INTERFACE

royalsocietypublishing.org/journal/rsif

Research

Cite this article: Yordanov P, Stelling J. 2020 Efficient manipulation and generation of Kirchhoff polynomials for the analysis of non-equilibrium biochemical reaction networks. *J. R. Soc. Interface* **17**: 20190828. http://dx.doi.org/10.1098/rsif.2019.0828

Received: 6 December 2019
Accepted: 30 March 2020

Subject Category:
Life Sciences–Mathematics interface

Subject Areas:
computational biology, systems biology, biomathematics

Keywords:
matrix-tree theorem, graph Laplacian, symbolic steady state, Markov process master equation, linear compartmental models, linear nonnegative systems

Author for correspondence:
Jörg Stelling
e-mail: joerg.stelling@bsse.ethz.ch

[†]Present address: Department of Systems Biology, Harvard Medical School, Boston, MA, USA.

Electronic supplementary material is available online at https://doi.org/10.6084/m9.figshare.c.4938138.

THE ROYAL SOCIETY
PUBLISHING

Efficient manipulation and generation of Kirchhoff polynomials for the analysis of non-equilibrium biochemical reaction networks

Pencho Yordanov[†] and Jörg Stelling

Department of Biosystems Science and Engineering and SIB Swiss Institute of Bioinformatics, ETH Zurich, Basel, Switzerland

PY, 0000-0002-4231-0256; JS, 0000-0002-1145-891X

Kirchhoff polynomials are central for deriving symbolic steady-state expressions of models whose dynamics are governed by linear diffusion on graphs. In biology, such models have been unified under a common linear framework subsuming studies across areas such as enzyme kinetics, G-protein coupled receptors, ion channels and gene regulation. Due to 'history dependence' away from thermodynamic equilibrium, these models suffer from a (super) exponential growth in the size of their symbolic steady-state expressions and, respectively, Kirchhoff polynomials. This algebraic explosion has limited applications of the linear framework. However, recent results on the graph-based prime factorization of Kirchhoff polynomials may help subdue the combinatorial complexity. By prime decomposing the graphs contained in an expression of Kirchhoff polynomials and identifying the graphs giving rise to equal polynomials, we formulate a coarse-grained variant of the expression suitable for symbolic simplification. We devise criteria to efficiently test the equality of Kirchhoff polynomials and propose two heuristic algorithms to explicitly generate individual Kirchhoff polynomials in a compressed form; they are inspired by algebraic simplifications but operate on the corresponding graphs. We illustrate the practicality of the developed theory and algorithms for a diverse set of graphs of different sizes and for non-equilibrium gene regulation analyses.

1. Introduction

Linear diffusion (of information, probabilities, concentrations) on graphs [1–3], and the mathematically equivalent linear non-negative and compartmental systems [4,5] are abundantly used in science to model linear dynamical processes. The great significance of Kirchhoff polynomials [6] stems from their role in linking the graph topologies to the symbolic steady-state expressions of such processes. In biology, analyses of linear diffusion processes on graphs, originating from areas as diverse as enzyme kinetics, G-protein-coupled receptors and gene regulation, have recently been unified under a common mathematical linear framework [7]. The linear framework represents a biological system as a labelled directed graph (graph, for short) having molecular states as vertices, state transitions as edges, and transition rate constants as edge labels. The dynamics on the graph have deterministic (linear ODEs) and stochastic (Markov process master equation) interpretations, and define how states (concentrations, respectively, probabilities) evolve over time [8]. Closed-form steady states of such linear framework models (LFMs) always exist and can be symbolically derived from initial conditions and the basis of the kernel of a matrix representation of LFMs, namely the graph Laplacian matrix [7]. For systems at thermodynamic equilibrium, the principle of detailed balance dictates 'history-independent' equilibrium steady states, for which the basis elements of the kernel can be derived from products of equilibrium constants

© 2020 The Author(s) Published by the Royal Society. All rights reserved.

along any path in the model graph [7]. However, a breakdown of detailed balance occurs when systems expend energy, leading to 'history-dependent' non-equilibrium steady states of substantially higher algebraic complexity [9]. Namely, away from equilibrium a basis element of the kernel of the graph Laplacian matrix becomes a homogeneous multivariate polynomial called the *Kirchhoff polynomial*, which according to Tutte's Matrix-Tree Theorem [10] can be equivalently obtained by (i) symbolically deriving all (j, j)-minors of the graph Laplacian matrix and summing them up, and by (ii) enumerating all spanning trees in the model graph, multiplying the symbolic labels in each tree, and adding the resulting monomials of all spanning trees.

Departure from equilibrium and the ensuing 'history dependence' pose a fundamental challenge—the number of spanning trees and, correspondingly, the size of Kirchhoff polynomials and symbolic steady-state expressions frequently grows super-exponentially with the size of the graph models [11]. Symbolic derivations bring great benefits in understanding non-equilibrium biological phenomena despite this seemingly unmanageable combinatorial explosion. In consequence, there has been a prolific development of software (reviewed in [12]) and of methods to derive steady states and rate equations of biological models that can be classified as falling within the linear framework using, among others, graph theoretical methods [13,14], systematic determinant expansion [15] and Wang algebra [16]. In computer science, advances have also been made to enumerate the set of all spanning trees from which Kirchhoff polynomials are obtained [17,18]. However, all existing exact methods and algorithms suffer from the aforementioned combinatorial explosion and they only offer limited and *ad hoc* manipulation of steady-state expressions. This hinders the in-depth understanding of the role of energy expenditure in biological systems, the extraction of general principles of eukaryotic gene regulation [9] and differential signalling [19], and the analysis of more detailed models that follow from advanced experimental techniques, such as phosphoproteomics [20].

An important step towards taming the combinatorial complexity is the realization that a model graph G can be efficiently decomposed into smaller graphs whose Kirchhoff polynomials are prime factors of the Kirchhoff polynomial of G [21]. This graph-based polynomial factorization provides a natural, compact representation that does not directly depend on the number of spanning trees but rather on directed graph connectivity. Here, we exploit the factorization to further develop theory and algorithms for dissecting and mitigating the seemingly intractable combinatorial complexity. Our approach aims to simplify expressions of Kirchhoff polynomials, bypassing the customary expensive symbolic generation and manipulation by computer algebra systems. Specifically, we consider the prime factors of all Kirchhoff polynomials in an expression as symbolic variables. The resulting coarse-grained expressions allow for symbolic simplification without explicit generation of the polynomials. To explicitly generate the Kirchhoff polynomials, e.g. as is needed for their repeated evaluation, we propose a recursive and an iterative heuristic algorithm inspired by algebraic simplification, but operating on the graphs alone. Applied to a collection of graphs, in particular, graph connectivity aware heuristics prove to be useful in practice, with large compressions and short running times. Furthermore, we extend the sharpness analysis of gene expression in development from [9] and show that away from equilibrium, four binding sites allow for previously unknown qualitative shapes of gene regulation functions (GRFs). The methods and algorithms are implemented in the Python package *KirchPy* available at https://gitlab.com/csb.ethz/KirchPy.

2. Background

2.1. Linear framework models

Let us consider the model of Ca^{2+}-dependent nuclear translocation of the nuclear factor of activated T cells (NFAT) from [22], which can be expressed in the linear framework. NFAT can be in one of three states—cytoplasmic phosphorylated (N_c^*), cytoplasmic dephosphorylated (N_c), or nuclear (N_n), and undergo four reactions—dephosphorylation, phosphorylation, nuclear import and nuclear export with respective rate constants r_1, r_2, r_3 and r_4. This system functions away from equilibrium because it contains multiple irreversible, and thus energy expending reactions.

The reaction scheme of NFAT can formally be represented as a simple labelled directed graph $G = (V, E)$ (figure 1*a*). The graph G is composed of a set of vertices $V(G) = \{v_{N_c^*}, v_{N_c}, v_{N_n}\}$ corresponding to NFAT states (correspondence marked in subscript), and a set of edges (ordered pairs of distinct vertices; no multiple parallel edges allowed) $E(G) = \{v_{N_c^*}v_{N_c}, v_{N_c}v_{N_c^*}, v_{N_c}v_{N_n}, v_{N_c}v_{N_c^*}\}$ corresponding to reactions. We associate a label $\ell(uv)$, standing for a mathematical expression, to each edge $uv \in E(G)$. For example, with $\ell(v_{N_c^*}v_{N_c}) = r_1$, we mark that the label associated with $v_{N_c^*}v_{N_c}$ is the rate constant r_1. Additionally, by $\ell(G)$, we define the *set of all edge labels* of G.

Here, we are primarily interested in LFMs that correspond to *strongly connected* graphs. Namely, graphs G in which there exists a directed path from u to v and from v to u for any two vertices $u, v \in V(G)$ (as in figure 1*a*); figure 1*d* shows an example for which this property does not hold. However, this does not limit the generality of the developed algorithms and theory.

LFMs are frequently obtained from more complicated models after applying the technique of timescale separation, stating that a part of a biochemical system operating much faster than the rest of the system can be assumed to have reached a steady state [7]. This model reduction could result in edge labels involving (nonlinear) algebraic expressions of kinetic parameters and species concentration terms. To retain the linearity of LFMs, concentration terms in labels must correspond to species not contained in $V(G)$. These could be species acting on the slow timescale or other entities as in the case of NFAT, where r_1 is assumed to be modulated by Ca^{2+} oscillations. We circumvent explicitly dealing with the arbitrary, though biologically significant, algebraic structure of the label expressions by regarding them as uninterpreted symbols $\ell(uv)$ that denote unique edge names.

We concentrate on the deterministic interpretation of LFM dynamics (also called Laplacian dynamics) and associate each vertex $v_i \in V(G)$ in G to a non-negative species concentration x_i and each edge to a mass-action reaction. In the resulting dynamical system, species concentrations associated with vertices flow in the direction of the edges at rates proportional to the concentrations on the edges' source vertices, where proportionality is set by the edge label $\ell(uv)$.

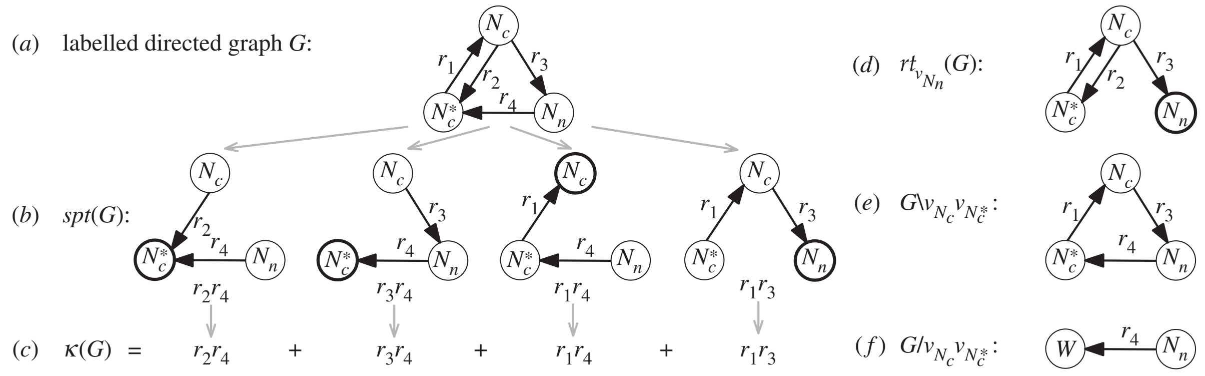


Figure 1. Example model for nuclear translocation of the nuclear factor of activated T cells (NFAT) [22]. (*a*) Graph *G*, (*b*) all its spanning trees, and (*c*) the corresponding Kirchhoff polynomial. (*d*) The graph obtained by rooting *G* at v_{N_n}, (*e*) the edge deleted graph $G\backslash v_{N_c}v_{N_c^*}$, and (*f*) the edge contracted graph $G/v_{N_c}v_{N_c^*}$. Vertex labels mark the states that they represent and *W* denotes a vertex obtained after edge contraction. Highlighted vertices are, respectively, roots of the corresponding spanning tree and of all spanning trees when rooting a graph.

The example model from figure 1*a* is *closed*, it does not exchange matter with the environment. The dynamics of closed LFMs can be expressed in the form

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \mathcal{L}(G)x, \tag{2.1}$$

where $x = (x_1, \ldots, x_n)^{\mathrm{T}}$ is the vector of species' concentrations corresponding to each vertex $v_1, \ldots, v_n \in V(G)$ and $\mathcal{L}(G)$ is the *graph Laplacian matrix* of G defined as

$$\mathcal{L}(G)_{ij} = \begin{cases} \ell(v_jv_i) & \text{if } i \neq j, \\ -\sum_{r\neq j}\ell(v_jv_r) & \text{if } i = j, \end{cases} \tag{2.2}$$

and $\ell(v_jv_i) = 0$ when $v_jv_i \notin E(G)$. For the example model, this means

$$x = \begin{pmatrix} x_{N_c^*} \\ x_{N_c} \\ x_{N_n} \end{pmatrix} \quad \text{and} \quad \mathcal{L}(G) = \begin{pmatrix} -r_1 & r_2 & r_4 \\ r_1 & -(r_2+r_3) & 0 \\ 0 & r_3 & -r_4 \end{pmatrix}. \tag{2.3}$$

In closed systems, the total amount of material x_t is conserved according to a single conservation law $x_1 + \cdots + x_n = x_t$. The system has a unique stable steady state that can be derived symbolically from initial conditions and the kernel of $\mathcal{L}(G)$ [8].

Graphs G for *open* LFMs with synthesis and degradation reactions are obtained by adding a vertex $v_\emptyset$ representing the environment to a *core graph* $\overline{G}$ (akin to closed systems, the core graph is composed of all non-synthesis and non-degradation reactions), and by introducing directed edges from $v_\emptyset$ to the synthesized species in $\overline{G}$ with labels s_i and edges labelled δ_i from the degraded species to $v_\emptyset$. The dynamics of open LFMs are defined in general form as

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \mathcal{L}(\overline{G})x - \Delta x + S,$$

where $\mathcal{L}(\overline{G})$ is the graph Laplacian matrix of the core graph, Δ is a diagonal matrix with $\Delta_{ii} = \delta_i$ the degradation rate constants of species i, and S is a vector $S_i = s_i$ comprising the synthesis rate constants. In open systems, the total amount of matter is not conserved, but synthesis and degradation at steady state are balanced: $\delta_1x_1 + \cdots + \delta_nx_n = s_1 + \cdots + s_n$. Similarly to closed systems, but assuring that the steady-state concentration at $v_\emptyset$ is always 1, the unique stable steady state for vertex v_i ($v_i \neq v_\emptyset$) can be derived symbolically. For more details on LFMs, see [7,8,23].

2.2. Spanning trees

A class of *subgraphs*, so-called spanning trees, connect non-equilibrium steady states of LFMs to model graph structure.

A graph H is a subgraph of a graph G if $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$, such that every edge in G between vertices in H is also an edge in H. For $V' \subseteq V(G)$, $G[V']$ denotes the *induced subgraph* of G by the set of vertices V'. A strongly connected component (SCC) of G is any largest (w.r.t. vertex inclusion) strongly connected induced subgraph of G. The definition implies that no two distinct SCCs share a vertex, that is, the SCCs $G_1, \ldots, G_k$ of a graph G induce a unique partition $V(G_1), \ldots, V(G_k)$ of $V(G)$. Furthermore, two distinct SCCs G_i and G_j can be connected by either a directed path from G_i to G_j, or from G_j to G_i, but not by both. The existence of such unidirectional paths induces a unique partial order on the SCCs $G_1, \ldots, G_k$.

A *rooted directed spanning tree* (spanning tree, for short) A is a subgraph of G that spans its vertex set such that there is a unique directed path from any vertex to a root vertex. We denote the set of all spanning trees of G by $spt(G)$, and the set of all spanning trees rooted at a vertex v by $spt_v(G)$ (see figure 1*b* for all spanning trees of the example graph). To obtain a graph containing only spanning trees rooted at a vertex v, we define the graph *rooting* operation rt, so that $rt_v(G)$ is the graph constructed from G by removing all edges outgoing from v (figure 1*d*). Likewise, we call a graph G *rooted* at a vertex v if v has no outgoing edges and v is reachable from every other vertex in G. Observe that $spt_v(G) = spt(rt_v(G))$. Graph G contains a spanning tree iff the partial order of the SCCs has exactly one maximal element, i.e. no other SCC is reachable from a maximal SCC. Such a maximal SCC is also called a *terminal SCC*.

2.3. Kirchhoff polynomials and steady states

A spanning tree A of a graph G with n vertices is a subgraph with $n-1$ edges $e_1, \ldots, e_{n-1} \in E(G)$ (we denote edges by e when not interested in the pairs of vertices defining them). In a *uniquely labelled graph* G, i.e. when no two edge labels in G are the same, A can also be represented as a monomial

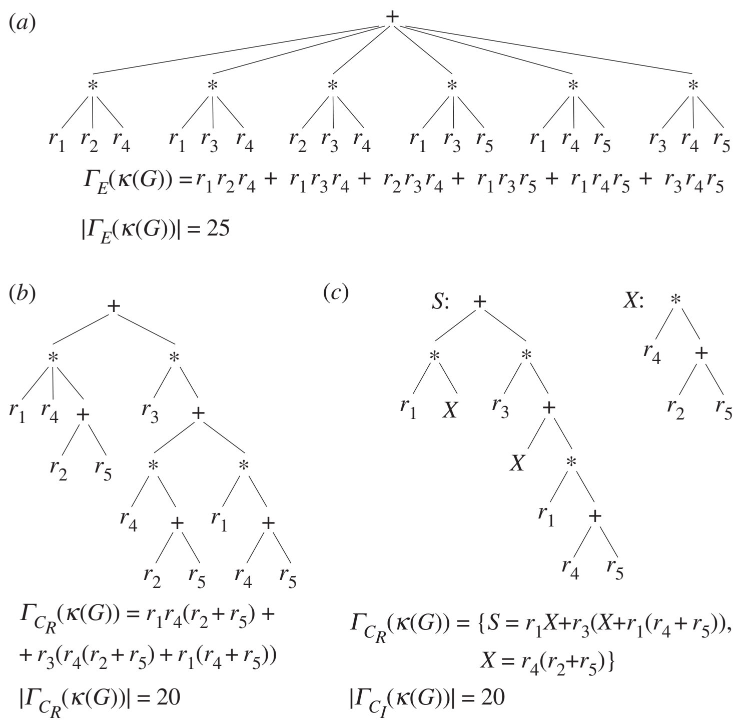

(a)
$\Gamma_E(\kappa(G)) = r_1r_2r_4 + r_1r_3r_4 + r_2r_3r_4 + r_1r_3r_5 + r_1r_4r_5 + r_3r_4r_5$
$|\Gamma_E(\kappa(G))| = 25$
(b)
$\Gamma_{C_R}(\kappa(G)) = r_1r_4(r_2+r_5) + r_3(r_4(r_2+r_5)+r_1(r_4+r_5))$
$|\Gamma_{C_R}(\kappa(G))| = 20$
(c)
S:
X:
$\Gamma_{C_R}(\kappa(G)) = \{S = r_1X+r_3(X+r_1(r_4+r_5)), X = r_4(r_2+r_5)\}$
$|\Gamma_{C_I}(\kappa(G))| = 20$

Figure 2. Algebraically equivalent representations of a Kirchhoff polynomial, their expression trees, and sizes for an example graph G (see electronic supplementary material, figure S1) in (*a*) the fully expanded representation, (*b*) a simplified representation, e.g. obtained by algorithm C_R and (*c*) a change of variables form (forest of expression trees), e.g. obtained by algorithm C_I. The size of a representation is the sum of the numbers of branch vertices and of leaves in the expression tree. With change of variables, each expression tree from the forest is assigned a pointer counting as 1 to the size of the representation and pointing to the leaves of other expression trees where it should be substituted to obtain the expression tree of the complete Kirchhoff polynomial. The pointer S denotes the 'starting' tree.

$\ell(e_1)\ell(e_2)\cdots\ell(e_{n-1})$ in the edge labels of G. Furthermore, the set of all spanning trees of G can be represented by a homogeneous multivariate polynomial over the variables $\ell(e_i)$, $e_i \in E(G)$. This polynomial is called the *Kirchhoff polynomial* $\kappa(G)$ (figure 1*c*)

$$\kappa(G) = \sum_{A\in spt(G)} \prod_{e_i\in E(A)} \ell(e_i). \tag{2.4}$$

Note that $\kappa(G) = 0$ when no spanning trees exist in G and $\kappa(G) = 1$ when G consists of a single vertex. We also denote the Kirchhoff polynomial of all spanning trees rooted at vertex v by $\kappa_v(G) = \sum_{A\in spt_v(G)} \prod_{e_i\in E(A)} \ell(e_i)$ (a shorter notation for $\kappa(rt_v(G))$).

A Kirchhoff polynomial $\kappa(G)$ can have multiple *algebraically equivalent representations* $\Gamma_i(\kappa(G))$ (i indexes all such representations) corresponding to different *expression trees* (figure 2). We consider expression trees in which the branch vertices represent the operations of n-ary addition or multiplication, and leaf vertices are the unique edge labels $\ell(G)$, the variables of $\kappa(G)$. We define the *size* of a representation of $\kappa(G)$, $|\Gamma(\kappa(G))|$, as the size of its corresponding expression tree. However, a change of variables requires an extended definition because it produces a forest of expression trees, and not a single tree (figure 2*c*). We define the size of a Kirchhoff polynomial in such a representation as the total number of branch vertices and leaves in the forest plus the number of expression trees in the forest. We need to account for the number of expression trees since each of them has a unique pointer indicating its location within the other expression trees.

The unique steady state of LFMs can be symbolically obtained from initial conditions and the kernel of the graph Laplacian matrix by employing *Tutte's Matrix-Tree Theorem* [10].

Theorem 2.1 (Tutte's Matrix-Tree Theorem). *Let G be a graph with n vertices then the minors $\mathcal{L}(G)_{(i,j)}$ of its Laplacian matrix can be expressed, up to a sign, by the Kirchhoff polynomial rooted at the vertex v_j corresponding to the j-th column of $\mathcal{L}(G)$ as*

$$\mathcal{L}(G)_{(i,j)} = (-1)^{n+i+j-1} \sum_{A\in spt_{v_j}(G)} \prod_{e\in E(A)} \ell(e) = \kappa_{v_j}(G).$$

As a result, the non-equilibrium steady-state concentration x_i^{SS} of species i associated with vertex v_i in a closed LFM with a strongly connected graph G is a fraction of Kirchhoff polynomials

$$x_i^{SS} = \frac{\kappa_{v_i}(G)}{\kappa(G)} x_t. \tag{2.5}$$

Correspondingly, for open systems and a vertex $v_i \neq v_\emptyset$

$$x_i^{SS} = \frac{\kappa_{v_i}(G)}{\kappa_{v_\emptyset}(G)}.$$

Note that the Kirchhoff polynomial $\kappa(G)$ in the denominator of the steady-state expression for closed systems acts as a non-equilibrium partition function [9]. For more details on LFMs, derivations, equilibrium steady states and steady states in non-strongly connected graphs see [7,8,23].

2.4. Deletion, contraction and prime factorization

By $G \setminus e$, we denote the graph obtained from G by deleting edge $e \in E(G)$ (figure 1*e*). Additionally, for a graph G and an edge $e = v_i v_j \in E(G)$, G/e is the *edge contracted* graph constructed from G by (i) removing the edge $v_j v_i$, if it exists, and all outgoing edges from v_i, i.e. $v_i u \in E(G)$ and (ii) fusing vertices v_i and v_j into a new vertex w (figure 1*f*). Edge contractions may give rise to graphs with multiple parallel edges between two vertices. To correct this, we replace m multiple parallel edges $e_1, e_2, \ldots, e_m$ from u to v with a single edge $e = uv$ so that $\ell(e) = \ell(e_1) + \ell(e_2) + \cdots + \ell(e_m)$.

The defined graph operations can be used to decompose $\kappa(G)$, given an edge $e \in E(G)$, into a sum of Kirchhoff polynomials according to the classic *deletion–contraction* identity [24]

$$\kappa(G) = \kappa(G \setminus e) + \ell(e)\kappa(G/e). \tag{2.6}$$

A Kirchhoff polynomial P is a *factor* of another Kirchhoff polynomial Q, if there exists a Kirchhoff polynomial R such that $Q = P \cdot R$. A Kirchhoff polynomial P that cannot be factorized into non-trivial factors is called *prime*. Correspondingly, for graphs, G′ *is a component (a prime component) of* G if $\kappa(G')$ *is a factor (a prime factor) of* $\kappa(G)$. Mihalák *et al.* [21] introduce graph decomposition rules that correspond to factorization steps of the Kirchhoff polynomial. In particular, the method yields in linear time graphs whose Kirchhoff polynomials are prime factors of the Kirchhoff polynomial of the original G

$$\kappa(G) = \prod_{i=1}^{n} \kappa(P_i),$$

where P_i are the prime components of G. A prime component P_i can be either (i) strongly connected or (ii) rooted at v such that $P_i \setminus v$ (here $\setminus$ denotes vertex deletion) is strongly connected and P_i does not have any non-trivial vertex dominators (a vertex u dominates a vertex w if every path from w to v goes through u). We call graphs with prime Kirchhoff polynomials also *prime graphs*. Importantly, the prime factorization is conditional on label uniqueness—when different edges have equal labels or there are variables shared across labels, the factorization is not guaranteed to be prime.

3. Efficient manipulation of Kirchhoff polynomials

Non-equilibrium steady states of LFMs are ratios (or more generally: expressions) of Kirchhoff polynomials. Similarly, any symbolic expression derived from steady-state LFMs through arithmetic and calculus will also comprise expressions of Kirchhoff polynomials. Examples are ratios of steady states, steady-state rate equations, EC_{50} values for steady-state dose–response curves, differential responses [19], and steady-state parameter sensitivities (expressions differentiated with respect to a reaction constant). Correspondingly, algebraic manipulation of expressions of Kirchhoff polynomials is important to understand when expressions can be simplified. For example, the steady state of a LFM can be simplified if the numerator and denominator share common factors. After all common factors are crossed out, numerator and denominator become relatively prime and further simplification is not possible.

To circumvent tedious symbolic manipulation of combinatorially complex algebraic expressions, we exploit properties of Kirchhoff polynomials that allow their implicit manipulation, that is, without explicitly generating polynomials in expanded form but working with the corresponding graphs. More precisely, we (i) find the prime components corresponding to prime factors of all Kirchhoff polynomials in the expression, (ii) determine which prime components generate identical Kirchhoff polynomials, and (iii) form a *coarse-grained representation* of the original expression by substituting prime components with symbolic variables, where prime graphs with equal Kirchhoff polynomials are assigned the same variable, and finally, (iv) symbolically simplify the coarse-grained expression. However, it is an open problem to efficiently determine which prime components generate equal Kirchhoff polynomials without their explicit generation.

3.1. Prime graphs with equal Kirchhoff polynomials

We consider Kirchhoff polynomial equality in the algebraic sense. By uniquely labelling a graph we assign identity to each edge through its label, that is, a label defines a particular reaction. Applying the graph operations of prime decomposition, edge deletion, edge contraction, and vertex rooting to a uniquely labelled graph preserves the identity of the reactions while the names of the vertices can change. However, when comparing two Kirchhoff polynomials originating from different sources, identical (different) reactions between the sources need to carry the same (different) labels to have a meaningful comparison.

A necessary condition for two polynomials to be equal is that they have the same set of variables corresponding to terms with non-zero coefficients. This condition cannot be transferred directly to compare the graphs generating Kirchhoff polynomials because the graphs may contain *nuisance edges* that do not participate in any spanning tree. With nuisance edges, the set of labels of two graphs that generate equal Kirchhoff polynomials will be different. However, if we compare only prime graphs we can prove that they do not contain nuisance edges because every edge participates in at least one spanning tree.

Theorem 3.1. *Let G be a prime graph, then each edge in G participates in at least one spanning tree.*

The absence of nuisance edges in prime graphs allows us to formulate a necessary condition for Kirchhoff polynomial equality.

Corollary 3.2. *Let G and H be two prime graphs with equal Kirchhoff polynomials, then G and H have equal sets of edge labels, i.e.* $\kappa(G) = \kappa(H) \Rightarrow \ell(G) = \ell(H)$.

Proof. Follows directly from theorem 3.1. □

The condition can be tested efficiently since it involves only a comparison between sets, but it is not a sufficient condition for Kirchhoff polynomial equality (figure 3*a*).

To obtain a graph-based sufficient condition of Kirchhoff polynomial equality, we define the term *λ-isomorphism* to denote a vertex bijection that is edge-preserving and enforces the corresponding edges to have identical labels.

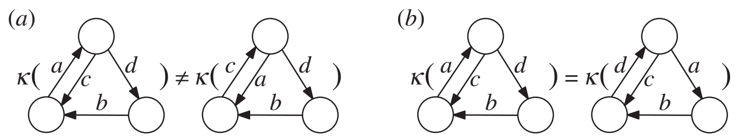


Figure 3. Examples for (*a*) two graphs with equal edge label sets but different Kirchhoff polynomials and for (*b*) two non-λ-isomorphic prime graphs with identical Kirchhoff polynomials.

Definition 3.3 (λ-isomorphism). Two labelled graphs G and H are called λ-isomorphic, denoted $G \simeq_\lambda H$, iff there exists a bijective mapping $\psi: V(G) \mapsto V(H)$, such that

1. $uv \in E(G)$ iff $\psi(u)\psi(v) \in E(H)$ and
2. $\ell(uv) = \ell(\psi(u)\psi(v))$.

Evidently, two λ-isomorphic graphs give rise to equal Kirchhoff polynomials because the graphs differ only by vertex names, and otherwise have identical topology and labels.

Observation 3.4. Let G and H be λ-isomorphic, then they generate identical Kirchhoff polynomials, i.e. $G \simeq_\lambda H \Rightarrow \kappa(G) = \kappa(H)$.

To derive a condition testing for λ-isomorphism we first define the so-called line graph $\mathscr{L}(G)$ associated with G.

Definition 3.5 (line graph). The line graph $\mathscr{L}(G)$ associated with the graph G satisfies the conditions

1. the vertices of $\mathscr{L}(G)$ are the *unique edge labels* of G, i.e. $V(\mathscr{L}(G)) \equiv \ell(G)$ and
2. two vertices $u, v \in V(\mathscr{L}(G))$ are joined by an edge uv iff $u = \ell(rs)$, $v = \ell(st)$ for $r, s, t \in V(G)$.

Theorem 3.6. *Two prime graphs G and H are λ-isomorphic iff the edge sets of their line graphs are equal, i.e.* $G \simeq_\lambda H \Leftrightarrow E(\mathscr{L}(G)) = E(\mathscr{L}(H))$.

Theorem 3.6 allows us to formulate a sufficient condition for prime Kirchhoff polynomial equality.

Corollary 3.7. *Let G and H be two uniquely labelled prime graphs whose line graphs have equal edge sets, then the Kirchhoff polynomials they generate are equal, i.e.* $E(\mathscr{L}(G)) = E(\mathscr{L}(H)) \Rightarrow \kappa(G) = \kappa(H)$.

Proof. Follows directly from observation 3.4 and theorem 3.6. □

The sufficient condition in corollary 3.7 is also cheap to evaluate since it only involves line graph construction, which has quadratic time complexity, and the comparison of two sets. The condition is not necessary for prime Kirchhoff polynomial equality (figure 3*b*).

3.2. Formulation of coarse-grained expressions

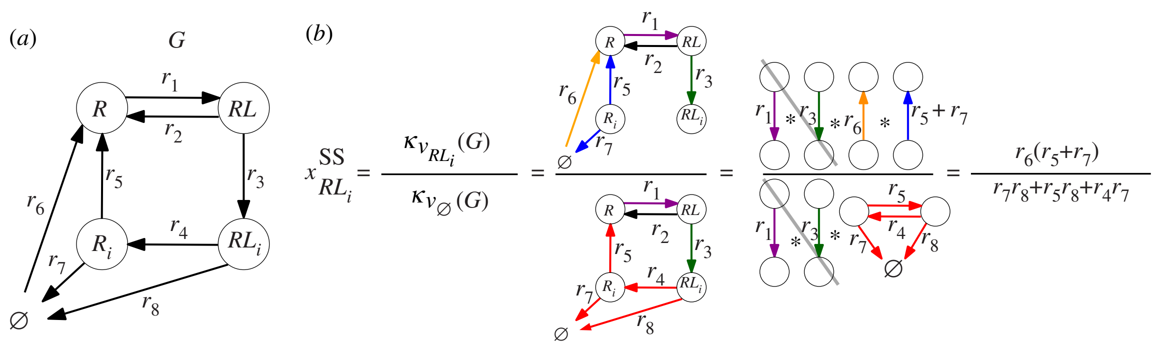
We use the conditions in corollary 3.2 and corollary 3.7 to assign identical variable names to prime graphs with equal Kirchhoff polynomials when formulating the coarse-grained description of an expression of Kirchhoff polynomials without their explicit generation. First, we apply the necessary condition to filter possible matches, and then the sufficient one to certify the equality. Pairs of prime graphs that are non-λ-isomorphic but have the same label sets require special attention. With such pairs, we cannot guarantee that we have identified all graphs with equal Kirchhoff polynomials, which translates to a lack of guarantees for maximal symbolic simplification of the coarse-grained description. However, without such pairs of graphs in the expression, we can guarantee the exhaustive identification of prime graphs with equal Kirchhoff polynomials. Furthermore, comparisons of prime graphs can be accelerated by realizing that each prime component of a graph is equal to at most one prime component of another graph, since prime factorization partitions the set of labels. Note that there might be other reasons that do not guarantee full simplification and contexts in which full simplification is guaranteed (see electronic supplementary material for details). Additionally, some proofs and derivations assume that the graph models have unique and irreducible expressions in their labels. If this assumption is not met, e.g. when different reactions have the same rate constant, rate constants are expressions that can be simplified, or rate constants contain symbols shared across different labels, then additional symbolic simplification might be required since the primality of the decomposition is not guaranteed and the manipulation formulae of the coarse-grained representation might not hold.

The study of non-equilibrium steady-state LFMs is not complete without efficient methods to apply differentiation and integration to Kirchhoff polynomials, for example, to derive parameter sensitivities. In electronic supplementary material, we show that the properties of Kirchhoff polynomials allow us to map differentiation and integration to graph operations, and thus to work with the implicit coarse-grained representation.

3.3. Application examples

To illustrate the manipulation of expressions of Kirchhoff polynomials in the coarse-grained representation, we first consider a simple open receptor trafficking model with graph G shown in figure 4*a*. It consists of species for an unbound surface receptor R, a cell surface ligand–receptor complex RL, their respective internalized counterparts R_i and RL_i, and a set of state transition, synthesis, and degradation reactions. Figure 4*b* shows the steady state for RL_i obtained by prime decomposing the graphs in the steady-state ratio and crossing out the common factors. Without complete generation of the polynomials $\kappa_{RL_i}(G)$ or $\kappa(G)$, we immediately see that the resulting expression does not depend on the rate constants r_1, r_2 and r_3.

The coarse-grained description is most instrumental in understanding large non-equilibrium systems when steady-state derivations are difficult or practically impossible. For example, one could study relative responses, expressed through ratios between the steady states of two species, which upon simplification could become decoupled from a subset of reaction rate constants. Biologically important examples for such relative responses are ratios of folded and misfolded protein conformations in proteostasis, where chaperones expend energy to



(a)
G
R
RL
R_i
RL_i
∅
r_1
r_2
r_3
r_4
r_5
r_6
r_7
r_8
(b)
$x^{SS}_{RL_i} = \frac{\kappa v_{RL_i}(G)}{\kappa v_{\emptyset}(G)}$
$\frac{r_6(r_5+r_7)}{r_7r_8+r_5r_8+r_4r_7}$

Figure 4. Simplification of expressions of Kirchhoff polynomials in the coarse-grained representation. (*a*) Simple receptor trafficking model *G* and (*b*) simplification of its steady-state expression. Prime components with equal Kirchhoff polynomials have the same colour; reaction constant r_2 is a label on a nuisance edge marked in black and does not partake in any prime component. Note that vertex labels are not important since they can change during edge contractions. Also, the symbol κ is omitted in front of the graphs for clarity.

alter the ratios [25], and ratiometric mechanisms in signalling, in which the ratio between unoccupied receptors and ligand-bound receptor complexes determines downstream effects [26]. Decoupling can be used to infer the connectivity of reaction networks, or design measurements to focus on (or isolate) the effect of certain reactions on the combinatorially complex steady state of a system.

Let us consider the detailed catalytic cycle of the prostaglandin H synthase 1 (PGHS) from [27] (for description see COX in electronic supplementary material, table S1), whose graph *G* is shown in electronic supplementary material, figure S2*a*, and contains 24 quadrillion spanning trees. We analyse the decoupling from reaction rate constants in all of its possible steady-state ratios by coarse-graining the relevant Kirchhoff polynomials and cancelling the common factors between the numerator and denominator. We find that the ratio between the steady states of species *E*21 and *E*17 (two states in the peroxidase cycle of the enzyme containing the arachidonic acid radical in the cyclooxygenase site and differing by the state of tyrosine 385) contains the fewest number of reaction dependencies—only five (see electronic supplementary material, figure S2*b*)—while the Kirchhoff polynomials $\kappa_{E21}(G)$ and $\kappa_{E17}(G)$ consist of trillions of spanning trees.

4. Compact generation of Kirchhoff polynomials

After simplifying an expression of Kirchhoff polynomials in its coarse-grained form, we find which labels have vanished (the corresponding reactions do not affect the expression) and which ones remain (the corresponding reactions might affect the function modelled by the expression). However, to symbolically obtain the simplified expression for further analysis or repeated evaluation, e.g. for parameter space exploration, we have to explicitly generate full-length Kirchhoff polynomials. The coarse-grained representation is advantageous here as well because we only need to generate the Kirchhoff polynomials for prime graphs with unequal Kirchhoff polynomials.

4.1. Recursive and iterative algorithms

Specifically, we extend the approach of [21] to Kirchhoff polynomial generation, namely, that of algebraic simplification—compression of the polynomial to an equivalent but more compact form. Thus we look for an algorithm *C* that takes a graph *G* as input and produces a representation of its Kirchhoff polynomial $\Gamma_C(\kappa(G))$ of size as small as possible. An ideal algorithm *C* would generate $\Gamma_C(\kappa(G))$ in a maximally compact form, bypassing explicit generation and tedious simplification. However, it is hard to even check if a Kirchhoff polynomial is fully simplified. Therefore, we aim to propose algorithms that, without guarantees for maximal compression, provide satisfactory results to practical problems.

The prime decomposition in [21] behaves as the ideal algorithm *C* for compression—in linear time, it produces a guaranteed maximally compact representation for a graph due to the irreducibility of each prime component. However, it cannot be applied to prime graphs, which can also have sizable Kirchhoff polynomials. To compress the Kirchhoff polynomials of the prime components, we rearrange prime Kirchhoff polynomials, particularly by taking a factor out from part of their monomials, such that we can further factorize parts of them. Without explicit generation, this is achieved through the deletion–contraction identity (equation 2.6), in which the modified graphs $G \setminus e$ and G/e could be amenable to further prime decomposition since *e*'s deletion and contraction could change the connectivity of *G*.

With this insight, we formulate the algorithm C_R (initially presented in [21]; see pseudocode in electronic supplementary material as algorithm 1). It takes a graph *G*, and recursively alternates between prime decomposition and edge deletion–contraction in every prime component until graphs are reduced to a single vertex or a single edge, whose polynomials are trivial to generate. C_R is easy to implement and produces an expression tree as in figure 2*b* that is more compact than the expanded form of the Kirchhoff polynomial. However, multiple recursive calls could unnecessarily work on large graphs with equal Kirchhoff polynomials.

We propose a second, iterative algorithm, C_I (for details and pseudocode see electronic supplementary material, algorithm 2). It employs the graph comparisons certifying Kirchhoff polynomial equality to eliminate the potential redundancy of multiply generating equal Kirchhoff polynomials. In contrast to C_R, C_I associates a unique pointer to every graph under study, and reduced graphs are added to a queue for further reduction, while remembering the partial expression tree they participate in. Then, the algorithm iterates over the graphs in the queue, reducing them only if their Kirchhoff polynomials are distinct from the Kirchhoff polynomials of all graphs already considered. Algebraically, this is equivalent to a change of variables—substituting identical parts of the Kirchhoff polynomial with identical symbols and explicitly generating them only once (see figure 2*c* for an example). The partial expression trees are then assembled to obtain a forest of expression trees marked with the pointers of the initializing graphs as in figure 2*c*. This forest corresponds to a set of Kirchhoff polynomials, which after being substituted into each other, gives rise to the complete Kirchhoff polynomial of graph *G*.

C_I's representation of the Kirchhoff polynomial is more compact than the expanded form and can still be easily evaluated and analysed. However, if there are few small graphs with equal Kirchhoff polynomials encountered during the reduction, compared to C_R, C_I might consume more memory (to remember pointers and already considered graphs), have longer running time (due to equality comparisons), and not provide significantly better compression (compare figure 2*b*, *c*). On the other hand, if the reduction encounters many large graphs with equal Kirchhoff polynomials, only C_I may generate practically relevant Kirchhoff polynomials.

An important ingredient of both algorithms G_R and C_I is the choice of an edge for the deletion–contraction operation (function GetEdgeForDelContr in electronic supplementary material, algorithms 1 and 2). It is unknown which edges to delete–contract to generate a maximally compressed Kirchhoff polynomial [21]. Therefore, we resort to a heuristic approach: we greedily select an edge to delete–contract such that a criterion on the decomposition properties is optimized. Since Kirchhoff polynomial generation results are instance specific, we explore different heuristics (see electronic supplementary material).

4.2. Performance evaluation

For performance analysis, we evaluated the running time and compression of C_R and C_I, where we define *compression* as the ratio of the size of the expanded representation of a Kirchhoff polynomial $|\Gamma_E(\kappa(G))|$ and the size of its representation produced by an algorithm *C*, $|\Gamma_C(\kappa(G))|$. Specifically, we applied 109 heuristics on a collection of example graphs of widely different complexity (see electronic supplementary material, table S1). Ten less complex graph models have tens to millions of spanning trees and two more complex models, HC4 and COXD, have up to quadrillion of spanning trees.

First, we analysed the less complex examples to compare the performance of the different heuristics (see electronic supplementary material for details). For more complex graphs, a random heuristic's performance quickly deteriorates, becoming orders of magnitude worse than heuristics informed by the graph connectivity (electronic supplementary material, figure S5). We normalized the performance measures over all heuristics separately for each example and divided them into groups (see electronic supplementary material, figures S3 and S4). *Post hoc* comparisons of sub-heuristic choices revealed that focusing deletion–contraction on edges relevant to the cycle structure of the graphs, and considering the edge deleted graphs and strongly connected components leads to significantly shorter running time and larger compression on average (see electronic supplementary material, tables S2 and S3).

Figure 5 compares the performance of algorithms C_R and C_I for the connectivity-informed heuristics. The performance data for examples of low complexity lie on or symmetrically around the 45° line, indicating that the two algorithms perform alike. However, the more complex the examples, the more apparent becomes the superiority of algorithm C_I over C_R. The difference is most striking for compression, implying that the change of variables benefits the compression of all models.

Finally, the results for the heuristics leading to the largest compression with C_I (table 1) show that, for larger graphs, the compressed form is orders of magnitude shorter than the number of spanning trees. Therefore, algebraic compressibility, rather than the number of spanning trees, is a hard bound for Kirchhoff polynomial generation. It is an open problem how to determine the compressibility of a graph, but we can get an impression by comparing HC4 and COXD. The compression results are expected because it is difficult to uncover strong connectivity and domination during the graph reduction procedure in dense graphs with many reversible edges; it is simpler to break open cycles in graphs with low density and many unidirectional edges.

4.3. Application to non-equilibrium gene regulation

Example HC4 from table 1 belongs to a family of LFMs used in [9] to explore possible biophysical mechanisms behind the sharp expression profile of the *hunchback* gene as a function of the transcription factor Bicoid in the early *Drosophila* embryo. In this family of hypercube graphs, vertices represent DNA microstates (patterns of a transcription factor (TF) bound to a gene), and edges and edge labels mark TF binding (with rates dependent on TF concentration) and unbinding. Graph topology is determined by the number *n* of TF binding sites at the gene, for example, $n = 3$ corresponds to a cube graph (see electronic supplementary material, figure S6) and $n = 4$ to the four-dimensional hypercube HC4. Under the stochastic interpretation of LFM dynamics, microstate probabilities evolve depending on the transition rates until a steady state is reached.

Estrada *et al*. [9] develop a sharpness analysis for relations between gene expression rate and TF concentration, called GRFs, that are derived from these LFMs. More precisely, GRFs are functions of steady-state microstate probabilities determined by a choice of an expression strategy. For example, in the *all-or-nothing* strategy, transcription is proportional to the steady-state probability of the microstate in which all TF sites are bound; microstate probabilities are, in turn, functions of TF concentrations. After normalization, two features are extracted from a GRF to evaluate its sharpness: (i) *steepness*—the GRF's maximal derivative and (ii) *position*—the TF concentration at which the maximal derivative is attained. A subsequent exploration of the GRF parameter space by a biased sampling algorithm aims to determine the boundaries of the feasible position-steepness region.

An important result of [9] is that energy expenditure is one possible explanation for the observed sharp response expression profiles in development. In particular, at thermodynamic equilibrium GRF position-steepness regions are restricted by the Hill function, which acts as a *Hopfield barrier*, whereas energy expenditure broadens the feasible position-steepness regions and allows for sharper responses. However, due to the large algebraic complexity of non-equilibrium steady states, the non-equilibrium position-steepness analysis in [9] is limited to models with up to $n = 3$ sites, while the *hunchback* P2 enhancer has 5–7 Bicoid binding sites.

The 2000-fold compression of the $n = 4$ sites model HC4 allows us to extend the non-equilibrium case analysis and explore how the number of binding sites affects the position-steepness regions. We focus on the all-or-nothing expression strategy and models with $n = 2$, 3, 4 binding

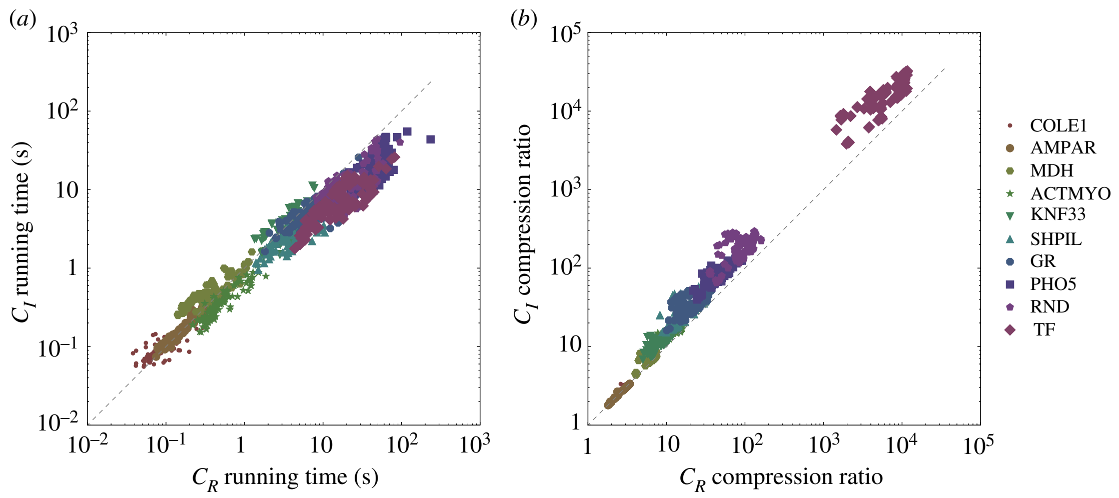


Figure 5. Running time (*a*) and compression (*b*) of algorithms C_R and C_I on the collection of less complex examples in electronic supplementary material, table S1. Each point represents the running time/compression for an example graph obtained by C_R and C_I using the same connectivity-informed heuristic. The dashed line marks equal performance for C_R and C_I. Examples are sorted by complexity (number of spanning trees): COLE1 has the lowest complexity and TF the highest.

Table 1. Performance of algorithm C_I with the heuristics leading to the largest compression on a set of example graphs (see graph descriptions in electronic supplementary material, table S1): size of the expression tree of the expanded Kirchhoff polynomial $|\Gamma_E(\kappa(G))|$, size of the compressed expression tree $|\Gamma_{C_I}(\kappa(G))|$, heuristic $\mathcal{H}$, compression calculated as the ratio $|\Gamma_E(\kappa(G))|/|\Gamma_{C_I}(\kappa(G))|$, and average running time in seconds obtained from 10 runs (one run for HC4 and COXD) of KirchPy on a Dell laptop with Intel i-7 CPU@2.10 GHz and 8 GB RAM.

G	$\|\Gamma_E(\kappa(G))\|$	$\|\Gamma_{C_I}(\kappa(G))\|$	$\mathcal{H}$	compression	time (s)
COLE1	157	46	2012	3.4	0.09
AMPAR	211	63	2102	3.3	0.07
MDH	1270	141	2001	9.0	0.27
ACTMYO	3561	142	3003	25.1	0.19
KNF33	15 553	940	3114	16.5	1.41
SHPIL	45 601	786	2204	58.0	1.58
GR	65 742	1280	2102	51.4	1.63
PHO5	640 513	4691	3215	136.5	10.61
RND	967 681	3281	1011	294.9	7.53
TF	38 746 801	1191	2002	32 533	1.84
HC4	679 477 249	333 599	1001	2036	1797
COXD	367 647 474 647 060 221	89 532	1010	4.1×10^{12}	661

sites whose non-dimensionalized kinetic parameters are sampled in the range $[10^3, 10^{-3}]$. We obtain position-steepness boundaries as described in [9], with differences mentioned in electronic supplementary material. Our results are shown in figure 6 and indicate that the $n = 3$ and $n = 4$ site models can achieve steepness of around 5.5 and 6.4, respectively, both exceeding the experimentally fitted Hill coefficient of 5 to the *hunchback* expression profile in response to Bicoid. Interestingly, $n = 4$ site models can generate GRFs with position values greater than 1, which are not attainable by Hill functions, equilibrium GRFs (since they are bounded by Hill functions acting as a Hopfield barrier), and non-equilibrium models with a lower number of sites. A normalized position value of 1 corresponds to a TF concentration at which the GRF is half-maximal, suggesting that $n \geq 4$ binding sites permit a wider class of GRF shapes in which the maximal steepness arises at TF concentrations larger or equal to the half-maximal TF concentration (see examples in electronic supplementary material, figure S7). Note that the increase of the position-steepness area from $n = 2$ to $n = 3$ sites is much larger than from $n = 3$ to $n = 4$ sites. We believe that, because of the vastly different dimensions of the sampled parameter spaces (six, 22 and 62 free parameters for models with $n =$ 2, 3 and 4 sites, respectively), the more the sites, the less precise the boundaries.

5. Conclusion

Here, we concentrated on biochemical models falling under the linear framework [7] and took an algebraic graph theory approach to describe their non-equilibrium steady states. The convenient correspondence between graphs and polynomials was essential in the development of theory and algorithms allowing us to manipulate and compress the combinatorially complex expressions. In particular, our

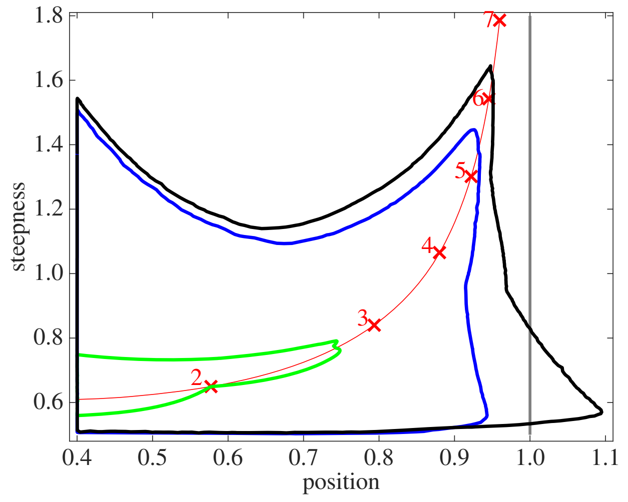


Figure 6. Position-steepness regions for non-equilibrium gene response functions corresponding to models with $n = 2$ (green), $n = 3$ (blue), $n = 4$ (black) transcription factor binding sites. The boundaries are obtained for the all-or-nothing expression strategy by sampling parameter values in the interval $[10^3, 10^{-3}]$. The *Hill line* defined by the position-steepness loci of Hill functions with coefficients ranging from 1.5 to 7 is shown in red; loci corresponding to integer Hill coefficients are marked with red crosses and numbers. The position asymptote of the Hill line is marked in grey.

coarse-grained representation permits the manipulation of otherwise symbolically intractable expressions of Kirchhoff polynomials. It also helps establish a structure–function relationship between a model and its steady-state response by identifying which reactions do not partake in the expression due to algebraic simplification and, in some cases, which reactions participate in it due to irreducibility.

To explicitly generate individual Kirchhoff polynomials from a simplified coarse-grained expression, our two proposed algorithms produce compressed polynomials that are algebraically equivalent to their fully expanded counterparts. We demonstrated the practical utility of the algorithms for a wide range of graph examples. The large compression results affirm the finding from [21] that Kirchhoff polynomial generation depends on graph connectivity and not on the (super) exponentially growing number of spanning trees. This calls for a more in-depth characterization of Kirchhoff polynomial compressibility based on connectivity. Additionally, compression allowed us to expand the non-equilibrium gene regulation analysis of [9] and conclude that with four transcription factor binding sites qualitatively different shapes of GRFs can be obtained.

A direction for improvement of the manipulation and simplification tools is the further development of Kirchhoff polynomial equality conditions, since the ones we present are not simultaneously necessary and sufficient, such that we cannot, in general, guarantee to identify all graphs giving rise to equal Kirchhoff polynomials in a coarse-grained expression. We anticipate that developments in Kirchhoff polynomial isomorphism similar to those for undirected graphs [28] and optimized deletion-contraction heuristics fuelled by recent advances in strong connectivity and 2-connectivity [29] could further improve the performance of our compression algorithms. Overall, we believe that our implementation of the methods in the package KirchPy, together with the theoretical insights into manipulation and generation of Kirchhoff polynomials, would (i) allow modelling and analysis efforts to catch up with the ever more comprehensive experimental data by promoting the construction and analysis of larger LFMs, (ii) enable the analysis of the functional significance of simplifications in classes of models, which can be useful in experimental design as well as for studying phenomena such as proteostasis, ratiometric and differential signalling, (iii) find applications beyond biology because of the equivalence of LFMs to continuous-time Markov chains, linear compartmental models, and linear non-negative systems in general, and (iv) offer an alternative for steady-state derivations that is more convenient than the direct application of the Matrix-Tree Theorem through naive spanning tree enumeration.

Data accessibility. The presented manipulation and generation theory and algorithms are implemented in the Python package KirchPy, which is available at https://gitlab.com/csb.ethz/KirchPy.

Authors' contributions. Conceptualization, P.Y. and J.S.; methodology and algorithms, P.Y.; investigation, P.Y.; writing, P.Y. and J.S. All authors gave final approval for publication.

Competing interests. We declare we have no competing interests.

Funding. This work has been supported by the EU FP7 project IFNAction (contract no. 223608).

Acknowledgements. We are grateful to Przemysław Uznański for insightful discussions and to Jeremy Gunawardena for critical comments and valuable suggestions.

References

1. Leighton F, Rivest R. 1986 The Markov chain tree theorem. MIT/LCS/TM-249, Laboratory for Computer Science, MIT, Cambridge, MA, 1983. Also in IEEE Transactions on Information Theory, IT-37 (6).
2. Biane P. 2015 Polynomials associated with finite Markov chains. In *In Memoriam Marc Yor-Séminaire de Probabilités XLVII*, pp. 249–262. Cham, Switzerland: Springer. See https://link.springer.com/chapter/10.1007/978-3-319-18585-9_12.
3. Weinzierl S. 2013 Feynman graphs. In *Computer algebra in quantum field theory* (eds C Schneider, J Blümlein). Texts & Monographs in Symbolic Computation (A Series of the Research Institute for Symbolic Computation, Johannes Kepler University, Linz, Austria). Vienna, Austria: Springer. See http://dx.doi.org/10.1007/978-3-7091-1616-6_16.
4. Farina L, Rinaldi S. 2000 *Positive linear systems: theory and applications*, vol. 50. New York, NY: John Wiley & Sons.
5. Haddad WM, Chellaboina V, Hui Q. 2010 *Nonnegative and compartmental dynamical systems*. Princeton, NJ: Princeton University Press.
6. Chung F, Yang C. 2000 On polynomials of spanning trees. *Ann. Comb.* **4**, 13–25. (doi:10.1007/PL00001273)
7. Gunawardena J. 2012 A linear framework for time-scale separation in nonlinear biochemical systems. *PLoS ONE* **7**, e36321. (doi:10.1371/journal.pone.0036321)
8. Mirzaev I, Gunawardena J. 2013 Laplacian dynamics on general graphs. *Bull. Math. Biol.* **75**, 2118–2149. (doi:10.1007/s11538-013-9884-8)
9. Estrada J, Wong F, DePace A, Gunawardena J. 2016 Information integration and energy expenditure in gene regulation. *Cell* **166**, 234–244. (doi:10.1016/j.cell.2016.06.012)
10. Tutte WT. 1948 The dissection of equilateral triangles into equilateral triangles. In *Mathematical Proc. of the Cambridge Phil. Soc.*, pp. 463–482, vol. 44. Cambridge, UK: Cambridge University Press.
11. Ahsendorf T, Wong F, Eils R, Gunawardena J. 2014 A framework for modelling gene regulation which accommodates non-equilibrium mechanisms.

BMC Biol. **12**, 102. (doi:10.1186/s12915-014-0102-4)

12. Qi F, Dash RK, Han Y, Beard DA. 2009 Generating rate equations for complex enzyme systems by a computer-assisted systematic method. *BMC Bioinf.* **10**, 238. (doi:10.1186/1471-2105-10-238)
13. King EL, Altman C. 1956 A schematic method of deriving the rate laws for enzyme-catalyzed reactions. *J. Phys. Chem.* **60**, 1375–1378. (doi:10.1021/j150544a010)
14. Chou KC, Forsén S. 1980 Graphical rules for enzyme-catalysed rate laws. *Biochem. J.* **187**, 829–835. (doi:10.1042/bj1870829)
15. Varon R, Garcia-Sevilla F, Garcia-Moreno M, Garcia-Canovas F, Peyro R, Duggleby RG. 1997 Computer program for the equations describing the steady state of enzyme reactions. *Comp. Appl. Biosci. CABIOS* **13**, 159–167. (doi:10.1093/bioinformatics/13.2.159)
16. Indge KJ, Childs R. 1976 A new method for deriving steady-state rate equations suitable for manual or computer use. *Biochem. J.* **155**, 567–570. (doi:10.1042/bj1550567)
17. Uno T. 1996 An algorithm for enumerating all directed spanning trees in a directed graph. In *Algorithms and computation* (eds T Asano, Y Igarashi, H Nagamochi, S Miyano, S Suri), vol. 1178 of LNCS, pp. 166–173. New York, NY: Springer.
18. Kapoor S, Ramesh H. 2000 An algorithm for enumerating all spanning trees of a directed graph. *Algorithmica* **27**, 120–130. (doi:10.1007/s004530010008)
19. Yordanov P, Stelling J. 2018 Steady-state differential dose response in biological systems. *Biophys. J.* **114**, 723–736. (doi:10.1016/j.bpj.2017.11.3780)
20. Gnad F, Gunawardena J, Mann M. 2011 PHOSIDA 2011: the posttranslational modification database. *Nucleic Acids Res.* **39**(suppl. 1), D253–D260. (doi:10.1093/nar/gkq1159)
21. Mihalák M, Uznański P, Yordanov P. 2016 Prime factorization of the Kirchhoff polynomial: compact enumeration of arborescences. In *Proc. of the Thirteenth Workshop on Analytic Algorithmics and Combinatorics (ANALCO 2016)*, pp. 93–105. Philadelphia, PA: Society for Industrial and Applied Mathematics Publications.
22. Tomida T, Hirose K, Takizawa A, Shibasaki F, Iino M. 2003 NFAT functions as a working memory of Ca^{2+} signals in decoding Ca^{2+} oscillation. *EMBO J.* **22**, 3825–3832. (doi:10.1093/emboj/cdg381)
23. Mirzaev I, Bortz DM. 2015 Laplacian dynamics with synthesis and degradation. *Bull. Math. Biol.* **77**, 1013–1045. (doi:10.1007/s11538-015-0075-7)
24. Levine L. 2011 Sandpile groups and spanning trees of directed line graphs. *J. Comb. Theory Ser. A* **118**, 350–364. (doi:10.1016/j.jcta.2010.04.001)
25. Barducci A, Rios PDL. 2015 Non-equilibrium conformational dynamics in the function of molecular chaperones. *Curr. Opin Struct. Biol.* **30**, 161–169. Folding and binding/nucleic acids and their protein complexes. See http://www.sciencedirect.com/science/article/pii/S0959440X15000172 10.1016/j.sbi.2015.02.008.
26. Bush A, Vasen G, Constantinou A, Dunayevich P, Patop IL, Blaustein M, Colman-Lerner A. 2016 Yeast GPCR signaling reflects the fraction of occupied receptors, not the number. *Mol. Syst. Biol.* **12**, 898. (doi:10.15252/msb.20166910)
27. Goltsov A, Lebedeva G, Humphery-Smith I, Goltsov G, Demin O, Goryanin I. 2010 *In silico* screening of nonsteroidal anti-inflammatory drugs and their combined action on prostaglandin H synthase-1. *Pharmaceuticals* **3**, 2059–2081. (doi:10.3390/ph3072059)
28. Bogner C, Weinzierl S. 2010 Feynman graph polynomials. *Int. J. Mod. Phys. A* **25**, 2585–2618. (doi:10.1142/S0217751X10049438)
29. Georgiadis L, Italiano GF, Parotsidis N. 2017 Strong connectivity in directed graphs under failures, with applications. In *Proc. of the Twenty-Eighth Annual ACM-SIAM Symp. on Discrete Algorithms*, pp. 1880–1899. See http://epubs.siam.org/doi/abs/10.1137/1.9781611974782.123.