The following model has been extensively used to describe the motion of active self-propelled microswimmers (e.g. bacteria, Janus particles, etc.) in two dimensions as used in [1].

The equation for the angular dynamics is a Langevin equation given by:

$$\dot{\theta} = \omega + \sqrt{2D_r}\mathcal{E}_\theta \tag{1}$$

$$\dot{x} = v\cos\theta + \sqrt{2D_t}\mathcal{E}_x \tag{2}$$

$$\dot{y} = v\sin\theta + \sqrt{2D_t}\mathcal{E}_y \tag{3}$$

where ω is the angular velocity of the swimmer, v the self-propulsion speed, D_r is the rotational diffusion coefficient, Dt is the translational diffusion coefficient, and $\mathcal{E}_\theta$, $\mathcal{E}_x$, $\mathcal{E}_y$ are zero-mean Gaussian, white, and independent to each other, each with autocorrelation function $\langle \mathcal{E}_\theta(t)\mathcal{E}_\theta(t')\rangle = \langle \mathcal{E}_x(t)\mathcal{E}_x(t')\rangle = \langle \mathcal{E}_y(t)\mathcal{E}_y(t')\rangle = \delta(t-t')$ variable. Also, the stochastic differential equation has the initial conditions $\theta(0) = x(0) = y(0) = 0$.

Analytical Formula for P(θ, t) and Stationary State

The equation for the angular dynamics is a Langevin equation given by:

$$\dot{\theta} = \omega + \sqrt{2D_r}\mathcal{E}_\theta \tag{4}$$

The probability density P(θ, t) can be derived from the Fokker-Planck equation, which describes the time evolution of the probability density function $P(\theta, t)$ associated with the Langevin equation. The Fokker-Planck equation is given by:

$$\frac{\partial P(\theta,t)}{\partial t} = -\frac{\partial}{\partial\theta}\left(A_\theta\, P(\theta,t)\right) + \frac{1}{2}\frac{\partial^2}{\partial\theta^2}\left(B_\theta\, P(\theta,t)\right). \tag{5}$$

Where A_θ is the drift coefficient, which represents deterministic changes in θ and B_θ is the square of the diffusion coefficient, which represents stochastic fluctuations in θ

Thus substituting the drift coefficient(ω) and the diffusion coefficient ($\sqrt{2D_r}$) in Equation 4 into the Fokker-Planck equation, we get

$$\frac{\partial P(\theta,t)}{\partial t} = -\frac{\partial}{\partial\theta}(\omega P(\theta,t)) + \frac{1}{2}\frac{\partial^2}{\partial\theta^2}\left((2D_r)P(\theta,t)\right)$$

$$\frac{\partial P(\theta,t)}{\partial t} = -\omega\frac{\partial P(\theta,t)}{\partial\theta} + D_r\frac{\partial^2 P(\theta,t)}{\partial\theta^2}. \tag{6}$$

We then solve the Fokker-Planck (FP) equation (Equation 6) to find the probability density $P(\theta, t)$. To do this, we use the Fourier transform with respect to θ. The Fourier transform of $P(\theta, t)$ is defined as:

$$\tilde{P}(k,t) = \int_{-\infty}^{\infty} P(\theta,t)e^{-ik\theta}\, d\theta,$$

where k is the Fourier variable and the inverse Fourier transform is:

$$P(\theta,t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \tilde{P}(k,t)e^{ik\theta}\, dk.$$

Applying the Fourier transform to the Folker-Planck equation, the derivatives transform as follows:

1. **First derivative in θ**:
The Fourier transform of the first derivative $\frac{\partial P(\theta,t)}{\partial \theta}$ is:

$$\mathcal{F}\left[\frac{\partial P(\theta,t)}{\partial \theta}\right] = \int_{-\infty}^{\infty} \frac{\partial P(\theta,t)}{\partial \theta} e^{-ik\theta}\, d\theta.$$

To evaluate this, we use **integration by parts**. Let $u = e^{-ik\theta}$ and $dv = \frac{\partial P(\theta,t)}{\partial \theta} d\theta$. Then:

$$du = -ike^{-ik\theta} d\theta, \quad v = P(\theta,t).$$

Applying integration by parts:

$$\int_{-\infty}^{\infty} \frac{\partial P(\theta,t)}{\partial \theta} e^{-ik\theta}\, d\theta = \left[P(\theta,t)e^{-ik\theta}\right]_{-\infty}^{\infty} + ik \int_{-\infty}^{\infty} P(\theta,t)e^{-ik\theta}\, d\theta.$$

The boundary term $\left[P(\theta,t)e^{-ik\theta}\right]_{-\infty}^{\infty}$ vanishes because $P(\theta,t)$ typically decays to zero as $\theta \to \pm\infty$.

Thus, we are left with:

$$\mathcal{F}\left[\frac{\partial P(\theta,t)}{\partial \theta}\right] = ik \int_{-\infty}^{\infty} P(\theta,t)e^{-ik\theta}\, d\theta = ik\tilde{P}(k,t).$$

So, the first derivative in θ transforms as:

$$\frac{\partial P(\theta,t)}{\partial \theta} \to ik\tilde{P}(k,t).$$

2. **Second derivative in θ**:
The Fourier transform of the second derivative $\frac{\partial^2 P(\theta,t)}{\partial \theta^2}$ is:

$$\mathcal{F}\left[\frac{\partial^2 P(\theta,t)}{\partial \theta^2}\right] = \int_{-\infty}^{\infty} \frac{\partial^2 P(\theta,t)}{\partial \theta^2} e^{-ik\theta}\, d\theta.$$

We apply **integration by parts** twice. First, let $u = e^{-ik\theta}$ and $dv = \frac{\partial^2 P(\theta,t)}{\partial \theta^2} d\theta$. Then:

$$du = -ike^{-ik\theta} d\theta, \quad v = \frac{\partial P(\theta,t)}{\partial \theta}.$$

Applying integration by parts:

$$\int_{-\infty}^{\infty} \frac{\partial^2 P(\theta,t)}{\partial \theta^2} e^{-ik\theta}\, d\theta = \left[\frac{\partial P(\theta,t)}{\partial \theta} e^{-ik\theta}\right]_{-\infty}^{\infty} + ik \int_{-\infty}^{\infty} \frac{\partial P(\theta,t)}{\partial \theta} e^{-ik\theta}\, d\theta.$$

The boundary term $\left[\frac{\partial P(\theta,t)}{\partial \theta} e^{-ik\theta}\right]_{-\infty}^{\infty}$ vanishes because $\frac{\partial P(\theta,t)}{\partial \theta}$ typically decays to zero as $\theta \to \pm\infty$. Using the result from the first derivative, we have:

$$\mathcal{F}\left[\frac{\partial^2 P(\theta,t)}{\partial \theta^2}\right] = ik \cdot \mathcal{F}\left[\frac{\partial P(\theta,t)}{\partial \theta}\right] = ik \cdot (ik\tilde{P}(k,t)) = -k^2\tilde{P}(k,t).$$

So, the second derivative in θ transforms as:

$$\frac{\partial^2 P(\theta,t)}{\partial \theta^2} \to -k^2 \tilde{P}(k,t).$$

3. **Time derivative**:
The Fourier transform of the time derivative $\frac{\partial P(\theta,t)}{\partial t}$ is:

$$\mathcal{F}\left[\frac{\partial P(\theta,t)}{\partial t}\right] = \int_{-\infty}^{\infty} \frac{\partial P(\theta,t)}{\partial t} e^{-ik\theta}\, d\theta = \frac{\partial \tilde{P}(k,t)}{\partial t}$$

(Since the Fourier transform is with respect to θ not t, we can interchange the order of differentiation and integration.)

So, the time derivative transforms as:

$$\frac{\partial P(\theta,t)}{\partial t} \to \frac{\partial \tilde{P}(k,t)}{\partial t}.$$

Substituting these into the Folker-Planck equation, we get the transformed equation:

$$\frac{\partial \tilde{P}(k,t)}{\partial t} = -i\omega k \tilde{P}(k,t) - D_r k^2 \tilde{P}(k,t) = (-i\omega k - D_r k^2)\tilde{P}(k,t)$$

This is a simple linear ODE of the form:

$$\frac{\partial \tilde{P}(k,t)}{\partial t} = \lambda(k)\tilde{P}(k,t),$$

where $\lambda(k) = -i\omega k - D_r k^2$ and the solution to this ODE is $\tilde{P}(k,t) = \tilde{P}(k,0)e^{\lambda(k)t}$

Here, $\tilde{P}(k,0)$ is the Fourier transform of the initial condition $P(\theta,0)$.

Assuming the initial condition is a delta function at $\theta = 0$:

$$P(\theta,0) = \delta(\theta).$$

The Fourier transform of the delta function is:

$$\tilde{P}(k,0) = 1.$$

Thus, the solution in Fourier space becomes:

$$\tilde{P}(k,t) = e^{(-i\omega k - D_r k^2)t}.$$

To find $P(\theta,t)$, we take the inverse Fourier transform of $\tilde{P}(k,t)$:

$$P(\theta,t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} e^{(-i\omega k - D_r k^2)t} e^{ik\theta}\, dk = \frac{1}{2\pi}\int_{-\infty}^{\infty} e^{-D_r t k^2 + i(\theta - \omega t)k}\, dk.$$

This integral can be evaluated using the Gaussian integral formula;

$$\int_{-\infty}^{\infty} e^{-ax^2+bx+c}\, dx = \sqrt{\frac{\pi}{a}} \exp\left(\frac{b^2}{4a} + c\right)$$

where $a = D_r t$, $b = i(\theta - \omega t)$ and $c = 0$, to obtain

$$\boxed{P(\theta, t) = \frac{1}{\sqrt{4\pi D_r t}} \exp\left(-\frac{(\theta - \omega t)^2}{4 D_r t}\right).}$$

Thus $P(\theta, t)$ is a Gaussian distribution with Mean ωt and Variance $2D_r t$.

Stationary State

In the stationary state, the probability density does not change with time. Thus

$$\frac{\partial P(\theta, t)}{\partial t} = -\omega \frac{\partial P_{\text{st}}(\theta)}{\partial \theta} + D_r \frac{\partial^2 P_{\text{st}}(\theta)}{\partial \theta^2} = 0.$$

The stationary equation is a second-order ODE and hence using the ansatz $P_{\text{st}}(\theta) = e^{\lambda\theta}$, we substitute into the ODE:

$$D_r \lambda^2 e^{\lambda\theta} - \omega \lambda e^{\lambda\theta} = 0.$$

Divide through by $e^{\lambda\theta}$:

$$D_r \lambda^2 - \omega \lambda = 0.$$

The solutions are:

$$\lambda = 0 \quad \text{or} \quad \lambda = \frac{\omega}{D_r}.$$

Thus, the general solution is:

$$P_{\text{st}}(\theta) = C_1 + C_2 e^{\frac{\omega}{D_r}\theta}.$$

The angle θ is periodic with period 2π, so:

$$P_{\text{st}}(\theta + 2\pi) = P_{\text{st}}(\theta).$$

Substituting the general solution:

$$C_1 + C_2 e^{\frac{\omega}{D_r}(\theta + 2\pi)} = C_1 + C_2 e^{\frac{\omega}{D_r}\theta}.$$

Simplify:

$$C_2 e^{\frac{\omega}{D_r} 2\pi} = C_2.$$

Case 1: $\omega = 0$ (inactive chiral Brownian motion)

If $\omega = 0$, then $e^{\frac{\omega}{D_r} 2\pi} = e^0 = 1$, and the equation becomes:

$$C_2 = C_2.$$

This holds for **any** C_2, so the general solution simplifies to:

$$P_{\text{st}}(\theta) = C_1 + C_2.$$

The periodicity condition is automatically satisfied, and the normalization condition requires:

$$\int_0^{2\pi} P_{\text{st}}(\theta)\, d\theta = 1.$$

Substituting $P_{\text{st}}(\theta) = C_1 + C_2$:

$$(C_1 + C_2) \cdot 2\pi = 1 \quad \Rightarrow \quad C_1 + C_2 = \frac{1}{2\pi}.$$

Thus, the stationary solution when there is inactive chiral Brownian motion ($\omega = 0$) is:

$$P_{\text{st}}(\theta) = \frac{1}{2\pi}.$$

Case 2: $\omega \neq 0$ **(active chiral Brownian motion)**

If $\omega \neq 0$, then $e^{\frac{\omega}{D_r}2\pi} \neq 1$, and the equation $C_2 e^{\frac{\omega}{D_r}2\pi} = C_2$ can **only** hold if $C_2 = 0$. Hence it is impossible to solve $P_s t(\theta)$ when $\omega \neq 0$.

Thus, the system does not reach a stationary state in the active chiral case ($\omega \neq 0$). Instead, the probability density $P(\theta, t)$ evolves over time [2]. This is because for the active chiral Brownian motion, the angle θ undergoes persistent rotation due to the drift term ω.

Comparing theoretical and simulated $P(\theta, t)$

Analysis of the angular dynamics $\theta(t)$ of an active chiral Brownian particle described by the stochastic differential equation:

$$\dot{\theta} = \omega + \sqrt{2D_r}\xi_\theta,$$

where $\omega = 10\,\text{rad}\,\text{s}^{-1}$ is the angular velocity, $D_r = 0.17\,\text{rad}^2/\text{s}$ is the rotational diffusion coefficient, and ξ_θ is Gaussian white noise, by comparing the theoretical probability density $P(\theta, t)$ with results from numerical simulations for $t = 1\,\text{s}$ and $t = 5\,\text{s}$.

For $t = 1\,\text{s}$, the expected mean of $\theta_{t=1s}$ is:

$$\langle \theta_{t=1s} \rangle = \omega t = 10\,\text{rad}.$$

and standard deviation of $\theta_{t=1s}$ is:

$$\sigma_{\theta_{t=1s}} = \sqrt{2D_r t} = \sqrt{2 \times 0.17 \times 1} \approx 0.58\,\text{rad}.$$

Figure 1 shows the comparison between the theoretical distribution (red line) and the simulated data (blue dots) for $t = 1s$. The distribution is centered at $\theta = 10\,\text{rad}$ with a narrow spread.

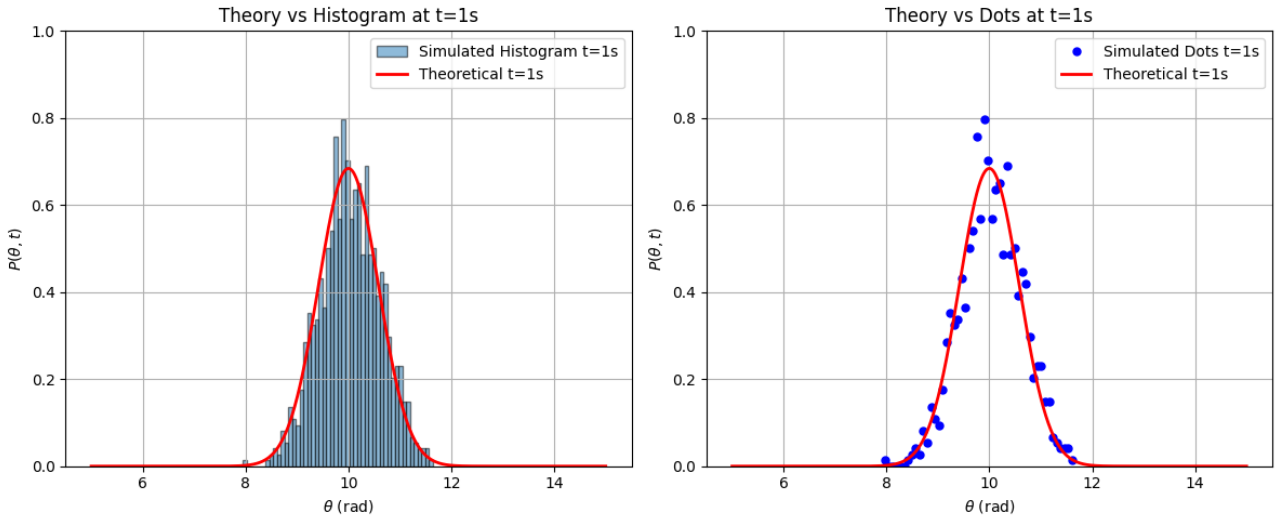

Figure 1: Probability density of θ at $t = 1\,\text{s}$ [3].

For $t = 5\,\text{s}$, the expected mean of θ is:

$$\langle\theta\rangle = \omega t = 50\,\text{rad}.$$

The standard deviation of θ is:

$$\sigma_\theta = \sqrt{2D_r t} = \sqrt{2 \times 0.17 \times 5} \approx 1.3\,\text{rad}.$$

The distribution is centered at $\theta = 50\,\text{rad}$ with a wider spread compared to $t = 1\,\text{s}$. Figure 2 shows the theoretical distribution (red line) and the simulated data (blue dots) at $t = 5s$.

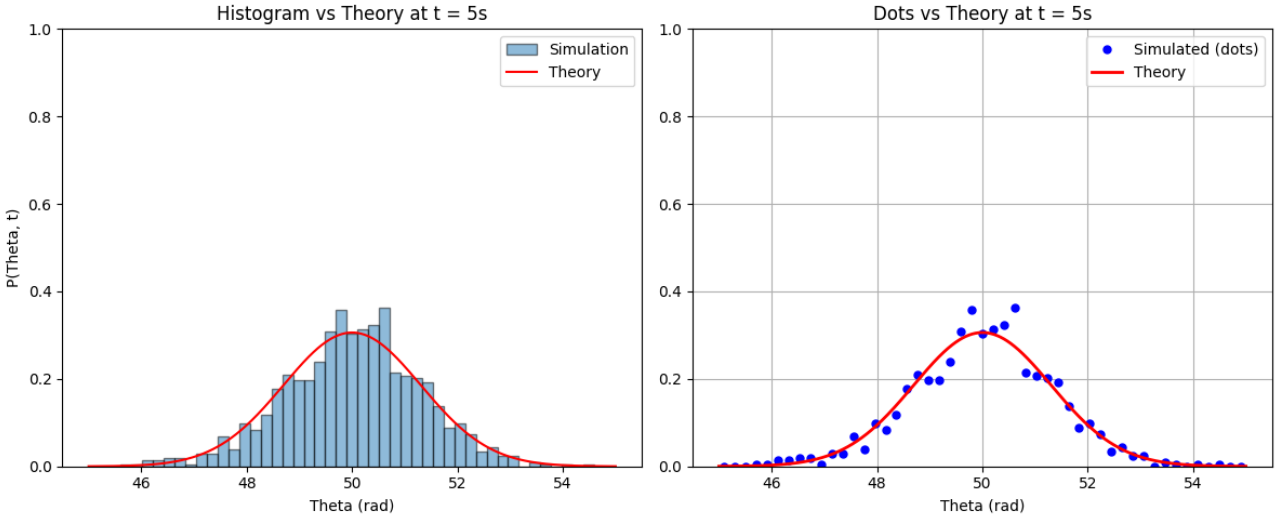

Figure 2: Probability density of θ at $t = 5\,\text{s}$ [3].

The theoretical and simulated results agree well for both $t = 1\,\text{s}$ and $t = 5\,\text{s}$. The distributions are centered at the expected mean $\langle\theta\rangle = \omega t$, and the spread increases with time due to

rotational diffusion. Thus the peak of the probability distribution of $t = 1s$ is much higher (≈ 0.7) compared to that of $t = 5s$ (≈ 0.3). This confirms the accuracy of the theoretical model.

Stochastic differential equation for $r(t)$

Let's derive the stochastic differential equation for the swimmer's distance from the center, denoted as:

$$r(t) = \sqrt{x^2(t) + y^2(t)} \tag{7}$$

using stochastic calculus. We will address both the Itô and Stratonovich forms in this derivation.

i. Itô

For Itô stochastic calculus, since $r(t)$ is a twice-differentiable function mapping from $\mathbb{R}^2$ to $\mathbb{R}$, and both $\dot{x}$ and $\dot{y}$ are stochastic differential equations, which can be written in Itô form as

$$dx = v\cos\theta dt + \sqrt{2D_t}\cdot dB_x \tag{8}$$

$$dy = v\sin\theta dt + \sqrt{2D_t}\cdot dB_y \tag{9}$$

where $dB_x = \mathcal{E}_x/dt$ and $dB_y = \mathcal{E}_y/dt$, and are independent Wiener processes.

The Itô SDE for $r(t)$ is derived using Itô's lemma, which for a function $r(x, y)$ of two stochastic processes $x(t)$ and $y(t)$ is given by:

$$dr = \frac{\partial r}{\partial x}dx + \frac{\partial r}{\partial y}dy + \frac{1}{2}\frac{\partial^2 r}{\partial x^2}(dx)^2 + \frac{1}{2}\frac{\partial^2 r}{\partial y^2}(dy)^2 + \frac{\partial^2 r}{\partial x \partial y}dxdy. \tag{10}$$

and the respective partial derivatives of $r(t) = \sqrt{x^2 + y^2}$ are

$$\frac{\partial r}{\partial x} = \frac{x}{\sqrt{(x^2+y^2)}} = \frac{x}{r}; \qquad \frac{\partial r}{\partial y} = \frac{y}{\sqrt{(x^2+y^2)}} = \frac{y}{r}$$

$$\frac{\partial^2 r}{\partial x^2} = \frac{1}{\sqrt{(x^2+y^2)}} - \frac{x^2}{\sqrt[3]{(x^2+y^2)}} = \frac{y^2}{r^3}; \qquad \frac{\partial^2 r}{\partial y^2} = \frac{1}{\sqrt{(x^2+y^2)}} - \frac{y^2}{\sqrt[3]{(x^2+y^2)}} = \frac{x^2}{r^3}$$

$$\frac{\partial^2 r}{\partial x \partial y} = \frac{\partial^2 r}{\partial y \partial x} = -\frac{xy}{\sqrt[3]{(x^2+y^2)}} = -\frac{xy}{r^3}$$

The variations of Equation 8 and Equation 9 are

$$(dx)^2 = \left(v\cos(\theta)dt + \sqrt{2D_t}dB_x\right)^2 = (v\cos(\theta)dt)^2 + 2v\cos(\theta)dt\cdot\sqrt{2D_t}dB_x + \left(\sqrt{2D_t}dB_x\right)^2$$

$$\therefore (dx)^2 = 0 + 0 + 2D_t(dB_x)^2 = 2D_t dt. \tag{11}$$

$$(dy)^2 = \left(v\sin(\theta)dt + \sqrt{2D_t}dB_y\right)^2 = (v\sin(\theta)dt)^2 + 2v\sin(\theta)dt\cdot\sqrt{2D_t}dB_y + \left(\sqrt{2D_t}dB_y\right)^2$$

$$\therefore (dy)^2 = 0 + 0 + 2D_t(dB_y)^2 = 2D_t dt. \tag{12}$$

$$
\begin{aligned}
dxdy = dydx &= \left(v\cos(\theta)dt + \sqrt{2D_t}dB_x\right)\left(v\sin(\theta)dt + \sqrt{2D_t}dB_y\right). \\
&= v\cos(\theta)dt \cdot v\sin(\theta)dt + v\cos(\theta)dt \cdot \sqrt{2D_t}dB_y \\
&\quad + \sqrt{2D_t}dB_x \cdot v\sin(\theta)dt + \sqrt{2D_t}dB_x \cdot \sqrt{2D_t}dB_y \\
\therefore dxdy &= 0 + 0 + 0 + 0 = 0. \qquad (13)
\end{aligned}
$$

Substituting the partial derivatives and stochastic differentials into Itô's lemma, we get:

$$
dr = \frac{x}{r}dx + \frac{y}{r}dy + \frac{1}{2}\frac{y^2}{r^3}(dx)^2 + \frac{1}{2}\frac{x^2}{r^3}(dy)^2 - \frac{xy}{r^3}dxdy.
$$

Substituting dx, dy, $(dx)^2$, $(dy)^2$, and $dxdy$,we get:

$$
\begin{aligned}
dr = &\frac{x}{r}\left(v\cos(\theta)dt + \sqrt{2D_t}dB_x\right) + \frac{y}{r}\left(v\sin(\theta)dt + \sqrt{2D_t}dB_y\right) \\
&+ \frac{1}{2}\frac{y^2}{r^3}(2D_t dt) + \frac{1}{2}\frac{x^2}{r^3}(2D_t dt) - \frac{xy}{r^3}(0).
\end{aligned}
$$

$$
dr = \frac{xv\cos(\theta) + yv\sin(\theta)}{r}dt + \frac{x\sqrt{2D_t}}{r}dB_x + \frac{y\sqrt{2D_t}}{r}dB_y + \frac{D_t y^2}{r^3}dt + \frac{D_t x^2}{r^3}dt.
$$

$$
= \left(\frac{xv\cos(\theta) + yv\sin(\theta)}{r} + \frac{D_t}{r}\right)dt + \frac{x\sqrt{2D_t}}{r}dB_x + \frac{y\sqrt{2D_t}}{r}dB_y \qquad (\text{ since } x^2 + y^2 = r^2)
$$

$$
dr = \left(\frac{v\left[x\cos(\theta) + y\sin(\theta)\right]}{r} + \frac{D_t}{r}\right)dt + \frac{x\sqrt{2D_t}}{r}\cdot dB_x + \frac{y\sqrt{2D_t}}{r}\cdot dB_y.
$$

Thus, the Itô SDE for $r(t)$ is:

$$
\boxed{dr = \left(\frac{v\left[x\cos(\theta) + y\sin(\theta)\right]}{r} + \frac{D_t}{r}\right)dt + \frac{\sqrt{2D_t}}{r}\left[x\cdot dB_x + y\cdot dB_y\right]}
$$

ii. Stratonovich

In the form of Stratonovich, we can write the stochastic differential equations as:

$$
d\theta = \omega dt + \sqrt{2D_r}\circ dB_\theta, \qquad (14)
$$

$$
dx = v\cos(\theta)dt + \sqrt{2D_t}\circ dB_x, \qquad (15)
$$

$$
dy = v\sin(\theta)dt + \sqrt{2D_t}\circ dB_y. \qquad (16)
$$

and the differential of the distance $r(t)$ can be computed using the chain rule:

$$dr = \frac{\partial r}{\partial x} \circ dx + \frac{\partial r}{\partial y} \circ dy.$$

Recall partial derivatives of r are:

$$\frac{\partial r}{\partial x} = \frac{x}{r}, \quad \frac{\partial r}{\partial y} = \frac{y}{r}.$$

and substituting these into the Stratonovich SDE for $r(t)$, we get:

$$dr = \frac{x}{r} \circ dx + \frac{y}{r} \circ dy.$$

Now substituting the Stratonovich SDEs for dx and dy as given by Equation 15 and Equation 16:

$$dr = \frac{x}{r} \circ \left(v\cos(\theta)dt + \sqrt{2D_t} \circ dB_x \right) + \frac{y}{r} \circ \left(v\sin(\theta)dt + \sqrt{2D_t} \circ dB_y \right).$$

Using the linearity of the Stratonovich integral, this simplifies to:

$$dr = \frac{xv\cos(\theta) + yv\sin(\theta)}{r} dt + \frac{x\sqrt{2D_t}}{r} \circ dB_x + \frac{y\sqrt{2D_t}}{r} \circ dB_y.$$

Thus, the Stratonovich SDE for $r(t)$ is:

$$\boxed{dr = \frac{v\left[x\cos(\theta) + y\sin(\theta)\right]}{r} dt + \frac{\sqrt{2D_t}}{r}\left[x \circ dB_x + y \circ dB_y\right].}$$

Numerical simulation with $\omega = 10\,\mathrm{rad\,s^{-1}}$

The following are simulations of the particle's motion using the Euler-Maruyama method.

Plot of $x(t)$ vs Time

The horizontal position $x(t)$ of the particle as a function of time is shown in Figure 3.

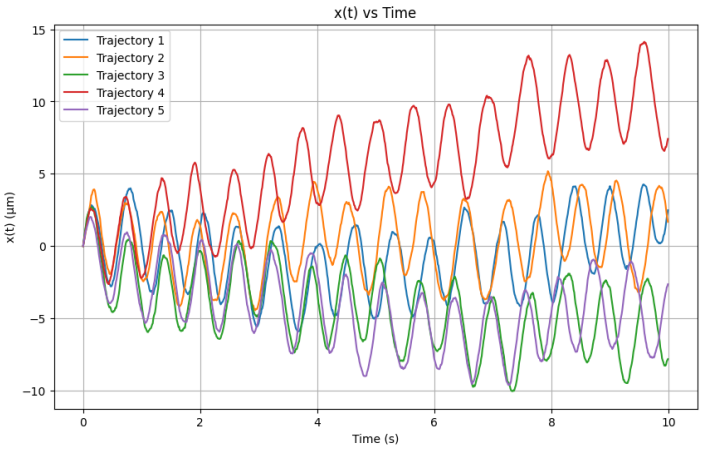

Figure 3: Plot of $x(t)$ vs time for five stochastic trajectories [3].

The plot shows oscillatory behavior due to the rotation caused by $\omega = 10\,\mathrm{rad\,s^{-1}}$, combined with noise from the diffusion term. The amplitude of the oscillations increases over time due to the self-propulsion.

Plot of $y(t)$ vs Time

The vertical position $y(t)$ of the particle as a function of time which models the motion of the particle in the y-direction is shown in Figure 4. The plot shows oscillatory behavior, but the phase of the oscillations is shifted compared to $x(t)$ due to the trigonometric $\cos(\theta_t)$ and $\sin(\theta_t)$.

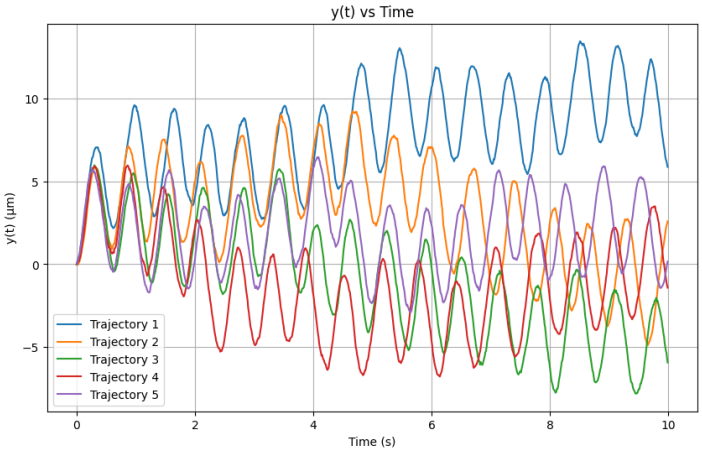

Figure 4: Plot of $y(t)$ vs time for five stochastic trajectories [3].

Plot of $y(t)$ vs $x(t)$

The 2D trajectory of the particle in the xy-plane is shown in Figure 5. The trajectories exhibit **chiral motion**, with the particles spiraling outward due to the combination of self-propulsion and rotation. The diffusion term causes the trajectories to deviate from perfect spirals.

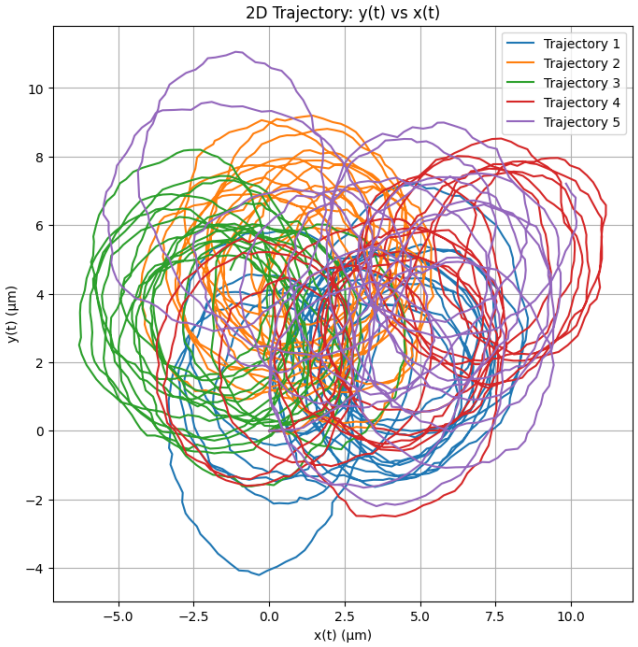

Figure 5: Plot of $y(t)$ vs $x(t)$ for five stochastic trajectories [3].

Plot of $r(t)$ vs Time

The distance ($r(t)$ from the origin as a function of time is shown in Figure 6. It increases over time due to the self-propulsion speed $v = 30\,\mu\mathrm{m\,s}^{-1}$, and is not linear due to the rotational motion ($\omega = 10\,\mathrm{rad\,s}^{-1}$) and the translational diffusion ($D_t = 0.2\,\mu\mathrm{m}^2/\mathrm{s}$), which add fluctuations.

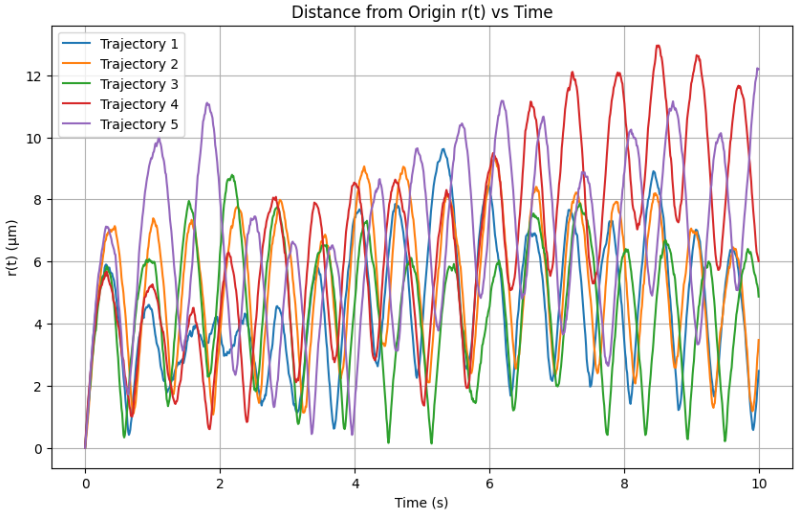

Figure 6: Plot of $r(t)$ vs time for five stochastic trajectories [3].

Plot of $dr(t)$ vs Time

The change in distance $dr(t) = r(t + \Delta t) - r(t)$ as a function of time is shown in Figure 7. The plot shows the fluctuations in the particle's distance from the origin over time.

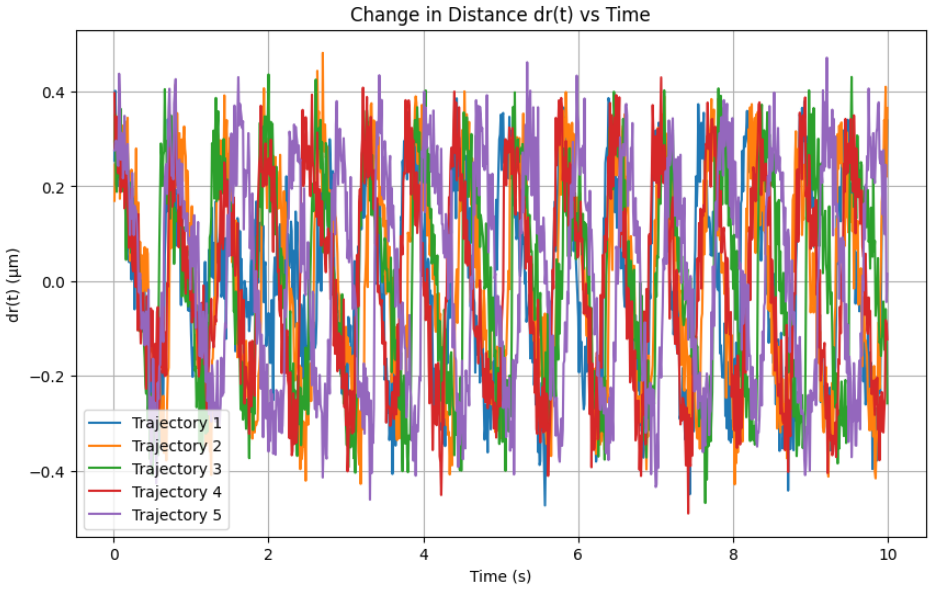

Figure 7: Plot of $dr(t)$ vs time for five stochastic trajectories [3].

Numerical simulations with $\omega = 0$ (inactive chiral)

Plot of $x(t)$ vs Time

The horizontal position $x(t)$ of the particle as a function of time for $\omega = 0$ is shown in Figure 8. The plot shows random motion in the x-direction, with no oscillatory behavior (unlike the $\omega = 10$ case).

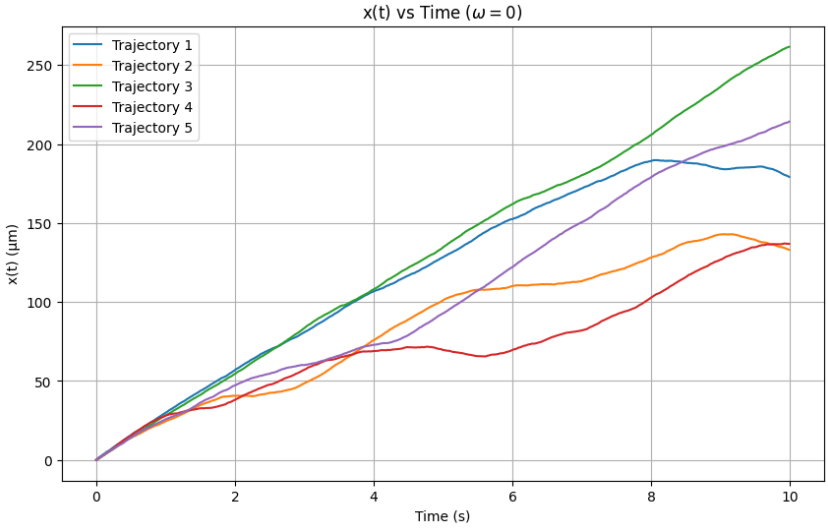

Figure 8: Plot of $x(t)$ vs time for five stochastic trajectories [3].

Plot of $y(t)$ vs Time

The vertical position $y(t)$ of the particle as a function of time which models the motion of the particle in the y-direction when $\omega = 0$ is shown in Figure 9. The plot shows random motion in the y-direction, similar to $x(t)$.

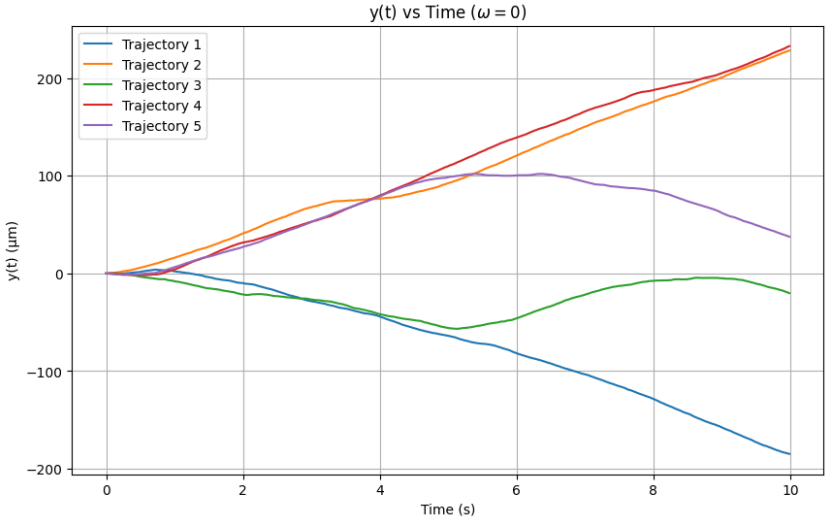

Figure 9: Plot of $y(t)$ vs time for five stochastic trajectories [3].

Plot of $y(t)$ vs $x(t)$

The 2D trajectory of the particle in the xy-plane when $\omega = 0$ is shown in Figure 10. The trajectories are random and unstructured, with no spiral-like patterns (unlike the $\omega = 10$). The particle's direction changes frequently, leading to a more diffusive motion.

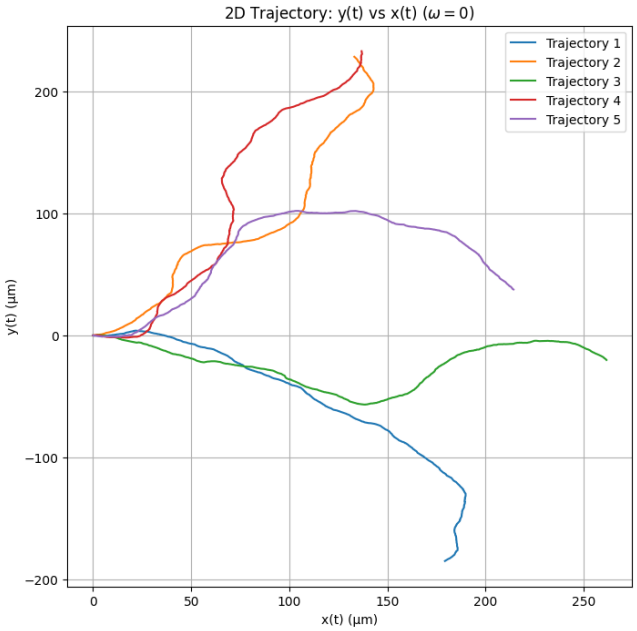

Figure 10: Plot of $y(t)$ vs $x(t)$ for five stochastic trajectories [3].

Plot of $r(t)$ vs Time

The distance ($r(t)$ from the origin as a function of time when $\omega = 0$ is shown in Figure 11. The distance $r(t)$ increases over time, but the increase is less smooth compared to the $\omega = 10$ case.

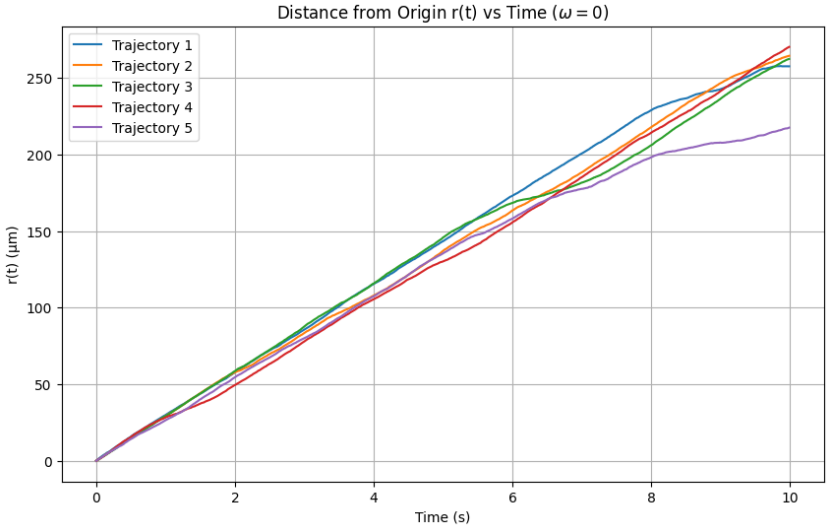

Figure 11: Plot of $r(t)$ vs time for five stochastic trajectories [3].

Plot of $dr(t)$ vs Time

The change in distance $dr(t) = r(t+\Delta t) - r(t)$ as a function of time is shown in Figure 12. The plot shows the fluctuations in the particle's distance from the origin over time. The change in distance $dr(t)$ fluctuates around a positive mean due to self-propulsion.

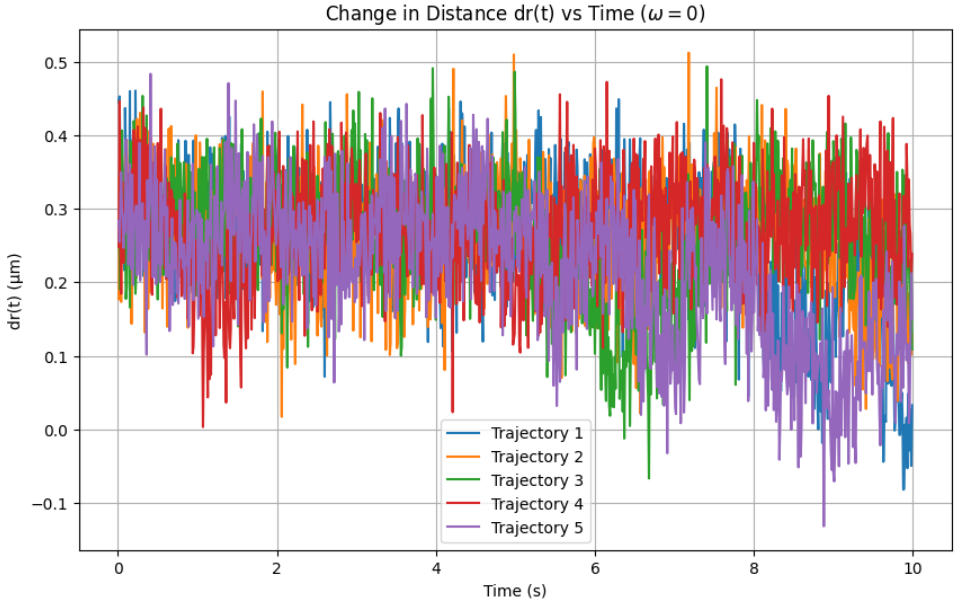

Figure 12: Plot of $dr(t)$ vs time for five stochastic trajectories [3].

In conclusion, the particle exhibits spiral-like trajectories due to the angular velocity ω while it doesn't when $\omega = 0$ (inactive chiral). Also, when $\omega = 0$, there is no oscillatory behavior when the particle moves in the x-direction or y-direction. The distance $r(t)$ increases more smoothly allowing the microswimmer to cover more distance when $\omega = 0$.

1 References

[1] C. Bechinger, R. Di Leonardo, H. L¨owen, C. Reichhardt, G. Volpe, and G. Volpe, Reviews of modern physics 88, 045006 (2016).

[2] L. Caprini and U. M. B. Marconi, Soft matter 15, 2627 (2019).

Appendices

Problem 1

```python
import numpy as np
import matplotlib.pyplot as plt

# Parameters
D_r = 0.17  # Rotational diffusion coefficient (rad^2/s)
omega = 10  # Angular velocity (rad/s)
dt = 0.01  # Time step (s)
T = 5  # Total time (s)
time_steps = np.linspace(0, T, int(T / dt) + 1)  # Time steps
N = len(time_steps)  # Number of time steps
num_trajectories = 1000  # Number of trajectories

# Initial theta
theta = np.zeros((num_trajectories, N))

# Euler-Maruyama simulation for theta(t)
for i in range(num_trajectories):
    for j in range(1, N):
        dW = np.sqrt(dt) * np.random.normal()  # Wiener increment
        theta[i, j] = theta[i, j-1] + omega * dt + np.sqrt(2 * D_r) * dW

# Theoretical distribution
def theoretical_distribution(theta, t):
    return (1 / np.sqrt(4 * np.pi * D_r * t)) * np.exp(-(theta - omega * t)
       **2 / (4 * D_r * t))
```

```python
import numpy as np
import matplotlib.pyplot as plt

# Given parameters
timesteps = 1000   # Total number of time steps
dt = 0.01          # Time step
time = np.arange(0, timesteps*dt, dt)
N = 1000           # Number of simulations

# Physical parameters
Dr = 0.17  # Rotational diffusion coefficient (rad^2/s)
Dt = 0.2   # Translational diffusion coefficient
v = 30     # Self-propulsion speed
omega = 10 # Angular velocity (rad/s)

# (b) Simulating theta(t) using Euler-Maruyama method
np.random.seed(42)  # For reproducibility
theta = np.zeros((N, timesteps))
for i in range(N):
    for t in range(1, timesteps):
        theta[i, t] = theta[i, t-1] + omega * dt + np.sqrt(2 * Dr * dt) * np
           .random.randn()
```

```
22
23 # Extract values at t = 1s and t = 5s
24 index_t1 = int(1/dt)
25 index_t5 = int(5/dt)
26 theta_t1 = theta[:, index_t1]
27 theta_t5 = theta[:, index_t5]
28
29 # Theoretical probability density function
30 def theoretical_p(theta_values, t):
31     return (1 / np.sqrt(4 * np.pi * Dr * t)) * np.exp(-((theta_values -
          omega * t)**2) / (4 * Dr * t))
32
33 # Create subplots
34 fig, axes = plt.subplots(1, 2, figsize=(12, 5))
35
36 # Plot results for t = 1s
37 t = 1
38 idx = np.argmin(np.abs(time - t))  # Find the index closest to time t
39 theta_t = theta[:, idx]  # Simulated theta values at time t
40
41 # Histogram of simulated theta (normalized)
42 hist, bin_edges = np.histogram(theta_t, bins=50, density=True)
43 bin_centers = (bin_edges[:-1] + bin_edges[1:]) / 2  # Center of bins
44
45 # Subplot 1: Theory vs Histogram
46 axes[0].hist(theta_t, bins=50, density=True, alpha=0.5, label=f'Simulated␣
      Histogram␣t={t}s', edgecolor='black')
47 theta_range = np.linspace(5, 15, 1000)  # Adjusted range around the expected
       mean (10 rad)
48 P_theta = theoretical_p(theta_range, t)
49 axes[0].plot(theta_range, P_theta, 'r-', label=f'Theoretical␣t={t}s',
      linewidth=2)
50 axes[0].set_xlabel(r'$\theta$␣(rad)')
51 axes[0].set_ylabel(r'$P(\theta,␣t)$')
52 axes[0].legend()
53 axes[0].set_title(f'Theory␣vs␣Histogram␣at␣t={t}s')
54 axes[0].grid(True)
55 axes[0].set_ylim(0, 1)
56
57 # Subplot 2: Theory vs Dots
58 axes[1].plot(bin_centers, hist, 'bo', label=f'Simulated␣Dots␣t={t}s',
      markersize=5)
59 axes[1].plot(theta_range, P_theta, 'r-', label=f'Theoretical␣t={t}s',
      linewidth=2)
60 axes[1].set_xlabel(r'$\theta$␣(rad)')
61 axes[1].set_ylabel(r'$P(\theta,␣t)$')
62 axes[1].legend()
63 axes[1].set_title(f'Theory␣vs␣Dots␣at␣t={t}s')
64 axes[1].grid(True)
65 axes[1].set_ylim(0, 1)
66
67 plt.tight_layout()
68 plt.show()
```

Listing 1: Plot of theoretical and simulated $\mathtt{P}(\theta, t = 1s)$

```python
import numpy as np
import matplotlib.pyplot as plt

# Given parameters
timesteps = 1000   # Total number of time steps
dt = 0.01          # Time step
time = np.arange(0, timesteps*dt, dt)
N = 1000           # Number of simulations

# Physical parameters
Dr = 0.17  # Rotational diffusion coefficient (rad^2/s)
Dt = 0.2   # Translational diffusion coefficient
v = 30     # Self-propulsion speed
omega = 10 # Angular velocity (rad/s)

# (b) Simulating theta(t) using Euler-Maruyama method
np.random.seed(42)  # For reproducibility
theta = np.zeros((N, timesteps))
for i in range(N):
    for t in range(1, timesteps):
        theta[i, t] = theta[i, t-1] + omega * dt + np.sqrt(2 * Dr * dt) * np
            .random.randn()

# Extract values at t = 5s
index_t5 = int(5/dt)
theta_t5 = theta[:, index_t5]

# Theoretical probability density function
def theoretical_p(theta_values, t):
    return (1 / np.sqrt(4 * np.pi * Dr * t)) * np.exp(-((theta_values -
        omega * t)**2) / (4 * Dr * t))

# Plot probability distributions
fig, axes = plt.subplots(1, 2, figsize=(12, 5))

# Histogram for t = 5s
theta_bins = np.linspace(45, 55, 50)
hist, bin_edges = np.histogram(theta_t5, bins=theta_bins, density=True)
bin_centers = (bin_edges[:-1] + bin_edges[1:]) / 2

axes[0].hist(theta_t5, bins=theta_bins, density=True, alpha=0.5, label='
    Simulation', edgecolor='black')
theta_range = np.linspace(45, 55, 1000)
axes[0].plot(theta_range, theoretical_p(theta_range, 5), 'r-', label='Theory
    ')
axes[0].set_title("Histogram␣vs␣Theory␣at␣t␣=␣5s")
axes[0].set_xlabel("Theta␣(rad)")
axes[0].set_ylabel("P(Theta,␣t)")
axes[0].set_ylim(0, 1)
axes[0].legend()

# Dots vs Theory for t = 5s
axes[1].plot(bin_centers, hist, 'bo', label='Simulated␣(dots)', markersize
    =5)
```

```
axes[1].plot(theta_range, theoretical_p(theta_range, 5), 'r-', label='Theory', linewidth=2)
axes[1].set_title("Dots vs Theory at t = 5s")
axes[1].set_xlabel("Theta (rad)")
axes[1].legend()
axes[1].grid(True)
axes[1].set_ylim(0, 1)

plt.tight_layout()
plt.show()
```

Listing 2: Plot of theoretical and simulated $\mathtt{P}(\theta, t = 5s)$

```
import numpy as np
import matplotlib.pyplot as plt

# Parameters
D_t = 0.2  # Translational diffusion coefficient
D_r = 0.17  # Rotational diffusion coefficient (rad^2/s)
v = 30  # Self-propulsion speed
omega = 10  # Angular velocity (rad/s)
dt = 0.01  # Time step (s)
T = 10  # Total time (s)
N = int(T / dt)  # Number of time steps
num_trajectories = 5  # Number of trajectories

# Initialize arrays
x = np.zeros((num_trajectories, N))
y = np.zeros((num_trajectories, N))
theta = np.zeros((num_trajectories, N))

# Euler-Maruyama simulation
for i in range(num_trajectories):  # Loop over trajectories
    for j in range(1, N):  # Loop over time steps
        # Wiener increments (random noise)
        dW_theta = np.sqrt(dt) * np.random.normal()  # Noise for theta
        dW_x = np.sqrt(dt) * np.random.normal()  # Noise for x
        dW_y = np.sqrt(dt) * np.random.normal()  # Noise for y

        # Update theta
        theta[i, j] = theta[i, j-1] + omega * dt + np.sqrt(2 * D_r) * dW_theta

        # Update x and y
        x[i, j] = x[i, j-1] + v * np.cos(theta[i, j]) * dt + np.sqrt(2 * D_t) * dW_x
        y[i, j] = y[i, j-1] + v * np.sin(theta[i, j]) * dt + np.sqrt(2 * D_t) * dW_y

# Time array
time = np.arange(0, T, dt)
```

Listing 3: Numerical simulation with $\omega = 10$

```
# Plot x(t) vs time
plt.figure(figsize=(10, 6))
for i in range(num_trajectories):
    plt.plot(time, x[i], label=f'Trajectory {i+1}')
plt.xlabel('Time (s)')
plt.ylabel('x(t) ( m )')
plt.title('x(t) vs Time')
plt.legend()
plt.grid()
plt.savefig('x_vs_time.png')  # Save the plot
plt.show()
```

Listing 4: Plot of horizontal position $x(t)$ for $\omega = 10$

```
# Plot y(t) vs time
plt.figure(figsize=(10, 6))
for i in range(num_trajectories):
    plt.plot(time, y[i], label=f'Trajectory {i+1}')
plt.xlabel('Time (s)')
plt.ylabel('y(t) ( m )')
plt.title('y(t) vs Time')
plt.legend()
plt.grid()
plt.savefig('y_vs_time.png')  # Save the plot
plt.show()
```

Listing 5: Plot of vertical position $y(t)$ for $\omega = 10$

```
# Plot y(t) vs x(t) (2D trajectory)
plt.figure(figsize=(8, 8))
for i in range(num_trajectories):
    plt.plot(x[i], y[i], label=f'Trajectory {i+1}')
plt.xlabel('x(t) ( m )')
plt.ylabel('y(t) ( m )')
plt.title('2D Trajectory: y(t) vs x(t)')
plt.legend()
plt.grid()
plt.savefig('y_vs_x.png')  # Save the plot
plt.show()
```

Listing 6: Plot of 2D trajectory of the particle in the xy-plane

```
# Compute r(t) and dr(t)
r = np.sqrt(x**2 + y**2)  # Distance from origin
# Plot r(t) vs time
plt.figure(figsize=(10, 6))
for i in range(num_trajectories):
    plt.plot(time, r[i], label=f'Trajectory {i+1}')
plt.xlabel('Time (s)')
plt.ylabel('r(t) ( m )')
plt.title('Distance from Origin r(t) vs Time')
plt.legend()
plt.grid()
plt.savefig('r_vs_time.png')  # Save the plot
plt.show()
```

Listing 7: Plot of distance $r(t)$ for $\omega = 10$

```
# Compute dr(t)
dr = np.diff(r, axis=1)  # Change in distance (dr/dt)
# Plot dr(t) vs time
plt.figure(figsize=(10, 6))
for i in range(num_trajectories):
    plt.plot(time[1:], dr[i], label=f'Trajectory {i+1}')
plt.xlabel('Time (s)')
plt.ylabel('dr(t) ( m )')
plt.title('Change in Distance dr(t) vs Time')
plt.legend()
plt.grid()
plt.savefig('dr_vs_time.png')  # Save the plot
plt.show()
```

Listing 8: Plot of change in distance $dr(t)$ for $\omega = 10$

```
import numpy as np
import matplotlib.pyplot as plt

# Parameters
D_t = 0.2  # Translational diffusion coefficient ( m ^2/s)
D_r = 0.17  # Rotational diffusion coefficient (rad^2/s)
v = 30  # Self-propulsion speed ( m /s)
omega = 0  # Angular velocity (rad/s) - set to 0 for non-chiral motion
dt = 0.01  # Time step (s)
T = 10  # Total time (s)
N = int(T / dt)  # Number of time steps
num_trajectories = 5  # Number of trajectories

# Initialize arrays
x = np.zeros((num_trajectories, N))
y = np.zeros((num_trajectories, N))
theta = np.zeros((num_trajectories, N))

```

```
# Euler-Maruyama simulation
for i in range(num_trajectories):  # Loop over trajectories
    for j in range(1, N):  # Loop over time steps
        # Wiener increments (random noise)
        dW_theta = np.sqrt(dt) * np.random.normal()  # Noise for theta
        dW_x = np.sqrt(dt) * np.random.normal()  # Noise for x
        dW_y = np.sqrt(dt) * np.random.normal()  # Noise for y

        # Update theta (no angular velocity, only rotational diffusion)
        theta[i, j] = theta[i, j-1] + omega * dt + np.sqrt(2 * D_r) *
            dW_theta

        # Update x and y
        x[i, j] = x[i, j-1] + v * np.cos(theta[i, j]) * dt + np.sqrt(2 * D_t
            ) * dW_x
        y[i, j] = y[i, j-1] + v * np.sin(theta[i, j]) * dt + np.sqrt(2 * D_t
            ) * dW_y

# Time array
time = np.arange(0, T, dt)
```

Listing 9: Numerical simulation for $\omega = 0$

```
# Plot x(t) vs time
plt.figure(figsize=(10, 6))
for i in range(num_trajectories):
    plt.plot(time, x[i], label=f'Trajectory {i+1}')
plt.xlabel('Time (s)')
plt.ylabel('x(t) ( m )')
plt.title('x(t) vs Time ($\omega = 0$)')
plt.legend()
plt.grid()
plt.savefig('x_vs_time_omega0.png')  # Save the plot
plt.show()
```

Listing 10: Plot of horizontal position (x(t)) of the particle for $\omega = 0$

```
# Plot y(t) vs time
plt.figure(figsize=(10, 6))
for i in range(num_trajectories):
    plt.plot(time, y[i], label=f'Trajectory {i+1}')
plt.xlabel('Time (s)')
plt.ylabel('y(t) ( m )')
plt.title('y(t) vs Time ($\omega = 0$)')
plt.legend()
plt.grid()
plt.savefig('y_vs_time_omega0.png')  # Save the plot
plt.show()
```

Listing 11: Plot of vertical position (y(t)) of the particle for $\omega = 0$

```
# Plot y(t) vs x(t) (2D trajectory)
plt.figure(figsize=(8, 8))
for i in range(num_trajectories):
    plt.plot(x[i], y[i], label=f'Trajectory {i+1}')
plt.xlabel('x(t) ( m )')
plt.ylabel('y(t) ( m )')
plt.title('2D Trajectory: y(t) vs x(t) ($\omega = 0$)')
plt.legend()
plt.grid()
plt.savefig('y_vs_x_omega0.png')  # Save the plot
plt.show()
```

Listing 12: Plot of 2D trajectory of the particle for $\omega = 0$

```

# Compute r(t)
r = np.sqrt(x**2 + y**2)  # Distance from origin

# Plot r(t) vs time
plt.figure(figsize=(10, 6))
for i in range(num_trajectories):
    plt.plot(time, r[i], label=f'Trajectory {i+1}')
plt.xlabel('Time (s)')
plt.ylabel('r(t) ( m )')
plt.title('Distance from Origin r(t) vs Time ($\omega = 0$)')
plt.legend()
plt.grid()
plt.savefig('r_vs_time_omega0.png')  # Save the plot
plt.show()
```

Listing 13: Plot of distance $r(t)$ for $\omega = 0$

```
# Compute dr(t)
dr = np.diff(r, axis=1)  # Change in distance (dr/dt)

# Plot dr(t) vs time
plt.figure(figsize=(10, 6))
for i in range(num_trajectories):
    plt.plot(time[1:], dr[i], label=f'Trajectory {i+1}')
plt.xlabel('Time (s)')
plt.ylabel(''dr(t) ( m )'')
plt.title('Change in Distance dr(t) vs Time ($\omega = 0$)')
plt.legend()
plt.grid()
plt.savefig('dr_vs_time_omega0.png')  # Save the plot
plt.show()
```

Listing 14: Plot of change in distance $r(t)$ as a function of time for $\omega = 0$