

# PSET1 Julia Code Sneha Kabaria-Correct

February 5, 2019

```
In [1]: #Import Plotting Packages
```

```
using Pkg; Pkg.add("Plots")
using Plots
Pkg.add("PyPlot")
```

```
Resolving package versions...
Updating `~/julia/Project.toml`
[no changes]
Updating `~/julia/Manifest.toml`
[no changes]
Resolving package versions...
Updating `~/julia/Project.toml`
[no changes]
Updating `~/julia/Manifest.toml`
[no changes]
```

```
In [2]: gr()
```

```
Out[2]: Plots.GRBackend()
```

```
In [3]: #PART A and B
```

```
In [4]: #Biological Constants
```

```
In [5]: #Closed to open complex =  $k_2 = .024 \text{ s}^{-1}$  (McClure Paper)
        kI = 0.024; #s-1
```

```
In [6]: # Elongation rate =  $e_X = 42 \text{ nts/sec}$ 
```

```
# Note for all links, they are too long to be pasted, so they are
# put onto multiple lines. All are official cited in a separate
# doc.
```

```
#All citations also included in constants document
```

```
# https://bionumbers.hms.harvard.edu/bionumber.aspx?id=108488&ve
# r=3&term=elongation+rate+in+E.+coli&org=
```

```

e_X = 42; # nt/sec
Lj = 3075; #nt/gene
kej = e_X/Lj #s-1

Out[6]: 0.013658536585365854

In [7]: #Find K_x,j = McClureSlope*kI (see paper), converted to units of [mM]
McClureSlope = 1.04*10^-3; #mM*s
K_xj = McClureSlope*kI #mM

Out[7]: 2.4960000000000001e-5

In [8]: #RNAP concentration = 30 nM
# https://bionumbers.hms.harvard.edu/bionumber.aspx?id
# =100194&ver=8&trm=rnap+e+coli+M&org=
R_xt=30*10^-6; #mM

In [9]: #Find the gene concentration
Gj_initial = 2500; # copies/cell
Gj_2 = Gj_initial / (6.02*10^23); #mol/cell
#Bionumbers: volume per cell = 6.7E-10 L/cell
# https://bionumbers.hms.harvard.edu/bionumber.aspx?id
# =108815&ver=1&trm=volume+of+e+coli+cell&org=
Gj_3= Gj_2 / (6.7*10^-16) #mol/L = M
Gj = Gj_3*1000 # mM

Out[9]: 3093.3004197832283

In [10]: tau = kej/kI
#also PART B

Out[10]: 0.5691056910569106

In [11]: rxj = kej*R_xt*(Gj/(tau*K_xj+(tau+1)*Gj)) #mM/s
#ANSWER TO rxj in PART A

Out[11]: 2.611398956088053e-7

In [12]: #PART C

In [13]: pyplot()

Out[13]: Plots.PyPlotBackend()

In [23]: #Constants
W1 = 0.26;
W2 = 300;
n = 1.5;
K_c = 0.30; #mM
#find f(I)
f(I) = (I^n)/(K_c^n+(I^n));
#find u(I)
u(I) = (W1 + W2*f(I))/(1 + W1 + W2*f(I))

```

Out[23]: u (generic function with 1 method)

```
In [24]: #growth rate, doubling time = 30 min
        grow = 30; #min
        #dilution is 1/grow, also convert to seconds
        B_term = 1/(30*60) #s-1
```

Out[24]: 0.0005555555555555556

```
In [25]: #global half life of mRNA = 5 min
        # https://bionumbers.hms.harvard.edu/bionumber.aspx?id=111927&ver
        # =2&term=mrna+half+life+e+coli&org=
        halflife = 5; #min
        #convert to seconds
        halflife2 = 5*60; #sec
        #convert to degradation rate, assuming first order kinetics
        kdeg = .693/halflife2 #s-1
```

Out[25]: 0.00231

```
In [26]: m_j(I) = (rxj*u(I)/(kdeg+B_term)) #units of mM
        #convert to uM for the plot
        m_j_2(I) = m_j(I)*1000 #uM
```

Out[26]: m\_j\_2 (generic function with 1 method)

```
In [27]: #lower bound
        a = m_j_2(0.0001)
```

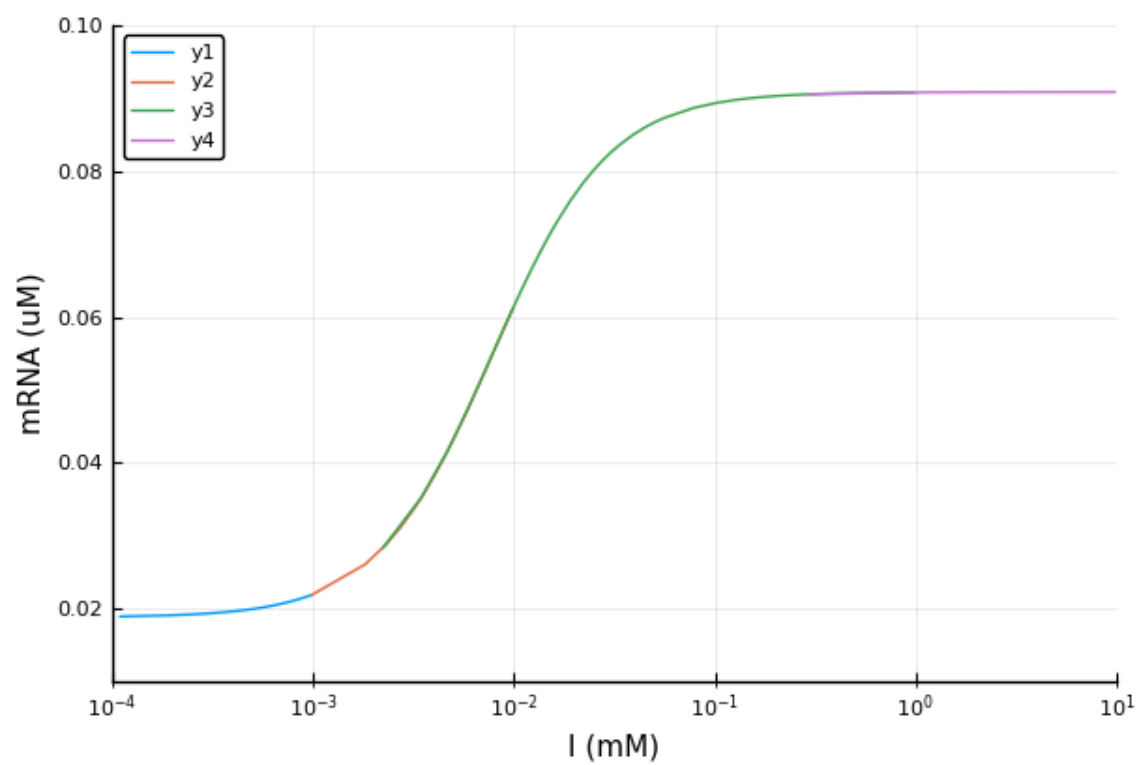
Out[27]: 0.018909382141098822

```
In [28]: #upper bound
        b = m_j_2(10)
```

Out[28]: 0.09082657086541693

```
In [29]: #Plotted the entire range incrementally
        #(will not plot all at once, possible due to Julia Box limitations)
        plot(m_j_2,0.0001,0.001,xaxis=:log,xlims=(0.0001,10),ylims=(1*10^-2,10*10^-2),
            xlabel = "I (mM)",ylabel="mRNA (uM)")
        plot!(m_j_2,0.0009,0.01,xaxis=:log,xlims=(0.0001,10),ylims=(1*10^-2,10*10^-2))
        plot!(m_j_2,0.001,1,xaxis=:log,xlims=(0.0001,10),ylims=(1*10^-2,10*10^-2))
        plot!(m_j_2,0.2,10,xaxis=:log,xlims=(0.0001,10),ylims=(1*10^-2,10*10^-2))
```

Out[29]:



In [ ]: