

Chemical mass-action systems as analog computers: implementing arithmetic computations at specified speed

David F. Anderson* and Badal Joshi†

April 9, 2024

Abstract

Recent technological advances allow us to view chemical mass-action systems as analog computers. In this context, the inputs to a computation are encoded as initial values of certain chemical species while the outputs are the limiting values of other chemical species. In this paper, we design chemical systems that carry out the elementary arithmetic computations of: identification, inversion, m th roots (for $m \geq 2$), addition, multiplication, absolute difference, rectified subtraction over non-negative real numbers, and partial real inversion over real numbers. We prove that these “elementary modules” have a speed of computation that is independent of the inputs to the computation. Moreover, we prove that finite sequences of such elementary modules, running in parallel, can carry out composite arithmetic over real numbers, also at a rate that is independent of inputs. Furthermore, we show that the speed of a composite computation is precisely the speed of the slowest elementary step. Specifically, the scale of the composite computation, i.e. the number of elementary steps involved in the composite, does not affect the overall asymptotic speed – a feature of the parallel computing nature of our algorithm. Our proofs require the careful mathematical analysis of certain non-autonomous systems, and we believe this analysis will be useful in different areas of applied mathematics, dynamical systems, and the theory of computation. We close with a discussion on future research directions, including numerous important open theoretical questions pertaining to the field of computation with reaction networks.

1 Introduction

An increasingly active area of scientific research is the study of chemical reaction systems as computational machines [1, 2, 3, 4, 5, 6, 7, 8, 9]. The broad goal of this nascent field is to develop systems that can operate in the niche of a (wet) cellular environment, rather than to directly compete with modern digital computers. Methods for the physical implementation of these systems include DNA-strand displacement, which was developed largely by Lulu Qian, David Soloveichik, and Erik Winfree [7, 9]. While this research area is still in its relative infancy, it is critical that the theoretical (mathematical) foundation of these methods and systems keeps pace with the physical implementations so as to (i) keep the algorithmic aspects of this field on a firm mathematical footing from the beginning, and (ii) provide a reliable (proof based) method for the development of the appropriate reaction network-based software design.

Under the assumptions that the chemical system is well-stirred and at a constant temperature, deterministic mass-action chemical systems are mathematically modeled as autonomous systems

*Department of Mathematics, University of Wisconsin-Madison, anderson@math.wisc.edu

†Department of Mathematics, California State University San Marcos, bjoshi@csusm.edu.

of ordinary differential equations [10]. To utilize such a system as a computational machine, we associate the inputs and outputs of the desired computation with certain state variables of the dynamical system. Most often, the desired output is encoded as a limiting value of a subset of the state variables that converge asymptotically to this limiting value. Since by necessity any recording of the system is at a finite time it is important to bound the error, which in this case is the deviation from the asymptotic value. The rate of this convergence, or alternatively the rate at which the error decays, determines the speed of the computation. The time chosen for the recording will in general be determined by the required level of accuracy, but should not be so large as to render the method impractical.

Because mass-action systems are modeled via systems of differential equations, they can naturally be viewed as analog computers. While digital computers now dominate the computational landscape in our society, it is interesting to note that analog computers have certain advantages over digital computers. For example, consider the problem of numerically solving the initial value problem

$$\dot{x}(t) = 1 - x(t), \quad x(0) = x_0 \in \mathbb{R}_{>0}, \quad (1)$$

If you were using a digital computer, you would naturally utilize some sort of discretization method (such as Euler’s method or a Runge-Kutta method) to approximate the solution to some desired accuracy. No matter the method, however, the key point is that one *must* discretize, as digital computers can only handle discrete computations. An incredible amount of scientific effort has gone into discretization methods for the numerical solution of ODEs, PDEs, and SDEs via digital computers over the past many decades, with such work still dominating portions of applied mathematics.

On the other hand, the system (1) above is quite naturally “solved” by simply considering the chemical reaction network modeled via



with an initial concentration of x_0 , so long as the rate constants are chosen to be equal to one (see section 2.1 for the details of mass-action models). In short: many problems that are difficult to solve for digital computers are quite simple to solve for a chemical system.

However, there is a natural converse to the above observation. While chemical systems will naturally be able to solve systems that are described via differential equations (see section 2.1 for the exact class of differential equations they solve), it is more difficult to use them to compute things that are naturally discrete, such as basic arithmetic (addition, multiplication, finding roots, division, etc.). Hence, while it took much thought by many people to develop algorithmic methods for the solution of continuous problems with digital computers, it is natural that we will require a fair amount of work to develop efficient methods for the use of chemical systems to solve these discrete problems.

We therefore focus here on algorithmic aspects of chemistry-based analog computing of arithmetic operations, as these operations are a cornerstone of discrete computation. Specifically, we focus on the development of elementary “chemical modules” that carry out basic arithmetic operations (such as inversion, roots, addition, multiplication, etc.) and we prove that the developed modules have a speed of computation that is independent of the inputs.¹ Moreover, we prove that these elementary modules can be run in parallel so as to carry out any desired arithmetic computation, also at speed that is independent of input.

There is existing work on algorithmic aspects of analog computation of arithmetic operations, even from the specific vantage point of reaction network implementation. Our work is closest

¹More specifically, we prove that the speed of computation for the elementary modules have a lower bound that is independent of inputs.

in approach to the one taken by [2]. However, while the authors of [2] considered algorithms for arithmetic computations using reaction networks, they did not include *speed of computation* in their consideration. Our stated purpose is to remedy this. We frame a general arithmetic computation as a finite sequence of elementary computations that are running in parallel. Since every element of the sequence is being computed in parallel, we need to consider time-evolving inputs to the elementary modules, where the output of a previous step is providing input into the next step even without having converged to its asymptotic value. We show input-independence of speed of any overall computation that is composed from multiple, but finitely many, elementary arithmetic operations.

Throughout this work, we assume that the rate constants of our models (which are the key system parameters, see section 2.1) are all exactly equal to 1. Our main results of this paper, Theorem 5.1 and Corollary 5.2, then concludes that any arithmetic computation carried out in the manner specified herein, even those consisting of a finite number of composite calculations being carried out in parallel, has a rate of convergence of at least 1 (loosely, this means the deviation of the model from the desired output decreases like e^{-t}). While it is important to our work that all the rate constants be equal, the choice of setting them equal to 1 is arbitrary. We discuss in Remark 5.4 how setting them all to $\sigma > 0$ is simply a time-change of our model and leads to a rate of convergence of σ . We discuss the possibility of generalizing to the situation of non-equal rate constants in section 7.

There are several bodies of research, outside the world of chemical reactions, on designing efficient hardware and software for analog as well as analog-digital hybrid computing [11, 12, 13, 14]. While we develop our results for a chemistry-based computer, the results apply equally well to any computational system that computes via nonlinear polynomial differential equations. We therefore hope that our theoretical results will find use in the broader sphere of computation.

The remainder of the paper is organized as follows. In the next subsection, section 1.1, we point out that there are two natural ways to use chemical systems as computational machines and compare/contrast their strengths and weaknesses. In section 2, we (i) formally introduce the mathematical models that will serve as our algorithms for analog computation, which are deterministic chemical reaction networks with mass-action kinetics, (ii) discuss issues related to these models, and (iii) demonstrate, via example, the central purpose of this paper: the development of reaction network modules that can carry out arithmetic at speeds that are independent of inputs. In section 3, we provide the main mathematical analysis of this paper. In particular, we provide a rather detailed analysis of certain non-autonomous dynamical systems. The work of section 3 is general enough that we hope and expect it to have applications in other areas of applied mathematics. In section 4, we provide the chemical reaction network modules that carry out the basic elementary operations discussed above (identification, addition, multiplication, inversion, etc.) and use the mathematical results of section 3 to prove that they have computational speeds that are independent of inputs. In section 5, we demonstrate how any calculation carried out via a combination of elementary operations using our constructions also has a computational speed that is independent of inputs. In section 6, we demonstrate how to expand the constructions to handle calculations involving all real numbers (as opposed to non-negative numbers). Finally, in section 7 we provide a discussion, including many directions for future research.

1.1 Arithmetic with reaction networks: comparing existing schemata

There are two distinct schemata that have been used for carrying out arithmetic with reaction networks, which we refer to as ‘conservation law based’ (for instance in [4, 15]) and ‘positive steady state based’ (for instance in [2] and the current paper). To understand one way that addition could

be performed in the conservation law based approach, we consider a simple analogy. Suppose we have two ‘input’ buckets and one ‘output’ bucket. The contents of the buckets may be discrete (e.g., marbles) or continuous (e.g., water). Pouring the contents of the two input buckets into an initially empty output bucket will result in the sum of the two inputs as the eventual volume of the output bucket. A reaction network based on this simple idea is



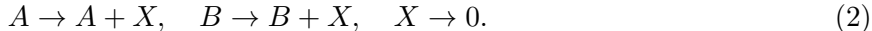
Here A and B are input species while X is the output species. Clearly, if the initial population of X , represented by x , is 0, we will have that, asymptotically, x converges to the sum of the initial populations of the input species:

$$\lim_{t \rightarrow \infty} x(t) = a(0) + b(0).$$

This is true regardless of the specific choice of rate constants for the chemical reactions (see section 2.1). In fact, it is true independent of whether the underlying model is discrete (integer counts) or continuous (concentrations), deterministic or stochastic. Consequently, the conservation law based schema has the potential to be used in both digital as well as analog computing.

Despite the simplicity and potential use in both digital- and analog-computing, the conservation law based schema does have significant drawbacks which we now discuss. Perhaps the most significant one is the ‘loss of memory’ suffered by the input species. At the end of the addition computation, all memory of what the inputs contained is lost forever. In fact, if addition is one of several elementary steps in a longer sequence of computations, then all inputs and all intermediate steps are forgotten. Only the final output is retained. This may become an issue if the same input is required multiple times. The positive steady state based schema utilized in the present paper does not suffer from this shortcoming. For every elementary operation, we explicitly require that all inputs are preserved while the output is computed and stored as the concentration of a dedicated new species.

To illustrate the ‘positive steady state based’ schema utilized in this paper let us once again consider the operation of addition, now accomplished with the reaction network



Assuming rate constants that are equal to one, the mass action differential equation for these reactions is (see section 2.1)

$$\dot{x}(t) = a(t) + b(t) - x(t).$$

We first consider the case where the reactions in (2) stand in isolation. Note that the concentrations of A and B are not changed by any of the reactions in (2). In particular, this means that the input species A and B retain their initial values, $a(t) = a(0)$ and $b(t) = b(0)$ for all time $t \geq 0$. Meanwhile, the concentration of the output species X asymptotically converges to $a(0) + b(0)$. Now imagine that the reactions in (2) are only some reactions in a much larger network. The inputs A and B may be outputs of some other subnetwork and may as yet be in the process of converging to their asymptotic values. Then, assuming the limits on the right exist, we have

$$\lim_{t \rightarrow \infty} x(t) = \lim_{t \rightarrow \infty} (a(t) + b(t)).$$

This example illustrates how an output from one computation may be fed into another computation as well as how inputs may be simultaneously used in several different computations since they are not degraded.

2 Introducing the basic problem

In this paper, we consider the task of carrying out the following common arithmetic functions via deterministic mass-action chemical systems: addition, multiplication, inversion, division, m th root (for integers $m \geq 2$), absolute difference of two numbers, the maximum of two numbers, and subtraction. Our particular goal, which we will describe more fully below, is to provide mass-action chemical reaction network modules for each such operation whose “speed of computation” is independent of the inputs (and hence outputs) to the problem *and* which can be combined to run in parallel so that any sequence of arithmetic computations (e.g., the square root of the inverse of the sum of two numbers: $\sqrt{1/(a+b)}$) will have a speed of computation that is also independent of inputs.

There are a number of questions/issues we wish to highlight from the start and are the focus of the present section.

1. What is the class of differential equation models we use in this paper? Said differently, what class of differential equations arise from deterministically modeled mass-action chemical systems?
2. Chemical concentrations are naturally non-negative. Hence, it is natural to consider problems such as the addition or division of two positive reals since the inputs and outputs are positive reals. However, how should one handle the situation in which one or more of the inputs are negative? Similarly, how should one handle the operation of minus, whose output can be negative even when the inputs are both positive?
3. Is it possible to ensure that the speed of the computation does not depend on the inputs and/or outputs to the problem? It is the recognition of this problem and its resolution in the affirmative that we consider to be the main contribution of this paper.

We handle each of the above questions in turn before moving to our general theory and constructions. Before proceeding, however, we first note that efficiency, accuracy, and speed are essential, pragmatic considerations when implementing computers. We need some assumptions, however, regarding the interface with a chemistry-based “CPU.” Specifically, we assume that we have (i) the ability to initiate chemical species with arbitrary precision, and (ii) the ability to measure chemical species with arbitrary precision.

2.1 Mass-action chemical systems and their associated ODEs

Consider an ordinary differential equation of the form

$$\dot{x}(t) = f(x(t)),$$

with $x(t) \in \mathbb{R}^d$ and $f : \mathbb{R}^d \rightarrow \mathbb{R}^d$. When does such an equation arise from a mass-action model of a chemical system?

Before answering, we first introduce mass-action models. Consider a finite number of chemical “species,” which we will denote by $\{X_1, \dots, X_d\}$, undergoing transformations via a finite number of possible reaction types. Reactions can be understood intuitively: they require certain numbers of the different species as inputs, and they output a different number of the various species. They are most naturally visualized in graphical format. For example, consider a system with three species,

$\{X_1, X_2, X_3\}$, and the following three reaction types



In the first reaction, one molecule of X_1 and one molecule of X_2 are converted to one molecule of X_3 . The second reaction reverses the first. The third reaction consists of the destruction of two X_3 molecules (perhaps they merge to become a fourth type of species that we are not modeling). In general, a chemical system will have a finite number of such reactions of the form



where $\nu, \nu' \in \mathbb{Z}_{\geq 0}^d$ and by $\nu \cdot X$ we mean the dot product: $\nu \cdot X = \nu_1 X_1 + \cdots + \nu_d X_d$. Note that if we were explicitly counting the numbers of molecules of each species, then one instance of the reaction (4) would change the system by addition of the vector $\nu' - \nu \in \mathbb{Z}^d$.

A differential equation model for the chemical concentrations arises by associating to each reaction a rate function. For example, if we denote the above reaction (4) by $\nu \rightarrow \nu'$ and denote the rate of that reaction when the system is in state $x \in \mathbb{R}_{\geq 0}^d$ by $\lambda_{\nu \rightarrow \nu'}(x)$, then the differential equation governing the dynamics is

$$\dot{x}(t) = \sum_{\nu \rightarrow \nu'} \lambda_{\nu \rightarrow \nu'}(x(t))(\nu' - \nu), \tag{5}$$

where the sum is over all reaction types (enumerated by $\nu \rightarrow \nu'$). The system is said to be governed by *mass-action* kinetics when the functions $\{\lambda_{\nu \rightarrow \nu'}\}$ have the following form,

$$\lambda_{\nu \rightarrow \nu'}(x) = c_{\nu \rightarrow \nu'} \prod_{i=1}^d x_i^{\nu_i},$$

where each $c_{\nu \rightarrow \nu'}$ is a positive constant and where we interpret 0^0 to be 1. Hence, for mass-action kinetics the rate of a reaction is a monomial in which the exponent for the i th species is the coefficient of X_i in $\nu \cdot X$. For example, assuming mass-action kinetics the differential equations governing the system (3) are

$$\dot{x}(t) = c_1 x_1 x_2 \begin{pmatrix} -1 \\ -1 \\ 1 \end{pmatrix} + c_2 x_3 \begin{pmatrix} 1 \\ 1 \\ -1 \end{pmatrix} + c_3 x_3^2 \begin{pmatrix} 0 \\ 0 \\ -2 \end{pmatrix},$$

where c_1, c_2 , and c_3 are some positive constants, and where we enumerated the constants sequentially (as opposed to the reaction $\nu \rightarrow \nu'$). It is common to incorporate the rate constants into the network graph, such as (3), by placing them next to the associated arrow.

We can now answer the first question posed above. If $f : \mathbb{R}^d \rightarrow \mathbb{R}$ satisfies the two properties below, then there is a mass-action chemical system whose associated ODE (5) is given by $\dot{x}(t) = f(x(t))$.

1. The function f is a polynomial. That is, for each $i \in \{1, \dots, d\}$,

$$f_i(x) = \sum_{a_1=0}^N \cdots \sum_{a_d=0}^N c_{i,a_1,\dots,a_d} x_1^{a_1} \cdots x_d^{a_d}, \tag{6}$$

where $N < \infty$ is a maximum exponent in the model, and the c_{i,a_1,\dots,a_d} are constants (which can be negative or zero).

2. If $c_{i,a_1,\dots,a_d} < 0$ in (6), then $a_i \geq 1$ for the associated monomial. In plain English, x_i must appear in any monomial of $f_i(x)$ that has a negative coefficient.

Note that the final condition guarantees that the non-negative orthant $\mathbb{R}_{\geq 0}^d$ is forward invariant under the dynamics. That is, so long as $x(0) \in \mathbb{R}_{\geq 0}^d$, then $x(t) \in \mathbb{R}_{\geq 0}^d$ for all $t \geq 0$.

It should be clear that any mass-action system satisfies the two conditions above. To see the converse, that is to see that for any dynamical system $\dot{x}(t) = f(x(t))$ satisfying the conditions above there is a mass-action chemical system with f as the governing dynamics, see [16] or, for a more modern treatment, [10].

2.2 The issue of negativity and the dual rail representation

Chemical concentrations are non-negative, but we wish to be able to represent negative values within our framework. To do so, we represent real numbers as the difference between two given chemical species. This is sometimes referred to as the *dual rail representation* of real numbers. The introduction we give to this topic in this section will be brief. See, for example, [4] for a more thorough discussion.

One of the early works to utilize the dual rail representation is [5], where, for example, they labeled Y^P and Y^C the amounts that a particular species, Y , had been “P”roduced and “C”onsumed, respectively. Then, the total amount of Y was $Y^P - Y^C$. In this example, if, for example, $Y^P = 10$ and $Y^C = 8$, then the total amount of Y is clearly 2. However, we could also represent the value 2 by having $Y^P = 2 + c$ and $Y^C = c$ for any $c \geq 0$. Thus, *how* we represent a given real number is a bit ambiguous, a priori.

For our purposes, we wish to do away with such ambiguity. Hence, we will split a number into its positive and negative parts. That is, we define the *dual rail map* $\varphi : \mathbb{R} \rightarrow \mathbb{R}_{\geq 0}^2$ via

$$a \mapsto (a_p, a_n) = \begin{cases} (a, 0), & \text{if } a > 0, \\ (0, -a), & \text{if } a \leq 0. \end{cases}$$

Here a_p and a_n are referred to as the *positive* and *negative parts*, respectively, of a . The map φ is injective and

$$\text{Im}(\varphi) = \{(a_p, a_n) \in \mathbb{R}_{\geq 0}^2 \mid a_p a_n = 0\} = \{(a_p, a_n) \in \mathbb{R}_{\geq 0}^2 \mid a_p = 0 \text{ or } a_n = 0\}.$$

Define the map $\psi : \mathbb{R}_{\geq 0}^2 \rightarrow \mathbb{R}$ via $(a_p, a_n) \mapsto a = a_p - a_n$. The map resulting from restricting the domain of ψ to $\text{Im}(\varphi)$ is the inverse of φ . We will refer to (a_p, a_n) as the canonical representation of a when $a_p a_n = 0$, and write $a = (a_p, a_n)$, for brevity.

2.3 Input independent speed of computation

Suppose we want to compute the reciprocal of a positive real $a \in \mathbb{R}_{> 0}$ as the long-term solution of a mass-action differential equation. A natural attempt would be to consider the reaction network



whose mass-action differential equation, assuming all reaction rate constants are equal to 1 and that the concentration of species A is a , is

$$\dot{x} = 1 - ax.$$

The differential equation has a unique solution for all time given by

$$x(t) = \frac{1}{a} + \left(x(0) - \frac{1}{a}\right) e^{-at}.$$

Clearly, $x(t) \xrightarrow{t \rightarrow \infty} 1/a$. Moreover, since a is the coefficient of t in the exponential decay e^{-at} , the rate of convergence to the long term solution is a . The rate of convergence is directly related to the speed of computing the reciprocal. Indeed, suppose that T_n is the smallest time required to compute $1/a$ accurately up to $n - 1$ digits. Then we have that

$$|x(T_n) - 1/a| = |x(0) - 1/a| e^{-aT_n} = 10^{-n},$$

which after applying logarithms and rearranging gives us

$$T_n = \frac{1}{a} \ln(|x(0) - 1/a|) + \frac{n \ln(10)}{a}.$$

In particular, we have that

$$\lim_{n \rightarrow \infty} \frac{T_n}{n \ln(10)} = \frac{1}{a}, \quad (7)$$

so that the time required to compute n decimal digits of $1/a$ scales not only with n but with the output $1/a$. Alternatively, the reciprocal

$$\lim_{n \rightarrow \infty} \frac{n \ln(10)}{T_n} = a, \quad (8)$$

gives the speed of computation. We will say that the variable x *computes the reciprocal of a at speed a* .

We will now show that we can remove this dependence of speed of computation of the reciprocal on the input (and output) to the system. Then, over the rest of the paper, we will show that this input-independence can be achieved for an arbitrary computation.

Consider the following reaction network



whose mass-action differential equation, assuming rate constants are equal to 1 and that the concentration of species A is a , is

$$\dot{x} = x(1 - ax). \quad (10)$$

For $x(0) = x_0$, the unique solution for all time $t \geq 0$ is given by

$$x(t) = \frac{x_0/a}{(1/a - x_0)e^{-t} + x_0},$$

and so the distance from the attracting steady state $1/a$ is

$$|x(t) - 1/a| = \left| \frac{x_0/a}{(1/a - x_0)e^{-t} + x_0} - \frac{1}{a} \right| = \frac{1}{a} \left(\frac{|x_0 - 1/a|}{x_0 e^t - (x_0 - 1/a)} \right).$$

This converges to zero like e^{-t} , and after a brief calculation, it is straightforward to see that

$$\lim_{n \rightarrow \infty} \frac{n \ln(10)}{T_n} = 1, \quad (11)$$

where, again, T_n is the smallest time needed to compute $1/a$ to $n - 1$ decimal places. That is, x computes the reciprocal of a at speed 1. A comparison of equations (7) and (11) highlights the input independence of the reciprocal computation in the latter construction.

3 Rate of convergence for certain non-autonomous systems

In this section, which holds much of the mathematics, we provide a rather detailed analysis of certain non-autonomous dynamical systems. These systems will have a structure like

$$\dot{x}(t) = f(g_1(t), g_2(t), x(t)),$$

where g_1 and g_2 are best viewed as forcing functions that are themselves the output from another such system. We will prove that for the particular systems we consider the following holds: global exponential convergence of the forcing functions, such as g_1 and g_2 , imply that the system solution also enjoys global exponential convergence. That is, that property is “fed forward” from the forcing functions to the solution. We will also demonstrate that, in most cases, the rate of convergence for the system solution is at least as large as the minimum rate of convergence of the forcing functions.

To make these notions precise we provide here a definition and a lemma.

Definition 3.1. Let $g : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$ be a real-valued function that converges to a real number $g^* \in \mathbb{R}$. The *rate of convergence* of g to g^* is defined to be

$$\rho = -\limsup_{t \rightarrow \infty} \frac{\ln |g(t) - g^*|}{t} \quad (12)$$

whenever $\rho \in (0, \infty]$. Alternatively, in the context of computation, we say that $g(t)$ computes g^* at *speed* ρ .

Lemma 3.2. Let $g : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$ be a real-valued function that converges to a real number $g^* \in \mathbb{R}$. Suppose there exist $c > 0$, $\rho > 0$, and $m \geq 0$ such that for all $t \geq 0$

$$|g(t) - g^*| \leq ct^m e^{-\rho t},$$

then g converges to g^* at a rate that is at least ρ .

Conversely, if g converges to g^ at a rate greater than ρ then there exists a $c > 0$ such that for all $t \geq 0$*

$$|g(t) - g^*| \leq ce^{-\rho t}.$$

Proof. To see the first conclusion, take logarithms, divide by t , apply limit superiors to both sides and apply Definition 3.1.

For the converse, suppose that g converges to g^* at rate $\rho_g > \rho$, and let $\varepsilon = \rho_g - \rho$. Then there is a $T > 0$ such that for any $t \geq T$, we have

$$\frac{\ln |g(t) - g^*|}{t} < -\rho_g + \varepsilon = -\rho. \quad (13)$$

Hence, for $t \geq T$, we have $|g(t) - g^*| < e^{-\rho t}$. Defining

$$c = \left(1 + \max_{t \in [0, T]} |g(t) - g^*|\right) e^{\rho T},$$

we have $|g(t) - g^*| \leq ce^{-\rho t}$ for all $t \geq 0$. □

3.1 Simple convergence results

We begin with three lemmas.

Lemma 3.3. *For each of $i \in \{1, 2\}$, let $g_i : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$ be a real-valued function that converges to $g_i^* \in \mathbb{R}$ at a rate that is at least ρ_{g_i} . Then the following hold.*

1. *For $i \in \{1, 2\}$, $\alpha \in \mathbb{R} \setminus \{0\}$, the scalar product αg_i converges to αg_i^* at rate that is at least ρ_{g_i} .*
2. *The sum $g_1(t) + g_2(t)$ converges to $g_1^* + g_2^*$ at rate that is at least*

$$\min\{\rho_{g_1}, \rho_{g_2}\}. \quad (14)$$

3. *The product $g_1(t)g_2(t)$ converges to $g_1^*g_2^*$ at rate that is at least*

$$\begin{cases} \min\{\rho_{g_1}, \rho_{g_2}\}, & \text{if } g_1^* \neq 0, g_2^* \neq 0, \\ \rho_{g_1}, & \text{if } g_1^* = 0, g_2^* \neq 0, \\ \rho_{g_2}, & \text{if } g_1^* \neq 0, g_2^* = 0, \\ \rho_{g_1} + \rho_{g_2}, & \text{if } g_1^* = 0, g_2^* = 0. \end{cases} \quad (15)$$

Proof. We first prove the conclusions assuming that the rate of convergence of the functions g_i is greater than ρ_{g_i} . In that case, Lemma 3.2 implies that there exist positive constants c_1 and c_2 such that

$$|g_i(t) - g_i^*| \leq c_i e^{-\rho_{g_i} t}, \quad i \in \{1, 2\}.$$

For the scalar product, we get the desired result from

$$|\alpha g_i(t) - \alpha g_i^*| \leq |\alpha| c_i e^{-\rho_{g_i} t}, \quad i \in \{1, 2\},$$

and then applying the first part of Lemma 3.2.

Applying the triangle inequality to the sum of g_1 and g_2 , we have

$$\begin{aligned} |g_1(t) + g_2(t) - (g_1^* + g_2^*)| &\leq |g_1(t) - g_1^*| + |g_2(t) - g_2^*| \\ &\leq c_1 e^{-\rho_{g_1} t} + c_2 e^{-\rho_{g_2} t}, \end{aligned}$$

and so the result again follows from Lemma 3.2.

Turning to the product, we note first that for $i \in \{1, 2\}$,

$$|g_i(t)| - |g_i^*| \leq ||g_i(t)| - |g_i^*|| \leq |g_i(t) - g_i^*| \leq c_i e^{-\rho_{g_i} t},$$

and so

$$|g_i(t)| \leq |g_i^*| + c_i e^{-\rho_{g_i} t}.$$

Now for the product $g_1(t)g_2(t)$ consider,

$$\begin{aligned} |g_1(t)g_2(t) - g_1^*g_2^*| &\leq |g_1(t)g_2(t) - g_1^*g_2(t)| + |g_1^*g_2(t) - g_1^*g_2^*| \\ &= |g_2(t)||g_1(t) - g_1^*| + |g_1^*||g_2(t) - g_2^*| \\ &\leq (|g_2^*| + c_2 e^{-\rho_{g_2} t}) c_1 e^{-\rho_{g_1} t} + |g_1^*| c_2 e^{-\rho_{g_2} t}, \end{aligned}$$

and therefore the result follows.

To complete the proof, we now assume that one or both of the functions g_i converge to their limiting values exactly at rates ρ_{g_i} . Then for an arbitrary $\varepsilon > 0$, the functions g_i converge to their limiting value at rates greater than $\rho_{g_i} - \varepsilon$. Then by what is proved earlier, the sum $g_1(t) + g_2(t)$ converges to $g_1^* + g_2^*$ at rate that is at least $\min\{\rho_{g_1}, \rho_{g_2}\} - \varepsilon$. But since $\varepsilon > 0$ is arbitrary, the rate of convergence is at least $\min\{\rho_{g_1}, \rho_{g_2}\}$. Similar considerations for the other two cases: scalar product and product of functions, completes the proof of the desired result. \square

Lemma 3.4. *Let $g : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$ be a real-valued function that converges to a nonzero real constant $g^* \in \mathbb{R} \setminus \{0\}$ at a rate that is at least ρ_g . Then $1/g(t)$ converges to $1/g^*$ at rate that is at least ρ_g .*

Proof. As in the proof of Lemma 3.3, we first assume that the convergence rate of g is greater than ρ_g . As in the previous lemma,

$$|g^*| - |g(t)| \leq ||g(t)| - |g^*|| \leq |g(t) - g^*| \leq ce^{-\rho_g t},$$

for some positive constant $c > 0$, and so

$$|g(t)| \geq |g^*| - ce^{-\rho_g t}.$$

In particular, there is a finite time $T_0 \geq 0$ such that for all $t > T_0$, $|g(t)| \geq |g^*|/2$. It then follows that for any $t > T_0$,

$$\left| \frac{1}{g(t)} - \frac{1}{g^*} \right| = \left| \frac{g(t) - g^*}{g(t)g^*} \right| \leq \frac{2ce^{-\rho_g t}}{(g^*)^2}.$$

But then, and as in the proof of Lemma 3.2, we can choose a $c' > 0$ sufficiently large so that for all $t \geq 0$,

$$\left| \frac{1}{g(t)} - \frac{1}{g^*} \right| \leq c'e^{-\rho_g t}.$$

Finally, arguing similarly to the proof of Lemma 3.3, we get the conclusion for the case when the convergence rate of g is exactly ρ_g . \square

Lemma 3.5. *Let $g : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$ be a real-valued function that converges to a non-negative constant $g^* \in \mathbb{R}_{\geq 0}$ at a rate that is at least ρ_g . Then for any $m \in \mathbb{R}_{>0}$, $g(t)^{1/m}$ converges to $(g^*)^{1/m}$ at rate that is at least*

$$\begin{cases} \rho_g, & \text{if } g^* > 0, \\ \rho_g/m, & \text{if } g^* = 0. \end{cases} \quad (16)$$

Proof. Similar to the proof of Lemma 3.3, it suffices to assume that the convergence rate of g is greater than ρ_g . By Lemma 3.2 there exists a $c > 0$ such that

$$|g(t) - g^*| < ce^{-\rho_g t}.$$

Suppose first that $g^* > 0$. Then there exists a $T_0 \geq 0$ such that $g(t) > g^*/2$ for all $t > T_0$. Then for $t > T_0$, we have

$$\begin{aligned} |g(t) - g^*| &= \left| g(t)^{1/m} - (g^*)^{1/m} \right| \left| \sum_{i=0}^{m-1} g(t)^i (g^*)^{m-i-1} \right| \\ &\geq \left| g(t)^{1/m} - (g^*)^{1/m} \right| \sum_{i=0}^{m-1} \frac{1}{2^i} (g^*)^i (g^*)^{m-i-1} \\ &= \left| g(t)^{1/m} - (g^*)^{1/m} \right| (2^m - 1) \left(\frac{g^*}{2} \right)^{m-1}. \end{aligned}$$

So for $t > T_0$

$$\left| g(t)^{1/m} - (g^*)^{1/m} \right| \leq \left(\frac{2}{g^*} \right)^{m-1} \frac{1}{2^m - 1} c e^{-\rho_g t},$$

and, again by reasoning similar to the end of the proof of Lemma 3.2, $g(t)^{1/m}$ converges to $(g^*)^{1/m} > 0$ at rate that is at least ρ_g .

When $g^* = 0$, there is a $c > 0$ such that

$$|g(t)| < c e^{-\rho_g t},$$

and so taking the m th root on both sides gives the result immediately. \square

3.2 Analysis of certain non-autonomous systems

Here we present our main mathematical results. These will be applied to our chemical systems in section 4.

Lemma 3.6. *Let $g_1 : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$ be a real-valued function that converges to $g_1^* \in \mathbb{R}$ at a rate that is at least ρ_{g_1} ; let $g_2 : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$ be a real-valued function that converges to a positive limit $g_2^* \in \mathbb{R}_{>0}$ at a rate that is at least ρ_{g_2} . We assume that the g_1, g_2 are smooth enough so that for any $x(0) = x_0 \geq 0$, the following non-autonomous differential equation has a unique solution $x : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$ for all time*

$$\dot{x}(t) = g_1(t) - g_2(t)x(t). \quad (17)$$

Then $x(t)$ converges to g_1^/g_2^* at rate that is at least*

$$\min\{\rho_{g_1}, \rho_{g_2}, g_2^*\}. \quad (18)$$

Proof. As in the proof of Lemma 3.3, we will assume that the convergence rates of g_1 and g_2 are greater than ρ_{g_1} and ρ_{g_2} respectively, and then just as in the proof of that lemma, the result will follow immediately for the case of equality. We first show that $x(t)$ is bounded above for all time $t \geq 0$. Since $g_2(t) \rightarrow g_2^* > 0$, there is a $T_0 > 0$ such that $g_2(t) > g_2^*/2$ for all $t > T_0$. Moreover, note that there must be a positive constant c_1 such that for all time $t > 0$, we have

$$g_1(t) \leq g_1^* + c_1 e^{-\rho_{g_1} t} \leq g_1^* + c_1.$$

Combining the two bounds gives an upper bound for $x(t)$. Since for any $t > T_0$,

$$\dot{x}(t) = g_1(t) - g_2(t)x(t) \leq g_1^* + c_1 - \frac{1}{2}g_2^* \cdot x(t),$$

we have that for any $t \geq 0$

$$x(t) \leq \max \left\{ x([0, T_0]), \frac{2(g_1^* + c_1)}{g_2^*} \right\}. \quad (19)$$

Now we show that $x(t)$ has the claimed rate of convergence to its limiting value g_1^*/g_2^* . First consider the differential equation obtained by replacing $g_2(t)$ with its limiting value g_2^*

$$\dot{z}(t) = g_1(t) - g_2^* \cdot z(t), \quad (20)$$

which has the unique solution

$$z(t) = e^{-g_2^* t} \left(z(0) + \int_0^t e^{g_2^* s} g_1(s) ds \right).$$

Therefore,

$$\begin{aligned} z(t) - \frac{g_1^*}{g_2^*} &= e^{-g_2^* t} \left(z(0) + \int_0^t e^{g_2^* s} g_1(s) ds \right) - \frac{g_1^*}{g_2^*} \\ &= e^{-g_2^* t} \left(z(0) - \frac{g_1^*}{g_2^*} + \int_0^t e^{g_2^* s} (g_1(s) - g_1^*) ds \right), \end{aligned}$$

and so

$$\begin{aligned} \left| z(t) - \frac{g_1^*}{g_2^*} \right| &\leq e^{-g_2^* t} \left(\left| z(0) - \frac{g_1^*}{g_2^*} \right| + \int_0^t e^{g_2^* s} |g_1(s) - g_1^*| ds \right) \\ &\leq e^{-g_2^* t} \left(\left| z(0) - \frac{g_1^*}{g_2^*} \right| + c \int_0^t e^{(g_2^* - \rho_{g_1}) s} ds \right), \end{aligned} \quad (21)$$

for some $c > 0$. The integral on the right hand side evaluates to

$$\int_0^t e^{(g_2^* - \rho_{g_1}) s} ds = \begin{cases} (e^{(g_2^* - \rho_{g_1}) t} - 1) / (g_2^* - \rho_{g_1}), & \text{if } g_2^* \neq \rho_{g_1} \\ t, & \text{if } g_2^* = \rho_{g_1}, \end{cases} \quad (22)$$

and so

$$\left| z(t) - \frac{g_1^*}{g_2^*} \right| \leq \begin{cases} e^{-g_2^* t} \left| z(0) - \frac{g_1^*}{g_2^*} \right| + \frac{c}{g_2^* - \rho_{g_1}} (e^{-\rho_{g_1} t} - e^{-g_2^* t}), & \text{if } g_2^* \neq \rho_{g_1}, \\ e^{-g_2^* t} \left| z(0) - \frac{g_1^*}{g_2^*} \right| + c t e^{-g_2^* t}, & \text{if } g_2^* = \rho_{g_1}. \end{cases} \quad (23)$$

After taking logarithms, then limit superiors and applying Definition 3.1, in either case, we have that $z(t)$ converges to g_1^*/g_2^* at least at rate $\min\{g_2^*, \rho_{g_1}\}$.

Now suppose that $x(t)$ is the solution to (17) with $x(0) = x_0$ and $z(t)$ is the solution to (20) with $z(0) = x_0$. We have

$$\begin{aligned} \frac{d}{dt}(z(t) - x(t)) &= -g_2^* z(t) + g_2(t) x(t) \\ &= -g_2^* (z(t) - x(t)) + (g_2(t) - g_2^*) x(t). \end{aligned}$$

Since $x(t)$ is bounded above, see (19), we have that $\alpha(t) := (g_2(t) - g_2^*) x(t)$ converges to 0 at least at rate ρ_{g_2} . Let $h(t) := z(t) - x(t)$, and so $h(0) = 0$ and $h(t)$ satisfies the differential equation

$$\dot{h}(t) = \alpha(t) - g_2^* h(t). \quad (24)$$

Comparing (24) with (20), $h(t) = z(t) - x(t)$ converges to 0 at least at rate $\min\{g_2^*, \rho_{g_2}\}$. Finally, since $x(t) = z(t) - h(t)$, a simple use of the triangle inequality gives us that $x(t) \rightarrow g_1^*/g_2^*$ at rate that is at least $\min\{\rho_{g_1}, \rho_{g_2}, g_2^*\}$. \square

Lemma 3.7. *For $i \in \{1, 2\}$, let $g_i : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$ be a real-valued function that converges to a positive limit $g_i^* \in \mathbb{R}_{> 0}$ at a rate that is at least ρ_{g_i} . We assume that the g_1, g_2 are smooth enough so that*

for any $x(0) = x_0 > 0$, the following non-autonomous differential equation has a unique solution $x : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$ for all time

$$\dot{x}(t) = x(t)(g_1(t) - g_2(t)x(t)^m), \quad (m \in \mathbb{Z}_{>0}). \quad (25)$$

Then $x(t)$ converges to $(g_1^*/g_2^*)^{1/m}$ at rate that is at least

$$\min\{\rho_{g_1}, \rho_{g_2}, mg_1^*\}. \quad (26)$$

Proof. Similar to Lemma 3.3 and Lemma 3.6, it suffices to assume that the convergence rate of g_i is greater than ρ_{g_i} for $i \in \{1, 2\}$. Note that the positive x -axis, $x > 0$, is invariant, i.e. if $x(0) > 0$, then $x(t) > 0$ for all $t > 0$. In fact, we argue that $x(t)$ has a positive lower bound as well as a positive upper bound for all time $t \geq 0$. The reasoning is similar to the one used in the previous lemma. Let $i \in \{1, 2\}$. Since $g_i(t) \rightarrow g_i^* > 0$, there is a $T_i > 0$ such that $g_i(t) > g_i^*/2$ for all $t > T_i$. Moreover, note that there must be a positive constant c_i such that for all time $t > 0$, we have

$$g_i(t) \leq g_i^* + c_i e^{-\rho_{g_i} t} \leq g_i^* + c_i.$$

For the upper bound of $x(t)$, consider that

$$\dot{x}(t) = x(t)(g_1(t) - g_2(t)x(t)^m) \leq x(t) \left(g_1^* + c_1 - \frac{1}{2}g_2^* \cdot x(t)^m \right),$$

for any $t > T_2$. Then we have that for any $t > 0$

$$x(t) \leq \max \left\{ x([0, T_2]), \left(\frac{2(g_1^* + c_1)}{g_2^*} \right)^{1/m} \right\}. \quad (27)$$

For the positive lower bound of $x(t)$, consider that

$$\dot{x}(t) = x(t)(g_1(t) - g_2(t)x(t)^m) \geq x(t)(g_1^*/2 - (g_2^* + c_2) \cdot x(t)^m),$$

for any $t > T_1$. Then we have that for any $t > 0$

$$x(t) \geq \min \left\{ x([0, T_1]), \left(\frac{g_1^*}{2(g_2^* + c_2)} \right)^{1/m} \right\}. \quad (28)$$

To show that we have the desired rate of convergence, we let

$$y(t) := \frac{1}{x(t)^m}.$$

Differentiating yields

$$\dot{y}(t) = -\frac{m}{x(t)^{m+1}} \cdot \dot{x}(t) = mg_2(t) - mg_1(t)y(t).$$

The functions mg_2 and mg_1 satisfy the hypotheses of Lemma 3.6, and so we have that $y(t)$ converges to g_2^*/g_1^* at rate that is at least $\min\{\rho_{g_1}, \rho_{g_2}, mg_1^*\}$. Then applying Lemma 3.4 gives us that the reciprocal of $y(t)$, i.e. $x(t)^m$ converges to g_1^*/g_2^* at rate that is at least $\min\{\rho_{g_1}, \rho_{g_2}, mg_1^*\}$. Finally applying Lemma 3.5 gives us that $x(t)$ converges to $(g_1^*/g_2^*)^{1/m}$ at rate that is at least $\min\{\rho_{g_1}, \rho_{g_2}, mg_1^*\}$. \square

Lemma 3.8. *Let $g_1 : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$ be a real-valued function that converges to a negative limit $g_1^* \in \mathbb{R}_{<0}$ at a rate that is at least ρ_{g_1} ; let $g_2 : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$ be a real-valued function that converges to a positive limit $g_2^* \in \mathbb{R}_{>0}$ at a rate that is at least ρ_{g_2} . We assume that g_1 and g_2 are smooth enough so that for any $x(0) = x_0 > 0$, the following non-autonomous differential equation has a unique solution $x : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$ for all time*

$$\dot{x}(t) = x(t)(g_1(t) - g_2(t)x(t)^m), \quad (m \in \mathbb{Z}_{>0}). \quad (29)$$

Then $x(t)$ converges to 0 at rate that is at least

$$\min\{\rho_{g_1}, -g_1^*\}. \quad (30)$$

Proof. Similar to the previous results, it suffices to assume that the convergence rate of g_i is greater than ρ_{g_i} for $i \in \{1, 2\}$. Since $g_2(t) \rightarrow g_2^* > 0$, there is a $T_0 > 0$ such that $g_2(t) > g_2^*/2$ for all $t \geq T_0$. So for any $t \geq T_0$, we have that

$$\dot{x}(t) \leq g_1(t)x(t),$$

which implies that $0 \leq x(t) \leq z(t)$ where $z(t)$ satisfies $z(T_0) = x(T_0)$ and for any $t \geq T_0$,

$$\dot{z}(t) = g_1(t)z(t).$$

This last equation satisfies the hypotheses of Lemma 3.6 with $g_1(t)$ of the lemma taken to be identically 0 and $g_2(t)$ of the lemma replaced with $-g_1(t)$. The conclusion is that $z(t)$ and therefore $x(t)$ converges to 0 at rate that is at least $\min\{\rho_{g_1}, -g_1^*\}$. \square

Lemma 3.9. *Let $g : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$ be a real-valued function that converges to 0 at a rate that is at least ρ_g . We assume that g is smooth enough so that for any $x(0) = x_0 > 0$, the following non-autonomous differential equation has a unique solution $x : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$ for all time*

$$\dot{x}(t) = x(t)(1 - g(t)x(t)^m), \quad (m \in \mathbb{Z}_{>0}).$$

Then for any $\varepsilon > 0$, there is a $c_\varepsilon > 0$ such that

$$x(t) \geq c_\varepsilon e^{(\min\{\rho_g/m, 1\} - \varepsilon)t}, \text{ for all } t \geq 0.$$

Proof. Note that the positive x -axis is invariant. We may therefore define

$$y(t) := \frac{1}{x(t)^m},$$

which satisfies the differential equation

$$\dot{y}(t) = mg(t) - my(t).$$

Since $y(t)$ satisfies the hypotheses of Lemma 3.6 with $g_1(t) := mg(t)$ and $g_2(t) \equiv m$, we have that $y(t)$ converges to 0 at rate that is at least $\min\{\rho_g, m\}$. Hence, for every $\varepsilon > 0$ there is a positive constant c_ε such that

$$y(t) \leq c_\varepsilon e^{-(\min\{\rho_g, m\} - m \cdot \varepsilon)t}.$$

In terms of the original variable $x(t)$ we have that

$$\frac{1}{x(t)^m} \leq c_\varepsilon e^{-(\min\{\rho_g, m\} - m \cdot \varepsilon)t}.$$

Taking the reciprocal and then the m th root gives the desired inequality. \square

4 Elementary operations

We construct reaction network modules that can carry out the elementary arithmetic operations of: identification, inversion, m th root (for $m \geq 2$), addition, multiplication, absolute difference, rectified subtraction, and partial real inversion (for reals represented via the dual rail representation). These operations form the core of basic arithmetic. Compositions of them will be taken up in the next section. Each of the elementary operations that we consider has one or two non-negative real inputs, hence referred to as a unary or binary operation, and one non-negative real output. In this section, we will show that the performance of each elementary operation retains from the input function the desired features of global exponential convergence, and convergence speed that is bounded from below independent of the inputs and output of computation. We will apply results from the previous section to prove these properties.

Throughout this section all rate constants are assumed to be equal to 1. Moreover, we assume throughout that the input functions (typically denoted as $a(t)$ or $b(t)$) are assumed to be smooth enough so that the associated differential equations have unique solutions for all relevant initial conditions and all time.

4.1 Identification

Lemma 4.1 (Identification). *Consider the reaction network and mass-action system*

$$\{A \rightarrow A + X, \quad X \rightarrow 0\}, \quad \dot{x}(t) = a(t) - x(t), \quad (31)$$

where $a(t)$ is a non-negative-valued function of time that converges to a non-negative constant a^* at a rate that is at least ρ_a . Then the concentration of species X , i.e. the variable x , computes identification $a^* \mapsto x^* = a^*$ at speed that is at least $\min\{\rho_a, 1\}$.

Proof. The result follows immediately by applying Lemma 3.6 with $g_1(t) := a(t)$ and $g_2(t) \equiv 1$. \square

Before proceeding further, we present an example of a composition of two identifications, which is arguably the simplest possible composition. We do so to point out how to utilize our results in that context. This topic will be extensively revisited in section 5.

Example 4.2. *Consider a sequence of two identification operations. We will use A to denote the input species, whose concentration is unchanged over time. Let Y denote a species whose long-term concentration is identified with that of A and X the species whose long-term value is identified with the long-term value of Y . The reaction network and system of ODEs for this computation are:*

$$\begin{aligned} A &\rightarrow A + Y, \quad Y \rightarrow 0, \quad Y \rightarrow Y + X, \quad X \rightarrow 0. \\ \dot{y}(t) &= a - y(t) \\ \dot{x}(t) &= y(t) - x(t). \end{aligned}$$

We can solve this system explicitly and thereby obtain exact rates of convergence. The solution to the initial value problem with $x(0) = x_0$ and $y(0) = y_0$ is

$$\begin{aligned} y(t) &= a(1 - e^{-t}) + y_0 e^{-t} \\ x(t) &= a(1 - (1 + t)e^{-t}) + (x_0 + ty_0)e^{-t} \end{aligned}$$

Thus both $y(t)$ and $x(t)$ converge to a at speed equal to 1. This is, of course, consistent with the conclusion from Lemma 4.1. Since the input species A is not changing, its rate of convergence is $\rho_a = +\infty$. By Lemma 4.1, $y(t)$ converges at speed ρ_y that is at least $\min\{\rho_a, 1\} = \min\{\infty, 1\} = 1$. Again, by application of Lemma 4.1, $x(t)$ converges at speed that is at least $\min\{\rho_y, 1\} = \min\{1, 1\} = 1$. \triangle

4.2 Inversion

Lemma 4.3 (Inversion). *Consider the reaction network and mass-action system*

$$\{X \rightarrow 2X, \quad A + 2X \rightarrow A + X\}, \quad \dot{x}(t) = x(t)(1 - a(t)x(t)), \quad (32)$$

where $a(t)$ is a non-negative-valued function of time that converges to a non-negative constant a^* at a rate that is at least ρ_a . Then the concentration of species X , i.e. the variable x , computes inversion $a^* \mapsto x^* = 1/a^*$ at speed that is at least $\min\{\rho_a, 1\}$.

Proof. The result follows immediately Lemma 3.7 with $m = 1$, $g_1(t) \equiv 1$, and $g_2(t) := a(t)$. \square

4.3 m th root

Lemma 4.4 (m th root). *Consider the reaction network and mass-action system*

$$\begin{aligned} \{Y \rightarrow 2Y, \quad A + (m+1)Y \rightarrow A + mY, \quad X \rightarrow 2X, \quad Y + 2X \rightarrow Y + X\} \\ \dot{y}(t) = y(t)(1 - a(t)y(t)^m), \\ \dot{x}(t) = x(t)(1 - y(t)x(t)), \end{aligned} \quad (33)$$

where $m \in \mathbb{Z}_{\geq 2}$ and $a(t)$ is a non-negative-valued function of time that converges to a non-negative constant a^* at a rate that is at least ρ_a . Then the concentration of species X , i.e. the variable x , computes the m th root $a^* \mapsto x^* = \sqrt[m]{a^*}$ at speed that is at least

$$\begin{cases} \min\{\rho_a, 1\}, & \text{if } a^* \neq 0, \\ \min\{\frac{\rho_a}{m}, 1\}, & \text{if } a^* = 0. \end{cases} \quad (34)$$

Proof. We first consider the case $a^* > 0$. The first differential equation satisfies the hypotheses of Lemma 3.7 with $m \geq 2$, $g_1(t) \equiv 1$ and $g_2(t) := a(t)$, and therefore $y(t)$ converges to $(a^*)^{-1/m}$ at rate that is at least $\min\{\rho_a, m\}$. Application of the same lemma again to the second differential equation with $m = 1$, $g_1(t) \equiv 1$ and $g_2(t) := y(t)$ gives us that $x(t)$ converges to $\sqrt[m]{a^*}$ at rate that is at least $\min\{\rho_a, 1\}$.

Now we consider the case when $a^* = 0$. We apply Lemma 3.9 to the first differential equation with $m \geq 2$ and $g(t) := a(t)$. Fix $\varepsilon > 0$. Then by Lemma 3.9, there exists a constant $c_\varepsilon > 0$ such that

$$y(t) \geq c_\varepsilon e^{\alpha t},$$

where $\alpha := \min\{\rho_a/m, 1\} - \varepsilon$. Therefore,

$$\dot{x}(t) \leq x(t)(1 - c_\varepsilon e^{\alpha t} x(t)). \quad (35)$$

Define a change of variables

$$z(t) = e^{\alpha t} x(t).$$

Clearly $z(0) = x(0)$ and $\dot{z}(t) = \alpha z(t) + e^{\alpha t} \dot{x}(t)$. Multiplying both sides of (35) with $e^{\alpha t}$, we get

$$e^{\alpha t} \dot{x}(t) \leq e^{\alpha t} x(t)(1 - c_\varepsilon e^{\alpha t} x(t)),$$

which we may rewrite as

$$\dot{z}(t) \leq z(t)(1 + \alpha - c_\varepsilon z(t)).$$

The solution $z(t)$ is bounded above by the solution of the autonomous system

$$\dot{w}(t) = w(t) (1 + \alpha - c_\varepsilon w(t)), \quad w(0) = z(0).$$

The explicit solution gives an upper bound for $x(t)$

$$x(t) = e^{-\alpha t} z(t) \leq e^{-\alpha t} w(t) = \frac{1 + \alpha}{c_\varepsilon (e^{\alpha t} - e^{-t}) + \frac{1+\alpha}{x_0} e^{-t}}.$$

The right hand side converges to 0 at a rate that is at least $\alpha = \min\{\rho_a/m, 1\} - \varepsilon$. But since the $\varepsilon > 0$ is arbitrary, the convergence is at rate $\min\{\rho_a/m, 1\}$. \square

4.4 Addition

Lemma 4.5 (Addition). *Consider the reaction network and mass-action system*

$$\{A \rightarrow A + X, \quad B \rightarrow B + X, \quad X \rightarrow 0\}, \quad \dot{x}(t) = a(t) + b(t) - x(t), \quad (36)$$

where $a(t)$ and $b(t)$ are non-negative functions of time that converge to non-negative constants a^* and b^* at rates that are at least ρ_a and ρ_b , respectively. Then the concentration of species X , i.e. the variable x , computes addition $(a^*, b^*) \mapsto x^* = a^* + b^*$ at speed that is at least $\min\{\rho_a, \rho_b, 1\}$.

Proof. By Lemma 3.3, the sum $a(t) + b(t)$ converges to $a^* + b^*$ at speed that is at least $\min\{\rho_a, \rho_b\}$. The result then follows from Lemma 3.6 with $g_1(t) := a(t) + b(t)$ and $g_2(t) \equiv 1$. \square

4.5 Multiplication

Lemma 4.6 (Multiplication). *Consider the reaction network and mass-action system*

$$\{A + B \rightarrow A + B + X, \quad X \rightarrow 0\}, \quad \dot{x}(t) = a(t)b(t) - x(t), \quad (37)$$

where $a(t)$ and $b(t)$ are non-negative functions of time that converge to non-negative constants a^* and b^* at rates that are at least ρ_a and ρ_b , respectively. Then the concentration of species X , i.e. the variable x , computes multiplication $(a^*, b^*) \mapsto x^* = a^* \cdot b^*$ at speed that is at least

$$\begin{cases} \min\{\rho_a, \rho_b, 1\}, & \text{if } a^* \neq 0, b^* \neq 0, \\ \min\{\rho_a, 1\}, & \text{if } a^* = 0, b^* \neq 0, \\ \min\{\rho_b, 1\}, & \text{if } a^* \neq 0, b^* = 0, \\ \min\{\rho_a + \rho_b, 1\}, & \text{if } a^* = 0, b^* = 0. \end{cases} \quad (38)$$

Proof. Applying Lemma 3.3 to the product $a(t)b(t)$ gives the lower bound on its rate of convergence. The result then follows by applying Lemma 3.6 with $g_1(t) := a(t)b(t)$ and $g_2(t) \equiv 1$. \square

4.6 Absolute difference

Lemma 4.7 (Absolute difference). *Consider the reaction network and mass-action system*

$$\begin{aligned} \{Y \rightarrow 2Y, \quad 2A + 3Y \rightarrow 2A + 2Y, \quad 2B + 3Y \rightarrow 2B + 2Y, \quad A + B + 3Y \rightarrow A + B + 5Y, \\ X \rightarrow 2X, \quad Y + 2X \rightarrow Y + X\}, \\ \dot{y}(t) = y(t) (1 - (a(t) - b(t))^2 y(t)^2), \\ \dot{x}(t) = x(t) (1 - y(t)x(t)), \end{aligned} \quad (39)$$

where $a(t)$ and $b(t)$ are non-negative functions of time that converge to non-negative constants a^* and b^* at rates that are at least ρ_a and ρ_b , respectively. Then the concentration of species X , i.e. the variable x , computes absolute difference $(a^*, b^*) \mapsto x^* = |a^* - b^*|$ at speed that is at least

$$\min\{\rho_a, \rho_b, 1\}. \quad (40)$$

Proof. By Lemma 3.3, the difference $a(t) - b(t)$ converges to $a^* - b^*$ at speed that is at least $\min\{\rho_a, \rho_b\}$. An application of the product part of the same Lemma 3.3 gives that $(a(t) - b(t))^2$ converges to $(a^* - b^*)^2$ at rate that is at least

$$\begin{cases} \min\{\rho_a, \rho_b\}, & \text{if } a^* \neq b^* \\ 2 \min\{\rho_a, \rho_b\}, & \text{if } a^* = b^*. \end{cases}$$

Finally, we apply Lemma 4.4 to the system (39) with $m = 2$ and the function $a(t)$ replaced with $(a(t) - b(t))^2$. \square

4.7 Rectified subtraction

Lemma 4.8 (Rectified subtraction). *Consider the reaction network and mass-action system*

$$\begin{aligned} \{Y \rightarrow 2Y, \quad 2A + 3Y \rightarrow 2A + 2Y, \quad 2B + 3Y \rightarrow 2B + 2Y, \quad A + B + 3Y \rightarrow A + B + 5Y, \\ A + Y + X \rightarrow A + Y + 2X, \quad B + Y + X \rightarrow B + Y, \quad Y + 2X \rightarrow Y + X\}, \\ \dot{y}(t) = y(t) (1 - (a(t) - b(t))^2 y(t)^2), \\ \dot{x}(t) = y(t) x(t) (a(t) - b(t) - x(t)), \end{aligned} \quad (41)$$

where $a(t)$ and $b(t)$ are non-negative functions of time that converge to non-negative constants a^* and b^* at rates that are at least ρ_a and ρ_b , respectively. Then the concentration of species X , i.e. the variable x , computes rectified subtraction

$$(a^*, b^*) \mapsto \begin{cases} a^* - b^* & \text{if } a^* > b^* \\ 0 & \text{if } a^* \leq b^* \end{cases}$$

at speed that is at least

$$\min\{\rho_a, \rho_b, 1\}. \quad (42)$$

Proof. By Lemma 3.3, the difference $a(t) - b(t)$ converges to $a^* - b^*$ at rate that is at least $\rho := \min\{\rho_a, \rho_b\}$.

First suppose that $a^* \neq b^*$. Similar to the proof of Lemma 4.7, $(a(t) - b(t))^2$ converges to $(a^* - b^*)^2$ at rate that is at least $\min\{\rho_a, \rho_b\}$. The differential equation for $y(t)$ satisfies the hypotheses of Lemma 3.7 with $m = 2$, $g_1(t) \equiv 1$ and $g_2(t) := (a(t) - b(t))^2$, and therefore $y(t)$ converges to $|a^* - b^*|^{-1} > 0$ at rate that is at least $\min\{\rho_a, \rho_b, 2\}$. Now rewrite the differential equation for $x(t)$ as

$$\dot{x}(t) = x(t) (y(t)(a(t) - b(t)) - y(t)x(t)), \quad (43)$$

and note that by Lemma 3.3, $y(t)(a(t) - b(t))$ converges to either $+1$ or -1 , depending on whether $a^* > b^*$ or $a^* < b^*$, at rate that is at least $\min\{\rho_a, \rho_b, 2\}$. When $a^* > b^*$, we apply Lemma 3.7 with $m = 1$, $g_1(t) := y(t)(a(t) - b(t))$, and $g_2(t) = y(t)$ to (43), and we conclude that $x(t)$ converges to $a^* - b^*$ at rate that is at least $\min\{\rho_a, \rho_b, 1\}$. When $a^* < b^*$, we apply Lemma 3.8 with $m = 1$,

$g_1(t) := y(t)(a(t) - b(t))$, and $g_2(t) = y(t)$ to (43), and we conclude that $x(t)$ converges to 0 at rate that is at least $\min\{\rho_a, \rho_b, 1\}$. In either case, we get the desired result when $a^* \neq b^*$.

Now consider the case that $a^* = b^*$. We have that $a(t) - b(t)$ converges to 0 at rate that is at least $\rho = \min\{\rho_a, \rho_b\}$. We fix an $\varepsilon > 0$. Then, there is a constant $c_\varepsilon > 0$ such that for all $t \geq 0$,

$$|a(t) - b(t)| < c_\varepsilon e^{-(\rho - \varepsilon)t}. \quad (44)$$

Similar to the proof of Lemma 4.7, $(a(t) - b(t))^2$ converges to 0 at rate that is at least $2 \min\{\rho_a, \rho_b\}$. The differential equation for $y(t)$ satisfies the hypotheses of Lemma 3.9 with $m = 2$, and $g(t) := (a(t) - b(t))^2$, and therefore there is a $c'_\varepsilon > 0$ such that for all $t \geq 0$, we have

$$y(t) \geq c'_\varepsilon e^{(\min\{\rho_a, \rho_b, 1\} - \varepsilon)t} = c'_\varepsilon e^{\nu t}, \quad (45)$$

where $\nu := \min\{\rho_a, \rho_b, 1\} - \varepsilon$.

We will now show that $x(t)$ converges to 0 at rate that is at least ν . As in our previous results, this will be enough to prove the result as $\varepsilon > 0$ is arbitrary. Note that the positive x -axis is invariant, i.e. for any $x(0) > 0$, $x(t) > 0$ for all time $t \geq 0$. Define

$$u(t) := \frac{1}{K} x(t) e^{\nu t}, \quad \text{where } K := c_\varepsilon + (\nu + 1)/c'_\varepsilon,$$

and so $u(t) > 0$ for all $t \geq 0$ whenever $x(0) > 0$. It suffices to show that if $u(t) \geq 1$, then $\dot{u}(t) \leq -1$. Indeed assuming this is true, we first show that $x(t)$ converges at the desired rate. Consider two cases: (i) $u(0) < 1$, since the region $u < 1$ is invariant we have that

$$x(t) = K u(t) e^{-\nu t} \leq K e^{-\nu t},$$

and (ii) $u(0) \geq 1$, in which case we have that $u(t) \leq u(0)$ for all $t \geq 0$, and so

$$x(t) = K u(t) e^{-\nu t} \leq K u(0) e^{-\nu t} = x(0) e^{-\nu t}.$$

Combining the two cases, we have that for all $t \geq 0$,

$$x(t) \leq \max\{x(0), K\} e^{-\nu t},$$

i.e. x converges to 0 at rate that is at least ν . Since $\varepsilon > 0$ is arbitrary, x converges to 0 at rate that is at least $\min\{\rho_a, \rho_b, 1\}$.

Now we prove the claim that if $u(t) \geq 1$ then $\dot{u}(t) \leq -1$. Note that we have

$$\begin{aligned} \dot{u} &= \frac{e^{\nu t}}{K} (\dot{x} + \nu x) = \nu u + u \frac{\dot{x}}{x} \\ &= \nu u + u y (a - b - K u e^{-\nu t}) \leq \nu u + u y (c_\varepsilon e^{(-\rho + \varepsilon)t} - K u e^{-\nu t}) \leq \nu u + u y (c_\varepsilon - K u) e^{-\nu t}, \end{aligned}$$

where we used (44), $\nu = \min\{\rho, 1\} - \varepsilon \leq \rho - \varepsilon$, and also that $u(t)$ and $y(t)$ are positive for all $t \geq 0$. Since we are assuming that $u \geq 1$, we then have

$$\begin{aligned} \dot{u} &\leq \nu u + u y (c_\varepsilon - K) e^{-\nu t} = \nu u - u y \left(\frac{\nu + 1}{c'_\varepsilon} \right) e^{-\nu t} \\ &\leq \nu u - u (\nu + 1) = -u \leq -1, \end{aligned}$$

where we used (45) and the assumption $u \geq 1$. This concludes the proof. \square

4.8 Partial real inversion

The input to the elementary operation of *partial real inversion* is a nonzero real number and can be positive or negative. Thus, the dual rail representation is used for the input, see section 2.2.

Lemma 4.9 (Partial real inversion). *Consider the reaction network and mass-action system*

$$\begin{aligned} &\{Y \rightarrow 2Y, \quad A_p + 2Y \rightarrow A_p + Y, \quad A_n + 2Y \rightarrow A_n + Y, \\ &A_p + Y + X \rightarrow A_p + Y + 2X, \quad A_n + Y + X \rightarrow A_n + Y, \quad 2A_p + Y + 2X \rightarrow 2A_p + Y + X\}, \quad (46) \\ &\dot{y}(t) = y(t) (1 - (a_p(t) + a_n(t))y(t)) \\ &\dot{x}(t) = y(t)x(t) (a_p(t)(1 - a_p(t)x(t)) - a_n(t)) \end{aligned}$$

where $a_p(t)$ and $a_n(t)$ are non-negative functions of time that converge to non-negative constants a_p^* and a_n^* at rates that are at least ρ_{a_p} and ρ_{a_n} , respectively. We assume that one and only one of a_p^* or a_n^* is positive, while the other is zero. Then the concentration of species X , i.e. the variable x , computes partial real inversion

$$(a_p^*, a_n^*) \mapsto \begin{cases} 1/a_p^*, & \text{if } a_p^* > 0, \\ 0, & \text{if } a_p^* = 0, \end{cases}$$

at speed that is at least

$$\min\{\rho_{a_p}, \rho_{a_n}, 1\}. \quad (47)$$

Proof. By Lemma 3.3, the sum $a_p(t) + a_n(t)$ converges to $a_p^* + a_n^* > 0$ at rate that is at least $\min\{\rho_{a_p}, \rho_{a_n}\}$. The first differential equation in (46) satisfies the hypotheses of Lemma 3.7 with $m = 1$, $g_1(t) \equiv 1$ and $g_2(t) := a_p(t) + a_n(t)$, and therefore $y(t)$ converges to $(a_p^* + a_n^*)^{-1}$ at rate that is at least $\min\{\rho_{a_p}, \rho_{a_n}, 1\}$.

The differential equation for $x(t)$ can be written as

$$\dot{x}(t) = x(t) (y(t)(a_p(t) - a_n(t)) - y(t)a_p(t)^2x(t))$$

By Lemma 3.3, the difference $a_p(t) - a_n(t)$ converges to its limit $a_p^* - a_n^*$ at rate that is at least $\min\{\rho_{a_p}, \rho_{a_n}\}$. By the product part of the same lemma, the product $y(t)(a_p(t) - a_n(t))$ converges to $+1$ or -1 , depending on whether $a_p^* > 0$ or $a_p^* = 0$, at rate that is at least $\min\{\rho_{a_p}, \rho_{a_n}, 1\}$.

First consider the case $a_p^* > 0$ and $a_n^* = 0$. By Lemma 3.3, the product $y(t)a_p(t)^2$ converges to a_p^* at rate that is at least $\min\{\rho_{a_p}, \rho_{a_n}, 1\}$. The differential equation for $x(t)$ satisfies the hypotheses of Lemma 3.7 with $m = 1$, $g_1(t) := y(t)(a_p(t) - a_n(t))$ and $g_2(t) := y(t)a_p(t)^2$, and therefore $x(t)$ converges to $1/a_p^*$ at rate that is at least $\min\{\rho_{a_p}, \rho_{a_n}, 1\}$.

Now consider the case $a_p^* = 0$ and $a_n^* > 0$. Suppose that $z(t)$ satisfies the differential equation $\dot{z}(t) = z(t)y(t)(a_p(t) - a_n(t))$, then by applying Lemma 3.6 with $g_1(t) \equiv 0$, and $g_2(t) := -y(t)(a_p(t) - a_n(t))$, we get that $z(t)$ converges to 0 at rate that is at least $\min\{\rho_{a_p}, \rho_{a_n}, 1\}$. Since we have

$$\begin{aligned} \dot{x}(t) &= x(t) (y(t)(a_p(t) - a_n(t)) - y(t)a_p(t)^2x(t)) \\ &\leq x(t)y(t)(a_p(t) - a_n(t)), \end{aligned}$$

for any $x(0) = z(0)$, we must have $x(t) \leq z(t)$ for all $t \geq 0$, and so $x(t)$ converges to 0 at rate that is at least $\min\{\rho_{a_p}, \rho_{a_n}, 1\}$. \square

5 Composite operations

5.1 Main theory

The elementary operations in section 4 can be combined to produce composite operations. Some operations, such as division, maximum, subtraction, which may be considered “elementary” from the point of view of arithmetic are in fact composite in our construction as they are produced by combining the operations in the previous section.

We start with a theorem about the speed of a general arithmetic operation, one that involves finitely many compositions of elementary operations. The result presented below is the most consequential theorem of this paper.

Theorem 5.1 (Composite computations). *Consider a computation that is composed from a finite number of elementary computations from section 4. Then the speed of this composite computation is at least that of the slowest elementary computation.*

Proof. The proof only requires combining the results in Lemmas 4.1-4.9. □

Example 4.2 is helpful here. When doing a sequence of two identifications, the intermediate variable converges like e^{-t} , while the output of the second identification converges like te^{-t} . For both variables, intermediate and output, the speed of convergence is 1.

For all elementary operations, we used the assumption that the mass action reaction rate constants are all equal. In fact, we set the rate constants to be equal to 1. If instead, all rate constants are taken to be $\sigma > 0$, a time change argument can be used to make them all equal to 1 in the modified system (see Remark 5.4 below). We now state two corollaries which assume that the rate constants are all equal to 1.

Corollary 5.2. *Consider a computation that is composed from a finite number of elementary computations from section 4. If none of the elementary computations are a root of zero, then the speed of the composite computation is at least 1.*

Corollary 5.3. *Consider a computation that is composed from a finite number of elementary computations from section 4. Suppose that of all the elementary steps in the composite computation, there are p elementary steps which require taking the m_j -th root of zero, $1 \leq j \leq p$. Then the speed of the composite computation is at least*

$$\prod_{j=1}^p \frac{1}{m_j}.$$

Remark 5.4. *Throughout this work, we have always chosen the rate constants of the reactions to be equal to 1. However, the choice of the value 1 was arbitrary. For example, suppose that the differential equation for our system, with a rate constant of 1, is*

$$\dot{x}(t) = f(x(t)).$$

Now suppose that we instead choose rate constants of $\sigma > 0$, and denote the system by

$$\dot{z}(t) = \sigma f(z(t)).$$

We then see that x and z are related by a simple time-change: $z(t) = x(\sigma t)$. You can see this by simply utilizing the chain rule:

$$\frac{d}{dt}x(\sigma t) = f(x(\sigma t)) \cdot \sigma.$$

Thus, we immediately conclude from Corollary 5.2 that if none of the elementary computations are the root of zero, the speed of the composite computation via z is σ , whereas from Corollary 5.3, we conclude the speed via z is $\prod_{j=1}^p \sigma/m_j$, when the conditions of that corollary are met. We discuss the possibility of having non-equal rate constants in section 7. \triangle

We now show how to compose operations from section 4 to carry out the operations *division*, *maximum*, and *subtraction*.

Division

Division, a^*/b^* , for $a^*, b^* > 0$, is a composite operation of our elementary operations. First we compute the reciprocal $b^* \mapsto 1/b^*$, and then the product $(a^*, 1/b^*) \mapsto a^*/b^*$, see Figure 2. The reaction network for computing division can be built simply by taking the union of reaction network *modules*, each of which computes an elementary operation, as follows

$$\begin{aligned} \{Z \rightarrow 2Z, \quad B + 2Z \rightarrow B + Z\}, \quad \dot{z}(t) = z(t)(1 - b(t)z(t)) \quad (\text{inversion}), \\ \{A + Z \rightarrow A + Z + X, \quad X \rightarrow 0\}, \quad \dot{x}(t) = a(t)z(t) - x(t) \quad (\text{multiplication}). \end{aligned} \quad (48)$$

Suppose $a(t)$ and $b(t)$ are real-valued functions that converge to $a^* \geq 0$ and $b^* > 0$, respectively. Then concentration of species X , i.e. the variable $x(t)$ computes a^*/b^* . The minimum speed of convergence can be determined as follows. By combining results from Lemmas 4.3, and 4.6, we have that

$$\begin{aligned} \rho_x &\geq \min\{\rho_a, \rho_z, 1\} \\ &\geq \min\{\rho_a, \min\{\rho_b, 1\}, 1\} = \min\{\rho_a, \rho_b, 1\}. \end{aligned}$$

In the previous short calculation, we have shown that specifically for a composition involving reciprocation and multiplication, the minimum speed of the composite computation is at least the minimum speed of the slowest elementary computation. This is, of course, an instance of the general result, Theorem 5.1.

Maximum

The maximum $\max\{a, b\}$ of two positive reals a and b can be found using the following formula

$$\max\{a, b\} = \frac{1}{2}(a + b + |a - b|),$$

which is a composition of the absolute difference, two binary additions and a multiplication by $1/2$. By Theorem 5.1, the speed of computation is bounded below.

Subtraction

The output $a - b$ of subtraction for two non-negative reals $a, b \geq 0$ is an arbitrary real number, and therefore requires the dual rail representation, i.e. two chemical species, the concentration of one is the positive part of $a - b$ and the concentration of the other is the negative part. We may write $a - b$ in dual rail representation as

$$a - b = (a \dot{-} b, b \dot{-} a).$$

In other words, a subtraction is simply two rectified subtractions, and therefore by Theorem 5.1, has the same lower bound on the speed as either of the two rectified subtractions.

5.2 Graphical representation for composite operations

Combining elementary arithmetic computations into a composite computation has similarities with how logic gates are combined in a Boolean logic operation. Following this analogy, we can set up a visual aid where the elementary operations of section 4 are thought of “elementary gates” which can be combined in various different ways. We first give a catalogue of the elementary gates, shown in Figure 1.

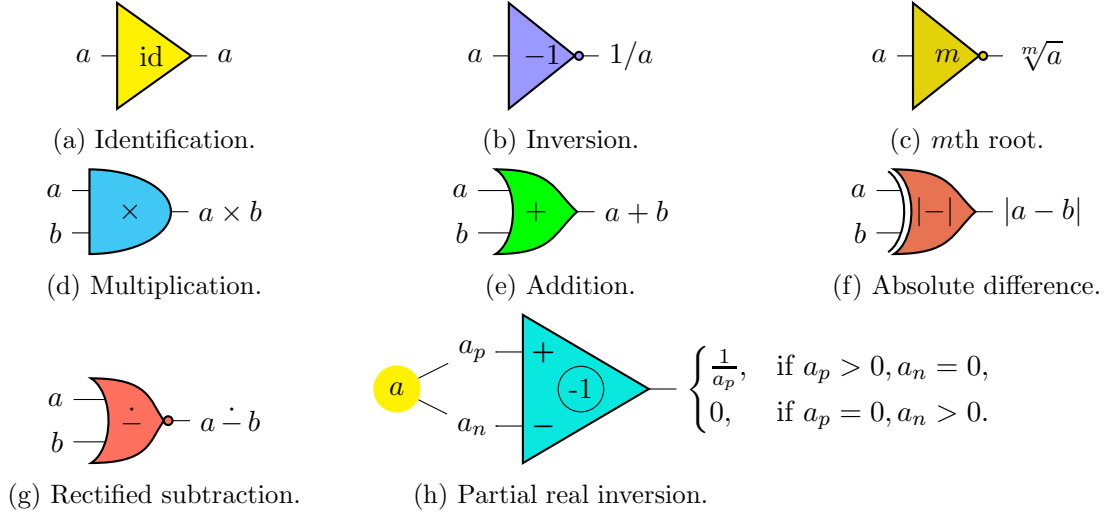


Figure 1: Elementary gates.

We can now put the gates together in different ways to generate the composite computations discussed in section 5.1. We start with division, shown in Figure 2 which only requires the elementary operations of identification, reciprocal, and multiplication. Figure 3 depicts a diagram of how the operation of taking the maximum of two non-negative reals can be constructed. Figure 4 shows a pictorial way of doing subtraction by combining two rectified subtractions.

We continue to use the diagrammatic representations of arithmetic computations in section 6. We show that each operation over reals can be decomposed into operations whose inputs and outputs are only non-negative reals. We give the diagrammatic decomposition into elementary arithmetic gates along with each description.

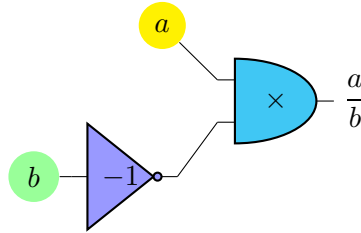


Figure 2: A graphical representation of division.

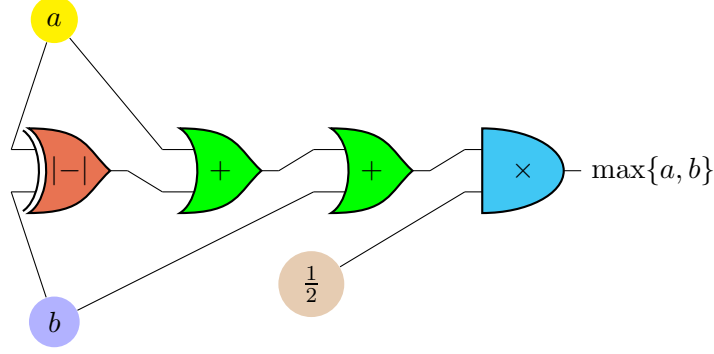


Figure 3: Maximum of two non-negative reals.

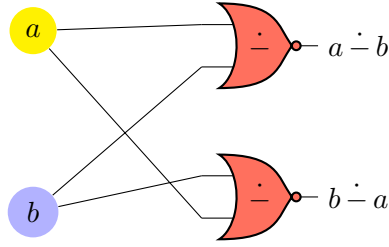


Figure 4: Subtraction.

6 Arithmetic operations on arbitrary real numbers

The previous section discussed arithmetic operations with non-negative inputs. In the final construction on subtraction, the output is any real number—positive, negative, or zero—even though the inputs are positive. We needed ‘dual rail’ to represent an arbitrary real number as a vector with two non-negative components. In this section, we continue to use the dual rail construction to represent an arbitrary real number which can be both an input and an output.

Addition and normalization

We define addition in the usual way, for any $a, b \in \mathbb{R}$,

$$a + b = (a_p, a_n) + (b_p, b_n) = (a_p + b_p, a_n + b_n).$$

The operation can be performed with ODEs/reaction networks by using two successive regular additions. Since the output of addition is not necessarily canonical, we define a normalization operation which brings the output into this form.

$$(a_p, a_n) = \begin{cases} (a_p - a_n, 0) & \text{if } a_p > a_n \\ (0, a_n - a_p) & \text{if } a_p \leq a_n. \end{cases}$$

The operation is performed using one dual rail subtraction.

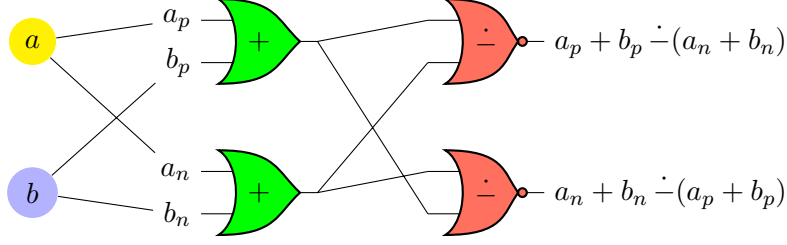


Figure 5: Real addition.

Additive inverse and subtraction

The negative of the real number a is defined via

$$x = -a = -(a_p, a_n) = (a_n, a_p) = (x_p, x_n).$$

This can be performed using two assignments. The system of ODEs is the following:

$$\begin{aligned}\dot{x}_p &= a_n - x_p, \\ \dot{x}_n &= a_p - x_n.\end{aligned}$$

Subtraction is defined as follows:

$$a - b = (a_p, a_n) - (b_p, b_n) = (a_p, a_n) + (-(b_p, b_n)) = (a_p, a_n) + (b_n, b_p) = (a_p + b_n, a_n + b_p).$$

Thus subtraction can be performed using two regular additions followed by a normalization to bring the output into canonical form.

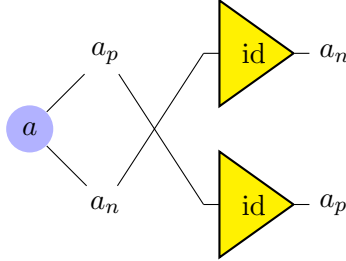


Figure 6: Real additive inverse.

Multiplication

Multiplication of arbitrary real numbers involves four multiplications and two additions of non-negative real numbers as shown by the following calculation:

$$\begin{aligned}a \cdot b &= (a_p, a_n) \cdot (b_p, b_n) = (a_p - a_n) \cdot (b_p - b_n) \\ &= a_p b_p + a_n b_n - a_p b_n - a_n b_p = (a_p b_p + a_n b_n, a_p b_n + a_n b_p).\end{aligned}$$

The normalization step is unnecessary since the output of multiplication is in canonical form whenever the inputs are in canonical form. Indeed, we have that

$$(a_p b_p + a_n b_n)(a_p b_n + a_n b_p) = (a_p^2 + a_n^2)b_p b_n + a_p a_n(b_p^2 + b_n^2) = 0,$$

where the last equality follows from $a_p a_n = 0$ and $b_p b_n = 0$.

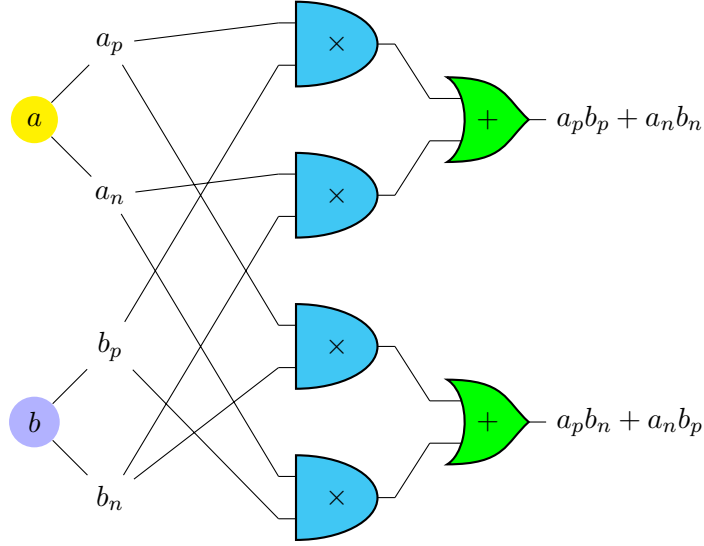


Figure 7: Real multiplication.

Inversion and Division

Consider the nonzero input $0 \neq a = (a_p, a_n)$ in canonical form, i.e. exactly one of a_p or a_n is positive, so that $a_p + a_n > 0$ and $a_p a_n = 0$. The multiplicative inverse in canonical form is

$$a^{-1} = \begin{cases} \left(\frac{1}{a_p}, 0 \right) & \text{if } a_p > 0, \\ \left(0, \frac{1}{a_n} \right) & \text{if } a_n > 0. \end{cases}$$

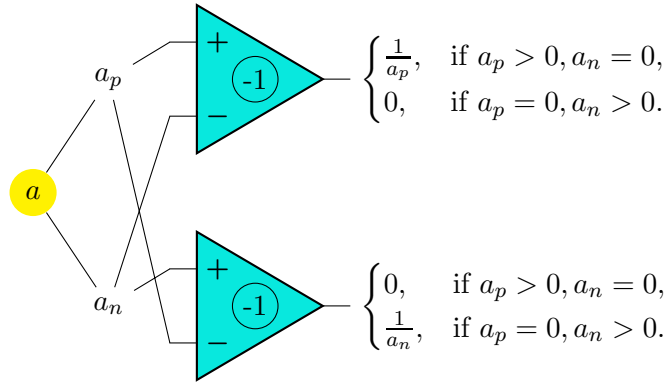


Figure 8: Real inversion.

Finally, division b/a is simply inversion of a followed by multiplication of b and $1/a$.

7 Discussion

In this paper, we focused on finding reaction networks and associated mass action systems that carry out arithmetic at a specified speed. We gave novel, explicit constructions for computing

elementary operations and showed how the elementary operations can be combined into composite computations involving a finite number steps. Furthermore, we proved rigorously that for any specified level of accuracy, all the constructions accomplish the computations at any desired speed.

We assumed that underlying our algorithmic considerations is an idealized analog computer that is built from a perfectly well-behaved mass action system. We anticipate introducing realistic constraints in future research that will strengthen and build on the idealized setup developed here. We now discuss some future research directions that are natural offshoots of the present paper.

Chemical reactions involving at most two molecules as reactants, sometimes referred to as bimolecular reactions, are considered the simplest to design and found most commonly in nature. The constructions given in this paper did not have this constraint and, in fact, did require multiple different reactant types with high molecularities. Is it possible to find constructions that compute arithmetic at input-independent speeds using only bimolecular reactants?

In some of our constructions, an intermediate variable was allowed to increase to indefinitely high values. For instance when computing rectified subtraction 41, the concentration of the species Y goes to infinity when $a^* = b^*$. This possibility of a variable increasing in an unbounded manner may be deemed unrealistic to implement using synthetic biochemistry, or at the very least is an energy-hungry construction. Is it possible to give an alternative construction for subtraction/rectified subtraction which works equally well algorithmically, has all the desired features, particularly including input-independent speed of computation, but also does not require any variable to increase in an unbounded manner?

We assumed throughout that all reactions have the same rate constant. This poses significant, perhaps unrealistic, design challenges. In the next improved iteration of construction, we may want to assume that the reactions occur at known, but not necessarily equal or controllable, rates. Given this significant constraint, how should the reaction networks for elementary operations and composite computations be modified to give an accurate output at input-independent speed?

A bottleneck in the speed of computation for our constructions is a root of zero. Can this bottleneck be removed? Are there constructions for which the speed of computation is not lowered even when an elementary step requires computing a root of zero. Or is the jump in speed when computing a root of zero as opposed to a root of a positive number, not merely a bug but a positive feature? Perhaps we may be able to put to good use the discrete jump in speed when the input to an m -th root is either a zero or a positive number. To be concrete, consider the elementary operation of computing a square root. When the input is $\varepsilon > 0$, the speed of computation is at least 1 while when the input is 0, the speed is halved. Can this phenomenon be a basis for designing a "zero detector" or an "equality detector"?

Arithmetic only requires composing finitely many elementary operations. Computation of general functions may need the use of power series and other techniques that involve limits and convergence. Approximating the value of a general function, using a power series for example, may require computing an arbitrarily large, even if finite, number of elementary operations. To compute a power series, we need to consider the initial rate of convergence of each elementary operation, not merely its asymptotic rate. The 'overhead' time required in each elementary operation may add up to a significant amount when considering a large, possibly infinite number of operations. Can our analysis be extended and what additional features or constraints are required when computing general functions?

An alternative approach, that does not require using power series, may be to develop specific algorithms for computing commonly used functions such as exponentials and logarithms, based on mass-action kinetics and whose speeds are input-independent. A fruitful research direction may be to explore the computational reach of reaction network-based hardware by developing versatile algorithms that broaden the suite of possible computations at input-independent speeds. Conversely,

hardware may impose additional constraints on the software thereby limiting computational accuracy or speed. Are there specific design aspects from the vantage of synthetic biochemistry or other practical engineering constraints that speed up or slow down computation? Multidisciplinary teams of researchers, representing synthetic biology, computer science as well as mathematics may be most poised to answer related questions and make rapid progress.

A research program could be founded on a broader territory of inquiry. As an instance, consider Boolean algebra, which is foundational to logical computations used in modern digital computers. One might ask: do there exist good, i.e. accurate and efficient, algorithms for performing Boolean algebra using reaction networks?

Acknowledgement

We would like to thank Josef Hofbauer (University of Vienna) for suggesting the change of variables used in the proof of Lemma 3.7, which simplified the analysis. This work was supported by a grant from the Simons Foundation (905083, Anderson). We also gratefully acknowledge NSF grant DMS-2051498.

References

- [1] David F Anderson, Badal Joshi, and Abhishek Deshpande. On reaction network implementations of neural networks. *Journal of the Royal Society Interface*, 18(177):20210031, 2021.
- [2] H. Buisman, Huub M.M. ten Eikelder, Peter Hilbers, and Anthony Liekens. Computing algebraic functions with biochemical reaction networks. *Artif. Life*, 15(1):5–19, 2009.
- [3] Daniele Cappelletti, Andres Ortiz-Muñoz, David F. Anderson, and Erik Winfree. Stochastic chemical reaction networks for robustly approximating arbitrary probability distributions. *Theoretical Computer Science*, 801:64–95, 2020.
- [4] Ho-Lin Chen, David Doty, Wyatt Reeves, and David Soloveichik. Rate-independent computation in continuous chemical reaction networks. *Journal of the ACM*, 70(3):1–61, 2023.
- [5] Ho-Lin Chen, David Doty, and David Soloveichik. Deterministic function computation with chemical reaction networks. *Natural computing*, 13:517–534, 2014.
- [6] Lulu Qian, Erik Winfree, and Jehoshua Bruck. Neural network computation with DNA strand displacement cascades. *Nature*, 475(7356):368–372, 2011.
- [7] Lulu Qian, David Soloveichik, and Erik Winfree. Efficient Turing-universal computation with DNA polymers. In *DNA Computing and Molecular Programming 16*, pages 123–140, 2011.
- [8] Lulu Qian and Erik Winfree. A simple DNA gate motif for synthesizing large-scale circuits. *J. R. Soc. Interface*, 8(62):1281–1297, 2011.
- [9] David Soloveichik, Georg Seelig, and Erik Winfree. DNA as a universal substrate for chemical kinetics. *PNAS*, 107(12):5393–5398, 2010.
- [10] Martin Feinberg. *Foundations of chemical reaction network theory*. Springer, 2019.
- [11] Yannis Tsividis. Not your father’s analog computer. *IEEE Spectrum*, 55(2):38–43, 2018.

- [12] Bruce J. MacLennan. The promise of analog computation. *International Journal of General Systems*, 43(7):682–696, 2014.
- [13] Sara Achour, Rahul Sarpeshkar, and Martin C. Rinard. Configuration synthesis for programmable analog devices with Arco. *ACM SIGPLAN Notices*, 51(6):177–193, 2016.
- [14] Ramiz Daniel, Jacob R. Rubens, Rahul Sarpeshkar, and Timothy K. Lu. Synthetic analog computation in living cells. *Nature*, 497(7451):619–623, 2013.
- [15] Tianqi Song, Sudhanshu Garg, Reem Mokhtar, Hieu Bui, and John Reif. Analog computation by DNA strand displacement circuits. *ACS synthetic biology*, 5(8):898–912, 2016.
- [16] Vera Hárs and János Tóth. On the inverse problem of reaction kinetics. *Qualitative theory of differential equations*, 30:363–379, 1981.