

# Mathematical Modelling and Scientific Computing in the Biosciences

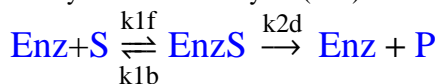
| 27 March 2007

## Lecture 2: Overview

- Singular Perturbation: Michaelis–Menten Kinetics
- Cooperative Phenomena, Hill–Function
- Motifs in Biological Circuits, Dynamical Properties
  - Negative Auto–Regulation
  - Feed–Forward Loop

## Singular Perturbation: Michaelis–Menten Kinetics

Catalytic reaction: enzyme (**Enz**) and substrate (**S**) forming complex **EnzS**, then giving rise to product (**P**) and enzyme:



Get the corresponding ODEs using Cellerator:

```
In[1]:= << cellerator.m;
EnzymeReactions = { {Enz + S ⇌ EnzS, k1f, k1b}, {EnzS → P + Enz, k2f} };
Cellerator™ 1.5.8 (1-July-2005) loaded 27-Mar-2007
09:32 using Mathematica Version 5.2 for Linux (June 20, 2005)

In[3]:= LargeBlue[x_] := StyleForm[x,
  FontColor → RGBColor[0, 0, 1], FontWeight → "Bold", FontSize → 15];
EnzymeODE = interpret[EnzymeReactions] // First // ColumnForm // LargeBlue

Out[4]//StyleForm=
Enz'[t] == k1b EnzS[t] + k2f EnzS[t] - k1f Enz[t] S[t]
EnzS'[t] == -k1b EnzS[t] - k2f EnzS[t] + k1f Enz[t] S[t]
P'[t] == k2f EnzS[t]
S'[t] == k1b EnzS[t] - k1f Enz[t] S[t]
```

## Singular Perturbation: Michaelis–Menten Kinetics

Last time, we simplified the ODE system to:

```
In[5]:= HighlightEpsRule =
  {ϵ → StyleForm[ϵ, FontColor → RGBColor[1, 0, 0], FontWeight → "Bold"]};

SimpEnzymeODE = {ϵ EnzS' [t̂] == Ŝ[t̂] - ((k1b + k2f) / (k1f STotal[0]) + 1) EnzS[t̂],
  Ŝ' [t̂] == -Ŝ[t̂] + EnzS[t̂] (k1b / (k1f STotal[0]) + Ŝ[t̂])};
% /. HighlightEpsRule // ColumnForm // LargeBlue
```

Out[7]//StyleForm=

$$\epsilon \text{EnzS}' [\hat{t}] == - \left( 1 + \frac{k1b+k2f}{k1f STotal[0]} \right) \text{EnzS} [\hat{t}] + \hat{S} [\hat{t}]$$

$$\hat{S}' [\hat{t}] == -\hat{S} [\hat{t}] + \text{EnzS} [\hat{t}] \left( \frac{k1b}{k1f STotal[0]} + \hat{S} [\hat{t}] \right)$$

with the small (dimensionless) parameter  $\epsilon \equiv \frac{\text{EnzTotal}[0]}{STotal[0]}$ .

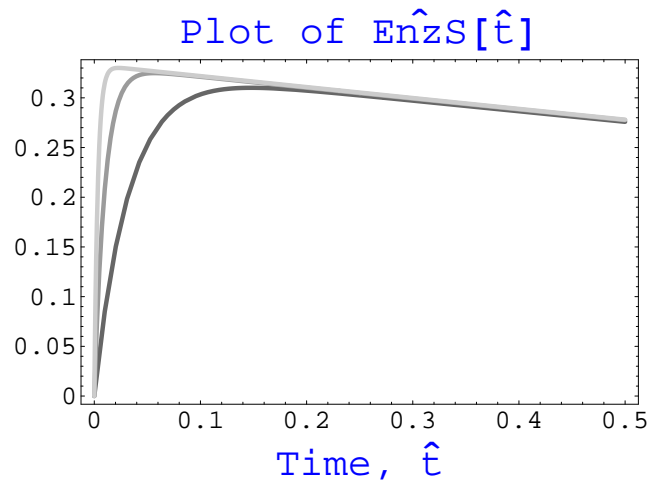


## Singular Perturbation: Michaelis–Menten Kinetics

In[210]:=

```
SolveAndPlot[EpsValue_] :=
Module[{}, ODESys = Join[SimpEnzymeODE, {EnzS[0] == 0, S[0] == 1}] /.
{k1f -> 1, k1b -> 1, k2f -> 1, STotal[0] -> 1, e -> EpsValue};
ODESoln = NDSolve[ODESys, {EnzS, S}, {t, 0, 1}];
Plot[EnzS[t] /. ODESoln, {t, 0, 0.5},
PlotStyle -> {GrayLevel[Mod[-0.4*Log[10, EpsValue], 1]], Thickness[0.008]},
Frame -> True, FrameLabel -> (LargeBlue /@ {"Time, t", ""}),
RotateLabel -> False, PlotRange -> All, DisplayFunction -> Identity];

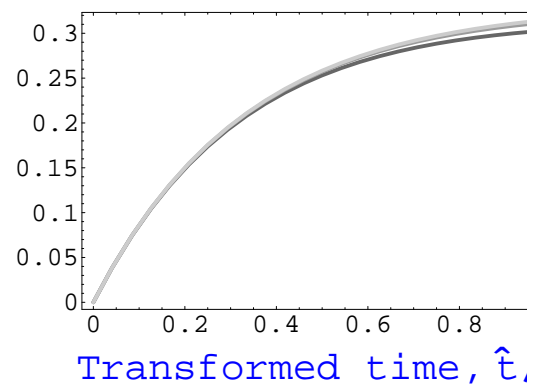
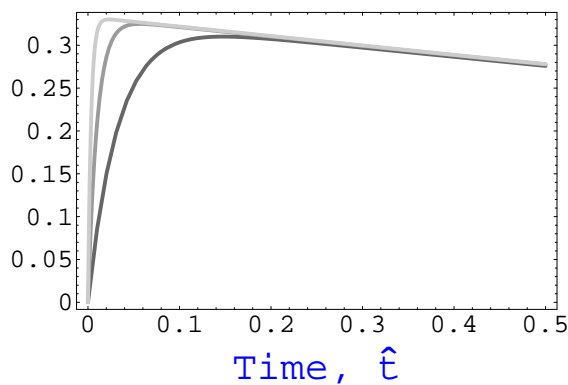
Show[SolveAndPlot /@ {0.1, 0.03, 0.01}, DisplayFunction -> $DisplayFunction,
PlotLabel -> ("Plot of EnzS[t]" // LargeBlue), ImageSize -> {400, 180}];
```



## Singular Perturbation: Michaelis–Menten Kinetics

In[244]:=

```
SolveAndPlotBlownUp[EpsValue_] := Module[{},
  ODESys = Join[SimpEnzymeODE, {EnzS[0] == 0, S[0] == 1}] /.
    {k1f → 1, k1b → 1, k2f → 1, STotal[0] → 1, ε → EpsValue};
  ODESoln = NDSolve[ODESys, {EnzS, S}, {t̂, 0, 1}];
  Plot[EnzS[t̂ * EpsValue] /. ODESoln, {t̂, 0, 1},
    PlotStyle → {GrayLevel[Mod[-0.4 * Log[10, EpsValue], 1]], Thickness[0.008]},
    Frame → True, FrameLabel → {LargeBlue /@ {"Transformed time, t̂/ε", ""}},
    RotateLabel → False, PlotRange → All, DisplayFunction → Identity];
  EpsList = {0.1, 0.03, 0.01};
  PlotOrig = SolveAndPlot /@ EpsList // Show;
  PlotBlownUp = SolveAndPlotBlownUp /@ EpsList // Show;
  Show[({PlotOrig, PlotBlownUp} // GraphicsArray), DisplayFunction → $DisplayFunction,
    ImageSize → {500, 180}];
```



6 of 25

## Singular Perturbation: Michaelis–Menten Kinetics

Observation: it seems that in terms of the new (magnified) time variable  $\tau \equiv \frac{\hat{t}}{\epsilon}$ , the inner (singular) solution is essentially independent of parameter  $\epsilon$ .

Regular perturbation failed because the solution is **not analytic** in the perturbative parameter  $\epsilon$ . Maybe the inner solution is an analytic function when in terms of the new variable  $\tau$ . Try this time–transformation.

```
In[124]:=
OrigODE =
{ε EnzS_I' [t̂] == S_I [t̂] - ((k1b+k2f) / (k1f STotal[0]) + 1) EnzS_I [t̂],
  S_I' [t̂] == -S_I [t̂] + EnzS_I [t̂] (k1b / (k1f STotal[0]) + S_I [t̂])};

Inner_ODE_IC = Join[%, {EnzS_I[0] == 0, S_I[0] == 1}];
(% /. {Derivative[n_][Func_][t_] → Derivative[n][Func][t] / e^n}) //
Simplify[#, Assumptions → {ε > 0}] &;
Inner_ODE_IC = % /. {t̂ → τ};
% /. HighlightEpsRule // ColumnForm

Out[128]=
EnzS_I[τ] (1 +  $\frac{k1b+k2f}{k1f STotal[0]}$ ) + EnzS_I' [τ] == S_I[τ]
EnzS_I[τ] ε ( $\frac{k1b}{k1f STotal[0]}$  + S_I[τ]) == ε S_I[τ] + S_I' [τ]
EnzS_I[0] == 0
S_I[0] == 1
```

7 of 25

## Singular Perturbation: Michaelis–Menten Kinetics

Now do perturbative analysis: expand the solution in terms of the small parameter,  $\epsilon$ :

```
In[222]:=
MaxOrd = 1;
S_Inner[t_] :=  $\sum_{i=0}^{\text{MaxOrd}} \epsilon^i S_I[i][t]$ ; EnzS_Inner[t_] :=  $\sum_{i=0}^{\text{MaxOrd}} \epsilon^i \text{EnzS}_I[i][t]$ ;
```

That is, we have expanded the inner solution as:

```
In[224]:=
{S_Inner[t], EnzS_Inner[t]} /. HighlightEpsRule

Out[224]=
{S_I[0][t] + ε S_I[1][t], EnzS_I[0][t] + ε EnzS_I[1][t]}
```

Therefore, the inner ODE system is:

```
In[225]:=
Expanded_ODE_IC =
Inner_ODE_IC /. {S_I → S_Inner, EnzS_I → EnzS_Inner} // Simplify;
% /. HighlightEpsRule // ColumnForm // StyleForm[#, FontSize -> 7] &

Out[226]//StyleForm=
(1 +  $\frac{k1b+k2f}{k1f STotal[0]}$ ) (EnzS_I[0][τ] + ε EnzS_I[1][τ]) + EnzS_I[0]' [τ] + ε EnzS_I[1]' [τ] == S_I[0][τ] + ε S_I[1][τ]
ε (EnzS_I[0][τ] + ε EnzS_I[1][τ]) ( $\frac{k1b}{k1f STotal[0]}$  + S_I[0][τ] + ε S_I[1][τ]) == ε (S_I[0][τ] + ε S_I[1][τ]) + S_I[0]' [τ] + ε
EnzS_I[0][0] + ε EnzS_I[1][0] == 0
S_I[0][0] + ε S_I[1][0] == 1
```

8 of 25

## Singular Perturbation: Michaelis–Menten Kinetics

Now let's obtain ODEs for each order. **Zeroth order** equations for the inner (singular) solution:

```
In[259]:=
<< PerturbationODE.m
OrderList = PolyOrderList[InnerODEIC, {S_I[τ], EnzS_I[τ]}, ε, 1];
OrderList[[1]] // ColumnForm

Out[261]=
EnzS_I[0][τ] +  $\frac{k_{1b} \text{EnzS\_I}[0][\tau]}{k_{1f} \text{STotal}[0]} + \frac{k_{2f} \text{EnzS\_I}[0][\tau]}{k_{1f} \text{STotal}[0]} + \text{EnzS\_I}[0]'[\tau] = \text{S\_I}[0][\tau]$ 
0 = S_I[0]'[τ]
EnzS_I[0][0] == 0
S_I[0][0] == 1

In[262]:=
Order0Soln =
DSolve[OrderList[[1]], {EnzS_I[0], S_I[0]}, τ // Simplify // Flatten;
% // ColumnForm

Out[263]=
EnzS_I[0] → Function[{τ},  $-\frac{(-1 + e^{\frac{\tau (-k_{1b} - k_{2f} - k_{1f} \text{STotal}[0])}{k_{1f} \text{STotal}[0]}) k_{1f} \text{STotal}[0]}{k_{1b} + k_{2f} + k_{1f} \text{STotal}[0]}$ ]
S_I[0] → Function[{τ}, 1]
```

**First order** equations for the inner (singular) solution:

```
In[264]:=
Order1Soln =
DSolve[OrderList[[2]] /. Order0Soln, {EnzS_I[1], S_I[1]}, τ // Simplify // Flatten;
```



9 of 25

## Singular Perturbation: Michaelis–Menten Kinetics

```
In[265]:=
PositivityAssumption = {STotal[0] > 0, k1f > 0, k1b > 0, k2f > 0};
LimitEnzS_Inner_Order0 =
Limit[EnzS_I[0] /. Order0Soln // Simplify // Last, τ → Infinity, Assumptions → %]

Out[266]=
 $\frac{k_{1f} \text{STotal}[0]}{k_{1b} + k_{2f} + k_{1f} \text{STotal}[0]}$ 

In[267]:=
LimitS_Inner_Order0 = Limit[S_I[0] /. Order0Soln // Simplify // Last, τ → Infinity]

Out[267]=
1
```

### Inner/Outer Solutions: Matching Condition

In the limit of  $\epsilon \rightarrow 0$ , the width of the boundary layer tends to zero.

Therefore, for the outer solution, its value at the boundary layer  $\rightarrow$  value at  $t = 0$ .

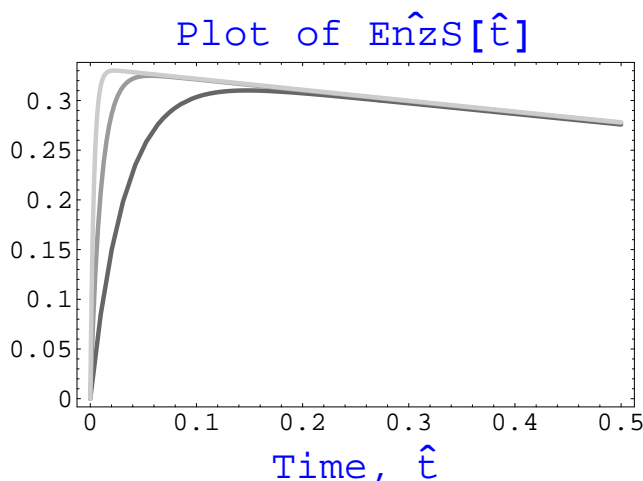
However, in the same limit of  $\epsilon \rightarrow 0$ , we have  $\tau \equiv \frac{t}{\epsilon} \rightarrow \infty$ .

Therefore, **matching condition** from continuity of solution requires that:

$$\lim_{\tau \rightarrow \infty} \{S\_Inner[\tau], \text{EnzS\_Inner}[\tau]\} = \lim_{t \rightarrow 0} \{S\_Outer[t], \text{EnzS\_Outer}[t]\}$$

Thus, the inner solution provides the initial conditions ( $t=0$ ) for the outer solution.

## Singular Perturbation: Michaelis–Menten Kinetics



### Inner/Outer Solutions: Matching Condition

In the limit of  $\epsilon \rightarrow 0$ , the width of the boundary layer tends to zero.

Therefore, for the outer solution, its value at the boundary layer  $\rightarrow$  value at  $t = 0$ .

However, in the same limit of  $\epsilon \rightarrow 0$ , we have  $\tau \equiv \frac{t}{\epsilon} \rightarrow \infty$ .

Therefore, **matching condition** from continuity of solution requires that:

$$\lim_{\tau \rightarrow \infty} \{S\_Inner[\tau], \text{EnzS\_Inner}[\tau]\} = \lim_{t \rightarrow 0} \{S\_Outer[t], \text{EnzS\_Outer}[t]\}$$

Thus, the inner solution provides the initial conditions ( $t=0$ ) for the outer solution.

## Singular Perturbation: Michaelis–Menten Kinetics

Using singular perturbation, we have derived the correct IC for the outer solution:

$$\text{In}[44] := \text{CorrectOuterIC} = \left\{ S\_Outer[0] == 1, \text{EnzS\_Outer}[0] == \frac{k1f S_{\text{Total}}[0]}{k1b + k2f + k1f S_{\text{Total}}[0]} \right\};$$

Compare this to the (inconsistent) IC obtained from regular perturbation (last lecture):

```
In[45]:= WrongOuterIC = {S_Outer[0] == 1, EnzS_Outer[0] == 0};
```

In particular, look at the outer ODE, at zeroth order in  $\epsilon$ :

```
In[46]:= OuterOrder0ODE =
  SimpEnzymeODE /. {EnzS → EnzS_Outer, S → S_Outer} /. {ε → 0} // Simplify;
% // ColumnForm
```

```
Out[47]= EnzS_Outer[τ] (1 +  $\frac{k1b+k2f}{k1f STotal[0]}$ ) == S_Outer[τ]
EnzS_Outer[τ] ( $\frac{k1b}{k1f STotal[0]}$  + S_Outer[τ]) == S_Outer[τ] + S_Outer'[τ]
```

```
In[82]:= Together // @ (OuterOrder0ODE[[1]] /. {τ → 0})
```

```
Out[82]=  $\frac{EnzS_Outer[0] (k1b + k2f + k1f STotal[0])}{k1f STotal[0]} == S_Outer[0]$ 
```



12 of 25

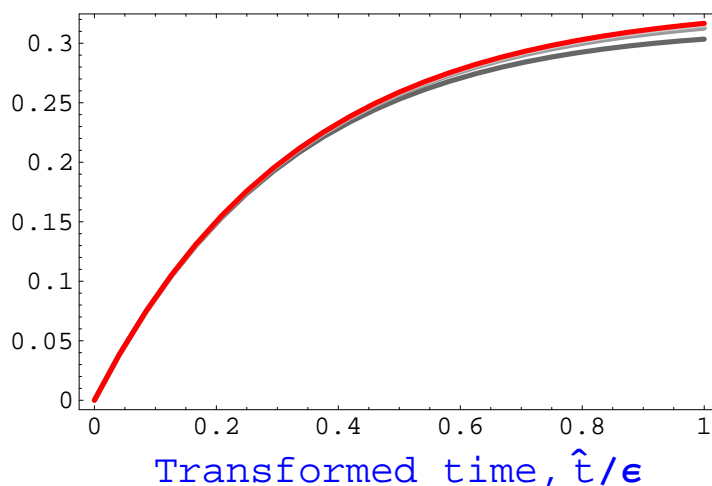
## Michaelis–Menten Kinetics: Inner Solution

```
In[49]:= EnzS_IOrder0[τ_] := Evaluate[EnzS_I[0] /. Order0Soln // Last];
HighlightRule[x_] := x → StyleForm[x, FontColor → RGBColor[0.0, 0.0, 1], FontSize → 12];
EnzS_IOrder0[τ] /. HighlightRule[τ]
```

```
Out[51]= 
$$-\frac{\left(-1 + e^{\frac{(-k1b-k2f-k1f STotal[0]) \tau}{k1f STotal[0]}}\right) k1f STotal[0]}{k1b + k2f + k1f STotal[0]}$$

```

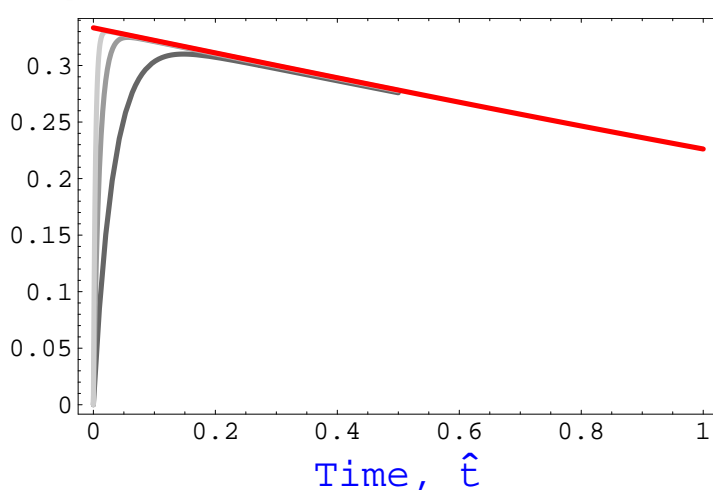
```
In[52]:= UnitConstantsRule = {STotal[0] → 1, k1b → 1, k1f → 1, k2f → 1};
EnzS_IOrder0Plot = Plot[EnzS_IOrder0[τ] /. %, {τ, 0, 1},
  PlotStyle → {Hue[1], Thickness[0.008]}, DisplayFunction → Identity];
Show[PlotBlownUp, EnzS_IOrder0Plot, DisplayFunction → $DisplayFunction];
```



13 of 25

## Michaelis–Menten Kinetics: Outer Solution

```
In[55]:= EnzS_Order0OuterRule = Solve[OuterOrder0ODE[[1]], EnzS_Out[ $\hat{t}$ ]] // Flatten;
S_OutODE = OuterOrder0ODE[[2]] /. EnzS_Order0OuterRule // Simplify;
S_OutSoln = NDSolve[Join[{}], {S_Out[0] == 1}] /. UnitConstantsRule,
  S_Out, { $\hat{t}$ , 0, 1}] // Flatten;
EnzS_OutSoln = EnzS_Order0OuterRule /. S_OutSoln /. UnitConstantsRule;
EnzS_Order0Plot = Plot[EnzS_Out[ $\hat{t}$ ] /. %, { $\hat{t}$ , 0, 1},
  PlotStyle -> {Hue[1], Thickness[0.008]}, DisplayFunction -> Identity];
Show[PlotOrig, %, DisplayFunction -> $DisplayFunction];
```

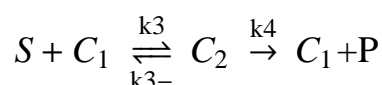
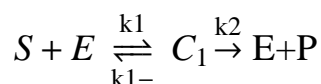


**Conclusion:** the outer perturbative solution captures the long-time dynamics well, especially for small values of  $\epsilon$ ; the initial behavior is well-captured by the inner solution.

14 of 25

## Cooperative Phenomenon

- Substrate binds to the enzyme on the [binding site](#).
- There are some enzymes with more than 1 binding site.
- Reaction between enzyme and substrate is called [cooperative](#), if when the enzyme is bound to a substrate molecule, other substrate molecule can bind to the remaining binding sites.
- Indirect interaction between distinct binding sites: [allosteric effect](#).



- Again, assume  $\epsilon \equiv \frac{\text{EnzymeTotal}}{\text{STotal}} \ll 1$

- From singular perturbation analysis, obtain expression for the substrate flux/reaction velocity (see book by J. Murray, *Intro. Math. Biol.*):

$$\left| \frac{ds}{dt} \right| = \frac{\text{EnzTotal STotal} (k_2 K_m' + k_4 \text{STotal})}{K_m K_m' + K_m' \text{STotal} + \text{STotal}^2}$$

15 of 25

## Cooperative Phenomenon: Hill–Function

- When cooperativity is suspected, usually assume reaction velocity to take the following **Hill–function** form:

$$\left| \frac{ds}{dt} \right| = \frac{Q * \text{STotal}^n}{K_m + \text{STotal}^n};$$

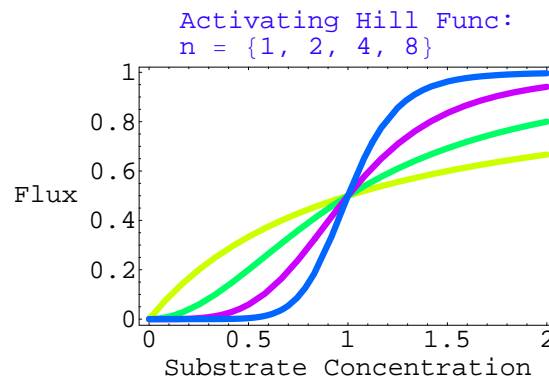
where  $n$  is usually taken to be an integer. Typically,  $2 \leq n \leq 4$ .

- $n$  takes the role number of molecules taking part in the reaction.

In the case  $n = 1$ , there is no cooperativity; reduce to Michaelis–Menten, as analyzed.

Larger  $n$ , higher non–linearity.

```
In[78]:= Hill_Function[x_, n_] := Q x^n / (Km + x^n); ListOfN = {1, 2, 4, 8};
Hill_Plots = Map[
  Plot[Hill_Function[x, #] /. {Q -> 1, Km -> 1}, {x, 0, 2}, DisplayFunction -> Identity,
    PlotStyle -> {Hue[#/5.0], Thickness[0.015]}] &, ListOfN];
ActivatingHill = Show[%, DisplayFunction -> $DisplayFunction, Frame -> True,
  FrameLabel -> {"Substrate Concentration", "Flux"}, RotateLabel -> False,
  PlotLabel -> ("Activating Hill Func:\nn = " <> ToString[ListOfN] //
    StyleForm[#, FontColor -> Hue[0.7]] &), ImageSize -> {500, 140}];
```

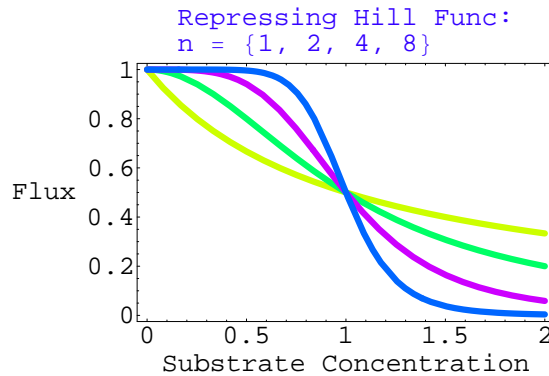


16 of 25

## Cooperative Phenomenon: Hill–Function

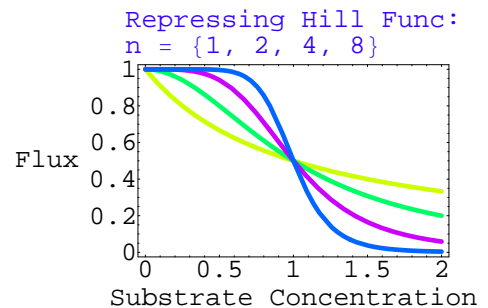
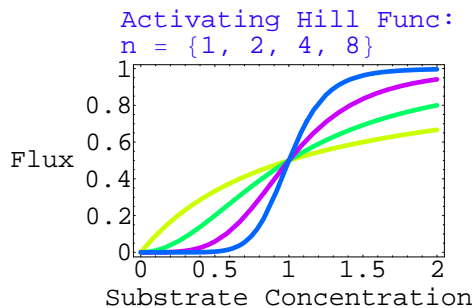
Besides **activating** Hill–functions, **repressing** Hill–functions are also commonly used in biological modelling.

```
In[81]:= Repressing_Hill_Function[x_, n_] := Q / (Km + x^n);
Repressing_Hill_Plots = Map[
  Plot[Repressing_Hill_Function[x, #] /. {Q -> 1, Km -> 1}, {x, 0, 2}, DisplayFunction ->
    Identity, PlotStyle -> {Hue[#/5.0], Thickness[0.015]}] &, ListOfN];
RepressingHill = Show[%, DisplayFunction -> $DisplayFunction, Frame -> True,
  FrameLabel -> {"Substrate Concentration", "Flux"}, RotateLabel -> False,
  PlotLabel -> ("Repressing Hill Func:\nn = " <> ToString[ListOfN] //
    StyleForm[#, FontColor -> Hue[0.7]] &), ImageSize -> {500, 140}];
```



## Gene Regulation

```
In[85]:= Show[GraphicsArray[{ActivatingHill, RepressingHill}], ImageSize -> { 500, 120}];
```

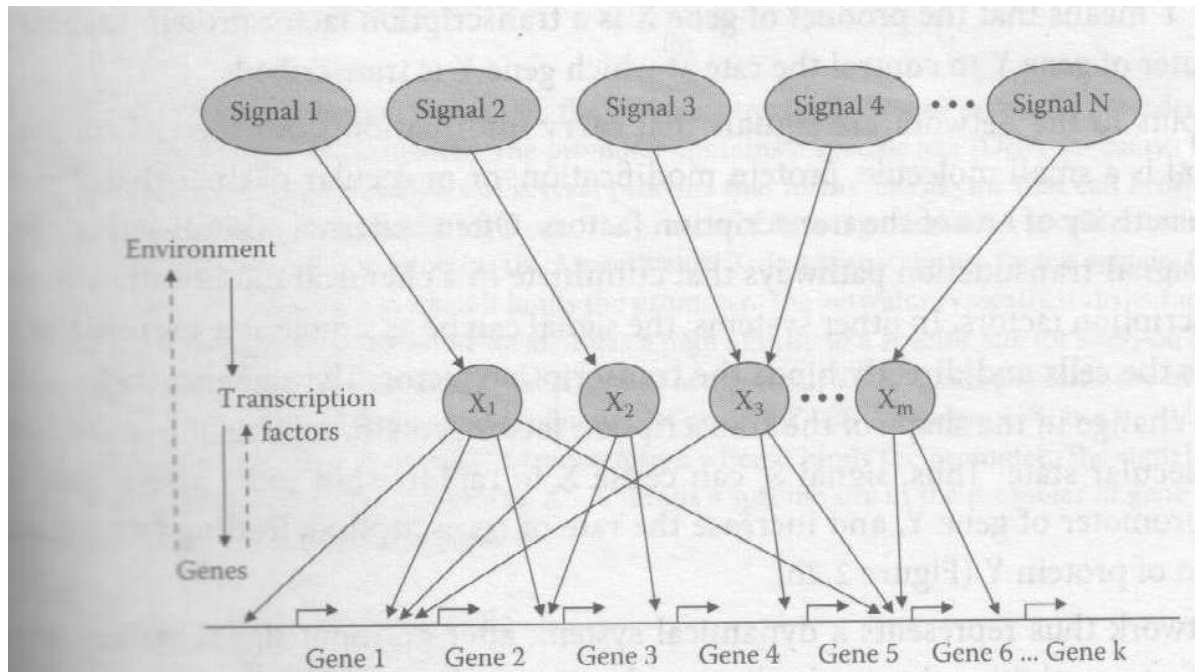


- Such activators (positive control) and repressors (negative control) are common in [gene regulation](#)
- Cell: thousands of interacting genes, acting together to respond to diverse signals
  - **external**: temperature, level of nutrients, harmful chemicals
  - **internal**: level of metabolites, damage to proteins, DNA, etc
- Information-processing capability: biological circuitry
- Environment  $\Leftrightarrow$  [transcription factors](#) (proteins transiting between active and inactive molecular states)
  - $\Leftrightarrow$  [genes](#)
- Set of interactions: [transcription network](#)

## Gene Regulation

- Schematic of interactions between the [environment](#), [transcription factors](#), [genes](#) (U. Alon, *Intro. Systems Biology*):

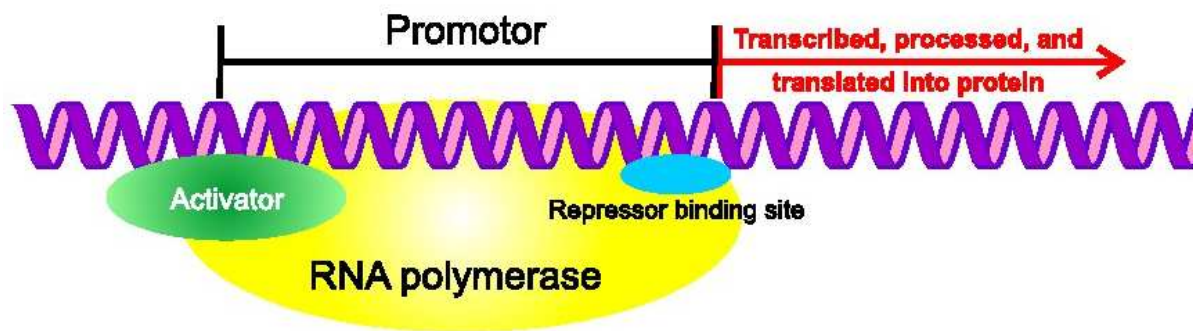
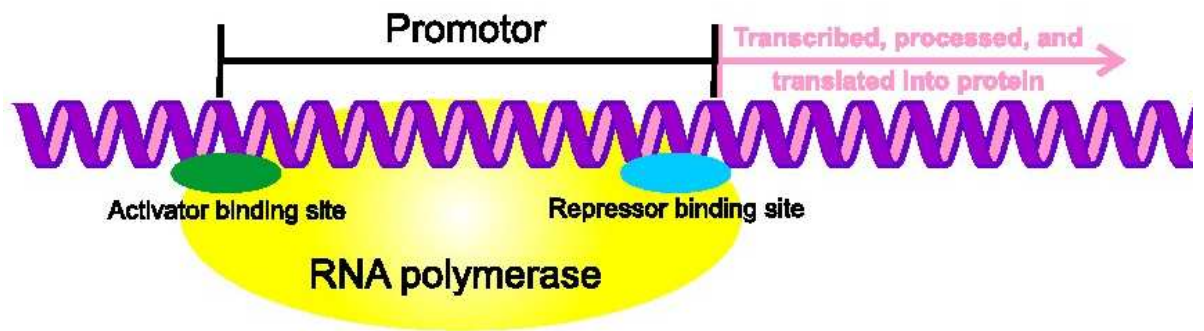
```
In[86]:= Import["~/Teaching/NoteBooks/Lecture3/GeneNetwork.jpg", "JPG"] //
Show[#, ImageSize -> {500, 250}] &;
```



- How do transcription factors influence the [on/off](#) states of genes?

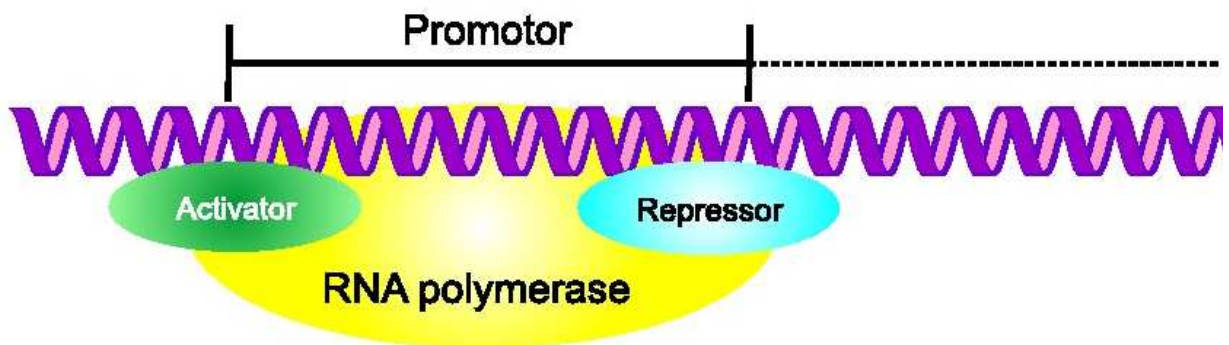
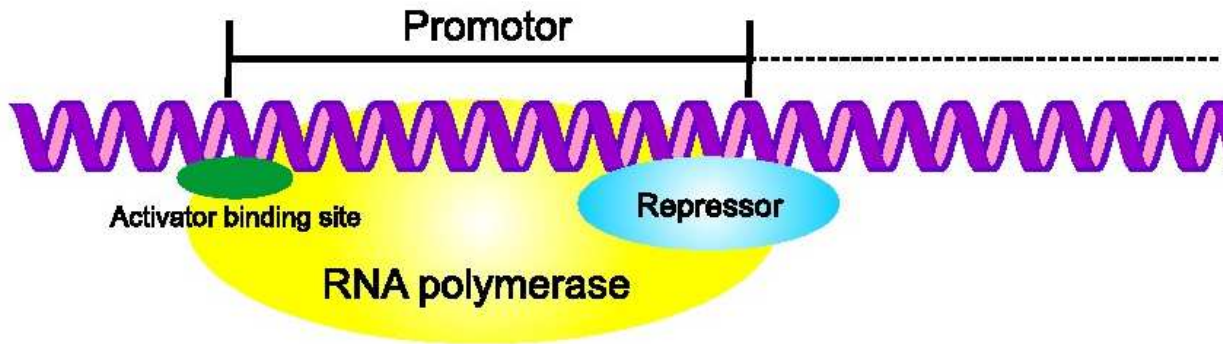
## Gene Regulation

```
In[87]:= Show[Import["~/Teaching/NoteBooks/Lecture3/State_Active.jpg", "JPG"],  
ImageSize -> {500, 320}];
```



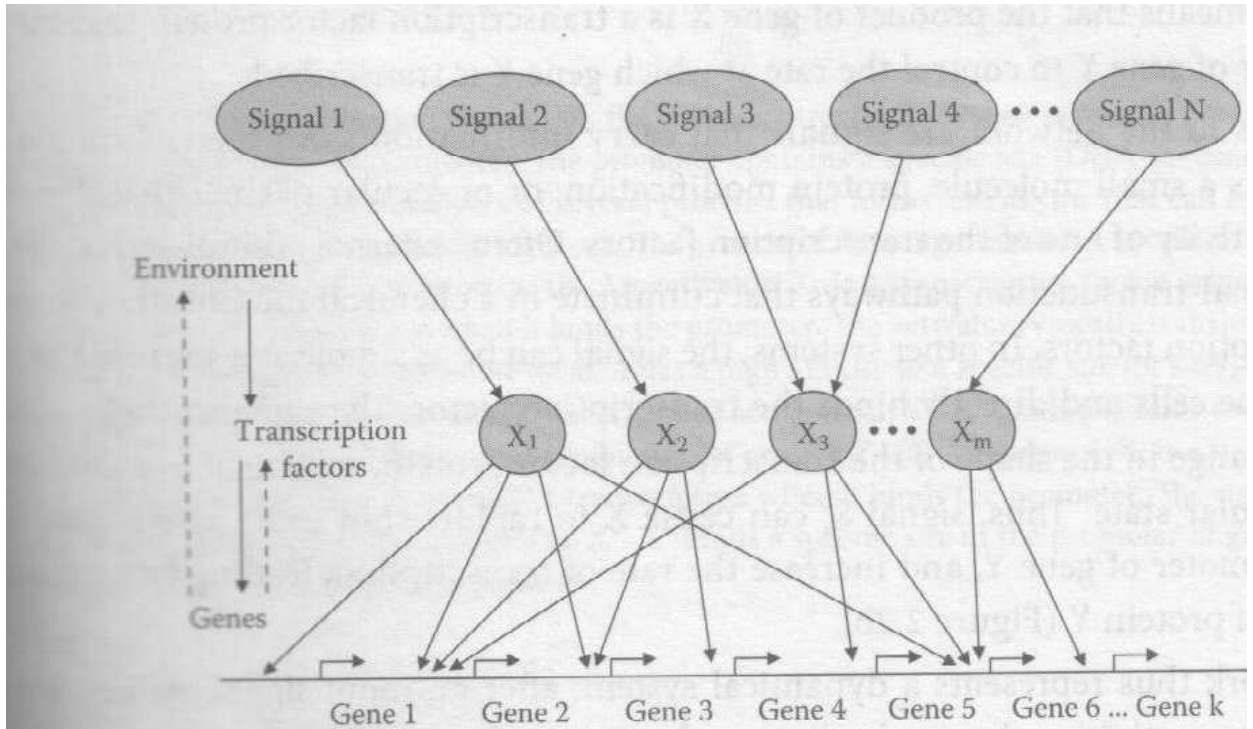
## Gene Regulation

```
In[88]:= Show[Import["~/Teaching/NoteBooks/Lecture3/State_Inactive.jpg", "JPG"],  
ImageSize -> {500, 320}];
```



## Transcription Network

```
In[89]:= Import["~/Teaching/NoteBooks/Lecture3/GeneNetwork.jpg", "JPG"];
Show[%, ImageSize -> {500, 280}];
```



- What kind of [patterns/motifs](#) are observed in gene transcription networks?

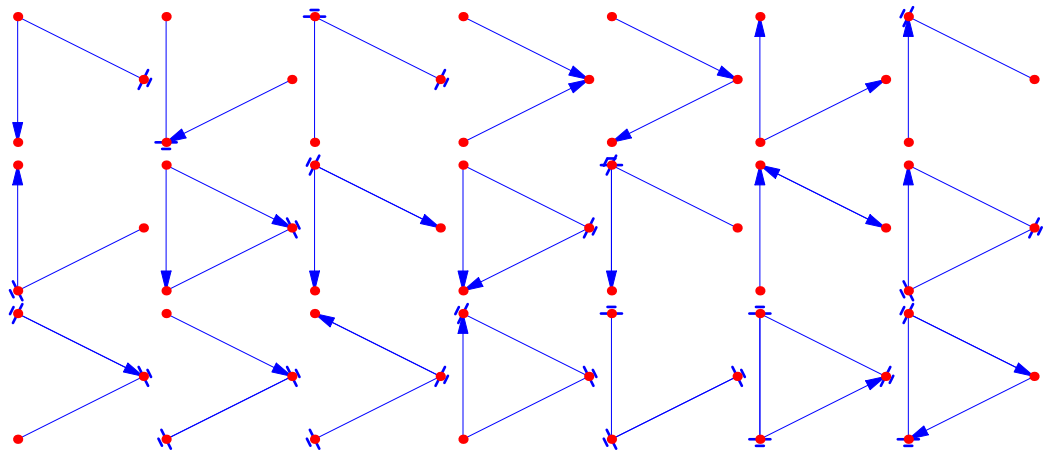
## Gene Networks

We can generate a list of random, connected, 3–node, activator/repressor graphs:

```

In[112]:=
<< DiscreteMath`Combinatorica`
PlotNum = 80; Prob = 0.3; NumVert = 3;
RandomGraphList =
Table[RandomGraph[NumVert, Prob, Type -> Directed], {PlotNum}];
DistinctConnectedGraphs = Pick[%, Map[(ConnectedQ[#] == True) &, %]] // Union;
RepActArrow[x_, y_] := Module[{}, Arrow[x, y, Hold[If[Random[Integer] == 0, HeadShape -> {{Thickness[0.02], Line[{{0, .08}, {0, -.08}}]}],
{Thickness[0.02], Line[{{0.1/2, .03}, {0.1/2, -.03}}]}], HeadShape -> Automatic]], HeadLength -> 0.15, HeadWidth -> 0.5];
GraphPlots = Map[
(ShowGraph[#, HeadLength -> 0.15, HeadWidth -> 0.5, EdgeColor -> Blue, VertexStyle -> Disk[Large],
Frame -> None, FrameTicks -> None, VertexColor -> Red, PlotRange -> All,
DisplayFunction -> Identity] /. Arrow[x_, y_, z_] -> RepActArrow[x, y]) &,
DistinctConnectedGraphs] // ReleaseHold;
Show[GraphicsArray[Partition[%, 7]], ImageSize -> {500, 170}, PlotRange -> All];

```



23 of 25

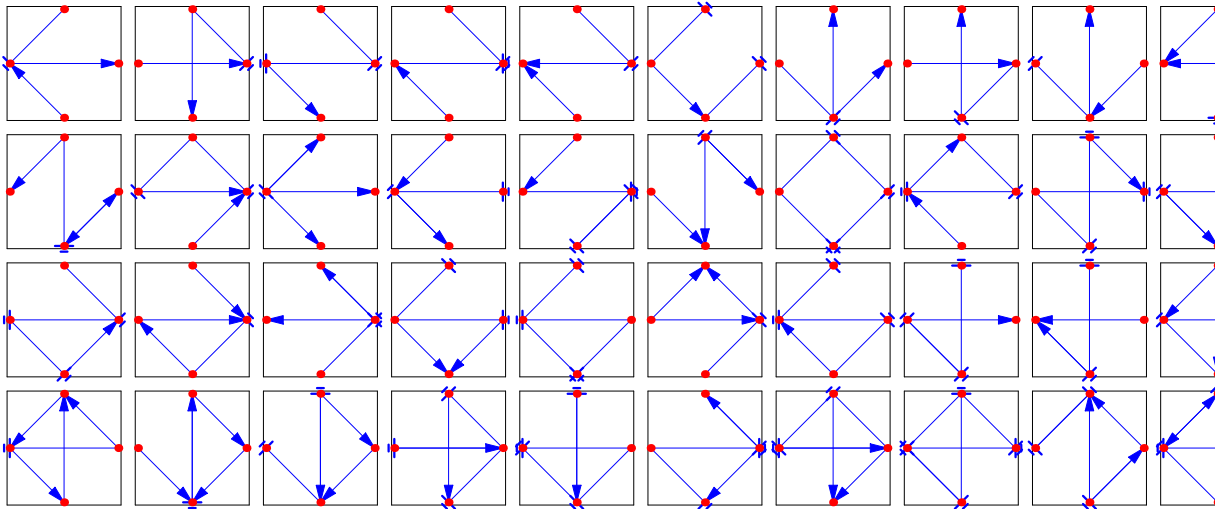
## Gene Networks

A larger list of 4-node graphs:

```

In[119]:=
NumVert = 4;
RandomGraphList =
  Table[ RandomGraph[NumVert, Prob, Type -> Directed], {PlotNum} ];
DistinctConnectedGraphs = Pick[%, Map[(ConnectedQ[#] == True) &, %]] // Union;
GraphPlots = Map[(ShowGraph[#, HeadLength -> 0.15,
  HeadWidth -> 0.5, EdgeColor -> Blue, VertexStyle -> Disk[Large],
  Frame -> True, FrameTicks -> None, VertexColor -> Red, PlotRange -> All,
  DisplayFunction -> Identity] /. Arrow[x_, y_, z_] -> RepActArrow[x, y]) &,
  DistinctConnectedGraphs] // ReleaseHold;
Show[GraphicsArray[Partition[%, 10]], ImageSize -> {500, 210}, PlotRange -> All];

```



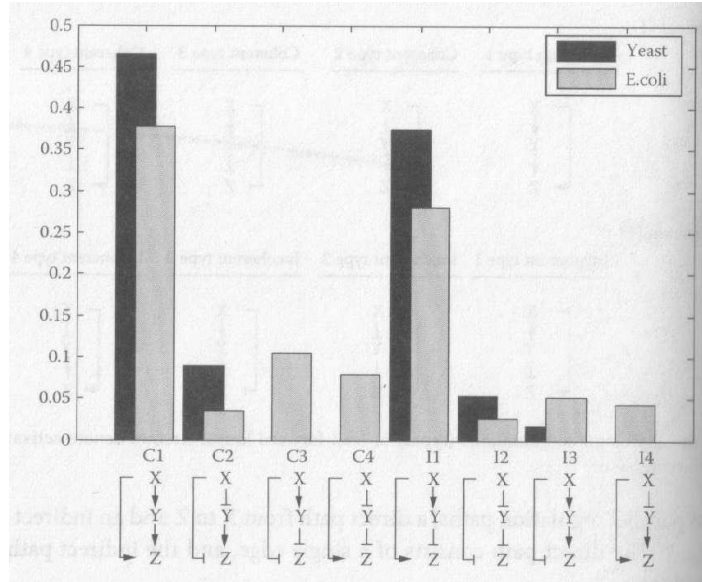
24 of 25

## Gene Networks

- It has been found that gene transcription networks are not random graphs:
  - **Motifs** occur much more abundantly
  - **Anti-Motifs** occur much more rarely

Relative abundance of 8 **Feed-Forward-Loop** (FFL) types in the transcription network of yeast and *E. coli*. (from *Mangan et. al, 2006*):

```
In[78]:= Import["~/Teaching/NoteBooks/Lecture3/FeedForwardMotifOccurrence.jpg", "JPG"];
Show[%, ImageSize -> {500, 220}];
```



25 of 25

## Network Motifs and Dynamical Implications

- Given that mutations occur randomly, it is probable that the abundance of observed motifs confer some dynamical properties that for its selection:
  - robustness to noise (environmental fluctuations)
  - information processing capabilities
- Negative Autoregulation:
  - insensitive to protein production rate (a noisy process)
- Feed-Forward Loop:
  - sign-sensitive delay
  - persistence detector/filtering of brief fluctuating signals