

Article

Not peer-reviewed version

---

# Automatic Generation of SBML Kinetic Models from Natural Language Texts using GPT

---

[Kazuhiro Maeda](#)<sup>\*</sup> and [Hiroyuki Kurata](#)

Posted Date: 7 March 2023

doi: 10.20944/preprints202303.0122.v1

Keywords: GPT; language model; kinetic modeling; simulation; systems biology; natural language processing



Preprints.org is a free multidiscipline platform providing preprint service that is dedicated to making early versions of research outputs permanently available and citable. Preprints posted at Preprints.org appear in Web of Science, Crossref, Google Scholar, Scilit, Europe PMC.

Copyright: This is an open access article distributed under the Creative Commons Attribution License which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

## Article

# Automatic Generation of SBML Kinetic Models from Natural Language Texts Using GPT

Kazuhiro Maeda <sup>1\*</sup>  and Hiroyuki Kurata <sup>1</sup> 

<sup>1</sup> Department of Bioscience and Bioinformatics, Kyushu Institute of Technology, 680-4 Kawazu, Iizuka, Fukuoka, 820-8502, Japan

\* Correspondence: kmaeda@bio.kyutech.ac.jp

**Abstract:** Kinetic modeling is an essential tool in systems biology research, enabling the quantitative analysis of biological systems and predicting their behavior. However, the development of kinetic models is a complex and time-consuming process. In this article, we propose a novel approach called KinModGPT, which generates kinetic models directly from natural language text. KinModGPT employs GPT-3 as a natural language interpreter and Tellurium as an SBML generator. We demonstrate the effectiveness of KinModGPT in creating SBML models from complex natural language descriptions of biochemical reactions. KinModGPT successfully generates valid SBML models from a range of natural language model descriptions of metabolic pathways, protein-protein interaction networks, and heat shock response. This article demonstrates the potential of KinModGPT in kinetic modeling automation.

**Keywords:** GPT; language model; kinetic modeling; simulation; systems biology; natural language processing

## 1. Introduction

Kinetic modeling is an essential technique in systems biology, as it allows for the quantitative analysis and prediction of complex biochemical systems such as metabolic pathways and gene regulatory networks [1]. Kinetic models are differential equation models that describe the dynamic behavior of biochemical systems based on the interactions between their molecular components. Kinetic models are stored and distributed as SBML (Systems Biology Markup Language) files [2,3]. In addition, there are several software tools, such as Tellurium [4,5], COPASI [6–8], and CADLIVE [9–11], that support kinetic modeling. Still, the development of kinetic models is time-consuming and expert work. First, modelers survey many research articles for known enzyme reactions, signal transduction pathways, and gene regulations contained in the system of interest and build the chemical reaction equations with kinetic parameters. Next, modelers translate them into kinetic rate equations. A few studies have been devoted to developing natural language processing-based methods that generate kinetic models from literature texts [12,13]. However, these studies focused on signal transduction, a specific type of biochemical systems. Moreover, their complex implementation makes it challenging for non-experts to modify or extend.

In recent years, there has been significant progress in artificial intelligence (AI) technologies, particularly in the field of generative models. Among these models, GPT (Generative Pre-trained Transformer) natural language models stand out as the most advanced. In fact, ChatGPT [14], one of the GPT models, has gained public attention due to the ability to engage in natural conversations. It has been shown that GPT models have the capability to pass graduate-level exams [15,16] and write program codes [17]. GPT models may allow researchers to automatically build kinetic models directly from research articles. To the best of our knowledge, the capabilities of GPT models for automatic kinetic model construction have not been investigated.


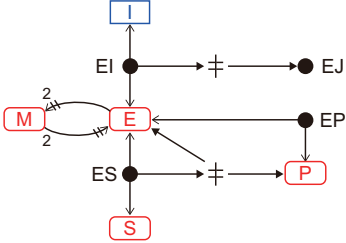
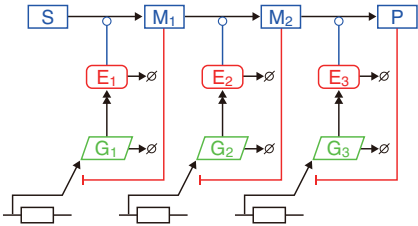
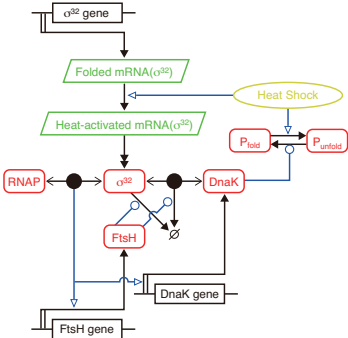
In this article, we aim to answer the following question: Can a GPT model generate an SBML kinetic model from a natural language description? While GPT-3 [18] alone cannot create valid SBML models, we propose a novel approach called KinModGPT, which combines GPT-3 as a natural language

interpreter and Tellurium as an SBML generator. Moreover, we demonstrate that KinModGPT can generate valid SBML models from natural language descriptions of biochemical reactions.

## 2. Results

### 2.1. GPT-3 alone cannot create valid SBML models

GPT-3 is a large language model trained on massive amounts of text data [18]. When prompted with a message to start the conversation, GPT-3 generates appropriate responses by repeatedly predicting the following word in the sequence. First, we tested whether GPT-3 can directly generate an SBML model from natural language text. We employed four test problems with biochemical systems with different complexities (Figure 1). Next, we asked GPT-3 to convert each model description to an SBML model. The instruction message for GPT-3 is shown in Scheme A1. The results are summarized in Table 1. For all the models tested, GPT-3 generated SBML-like models. However, upon inspecting these models using the Online SBML validator [19], we discovered that all the generated SBML models were invalid, containing some errors, such as missing required attributes. The invalid SBML files cannot be imported by widely-used modeling tools.

Model name	Network map	Model description
Decay		Protein P decays. The initial concentration is 1 uM.
HIV		Molecules of M bind to form E, and E dissociates back into two Ms. Additionally, E and S can bind to form ES, which then dissociates back into E and S, while E and P can bind to form EP, which dissociates back into E and P. Furthermore, ES can be converted into E and P. E and I can also bind to form EI, which dissociates back into E and I. Finally, EI is converted into EJ.
Three-step		Substrate S is converted into product P through intermediates M1 and M2. The metabolic reactions are catalyzed by three enzymes, E1, E2, and E3. The expression of mRNAs G1, G2, and G3 is repressed by the metabolites M1, M2, and P, respectively. The proteins E1, E2, and E3 are translated from G1, G2, and G3, respectively. E1, E2, E3, G1, G2, and G3 degrade. The initial concentration of S is 1.
Heat shock response		s70 and RNA polymerase (RNAP) bind together to form s70_RNAP, which then dissociates into s70 and RNAP. Pg and s70_RNAP bind to create Pg_s70_RNAP, which can dissociate back into its components. RNAP and s32 bind to create RNAP_s32, which can dissociate into RNAP and s32. Ph and RNAP_s32 bind to form Ph_RNAP_s32, which then dissociates back into Ph and RNAP_s32. Additionally, s32 and DnaK form s32_DnaK, which can dissociate into s32 and DnaK, while s32 and FtsH bind to create s32_FtsH, which can dissociate back into s32 and FtsH. Similarly, Punfold and DnaK bind to form Punfold_DnaK, which can dissociate into Punfold and DnaK. (continued)

**Figure 1.** Test problems. We tested whether KinModGPT can create SBML models from the natural language model descriptions. For the reaction network maps, CADLIVE notation was used [9–11]. For simplicity, only important reactions are shown in the reaction network map for the heat shock response model. The full model description for the heat shock response is provided in Scheme A3.

**Table 1.** Summary of the computational experiments. We converted four natural language model descriptions into SBML using either the GPT-3 only approach or KinModGPT. The validity of the generated SBML files was verified using the Online SBML Validator [19], and their consistency with the original model descriptions was manually inspected.

Method	Model name	Are SBML models created?	Are the created SBML models valid?	Are the created SBML models consistent with their model descriptions?
GPT-3 only	Decay	Yes	No	N/A
	HIV	Yes	No	N/A
	Three-step	Yes	No	N/A
	Heat shock response	Yes	No	N/A
KinModGPT	Decay	Yes	Yes	Yes
	HIV	Yes	Yes	Yes
	Three-step	Yes	Yes	Yes
	Heat shock response	Yes	Yes	Yes

2.2. KinModGPT can create valid SBML models

To generate an SBML model from natural language text, we introduce the strategy named KinModGPT, as outlined in Figure 2. First, KinModGPT translates the natural language descriptions of biochemical reactions into Antimony language [20], a human-readable model definition language. Next, KinModGPT converts the resulting Antimony model into the SBML model using Tellurium [4,5]. For KinModGPT, we provided the following prompt to GPT-3. First, we told GPT-3 that the task was to translate the descriptions of biochemical reactions written in natural language into Antimony. Second, we provided a conversion rule table that showed how each chemical reaction is represented in natural language and Antimony (Scheme A2). Then, we input the natural language model descriptions. The Python code of KinModGPT and the created Antimony and SBML models are provided as Supplementary Materials.

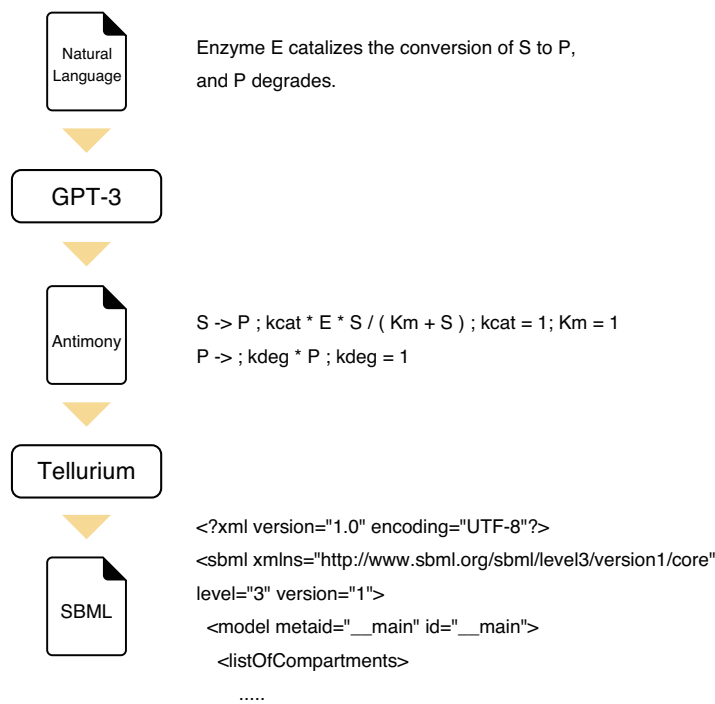


Figure 2. Overview of KinModGPT.

We tested whether KinModGPT can create valid SBML models from text model descriptions. KinModGPT successfully generated an SBML model for each test problem. We confirmed their validity using the Online SBML validator (Table 1). Indeed, these SBML models could be imported by a widely-used modeling tool, COPASI [6–8], demonstrating that KinModGPT can create ready-to-simulate SBML models from natural language text.

To further examine the models, we intensively analyzed each model. The Antimony model for decay is shown in Scheme 1. It is worth noting that KinModGPT does not require an exact match between the conversion rules and model descriptions. Indeed, KinModGPT successfully interpreted “Protein P decays. The initial concentration is 1 uM.” as the combination of the two expressions: “X degrades (or decays)” and “X (concentration) is Y M (or mM or uM or nM or pM).” The generated SBML model could be simulated as it was (Figure 3).



Scheme 1. The decay model in Antimony language, created by KinModGPT.

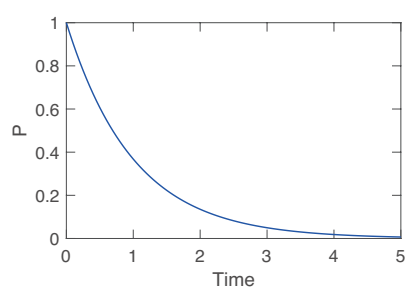


Figure 3. Simulation of the created SBML model for the decay model.

In the model description of the HIV model (Figure 1), five sentences describe ten reactions. Indeed, the second sentence contains four reactions. KinModGPT extracted the necessary information from

these complex sentences and generated a valid SBML model. The Antimony model is provided in Scheme 2. The SBML model could be simulated by modeling tools without any modifications. However, to check whether the SBML model reproduces the behavior of the HIV proteinase system, we manually set known realistic values to kinetic parameters. As shown in Figure 4, the substrate (S) is converted into the product (P) over time. Moreover, the inactive enzyme-inhibitor complex (EJ) gradually increases. This behavior is consistent with the network map (Figure 1), model description, and literature [21–23].

---

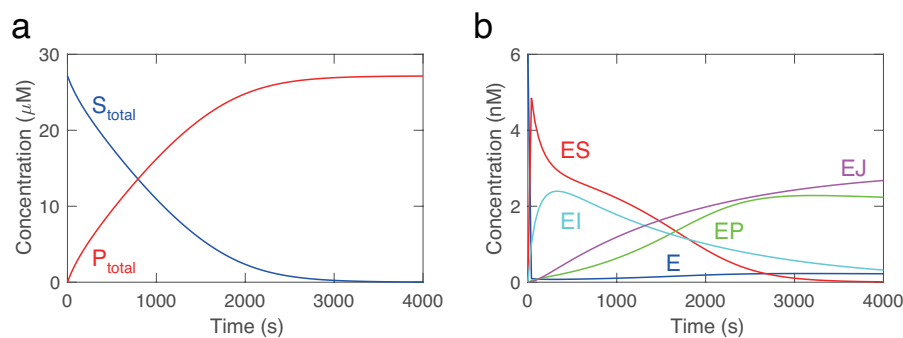
```

M + M -> E ; ka_M_M_E * M * M ; ka_M_M_E = 1
E -> M + M ; kd_E_M_M * E ; kd_E_M_M = 1
E + S -> ES ; ka_E_S_ES * E * S ; ka_E_S_ES = 1
ES -> E + S ; kd_ES_E_S * ES ; kd_ES_E_S = 1
E + P -> EP ; ka_E_P_EP * E * P ; ka_E_P_EP = 1
EP -> E + P ; kd_EP_E_P * EP ; kd_EP_E_P = 1
ES -> E + P ; kc_ES_E_P * ES ; kc_ES_E_P = 1
E + I -> EI ; ka_E_I_EI * E * I ; ka_E_I_EI = 1
EI -> E + I ; kd_EI_E_I * EI ; kd_EI_E_I = 1
EI -> EJ ; kc_EI_EJ * EI ; kc_EI_EJ = 1

```

---

**Scheme 2.** The HIV model in Antimony language, created by KinModGPT.



**Figure 4.** Simulation of the created SBML model for the HIV model.  $S_{total}$  and  $P_{total}$  represent the total S concentration ( $S_{total} = S + ES$ ) and the total P concentration ( $P_{total} = P + EP$ ), respectively. We tuned the kinetic parameters before simulation.

The three-step problem highlights the remarkable capability of KinModGPT. The model description's first two complex sentences, "Substrate S is converted into product P through intermediates M1 and M2. The metabolic reactions are catalyzed by three enzymes, E1, E2, and E3.", contain four metabolites and three enzyme reactions. These sentences may be challenging even for experienced modelers to interpret. However, KinModGPT successfully interpreted and translated the sentences into the Antimony model (Scheme 2). Next, we tested whether the created SBML model could reproduce a reasonable behavior. As shown in Figure 5, the substrate (S) is converted into the product (P) through two intermediate metabolites (M1 and M2). The expression of the third enzyme (E3) is repressed compared to the first and second enzymes because the accumulated P represses the expression of E3. This behavior matches the network map and model description shown in Figure 1.

---

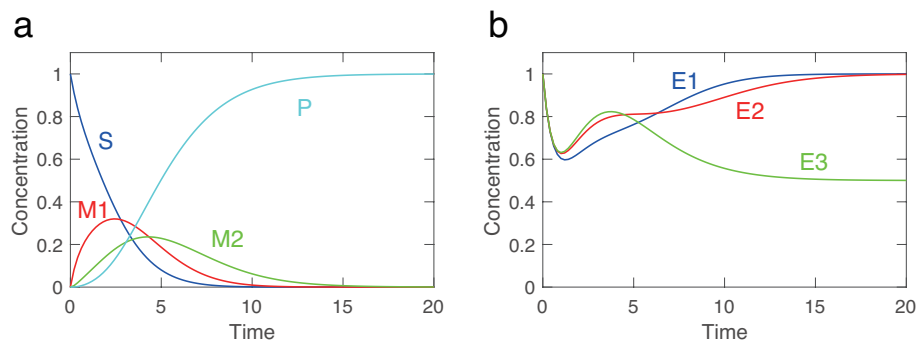
```

S -> M1 ; kcat_E1_S_M1 * E1 * S / ( Km_E1_S_M1 + S ) ; kcat_E1_S_M1 = 1 ; Km_E1_S_M1 = 1 ; E1 = 1
M1 -> M2 ; kcat_E2_M1_M2 * E2 * M1 / ( Km_E2_M1_M2 + M1 ) ; kcat_E2_M1_M2 = 1 ;
Km_E2_M1_M2 = 1 ; E2 = 1
M2 -> P ; kcat_E3_M2_P * E3 * M2 / ( Km_E3_M2_P + M2 ) ; kcat_E3_M2_P = 1 ; Km_E3_M2_P = 1 ; E3
= 1
-> G1 ; km_G1_M1 * K_G1_M1 ^ n_G1_M1 / ( K_G1_M1 ^ n_G1_M1 + M1 ^ n_G1_M1 ) ; km_G1_M1
= 1 ; K_G1_M1 = 1 ; n_G1_M1 = 1
-> G2 ; km_G2_M2 * K_G2_M2 ^ n_G2_M2 / ( K_G2_M2 ^ n_G2_M2 + M2 ^ n_G2_M2 ) ;
km_G2_M2 = 1 ; K_G2_M2 = 1 ; n_G2_M2 = 1
-> G3 ; km_G3_P * K_G3_P ^ n_G3_P / ( K_G3_P ^ n_G3_P + P ^ n_G3_P ) ; km_G3_P = 1 ; K_G3_P
= 1 ; n_G3_P = 1
-> E1 ; kp_G1_E1 * G1 ; kp_G1_E1 = 1
-> E2 ; kp_G2_E2 * G2 ; kp_G2_E2 = 1
-> E3 ; kp_G3_E3 * G3 ; kp_G3_E3 = 1
G1 -> ; kdeg_G1 * G1 ; kdeg_G1 = 1
G2 -> ; kdeg_G2 * G2 ; kdeg_G2 = 1
G3 -> ; kdeg_G3 * G3 ; kdeg_G3 = 1
E1 -> ; kdeg_E1 * E1 ; kdeg_E1 = 1
E2 -> ; kdeg_E2 * E2 ; kdeg_E2 = 1
E3 -> ; kdeg_E3 * E3 ; kdeg_E3 = 1
S = 1

```

---

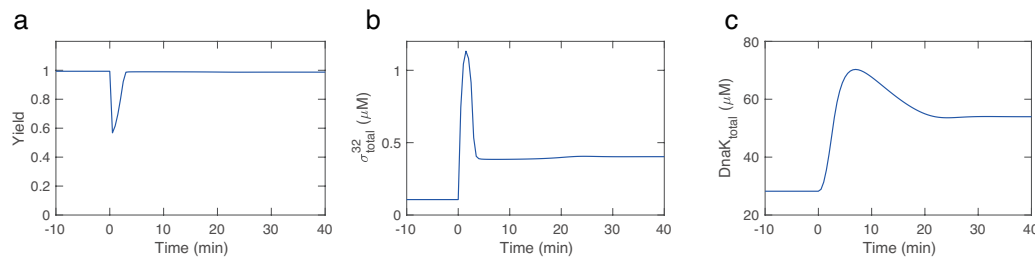
**Scheme 3.** The three-step model in Antimony language, created by KinModGPT.



**Figure 5.** Simulation of the created SBML model for the the three-step model.

Finally, KinModGPT has successfully created an SBML model for the heat shock response model that consists of 25 variables and 50 rate equations, demonstrating its potential for developing complex, realistic kinetic models. The Antimony model is provided in Schemes A4 and A5. To test the created SBML model, we assigned realistic parameter values and simulated its behavior. Upon heat shock, proteins are rapidly denatured, and thus the yield (the fraction of folded proteins in the total protein pool) decreases (Figure 6). However, the  $\sigma^{32}$  transcription factor is produced, which then initiates the expression of the chaperone protein DnaK. Denatured proteins are then quickly refolded by DnaK, and thus the yield is recovered. This behavior is consistent with literature [24–26].





**Figure 6.** Simulation of the created SBML model for the the heat shock response model. Heat shock occurs at 0 min and is implemented through an increase in the rate constant for protein denaturing. Yield is the fraction of folded proteins in a pool of total proteins, i.e.,  $Yield = P_{fold} / (P_{fold} + P_{unfold} + P_{unfold\_DnaK})$ . We tuned the kinetic parameters before simulation.

### 3. Discussion

In this article, we explored the possibilities of using GPT models for kinetic modeling automation. We developed KinModGPT by integrating GPT-3 [18] and Tellurium [4,5]. KinModGPT successfully converted a kinetic model written in a natural language text into the SBML model. Furthermore, all the created SBML models could be imported by widely-used modeling tools. To our knowledge, this work presents the first method applying GPT models to kinetic modeling, contributing to advances in systems biology.

How did GPT-3 fail to generate valid SBML files without any help from a modeling tool? Despite the effectiveness of language models, such as GPT-3, in generating natural sentences, their output may lack the precision required to generate accurate models. In contrast, even a single missing tag in SBML can lead to errors in the model. GPT-3 could not generate any valid SBML models, not even for the simplest models with one variable and one reaction. It is challenging to fix the generated invalid SBML models, as it requires manual review of the SBML files in XML format.

Instead of creating SBML models directly from natural language texts, KinModGPT employs Antimony language as an intermediate representation. Since Antimony language is simpler than SBML, GPT-3 can translate a natural language model description into Antimony language without errors. Then, the modeling tool, Tellurium, creates an SBML model from the Antimony model. For all four test problems, there were no conversion errors. However, even if there were some errors, they could be easily corrected by using modeling tools or by directly editing the Antimony models. This is another advantage of KinModGPT over the GPT-3 only approach.

Despite its promising results, KinModGPT has some limitations. Firstly, the current version of the natural language-Antimony conversion rule table covers only a part of biochemical reactions. However, the table can easily be expanded or tailored to a specific application domain. Moreover, we may be able to eliminate the need for manual rule definition by fine-tuning or by retraining GPT-3 with a large number of “Rosetta stones,” i.e., natural language model descriptions and their Antimony counterparts. Another limitation is that the current version of KinModGPT cannot automatically set appropriate kinetic parameter values. Thus, kinetic parameters must be tuned in the downstream modeling process to create a realistic model that complies with experimental data [27–30].

Gyori et al. developed the Integrated Network and Dynamical Reasoning Assembler (INDRA), which automatically builds kinetic models from natural language texts [12]. While INDRA focuses on modeling cell signaling pathways, its applicability to other types of biochemical systems is uncertain. In contrast, we have demonstrated that KinModGPT can effectively model various biochemical systems, including metabolic pathways, protein-protein interactions, and gene regulations. KinModGPT also offers the advantage of extensibility. Due to its simple implementation, modelers can easily customize the conversion rules for translating natural language texts to Antimony. For example, we modeled enzyme reactions using the simplest irreversible Michaelis-Menten equation in the three-step model; however, by modifying the conversion rule table (written in a simple text file), modelers can easily

switch to a reversible equation. In addition, as GPT-3 is multilingual, the conversion rules can be written in languages other than English.

It should be noted that the purpose of this article is not to provide a perfect solution for kinetic modeling automation but rather to demonstrate the potential of GPT models in this field. With the continued refinement of KinModGPT, AI may be able to extract information from many relevant articles to generate a kinetic model automatically. Such a development could significantly accelerate model development and improve an understanding of complex biological systems.

## 4. Materials and Methods

### 4.1. GPT-3

We chose GPT-3 (text-davinci-003) [18], one of OpenAI's GPT series with an open API. Trained on massive amounts of text data, GPT-3 is a powerful tool for natural language processing tasks, including language translation, text summarization, and question answering. By predicting the next word in a sequence given preceding words, GPT-3 can generate human-like natural language text. In GPT-3, the parameter called *temperature* determines the randomness of responses. In this study, we set *temperature* = 0 for reproducibility.

### 4.2. Tellurium

Tellurium [4,5] is a Python library for modeling and simulating biochemical systems. For model development, Tellurium employs Antimony [20], a human-readable, text-based language that facilitates the creation of kinetic models. Additionally, Tellurium can convert Antimony models to SBML models.

### 4.3. Test problems

We utilized four test problems to benchmark KinModGPT (Figure 1). The first problem, decay, is the simplest scenario in which an SBML model is generated from just two sentences. This model has a single variable and a single rate equation. The HIV model represents the mechanism of irreversible inhibition of HIV proteinase [21–23]. It is comprised of 9 variables and 10 rate equations, and its model description consists of 5 sentences. The three-step model describes a hypothetical metabolic pathway with 3 enzyme reactions and 3 gene regulations. This model consists of 10 variables and 15 rate equations, described in 5 sentences. Finally, the heat shock response model represents the realistic, complex regulatory mechanisms that confer robustness to heat shock in *Escherichia coli* [24–26]. This model consists of 25 variables and 50 rate equations, described in 20 sentences.

**Author Contributions:** Conceptualization, K.M.; software, K.M.; investigation, K.M. and H.K.; writing—original draft preparation, K.M. and H.K. All authors have read and agreed to the submitted version of the manuscript.

**Funding:** This work was supported by Grant-in-Aid for Scientific Research (C) (22K12247) and Grant-in-Aid for Scientific Research (B) (22H03688) from the Japan Society for the Promotion of Science. This work was further supported by JST PRESTO (JPMJPR20K8).

**Data Availability Statement:** The program code for the computational experiments and the created Antimony and SBML models are provided as Supplementary Materials.

**Conflicts of Interest:** The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

## Abbreviations

The following abbreviations are used in this manuscript:

GPT	Generative Pre-trained Transformer
SBML	Systems Biology Markup Language
COPASI	COmplex PATHway Simulator
CADLIVE	Computer-Aided Design of LIVing systEms
INDRA	Integrated Network and Dynamical Reasoning Assembler

## Appendix A

---

You are a program that converts biochemical reactions written in natural language into SBML (Systems Biology Markup Language).

Convert the biochemical reactions listed below into SBML. No need to provide further explanations, just present the converted result.

---

**Scheme A1.** Instruction message for the GPT-3 only approach. This instruction is followed by a model description (see Figure 1).

---

You are a program that converts biochemical reactions written in natural language into Antimony language. First, remember the following conversion rules.

### # Conversion rules

Natural language	Antimony language
E catalyzes the conversion of X to Y	$X \rightarrow Y$ ;  $k_{cat\_E\_X\_Y} * E * X / (K_{m\_E\_X\_Y} + X)$ ;  $k_{cat\_E\_X\_Y} = 1$ ;  $K_{m\_E\_X\_Y} = 1$ ;  $E = 1$
X is phosphorylated	$X \rightarrow X\_P$ ;  $V_p\_X * X / (K_{m\_X} + X)$ ;  $V_p\_X = 1$ ;  $K_{m\_X} = 1$
X is converted into Y	$X \rightarrow Y$ ;  $k_{c\_X\_Y} * X$ ;  $k_{c\_X\_Y} = 1$
X and Y bind to form Z	$X + Y \rightarrow Z$ ;  $k_{a\_X\_Y\_Z} * X * Y$ ;  $k_{a\_X\_Y\_Z} = 1$
X dissociates into Y and Z	$X \rightarrow Y + Z$ ;  $k_{d\_X\_Y\_Z} * X$ ;  $k_{d\_X\_Y\_Z} = 1$
X is produced (or transcribed)	$\rightarrow X$ ;  $k_{m\_X}$ ;  $k_{m\_X} = 1$
Expression of X is repressed (or negatively regulated or downregulated) by R	$\rightarrow X$ ;  $k_{m\_X\_R} * K_{X\_R}^n / (K_{X\_R}^n + R^n)$ ;  $k_{m\_X\_R} = 1$ ;  $K_{X\_R} = 1$ ;  $n_{X\_R} = 1$
Expression of X is activated (or positively regulated or upregulated) by A	$\rightarrow X$ ;  $k_{m\_X\_A} * A^n / (K_{X\_A}^n + A^n)$ ;  $k_{m\_X\_A} = 1$ ;  $K_{X\_A} = 1$ ;  $n_{X\_A} = 1$
Y is translated from X	$\rightarrow Y$ ;  $k_{p\_X\_Y} * X$ ;  $k_{p\_X\_Y} = 1$
X degrades (or decays)	$X \rightarrow$ ;  $k_{deg\_X} * X$ ;  $k_{deg\_X} = 1$
X (concentration) is Y M (or mM or uM or nM or pM)	$X = Y$

### # Examples

"The expression of G is negatively regulated by R." is converted into  $\rightarrow G$ ;  $k_{m\_G\_R} * K_{G\_R}^n / (K_{G\_R}^n + R^n)$ ;  $k_{m\_G\_R} = 1$ ;  $K_{G\_R} = 1$ ;  $n_{G\_R} = 1$   
 "G is upregulated by A." is converted into  $\rightarrow G$ ;  $k_{m\_G\_A} * A^n / (K_{G\_A}^n + A^n)$ ;  $k_{m\_G\_A} = 1$ ;  $K_{G\_A} = 1$ ;  $n_{G\_A} = 1$

Using the conversion rules provided, convert the biochemical reactions listed below into Antimony language. After converting each reaction, create a bullet point list that includes all the resulting expressions. In the list, show one reaction per line. No need to provide further explanations, just present the list. Start each line with '- '.

---

**Scheme A2.** Instruction message for KinModGPT. This instruction is followed by a model description (see Figure 1).

**Scheme A3.** Full model description for the heat shock response model.

s70 and RNA polymerase (RNAP) bind together to form s70\_RNAP, which then dissociates into s70 and RNAP. Pg and s70\_RNAP bind to create Pg\_s70\_RNAP, which can dissociate back into its components. RNAP and s32 bind to create RNAP\_s32, which can dissociate into RNAP and s32. Ph and RNAP\_s32 bind to form Ph\_RNAP\_s32, which then dissociates back into Ph and RNAP\_s32. Additionally, s32 and DnaK form s32\_DnaK, which can dissociate into s32 and DnaK, while s32 and FtsH bind to create s32\_FtsH, which can dissociate back into s32 and FtsH. Similarly, Punfold and DnaK bind to form Punfold\_DnaK, which can dissociate into Punfold and DnaK. D and s70\_RNAP bind to form D\_s70\_RNAP, which can dissociate into D and s70\_RNAP, and D and RNAP\_s32 bind to form D\_RNAP\_s32, which can dissociate into D and RNAP\_s32. RNAP and D bind to form RNAP\_D, and RNAP\_D can dissociate into RNAP and D. s32\_DnaK and FtsH bind to form s32\_DnaK\_FtsH, which then dissociates into s32\_DnaK and FtsH. s32\_FtsH is converted into FtsH, while s32\_DnaK\_FtsH is converted into DnaK and FtsH. Similarly, Pfold is converted into Punfold, while Punfold\_DnaK is converted into Pfold and DnaK.

mRNA\_s32 is upregulated by Pg\_s70\_RNAP. Similarly, mRNA\_DnaK and mRNA\_FtsH are positively regulated by Ph\_RNAP\_s32. mRNA\_Protein is transcribed without regulation. s32, FtsH, DnaK, and Pfold are translated from mRNA\_s32, mRNA\_FtsH, mRNA\_DnaK, and mRNA\_Protein, respectively. All the mRNAs (mRNA\_s32, mRNA\_DnaK, mRNA\_FtsH, and mRNA\_Protein) decay. s32, s32\_DnaK, s32\_FtsH, s32\_DnaK\_FtsH, FtsH, DnaK, Punfold\_DnaK, Pfold, and Punfold decay. RNAP\_s32 is degraded into RNAP. Similarly, Ph\_RNAP\_s32 is degraded into Ph and RNAP. D\_RNAP\_s32 is degraded into RNAP\_D.

**Scheme A4.** The first half of the heat shock response model in Antimony language, created by KinModGPT.

```
s70 + RNAP -> s70_RNAP ; ka_s70_RNAP * s70 * RNAP ; ka_s70_RNAP = 1
s70_RNAP -> s70 + RNAP ; kd_s70_RNAP * s70_RNAP ; kd_s70_RNAP = 1
Pg + s70_RNAP -> Pg_s70_RNAP ; ka_Pg_s70_RNAP * Pg * s70_RNAP ; ka_Pg_s70_RNAP = 1
Pg_s70_RNAP -> Pg + s70_RNAP ; kd_Pg_s70_RNAP * Pg_s70_RNAP ; kd_Pg_s70_RNAP = 1
RNAP + s32 -> RNAP_s32 ; ka_RNAP_s32 * RNAP * s32 ; ka_RNAP_s32 = 1
RNAP_s32 -> RNAP + s32 ; kd_RNAP_s32 * RNAP_s32 ; kd_RNAP_s32 = 1
Ph + RNAP_s32 -> Ph_RNAP_s32 ; ka_Ph_RNAP_s32 * Ph * RNAP_s32 ; ka_Ph_RNAP_s32 = 1
Ph_RNAP_s32 -> Ph + RNAP_s32 ; kd_Ph_RNAP_s32 * Ph_RNAP_s32 ; kd_Ph_RNAP_s32 = 1
s32 + DnaK -> s32_DnaK ; ka_s32_DnaK * s32 * DnaK ; ka_s32_DnaK = 1
s32_DnaK -> s32 + DnaK ; kd_s32_DnaK * s32_DnaK ; kd_s32_DnaK = 1
s32 + FtsH -> s32_FtsH ; ka_s32_FtsH * s32 * FtsH ; ka_s32_FtsH = 1
s32_FtsH -> s32 + FtsH ; kd_s32_FtsH * s32_FtsH ; kd_s32_FtsH = 1
Punfold + DnaK -> Punfold_DnaK ; ka_Punfold_DnaK * Punfold * DnaK ; ka_Punfold_DnaK = 1
Punfold_DnaK -> Punfold + DnaK ; kd_Punfold_DnaK * Punfold_DnaK ; kd_Punfold_DnaK = 1
D + s70_RNAP -> D_s70_RNAP ; ka_D_s70_RNAP * D * s70_RNAP ; ka_D_s70_RNAP = 1
D_s70_RNAP -> D + s70_RNAP ; kd_D_s70_RNAP * D_s70_RNAP ; kd_D_s70_RNAP = 1
D + RNAP_s32 -> D_RNAP_s32 ; ka_D_RNAP_s32 * D * RNAP_s32 ; ka_D_RNAP_s32 = 1
D_RNAP_s32 -> D + RNAP_s32 ; kd_D_RNAP_s32 * D_RNAP_s32 ; kd_D_RNAP_s32 = 1
RNAP + D -> RNAP_D ; ka_RNAP_D * RNAP * D ; ka_RNAP_D = 1
RNAP_D -> RNAP + D ; kd_RNAP_D * RNAP_D ; kd_RNAP_D = 1
s32_DnaK + FtsH -> s32_DnaK_FtsH ; ka_s32_DnaK_FtsH * s32_DnaK * FtsH ; ka_s32_DnaK_FtsH = 1
s32_DnaK_FtsH -> s32_DnaK + FtsH ; kd_s32_DnaK_FtsH * s32_DnaK_FtsH ; kd_s32_DnaK_FtsH = 1
s32_FtsH -> FtsH ; kc_s32_FtsH * s32_FtsH ; kc_s32_FtsH = 1
s32_DnaK_FtsH -> DnaK + FtsH ; kc_s32_DnaK_FtsH * s32_DnaK_FtsH ; kc_s32_DnaK_FtsH = 1
Pfold -> Punfold ; kc_Pfold_Punfold * Pfold ; kc_Pfold_Punfold = 1
Punfold_DnaK -> Pfold + DnaK ; kc_Punfold_DnaK * Punfold_DnaK ; kc_Punfold_DnaK = 1
```

**Scheme A5.** The second half of the heat shock response model in Antimony language, created by KinModGPT.

---

```

-> mRNA_s32 ; km_mRNA_s32_Pg_s70_RNAP * Pg_s70_RNAP ^ n_mRNA_s32_Pg_s70_RNAP / (
K_mRNA_s32_Pg_s70_RNAP ^ n_mRNA_s32_Pg_s70_RNAP + Pg_s70_RNAP ^
n_mRNA_s32_Pg_s70_RNAP ) ; km_mRNA_s32_Pg_s70_RNAP = 1 ; K_mRNA_s32_Pg_s70_RNAP =
1 ; n_mRNA_s32_Pg_s70_RNAP = 1
-> mRNA_DnaK ; km_mRNA_DnaK_Ph_RNAP_s32 * Ph_RNAP_s32 ^
n_mRNA_DnaK_Ph_RNAP_s32 / ( K_mRNA_DnaK_Ph_RNAP_s32 ^
n_mRNA_DnaK_Ph_RNAP_s32 + Ph_RNAP_s32 ^ n_mRNA_DnaK_Ph_RNAP_s32 ) ;
km_mRNA_DnaK_Ph_RNAP_s32 = 1 ; K_mRNA_DnaK_Ph_RNAP_s32 = 1 ;
n_mRNA_DnaK_Ph_RNAP_s32 = 1
-> mRNA_FtsH ; km_mRNA_FtsH_Ph_RNAP_s32 * Ph_RNAP_s32 ^ n_mRNA_FtsH_Ph_RNAP_s32
/ ( K_mRNA_FtsH_Ph_RNAP_s32 ^ n_mRNA_FtsH_Ph_RNAP_s32 + Ph_RNAP_s32 ^
n_mRNA_FtsH_Ph_RNAP_s32 ) ; km_mRNA_FtsH_Ph_RNAP_s32 = 1 ;
K_mRNA_FtsH_Ph_RNAP_s32 = 1 ; n_mRNA_FtsH_Ph_RNAP_s32 = 1
-> mRNA_Protein ; km_mRNA_Protein ; km_mRNA_Protein = 1
-> s32 ; kp_mRNA_s32_s32 * mRNA_s32 ; kp_mRNA_s32_s32 = 1
-> FtsH ; kp_mRNA_FtsH_FtsH * mRNA_FtsH ; kp_mRNA_FtsH_FtsH = 1
-> DnaK ; kp_mRNA_DnaK_DnaK * mRNA_DnaK ; kp_mRNA_DnaK_DnaK = 1
-> Pfold ; kp_mRNA_Protein_Pfold * mRNA_Protein ; kp_mRNA_Protein_Pfold = 1
mRNA_s32 -> ; kdeg_mRNA_s32 * mRNA_s32 ; kdeg_mRNA_s32 = 1
mRNA_DnaK -> ; kdeg_mRNA_DnaK * mRNA_DnaK ; kdeg_mRNA_DnaK = 1
mRNA_FtsH -> ; kdeg_mRNA_FtsH * mRNA_FtsH ; kdeg_mRNA_FtsH = 1
mRNA_Protein -> ; kdeg_mRNA_Protein * mRNA_Protein ; kdeg_mRNA_Protein = 1
s32 -> ; kdeg_s32 * s32 ; kdeg_s32 = 1
s32_DnaK -> ; kdeg_s32_DnaK * s32_DnaK ; kdeg_s32_DnaK = 1
s32_FtsH -> ; kdeg_s32_FtsH * s32_FtsH ; kdeg_s32_FtsH = 1
s32_DnaK_FtsH -> ; kdeg_s32_DnaK_FtsH * s32_DnaK_FtsH ; kdeg_s32_DnaK_FtsH = 1
FtsH -> ; kdeg_FtsH * FtsH ; kdeg_FtsH = 1
DnaK -> ; kdeg_DnaK * DnaK ; kdeg_DnaK = 1
Punfold_DnaK -> ; kdeg_Punfold_DnaK * Punfold_DnaK ; kdeg_Punfold_DnaK = 1
Pfold -> ; kdeg_Pfold * Pfold ; kdeg_Pfold = 1
Punfold -> ; kdeg_Punfold * Punfold ; kdeg_Punfold = 1
RNAP_s32 -> RNAP ; kdeg_RNAP_s32 * RNAP_s32 ; kdeg_RNAP_s32 = 1
Ph_RNAP_s32 -> Ph + RNAP ; kdeg_Ph_RNAP_s32 * Ph_RNAP_s32 ; kdeg_Ph_RNAP_s32 = 1
D_RNAP_s32 -> RNAP_D ; kdeg_D_RNAP_s32 * D_RNAP_s32 ; kdeg_D_RNAP_s32 = 1

```

---

## References

1. Kitano, H., Systems biology: a brief overview. *Science* **2002**, 295, (5560), 1662-4.
2. Hucka, M.; Finney, A.; Sauro, H. M.; Bolouri, H.; Doyle, J. C.; Kitano, H.; Arkin, A. P.; Bornstein, B. J.; Bray, D.; Cornish-Bowden, A.; Cuellar, A. A.; Dronov, S.; Gilles, E. D.; Ginkel, M.; Gor, V.; Goryanin, II; Hedley, W. J.; Hodgman, T. C.; Hofmeyr, J. H.; Hunter, P. J.; Juty, N. S.; Kasberger, J. L.; Kremling, A.; Kummer, U.; Le Novère, N.; Loew, L. M.; Lucio, D.; Mendes, P.; Minch, E.; Mjolsness, E. D.; Nakayama, Y.; Nelson, M. R.; Nielsen, P. F.; Sakurada, T.; Schaff, J. C.; Shapiro, B. E.; Shimizu, T. S.; Spence, H. D.; Stelling, J.; Takahashi, K.; Tomita, M.; Wagner, J.; Wang, J.; Forum, S., The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. *Bioinformatics* **2003**, 19, (4), 524-31.
3. Keating, S. M.; Waltemath, D.; König, M.; Zhang, F.; Dräger, A.; Chaouiya, C.; Bergmann, F. T.; Finney, A.; Gillespie, C. S.; Helikar, T.; Hoops, S.; Malik-Sheriff, R. S.; Moodie, S. L.; Moraru, II; Myers, C. J.; Naldi, A.; Olivier, B. G.; Sahle, S.; Schaff, J. C.; Smith, L. P.; Swat, M. J.; Thieffry, D.; Watanabe, L.; Wilkinson, D. J.; Blinov, M. L.; Begley, K.; Faeder, J. R.; Gomez, H. F.; Hamm, T. M.; Inagaki, Y.; Liebermeister, W.; Lister, A. L.; Lucio, D.; Mjolsness, E.; Proctor, C. J.; Raman, K.; Rodriguez, N.; Shaffer, C. A.; Shapiro, B. E.; Stelling, J.; Swainston, N.; Tanimura, N.; Wagner, J.; Meier-Schellersheim, M.; Sauro, H. M.; Palsson, B.; Bolouri, H.;



- Kitano, H.; Funahashi, A.; Hermjakob, H.; Doyle, J. C.; Hucka, M.; members, S. L. C., SBML Level 3: an extensible format for the exchange and reuse of biological models. *Mol Syst Biol* **2020**, *16*, (8), e9110.
4. Choi, K.; Medley, J. K.; Konig, M.; Stocking, K.; Smith, L.; Gu, S.; Sauro, H. M., Tellurium: An extensible python-based modeling environment for systems and synthetic biology. *Biosystems* **2018**, *171*, 74-79.
5. Medley, J. K.; Choi, K.; Konig, M.; Smith, L.; Gu, S.; Hellerstein, J.; Sealfon, S. C.; Sauro, H. M., Tellurium notebooks-An environment for reproducible dynamical modeling in systems biology. *PLoS Comput Biol* **2018**, *14*, (6), e1006220.
6. Hoops, S.; Sahle, S.; Gauges, R.; Lee, C.; Pahle, J.; Simus, N.; Singhal, M.; Xu, L.; Mendes, P.; Kummer, U., COPASI—a COMplex PATHway SIMulator. *Bioinformatics* **2006**, *22*, (24), 3067-74.
7. Mendes, P.; Hoops, S.; Sahle, S.; Gauges, R.; Dada, J.; Kummer, U., Computational modeling of biochemical networks using COPASI. *Methods Mol Biol* **2009**, *500*, 17-59.
8. Bergmann, F. T.; Hoops, S.; Klahn, B.; Kummer, U.; Mendes, P.; Pahle, J.; Sahle, S., COPASI and its applications in biotechnology. *J Biotechnol* **2017**, *261*, 215-220.
9. Kurata, H.; Matoba, N.; Shimizu, N., CADLIVE for constructing a large-scale biochemical network based on a simulation-directed notation and its application to yeast cell cycle. *Nucleic Acids Res* **2003**, *31*, (14), 4071-84.
10. Kurata, H.; Masaki, K.; Sumida, Y.; Iwasaki, R., CADLIVE dynamic simulator: direct link of biochemical networks to dynamic models. *Genome Res* **2005**, *15*, (4), 590-600.
11. Kurata, H.; Inoue, K.; Maeda, K.; Masaki, K.; Shimokawa, Y.; Zhao, Q., Extended CADLIVE: a novel graphical notation for design of biochemical network maps and computational pathway analysis. *Nucleic Acids Res* **2007**, *35*, (20), e134.
12. Gyori, B. M.; Bachman, J. A.; Subramanian, K.; Muhlich, J. L.; Galescu, L.; Sorger, P. K., From word models to executable models of signaling networks using automated assembly. *Mol Syst Biol* **2017**, *13*, (11), 954.
13. Todorov, P. V.; Gyori, B. M.; Bachman, J. A.; Sorger, P. K., INDRA-IPM: interactive pathway modeling using natural language with automated assembly. *Bioinformatics* **2019**, *35*, (21), 4501-4503.
14. Roose, K., The Brilliance and Weirdness of ChatGPT. *New York Times* **Dec 26, 2022**.
15. Terwiesch, C. Would Chat GPT Get a Wharton MBA? A Prediction Based on Its Performance in the Operations Management Course; Mack Institute for Innovation Management at the Wharton School, University of Pennsylvania: **2023**.
16. Choi, J. H.; Hickman, K. E.; Monahan, A.; Schwarcz, D., ChatGPT Goes to Law School. *SSRN* **2023**.
17. Bussler, F., Will GPT-3 Kill Coding? **Jul 21, 2020**. Available online: <https://towardsdatascience.com/will-gpt-3-kill-coding-630e4518c04d> (accessed on Mar 4, 2023).
18. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A., Language models are few-shot learners. *Advances in neural information processing systems* **2020**, *33*, 1877-1901.
19. Bergmann, F. T.; Hucka, M.; Bornstein, B. J.; Jouraku, A. Online SBML Validator. Available online: [https://synonym.caltech.edu/validator\\_servlet/](https://synonym.caltech.edu/validator_servlet/) (accessed on Mar 4, 2023)
20. Smith, L. P.; Bergmann, F. T.; Chandran, D.; Sauro, H. M., Antimony: a modular model definition language. *Bioinformatics* **2009**, *25*, (18), 2452-4.
21. Ji, X.; Xu, Y., libSRES: a C library for stochastic ranking evolution strategy for parameter estimation. *Bioinformatics* **2006**, *22*, (1), 124-6.
22. Kuzmic, P., Program DYNAFIT for the analysis of enzyme kinetic data: application to HIV proteinase. *Analytical biochemistry* **1996**, *237*, (2), 260-73.
23. Mendes, P.; Kell, D., Non-linear optimization of biochemical pathways: applications to metabolic engineering and parameter estimation. *Bioinformatics* **1998**, *14*, (10), 869-83.
24. Kurata, H.; El-Samad, H.; Yi, T.-M.; Khammash, M.; Doyle, J. In Feedback Regulation of the Heat Shock Response in E. coli, Conference on Decision and Control, Orlando, Florida, USA, 2001; Orlando, Florida, USA, 2001; pp 837-42.
25. El-Samad, H.; Kurata, H.; Doyle, J. C.; Gross, C. A.; Khammash, M., Surviving heat shock: control strategies for robustness and performance. *Proc Natl Acad Sci U S A* **2005**, *102*, (8), 2736-41.
26. Kurata, H.; El-Samad, H.; Iwasaki, R.; Ohtake, H.; Doyle, J. C.; Grigorova, I.; Gross, C. A.; Khammash, M., Module-based analysis of robustness tradeoffs in the heat shock response system. *PLoS Comput Biol* **2006**, *2*, (7), e59.
27. Jaqaman, K.; Danuser, G., Linking data to models: data regression. *Nat Rev Mol Cell Biol* **2006**, *7*, (11), 813-9.

28. Banga, J. R., Optimization in computational systems biology. *BMC Syst Biol* **2008**, *2*, (1), 47.
29. Ashyraliyev, M.; Fomekong-Nanfack, Y.; Kaandorp, J. A.; Blom, J. G., Systems biology: parameter estimation for biochemical models. *FEBS J* **2009**, *276*, (4), 886-902.
30. Maeda, K.; Boogerd, F. C.; Kurata, H., RCGAToolbox: A Real-coded Genetic Algorithm Software for Parameter Estimation of Kinetic Models. *IPSI Transactions on Bioinformatics* **2021**, *14*, (0), 30-35.

**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.