Enhancing Theoretical Understanding of the Onset of Type 1 Diabetes

David Li
Basha High School
5990 South Val Vista Drive, Chandler, Arizona 85249

Abstract

Type 1 diabetes, or autoimmune diabetes, is a disease where the body's immune system destroys insulin-producing pancreatic beta cells. Previous experiments in NOD (non-obese diabetic mice) revealed that the level of immune cells in the weeks before symptoms appeared followed a characteristic oscillatory or "spiking" pattern. This experiment enhances the mathematical modeling of this phenomenon by analyzing one parameter, the level of beta cells, as it varies continuously (rather than leaving it constant), which, based on research in systems in other fields, can reveal unexpected behavior. The experiment found the oscillations indicative of diabetes would either begin earlier or later than expected mathematically, implying that the disease may be detectable earlier, potentially allowing more time for treatment for those affected.

1 Introduction

In type 1 diabetes, also known as autoimmune diabetes, the body's own immune system attacks and destroys insulin-producing pancreatic beta cells, leading to an insulin shortage and causing symptoms [1]. Currently, the causes and cures are largely unknown [2]. However, previous experiments have shown that in NOD (non-obese diabetic) mice, a standard model for diabetic research, the level of T cells (a specific type of immune cell) fluctuates cyclically in the weeks leading up to the appearance of symptoms, as depicted in figure 1, top: after the "spikes" occur (black), the percentage of diabetic mice increases dramatically (in gray) [[3] cited in [4]]. To better understand the mechanism underlying these oscillations, Mahaffy and Edelstein-Keshet constructed a mathematical model of the immune–pancreas system. One parameter in the model is the level of pancreatic beta cells, which slowly decreases over time; at a certain level, the fluctuations described experimentally occur, and then diabetes appears [4].

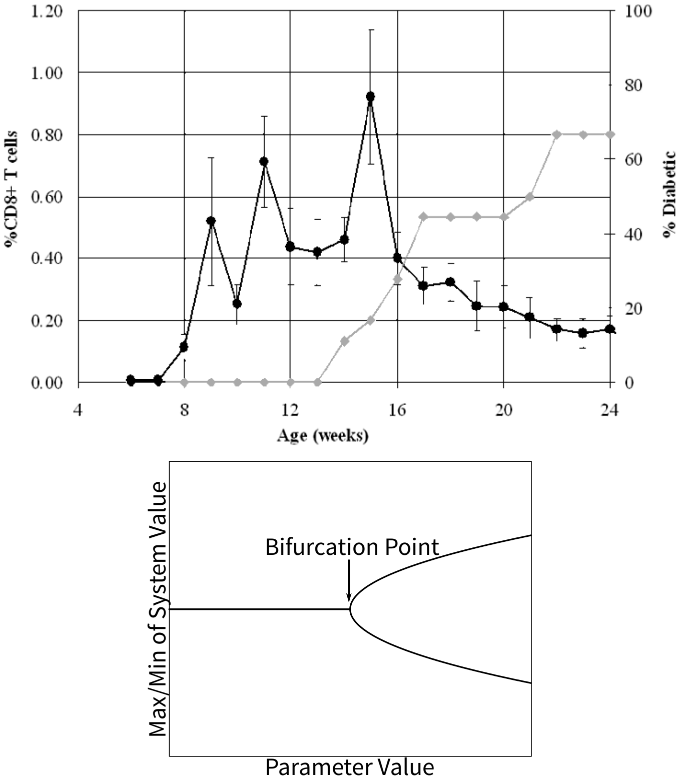

Figure 1 Top: From [4]; experimental measurement of T cell levels in NOD mice over time, with diabetes occurring after "spikes". Below: Example of a Hopf bifurcation.

Originally, the researchers analyzed the model's behavior at various constant parameter values [4], searching for *bifurcations*, or qualitative changes in behavior that occur when a parameter reaches a certain threshold [5]. For instance, a system may be constant at one parameter value; if the parameter reaches a certain level, the system may then oscillate between two values (see figure 1) [5]. Herein, this experiment will improve the diabetes model by applying research demonstrating that in certain systems, slowly varying the parameter value while the model runs changes the qualitative nature of the system [6]. This more accurately reflects what occurs biologically, as the original paper explicitly states the parameter continuously slowly falls. To summarize: for the original *static* analysis, the authors re-ran the model multiple times, each time setting the parameter to a fixed value. For the *continuous* analysis here, the parameter starts at a given value and then continuously decreases over time. Thus, this experiment will help scientists and medical professionals better understand and apply the theoretical results of Mahaffy and Edelstein-Keshet's work as well as the experimental results they cited in predicting and understanding the onset of type 1 diabetes and the behavior of the immune system in this disease.

Furthermore, when varying the parameter slowly, previous research noted that numerical precision greatly affected the results; if the calculations truncated too many digits (e.g. by using "single-precision" numbers), the results simply came out wrong [6]. This experiment will address that with `mpmath`, a modern library for arbitrary-precision calculations that allows calculations at any precision [7].

2 Question & Hypothesis

Question: How does treating the peptide clearance rate δ_p (= 1 / pancreatic beta cell level) as a *continuously* and slowly varying parameter affect the qualitative behavior of the scaled reduced immune model developed by Mahaffy and Edelstein-Keshet, and how can these findings be applied to understanding and predicting the onset of type 1 diabetes?

Hypothesis: If the original model for the level of immune cells in the weeks before the onset of type 1 diabetes is analyzed with both a continuously varying and a static peptide clearance rate δ_p, then in the former analysis, the oscillations present in the original model that indicate the onset of diabetes will begin later than in the new, latter model because previous research has shown such behavior is delayed in other systems when similarly analyzed with a continuously varying parameter.

3 Experimental Materials

- Computer (x86-64 architecture running a Linux distribution)
  The exact specifications and the operating system do not matter, as the software used runs on all platforms. A relatively recent (past 5 years or so) computer will suffice.
- Computer Software
  - Python 3.3.3 with `mpmath` (http://www.python.org, http://www.mpmath.org), used to calculate the data for the plots.
  - XPP AUTO 7.0 (http://www.math.pitt.edu/~bard/xpp/xpp.html), used to calculate the data for the bifurcation diagrams.
  - ConTeXt 2014.01.03 (http://wiki.contextgarden.net), used to generate the plots.
  - Gnuplot 4.6.4 (http://www.gnuplot.info), used to generate plots.
- Project Source Code: https://github.com/lidavidm/scifair-type1diabetes-model.

Here are the version numbers of the software programs as reported themselves:

```
% python --version
Python 3.3.3
% ./xppaut -version
XPPAUT Version 7.0
% python -c "import mpmath; print(mpmath.__version__)"
0.17
% gnuplot --version
gnuplot 4.6 patchlevel 4
% context --version
mtx-context     | current version: 2014.01.03 00:40
```

4 Experimental Procedures

These procedures represent a high-level overview; see "Detailed Procedures" on page 11 for details.

XPP AUTO, used in the original paper, is used to compute a bifurcation diagram (figure 2), which shows, at a particular level of beta cells, possible states of the model, or the *maximum and minimum values of A over time for a trial run of the model with the particular parameter level indicated on the x-axis*. Thus, it shows at what *static* parameter levels diabetes can occur; the higher the T cell level, the more beta cells can be destroyed. On a steady state, the system remains at a fixed level of T cells; in contrast, the level oscillates on a periodic state. If a state is steady, then introducing a small deviation will result in the system returning to the original state, while an unstable state will move towards another state. The points labeled

"HB" are *Hopf bifurcations*, points at which the system changes qualitative behavior—from stable to periodic or vice-versa.

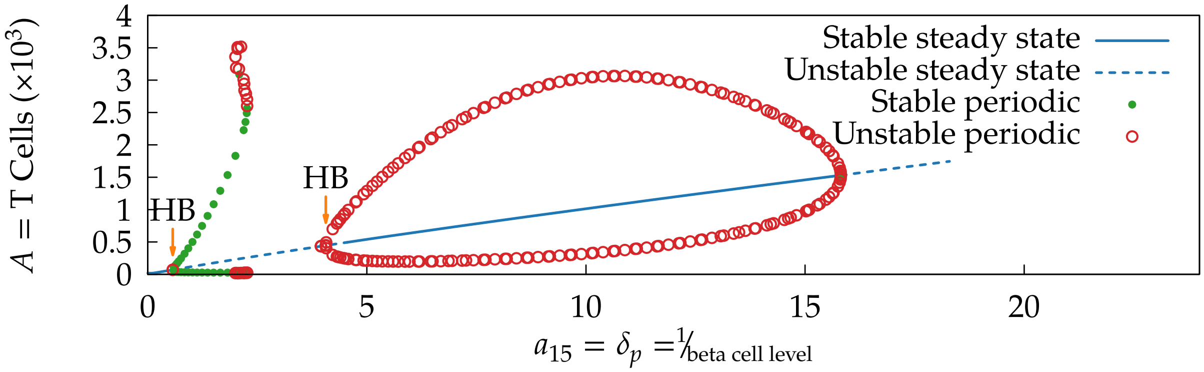

Figure 2 The bifurcation diagram, generated by XPP AUTO and plotted with Gnuplot.

Now, for the *continuous* analysis, the model starts at a particular a_{15} (= 1/beta cell) level, which varies linearly over time. Because this variation is linear, at each time the a_{15} level is unique, and thus the A vs t plot of this model can be overlaid on the bifurcation diagram as a A vs a_{15} plot, allowing them to be compared (see the diagrams below).

The model equations are

$$\frac{dA}{dt} = (a_6 + a_7 M) f_1(p) - a_8 A - a_9 A^2 \qquad \frac{dM}{dt} = a_{10} f_2(p) A - f_1(p) a_7 a_{16} M - a_{11} M$$

$$\frac{dE}{dt} = a_{12}(1 - f_2(p))A - a_{13} E \qquad p = \frac{a_{14}}{a_{15}} EB \qquad f_1 = \frac{p^{a_1}}{{a_2}^{a_1} + p^{a_1}} \qquad f_2 = \frac{a_4 {a_5}^{a_3}}{{a_5}^{a_3} + p^{a_3}}$$

with $a_{15} = \delta_p =$ constant for the bifurcation diagram or $\frac{da_{15}}{dt} = \frac{d\delta_p}{dt} =$ rate of change for the continuous analysis[1]. Here, A represents the level of activated T cells, B the level of pancreatic beta cells, E the effector T cells, M the memory T cells, and p the peptide level. (The fluctuation of activated T cells matters the most for the onset of diabetes.) In the model, pancreatic beta cells that undergo apoptosis (cell death) generate peptide; in lymph nodes, T cells become activated and "recognize" this particular peptide. These cells then become either effector cells, which destroy more pancreatic cells, thus producing more peptide, or memory cells, which encounter the peptide and trigger an immune response.

Because B and δ_p are constant, an increase in the latter is equivalent to a decrease in the former (and vice-versa), so the latter was increased in Mahaffy's analysis, which is equivalent to a decrease in beta cell levels [4]. Additionally, a 4th order Runge-Kutta ODE solver written in Python is also used to compute the numerical solution in the second case. Runge-Kutta is a standard algorithm for solving ODEs; it is used here because Dr. Baer's

[1] The rate of change depends on the parameter range in consideration (see the appendix).

work noted that the continuous analysis could lead to numerical precision errors, and the Python implementation allows for arbitrary-precision computations [6][7][8].

This experiment will also generate model-over-time diagrams, which are overlaid over the bifurcation diagram to generate a "combined diagram". Since the parameter will continuously and linearly vary over time, a correspondence between the points of the two graphs exists (i.e. for the model-vs-time graph, at a certain time, the parameter will be at a unique particular value in the X-range of the bifurcation diagram), and therefore the two modes of simulation can be compared – the goal of this experiment. (See the appendix for the implementation details of processing the data and generating the diagrams.)

5 Results & Discussion

Recall the onset of diabetes is roughly predicted by the appearance of oscillations. "HB" indicates the location of a Hopf bifurcation.

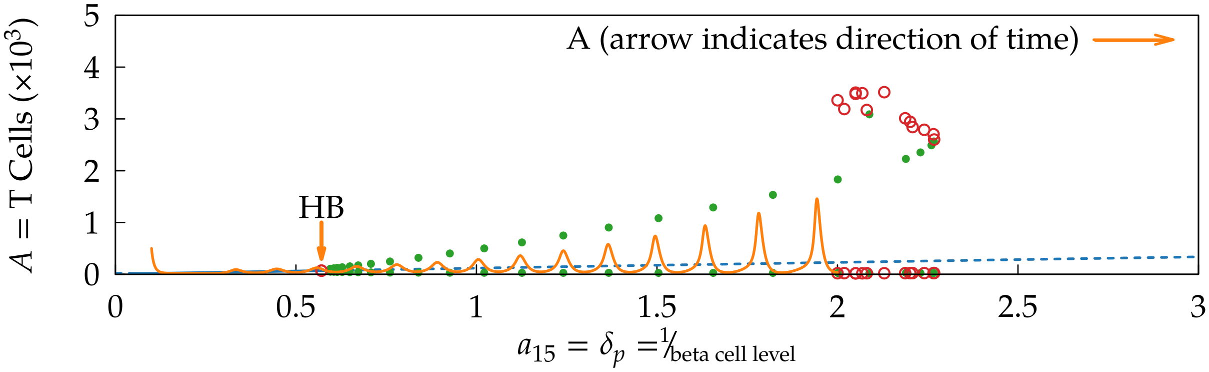

Figure 3 Parameter range 0.1 to 2 over 200 days

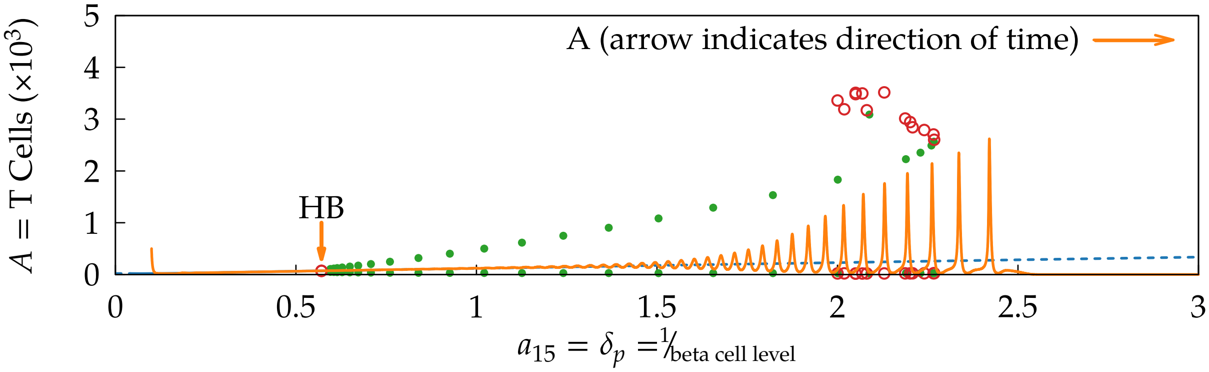

Figure 4 Parameter range 0.1 to 2 over 1000 days

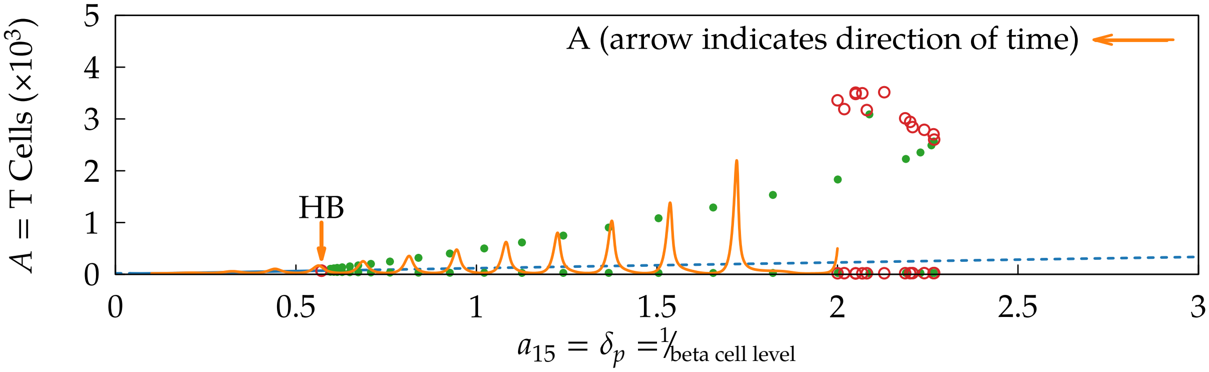

Figure 5 Parameter range 2 to 0.1 over 200 days

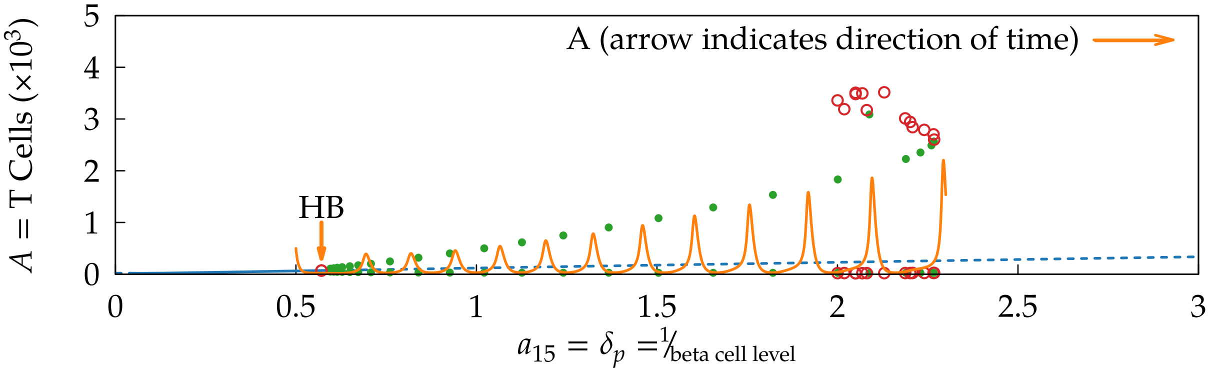

Figure 6 Parameter range 0.5 to 2.3 over 200 days

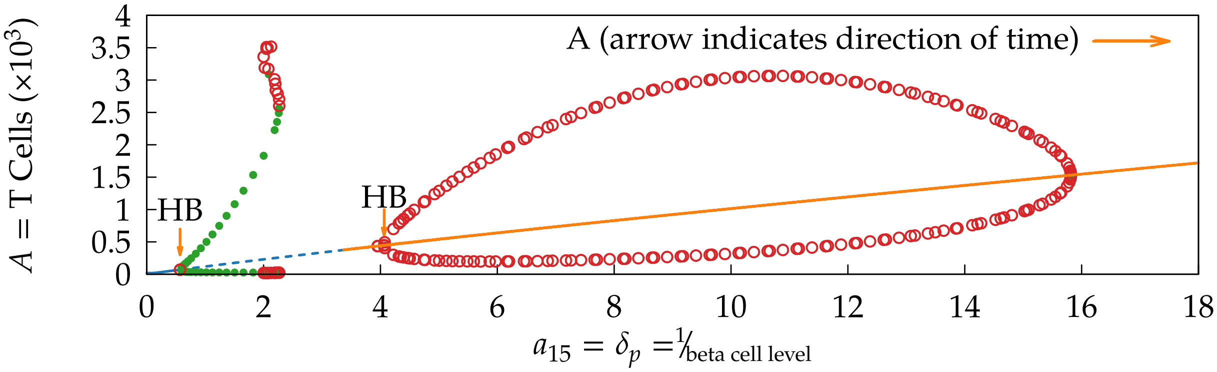

Figure 7 Demonstration of ramping of system through the second Hopf bifurcation (increasing parameter)

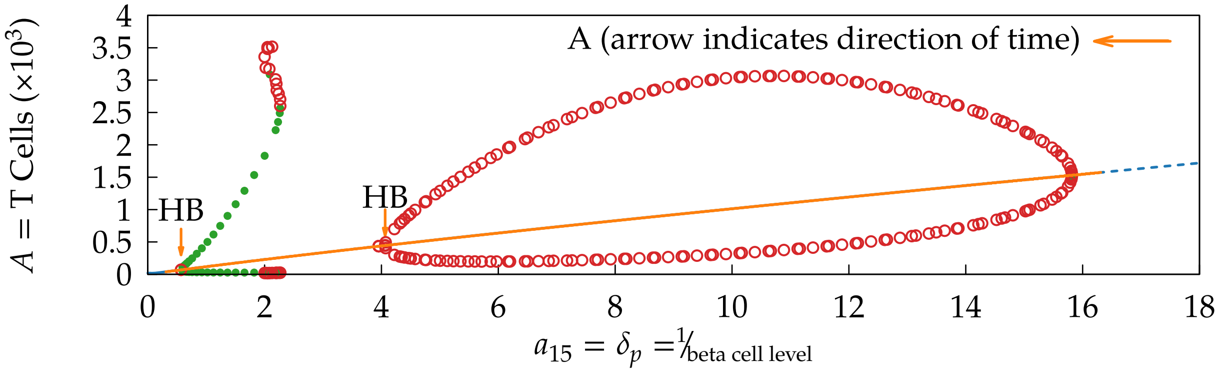

Figure 8 Demonstration of ramping of system through the second Hopf bifurcation (decreasing parameter)

To read the plots, note that each contains two graphs: the continuous model, in orange, and the bifurcation diagram, in blue/green/red. The latter shows, at a particular δ_p, the level of T cells the static model "settles into" over time. If multiple values are shown for a particular δ_p, then the system oscillates between those values over time (and in this model, oscillations imply symptoms of diabetes [4]). Meanwhile, the continuous model (orange), which is the focus of this experiment, shows T cell level vs. time; since δ_p varies linearly with time, each δ_p corresponds to exactly one unique time value. The arrow in the upper right indicates the direction of time.

In the original experiments with mice, the appearance of symptoms of diabetes corresponded with cyclic fluctuations in the level of T cells [3]. Looking at figure 3, as the level of T cells decreases over time in the continuous model, some oscillations appear before the point they are expected to (the *Hopf bifurcation* point, labeled "HB"). Given the large error bars in the original experiments, however, such small oscillations may not be of note and may explain why only a few oscillations were found in the mice. Indeed, if the system is started closer to the Hopf bifurcation in figure 6, the oscillations begin at the expected time. When the same amount of beta cell destruction as in figure 3 is spread out over a longer timeframe—1000 days rather than 200 (the experiments covered roughly 200 days)—the oscillations occur far later than they should, as seen in figure 4. Thus, theoretically, a way to prevent beta cell destruction would only delay the onset of symptoms. Meanwhile, in figure 5, the beta cell level increases at a rate that outpaces the destruction caused by the immune system. In this scenario, the immune system still displays the response characteristic of the onset of diabetes—but as beta cell level is increasing, this response does not matter.

No cyclic behavior occurred at all for the second Hopf bifurcation, as demonstrated by figure 7 and figure 8. These diagrams can be interpreted best in terms of δ_p, the peptide clearance rate. In the model, pancreatic cells die and produce peptide, some of which is not cleared away and produces peptide, triggering an immune response—δ_p controls how much peptide is left over, in essence. For figure 7, where the beta cell level decreased, killing cells causes no oscillatory response because the high clearance rate prevents the immune system from “noticing” the peptide, in effect. Similarly, in figure 8, where beta cells regenerate, the presence of immune cells implies that some beta cells die, but again, the high clearance rate prevents any adverse reaction.

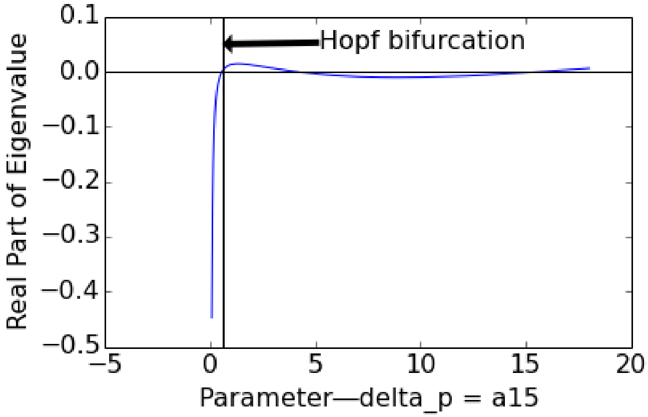

Figure 9 Plot of eigenvalues of system versus parameter value. The areas under the curve before and after the bifurcation do not equal.

To determine when exactly the model “destabilizes” and begins oscillations, the eigenvalues of the system of ODEs can be plotted versus the parameter (i.e. versus time); generally such systems destabilize when the positive area under this plot after the bifurcation equals the area under the plot before [6]. (By definition, a Hopf bifurcation is when the real part of a pair of complex conjugate eigenvalues crosses through zero as the parameter changes [5].) However, as evidenced by the above plot, these areas do not equal, and thus, the condition does not apply to this model.

As previously described, this experiment performed calculations both using XPP and with Python and `mpmath` to avoid error with numerical roundoff. This results only shows results from XPP as in all cases studied herein, the calculations matched qualitatively (results were not compared at a numerical level). However, `mpmath` can help in cases where the necessary numerical precision exceeds that the system can provide. For instance, some non-standard implementations of Fortran provide a quadruple-precision type `real*16` [9] with 113 bits of significand precision [10]. `mpmath`, in contrast, can provide any precision necessary (though being implemented in software, is slower). Consider a hypothetical immune system where $d\delta_p/dt$ is 0.0002. As shown by figure 10 on page 9, using “double-precision” numbers leads to the system incorrectly destabilizing prematurely. With `mpmath` set to a higher precision, the system behaves correctly; thus, Python improved the model significantly with higher precision.

6 Conclusion & Future Research

The results both confirm and refute the hypothesis—depending on the timescale, the oscillations that are indicative of the onset of diabetes may start either earlier or later than expected. Taking into account the error bars on the original data, the simulations here still show the characteristic pattern of low T cell levels that then transition into oscillatory “spiking”. Thus, this confirms the veracity of the original model under a more biologically

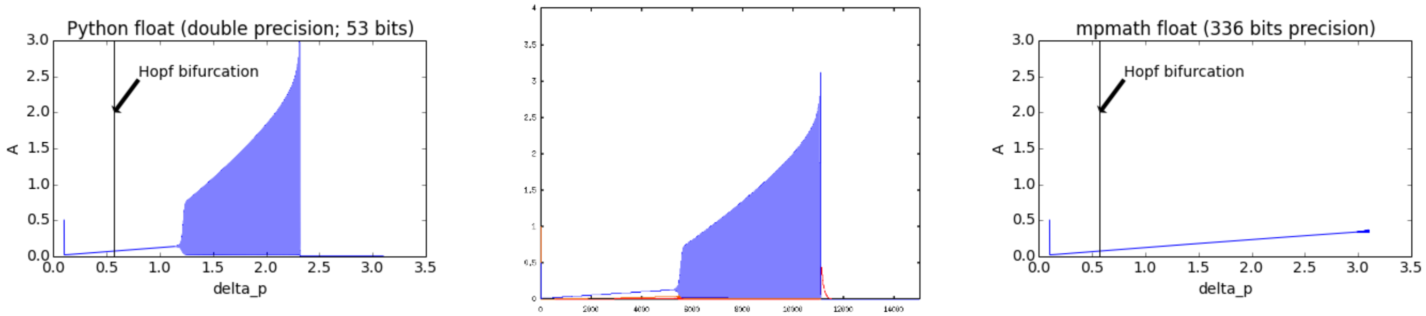

Figure 10 Double-precision (in Python, left; in XPP, center) leads to incorrect results. The precision of Python's float may vary by platform; as reported by `sys.float_info` the bits of precision here is 53.

accurate continuous analysis. However, the results herein further suggest that 1) with sensitive enough tests, the oscillations characteristic of the disease may be detectable earlier than expected, and that 2) slowing the rate of beta cell destruction may delay, but not prevent, those oscillations. Therefore, researchers interested in completely preventing this disease may want to investigate other variables of the model, and being able to theoretically detect the onset of type 1 diabetes earlier than expected before would allow more time for treatment and hopefully a higher quality of life for those affected.

Experimental error in this model may result from round-off due to finite numerical precision. To address this, `mpmath` with Python allowed the calculations to be conducted at arbitrary precision; this data was then compared with the result from XPP. Multiple trials were not necessary as the calculations were deterministic. (Internally, `mpmath` represents values with an arbitrary-length integer mantissa and exponent; the library also guarantees the correctness of basic arithmetic operations, the only ones used here [7].) This experiment additionally demonstrated the viability of Python in this type of research by comparing results from Python to those from the standard tool XPP and showing when the former is more capable.

In this analysis, the level of beta cells varies linearly, but in reality, beta cells regenerate. Thus, extending the system with a model of beta cell regeneration would make it more biologically accurate. For instance, models of type 2 diabetes often account for this and could serve as a starting point for research [Dr. Jiaxu Li, University of Louisville, personal communication, December 6, 2013]. Additionally, conducting further experiments in mice to obtain more data on the oscillations would help fine-tuning the model; note both the large error bars in the original data (see introduction) and the parameter sensitivity of this model [4].

7 References

[1]: Wisse, Brent (2013). Type 1 Diabetes. In *A.D.A.M. Medical Encyclopedia*. Retrieved from http://www.ncbi.nlm.nih.gov/pubmedhealth/PMH0001350/.

[2]: Daneman, D. (2006). Type 1 diabetes. *The Lancet*, *367*.

[3]: Trudeau, J. D. & Kelly-Smith, C. & Verchere, C. B. & Elliott, J. F. & Dutz, J. P. & Finegood, D. T. & Santamaria, P. & Tan, R. (2003). Prediction of spontaneous

autoimmune diabetes in NOD mice by quantification of autoreactive T cells in peripheral blood. *Journal of Clinical Investigation, 111*.

[4]: Mahaffy, J. M. & Edelstein-Keshet, L. (2007). Modeling Cyclic Waves of Circulating T Cells in Autoimmune Diabetes. *SIAM Journal on Applied Mathematics, 67*.

[5]: Voorn, G. (2006). PhD mini course: introduction to bifurcation analysis. Retrieved from http://www.bio.vu.nl/thb/research/project/globif/Globif_text.pdf.

[6]: Baer, S. & Erneux, T. & Rinzel, J. (1989). The Slow Passage Through a Hopf Bifurcation: Delay, Memory Effects, and Resonance. *SIAM Journal on Applied Mathematics, 49*.

[7]: Johansson, F. (2011). Precision and representation issues. *mpmath v0.17 documentation*. Retrieved on September 14, 2013 from http://mpmath.googlecode .com/svn/trunk/doc/build/technical.html.

[8]: Gonsalves, R. J. (2009). Runge-Kutta Methods for ODE Systems. *Computational Physics*. Retrieved on October 24, 2013 from http://www.physics.buffalo.edu /phy410-505-2009/topic3/lec-3-2.pdf.

[9]: Free Software Foundation (2014). KIND Type Parameters. *The GNU Fortran Compiler*. Retrieved on March 14, 2014 from http://gcc.gnu.org/onlinedocs /gfortran/KIND-Type-Parameters.html.

[10]: Zuras, D. & Cowlishaw, M. (2008). IEEE Standard for Floating-Point Arithmetic. *IEEE Std 754-2008, 1*.

8 Acknowledgments

Thanks goes to Dr. Michael McKelvy, Dr. Stephen Baer, Dr. Joseph Mahaffy, and Dr. Jiaxu Li for all their help, and to my parents for driving me everywhere.

Appendix A. Detailed Procedures

Note: familiarity with the Linux command line is assumed.

1. Download the necessary experiment software and experiment files (location TBD).
2. Extract the experiment files to a directory.
3. Open a shell and `cd` to the directory containing the experiment files.
4. To generate plots of the system over time:
   a. Run `python3 rk4_ode.py` to generate the data via the Runge-Kutta method.
   b. Run `./time_diagrams_xppaut.sh` to generate the data via XPP AUTO.
   c. Run `context time_diagrams.tex` to generate a PDF containing the plots.
5. Then, to generate the plots of the system vs the bifurcation parameters (the peptide clearance rate, δ_p; fraction of memory T cells produced, a; T cell competition parameter ε, and peptide level for ½ max memory cells, k_2):
   a. Run `../path_to_xpp/xppaut -runnow mk_static.ode` to launch XPP AUTO. Bring up AUTO and save the points for the complete bifurcation diagram in `mk_static.dat`. If other parameters are to be analyzed, also save the diagrams for those cases.
      i. Press `f a` to bring up AUTO.
      ii. Press `f l` and choose `mk_static.ode.auto` to load the parameters for the bifurcation diagram.
      iii. Press `r s` to generate the diagram.
      iv. Press `g`, `<TAB>` over to the bifurcation point (labeled `HB` on the bottom), press `<ENTER>` to select it, and then `rp` to generate the branch.
      v. Repeat for each bifurcation point.
      vi. Press `f a` and save the points in `mk_static.dat`.
      vii. Close XPP AUTO (`Control-C` the program in the terminal).
   b. Repeat step a for `mk_static_epsilon.ode` with `mk_epsilon.ode.auto` and `mk_static_k2.ode` with `mk_k2.ode.auto`, and then for `mk_continuous_epsilon.ode` and `mk_continuous_k2.ode` with the corresponding `.auto` files. Name the result file after the parameter and `.ode` file (e.g. `mk_continuous_epsilon.dat`, `mk_static_k2.dat`).
   c. Run `python3 bifurcation_diagrams.py` to process the plot data.
   d. Run `context bifurcation_diagrams.tex` to generate a PDF containing the plots.

Parameters for XPP AUTO and Python

The parameters in the following table are *not* parameters for the model, but rather for the programs used to generate the model data. All parameters not listed were left at default values.

Parameter	**Value**
Integration time (the time the system will be run for)	200 days
Integration step (RK4, both Python and XPP AUTO)	0.05 days
`mpmath` Decimal Precision (`mpmath.mp.dps`)	600
AUTO Par 1	`a15`
AUTO Hi-Lo Y-axis	`A`

Parameter	**Value**
AUTO Hi-Lo Main Parameter	`a15`
AUTO Hi-Lo ranges	X: $[0, 18]$ Y: $[0, 3]$
AUTO Par Max	18

Initial Conditions

For the model when processed over time:

$$A = 0.5 \quad M = 0 \quad E = 1 \quad B = 1 \quad a_{15} = \text{See below}$$

For the model when generating bifurcation diagrams for the parameter a_{15}:

$$A = 0.01961524 \quad M = 2.802597 \times 10^{-45} \quad E = 0.006538414 \quad B = 1 \quad a_{15} = 0$$

Parameter Ranges

As the total time is 200 days, the parameter will be varied at a rate of $(\delta_{p_2} - \delta_{p_1})/200$ per day. These ranges were chosen based on a reading of Mahaffy's paper [4]. Although the authors did not call out explicit parameter ranges, the ranges here represent either the domains of graphs given or are near bifurcation points they noted. Also, they tended to include a parameter value of 0, which has been adjusted to 0.1 here to avoid a division-by-zero error. XPP AUTO does not return an error when this occurs, but Python does, so the adjustment avoids conflicts and undocumented behaviors. This will not adversely affect the impact of this experiment; because the original analysis used static parameter values, the overall range of values does not matter as long as it includes the range used in the continuous analysis to be done here for comparison.

δ_p, the peptide clearance rate
0.1 to 2
2 to 0.1
0.5 to 2.3
16.31 to 0.31
3.369 to 19.369