Biochemical Data

Goals and Objectives

-explain the relationship between the Philadelphia chromosome and CML

-describe the function of the BCR-Abl fusion kinase

-discuss the mechanism of imatinib action

-define the association and dissociation equilibrium constants in terms of reaction concentrations

-calculate the KD of a biomolecular reaction from a hyperbolic binding equation from a graph of data

-illustrate a first-order dissociation reaction

-estimate the half-life for the decay of the reactants from a first-order dissociation reaction

-plot biochemical data using MATLAB­

**Fitting experimental biochemical data – equibrilium and kinetic data**

**Why do we care about defining these biochemical parameters? CML, BCR-Abl and imatinib**

CML – 1-2 per 100,000 people

Treated with tyrosine kinase inhibitor Gleevec or imatinib

Before that, you had about 30% five-year survival – now it’s 90%

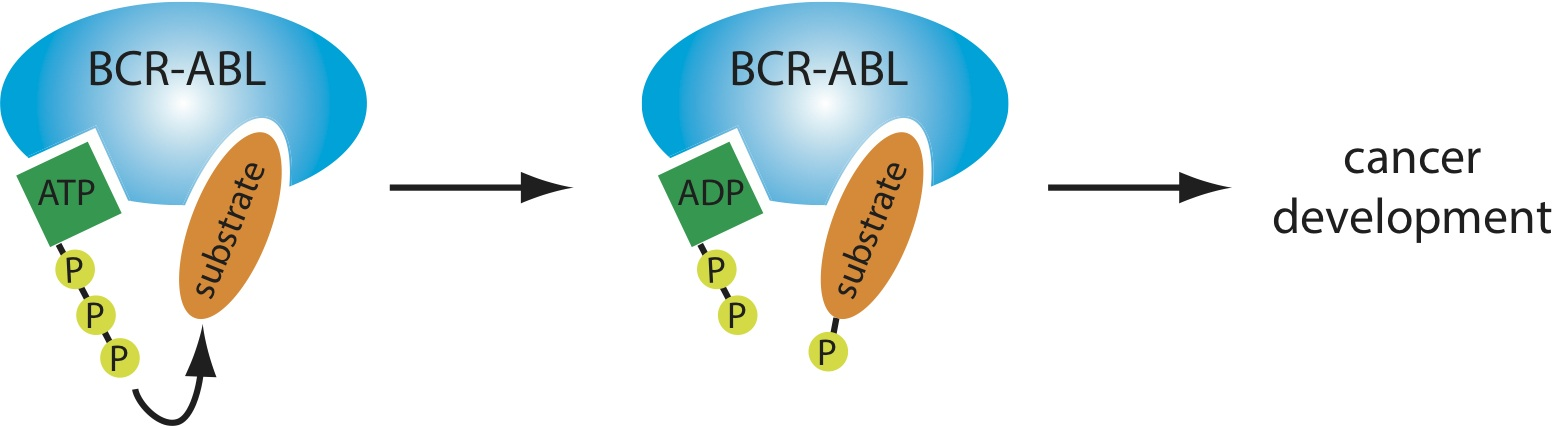
We know in detail how it works biochemically and these biochemical parameters help us understand why

CML is associated with a characteristic chromosomal translocation – a break and fusion between the *abl* gene on chromosome 9 and the *bcr* (breakpoint cluster region) region on chromosome 22 – known as the Philadelphia chromosome

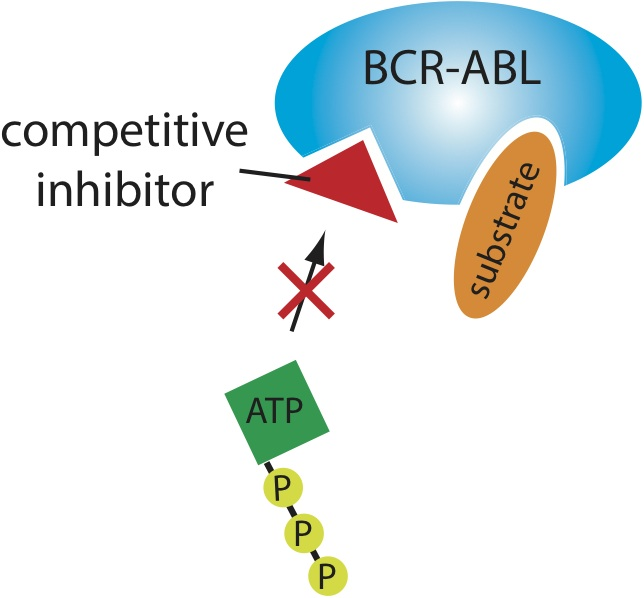
It generates bcr-abl which has kinase activity. Abl is a kinase implicated in control of the cell cycle. In normal cells, the Abl kinase is subject to regulation by cell cycle checkpoint proteins. However, cells with the Philadelphia chromosome produce a constitutively active BCR-Abl fusion protein that is not subject to regulation. Thus, the BCR-Abl fusion protein initiates a signaling cascade that speeds up cell division, leading to cancer.

**BCR-Abl kinase is responsible for CML development and imatinib is a competitive inhibitor**

The figure below illustrates the activity of the BCR-Abl fusion protein. The Abl portion of the protein has kinase activity, meaning it can take a phosphate group from ATP and transfer it to a substrate protein (or other substance). This reaction yields adenosine diphosphate (ADP) and a phosphorylated substrate. The phosphorylated substrate protein then goes on to initiate a signaling cascade that leads to cell proliferation. Because BCR-Abl is constitutively active, its presence leads to uncontrolled cell proliferation and the development of cancer.



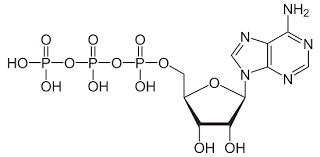
A major therapeutic strategy for CML treatment is to inhibit the BCR-Abl kinase by preventing ATP binding to it. One way to do this is by designing a drug inhibitor that binds tightly to the same site as ATP, thereby preventing ATP binding. This kind of molecule is called a competitive inhibitor.



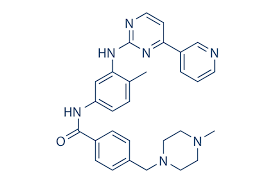
Gleevec is a competitive inhibitor of ATP for BCR-Abl. Without ATP, there is no phosphate to transfer to the substrate, there is no initiation of the signaling cascade, and there is no cancerous phenotype.

Let’s look in higher resolution at how these molecules and proteins actually look.

This is ATP:



And this is imatinib:



They share some aromatic ring structures, but for the most part they look pretty different. This is important because ATP participates in a massive number of reactions, and something that looks just like ATP would have off-target effects and likely kill cells or at least have limiting side effects. If you want to inhibit BCR-Abl specifically, you need something that just really binds BCR-Abl.

On the left is ATP binding to BCR-Abl in green, and on the right is imatinib binding to BCR-Abl in red.

ATP and imatinib bind to the same site in BCR-Abl, the definition of competitive inhibition.

For imatinib to be an effective competitive inhibitor, it has to bind more tightly to BCR-Abl than ATP. ATP is present in the cell in large quantities, and since we are not going to get molar quantities of a drug into the cell we want our drug to be able to bind very tightly. (Furthermore, ideally we want it to bind BCR-Abl more effectively than wild-type Abl or other tyrosine kinases, but that may not be covered in this lecture.)

**The equilibrium constant**

How tightly does imatinib bind to BCR-Abl?

A + B ↔ AB

We need some kind of assay to determine whether we are at equilibrium – you need to make measurements. If you measure free A for example, and then wait a bit, and then measure free A later and it’s the same, then you’re at equilibrium

Let’s define some of these biochemical parameters that we use to talk about how tightly A associates with B

Equilibrium constants – big K

(small k has to do with kinetic processes)

KA = association constant

KD = dissociation constant

KA = [AB]/[A][B] M-1

KD = [A][B]/[AB] M

Biochemists typically discuss binding affinities in terms of KD because units of molar are easier to understand intuitively.

**Hyperbolic binding equation**

Assuming we have an assay to measure the fraction of B bound in the complex, we can experimentally determine the KD in this manner.

Fraction B bound = θB = [AB]/([B]+[AB]) = 1 / ([B]/[AB] + 1)

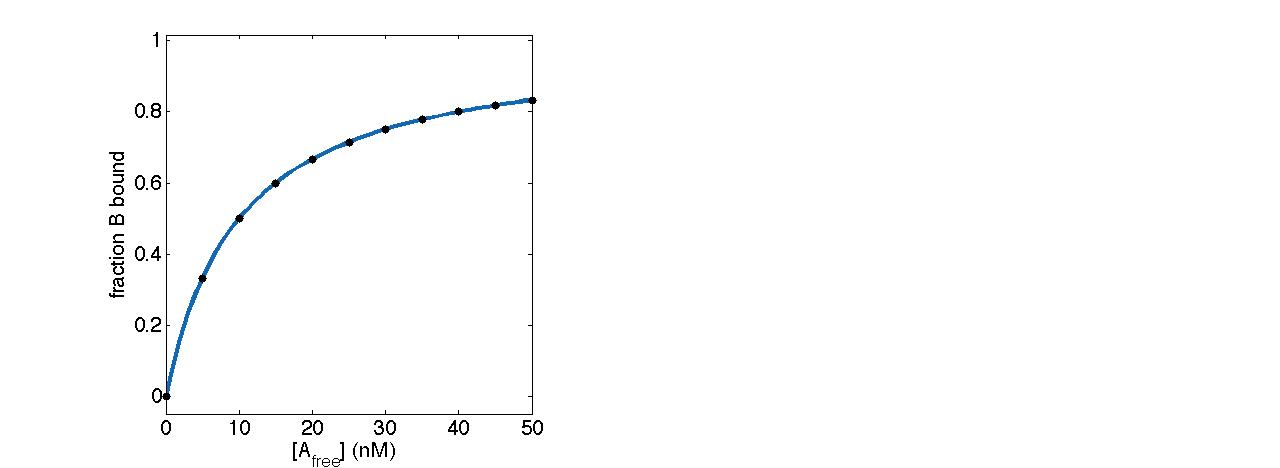
Substitute in the expression for KD:

1 / (KD/[A] + 1) = [A] / (KD + [A])

θB = [A] / (KD + [A])

NOTE This equation is only valid in this bimolecular reaction model A + B ↔ AB

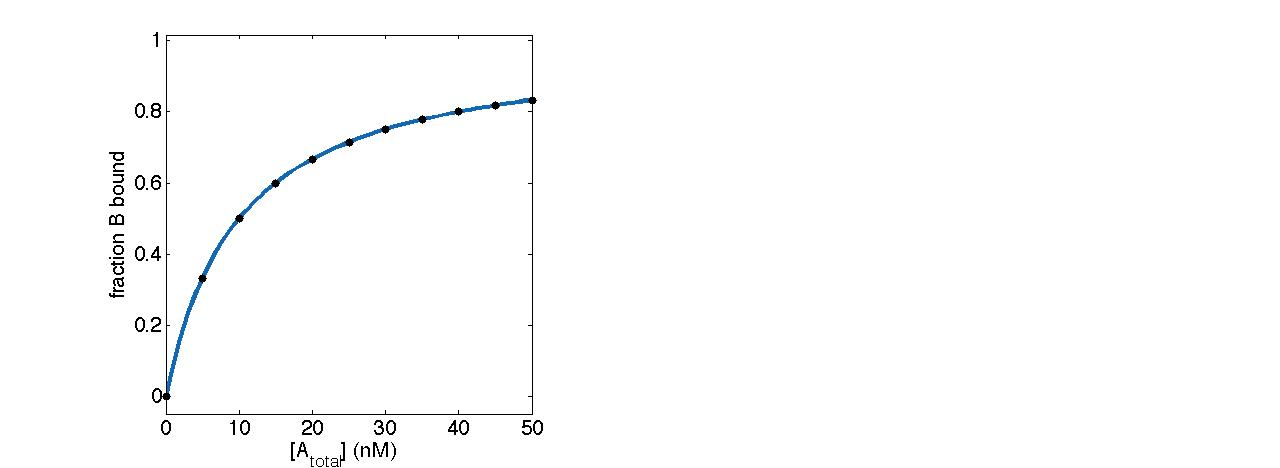
If we graph this with [A]free on the X axis and θB on the Y axis, we get a rectangular hyperbola:



It’s really easy to know how much A you added to the test tube but harder to know how much A is free…but if we choose our initial concentration of B so it’s very low, then B is limiting:

If [B0] << KD then [A]total ~= [A]free

And we need no assay for [A] and can just use the total [A] in our calculations



Note the changed x-axis.

When [A] = KD, θB = 0.5

You should make a full curve to determine your KD because you will have measurement error and then you can use all your data to see how your system behaves at all the different concentrations of A, and you can fit it all to a model and get a more accurate representation of the KD.

**Reason for B << Kd**

question posted 27 days ago by **[NicolaFalcioni](https://courses.edx.org/courses/course-v1:MITx+7.QBWx+3T2017/discussion/forum/users/2915959)**

Given the equation, borrowed from general chemistry, for the Kd, and the formula ΘB= [A]/(Kd+[A]), that we derived, how come the approximation that is done in order to consider Atotal = Afree is B << Kd? Wouldn't it be more sensible to write something like Afree >> AB? And if these two conditions are equivalent, how can I demonstrate it mathematically?

This post is visible to everyone.

Add a Response

0 responses

1. [**gill\_signals**](https://courses.edx.org/courses/course-v1:MITx+7.QBWx+3T2017/discussion/forum/users/19849580)

3 minutes ago

* + Vote for this post, there are currently 0 votes

Those two statements are equivalent ways of saying that B is the limiting reactant.

I'll work through the equation for KD using the substitution [A]total for [A]==[A]free:

KD=[A]free[B]/[AB]

[A]free=[A]total−[AB]

KD=([A]total−[AB])[B]/[AB]

KD=[A]total[B]/[AB]−[AB][B]/[AB]

KD=[A]total[B]/[AB]−[B]

[B]+KD=[A]total[B]/[AB]

So far this is just math - we haven't made any assumptions about the values of [B] or [A].

Compare the first and last lines. The only difference in the case when we consider total [A] instead of free [A] is the presence of [B] on the left side of the equation. If B is very very small compared to Kd - that is, if B << Kd - then these two equations are equivalent, or [A]total = [A]free.

It's more precise to define in terms of B and Kd because we can measure free B with our assay but our assay does not measure free A or AB - we have no way to tell whether total A is roughly equal to free A, and we have no way to measure the complex AB we can only assume it based on our measurements of B. This is the way we defined our problem - the lecture and problem description explained that we are assuming our assay measures free [B].

Hope that helps!

So far, everything that has been discussed has been in equilibrium. Concentrations are constant with respect to time and we ignore the kinetics of the process. Kinetics describe how the reactants change with respect to time before equilibrium is reached.

For these purposes, we will ignore the back reaction and we will talk about dissociation – the AB complex falling apart and how that happens with respect to time. We express this in terms of a rate law.

**First-Order Rate Law**

Rate laws describe the change in concentration of a reactant or product molecule with respect to time. A rate law is a function of the concentrations of molecules involved in the reaction as well as a rate constant *k*.

The rate law for a first-order dissociation reaction of a complex is described by the model below:

AB → A + B

Note that the arrow above goes in only one direction. This model assumes that the dissociation reaction is irreversible. The rate law is shown below.

Rate = -d[AB]/dt = d[A]/dt = d[B]/dt = k[AB]

{\text {rate}}=-\frac{d[AB]}{dt}=\frac{d[A]}{dt}=\frac{d[B]}{dt}=k[AB]

This is a **first**-order reaction because the rate depends on the concentration of only **one** reactant, [AB]. The rate of reaction has units of molar/second. The units of the first order rate constant are 1/second.

**Integrated solution to first-order rate law and half-life**

We can obtain the integrated solution to the first-order rate law by taking:

-d[AB]/dt = k[AB]

And rearranging it:

d[AB]/[AB] = -k dt

integrate both sides:

\int \frac{d[AB]}{[AB]}=-k\int dt

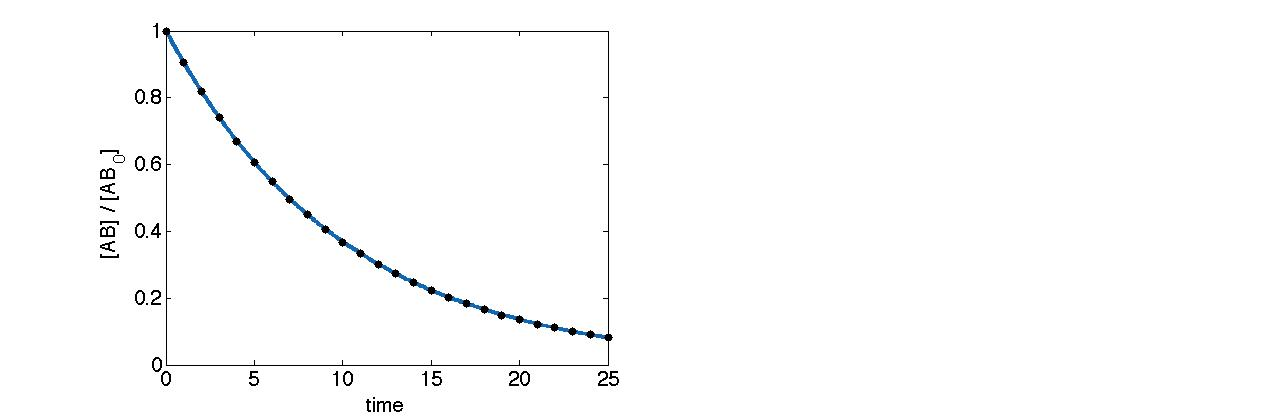
ln [AB] – ln [AB0] = -kt

plot(x,y)…at x=0, y=b+a…k where fluorescence range is in the middle…

ln [AB]/[AB0] = -kt

[AB]/[AB0] = e-kt

First order reactions exhibit an exponential decay of reactants as shown below:



The reaction half-life *t*1/2 is the time required for half the reactants to be converted to products – the time at which [AB]/[AB0] = 0.5. We can relate half-life to the kinetic relate constant like this:

0.5 = e-kt

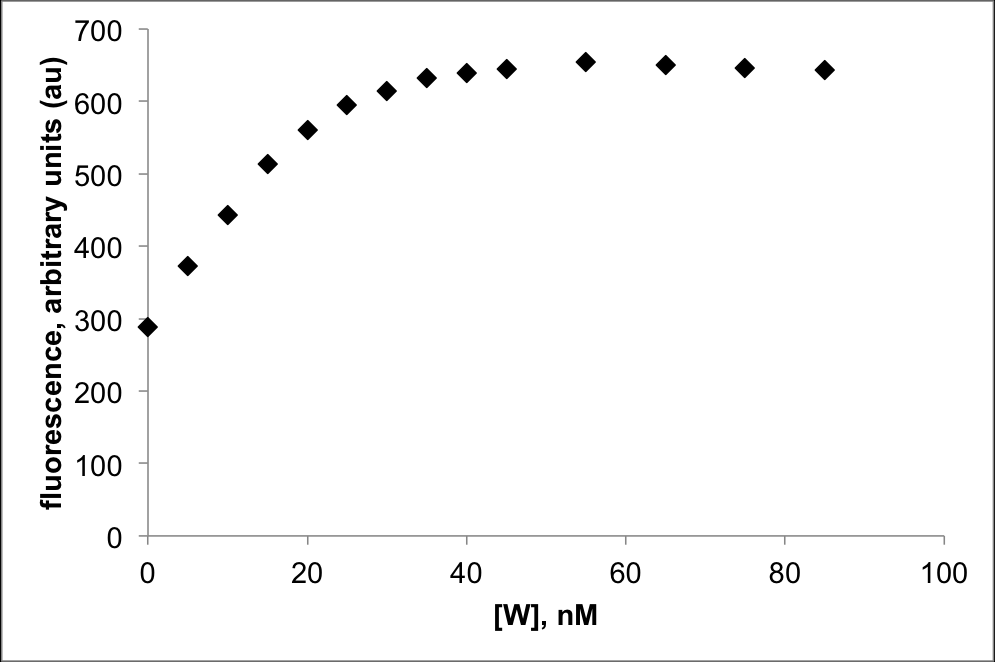
ln 2 = kt

t = ln 2/k = 0.693/k

The half-life gives us a more intuitive sense of how quickly a reaction occurs.

**Applying a model to data**

Suppose you perform an experiment measuring the equilibrium binding of an inhibitor molecule W to your protein E. You set up samples with constant [E] and vary [W], wait 24h, and measure fluorescence of E to obtain these data:



This looks like the graph of the data from the hyperbolic binding equation we used from the bimolecular equilibrium binding data model (shown here with adjusted variable names for this experiment):

ΘE = [W]/(KD + [W]) \Theta\_E = \frac{[W]}{K\_D + [W]}

But the experimental data has units of fluorescence, in arbitrary units (au) dependent on the instrument used to collect the data, as a function of the concentration of W.

The hyperbolic binding equation, on the other hand, uses the fraction of E bound by W, ΘE, as the Y value. But since we are measuring an experimental value (fluorescence) rather than ΘE ­directly, our equation must incorporate the nuances of this experimental measurement or the KD we incorporate will be incorrect. We need to fit fluorescence F to the data instead, and this is adjusted by a blank or baseline measurement.

F = ΘE([W]) + B

A good guess for B is the value of F when ΘE([W]) = 0. In this case, roughly 300.

No two scientific instruments will be the same. We need to add some constant term to our equation to take the variability of the instrument into account. We will represent this as A in:

F = AΘE([W]) + B

Practically, it’s easiest to look at a part of the data where you know what ΘE should be to figure out how to choose a value for A. So, a good guess for A is the fluorescence value F at saturating values of W, minus B. In this case, roughly 650-300 = 350.

**The MATLAB \*fit\* function**

How do we fit an exponential function to our data?

Specify dependent and independent variables, then type of fit.

f = fit(x,y,fittype)

regression: polynomial ‘poly1’, ‘poly2’, …, exponential ‘exp1’, ‘exp2’, …

interpolation: linear interpolation ‘linearinterp’, nearest interpolation ‘nearestinterp’

spline: ‘cubicspline’, ‘smoothingspline’

single term exponential ‘exp1’

y = aebx

a = initial population

b = growth rate

x = time

y = population

f = fit(time, pop, ‘exp1’)

output is fit object

can inspect variables like growth rate or access it directly with **f.b**

you can also call the fit variable **f** at a certain time point with **f(10.5) to see f at 10.5 hours**

**Working with custom fit types**

When fitting curves to data, we sometimes need to fit custom models. We can enter our own equation and define a custom model using the fittype function.

ft = fittype(formula,Name,Value)

Write the formula as a MATLAB expression and enclose it in single quotes.

formula = ‘K\*P0\*exp(r\*t)/(K+P0(exp(r\*t)-1))’;

Provide information about our model using name-value pairs. Include the independent and dependent variables and names of the coefficients. If there are multiple coefficients, group multiple values in curly braces.

ft = fittype(formula, …

‘Independent,‘t’, …

‘Dependent’,‘P’, …

‘Coefficients’,{‘K’,”P0’,’r’});

Next, we pass it to the FIT function along with our independent variable y=time and dependent variable x=pop.

% The following code is used to generate noisy linear data with initial slope 2 and intercept 1.

%

x = (0:100)'; % column vector of x-values;

y = 2\*x+1+randn(length(x),1); % generate colmn vector of y-data + random noise

% Enter your solution below.

fitObj = fit(x,y,'poly1')

m = fitObj.p1;

b = fitObj.p2;

Fit the linear data below defined by the vectors x and y. Take a best guess for what m and b are and assign them to m0 and b0 that will be used as starting estimates for the fit.

To find out the order in which constants must be entered, you can use:

coeffs = coeffnames(ft)

x = [1;2;3;4;5;6;7;8;9;10];

y = [5;7;9;11;13;15;17;19;21;23];

m0 = 2;

b0 = 3;

ft = fittype('m\*x + b')

f = fit(x, y, ft,'StartPoint', [b0, m0])

In the following exercise, a set of noisy linear data is generated for you and stored in the vectors x and y.

1. First, use the fittype function to create a custom linear fit type variable 'ft' which corresponds to the mathematical formula y=mx+b, i.e. with fit parameters 'm' and 'b'.
2. Then, use the fit function along with the fit type variable to fit a curve to the linear data. Make sure you also provide the fit function with initial estimates for the fit parameters.
3. Lastly, extract the slope and y-intercept of the fitted line from the fit object and store the data in the variables 'm' and 'b' respectively.

% The following code is used to generate noisy linear data with initial slope 2 and intercept 1.

%

x = (0:100)'; % column vector of x-values;

y = 2\*x+1+randn(length(x),1); % generate column vector of y-data + random noise

% Enter your solution below.

% create the custom linear fit type with fit parameters m and b

formula = 'm\*x+b';

ft = fittype(formula, ...

'Independent', 'x',...

'Dependent', 'y',...

'Coefficients', {'m','b'});

% use the fit function, the fit type variable, and initial parameter estimates

% to fit a linear curve to the data

fitObj = fit(x, y, ft,...

'StartPoint', [y(end), x(end)])

% extract the fit parameters

m = fitObj.m;

b = fitObj.b;

Now we will fit equilibrium binding data for a simple bimolecular model,

B+I\rightleftharpoons BI

Suppose B is BCR-Abl and I is imatinib. We will determine the binding affinity of the inhibitor imatinib.

Line 1 reads the .csv into MATLAB and stores the data in matrix m. The numbers 1 and 0 tell MATLAB to read values from row 2, column 1.

Line 3: defines x-axis data

Line 4: defines y-axis data

Line 5: initial guess for K\_D as k0 – you must provide this

Line 7: specifies the appropriate fitting equation – you must write a MATLAB expression using single lowercase letters for your variables

Line 8: determines best fit of the data

Line 9: plots the data and the fit. You can plot a fit in addition to x/y data by putting the fit in front of the x/y parameters like plot(YOURFIT, x, y)

m = csvread('7QBW\_Day2\_Biochem\_example1\_data.csv',1,0);

x = m(:,1);

y = m(:,2);

k0 = 1;

ft = fittype('x/(x+k)')

f = fit(x, y, ft, ...

'StartPoint', k0)

plot(f, x, y)

xlabel('nanomolar [nM]')

ylabel('fraction bound')

ft =

General model:

ft(k,x) = x/(x+k)

f =

General model:

f(x) = x/(x+k)

Coefficients (with 95% confidence bounds):

k = 72.15 (62.72, 81.59)

m = csvread('example2\_data.csv',1,0);

x = m(:,1);

y = m(:,2);

% Fit parameters

% Enter appropriate initial guesses. Delete any that are not needed

a0 = 2266;

b0 = 400;

k0 = 5000;

ft = fittype('b+a\*x/(x+k)');

f = fit(x,y,ft,'StartPoint', [a0,b0,k0])

plot(f,x,y)

xlabel('nanomolar [nM]', 'Fontsize', 12)

ylabel('arbitrary units', 'Fontsize', 12)

set(gca,'XTickLabel',{'0';'50,000';'100,000';'150,000'})

m = csvread('7QBW\_Day2\_Biochem\_example3\_data.csv',1,0);

x = m(:,1);

y = m(:,2);

% Fit parameters

% Enter appropriate initial guesses. Delete any that are not needed

a0 = 1480;

b0 = 222;

k0 = .001;

ft = fittype('b+a\*exp(-k\*x)')

f = fit(x,y,ft,'StartPoint',[a0,b0,k0])

plot(f,x,y)

xlabel('time [seconds]')

ylabel('arbitrary units')