Data-driven network models for genetic circuits from time-series data with incomplete measurements

Enoch Yeung1, Jongmin Kim2, Ye Yuan3, Jorge Gonçalves4 and Richard M. Murray5,6

1Center for Biological Engineering, Biomolecular Science and Engineering Program, Department of Mechanical Engineering, Center for Control, Dynamical Systems, and Computation, University of California, Santa Barbara, CA, USA
2Department of Life Sciences, POSTECH, Pohang, South Korea
3School of Artificial Intelligence and Automation, Hua Zhong University of Science and Technology, Wuhan, People’s Republic of China
4Systems Biology Research Group, University of Luxembourg, Belvaux, Luxembourg
5Control and Dynamical Systems, and 6Biology and Biological Engineering, California Institute of Technology, Pasadena, CA, USA

Synthetic gene networks are frequently conceptualized and visualized as static graphs. This view of biological programming stands in stark contrast to the transient nature of biomolecular interaction, which is frequently enacted by labile molecules that are often unmeasured. Thus, the network topology and dynamics of synthetic gene networks can be difficult to verify in vivo or in vitro, due to the presence of unmeasured biological states. Here we introduce the dynamical structure function as a new mesoscopic, data-driven class of models to describe gene networks with incomplete measurements of state dynamics. We develop a network reconstruction algorithm and a code base for reconstructing the dynamical structure function from data, to enable discovery and visualization of graphical relationships in a genetic circuit diagram as time-dependent functions rather than static, unknown weights. We prove a theorem, showing that dynamical structure functions can provide a data-driven estimate of the size of crosstalk fluctuations from an idealized model. We illustrate this idea with numerical examples. Finally, we show how data-driven estimation of dynamical structure functions can explain failure modes in two experimentally implemented genetic circuits, a previously reported in vitro genetic circuit and a new E. coli-based transcriptional event detector.

1. Introduction

Synthetic gene networks fulfil diverse roles in realizing circuit logic [1] and timing in living organisms [2], ranging from single-input inverters [3,4] to combinatorial input logic gates [5,6], reduction in DNA synthesis and sequencing costs have made it possible to build increasingly complex genetic circuits with tens to hundreds of components. However, the ability to build novel biological circuitry often outpaces our ability to revise designs or to verify what has been built behaves as intended. As the fields of synthetic and systems biology continue to build and integrate on successes of circuit and device-level complexity to engineer entire genetic systems or pathways, we are consistently seeing failure modes that arise from a lack of modularity, e.g. retroactivity [7–9], and context effects [10].

Likewise, the expansion of CRISPR-based methods for genome editing [11,12] has led to new network control problems in systems biology, e.g. design of
minimal genomes [13], reprogramming regulatory networks for phenotype control [14], and fine-tuned optimization of metabolic pathways. The dynamical system in these design challenges is often a complex network of interacting genes, mRNA, proteins and metabolites. The expansion in DNA sequencing read depth has made it possible to profile individual genes via the transcriptome [15], which combined with quantitative proteomics [16] or metabolomics [17], enables systems-level analysis of network activity. But prohibitive sampling and library preparation costs make obtaining highly time-resolved omics measurements hard. This makes it difficult to infer dynamic network activity at the scale of whole cell models [18] without extensive experimental investment.

Dynamic network models that describe the intricate interactions between every biomolecular state or species are referred to as state-space models. Two key variables that often determine the behaviour of these network models are network topology [19,20] and parametric realization [21,22]. The structure of a network is generally determined by how states in the system causally affect each other [22]; edges in the network are determined by causal dependence while nodes are determined by the states of the system [23].

Identifying the active, dynamic network structure of a biological network is critical, since the hypothesized network architecture of a genetic circuit may be very different from the realized network architecture using a specific collection of parts, sequences, and composition approach [24]. While network structure alone does not determine dynamical behaviour, parametric information is also important in determining what dynamical behaviours a system can achieve [25]. Rather, network structure, or topology, often defines or narrows the possible behaviours a system can achieve. Without any structural constraints, a dynamical system can have arbitrary input–output behaviour. Once network structure is imposed, the set of realizable input–output trajectories can be reduced [26,27]. If the realized network differs significantly from the intended network design, the dynamics of the system may produce faults or glitches when appropriately excited or interrogated [28,29]. Getting the actual network topology to match the intended network motif is thus a key element to robust synthetic biological design.

In systems and synthetic biology, canonical network motifs are broadly accepted as enabling useful dynamical behaviour [27,30]. For example, an incoherent feedforward loop can be used for fold-change detection or adaptation [31,32]. A cyclic network of repressors is associated with either oscillations [33–35] or multi-stability [36] while a dual negative feedback network of two nodes is used as memory module or toggle switch [37]. Still, the active, dynamic network architecture of most realizations of these network motifs in the form of genetic circuits are not formally characterized or catalogued [38]. Systematic, generalizable tools that can discover and model dynamic network topology from data are valuable [1].

Circuit network discovery is, at its core, a network reconstruction problem. Given a desired network motif and a physical system, we need to use measurements of the system to determine if the actual, active network of the system matches the intended design. There have been many network reconstruction algorithms developed for natural and synthetic biological networks [24,39–45]. Historically, the approach to discovering network interactions has involved direct perturbation of biochemical species or components in a network [41,45,46]. Individual nodes are perturbed and depending on if nearby nodes positively or negatively correlate, an activating or repressing relationship between two network nodes can be inferred. In [24], this framework was taken a step further, by showing that the behaviour of direct and indirect links in a benchmark circuit is network topology dependent. This provided a means for using steady-state perturbation data [39,45,47] to estimate network models. Furthermore, these steady-state estimation algorithms have been verified using a benchmark synthetic gene circuit [43]. More recently, the authors in [42] and [40] showed that retroactivity in gene networks can paradoxically confound network predictions that are based wholly on correlation measures. The core issue is that even when measurement data for all biological states are available, causality is difficult to determine from steady-state measurement data affected by back-action or retroactivity in genetic networks [40].

At the single-cell level, the reconstruction problem for biological networks introduces challenges of inferring non-linear stochastic models from noisy data [44,48,49]. In [48], the authors show that by comparing average abundances, molecule lifetimes, covariances and magnitude of step, they can map pairwise interaction dynamics, even when the rest of the system is completely unspecified. The key observation is that assembly stoichiometry of new molecules is fixed, so unbalanced production of linked precursor components will exacerbate imbalance further, resulting in empirically observed large fluctuations. Furthermore, Hilfinger et al. [49] showed that there are statistical invariants for certain kinds of network interactions, which can be used to evaluate and challenge existing hypotheses of stochastic gene interaction. More recently, Wang et al. showed that effective stoichiometric spaces can be used to determine network structure from the covariances of single-cell multiplex data [44]. These studies show that it is possible to infer meaningful structural information about a genetic network, even when only a portion of the network states are observed and the data are fundamentally noisy.

In this paper, we introduce a class of mesoscopic network reconstruction models with adaptable resolution, commensurate with the depth or coverage of the circuit states (or genome) available from fluorimetric, spectometry-based, or sequencing based measurements. Our method is distinct in that we consider the use of high-resolution time-series data, but where only partial measurement of the network’s nodes is feasible. Furthermore, we consider dynamic measurements of bulk culture rather than single cell, where we benefit from the assumptions of high molecular copy number and large reaction volumes [49]. Specifically, we present the dynamical structure function, an abstract model class from linear time-invariant systems theory and show it can be used as a generalized representation of measured interactions between biological or biochemical states. The contributions of this paper are: (1) we show how a dynamical structure function can encode both direct and crosstalk network interactions, by way of theorem and simulated examples, (2) we develop a direct estimation algorithm and code to directly estimate the dynamical structure function, as well as visualization tools to monitor repression and activation in genetic circuits and (3) we demonstrate this theory on two experimental systems: (A) an in vitro genelet repressilator from the synthetic biology literature and (B) a novel transcriptional event detector that we build specifically to illustrate dynamical structure reconstruction.
2. Representing network interactions in partially measured biological networks with dynamical structure functions

In both systems and synthetic biology, discovering (or verifying) the network of a (engineered) biological system is an important problem. However, discovering an entire biochemical reaction network is typically ill-posed, since many network dynamics occur simultaneously from host or environmental context [50], loading effects [51], or unanticipated retroactivity effects [7,51–54]. Even without these effects, the reconstruction problem is equivalent to finding a unique realization for the dynamical system from direct measurements of every state of the system. Unique realization problems are difficult, unless the system of interest has specific structure, e.g. measurement functions of the state that are diffeomorphic [55,56]. On the other hand, there are many inputs that can be used to perturb the system of interest, e.g. silencing RNA [57], genetic knockouts [58] and small chemical inducers [59]. Using these inputs, it is straightforward to reconstruct the system transfer function \( G(s) \) of the system [60,61], where

\[
Y(s) = G(s)U(s),
\]

\( Y(s) \) is a system’s output, \( U(s) \) a system’s input and \( s \) is the Laplace variable. However, the standard system transfer function \( G(s) \) only models closed-loop input-to-output dependencies. It has no direct information about how chemical species within the system are interacting with each other.

There are other kinds of transfer function models which have the potential to describe the network of interactions between chemical species, for example, the dynamical structure function [62]. The dynamical structure function is a representation derived from linear systems theory, and thus can be used to model transients of a genetic circuit around an operating point or even unstable network dynamics diverging from an equilibrium point. It is a more detailed description of network structure than the system transfer function \( G(s) \) since it models causal interactions between measured outputs, in addition to the causal dependencies of outputs on input variables. Most notably, necessary and sufficient conditions for recovery of dynamical network models have been developed and well-studied [22,62–66], but so far no open-source algorithms, code bases, or applications of this theory have been developed directly for synthetic biology.

2.1. Dynamical structure functions

Here we introduce the mathematical formulation of a dynamical structure function, formulated in the context of biological analysis. The dynamical structure function is formulated on the premise that partial, rather than full, state measurements are accessible. The measured states are denoted as \( y \in \mathbb{R}^p \) and the hidden or latent states are denoted as \( x_h \in \mathbb{R}^{n-p} \), where \( n \) is the dimension of the whole state. We then denote the state of the dynamical system

\[
x = [y^T \ x_h^T]^T \in \mathbb{R}^n.
\]

We let \( u \in \mathbb{R}^m \) denote exogenous inputs that can be introduced to influence the dynamics of the state \( x \). With the exception of oscillators, many biochemical reaction networks converge to a steady state. Moreover, it is generally the case that the parameters of biochemical reaction networks are time-invariant [67], so long as macroscopic experimental settings of the system such as temperature, growth media and dissolved oxygen content remain fixed. Therefore, while the model of a biochemical reaction network is of the form

\[
\begin{align*}
\dot{y} &= f_y(y, x_h, u), \\
\dot{x}_h &= f_{x_h}(y, x_h, u), \\
y(0) &= y_0, \\
x_h(0) &= x_{h,0}
\end{align*}
\]

and

\[
y = [I_{p \times p} \ 0] \begin{bmatrix} y \\ x_h \end{bmatrix},
\]

we will suppose that we can linearize the system about either an equilibrium point, a nominal operating point, or even an (unstable or oscillatory) initial condition to extract network dynamics. In biological systems, networks are almost never precisely linear, but we presume to model local fluctuations or perturbations from a target point in the state space. As we will see in the following, this will be enough to extract relevant network information. Proceeding with the linearization, we can write the system in the form

\[
\begin{align*}
\dot{y} &= A_{11} y + A_{12} x_h + B_1 u, \\
\dot{x}_h &= A_{21} y + A_{22} x_h + B_2 u
\end{align*}
\]

and

\[
y = C \begin{bmatrix} y \\ x_h \end{bmatrix},
\]

where

\[
C = [I_{p \times p} \ 0].
\]

We also assume the system’s initial condition of the linearized system is \( x(0) = 0 \), and the entries in \( A \in \mathbb{R}^{n \times n} \) and \( B \in \mathbb{R}^{n \times m} \) are calculated as

\[
\begin{align*}
A_{11} &= \frac{\partial f_y(y, x_h, u)}{\partial y} \bigg|_{y=x, x_h=x_h, u=u}, \\
A_{12} &= \frac{\partial f_y(y, x_h, u)}{\partial x_h} \bigg|_{y=x, x_h=x_h, u=u}, \\
A_{21} &= \frac{\partial f_{x_h}(y, x_h, u)}{\partial y} \bigg|_{y=x, x_h=x_h, u=u}, \\
A_{22} &= \frac{\partial f_{x_h}(y, x_h, u)}{\partial x_h} \bigg|_{y=x, x_h=x_h, u=u}, \\
B_1 &= \frac{\partial f_y(y, x_h, u)}{\partial u} \bigg|_{y=x, x_h=x_h, u=u}, \\
B_2 &= \frac{\partial f_{x_h}(y, x_h, u)}{\partial u} \bigg|_{y=x, x_h=x_h, u=u}.
\end{align*}
\]

Taking Laplace transforms, solving for \( X_h(s) \) and replacing it in \( Y(s) \) we obtain

\[
sY = W(s)Y(s) + V(s)U(s),
\]

where

\[
W(s) = A_{11} + A_{12}(sI - A_{22})^{-1}A_{21},
\]

\[
V(s) = B_1 + A_{12}(sI - A_{22})^{-1}B_2.
\]

Defining \( D(s) = \text{diag}(W(s)) \) and subtracting \( D(s)Y(s) \) from both sides of equation (2.3) and solving for \( Y(s) \) we obtain the following equation:

\[
Y(s) = Q(s)Y(s) + P(s)U(s),
\]

where

\[
Q(s) = (sI - D(s))^{-1}(W(s) - D(s))
\]

is a \( p \times p \) matrix transfer function and

\[
P(s) = (sI - D(s))^{-1}V(s)
\]

is a \( p \times m \) matrix transfer function. In our use of the term transfer function here, we distinguish between the system transfer function \( G(s) \) that describes the closed loop relationship between inputs and outputs and the matrix transfer functions.
Q(s) and P(s) that encode open-loop causal relationships. Each entry $Q_{ij}(s)$ is a transfer function that describes the open-loop causal dependency of measured state $Y_i(s)$ on measured state $Y_j(s)$. Similarly, the transfer function $P_{ij}(s)$ describes the open-loop causal dependency of measured state $Y_i(s)$ on input $U_j(s)$. The matrix pair $(Q(s), P(s))$ is known as the dynamical structure function, where $Q(s)$ is referred to as the network structure and $P(s)$ as the control structure. Finally, we define the time-dependent dynamical structure function $Q(t)$, $P(t)$ as the inverse Laplace transform of the dynamical structure function, where

$$Q(t) = L^{-1}(Q(s))$$

and

$$P(t) = L^{-1}(P(s)).$$

Note that $Q(s)$ is defined as $Q(s) = (sI - D)^{-1}(W - D)$ rather than $Q_{ij}(s) = 1/sW(s)$. This guarantees that the diagonal entries of $Q(s)$ are 0, which implies that any non-zero terms $Q_{ij}(s)$ are strictly proper transfer functions and thus descriptions of open causal interactions among measured $Y_i$ and $Y_j$. This also means that $Q(s)$, defined in this way, is unique and has $p$ fewer transfer functions to identify on its diagonal. This construction of $Q(s)$ and $P(s)$ ultimately ensures identifiability [62] under reasonable assumptions of independent input perturbation [24,39-45]. Furthermore, if $Q(s) = W/s$, then we would face two simultaneous challenges in estimation: (1) disentangling autoregulatory dynamics ($Y_i$ to $Y_i$) from pairwise interactions ($Y_i$ to $Y_j$) and (2) too many unknown parameters in both $Q(s)$ and $P(s)$. Lastly, we find that studying the pairwise interactions $Q_{ij}(s)$ can already elicit important functional information about a genetic network, as illustrated by the next two examples.

### 2.1.1. Example: the dynamical structure function of an idealized incoherent feedforward loop

Consider the following synthetic biology design problem: design and implement an incoherent feedforward loop. Specifically, we consider implementing a feedforward loop using the synthetic parts $p_{Lac}$-LasR-CFP-LVA, $p_{Las}$-TetR-YFP-LVA, and $p_{Las-Tet}$-RFP-LVA and IPTG, $C_6O_9H_{12}$ - HSL, and aTc as inputs (figure 1). We model the protein concentration of LasR-CFP, TetR-YFP, and RFP as $x_1$, $x_2$ and $x_3$, respectively. We denote the corresponding mRNA species for each of these proteins as $m_1$, $m_2$, and $m_3$. A simple model without any loading effects, describing the dynamics of these states can be written as

$$
\begin{align*}
\dot{x}_1 &= \rho_1 m_1 - \frac{C_0 x_1/k_L1}{1 + x_1/k_L1} \\
\dot{x}_2 &= \rho_2 m_2 - \frac{C_0 x_2/k_L2}{1 + x_2/k_L2} \\
\dot{x}_3 &= \rho_3 m_3 - \frac{C_0 x_3/k_L3}{1 + x_3/k_L3} \\
\dot{m}_1 &= \frac{\alpha_1 u_1}{k_M1 + u_1} - \delta_m m_1 \\
\dot{m}_2 &= \frac{\alpha_2 x_1 u_2/k_M2}{1 + x_1 u_2/k_M2} - \delta_m m_2 \\
\dot{m}_3 &= \frac{\alpha_3 x_2 u_3/k_M3}{1 + x_2 u_3/k_M3} - \delta_m m_3
\end{align*}
$$

and

$$y = [I_{3 \times 3} \ 0_{3 \times 3}] [x^T \ \tilde{m}^T]^T,$$

where protein production rates $\rho_1 = 641.4$, $\rho_2 = 585.1$, $\rho_3 = 652.8$ nM s$^{-1}$ [68], mRNA production rates $\alpha_1 = 7.8$, $\alpha_2 = 7.1$, $\alpha_3 = 7.92$ nM s$^{-1}$ [69], degradation Michaelis constants $k_{d1} = k_{d2} = k_{d3} = 200$ nM, input Michaelis constants $k_{M1} = k_{M2} = k_{M3} = 4000$ nM [67], degradation rate constant $C_0 = 1$ nM s$^{-1}$ and mRNA degradation or dilution rate $\delta_m = 10$ nM s$^{-1}$ [70].

Based on the parameters selected above, the system yields a stable equilibrium point $(x, u)$ which becomes the point about which we linearize this example system. The dynamical structure function for this system is derived following the procedure outlined above: first, we take Laplace transforms; second, we eliminate the hidden mRNA states of $x_1$, $x_2$ and $x_3$, namely $m_1$, $m_2$, $m_3$. The network structure $Q(s)$ and control structure $P(s)$ matrix transfer functions evaluate to the following:

\[
\begin{pmatrix}
0 & 0 \\
0 & 0
\end{pmatrix}
\]
The network, with edge weight functions corresponding to the entries of $Q'(s)$, is drawn in figure 1b. Note that if we take $s \in \mathbb{R}_{>0}$, the sign of the entries in $Q'(s)$ coincides with the form of transcriptional regulation implemented by TetR and LasR, respectively. In [71], it was shown that the sign definite properties of entries in $Q[\mathbb{R}_{>0}]$ are useful for reasoning about the monotonicity of interactions between measured outputs and how fundamental limits in system performance relate to network structure.

Let us now consider $Q'(t)$ as defined above. We remark that $y(t) = \int_0^t Q'(\tau)y(\tau - \tau) d\tau$ follows from the equation

$$L^{-1}(Y(s)) = L^{-1}(Q'Y(s) + P'U(s)),$$

whenever $u(t) \equiv 0$ such that $U(s) = 0$. This argument holds in general for any system of the form (2.2). In particular, the entries $Q'(t)$ act as convolution kernels, and taken with the integral, define an operator for mapping $y(t)$ to $y(t)$. $[Q'(t)]_a$ also models the isolated impulse response of $y(t)$ to an impulse $y(t)$, while assuming all other elements are off or set to zero. In this way, $[Q'(t)]_a$ encodes the time-dependent gain of $y(t)$ on $y(t)$ assuming all other nodes in the network are momentarily off. In the case of our example, we can see that the network structure of this incoherent feedforward loop is dynamical, hence our usage of the term dynamical structure function to describe the network structure among the measured chemical species $y(t)$. In this particular case, the time-domain analogue of the dynamical structure (or dynamical structure convolution kernel) is given as

$$Q'(t) = \begin{bmatrix} 0 & 0 & 0 \\ Q_{21}(t) & 0 & 0 \\ Q_{31}(t) & Q_{32}(t) & 0 \end{bmatrix},$$

where

$$Q_{21}(t) = (5.5 \times 10^{-3}) e^{-(8.3\times10^{-4})t} \sinh((6.0 \times 10^{-5})t),$$

$$Q_{31}(t) = (4.1 \times 10^{-3}) e^{-(4.9\times10^{-7})t} \sinh((4.1 \times 10^{-3})t),$$

and

$$Q_{32}(t) = -(9.7 \times 10^{-6}) e^{-(4.9\times10^{-7})t} \sinh((4.1 \times 10^{-3})t).$$

A visualization of each of these impulse kernel functions and their corresponding location in the dynamic adjacency matrix, defined by $Q'(t)$, is given in figure 2. Note how the activating or repressing nature of genetic regulation is encoded by the positive or negative sign of the corresponding kernel response. In addition to uncovering the Boolean network of interactions between biological states, the dynamical network convolution kernel $Q'(t)$ reveals the time scales of response of each network edge, as well as the amplitude and the rate of decay of the gain from the time of impulse. Similar response profiles can be generated for step function inputs, though finite impulse inputs are typically more common in biological networks. Interestingly, the transfer function $G'(s)$ of the system is likewise lower triangular, reflecting the feedforward network topology in the genetic circuit. Specifically, $G'(s)$ has a sparsity structure of the form

$$G'(s) = \begin{bmatrix} G_{21}'(s) & 0 & 0 \\ G_{22}'(s) & G_{23}'(s) & 0 \\ G_{32}'(s) & G_{33}'(s) & \end{bmatrix}.$$

2.1.2. Example: the dynamical structure function of an incoherent feedforward loop with crosstalk

In prototyping a feedforward loop, it is important to anticipate in vivo context effects. We consider the same biocircuit as described in example 2.1.1, except now we specifically consider loading effects frequently neglected in the design process of synthetic biology. First, we note that each gene may be susceptible to loading effects [7]. For each gene in figure 1a, a degradation tag is added, to provide tunability, to the rate of degradation of the protein. Inside the cell, a protease called ClpXP targets these degradation tags and degrades the associated protein. Different tags can be incorporated to modulate
the gain of the degradation process. Furthermore, these degradation tags can be subject to mutagenesis experiments, as a means to modulate tunability.

Tunability of degradation introduces a tradeoff in performance. Since the ClpXP protease is a housekeeping protein expressed to form a common pool of proteases for all genes in the cell, there is a limit to the supply of free ClpXP protein in any instant of the cell’s growth cycle. When there are too many degradation-tagged proteins [72], the overloading of the protein degradation queue can trigger unwanted effects such as stress response. More directly, the competition for scarce proteases can induce coupled dynamics or a virtual or indirect interaction between two genes competing for the same protease pool. Even if the genes were engineered to have no direct transcriptional or translational cross-regulation, the competition for the same protease effectively couples the protein states of both genes. Modifying the above model to account for these type of loading effects yields

\[
\begin{align*}
\dot{x}_1 &= \frac{c_0 x_1}{k_{d1} + x_1/k_{d2} + x_2/k_{d2} + x_3/k_{d3}} - \frac{c_0 x_1}{1 + x_1/k_{d1} + x_2/k_{d2} + x_3/k_{d3}} \\
\dot{x}_2 &= \frac{c_0 x_2}{k_{d2} + x_1/k_{d1} + x_2/k_{d2} + x_3/k_{d3}} - \frac{c_0 x_2}{1 + x_1/k_{d1} + x_2/k_{d2} + x_3/k_{d3}} \\
\dot{x}_3 &= \frac{c_0 x_3}{k_{d3} + x_1/k_{d1} + x_2/k_{d2} + x_3/k_{d3}} - \frac{c_0 x_3}{1 + x_1/k_{d1} + x_2/k_{d2} + x_3/k_{d3}} \\
\dot{m}_1 &= \frac{a_0 x_1}{k_{m1} + u_1} - \delta_m m_1 - \frac{a_0 x_1}{k_{m2} + u_2} m_2 \\
\dot{m}_2 &= \frac{a_0 x_2}{k_{m2} + u_2} m_2 - \delta_m m_2 - \frac{a_0 x_2}{k_{m3} + u_3} m_3 \\
\dot{m}_3 &= \frac{a_0 x_3}{k_{m3} + u_3} m_3 - \delta_m m_3 - \frac{a_0 x_3}{k_{m4} + u_4} m_4 \\
y &= [x_{1,3}, x_{2,3}, x_{3,3}] [x^T, m^T]^T.
\end{align*}
\]

(2.9)

We use the same parametric values as before. Computing the dynamical structure function, we obtain \(Q'(s)\) as

\[
\begin{align*}
&\begin{pmatrix}
  1.6 \times 10^{-5} & 0.041 \\
  0 & 2.2 \times 10^{-7}
\end{pmatrix} \\
&\begin{pmatrix}
  3.8 \times 10^{-4} & 7.4 \times 10^{-4} \\
  3.8 \times 10^{-4} & 4.4 \times 10^{-4}
\end{pmatrix}
\end{align*}
\]

and \(P'(s)\) as

\[
\begin{align*}
&\begin{pmatrix}
  6.7 \times 10^{-7} & 0 & 0 \\
  0 & 244.0 & 0 \\
  0 & 0 & 2.8
\end{pmatrix} \\
&\begin{pmatrix}
  2.1 \times 10^{-7} & 3.2 \times 10^{-7} \\
  3.2 \times 10^{-7} & 2.1 \times 10^{-7}
\end{pmatrix}
\end{align*}
\]

(2.10)

Note that \(Q'(s)\) is no longer lower-triangular, but fully connected. Introducing loading effects creates additional coupling between nodes in the network. If the coupling is significant, the designed network interactions of the incoherent feedforward loop are overcome by the crosstalk network interactions [8,20,51,53,71,73,74]. Thus, the coupling that is introduced into the biochemical reaction network by loading effects is reflected in the structure of \(Q'(s), P'(s)\) (see figure 3).

By contrast, the input–output transfer function \(G'(s)\) of the crosstalk system only characterizes how system outputs causally depend on inputs. When calculated, \(G'(s)\) also is a full matrix like \(Q'(s)\) of sixth-order SISO transfer functions

\[
G'(s) = \begin{bmatrix}
G_{11}'(s) & G_{12}'(s) & G_{13}'(s) \\
G_{21}'(s) & G_{22}'(s) & G_{23}'(s) \\
G_{31}'(s) & G_{32}'(s) & G_{33}'(s)
\end{bmatrix}(s),
\]

but all structural information about how loading effects cause interference among measured system states \(Y(s)\) is mixed with the information about how outputs causally depend on inputs \(U(s)\) in \(G(s)\). An identification algorithm of entries in \(G(s)\) will thus be unable to quantify the size of crosstalk or interference among system states \(Y(s)\).

To what extent can the entries of \((Q(s), P(s))\) be used to quantify the size of crosstalk in a synthetic gene network? The following theorem shows that the dynamical structure function can quantify crosstalk-induced deviation from idealized or ‘designed’ system behaviour.

**Theorem 2.1.** Let \(L\) denote the two-sided Laplace operator. Suppose we have a system model that incorporates the effect of crosstalk:

\[
\begin{align*}
\dot{y}' &= f_0(y', x'_0, u), \quad y'(0) = 0 \\
\dot{x}' &= f_0(y', x'_0, u), \quad x'_0(0) = 0
\end{align*}
\]

(2.11)

where \(y', x'\in\mathbb{R}^n\) is a vector of measured states in the output, \(x'_0\in\mathbb{R}^{n-\rho}\) are the unmeasured states of the system, \(x' = (y', x'_0)\) is the full system state, and \(u\in\mathbb{R}^m\) is a vector of system inputs. Furthermore, suppose we have a idealized system model to simulate system dynamics in the absence of crosstalk:

\[
\begin{align*}
\dot{y} &= f_0(y', x'_0, u), \quad y(0) = 0 \\
\dot{x}' &= f_0(y', x'_0, u), \quad x'_0(0) = 0
\end{align*}
\]

(2.12)

where \(y\in\mathbb{R}^n\) is a vector of the measured states in the output, \(x'_0\in\mathbb{R}^{n-\rho}\) are the unmeasured states of the system, \(x' = (y', x'_0)\) is the full system state, and \(u\in\mathbb{R}^m\) is a vector of system inputs. Let \((Q'(s), P'(s))\) and \((Q(s), P(s))\) denote the respective dynamical structure functions calculated for each linearized system about the origin. Let

\[
\xi(t) = x'(t) - x(t)
\]

denote the deviation of the crosstalk state from the idealized state. Then

\[
\frac{\partial \xi(t)}{\partial Y_j} = Q'_j(s) - Q_j(s) + \frac{\partial}{\partial Y_j} \mathcal{L}(O(x^2)),
\]

and in particular, if

\[
Q'_j(s) = 0,
\]

then

\[
\frac{\partial \xi(t)}{\partial Y_j} = -Q_j(s) + \frac{\partial}{\partial Y_j} \mathcal{L}(O(x^2)),
\]

and can be estimated from input output data \((Y(s), U(s))\).

**Proof.** The proof of this theorem is provided in the electronic supplementary material (theorem C7).

There are two ways to apply this result. First, if one has an idealized or reference dynamical structure model of the circuit, then this can be compared to the dynamical structure model estimated from the data. Second, in the absence of a reference model, the dynamical structure model can be estimated directly from data to observe new edge functions or gain mismatch directly from the data-driven dynamical structure model. In the latter scenario, if the crosstalk interactions are mediated on an existing edge, it will not be possible to separate the magnitude of the crosstalk from the existing (intended) dynamics of a given active edge in the network. However, the discovered
edge dynamics can be compared to the intended behaviour, e.g.
intentional activation or repression over a certain growth phase of
the cells.

When the assumptions of theorem 2.1 are satisfied, the
entries of $Q_0(s)$ can be used to quantify the size of the crosstalk in the system. Example 1 in the electronic supplementary material shows how the $H_2$ gain of entries in $Q_0(s)$ quantify crosstalk due to enzyme loading. Furthermore, we can compute $Q_0(t) = L^{-1}(Q_0(s))$, which gives a simulated response model of how crosstalk gain fluctuates over time to affect $Y_j(t)$ in response to an impulse in $Y_j(t)$. This response model can also be used to estimate the relevant time scales where crosstalk gain is as large as the gain of designed interactions between circuit components.

Knowledge of the active crosstalk in a genetic circuit can be useful for revising closed-loop design or designing biological controllers to compensate for crosstalk effects. As discussed above, discovering the network model of a complex biological network can be useful for design, verification, analysis or control. In the next section, we formally introduce the theoretical conditions under which identification of the dynamical structure function is possible, as well as the formal problem of network reconstruction.

3. Direct estimation algorithm for dynamical structure functions

In this section, we will introduce a direct estimation algorithm for estimating the dynamical structure function. The dynamical structure function is a tuple of matrix transfer functions $(Q(s), P(s))$ and can be directly estimated from experimental or simulation data so long as the data and the conditions of the experiment satisfy the assumptions of the following theorem, originally shown in [62].

\textbf{Theorem 3.1.} Given a $p \times m$ transfer function $Q(s)$, dynamical structure reconstruction is possible from partial structure information if and only if $p - 1$ entries in each column of $[Q(s) P(s)]^*$ are known, to specify linearly independent elements of the nullspace of the matrix function $[G(s) - I]^*$.

\textbf{Proof.} See the proof of theorem 2 in section IIIB of [62].

It follows from this theorem that without additional structural information about the columns of the matrix $[Q(s) P(s)]^*$, it is not possible to identify $Q(s)$, $P(s)$. In synthetic gene circuits with complex internal network interactions encoded by $Q(s)$, we can still solve for the structure and parameters of $Q(s)$, as long as $p - 1$ elements of $P(s)$ are known. For example, targeted gene knockdowns (CRISPRi) or knockouts (engineered genomic mutations) can be engineered so that $P(s)$ is a diagonal matrix transfer function. This guarantees that $p - 1$ elements of each column of $P(s)$ are known, which satisfies the conditions of theorem 3.1. In this set-up, all genes or biological states in the network of interest are (A) monitored by some measurement channel over time and (B) independently perturbed by a diagonal element in $P(s)$. These are necessary and sufficient conditions for reconstruction of $Q(s)$ and $P(s)$ and corroborate the general network reconstruction conditions developed for network reconstruction of full state measurement systems [24,39–45].

Under the above premises, the task is to estimate the diagonal transfer function entries of $P(s)$ and to estimate all off-diagonal entries of $Q(s)$. Recall from the derivation in §2.1 that the diagonal entries of $Q(s)$ are set to 0 by subtracting the diagonal entries of the precursor $W(s)$ transfer function matrix from $W(s)$. Furthermore, by left-multiplying $(sI - D)^{-1}$ with $W(s) + D(s)$, this guarantees that the representation for $Y(s) = Q(s)Y(s) + P(s)U(s)$ yields a unique $Q(s)$ and $P(s)$. The entries of $Q(s)$ and $P(s)$ are all strictly proper rational transfer functions and thus encode causal dynamics.

\textbf{Problem 3.2 (Network reconstruction and estimation).} Given output measurements $Y(s)$ and input measurements $U(s)$ for a system (2.1) satisfying identifiability conditions described in theorem 3.1, the network reconstruction problem...
is to find the dynamical structure function \( Q(s), P(s) \), as defined in (2.6) and (2.7) that satisfies the equation

\[
Y(s) = Q(s)Y(s) + P(s)U(s).
\]

The network estimation problem is to find an estimate dynamical structure function \( \hat{Q}(s), \hat{P}(s) \) to solve the minimization problem

\[
\min_{\hat{Q}, \hat{P}} \| Y(s) - \hat{Q}(s)Y(s) - \hat{P}(s)U(s) \|.
\]

In practice, there are two approaches to solve for \( Q(s) \) and \( P(s) \). The first is to estimate the transfer function \( G(s) \) using a standard transfer function estimation routine, followed by inversion of \( G(s) \) and calculation of the entries of \( \hat{P}(s) \) and subsequently the entries of \( \hat{Q}(s) \). This approach has the drawback of relying on inversion of the matrix transfer function \( G(s) \), which often requires symbolic inversion and is thus prone to numerical instability and scaling issues for larger networks.

The second approach, which we propose here and develop code for, is to identify the dynamical structure function directly from data by writing the model estimation problem in normal form. First, we can denote the discrete-time approximations of \( \hat{Q}(s) \) and \( \hat{P}(s) \) as \( \hat{Q}(z), \hat{P}(z) \), where \( z \) denotes the \( z \)-transform. \( \hat{Q}(z), \hat{P}(z) \) will follow the same identifiability conditions [62].

Specifically, we have that

\[
Y(z) = \hat{Q}(z)Y(z) + \hat{P}(z)U(z),
\]

which given that \( \hat{Q}(z) \) and \( \hat{P}(z) \) share the same denominator, we can multiply the characteristic polynomial of \( \hat{Q}(z) \) on both sides, to obtain

\[
(z^{-n_i} + a_1 z^{-n_i+1} + \ldots + a_{n_i}) Y(z) = \hat{Q}_{\text{num}}(z)Y(z) + \hat{P}_{\text{num}}(z)U(z).
\]

Here, the matrices \( \hat{Q}_{\text{num}}(z), \hat{P}_{\text{num}}(z) \) are matrices that contain the numerator entries of each transfer function in \( \hat{Q}(z) \) and \( \hat{P}(z) \). Namely

\[
\hat{Q}_i(z) = \begin{bmatrix} \hat{Q}_{\text{num}}(z) \end{bmatrix}_i = \frac{[\hat{Q}_{\text{num}}(z)]_i}{(z^{-n_i} + a_1 z^{-n_i+1} + \ldots + a_{n_i})}
\]

\[
\hat{P}_i(z) = \begin{bmatrix} \hat{P}_{\text{num}}(z) \end{bmatrix}_i = \frac{[\hat{P}_{\text{num}}(z)]_i}{(z^{-n_i} + a_1 z^{-n_i+1} + \ldots + a_{n_i})}.
\]

We can express the known quantity \( z^{-n_i}Y(z) \) after inverse \( z \)-transforming to obtain

\[
y[t - n_i] = -a_1 y[t - n_i + 1] - a_2 y[t - n_i + 2] - \ldots - a_{n_i} y[t - n_i] - \Theta_{\text{Qnum}}Y[t-\ldots-1] - \Theta_{\text{Pnum}}U[t-\ldots-1].
\]

where \( Y_{t \ldots t-1} \) denotes a matrix containing time-shifted entries \( y[t - m], y[t - m + 1], y[t - m + 2], \ldots y[t] \) of a time series or time trace \( y[t] \), \( t \in \mathbb{N} \). Similarly, \( U_{t \ldots t-1} \) denotes a matrix containing time-shifted entries \( u[t - m], u[t - m + 1], u[t - m + 2], \ldots u[t] \) of a time series or time trace \( u[t] \), \( t \in \mathbb{N} \). The terms \( \Theta_{\text{Qnum}} \) and \( \Theta_{\text{Pnum}} \) contain the coefficients of the numerator polynomials of off-diagonal elements in \( \hat{Q}(s) \) and the diagonal elements in \( \hat{P}(s) \), respectively.

The above equation can be thus written in normal form as

\[
b = v^T \Theta.
\]

Here \( b \) and \( v \) are dependent on the elements of a single time trace of output measurements \( y[t] \) and input measurements \( u[t] \). This equation can be also stacked for multiple time traces \( y^1[t], y^2[t], \ldots, y^N_{\text{exp}}[t] \) collected from \( N_{\text{exp}} \) different conditions or experimental replicates, or by simply staggering the time horizon, to obtain the stacked normal form equations

\[
B = X^T \Theta
\]

where \( B \) is a stacked matrix of \( b^1, b^2, \ldots, b^{N_{\text{exp}}} \) and \( X \) is a stacked matrix of \( v^1, v^2, \ldots, v^{N_{\text{exp}}} \) of time-series data and \( \Theta \) contains the coefficients determining the characteristic polynomial and the numerators of the dynamical structure function. Once these estimates are obtained, a standard discrete-to-continuous transformation can be used to estimate \( Q(s) \) and \( P(s) \). The steps of our algorithm are summarized in algorithm 1 and the full MATLAB code is provided at the Github repository https://github.com/YeungRepo/NetRecon.

### Algorithm 1. Direct QF estimation algorithm

1. **Input:**\( Y_{\text{Data}}, U_{\text{Data}} \in \mathbb{R}^{N_{\text{exp}} \times N_{\text{exp}}}. \)
2. **Result:** Dynamical structure function \( \hat{Q}(s), \hat{P}(s) \).
3. **Initialize:**
   1. \( n_{\text{dmin}} \) (min. num. of poles),
   2. \( n_{\text{dmax}} \) (max. num. of poles),
   3. \( t_{\text{max}} \) (max. horizon of subsampled time-series trace)
4. for \( n_{\text{ord}} = n_{\text{dmin}}, \ldots, n_{\text{dmax}} \) do
5.   while \( 1 < t_{\text{max}} < N - n_{\text{ord}} - 1 \) do
6.     \( Y_{LH} = [] \) (stacked snapshots of time-series data corresponding to char. polynomial of \( Q(s) \))
7.     \( Y_{RH} = [] \) (stacked snapshots of time-series data corresponding to numerator terms of \( Q(s) \))
8.     \( U_{LH} = [] \) (stacked snapshots of time-series data corresponding to numerator terms of \( P(s) \))
9.     for \( c = 1, \ldots, p \) do
10.        if \( c < p \) then
11.           \( c_{\text{alt}} = \text{mod}(y_{\text{alt}} + c), p) \)
12.        else
13.           \( c_{\text{alt}} = y_{\text{alt}} + c \)
14.        end
15.     end
16.     \( t_{\text{alt}} = N - (r - 1) - n_{\text{ord}} \)
17.     \( t_{\text{t}} = N - (r - 1) \)
18.     \( Y_{LH, \text{append}}(Y[t : t_{\text{alt}}, c_{\text{alt}}, \exp]) \)
19.     \( Y_{RH, \text{append}}(Y[t : t_{\text{t}}, \exp]) \)
20.     end
21.     \( Y_{LH} = \hat{Y}_{LH} \) (stacked of different conditions)
22.     \( Y_{RH} = \hat{Y}_{RH} \) (stacked of different conditions)
23.     \( U_{LH} = \hat{U}_{LH} \) (stacked of different conditions)
24.     \( U_{RH} = \hat{U}_{RH} \) (stacked of different conditions)

### References

[62] Yeung, R.Y., Hui, K.W., Yeung, G.Y., Sze, K.L., 2021. NetRecon: Large-Scale Network Estimation with Subsampled Measurements. *J. R. Soc. Interface*. 18, 20210413.
A couple remarks are in order. First, there are two distinct hyperparameters that require optimization in generating the estimates for $Q(z)$ and $P(z)$: (1) $n_d$, the order of the characteristic polynomial and (2) the selection of the number of subsampled timepoints $h_{\max}$ in given time-series traces. The optimal value of $h_{\max}$ will depend on the dataset, to ensure the condition number of the matrix $X$ must be small. In high-resolution time-series measurements, a small or short subsampled horizon $h_{\max}$ may produce virtually identical data if the transient has a slow rate of change, which can result in ill-conditioning of $X$. We optimize $n_d$, $h_{\max}$ to minimize the $n$-step L1 prediction error across all $N_{\exp}$ experimental samples. In practice, we observe that this criterion, by necessity, guarantees an appropriate selection of $h_{\max}$ as well as a lower optimal $n_d$ value. In general, optimization of $n_d$ is non-convex, as this corresponds to an estimate of the model order. In the examples below, we leverage knowledge about the system to inform an initial estimate for $n_d$ and perform a local optimization. A generalized specification of $n_d$, following the standard calculation of the rank of a Hankel matrix, but generalized for dynamical structure functions, is the subject of future work.

Secondly, at face value, the routine described above may appear similar to a classic transfer function estimation routine, analogous to the tools developed in [60] for MATLAB. The primary difference in algorithm 1 from a standard transfer function estimation procedure for $[Q(z) \quad P(z)]$ is that we must impose structural information about $Q(z)$ and $P(z)$ on the estimation process. This results in a structured system identification problem, which typically is non-convex and difficult to initialize properly. Specifically, the diagonal entries of the estimated $Q(z)$ and the off-diagonal entries of $P(z)$ must be exactly 0, as per the premises of [62]. Again, these structural constraints guarantee a unique representation of the dynamical structure function and identifiability of the model. When using standard transfer function estimation tools (in MATLAB’s System Identification Toolbox), we found repeatedly over thousands of numerical trials that imposing structural constraints as model priors sometimes resulted in (A) models with extremely poor $n$-step prediction capacity (forecasting) or (B) stable transfer function models that were unable to capture transient (divergent) dynamics.

If the dynamics of the system appear to behave like a linear, stable system, MATLAB system identification routines can be adapted to perform reconstruction (e.g. the experimental data for the in vitro transcriptional event detector). However, for many in vivo gene networks, dynamics are nonlinear or respond to perturbations with transients that are unlike asymptotically stable linear dynamics. The latter observation violates the premise of many transfer function estimation routines in the MATLAB System Identification Toolbox. This motivated the development of a direct estimation algorithm, that mirrors the standard estimation of a discrete-time transfer function model, but where structural constraints of $Q(z)$ and $P(z)$ are directly encoded in the formulation of the normal form of equations (lines 19–28, 32 and 36 of algorithm 1).

### 4. The dynamical structure of an in vitro genelet repressilator

We now illustrate the process of data-driven estimation of dynamical structure functions, using experimental data. In this section, we take as a first test case the synthetic genelet repressilator developed by Kim & Winfree [75]. The genelet repressilator consists of three DNA switches that repress one another through indirect sequestration. Specifically, each DNA switch transcribes its mRNA product only when its activator strand binds to complete its T7 RNA polymerase promoter sequence. The RNA product produced from each DNA switch, in turn, acts as an inhibitor to the downstream switch by binding to the downstream switch’s DNA activator molecule. Thus, by sequestering the DNA activator from completing the T7 RNA polymerase promoter region, the mRNA product of the upstream switch inhibits activation of the downstream switch. Figure 4a shows the mechanistic design of the genelet switch.

![Figure 4](https://royalsocietypublishing.org/journal/rsif/article-pdf/18/1/20210413/20210413.pdf)
The genelet switch relies heavily on RNase H to degrade any activator–mRNA inhibitor complexes. Without degradation, the binding of activator to mRNA inhibitor is much faster than unbinding and so sequestration is effectively irreversible. Thus, in order for the repressilator to function properly, RNase H must degrade its target substrates sufficiently fast. If RNase H is saturated with high levels of a particular substrate, this slows the degradation of other substrates, creating a crosstalk interaction between competing DNA–RNA complexes.

By performing network reconstruction on the genelet repressilator, we can determine how much crosstalk exists in the biocircuit. Furthermore, we can validate our dynamical structure reconstruction algorithm in an in vitro setting, by deliberately attenuating one of the components to create a gain imbalance. We can see if the reconstruction process recovers the deliberate imbalance we introduce into the genelet repressilator, event when simply measuring local perturbations of an operating point for a normally oscillatory circuit.

To reconstruct \( Q(s) \) and \( P(s) \), we performed a single experiment with three perturbations applied in series \([24,39–45,62]\). To perturb each switch, we pipetted a small perturbative concentration of DNA inhibitor (a DNA analogue of RNA inhibitor). Since DNA is not degradable in a T7 expression system by RNase H, it effectively acts as a step input since it binds to DNA activator and does not degrade. In this way, our perturbation design ensures sufficiency of excitation and independent perturbation of each activator (and downstream switch), thereby satisfying the identifiability conditions in \([62]\) and the persistence of excitation conditions described in \([60]\). Furthermore, we attenuated the concentration of the third switch \( T_3 \) by 20%, to create a deliberate gain imbalance for evaluating our reconstruction algorithm.

A detailed model of the repressilator can be found in the supplement of \([75]\). Since the derivation is lengthy, it suffices to write the idealized dynamical structure function \( \tilde{Q}(s) \) of this system, corresponding to the detailed model provided in \([75, \text{supplementary §1.6}]\). The structure is obtained by linearizing the system, transforming into the Laplace domain, eliminating hidden variables to obtain the following:

\[
\begin{bmatrix}
0 & 0 & \tilde{Q}_{10}(s) \\
\tilde{Q}_{10}(s) & 0 & 0 \\
0 & \tilde{Q}_{20}(s) & 0
\end{bmatrix}
\]

reflecting the cyclic structure of the system. This represents an idealized model of the system. As stated in theorem 2.1 every entry where \( \tilde{Q}_{ij}(s) = 0 \), the corresponding entry in \( Q(s) \) estimated directly from experimental data will be a crosstalk interaction present in the network. Here \( Q(s) \) is used to denote \( Q'(s) \), the dynamical structure function estimated directly from data.

The experimental data used to fit \( Q(s) \) and \( P(s) \) are plotted in figure 5, along with their respective fits. For each row \( i \) of \( Q(s) \), we use \( Y_i, j \neq i \) and \( U_i \) as inputs and \( Y_i \) as the output for a direct MIMO \( p \times 1 \) transfer function estimation problem. The impulse response for the convolution kernel \( Q(t) \) of the reconstructed \( Q(s) \) is plotted in figure 6.

If we compute the corresponding \( H_\infty \) gain of each entry in \( Q_0(s) \) and scale by the maximum gain, we obtain

\[
\|Q(s)\|_\infty = \begin{bmatrix} 0 & 0.07 & 0.73 \\ 1.0 & 0 & 0.3 \\ 0.053 & 0.17 & 0 \end{bmatrix}.
\]

We see significant crosstalk on the edge \( Q_{23}(s) \) and minor crosstalk from entries \( Q_{13}(s) \) and \( Q_{12}(s) \). This crosstalk need not occur simultaneously, since the \( H_\infty \) gain calculates the worst-case or maximum gain over all possible frequencies. With the exception of \( Q_{23}(s) \), all other crosstalk entries have strictly smaller \( H_\infty \) gain than the designed edge. Examining the impulse response of the convolution kernel confirms these observations; the crosstalk edge \( Q_{23}(s) \) has a larger impulse response than designed edge \( Q_{23}(s) \). The response of output is normalized per the maximum signal gain achieved, using the technique described in \([75]\), in arbitrary fluorescent units (a.f.u.).

As intended in the design of the experiment, our estimated network model shows a gain imbalance between the designed edges \( Q_{23}(s), Q_{13}(s) \) and \( Q_{23}(s) \). It is well known that in order for a repressilator to stably oscillate \([75]\), it needs to have approximately the same gain along each edge in the network. This example verifies that our reconstruction algorithm can identify important functional dynamics of a genetic circuit; especially for debugging purposes. The linearization scheme is valid, so long as we model fluctuations in dynamics from a nominal initial condition, even if the initial condition is not stable or leads to oscillatory dynamics. Our results here illustrate how linearized models can provide insight into local dynamics. In this simple, controlled dataset, we know we can increase the gain of the edge in \( Q_{23}(s) \) by adjusting the binding affinity of the activator DNA with its inhibitor RNA, or by increasing the concentration of the corresponding downstream switch \( T_3 \). Note that this design insight may not be obvious by direct examination of experimental trajectories of each switch in figure 5. As long as we have an idealized network model, we can measure the deviation from that model \( Q'(s) \) in the network model identified from the data \( Q' = Q(s) \) and identify edges or nodes in our network that need tuning.

5. The dynamical structure of an in vivo transcriptional event detector

We now introduce a new transcriptional event detector circuit, one that is designed, built, and constructed for illustrating the use of our dynamical structure estimation algorithm in in vivo circuit design. Event detectors are useful because of their ability to perform temporal logic. Making temporal logic decisions enable applications such as programmed differentiation, where the goal is to perform some operation based on combinatorial and temporal sequences of events that dictate cell fate.

So far there are two demonstrations of temporal logic gates: (1) a temporal logic gate that differentiates start times of two chemical outputs \([76]\) and (2) a molecular counter that counts the number of sequential pulses of inducers \([77]\). Both event detectors use serine integrases to perform irreversible recombination, while \([77]\) demonstrates the use of transcription-based event detecting to perform event counting. The advantage of an integease-based approach is the persistent nature of DNA-based memory. At the same time, the drawback of integease-based event detection is that it is limited to one-time use.

By contrast, transcription based event detectors use proteins instead of DNA to encode a memory state \([77,78]\). The advantage of a transcription-based event detector is that
proteins are labile, since they are diluted through cell growth or can be tagged for degradation. Thus, a transcriptional event detector’s memory state can be reset after some period of time. On the other hand, maintaining protein state over multiple generations is metabolically expensive [51] and the dynamics of the circuit can become sensitive to production and growth phase of the cells. Therefore, a transcriptional event detector biocircuit must be designed with precise timing, balance of production rates, and carefully tuned gain of each transcriptional regulator. This presents a suitable application for our network reconstruction algorithm.

5.1. Designing a transcriptional event detector

We designed our transcriptional event detector to be made of two constitutively expressed relay genes, AraC and LacI, and an internal toggle switch. The two relay genes transmit the arrival of two distinct induction events (arabinose and HSL) to relay output promoters pBAD and pLas, respectively, which drive production of a fluorescent response in two relay promoters. To record these induction events historically, the output of each relay gene is coupled to one of two combinatorial promoters (pBAD-Lac or pLas-Tet) in a toggle switch. Each combinatorial promoter implements NIMPLY logic, e.g. pBAD-Lac (pLas-Tet) expresses TetR (LacI) only when arabinose (HSL) and AraC (LasR) are present and LacI (TetR) is absent. Thus, when one analyte (e.g. arabinose) arrives, it triggers latching of the toggle switch only if the toggle switch is unlatched to begin with or the prior latching protein state has been diluted out. The relay outputs thus transmit the current or recent induction event state while the toggle switch maintains the historical induction event state. Depending on the order of arrival of each inducer, we obtain different biocircuit states. Figure 7 details the genetic elements in the event detector biocircuit and the designed component interaction network.

We can write down an idealized model for the event detector (assuming no crosstalk), assuming first-order degradation and production, with Hill functions encoding the NIMPLY logic of each promoter in the memory module:

\[
\begin{align*}
\dot{x}_1 &= \rho_1 m_1 - \delta_s x_1, \\
\dot{x}_2 &= \rho_2 m_2 - \delta_s x_2, \\
\dot{x}_3 &= \rho_3 m_3 - \delta_s x_3, \\
\dot{x}_4 &= \rho_4 m_4 - \delta_s x_4, \\
\dot{m}_1 &= \frac{k_1 (u_1 + s_1 / K_{M,1})}{1 + u_1 (s_1 / K_{M,1})} - \delta_m m_1, \\
\dot{m}_2 &= \frac{k_2 (u_2 + s_2 / K_{M,2})}{1 + u_2 (s_2 / K_{M,2})} - \delta_m m_2, \\
\dot{m}_3 &= \frac{k_3 (u_3 + s_3 / K_{M,3})}{1 + u_3 (s_3 / K_{M,3})} - \delta_m m_3, \\
\dot{m}_4 &= \frac{k_4 (u_4 + s_4 / K_{M,4})}{1 + u_4 (s_4 / K_{M,4})} - \delta_m m_4 \\
\end{align*}
\]

\[
y = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x^T \\ m^T \end{bmatrix}^T,
\]

where the measured outputs of the system are \(y_i = x_i, \ i = 2, 3\), \(\rho_i\) is the translation rate of \(m_i\) into \(x_i\), \(\delta_s\) is the effective dilution rate of \(x_i, \ i = 1, \ldots, 4\), \(\delta_m\) is the combined dilution and degradation rate of \(m_i, \ i = 1, \ldots, 4\), \(K_{M,i}\) is the Michaelis constant.

Figure 5. Time-series experimental data from in vitro network perturbation experiments of a T7 RNAP genelet repressilator. Three outputs are measured simultaneously, \(y_1, y_2, y_3\) corresponding to DNA switches \(T_{11}, T_{12}, T_{23}\). DNA homologues of the RNA inhibitors in the genelet circuit.
for $u_i$, $k_i$ is the leaky catalytic transcription rate, $k_i$ is the catalytic transcription rate for $m_i$, and $u_1$, $u_2$ are arabinose and HSL, respectively.

Again, the dynamical structure function for this system is calculated by linearizing the system about a nominal initial condition, $(x_0, m_0)$, taking a Laplace transform and solving out the hidden variables $m_1, ..., m_4$. We present a simplified case here, assuming algebraic symmetry of the parameters $k_i = k$, $p_i = p$, $k_{M_i} = k_M$ as it does not qualitatively change the structure of $(Q(s), P(s))$. We obtain

$$Q^s(s) = \begin{bmatrix} 0 & Q_{12}(s) \\ Q_{21}(s) & 0 \end{bmatrix}$$

and

$$P^s(s) = \begin{bmatrix} P_{11}(s) & 0 \\ 0 & P_{22}(s) \end{bmatrix},$$

where $P_{ii}(s) = \rho/(\delta_m + s)(\delta_p + s)$ for $i = 1, 2$ and

$$Q_{12}(s) = \frac{-k\rho(k_1 + u_1/k_M)}{k_M(\delta_m + s)(\delta_p + s)(u_1/k_M + x_3/k_M + 1)^2}$$

and

$$Q_{21}(s) = \frac{-k\rho(k_2 + u_2/k_M)}{k_M(\delta_m + s)(\delta_p + s)(u_2/k_M + x_2/k_M + 1)^2}.$$

In the absence of protein degradation, $Q_{12}(s)$ and $Q_{21}(s)$ can be approximated with first-order SISO transfer functions. These expressions for $Q(s)$ and $P(s)$ are for the idealized dynamical structure function of the alternative system. Notice that $Q_{12}(s)$ and $Q_{21}(s)$ are strictly negative transfer functions, indicating the repression present in an idealized simulation of the event detector circuit. This is the intended dynamical network structure of the event detector, in the absence of all genetic crosstalk or context effects.

Depending on the abundance of transcription factors such as LacI, TetR and AraC, as well as commonly shared transcriptional and translational proteins, the actual dynamical structure function $Q^s(s)$ may not exhibit monotonic repression or may even unveil unwanted interactions. We can investigate these interactions under a range of conditions with dynamical structure estimation.

We constructed a biological implementation of the event detector, using the design specified in figure 7. The logical components containing the relays and the memory module
and relative ordering of two events occurring within a finite time horizon. Similarly, arrival of event type B triggers transient reporter for B (top) and latching of the toggle in an A-dominant state as a memory detector. Arrival of event type A triggers transient reporter for A (bottom) and latching of the toggle in a B-dominant state as a memory detector; a network model of the dynamic graph of the event detector can be reconstructed using dynamical structure function reconstruction experiments.

module of the event detector to latch in the correct state in response to the first input, followed by a challenge to ignore the second input signal while the relays detect and read out the second input signal. The data for both of these in vivo tests are plotted in figure 8b,c.

The event detector showed the correct latching response in all tests at standard maximum induction concentrations of arabinose (1 mM) and working induction concentrations of 1 μM HSL. For example, figure 8c shows that when the event detector is given arabinose followed by HSL, it generates the correct fluorescent response of YFP, with lower expressions level of RFP. Conversely, when we add HSL first, followed by arabinose, RFP signal ramps up immediately beginning as early as 1–2 h after induction while YFP expression is abolished to background levels.

We evaluated the performance of our transcriptional event detector circuit using a temporal logic test. A standard temporal logic experiment for any two-input event detector is to present the inputs, presenting HSL quorum sensing molecule to the network architecture in (b) and latching of the toggle in a B-dominant state as a memory state. (c) A diagram showing the synthetic biocircuit parts used to implement the network architecture in (b). (d) The arabinose and HSL inducers independently perturb distinct elements of the memory module in the event detector; a network model of the dynamic graph of the event detector can be reconstructed using dynamical structure function reconstruction experiments.

Figure 7. (a) Left: We design an event detector to determine the identity and relative ordering of two events $E_1$ and $E_2$ occurring within a finite time horizon. (b) A schematic showing the logic of the circuit for the event detector. Arrival of event type A triggers transient reporter for A (top) and latching of the toggle in an A-dominant state as a memory state. Similarly, arrival of event type B triggers transient reporter for B (bottom) and latching of the toggle in a B-dominant state as a memory state. (c) A diagram showing the synthetic biocircuit parts used to implement the network architecture in (b). (d) The arabinose and HSL inducers independently perturb distinct elements of the memory module in the event detector; a network model of the dynamic graph of the event detector can be reconstructed using dynamical structure function reconstruction experiments.

Figure 8. A plot of data from in vivo plate reader experiments, testing the temporal logic properties of the event detector diagrammed in figure 7. Note that at 1 μM HSL and 1 mM arabinose, the event detector functions properly, expressing different levels of YFP and RFP depending on the order of arrival of arabinose and HSL. At 1 nM HSL and 1 μM arabinose induction concentrations, the temporal logic properties of the event detector are completely abolished.

were encoded on to a plasmid vector with a kanamycin resistance marker and a ColE1 (high copy) replication origin. The fluorescent reporter elements with the relay promoters and readouts for the toggle switch were encoded on a plasmid vector with chloramphenicol resistance and the p15 replication origin.

5.2. Event detector latching experiments

We evaluated the performance of our transcriptional event detector circuit using a temporal logic test. A standard temporal logic experiment for any two-input event detector is to evaluate the effect of varying the order of presentation of two input signals. In one test, we present the first input, arabinose, for 7.5 h, followed by induction of the second input, a homoserine lactone (HSL) quorum sensing molecule to activate the pLas-Tet promoter. In the second test, we swap the order of the inputs, presenting HSL quorum sensing molecule to the event detector for 7.5 h, then present arabinose inducer as a second input. Both tests evaluate the ability of the memory detector to latch in the correct state in response to the first input, followed by a challenge to ignore the second input signal while the relays detect and read out the second input signal. The data for both of these in vivo tests are plotted in figure 8b,c.

The event detector showed the correct latching response in all tests at standard maximum induction concentrations of arabinose (1 mM) and working induction concentrations of 1 μM HSL. For example, figure 8c shows that when the event detector is given arabinose followed by HSL, it generates the correct fluorescent response of YFP, with lower expressions level of RFP. Conversely, when we add HSL first, followed by arabinose, RFP signal ramps up immediately beginning as early as 1–2 h after induction while YFP expression is abolished to background levels.

We tested a variety of combinations of high and low concentrations for arabinose and HSL. When the concentration of HSL was decreased to 1 nM, we observed consistent leaks in the memory module in either the YFP channel or the RFP channel. Decreasing arabinose down to 1 μM still allows for latching of high YFP expression, but in the presence of 1 μM HSL, any arabinose latching is reversed by HSL induction (data not plotted). Conversely, when we attenuate HSL induction to 1 nM, HSL does not prevent arabinose from reversing a HSL latch on the memory module; see figure 8b. This leak is significant enough in the 1 nM HSL induction level that the difference in signal between the arabinose–HSL induction
scenario versus the HSL–arabinose induction scenario vanished. This temporal logic response profile is evident of a glitch in the event detector circuit that occurs at lower HSL and arabinose concentrations.

5.3. Network reconstruction experiments to debug circuit failure

We conducted 4 in vivo network reconstruction experiments (2 inducers versus 2 concentrations), recording time-series data of the memory module relay elements, YFP and RFP. The memory module is designed using two hybrid promoters, so from a design standpoint, verification of the memory module was most critical. The arabinose inducer targets the pAra-Lac promoter, while the HSL inducer targets the pLas-Tet promoter (see electronic supplementary material for sequences).

As shown in the model (5.1) of the event detector, the actual event detector we constructed exhibits nonlinear response. However, for any one parametric concentration regime, e.g. at a fixed arabinose or HSL concentration, the response of the system behaves similar to that of a linear system. Thus, we estimated a dynamical structure function for both conditions of the reconstruction experiment. The one-step accuracies in fitting dynamical structure models to the low gain condition (1 μM arabinose, 1 mM HSL) and high gain condition (1 mM arabinose and 1 μM HSL) were 99.996% and 99.995%, respectively.

As in the case of the genelet repressilator, we can plot a dynamical network graph for the in vivo event detector to understand how the memory module components labelled by YFP and RFP, representing TetR and LacI, respectively, interact with each other. A movie visualizing the dynamics of the edges of the graph is available for download (see electronic supplementary material). Each edge represents the convolution kernel response of the edge to an impulse applied to that input. All responses are superimposed to form a dynamical graph. Snapshots of the graph are plotted in figure 10, while time-lapse responses of the weights of each edge are plotted in figure 9. Again as with the repressilator, we can see that the regulatory nature of edges in the event detector’s memory module manifests as two edges with negative or positive values indicating repression or activation, respectively.

The reconstructed network of our transcriptional event detector reveals the functional relationship between states in the circuit at different concentration regimes. At lower concentrations of arabinose and HSL, the reconstructed transcriptional event detector network reveals functional cause of failed circuit latching. Both edges in the memory module did not repress their target promoters as intended, while the pLas-Tet promoter appears to enact a much higher gain of activated expression from HSL induction than does the activated expression of the pAra-Lac promoter in response to arabinose.

In the high gain setting, where arabinose is induced at 1 mM and HSL is induced at 1 μM, we see that the memory module exhibits the proper mutually repressing motif characteristic of the genetic toggle switch up after the arrival of the HSL inducer. The repression in both edges steps up their gain as t approaches 4 h, which is roughly the time when we see a plateauing of production in the RFP signal in figure 8c. From our reconstruction model, we can see that the edges are well balanced at the higher concentration of inducers. At the low gain of inducers, the network is completely inactive, even though the genetic sequence of the circuit is the same. This example shows that our network verification algorithm can be used to determine the conditions, or the performance envelope, under which the circuit is functioning properly. Even though the underlying model of our system is a linear approximation to a nonlinear system, we can test the system at multiple initial conditions, operating points, or equilibria, to quantify network behaviour of the system locally. Taken in aggregate, these can provide a

![Figure 9](image-url)  
**Figure 9.** Impulse response of the estimated convolution kernel Q(t) matrix when the event detector biocircuit is induced with (a) 1 nM HSL and 1 μM arabinose or (b) 1 μM HSL and 1 mM arabinose. Q(t) is estimated directly from experimental data, transformed into the frequency domain, and simulated in time as a function of hours from arrival time of an inducer input.

![Figure 10](image-url)  
**Figure 10.** A visualization of the impulse response of the estimated convolution kernel Q(t) matrix when the event detector biocircuit is induced with low (a) versus high (b) concentrations of arabinose and HSL inducer. The width of edges in this graph coincide with the magnitude of the impulse response, while colouring is red if the sign of the impulse response for a given edge is negative (repression) and green if the given edge is positive (activation).
6. Conclusion

The dynamical structure function models the dependencies among measured states. It is a flexible representation of network structure that naturally adapts to the constraints imposed by experimental measurement. Since identifiability conditions of the dynamical structure function have been well characterized, appropriate experimental design can ensure that the process of network reconstruction produces a sensible answer.

In this work, we introduced a network reconstruction algorithm and a code base for reconstructing the dynamical structure function from data, to enable discovery and visualization of graphical relationships in a genetic circuit diagram as time-dependent functions rather than static, unknown weights. We proved a theorem, showing that dynamical structure functions can provide a data-driven estimate of the size of crosstalk fluctuations from an idealized model. We then illustrated these findings with numerical examples. Next, we used an in vitro genetic circuit, deliberately tuned with gain imbalance, to validate our algorithm on experimental data. Finally, we built a new E. coli based transcriptional event detector and showed how estimation of the dynamical structure reveals active and inactive network states, depending on inducer concentration. These results show how the dynamical structure function characterizes the operational or active network. They also provide a route for future study of relationships between environmental parameters, active network dynamics, and biocircuit performance.

7. Experimental methods

All plasmids were constructed using either Golden Gate assembly [79] or Gibson isothermal assembly [80] in E. coli. Plasmids were sequence verified in JM109 cloning strains and transformed into the strain MG1655ΔlacI, provided as a courtesy by R. J. Krom and J. J. Collins. The event detector was transformed into the strain MG1655. Plasmids were sequence verified in JM109 cloning strains and transformed into the strain MG1655ΔlacI, provided as a courtesy by R. J. Krom and J. J. Collins. The event detector was transformed into the strain MG1655.

Data accessibility. All data files, network reconstruction code and visualization scripts can be obtained from the GitHub repository https://github.com/YeungRepo/NetworkRecon.

Authors’ contributions. E.Y. wrote the paper. E.Y. J.G., Y.Y., J.K. and R.M.M. edited drafts of the paper. E.Y. and J.K. designed and carried out experiments and processed experimental data. E.Y. performed analysis and modelling. J.G. and R.M.M. secured research funding. R.M.M. supervised the research process.

Competing interests. We declare we have no competing interests.

Funding. This work was supported by the Engineering and Physical Sciences Research Council, the Luxembourg National Research Foundation, Air Force Office of Scientific Research, grant no. FA9550-14-1-0060, the Defense Advanced Research Projects Agency, grant nos. HR0011-12-C-0065 and FA8750-19-2-0502, the Army Research Office Young Investigator Program, grant no. W911NF-20-1-0165, the National Science Foundation, grant no. 1317291, and the John and Ursula Kanel Charitable Foundation.

Acknowledgements. We would like to acknowledge Sean Warnick, Shara Balakrishnan, vipul Singh and Anandh Swaminathan for insightful conversations on network reconstruction algorithms. We would like to especially thank Shara Balakrishnan for editorial comments through the writing process. We would like to thank and acknowledge Zachary Sun, Victoria Hsiao, Ophelia Venturelli, Charmyna Hayes, Emmanuel de los Santos and Joe Meyerowitz for guidance with experimental techniques.

Appendix A. Supplementary information

A.1. Experimental methods for circuit preparation, assembly and testing

A.1.1. The repressilator genelet circuit

The DNA sequences for the T31, T12, T23 switch were obtained as a gift from the Winfree lab, mirroring the design identically of the repressilator genelet circuit used in [75]. Oligonucleotides were ordered with functionalized fluorophores or quenchers, corresponding to the original design of the genetic repressor. DNA sequences were suspended in Tris-EDTA buffer for primary stock storage, while all genelet switches T12, T31, T23 added at concentrations of 75 nM, 75 nM and 60 nM, respectively to match previous tuning experiments to balance the repressilator, with 7.5 mM working concentration of mono-NTP solution, 24 mM MgCl2, and 1× T7 expression system buffer.

DNA analogues of RNA inhibitors were added to seques- ter DNA activator signal from the switches as an effective step input perturbation to each node. The switches produced a RNA signal that was designed to interfere with formation of a complete promoter region of the next downstream switch in the repressilator circuit. Adding DNA served as a step perturbation to the corresponding switch. Each DNA moiety added thus had the effect of an activator. Activator DNA molecules A1, A2 and A3, each containing Iowa Black quencher were added at 75 nM, 80 nM and 75 nM working concentration at 20 min from the onset of the reaction, to determine the maximum range of quenching. At 58 min, we added 0.7 μl of pyrophosphatase, 3 μl of T7 RNA Polymerase and 2.2 μl of RNase H to achieve identical working concentrations to those described in [75].

A.2. The transcriptional event detector circuit

The transcriptional event detector circuit, as illustrated in figure 7 in the main text, is composed of four distinct gene expression cassettes that define the regulatory logic of the circuit and four distinct gene expression cassettes that generate the fluorescent reporter elements of the circuit. Each gene cassette defines a transcriptional unit, with a promoter element, an...
RBS, a coding sequence, and a terminator sequence. Each gene cassette was cloned using a 5 part Golden Gate assembly, with a type II BsaI restriction enzyme and overhang sequences from [81,82]. Each assembled gene cassette was cloned in JM109 E. coli cloning strains and sequence verified at Eurofins Genomic, by Sanger sequencing. Assembled plasmids were engineered to enable a second stage Golden Gate assembly, using the BsaI type II restriction enzyme, and assembled to either (1) form a master regulatory logic plasmid (pEY15K), comprised of four distinct gene expression cassette driving transcription factor or allosteric response or (2) form a master reporter plasmid comprised of four distinct reporter elements (pEY14C).

Both stage 2 assembled regulatory logic and reporter plasmids were sequence verified using Sanger sequencing (Eurofins Genomics) and transformed into MG1655ΔlacI (a gift from the Collins laboratory). The sequences for all individual plasmids and the circuit plasmids are listed in table 1.

Appendix B. Sequences of genetic circuit components

The sequences for all genetic components and circuits for the event detector circuit are listed in table 1. All genelet repressors are identical to the sequences used and listed in [75]. All ribosome binding site (RBS) sequences were derived from the bicistronic design (BCD) ribosome binding site library [83], while all terminator sequences were drawn from the synthetic terminator library characterized in [84].

Appendix C. Quantifying crosstalk in biochemical reaction networks

A common way that crosstalk arises in biochemical reaction networks is when species compete for commonly shared enzymes. When this occurs, the sequestration of an enzyme by one competing species makes the enzyme less accessible to other competing species. For example, when two mRNA are competing for a single ribosome, the binding of one mRNA to the ribosome during translation makes it less accessible to other mRNA. At the core of any such crosstalk is a sudden increase in the dependency of one biochemical state on another. Though enzyme loading may be a common source of crosstalk, such interactions can be modelled at a higher level of abstraction, namely how the dynamics of a given state are affected by the concentration fluctuations of other states.

Nerely every synthetic gene network implements causal dependencies among states. Often, these ‘designed’ interactions take the form of transcription factor binding, sense–anti-sense mRNA regulation, and sequestration events. In practice, every physical system exhibits trajectories that are a mixture of the consequences of both interaction types: designed and crosstalk interactions. Throughout the course of this paper, we will denote the physical system of interest in our models as

\[
\begin{align*}
\dot{y} &= f_y(y, x_h, u), \\
\dot{x}_h &= f_{x_h}(y, x_h, u), \\
\end{align*}
\]

and

\[
\left[ \begin{array}{c} y \\ x_h \end{array} \right] = \left[ I_{阶段} \quad 0 \right] \left[ \begin{array}{c} y \\ x_h \end{array} \right].
\]

(C1)

To quantify crosstalk in such systems, we can compare the dynamics of system (C 1) against the dynamics of a reference or alternative system that is free of crosstalk. Such a reference system will still retain the desired interaction dynamics and reflects the idealized model often used to design a synthetic gene network, e.g. the feed forward loop model in example 2.1.1. Moreover, it can represent the desired behaviour of the system in a regime where the magnitude of crosstalk effects are supposed to be minimal or engineered in such a way that they are suppressed [7]. We write the reference system as

\[
\begin{align*}
y &= f_y(y, x_h, u), \\
\dot{y}(0) &= y_0, \\
\dot{x}_h &= f_{x_h}(y, x_h, u), \\
x_h(0) &= x_{h,0} \\
\end{align*}
\]

and

\[
\left[ \begin{array}{c} y \\ x_h \end{array} \right] = \left[ I_{阶段} \quad 0 \right] \left[ \begin{array}{c} y \\ x_h \end{array} \right].
\]

(C2)

Remark C.1. For the comparison between the alternative and crosstalk system to be fair, it is important that (C4) satisfies internal equivalence [85]. Specifically, we will suppose that any parameters or dynamics unassociated with crosstalk, e.g. interaction dynamics, catalytic reactions, or anabolic reactions with no loading effects, are held fixed. Thus, as we compare the behaviour of both systems, any differences in the hidden state \(x_0\) or output \(y\) dynamics are purely due to effects of crosstalk.

With the definition of an alternative system in place, it becomes possible to reason about the size of crosstalk, by comparing the dynamics of both systems. In particular, we can develop a rigorous notion for describing the amount of crosstalk arising from the difference of trajectories in both systems.

Definition C.2 (Crosstalk trajectory). Consider two systems, a crosstalk system and an alternative or reference system, initialized from the same initial condition \(x(0)\). For each initial condition \(x(0) = (y(0), x_h(0)) \in \mathbb{R}^3\) and input trajectory \(u(t)\), we define the crosstalk trajectory \(\xi(t)\) as

\[
\xi(t) = x^c(t) - x^c(t).
\]

The crosstalk trajectory is a time-evolving vector that describes the deviation of the physical system (subject to crosstalk) from the reference system’s trajectory. With this notion of crosstalk, we can also make precise the concept of crosstalk between states. We note that in writing the following quantity of interest \((\partial / \partial y)\xi_c\) it is with a slight abuse of notation, since \(\xi_c = x^c(t) - x^c(t)\). Mathematically, we are computing the 4th partial derivative of each term in \(\xi_c = x^c(t) - x^c(t)\). Thus, to be clear, when we write \((\partial / \partial y)\xi_c\) it will be implicit that we mean \((\partial / \partial x^c)\xi^c(t) = (\partial / \partial x)\xi^c(t)\).

Definition C.3 (Directed crosstalk). Given an initial condition of \((x(0), y(0))\) and input trajectory \(u(t)\) we say that a chemical species \(x_i\) exerts a crosstalk effect on chemical species \(x_j\) if the \(i\)th component of the crosstalk trajectory \(\xi(t)\) has non-zero partial derivative

\[
\frac{\partial}{\partial x_i} \xi(t) \neq 0,
\]

for some initial condition of \((x(0), y(0))\) and input trajectory \(u(t)\). In general, we will refer to \((\partial / \partial x)\xi_i(t)\) as the crosstalk sensitivity of \(x_i\) to \(x_j\).
Table 1. Table of genetic sequences for all parts used to make the transcriptional event detector circuit.

| sequence ID | sequence description | DNA sequence |
|-------------|----------------------|--------------|
| pAra-Lac    | hybrid promoter      | CATACGTATT... |
| BCD2        | RBS                  | GGCGCAATG... |
| TetR        | transcription factor | ATGTCTAGA... |
| ECK120033736| terminator           | AACGCATG... |
| BCD9        | RBS                  | GGCGCAATG... |
| AraC        | transcription factor | ATGTCTAGA... |
| ECK120029600| terminator           | TTAGGC...    |

(Continued.)
Table 1. (Continued.)

| sequence ID | sequence description | DNA sequence |
|-------------|----------------------|--------------|
| pLas-Tet    | hybrid promoter      | TTCTTCGAGCCTAGCAAGGGTCCGGGTTCACCGAAATCTA | TCTCATTTGCTAGTTAAATAATTTGAAATCTGTTGCTTAAATTC |
|             |                      | CTTATGAGACTGCTGACCTGAGCTGTGCAATTTTCATGGAATGACGTTGAGAGG | |
| BCD10       | RBS                  | GGCCGCAAGTTCACTTTAAAAGAGGATCAACATGAAAGAGA | ATTTTCTACTGAAACATTTGCTGAGGGACGATGTTTCCTG |
|             |                      | ATTTTCGTACTGAAACATCTGATTATCATGCGGAGGATCTGATT | |
| Lacl        | transcription factor  | ATGAAACCAGTAGTATGATGCTGGCAAGATCCGCGTGTTT | |
|             |                      | GCATTGCTGTTCTTGATGGCTAGCTCAGTCCTAGGTACAATGCAAGGACG | |
|             |                      | ATGGCCTTGGTTGACGGTTTTCTTGAGCTGGAACGCTCAAGGATGCTG | |
|             |                      | GCGAGCGACCTTGGATTCTCGAAGATCCTGTTCGGCCTG | |
|             |                      | GCCGGCTGCGGTTTTGTGGCCAGATTATCACGGCGTCAAACCAG | |
|             |                      | AATGCTTCTGGAGACCCAGGATGCGAAGCTTCAATTCCTGGAACGAGG | |
|             |                      | GCAGATCTGGTGGATGGCAAGCTGAGGTCGCAAGGAGCTGTTGA | |
|             |                      | GCCGGCATTAAAGTTCTCTGCGGCTGCTGTCGTTGCGTG | |
|             |                      | TGCATAAATATCTACTCGCAATACATCGAGCGATACGGGAAAC | |
|             |                      | GGGAGGCGACTGAGTGCCATGTCCGGTTTTTCAAAACAAAATG | |
|             |                      | GAAATGCTGGAATAGGGGACTCTTGACCAGTGCTG | |
|             |                      | CAAGATGATAAGGGGCTGCTGCCGCTGCAATACCCGCTTACGAGAT | |
|             |                      | CGCGGGCTGGCCTTTCTGGGAACCGTC | |
|             |                      | CATCTACCAGACGCGAAAGCAGCACGAGTTCTTCGAGGAA | |
|             |                      | GCCTCGGGCCGCGGCTGGATGATGCTGACTGCAAACGATTCGCGG | |
|             |                      | (Continued.) |
| sequence ID | sequence description | DNA sequence |
|-------------|----------------------|--------------|
| CTGATGTTGTGCCGGACACTCCGGCGCTGAGCCTC | CFP CDS | ATGAGTAAAGGAGAAGAACTTTTCACTGGAGTTGTCCCAATTTC |
| AGGGGGAAGCAACTTGTAAGCTCCATGCAAAATTTCATGGAATATGCTGCTGCTGAGAGAAGGTCGAGAGAATCGAATTAAAAGGTATTGTATTGAAAA |
| ATGGCTTCCTCCGAAGATGTATCAAAGAGTTCATGCGTTTCAAAGTTCGTATGGAAGGTTCCGTTAACGGTCACGAGTTCGAAATCGAAGGTGAAGGTGAAGGTGAAATACCTACAGTTAATAGAATCGAATTAAAAGGTATTGTATTGAAAA |
| mRFP CDS | ATGGCTTCCTCCGAAGATGTATCAAAGAGTTCATGCGTTTCAAAGTTCGTATGGAAGGTTCCGTTAACGGTCACGAGTTCGAAATCGAAGGTGAAGGTGAAATACCTACAGTTAATAGAATCGAATTAAAAGGTATTGTATTGAAAA |

(Continued.)
In both systems, we have supposed that time has been rescaled and the input $u(t)$. This dependency is consistent with the parametric sensitivity of biological function. Many genetic circuits in bacteria behave acceptably in one initial condition and for one input condition, e.g. in log-phase with an attenuated amount of a small molecule or sugar compound, but exhibit significantly different behavior when input concentrations are increased by an order of magnitude or subject to an alternative preparation method prior to the experiment. The latter imposes a state history that defines a distinct initial condition, which can drive a biological network to a highly coupled or decoupled state.

Example C.4. Consider two mRNA species $m_1$ and $m_2$ competing for the same degradation enzyme $D$ in a physical system. For simplicity of exposition, suppose their production dynamics do not depend on each other and can be modelled as $P_1(t)$ and $P_2(t)$, respectively. The crosstalk system is given as

$$m_1 = P_1(t) - \frac{Dm_1/k_{M1}}{1 + m_1/k_{M1} + m_2/k_{M2}}$$

and

$$m_2 = P_2(t) - \frac{Dm_2/k_{M2}}{1 + m_1/k_{M1} + m_2/k_{M2}},$$

while the reference system is given as

$$\hat{m}_1 = P_1(t) - \frac{D0m_1/k_{M1}}{1 + m_1/k_{M1}}$$

and

$$\hat{m}_2 = P_2(t) - \frac{D0m_2/k_{M2}}{1 + m_1/k_{M1}}.$$

In both systems, we have supposed that time has been rescaled so that the customary parameter $\omega_m$ for degradation is unity. The crosstalk sensitivities of $m_1$ and $m_2$ (with respect to each other) are given as

$$\frac{\partial \xi_1}{\partial m_2} = \frac{\partial}{\partial m_2} \int_0^t \frac{-D_0m_1/k_{M1}m_2/k_{M2}}{(1 + m_1/k_{M1})(1 + m_1/k_{M1} + m_2/k_{M2})},$$

and

$$\frac{\partial \xi_2}{\partial m_1} = \frac{\partial}{\partial m_1} \int_0^t \frac{-D_0m_1/k_{M1}m_2/k_{M2}}{(1 + m_1/k_{M1})(1 + m_1/k_{M1} + m_2/k_{M2})},$$

respectively. The crosstalk sensitivity between $m_1$ and $m_2$ is non-zero whenever $m_1$ or $m_2$ have non-zero initial condition.

Remark C.5. In synthetic biocircuit design, two chemical species $x_i$ and $x_j$ are often declared orthogonal when there is no designed interaction between them. Mathematically, in the crosstalk free system, this corresponds to

$$\frac{\partial}{\partial x_i} x_j^*(t) = 0,$$

for all $x(0)$ and $u(t)$. In such a situation, $\zeta(x_u, x_i) \neq 0$ if and only if

$$\frac{\partial}{\partial x_i} x_j^*(t) = \int_0^t\frac{f_j(y, x_i, u)\, d\tau}{0} \neq 0.$$

This condition is interesting in experimental settings since a computational estimate of $\frac{\partial}{\partial x_j} x_i^*(t)$ from perturbation experiments coincides with a direct estimate of the sensitivity of the crosstalk $\frac{\partial}{\partial x_i} \xi_i$. More specifically, when $x_i$ and $x_j$ are measured outputs of the system, we will show in the following that quantifying $\|Q_i(s)\|$ is directly related to an estimate of the crosstalk sensitivity $\frac{\partial}{\partial x_i} \xi_i(t)$ near the equilibrium point $x_i^*$.

Remark C.6. In general, estimating the crosstalk sensitivities for the nonlinear systems (C 1) and (C 4) can be challenging if either $x_i$ and $x_j$ are not measured directly. Firstly, if experimental data are available, they will often consist of data for the measured species $y$ in the crosstalk system, but not the reference system. Second, if only one of the species $x_i$ (or none) is available for measurement, even if perturbation of $x_i$ is possible, a nonlinear observer is required to estimate the trajectory of $x_i(t)$. Unless the parameters of $f_i(x, u)$ are known a priori (which is generally not the case), this then also requires system identification of the parameters of $f_i(x, u)$ and $f_j(x, u)$ which often results in a non-convex optimization problem.
Thus, our goal is to estimate the observed crosstalk between measured species \( Y_i \) and \( Y_j \). This crosstalk estimate will invariably include the dynamics of unmeasured chemical species (such as ATP, RNAP, untagged mRNA and protein species, DNA–protein complexes etc.). From a synthetic biology design standpoint, this is not a disadvantage, since the goal is to design a synthetic gene network with an abstracted circuit architecture operating reliably in the context of many unmeasured species. In any genetic circuit, there are always additional biochemical compounds that are unmeasured. As stated in the main text, theorem 2.1 shows that always additional biochemical compounds that are unmeasured. Since the goal is to design a synthetic gene network with an abstracted circuit architecture operating reliably in the context of many unmeasured species, this is not a disadvantage.

\[ Y_c = f_y(y_c, x_s, u), \quad y'(0) = 0 \] 
\[ x'_s = f_x(y_c, x_s, u), \quad x'_s(0) = 0, \]  
\( y_c \in \mathbb{R}^p \) is a vector of measured states in the output, \( x'_s \in \mathbb{R}^q \) are the unmeasured states of the system, \( x = (y_c, x'_s) \) is the full system state, and \( u \in \mathbb{R}^m \) is a vector of system inputs. Furthermore, suppose we have an idealized system model to simulate system dynamics in the absence of crosstalk

\[ \hat{y}' = f_y'(y'_c, x'_s, u), \quad y'(0) = 0 \] 
\[ \hat{x}'_s = f_x'(y'_c, x'_s, u), \quad x'(0) = 0, \]  
where \( y'_c \in \mathbb{R}^p \) is a vector of the measured states in the output, \( x'_s \in \mathbb{R}^q \) are the unmeasured states of the system, \( x'' = (y'_c, x'_s) \) is the full system state, and \( u \in \mathbb{R}^m \) is a vector of system inputs. Let \( (Q'(s), P'(s)) \) and \( (Q(s), P(s)) \) denote the respective dynamical structure functions calculated for each linearized system about the origin. Let

\[ \xi(t) = x''(t) - x'(t) \]

denote the deviation of the crosstalk state from the idealized state. Then

\[ \frac{\partial \mathcal{L}(\xi)}{\partial Y_i} = Q'_i(s) - Q_0(s) + \frac{\partial}{\partial Y_i} \mathcal{L}(O(x'')^2), \]  
and in particular, if

\[ Q'_i(s) = 0 \]

then

\[ \frac{\partial \mathcal{L}(\xi)}{\partial Y_i} = -Q_0(s) + \frac{\partial}{\partial Y_i} \mathcal{L}(O(x')^2), \]

and can be estimated from input output data \( (Y(s), U(s)) \).

Proof. First, note that the Laplace transform of \( \mathcal{L}(\xi(t)) = \mathcal{L}(x'' - x') = \mathcal{L}(X_c - X_s(s)) \), which can be decomposed into its measured and unmeasured states

\[
\begin{bmatrix}
Y_c(s) \\
X'_s(s)
\end{bmatrix} = \begin{bmatrix}
Q'Y_c(s) - Q'Y_s(s) + (P' - P')U(s) + \mathcal{L}(O(x')) \\
X'_s(s) - X'_s(s) + \mathcal{L}(O(x'))
\end{bmatrix}.
\]
In figure 11, we plot the time-lapse response of \( y_2(t) \) and \( y_3(t) \) for varying parameter values of \( k_{2,d} \) in equation (2.9). The \( k_{2,d} \) parameter is a Michaelis constant that determines the effective affinity of substrate \( x_2 \) in binding with \( C_0 \). As \( k_{2,d} \) increases, the affinity of substrate \( x_2 \) is diminished, relative to the affinity of \( x_1 \) and \( x_3 \). Attenuating \( k_{2,d} \) can be viewed as similar to swapping out a strong degradation marker for protease degradation with a weaker degradation marker on the species \( x_2 \). In the experimental literature, there are multiple degradation markers for proteins that confer varying binding affinities to an associated protease [88]. In our simulation, we consider five potential values for \( k_{2,d} \): 500, 1625, 2750, 3875 and 5000 \( \mu \)M corresponding to five artificial LVA markers of varying strengths for the protease ClpXP frequently used in \( E. coli \). Note that as we decrease the affinity of \( y_2 \) for ClpXP, this also coincides with an increased \( \xi_2 \) crosstalk magnitude. Here, we have computed \( \xi_2 = y_2^0(t) - y_2^1(t) \). We find that \( |\xi_2| \) increases as \( k_{2,d} \) increases. In figure 11b–d, \( \xi_2 \) is plotted as a percentage of maximum absolute change across all values of \( k_{2,d} \).

We see that the time-lapse response of \( y_2(t) \) increases monotonically for all \( t \) as the crosstalk \( \xi_2(t) \) increases. This is consistent with biological intuition, since an increase in competition for resource loading (an increase in \( k_{2,d} \)) results in prolonged lifetimes of each individual \( y_2 \) (TetR-YFP) protein. This in turn results in higher repression levels of \( y_3 \) in the incoherent feedforward loop. Increased competition for ClpXP from substrates \( y_3 \) and \( y_1 \) have the effect of damping \( y_3 \) dynamics and reinforcing the pulsatile response of the IFIIL. The crosstalk in this circuit thus has the effect of effectively strengthening the negative regulation of \( y_3 \) on \( y_2 \), encouraging the downward transient after 0.75 h. Our network analysis shows we can improve the robustness of an IFIIL’s pulse by attenuating the relative binding affinity of the repressor to its protease.

In general, crosstalk effects do not necessarily reinforce the feedback architecture of a biocircuit. This underscores the importance of having techniques for quantifying crosstalk in a synthetic gene network and validating that designed interactions are dominant over crosstalk interactions.
