Path Integration for Ants

May 29th, 2024

Max Brodeur
SCIPER: 368410

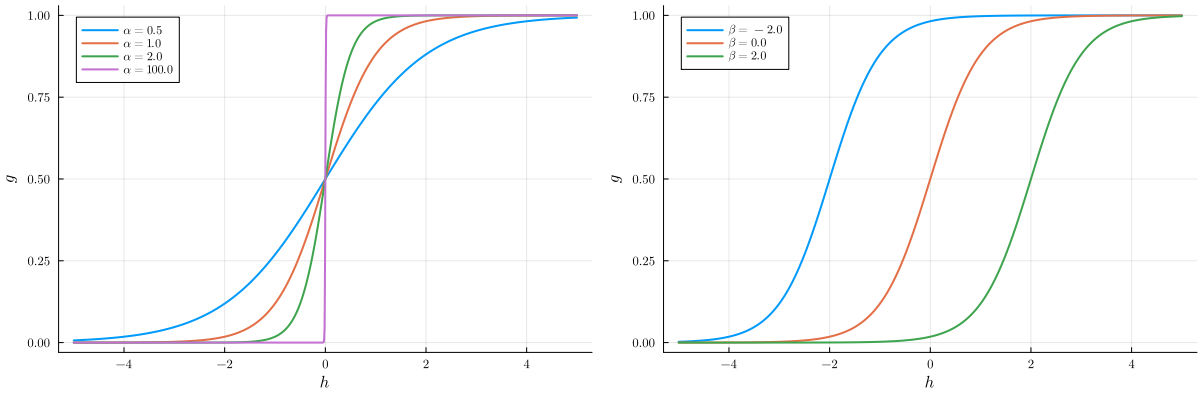
Mathilde Simoni
SCIPER: 371423

Introduction

In this project, we aim to implement a circuit capable of integrating the trajectory of an ant in two dimensions. To achieve this goal, we use a network of interconnected bump attractors and make use of their stability to encode the position of the ant throughout time. The head direction of agent is encoded with a single bump attractor which is given as external input the generated ant trajectory. The x and y positions are then encoded with two coupled bump attractors that update the x and y locations respectively at every time-step based on head direction.

Ex. 0 Getting Started: Poisson Neurons

0.1



Effect of α, $\beta = 0$ Effect of β, $\alpha = 1.0$

Figure 1: Sigmoid g vs. Potential h

The first observation is that as $\alpha \to \infty$, the transfer function $g(h)$ converges to the step function, resulting in a more pronounced threshold effect.

In addition, β translates $g(h)$ horizontally. This means that as β increases, the neurons need a higher potential h to fire. Conversely, a lower β causes the neurons to fire at lower potential values.

0.2

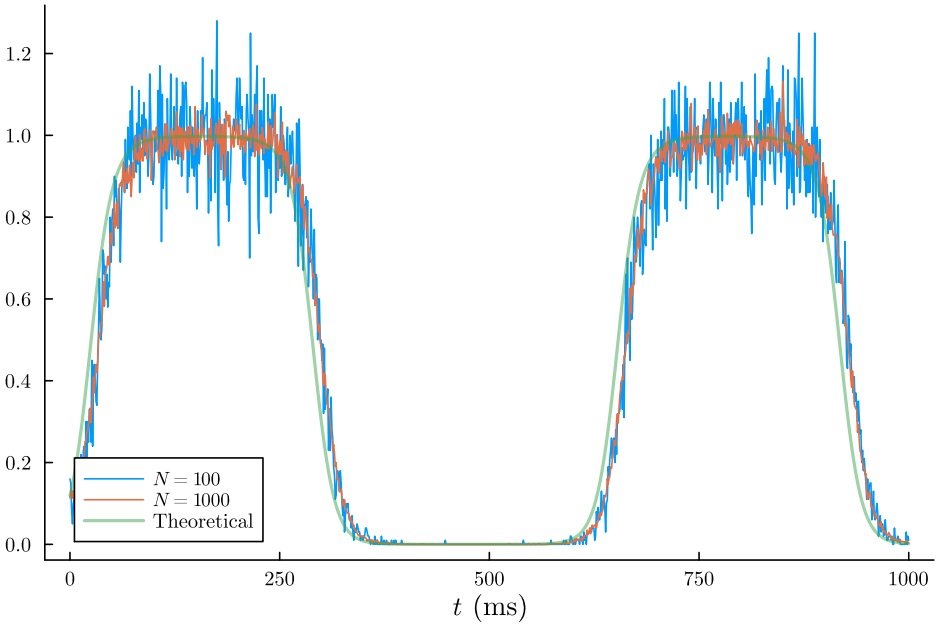


Figure 2: Population firing rate for ranging population sizes and theoretical rate neurons.

The simulation with $N = 100$ neurons shows noise when compared to the theoretical firing rate. This noise arises from the to the stochastic nature of the Poisson process used to generate the spikes. However, this noise doesn't appear at the bottom of the wave since sampling from a zero probability always gives 0.

Increasing the number of neurons to $N = 1000$ reduces the noise due to the Law of Large Numbers. As the number of neurons increases, the mean number of spikes per ms converges to the expected value, which is here the theoretical rate.

Ex. 1 Bump Attractor

1.1

Figure 3 and Figure 4 show a raster plot fo the activity for different values of J. For $J = 0$, the connection strength is too weak to produce a bump. When $J = 4.6$, a bump occasionally forms but lacks consistency. However, increasing the interaction strength to $J \geq 5$ consistently results in a converging bump.

The bump formation can be explained by the weight $w(x_i, x_j) = \cos(x_i - x_j)$ used in the simulation. A neuron at location x_i is mostly influenced by neighboring neurons with small $|x_i - x_j|$ because the cosine weight is the highest for small differences. Conversely, far away neurons at a distance π exert a negative influence due to the negative cosine weight.

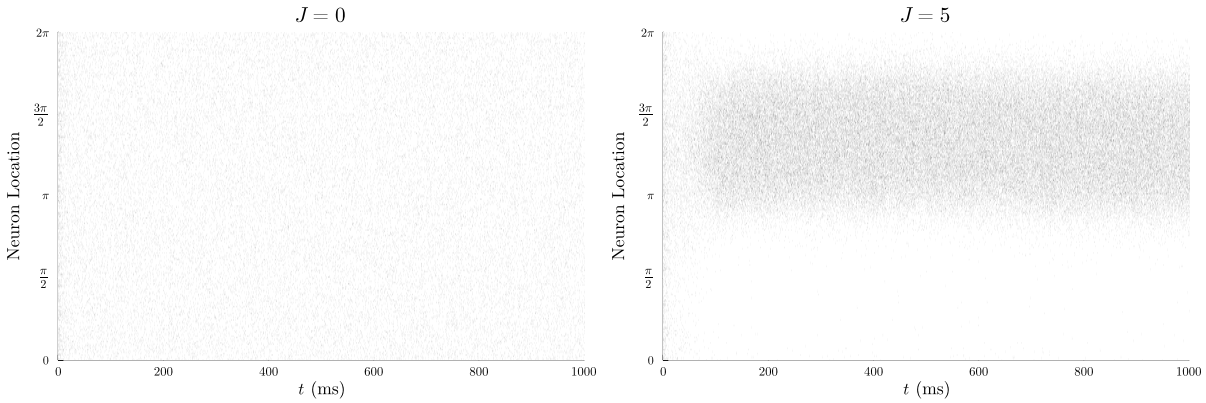


Figure 3: Stable vs. Unstable values of J

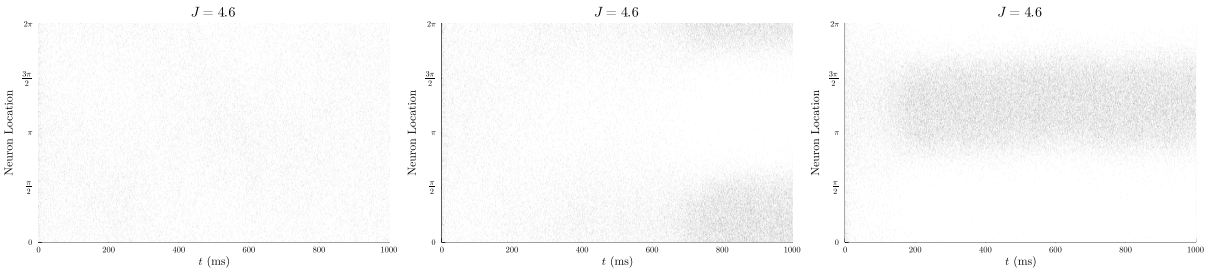


Figure 4: $J = 4.6$ does not consistently produce stable bumps.

1.2

In order to locate the find the location of the bump θ_{bump} over time, we use a weighted version of the circular mean (wikipedia). The weights associated to each neuron are 1 if it produces a spike, 0 otherwise. We also average the location of the bump over bins of 10ms. This reduces the noise due to the stochastic nature of the simulation and allows to precisely locate the center of the bump. Figure 5 shows the bump location on top of the activity. Initially, it exhibits noise, but stabilizes as the bump converges.

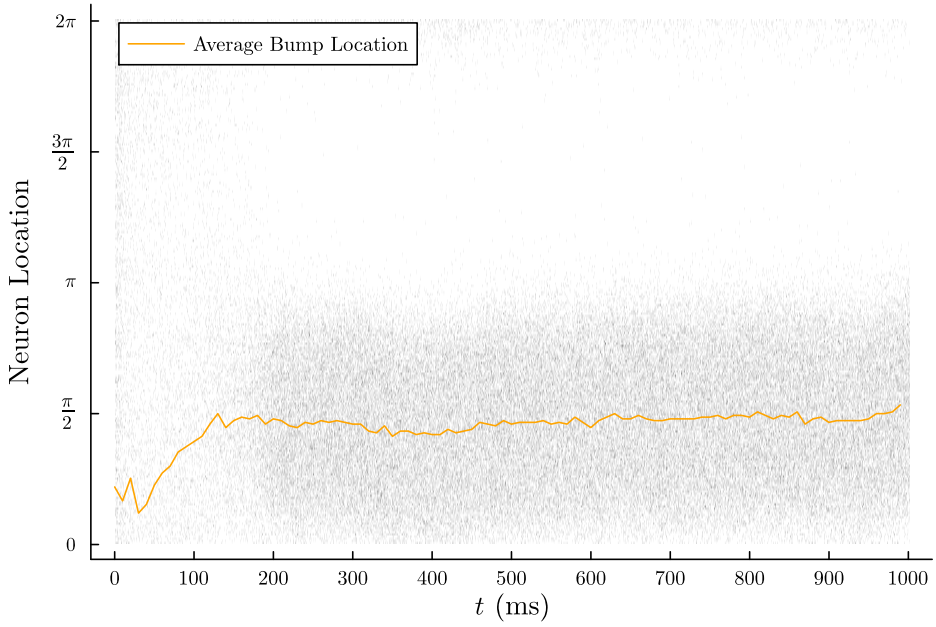


Figure 5: Average Location of the Bump

If we simulate the network for a larger timescale (2000 ms), a *drift* can appear.

1.3

In order to increase the stability of the bump throughout time, we first notice that increasing the value of the characteristic timescale τ reduces the drift (Figure 6). This observation is confirmed by the authors of [1] which states that "the bump's dispersion rate [...] depends on the neuronal time constant and decreases as the time constant decreases" because it is less subject to the noise causing the *drift*.

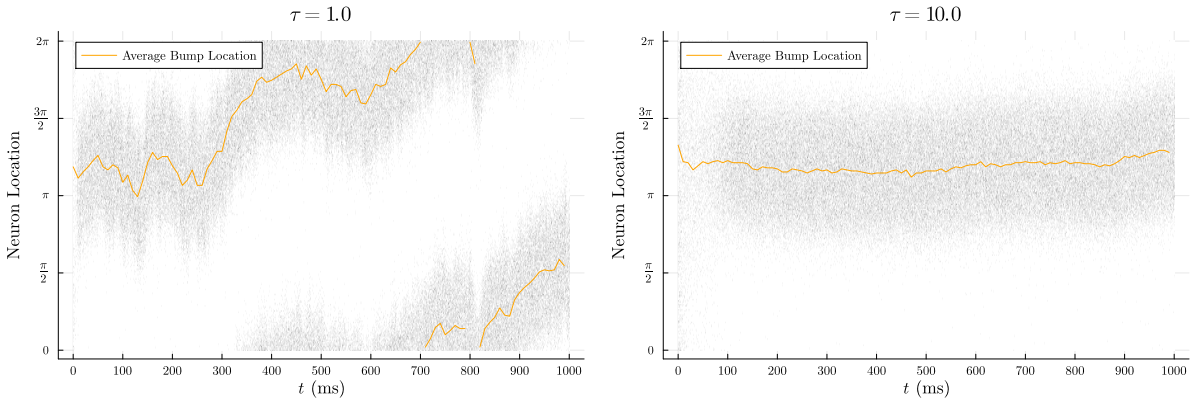


Figure 6: Effect of the time scale on bump location drift.

Additionally, we notice that the stability of the bump increases with the number of neurons in the network, due to the Law of Large Numbers For instance, Figure 7 shows the result of two simulations with 100, 300, 500 and 1000 neurons.

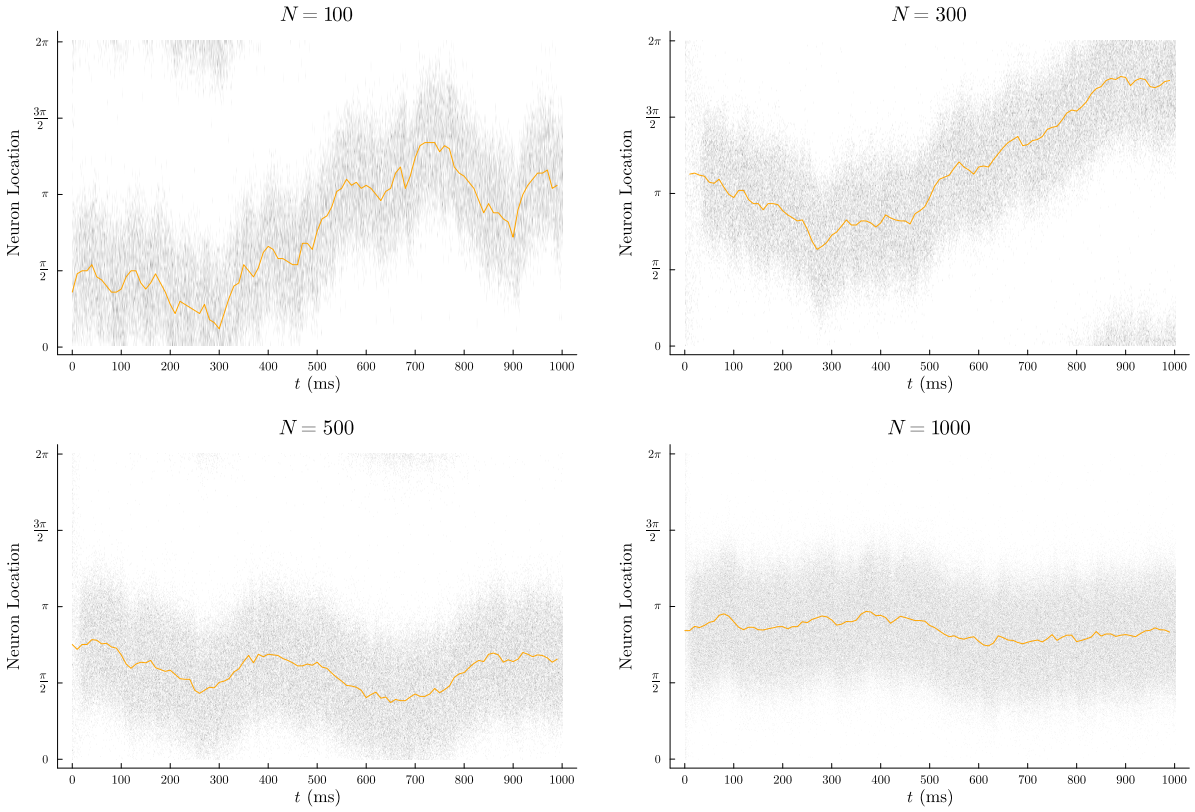


Figure 7: Stability of the Bump for Different Network Sizes

1.4

Figure 8 shows the external current as a function of the x_i (neuron location) for different times

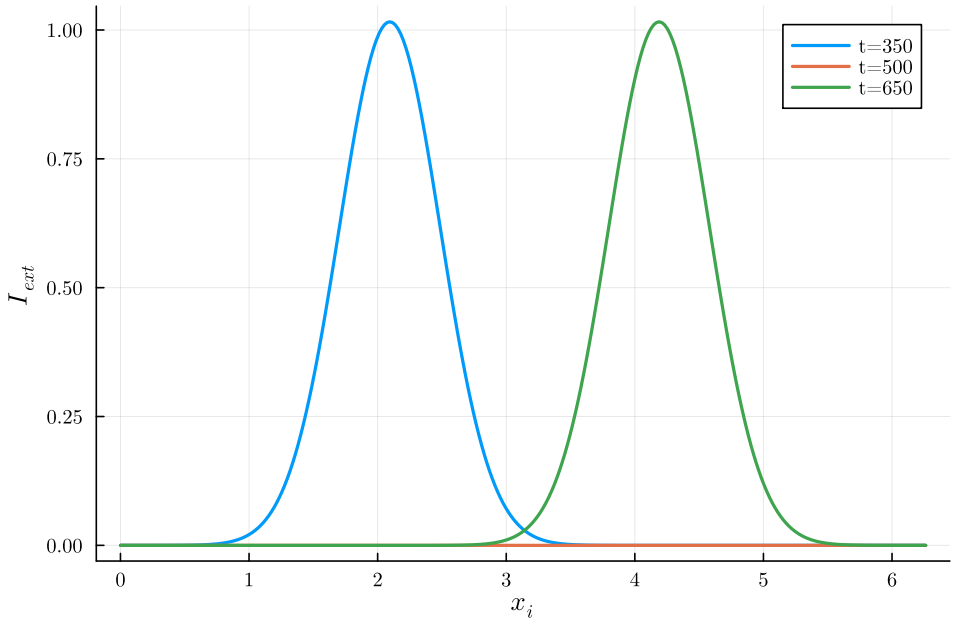


Figure 8: External Current at Different Times

Simulating the network with the newly added external input gives the following network activity:

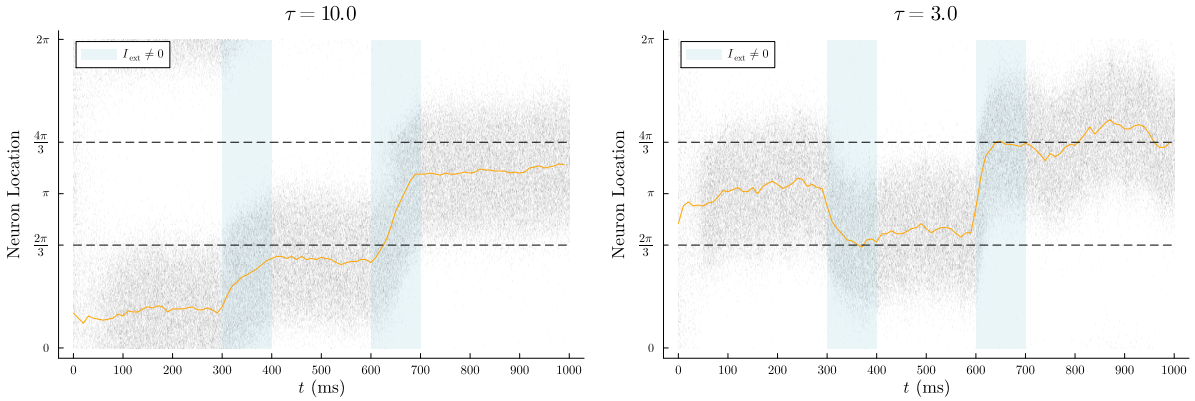


Figure 9: Raster Plot of the Network Activity

We observe that the bump location moves to the peak of the external input added to the network ($\frac{2\pi}{3}$ first and then $\frac{4\pi}{3}$). In addition, the change in bump location is faster when the characteristic timescale τ is smaller. This makes sense as a smaller τ is associated with a faster convergence of the neuron potential to an external input in the initial ODE describing the neuronal dynamics.

1.5

By modifying the weight as indicated in the instructions, the peak of the cosine weight function is shifted. Hence, we expect the bump to move throughout time. The weight function becomes:

$$\begin{aligned} w\big(x_i - \varphi, x_j\big) &= \cos\big((x_i - \varphi) - x_j\big) \\ &= \cos(x_i - \varphi)\cos\big(x_j\big) + \sin(x_i - \varphi)\sin\big(x_j\big) \end{aligned}$$

Therefore, the input to a neuron can be updated as follows:

$$I_i(t) = J(\cos(x_i - \varphi)m_{\cos}(t) + \sin(x_i - \varphi)m_{\sin}(t))$$

with $m_{\cos}(t)$ and $m_{\sin}(t))$ the collective variables previously defined.

Figure 10 shows the results of applying this new weight function for different values of input angle φ. It confirms our initial intuition about the change of the bump location. In addition, we observe that a higher φ produces faster changes of the bump location and that the sign of φ dictates the direction of the drift.

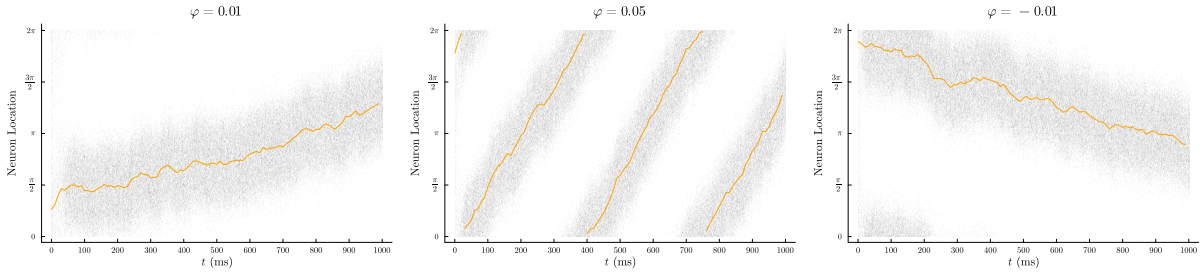


Figure 10: Raster plots of the network activity with varying input angle.

Ex. 2 Integrator

2.1

The input received for each neuron in the left population and the right population is as follows:

$$\begin{aligned} I_i^L(t) &= \frac{J}{N}\sum_{j=1}^{N} w_{L\to L}\big(x_i^L, x_j^L\big)S_j(t) + \frac{J}{N}\sum_{j=1}^{N} w_{R\to L}\big(x_i^L, x_j^R\big)S_j(t) \\ &= \frac{J}{N}\sum_{j=1}^{N} \cos\big(x_i^L + \theta - x_j^L\big)S_j(t) + \frac{J}{N}\sum_{j=1}^{N} \cos\big(x_i^L + \theta - x_j^R\big)S_j(t) \\ &= J\big(\cos\big(x_i^L + \theta\big)\big(m_{\cos}^L(t) + m_{\cos}^R(t)\big) + \sin\big(x_i^L + \theta\big)\big(m_{\sin}^L(t) + m_{\sin}^R(t)\big)\big) \end{aligned}$$

$$\begin{aligned} I_i^R(t) &= \frac{J}{N}\sum_{j=1}^{N} w_{R\to R}\big(x_i^R, x_j^R\big)S_j(t) + \frac{J}{N}\sum_{j=1}^{N} w_{L\to R}\big(x_i^R, x_j^L\big)S_j(t) \\ &= \frac{J}{N}\sum_{j=1}^{N} \cos\big(x_i^R - \theta - x_j^R\big)S_j(t) + \frac{J}{N}\sum_{j=1}^{N} \cos\big(x_i^R - \theta - x_j^L\big)S_j(t) \\ &= J\big(\cos\big(x_i^R - \theta\big)\big(m_{\cos}^L(t) + m_{\cos}^R(t)\big) + \sin\big(x_i^R - \theta\big)\big(m_{\sin}^L(t) + m_{\sin}^R(t)\big)\big) \end{aligned}$$

with the 4 collective variables defined as:

$$m_{\cos}^L(t) = \sum_{j=1}^{N} \cos\big(x_j^L\big)S_j(t) \qquad m_{\cos}^R(t) = \sum_{j=1}^{N} \cos\big(x_j^R\big)S_j(t)$$

$$m_{\sin}^L(t) = \sum_{j=1}^{N} \sin\big(x_j^L\big)S_j(t) \qquad m_{\sin}^R(t) = \sum_{j=1}^{N} \sin\big(x_j^R\big)S_j(t)$$

Why push pull?

This connectivity can be considered a "push-pull system" because out of its four weight equations, two of them make the bumps move away from each other (push) and two of them make the bumps move closer to each other (pull). Specifically, the $w_{L\to L}, w_{R\to R}$ equations have the effect of moving the bumps both down and up respectively, by the same reasoning for the results shown in Figure 10. Conversely, the $w_{L\to R}$ and $w_{R\to L}$ equations have the effect of activating the neurons of the opposing bumps that are in the other direction of the movement induced by the "push" equations. This explains the "pulling" behavior of these equations.

2.2

Even though the interaction strength $J = 3$ (which is smaller than the minimal interaction weight for a single bump attractor), we observe in Figure 11 that the two bumps are stable over time. This shows the the coupled bump attractor network produces a bump with higher stability than the single bump attractor.

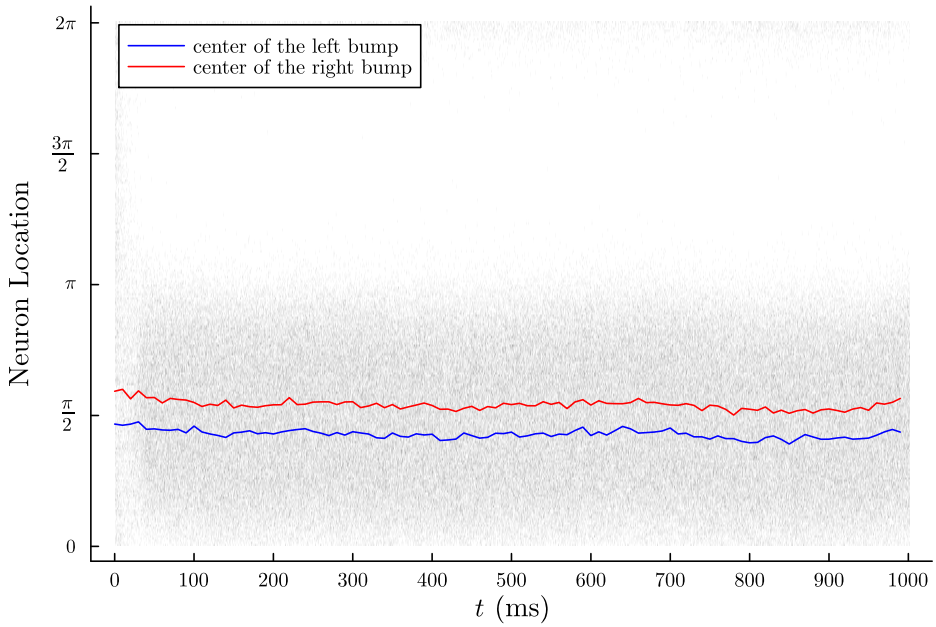


Figure 11: Location of the bumps throughout time

2.3

In question 1.4, we observed that giving an external input to a bump attractor network moves the location of the bump to the external input's peak value. Therefore, we first run the simulation for 500 time-steps with an external input $I_{\text{ext, init}}$ sampled from the Normal distribution centered at π. This way, the mean location of the two bumps moves to $\theta_{\text{bump mean}} \approx \pi$ before the start of the simulation. This method also reduces the initial noise that was observed in the previous plots:

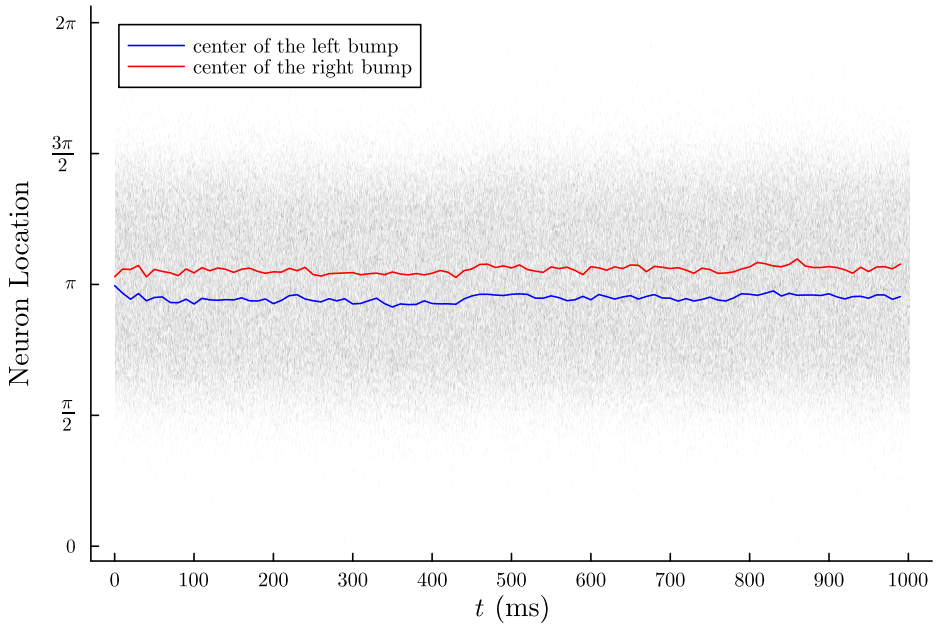


Figure 12: Location of the Bumps Throughout Time

2.4

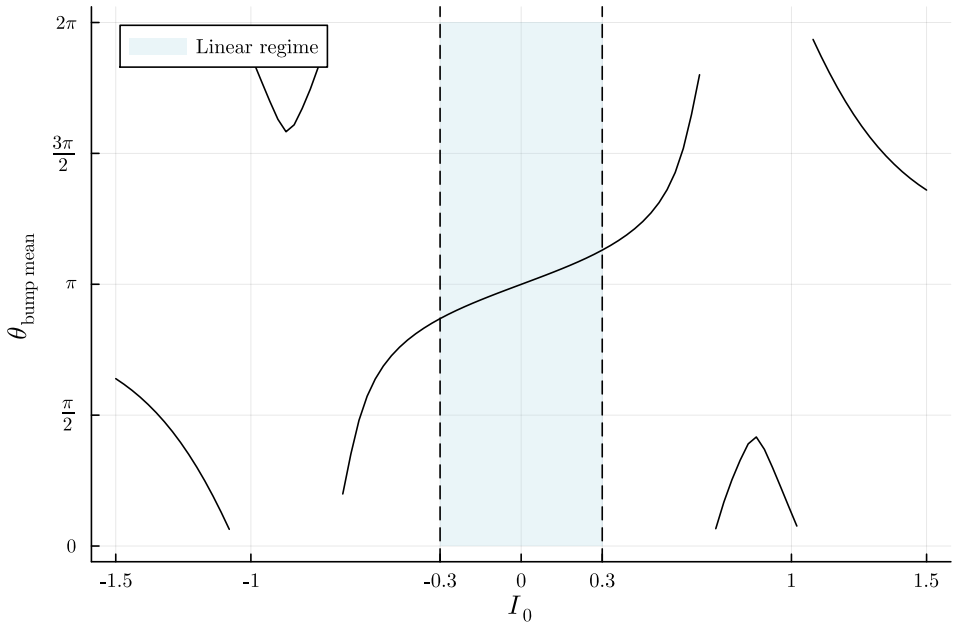


Figure 13: Final Bump Location Versus Input Strength

We observe that the final location of the bump has a linear relationship with the input strength when $|I_0| < 0.3$.

2.5

Unlike with the single bump attractor, the convergence value of each bump depends on the state of the other network. This creates a relationship that regulates each network's bump location and provides a mechanism to achieve an equilibrium state of both networks. This equilibrium state does not depend on the actual location of both networks, but of their *relative* locations to one another.

Provided a way to control the initialized location of the bumps as discussed in 2.3, and a valid mapping between positions and angles, we can use this coupled bump attractor network to perform integration

given input head angles. We can use the head angle θ at a certain time step and extract an x step value by taking $\cos(\theta)$. By assuming a constant speed, adding up these x steps in time corresponds to integration of the x coordinate.

The last step is to have the coupled bump attractor network "record" the current x coordinate using its equilibrium state. To achieve this, we provide the cosine value as input to both populations in such a way that the left bump attractor "pulls harder" when the value is negative (and so the x position is moving left), and vice-versa with the right attractor. This is exactly the behavior using the term described in 3.3.

Ex. 3 Path Integration

3.1

To generate smooth random trajectories, we generate the head direction values by means of a Wiener process, in which the head direction is sampled from a normal distribution centered at the last head direction step and with parametric standard deviation:

$$\theta(t_{n+1}) \leftarrow \mathcal{N}\left[\theta(t_n), \sigma\right]$$

where σ is a parameter representing the volatility of the head direction movement.

3.2

We test the head direction bump attractor for two different volatility values $\sigma = 0.01$ and $\sigma = 0.1$. In both cases, the bump location θ^H_{bump} matches well the current head direction θ^H_{input}, with a small delay due to the timescale τ.

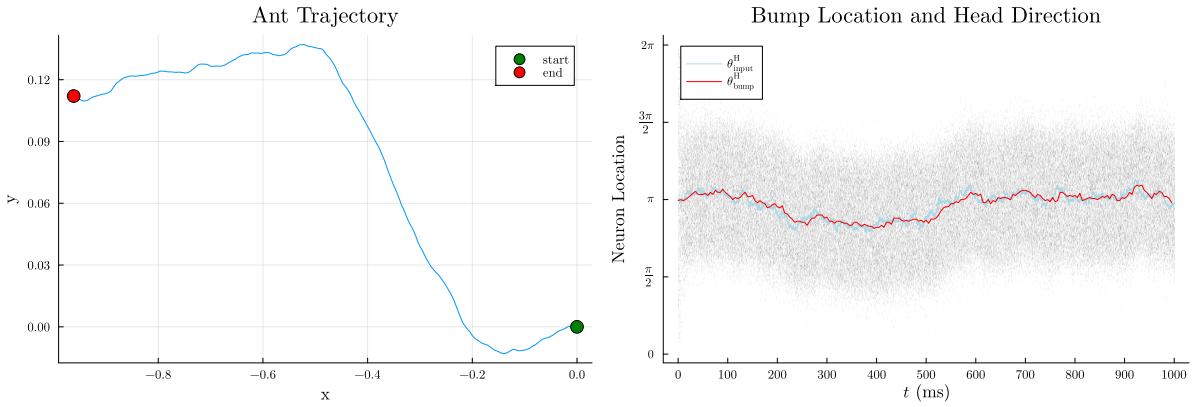


Figure 14: $\sigma = 0.01$

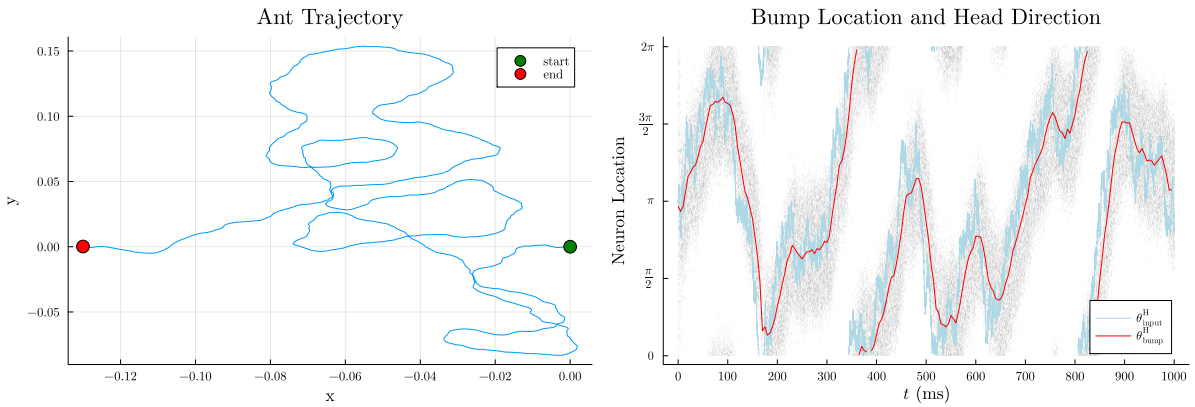


Figure 15: $\sigma = 0.08$

3.3

The input coming from the head network takes the following form

$$I_{\text{head}}^{L/R}(t) = \mp \frac{J_{\text{head}}}{N} \sum_{j=1}^{N} \cos\left(x_j^H\right) S_j^H(t)$$

We previously determined that to stay in a linear regime, the bump attractors must receive input within the range $I_{\text{head}} \in [-0.3, 0.3]$. Therefore, we must determine the values of J_{head} such that the input stays within that range. After simulating the head network over a variety of trajectories, and by setting $J_{\text{head}} = 1$, we observe that $I_{\text{head}} \in [-0.6, 0.6]$ as displayed in Figure 16. Therefore, using a value of $J_{\text{head}} = 0.5$ is a suitable choice to obtain values within our desired range. In fact, the value of J_{head} is inversly proportional to that of Δt. Indeed, the scaling of the spike train values scales inversely to Δt, and so for a smaller value, we need an equally larger one for J_{head}. In fact, the stable value is probably $J_{\text{head}} = 5\Delta t$ where Δt is the step size used for the head direction network, and thus the inverse of the spike train magnitudes.

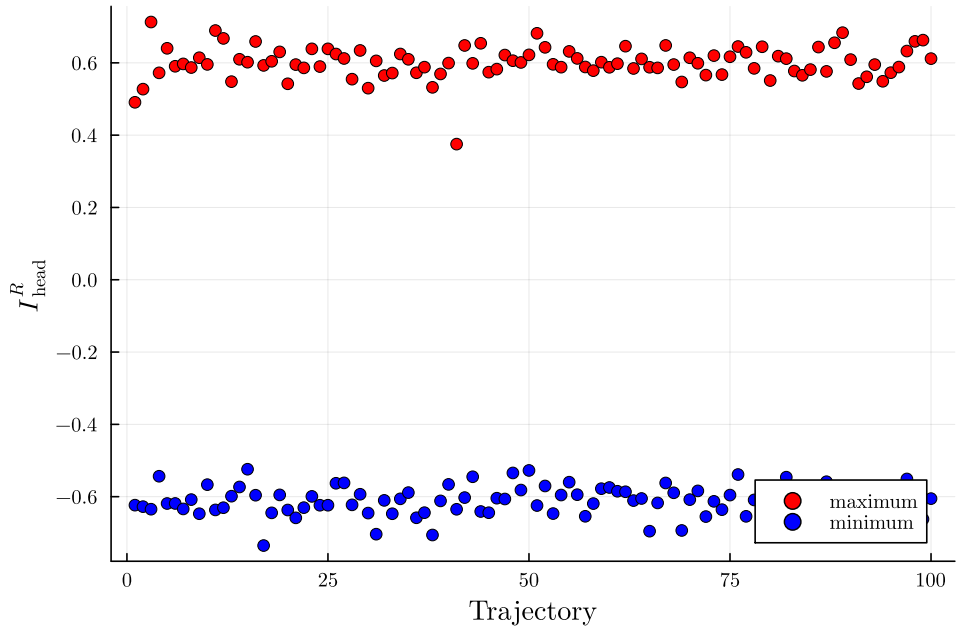


Figure 16: Maximum and Minimum values of I_{head}^R over 100 random trajectories.

3.4

The full circuit consists of 1 single bump attractor encoding the head direction, and 2 coupled bump attractors encoding the x and y positions, each composed of a left and right population. The input to each population is as follows:

$$I_i^H(t) = J(\cos(x_i) m_{\cos}(t) + \sin(x_i) m_{\sin}(t)) + I_0 \cos\left(x_i^H - \theta_H(t)\right),$$

$$I_i^{X,L}(t) = J\left(\cos\left(x_i^{X,L} + \theta\right)\left(m_{\cos}^{X,L}(t) + m_{\cos}^{X,R}(t)\right) + \sin\left(x_i^{X,L} + \theta\right)\left(m_{\sin}^{X,L}(t) + m_{\sin}^{X,R}(t)\right)\right) - \frac{J_{\text{head}}}{N} \sum_{j=1}^{N} \cos\left(x_j^H\right) S_j^H(t)$$

$$I_i^{X,R}(t) = J\left(\cos\left(x_i^{X,R} - \theta\right)\left(m_{\cos}^{X,L}(t) + m_{\cos}^{X,R}(t)\right) + \sin\left(x_i^{X,R} - \theta\right)\left(m_{\sin}^{X,L}(t) + m_{\sin}^{X,R}(t)\right)\right) + \frac{J_{\text{head}}}{N} \sum_{j=1}^{N} \cos\left(x_j^H\right) S_j^H(t)$$

$$I_i^{Y,L}(t) = J\left(\cos\left(x_i^{Y,L} + \theta\right)\left(m_{\cos}^{Y,L}(t) + m_{\cos}^{Y,R}(t)\right) + \sin\left(x_i^{Y,L} + \theta\right)\left(m_{\sin}^{Y,L}(t) + m_{\sin}^{Y,R}(t)\right)\right) - \frac{J_{\text{head}}}{N} \sum_{j=1}^{N} \sin\left(x_j^H\right) S_j^H(t)$$

$$I_i^{Y,R}(t) = J\left(\cos\left(x_i^{Y,R} - \theta\right)\left(m_{\cos}^{Y,L}(t) + m_{\cos}^{Y,R}(t)\right) + \sin\left(x_i^{Y,R} - \theta\right)\left(m_{\sin}^{Y,L}(t) + m_{\sin}^{Y,R}(t)\right)\right) + \frac{J_{\text{head}}}{N} \sum_{j=1}^{N} \sin\left(x_j^H\right) S_j^H(t)$$

3.5 Integrating an entire trajectory

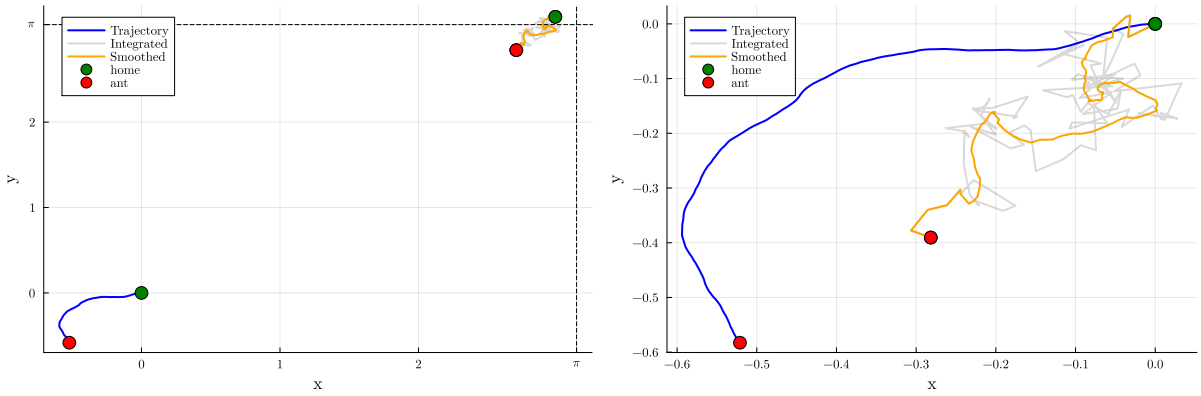


Figure 17: Integrated trajectory with volatility $\sigma = 0.01$ centered around π (left) and centered at the origin (right).

Directly plotting the bump locations without any modifications gives an integrated trajectory starting at $(x, y) = (\pi, \pi)$ since the bump attractors are initialized at equilibrium around pi. After centering at the origin, as shown in Figure 17, we observe that our network does not quite solve the task.

Modifying the parameters

As seen in 1.3 with Figure 7, increasing the size of our network helps with the stability of the bump attractors. From Figure 18 we conclude that this indeed helps the integration.

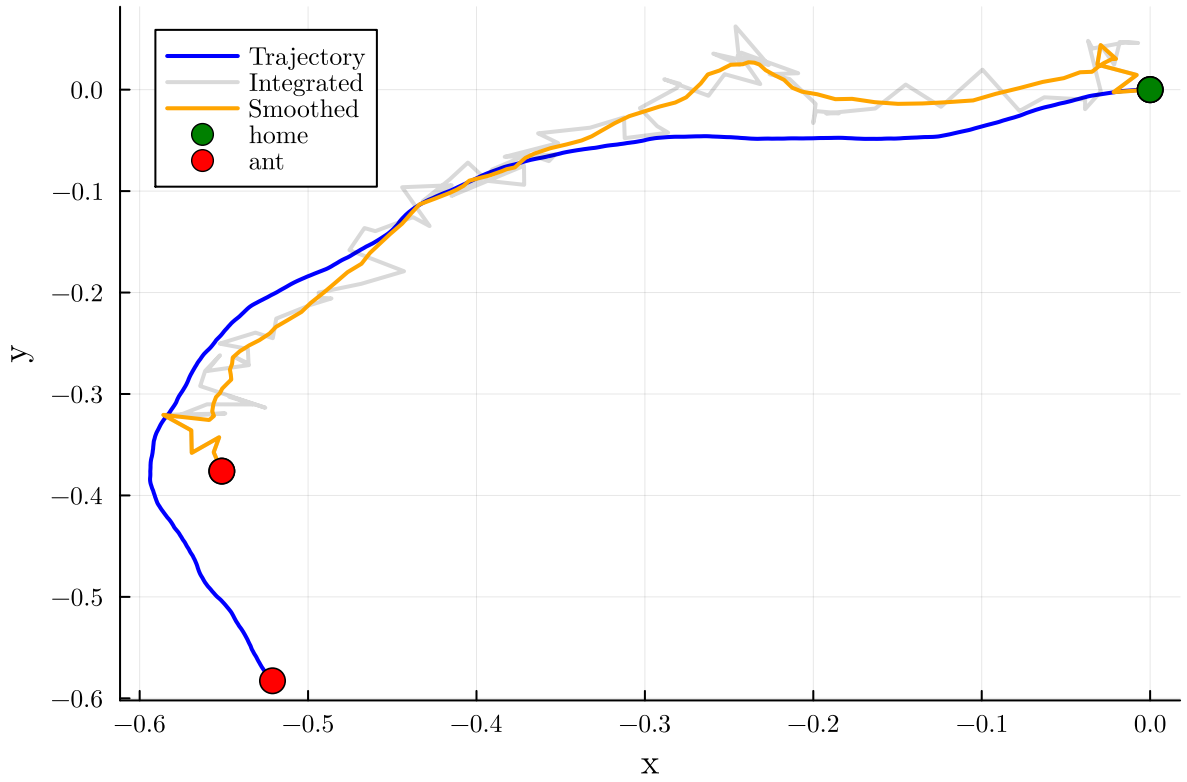


Figure 18: Trajectory integration with increased network size N.

However, it still isn't performing too well. We notice that the scale on which the trajectory's position is integrated is very small compared to the angle range of the integrator networks. Indeed, as shown in , the integrated positions are almost flat lines at $\theta_{\text{bump}} = \pi$. We conclude from this that the scale of the integrated trajectory makes it very sensitive to the noise of the integrator network, and thus loses

some precision. Upon investigation, we found that increasing the integrator network's offset angle between their two populations increases the magnitude of the integrator bump attractor variations, in the same way the angle φ affected the bump in 1.5 as shown in Figure 10. However, setting the angle to large risks the integrated trajectory to wrap around the circle and thus losing its interpretability. We found that setting $\theta = 30^{\circ}$ yields nice results, as shown in .

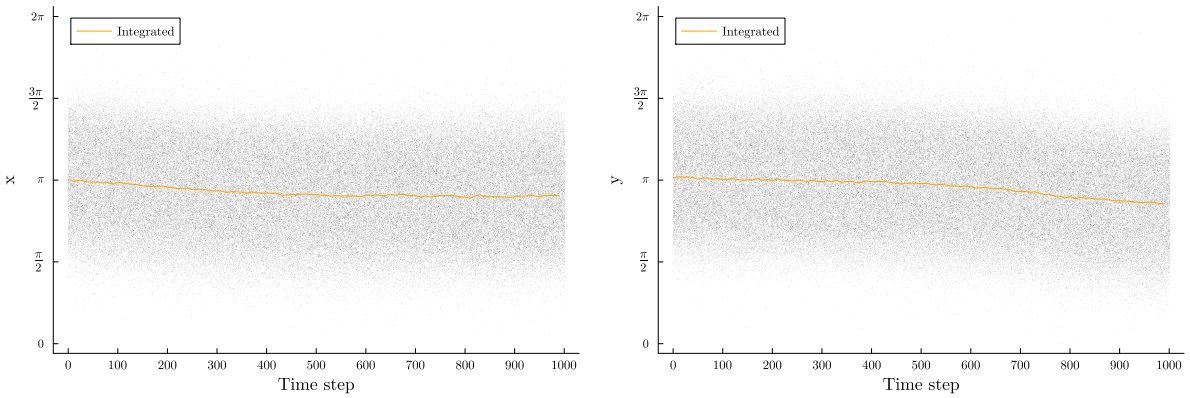


Figure 19: The integrated position is very close to $\theta = \pi$ and therefore is very sensitive to the noisy networks.

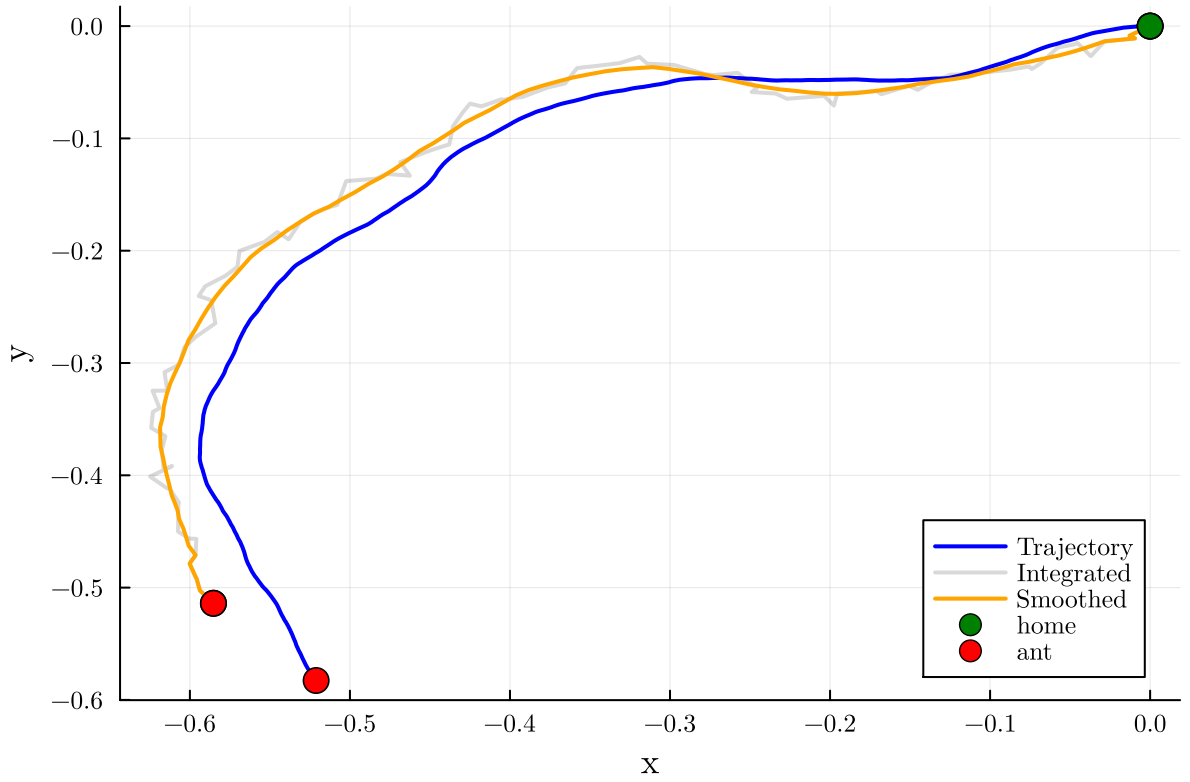


Figure 20: Trajectory integration with increased offset angle and scaled down for comparison with trajectory.

Sanity check: taking values of J_{head} outside of the linear range produces scaled trajectories.

Setting $J_{\text{head network}} = 5$ but $J_{\text{X/Y networks}} = 3$ as indicated in 2.2 produces more accurate results.

Increasing N gives more stable results.

3.6

The argument in favor of this model's biological accuracy is in fact the stochastic drift it displays. Indeed, as explained in [1], the neuronal noise of the bump attractor network induces a stochastic drift, which models the homing distance error observed in experiments with ant populations. However, the size of the network required to obtain the same drift as the one observed in nature would need a "multiple of 6 million", when the entire brain of insects like ants is estimated to have 200 000 neurons in total.

3.7

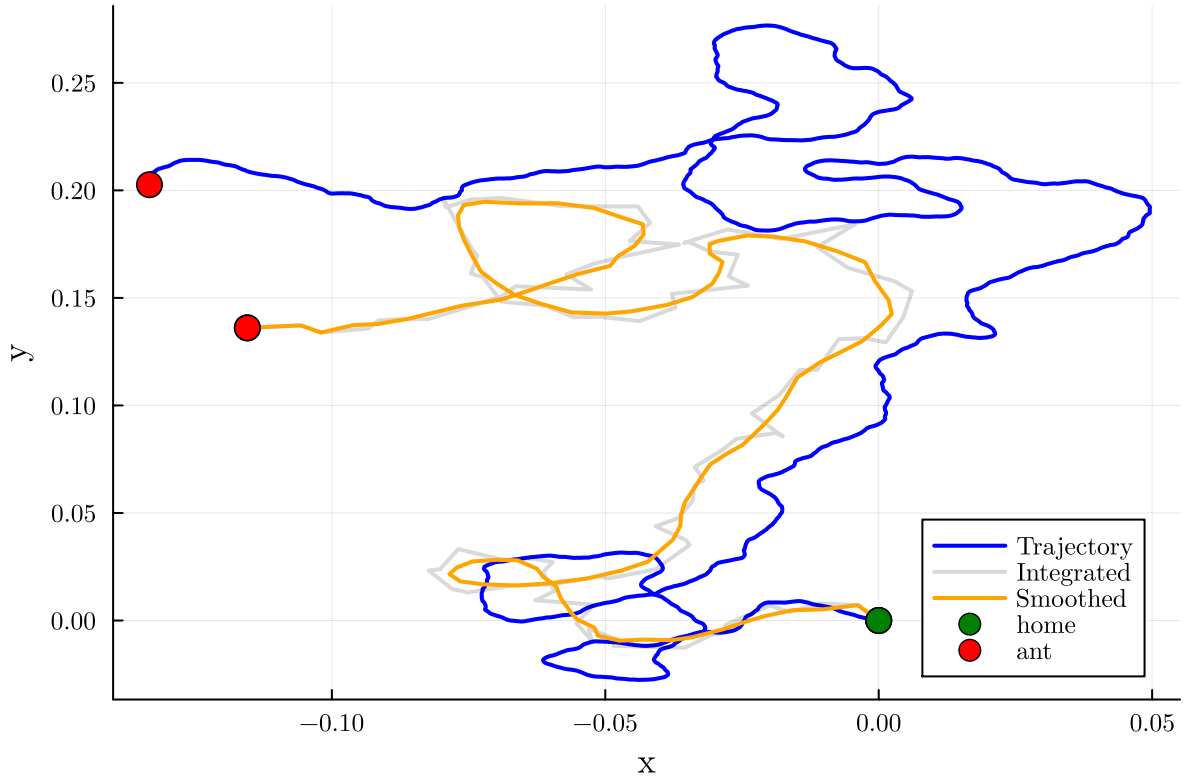


Figure 21: Integration of a $\sigma = 0.08$ trajectory.

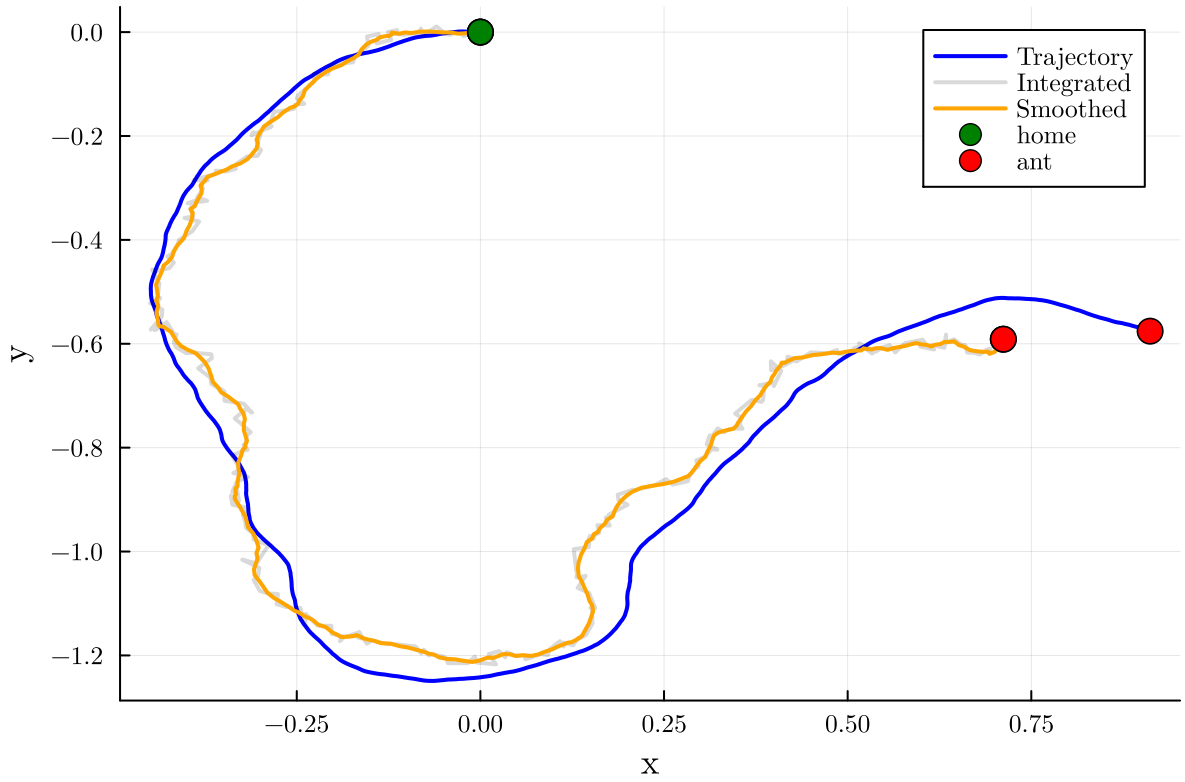


Figure 22: Integration of a $T = 3000$s trajectory. Note: we needed to reduce the offset angle to $\theta = 20$ in order to stay within the circle.

3.8 Conclusion

Bump attractor neural networks can be used to perform path integration. The single bump attractor network can be used to record the head direction and two coupled bump attractors to integrate the position. The connection strength between the head direction network and the position integrators must satisfy a relationship similar to $J_{\text{head}} = 5\Delta t$ in order to obtain external inputs that keep the results in a linear regime. Increasing the number of neurons in the network and manipulating the offset angle between the integrator networks helps reduce the noise in the resulting trajectory.

Bibliography

[1] I. Pisokas and M. H. Hennig, "Can the Insect Path Integration Memory be a Bump Attractor?," *bioRxiv*, 2022, doi: 10.1101/2022.04.05.487126.