% Final Exam Report

% by Ashfaq Khan

**B.1.1** Applying the flow balance principle with Fig.1, we can obtain the flow balance equations.

p1’(t) = λ4,1p4(t) – [λ1,2 + u1λ1,3 +λ1,3 (1 – u1)](p2(t) + p3(t))kp1(t)

p2’(t) = [λ1,2 + u1λ1,3](p2(t) + p3(t))kp1(t) – λ2,4p2(t)

p3’(t) = λ1,3(1 – u1)(p2(t) + p3(t))kp1(t) – (λ3,4 + λ3,5)p3(t)

p4’(t) = λ2,4p2(t) + λ5,4p5(t)+(λ3,4 + u3λ3,5)p3(t) – λ4,1p4(t)

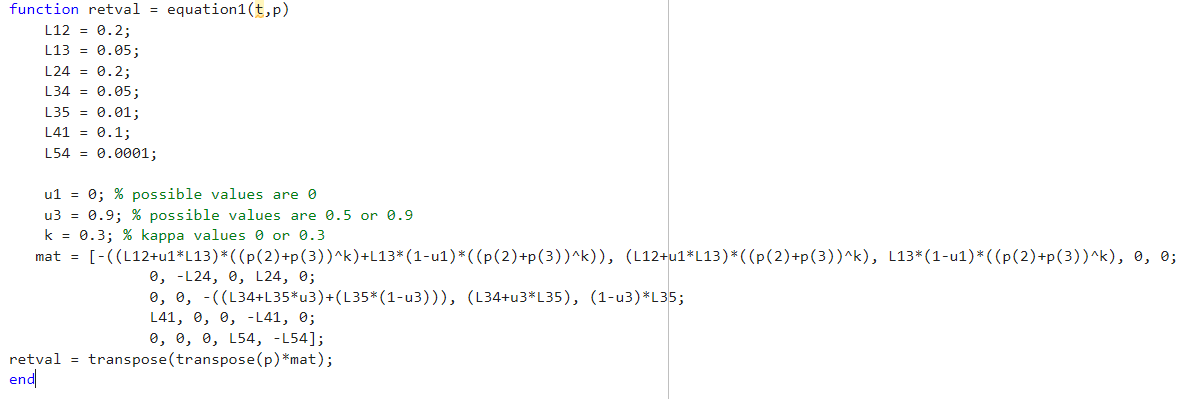
p5’(t) = λ3,5(1 – u3)p3(t) - λ5,4(t)p5(t)

**

Plot 1: Population Dynamics, k = 0, u1, u3 = 0,0.5 Plot 2: Population Dynamics, k = 0, u1, u3 = 0,0.9

**

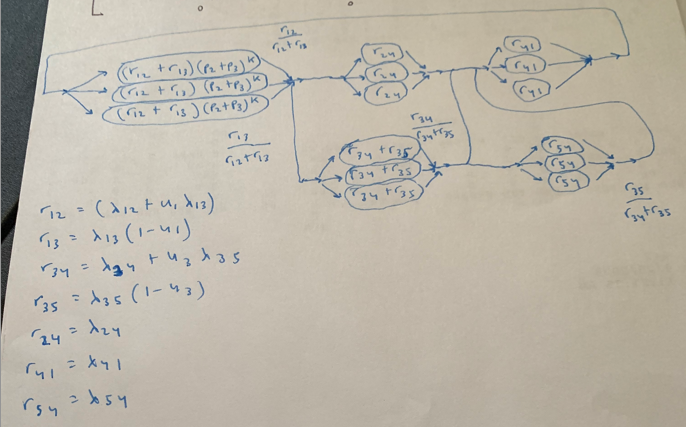
Plot 3: Population Dynamics, k = 0.3, u1, u3 = 0,0.5 Plot 4: Population Dynamics, k = 0.3, u1, u3 = 0,0.9



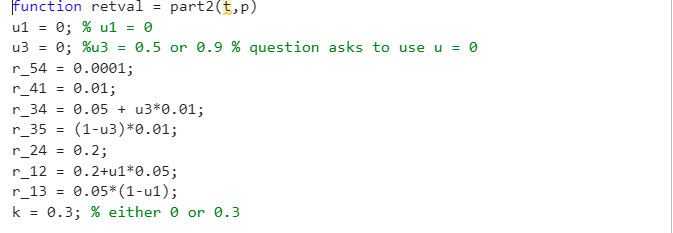
*Plot is from equation1.m to simulate the population dynamics.*

These are the plots for the Population Dynamics observed from simulating the kappa values and the U1 and U3 values from table 1 until the populations settle to equilibrium for each case.

**B.1.2**



**Fig.F2 A closed queueing network realization of the epidemiological process.**



*Plot is from part2.m to simulate the population dynamics.*



Plot 5: Population Dynamics, k = 0, u1, u3 = 0,0

This is the observed population dynamics for N = 3 when u = (0, 0) and for k = 0.



Plot 6: Population Dynamics, k = 0.3, u1, u3 = 0,0.5

This is the observed population dynamics for N = 3 when u = (0, 0) and for k = 0.3.

\* see MATLAB code in part2.m and the % part b 1.2 section in solverscript.m

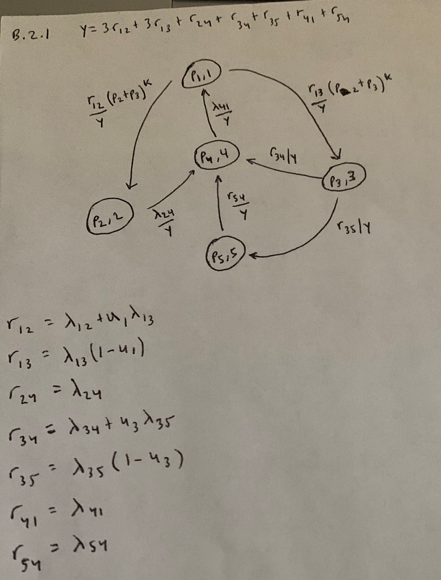
**B.1.3** Population dynamics where N = 30

**B.1.4** Summary of B.1

From plotting the population dynamics, it is clear the p values add up to 1 throughout the simulation which is what is expected. For the N = 3 system simulation of population dynamics, we get that same expected result of the p values summing up to 1. Selecting the different u1 and u3 values, along with the provided kappa values, the behavior of the curves are slightly different in the middle of the simulation, but otherwise the plot of the population dynamics is fairly similar to the plot of the population dynamics observed from the earlier project.

**B.2.1** Convert into probability transition diagram using gamma.

Express explicitly the transition probability matrix for the uniformized chain in terms of lambda and gamma.



We can create the transition probability matrix P = [pij]

P = [ - (r12/γ (p2 + p3)k + r13/γ (p2 + p3)k r12/γ (p2 + p3)k r13/γ (p2 + p3)k 0 0

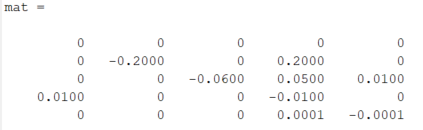
0 - r24/γ 0 r24/γ 0

0 0 - (r34/γ + r35/γ) r34 /γr35/γ

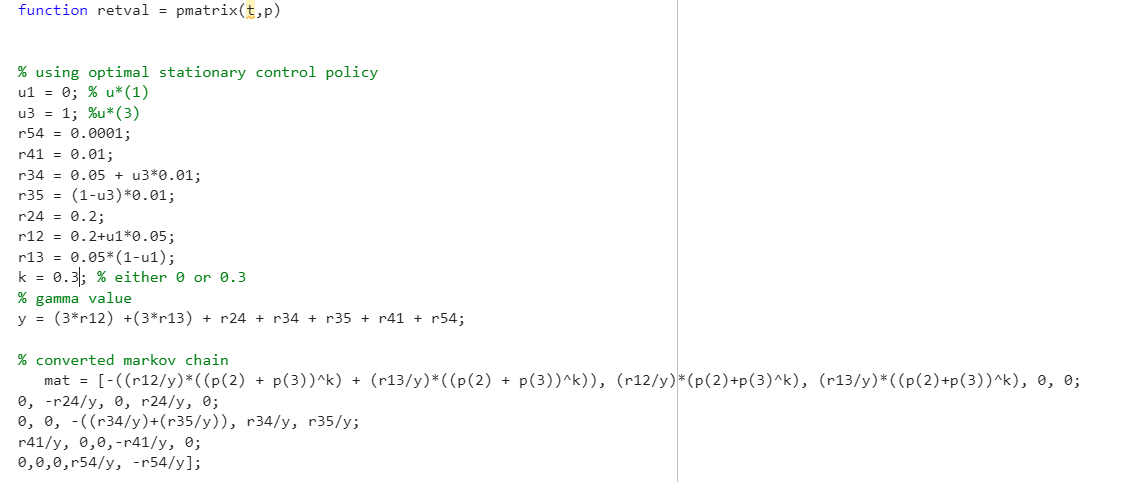
r41/γ 0 0 - r41/γ 0

0 0 0 r54/γ - r54/γ]

Where γ = 3r12 +3r13 + r24 + r34 + r35 + r41 + r54

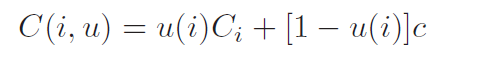


*Ran mat to get the matrix for the uniformized markov chain*

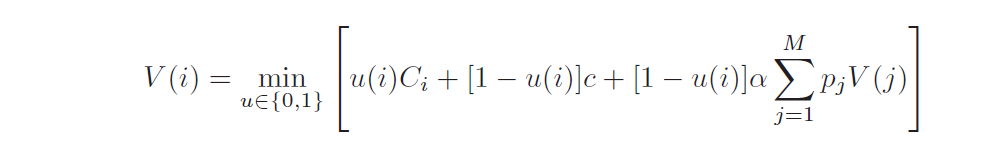


*Plot is from pmatrix.m to simulate the population dynamics.*

The cost structure is defined as follows:

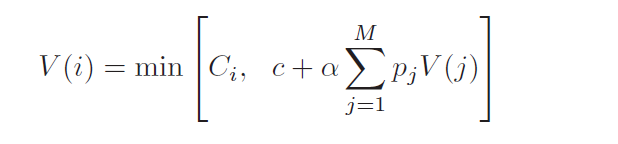


Solve for the stationary control policy u\*(x) = (u\* (1), u\* (3)) that minimizes the total expected discounted cost Vpi(x).

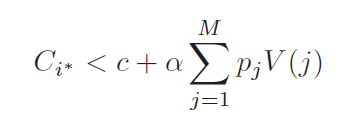


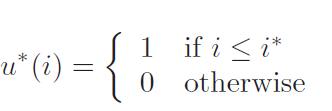
Step 1 is to write down the optimality equation. This is equation 9.34 from the textbook showing the optimality equation.

Can be simplified into equation 9.35.



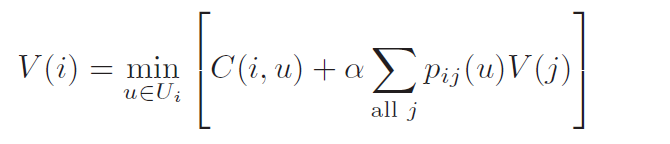
Step 2 is to characterize the optimal control actions. We must define an i\* by which I <= i\*, that is to hold our job until we are assigned any processor.





Step 3 is to try and obtain a closed form expression for the optimal policy.





We have the unconditioned total expected discount cost

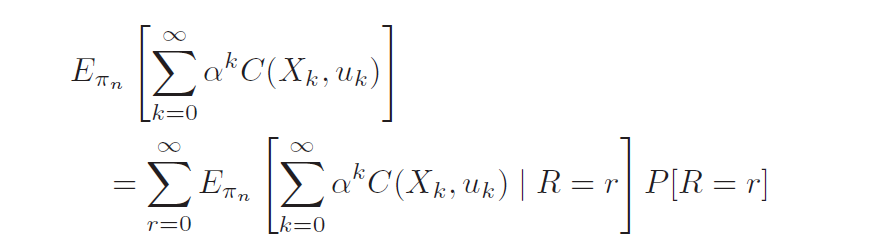


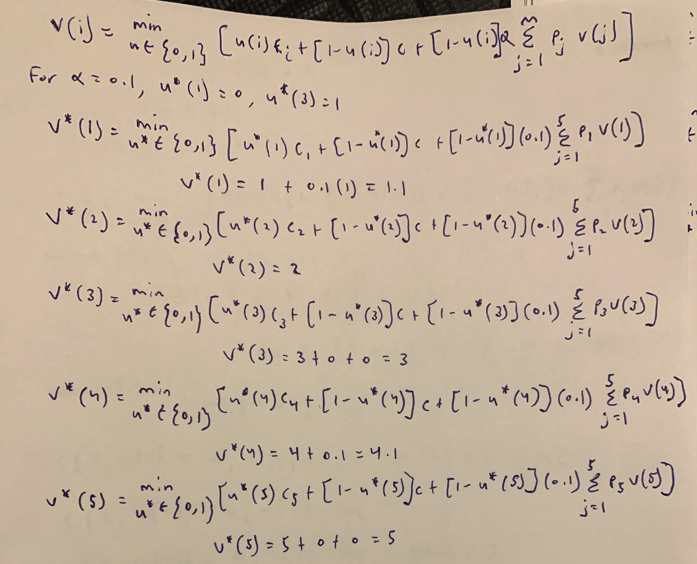
Table 3: Optimal policy, cost-to-go, and steady-state probability

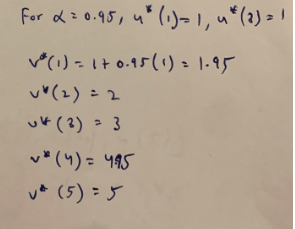
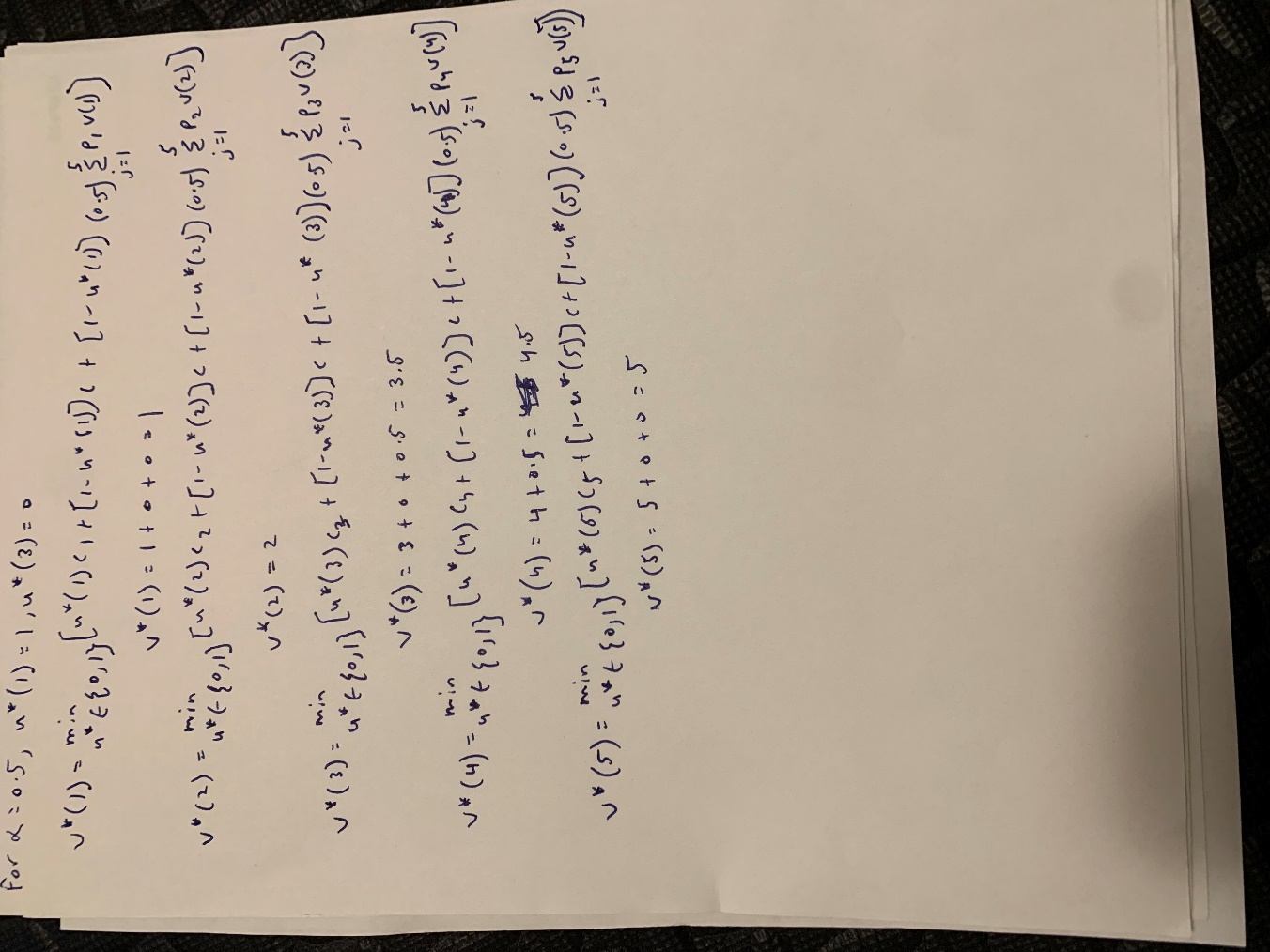
|  |  |  |
| --- | --- | --- |
| alpha | u\*(1), u\*(3) | V\*(1), V\*(2), V\*(3), V\*(4), V\*(5) |
| 0.1 | 0, 1 | 1.1, 2, 3, 4.1, 5 |
| 0.5 | 1, 0 | 1, 2, 3.5, 4.5, 5 |
| 0.95 | 1, 1 | 1.95, 2, 3, 4.95, 5 |

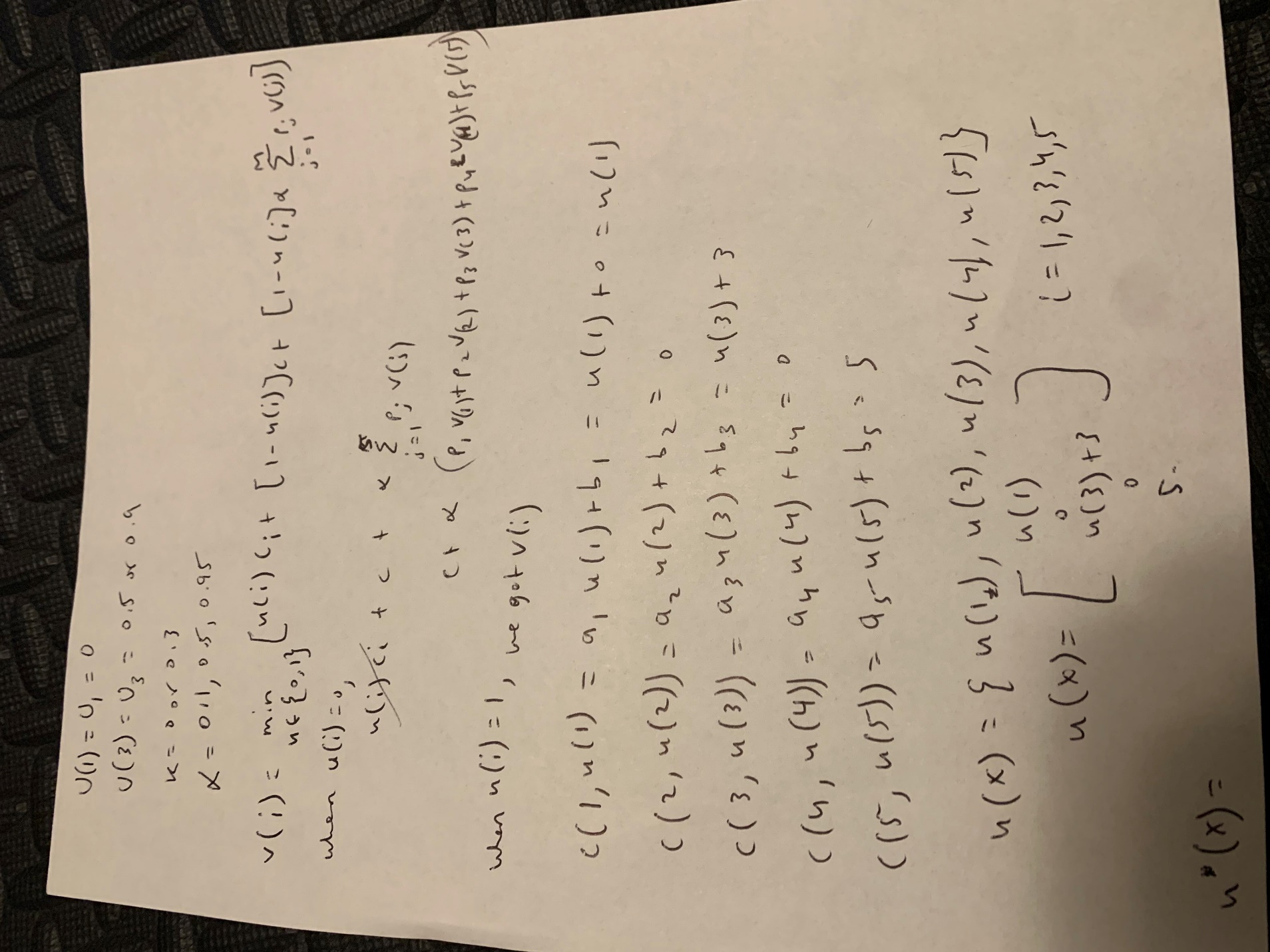


Plot 7: Population Dynamics, k = 0.3, u\*1, u\*3 = 0, 1

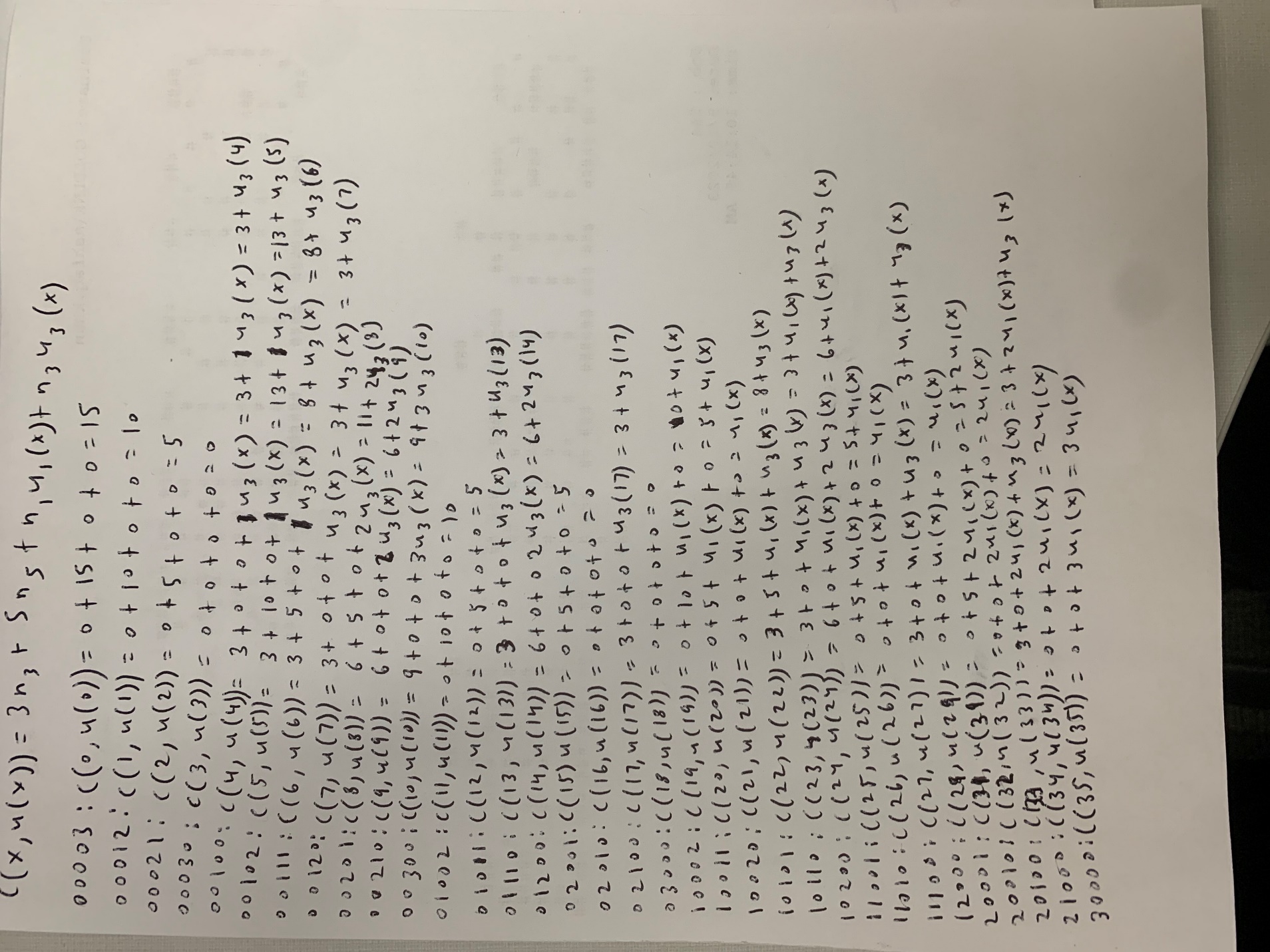
This is using u\*(1) and u\*(3) which is the optimal stationary policy and simulating the mean population dynamics for k = 0.3.





­­­­­­

**B2.2** Solving for MDP for the PCMC from B.1.2 where N = 3, k > 0





Plot 8: Mean Population Dynamics, k = 0.3, u\*1, u\*3 = 0, 0 and c(x, u(x))

Table 4: Optimal policy, cost-to-go, and steady-state- probability

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| x | u\*(x) | V\*(x) | C(x, u(x)) | pix(u\*) |
| 00003 | 0 |  | 15 |  |
| 00012 | 0 |  | 10 |  |
| 00021 | 0 |  | 5 |  |
| 00030 | 0 |  | 0 |  |
| 00100 | 0 |  | 3 |  |
| 00102 | 0 |  | 13 |  |
| 00111 | 0 |  | 8 |  |
| 00120 | 0 |  | 3 |  |
| 00201 | 0 |  | 13 |  |
| 00210 | 0 |  | 8 |  |
| 00300 | 0 |  | 12 |  |
| 01002 | 0 |  | 10 |  |
| 01011 | 0 |  | 5 |  |
| 01110 | 0 |  | 3 |  |
| 01200 | 0 |  | 8 |  |
| 02001 | 0 |  | 5 |  |
| 02010 | 0 |  | 0 |  |
| 02100 | 0 |  | 3 |  |
| 03000 | 0 |  | 0 |  |
| 10002 | 0 | 0 | 10 |  |
| 10011 | 0 | 0 | 5 |  |
| 10020 | 0 | 0 | 1 |  |
| 10101 | 0 | 0 | 8 |  |
| 10110 | 0 | 0 | 3 |  |
| 10200 | 0 | 0 | 6 |  |
| 11001 | 0 | 0 | 5 |  |
| 11010 | 0 | 0 | 1 |  |
| 11100 | 1 | 1.95 | 5 |  |
| 12000 | 1 | 2 | 1 |  |
| 20001 | 0 | 0 | 7 |  |
| 20010 | 0 | 0 | 2 |  |
| 20100 | 0 | 0 | 6 |  |
| 21000 | 1 | 4.95 | 2 |  |
| 30000 | 0 | 0 | 3 |  |

**B.2.3** Conclusion

Created a P matrix for the uniformed chain using gamma. The cost associated, c(x, u(x)), was calculated using the Table 2 coefficient values for ax and bx. We know that if processor 1 is rejected, we automatically reject the other processors. Using the u\* values, which are the optimal stationary policy values, the population dynamics were simulated. The V\* values, which are the minimum cost-to-go values were calculated and entered into the table. From this, we can determine that the plot using the optimal stationary policy has more fluctuation than without the optimal stationary policy in part B.1.1 and B1.2 where u(x) = 0. Then in part B.2.2, we created a new c(x. u(x)) cost and using the 35 states from our N = 3 system it was possible to update table 4 with u values and the steady state probability.