




# A MATLAB toolbox for modeling genetic circuits in cell-free systems<sup>†</sup>

Vipul Singhal <sup>1,\*</sup>, Zoltan A. Tuza <sup>2</sup>, Zachary Z. Sun <sup>3</sup>, and Richard M. Murray<sup>4</sup>

<sup>1</sup>Spatial and Single Cell Systems Domain, Genome Institute of Singapore, 60 Biopolis St, 138672, Singapore, <sup>2</sup>Department of Bioengineering, Imperial College London, Exhibition Rd, South Kensington, SW7 2BU, London, UK, <sup>3</sup>Tierra Biosciences, 1933 Davis St #223, 94577, CA, USA and <sup>4</sup>Control and Dynamical Systems and Biology and Biological Engineering, California Institute of Technology, 1200 E California Blvd, 91125, CA, USA

\*Corresponding author: E-mail: vipul\_singhal@gis.astar.edu.sg

<sup>†</sup>This work was performed while all authors were at Caltech. V.S. was with the Computation and Neural Systems option, Z.A.T. was a visiting graduate student with the Department of Control and Dynamical Systems, and Z.Z.S. was with the Biology option.

## Abstract

We introduce a MATLAB-based simulation toolbox, called *txtlsim*, for an *Escherichia coli*-based Transcription–Translation (TX–TL) system. This toolbox accounts for several cell-free-related phenomena, such as resource loading, consumption and degradation, and in doing so, models the dynamics of TX–TL reactions for the entire duration of solution phase batch-mode experiments. We use a Bayesian parameter inference approach to characterize the reaction rate parameters associated with the core transcription, translation and mRNA degradation mechanics of the toolbox, allowing it to reproduce constitutive mRNA and protein-expression trajectories. We demonstrate the use of this characterized toolbox in a circuit behavior prediction case study for an incoherent feed-forward loop.

**Key words:** cell-free synthetic biology; genetic circuits; mathematical modelling; chemical reaction networks; parameter inference

## 1. Background

One of the goals in synthetic biology is the engineering of genetic circuits to function in a predictable manner, so that these circuits may be used to control cellular behavior (1, 2). The design-build-test-learn (DBTL) cycle of these circuits *in vivo*, however, can be time consuming and expensive. In disciplines like electrical and aeronautical engineering, the design-build-test cycle is accelerated with the help of rapid prototyping environments like breadboards and wind tunnels, and associated dynamics simulation software such as PSpice and XFlow (3, 4). Analogously, it should be possible to accelerate the design of novel biological systems in synthetic biology using appropriate

rapid prototyping tools, an end to which cell-free protein synthesis (CFPS) systems have gained significant traction.

More generally, CFPS systems have been used in synthetic biology as a versatile tool for biosensing (5), for the manufacture of therapeutics (6), as a platform for discovery of biosynthetic pathways and biomaterials (7), as a tool for building artificial cells (8, 9), and as a platform for prototyping genetic circuits (10, 11). These systems can be reconstituted from individual molecules (12) or based on cell lysates (13), and can be used in either solution phase, encapsulated in liposomes (14, 15), in freeze-dried paper-based modes (16), or as part of microfluidic systems (17). Cell lysate-based systems, in particular, are made

Submitted: 11 August 2020; Received (in revised form): 7 December 2020; Accepted: 23 December 2020

© The Author(s) 2021. Published by Oxford University Press.

This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (<http://creativecommons.org/licenses/by-nc/4.0/>), which permits non-commercial re-use, distribution, and reproduction in any medium, provided the original work is properly cited. For commercial re-use, please contact [journals.permissions@oup.com](mailto:journals.permissions@oup.com)

of three components: a cytoplasmic extract, a buffer containing energy and raw material molecules, and a solution containing the DNA that encodes the circuit to be implemented. One example of such a system based on *Escherichia coli* extract is the TX-TL (transcription–translation) system (13, 18).

Cell-free systems have numerous advantages that make them suitable as circuit prototyping platforms in synthetic biology. First, since the DNA encoding the circuit is not constrained by the need for replication, restrictions due to plasmid selection markers and antibiotic compatibility are lifted. This allows for the rapid exploration of genetic circuit variants by trying different combinations of DNA species. Second, time-consuming cloning and transformation steps may be bypassed by using linearized DNA in the form of polymerase chain reaction (PCR) products or *de novo* synthesized fragments, which speeds up the DBTL cycle.

Examples of software for simulating general biochemical reaction networks include TABASCO (19), COPASI (20), Bioscrape (21) and MATLAB Simbiology<sup>®</sup>. TABASCO and COPASI are fast general purpose solvers that can incorporate stochastic simulations into circuit dynamics. Bioscrape is a Python-based simulator that leverages the speed of Cython to perform fast stochastic simulations with time delays, cell lineage tracking, and Bayesian parameter inference. Simbiology<sup>®</sup> is a MATLAB toolbox that follows the Systems Biology Markup Language (SBML) in its structure, in that it allows for the specification of the standard SBML features such as models, compartments, reactions, species, parameters, rules and events in an object-oriented manner. It works well with MATLAB's ordinary differential equation solvers, local and global optimization methods, plotting tools, and other functionalities.

Examples of modeling studies specific to cell-free systems include the one performed by Stögbauer *et al.* (22), who described a minimal rate equation model of the PURE cell-free system (12), and performed a fit of their model parameters to the experimentally measured time courses of both the expressed protein and mRNA. A more detailed model was presented by Mavelli *et al.* (23, 24), where the authors accounted for nucleotide and amino acid usage, considered tRNA aminoacylation separately from translation elongation, and modeled the energy regeneration system. Neither of these studies, however, explored fitting their model to an *E. coli* extract. Earlier work by Karzbrun and Noireaux (25), on the other hand, did fit models to protein and mRNA data from the TX-TL *E. coli* extract, but restricted their analysis to the first 60 minutes of system behavior. More recently, Moore *et al.* (26) have performed extensive characterization of parts in non-model bacteria extracts, such as those made from *Bacillus megaterium*. None of these studies, however, provide a software toolbox or attempted to use their characterized models to predict and validate the behavior of whole circuits.

In this article, we build on our initial work in (27) to describe a MATLAB-based software toolbox called `txtlsim` for prototyping genetic circuits in TX-TL. This toolbox comes with a library of parts that can be combined in different combinations to build circuit models that are predictive of the behavior of circuits *in vitro*. It does this by accounting for the loading of finite enzymatic resources, which can introduce complex interactions between otherwise non-interacting elements of genetic circuits (28). Furthermore, it models the consumption of resources like nucleotides and amino acids, and does so without needing to model elongation processes at the single base or codon resolution. Another feature of the toolbox is its simple user interface, which requires only a few lines of code and allows the user to specify a genetic circuit at the level of promoters and genes, abstracting away all lower level interactions.

Finally, we demonstrate a Markov chain Monte Carlo (MCMC) based multi-stage Bayesian inference procedure for characterizing the toolbox's parameters, and use the characterized models to predict and experimentally validate the behavior of an incoherent feed-forward loop under a variety of experimental conditions.

The rest of this article is organized as follows. In Section 2, we describe the implementation details of the toolbox, including the biochemical equations, and show sample code for creating models of genetic circuits. Section 3 describes inference of the 'core' toolbox parameters, which are parameters associated with transcription, translation and RNA and resource degradation. In Section 4, we demonstrate the predictive capabilities of the toolbox using an incoherent feed-forward loop as an example circuit. In Section 5, we discuss the results and possible future work. Finally, in Section 6, we describe the experimental and computational methods used for data collection and parameter inference.

## 2. Implementation

In this section, we describe the implementation details of `txtlsim` that make it useful for modeling TX-TL experiments.

First, modeling the reactions associated with TX-TL requires explicit accounting of the interactions of DNA and RNA with enzymatic resource species such as RNA polymerases, ribosomes, ribonucleases (RNases), and transcription factors. This quickly increases the complexity of the chemical reaction network being built, due to effects such as resource loading and retroactivity (28). This complexity is further compounded by the need to account for nucleotide and amino acid consumption and degradation. The `txtlsim` toolbox abstracts away the need for the specification of these low-level mechanisms and interactions, allowing the user to specify genetic circuits at the level of genes. Indeed, `txtlsim` comes with a library of parts that can be combined in different combinations to build genetic circuit models. Secondly, it is able to predict the behavior of a genetic circuit, based only upon the characterization of its constituent parts, as discussed in Section 4.

The `txtlsim` toolbox is written using MATLAB Simbiology<sup>®</sup>, enabling its models to be compatible with MATLAB's visualization, simulation and parameter estimation capabilities. Genetic circuits are specified using a set of input strings that specify DNA, small molecules and other miscellaneous species. The toolbox then generates a deterministic mass action kinetics model of this circuit's mechanics in TX-TL. A typical TX-TL model, specified by the user at the resolution of whole DNA, mRNA and protein species, comprises mechanisms for transcription, translation, RNA degradation, transcriptional regulation and the eventual inactivation of TX-TL itself. Other mechanisms, such as linear DNA and protein degradation or *gamS*-mediated protection against nucleases, may also be included (39).

### 2.1 User interface

We highlight several features of `txtlsim`. First, the toolbox requires only a few lines of code to generate a relatively complex chemical reaction network model of TX-TL. For instance, the constitutive expression of green fluorescent protein (GFP) can be modeled using the code shown below.

```
% Set up three Simbiology model objects.
tube1 = txtl_extract('E1'); % extract
tube2 = txtl_buffer('E1'); % buffer
tube3 = txtl_newtube('constitutive_expression');
```

```

% Add DNA to model object by specifying promoter, RBS, CDS,
concentration and type of DNA.
txtl_add_dna(tube3, 'pOR2OR1(50)', 'utr1(20)',...
'deGFP(1200)', 30, 'plasmid'); % reporter DNA

% Combine the three model objects into one.
Model_obj = txtl_combine([tube1, tube2, tube3]);

% Simulate and plot.
simData = txtl_runsim(Model_obj, 14*60*60);
txtl_plot(simData, Model_obj);

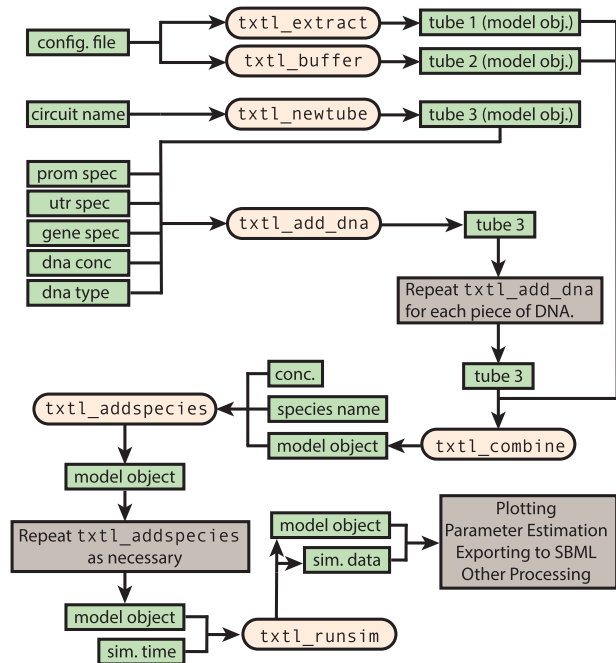
```

The set of commands shown in the snippet above mimic the actual experimental protocol of setting up a TX-TL experiment. The functions `txtl_extract` and `txtl_buffer` access extract and buffer specific parameter configuration files, selected by the input string 'E1' here, to set up two Simbiology® model objects called `tube1` and `tube2`. Users can use these to store parameters for each new extract batch, or transfer parameters corresponding to different batches between laboratories.

Next, the `txtl_newtube` and `txtl_add_dna` commands are used to initialize a new Simbiology® model object and add DNA to this model object, respectively. In its most common use case, the `txtl_add_dna` function allows for specification of a promoter, an untranslated region and a coding sequence to form a transcriptional unit on the specified DNA, along with the concentration of the DNA added, and whether it is a linear fragment or plasmid DNA. For example, in the call to `txtl_add_dna` above, the promoter, ribosome binding site and coding sequence (CDS) are specified by the strings 'pOR2OR1', 'utr1' and 'deGFP', respectively. These strings, each describing a component of the transcriptional unit, are used by `txtl_add_dna` to access a library containing code and parameter files associated with these components. These component files specify the reactions and species associated with each component, and encode interactions with other components. This allows `txtlsim` to automatically link different transcriptional units into a genetic circuit.

The `txtl_combine` command is used to combine the three model objects (`tube1`, `tube2` and `tube3`) into a model object, `Mobj`, which is then simulated using `txtl_runsim`, with the results plotted using `txtl_plot`. The flowchart in [Figure 1](#) depicts the order these commands need to be specified in.

[Figure 2](#) shows the result of the `txtl_plot` command, which is arranged into three panels. The top panel shows the protein species that exist within the TX-TL system. The bottom left panel shows RNA (solid) and DNA (dashed) dynamics. The bottom right panel (normalized to 1) shows the dynamics of enzymatic and consumable resources. The enzymatic resources are ribosomes, RNA polymerases, and RNases; the consumable species in the model are the four nucleotides (ATP and GTP, summed into a so-called species AGTP, and CTP and UTP, summed into CUTP), and amino acids (AA). Each RNA species is plotted as the total concentration of all its forms (bound and unbound). The remaining species are plotted as the concentration of free species (unbound). For example, the concentration of free ribosomes is plotted as the dashed green curve in the bottom right panel. This concentration drops as ribosomes bind to mRNA during the first two hours of the experiment, and then rises as the concentration of mRNA drops and ribosomes unbind.



**Figure 1.** Flowchart of the user-level code. The `txtl_add_dna` command is the main command that is used to specify the DNA to be added to the model. This allows for all the reactions and species associated with that DNA to be set up in the model. The model is contained in a Simbiology® model class object, and is simulated using the `txtl_runsim` command.

The plots in [Figure 2](#) were generated using parameters found by fitting the deGFP and mRNA curves to corresponding data, as described in [Section 4](#).

## 2.2. The modeling framework of the `txtlsim` toolbox

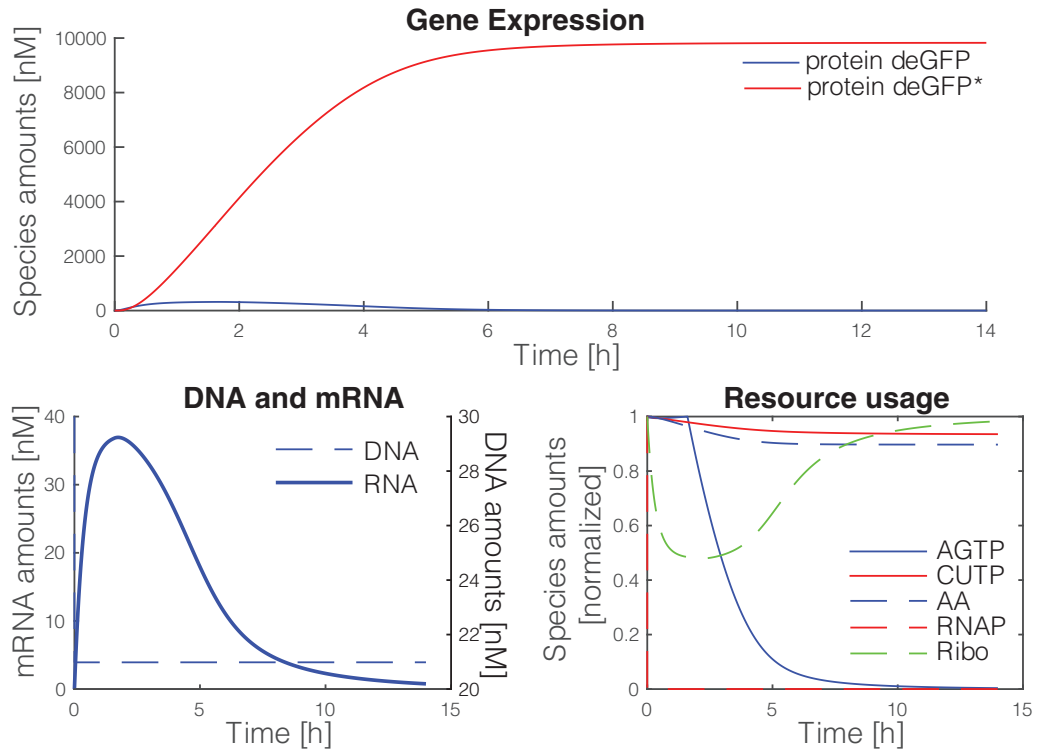
In this section, we describe the typical reaction network generated by `txtlsim` when a transcriptional unit is expressed. More complex networks made out of multiple transcriptional units interacting via transcription factor-mediated regulation are simply iterations of this canonical network, but coupled via enzymatic and consumable resources, and the relevant regulatory interactions. The species in the software toolbox may be divided into five broad categories: DNA, mRNA, proteins, miscellaneous species such as inducers or nucleotides, and the biochemical complexes formed by combining these in defined ways.

The species follow a naming convention, which allows for the automated decision making involved in the reaction network generation. This naming convention is described in [Supplementary Section S3.2](#).

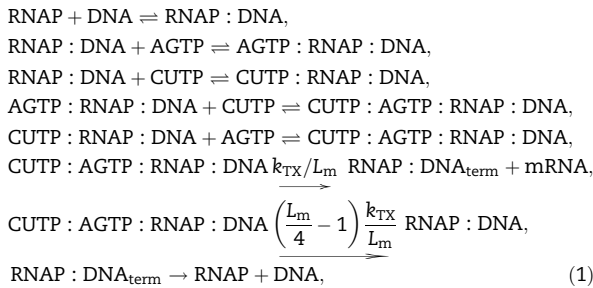
The main processes associated with each transcriptional unit are transcription, translation and RNA degradation ([Supplementary Figure S3](#)). Other processes include DNA and protein degradation, transcription factor mediated activation and repression, and inducer action.

### 2.2.1 Transcription

Transcription is modeled as a four-step process (RNA polymerase binding, nucleotide binding, elongation and termination) using the reactions



**Figure 2.** Standard output of txtsim. Top: Gene-expression profiles for unfolded (deGFP) and folded (deGFP\*) reporter protein. Bottom left: left axis: total mRNA profile, right axis: free DNA profile. Bottom right: Normalized resource loading and consumption  $AGTP = ATP + GTP$ ,  $CUTP = CTP + UTP$ . The mRNA curve for each RNA corresponds to the total concentration of that RNA in all its bound forms (i.e. bound to ribosomes, RNase, etc.). The remaining species (proteins, DNA and resources) correspond to the free (unbound) species concentrations.



where a complex formed by two species, e.g. RNAP and DNA, is denoted as  $RNAP : DNA$ .

The catalytic machinery of transcription is lumped into a single species, denoted by RNAP. It is assumed to encompass RNA polymerases, sigma factors, and other cofactors. Transcription factors are modeled separately, because they are needed for transcriptional regulation, are user defined, and because various distinct transcription factors may exist in a single circuit.

While all four nucleotides (ATP, GTP, CTP and UTP) are used as raw materials for mRNA synthesis, ATP and GTP are also used as a source of energy for translation. Thus, instead of lumping all four nucleotides into a single generic nucleotide species that can both be incorporated into mRNA and be used as an energy source in translation (as was done in, for instance (23)), we lump ATP and GTP into a species AGTP and CTP and UTP into a species CUTP. We note that each molecule of AGTP is defined to

correspond to a molecule of ATP *and* one of GTP (as opposed to “or”, a distinction that has implications for the stoichiometry of the transcription and translation reactions). We denote this definition  $AGTP = ATP + GTP$ . Similarly, we define  $CUTP = CTP + UTP$ .

After the binding of the catalytic and consumable species, the production of mRNA itself is divided into two reactions. The first reaction models mRNA production as a single step, where the bound complex creates an mRNA molecule without modeling elongation along the DNA at the base pair resolution, or modeling polysome formation, where multiple RNAP can bind onto the DNA. This choice to model mRNA production in a single step is made to keep the model relatively simple in light of the fact that at present, sub-transcript resolution mechanisms of regulation (such as attenuator-antisense RNA binding (40)) are not modeled in the toolbox. If a polysome model is needed, the ribosome flow model (41) could be incorporated in future releases of the toolbox.

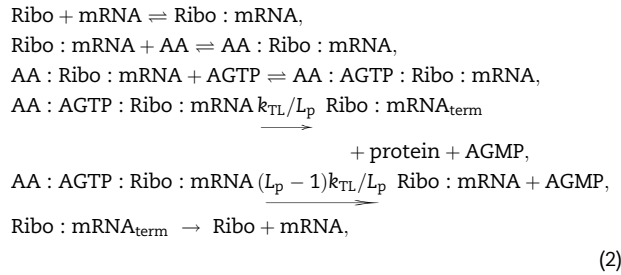
On the other hand, accounting for nucleotide consumption, which is generally done via base-pair resolution models, is accomplished using a ‘consumption’ reaction. This latter reaction uses up AGTP and CUTP without producing mRNA, and is used to balance the consumption of nucleotides with the production of mRNA. We have discussed the consumption reaction in [Supplementary Section S4.1](#).

Finally, at the end of mRNA production, a termination complex  $RNAP : DNA_{term}$  forms, which then dissociates into RNAP and DNA in a separate reaction.



### 2.2.2 Translation

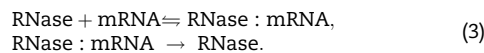
Translation is modeled analogously, with the reactions



where  $L_p$  is the length of the protein in amino acids, and  $k_{\text{TL}}$  is the translation rate. The protein production step is once again modeled in two reactions, a protein production reaction and a resource consumption reaction, and we once again avoid modeling polysome formation or elongation at the single amino-acid resolution. We also use AGTP as an energy source, and its conversion to  $\text{AGMP} \triangleq \text{AMP} + \text{GMP}$  is used to model the four high energy phosphate bonds needed for a single amino acid incorporation (two for tRNA charging and two for elongation). We note that this is an approximation, because in the true model, while  $\text{ATP} \rightarrow \text{AMP} + \text{PPi}$  provides two of the phosphate bonds for the tRNA charging step, the two bonds for elongation are supposed to come from two GTP molecules:  $2 \text{GTP} \rightarrow 2\text{GDP} + 2\text{Pi}$ . These considerations are discussed in greater detail in [Supplementary Section S4.2.2](#).

### 2.2.3 RNA degradation

RNA degradation is mediated by RNases, and is implemented as an enzymatic reaction,



Similar binding and degradation reactions are set up for mRNA in its various bound forms, such as  $\text{Ribo} : \text{mRNA}$ ,  $\text{AA} : \text{Ribo} : \text{mRNA}$ , etc., which result in the degradation of the mRNA and return of the remaining species to the species pool. The full set of these reactions is described in [Supplementary Section S5.2](#).

### 2.2.4 AGTP regeneration system

In batch mode CFPS systems, the ability to express mRNA and proteins diminishes as the experiment proceeds, eventually reaching zero. There are numerous reasons for this exhaustion, including the depletion of nucleotides, the accumulation of inorganic phosphate molecules and the resulting sequestration of magnesium ions, and even the accumulation of magnesium phosphate (42). The depletion of ATP and GTP can be temporarily halted by the use of energy regeneration systems involving a phosphate donor, such as creatine phosphate, phosphoenolpyruvate and 3-phosphoglyceric acid (18). The accumulation of phosphate molecules can be slowed down using, for instance, a pyruvate-based system or dual systems that combine phosphate donor systems with a glucose system (42).

Eventually, as the regeneration systems are themselves depleted, and waste by-products accumulate, transcription and translation slow down, and eventually stop. Noireaux et al. showed that ATP levels were constant for the first 3–6 h after the start of the reaction, independent of whether an eGFP reporter protein was expressed [(43) [Figure 1B](#)], after which they dropped exponentially.

We model this mechanism as a reversible degradation-regeneration reaction involving AGTP and AGMP,

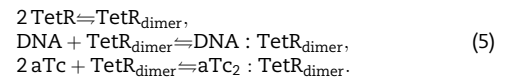


After  $\tau_{\text{ATP}}$  seconds, the reverse (regeneration) reaction stops, leading to pure degradation of the energy resources in the system.

For simplicity, we use this inactivation of the regeneration system and the subsequent depletion of ATP and GTP as the mechanism by which TX-TL depletes its capacity for gene expression. That is, we use this mechanism to model the deactivation of TX-TL due to all effects, including the accumulation of inorganic phosphates or depletion of magnesium ions. The parameters involved in this mechanism ( $\tau_{\text{ATP}}$ , for instance) are estimated using experimental data, and may be thought of as effective parameters capturing the multiple causes of TX-TL inactivation.

### 2.2.5 Other reactions

Additionally, txtlsim can model linear DNA degradation mediated by RecBCD, which is a three subunit enzyme that unwinds DNA, and RecBCD sequestration by the GamS protein (39). The TX-TL system has no innate protein degradation, and degradation of tagged proteins can be mediated by the ClpXP protease (44, 45) and transcription factor-mediated regulation, among other mechanisms. For brevity, we describe just the transcriptional repression and induction reactions here. Repression by the dimerizable protein TetR and its sequestration by the inducer anhydrotetracycline (aTc) are modeled as



## 2.3 Circuit example

The incoherent feed-forward loop (IFFL) is a genetic circuit involving an activator, a repressor and a reporter ([Figure 4A](#)). Owing to the circuit's network topology, the repression of the reporter is delayed with respect to its activation. The IFFL used in this article uses LasR as an activator, expressed constitutively under the control of a pLac promoter. The repressor is TetR, which is under the control of an engineered pLas promoter. We also combined this las-activatable promoter with a tetO operator site to form the combinatorial promoter used to control the deGFP expression (Section 6.4). The code below shows the commands needed to set up the IFFL in txtlsim. This gene circuit will be used to demonstrate the predictive capabilities of the toolbox (Section 4 and [Supplementary Section S6](#)).

```

% Set up three Simbiology model objects.
tube1 = txtl_extract('E1'); % extract
tube2 = txtl_buffer('E1'); % buffer
tube3 = txtl_newtube('las-tet-IFFL');
% Set up LasR activator DNA.
txtl_add_dna(tube3, 'plac(50)', 'utr1(20)', 'LasR(1000)', 2,
'plasmid');
% Set up TetR repressor DNA.
txtl_add_dna(tube3, 'plas(50)', 'utr1(20)', 'TetR(1000)',
0.1, 'plasmid');
% Set up deGFP reporter DNA.
txtl_add_dna(tube3, 'plas_ptet(50)', 'utr1(20)',
'deGFP(1000)', 2, 'plasmid');
% Combine model objects and add inducers.
Model_obj = txtl_combine([tube1, tube2, tube3]);

```

```

txtl_addspecies(Model_obj, 'OC12HSL', 1000);
txtl_addspecies(Model_obj, 'aTc', 1000);
% Simulate and plot.
simData = txtl_runsim(Model_obj, 14*60*60);
txtl_plot(simData, Model_obj);

```

## 2.4 Managing chemical reaction network complexity

In lower-level specifications of reaction networks, where the user must specify each chemical reaction and interaction, such as Simbiology®, Bioscrape (21) or even directly stated ODEs, modeling the IFFL at the level of detail of the txtlsim toolbox would require several tens to over a hundred equations, all of which would need to be manually specified. Furthermore, modifying the network would entail manually updating the relevant equations, a process that is both time consuming and error prone. The txtlsim toolbox, on the other hand, allows the user to specify genetic circuits at a higher level of abstraction, allowing for rapid testing of different designs.

The rationale behind including the above reactions to model circuits is to account for the consumption of the limited pool of nucleotides and amino acids, and the loading of the finite catalytic and regulatory machinery (RNA polymerases, ribosomes, transcription factors, etc.). The consumption and degradation of nucleotides and amino acids is thought to underlie the inactivation of the gene expression capability, and is therefore important to model to capture the full curves of TX–TL experiments. Coupling between different parts of a circuit, via the loading of enzymatic resources (28) or regulatory elements, has been shown to introduce unintended interactions between parts of genetic circuits in both TX–TL and *in vivo* (46). The txtlsim toolbox incorporates these types of effects naturally, since it is built on mass action—as opposed to Michaelis–Menten or Hill—kinetics, allowing for enzymatic loading to be modeled by the explicit formation of complexes and a drop in the concentration of free enzyme. The use of such mass action kinetic models also means that the models are extensible, in the sense that once a species exists, new types of interaction can be added without modifying any of the existing equations—a property that does not hold for Michaelis–Menten or Hill kinetics in general. Finally, models created with txtlsim can be converted into SBML, and may be exported into any other SBML compatible environment for analysis.

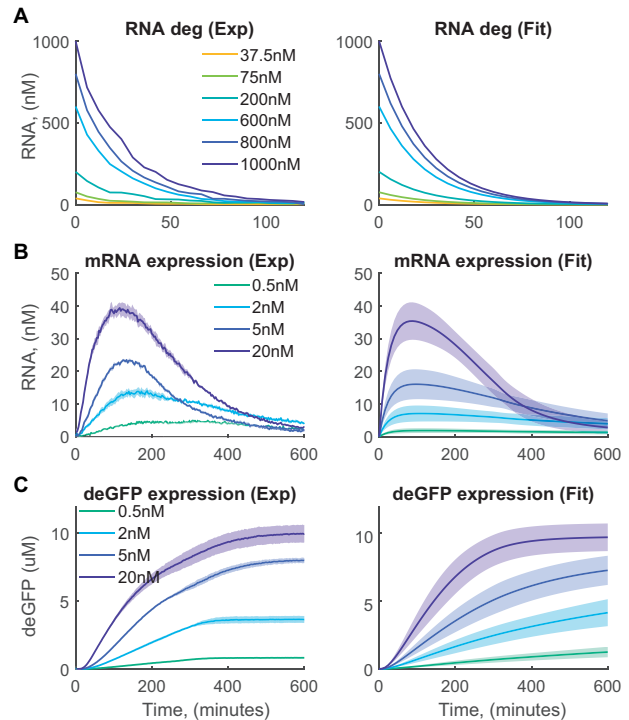
In [Supplementary Section S3.1](#), we describe the software architecture that allows for the automatic generation of these reactions and the interactions between them without the need for the user to specify them explicitly.

## 3. Inferring the parameters associated with the core mechanisms in TX–TL

### 3.1 Overview

Our parameter inference was divided into two stages: the inference of parameters associated with the core mechanisms of the toolbox (transcription, translation, RNA degradation, energy regeneration) and the parameters associated with the parts of genetic circuits (repressors, activators, reporters). The core inference stage is described in this section, while the inference of part specific parameters is described in Section 4.

In the core inference stage, we used trajectories of the degradation of spiked-in RNA and expression of mRNA and protein ([Figure 3](#)) to infer parameters associated with transcription, translation, RNA degradation and AGTP regeneration. We used a Bayesian approach to infer the parameters, so that instead of



**Figure 3.** Estimating the core TX–TL parameters. Experimental data is from [25,47]. Shaded regions indicate standard error over three replicates (left), and model simulations based on the inferred parameters (right). (A) Decay of purified deGFP–MGapt transcripts initiated at six different mRNA concentrations. (B) Transcription kinetics reported by a Malachite Green aptamer. (C) Translation kinetics reported by deGFP. Rows (B) and (C) show four different concentration of plasmid DNA that was added to each TX–TL master mix at the beginning of the experiment.

inferring point estimates for parameter values, we estimated the entire distribution of parameters ([Supplementary Figure S7](#)), which gives a better sense of the correlations between parameter values, and the uncertainty in the parameter estimates. The parameters associated with transcription and translation were, for example, the elongation rates, the dissociation constants associated with the binding of enzymes (RNA polymerases and ribosomes) or resources (nucleotides and amino acids) and the initial concentration of the enzymes. The parameters associated with RNA degradation were the dissociation constant of the binding of the RNase, the forward catalysis rate for the reaction and the initial concentration of the RNase. For the energy regeneration system, the parameters were the forward and reverse rate of AGTP degradation ( $\alpha_{atp}$ ,  $\delta_{atp}$ ) and the time when the AGTP regeneration switches off,  $\tau_{atp}$ . The full list of parameters inferred is shown in [Table 1](#), which is described in greater detail in this section. Here we simply note that the values in the ‘nominal’ parameter value column are reported after transformation using a logarithm of base-*e* (so that numbers between 0 and 1 show up as negative values) and that this table describes multiple independent attempts at inferring the parameters, where some parameters are fixed (marked by asterisks) while the others were inferred. We found that the parameter fitting was computationally tractable for up to approximately 15 free parameters. This is discussed in detail in the rest of this section.

Once the parameters for the core mechanisms have been inferred, the toolbox can be used to simulate the basic mechanisms of TX–TL. The prediction of circuit behavior, in contrast, requires inference of the parameters associated with the parts

**Table 1.** Runs 1–5: different combinations of parameters that were fixed during the core parameter inference

Parameter	$\log_e(\text{nominal})$	Run 1	Run 2	Run 3	Run 4	Run 5
$\text{TX}_{\text{cat}}$	4.9	est	est	est	est	est
$\tau_{\text{atp}}$	9.2	est	est	est	*	*
$\delta_{\text{atp}}$	-9.5	est	est	est	*	*
$\alpha_{\text{atp}}$	-3.9	est	est	*	*	*
$\text{pol}_{\text{Kd}}$	9.5	est	est	est	est	est
$\text{pol}_{\text{F}}$	1.5	*	*	*	*	*
$\text{pol}_{\text{term}}$	3.3	est	est	est	est	*
$n_{\text{Kd1}}$	2.9	est	*	*	*	*
$n_{\text{F1}}$	0	*	*	*	*	*
$n_{\text{Kd2}}$	14.0	est	*	*	*	*
$n_{\text{F2}}$	0	*	*	*	*	*
$\text{RNase}_{\text{Kd}}$	9.2	est	est	est	*	*
$\text{RNase}_{\text{F}}$	0	*	*	*	*	*
$\text{RNase}_{\text{cat}}$	-4.4	est	est	est	est	*
$\text{pol}$	1.4	est	est	est	est	est
$\text{RNase}$	6.5	est	est	est	est	*
$\text{TL}_{\text{cat}}$	0.5	est	est	est	est	est
$\text{GFP}_{\text{mat}}$	-6.1	est	est	*	*	*
$\text{Ribo}_{\text{Kd}}$	11.2	est	est	est	est	est
$\text{Ribo}_{\text{F}}$	-0.2	*	*	*	*	*
$\text{aa}_{\text{Kd}}$	6.6	est	*	*	*	*
$\text{aa}_{\text{F}}$	-0.3	*	*	*	*	*
$\text{TL}_{\text{n,Kd}}$	14.5	est	*	*	*	*
$\text{TL}_{\text{n,F}}$	-1.2	*	*	*	*	*
$\text{Ribo}_{\text{term}}$	5.4	est	est	est	est	*
$\text{Ribo}$	7.3	est	est	est	est	est

Asterisks denote the parameters are fixed to the corresponding values at the nominal point.

of the circuits. In Section 4, we discuss a case study involving the IFFL circuit (described in Section 2.3), where we first follow a multi-stage procedure to infer the parameters associated with the parts of the IFFL, and then use the resulting characterized models to predict the behavior of the IFFL and compare this to corresponding experimental data.

### 3.2 Core parameter inference

In this section, we estimate the parameters associated with the core mechanisms of TX-TL, such as transcription, translation and RNA and energy regeneration (equations (1)–(4)). Our parameter inference is performed in a Bayesian framework, with an MCMC sampler used to construct the posterior distribution of parameters, conditioned on the data and models (See Section 6, Materials and Methods). The experimental data used for estimating the core parameters is from (25, 47), and comprises fluorescence measurements of constitutive protein and mRNA expression, along with the degradation of spiked in mRNA (Figure 3, left column). More details about the data can be found in Supplementary Section S5.1.

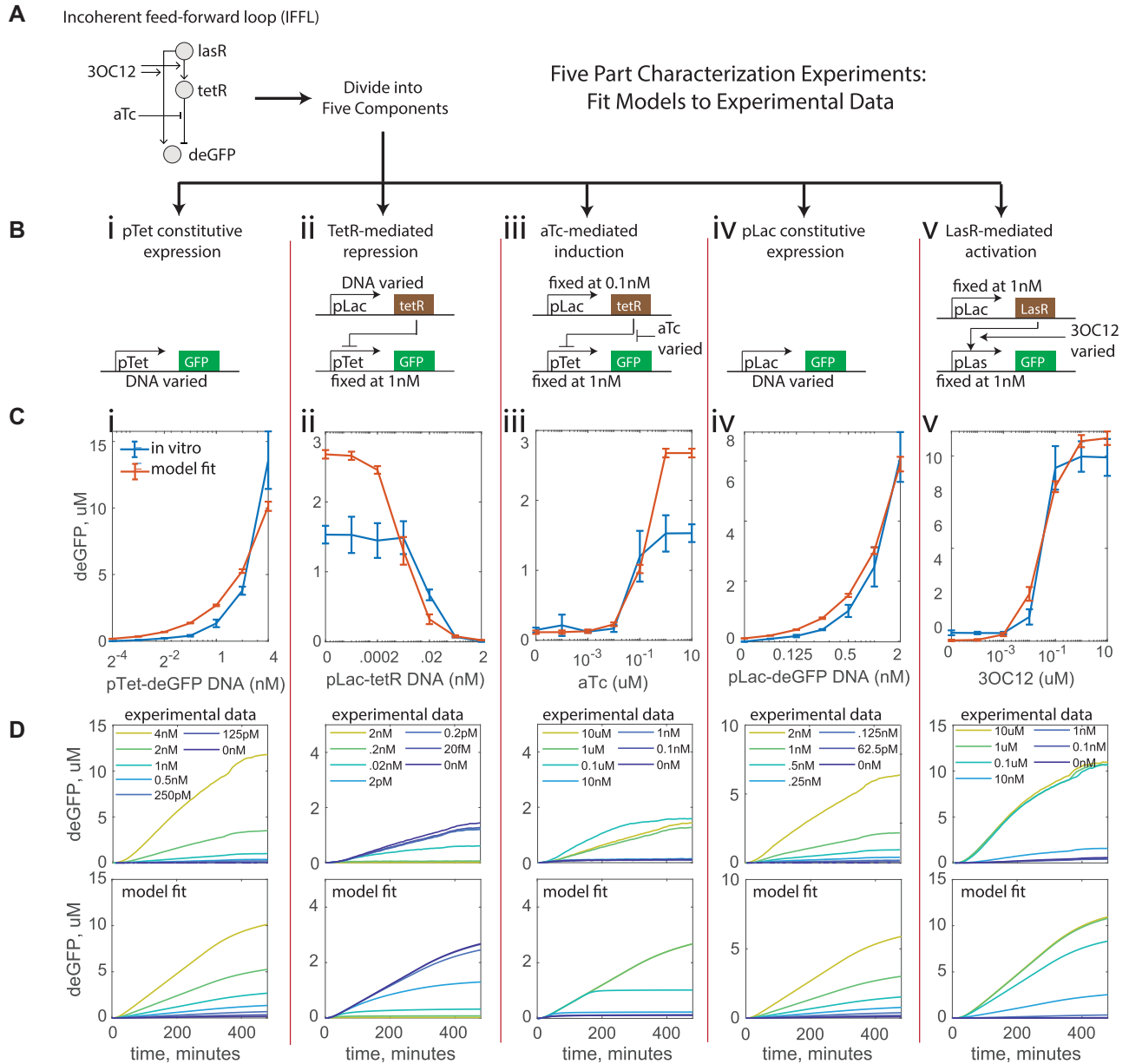
The column on the right of Figure 3 shows the results of fitting txtlsim models to this data. There are a total of 26 parameters in these models, of which several are non-identifiable (48, 49). Briefly, non-identifiability of parameters corresponds to the situation where multiple values of that parameter give the same output behavior of the model, precluding the possibility of uniquely identifying these parameters from the data at hand. Often, parameter non-identifiability is observed when one

parameter’s value compensates for the value of another parameter, so that the overall input-output behavior of the model remains unchanged. Examples of non-identifiability and parameter values trading off against each other can be seen in, for instance, the initial RNA polymerase concentration and the transcriptional elongation rate ( $\text{pol}$ ,  $\text{TX}_{\text{cat}}$ ) during the characterization of the core parameters, Supplementary Figure S7, and the transcriptional and translational elongation rate parameters ( $\text{TX}_{\text{cat}}$ ,  $\text{TL}_{\text{cat}}$ ), or the dissociation constant for the binding of RNA polymerase to the ptet or plac promoters ( $\text{pol}_{\text{Kd,tet}}$ ,  $\text{pol}_{\text{Kd,lac}}$ ) during the IFFL circuit characterization, Supplementary Figure S9. See Supplementary Section S7 for formal definitions of parameter non-identifiability and parameter covariation.

Unlike point estimation methods, MCMC-based sampling gives an estimate of the entire joint parameter distribution, and can be used to gain insight into which parameters are non-identifiable and may therefore be fixed during parameter inference. Examples of such parameters include the forward reaction rates associated with reversible reactions, and in some cases, might even include the dissociation constants themselves. The forward reaction rates set the timescales at which fast reversible reactions reach quasi-steady-state, and were found to be non-identifiable once they were large enough for time-scale separation to be achieved. Some of the dissociation constants, especially those associated with reactions embedded deeper inside the reaction network, also had broad distributions, indicating possible non-identifiability. Examples of reactions with such dissociation constants include those involving nucleotides and amino acids binding to their respective transcription and translation complexes.

Table 1 and Supplementary Figures S4–S6 show five independent parameter inference runs involving models of constitutive expression and RNA degradation, and experimental data from (47). In these runs, different combinations of parameters were fixed to nominal values (reported here after  $\log_e$  transformation, so that negative values correspond to parameter values between 0 and 1; this transformation was performed for technical reasons described in the Methods Section 6.7) as part of an exploratory scheme designed to help elucidate the trade-off between model fidelity and the computational tractability of the parameter inference procedure as a function of the number of free and fixed parameters. In Run 1, only the forward rates in the various reversible reactions were fixed, resulting in a 19-dimensional parameter space, which was too large to be searched efficiently. In Run 2, in addition to the forward reaction rates, the dissociation constants of the reactions involved in the binding of nucleotides and amino acids to the transcription and translation complexes were also fixed, leading to a more manageable search space. In the remaining runs, we successively increased the number of parameters that were fixed, allowing for the parameter space to be searched more efficiently. We found that Runs 3 and 4 were able to fit the data well while still allowing the parameter space to be searched relatively quickly.

The nominal parameter point shown in the first column provided the values to fix the parameters to, and was found using a combination of MCMC and manual tuning (details in Supplementary Section S5). Ultimately, the fits shown in Figure 3 were generated using the parameter fixing profile of Run 3, which had a small enough number of free variables to search over efficiently, while still leaving the model flexible enough to fit the data. Supplementary Figure S7 shows the marginal distributions of the core parameters estimated during Run 3, using data from both (25, 47).



**Figure 4.** Part characterization and parameter fitting for the incoherent feed-forward loop (IFFL, schematic in **A**) with the parameter fixing profile of Stage 2d (**Table 3**). (**B**) Schematics describing the five part characterization experiments used to infer the part-specific parameters. (**C**) Endpoint curves (mean, standard error at 480 min) corresponding to the experimental data (blue,  $n = 3$ ) and corresponding parameter fitting trajectories (orange,  $n = 50$ , sampled from the posterior parameter distribution and simulated). The posterior distributions were generated by fitting the full time-course trajectories to the data (**D**).

#### 4 Case study: experimentally validated prediction of gene circuit behavior

The main role of the `txtlsim` toolbox is to be a simulator for the TX-TL system. This section highlights this role using a case study involving the prediction and experimental validation of the behavior of an IFFL.

First, we collected experimental data involving the components of an IFFL in five different experiments (**Figure 4A, B** and **Table 2**, top half). Second, we estimated the parameters of the building blocks of IFFL, with some of the previously estimated core parameters (**Supplementary Figure S7**) providing the approximate ranges of values to search over, while others providing the values to fix the parameters to. Third, we assembled the

whole gene circuit model of the IFFL using these characterized parts in `txtlsim` (Section 2.3) and simulated it. We also collected experimental data about the dynamical behavior of the whole IFFL in TX-TL under five sets of experimental perturbations (**Figure 5A**, and lower part of **Table 2**), and compared the simulated model with the experimental data (**Figure 5B and C**).

##### 4.1 Part characterization

The part characterization experiments involved collecting data on the isolated behavior of the various components of the IFFL. The experiments we chose to characterize the components were pTet constitutive expression, TetR-mediated repression,



Table 2. IFFL part characterization and circuit behavior prediction experiments (Figures 4 and 5)

Exp. type	experiment	Species varied	Species fixed
Characterization	Constitutive pTet expression	pTet-UTR1-deGFP: 4, 2, 1, 0.5, 0.25, 0.125 and 0.0625 nM	pTet-UTR1-deGFP: 1 nM
	Constitutive pLac expression	pLac-UTR1-deGFP: 2, 1, 0.5, 0.25, 0.125, 0.0625 and 0.0313 nM	pLac-UTR1-TetR: 0.1 nM
	TetR-mediated repression	pLac-UTR1-TetR: 2, 0.2, 0.02, 0.002, $2 \times 10^{-4}$ , $2 \times 10^{-5}$ and 0 nM	pTet-UTR1-deGFP: 1 nM
	aTc-mediated induction	aTc: 10, 1, 0.1, 0.01, 0.001, $1 \times 10^{-4}$ and $1 \times 10^{-5}$ $\mu$ M	pLac-UTR1-LasR: 1 nM
	3OC12HSL-mediated induction	3OC12 at 10, 1, 0.1, 0.01, 0.001, $1 \times 10^{-4}$ and $1 \times 10^{-5}$ $\mu$ M	pLas-UTR1-deGFP: 1 nM
Prediction	3OC12 induction	3OC12 at 10, 1, 0.1, 0.01, 0.001, $1 \times 10^{-4}$ and $1 \times 10^{-5}$ $\mu$ M	Unless in the 'Species Varied' column:
	LasR activation	DNA pLac-UTR1-LasR: 2, 1, 0.5, 0.25, 0.125, 0.0625 and 0.03125 $\mu$ M	pLac-UTR1-LasR: 1 nM
	aTc induction	aTc: 10, 1, 0.1, 0.01, 0.001, $1 \times 10^{-4}$ and $1 \times 10^{-5}$ $\mu$ M	pLas-UTR1-TetR: 0.1 nM
	TetR repression	DNA pLas-UTR1-TetR: 1, 0.1, 0.01, 0.001, $1 \times 10^{-4}$ , $1 \times 10^{-5}$ and 0 $\mu$ M	pLas_tetO-UTR1-deGFP: 1 nM
	Reporter DNA	pLas_tetO-UTR1-deGFP: 4, 2, 1, 0.5, 0.25, 0.125 and 0.0625 $\mu$ M	aTc: 10 $\mu$ M
			3OC13: 1 $\mu$ M

aTc-mediated induction, pLac constitutive expression and 3OC12HSL-mediated induction (Figure 4B, and Table 2, top half).

We used three models to fit the five IFFL characterization data sets shown in Table 2. The constitutive pLac expression and the 3OC12HSL-mediated induction data sets had their own models, while the remaining three data sets: constitutive pTet expression, TetR-mediated repression and aTc-mediated induction only needed a single model, with three different sets of initial conditions accounting for their differences. Model equations and associated reaction rate parameters can be found in Supplementary Section S6.2.

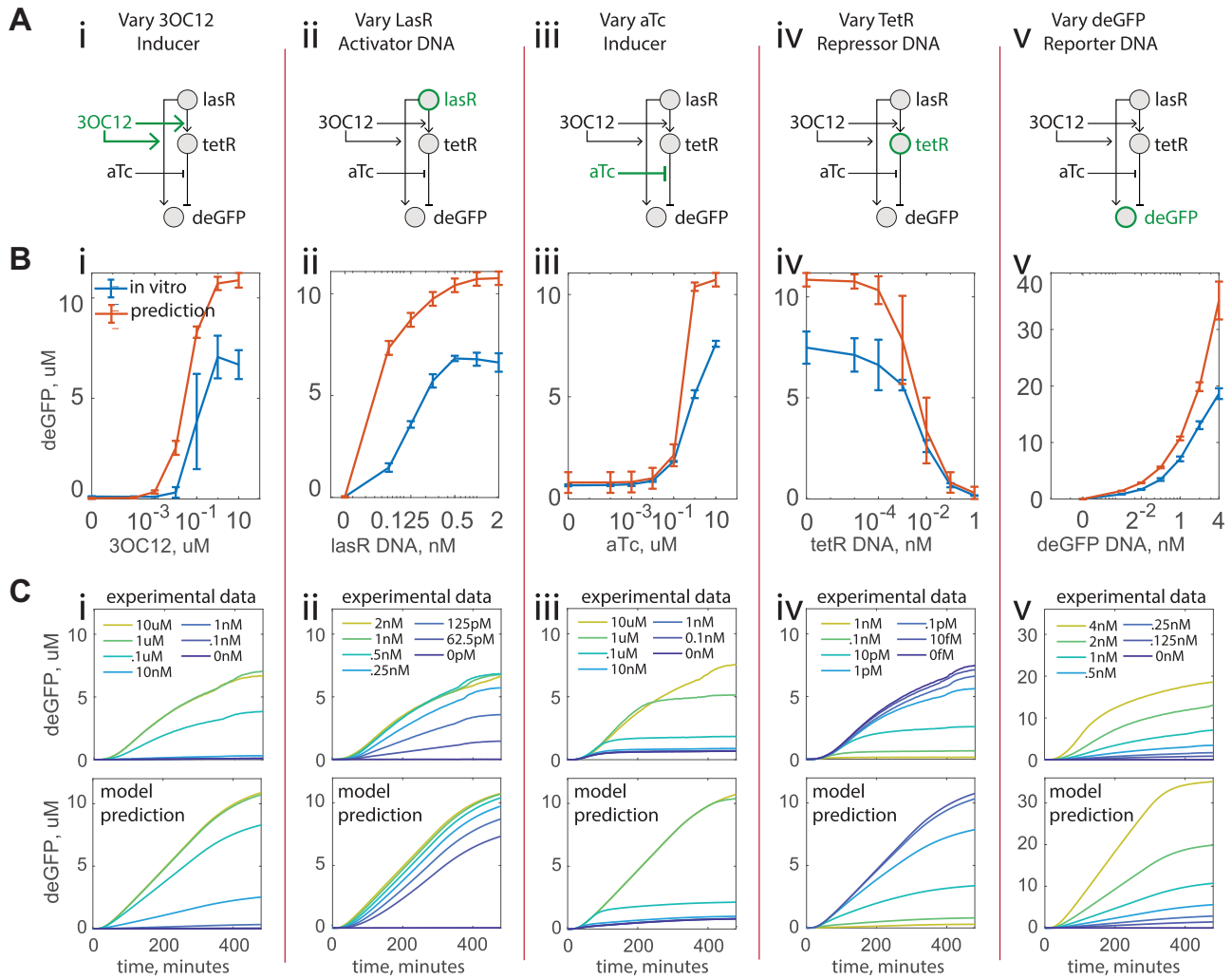
The inference of the parameters associated with the parts of the IFFL was also performed in a consensus Bayesian inference framework (Section 6, Materials and Methods). In total, there were 42 parameters in the model, as shown in the first column of Table 3. The first 26 parameters were those associated with the core mechanisms in the toolbox, and were described in Section 3. The next 8 parameters were associated with the Tet-repression system, and the final 8 parameters were those associated with the Las-activation system.

Table 3 summarizes the multi-stage parameter estimation scheme that we employed to search the parameter space. This approach was needed because the 42 dimensional space was too large to be searched efficiently. All columns except the one labeled 'Init.' describe a parameter inference run in terms of which parameters were estimated, and which ones were fixed. There are four types of symbols in this table: Asterisks, the phrase 'fixed: p' (where p is a numerical parameter value), numerical values and the word 'free'. Asterisks indicate that the value was the same as the value in the previous column. The phrase 'fixed: 1.5' means that that parameter value was fixed to 1.5 at that stage. A numerical value indicates that that parameter was freely estimated at that stage, and that value was picked from the resulting distribution (jointly with any other such fixed values) and fixed in the next stage (therefore, every numerical value is necessarily followed by an asterisk). Finally, the word 'free' means that that parameter was freely estimated at that stage, but no value was picked for fixing in the next stage (and is therefore never followed by an asterisk in the next column). As in the previous section, all parameter values are log-transformed (base-e).

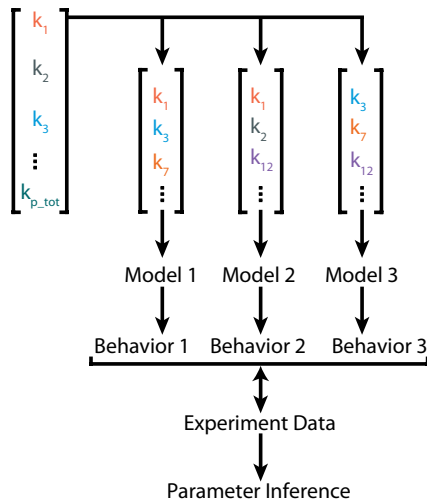
The first 26 (core) parameters are specified by the 'Init.' (initialization) and 'St. 1' (Stage 1) columns of the table. Initialization refers to the initial parameter point found using MCMC and manual parameter tuning in Section 3. The numbers shown in the Stage 1 column correspond to a particular parameter point from the distribution found in Run 3 during the core inference stage.

In Stage 2a, the constitutive pTet expression, TetR-mediated repression and aTc-mediated induction circuits described in Table 2 and Figure 4 were characterized. In addition to the core parameters, there were eight additional Tet-system-associated parameters in the model. Stage 2b had the same models, but with some of the core and Tet-system parameters re-estimated. In Stage 2c, the 3OC12HSL-mediated induction (activation of the pLas promoter by the LasR activator) system was introduced, along with 8 new parameters associated with this model. The forward rates are fixed to a value of zero (in  $\log_e$ -space), and the parameters marked 'free' were estimated. In Stages 2d-f, we estimated different combinations of parameters, and in doing so, explored the trade-off between model fidelity and the computational tractability of the MCMC runs. The characterization results from the probability distribution resulting from Stage 2d are shown in Figure 4, and the pairwise marginal probability

Five Model Validation Experiments: Compare Model Predictions to Experimental Data



**Figure 5.** Model predictions and experimental validation for the incoherent feed-forward loop (IFFL), with parameters found in Stage 2d. (A) Schematics describing the five perturbations of the IFFL that were used for the validation of model predictions. (B) IFFL behavior under these perturbations. The nominal IFFL conditions are described in the main text. Endpoint measurements of mean and standard error for experimental (blue,  $n = 3$ ) and predicted (orange,  $n = 50$ ) values ( $t = 480$  min). (C) Corresponding experimental and model prediction trajectories.



**Figure 6.** Parameter overlap in the consensus parameter inference problem.

distributions are shown in [Supplementary Figure S9](#). The fitting trajectories and parameter distributions resulting from Stage 2f are shown in [Supplementary Figures S10 and S11](#).

**4.2 Model prediction and experimental validation**

Using the part-specific parameters inferred in the previous section, we created *in silico* predictions of the behavior of the whole IFFL, and compared these to corresponding TX-TL data. We measured the behavior of the IFFL under five sets of perturbations away from a nominal IFFL. This nominal condition was: pLac-UTR1-LasR at 1 nM; IPTG at 1 mM (sequestering any native LacI in the extract); the LasR inducer 3OC12HSL at 1  $\mu$ M; the repressor DNA pLas-UTR1-TetR at 0.1 nM; the reporter DNA plas\_tetO-UTR1-deGFP at 1 nM; and the TetR inducer aTc at 10  $\mu$ M.

With this nominal IFFL, we collected the deGFP expression levels under perturbations of 3OC12HSL, the LasR DNA, aTc, the TetR DNA and the deGFP DNA, as listed in lower half of [Table 2](#). The results of these experiments are shown in [Figure 5B and C](#).

**Table 3.** Step-wise parameter inference strategy for IFFL part characterization

Stage Model	Init. const.	St. 1 const.	St. 2a tet	St. 2b tet, lac	St. 2c tet, lac, las	St. 2d tet, lac, las	St. 2e tet, lac, las	St. 2f tet, lac, las
$pol_{F, lac}$ (or p70)	1.5	*	*	*	*	*	*	*
$pol_{Kd, lac}$ (or p70)	9.5	13.6	*	Free	Free	Free	Free	Free
$n_{F1}$	0	*	*	*	*	*	*	*
$n_{Kd1}$	2.9	*	*	*	*	*	*	*
$n_{F2}$	0	*	*	*	*	*	*	*
$n_{Kd2}$	14.0	*	*	*	*	*	*	*
$Ribo_F$	-0.2	*	*	*	*	*	*	*
$aa_F$	-0.3	*	*	*	*	*	*	*
$aa_{Kd}$	6.6	*	*	*	*	*	*	*
$TL_{n,F}$	-1.2	*	*	*	*	*	*	*
$TL_{n,Kd}$	14.5	*	*	*	*	*	*	*
$RNase_F$	0	*	*	*	*	*	*	*
$GFP_{mat}$	-6.1	*	*	*	*	*	*	*
$\alpha_{atp}$	-3.9	*	*	*	*	*	*	*
$TX_{cat}$	4.9	2.4	*	2.3	*	Free	3.1	*
$pol_{term}$	3.3	4.4	*	*	*	Free	Free	Free
$pol$	1.4	1.6	*	Free	Free	Free	Free	Free
$TL_{cat}$	0.5	3.3	*	3.7	*	Free	3.4	*
$Ribo_{Kd}$	11.2	0.05	*	*	*	*	*	*
$Ribo_{term}$	5.4	2.8	*	*	*	Free	Free	Free
$Ribo$	7.3	4.2	*	Free	Free	Free	Free	Free
$RNase_{Kd}$	9.2	15.6	*	*	*	*	*	*
$RNase_{cat}$	-4.4	-0.2	*	*	*	*	*	*
$RNase$	6.5	8.6	*	9.2	*	*	*	*
$\tau_{atp}$	9.2	8.9	*	10.1	*	9.7	*	*
$\delta_{atp}$	-9.5	-9.7	*	*	*	*	*	*
$pol_{F, tet}$			Fixed: 1.5	*	*	*	*	*
$pol_{Kd, tet}$			Free	Free	Free	Free	Free	Free
$rep_{Kd}$			Free	-2.7	*	-0.5	*	*
$rep_F$			1.3	*	*	*	*	*
$aTc_{Kd}$			Free	-6.0	*	Free	-2.0	*
$aTc_F$			1.6	*	*	*	*	*
$dim_{Kd}$			-10.0	*	*	*	*	*
$dim_F$			1.4	*	*	*	*	*
$pol_{F, las}$					Fixed: 0	*	*	*
$pol_{Kd, las}$					Free	Free	Free	Free
$3OC12_{Kd}$					Free	13.0	*	*
$3OC12_F$					Fixed: 0	*	*	*
$pLas - pol_{TF, F}$					Fixed: 0	*	*	*
$pLas - pol_{TF, Kd}$					Free	Free	Free	Free
$pLas_{TF, F}$					Fixed: 0	*	*	*
$pLas_{TF, Kd}$					Free	Free	Free	Free

The blue curves in [Figure 5B](#) show the expression levels of the deGFP under the various perturbations at 480 min (i.e. end-point measurements). The error bars are the standard errors of three technical replicates. [Figure 5C](#), top row, shows the trajectories of the full 480 min of these experiments.

The model prediction trajectories were generated by sampling points from the joint posterior parameter distribution resulting from Stage 2d in [Table 3](#), and simulating the IFFL model at each of these parameter points. The orange curves in [Figure 5B](#) are the mean and standard errors of the deGFP species concentration at 480 min for 50 of these trajectories. Similarly, the predictions shown in the bottom row of [Figure 5C](#) are the mean trajectories of the same 50 trajectories. Similar model predictions from parameter values found in Stage 2f are shown in [Supplementary Figure S12](#).

## 5. Discussion

Synthetic biology is an attempt at incorporating engineering principles into the design of novel biological functions. These principles include the standardization of parts, the principled composition of these parts into larger systems, the use of abstraction layers to decouple phenomena at different scales, and the use of rapid prototyping and predictive modeling to speed up the engineering process.

The rapid prototyping paradigm has been implemented in synthetic biology using cell-free systems like TX-TL, which, among various other uses in biomanufacturing, biosensing, therapeutics and artificial cells, have found utility as a tool for testing genetic circuits *in vitro*. In this article, we have described an *in silico* modeling toolbox called `txtlsim` to accompany TX-TL. This toolbox is built using MATLAB Simbiology<sup>®</sup>, and closely

mimics the species, reactions and chemical reaction network dynamics of TX-TL.

In particular, `txtlsim` has a number of specific features that make it suited as a tool for circuit behavior prediction in TX-TL. First, it explicitly models the usage of enzymes like RNA polymerases and ribosomes using mass action kinetics. Unlike Hill or Michaelis-Menten kinetics, this accounts for the loading of enzymatic or consumable resources once the binding reactions involving these resource species are specified. Accounting for the usage of amino acids and nucleotides usually requires modeling transcription and translation at the single base or amino acid resolution (50, 51), making the genetic circuit models large and unwieldy. We avoid this by modeling these processes in a single-step reaction, while accounting for the resource usage by a separate consumption reaction, with a reaction rate that maintains the correct stoichiometry.

This toolbox also has a library of parts that can be composed to build a spectrum of circuits. As discussed in this study, the circuit models built out of part models that have characterized reaction rate parameters should be predictive of the *in vitro* behavior of the corresponding circuits. We demonstrated this using the IFFL circuit. We first characterized the parameters associated with the core transcription, translation and mRNA degradation mechanics of the toolbox, followed by the parameters associated with the parts of the IFFL. The characterized models were then combined into a model of the IFFL, and we verified that the predicted model behavior matched the corresponding experimental data.

Parameter inference of the individual circuit parts may be performed using MATLAB's in-built optimization tools, or using the MCMC based consensus Bayesian inference tools provided with this toolbox. The Bayesian approach gave estimates of the joint distribution of the part-parameters, conditioned on the data, models and any fixed parameters. Visualizing the (marginalized) probability densities can be used to indicate which parameters might be non-identifiable, or even co-varying with other parameters. This allowed us to fix the values of highly non-identifiable parameters, making otherwise computationally intractable inference problems tractable. Furthermore, the ability to visualize the co-variation (52) between parameters allows us to perform this fixing while respecting the relationships between parameters. These considerations were crucial for performing the inference of both the core model parameters and the circuit-part specific parameters.

We note that while the behavior of the circuit predicted by the model is qualitatively correct, it is often far from the behavior in a quantitative sense. Furthermore, we note that we had to split the parameter inference into multiple stages, where subsets of parameters were estimated at each stage, with parameters from previous stages fixed. The decisions about which parameters to fix, and which ones to estimate at each stage relied on notions of parameter identifiability, but were also very much determined heuristically. This point is related to the fact that, in general, inference and optimization over high dimensional parameter spaces is still very much an art, requiring significant trial and error on the part of the modeler. With regards to both of these points, we note that the emphasis here is on the toolbox itself as a tool for approximate circuit behavior modeling during the initial stages of circuit design, as opposed to the precision of the parameter inference as presented here *per se*, or the presented parameter inference approach as a general prescription.

We also note that the uncertainty in the data, in the parameter estimates, and in the final predictions are intertwined. At

the simplest level, uncertainty propagates forward through the process: the uncertainty in the data (both due to measurement and intrinsic noise) leads to uncertainty in the parameter estimates. This, along with structural non-identifiability of the parameters (49) leads to uncertainty in the prediction, especially when parameters from different models are combined. More complex models of uncertainty propagation can also be considered. For example, when parameters covary with respect to each other, and different combinations of these parameters are taken from different models and experiments (as was done in this article), the final predictions can have uncertainty in them because of parameter inconsistencies (52). We leave a formal investigation of this issue as future work.

Despite the features present in this version of `txtlsim`, there are several directions it can be extended in. Most simply, various new circuit features may be added to the library of parts, such as antisense RNA-mediated transcriptional regulation (11) or integrase-mediated DNA recombination (53). Capabilities for accounting and correcting for extract batch variation (52, 54), studying the identifiability of the circuits, for modeling TX-TL circuits in vesicle (15, 24, 55), paper based (16), microfluidic (17) or clay microgel (56) based modes, or for predicting the *in vivo* behavior from the *in vitro* behavior may also be added, although these tasks present significant research challenges.

As the field of synthetic biology matures, we expect computational modeling to play an increasingly predictive role in the design of genetic circuits, just as it has played in electrical, mechanical and aeronautical engineering.

## 6. Materials and methods

### 6.1 TX-TL extract and buffer preparation

Preparation and execution of TX-TL was according to previously described protocols (13), with a modification of the strain used to ExpressIQ (New England Biolabs).

Briefly, the cells were grown to an  $OD_{600}$  of 1.5, pelleted and washed. They were then lysed using bead beating, and centrifuged to remove the beads and cell debris. The supernatant was incubated at 37°C for 80 min, and then centrifuged to remove endogenous nucleic acids. The supernatant was dialyzed against a pH8.2 buffer containing Mg-glutamate, K-glutamate, Tris and DTT. Finally, the extract was centrifuged and the supernatant was flash-frozen in liquid nitrogen and stored at -80°C.

The buffer had the following components: 9.9 mg/ml protein, 9.5 mM Mg-glutamate, 95 mM K-glutamate, 0.33 mM DTT, 1.5 mM each amino acid except leucine, 1.25 mM leucine, 50 mM HEPES, 1.5 mM ATP and GTP, 0.9 mM CTP and UTP, 0.2 mg/ml tRNA, 0.26 mM CoA, 0.33 mM NAD, 0.75 mM cAMP, 0.068 mM folinic acid, 1 mM spermidine, 30 mM 3-PGA, 2% PEG-8000.

Both the extract and buffer were stored at -80°C in separate tubes, with enough volume for seven reactions per tube.

### 6.2 TX-TL experiment

A 384-well microplate (Nunc) was used for the experiments, and the appropriate concentrations and volumes of DNA and inducers to be used in each reaction were calculated using the spreadsheets provided in (13). The extract and buffer were thawed for 20 min on ice, mixed in the prescribed ratios, and pipetted into each well being used in the microplate, which was also placed on ice. The DNA was then added to each well according to the spreadsheet. All the pipetting was done to avoid bubbles, the plate was sealed, and spun at 4000g for 45 s



at 4°C to distribute the mix evenly at the bottom of the wells and remove any bubbles that might have been introduced. The plate was placed in a Synergy H1/MF microplate reader (Biotek). Settings used for deGFP measurement were: excitation/emission 485 nm/515 nm, at gain 61, measured every 8 min for 8 h.

### 6.3 Plasmid construction

DNA was cloned using standard molecular biology procedures (29, 30, 31) and propagated in a JM109 recA- lacIQ (Zymo Research) strain for purification. Small-scale purifications were done by miniprep (PureYield, Promega) followed by a PCR purification for desalting (QiaQuick, Qiagen). Large-scale purifications were done by midiprep or maxiprep (NucleoBond Xtra Midi or NucleoBond Xtra Maxi, Macherey-Nagel). All plasmids were isolated in stationary phase and sequenced before use.

### 6.4 IFFL part screening in TX-TL

We first characterized a LasR-responsive promoter, pRsaL (Porig), from previously published work by testing its ability to express deGFP in the presence of LasR and 3OC12HSL (Supplementary Figure S8A) (32). While the promoter was responsive to LasR, the  $V_{\max}$  of the promoter was lower than anticipated and the dynamic range was under 6-fold (Supplementary Figure S8B, C). To find a more robust part, we used TX-TL to screen four more promoters taken from the Registry of Standard Biological Parts or from RNAseq data of known responsive elements (33). Out of our screen, P1 showed a 7-fold improved  $V_{\max}$  over Porig and a 29-fold dynamic range (Supplementary Figure S8B, C). We also characterized the basal leakiness of the promoters without LasR present (Supplementary Figure S8D). Finally, we confirmed the result from our extract was generalizable by testing all 5 promoters for  $V_{\max}$  in 11 independently made extracts using the same method (13) but over four *E. coli* strains (Supplementary Figure S8E). We then used P1 for the downstream LasR-responsive promoter due to its high  $V_{\max}$ . To engineer a TetR-repressible, LasR activatable combinatorial promoter, we tried two placements of the tetO operator sites (Supplementary Figure S8F-1) and characterized the response of these variants under aTc activation.

### 6.5 Modeling framework

We assume mass action kinetics, along with a well stirred, constant temperature and volume assumption on our reactions. This allows us to model the chemical equations as a set of ordinary differential equations (ODEs) with the reaction rate parameters and the unknown initial concentrations as the parameters of the system. Formally, we define an experiment  $\mathcal{H} = (S, x_0, \bar{y})$  to be the execution of a system  $S$  under initial conditions  $x_0$  and output measurements  $\bar{y}$ , where the bar denotes the assumption that experimental data reflects the ground truth. With each experiment, we associate an initialized, parametrized ODE model  $M_i$ , with the general structure

$$\begin{aligned} \dot{x} &= f(x, \theta), \\ y &= h(x, \theta), \quad x(0) = x_0(\theta), \end{aligned} \quad (6)$$

where the state vectors, which encode the species concentrations, are  $x, x_0 \in \mathbb{R}_+^n$ , and are assumed to exist for all  $t \geq 0$ . The parameter vector symbol is  $\theta \in \Omega \subseteq \mathbb{R}^p$ , where  $\Omega$  is the set of all possible parameter points of interest. The output is denoted  $y \in \mathbb{R}^S$ , where  $S$  is the number of output variables.

### 6.6 Experiment ensemble and consensus parameter inference

In general, we have multiple experiments informing some common set of parameters, where a given experiment may not inform every parameter, but every parameter is informed by at least one experiment.

Consider an ensemble of experiments  $\{\mathcal{H}_1, \dots, \mathcal{H}_I\}$ , and an associated ensemble of models  $\{M_1, \dots, M_I\}$ , where model  $M_i$  with parameters  $\theta^{(i)} \in \mathbb{R}^p$  captures the evolution of the system in experiment  $\mathcal{H}_i$  under the specified initial conditions.

The different experiments range over different doses (initial conditions), replicates and genetic circuits (systems). The data are collected at a given sampling rate, which discretizes the trajectories. For the  $i$ -th experiment, the discretization of  $\bar{y}^{(i)}$  may be written as a matrix,  $\bar{Y}^{(i)} \in \mathbb{R}^{T^{(i)} \times S^{(i)}}$ , where we note that the number of time points  $T^{(i)}$  and measured output variables  $S^{(i)}$  will in general depend on  $i$ . We concatenate the set of these matrices into a block matrix

$$\bar{Y} \triangleq \left[ \bar{Y}^{(1)} \mid \bar{Y}^{(2)} \mid \dots \mid \bar{Y}^{(I)} \right],$$

with the appropriate padding of zeros when the number of time points differ between experiments.

We collect all the parameters from the ensemble into a *master vector*,  $\Psi \in \mathbb{R}^{p_{\text{tot}}}$ , counting a parameter that appears in multiple  $\theta^{(i)}$ 's only once, so that  $p_{\text{tot}} \leq \sum_{i=1}^I p_i$ . The individual parameter vectors can be related to the master vector via a binary *membership matrix*  $\Gamma \in \{0, 1\}^{p_{\text{tot}} \times I}$ , where the  $(k, j)$ -th entry is 1 if the  $k$ -th element of  $\Psi$  is present in  $\theta^{(j)}$ , and 0 otherwise.

If we group the individual parameter vectors into a matrix  $\Theta \triangleq [\theta^{(1)} \dots \theta^{(I)}]$ , and consider  $\text{diag}(\Psi)$  as the square matrix with the elements of  $\Psi$  on the diagonal, and zeros elsewhere, then the matrix equation  $\Theta = \text{diag}(\Psi) \cdot \Gamma$  relates the master vector to the individual models' parameters via the membership matrix. Consensus parameter inference is described visually in Figure 6.

### 6.7 Bayesian parameter inference

In general, there are two broad classes of methods for estimating parameters of models: the optimization approach and the Bayesian approach. In the optimization approach, one searches for a point in the parameter space that minimizes an *energy function*, which penalizes the difference between the model's behavior and the experimental data. Such methods are usually faster than the Bayesian approach, and work best when the energy function has a single global minimum or, if there are multiple minima, these minima are isolated from one another. In contrast, the Bayesian approach involves estimating the probability distribution of the parameters, given the data. While estimating these distributions can be computationally demanding, this approach has the advantage of giving insight into the joint distributions of the parameters, and in doing so highlighting not just the presence of global or locally isolated minima, but also showing correlations between parameters. As described in Section 3.2, this helps identify the non-identifiability of the parameters, and helps the modeler understand how the data constrains the values of the parameters. Another advantage of the Bayesian approach is for model selection. The probabilities associated with different parameter estimates can be used to assign relative confidence to different parameter values, something that is not possible with the point estimates found via optimization based methods.

We wish to determine the probability density of the parameters, conditioned on the experiments, models, and consensus pattern,

$$p(\Psi \mid \{\mathcal{H}_i\}_{i=1}^I, \{\mathbf{M}_i\}_{i=1}^I, \Gamma), \quad (7)$$

where the data for experiment  $i$  are included by virtue of being an element of the experiment tuple,  $\mathcal{H}_i$ . In what follows, we simplify notation by replacing  $\{\mathcal{H}_i\}_{i=1}^I$  with the data matrix  $\bar{Y}$ , as defined in the previous section. We also drop  $\Gamma$  and  $\{\mathbf{M}_i\}_{i=1}^I$ , though these are assumed. Then, Bayes rule gives

$$p(\Psi \mid \bar{Y}) = \frac{p(\bar{Y} \mid \Psi) \cdot p(\Psi)}{p(\bar{Y})}.$$

We assume the prior to be uninformative (uniform within a hypercube, and zero outside). The likelihood function involves data from all the experiments, measured species, replicates and time points, and is defined as

$$\mathcal{L}(\Psi) \triangleq p(\bar{Y} \mid \Psi) = \frac{1}{(2\pi\sigma^2)^{N/2}} \exp\left(-\frac{\|r(\Psi)\|_2^2}{2\sigma^2}\right).$$

The vector  $r(\Psi)$  is defined as

$$r(\Psi) \triangleq \text{vec}\left(W \odot (\bar{Y} - \hat{Y}(\Psi))\right),$$

with the  $\odot$  symbol denoting the element-wise (Hadamard) product, and  $\text{vec}$  denoting the column-wise reshaping of a matrix into a vector.  $N$  denotes the total number of data points in the output of the models. The model predictions  $\hat{Y}$  depend on the parameter values  $\Psi$ , and are arranged the same way as  $\bar{Y}$ . The matrix  $W$  has the same shape as  $\bar{Y}$ , and contains weights to normalize for the different magnitudes of different output variables. For example, the concentrations of proteins are often much greater than those of mRNA, and fitting performance can be improved greatly by normalizing these data sets using  $W$ . Finally, we follow the standard practice of working with log-probabilities, which improves both the speed and stability of the numerical computations [(34), chapter 22]. We also note that all the parameter values were transformed using  $\log_e$ , so that the nonnegative orthant (where the parameters are naturally restricted to) mapped invertibly to the entire Euclidean space. This allowed the parameter inference to be performed without the nonnegativity constraint, greatly simplifying the problem.

In general, there is no analytical description of the parameter distributions associated with biochemical reaction networks. The standard approach is to use Markov chain Monte Carlo (MCMC) methods to sample from the desired parameter distribution, and construct an approximation to this distribution. This is done by constructing a Markov chain that has this distribution as its stationary distribution. It does this by performing a random walk in parameter space, such that the probability of being in a given region is proportional to the desired probability density in that region (35). We used the MATLAB implementation from (36) of the emcee sampler (37, 38), with some modifications for better walker initialization and handling of numerical ill-conditioning.

## Supplementary Data

Supplementary Data are available at SYN BIO Online.

## Code, data and plasmid availability

The code for this toolbox is available at [https://github.com/vipul\\_singhal02/txtlsim\\_buildacell](https://github.com/vipul_singhal02/txtlsim_buildacell), along with multiple tutorials, and scripts to download the data and generate the figures in both the paper and the [supplementary information](#). [Supplementary section S1](#) gives information on the plasmids and their availability.

## Competing interests

V.S. and Z.A.T. declare no conflicts of interest. R.M.M. and Z.Z.S. declare a conflict of interest: R.M.M. and Z.Z.S. hold ownership in Tierra Biosciences (formerly Synvirobio).

## Author contributions statement

R.M.M. conceived of the txtlsim toolbox, and V.S., Z.A.T. and R.M.M. wrote it. Z.Z.S. conceived of the experiments and collected the data, and V.S. conceived of and performed the parameter inference strategy. V.S. wrote the manuscript. All authors edited it.

## Acknowledgments

We thank Kyle Martin, Shaobin Guo and Clarmyra Hayes for laboratory assistance, and Vincent Noireaux and Michael Jewett for advice on extract preparation methods. We thank Anandh Swaminathan, William Poole and Samuel Clamons for useful discussions about the computational methodology.

## Funding

This material is based upon work supported in part by the Defense Advanced Research Projects Agency (DARPA/MTO) Living Foundries program, contract number HR0011-12-C-0065 (DARPA/CMO); the Air Force Office of Scientific Research, grant number FA9550-14-1-0060; and DARPA SBIR contract W911NF-16-P-0003 (Caltech subcontract from Synvirobio, Inc.), and a Caltech Grubstake Grant.

V.S. was supported by the National Science Scholarship (NSS-PhD), Agency for Science, Technology, and Research (A\*STAR), Singapore. Z.A.T. was supported by the Fulbright Scholarship. Z.Z.S. was supported by a UCLA/Caltech Medical Scientist Training Program fellowship, and a National Defense Science and Engineering Graduate fellowship.

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing official policies, either expressly or implied, of the Defense Advanced Research Projects Agency or the U.S. Government.

## References

1. Elowitz, M.B. and Leibler, S. (2000) A synthetic oscillatory network of transcriptional regulators. *Nature*, 403, 335–338.
2. Gardner, T.S., Cantor, C.R. and Collins, J.J. (2000) Construction of a genetic toggle switch in *Escherichia coli*. *Nature*, 403, 339–342. January
3. Xflow: Fluids simulations to improve real-world performance. Software, Dassault Systems, 2020.
4. Nagel, L.W. and Pederson, D.O. (1973) Spice (simulation program with integrated circuit emphasis). Technical Report

- UCB/ERL M382, EECS Department, University of California, Berkeley, April.
5. Pardee, K., Green, A.A., Takahashi, M.K., Braff, D., Lambert, G., Lee, J.W., Ferrante, T., Ma, D., Donghia, N., Fan, M. et al. (2016) Rapid, low-cost detection of Zika virus using programmable biomolecular components. *Cell*, 165, 1255–1266.
  6. Pardee, K., Slomovic, S., Nguyen, P.Q., Lee, J.W., Donghia, N., Burrill, D., Ferrante, T., McSorley, F.R., Furuta, Y., Vernet, A. et al. (2016) Portable, on-demand biomolecular manufacturing. *Cell*, 167, 248–259.e12.
  7. Tinafar, A., Jaenes, K. and Pardee, K. (2019) Synthetic biology goes cell-free. *BMC Biology*, 17, 64.
  8. Elani, Y., Law, R.V. and Ces, O. (2015) Protein synthesis in artificial cells: using compartmentalisation for spatial organisation in vesicle bioreactors. *Physical Chemistry Chemical Physics*, 17, 15534–15537.
  9. Noireaux, V., Bar-Ziv, R., Godefroy, J., Salman, H. and Libchaber, A. (2005) Toward an artificial cell based on gene expression in vesicles. *Physical Biology*, 2, P1–P8.
  10. Niederholtmeyer, H., Sun, Z.Z., Hori, Y., Yeung, E., Verpoorte, A., Murray, R.M. and Maerkl, S.J. (2015) Rapid cell-free forward engineering of novel genetic ring oscillators. *eLife*, 4, e09771.
  11. Takahashi, M.K., Chappell, J., Hayes, C.A., Sun, Z.Z., Kim, J., Singhal, V., Spring, K.J., Al-Khabouri, S., Fall, C.P., Noireaux, V. et al. (2015) Rapidly characterizing the fast dynamics of RNA genetic circuitry with cell-free transcription–translation (TX-TL) Systems. *ACS Synthetic Biology*, 4, 503–515. doi: 10.1021/sb400206c.
  12. Shimizu, Y., Inoue, A., Tomari, Y., Suzuki, T., Yokogawa, T., Nishikawa, K. and Ueda, T. (2001) Cell-free translation reconstituted with purified components. *Nature Biotechnology*, 19, 751–755.
  13. Sun, Z.Z., Hayes, C.A., Shin, J., Caschera, F., Murray, R.M. and Noireaux, V. (2013) Protocols for implementing an *Escherichia coli* based TX-TL cell-free expression system for synthetic biology. *Journal of Visualized Experiments: JoVE*, 16, e50762.
  14. Shin, J. and Noireaux, V. (January 2012) An *E. coli* cell-free expression toolbox: application to synthetic gene circuits and artificial cells. *ACS Synthetic Biology*, 1, 29–41.
  15. Stano, P. (2019) Gene expression inside liposomes: from early studies to current protocols. *Chemistry – A European Journal*, 25, 7798–7814.
  16. Pardee, K., Green, A.A., Ferrante, T., Cameron, D.E., Daley, Keyser, A., Yin, P. and Collins, J.J. (2014) Paper-based synthetic gene networks. *Cell*, 159, 940–954.
  17. Niederholtmeyer, H., Stepanova, V. and Maerkl, S.J. (2013) Implementation of cell-free biological networks at steady state. *Proceedings of the National Academy of Sciences*, 110, 15985–15990.
  18. Shin, J. and Noireaux, V. (June 2010) Efficient cell-free expression with the endogenous *E. coli* RNA polymerase and sigma factor 70. *Journal of Biological Engineering*, 4, 8.
  19. Kosuri, S., Kelly, J.R. and Endy, D. (December 2007) TABASCO: a single molecule, base-pair resolved gene expression simulator. *BMC Bioinformatics*, 8, 480.
  20. Hoops, S., Sahle, S., Gauges, R., Lee, C., Pahle, J., Simus, N., Singhal, M., Xu, L., Mendes, P. and Kummer, U. (2006) COPASI—a Complex Pathway Simulator. *Bioinformatics*, 22, 3067–3074.
  21. Swaminathan, A., Hsiao, V. and Murray, R.M. (2017) Quantitative modeling of integrase dynamics using a novel python toolbox for parameter inference in synthetic biology. *bioRxiv*, 121152. doi: 10.1101/121152.
  22. Stögbauer, T., Windhager, L., Zimmer, R. and Rädler, J.O. (2012) Experiment and mathematical modeling of gene expression dynamics in a cell-free system. *Integrative Biology*, 4, 494–501.
  23. Mavelli, F., Marangoni, R. and Stano, P. (2015) A simple protein synthesis model for the PURE system operation. *Bulletin of Mathematical Biology*, 77, 1185–1212.
  24. Mavelli, F. and Stano, P. (2015) Experiments on and numerical modeling of the capture and concentration of transcription-translation machinery inside vesicles. *Artificial Life*, 21, 445–463.
  25. Karzbrun, E., Shin, J., Bar-Ziv, R.H. and Noireaux, V. (2011) Coarse-grained dynamics of protein synthesis in a cell-free system. *Physical Review Letters*, 106, 048104.
  26. Moore, S.J., MacDonald, J.T., Wienecke, S., Ishwarbhai, A., Tsipa, A., Aw, R., Kylilis, N., Bell, D.J., McClymont, D.W., Jensen, K. et al. (2018) Rapid acquisition and model-based analysis of cell-free transcription–translation reactions from nonmodel bacteria. *Proceedings of the National Academy of Sciences*, 115, E4340–E4349.
  27. Tuza, Z.A., Singhal, V., Kim, J. and Murray, R.M. (2013) An in silico modeling toolbox for rapid prototyping of circuits in a biomolecular “breadboard” system. *52nd IEEE Conference on Decision and Control*, Firenze, Italy, pp.1404–1410.
  28. Gyorgy, A. and Del Vecchio, D. (2014) Limitations and trade-offs in gene expression due to competition for shared cellular resources. In *53rd IEEE Conference on Decision and Control*, pp. 5431–5436. IEEE.
  29. Gibson, D.G., Young, L., Chuang, R.-Y., Venter, J.C., Hutchison, C.A. and Smith, H.O. (2009) Enzymatic assembly of DNA molecules up to several hundred kilobases. *Nature Methods*, 6, 343–345.
  30. Sarrion-Perdigones, A., Falconi, E.E., Zandalinas, S.I., Juárez, P., Fernández-del-Carmen, A., Granell, A. and Orzaez, D. (2011) GoldenBraid: an iterative cloning system for standardized assembly of reusable genetic modules. *Plos One*, 6, e21622.
  31. Engler, C., Kandzia, R. and Marillonnet, S. (2008) A one pot, one step, precision cloning method with high throughput capability. *Plos One*, 3, e3647.
  32. Tamsir, A., Tabor, J.J. and Voigt, C.A. (January 2011) Robust multicellular computing using genetically encoded NOR gates and chemical ‘wires’. *Nature*, 469, 212–215.
  33. Wurtzel, O., Yoder-Himes, D.R., Han, K., Dandekar, A.A., Edelheit, S., Greenberg, E.P., Sorek, R. and Lory, S. (2012) The single-nucleotide resolution transcriptome of *Pseudomonas aeruginosa* grown in body temperature. *PLoS Pathogens*, 8, e1002945.
  34. MacKay, D.J.C. (2002) *Information Theory, Inference & Learning Algorithms*. Cambridge University Press, New York, NY, USA.
  35. Christensen, N., Meyer, R. and Libson, A. (2004) A Metropolis–Hastings routine for estimating parameters from compact binary inspiral events with laser interferometric gravitational radiation data. *Classical and Quantum Gravity*, 21, 317–330.
  36. Grinstead, A. (2015) Gwmcmc: An implementation of the Goodman and Weare MCMC sampler for MATLAB. GitHub Repository, March.
  37. Foreman-Mackey, D., Hogg, D.W., Lang, D. and Goodman, J. (2013) emcee: the MCMC Hammer. *Publications of the Astronomical Society of the Pacific*, 125, 306–312.
  38. Goodman, J. and Weare, J. (2010) Ensemble samplers with affine invariance. *Communications in Applied Mathematics and Computational Science*, 5, 65–80.
  39. Sun, Z.Z., Yeung, E., Hayes, C.A., Noireaux, V. and Murray, R.M. (June 2014) Linear DNA for rapid prototyping of synthetic biological circuits in an *Escherichia coli* based TX-TL cell-free system. *ACS Synthetic Biology*, 3, 387–397.
  40. Lucks, J.B., Qi, L., Mutalik, V.K., Wang, D. and Arkin, A.P. (May 2011) Versatile RNA-sensing transcriptional regulators for

- engineering genetic networks. *Proceedings of the National Academy of Sciences*, 108, 8617–8622.
41. Reuveni, S., Meilijson, I., Kupiec, M., Rupp, E. and Tuller, T. (2011) Genome-scale analysis of translation elongation with a ribosome flow model. *PLOS Computational Biology*, 7, e1002127.
  42. Kim, H.-C. and Kim, D.-M. (2009) Methods for energizing cell-free protein synthesis. *Journal of Bioscience and Bioengineering*, 108, 1–4.
  43. Noireaux, V., Bar-Ziv, R. and Libchaber, A. (2003) Principles of cell-free genetic circuit assembly. *Proceedings of the National Academy of Sciences*, 100, 12672–12677.
  44. Baker, T.A. and Sauer, R.T. (2012) ClpXP, an ATP-powered unfolding and protein-degradation machine. *Biochimica Et Biophysica Acta*, 1823, 15–28.
  45. Sun, Z.Z., Kim, J., Singhal, V., and Murray, R.M. (2015) Protein degradation in a TX-TL cell-free expression system using ClpXP protease. *bioRxiv*, 019695. doi: 10.1101/019695.
  46. Del Vecchio, D., Ninfa, A.J. and Sontag, E.D. (2008) Modular cell biology: retroactivity and insulation. *Molecular Systems Biology*, 4, 161.
  47. Siegal-Gaskins, D., Tuza, Z.A., Kim, J., Noireaux, V. and Murray, R.M. (2014) Gene circuit performance characterization and resource usage in a cell-free “breadboard”. *ACS Synthetic Biology*, 3, 416–425.
  48. Chis, O.-T., Banga, J.R. and Balsa-Canto, E. (November 2011) Structural identifiability of systems biology models: a critical comparison of methods. *PLoS ONE*, 6, e27755.
  49. Raue, A., Kreutz, C., Maiwald, T., Bachmann, J., Schilling, M., Klingmüller, U. and Timmer, J. (2009) Structural and practical identifiability analysis of partially observed dynamical models by exploiting the profile likelihood. *Bioinformatics*, 25, 1923–1929.
  50. Algar, R., Ellis, T. and Stan, G.-B. (2013) Modelling the burden caused by gene expression: an in silico investigation into the interactions between synthetic gene circuits and their chassis cell. *arXiv:1309.7798 [q-Bio]*, September arXiv: 1309.7798.
  51. Arnold, S., Siemann, M., Scharnweber, K., Werner, M., Baumann, S. and Reuss, M. (March 2001) Kinetic modeling and simulation of in vitro transcription by phage T7 RNA polymerase. *Biotechnology and Bioengineering*, 72, 548–561.
  52. Singhal, V. and Murray, R.M. Transforming Data Across Environments Despite Structural Non-Identifiability. In *2019 American Control Conference (ACC)*, pp. 5639–5646, July 2019.
  53. Hsiao, V., Hori, Y., Rothmund, P.W. and Murray, R.M. (May 2016) A population-based temporal logic gate for timing and recording chemical events. *Molecular Systems Biology*, 12, 869.
  54. Gyorgy, A. and Murray, R.M. (2016) Quantifying resource competition and its effects in the tx-tl system. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pp. 3363–3368.
  55. Altamura, E., Carrara, P., D’Angelo, F., Mavelli, F. and Stano, P. (2018) Extrinsic stochastic factors (solute partition) in gene expression inside lipid vesicles and lipid-stabilized water-in-oil droplets: a review. *Synthetic Biology*, 3, 1. doi: 10.1093/synbio/ysy011.
  56. Jiao, Y., Liu, Y., Luo, D., Huck, W.T.S. and Yang, D. (September 2018) Microfluidic-assisted fabrication of clay microgels for cell-free protein synthesis. *ACS Applied Materials & Interfaces*, 10, 29308–29313.