An Extension of the Swarmalator Model

Ankith Anil Das
470485327
Faculty of Science, Mathematics
The University of Sydney
Email: aani9804@uni.sydney.edu.au

Synchronization occurs at many natural and technological systems. Such an emergent properties is observed in cardiac pacemaker cells, Japanese tree frogs, colloidal suspensions of magnetic particles, and other biological and technological systems in which synchronization interact. We consider the system where oscillators can sync and swarm. A detailed analysis was proposed by Kevin P. O'Keeffe in the paper "Oscillators the sync and swarm". We studied an extended model of the Swarmalator model proposed in the paper. Understanding the dynamics of this model could possibly give insight to the generalized model where the phase coupling function could be a fourier expansion of a function.

1 Introduction

2 The Model

We consider swarmalators free to move in the plane. The governing equations are

$$\dot{\mathbf{x}}_i = \mathbf{v}_i + \frac{1}{N}\sum_{j=1}^{N}\left[\mathbf{I}_{\mathrm{att}}\left(\mathbf{x}_j - \mathbf{x}_i\right)F\left(\theta_j - \theta_i\right) - \mathbf{I}_{\mathrm{rep}}\left(\mathbf{x}_j - \mathbf{x}_i\right)\right] \tag{1}$$

$$\dot{\theta}_i = \omega_i + \frac{K}{N}\sum_{j=1}^{N} H_{\mathrm{att}}\left(\theta_j - \theta_i\right)G\left(\mathbf{x}_j - \mathbf{x}_i\right) \tag{2}$$

for $i = 1,...,N$, where N is the number of swarmalators, $\mathbf{x}_i$ is the position of the $i-$th swarmalator, and θ_i, ω_i, and $\mathbf{v}_i$ are it's phase, natural frequency and background velocity. The functions $\mathbf{I}_{att}$ and $\mathbf{I}_{rep}$ represent spatial attraction and repulsion between the swarmalators where as phase interaction is governed by $\mathbf{H}_{att}$. We considered the following model:

$$\dot{\mathbf{x}}_i = \mathbf{v}_i + \frac{1}{N}\left[\sum_{j\neq i}^{N}\frac{\mathbf{x}_j - \mathbf{x}_i}{\left|\mathbf{x}_j - \mathbf{x}_i\right|}\left(1 + J\cos\left(\theta_j - \theta_i\right)\right) - \frac{\mathbf{x}_j - \mathbf{x}_i}{\left|\mathbf{x}_j - \mathbf{x}_i\right|^2}\right] \tag{3}$$

$$\dot{\theta}_i = \omega_i + \frac{K}{N}\sum_{j\neq i}^{N}\frac{\gamma_1 \sin\left(\theta_j - \theta_i\right) + \gamma_2 \sin\left(2\left(\theta_j - \theta_i\right)\right)}{\left|\mathbf{x}_j - \mathbf{x}_i\right|} \tag{4}$$

We considered identical swarmalators so that $\omega_i = \omega$ and $\mathbf{v}_i = \mathbf{v}$. Using this assumption, using a suitable choice of reference frame we can set $\omega = 0$, and $\mathbf{v} = \mathbf{0}$. The system has four parameters $(J, K, \gamma_1, \gamma_2)$.

The parameter J measures the extend to which phase similarity enhances spatial attraction. For $J > 0$, swarmalators prefer to be near other swarmalators with similar phase. When $J < 0$, the opposite behavior is observed: swarmalators attract those with opposite phase. When $J = 0$, they show no phase based spatial behavior, i.e, their spatial attraction is independent of phase. To maintain $\mathbf{I}_{att} > 0$, we constrain J to $-1 \leq J \leq 1$. The parameter K is the phase coupling strength. It scales γ_1, and γ_2 parameters. The relative strengths of γ_1 and γ_2 determine the stability of one or two clusters.

3 Optimization of the ode solver

The simulations were run using MATLAB's ODE integrator 'ode45'. Absolute and Relative tolerance for the integrator was set to 10^{-6}. Before large computations were performed, optimization of the existing code was necessary. As this project involves computations of large matrices using ode45, reducing function call overheads would greatly reduce the computation time. The initial code was profiled using MATLAB's performance profiler and bottlenecks were identified. Calculating the pairwise inverse distance was the most expensive in the ode function. Simulations were run for $N = 100$ and for $T = 10$ time units and the original code took 70 seconds. This was too slow for any long time simulations. After removing unwanted function calls, changing the algorithm, and using functions which supports vectorization, the new code took 0.509 sec for the same task. The performance was good enough for a user interactive simulation where the user can change the parameters of the system and can see the results almost instantly.

4 Single Cluster Stability