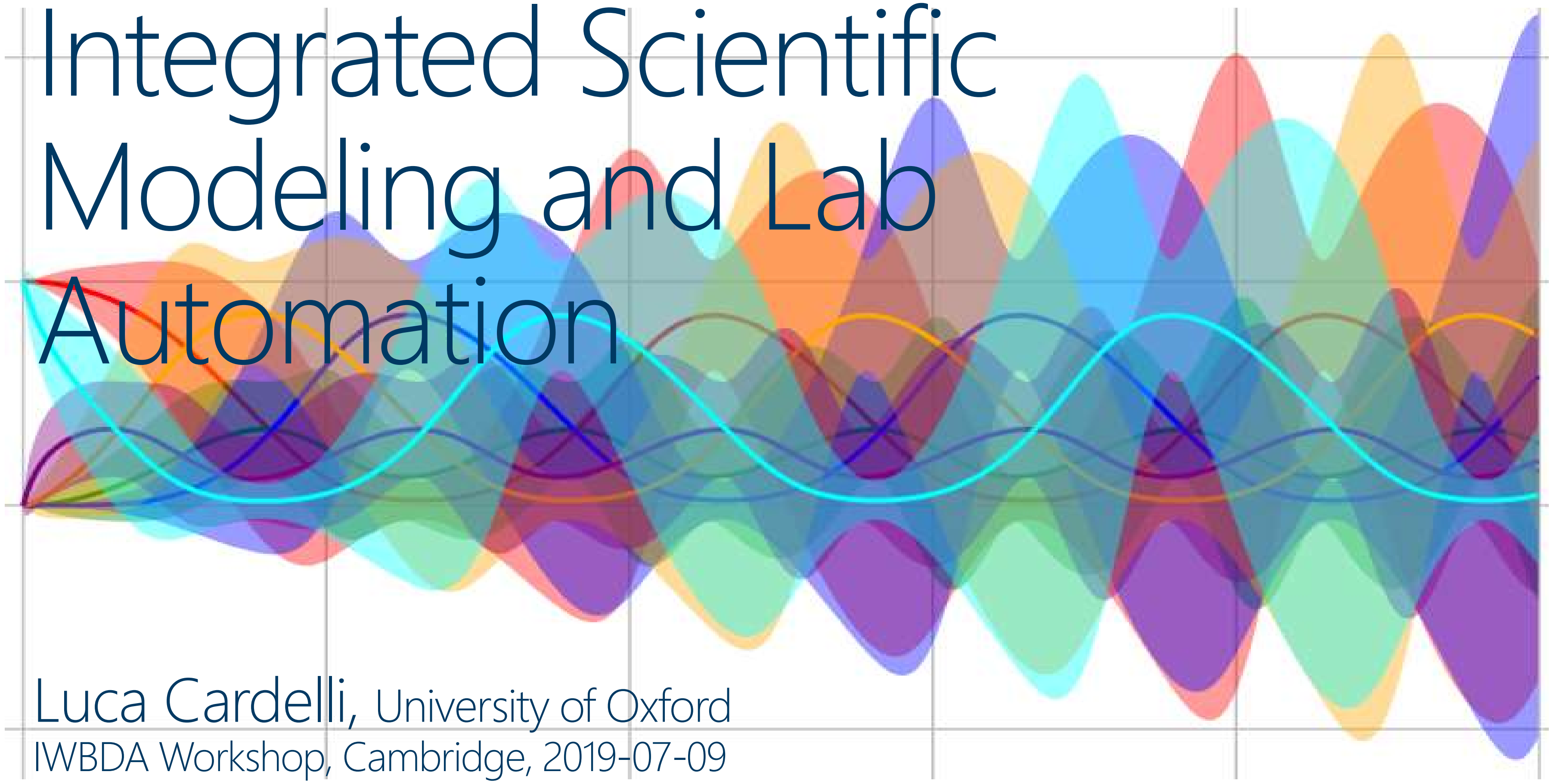


Integrated Scientific Modeling and Lab Automation

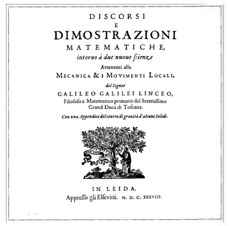
Luca Cardelli, University of Oxford

IWBDA Workshop, Cambridge, 2019-07-09

Discovery through Observation

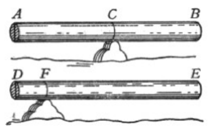
The Scientific Method ~ 1638

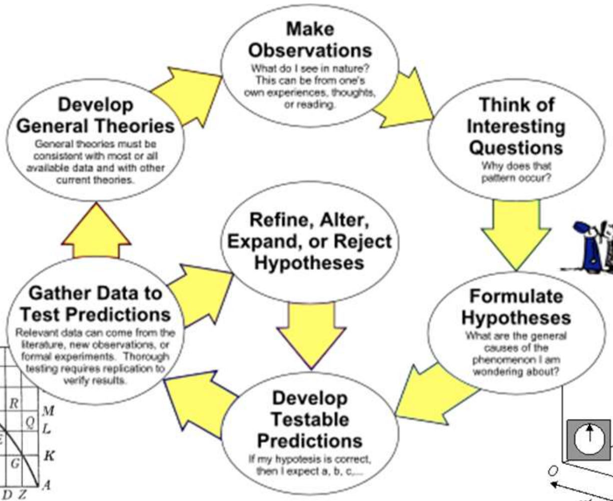
1 Guy



DISCORSI
E
DIMOSTRAZIONI
MATEMATICHE,
intorno à due nuove ſcienze
Attenenti alla
Mecanica & i Movimenti Locali,
del Signor
GALILEO GALILEI LINCEO,
Filoſofo e Matematico primario del Sereniſſimo
Grand Duca di Toſcana.
Con una Appendice del centro di gravità d'alcuni Solidi.

IN LEIDA,
Appreſſo gli Elſevirii. M. D. C. XXXVIII.





Make Observations What do I see in nature? This can be from one's own experiences, thoughts, or reading.

Think of Interesting Questions Why does that pattern occur?

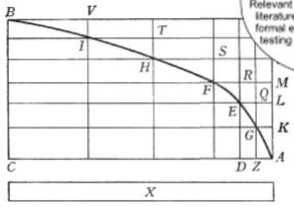
Formulate Hypotheses What are the general causes of the phenomenon I am wondering about?

Develop Testable Predictions If my hypothesis is correct, then I expect a, b, c,...

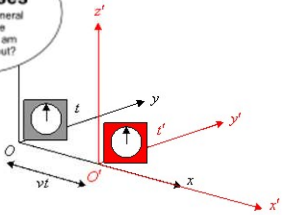
Gather Data to Test Predictions Relevant data can come from the literature, new observations, or formal experiments. Thorough testing requires replication to verify results.

Refine, Alter, Expand, or Reject Hypotheses

Develop General Theories General theories must be consistent with most or all available data and with other current theories.



Garland, Jr., Theodore. "The Scientific Method as an Ongoing Process". U C Riverside.



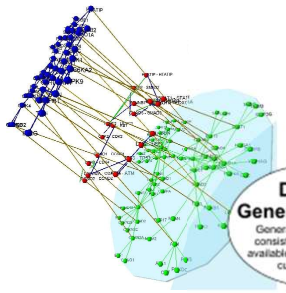
Discovery through Collaboration

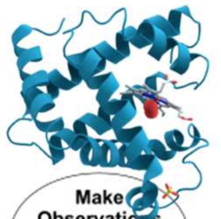
The Scientific Method ~ 2000's

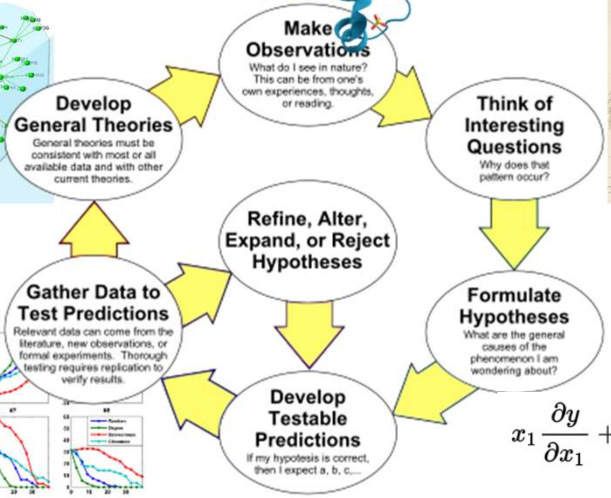
1 Lab

1 protein = 30 people / 30 years

Humans have >250,000 proteins ☹







Make Observations
What do I see in nature? This can be from one's own experiences, thoughts, or reading.

Think of Interesting Questions
Why does that pattern occur?

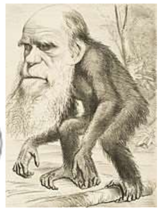
Formulate Hypotheses
What are the general causes of the phenomenon I am wondering about?

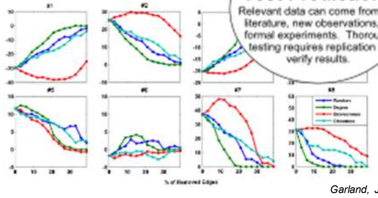
Develop Testable Predictions
If my hypotesis is correct, then I expect a, b, c,...

Gather Data to Test Predictions
Relevant data can come from the literature, new observations, or formal experiments. Thorough testing requires replication to verify results.

Refine, Alter, Expand, or Reject Hypotheses

Develop General Theories
General theories must be consistent with most or all available data and with other current theories.





$$x_1 \frac{\partial y}{\partial x_1} + x_2 \frac{\partial y}{\partial x_2} = y$$

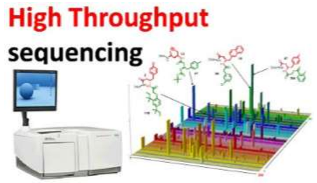
Garland, Jr., Theodore. "The Scientific Method as an Ongoing Process". U C Riverside.

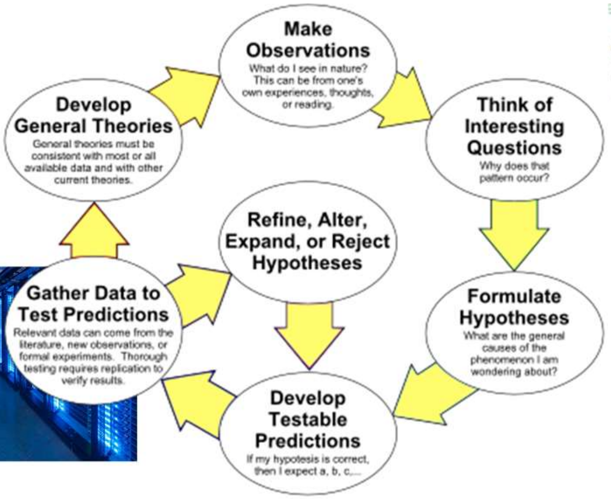
Discovery through Automation

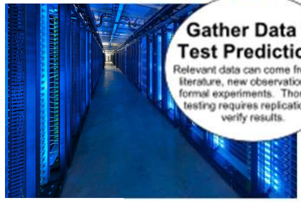
The Scientific Method ~ 2020's

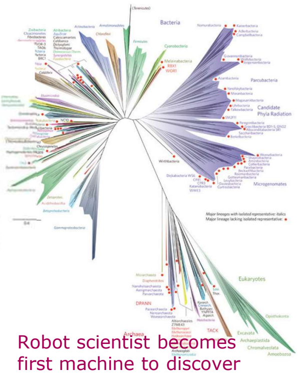
1 Program

```
while (true) {
      predict();
      falsify();
}
```




Make Observations
What do I see in nature? This can be from one's own experiences, thoughts, or reading.
Think of Interesting Questions
Why does that pattern occur?
Formulate Hypotheses
What are the general causes of the phenomenon I am wondering about?
Develop Testable Predictions
If my hypotesis is correct, then I expect a, b, c,...
Gather Data to Test Predictions
Relevant data can come from the literature, new observations, or formal experiments. Thorough testing requires replication to verify results.
Develop General Theories
General theories must be consistent with most or all available data and with other current theories.
Refine, Alter, Expand, or Reject Hypotheses





Robot scientist becomes first machine to discover new scientific knowledge

Ross King

Garland, Jr., Theodore. "The Scientific Method as an Ongoing Process". U C Riverside.

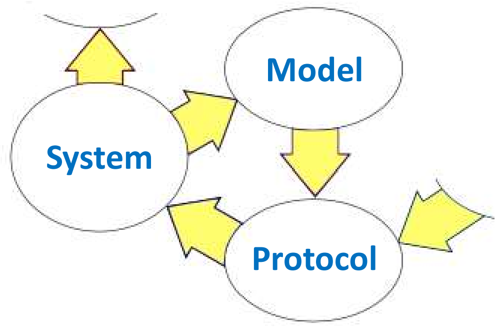
The Inner Loop

- A *model* is refined by testing a (fixed) *protocols* against a *systems*
- A *protocol* is refined by testing a (fixed) *model* against a *systems*

- Today: publication does not accurately reflect execution

  - Model: poorly-maintained matlab script
  - Protocol: poorly-described manual steps in the lab
  - System: poorly-characterized and hardly "resettable"

  - ⇒ Crisis in biology: experiments are done once and are hard to reproduce
    http://www.nature.com/news/reproducibility-1.17552



The Inner Loop

- Tomorrow, automation

Nodes

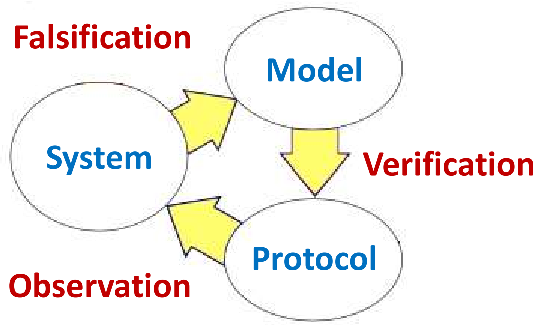
- Model: unambiguous (mathematical) description (CompBio)
- Protocol: standardized (engineered) parts and procedures (SynthBio)
- System: characterized (biological) organism and foundries (SysBio)

Arcs

- Verification: simulation / analysis / model checking / theorem proving
- Observation: lab automation
- Falsification: statistical inference / model reduction

Lifecycle

- Performance evaluation/optimization: of model+protocol+system combined
- Management: version control, equipment monitoring, data storage



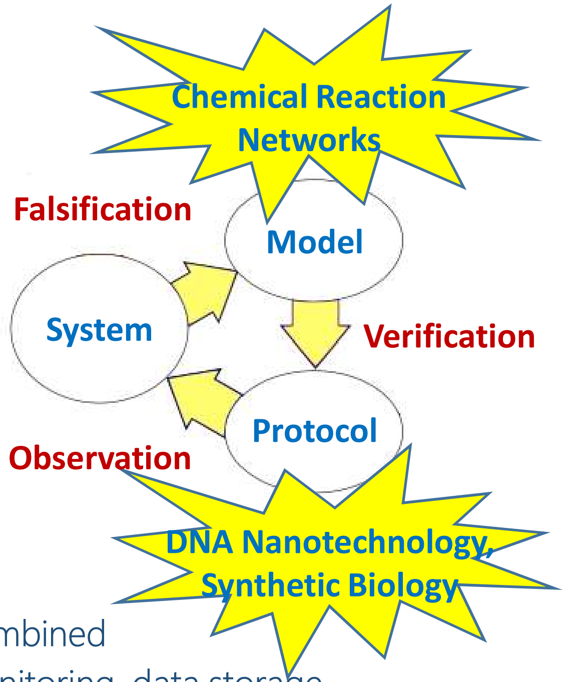
The Inner Loop

- Tomorrow, automation

Nodes
  - Model: unambiguous (mathematical) description (CompBio)
  - Protocol: standardized (engineered) parts and procedures (SynthBio)
  - System: characterized (biological) organism and foundries (SysBio)

Arcs
  - Verification: simulation / analysis / model checking / theorem proving
  - Observation: lab automation
  - Falsification: statistical inference / model reduction

Lifecycle
  - Performance evaluation/optimization: of model+protocol+system combined
  - Management: version control, equipment monitoring, data storage


Chemical Reaction Networks
Falsification
Model
System
Verification
Protocol
Observation
DNA Nanotechnology, Synthetic Biology

Why are chemical reactions interesting?

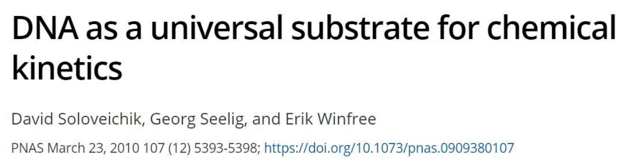
$$X + Y \ \text{->}^{r}\ Z + W$$

- A fundamental model of kinetics in the natural sciences
- A fundamental mathematical structure, rediscovered in many forms
  - Vector Addition Systems, Petri Nets, Bounded Context-Free Languages, Population Protocols, …
- A description of mechanism rather than just behavior
  - A way of describing and comparing biochemical algorithms
  - Enabling addition analysis techniques, e.g. evolution of mechanism through unchanging behavior
- A programming language (coded up in the genome) by which living things manage the processing of matter and information

Also, a formal language we can implement with real (DNA) molecules

- ANY collection of abstract chemical reactions can be implemented with specially designed DNA molecules, with accurate kinetics (up to time scaling).

- A situation where we can "systematically compile" (synthesize) a model, run an (automated) protocol, and observe (sequence) the results in a closed loop.



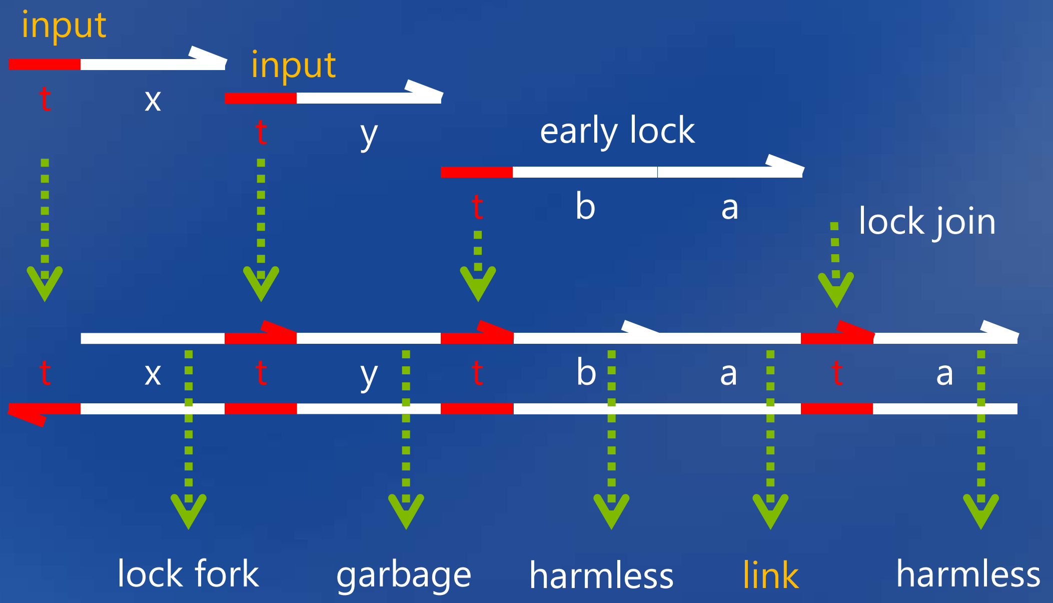
DNA as a universal substrate for chemical kinetics

David Soloveichik, Georg Seelig, and Erik Winfree

PNAS March 23, 2010 107 (12) 5393-5398; https://doi.org/10.1073/pnas.0909380107

Reaction x + y → z + w

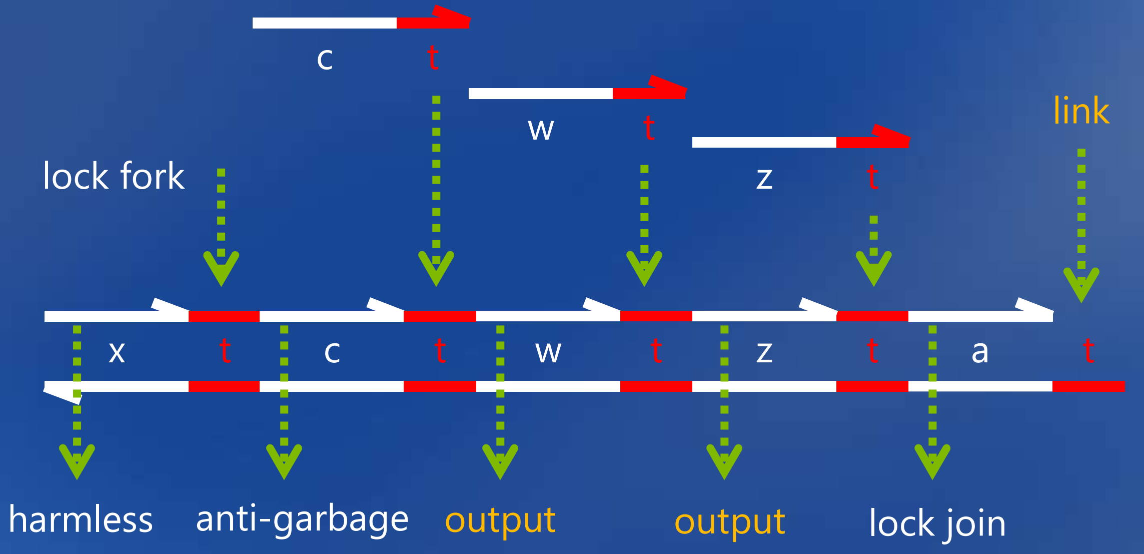
reactants half



"join" structure

Reaction $x + y \rightarrow z + w$

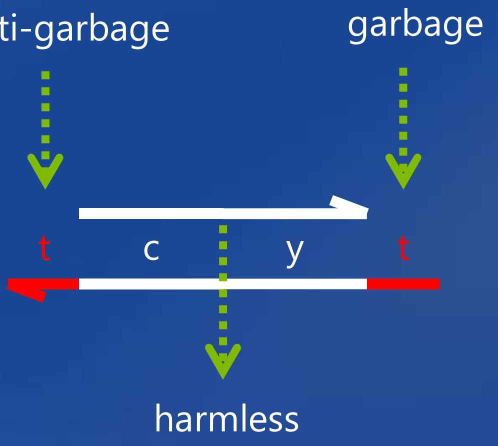
products half



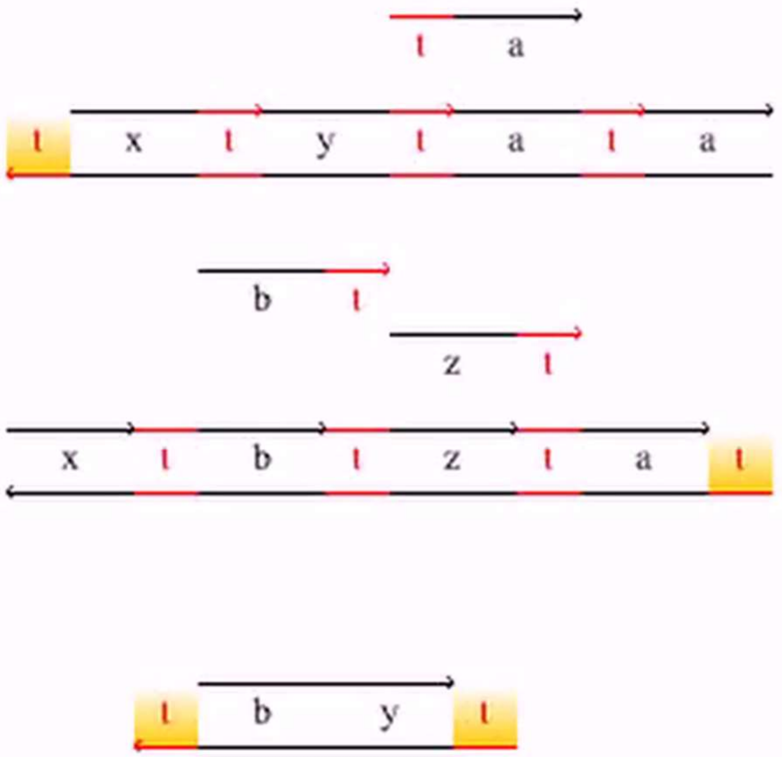
"fork" structure

Reaction $x + y \rightarrow z + w$

garbage collection

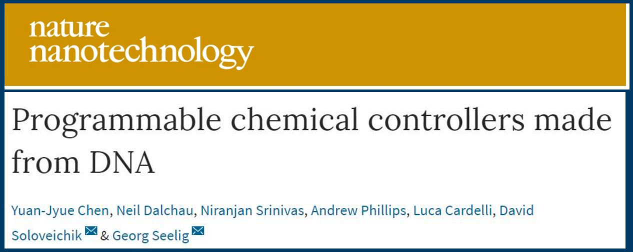


Join $x+y \rightarrow z$



DNA Implementation of the Approximate Majority Algoithm

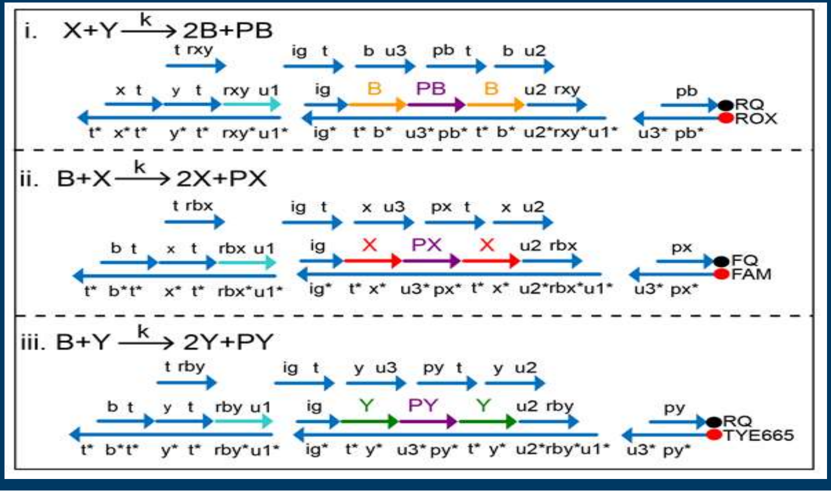
$$X + Y \rightarrow 2B$$
$$B + X \rightarrow 2X$$
$$B + Y \rightarrow 2Y$$

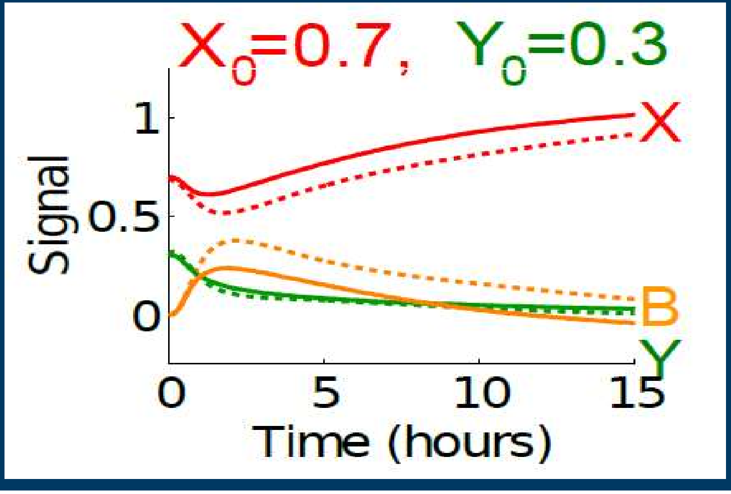


nature nanotechnology

Programmable chemical controllers made from DNA

Yuan-Jyue Chen, Neil Dalchau, Niranjan Srinivas, Andrew Phillips, Luca Cardelli, David Soloveichik & Georg Seelig





Experimental-Protocol Languages for Chemical Reaction Networks

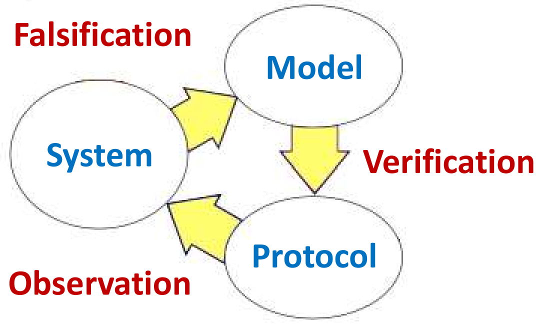
Automating “the whole thing”

- Protocols: sets of steps to direct lab machinery (or people)
  - Published (possibly) in specialized journals. With varying accuracy.
- Models: sets of equations to predict the results of lab experiments
  - Published (possibly) in Auxiliary Online Materials. With lots of typos.

- Protocols know nothing about models
  - What hypothesis is the protocol trying to test? It is not written in the protocol.
- Models know nothing about protocols
  - What lab conditions are being used to test the model? It is not written in the model.
- While presumably talking about the same system
  - Through the experiment.

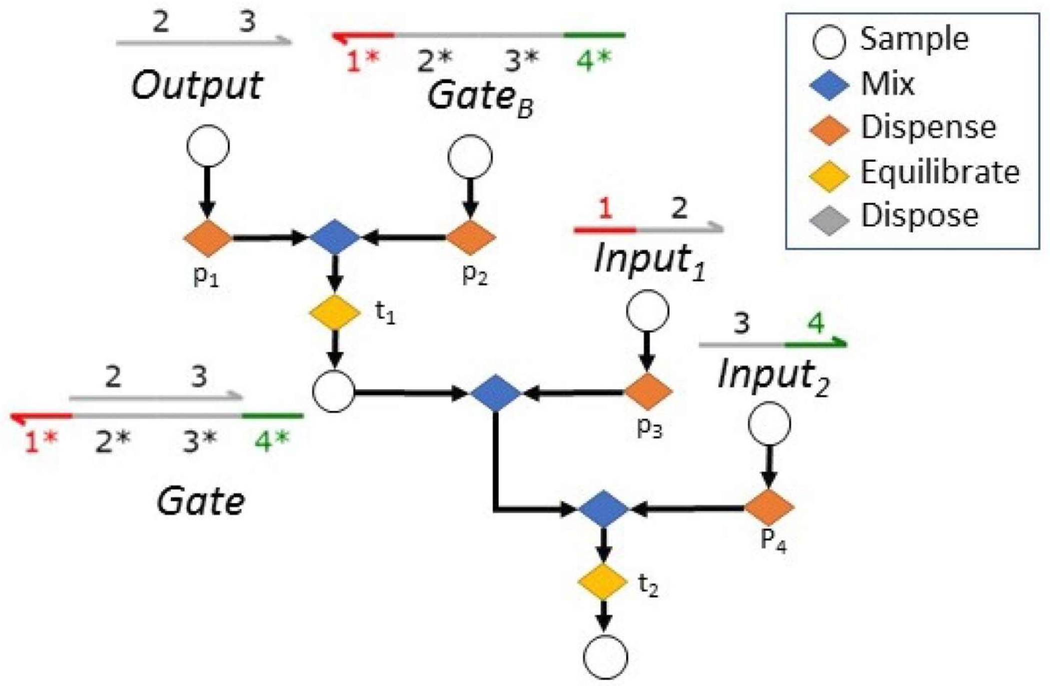
- Reproducibility crisis
  - Experiments are hard to reproduce.
  - Even models are hard to reproduce!

- Similar to a classical problem in C.S.
  - Documentation (model) gets out of step from code (protocol) if their integration is not automated.



A Protocol

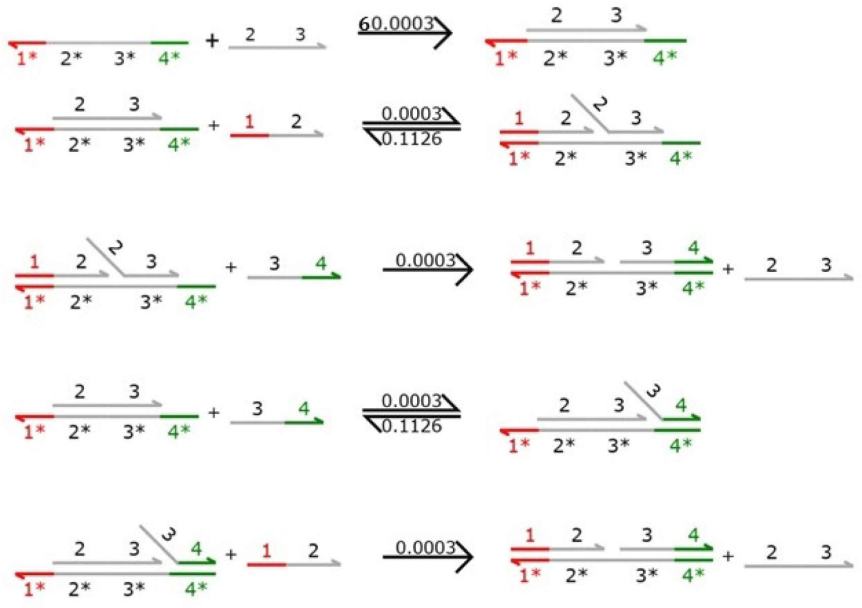
For DNA gate assembly and activation in vitro



Protocol steps (liquid handing)

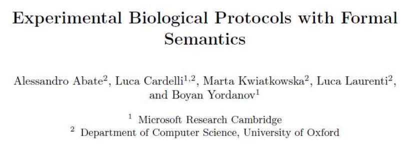
A Model

A Chemical Reaction Network, provided explicitly or (in this case) generated from a higher-level description of the initial strands, according to the DNA strand displacement rules



An Integrated Description

This requires a language



Experimental Biological Protocols with Formal Semantics

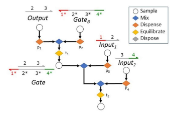
Alessandro Abate[2], Luca Cardelli[1,2], Marta Kwiatkowska[2], Luca Laurenti[2], and Boyan Yordanov[1]

[1] Microsoft Research Cambridge
[2] Department of Computer Science, University of Oxford

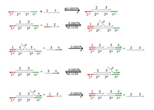
$$\mathcal{C} = (\mathcal{A}, \mathcal{R})$$

$$\begin{array}{rrl} P = & x & (\textit{sample variable}) \\ & (x_0, V, T) & (\textit{initial condition}) \\ & Mix(P_1, P_2) & (\textit{mix samples}) \\ & let\, x = P_1\, in\, P_2 & (\textit{define variable}) \\ & let\, x, y = Dispense(P_1, p)\, in\, P_2 & (\textit{dispense samples}) \\ & Equilibrate(P, t) & (\textit{let time pass}) \\ & Dispose(P) & (\textit{discard } P) \end{array}$$

The CRN can be computed from *{Input$_1$, Input$_2$, Output, Gate}*, and its initial conditions and evolution are determined by the protocol steps.



+



=

$$Input_1 =< 1^*\ 2 > \quad Output =< 2\ 3 >$$
$$Input_2 =< 3\ 4^* > \quad Gate = \{1^*\}[2\ 3]\{4^*\}$$

$$\begin{aligned} P_1 = & let\, In1 = ((Input1, 100.0nM), 0.1mL, 25.0^oC)\, in \\ & let\, In2 = ((Input2, 100.0nM), 0.1mL, 25.0^oC)\, in \\ & let\, GA = ((Output, 100.0nM), 0.1mL, 25.0^oC)\, in \\ & let\, GB = ((Gate_B, 100.0nM), 0.1mL, 25.0^oC)\, in \\ & let\, sGA, \_ = Dispense(GA, p_1)\, in \\ & let\, sGB, \_ = Dispense(GB, p_2)\, in \\ & let\, sIn1, \_ = Dispense(In1, p_3)\, in \\ & let\, sIn2, \_ = Dispense(In1, p_4)\, in \\ & Observe(Equilibrate(Mix(Mix(Equilibrate(\\ & \quad Mix(sGA, sGB), t_1), sIn1), sIn2), t_2), idn). \end{aligned}$$

Language Semantics (deterministic)

The deterministic case is a warm-up exercise, but simple to explain

Each program denotes a final state <concentrations, volume, temperature>

$[\![P]\!]^\rho$ is the final state produced by a protocol P for a fixed CRN $\mathcal{C} = (\mathcal{A}, \mathcal{R})$:

$$
\begin{aligned}
&[\![x]\!]^\rho = \rho(x)\\
&[\![x_0, V, T]\!]^\rho = (x_0, V, T)\\
&[\![Mix(P_1, P_2)]\!]^\rho =\\
&\quad let\,(x_0^1, V_1, T_1) = [\![P_1]\!]^\rho\\
&\quad let\,(x_0^2, V_2, T_2) = [\![P_2]\!]^\rho\\
&\quad (\frac{x_0^1 V_1 + x_0^2 V_2}{V_1 + V_2}, V_1 + V_2, \frac{T_1 V_1 + T_2 V_2}{V_1 + V_2})\\
&[\![let\, x = P_1\, in\, P_2]\!]^\rho =\\
&\quad let\,(x_0, V, T) = [\![P_1]\!]^\rho\\
&\quad let\,\rho_1 = \rho\{x \leftarrow (x_0, V, T)\}\\
&\quad [\![P_2]\!]^{\rho_1}
\end{aligned}
$$

$$
\begin{aligned}
&[\![let\, x, y = Dispense(P_1, p)\, in\, P_2]\!]^\rho =\\
&\quad let\,(x_0, V, T) = [\![P_1]\!]^\rho\\
&\quad let\,\rho_1 = \rho\{x \leftarrow (x_0, V \cdot p, T), y \leftarrow (x_0, V \cdot (1-p), T)\}\\
&\quad [\![P_2]\!]^{\rho_1}\\
&[\![Equilibrate(P, t)]\!]^\rho =\\
&\quad let\,(x_0, V, T) = [\![P]\!]^\rho\\
&\quad [\![(\mathcal{A}, \mathcal{R}, x_0), V, T)]\!](H)(t)\\
&[\![Dispose(P)]\!]^\rho = (0^{|\mathcal{A}|}, 0, 0),
\end{aligned}
$$

State produced by CRN $\mathcal{C} = (\mathcal{A}, \mathcal{R})$ at time t:

$$
\begin{aligned}
&[\![((\mathcal{A},\mathcal{R}, x_0), V, T)]\!](H)(t) =\\
&\quad let\, G : [0...H) \to \mathbb{R}^{|\mathcal{A}|}\ be\ the\ solution\ of\ G(t') = x_0 + \int_0^{t'} F(V,T)(G(s))ds\\
&\quad (G(t), V, T)
\end{aligned}
$$

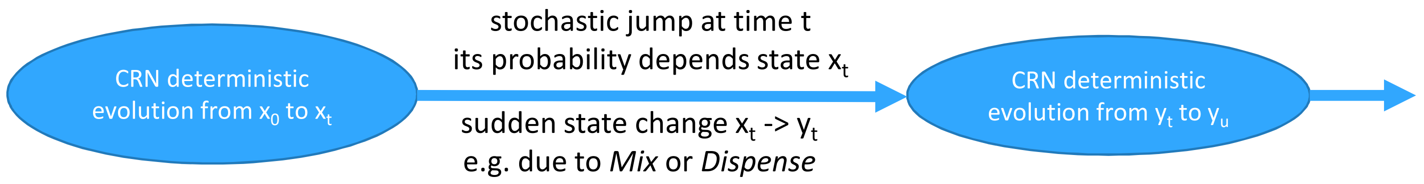
Language Semantics (stochastic)

Dispense has a volume uncertainty.

Equilibrate has a time uncertainty.

Reactions have rate uncertainty.

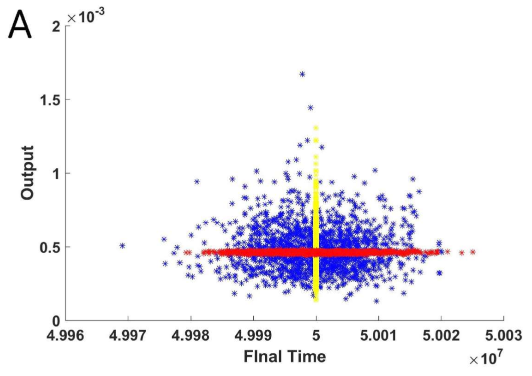
Each program now represents a Hybrid System with stochastic jumps between deterministic evolutions:


CRN deterministic evolution from x_0 to x_t
stochastic jump at time t its probability depends state x_t
sudden state change x_t -> y_t e.g. due to *Mix* or *Dispense*
CRN deterministic evolution from y_t to y_u

Which in turn denotes a *Piecewise Deterministic Markov Process (PDMP)*

Stochastic Analysis

- We can ask: what is the probability of a certain outcome given uncertainties in *both the protocol and the model*?
- Conversely: which parameters of *both the protocol and the model* best fit the observed result?

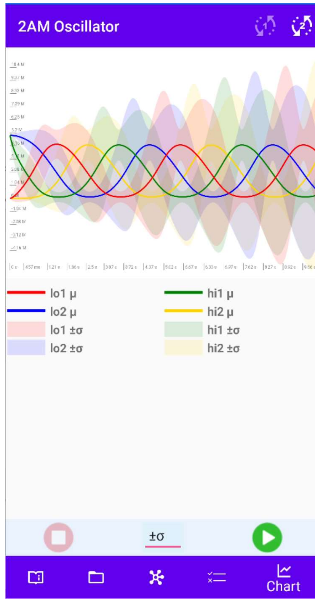


1500 executions including protocol uncertainty due timing and pipetting errors (red).
1500 executions including only model uncertainty about rates of the CRN (yellow).
1500 executions including both sources of uncertainty (blue).

We may estimate by Statistic Model Checking, e.g. the probability that Output will fall in a certain range, given distributions over uncertain model and protocol parameters.

Kaemika

- A prototype language for chemical models & protocols



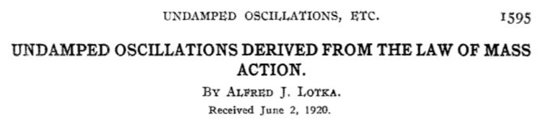
- Android app:
  Search "Kaemika" in the Play Store
  https://play.google.com/store/apps/details?id=com.kaemika.Kaemika

Describing a Model

- *Species* and *reactions*
  - Are characterized by a initial values and rates
- *Kinetics*
  - Assume a model of matter (deterministic of stochastic) e.g. for simulations
- *Programming abstractions*
  - Help assemble large models as compositions of modules

Species and Reactions

```
//======================================
// Lotka 1920, Volterra 1926
// (simplified with all rates = 1)
//======================================

number x1_0 =? uniform(0,1) // random x1_0
number x2_0 =? uniform(0,1) // random x2_0

species x1 @ x1_0 M      // prey
species x2 @ x2_0 M      // predator

x1 -> x1 + x1          {1} // prey reproduces
x1 + x2 -> x2 + x2     {1} // predator eats prey
x2 -> #                {1} // predator dies

equilibrate for 40
```

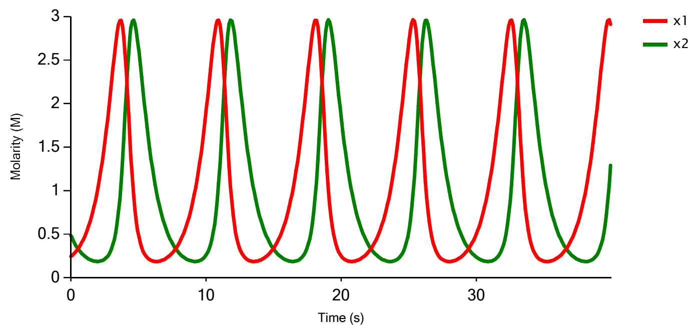


UNDAMPED OSCILLATIONS, ETC. 1595

UNDAMPED OSCILLATIONS DERIVED FROM THE LAW OF MASS ACTION.

By Alfred J. Lotka.

Received June 2, 1920.



Common action:

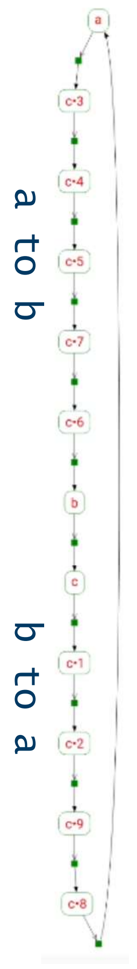
- Run simulations
- Vary parameters
- Inspect reaction graphs
- Extract equations
- Intrinsic noise (via LNA)

Writing Models Compositionally

- Functional-monadic approach
  - *Functions* take *data* as parameters and produce *data* as results
  - *Networks* take *data* as parameters and produce *effects* as results
  - *Data* is *numbers*, *species*, *functions*, *networks*, *flows*, etc.
  - *Effects* are *species creation*, *reaction definitions*, *and sample handling*
  - A program execution produces both a final *result* and a sequence of *effects*
- (Temporal) Flows
  - Flows are functions of time (mostly real-valued)
  - Can be assembled programmatically (as a data structure)
  - Can be used as *rates* (leading to programmable kinetics)
  - Can be *observed* at specific times (leading to protocol observations)
  - Can be *plotted* over time (leading to chart series and legends)

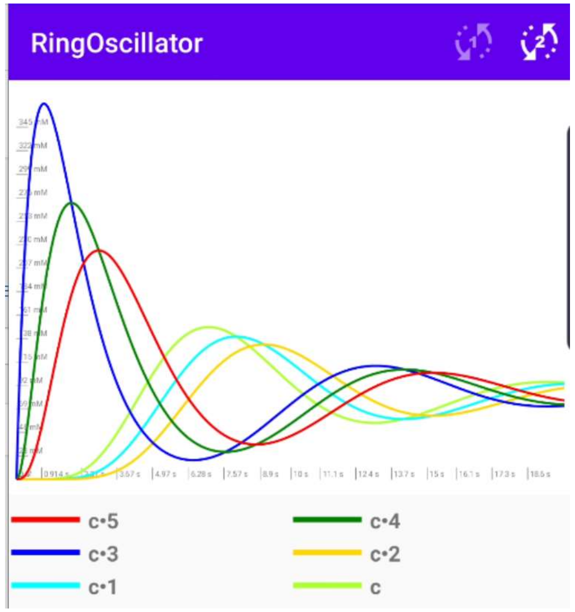
Ex: Ring Oscillator

```
First build a chain of reactions from a to b
with n intermediate species c_i
a -> c_0 -> c_1 ->  ... -> c_n-1 -> b

network Erlang(species a b, number n) {
   if n <= 0 then
      a -> b                          // just one reaction from a to b
   else
      species c @ 0 M                 // new intermediate species c, initially 0
      if n <= 3 then report c end  // plot (report) at most 3 of those
      Erlang(a, c, n-1)               // make a chain from a to c with n-1 steps
      c -> b                          // plus one reaction from c to b
   end
}
```



Ex: Ring Oscillator

```
Then connect two such chains in a loop
to produce a dampened ring oscillator

network RingOscillator(species a b, number n) {
    Erlang(a,b,n/2)
    Erlang(b,a,n/2)
}

Initialize some species and activate the oscillator

species a @ 1 M
species b @ 0 M
RingOscillator(a, b, 10)

Simulate the reactions and produce a plot
Multiple 'c' species are distinguished by a suffix

equilibrate for 20
```



Describing a Protocol

- *Samples* (e.g. test tubes)
  - Are characterized by a volume and a temperature
  - Contain a specified set of species
  - Evolve according to reactions that operates on those species

- *Operations* (e.g. liquid handling)
  - Accept and produce samples
  - Accepted samples are *used up* (they can only be operated-on once)

Samples (and their volume)

- Samples contain concentrations of species, acted over by reactions.
- Each sample has a fixed volume and a fixed temperature through its evolution.
- Sample concentrations are in units of M = mol/L.
- The default implicit sample is called the 'vessel' {1 mL, 20 C}

```
Create a new empty sample 's' with
given volume and temperature:

sample s {1mL, 20C}
```

```
Declare two new species, but do not
initialize them: they can be used in
several samples:

species {a, b}
```

```
Initialize the amount of 'a' in 's'
at '1' (M), similarly for 'b'.

amount a @ 1 M in s
amount b @ 2 M in s
```

```
The amount can also be given in
grams (if molar mass is specified) and
the resulting concentration is then
relative to sample volume.

sample t {1mL, 20C}

species {NaCl#58.44}

amount NaCl @ 8g in t
```

Samples (and their temperature)

```
Declare a new temperature-dependant reaction
(it can operate in any sample
where all its species initialized).

a + b -> {2, 5} c

'2' is collision frequency,
'5' (J*mol^-1) is activation energy
(default is '{1, 0}')

In each sample, the reaction rate is then
dependent on the sample temperature 'T' via
the activation energy and the gas constant 'R'
by Arrhenius' formula: 2*e^(-5/(R*T))
```

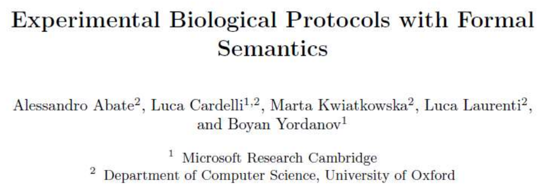
Samples (and their evolution)

```
The sample 's' evolves
according to the relevant reactions
resulting in a new sample 's1' after time '3'.

equilibrate s1 := s for 3

•  Sample 's' can no longer be used after this: it has been consumed.
•  Sample 's1' has the same volume and temperature as 's'.
•  Sample 's1' contains the same species as 's'
   in usually altered amounts.

'equilibrate s := s for 3'
reuses the old name for the new sample.

'equilibrate s for 3' is an abbreviation
for 'equilibrate s := s for 3'.
```

Samples (and their operations)

```
Mix two samples into one

        mix A := B with C

Split a sample into two

        split B,C := A by 0.5

Transfer a sample to a new volume, temperature

        transfer A{1L, 20C} := B

Let a sample evolve

        equilibrate A := B for 3

Throw away a sample

        dispose C
```

These are based on our paper, but now these are *effects*, not algebraic operation. So they are used like imperative statements (":=") rather than expression.



Experimental Biological Protocols with Formal Semantics

Alessandro Abate[2], Luca Cardelli[1,2], Marta Kwiatkowska[2], Luca Laurenti[2], and Boyan Yordanov[1]

[1] Microsoft Research Cambridge
[2] Department of Computer Science, University of Oxford

Flows

- Flows are a powerful facility for representing time series, they can appear in rate expressions, in report (plotting) expressions, and in protocol observation.
- A flow is a closed expression (essentially a data structure) representing a *value* v(t,s) at a time $t \geq 0$ in sample s (or a *distribution of values* if LNA is active).
- I.e. a flow denotes a function λ(t,s) v(t,s).

`time`	λ(t,s) t	
`3.5`	λ(t,s) 3.5	
`kelvin`	λ(t,s) temperature(s)	
`a` (a species)	λ(t,s) a(t,s)	concentration of **`a`** in the sample
$\texttt{op}(f_1,...,f_2)$	λ(t,s) $op(f_1(t,s),...,f_2(t,s))$	e.g.: `sin(time+1)`, `2*a - 3*b`
$\texttt{cond}(f_1,f_2,f_3)$	λ(t,s) if $f_1(t,s)$ then $f_2(t,s)$ else $f_3(t,s)$	conditional flows, e.g. `cond(a<b, a, b)`
`poisson(f)`	λ(t,s) poisson(f(t,s))	mean and variance equal to f(t,s)
$\texttt{cov}(f_1,f_2)$	λ(t,s) $cov(f_1(t,s),f_2(t,s))$	covariance of any two linear combinations of species
`∂f`	∂_t(λ(t,s) f(t,s))	first time deriviative (based on the mass action equations)

Observations

`observe(f, s)`	observe a flow f in sample s (at the "current" time)
`observe(kelvin,s)`	temperature of s
`observe(volume,s)`	volume of s (L)
`observe(a,s)`	concentration of a in s (mol/L)
`observe((a-b)^2,s)`	combined observations
`observe(time,s)`	e.g., observe the endtime of a simulation
`observe(var(a),s)`	observe noise (requires LNA active)
`observe(∂a,s)`	time derivative of a's concentration

- Conditional protocol execution

```
if (observe(a, s) > 3.5) then ... else ...
```

  If the concentration of a in sample s > 3.5 ... (typically tested at a time *between* equilibrates)

- Protocol optimization

```
argmin(objectiveFunction, initialGuess, tolerance)
```

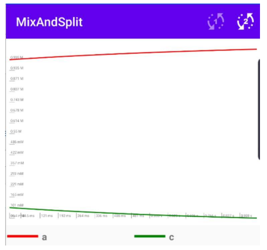
  where `objectiveFunction = fun(parameters) ... observe((objective - outcome)^2, s) ...`

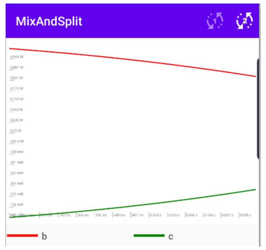
  Compute an error that depends on a choice of parameters,
  for the gradient descent minimization of an objective function

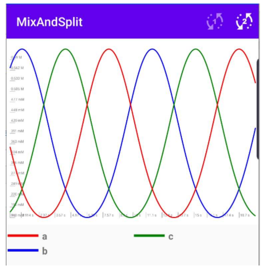
Ex: Sample Manipulation

```
species {c}

sample A
species a @ 1M in A
amount c @ 0.1M in A
a + c -> a + a
equilibrate A1 := A for 1

sample B
species b @ 1M in B
amount c @ 0.1M in B
b + c -> c + c
equilibrate B1 := B for 1

split C,D := A1 by 0.5
dispose C

mix E := D with B1
a + b -> b + b

equilibrate F := E for 20
dispose F
```

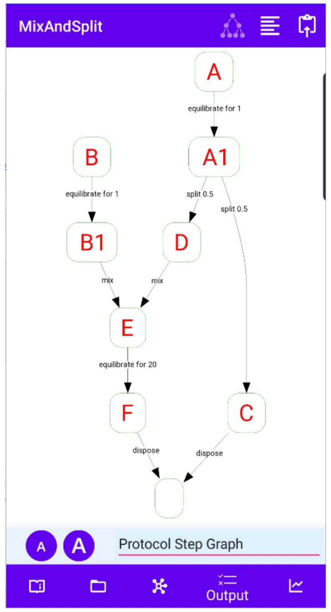
Multiple equilibration steps



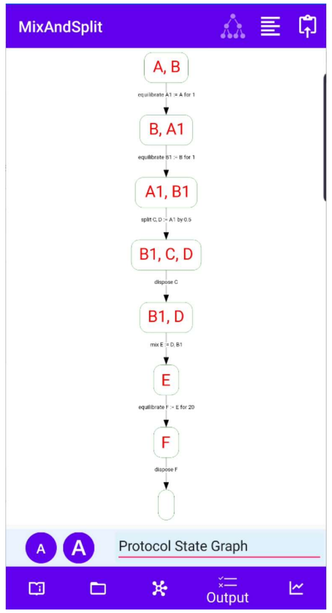




"Protocol step graph"



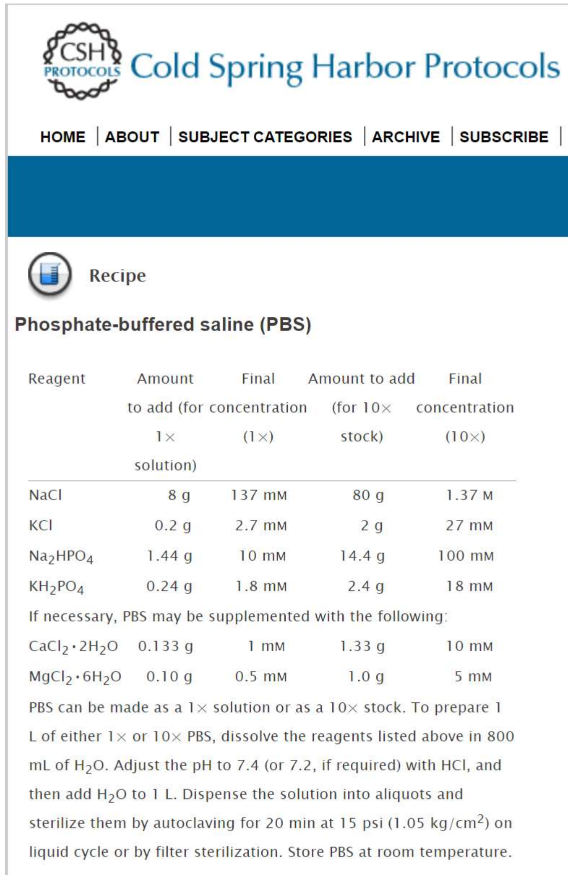
"Protocol state graph"



PDMP ("System Equations")

Ex: Phosphate-buffered saline (PBS)

```
species {NaCl#58.44, KCl#74.5513, NA2HPO4#141.96, KH2PO4#136.086}
report NaCl, KCl, NA2HPO4, KH2PO4

function MakePBS() {
   define
      sample PBS {800mL, 20C}
      amount NaCl @ 8g in PBS
      amount KCl @ 0.2g in PBS
      amount NA2HPO4 @ 1.44g in PBS
      amount KH2PO4 @ 0.24g in PBS

      sample topup {200mL, 20C}
      mix PBS with topup
   yield Autoclave(PBS, 20*60)
}

function Autoclave(sample PBS, number t) {
   define
      // increase temperature, preserve volume:
      transfer hot { observe(volume,PBS)L, 121C } := PBS
      // bake
      equilibrate hot for t
      // decrease temperature, preserve volume:
      transfer PBS { observe(volume,hot)L, 20C } := hot
   yield PBS
}

sample PBS = MakePBS()
```



CSH PROTOCOLS Cold Spring Harbor Protocols
HOME | ABOUT | SUBJECT CATEGORIES | ARCHIVE | SUBSCRIBE |
Recipe

Phosphate-buffered saline (PBS)

Reagent	Amount to add (for 1× solution)	Final concentration (1×)	Amount to add (for 10× stock)	Final concentration (10×)
NaCl	8 g	137 mM	80 g	1.37 M
KCl	0.2 g	2.7 mM	2 g	27 mM
Na_2HPO_4	1.44 g	10 mM	14.4 g	100 mM
KH_2PO_4	0.24 g	1.8 mM	2.4 g	18 mM
If necessary, PBS may be supplemented with the following:				
$CaCl_2 \cdot 2H_2O$	0.133 g	1 mM	1.33 g	10 mM
$MgCl_2 \cdot 6H_2O$	0.10 g	0.5 mM	1.0 g	5 mM

PBS can be made as a 1× solution or as a 10× stock. To prepare 1 L of either 1× or 10× PBS, dissolve the reagents listed above in 800 mL of H_2O. Adjust the pH to 7.4 (or 7.2, if required) with HCl, and then add H_2O to 1 L. Dispense the solution into aliquots and sterilize them by autoclaving for 20 min at 15 psi (1.05 kg/cm^2) on liquid cycle or by filter sterilization. Store PBS at room temperature.

http://cshprotocols.cshlp.org/content/2006/1/pdb.rec8247

Ex: Serial Dilution

```
network SerialDilution(number count, sample s, network f) {
  if count > 0 then
    sample solvent {9*observe(volume,s) L, observe(kelvin,s) K}
    mix s with solvent
    split s, dilution := s by 0.1
    f(dilution)
    SerialDilution(count-1, s, f)
  end
}

initial sample to be diluted:

sample init {1mL, 25C}
species A @ 1M in init
species B @ 1M in init
A + B ->{20} A
A -> #

apply this network to each dilution;
note that this invokes a simulation
each time in each solution

network test(sample s) {
   equilibrate s for 10
   dispose s
}

dilute 4 times

SerialDilution(4, init, test)
```

Prepare a series of increasingly diluted solutions and apply a network f to each (f can add species and reactions to the solutions)

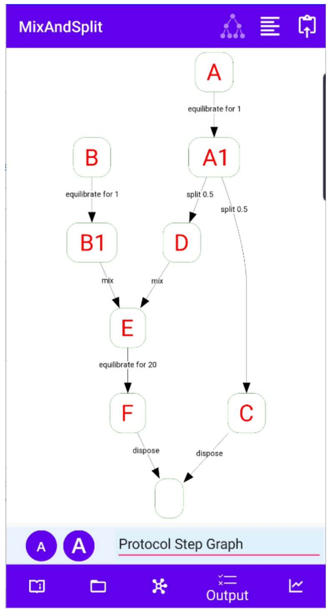
```
RESULT:
sample init {1mL, 298.2K} {A = 1M, B = 1M}
sample s2 {1mL, 298.2K} {A = 100mM, B = 100mM}
sample s4 {1mL, 298.2K} {A = 10mM, B = 10mM}
sample s7 {1mL, 298.2K} {A = 1mM, B = 1mM}
sample s10 {1mL, 298.2K} {A = 100uM, B = 100uM}
```

Extracting both Model and Protocol

From the script

```
species {c}

sample A
species a @ 1M in A
amount c @ 0.1M in A
a + c -> a + a
equilibrate A1 := A for 1

sample B
species b @ 1M in B
amount c @ 0.1M in B
b + c -> c + c
equilibrate B1 := B for 1

split C,D := A1 by 0.5
dispose C

mix E := D with B1
a + b -> b + b

equilibrate F := E for 20
dispose F
```

The protocol



The (final) model (sample E)

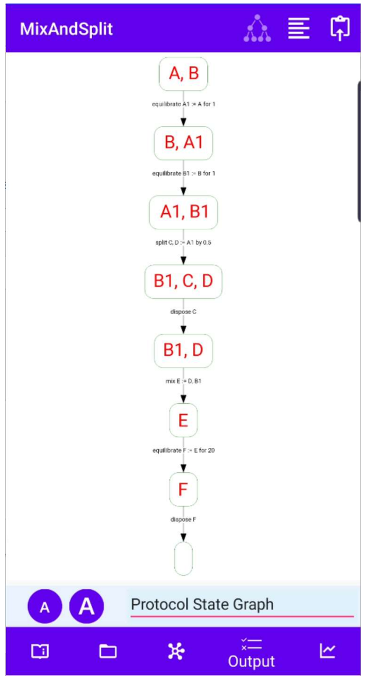
```
STATE_5
sample E {1.5mL, 293.2K} {
   a = 354.5mM
   c = 178mM
   b = 0.5674M
   consumed
   a + c -> a + a
   b + c -> c + c
   a + b -> b + b
}

KINETICS for STATE_5 (sample E) for 20 time units:
∂a = a * c - a * b
∂c = c * b - a * c
∂b = a * b - c * b
```

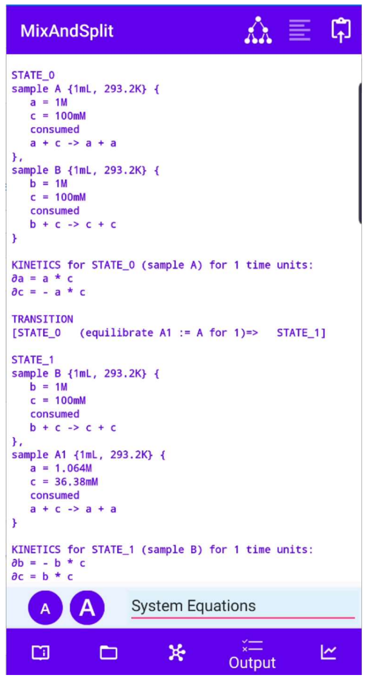
Extracting both Model and Protocol

From the script

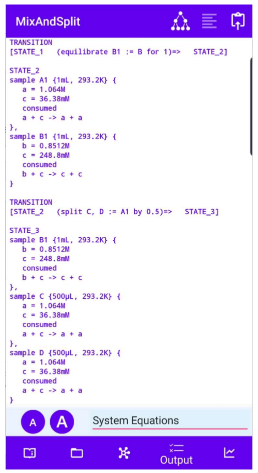
```
species {c}

sample A
species a @ 1M in A
amount c @ 0.1M in A
a + c -> a + a
equilibrate A1 := A for 1

sample B
species b @ 1M in B
amount c @ 0.1M in B
b + c -> c + c
equilibrate B1 := B for 1

split C,D := A1 by 0.5
dispose C

mix E := D with B1
a + b -> b + b

equilibrate F := E for 20
dispose F
```

The full story (Hybrid system)

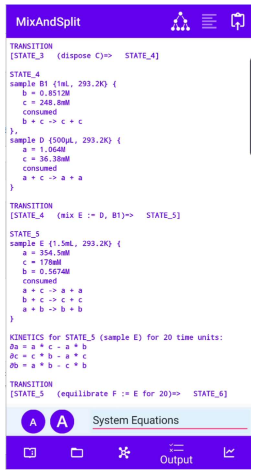




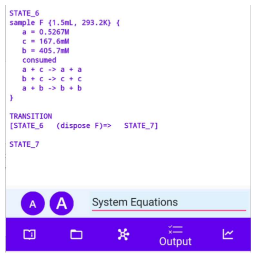
```
STATE_0
sample A {1mL, 293.2K} {
   a = 1M
   c = 100mM
   consumed
   a + c -> a + a
},
sample B {1mL, 293.2K} {
   b = 1M
   c = 100mM
   consumed
   b + c -> c + c
}

KINETICS for STATE_0 (sample A) for 1 time units:
∂a = a * c
∂c = - a * c

TRANSITION
[STATE_0   (equilibrate A1 := A for 1)=>   STATE_1]

STATE_1
sample B {1mL, 293.2K} {
   b = 1M
   c = 100mM
   consumed
   b + c -> c + c
},
sample A1 {1mL, 293.2K} {
   a = 1.064M
   c = 36.38mM
   consumed
   a + c -> a + a
}

KINETICS for STATE_1 (sample B) for 1 time units:
∂b = - b * c
∂c = b * c
```



```
TRANSITION
[STATE_1   (equilibrate B1 := B for 1)=>   STATE_2]

STATE_2
sample A1 {1mL, 293.2K} {
   a = 1.064M
   c = 36.38mM
   consumed
   a + c -> a + a
},
sample B1 {1mL, 293.2K} {
   b = 0.8512M
   c = 248.8mM
   consumed
   b + c -> c + c
}

TRANSITION
[STATE_2   (split C, D := A1 by 0.5)=>   STATE_3]

STATE_3
sample B1 {1mL, 293.2K} {
   b = 0.8512M
   c = 248.8mM
   consumed
   b + c -> c + c
},
sample C {500μL, 293.2K} {
   a = 1.064M
   c = 36.38mM
   consumed
   a + c -> a + a
},
sample D {500μL, 293.2K} {
   a = 1.064M
   c = 36.38mM
   consumed
   a + c -> a + a
}
```



```
TRANSITION
[STATE_3   (dispose C)=>   STATE_4]

STATE_4
sample B1 {1mL, 293.2K} {
   b = 0.8512M
   c = 248.8mM
   consumed
   b + c -> c + c
},
sample D {500μL, 293.2K} {
   a = 1.064M
   c = 36.38mM
   consumed
   a + c -> a + a
}

TRANSITION
[STATE_4   (mix E := D, B1)=>   STATE_5]

STATE_5
sample E {1.5mL, 293.2K} {
   a = 354.5mM
   c = 178mM
   b = 0.5674M
   consumed
   a + c -> a + a
   b + c -> c + c
   a + b -> b + b
}

KINETICS for STATE_5 (sample E) for 20 time units:
∂a = a * c - a * b
∂c = c * b - a * c
∂b = a * b - c * b

TRANSITION
[STATE_5   (equilibrate F := E for 20)=>   STATE_6]
```

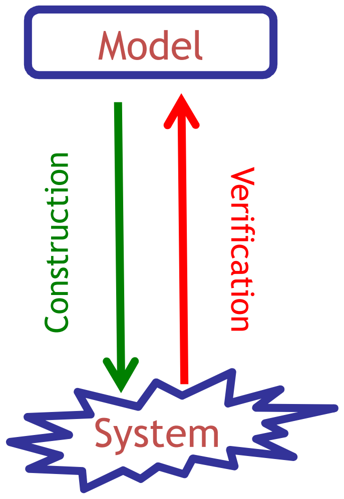


```
STATE_6
sample F {1.5mL, 293.2K} {
   a = 0.5267M
   c = 167.6mM
   b = 405.7mM
   consumed
   a + c -> a + a
   b + c -> c + c
   a + b -> b + b
}

TRANSITION
[STATE_6   (dispose F)=>   STATE_7]

STATE_7
```

Conclusions

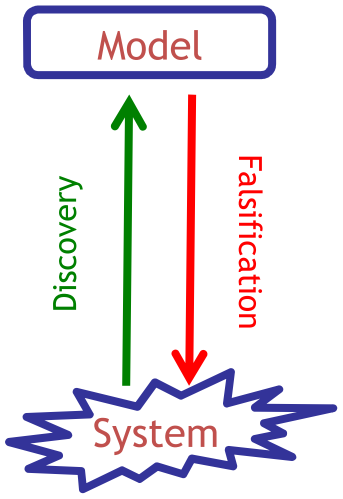
Scientific Method vs. Engineering Method

Engineering Method



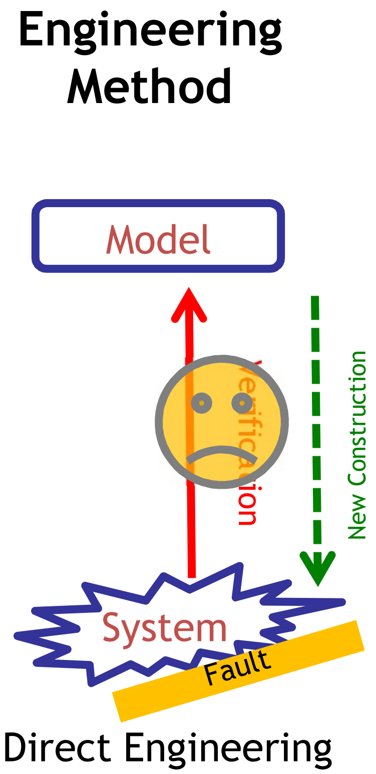
Direct Engineering
(Synthetic Biology)

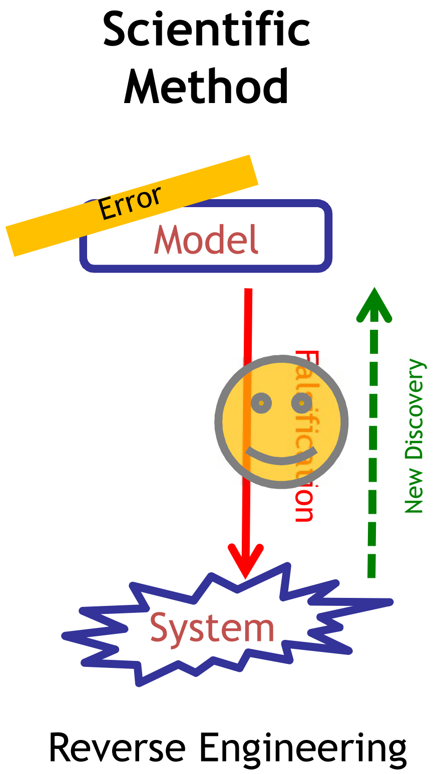
Scientific Method



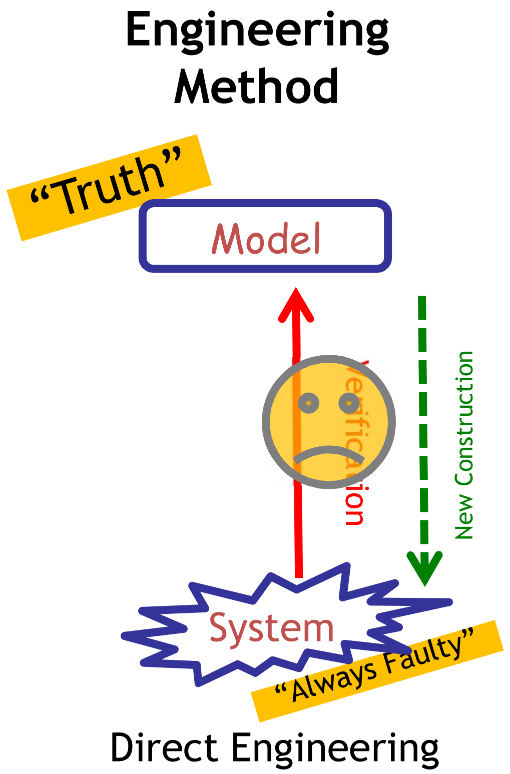
Reverse Engineering
(Systems Biology)

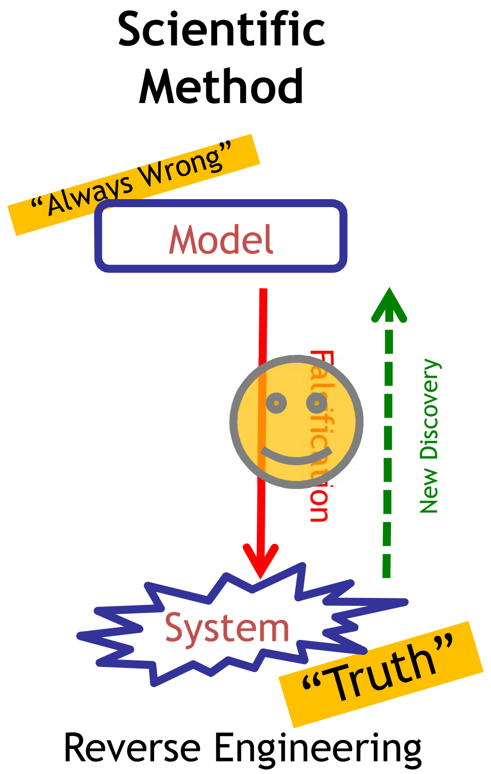
Scientific Method vs. Engineering Method





Scientific Method vs. Engineering Method



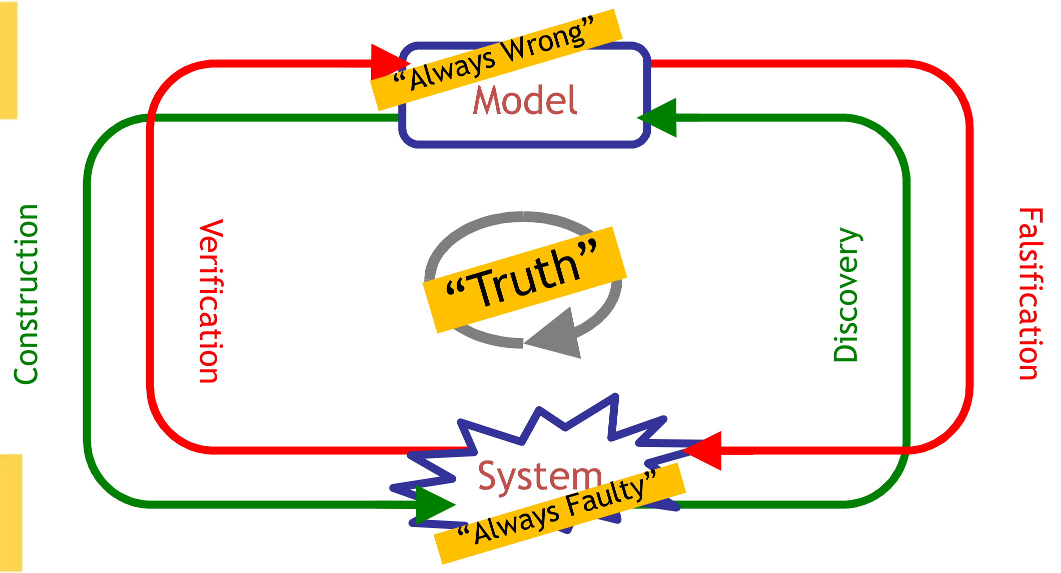


Scientific Method vs. Engineering Method

When the models and the systems are *both* too complex to *either* be the full Truth

Closed Loop Method

The model is always somewhat wrong in its predictions



The Truth is not something you ever "have" but something you "maintain"

The system is always somewhat faulty in its execution

We need a closed-loop formalized description of the whole method