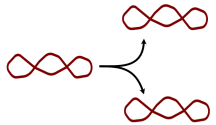
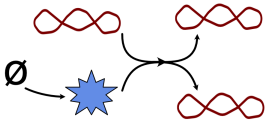
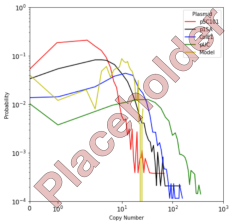
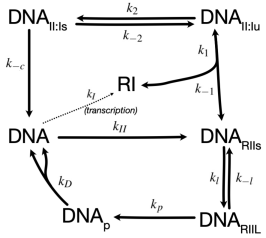
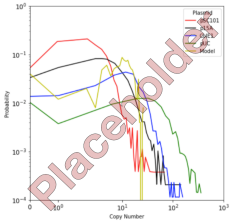
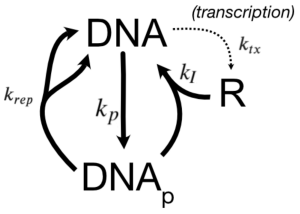
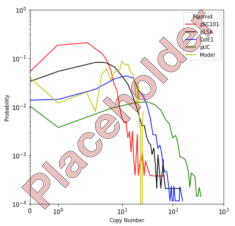
How to model DNA replication in stochastic models of synthetic gene circuits (and why)

Samuel Clamons[1] **and Richard Murray**[1]

[1]**Caltech, Pasadena, CA, United States**

CONTENTS

Model	CRN Reactions	Description	Diagrams	Distribution Fits
Trivial Replication (don't do this!)	$DNA \to DNA + DNA$	• DNA copies itself without restriction		Not applicable (no non-trivial steady state).
Dummy-Triggered Replication	$\emptyset \to R$ $R + DNA \xrightarrow{\text{fast}} DNA + DNA$	• Replication is triggered at zero-order rate. • Replication trigger is consumed by replication. • "Hack" to produce DNA at a rate independent of DNA concentration, while remaining a CRN model.		
Brendel & Perelson's ColE1 Replication Model	$DNA \xrightarrow{k_{II}} DNA_{RIIs}$ $DNA_{RIIs} \xrightarrow{k_I} DNA_{RIIL}$ $DNA_{RIIL} \xrightarrow{k_{-I}} DNA$ $DNA_{RIIL} \xrightarrow{k_p} DNA_p$ $DNA_p \xrightarrow{k_D} DNA + DNA$ $DNA_{RIIs} + RI \underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} DNA_{II:Iu}$ $DNA_{II:Iu} \underset{k_{-2}}{\overset{k_2}{\rightleftharpoons}} DNA_{II:Is}$ $DNA_{II:Is} \xrightarrow{k_{-c}} DNA$ $DNA \xrightarrow{k_I} DNA + RI$ $RI \xrightarrow{\gamma_I + \gamma} \emptyset$	• Mechanistic, biologically-plausible model of ColE1 replication. • DNA produces RNAI, which inhibits replication. • DNA replicates with self-produced RNAII primer, which is destructively blocked by RNAI.		
Three-Species (Simplified) ColE1 Replication	$DNA \xrightarrow{k_p} DNA_p$ $DNA_p \xrightarrow{k_{rep}} DNA + DNA$ $DNA \xrightarrow{k_{tx}} DNA + R$ $DNA_p + R \xrightarrow{k_I} DNA$ $R \xrightarrow{\gamma_I + \gamma} \emptyset$	• Inspired by Brendel & Perelson ColE1 model, but simplified. • RNAI-binding loop is reduced to a single reaction. • Replication loop is reduced to two reactions and one state. • With some assumptions, reduces to zero-order replication, equivalent to dummy-triggered replication.		

2 MODEL DETAILS

2.1 What is this?

This document is a quick guide to several different effective ways to add a replicating DNA species to a Chemical Reaction Network (CRN) model of a biocircuit, along with a few warnings about how *not* to do it. We've provided three different DNA replication models of varying complexity and biological plausibility.

2.2 What's an "effective" model of DNA replication?

This guide is written for engineers and biologists who need some way to represent replicating DNA in their biocircuit models. In most cases, the most important feature of such a model is that it can hold a DNA species at (roughly) a constant concentration. We will consider a model that can do this a "good" model. An even better property of a model of replicating DNA is to produce a realistic *distribution* of copy numbers over time and across a simulated population – we will also consider how well models can do this in Section 3.

For the purposes of this document, an "effective" or "good" model of replicating DNA is a CRN model that can stably hold a DNA species at a roughly-fixed copy number. A *really* good model of replicating DNA is a CRN model that accurately reproduces the real-world steady-state copy number distribution of a DNA species.

This document was written to help engineers who might need to model replicating DNA. As such, "effective" does *not*, for the purposes of this document, imply that the model accurately represents a real mechanism of DNA replication, or that it can provide any insight into natural replication systems.

2.3 Who is this for?

This guide is intended for synthetic biologists and systems biologists who use CRN models and who might need to model a biocircuit with explicitly-tracked DNA species. In particular, it is intended for those working with stochastic models – for deterministic models, the trivial model is probably fine (see Section 4.3).

2.4 Why should I care?

See Section 4.

2.5 What's wrong with the trivial model? Why shouldn't I use it?

The trivial replication model is the most obvious way to model replication of a DNA species *DNA* – simply have *DNA* copy itself from time to time at some rate, say α. This can work just fine in the deterministic limit of high concentrations and ample mixing – together with a dilution term $DNA \xrightarrow{\gamma} \emptyset$, it translates to an ODE like

$$\frac{DNA}{dt} = (\alpha - \gamma)DNA$$

.

This has a steady-state value when $\alpha = \gamma$, though we might be suspicious that it is not a *well-defined* steady state – whatever concentration *DNA* starts at, it will stay at.

The problem is that in anything other than the infinite deterministic limit, the trivial replication mechanism doesn't keep copy number stable. Figure 1A shows what you get

if you try simulating this mechanism in a stochastic regime, using Gillespie's Stochastic Simulation Algorithm (SSA) [4].

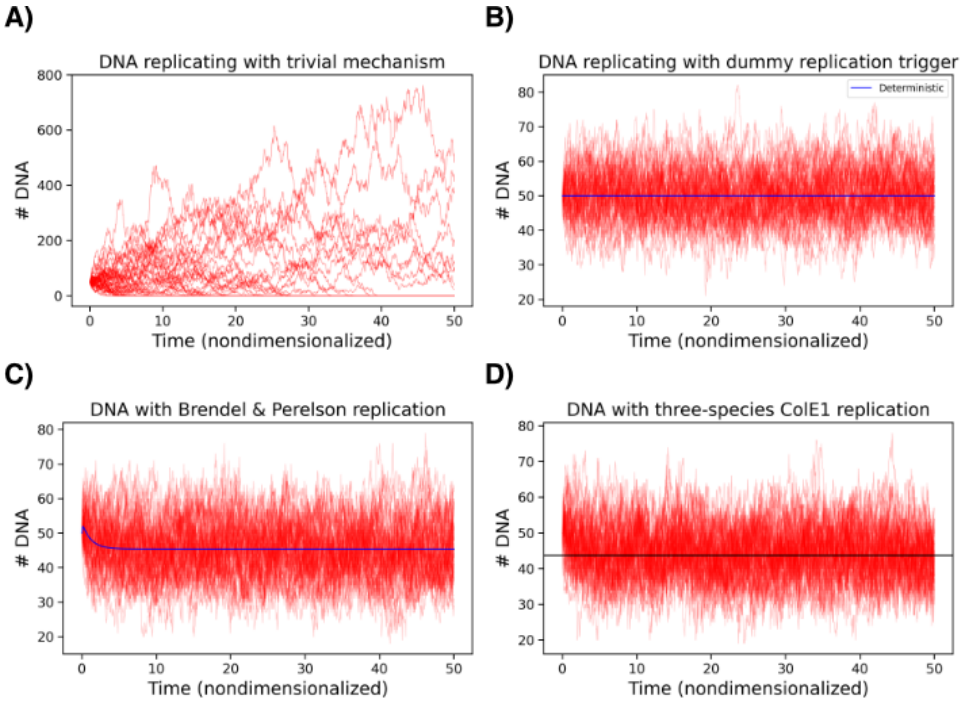


Figure 1. Typical dynamics of DNA replicating with different mechanisms. Each line is one simulated cell. **A)** Trivial $DNA \rightarrow DNA + DNA$ mechanism. The distribution does not go to steady state unless all cells lose their plasmids – cells with high DNA concentration will tend to gain copies indefinitely. **B)** Dummy-triggered replication. **C)** Brendel & Perelson's ColE1 replication mechanism. **D)** Three-species reduced ColE1 replication mechanism.

Each line in Figure 1A is a sample trajectory for a DNA species starting at 50 copy number and allowed to run with $\gamma = \alpha = 1$. Cell growth is implemented with the dilution reaction (i.e., this is a BioSCRAPE SSA simulation, not a BioSCRAPE lineage simulation). Fifty independent results are shown – most of them collapsed down to copy number 0 by about $t = 150$.

The trivial replication mechanism is not stable. Any fluctuation to a higher copy number leads to an increase in overall DNA replication rate, which leads to higher copy number. On the other hand, fluctuations to lower copy number reduce overall DNA replication rate, which leads to lower copy number. A DNA species replicating this way is effectively undergoing a random walk in concentration with an absorbing state at 0, which is why real self-replicating DNAs either replicate much faster than dilution (and so grow indefinitely until they use up one or another of their host's resources) or employ a copy number control system of some kind.

2.6 Real replication involves delays. Can't we salvage the trivial model by adding a delay on the order of cell replication time?

Not really. Adding a delay doesn't solve the inherent instability of the trivial model. It just makes the DNA population disappear/explode in discrete bursts, instead of more-or-

less continuously.

2.7 What in the world is that species R in the first model? What kind of molecule is the trigger supposed to represent?

R is a hack to get zero-order *DNA* production while keeping them model a CRN. It does not represent any real molecule.

The dummy replication trigger model represents an attempt to fix the fundamental problem of the trivial replication mechanism (*i.e.*, that it fails because first-order self-replication coupled with first-order dilution is unstable). The trivial mechanism's ODE description looks like $\frac{dDNA}{dt} = (\alpha - \gamma)DNA$, which has no non-zero steady state. If replication were, instead, zero-order (that is, constant with DNA count rather than proportional to it), then we would have something more like $\frac{dDNA}{dt} = \alpha - \gamma DNA$, which has steady state α/γ.

The problem is that a CRN can't directly represent zero-order self-replication because the reaction $DNA \rightarrow DNA + DNA$ will always have a rate dependent on the concentration of DNA! A CRN can, however, *emulate* zero-order replication by tying a normal replication reaction to the zero-order production of a rate-limiting dummy molecule that is consumed by the reaction.

We recommend this model for most applications for any DNA with large enough copy number that it won't readily be lost (this threshold is parameter-dependent, but as a guideline, if a DNA doesn't need active partitioning to maintain, this model is probably fine). It isn't *just* a hack to get meaningful steady-state behavior – in the right parameter regime, Brendel & Perelson's model of ColE1 plasmid replication reduces to essentially this model (see section 2.10).

2.7.1 Why use the dummy trigger? Why not just have plasmid directly replicate at zero-order rate?

I settled on the dummy trigger mechanism because it was trivial to implement in CRN-based simulators. If you are using a simulator where you can assign arbitrary (non-CRN-like) propensities to reactions, then you can certainly have the reaction $DNA \rightarrow DNA + DNA$ occur at zero-order propensity and omit the dummy trigger species.

Even staying strictly within the world of CRNs, in many cases, you could instead use the simpler model $\emptyset \rightarrow DNA$. The dummy replication trigger model is more general, however, and can handle some cases where direct zero-order DNA production fails, such as:

- If you need to do something other than simply make a new plasmid during replication (for example, when replication triggers an unbinding event), or
- If you have more than one competing, inheritable DNA species (for example, when tracking a circuit using replicating DNA modified by an integrase).

2.7.2 The dummy trigger looks suspiciously like a DNA polymerase. Could you instead model replication using a limiting polymerase to keep replication zero-order?

The simplest model of a DNA polymerase would look something like $DNA + \text{Polymerase} \rightarrow DNA + DNA + \text{Polymerase}$, which occurs at a rate proportional to the concentration of *DNA*. This is functionally identical to the trivial model, and is just as unstable.

You could instead use a Michaelis-Menten polymerase with reactions

$$DNA + \text{Polymerase} \leftrightarrow DNA : \text{Polymerase} \rightarrow DNA + DNA + \text{Polymerase}$$

.

This polymerase model should approximate zero-order replication for (sufficiently) high-copy plasmids if the production reaction is sufficiently rate-limiting. This is arguably a more biologically *plausible* model – though notably, real-world plasmids typically use copy number control mechanisms beyond rate limitation by host polymerase, so it is not clear that this model's *plausibility* should translate to better *biological plausibility*.

2.8 There's a lot going on in the Brendel & Perelson model. What is this, and what are all those states?

Real cells keep DNA concentrations constant. One principled way of modeling DNA replication is to simply describe how they do that. There are a number of different DNA copy control mechanisms, but one of the simplest and best-understood is the ColE1 copy number control system, described in model form by Brendel & Perelson in 1993 [1].

Briefly, ColE1 plasmids replicate using a *cis*-acting RNA primer, called RNAII. That primer can be blocked by a complementary RNA called RNAI, which is constitutively produced by the ColE1 plasmid. This means that the per-plasmid rate of ColE1 replication drops as the concentration of ColE1 plasmids increases, which gives ColE1 a finite steady state.

In the absence of RNAI, a ColE1-based *DNA* species are duplicated by initiation of RNAII transcription ($DNA \rightarrow DNA_{RIIs}$), elongation of RNAII ($DNA_{RIIs} \rightarrow DNA_{RIIL}$), attachment of a DNA polymerase to the RNAII-primed DNA complex ($DNA_{RIIL} \rightarrow DNA_P$), and, finally, replication by the attached DNA polymerase ($DNA_P \rightarrow DNA + DNA$). If a DNA polymerase does not bind to an elongated RNAII, the RNAII will be cleaved off, returning the plasmid to its initial state ($DNA_{RIIL} \rightarrow DNA$).

ColE1 does, however, produce RNAI ($DNA \rightarrow DNA + RI$). RNAI can bind to RNAII while it's being transcribed, blocking elongation ($DNA_{RIIs} + RI \rightarrow DNA_{II:Iu}$). The RNAI:RNAII complex formed this way is unstable and can reverse easily, but can switch into a much more stable form ($DNA_{II:Iu} \rightarrow DNA_{II:Is}$) from which both RNAs can be cleaved off the plasmid by RNase action, resetting the plasmid ($DNA_{II:Is} \rightarrow DNA$). This serves as the negative feedback mechanism that lowers ColE1's replication rate as its copy number increases.

In their original description of this model, Brendel and Perelson also describe optional sequestration of the $DNA_{II:Is}$ state by *Rom* binding, which leads to a lower steady state copy number [1]. Freudenau *et al* further extend this model with additional feedback by uncharged tRNA, which occurs under starvation conditions, and experimentally parameterize the model. For simplicity, I omit these two extensions in this work.

2.9 What's the relationship between the "Brendel & Perelson" and the "Reduced ColE1" model?

The reduced Brendel & Perelson ColE1 model is my attempt to capture the most important phenomenological features of the Brendel & Perelson model using fewer

DNA states and fewer reactions (a little under half of each). In the three-species model, DNA still produces a regulatory RNA species R. It can initiate replication by going into a state DNA_P. From here, it can either finish replicate ($DNA_P \rightarrow DNA + DNA$) or be reset by consuming an RNAI molecule ($DNA_P + R \rightarrow DNA$).

2.10 What's the relationship between the "Reduced ColE1" model and the "dummy replication trigger" model?

Under realistic parameter regimes (specifically, when R dynamics are fast and k_p is relatively fast compared to k_{rep}), this three-species ColE1 model can be approximated by a single-species model equivalent to the dummy-triggered replication model with replication rate $k_{rep}k_p(\gamma+\gamma_I)/(k_I(k_{tx}-k_p))$.

This approximation requires:

1. Dynamics of creation and destruction of R must be fast (i.e., R must be at quasi-steady state compared to DNA and DNA_p.
2. Replication must be bottlenecked by the $DNA_p \rightarrow DNA + DNA$ reaction (i.e., $k_p \gg k_{rep}$). This is true for the three-species model with parameters fit against simulations from the full Brendel & Perelson ColE1 model.
3. $k_{tx} > k_p > \gamma$ and $k_{rep} > \gamma$ (required for stability of the three-species ColE1 model).

Since R dynamics are fast, we'll assume that R is at quasi-steady state:

$$\frac{dR}{dt} = k_{tx}DNA - (\gamma+\gamma_I)R - k_I DNA_p R$$

$$\text{At steady state: } R = \frac{k_{tx}DNA}{(\gamma+\gamma_I)+k_I DNA_p}$$

Because k_{rep} is, by assumption, relatively slow, we can think also consider the quasi-steady state concentration of DNA_p. At quasi-steady state, the equilibrium condition between DNA and DNA_p means that

$$k_I * DNA_p * \frac{k_{tx}DNA}{(\gamma+\gamma_I)+k_I DNA_p} = k_p * DNA$$

so

$$DNA_p = \frac{k_p(\gamma+\gamma_I)}{k_I(k_{tx}-k_p)}$$

Surprisingly, we have discovered that the concentration of DNA_p is roughly *constant* regardless of the copy number of the plasmid. As long as this is true, the rate of replication (i.e., the rate at which DNA_p goes to $2DNA$) is

$$k_p * DNA_p = \frac{k_{rep}k_p(\gamma+\gamma_I)}{k_I(k_{tx}-k_p)}$$

.

Since this is constant, we're back to the zero-order (e.g., dummy-triggered) replication model where DNA species replicate at a constant, DNA-independent rate! This

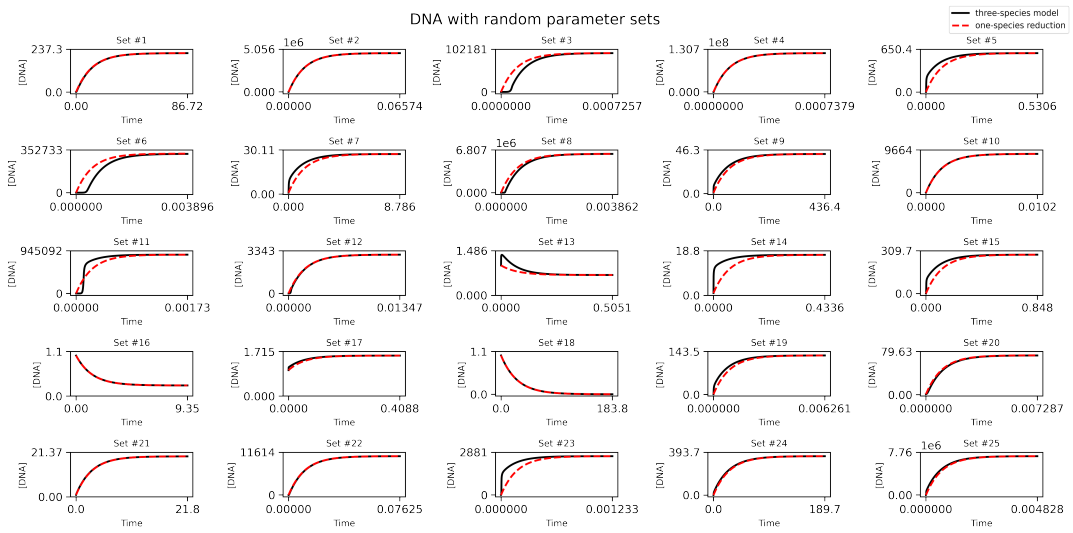


Figure 2. Simulated trajectories of replicating DNA the three-species ColE1 model (black lines), with trajectories for the equivalent one-species reduced model (dashed red lines), for 25 random parameter sets chosen so that DNA has a finite, nonzero steady state and $k_p > k_{rep}$.

zero-order, one-species reduced model will bring DNA to the same copy number and usually has behavior qualitatively very similar to the three-species model (Fig. 2).

Note that in the opposite regime, where $k_{rep} \gg k_p$, a similar reduction can be made to a single-species model where DNA replicates at rate

$$\frac{k_p}{1+\beta DNA}, \text{ where } \beta = \frac{k_I k_{tx}}{k_{rep}(\gamma+\gamma_I)}$$

See the ipython notebook "simple_bp_model_reduction.ipynb" in the code accompanying this document for more details.

3 EVALUATING MODELS

3.1 What's going on in the "steady-state distributions" plots?

These plots demonstrate how closely the distribution of DNA copy numbers predicted by each model hews to real-world plasmid copy number data. Each color represents a different plasmid with a different copy number (and, in real cells, a different replication and copy number control mechanism). The dotted curves are empirical distributions measured by microscopy, while the solid curves are generated from BioSCRAPE lineage simulations run to a "reasonable steady-state time."

3.2 What do the "empirical distributions" come from?

Empirical distributions were pulled from Figure 1e of Shao *et al.*, Nature Communications 2021 [11]. Copy numbers were measured by microscopy from approximately 1-2 thousand growing *E. coli* cells bearing plasmids with binding sites for a GFP-fused transcription factor. I extracted frequencies from the figure using WebPlotDigitizer, rounding each data point to the nearest whole plasmid copy number.

3.3 How did you fit the models against the empirical distributions?

3.3.1 Dummy replication trigger

The dummy replication trigger model only has one relevant parameter to tune, namely the production rate of R (dilution rate is known and simply sets a timescale for the system, and the actual replication reaction merely needs to be fast enough to consume all R quickly relative to other processes). For each empirical plasmid distribution, R production rate was set to make the analytic steady state of the model (R production rate divided by dilution rate) equal to the observed mean plasmid copy number.

3.3.2 Brendel & Perelson, reduced Brendel & Perelson

The two Cole1 models required somewhat more complex fitting. In brief, parameters were optimized with Scipy on simulations using BioSCRAPE lineages in turbidostat mode.

For different parameter sets, I used BioSCRAPE lineages to grow a population out to a reasonable steady-state time with simulated copy number distributions extracted from the final time. Parameters were fit to each plasmid's empirical observations by computing least squared distance between the empirical distribution and simulated distribution at each copy number.

Importantly, one of Shao *et al*'s surprising findings was that large numbers of "plasmid bearing" cells do not appear to actually carry plasmids, *even in the presence of antibiotic selection for those plasmids*. Therefore, for the ColE1 models to have any hope of closely reproducing real-world copy number measurements, my simulations must be able to handle common cases of total plasmid loss. As the models are written, with no other modification, plasmid loss is an absorbing state, and so all reasonable plasmid distributions would quickly tend toward total loss; therefore, I had to include some form of selection in my models.

BioSCRAPE allows the use of "growth rules" that tie growth (and replication) rate to the concentrations of molecules in the cell. We can therefore somewhat crudely represent antibiotic selection by applying the growth rule that cells stop growing when they have zero DNA. This allows cells with plasmid loss to persist in the population while also being subject to antibiotic selection, as by a bacteriostatic antibiotic (Shao *et al* measure copy numbers of plasmids bearing ampicillin resistance).

3.4 Why don't you have steady-state distributions for the trivial model?

The trivial model is unstable and has no steady state distribution, aside from the absorbing state of total plasmid loss.

3.5 So these models will hold plasmids at constant copy number ($\pm$ noise)?

Almost. What "copy number control" mechanisms really act on is DNA *concentration*, not copy number. For most purposes, this isn't a meaningful difference, but do note that the empirical distributions used in this report are distributions of *copy number*, and the models were parameterized accordingly.

3.6 How fast are these models to simulate, relative to each other?

Table 1 shows representative wall-clock run times for various stochastic simulations using each of the three replication models. As you can see, dummy-triggered replication

is typically much faster than either ColE1 model, and different simulation conditions can make either ColE1 model faster than the other.

All simulations were performed inside Jupyter Lab notebooks on a mid-2014 macbook pro (2.8 GHz Intel Core i5 processor, ample 1600 MHz DDR3 RAM).

System	# Cells	Dummy-Triggered	Full ColE1	3-Species ColE1
DNA only (non-lineage)	50	0.27 sec	13 sec	10 sec
DNA only (guessed)	64	0.35 sec	11 sec	7.5 sec
DNA only (hand-tuned)	64	0.52 sec	###	###
DNA only (optimized)	64	###	###	###
5-node CRISPRlator	1	###	###	###
5-node CRISPRlator	64	###	N/A	N/A
TLG, copy number 30	1,056	5 hr, 14 min	N/A	N/A
TLG, copy number 100	1,056	17 hours, 9 min	N/A	N/A

Table 1. Wall-clock run times for different simulations with each mechanism, where tested. TLG = Temporal Logic Gate. "guessed", "hand-tuned", and "optimized" refer to the parameter sets used for simulation. See Section 4.4 for details of the 5-node CRISPRlator; see Section 4.5 for details of the temporal logic gate (TLG).

3.7 Which model should I use? Why?

That depends on what you need from your model. If you just want to see the expected behavior of your circuit under stochastic assumptions with minimal fuss, I recommend using the dummy-triggered replication – it is the simplest to implement of the working models, and runs much more quickly than either ColE1-based model. In the right parameter regimes, the dummy-triggered replication model approximates the other two quite well (see Section 2.10).

You might need to use a more realistic replication model if you anticipate coupling your circuit's DNA to other reactions (e.g., RNA degradation machinery) or if you anticipate a reviewer who asks you to use a more realistic replication model. In this case, either the Brendel & Perelson ColE1 replication model or the reduced, three-species ColE1 replication model should suffice. You may want to try both to see which runs most efficiently; counterintuitively, sometimes the full ColE1 model is faster to run than the reduced model, although it is virtually always more effort to implement.

3.8 What's missing from these models?

All of the models presented here assume random (binomial) partitioning of DNA. This is realistic for some DNA and not for others, and your milage may vary accordingly. Non-random partitioning can be added straightforwardly to Bioscrape lineages simulations, if desired.

As written, these models also do not have any kind of antibiotic selection or other maintenance system, and nothing will stop a simulated DNA from dying out. This may be a problem for simulations of low copy number, especially since empirical measurements of plasmid copy number suggest that 0 is a perfectly reasonable number of plasmids to expect to find in a cell! The lineage simulations used for parameter-fitting

include an added selection mechanism roughly mimicking ampicillin resistance (cells with no plasmid do not grow, and are likely to be pushed out by growing cells).

4 MOTIVATION

4.1 Why did you write this?

I wrote this guide because I kept making models involving explicit DNA species, for a variety of reasons, and the simplest hacks I kept coming up with to handle them kept not working. This was doubly true when I started running stochastic simulations of biocircuits. I kept running into problems like transcription factors "hiding" from dilution by binding to DNA, or total plasmid loss, or plasmids spiraling out to infinite concentration and crashing Python. The standard modeling tools in my toolbelt didn't cover cases with replicating DNA.

Eventually I figured out a couple of replication mechanisms that didn't totally crash and burn or compromise on possibly-important kinetic details. Hopefully this document will save other modelers in similar situations the trouble.

4.2 I've never had to model DNA replication in my models. Why would I ever need to?

Most biocircuit models don't even include species representing DNA, much less model their replication. We can generally get away with this because of one or both of two common assumptions:

1. **The concentration of DNA species is held constant by the cell.** Most DNAs (either chromosomal or plasmid) are copy-number controlled by more or less complex cellular processes, and we can usually assume that these processes are functioning properly in the background.

2. **We do not need to track binding of species to the DNA.** Typically we assume that the actions of DNA-binding proteins (e.g. transcription factors) can be well-approximated by a Hill function or some other simple transfer function. The Hill assumption eliminates the need to explicitly track bound and unbound DNA states.

As I've said, these assumptions are often good enough to use. However, there are some biocircuits that violate these assumptions. For example:

1. When binding and unbinding of a transcription factor is slow relative to other processes (as can be the case with, for example, dCas9-based CRISPRi transcription factors [6]), we may need to explicitly model binding and unbinding kinetics to understand whether or not that transcription factor will function in a circuit of interest.

2. Any time the state of a piece of DNA is malleable, inheritable, and important for circuit function, we may wish to track that DNA explicitly. Recombinase-based circuits that function by flipping, removing, or integrating chunks of DNA are a clear example of this case, particularly those such circuits on plasmids [9, 5].

3. Any time we expressly wish to know the effect of the noise in copy number of a DNA species, we will likely want a way to track it.

Another case in which explicit DNA tracking might be useful would be if we wished to predict the transfer function of a transcription factor from purely kinetic parameterization, rather than empirically, though this is rare. Usually it is easier to put a transcription factor in a cell, induce its expression, and watch its effects than it is to directly measure its binding and unbinding rates; however, sometimes kinetic data may be available in the literature for a transcription factor for which there is no good direct transfer curve measurement, and modeling the transcription factor kinetically (rather than with a Hill assumption) may be appropriate.

4.3 So these models are only useful for stochastic simulations?

More or less. In general, you can get away with using the trivial replication mechanism $DNA \rightarrow DNA + DNA$ if you're only using deterministic models, because you can exactly balance DNA replication against dilution.

4.4 When would you *actually need* to model DNA replication? Give me a concrete example.

Here's one: the CRISPRlator (Figure 3).

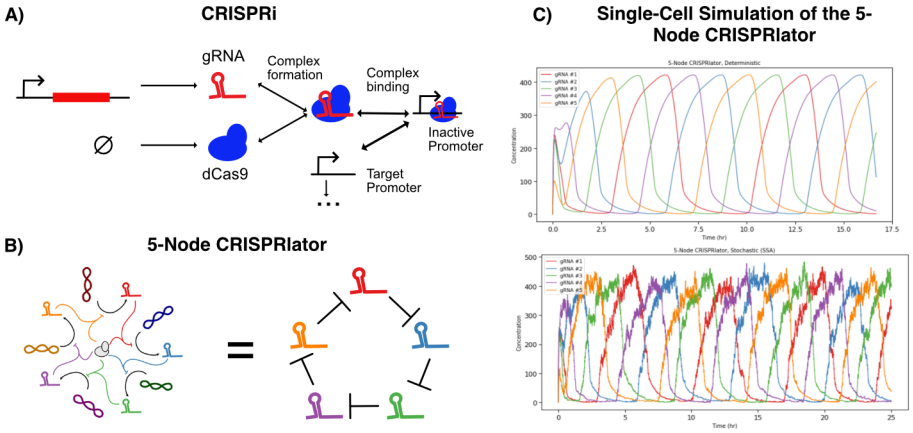


Figure 3. **A)** A simple model of general CRISPRi systems. Guide RNAs (gRNAs) complex with a shared pool of dCas9. dCas9:gRNA complexes bind to DNA targets, blocking transcription from those targets. Targets can be other gRNAs, as in the 5-node CRISPRi oscillator (CRISPRlator) **(B)**. **C)** Deterministic and stochastic (SSA, non-lineage) simulation of the 5-node CRISPRlator.

The CRISPRlator is just one example of a CRISPR interference (CRISPRi) circuit. CRISPRi circuits are transcriptional networks that use guide RNAs (gRNAs) that bind to catalytically deactivated dCas9 (dCas9) to form (theoretically) independent, orthogonal repressors [7, 8, 3]. The CRISPRlator in particular is a ring oscillator made of an odd number of gRNAs that repress each other in a cycle. With the right parameters, the CRISPRlator will naturally oscillate, with gRNAs activating in pulses, in turn [10].

CRISPRi is an appealing system for several reasons, but CRISPRi also has potential drawbacks compared to traditional gene expression networks [2, 12]. In particular, dCas9

has shockingly slow binding kinetics – a single dCas9 molecule has been calculated to take an average of *six hours* to find a single DNA target in an *E. coli* cell, necessitating large numbers of targets and/or dCas9 molecules for circuits to operate on reasonable timescales [6]. One might use simulation determine whether a particular CRISPRi circuit will function with particularly slow binding times.

Note that to answer this question, we have to dispose of the usual Hill function approximation of gene repression – one of the most common simplifying assumptions in the gene circuit literature – because that assumption *assumes* fast binding kinetics. To see the effects of slow binding kinetics, we have to explicitly model slow binding kinetics, which means explicitly tracking of bound and unbound DNA species. DNA replication might be a nice feature in such a model.

You may also simply want to know how fluctuations in plasmid copy number affect the performance of the CRISPRlator. Can the CRISPRlator function if DNA copy numbers fluctuate in ways consistent with noisy DNA replication? Can it survive random partitioning? Simulations with DNA replication could help answer those questions.

4.4.1 *Is there* really *no other way to model this circuit?*

For a deterministic or SSA simulation, you could assume that DNA concentrations are held constant by the cell (although you'll have to take care to account for the unbinding reactions implicit in balanced replication/dilution so that binding to DNA isn't a dilution-free 'tax haven' for repressors!). This lets you model the CRISPRlator without any need for explicit DNA replication.

Things get tricker if you want to *also* track the CRISPRlator in a lineage of growing, dividing cells. You may want to do this to, for example, see whether or not a CRISPRlator will stay synchronized across a population, or if you want to couple the CRISPRlator to another circuit in an agent-based model with spatial dynamics.

If you try to simulate the CRISPRlator in a lineage model without replicating DNA, you quickly run into a few problems. The first is that if a cell keeps a constant *copy number* of a plasmid while *growing in volume*, then the concentration of plasmid will consistently and unrealistically drop by a factor of two with every growth cycle. You will also run into a thorny question of how to add more plasmids to cells when they divide. The obvious thing to do is to instantaneously replicate all of the plasmids at cell division time, but this will create a bunch of un-repressed plasmids all at once, which can scramble the state of the CRISPRlator and kill oscillations.

There may other ways to make a lineage model of the CRISPRlator work, but the best way I have found is to add explicit DNA replication mechanisms.

4.4.2 *Does the CRISPRlator work in stochastic simulations with replicating DNA?*

Yes. Figure 4 shows representative simulations of a single-cell 5-node CRISPRlator using each DNA replication method.

More interestingly, we can now simulate lengthy lineages of growing, replicating cells expressing the CRISPRlator. Figure 4E shows results from a lineage simulation using the dummy-triggered replication model parameterized identically to the one used for Figure 4B. This simulation begins with a single cell, which grows and divides every doubling of cell size. The total population is capped at 64 by killing a random cell whenever a cell division would bring the population above 64. Each dot is the

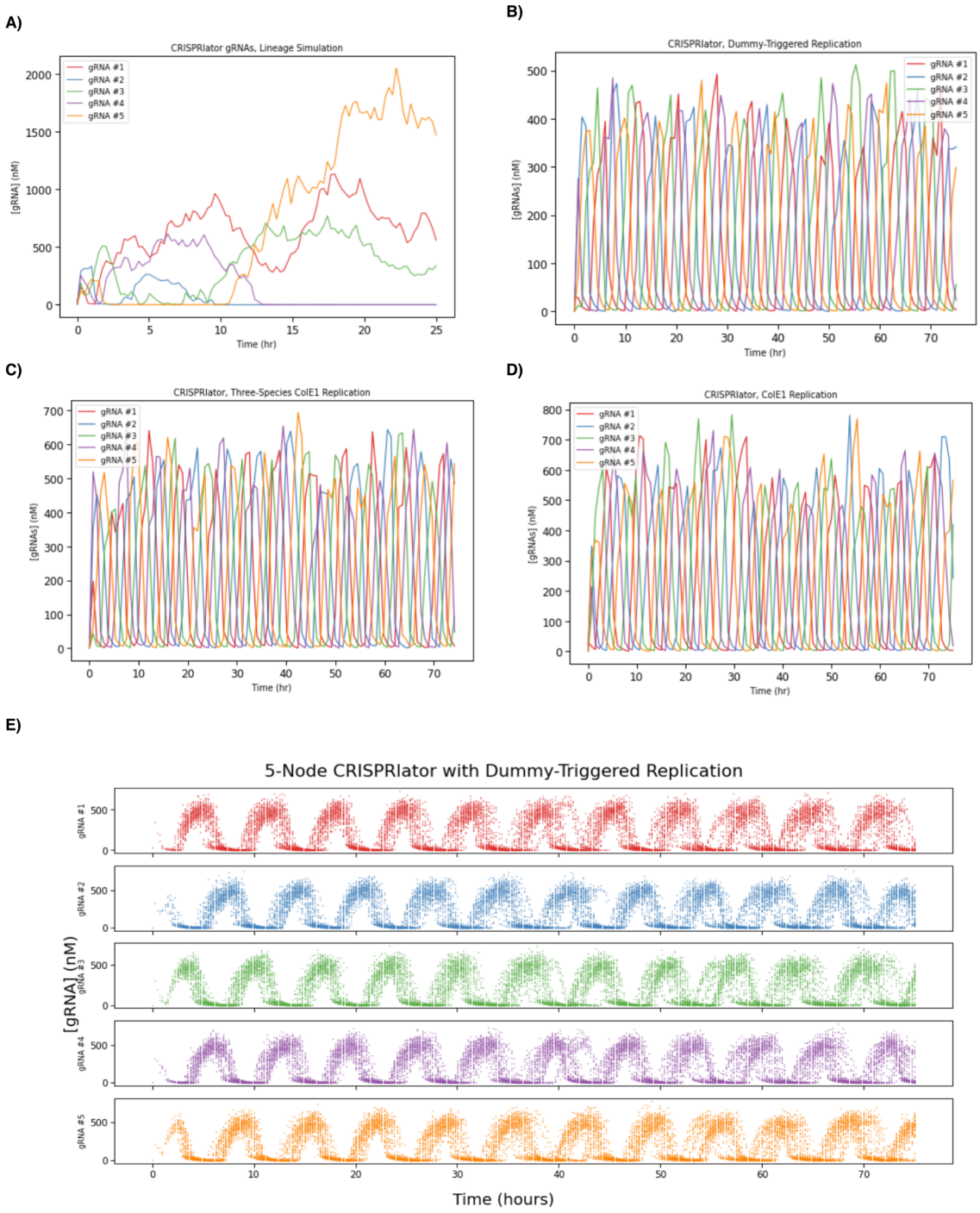


Figure 4. The five-node CRISPRlator in a single growing cell with DNA replication modeled using **A)** the trivial replication model (note the shorter time scale), **B)** dummy-triggered replication, **C)** Brendel & Perelson's ColE1 replication model, and textbfD) the simplified 3-species ColE1 replication model. Each line is a sum of the *concentrations* of all species containing one of the five gRNAs. **E)** Lineage simulation of the 5-node CRISPRlator in growing, dividing cells over approximately 150 generations, with a population cap of 64 cells.

concentration of total gRNA for one of the five gRNA in one cell at one time. These simulations show that over 150 generations, the CRISPRlator remains largely coherent – although increasingly less so as the simulation progresses (note the spread in the "tails" where each gRNA species falls to zero concentration).

Finally, we can simulate the effect of dropping down the copy number of the plasmids bearing the CRISPRlator. Figure 5 shows representative traces from one of the five gRNAs in CRISPRlator variants on plasmids varying from single-copy to copy number ten. Guide RNA production rates are scaled so that the expected gRNA production rate are the same as in the simulations used for Figure 4, and selection has been added to the models to counteract plasmid loss (see Section 3.3.2 for details).

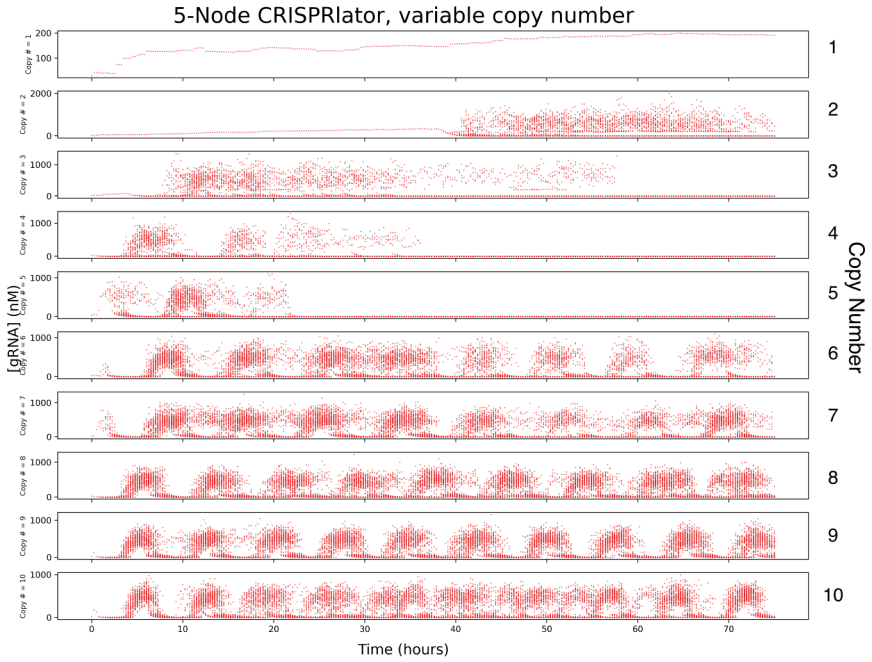


Figure 5. 5-node CRISPRlator at different copy numbers.

4.5 Okay, but that's just one example, and I don't care particularly about CRISPRi circuits. Can you give another concrete example?

Consider the integrase-based temporal logic circuit [5], which is a special case of the general recombinase-based state machine [9]. This circuit uses two different serine DNA integrases *intA* and *intB* under inducible control of molecules A and B, respectively, to flip pieces of a shared reporter module. Induction of either integrase permanently alters the reporter module so that the circuit produces different outputs depending on whether it has been exposed to A only, B only, A followed by B ($A \Rightarrow B$), B followed by A ($B \Rightarrow A$), or no inducer at all.

A single cell expressing a temporal logic circuit is an imperfect sensor. Even if the cell is exposed to $A \Rightarrow B$, it is possible that it could, by stochastic fluctuation, not flip with integrase *intA* for so long that it eventually runs into a B molecule and integrates first with *intB*. The shorter the time delay between the introduction of A and B, the more likely this sort of mis-firing will be.

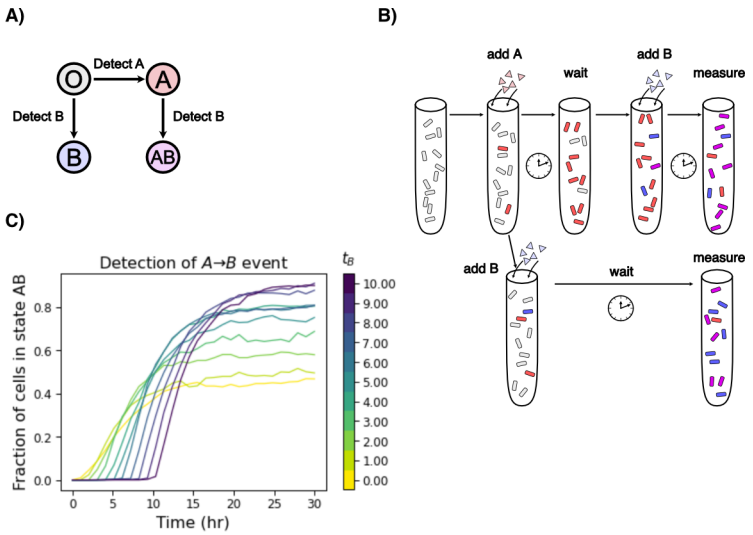


Figure 6. Overview of the temporal logic gate. **A)** State machine describing a single temporal logic gate unit. "Detection" is classically implemented with integrases whose activities are induced by signals *A* and *B*. **B)** The behavior of a population of cells where each cell contains a single temporal logic gate. The final distribution of cell states can be used as a readout of the time delay between introduction of *A* and introduction of *B*. **C)** Representative simulation of a genomically-integrated temporal logic gate (i.e., without explicitly replicating DNA species). Each curve is the fraction of cells in state *AB* over time for a different delay time t_B between introduction of *A* and *B*.

In a large population of temporal-logicl-circuit bearing cells, this imperfection can be exploited to create a population-level, analog measurement of not just which input appeared first, but the *time delay* between the appearance of the two. If one input appears rapidly after the other, then many cells will "misfire" and report the wrong temporal value; if the two inputs are separated by a significant amount of time, then almost all cells will react to the first input before the second input arrives, and the population will homogeneously report the correct value. With proper calibration, the final distribution of cell states can be used to back out the time difference between inputs.

Equivalently, this type of population-level circuit can be used to measure differences in *concentration* of inputs that appear at the same time.

Importantly, the population-level circuit assumes that each cell has a single, simple state. This is achieved by using a chromosomally-integrated reporter module, so that the cell has a single, unique state (or, at least, will after a round or two of division). One could easily imagine performing the same type of population-level measurement in a single cell by using a state-bearing plasmid with a high copy number. This could make the circuit much more compact, and potentially simpler to sample depending on the circuit's intended environment.

Will a single-cell, plasmid-based temporal logic gate still perform as advertised? There are practical, integrase-related difficulties with making such a circuit – as originally designed, multiple copies of the gate's reporter module would recombine in unexpected and unwanted ways – but even setting those aside, moving a population-level circuit into a single cell adds new complications and sources of noise. For one thing, recording

events between modules are no longer independent – transcription of a single integrase mRNA can produce a number of module-flipping events. The frequency of a plasmid state in a cell might also not be stable over generations, due to random partitioning at division.

A straightforward way to ask whether or not a single-cell temporal logic gate can still give analog measurements is to simulate it. To do so, we should use stochastic simulation (since stochastic fluctuations are likely important for plasmid dynamics) and we will need to model plasmid replication (since stateful plasmids are the principle species acted on by the circuit).

4.5.1 Is there really *no other way to model this circuit?*

You could probably get away with representing each cell as a finite Markov chain with transition rates fixed by the presence or absence of each signal molecule, with the state of the chain encoding the number of plasmids in that cell with each reporter state. This could be a good approximation *if* you are sure you can approximate each cell's (and each plasmid's) internal dynamics with Markovian state transitions. You could easily add partitioning dynamics to this sort of model by splitting each cell at fixed intervals, with plasmids duplicated and binomially partitioned at division time.

What you will lose with the simpler Markov chain description of the temporal logic gate is the ability to couple the gate's activity to arbitrary dynamic gene networks. The example shown here arguably doesn't strictly need the full power of BioSCRAPE's stochastic molecular simulator, but it would not be difficult to imagine tying integrase activity to, for example, a feed-forward pulse generator, so that the gate only detects the $A \rightarrow B$ transition if B appears within a short window after the appearance of A.

4.5.2 Does the temporal logic gate work when modeled on a replicating plasmid?

It does, although with more noise than the original genome-located temporal logic gate. See Figure 7.

5 DATA/CODE AVAILABILITY AND REPRODUCIBILITY

5.1 What software did you use to simulate this stuff?

All simulations were performed in Python using the BioSCRAPE cell simulator, with software and package versions given in Table 2. Copy number data from [11] was extracted from Figure 1e using WebPlotDigitizer).

Software/Package	Version
Python	3.8.1
ipykernel	5.1.4
IPython	7.12.0
Jupyter Lab	2.1.5
Numpy	1.20.2
Matplotlib	3.4.1
Scipy	1.6.2
BioSCRAPE	1.0.3

Table 2. Software versions used for this report.

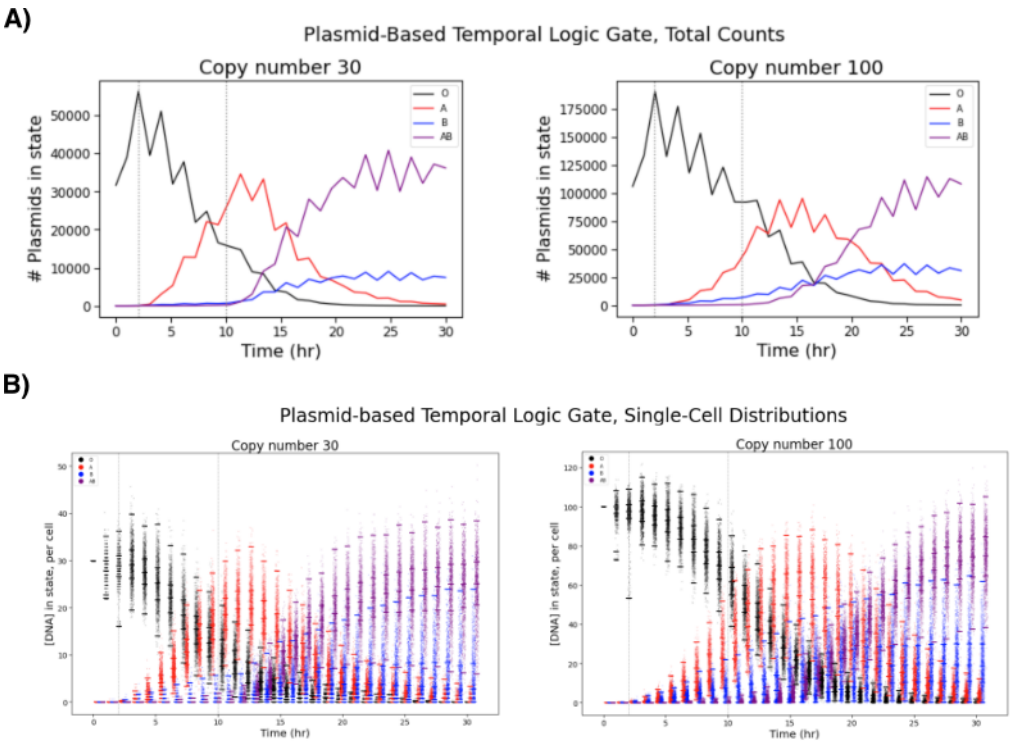


Figure 7. The temporal logic gate on a replicating plasmid. **A)** Total counts of plasmids of copy number 30 or 100 in each state across a population of 1,056 cells introduced to *A* at time 2 hours and *B* at time 10 hours. **B)** Same data as in **A**, disaggregated into individual cells' copy numbers of plasmids in each state. Points are shifted and jittered slightly on the time axis for visibility. Solid ticks mark the 1st, 25th, 50th, 75th, and 99th percentile copy numbers for each state at each time.

For this document, code was run using a mid-2014 MacBook Pro running macOS 10.14.6.

5.2 Do you have code for any of this? Where can I get it?

All code used in this document, along with examples and some additional analysis, can be found in the "code" folder in this document's GitHub repository at `https://github.com/sclamons/plasmid_replication_modeling`. All code is packaged as interactive Jupyter notebooks:

- **summary_table_plots.ipynb**: Descriptions of the models, simulations of just replicating DNA, and parameterization. Used for Figure 1, Table 1, and most of the main table. Start here.
- **simple_bp_model_reduction.ipynb**: Semi-rigorous derivations for reductions of the three-species ColE1 replication model (see Section 2.10), along with simulations to back up those derivations. Used for Figure 2.
- **CRISPRessilator.ipynb**: Worked examples of the five-node CRISPRlator (see Section 4.4) with no replication mechanism, trivial replication, dummy-triggered replication, and the full and three-species ColE1 replication mechanisms, as well

as simulations using dummy-triggered replication at a variety of copy numbers. Used for Figures 3, 4, and 5, and for Table 1.

- **temporal_logic_gate.ipynb**: Worked examples of the temporal logic gate (see Section 4.5 either as a single-copy genomic circuit or as a plasmid-based circuit on DNA with dummy-triggered replication. Used for Figures 6 and 7, and Table 1.

You will need to install the BioSCRAPE package to run the example notebooks ("pip install bioscrape"). BioSCRAPE is a performance-optimized, Cython-based Python package for efficient ODE and SSA simulation of biocircuits. See `https://github.com/biocircuits/bioscrape` and "Fast and flexible simulation and parameter estimation for synthetic biology using bioscrape" for more information.

REFERENCES

[1] Brendel, V. and Perelson, A. S. (1993). Quantitative Model of ColE1 Plasmid Copy Number Control. *Journal of Molecular Biology*, 229.

[2] Clamons, S. E. and Murray, R. M. (2017). Modeling Dynamic Transcriptional Circuits with CRISPRi. *bioRxiv*.

[3] Gander, M. W., Vrana, J. D., Voje, W. E., Carothers, J. M., and Klavins, E. (2016). Digital logic circuits in yeast with CRISPR-dCas9 NOR gates. *Nature Communications*, 8(1):15459.

[4] Gillespie, D. T. (1977). Exact stochastic simulation of coupled chemical reactions. *Journal of Physical Chemistry*, 81(25):2340–2361.

[5] Hsiao, V., Hori, Y., Rothemund, P. W., and Murray, R. M. (2016). A population-based temporal logic gate for timing and recording chemical events. *Molecular Systems Biology*, 869(12).

[6] Jones, D. L., Leroy, P., Unoson, C., Fange, D., Ćurić, V., Lawson, M. J., and Elf, J. (2017). Kinetics of dCas9 target search in *Escherichia coli*. *Science*, 357:1420–1424.

[7] Kiani, S., Beal, J., Ebrahimkhani, M. R., Huh, J., Hall, R. N., Xie, Z., Li, Y., and Weiss, R. (2014). CRISPR transcriptional repression devices and layered circuits in mammalian cells. *Nature Methods*, (11):723–726.

[8] Nielsen, A. A. K. and Voigt, C. A. (2014). Multi-input CRISPR/Cas genetic circuits that interface host regulatory networks. *Molecular Systems Biology*, 10(11):763.

[9] Roquet, N., Soleimany, A. P., Ferris, A. C., Aaronson, S., and Lu, T. K. (2016). Synthetic recombinase-based state machines in living cells. *Science*, 353(aad8559).

[10] Santos-Moreno, J., Tasiudi, E., Stelling, J., and Schaerli, Y. (2020). Multistable and dynamic CRISPRi-based synthetic circuits. *Nature Communications*, 11(2746).

[11] Shao, B., Rammohan, J., Anderson, D. A., Alperovich, N., Ross, D., and Voigt, C. A. (2021). Single-cell measurement of plasmid copy number and promoter activity. *Nature Communications*, 1475(12).

[12] for Engineered dCas9 with reduced toxicity in bacteria: impliciations for genetic circuit design. *Nucleic Acids Research*.