Biomath Communications

www.biomathforum.org/biomath/index.php/conference

An algorithm for building and integrating dynamic systems based on reaction networks

Petar Chernev
Sofia University "St. Kliment Ohridski"
prchernev@gmail.com

Keywords: reaction networks, dynamic systems

Abstract

In the present work we give an overview and implementation of an algorithm for building and integrating dynamic systems from reaction networks. Reaction networks have their roots in chemical reaction network theory, but their nature is general enough that they can be applied in many fields to model complex interactions. Our aim is to provide a simple to use program that allows for quick prototyping of dynamic models based on a system of reactions. After introducing the concept of a reaction and a reaction network in a general way, not necessarily connected to chemistry, we outlay the algorithm for building its associated system of ODEs. Finally, we give a few example usages where we examine a range of growth-decay models in the context of reaction networks.

Citation: Petar Chernev, An algorithm for building and integrating dynamic systems based on reaction networks, Biomath Communications 8, pp. 49-74, `https://doi.org/10.11145/bmc.2021.08.285`

1 Introduction

Reaction networks have their origin in chemistry. As the name suggests, they consist of a number of reactions between elements. In chemistry a reaction represents an interaction between two or more molecules which produces molecules of one or more other (or the same, when we have a catalyst) elements.

The formalism of reaction networks has proven useful in modeling a wide range of systems [1]. Reaction networks are a natural way to represent systems in which the interactions among entities are transformational in nature. In this context, the familiar chemical reaction becomes a specific example of an interaction between elements of a system which results in the quantity of these elements diminishing and the quantity of other elements increasing. The same concept is readily applicable to a wide range of systems, including ecological [2], social [3] and political [4] ones.

We seek to implement an algorithm for simulating the dynamics of a system that has been described using the formalism of reaction networks. This is done by defining a small domain specific language for representing reaction networks that matches the natural way chemical reactions are written out. This language is then parsed using regular expressions [10] to determine the underlying structure of the reaction network (stroichiometry). From this a system of ODEs is build in a form which can be integrated using the `scipy`[11] library.

There are well documented and extensive solutions to this problem ([15], [14]), which come with a plenitude of other features as well and allow for a much deeper and more complete analysis of a reaction network system. Our goal is to present a streamlined and easy to use program that can be used for quick prototyping of different configurations of reaction networks.

In the next section we introduce the main concepts of the reaction networks formalism and the construction of systems of ODEs from systems of reaction networks. In section 4, we lay out the algorithm that we use to automatically building the right hand sides of the

ODEs given the structure of the reaction network. After that we give details on the actual implementation of said algorithm in code and then give a few examples of the usage of our program to analyze a number of reaction networks.

2 Reaction network kinetics

We generalize the concept of a reaction network in the following way: the elements (labeled with capital letters $\{A, B, ..., A_i, ...\}$) and reactions are abstract and we are not concerned with the real world representation of these elements and reactions. A reaction represents interaction between members of the set of elements (reactants) that produce other elements of the set (products). A reaction is thus an ordered pair of multisets - the first one containing reactants and the second one products, along with real number denoting the *rate* of the reaction. For example:

$$A + B \xrightarrow{k} C. \tag{1}$$

Here we have a reaction between the reactants A and B that react with rate k to produce C. If an element appears on both sides of the reaction, it is said to be a *catalyst.* For example:

$$A + B \xrightarrow{k} A + C. \tag{2}$$

The element A catalyses the reaction - it's mass does not change, but it is needed for the reaction to take place.

It's important that the two sides of the reactant are multisets, i.e. a reactant or a product can appear multiple times in either side:

$$A + B \xrightarrow{k} C + C + D, \tag{3}$$

in which case the mass of the element C increase twice as fast as the mass of the element D.

Each reaction can be translated to a system of ODEs that describe the dynamics of the reaction using the mass action principle. We describe the mass of each element $A, B, ...$ by a function of time, denoted with the corresponding lower case letter $a(t), b(t), ...$. By the mass action principle the rate of change of the mass of each element is proportional to the masses of the reactants on the left side of the arrow and the rate of the reaction. Whether the mass of the element increases of decreases depends whether it's a reactant or a product. For example, for the reaction (1), we would have;

$$\begin{aligned}\dot{a}(t) &= -ka(t)b(t)\\ \dot{b}(t) &= -ka(t)b(t)\\ \dot{c}(t) &= ka(t)b(t).\end{aligned}$$

If an element is a catalyst, it appears on both sides of the reaction. In this case, it's rate of change would have two terms with opposite signs that cancel out. For the reaction (2):

$$\dot{a}(t) = -ka(t)b(t) + ka(t)b(t) = 0, \tag{4}$$

which shows that the mass of a catalyst does not change.

If an element appears more than once on either side of the reaction, this is reflected by a multiplicative factor in the equations For the reaction

$$A + B \xrightarrow{k} A + A + C$$

we have the equation:

$$\dot{a}(t) = -ka(t)b(t) + 2ka(t)b(t) = ka(t)b(t)$$

Additionally, if an element appears multiple times on the left side of the reaction, this is reflected by an exponential factor in the ODEs. For example, for the reaction

$$A + A \xrightarrow{k} B$$

we would have the equations:

$$\begin{aligned}\dot{a}(t) &= -2ka(t)^2\\ \dot{b}(t) &= ka(t)^2.\end{aligned}$$

We see that we have both the multiplicative factor 2 as in the previous point, but also an exponential factor 2 in the flow law. This can be seen intuitively from the chemical origin of the reactions. If an element appears twice as a product of a reaction, each time the reaction occurs, two units of mass of the product are produced - we have a linear dependency between the rate of change of the mass and the intensity of the reaction.

The same logic leads to the multiplicative factor if the element appears twice on the left hand side - two molecules of the element are needed for the reaction to occur. In this case however, the mass of the element also influences the probability of the reaction occurring. At very low concentrations of the reactant, the reaction becomes very unlikely to occur. This means that the rate of the change of the reactant tends asymptotically to zero. This is reflected in the exponential dependency of the negative rate of change to the reactant mass.

2.1 A system of reactions - a reaction network

We can easily extend the above to support systems of multiple reactions. If an element is present in more than one reactions, it's rate of change is the sum of the terms for each individual reaction. For example, for the system of reactions:

$$A + B \xrightarrow{k_1} C$$
$$D + C \xrightarrow{k_2} B$$

we would have the following system of ODEs (we omit the obvious dependence on the time t for brevity):

$$\begin{aligned}
\dot{a} &= -k_1 ab \\
\dot{b} &= -k_1 ab + k_2 dc \\
\dot{c} &= k_1 ab - k_2 dc \\
\dot{d} &= -k_2 dc
\end{aligned}$$

Our goal is to implement an algorithm for finding the functions on the right hand side of the system given a reaction network. This system can then be integrated using any method for integration of first order ODEs.

3 Formal definition of a reaction

In this section we will define a reaction in such a way that will allow us to more easily outlay the algorithm in the next section. Let $E = \{A_i\}_{i=1}^n$ denote the set of elements that appear in a reaction (both reactants and products). As we said earlier, a reaction can be represented as an ordered pair of multisets with elements in E. However, we also want a convenient way of determining if an element appears on either side of a reaction at all. It is more fitting to represent these mutlisets as tuples of coefficients corresponding to each element, where a coefficient of 0 signifies that an element does not belong to the multiset, i.e. does not appear in this side of the reaction.

Following the notation in [6], we define a reaction ρ as:

$$\rho : \sum_{i=1}^{n} l_i A_i \xrightarrow{k} \sum_{i=1}^{n} r_i A_i \tag{5}$$

where l_i and r_i are the coefficients of the elements in the reaction and k is the rate of the reaction. If a coefficient is 0 it means that the corresponding element does not appear on the corresponding side of the equation.

For ease of use with the alphabetical notation of the elements $(A, B, \dots)$ we also write the coefficients as functions of the elements:

$$l : E \to \mathbb{Z} \tag{6}$$

$$A_i \mapsto l(A_i) = l_i \tag{7}$$

for the reactant side and

$$r : E \to \mathbb{Z} \tag{8}$$
$$A_i \mapsto r(A_i) = r_i \tag{9}$$

for the product side. This allows us to write for example $l(B)$ without overburdening the index notation.

For example, for the reaction

$$A + A + B \xrightarrow{k} B + B + C + D$$

the set of all elements is $E = \{A, B, C, D\}$. The left hand side (reactant) coefficients are:

$$l(A) = 2, \quad l(B) = 1,$$
$$l(C) = l(D) = 0,$$

while the right hand side (product) coefficients are:

$$r(B) = 2, \quad r(C) = 1$$
$$r(D) = 1, \quad r(A) = 0.$$

4 Building systems of ODEs from reaction networks

In this section we will lay out a procedure for building the ODEs governing the dynamics of a reaction network system. In this we follow [5] along with an application of our definitions from the previous section.

4.1 Single reaction

Let's first look at an example of how the system of ODEs for a single reaction is built. We have the reaction:

$$A + A + B \xrightarrow{k} A + C$$

The corresponding system is:

$$\begin{aligned}
\dot{a}(t) &= -2ka(t)^2b(t) + ka(t)^2b(t) = -ka(t)^2b(t) \\
\dot{b}(t) &= -ka(t)^2b(t) \\
\dot{c}(t) &= ka(t)^2b(t)
\end{aligned}$$

As we can see, all the ODEs are of the form $\dot{f}(t) = s_f ka(t)^2b(t)$, where $s_f \in \mathbb{Z}$. We call

$$\varphi(t) := ka(t)^2b(t)$$

the *flow law* of the reaction. It represents the intensity of the reaction and it is determined only by the masses of the reactants (left side of the reaction) and the rate k of the reaction. If our reactions represent interaction between discrete elements (ex. molecules in a solution), we can think of the rate law as a number of interaction per unit time.

Given the rate law, we can determine the rate of change of the masses of the elements by looking at the amounts of an element that are consumed or produced by the reaction. In our example reaction, one unit of the element C is produced, so $s_c = 1$ and $\dot{c}(t) = ka(t)^2b(t)$. Similarly, one unit of the element B is consumed by the reaction, so $s_b = -1$. For the element A, two units of mass are needed for the reaction to occur and one unit of mass is left after the reaction. It follows that $s_a = -2 + 1 = -1$. As we can see, the coefficient s_f in the ODEs corresponds to the difference of the coefficients $r(A) - l(A)$.

More rigorously, let $E = \{A_i\}_{i=1}^n$ be the elements of a reaction. A reaction ρ is a 3-tuple (l, r, k), where l and r are the coefficient functions. Let's denote:

$$s_i = r(A_i) - l(A_i),$$

which we will call the *stoichiometry* of the element A_i. Let $a_i(t)$ be the mass of the element A_i at time t. Then the rate law of the reaction is:

$$\varphi(t) = k \prod_{j=1}^{n} a_j(t)^{l(A_j)}$$

and the rate of change of the mass $a_i(t)$ is:

$$\dot{a}_i(t) = s_i \varphi(t)$$

Both when generalizing for a system of reactions and when implementing the algorithm in code, it will be easier to work in vector notation. Without loss of generality, we can order the elements in an n-tuple $\boldsymbol{A} = (A_1, \ldots, A_n)$. Using this ordering we can define the vector of masses $\boldsymbol{a}(t) = (a_1(t), \ldots, a_n(t))^T$ and a *stoichiometry vector* of the reaction $\boldsymbol{s} = (s_1, \ldots, s_n)^T$. Then:

$$\dot{\boldsymbol{a}}(t) = \varphi(t)\boldsymbol{s} \tag{10}$$

We can see that the rate law $\varphi(t)$ of a reaction depends only on the left side of the reaction. That is to say, the intensity of the reaction does not depend on the concentrations of the products of the reaction (unless those products are also reactants). The stoichiometry vector on the other hand gives us a notion of the effects of the reaction on the masses of the elements and it depends on both sides of the reaction.

4.2 Reaction network

The generalization of the above to a reaction network is straightforward. As we said, the rate of change of an element that is present in more than one reaction is just the sum of the rates of change that are caused by each reaction individually. A reaction network is a set of reactions

$$\{\rho_i = (l_i, r_i, k_i)\}_{i=1}^{p} \tag{11}$$

between a set of elements E. Here l_i and r_i are the reactant and product coefficient functions for the i-th reaction.

As we did for the elements we can order the reaction in a p-tuple $\boldsymbol{\rho} = (\rho_1, \ldots, \rho_p)$ without loss of generality. Using this we can define a *vector of rate laws* of the reaction network $\boldsymbol{\varphi}(t) = (\varphi_1(t), \ldots, \varphi_p(t))^T$, where

$$\varphi_j(t) = k_j \prod_{i=1}^{n} a_i(t)^{l_j(A_i)}.$$

For a single reaction we labeled as s_i the stoichiometry of the element A_i. For a reaction network we define

$$s_{ij} := r_j(A_i) - l_j(A_i)$$

the stoichiometry of the element A_i in the reaction ρ_j.

The rate of change of the element A_i due to the reaction ρ_j is $s_{ij}\varphi_j(t)$. The rate of change of an element in a reaction network is the sum of the rates of change due to each individual reaction:

$$\dot{a}_i(t) = \sum_{j=1}^{p} s_{ij}\varphi_j(t)$$

In vector notation the rate of change of the mass vector $\boldsymbol{a}(t) = (a_1(t), \dots, a_n(t))^T$ is:

$$\dot{\boldsymbol{a}}(t) = \boldsymbol{S} \cdot \boldsymbol{\varphi}(t), \tag{12}$$

where we have defined $\boldsymbol{S} = (s_{ij}) \in \mathbb{Z}^{(n \times p)}$ as the *stoichiometry matrix* of the reaction network.

We have arrived at our desired representation of the system of ODEs that describes the dynamics of a reaction network. In the next section we will describe briefly how this is implemented in the programming language `python` [8].

5 Implementation

The functionality that leverages this algorithm can be found in the git repository `https://github.com/PetarChernev/reaction-networks` hosted on GitHub [9]. Here we will give a quick overview of the implementation.

1. We take as input the definitions of the reaction network separated by `;` (and optionally newlines) and an array of numbers for the rates of the reactions. For example, the Gompertz reaction networks is given by:

   ```
   S + X -> 2X + S;
   S -> Q
   ```

   with some rates

   ```
   rates = [1, 0.2]
   ```

2. Using regex [10] we determine the set of all elements E and define the reactions in a manner that complies with the definition of reaction (5). In `python` this can be done out of the box using the `collection.Counter` builtin class. It accepts a collection of items (in our case capital letter characters) and gives you the count of how many times each item is encountered in the collection, returning 0 if the item is not encountered in the collection.

3. Using the above `Counter` objects, we calculate the stoichiometry matrix of the reaction network, which is static throughout the integration of the system.

4. The rate laws for the reaction network are expressed as a product of the masses of the elements as functions of time. They are defined as a callback function that accepts the mass vector of the system as an argument. This function is called at each step of the integration to determine the rate laws at this time.

5. Given a vector of initial masses, the resulting IVP is integrated using the functionality in the `scipy`[11] library, namely `scipy.integrate.solve_ivp`. By default this function uses an explicit Runge-Kutta method of order 5(4).

6. The result of the integration can then be plotted using the `matplotlib`[13] package.

6 Example usage

In this section we give a few examples of the usage of our implementation.

6.1 $x(t)$ for the logistic, Gompertz and mixed models

In this section we follow the definitions in [7] and show how our implementation can be used to simulate and visualize the models in question.

We are interested in examining the differences between the logistic and the Gompertz growth and decay models. The logistic model is defined by the reaction network:

$$S + X \xrightarrow{k} 2X,$$

meaning that X self catalyses the reaction. The reaction networks of the Gompertz model is

$$\begin{gathered} S + X \xrightarrow{k} 2X + S \\ S \xrightarrow{\nu} Q \end{gathered}$$

Here both X and S are catalysts of the first reaction. The second reaction represents a decay of S with time. Q is an external reactant - its mass does not influence the dynamics of the system and its mass and rate of change are ignored when building the ODEs of the system.

Additionally, we examine a mixed model. In the logistic model the element S turns into the element X with time, which leads to an eventual equilibrium of the system. In the Gompertz model the element S is not lost in the first reaction, but rather decreases with time into the external element Q, which again leads to equilibrium, as it is a catalyst of the first reaction. We can combine the two models in the mixed model:

$$S + X \xrightarrow{k} 2X + S$$
$$S \xrightarrow{\nu} X$$

Here again S decreases over time and it is required for both reaction to occurs, which leads to equilibrium. However there is not external element Q - the second reaction turns S into X.

From [7] we know that the dynamical systems of two variables induced by the reaction networks in question have a first integral of the system which remains constant during the evolution of the system:

$$c(s(t), x(t)) = c(s(0), x(0)), \quad \forall t \tag{13}$$

The exact forms of $c(s, x)$ are such that for any chosen $x(0)$, we can choose $s(0)$, such that $c(s(0), x(0)) = c(0, 1)$.

For the three models we are examining $s(\infty) = 0$. Using the second property we can make sure that $x(t)$ tends asymptotically to 1 for any value of $x(0)$. This allows is to plot the trajectories of $x(t)$ for the three models on the same scale and compare their behaviour. Some example plots are given below (Fig. 1, 2, 3)

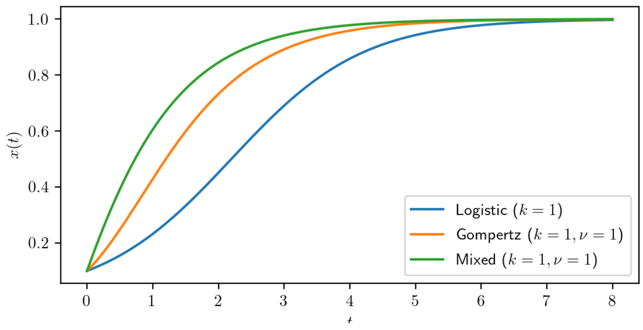

Figure 1: $x(t)$ for the logistic, Gompertz and mixed models with all rates equal to 1

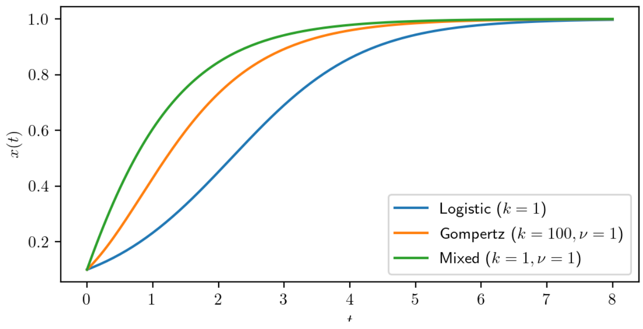

Figure 2: $x(t)$ for the logistic, Gompertz and mixed models. We can see that just changing k for the Gompertz model does not affect the trajectory, because of the scaling of $c(s(0), x(0))$ that we do to keep the asymptotic behavior.

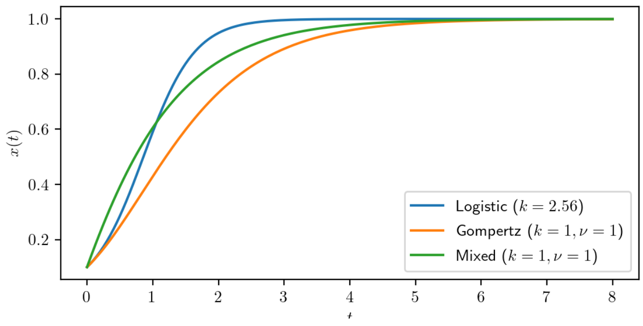

Figure 3: $x(t)$ for the logistic, Gompertz and mixed models

6.2 Vector fields of the dynamic system for the logistic and Gompertz models.

Every system of ODEs of the form

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

defines a vector field over the state space of possible values of $\boldsymbol{x}$. Using equation (12), we can do this for our reaction networks. In Fig. 4 and Fig. 5 we have the vector fields for the logistic and the Gompertz models respectively. We can see that the vector fields reflect the integral curves for the systems that arise from the constant functions $c(s(t), x(t))$ mentioned in the previous point. For the logistic model we have [7] :

$$c(s(t), x(t)) = s(t) + x(t) = \text{const.}$$

and so we see the integral curves are straight lines $x + s - c$. For the Gompertz model we have [7]:

$$c(s(t), x(t)) = \frac{k}{\nu} s(t) + \ln x(t) = \text{const.}$$

and the integral curves are of the form $\frac{k}{\nu} s(t) + \ln x(t) - c$.

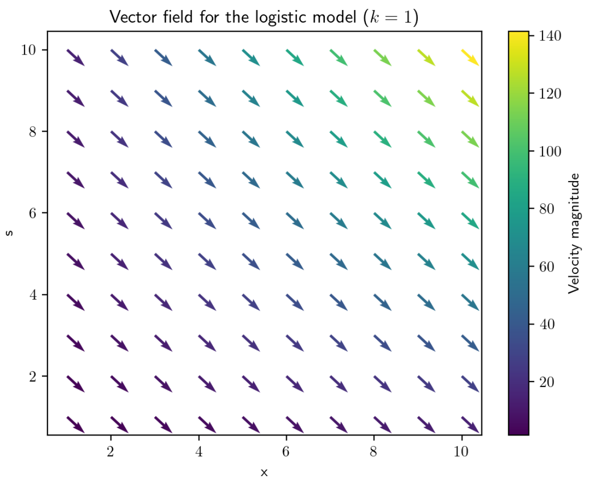

Figure 4: The vector field corresponding to the dynamic system for the logistic model.

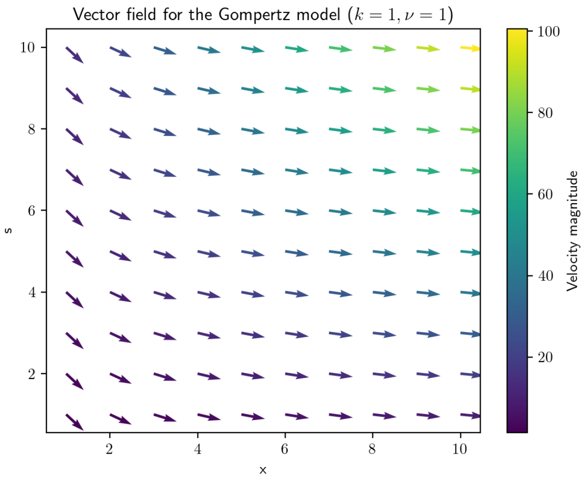

Figure 5: The vector field corresponding to the dynamic system for the Gompertz model.

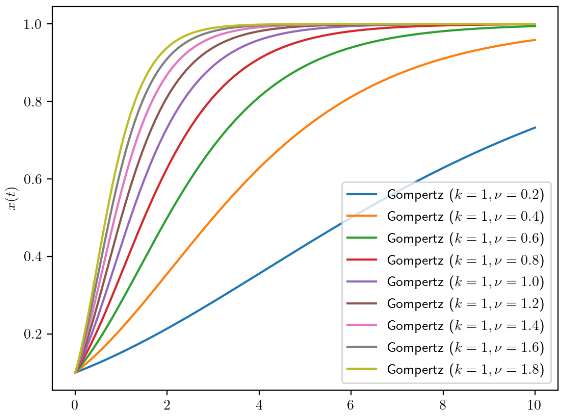

Figure 6: $x(t)$ for the Gompertz for different values of ν.

6.3 $x(t)$ for the Gompertz model with different values of ν.

In Fig. 6 we plot the trajectory of $x(t)$ for the Gompertz model for different values of the second reaction coefficient ν.

6.4 Comparison of the trajectory of $x(t)$ for the logistic and Gompertz models.

As we can see in Fig. 1, for $k_{logistic} = k_L = 1$ and $k_{Gompertz} = k_G = 1, \nu_G = 1$, $x_L(t)$ for the logistic model is always lower than or equal to $x_G(t)$ for the Gompertz model. In Fig. 3 however, we have increased k_L to 2.56. This causes the logistic model $x_L(t)$ to cross over the Gompertz model $x_G(t)$. We are interested in seeing for which values of the parameters k_L and ν there is a crossover and for which there is an inequality for all t. We can do this by integrating the systems for different values of the parameters and comparing the solutions. The result of this computation is shown in Fig. 7.

This numerical integration of pairs of system is really computationally expansive. We can examine this behavior by calculating the difference between the trajectories directly. From [7] we know that $x_L(t)$ can be expressed as:

$$x_L(t) = \frac{x_0}{(1 - x_0)e^{-k_L t} + x_0}$$

and $x_G(t)$ as:

$$x_G(t) = x_0^{\exp(-\nu t)}$$

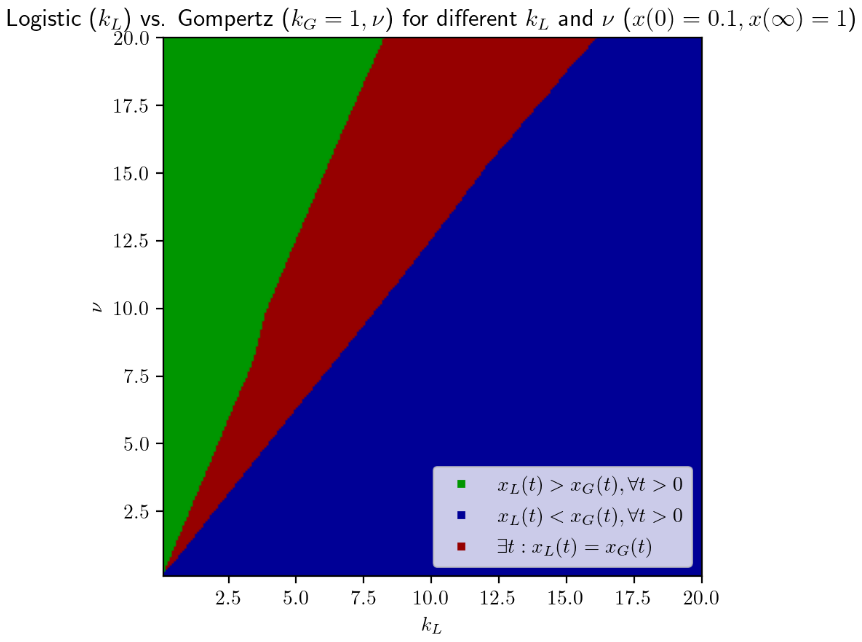

Figure 7: A comparison of the trajectory of $x(t)$ for the logistic and the Gompertz models for different values of k_L and ν by numerical integration of pairs of systems.

We are interested in the sign of the difference:

$$\begin{aligned}
\delta(t; k_L, \nu) &= x_L(t) - x_G(t) \\
&= \frac{x_0}{(1 - x_0)e^{-k_L t} + x_0} - x_0^{\exp(-\nu t)} \\
&= \frac{x_0}{e^{-k_L t} + (1 - e^{-k_L t})x_0} - x_0^{\exp(-\nu t)} \\
&= \frac{x_0 - e^{-k_L t} x_0^{\exp(-\nu t)} - (1 - e^{-k_L t})x_0^{1+\exp(-\nu t)}}{e^{-k_L t} + (1 - e^{-k_L t})x_0}
\end{aligned}$$

The denominator is strictly positive. By calculating the numerator for different values of t at different pairs of values for the pa-

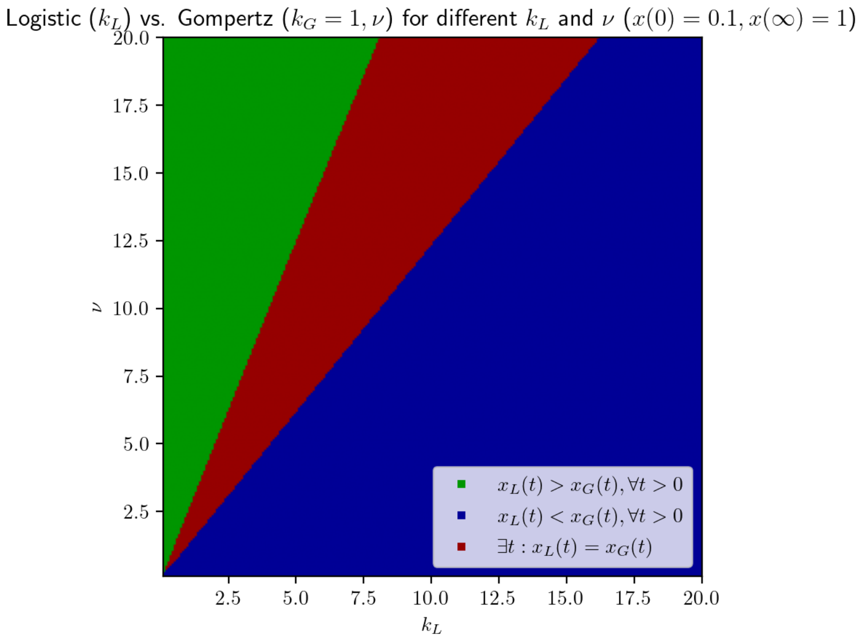

Figure 8: A comparison of the trajectory of $x(t)$ for the logistic and the Gompertz models for different values of k_L and ν by computation of the theoretical difference $\delta(t) = x_L(t) - x_G(t)$.

rameters (k_L, ν), we can examine the relative behavior of the two trajectories. The result of this computation are given in Fig. 8

We can see from the two figures that we get similar results. There is a slight nonlinearity in the crossover region for the figure derived from integration of pairs of systems. This is due to the fact that both trajectories of the pairs of systems tend very quickly to 1 for higher values of the parameters, and so the computed difference $\delta(t; k_L, \nu)$ is very close to 0 for the majority of the time interval. This causes inaccuracies in detecting a crossover. This is yet another advantage of computing an explicit theoretical formula for the difference.

6.5 Gompertz-Bateman reaction network

We apply the algorithm to the Gompertz-Bateman growth-decay model [12]:

$$S \xrightarrow{k_1} P \xrightarrow{k_2} Q$$

$$P + X \xrightarrow{k} 2X + P,$$

where Q is an external reactant. We can also use the program to print the system of ODEs that is generated by the reaction network in question:

```
p'=k_1s-k_2p
s'=-k_1s
x'=kpx
```

or generate valid LaTeX for an `align` environment:

$$\begin{aligned} \dot{p}(t) &= k_1 s - k_2 p \\ \dot{s}(t) &= -k_1 s \\ \dot{x}(t) &= kpx \end{aligned}$$

This dynamical system has a conservation relation:

$$c(t) = \frac{k}{k_2}(p - s) + \ln x = const. \tag{14}$$

We can use this first integral of the system to determine the initial conditions in such a way that the system reaches a steady state with $x(\infty) = 1, p(\infty) = s(\infty) = 0$:

$$\begin{aligned} c(0) &= \frac{k}{k_2}(p_0 + s_0) + \ln x_0 = c(\infty) = 0 \\ x_0 &= e^{-\frac{k}{k_2}(p_0+s_0)} \end{aligned}$$

If we choose initial masses $p_0 = s_0 = 1$, we see that we need to set $x_0 = e^{-2\frac{k}{k_2}}$ to achieve the desired behaviour $x(\infty) = 1$. A plot of a numerical simulation of the system with these initial conditions and reaction rates $k = k_1 = k_2 = 1$ is shown in Fig. 9.

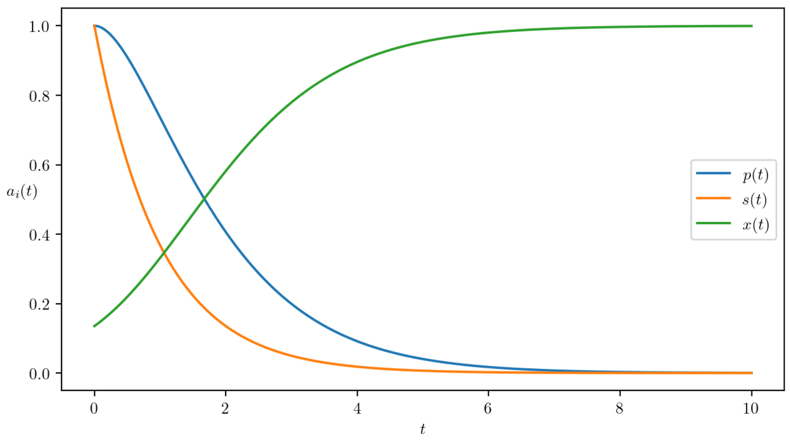

Figure 9: The mass trajectories for the Gompertz-Bateman growth-decay model with a steady state at $x(\infty) = 1$ and reaction rates $k = k_1 = k_2 = 1$.

6.6 Modified Gompertz-Bateman growth-decay model

We can modify the above model by removing the external reactant Q and having the reactant P turn into X directly, as well as being a catalyst to the growth function:

$$S \xrightarrow{k_1} P \xrightarrow{k_2} X$$
$$P + X \xrightarrow{k} 2X + P,$$

The induced dynamic system is:

$$\begin{aligned}\dot{p}(t) &= k_1 s - k_2 p\\ \dot{s}(t) &= -k_1 s\\ \dot{x}(t) &= kpx + k_2 p\end{aligned}$$

Proposition 1. *The above system has a conservation relation:*

$$c(t) = s(t) + p(t) + \frac{1}{\gamma}\ln(\gamma x(t) + 1), \tag{15}$$

where $\gamma = \frac{k}{k_2}$.

Proof. We take the sum:

$$\dot{s} + \dot{p} = -k_2 p = -k_2 \frac{\dot{x}}{kx + k_2} = \frac{\dot{x}}{\gamma x + 1}$$

Integrating this gives:

$$\begin{aligned} s(t) + p(t) &= -\int_0^t \frac{dx}{d\tau}\frac{1}{\gamma x + 1} d\tau + c \\ &= -\frac{1}{\gamma}\int_0^t \frac{d(\gamma x + 1)}{d\tau}\frac{1}{\gamma x + 1} d\tau + c \\ &= -\frac{1}{\gamma}\int_0^t \frac{d\ln(\gamma x + 1)}{d\tau} d\tau + c \\ &= -\frac{1}{\gamma}\ln(\gamma x(t) + 1) + c \end{aligned}$$

This gives the first integral:

$$c = s(t) + p(t) + \frac{1}{\gamma}\ln(\gamma x(t) + 1), \tag{16}$$

which remains constant throughout the integration of the system. □

Using the above relation and having fixed s_0 and p_0, we can once again choose x_0 such that $x(\infty) = 0$. For the initial conditions we have:

$$c(0) = s_0 + p_0 + \frac{1}{\gamma}\ln(\gamma x_0 + 1). \tag{17}$$

The mass $s(t)$ is strictly decreasing, so in the steady state we have $s(\infty) = 0$. The mass $p(t)$ has a positive term in the derivative, which is proportional to $s(t)$. In the limit it is also strictly decreasing:

$$\lim_{t\to\infty} \dot{p}(t) = -k_2 p(t),$$

so again in the steady state we have $p(\infty) = 0$. In a sense the reaction network "uses up" the reactants S and P to produce the reactant X. We want to impose the condition $x(\infty) = 1$. In the steady state, the conservation relation is:

$$c(\infty) = \frac{1}{\gamma} \ln(\gamma + 1) \tag{18}$$

Since we have $c(0) = c(\infty)$, we can solve (17) and (18) for x_0:

$$x_0 = \frac{1}{\gamma}\Big(\frac{\gamma + 1}{e^{\gamma(s_0 + p_0)}} - 1\Big) \tag{19}$$

The masses of the reactants are non-negative, so we must have $x_0 \geq 0$. This imposes a constraint on the possible initial conditions for s_0 and p_0 such that $x(\infty) = 1$:

$$s_0 + p_0 \leq \frac{1}{\gamma} \ln(\gamma + 1). \tag{20}$$

In Fig. 10 we can see a plot of the trajectories of the masses for a modified Gompertz-Bateman system with $k = k_1 = k_2 = 1$. With these reaction rates, the constraint (20) becomes:

$$s_0 + p_0 \leq \ln 2 = 0.693. \tag{21}$$

For the plotted solution we have chosen initial conditions $s_0 = 0.5$, $p_0 = 0$, which leads to a choice of $x_0 = 0.213$ under our condition $x(\infty) = 1$.

We can see the the mass of the reactant P increases at first due to the reaction $S \xrightarrow{k_1} P$, which causes an acceleration in the growth of the reactant X. That is until a maximum is reached, after which the time derivative of $p(t)$ is dominated by the $-k_2 p$ term. After that point, both $s(t)$ and $p(t)$ decrease until a steady state is reached.

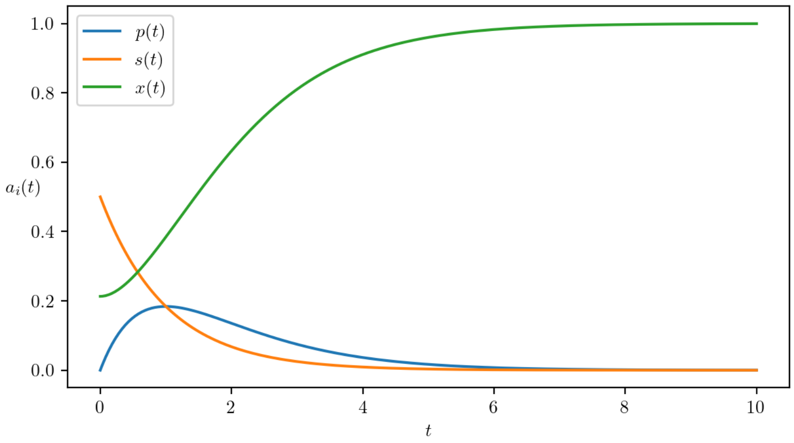

Figure 10: The mass trajectories for the modified Gompertz-Bateman growth-decay model with a steady state at $x(\infty) = 1$ and reaction rates $k = k_1 = k_2 = 1$.

7 Conclusion

In the present work, we have out lined a straightforward algorithm for building a system of ODEs from a reaction network. We have implemented this algorithm in a GitHub repository along with functionality to solve the IVPs given by said systems of ODEs and plot the solutions in different forms and show some example applications of the algorithm. We also implement functionality for choosing the initial conditions based on asymptotic conditions on the mass trajectories at infinity by use of first integrals of the system.

Compared to well known and established systems in the same domain ([15], [14]), our implementation is extremely simple. This simplicity however, provides a streamlined and simple to use tool for quick iteration over different configurations of reaction networks and their associated dynamical systems.

References

[1] Tomas Veloz and Pablo Razeto-Barry. *Reaction Networks as a Language for Systemic Modeling: Fundamentals and Examples*, Systems 2017, 5(1), 11
`https://doi.org/10.3390/systems5010011`

[2] Pascual, M.; Dunne, J.A. *Ecological Networks: Linking Structure to Dynamics in Food Webs*; Oxford University Press: Oxford, UK, 2006.

[3] Peter Dittrich and Lars Winter. *Reaction Networks as a Formal Mechanism to Explain Social Phenomena*, Proceeding of The Fourth International Workshop on Agent-based Approaches in Economics and Social Complex Systems (AESCS 2005) July 9-13, 2005, Tokyo Institute of Technology, Japan, p. 433–446, 2005
`https://users.fmi.uni-jena.de/~dittrich/p/DW2005.pdf`

[4] Peter Dittrich and Lars Winter. *Chemical Organizations in a Toy Model of the Political System*, Adv. Complex Syst. 2008, 11, 609–627 `http://users.minet.uni-jena.de/~dittrich/p/DW2007eccs.pdf`

[5] Luca Cardelli. *From Processes to ODEs by Chemistry*, IFIPAICT, volume 273
`https://link.springer.com/chapter/10.1007/978-0-387-09680-3_18`

[6] Hans-Michael Kaltenbach, *A unified view on bipartite species-reaction and interaction graphs for chemical reaction networks*, `https://arxiv.org/abs/1210.0320`

[7] S. Markov, *The Gompertz Model Revisited and Modified Using Reaction Networks: Mathematical Analysis*, Biomath X, 2 (2021), accepted for publication

[8] Python 3.9, Python Software Foundation,
`https://www.python.org/`

[9] `github.com`, GitHub, Inc., see Introduction to GitHub

[10] Aho, A. V. (1991). *Algorithms for finding patterns in strings*, Handbook of theoretical computer science (vol. A): algorithms and complexity. MIT Press, Cambridge, MA.
`https://mitpress.mit.edu/books/handbook-theoretical-computer-science-volume`

[11] Pauli Virtanen, Ralf Gommers et al. (2020) *SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python*, Nature Methods, 17(3), 261-272.
`https://www.nature.com/articles/s41592-019-0686-2`

[12] S. Markov (2019). *On a Class of Generalized Gompertz-Bateman Growth-decay Models*, Biomath Communications.
`https://www.biomathforum.org/biomath/index.php/conference/article/view/1311`

[13] J. D. Hunter, *Matplotlib: A 2D Graphics Environment*, Computing in Science & Engineering, vol. 9, no. 3, pp. 90-95, 2007.
`https://ieeexplore.ieee.org/document/4160265`

[14] Stefan Hoops, Sven Sahle, Ralph Gauges, Christine Lee, Jürgen Pahle, Natalia Simus, Mudita Singhal, Liang Xu, Pedro Mendes, Ursula Kummer. *COPASI—a COmplex PAthway SImulator*, Bioinformatics, Volume 22, Issue 24, 15 December 2006, Pages 3067–3074
`https://doi.org/10.1093/bioinformatics/btl485`

[15] Laurence Calzone, François Fages, Sylvain Soliman. *BIOCHAM: an environment for modeling biological systems and formalizing experimental knowledge*, Bioinformatics, Volume 22, Issue 24, 15 December 2006, Pages 3067–3074
`https://doi.org/10.1093/bioinformatics/btl485`