3. Fixed points, Gröbner bases, & computational algebra

Goal: Develop techniques that can be used to analyze models where the state space (2^n nodes) is large.

First, consider a more refined model for the lac operon.

Add variables:

$\left.\begin{array}{ll} P & \text{lac permease} \\ B & \beta\text{-galactosidase} \end{array}\right\}$ instead of E; lac Z polypeptide

- A allolactose
- R repressor protein lac I
- C catabolite activator protein CAP

New feature: Distinguish between 3 levels of lactose & allolactose:

none, low, high.

This is in contrast to proteins & enzymes, which are either in abundance, or absent (concentration levels when expressed are thousands of times lower than when they're not.)

Several ways to do this:

(i) Use states $S = \{0, 1, 2\}$ instead of $\{0, 1\}$

0 = none
1 = low
2 = high

or (ii) Introduce additional variables: L_ℓ and A_ℓ that represent "low levels."

- No lactose: $L_\ell = 0$, $L = 0$
- low lactose: $L_\ell = 1$, $L = 0$
- high lactose: $L_\ell = 1$, $L = 1$

The other possibility, $L_\ell = 0$, $L = 1$ is meaningless; ignore it.

Assumption: High levels of lactose or allolactose at time t means at least low levels at time $t+1$.

Proposed model:

$f_M = \overline{R} \wedge C$ "no repressor protein & high concent. of CAP"

$f_P = M$

$f_C = \overline{G_e}$

$f_A = L \wedge B$

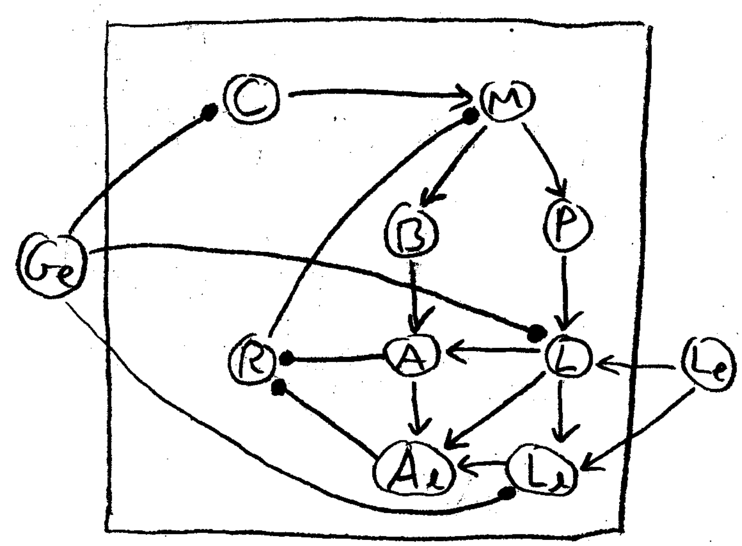
$f_L = \overline{G_e} \wedge P \wedge L_e$

$f_B = M$

$f_R = \overline{A} \wedge \overline{A_\ell}$ "no allactose"

$f_{A_\ell} = A \vee L \vee L_e$

$f_{L_\ell} = \overline{G_e} \wedge (L \vee L_e)$



wiring diagram

<u>Problem</u>: State space has size $2^9 = 512$ nodes.

This is managable (barely).

But some Boolean networks are too big.

<u>Ex</u>: A model for T cell receptor signaling contains 94 nodes, so $2 \cdot 10^{28}$ nodes in the state space.

<u>Goal</u>: How do we find the fixed points easily?

<u>Recall</u>: A fixed point $(p_1, p_2, \ldots, p_n)$ satisfies

$$
\begin{aligned}
p_1 &= f_{x_1}(p_1, \ldots, p_n) \\
p_2 &= f_{x_2}(p_1, \ldots, p_n) \\
&\vdots \\
p_n &= f_{x_n}(p_1, \ldots, p_n)
\end{aligned}
$$

<u>Approach</u>: Computational algebra (= abstract algebra) using algebraic geometry (= polynomial algebra) & <u>Gröbner bases</u>.

Problem rephrased: Find sol'ns to the system of <u>polynomial eq'ns</u>:

$$
\begin{cases}
f_{x_1}(x_1, \ldots, x_n) - x_1 = 0 \\
f_{x_2}(x_1, \ldots, x_n) - x_2 = 0 \\
\quad \vdots \qquad\qquad\qquad \vdots \\
f_{x_n}(x_1, \ldots, x_n) - x_n = 0
\end{cases}
$$

Step 1: Write functions in polynomial form.

Recall
$$x_1 \wedge x_2 = x_1 x_2$$
$$x_1 \vee x_2 = x_1 + x_2 + x_1 x_2$$
$$\overline{x_1} = x_1 + 1$$

Ex:
$$\begin{aligned} f_{A_e} = A \vee L \vee L_e &= (A \vee L) \vee L_e \\ &= (A + L + AL) \vee L_e \\ &= A + L + AL + L_e + (A + L + AL) L_e \\ &= A + L + AL + L_e + AL_e + LL_e + ALL_e \end{aligned}$$

Do this for remaining 8 functions (exercise).

Step 2: Use computational algebra software to find a Gröbner basis of these polynomials.

Gröbner bases are a generalization of Gaussian elimination, but for systems of polynomials instead of linear equation.

Gaussian elimination: Input linear system.

e.g., $\begin{cases} x + 2y = 1 \\ 3x + 8y = 1 \end{cases}$ or

$$\left[\begin{array}{cc|c} 1 & 2 & 1 \\ 3 & 8 & 1 \end{array}\right] \sim \left[\begin{array}{cc|c} 1 & 2 & 1 \\ 0 & 2 & -2 \end{array}\right] \sim \left[\begin{array}{cc|c} 1 & 0 & 3 \\ 0 & 2 & -2 \end{array}\right] \sim \left[\begin{array}{ccc} 1 & 0 & 3 \\ 0 & 1 & -1 \end{array}\right]$$

Ouput: A simpler system e.g., $\begin{cases} x = 3 \\ y = -1 \end{cases}$ with the same solutions!

Remark: IF the output system is "uppertriangular", then we can back-substitute & solve completely. e.g.,
$$\begin{cases} x + \quad z = 2 \\ y - z = 8 \\ 0 = 0 \end{cases}$$

Gröbner bases give us a much simplier set of polynomials that have the same solution set as the original system.

The theory is deep, but it's implemented in software packages.

* Free open source package: $\boxed{\text{SAGE}}$ www.sagemath.org

Select "Try Sage online" & create a notebook account. (sagenborg)

Example: let's solve the following system:
$$\begin{cases} x^2+y^2+z^2-1=0 \\ x^2+z^2-y=0 \\ x-z=0 \end{cases}$$

Enter in SAGE (hit Shift + Return after each line

```
P.<x,y,z>=PolynomialRing(RR, 3, order='lex'); P
I = ideal(x^2 + y^2 + z^2 - 1, x^2 + z^2 - y, x-z); I
B = I.groebner_basis(); B
```

Output: [x-z, y-2*z^2, z^4+1/2*z^2-1/4]

That is,
$$\begin{cases} z^4+\frac{1}{2}z^2-\frac{1}{4}=0 \\ y-2z^2=0 \\ x-z \end{cases}$$
now back-substitute!

<u>Note</u>: This system is "upper triangular", i.e., we can

- Solve for z in Eq 1
- Plug into Eq 2 & solve for y
- Plug into Eq 3 & solve for x.

<u>Solns</u>: $z = \pm\sqrt{\frac{-1+\sqrt{5}}{4}}$; $y = 2z^2$; $x = z$

$\Rightarrow$ $z = \sqrt{\frac{-1+\sqrt{5}}{4}}$; $y = \frac{-1+\sqrt{5}}{2}$; $x = \sqrt{\frac{1+\sqrt{5}}{4}}$ ← Soln 1

and $z = -\sqrt{\frac{-1+\sqrt{5}}{4}}$; $y = \frac{-1+\sqrt{5}}{2}$; $x = -\sqrt{\frac{-1+\sqrt{5}}{4}}$ ← Soln 2

<u>lac operon example</u>: $\underline{f_{x_i}(x_1,\dots,x_n) - x_i}$ (recall: $x_i = -x_i$)

(Put $L_e = a = 0$, $G_e = g = 0$)

M	x_1	$x_1 + x_4 x_5 + x_4 = 0$
P	x_2	$x_1 + x_2 = 0$
B	x_3	$x_1 + x_3 = 0$
C	x_4	$x_4 + (g+1) = 0$
R	x_5	$x_5 + x_6 x_7 + x_6 + x_7 + 1 = 0$
A	x_6	$x_6 + x_3 x_8 = 0$
A_ℓ	x_7	$x_8 + x_7 + x_8 + x_9 + x_8 x_9 + x_6 x_8 + x_6 x_9 + x_6 x_8 x_9 = 0$
L	x_8	$x_8 + a(g+1)x_2 = 0$
L_ℓ	x_9	$x_9 + (g+1)(x_8 + a x_8 + a) = 0$

<u>Output</u> (SAGE): $[x_1, x_2, x_3, x_4+1, x_5+1, x_6, x_7, x_8, x_9]$

We have found the (unique) fixed point: when $L_e = a = 0$, $G_e = g = 0$

$$(M, P, B, C, R, A, A_\ell, L, L_\ell) = (x_1, x_2, \dots, x_9) = (0,0,0,1,1,0,0,0,0) \quad \underline{\text{OFF}}$$

<u>Exercise</u>: • If $(L_e, G_e) = (a, g) = (0,1)$, then the fixed point is

$$= (0,0,0,0,1,0,0,0,0) \quad \underline{\text{OFF}}$$

• If $(L_e, G_e) = (a, g) = (0,1)$, the fixed point is

$$= (0,0,0,0,1,0,0,0,0) \quad \underline{\text{OFF}}$$

• If $(L_e, G_e) = (a, g) = (1,1)$, the fixed point is

$$= (1,1,1,1,0,1,1,1,1) \quad \underline{\text{ON}}$$

<u>Question</u>: Do these make biological sense? (They do!)

• Predicts operon is <u>on</u> only when external lactose is available and external glucose is not.

In these 3 cases, all variables (except repressor protein) are present

• When glucose is available, operon is <u>off</u>.

<u>Remark</u>: We didn't need to analyze all $2^9 = 512$ states.

Most of them are irrelevant for us.