1 Preface

1.1 Context

This four-month USRA project was undertaken with the supervision of Professor Wayne Nagata and Professor Priscilla Greenwood. Here we investigated the stochastic Morris-Lecar neuron, comparing different models and measuring statistics such as interspike interval (ISI) distributions and exit times.

This report is a summary of the work the author has done, and serves as a companion to the codebase developed for this project. There are two types of readers this report targets: knowledgable researchers (such as Professor Priscilla and Professor Nagata) who can find results and code annotations handy in this report, and new students who would additionally benefit from the discussions of theory and resource recommendations.

1.2 Acknowledgements

I express my gratitude towards Professor Nagata and Professor Greenwood for introducing me to mathematical neuroscience, as well as engaging me more into nonlinear dynamics and stochastics. I have learned a lot from Professor Greenwood and Professor Nagata, and I thank them for supervising me.

I also thank the Natural Sciences and Engineering Research Council of Canada for helping fund this project with a USRA.

1.3 Contact Me

I can be reached out to by email (as of the time of this writing) at jwu277@gmail.com.

2 Code Usage

The code developed for this project can be found (as of the time of this writing) at `https://github.com/jwu277/usra2021`. The software is written in Python 3.

2.1 Code Structure

The project repoository is divided into three directories:

1. `neurons`: these files contain classes that are numerical implementations of various neuron models.
2. `scripts`: these files contain runnable scripts that perform various simulations.
3. `util`: these files contain various useful functions that files in the `neurons` and `scripts` directories may use.

Functions and variables starting with an underscore (e.g. `_a`) are intended to be used as private entities (i.e. only used within the file). Another thing is that each `neuron` file except for `neuron.mlr` has a corresponding `scripts` file with the same name that can be run for a quick simulation of the neuron. This mapping is not surjective, however. Now, some scripts will have a line similar to

```
THREADS = 12
```

Feel free to change the number to the appropriate number of threads in your machine. Finally, scripts will sometimes have parameters such as `tmax`, `trials`, etc. As with most variables in the scripts, these can be changed to suit your needs.

A description of the files in the repository is given in **Table 2.1**. Of course, the code files themselves contain more detailed annotations.

File	Description	Notes
`neurons/bm.py`	Binary Markov (BM) neuron	
`neurons/izhikevich.py`	Izhikevich neuron	
`neurons/lif.py`	Linear integrate and fire neuron	
`neurons/ml.py`	Morris-Lecar neuron	
`neurons/mlj.py`	Jacobi Morris-Lecar neuron	Needs to be initialized with `init` before usage

`neurons/mll.py`	Linear Morris-Lecar neuron ($\tilde{\tilde{\mathbf{x}}}$)	Needs to be initialized with `init` before usage
`neurons/mlr.py`	Linear Modulus Morris-Lecar neuron	Not really used. Simulates the modulus of the linear Morris-Lecar neuron.
`neurons/modelb.py`	"Model B" neuron; related to the patched model.	Patched model still needs to be investigated further, so work can still be done here.
`scripts/bm.py`	ISI distribution simulation for the binary Markov neuron	
`scripts/contour.py`	TODO	ipsum
`scripts/det.py`	Compares deterministic trajectories of different neuron models	
`scripts/docheck.py`	TODO	ipsum
`scripts/dtstab.py`	Compares different values of dt and numerical methods(currently Euler and RK45)	$dt = 0.1$ suffices. Also, no appreciable difference was observed between the Euler and RK45 methods.
`scripts/elstats.py`	TODO	ipsum
`scripts/isih.py`	ISI histogram generator for Morris-Lecar neuron	
`scripts/izhikevich.py`	ISI distribution simulation for the Izhikevich neuron withPoisson current	Not really used and might have inconsistent units
`scripts/lif.py`	ISI distribution simulation for the LIF neuron	
`scripts/longtin.py`	Simulation of Longtin's stochastic bistable system	Simulation does not reproduce Longtin's results
`scripts/ml.py`	Simulation of a single trajectory of the Morris-Lecar neuron	
`scripts/mlj.py`	Simulation of a single trajectory of the Jacobi Morris-Lecar neuron	

`scripts/mll.py`	Simulation of a single trajectory of the linear Morris-Lecar neuron	
`scripts/mlrtrials.py`	Simulates distribution of exit times for the `mlr` neuron	
`scripts/modelb.py`	Simulation of a single trajectory of the Model B neuron	
`scripts/pdist.py`	Simulates Poincare statistics (described later in this report) for a fixed values of ψ	
`scripts/phi_noise.py`	Saves a run of Morris-Lecar neuron for certain values ofϕ and N_K	Filename for save should be renamed as desired
`scripts/plotter.py`	Makes an animated plot of a phase portrait trajectory	Requires ImageMagick to be installed
`scripts/poincare.py`	Computes statistics for the L Poincare section	
`util/current.py`	Generates $I(t)$ signal	
`util/dyn.py`	Used to find fixed point of a system	
`util/integrate.py`	Essentially a wrapper for `scipy.integrate.solve_ivp`	
`util/isi.py`	Computes spike times and ISIs for neuron signals	Not useful for Morris-Lecar neuron. Used in BM, Izhikevich, and LIF neurons.
`util/ito.py`	Euler-Maruyama method implementation	
`util/ml.py`	Computes ISIs, orbit times, etc. for Morris-Lecar signals	
`util/plot.py`	Handy plotting functions	

Table 2.1: *Descriptions of all the code files in the project repository.*

2.2 Running Scripts

First, a new user will likely need to run

```
python -m pip install -r requirements.txt
```

in a terminal at the root of the repository. Note: `python` is taken to be the alias for Python 3. After that, the dependencies should be installed and scripts can now be run. To do so, execute

```
python -m scripts.<scriptname>
```

in a terminal again at the root of the repository. Here `<scriptname>` is the name of the script file (excluding the `.py` suffix) to be run. For example, to run a simulation of ISIs for a binary Markov neuron, run

```
python -m scripts.bm
```

As of the writing of this report, a plot should appear after a bit of computation time (exit the plot to terminate the program).

3 Morris-Lecar Neuron

3.1 Circuit Model

The Morris-Lecar description [9] models a neuron as having a membrane potential v, defined to be the difference in voltage between the inside and outside of the neuron cell. Current flows through potassium and calcium channels in the membrane, labelled as I_K and I_{Ca}, respectively. There is also some current arising form other ions, which is collectively totalled as the leak current I_L. The combined potassium, calcium, and leak channels each have an effective conductance (or, taking reciprocals, resistance) and potential. Finally, the membrane has a capacitance C. **Figure 3.1** depicts a circuit for this model.

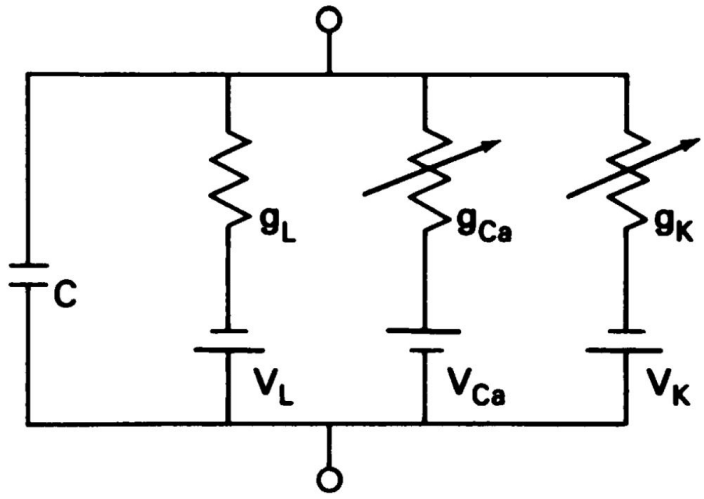


Figure 3.1: *Equivalent circuit of Morris-Lecar neuron model. Only g_{Ca} and g_K may vary with time–the other parameters in this diagram are constants. This diagram is copied from [9].*

Consequently, the membrane potential obeys the following first order differential equation:

$$C\frac{dv}{dt} = I - g_K(v - V_K) - g_{Ca}(v - V_{Ca}) - g_L(v - V_L) \tag{3.1}$$

Nominally, the potassium and calcium conductances are non-constant. Rather, they obey the following equations:

$$g_K = \bar{g}_K w \tag{3.2}$$

$$g_{Ca} = \bar{g}_{Ca} m \tag{3.3}$$

where $\bar{g}_K, \bar{g}_{Ca}$ are constant.

At constant v, the parameters $x = w, m$ are governed by first order differential equations of the following form:

$$\frac{dx}{dt} = \lambda_x(v)(x_\infty(v) - x) \tag{3.4}$$

However, the m timescale is much shorter than the w timescale, so that $m \approx m_\infty(v)$ in (3.1). Furthermore, to be consistent with [5], we may re-arrange the $x = w$ version of (3.4) to be of the following form:

$$\frac{dw}{dt} = \alpha(v)(1 - w) - \beta(v)w \tag{3.5}$$

Finally, we have

$$\alpha(v) = \frac{1}{2}\phi \cosh\left(\frac{v - V_3}{2V_4}\right)\left(1 + \tanh\left(\frac{v - V_3}{V_4}\right)\right) \tag{3.6}$$

$$\beta(v) = \frac{1}{2}\phi \cosh\left(\frac{v - V_3}{2V_4}\right)\left(1 - \tanh\left(\frac{v - V_3}{V_4}\right)\right) \tag{3.7}$$

$$m_\infty(v) = \frac{1}{2}\left(1 + \tanh\left(\frac{v - V_1}{V_2}\right)\right) \tag{3.8}$$

3.2 Morris-Lecar Parameters

The parameters from [5] used in this project are given in **Table 3.1**. Sometimes I and/or ϕ are varied to assess the bifurcational behaviuor of the system.

Variable	Value	Units
C	20	$\mu F/cm^2$
g_L	2.0	mS/cm^2
$\bar{g}_{Ca}$	4.4	mS/cm^2
$\bar{g}_K$	8	mS/cm^2
V_L	-60	mV
V_{Ca}	120	mV
V_K	-84	mV

Variable	Value	Units
V_1	-1.2	mV
V_2	18.0	mV
V_3	2.0	mV
V_4	30.0	mV
ϕ	0.04	ms^{-1}
I	90	$\mu A/cm^2$

Table 3.1: *Values of parameters used in the Morris-Lecar model. These values are assumed for the remainder of the report unless specified otherwise.*

3.3 Helpful Resources

The books [5] and [12] provide good introductions to the Morris-Lecar model.

4 Dynamics

4.1 Bautin Bifurcation

Consider a 2-dimensional dynamical system $(x(t), y(t)) \in \mathbb{R}^2$. We may equivalently define the complex variable $z = x + iy \in \mathbb{C}$. Alternatively, we may express this system in polar coordinates: $z = re^{i\varphi}$. Now, let us consider the system

$$\frac{dz}{dt} = \lambda z \left(1 + k_1 |z|^2 + k_2 |z|^4 + O\left(|z|^5\right)\right) \tag{4.1}$$

$$= \lambda z + c_1 z |z|^2 + c_2 z |z|^4 + O\left(|z|^6\right) \tag{4.2}$$

Here $c_j = \lambda k_j$, and $\lambda, c_j \in \mathbb{C}$ for $j = 1, 2$. Let us rewrite $\lambda = \alpha + i\omega, c_j = l_j + im_j$, where $\alpha, \omega, l_j, m_j \in \mathbb{R}$. Then taking real and imaginary parts of (4.2) gets us

$$\frac{dx}{dt} = (\alpha + l_1 r^2 + l_2 r^4)x - (\omega + m_1 r^2 + m_2 r^4)y \tag{4.3}$$

$$\frac{dy}{dt} = (\omega + m_1 r^2 + m_2 r^4)x + (\alpha + l_1 r^2 + l_2 r^4)y \tag{4.4}$$

Here we dropped terms of order $O\left(|z|^6\right)$ or higher. We do so by assuming that $l_2 < 0$ (and is thus nonzero). The parameter l_2, known as the *second Lyapunov coefficient*, is indeed negative for the Morris-Lecar neuron (as we shall discuss later). Doing some calculus, the equations for r and φ become:

$$\frac{dr}{dt} = r(\alpha + l_1 r^2 + l_2 r^4) \tag{4.5}$$

$$\frac{d\varphi}{dt} = \omega + m_1 r^2 + m_2 r^4 \tag{4.6}$$

We see that there is a fixed point at $r = 0$. Furthermore, there are circular limit cycles at r^2 satisfying the quadratic

$$f(r^2) = r^4 + \frac{l_1}{l_2} r^2 + \frac{\alpha}{l_2} = 0 \tag{4.7}$$

Since these limit cycles occur at $r > 0$, we have precisely one limit cycle for each r^2 root. The discriminant Δ of the quadratic (4.7) is

$$\Delta = \left(\frac{l_1}{l_2}\right)^2 - 4\frac{\alpha}{l_2} = \frac{1}{l_2^2}\left(l_1^2 - 4\alpha l_2\right) \tag{4.8}$$

The sign of Δ is therefore just the sign of $l_1^2 - 4\alpha l_2$. Then a necessary condition for there to be limit cycles i.e. positive real roots for r^2 is:

$$\alpha \geq \frac{l_1^2}{4l_2} \tag{4.9}$$

Now assuming that $\Delta \geq 0$ i.e. the roots for r^2 are all real, we use Vieta's formulas to note that the sum and product of the roots are $-\frac{l_1}{l_2}, \frac{\alpha}{l_2}$, repsectively.

For $\Delta \geq 0$, exactly one positive root exists precisely when $\alpha > 0 \therefore \frac{\alpha}{l_2} < 0$. But we immediately have $\alpha > 0 \geq \frac{l_1}{4l_2}$, satisfying the discriminant condition.

Now if $\alpha = 0$, then at least one root must coincide with $r = 0$. The other root is thus equal to $-\frac{l_1}{l_2}$ and is positive iff $l_1 > 0$.

Next, if $\alpha < 0$, both roots have the same sign. If $l_1 \leq 0$ then the sum of the roots is nonpositive, resulting in both roots being nonpositive. Thus there are no limit cycles for $\alpha < 0, l_1 \leq 0$. However, if $\alpha < 0$ and $l_1 > 0$, there are two positive roots for r^2 if $\alpha > \frac{l_1^2}{4l_2}$. If instead, $l_1 > 0$ and $\alpha = \frac{l_1^2}{4l_2} < 0$ then there is a repeated positive root and therefore one limit cycle.

Finally, the stability of the limit cycles and $r = 0$ fixed point is determined by the sign of $f(r^2)$ in between these roots. Since $l_2 < 0$, the outermost limit cycle (or fixed point if none exists) attracts from $r = \infty$. Meanwhile, the stability of the fixed point is determined by the sign of α (or l_1 if $\alpha = 0$, or $l_2 < 0$ if $\alpha, l_1 = 0$). Finally, if there are two limit cycles, the inner one is unstable while the outer one is stable since $f(r^2)$ is a downwards parabola in r^2.

Put all together, the bifurcation diagram is given in **Figure 4.1**

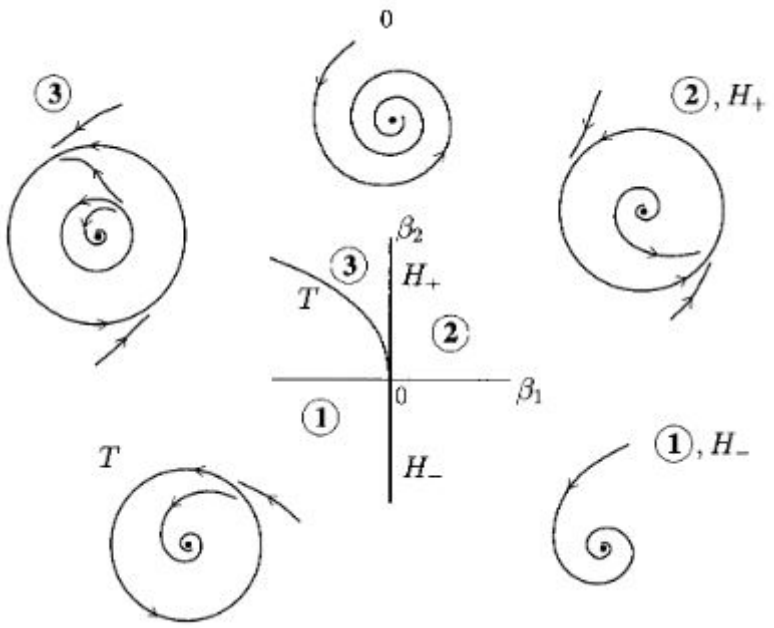


Figure 4.1: *Bifurcation diagram of Bautin bifurcation. This figure was obtained from [6]. This diagram labels α and l_1 as β_1 and β_2, respectively.*

The curve $H_- = \{\alpha = 0, l_1 < 0\}$, in **Figure 4.1** is a *Hopf bifurcation.* For a fixed $l_1 < 0$,

varying α near zero affects the stability of the system; $\alpha < 0$ has a limit cycle whereas $\alpha > 0$ does not. Similarly, $H_+ = \{\alpha = 0, l_1 > 0\}$ is also a Hopf bifurcation. This time, a limit cycle is produced at the fixed point when varying $\alpha > 0$ to $\alpha < 0$. In general, the Hopf bifurcation is a local codim-1 bifurcation, where a limit cycle appears from/disappears into a fixed point.

The curve $T = \{l_1 > 0, \alpha = \frac{l_1^2}{4l_2}\}$ is a *saddle node of limit cycles (SNLC) bifurcation*. Here the two limit cycles in region 3 collapse into a single limit cycle on T. Varying a parameter more (e.g. decreasing α or l_1) will cause the semi-stable limit cycle to disappear. Like the Hopf bifurcation, the SNLC bifurcation is a local codim-1 bifurcation.

Finally, a *Bautin bifurcation* occurs near the point $\alpha, l_1 = 0$ in this model system. Since two parameters are varied in this bifurcation (α, l_1), this is a local codim-2 bifurcation.

4.2 Morris-Lecar Dynamics

A Bautin bifurcation exists for the Morris-Lecar neuron in the parameters (I, ϕ) near $(I, \phi) = (90, 0.04)$. The bifurcation diagram is given in **Figure 4.2**. We see that

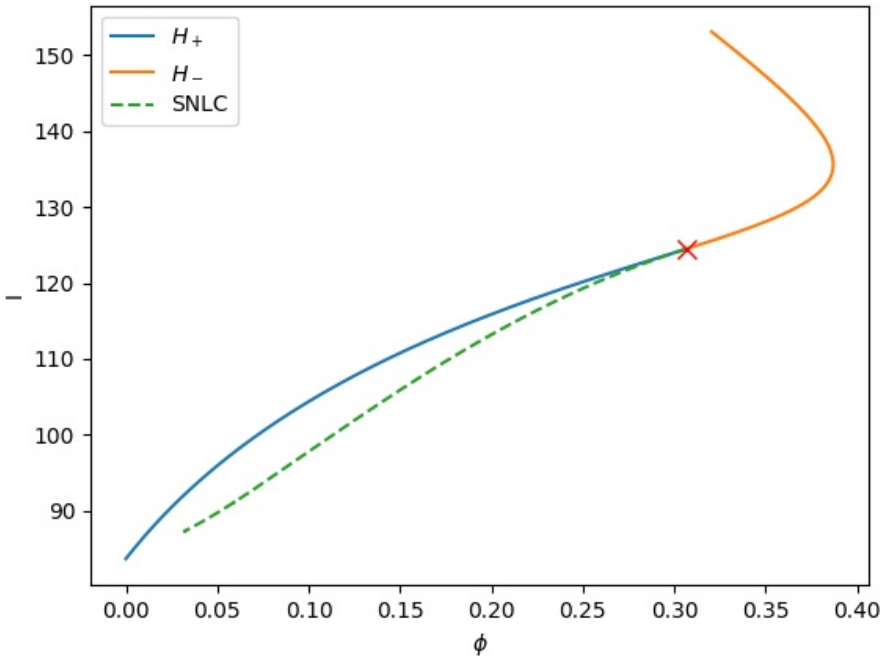


Figure 4.2: *Bautin bifurcation diagram, produced using [3]. The red cross represents the origin of the Bautin bifurcation.*

$(I, \phi) = (90, 0.04)$ corresponds to a system of type 3 in **Figure 4.1**. The salient feature

here is that there is a fixed point inside an unstable limit cycle inside a stable limit cycle. Corroborating this result, we keep $\phi = 0.04$ fixed and vary I only. That gives us the bifurcation diagram in **Figure 4.3**.

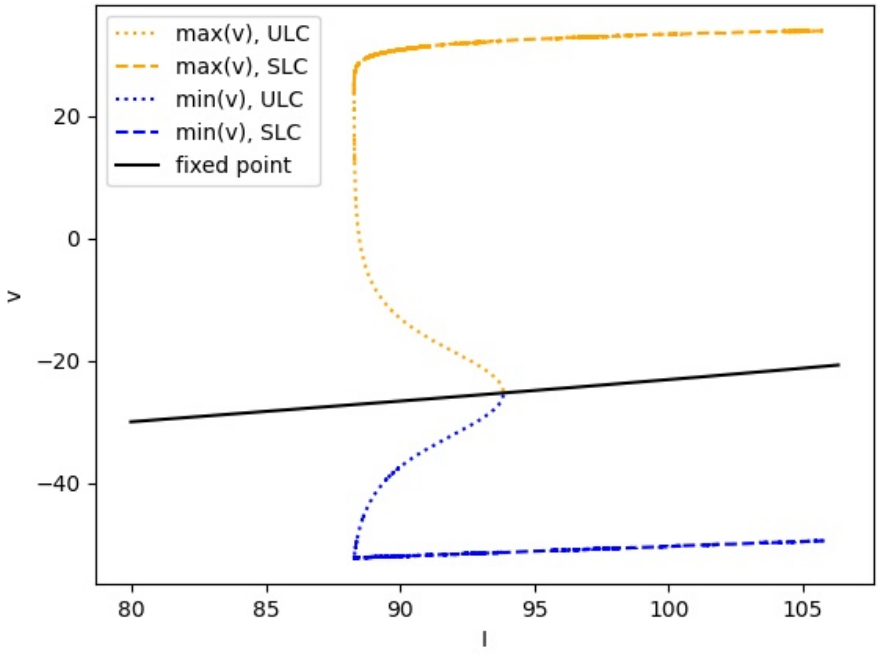


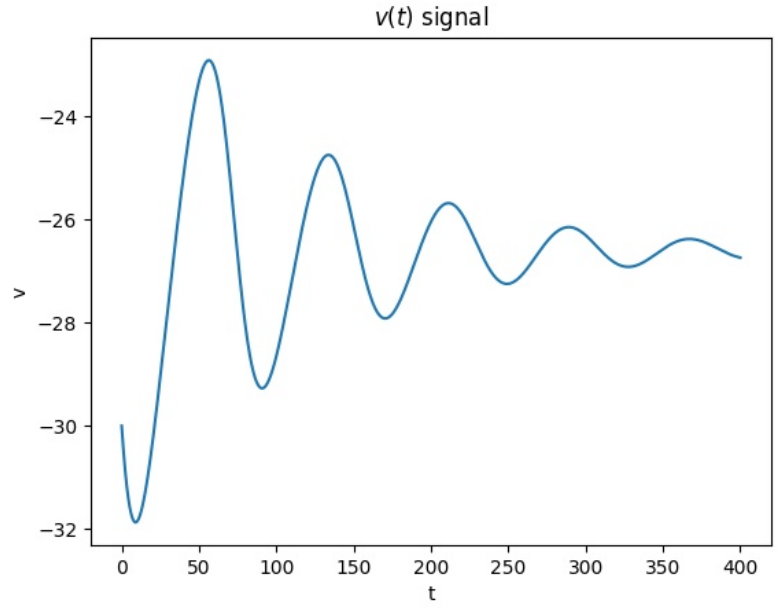
Figure 4.3: *Bifurcation diagram, produced using [3], with $\phi = 0.04$ fixed and I varied. The value of v at the fixed point, as well as extremal values of v along the stable (SLC) and unstable (ULC) limit cycles are tracked.*

In the (v, w) phase plane, trajectories closing in on the fixed point represent *quiescent* oscillations. In contrast, trajectories orbiting near the stable limit cycle exhibit *spiking* oscillations. Since this system is bistable, these are the only types of oscillations that can arise. The switching between the quiescent and spiking states describe neuronal behaviour and is of interest to us. Example phase portraits and corresponding $v(t)$ signals of deterministic quiescent and spiking oscillations are given in **Figure 4.4** and **Figure 4.5**, respectively. Note that the phase portrait flows counterclockwise.

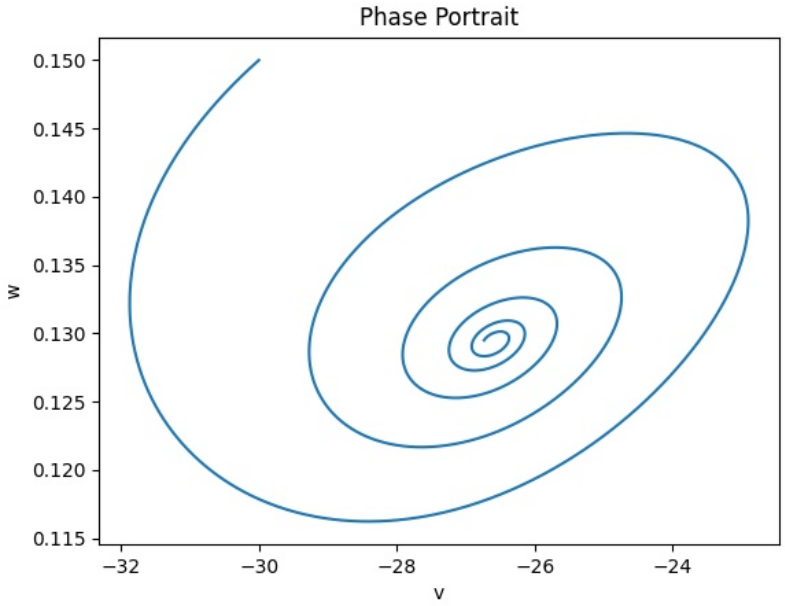
These figures can be reproduced by running the `ml` script:

```
python -m scripts.ml
```

To get a deterministic trajectory, set the `stochastic_method` variable in the `scripts/ml.py` file to `None`:

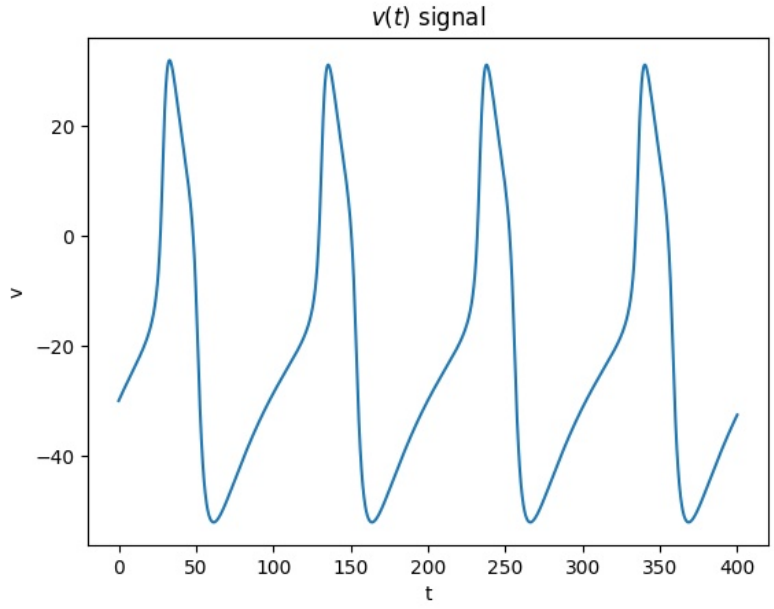


(a) $v(t)$ *signal*

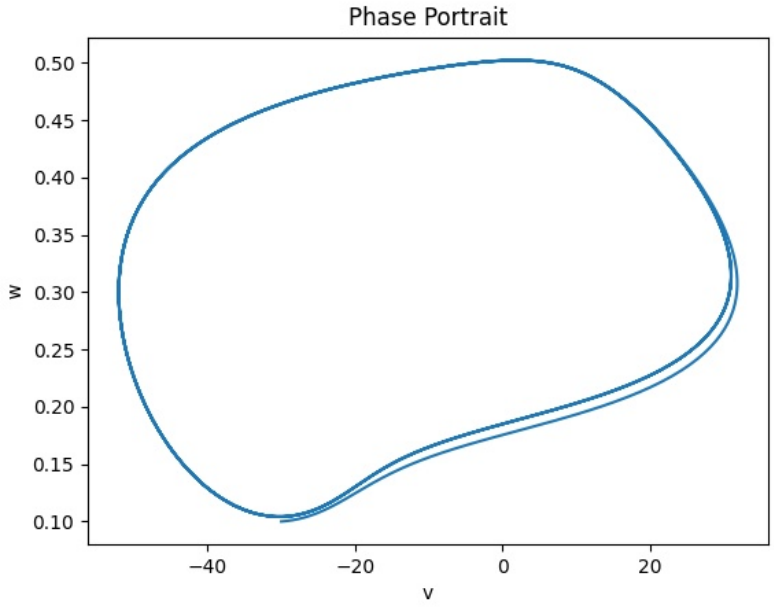


(b) Phase portrait

Figure 4.4: *Quiescent trajectory with initial point* $(-30.0, 0.15)$ *and* $t_{max} = 400$. *As is the case for this report, voltage and time have units of* mV *and* ms*, respectively.*



(a) $v(t)$ signal



(b) Phase portrait

Figure 4.5: *Spiking trajectory with initial point* $(-30.0, 0.1)$ *and* $t_{max} = 400$.

```
stochastic_method = None
```

For a quiescent trajectory, set the initial condition

```
x0 = np.array([-30.0, 0.15])
```

and run the script. Meanwhile, for a spiking trajectory, set the initial condition

```
x0 = np.array([-30.0, 0.1])
```

and run the script.

4.3 Scaling Results

The results in this section were obtained with the use of `MATCONT` [3].

Some potentially interesting scaling results in the Morris-Lecar neuron were obtained. First, along the H_+ curve in the Bautin bifurcation diagram, the fixed point has two imaginary eigenvalues that are conjugates of each other (since the real part goes to zero for Hopf bifurcations). It turns out that this imaginary part ω scales as $\sqrt{\phi}$:

$$\omega \sim \sqrt{\phi} \tag{4.10}$$

This fit is compared to the numerical data in **Figure 4.6**.

This agrees with the theoretical result obtained in [1] (in that paper, the authors use ε and λ in place of ϕ and I, respectively).

The scaling between I and ϕ along the H_+ and SNLC curves for small ϕ was also investigated. Along the H_+ curve we have:

$$I - I_0 \sim \phi \tag{4.11}$$

Along the SNLC curve we get:

$$I - I_0 \sim \phi^{4/3} \tag{4.12}$$

Here $I_0 = 83.6$ is the value of I in the $\phi \to 0^+$ limit (the H_+ and SNLC curves appear to approach the same point). These fits are compared to numerical data in **Figure 4.7**.

4.4 Helpful Resources

The book [11] provides an excellent introduction to bifurcation theory. A more advanced book on bifurcation theory is [6]. Meanwhile, the software `MATCONT` [3], which can be run through MATLAB, is really useful for numerical bifurcations. A helpful tutorial guide for `MATCONT` is [8].

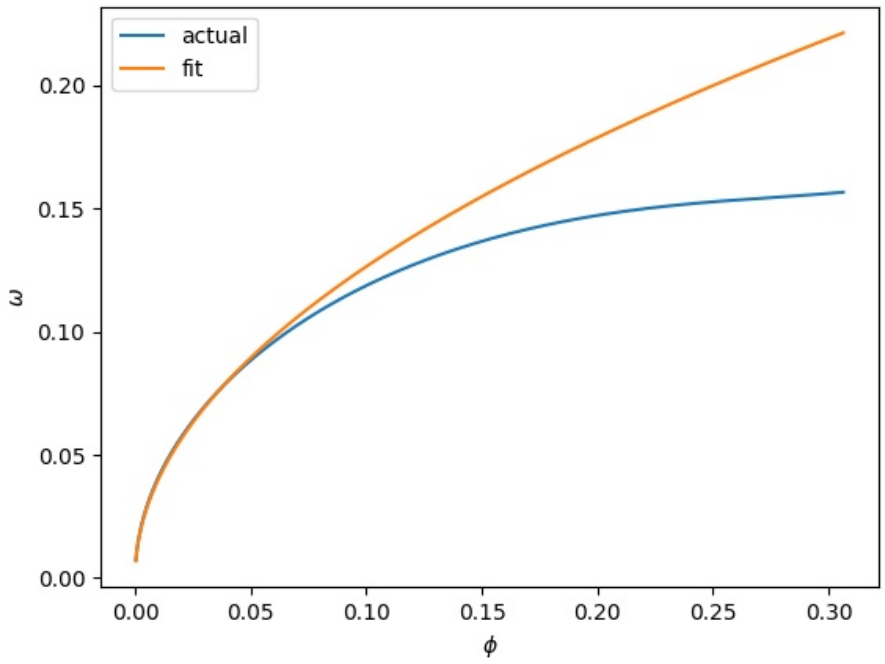


Figure 4.6: *Comparison of fit power law scaling and actual numerical data for $\omega(\phi)$ at small ϕ. The fit obeys the equation $\omega = 0.4\sqrt{\phi}$.*

5 Stochastics

5.1 Itô Calculus

X_t is a *Wiener process* iff it is a continuous-time stochastic process satisfying the following conditions [7]:

1. **Stationary increments:** If $s \leq t$, $X_t - X_s$ has the same distribution as $X_{t-s} - X_0$.
2. **Independent increments:** If $s \leq t$, $X_t - X_s$ is independent of X_r for all $r \leq s$.
3. **Continuous paths:** X_t is a continuous function of t.

Here, a Wiener process is characterized by three paramters: the initial condition X_0, the drift m, and the variance parameter σ^2. For $s \leq t$, the increment $X_t - X_s$ is distributed as a Gaussian with mean $m(t-s)$ and variance $\sigma^2(t-s)$:

$$X_t - X_s \sim N\left(m(t-s), \sigma^2(t-s)\right) \tag{5.1}$$

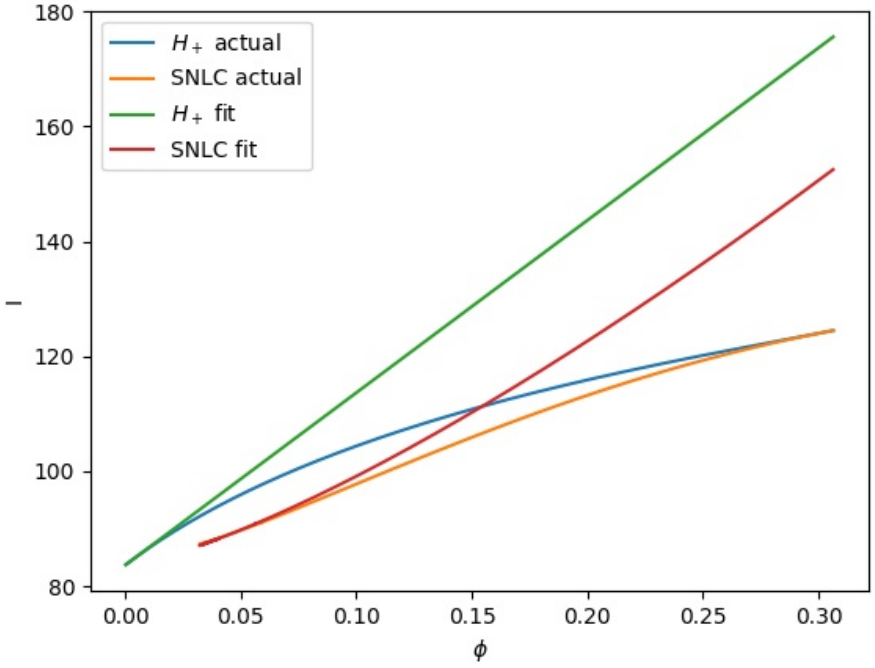


Figure 4.7: *Comparison of fit power law scaling and actual numerical data for $I(\phi)$ at small ϕ along the H_+ and SNLC bifurcation curves. The fits obey the relationships $I = 83.6 + 300\,\phi$ and $I = 83.6 + 333.3\,\phi^{4/3}$ along the H_+ and SNL curves, respectively.*

The *standard* Wiener process B_t has $B_0 = 0$, $m = 0$, and $\sigma^2 = 1$. We may therefore write

$$dX_t = mdt + \sigma dB_t \tag{5.2}$$

Note that dB_t is distributed as

$$dB_t \sim \sqrt{dt}Z \tag{5.3}$$

where Z is a standard normal random variable. Now in general, if f is a function of X_t, then we may write

$$df(X_t) = g(X_t)dt + h(X_t)dB_t \tag{5.4}$$

for some functions $g(X_t), h(X_t)$ (Itô's lemma gives a more precise result, but we will not utilize it for these purposes).

The n-dimensional standard Wiener process $\mathbf{B_t}$ is simply a vector of n independent standard Wiener processes. Then an n-dimensional dyanmical system $\mathbf{X}$ can be expressed in the form

$$d\mathbf{X} = \mathbf{g}(\mathbf{X})dt + \mathbf{H}(\mathbf{X})\mathbf{dB_t} \tag{5.5}$$

Here $\mathbf{g}$ is a vector while $\mathbf{H}$ is a matrix.

5.2 Stochastic Morris-Lecar

We can model the neuron as having N_K potassium channels (in this project, we are typically on the order of $N_K \sim 1000$), each of which can be fully open or closed. The opening rate (i.e. probability rate of transition from the closed state to the open state) and closing rate for each channel would be $\alpha(v)$ and $\beta(v)$ from (3.5), respectively. Then over a small time increment dt, the probability of a closed channel opening is $\alpha(v)dt$. Likewise, the probability of an open channel closing is $\beta(v)dt$.

Over a time increment of dt, we have

$$dw = w(t+dt) - w(t) \sim \frac{\text{Binom}\left(N_K(1-w), \alpha dt\right) - \text{Binom}\left(N_K w, \beta dt\right)}{N_K} \tag{5.6}$$

Note that $\text{Binom}\left(n, p, =\right) \sum_{j=1}^{n} I_j$, where each I_j is an independent indicator random variable with probability p. By the central limit theorem, when n is large

$$\text{Binom}\left(n, p\right) \sim np + \sqrt{np(1-p)}Z \tag{5.7}$$

where Z is a standard normal random variable. Then (5.6) becomes

$$dw \sim \frac{\left(N_K(1-w)\alpha dt + \sqrt{N_K(1-w)\alpha dt(1-\alpha dt)}Z_1\right) - \left(N_K w\beta dt + \sqrt{N_K w\beta dt(1-\beta dt)}Z_2\right)}{N_K} \tag{5.8}$$

Here Z_1 and Z_2 are independent normal random variables. This is because the closed and open channels are independent (as we assumed all the channels to be independent). Since dt is small, $1 - \alpha dt \approx 1$. So

$$dw \sim \left(\alpha(1-w) - \beta w\right) dt + \frac{1}{\sqrt{N_K}}\sqrt{\alpha(1-w) + \beta w}\sqrt{dt}Z \tag{5.9}$$

As a system of stochastic differential equations, we have

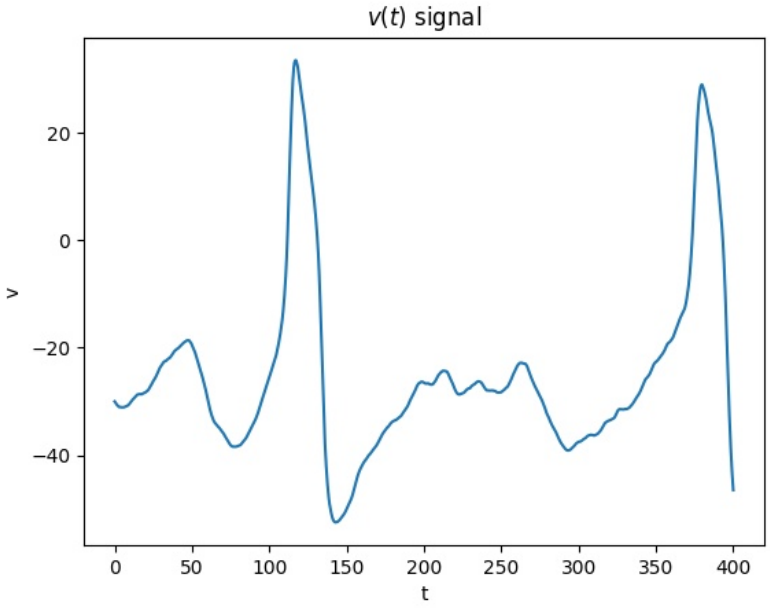
$$\begin{pmatrix} v \\ w \end{pmatrix}' = \begin{pmatrix} f(v,w) \\ g(v,w) \end{pmatrix} dt + \begin{pmatrix} 0 & 0 \\ 0 & h(v,w) \end{pmatrix} \mathbf{dB_t} \tag{5.10}$$

$$f(v,w) = \frac{I - \bar{g}_K w(v - V_K) - \bar{g}_{Ca} m_\infty(v)(v - V_{Ca}) - g_L(v - V_L)}{C} \tag{5.11}$$

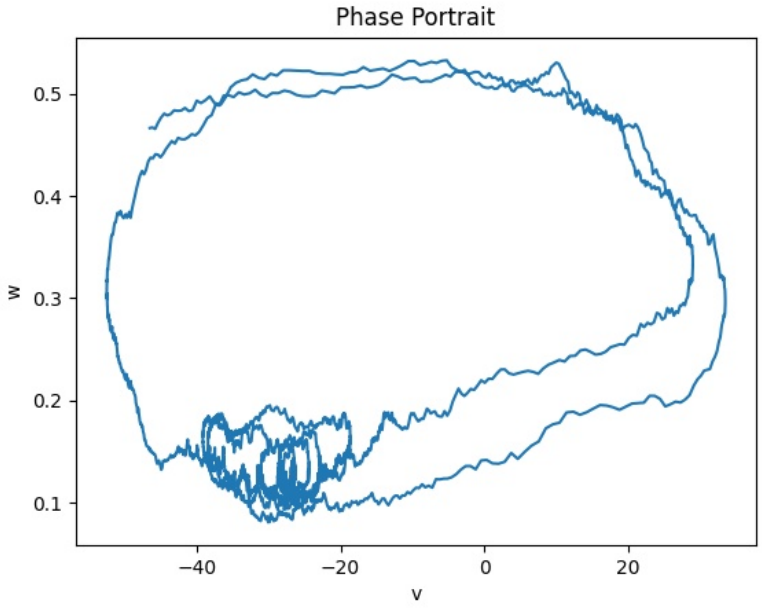
$$g(v,w) = \alpha(v)(1-w) - \beta(v)w \tag{5.12}$$

$$h(v,w) = \frac{1}{\sqrt{N_K}}\sqrt{\alpha(v)(1-w) + \beta(v)w} \tag{5.13}$$

Example phase portraits and corresponding $v(t)$ signals of stochastic quiescent and spiking oscillations are given in **Figure 5.1** and **Figure 5.2**. Note that the trajectory in **Figure 5.1** transitions between quiescent and spiking states, as opposed to staying in the quiescent state like its deterministic counterpart. Such transitioning between quiescent and spiking states make the neuronal behaviour interesting and is worthy of closer investigation.

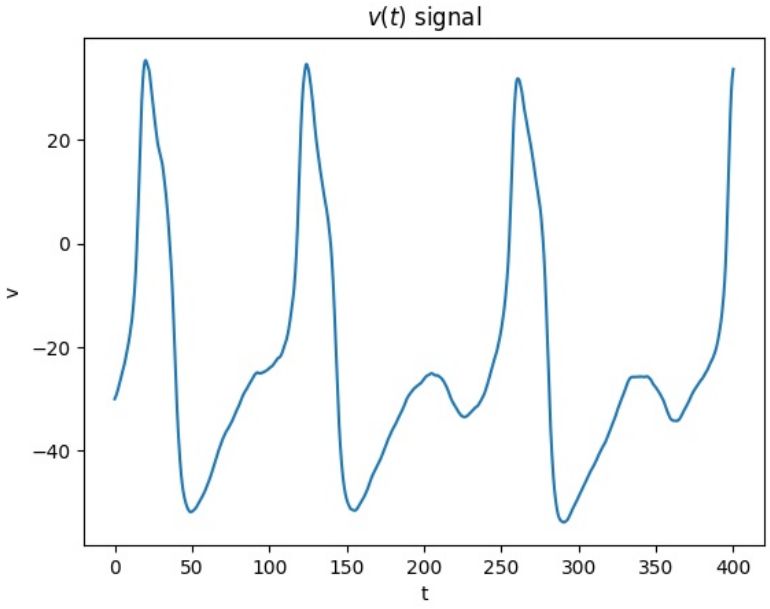


(a) $v(t)$ signal

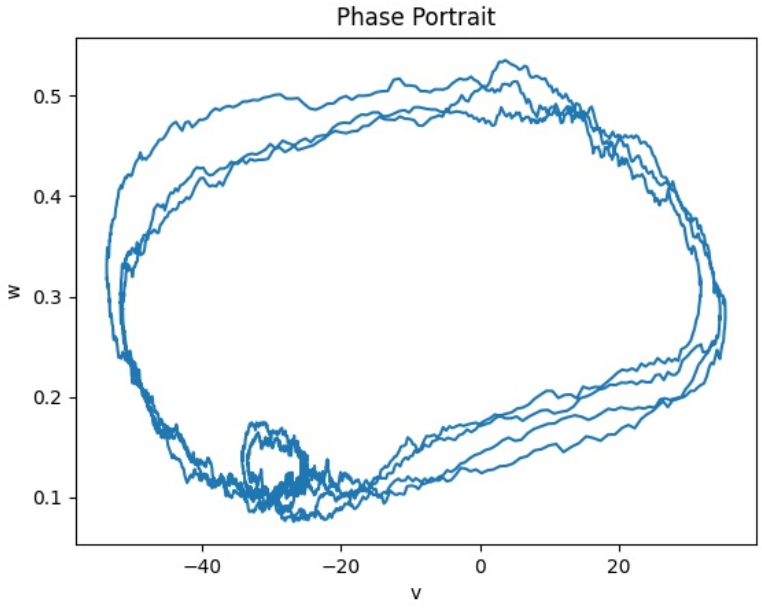


(b) Phase portrait

Figure 5.1: *Stochastic trajectory with initial point* $(-30.0, 0.15)$ *and* $t_{max} = 400$. *Parts of the trajectory are quiescent while other parts are spiking.*



(a) $v(t)$ signal



(b) Phase portrait

Figure 5.2: *Stochastic spiking trajectory with initial point* $(-30.0, 0.1)$ *and* $t_{max} = 400$. *Parts of this trajectory are quiescent.*

5.3 Euler-Maruyama Method

The Euler-Maruyama method is a generalization of the Euler method to stochastic differential equations of the form (5.5). Here, a small finite time increment dt is chosen. Then, starting from an initial condition $\mathbf{X}(0)$, we iteratively compute

$$\mathbf{X}(t+dt) = \mathbf{X}(t) + \mathbf{g}(\mathbf{X})dt + \mathbf{H}(\mathbf{X})\mathbf{Z}\sqrt{dt} \tag{5.14}$$

Here $\mathbf{Z}$ is a randomly generated vector, where each entry is obtained from an independent unit normal distribution. The resulting path $\mathbf{X}(t)$ will be distributed approximately as the true distribution; thus, many trials of the Euler-Maruyama method can be used to obtain statistics for the process (e.g. ISIs, which will be discussed later).

To ease computations, only the second entry of $\mathbf{Z}$ needs to be generated for the 2-dimensional stochastic Morris-Lecar model (since only $\mathbf{H_{22}}$ is nonzero). Furthermore, the random numbers can all be generated at the start rather than at every iteration.

The Euler-Maruyama method is implemented in the project's software. One such script that utilizes this is `scripts/ml.py`. Here a stochastic trajectory of the Morris-Lecar neuron can be simulated using the Euler-Maruyama method. This can be done by first setting the appropriate stochastic method in the file:

```
stochastic_method = 'euler'
```

Then the script can be run in a terminal:

```
python -m scripts.ml
```

The plots in **Figure 5.1** and **Figure 5.2** can be reproduced (up to stochastic noise) by running this script with the Euler-Maruyama method.

5.4 Helpful Resources

The book [7] provides a great introduction to stochastic calculus.

6 Interspike Intervals

One statistic of interest in this project is the ISI distribution. Intuitively, these intervals are the lengths of time between successive spikes in the membrane potential. Adding stochastics to the Morris-Lecar neuron results in a probabilistic distribution of intervals.

6.1 Defining Intervals

Defining interspike intervals boils down to defining spike times for a signal. Looking at the $v(t)$ signal in **Figure 4.5a**, we see that these ought to be the distinct local maxima in the $v(t)$ signal. However, not all $v(t)$ maxima qualify as spikes. For example, the quiescent signal in **Figure 4.4a** exhibits maxima, yet should not count as having any spikes. To complicate matters further, the noisy signals in **Figure 5.1a** and **Figure 5.2a** have both quiescent and spiking maxima. An easy way of differentiating these two maxima is to introduce a threshold (e.g. 20mV) that a maxima must exceed to constitute a spike.

The ISIs in a signal can then be computed from the differences of these spike times. A given signal will typically have many spike times, giving rise to several ISIs. If we assume the stochastic Morris-Lecar neuron to be ergodic–which it ought to be at least approximately so, considering (granted, I have not proved this, or even gone much beyond *considering* these assertions) that (a) the system can be expanded as an Ornstein–Uhlenbeck process near the fixed point and stable limit cycle, which should be ergodic, and (b) a finite transition probability exists between the two stable states–then running the system for a long time will give us the approximate ISI distribution.

Spike times and ISIs for a Morris-Lecar signal can be obtained using functions from the `util/ml.py` file. To reuse software components, the spike times are computed slightly differently than described in this section, however. *Orbit times* (times at which an oscillation begins) are first computed, from which spiking times are determined. The orbit time calculation requires a `vdiff` parameter that represents at least how much maxima and minima of $v(t)$ should differ by. Meanwhile, the `sthresh` parameter is simply the spiking voltage threshold.

6.2 Results

Numerically simulated ISI histograms for the stochastic Morris-Lecar neuron at various values of N_K are given in **Figure 6.1**. Recall from (5.13) that the noise amplitude is proportional to $N_K^{-1/2}$. The salient features of these histograms are the discrete peaks followed by the exponential tail.

The discrete peaks and exponential tail can be explained as follows. Between every two consecutive spikes, there is one spiking oscillation and $n \geq 0$ quiescent oscillations. The

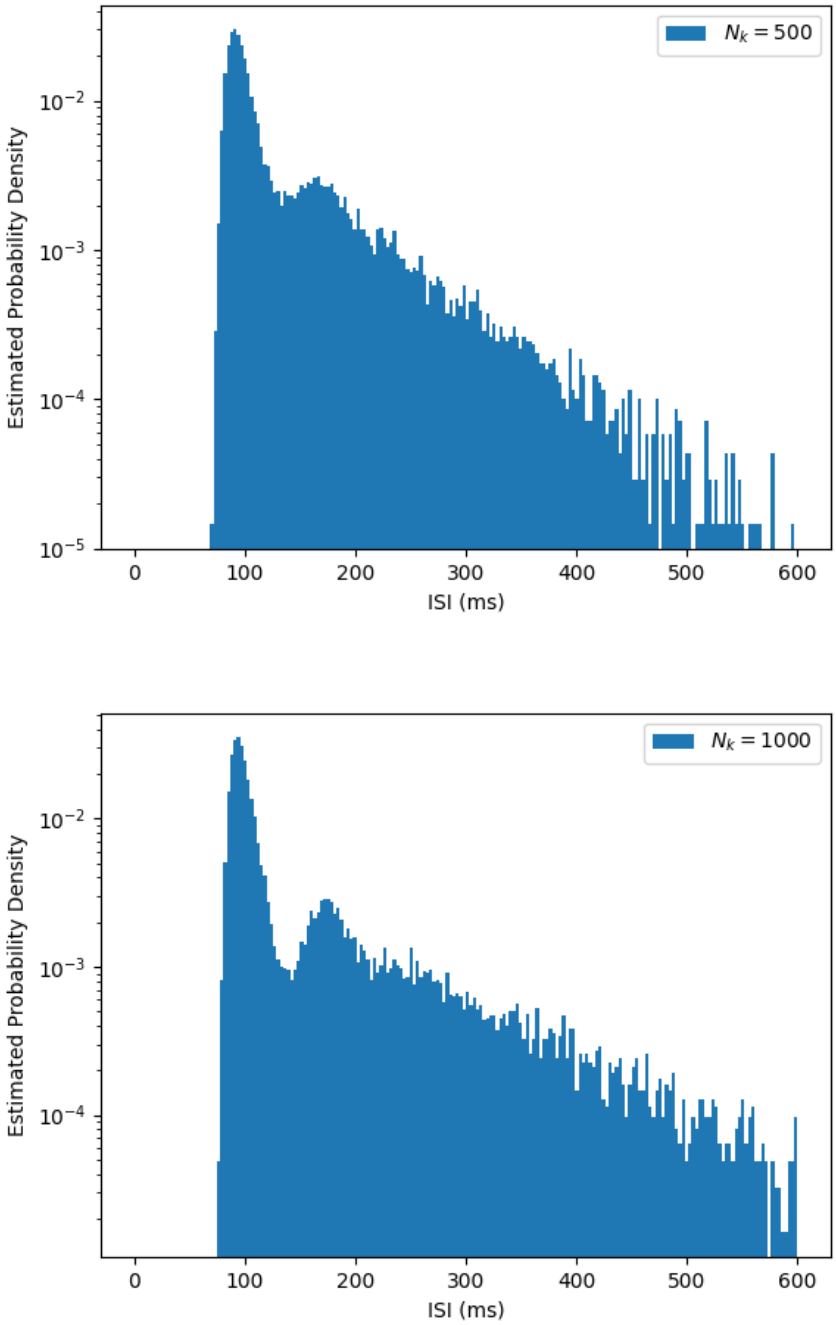


Figure 6.1: *ISI histograms for various values of N_K. These histograms were produced by running 3200 trials of the Morris-Lecar neuron for 1000 ms each, with the initial point* $(v, w) = (-40.0, 0.42)$.

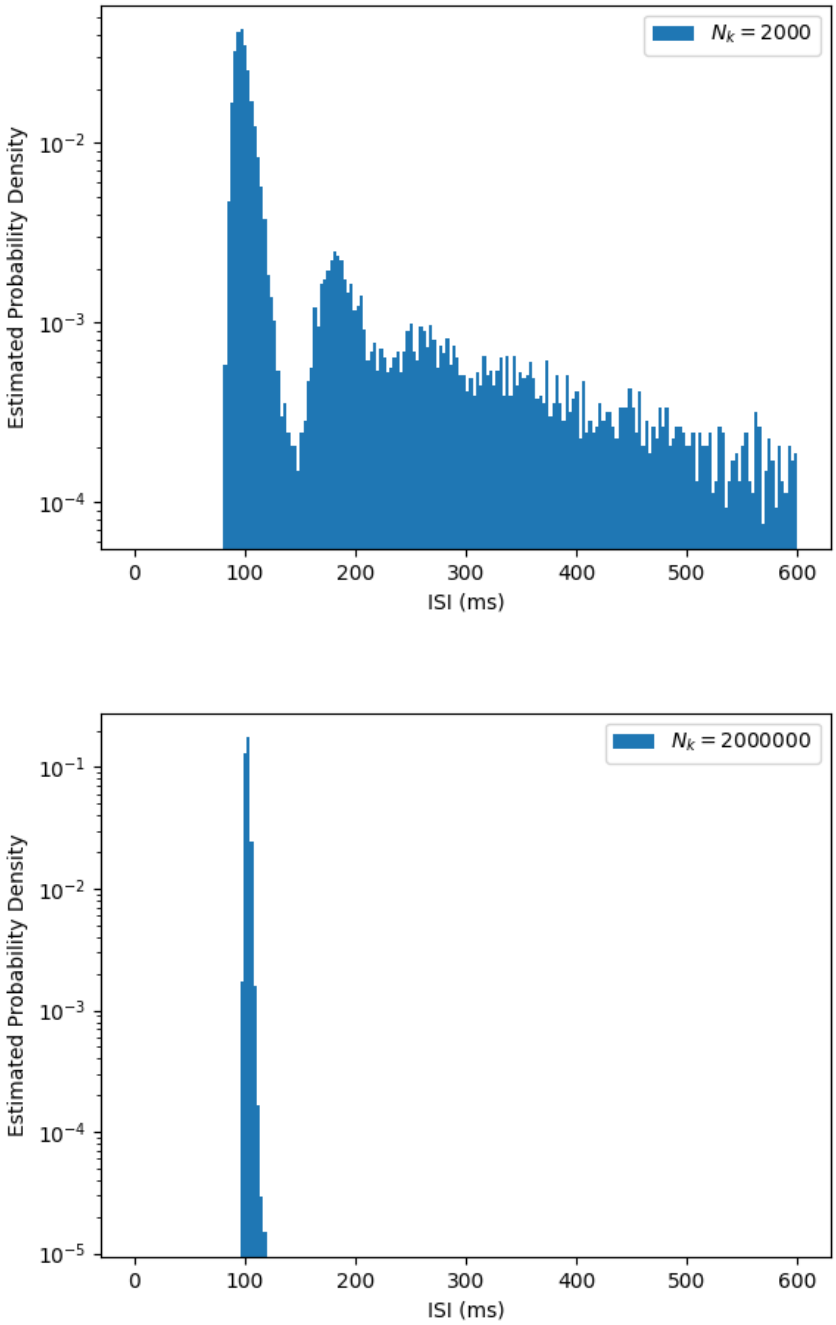


Figure 6.1: *ISI histograms for various values of N_K. These histograms were produced by running 3200 trials of the Morris-Lecar neuron for 1000ms each, with the initial point $(v, w) = (-40.0, 0.42)$. (cont.)*

total time T between two such spikes is then

$$T = S + \sum_{i=1}^{n} Q_i \tag{6.1}$$

where S is the time duration of the spiking oscillation and Q_i is the time duration of the i-th quiescent oscillation. As we might expect, S and Q_i are distributed unimodally with expectation values of approximately 100ms and 80ms, respectively. The standard deviations appear to be small compared to the EVs for these times.

Thus the first discrete peak (ordering in terms of increasing ISI) corresponds to $n = 0$. Indeed, this peak is centred at around 100 ms, with the width being attributed to the deviation in S. Likewise, the second, third, etc. peaks correspond to $n = 1, 2$, etc. The centres of these peaks would be located at $100 + 80n$ milliseconds, with the width increasing with increasing n. This latter assertion arises from how standard deviation increases in quadrature as more independent RVs are added (we intuitively expect S and Q_i to be approximately independent–a claim we scrutinize more closely when we look at Poincare maps).

We also expect the peak heights to decay exponentially with n. This can be seen from the following argument: if we assume quiescent oscillation to be approximately independent of each other, then each quiescent oscillation has a probability p of transitioning to a spiking oscillation. Then $n \geq 1$ should follow a geometric distribution–hence the exponential decay in peak height.

Now, once n becomes large enough, the successive peaks in the ISI histogram become so wide that they blend in with each other when superposed. The remaining feature of these blended peaks is therefore their heights (i.e. probability density) in the histogram. Per the previous discussion, this results in the exponential tail.

The numerically obtained ISI histograms agree with those found elsewhere, namely in [10]. The results in this report can be reproduced using the `scripts.isih` script. Some parameters that can be changed in the script include `Nk`, `tmax`, and `epochs`. These correspond to N_K, the duration of a single trajectory, and the number of trajectories to simulate, respectively.

7 Poincare Maps

7.1 Background

A *Poincare section* is a cross-sectional subset of a system's phase space that is transverse to oscillatory trajectories. For example, if we let (v_{eq}, w_{eq}) be the fixed point of the Morris-Lecar neuron, we can define the two Poincare sections L, L' as:

$$L = \{(v, w)\ s.t.\ v = v_{eq}, w \leq w_{eq}\} \tag{7.1}$$
$$L' = \{(v, w)\ s.t.\ v = v_{eq}, w \geq w_{eq}\} \tag{7.2}$$

These subsets are depicted in **Figure 7.1**.

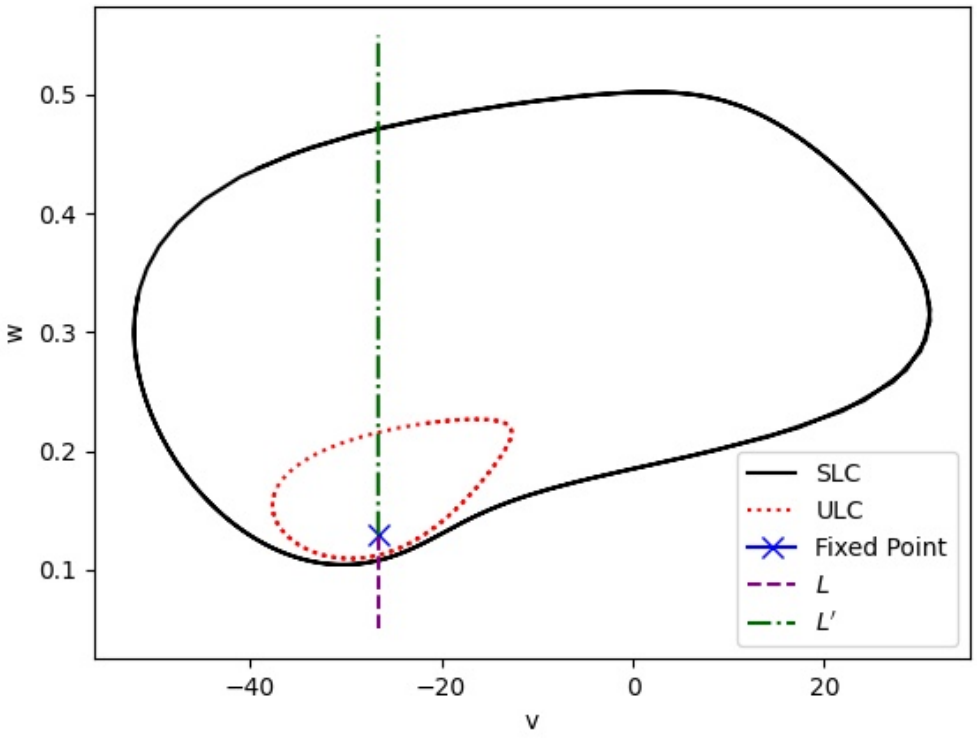


Figure 7.1: *Plot of the Poincare sections in the (v, w) phase plane. The fixed point and limit cycles are included for comparison.*

Indeed, trajectories such as those in **Figure 4.4** and **Figure 4.5** cross each of L, L' once per oscillation. More formally, for a deterministic system, if we abbreviate $\mathbf{x}(t) = (v(t), w(t))$, then a time t^* is a *crossing time* through a Poincare section S iff $\mathbf{x}(t) \in S$. So long as the

trajectories are not tangent to the subset, there are only countably many crossing times. This is intuitively clear because trajectories must "shoot through" the section and cannot remain on it for more than an instant of time. We associate with every crossing time t^* a *crossing point* $\mathbf{x}(t^*) \in S$. We can thus define a function $P : S \to S$ for a Poincare section S, where $P(\mathbf{x}) = \mathbf{y}$ iff a trajectory starting at $\mathbf{x}$ has its next crossing point at $\mathbf{y}$ (the existence of $\mathbf{y}$ is guarenteed by how S is transverse to oscillatory motion by definition; meanwhile, the uniqueness comes from the Well-Ordering Principle). The $P(\mathbf{x})$ function is known as a *Poincare map*. We can similarly define the function $T(\mathbf{x}) : S \to \mathbf{R}$, where $T(\mathbf{x})$ is the next crossing time of a trajectory starting at $\mathbf{x}$. We shall call $T(\mathbf{x})$ the *Poincare timer*.

The L and L' sections are one-dimensional, so it helps to parametrize them by a single real parameter. Using the parameters $\psi, \psi' \geq 0$, we can parametrize $\psi \leftrightarrow (v_{eq}, w_{eq} - \psi)$ and $\psi' \leftrightarrow (v_{eq}, w_{eq} + \psi)$ for L and L', respectively. We then have the functions $P(\psi)$ and $T(\psi)$ (or $P(\psi')$ and $T(\psi')$) as the Poincare map and Poincare timer.

As an example, the Poincare map for L is plotted in **Figure 7.2**. Notice that there are

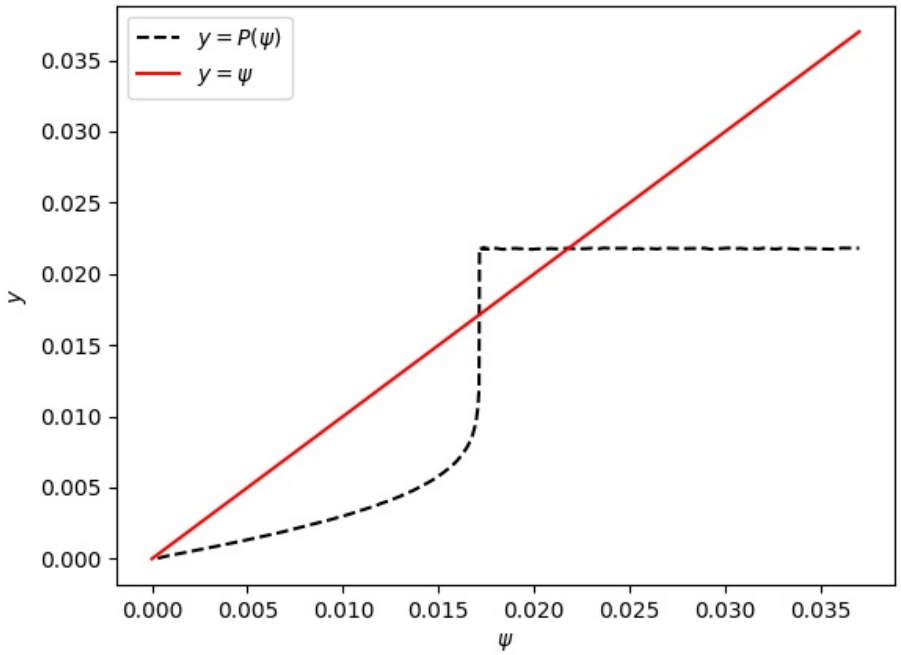


Figure 7.2: *Deterministic Poincare map for the L section.*

three fixed points (i.e. $P(\psi) = \psi$) of the Poincare map. In order of increasing ψ, these correspond to the system's fixed point, unstable limit cycle, and stable limit cycle. Of course, fixed points of a deterministic Poincare map will always correspond to equilibria or limit cycles, and vice versa. The stability of the system's equilibria and limit cycles can

then be inferred from the Poincare map. Here, we see that $P(\psi) < \psi$ for points between the fixed point and ULC. This implies that the fixed point is stable while the ULC is indeed unstable from the inside. This is because trajectories in this region move closer towards the fixed point after each oscillation. Similarly, the fact that $P(\psi) > \psi$ between the ULC and SLC further confirms that the ULC is unstable from the outside and that the SLC is stable from the inside. Finally, $P(\psi) < \psi$ outside the SLC shows that the SLC is indeed stable from the outside.

Finally, for a stochastic system, $P(\psi)$ and $T(\psi)$ are no longer real numbers but are instead random variables. The distributions of $P(\psi)$ and $T(\psi)$, and how they change with ψ become of interest.

7.2 Numerical Implementation

Crossing times for L and L' can be computed numerically from a given signal by first keeping track of v_{eq} crossings for $v(t)$. In an e.g. Euler-Maruyama simulation, the signal will be an array of values $v[i]$, with i being the index. The indices i correspond to times t_i i.e. $v[i] \approx v(t_i)$. Then since $v(t)$ crosses L rightwards (recall that the trajectories floe counterclockwise in the phase plane), we check that $v[i] \leq v_{eq}$ and $v[i+1] \geq v_{eq}$. Finally, we check that the value of w is correct: $w[i] \leq w_{eq}$. When these criteria are satisfied, t_i is accepted as a crossing time. Likewise, $(v[i], w[i])$ is accepted as the associated crossing point. A similar case holds for L' except $v[i] \geq v_{eq}$, $v[i+1] \leq v_{eq}$, and $w[i] \geq w_{eq}$.

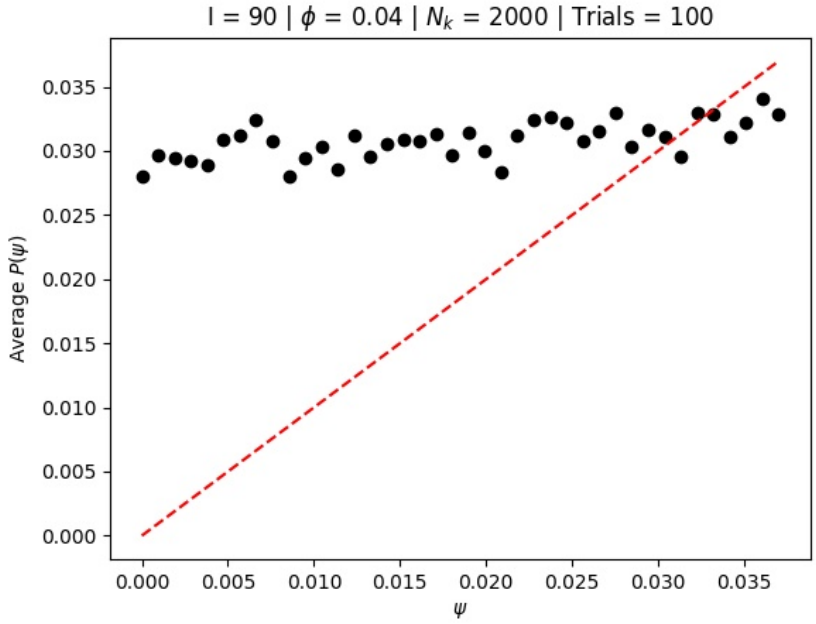
The `scripts.poincare` script uses this logic to compute $P(\psi)$ and $T(\psi)$ distributions, as well as how frequently a given oscillation spikes for a certain ψ.

7.3 Results

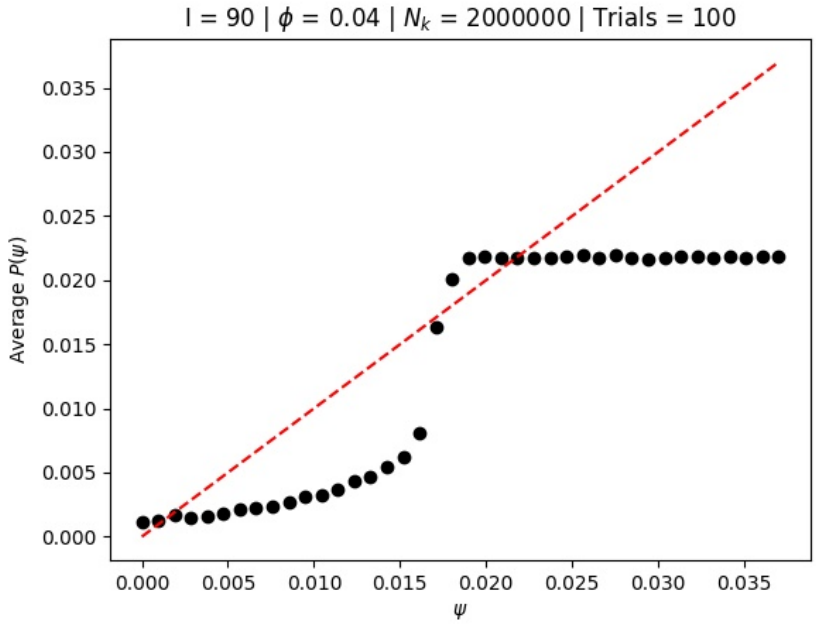
For noise at the $N_K = 2000$ level (recall that noise amplitude is proportional to $N_K^{-1/2}$), the distinct sigmoid shape of the Poincare map in **Figure 7.2** (we consider $\mathbb{E}[P(\psi)]$ for the stochastic system) disappears. Of course, there are levels of noise that are weak enough to retain the sigmoidal shape of $\mathbb{E}[P(\psi)]$ while still having an appreciable variance in $P(\psi)$. This latter quality allows for transitions between spiking and quiescent states, and gives rise to a distribution of ISIs. Plots of $\mathbb{E}[P(\psi)]$, $\mathbb{E}[T(\psi)]$, and spiking probabilities versus ψ for $N_K = 2000, 2 \times 10^6$ are given in **Figure 7.3**.

For weak noise (e.g. $N_K = 2 \times 10^6$), the $P(\psi)$ distributions is bimodal when ψ is near the ULC i.e. $\psi \approx 0.017$. If ψ is much more or less than this, the distribution becomes seemingly unimodal. A few of these simulated distributions are given in **Figure 7.4**.

One explanation of the unimodal/bimodal distributions is that the modes correspond to the stable states of the system i.e. the fixed point and SLC. So inside the ULC, the system will be attracted to the fixed point and therefore exhibit the distribution associated with the

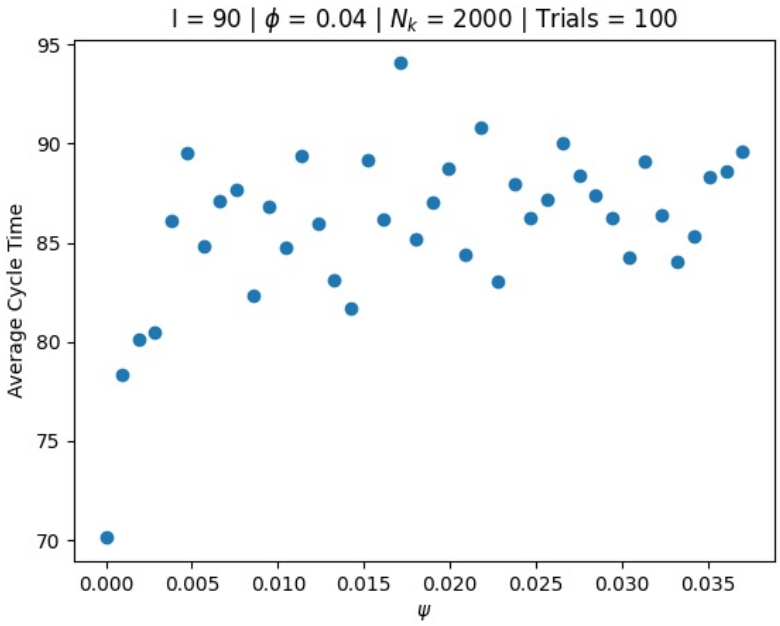


(a) $\mathbb{E}[P(\psi)]$ versus ψ for $N_K = 2000$

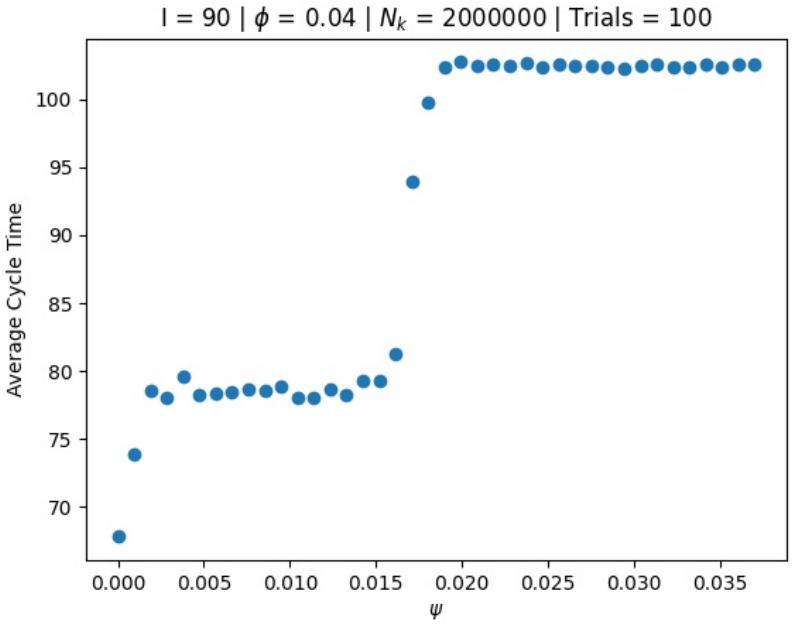


(b) $\mathbb{E}[P(\psi)]$ versus ψ for $N_K = 2000000$

Figure 7.3: *Plots of various Poincare statistics. Each of the 100 trials constituted running a single oscillation for every value of* ψ.

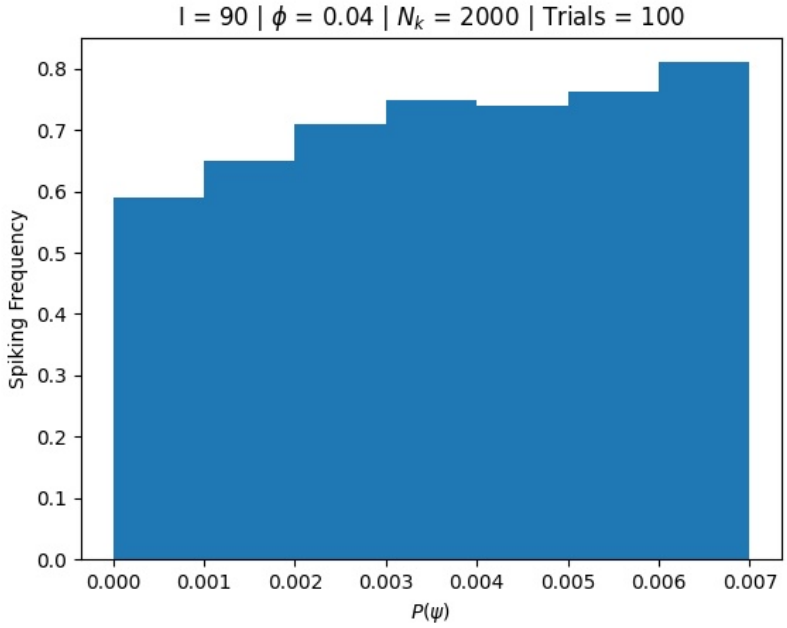


(c) $\mathbb{E}[T(\psi)]$ versus ψ for $N_K = 2000$

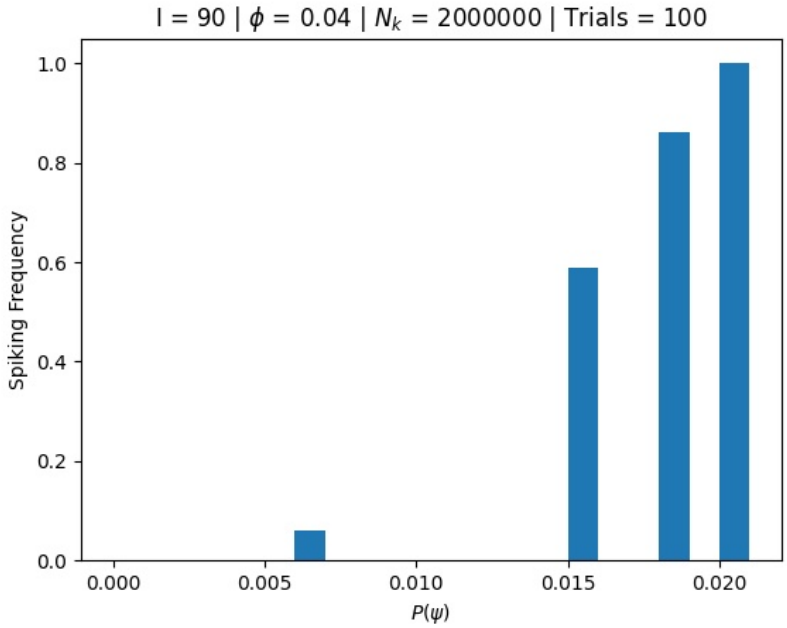


(d) $\mathbb{E}[T(\psi)]$ versus ψ for $N_K = 2000000$

Figure 7.3: *Plots of various Poincare statistics. Each of the 100 trials constituted running a single oscillation for every value of ψ. (cont.)*

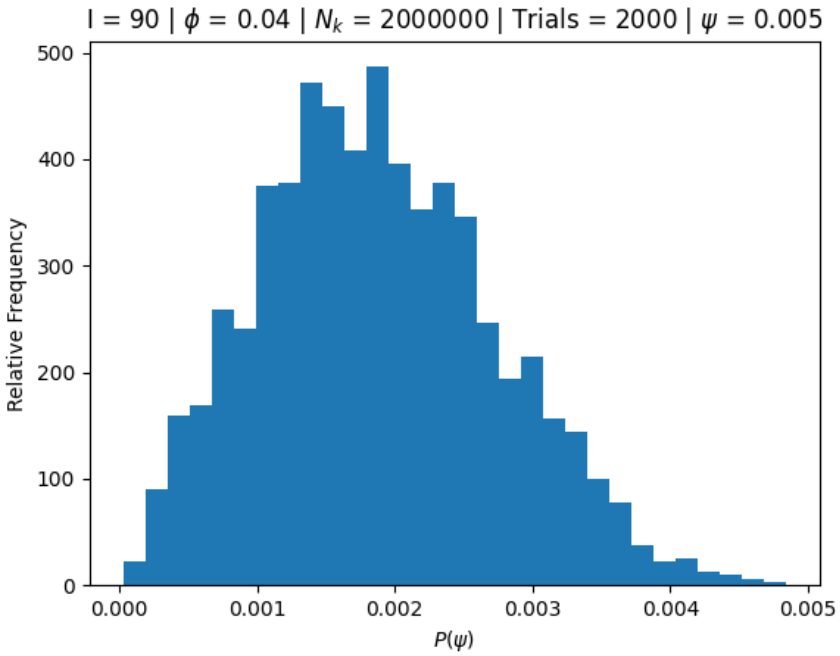


(e) Spiking frequency versus ψ for $N_K = 2000$

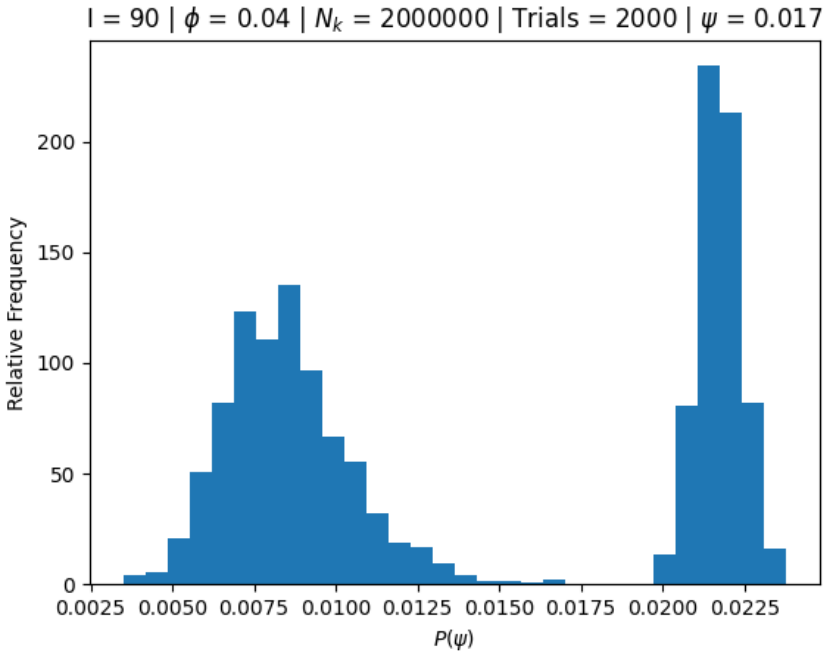


(f) Spiking frequency versus ψ for $N_K = 2000000$

Figure 7.3: *Plots of various Poincare statistics. Each of the 100 trials constituted running a single oscillation for every value of ψ. (cont.)*

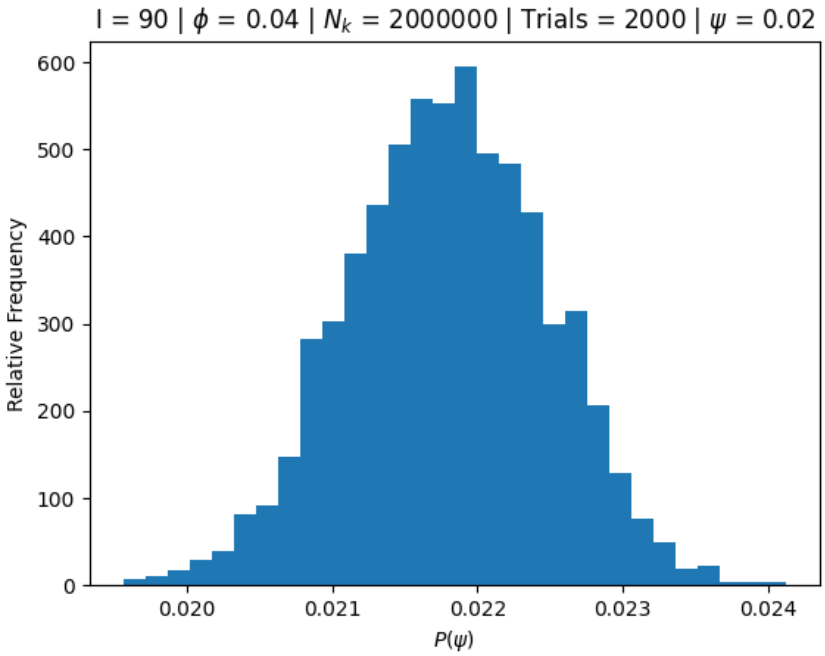


(a) $\psi = 0.005$



(b) $\psi = 0.017$

Figure 7.4: *Simulation of the distribution of $P(\psi)$ for various values of ψ.*



(c) $\psi = 0.02$

Figure 7.4: *Simulation of the distribution of $P(\psi)$ for various values of ψ. (cont.)*

fixed point (as in **Figure 7.4a**). Now, outside the ULC, the system is attracted to the SLC and thus produces the corresponding mode (as in **Figure 7.4c**). However, near the ULC, the stochastic noise can push the trajectory across to either side of the ULC. This results in a superposition of the fixed point and SLC modes in the histogram (as in **Figure 7.4b**).

The mode associated with the SLC is indeed maximal at approximately the value of ψ corresponding to the SLC (i.e. the intersection of the SLC and L). However, the mode associated with the fixed point is not maximal at $\psi = 0$, where the fixed point is located. One reason this discrepancy arises is that the SLC attraction is strong in the sense that deterministic trajectories typically converge near it within one oscillation. In contrast, trajectories will spiral towards the fixed point (at the speed of exponentially decay) as seen in **Figure 4.4b**. So if we naively assume the modes to be maximal at the deterministic $P(\psi)$, we would expect the SLC mode to exist as such and the fixed point mode to be offset from $\psi = 0$. Another reason for the offset is the one-sidedness of L i.e. that ψ is nonnegative. To illustrate, if a deterministic were to reach a small neighbourhood of the fixed point (for concreteness we make the neighbourhood diameter much smaller than the noise amplitude multiplied by the square root of the time scale i.e. the oscillation period 80 ms), Brownian motion in *any* direction will cause a deviation from the fixed point. If the system is perturbed in direction of decreasing w, $P(\psi)$ would increase as usual. However, if the system is instead disturbed in the direction of increasing w, the trajectory will be above the fixed point and effectively become a trajectory that is halfway through an oscillation;

the phase portrait will then circle around and hit L at a location below the fixed point.

The script used to generate the various Poincare statistics in **Figure 7.3** can be found in `scripts.poincare`. Variables of interest to change are `Nk`, `psi_range` (which defines the values of ψ to be probed) and `trials` (the number of trials to perform).

Meanwhile, the script used to generate the distributions of $P(\psi)$ for specific values of ψ is contained in `scripts.pdist`. This script additionally generates histograms for $T(\psi)$ and spiking frequency for fixed ψ. Some variables to change here include `Nk`, `psi`, and `trials`.

8 Alternative Dynamics

8.1 Jacobi Dynamics

Physically, w should be bounded in $[0,1]$. However, the equations described in (5.10)-(5.13) imply a finite (albeit small) possibility of w being outside of $[0,1]$. *In practice*, this issue never really manifests. For theoretical satisfaction, however, we may instead use–for a fixed v–a bounded diffusion process for w.

Following [4], we choose the simplest diffusion process that mimics the system. Arguably, we wish to preserve the dynamic part $g(v,w)$ while having as simple of a variance coefficient $h(v,w)^2$ as possible. The preservation of $g(v,w)$ can be vouched for as it, loosely speaking, determines the "average" behaviour of the system, as well as defining the $N_K \to 0$ limit behaviour. Seeing that $g(v,w)$ is linear in w, we consider Pearson diffusions, where $h(v,w)^2$ is a polynomial of degree at most two. It turns out that the Jacobi diffusion is the only Pearson diffusion bounded in a finite interval. This renders a Jacobi diffusion as the "simplest" bounded diffusion process appropriate for the stochastic Morris-Lecar system.

We follow the discussion [4] on the Jacobi diffusion here. The Jacobi diffusion bounded on $(0,1)$ has the form (for fixed v)

$$dw = -\theta(w-\mu)dt + \gamma\sqrt{2\theta w(1-w)}dB_t \tag{8.1}$$

Equating the dynamic part gets us

$$\theta = \alpha + \beta, \quad \mu = \frac{\alpha}{\alpha+\beta} \tag{8.2}$$

Note that we necessarily have $\alpha(v), \beta(v) \geq 0$ as can be seen from (3.6)-(3.7). The Jacobi diffusion is *ergodic* for

$$\gamma^2 \leq \mu, 1-\mu \tag{8.3}$$

To retain this property, we can rewrite $\gamma = \sigma^*\delta(v)$, where σ^* is a constant that can be fit (to e.g. match $h(v,w)$ at the fixed point) and $\delta(v)$ is a simple function of v. Of course, such a choice of $\sigma^*, \delta(v)$ is not unique since we can rewrite $\sigma^* \mapsto x\sigma^*$, $\delta(v) \mapsto \frac{\delta(v)}{x}$ for any nonzero x. We therefore assert the condition that $\sigma^* \in [0,1]$ and that (8.3) holds for all σ^* in that range. When fitting σ^*, the ergodicity requirement becomes verifying that $\sigma^* \in [0,1]$.

γ^2 is largest when $\sigma^* = 1$, so our task becomes assuring that $\delta(v)^2 \leq \mu, 1-\mu$ i.e.

$$\delta(v)^2 \leq \frac{\alpha(v)}{\alpha(v)+\beta(v)}, \frac{\beta(v)}{\alpha(v)+\beta(v)} \tag{8.4}$$

Naively, we could define

$$\delta(v) = \sqrt{\min\left(\frac{\alpha(v)}{\alpha(v)+\beta(v)}, \frac{\beta(v)}{\alpha(v)+\beta(v)}\right)} \tag{8.5}$$

However, the piecewise definition comes at the cost of differentiability. To make $\delta(v)$ analytically "well-behaved", we instead define $\delta(v)^2$ to be the harmonic sum of $\mu, 1-\mu$ (as is done for e.g. parallel resistors):

$$\delta(v)^2 = \frac{\mu(1-\mu)}{\mu + (1-\mu)} = \frac{\alpha\beta}{\alpha+\beta} \tag{8.6}$$

Thus

$$\gamma = \sigma^* \sqrt{\frac{\alpha\beta}{\alpha+\beta}} \tag{8.7}$$

Equation (8.1) now becomes

$$dw = (\alpha(v)(1-w) - \beta(v)w)dt + \sigma^* \sqrt{2\frac{\alpha(v)\beta(v)}{\alpha(v)+\beta(v)}w(1-w)}dB_t \tag{8.8}$$

The fixed point of the system occurs at approximately $(v_{eq}, w_{eq}) = (-26.6, 0.129)$. Plugging this into (5.13) and the coefficient in front of dB_t in (8.8) gives

$$h(v_{eq}, w_{eq}) = \frac{0.1003}{\sqrt{N_K}} = 0.03365\sigma^* \tag{8.9}$$

That gives us

$$\sigma^* = \frac{2.98}{\sqrt{N_K}} \tag{8.10}$$

Since N_K is on the order of 1000, and certainly larger than 9, we see that typical fits of σ^* are indeed ergodic.

8.2 Linearized Dynamics

Near the fixed point, we can expand (5.10) as a linear model as in [4]. Since the deterministic part is zero to zeroth order, we expand it to first order. In contrast, the stochastic part $h(v, w)$ is nonzero to zeroth order, so we need only expand that to zeroth order. That gets us

$$\begin{pmatrix} v \\ w \end{pmatrix}' \approx \mathbf{M} \begin{pmatrix} v \\ w \end{pmatrix} dt + \begin{pmatrix} 0 \\ \frac{0.1003}{\sqrt{N_K}} dB_t \end{pmatrix} \tag{8.11}$$

Here we used the value of (8.9) for the stochastic part. Meanwhile, $\mathbf{M}$ is the Jacobian matrix of the dynamic part in (5.10):

$$\mathbf{M} = D \begin{pmatrix} f(v,w) \\ g(v,w) \end{pmatrix} \tag{8.12}$$

The linear approximation is valid in a small region near the fixed point. Of course, this region is necessarily within the unstable limit cycle. Numerical experiemnts can provide more clarity into the size of this region. Such a region would be elliptical, however. This is because the coordinates v, w are not symmetrical in the sense that $\mathbf{M}$ does not stretch and shear them uniformly. To remove this complication and simplify the equation, we can define the matrix

$$\mathbf{Q} = \begin{pmatrix} \omega & m_{11} + \lambda \\ 0 & m_{21} \end{pmatrix} \tag{8.13}$$

Then we define

$$\mathbf{A} = \mathbf{Q}^{-1}\mathbf{M}\mathbf{Q} = \begin{pmatrix} -\lambda & \omega \\ -\omega & -\lambda \end{pmatrix} \tag{8.14}$$

$$\tilde{\mathbf{x}} = \mathbf{Q}^{-1} \begin{pmatrix} v \\ w \end{pmatrix} \tag{8.15}$$

$$\mathbf{c} = \mathbf{Q}^{-1} \begin{pmatrix} 0 \\ \frac{0.1003}{\sqrt{N_K}} dB_t \end{pmatrix} \tag{8.16}$$

We then obtain the symmetrized equation

$$d\tilde{\mathbf{x}} = \mathbf{A}\tilde{\mathbf{x}} + \mathbf{c} \tag{8.17}$$

In the transformed space $\tilde{\mathbf{x}}$, it makes sense to talk about circular regions as opposed to elliptical regions in the original coordinate system. Finally, we can further simplify this equation by factoring out rotations. If we define

$$\tilde{\tilde{\mathbf{x}}} = R_{\omega t}\tilde{\mathbf{x}} = \begin{pmatrix} \cos\omega t & -\sin\omega t \\ \sin\omega t & \cos\omega t \end{pmatrix} \tilde{\mathbf{x}} \tag{8.18}$$

Then we obtain the equation

$$d\tilde{\tilde{\mathbf{x}}} = -\lambda\tilde{\tilde{\mathbf{x}}} + R_{\omega t}\mathbf{c} \tag{8.19}$$

Finally, it helps to define

$$\mathbf{b} = \mathbf{Q}^{-1} \begin{pmatrix} 0 \\ \frac{0.1003}{\sqrt{N_K}} \end{pmatrix} \;\therefore\; \mathbf{c} = \mathbf{b}\, dB_t \tag{8.20}$$

Now we can define a one-parameter model near the fixed point as follows. If we first define (actually, that $|\mathbf{b}|$ is the trace of the covariance matrix):

$$\tau = \frac{|\mathbf{b}|}{\sqrt{2}} \tag{8.21}$$

$$\mathbf{U}_t = \frac{\sqrt{\lambda}}{\tau}\tilde{\tilde{\mathbf{x}}}_{t/\lambda} \tag{8.22}$$

Then the $\mathbf{U}_t$ process is scaled to have unit decay rate:

$$d\mathbf{U}_t = -\mathbf{U}_t dt + \frac{1}{\tau} R_{\omega t/\lambda}\mathbf{c} \tag{8.23}$$

According to [2], the $\mathbf{U}_t$ process converges to the simpler process $\mathbf{S}_t$ (where $\mathbf{B}_t$ denotes a standard 2-dimensional Brownian motion)

$$d\mathbf{S}_t = -\mathbf{S}_t dt + \mathbf{dB_t} \tag{8.24}$$

when $\lambda/\omega \to 0$. The hypothesis to this claim is reasonable to assume, seeing that λ is an order of magnitude less than ω for the $I = 90, \phi = 0.04$ neuron: $\lambda = 0.00860, \omega = 0.0801$. So the stochastic system is indeed approximately a unit Ornstein–Uhlenbeck process *after an appropriate coordiantes and time transformation* near the fixed point. We can subsequently treat the 2-norm of $\mathbf{S}_t$ (recall that $\mathbf{S}_t$ is a 2-dimensional stochastic process) as a one-parameter system thanks to the isotropy of (8.24).

$$R_t = |\mathbf{S}_t| \tag{8.25}$$

It can be shown with Itô calculus that this "R" model obeys the equation

$$dR = \left(\frac{1}{2R} - R\right) dt + dB_t \tag{8.26}$$

where B_t is a standard one-dimensional Brownian motion.

8.3 Numerical Comparison

A comparison of the Euler-Maruyama, Jacobi, and linear models for a deterministic Morris-Lecar neuron is given in **Figure 8.1**. Since the Euler-Maruyama and Jacobi models differ only in the noise term, the two have the same deterministic dynamics. Otherwise, the linear and nonlinear models appear to exhibit similar behaviour (at least, qualitatively) near the fixed point for the trajectories in **Figure 8.1**. A comparison of these models with a stochastic neuron is given later in this report.

The `mlj`, `mll`, and `mlr` neurons (in the `neurons` subdirectory) correspond to the Jacobi, linear, and "R" Morris-Lecar models, respectively. These can be instantiated through the constructor or from an existing `ml` neuron with `ml.gen_model`. After that, the `init` method must be called on these neurons before they are used.

8.4 Helpful Resources

The article [4] discusses the Jacobi, linear, and "R" models as well.

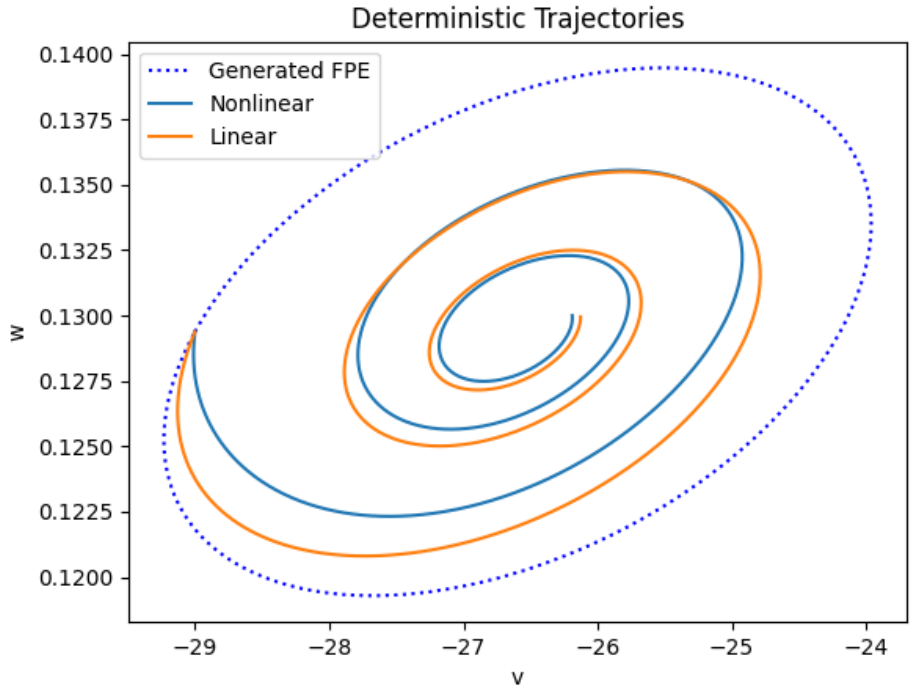


Figure 8.1: *Comparison of quisecent deterministic trajectories for nonlinear (Euler-Maruyama and Jacobi) and linear models of the Morris-Lecar neuron. The fixed point ellipse (FPE) is defined later in the report.*

9 Patched Model

9.1 Description

D

9.2 Motivation

S

9.3 Current Results

T

10 Next Steps

References

[1] S. M. BAER and T. ERNEUX. Singular hopf bifurcation to relaxation oscillations. *SIAM journal on applied mathematics*, 46(5):721–739, 1986.

[2] Peter H. Baxendale and Priscilla E. Greenwood. Sustained oscillations for density dependent markov processes. *Journal of mathematical biology*, 63(3):433–457, 2011.

[3] A. Dhooge, W. Govaerts, and Yu Kuznetsov. Matcont: A matlab package for numerical bifurcation analysis of odes. *ACM transactions on mathematical software*, 29(2):141–164, 2003.

[4] Susanne Ditlevsen and Priscilla Greenwood. The morris–lecar neuron model embeds a leaky integrate-and-fire model. *Journal of mathematical biology*, 67(2):239–259, 2013;2011;.

[5] Priscilla E. Greenwood, Lawrence M. Ward, SpringerLink (Online service), SpringerLINK ebooks Mathematics, and Statistics. *Stochastic Neuron Models*, volume 1.5. Springer International Publishing, Cham, 1st 2016. edition, 2016.

[6] Yuri A. Kuznetsov, SpringerLINK eBooks English/International Collection (Archive), and SpringerLink (Online service). *Elements of Applied Bifurcation Theory*, volume 112. Springer New York, New York, NY, third;third; edition, 2004;2013;.

[7] Gregory F Lawler. Stochastic calculus: An introduction with applications. *American Mathematical Society*, 2010.

[8] Hil Meijer. Matcont tutorial: ode gui version, 2016.

[9] C. Morris and H. Lecar. Voltage oscillations in the barnacle giant muscle fiber. *Biophysical journal*, 35(1):193–213, 1981.

[10] Peter F. Rowat and Priscilla E. Greenwood. Identification and continuity of the distributions of burst-length and interspike intervals in the stochastic morris-lecar neuron. *Neural computation*, 23(12):3094–3124, 2011.

[11] Steven H Strogatz. *Nonlinear dynamics and chaos with student solutions manual: With applications to physics, biology, chemistry, and engineering.* CRC press, 2018.

[12] David H. Terman and G. B. Ermentrout. *Foundations of mathematical neuroscience*, volume 35. Springer, 2010.