CHEME 5440/7770: Prelim 1 Q2&Q3

Yujia Huang, yh945
Smith School of Chemical and Biomolecular Engineering, Cornell University, Ithaca NY 14850

2. a) Formulate a three micro-state model for PEK activity

- **State s = 0**: no effector+no substrate (base state, no activity)
- **State s = 1**: no effector+substrate (the data shows low activity)
- **State s = 2**: effector+substrate (activity)

take the form

$$\hat{r}_j = r_j v(\dots)_j$$

The probability of each microstate is given by

$$p_i = \frac{1}{Z} \times f_i \exp\left(-\beta\epsilon_i\right) \qquad i = 0, 1, 2, \dots, \mathcal{S}$$

where

$$W_i = \exp\left(-\beta\epsilon_i\right)$$

$$Z = \sum_{s=0}^{\mathcal{S}} f_i \exp\left(-\beta\epsilon_i\right)$$

which gives:

$$p_i = \frac{f_i \exp\left(-\beta\epsilon_i\right)}{\sum_{s=0}^{\mathcal{S}} f_i \exp\left(-\beta\epsilon_i\right)} \qquad i = 0, 1, 2, \dots, \mathcal{S}$$

Given these microstates, we know that enzyme E can catalyze its reaction in microstate $s = 1$ and $s = 2$, thus:

$$v(\dots)_j = \frac{f_1 \exp\left(-\beta\epsilon_1\right) + f_2 \exp\left(-\beta\epsilon_2\right)}{\sum_{s=0}^{2} f_s \exp\left(-\beta\epsilon_s\right)}$$

let $j = 1$

$$r_1 = k_{cat} E_1 (\frac{F6P}{K_{K6P} + F6P})(\frac{ATP}{K_{ATP} + ATP})$$

$$\hat{r}_1 = r_1 v(\dots)_1$$

2. b) Estimate the parameters by using the dataset in Table 1

From definition we know $\epsilon_0 = 0$, then $W_0 = 1$ and we also know f_0 and f_1 are both set to 1. So I drag the buttons to estimate ϵ_1, ϵ_2, the binding constant Kd and an order parameter n to get W_1, W_2 and f_2 to match the dataset in table 1.

Kd (mM) W1 W2

```
(1.54, 0.046, 260.0)
```

```
DSM_parameters
```

calculate v_j

```
v = [0.04397705544933078, 0.05389273057560039, 0.0824382077(
```

```julia
begin
    # get Effector - A
    A = [0:0.01:1;]
    v = A
    for i in eachindex(A)
        Kd = DSM_parameters[1]
        W0 = 1
        W1 = DSM_parameters[2]# state 1 (E bound to S,
        but no A)
        W2 = DSM_parameters[3] # state 2 (E bound to A)

        # setup system -
        R = 8.314           # units: J/mol-K
        T = 273.15 + 25.0   # units: K
        β = 1/R*T

        # setup binding parameters for state 2 -
        n = 2.0

        # compute the state-specific factor-
        f0 = 1.0 # state 0
        f1 = 1.0 # state 1
        f2 = ((A[i]/Kd)^(n))/(1+(A[i]/Kd)^(n)) # state 2

        # compute the v variable -
        microstate_0 = f0.*W0
        microstate_1 = f1.*W1
        microstate_2 = f2.*W2

        Z = microstate_0 + microstate_1 + microstate_2
        p1 = (1/Z)*microstate_1
        p2 = (1/Z)*microstate_2
        v[i] = p1 + p2
    end
    # show -
    with_terminal() do
        println("v = $(v)")
    end
end
```

Show the r_bar calculated when the concentration of the effector is the same as the ones that are shown in table 1. See if they are matched to each other.

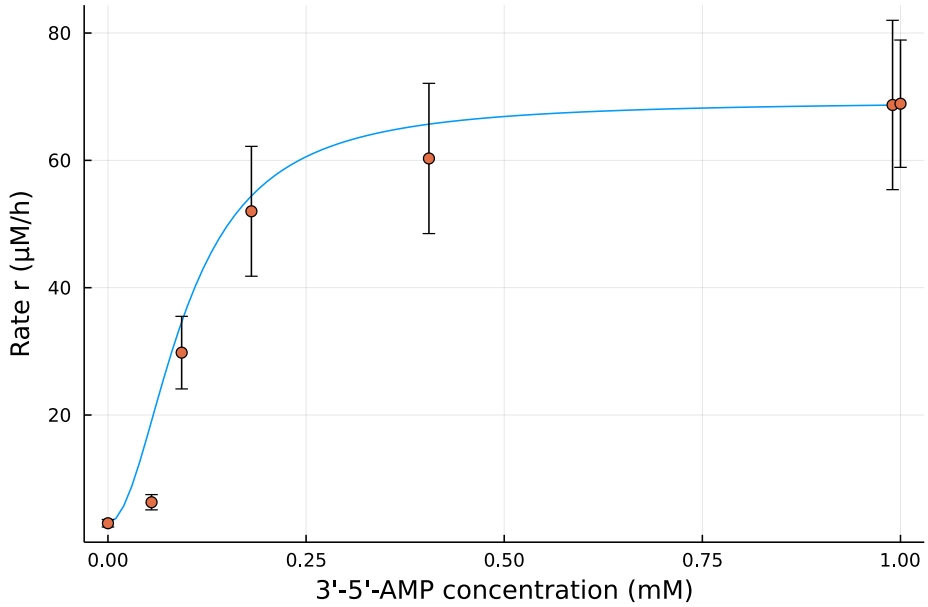
```
r_bar[1,5,9,18,40,99,100] = 3.05991612706268,12.60745895571
```

```
begin
# Set up some parameters
    S1 = 0.1 # units: mM -- concentration for F6P
    S2 = 2.3 # units: mM -- concentration for ATP
    E = 0.12 # units: μM
    K_F6P = 0.11 # units: mM
    K_ATP = 0.42 # units: mM
    kcat = 0.4 # units: s^-1

# calculate the rate
    r_bar = (kcat*E)*(S1/(S1+K_F6P))*
    (S2/(S2+K_ATP))*v*3600

# show -
    with_terminal() do
    println("r_bar[1,5,9,18,40,99,100] =
    $(r_bar[1]),$(r_bar[5]),$(r_bar[9]),$(r_bar[18]),$(r_b
    ar[40]),$(r_bar[99]),$(r_bar[100])")
    end
end
```

so the final results I choose are kd = 1.54, W1 = 0.046, W2 = 260.0(set the buttons to the numbers respectively to see the final results)

2. c) Plot the converted data with errorbars

from the image we can see the proposed model describes the data well except for the second one




```
begin

    # 3'-5'-AMP concentration -
    x = 0:0.01:1
    conc = [0, 0.055, 0.093, 0.181, 0.405, 0.990, 1.0]
    rate = [3, 6.3, 29.8, 52.0, 60.3, 68.7, 68.9]
    std = [0.59, 1.2, 5.7, 10.2, 11.8, 13.3, 10.0]

    # overall rate -
    y = r_bar

    # plot -
    plot(x,y,label="r")
    plot!(conc,rate,yerror=std,seriestype = :scatter,
    legend =false)
    xlabel!("3'-5'-AMP concentration (mM)",fontsize=18)
    ylabel!("Rate r (μM/h)",fontsize=18)

end
```

3. a) convert the <n> values in Table 2

```
[0.26125, 0.28875, 0.56375, 0.92125, 1.1825, 1.27875, 1.27875]
```

```
begin
    n_array = [19;21;41;67;86;93;93;]; #units: nM from
    the assume(ii)
    mc = 2 * 10^(-13) # units:g
    Vc = 2.75 # units: μm^3

    n_array_new = n_array * Vc / mc * 10^(-15)
end
```

3. b) Derive the gain function κ_x and formulate $\bar{u}_i$

first for κ_x

from the equation

$$\frac{dm_i}{dt} = r_{X,i}\bar{u}_i - (\theta_{m,i} + \mu)m_i$$

we can get steady-state mRNA abundance m*

$$m^* = \kappa_x(G, \dots)\bar{u}(I, k)$$

Therefore

$$\kappa_x = \frac{r_{X,i}}{\theta_{m,i} + \mu}$$

and also

$$r_{X,i} = V_{max,i}\frac{[G_i]}{K_{X,i} + [G_i]}$$

where $V_{max,i} \equiv k_{3,i}[RNAP]_T$, and $k_{3,i} \sim \langle e_X \rangle L_i^{-1}$

The saturation constant of transcription $K_{X,i}$ (units: conc) is defined as the ratio of elementary rate constants:

$$K_{X,i} \equiv \frac{k_{2,i} + k_{3,i}}{k_{1,i}}$$

We know that the RNAP dissociation constant $K_{D,i}$:

$$K_{D,i} = \frac{k_{2,i}}{k_{1,i}}$$

for the lac promoter in *E. coli* is [$K_D \sim 550$ nM (units: nM or molecules/cell)]

then for $\bar{u}_i$

using the Discrete state model for promoter functions (same as Q2)

- **State s = 0**: base state, no transcription possible
- **State s = 1**: RNAP bound to G_i at I = 0. (the data shows transcription possible although low)
- **State s = 2**: RNAP + inducer bound to G_i (transcription possible)

The probability of each microstate is given by

$$p_i = \frac{1}{Z} \times f_i \exp\left(-\beta \epsilon_i\right) \qquad i = 0, 1, 2, \ldots, \mathcal{S}$$

where

$$W_i = \exp\left(-\beta \epsilon_i\right)$$

$$Z = \sum_{s=0}^{\mathcal{S}} f_i \exp\left(-\beta \epsilon_i\right)$$

which gives:

$$p_i = \frac{f_i \exp\left(-\beta \epsilon_i\right)}{\sum_{s=0}^{\mathcal{S}} f_i \exp\left(-\beta \epsilon_i\right)} \qquad i = 0, 1, 2, \ldots, \mathcal{S}$$

Finally, we relate the probability that promoter P is in microstate s back to the $\bar{u}(\ldots)$ control function by computing the overall probability that the desired event happens, e.g., promoter P undergoes transcription. We know if $\Omega = \{1, 2, \ldots, \mathcal{S}\}$, then we can define the subset $\mathcal{A} \subseteq \Omega$ in which the desired event happens (in this case transcription). Given $\mathcal{A}$, the $\bar{u}(\ldots)$ function becomes:

$$\bar{u} = \sum_{s \in \mathcal{A}} p_s$$

3. c) Use the data in Table 2 to estimate the discrete state promoter model parameters in $\bar{u}_i$ and the gain κ_x(G...)

K (mM) W1*new* *W*2new

```
(0.065, 0.27, 100)
```

```
RNA_parameters
```

calculate $\bar{u}$, set n = 2

```
u = [0.21259842519685038, 0.2127446933906952, 0.21318138110
```

```
begin
    # get Inducer - I
    I = [0:0.0001:1;]
    u = I

    for i in eachindex(I)

        K = RNA_parameters[1]
        W0_new = 1
        W1_new = RNA_parameters[2]# state 1
        W2_new = RNA_parameters[3] # state 2

        # setup system -
        R = 8.314           # units: J/mol-K
        T = 273.15 + 25.0   # units: K
        β = 1/R*T

        # setup binding parameters for state 2 -
        n_new = 2.0

        # compute the state-specific factor-
        f0_new = 1.0 # state 0
        f1_new = 1.0 # state 1
        f2_new = ((I[i]/K)^(n_new))/(1+I[i]/K)^(n_new)#
    state 2

        # compute the v variable -
        microstate_0_new = f0_new.*W0_new
        microstate_1_new = f1_new.*W1_new
        microstate_2_new = f2_new.*W2_new

        Z_new = microstate_0_new + microstate_1_new +
        microstate_2_new
        p1_new = (1/Z_new)*microstate_1_new
        p2_new = (1/Z_new)*microstate_2_new
        u[i] = p1_new + p2_new

    end

    # show -
    with_terminal() do
        println("u = $(u)")
    end
end
```

calculate κ_x(G...) and $m*$, try to find the proper values match the $m*$ to the converted data from copy numbers that are given in table 2 when with the same concentration I (set Kd = 430)

m = 0.27260442611792557,0.27556961637597216,0.5542793685586

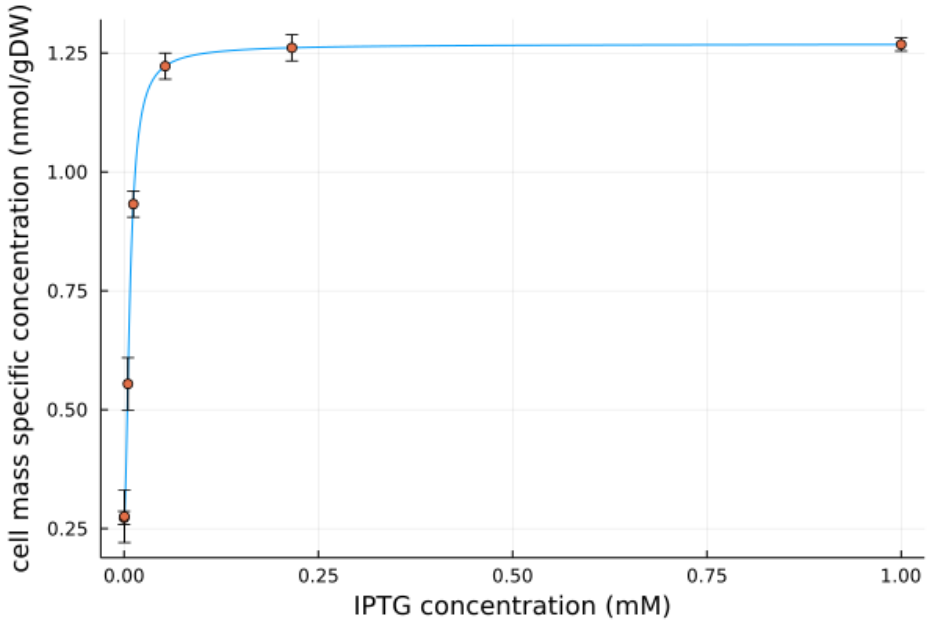
```
begin

    # setup the parameters -
    parameters = Dict{String, Any}()

    # compute values given or estimate -
    k₁ = 100.0  # units: 1/conc-t
    eₓ = 35.0   # units: nt/s
    L = 3075.0  # units: nt
    KD = 430.0  # units: nM -- estimate
    Kₓ = (eₓ*(1/L) + KD*k₁)/(k₁)

    # get parameters from given -
    parameters["RNAP_copy_number"] = 4600.0
            # units: copies/cell
    parameters["RNAP_elongation_rate"] = eₓ
            # units: nt/s
    parameters["gene_coding_length"] = L
            # units: nt
    parameters["mRNA_half_life"] = 5*(60)
            # units: s
    parameters["gene_copy_number"] = 2
            # units: copies/cell
    parameters["saturation_constant_transcription"] = Kₓ
            # units: copies/cell
    parameters["doubling_time"] = 40*60
          # units: s

    # get values from the parameters dictionary -
    RNAP = parameters["RNAP_copy_number"]
    eₓ = parameters["RNAP_elongation_rate"]
    Kₓ = parameters["saturation_constant_transcription"]
    G = parameters["gene_copy_number"]
    t_half_life = parameters["mRNA_half_life"]
    L = parameters["gene_coding_length"]
    td = parameters["doubling_time"]


    # compute Vmax and the constants -
    Vmax = RNAP*(eₓ)*(1/L)
    θₓ = -(log(0.5)/t_half_life)
    μ = log(2)/td

    # compute r_x,i
    r_x = Vmax * G/(Kₓ+G)

    # compute m* -- units nmol/gDW
    m = r_x/(θₓ+μ)*u * Vc / mc * 10^(-15)

    # show -
    with_terminal() do
```

```
        println("m =
        $(m[1]),$(m[5]),$(m[50]),$(m[120]),$(m[530]),$(m[2160]
        ),$(m[10000])")
        end
    end
```

Finally I choose K = 0.065, W1 = 0.27, W2 = 100 for my model (set the buttons to the numbers)

3. d) Plot the converted data and the estimated average copy number from the model

from the plot we can see the model fits well




```
begin
   # IPTG concentration -
    xx = 0:0.0001:1
    concI = [0, 0.0005, 0.005, 0.012, 0.053, 0.216, 1.0]
    m_array =
    [m[1];m[5];m[50];m[120];m[530];m[2160];m[10000];];
    low_array = [1, 4, 4, 2, 2, 2, 1]
    low_array_new = low_array* Vc / mc * 10^(-15)
    high_array = [1, 5, 3, 2, 2, 2, 1]
    high_array_new = low_array* Vc / mc * 10^(-15)

    # copynumber -
    yy = m

    # plot -
    plot(xx,yy,label="copynumber")
    plot!(concI,m_array,yerror=
    (low_array_new,high_array_new),seriestype = :scatter,
    legend = false)
    xlabel!("IPTG concentration (mM)",fontsize=18)
    ylabel!("cell mass specific concentration
    (nmol/gDW)",fontsize=18)
end
```

```
begin
    # import some packages -
    using PlutoUI
    using PrettyTables
    using LinearAlgebra
    using Plots
end
```