Oscillator-Based Potts Machine (OPM) for the Implementation of the Vector Potts Model

Jaijeet Roychowdhury
Sayan Seal

Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2022-198
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2022/EECS-2022-198.html

August 11, 2022

Oscillator-Based Potts Machine (OPM) for the Implementation of the Vector Potts Model

by

Sayan Seal

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in partial satisfaction of the requirements for the degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

Committee:

Professor Jaijeet Roychowdhury
Research Advisor

Aug 10, 2022

(Date)

* * * * * * *

Laura Waller

Professor Laura Waller
Second Reader

8/11/22

(Date)

Oscillator-Based Potts Machine (OPM) for the Implementation of the Vector Potts Model

Jaijeet Roychowdhury and Sayan Seal
Department of Electrical Engineering and Computer Sciences, The University of California, Berkeley, CA, USA
Emails: jr@berkeley.edu, sayan_seal@berkeley.edu

***Abstract*—An approach towards the implementation of the Vector Potts Model using a network of coupled nonlinear oscillators has been presented in this technical report. The oscillator systems, under the influence of N-SHIL (Sub-Harmonic Injection Locking), show phase dynamics that have an underlying Lyapunov function, analogous to the Vector Potts Hamiltonian with N states. The key concept used here is that there are N equally spaced stable locks under the influence of N-SHIL, which has been shown using the Stability Theorem and linearization. The coupled oscillator network tends to minimize the Lyapunov function naturally over time, indicating the minimization of the corresponding Hamiltonian. Global minimum Hamiltonians of the Vector Potts problems can be obtained by adding appropriate amounts of noise to this system, as well as smoothly switching SHIL on and off multiple times. The proposed method has been applied on several examples of random graphs, that have been generated using the rudy graph generator, for assessing performance and demonstrating effectiveness.**

Keywords: Vector Potts Model, Coupled oscillators, OPM, N-SHIL, Kuramoto equation, Lyapunov function, Stable locks.

I. Introduction

The Vector Potts Model [1] plays a significant role in statistical mechanics and is a generalization of the Ising Model [2, 3]. It consists of a number of discrete variables called *spins* placed on a lattice, that can take one of N possible states uniformly distributed about a circle, *i.e.*, an angle of 2π. The model has an associated energy function known as the Vector Potts Hamiltonian. The aim of the Vector Potts problem is to assign values to the Vector Potts spins in such a manner that the associated Hamiltonian is minimized. This model is also known as the clock model and is extremely interesting due to its role in the study of phase transitions and other important concepts in solid-state physics.

In this technical report, we present a novel approach for implementing the Vector Potts model using coupled nonlinear oscillator networks. At first, the theoretical concepts that map the phase dynamics shown by an oscillator system to the discrete Vector Potts Hamiltonian of the corresponding coupling graph is studied. After that, the concepts are applied to realize the Oscillator-Based Potts Machine (OPM). The developed OPM is finally applied on some simple as well as complex random graphs generated using the rudy graph generator [4] to show the efficacy of the scheme.

We first define the Vector Potts Hamiltonian and then demonstrate the theory behind OPM. Initially, we modify the Perturbation Projection Vector (PPV) [5, 6] equation for oscillators and define it in terms of the phase differences of the oscillators. Then, the modified PPV equation is used to derive the generalized Kuramoto equation for oscillators [7–9] in the presence of a N^{th} harmonic SYNC input that induces N-SHIL (Sub-Harmonic Injection Locking) to the oscillator network. N-SHIL causes the phase difference values of the oscillators to settle at or near one of N distinct equispaced stable values in the range $[0,1)$ [10], given by $\frac{k}{2N}$ $(k = 1,3,\cdots,2N-1)$. Moreover, the Kuramoto model, in the presence of N-SHIL, has an underlying Lyapunov function which is non-increasing with time [11]. This Lyapunov function governs the oscillator network's dynamics and is identical to the Vector Potts Hamiltonian of the network coupling graph at the discrete stable phase difference values of $\frac{k}{2N}$ $(k = 1,3,\cdots,2N-1)$.

Therefore, we establish that the dynamics of the oscillator networks, under the influence of N-SHIL, evolve naturally to minimize the associated Lyapunov function, signifying the minimization of the Vector Potts Hamiltonian, and find good solutions to the Vector Potts problems. Appropriate amounts of noise can be added to the system, along with varying the coupling parameters with time such as switching the Sub-Harmonic Injection Locking (SHIL) on and off several times, so that phase dynamics converge to the global minimum instead of a local minimum.

We demonstrate the effectiveness of our method with the help of some random graphs, generated using the rudy graph generator [4] and considering different values of N. In each of these simulated cases, with the appropriate choice of the parameters, the solution converges to the discrete stable values, as expected, and is able to find the global minimum solution, leading to a globally minimized Vector Potts Hamiltonian of the corresponding network coupling graph.

The fact that the global minimum value is actually

obtained, has also been verified with the help of a separate program that uses the Brute Force approach to find the minimum Hamiltonian, by computing the Hamiltonian for each permutation of the Potts states, and comparing the computed values to find the minimum.

Over the years, research has been conducted on the Vector Potts Model and its use in lattice statistics, and the special case of $N = 2$ (Ising Model) has received recognition in particular for its relatively simpler nature and easier implementation. Numerous methods for the physical implementation of the Ising Model have been proposed; but limited research has been conducted on the physical realization of the general case of the Vector Potts Model.

The D-Wave Systems use a quantum computing approach [12, 13] by taking superconducting loops as spins and Josephson junction devices as connections, and can use the concept of quantum tunneling for increasing performance. However, the machines are unable to function at room temperature and require costly cooling mechanisms. There are other non-quantum implementations of the Ising Model that can operate at room temperature. One such method makes the use of laser pulses traveling on long optical fibers and FPGA mechanism [14–16]. These machines, though more compact than the quantum Ising machines, may face integration and miniaturization issues because of the use of long fibers.

Ising machines have also been implemented using digital circuits such as CMOS SRAM cells as spins and digital logic gates as means of coupling [17], as well as networks of self-sustaining coupled oscillators [18]. The method described in [17] does not guarantee a global optimal energy state due to variability.

All the methods mentioned above focus only on the realization of the $N = 2$ special case of the Vector Potts Model, and not on implementing the general case. The optical implementation of a three-state Potts model, by using three-photon down-conversion oscillators, has been proposed in [19]. Our scheme can be used for classically implementing the generalized version of the Vector Potts Model using interesting inherent physics.

In the rest of the report, we first define the Vector Potts Model and its associated Hamiltonian in Sec. II. Then, in Sec. III, we demonstrate the general theory behind Oscillator-Based Potts Machine (OPM), *i.e.*, we explain the mapping between the Lyapunov function associated with the phase dynamics of the coupled nonlinear oscillator network and the Vector Potts Hamiltonian. Finally, we show the efficacy and validity of the proposed scheme with the help of randomly generated graphs of different sizes, generated using the rudy graph generator [4] in Sec. IV.

II. Vector Potts Model

The Vector Potts Model [1], also known as the Clock Model and the Planar Potts Model, is used for studying the behavior of ferromagnets and some other important concepts in solid-state physics. It is a generalization of the famous Ising model [2, 3] and was named after the Australian mathematician, Renfrey Potts. The model was explained by him in his Ph.D. thesis in 1951 and suggested to him by his advisor, Cyril Domb. This model consists of n discrete variables called *spins* $\{s_i\}$ that are placed on a lattice (generally a 2d Euclidean lattice), the values of which are uniformly distributed on a circle, at angles given by

$$\theta_{s_i} = \frac{2\pi s_i}{N}, \tag{1}$$

where $s_i \in \{0, 1, \cdots, N-1\}$. The associated Vector Potts interaction Hamiltonian is given by

$$H = \sum_{i,j,i<j} J_{ij} \cos(\theta_{s_i} - \theta_{s_j}), \tag{2}$$

where J_{ij} represents the coupling between the Vector Potts spins s_i and s_j. The spins take up values in such a manner that the Hamiltonian is minimized. The Vector Potts Model is an important model in statistical mechanics for studying phase transitions.

III. Oscillator-Based Potts Machine

In this section, we demonstrate the basic theory behind the implementation of the Vector Potts Model using a network of coupled nonlinear oscillators. At first, we formulate the Perturbation Projection Vector (PPV) equation for oscillators in terms of the phase difference in Sec. III-A. Using this equation, the general Kuramoto equation for oscillators, under the influence of N-SHIL (Sub-Harmonic Injection Locking), has been derived in Sec. III-B. In Sec. III-C, the Lyapunov function for the general Kuramoto Equation (derived in Sec. III-B) has been shown. Under the influence of N-SHIL, there are N equally spaced stable locks $\in [0, 1)$ for the phase differences of the oscillators, which have been computed in Sec. III-D. Finally, using these stable solutions, the equivalence of the Lyapunov function, shown in Sec. III-C, and the Vector Potts Hamiltonian (2) has been established in Sec. III-E.

A. PPV equation for oscillators in terms of phase difference

The Differential Algebraic Equation (DAE) for a circuit involving oscillators [20] is given by

$$\frac{d}{dt}\vec{q}(\vec{x}(t)) + \vec{f}(\vec{x}(t)) + \vec{b}(t) = \vec{0}, \tag{3}$$

where $\vec{x}$ represents the internal state, $\vec{f}(\cdot)$ and $\vec{q}(\cdot)$ represent the static and time-varying terms respectively, and $\vec{b}(t)$ represents a small perturbation to the system. If $\vec{b}(t) = 0$, the self-sustaining oscillators produce a T_0-periodic natural oscillation denoted by $\vec{x}_s(t)$ such that the natural frequency is given by

$$f_0 = \frac{1}{T_0}. \tag{4}$$

Using (4), it can be written that

$$\vec{x}_s(t) = \vec{x}_{1s}(f_0 t), \tag{5}$$

where $\vec{x}_{1s}(t)$ is a 1-periodic oscillation.

The Perturbation Projection Vector (PPV) equation for the oscillator system under the influence of $\vec{b}(t)$ is given by

$$\frac{d}{dt}\alpha(t) = \vec{p}^{\mathrm{T}}(t+\alpha(t))\cdot\vec{b}(t), \tag{6}$$

where $\vec{p}(t)$ is known as the Perturbation Projection Vector (PPV) [6] or the Phase Response Curve (PRC) [21], which is a T_0-periodic function of time and $\alpha(t)$ represents the time-shift caused by the external perturbation. Under such conditions, the oscillator's response, as has been shown in [5], can be written as

$$\begin{aligned}\vec{x}(t) &= \vec{x}_s(t+\alpha(t)) + \vec{y}(t)\\ &= \vec{x}_{1s}(f_0(t+\alpha(t))) + \vec{y}(t),\end{aligned} \tag{7}$$

where $\vec{y}(t)$ (deviation from the natural oscillation) is small if $\vec{b}(t)$ is small.

It is assumed that the perturbation $\vec{b}(t)$ is of the form

$$\vec{b}(t) = \vec{b}_1(f^*t + \phi_{in}(t)), \tag{8}$$

where $\vec{b}_1(\cdot)$ is a 1-periodic function, f^* is some nominal input frequency[1] and $\phi_{in}(t)$ is some input phase that is suitable for 1-periodic functions.

Let the phase of the oscillator $\phi(t)$ be defined as

$$\phi(t) = f_0 t + f_0\alpha(t). \tag{9}$$

Using (9), the following is true

$$\vec{x}_s(t+\alpha(t)) = \vec{x}_s(\frac{\phi(t)}{f_0}). \tag{10}$$

Let the phase difference between the nominal input phase and the oscillator response phase be defined as

$$\Delta\phi(t) = \phi(t) - f^*t. \tag{11}$$

Therefore, using (9) and (11), we get

$$\phi(t) = f_0(t+\alpha(t)) = f^*t + \Delta\phi(t) \tag{12}$$

$$\Rightarrow (t+\alpha(t)) = \frac{f^*t+\Delta\phi(t)}{f_0} \tag{13}$$

$$\begin{aligned}\Rightarrow \vec{x}_s(t+\alpha(t)) &= \vec{x}_s(\frac{f^*t+\Delta\phi(t)}{f_0})\\ &= \vec{x}_{1s}(f^*t+\Delta\phi(t)).\end{aligned} \tag{14}$$

Rewriting (6) in terms of $\Delta\phi(t)$ using (13),

$$\alpha(t) = \frac{f^*t+\Delta\phi(t)}{f_0} - t \tag{15}$$

$$\Rightarrow \frac{d}{dt}\alpha(t) = \frac{f^*-f_0}{f_0} + \frac{1}{f_0}\frac{d}{dt}\Delta\phi(t) \tag{16}$$

$$\begin{aligned}\Rightarrow \frac{d}{dt}\Delta\phi(t) = f_0 - f^* + f_0\vec{p}(\frac{f^*t+\Delta\phi(t)}{f_0})\cdot\vec{b}_1(f^*t\\ +\phi_{in}(t)).\end{aligned} \tag{17}$$

Let the 1-periodic version of $\vec{p}(t)$ in (6), denoted by $\vec{p}_1(t)$, be defined as

$$\vec{p}_1(t) \triangleq \vec{p}(tT_0). \tag{18}$$

Using (18) in (17), we obtain the PPV equation in terms of $\Delta\phi(t)$, given by

$$\begin{aligned}\frac{d}{dt}\Delta\phi(t) = f_0 - f^* + f_0\vec{p}_1(f^*t+\Delta\phi(t))\cdot\vec{b}_1(f^*t\\ +\phi_{in}(t)).\end{aligned} \tag{19}$$

B. General Kuramoto equation for oscillators in the presence of N-SHIL

Let there be n coupled oscillators with phases denoted by $\phi_i(t)$ and averaged phases denoted by $\phi_{a,i}(t)$ such that $\phi_{a,i}(t) \simeq \phi_i(t) (i = 1,\cdots,n)$. The oscillators have identical steady-state waveforms (1-periodic forms) denoted by $\vec{x}_{1s}(t)$ given by (5). However, their frequencies f_i can be different, *i.e.*,

$$\vec{x}_{s,i}(t) = \vec{x}_{1s}(f_i t). \tag{20}$$

Moreover, the same nominal input frequency f^* is assumed for all the oscillators (though not a restriction since individual oscillator features are captured in $\phi_i(t)$). From (7), the perturbed response of the i^{th} oscillator (ignoring the $y(t)$ small orbital deviation) is

$$\begin{aligned}\vec{x}_i(t) &= \vec{x}_{1s}(f_i(t+\alpha_i(t)))\\ &= \vec{x}_{1s}(f^*t+\Delta\phi_i(t)). \qquad \text{(using (14))}\end{aligned} \tag{21}$$

The input to the i^{th} oscillator is a weighted sum of the outputs of all the other oscillators, added with a N^{th} harmonic SYNC input, and is given by

$$\begin{aligned}\vec{b}_i(t) = \sum_{\substack{j=1\\ j\neq i}}^{n}\beta_{ij}\vec{x}_{1s}(f^*t+\Delta\phi_i(t))\\ +\vec{K}_s\cos(2\pi N(f^*t+\phi_s(t))),\end{aligned} \tag{22}$$

where $\vec{K}_s$ is the vector version of SYNC coupling and $\phi_s(t)$ is the phase of SYNC input. We assume the same 1-periodic PPV for all the oscillators such that

$$\vec{p}_{1,i}(t) \equiv \vec{p}_1(t). \tag{23}$$

From (19), the generalized Kuramoto equation [7–9] with SYNC input $\vec{K}_s\cos(2\pi N(f^*t+\phi_s(t)))$ can be derived, which is given by

$$\begin{aligned}\frac{d}{dt}\Delta\phi_i(t) = f_i - f^* + f_i\sum_{\substack{j=1\\ j\neq i}}^{n}\beta_{ij}\,g_c(\Delta\phi_i(t)-\Delta\phi_j(t))\\ +f_i\,g_s(\Delta\phi_i(t)-\phi_s(t))\end{aligned} \tag{24}$$

$$\begin{aligned}\text{with}\quad g_c(\tau) &\triangleq \int_0^1 \vec{p}_1(t+\tau)\cdot\vec{x}_{1s}(t)dt\\ &= \sum_{l=-\infty}^{\infty}\vec{p}_{1,l}\cdot\vec{x}_{1s,-l}e^{j2\pi l\tau},\end{aligned} \tag{25}$$

$$\begin{aligned}\text{and}\quad g_s(\tau) &\triangleq \int_0^1 \vec{p}_1(t+\tau)\cdot\vec{K}_s\cos(2\pi Nt)dt\\ &= \frac{1}{2}[\vec{p}_{1,N}e^{j2\pi N\tau} + \vec{p}_{1,-N}e^{-j2\pi N\tau}]\cdot\vec{K}_s,\end{aligned} \tag{26}$$

where $\vec{p}_{1,l}$ and $\vec{x}_{1s,-l}$ are the Fourier coefficients of $\vec{p}_1(\cdot)$ and $\vec{x}_{1s}(\cdot)$ respectively, and $g_c(\cdot)$ and $g_s(\cdot)$ are general 1-periodic functions.

Let there be the following special conditions:

$$f_i = f_0\,\forall\, i, \tag{27}$$

$$f^* = f_0, \tag{28}$$

[1]For FHIL, we usually assume locking to f^*, but this is not a strict rule. For *e.g.*, locking can be done to $\frac{f^*}{N}$ for N-SHIL.

$$\vec{x}_{1s}(t) = \vec{X}_{1s}\cos(2\pi t) \quad \text{(where } \vec{X}_{1s} \text{ is real)}, \tag{29}$$

$$\phi_s(t) \equiv 0, \quad \text{and} \tag{30}$$

$$\vec{p}_1(t) = \vec{a}_0 + \vec{a}_1\sin(2\pi t) + \vec{a}_2\sin(2\pi\cdot 2t) + \cdots + \vec{a}_N\sin(2\pi\cdot Nt) + \cdots, \quad \text{(where } \vec{a}_i \forall\, i \text{ are real.)}. \tag{31}$$

Using the conditions (27)-(31) and (23), $g_c(\tau)$ from (25) becomes

$$g_c(\tau) = \frac{\vec{a}_1\cdot\vec{x}_{1s}}{2}\sin(2\pi\tau). \tag{32}$$

Similarly, $g_s(\tau)$ from (26) becomes

$$g_s(\tau) = \frac{1}{2}\vec{a}_N\cdot\vec{K}_s\sin(2\pi N\tau). \tag{33}$$

Using (32) and (33), the Kuramoto equation (24) becomes

$$\frac{d}{dt}\Delta\phi_i(t) = f_0\sum_{\substack{j=1\\ j\neq i}}^{n}\beta_{ij}\frac{\vec{a}_1\cdot\vec{x}_{1s}}{2}\sin(2\pi(\Delta\phi_i(t)-\Delta\phi_j(t))) + f_0\frac{\vec{a}_N\cdot\vec{K}_s}{2}\sin(2\pi N\Delta\phi_i(t)) \tag{34}$$

$$\Rightarrow \frac{d}{dt}\Delta\phi_i(t) = K\sum_{\substack{j=1\\ j\neq i}}^{n}\beta_{ij}\sin(2\pi(\Delta\phi_i(t)-\Delta\phi_j(t))) + K_s\sin(2\pi N\Delta\phi_i(t)), \tag{35}$$

where $K \triangleq f_0\frac{\vec{a}_1\cdot\vec{x}_{1s}}{2}$ and $K_s \triangleq f_0\frac{\vec{a}_N\cdot\vec{K}_s}{2}$.

C. Lyapunov function corresponding to the Kuramoto equation

Let $\overrightarrow{\Delta\phi}(t)$ be defined as

$$\overrightarrow{\Delta\phi}(t) \triangleq [\Delta\phi_1(t), \Delta\phi_2(t), \cdots, \Delta\phi_n(t)]^{\mathrm{T}}. \tag{36}$$

Claim: The global Lyapunov function for (35) can be written as

$$E(\overrightarrow{\Delta\phi}) = \frac{N}{2}K\sum_{\substack{i,j\\ j\neq i}}^{n}\beta_{ij}\cos(2\pi(\Delta\phi_i-\Delta\phi_j)) + K_s\sum_{i=1}^{n}\cos(2\pi N\Delta\phi_i), \tag{37}$$

where $\overrightarrow{\Delta\phi}$ is given by (36).

Proof: Being a global Lyapunov function, the coupled oscillator system should tend to minimize (37) as it evolves over time [11], *i.e.*, $\frac{d}{dt}(E(\overrightarrow{\Delta\phi}(t)))$ must be less than or equal to 0.

$\frac{d}{dt}(E(\overrightarrow{\Delta\phi}(t)))$ is given by

$$\frac{d}{dt}(E(\overrightarrow{\Delta\phi}(t))) = \sum_{k=1}^{n}[\frac{\partial(E(\overrightarrow{\Delta\phi}))}{\partial(\Delta\phi_k)}\Big|_{\Delta\phi_k(t)}\cdot\frac{d}{dt}(\Delta\phi_k(t))]. \tag{38}$$

Therefore, in order to show that (37) is indeed a global Lyapunov function, E needs to be differentiated first with respect to $\overrightarrow{\Delta\phi}$. It can be observed that the first component of E in (37) consists of (n^2-n) number of $\cos(\cdot)$ terms. For any index k, $\Delta\phi_k$ appears $(n-1)$ times as the subtrahend inside $\cos(\cdot)$; these terms are given by $\beta_{kl}\cos(2\pi(\Delta\phi_k-\Delta\phi_l))$ $(l=1,\cdots,n$ and $l\neq k)$, and another $(n-1)$ times as the minuend inside $\cos(\cdot)$; these terms are given by $\beta_{lk}\cos(2\pi(\Delta\phi_l-\Delta\phi_k))$ $(l=1,\cdots,n$ and $l\neq k)$.

So, $\frac{\partial(E(\overrightarrow{\Delta\phi}))}{\partial(\Delta\phi_k)}$ can be written as

$$\frac{\partial(E(\overrightarrow{\Delta\phi}))}{\partial(\Delta\phi_k)} = \frac{N}{2}K\sum_{\substack{l=1\\ l\neq k}}^{n}\beta_{kl}\frac{\partial}{\partial(\Delta\phi_k)}[\cos(2\pi(\Delta\phi_k-\Delta\phi_l))] + \frac{N}{2}K\sum_{\substack{l=1\\ l\neq k}}^{n}\beta_{lk}\frac{\partial}{\partial(\Delta\phi_k)}[\cos(2\pi(\Delta\phi_l-\Delta\phi_k))] + K_s\frac{\partial}{\partial(\Delta\phi_k)}(\cos(2\pi N\Delta\phi_k)) \tag{39}$$

$$= -\frac{N}{2}K\sum_{\substack{l=1\\ l\neq k}}^{n}\beta_{kl}\sin(2\pi(\Delta\phi_k-\Delta\phi_l))\cdot 2\pi + \frac{N}{2}K\sum_{\substack{l=1\\ l\neq k}}^{n}\beta_{lk}\sin(2\pi(\Delta\phi_l-\Delta\phi_k))\cdot 2\pi - K_s\sin(2\pi N\Delta\phi_k)\cdot 2\pi N \tag{40}$$

$$= -2\pi N[\frac{1}{2}K\sum_{\substack{l=1\\ l\neq k}}^{n}2\beta_{kl}\sin(2\pi(\Delta\phi_k-\Delta\phi_l)) + K_s\sin(2\pi N\Delta\phi_k)] \quad \text{(using } \beta_{kl}=\beta_{lk}\forall k,l \text{ and } \sin(-x)=-\sin(x)) \tag{41}$$

$$= -2\pi N\frac{d}{dt}\Delta\phi_k(t). \quad \text{(using (35))} \tag{42}$$

Note that in (39), terms not involving k at all have been dropped since $\frac{\partial}{\partial(\Delta\phi_k)} = 0$ for those terms.

Substituting (42) in (38), we get

$$\frac{d}{dt}(E(\overrightarrow{\Delta\phi}(t))) = \sum_{k=1}^{n}[\frac{\partial(E(\overrightarrow{\Delta\phi}))}{\partial(\Delta\phi_k)}\Big|_{\Delta\phi_k(t)}\cdot\frac{d}{dt}(\Delta\phi_k(t))] = -2\pi N\sum_{k=1}^{n}(\frac{d}{dt}\Delta\phi_k(t))^2 \leq 0. \tag{43}$$

Hence, (37) is a global Lyapunov function that the coupled oscillators minimize over time.

D. Computing stable locks

When $K=0$, (35) reduces to

$$\frac{d}{dt}\Delta\phi_i(t) = f(\Delta\phi_i) = K_s\sin(2\pi N\Delta\phi_i),\ i=1,\cdots,n. \tag{44}$$

The equilibrium solutions of (44) can be computed using the Stability Theorem and used to accurately predict the injection-locked states under N-SHIL. The equilibrium equation for (44) can be written as

$$\frac{d}{dt}\Delta\phi_i(t) = 0, \qquad i=1,\cdots,n \tag{45}$$

$$\Rightarrow K_s\sin(2\pi N\Delta\phi_i) = 0 \tag{46}$$

$$\Rightarrow 2\pi N\Delta\phi_i = k\pi, \qquad k=0,1,\cdots \tag{47}$$

$$\Rightarrow \Delta\phi_i = \frac{k}{2N}. \tag{48}$$

Now, there are $2N$ distinct solutions for (48) in the interval $\Delta\phi_i \in [0,1)$, given by $\Delta\phi_k^* = \frac{k}{2N}$ $(k = 0,1,\cdots,2N-1)$.

Of these $2N$ solutions, if $k = 2k^{'}$ $(k^{'} = 0,1,\cdots,N-1), \Delta\phi_k^* = \frac{k}{2N} = \frac{k^{'}}{N}$. Therefore, using (44),

$$\begin{aligned} f^{'}(\Delta\phi_i)\Big|_{\Delta\phi_k^*} &= K_s \cdot 2\pi N \cos(2\pi N \Delta\phi_k^*) \\ &= K_s \cdot 2\pi N \cos(2\pi k^{'}) \\ &= K_s \cdot 2\pi N \quad [\text{using } \cos(2\pi k^{'}) = 1] \\ &> 0. \end{aligned} \tag{49}$$

Again, if $k = 2k^{'}+1$ $(k^{'} = 0,1,\cdots,N-1), \Delta\phi_k^* = \frac{k}{2N} = \frac{2k^{'}+1}{2N}$. Therefore, using (44),

$$\begin{aligned} f^{'}(\Delta\phi_i)\Big|_{\Delta\phi_k^*} &= K_s \cdot 2\pi N \cos(2\pi N \Delta\phi_k^*) \\ &= K_s \cdot 2\pi N \cos(\pi(2k^{'}+1)) \\ &= -K_s \cdot 2\pi N \quad [\text{using } \cos(\pi(2k^{'}+1)) = -1] \\ &< 0. \end{aligned} \tag{50}$$

Applying Taylor series approximation in (44) about $\Delta\phi_k^* = \frac{k}{2N}$ $(k = 0,1,\cdots,2N-1)$ and neglecting all higher order (higher than 1^{st} order) terms,

$$\frac{d}{dt}\Delta\phi_i(t) = f(\Delta\phi_i(t)) = f(\Delta\phi_k^*) + f^{'}(\Delta\phi_k^*) \cdot \delta\phi_i(t) \tag{51}$$

$$\begin{aligned} &(\text{where } \Delta\phi_i(t) = \Delta\phi_k^* + \delta\phi_i(t);\ i = 1,\cdots,n) \\ &= 0 + f^{'}(\Delta\phi_k^*) \cdot \delta\phi_i(t) \\ &\qquad (\text{using (46)}) \end{aligned} \tag{52}$$

$$= f^{'}(\Delta\phi_k^*) \cdot \delta\phi_i(t). \tag{53}$$

Now, if $\Delta\phi_k^* = \frac{k}{2N}$ $(k = 0,2,\cdots,2N-2)$, $f^{'}(\Delta\phi_k^*) > 0$ (using (49)). Therefore,

$$\begin{aligned} &\delta\phi_i(t) > 0 \qquad (\text{right neighborhood}) \\ &\Rightarrow f^{'}(\Delta\phi_k^*) \cdot \delta\phi_i(t) > 0 \\ &\Rightarrow \frac{d}{dt}\Delta\phi_i(t) = f(\Delta\phi_i(t)) > 0. \qquad (\text{using (53)}) \end{aligned} \tag{54}$$

$$\begin{aligned} &\text{and } \delta\phi_i(t) < 0 \qquad (\text{left neighborhood}) \\ &\Rightarrow f^{'}(\Delta\phi_k^*) \cdot \delta\phi_i(t) < 0 \\ &\Rightarrow \frac{d}{dt}\Delta\phi_i(t) = f(\Delta\phi_i(t)) < 0. \qquad (\text{using (53)}) \end{aligned} \tag{55}$$

Using (54) and (55), it can be seen that at $\Delta\phi_k^* = \frac{k}{2N}$ $(k = 0,2,\cdots,2N-2)$, over time, the neighboring solutions move away from the equilibrium and hence these $\Delta\phi_k^*$ are unstable solutions to (45).

Again, if $\Delta\phi_k^* = \frac{k}{2N}$ $(k = 1,3,\cdots,2N-1)$, $f^{'}(\Delta\phi_k^*) < 0$ (using (50)). Therefore,

$$\begin{aligned} &\delta\phi_i(t) > 0 \qquad (\text{right neighborhood}) \\ &\Rightarrow f^{'}(\Delta\phi_k^*) \cdot \delta\phi_i(t) < 0 \\ &\Rightarrow \frac{d}{dt}\Delta\phi_i(t) = f(\Delta\phi_i(t)) < 0. \qquad (\text{using (53)}) \end{aligned} \tag{56}$$

$$\begin{aligned} &\text{and } \delta\phi_i(t) < 0 \qquad (\text{left neighborhood}) \\ &\Rightarrow f^{'}(\Delta\phi_k^*) \cdot \delta\phi_i(t) > 0 \\ &\Rightarrow \frac{d}{dt}\Delta\phi_i(t) = f(\Delta\phi_i(t)) > 0. \qquad (\text{using (53)}) \end{aligned} \tag{57}$$

Using (56) and (57), it can be seen that at $\Delta\phi_k^* = \frac{k}{2N}$ $(k = 1,3,\cdots,2N-1)$, over time, the neighboring solutions move towards the equilibrium and hence these $\Delta\phi_k^*$ are stable solutions to (45).

So, it can be seen that there are exactly N equally spaced stable locks under the influence of N-SHIL [10] due to the K_s (SYNC) terms in (45) such that $\Delta\phi_k^* = \frac{k}{2N}$ $(k = 1,3,\cdots,2N-1)$.

When $K_s >> K$, the coupling terms represent only small perturbations to the locks; hence $\frac{k}{2N}$ $(k = 1,3,\cdots,2N-1)$ remain good approximation to the stable locks.

E. Equivalence of the Lyapunov function and the Vector Potts Hamiltonian

At the discrete points where $\Delta\phi_k^* = \frac{k}{2N}$ $(k = 1,3,\cdots,2N-1)$,

$$\cos(2\pi N \Delta\phi_k^*) \equiv -1. \tag{58}$$

Using (58), (37) reduces to

$$\begin{aligned} E(\overrightarrow{\Delta\phi}) &= \frac{N}{2} K \sum_{\substack{i,j \\ j\neq i}}^{n} \beta_{ij} \cos(2\pi(\Delta\phi_i - \Delta\phi_j)) - nK_s \\ &(\text{where } \Delta\phi_i = \Delta\phi_{k_i}^* = \frac{k_i}{2N} \text{ and } \Delta\phi_j = \Delta\phi_{k_j}^* = \frac{k_j}{2N}; \\ &\qquad k_i, k_j = 1,3,\cdots,2N-1) \end{aligned} \tag{59}$$

$$\begin{aligned} &= \frac{N}{2} K \cdot 2 \sum_{\substack{i,j \\ i<j}}^{n} \beta_{ij} \cos(2\pi(\Delta\phi_i - \Delta\phi_j)) - nK_s \\ &\qquad (\text{using } \beta_{ij} = \beta_{ji}\ \forall i,j) \end{aligned} \tag{60}$$

$$= NK \sum_{\substack{i,j \\ i<j}}^{n} \beta_{ij} \cos(2\pi(\Delta\phi_i - \Delta\phi_j)) - nK_s, \tag{61}$$

where nK_s is a constant.

Hence, it can be observed that (61) is a scaled version of the discrete Vector Potts Hamiltonian (2) with a constant offset and $\beta_{ij} = J_{ij}\ \forall i,j$. Moreover, for (2) to be equivalent to (61), θ_k defined in (1) becomes

$$\begin{aligned} \theta_k &= \frac{2\pi k}{2N} \qquad (k = 1,3,\cdots,2N-1) \\ &= \frac{\pi k}{N}. \end{aligned} \tag{62}$$

Therefore, we have successfully established that a network of self-sustaining, coupled oscillators, under the influence of N-SHIL, can be used for realizing the Vector Potts model.

It is to be noted that the Lyapunov function (37) will have several local minima, and the solutions obtained by OPM are not always guaranteed to settle at or near the global minimum. But, the introduction of noise via K_n (a noise level parameter), and smoothly varying the parameter K_s with time can increase the chances of the solutions to settle at or near the global minimum.

IV. Results

In this section, we apply OPM to several randomly generated graphs, generated using the rudy graph generator [4] in order to confirm the validity of the scheme and to assess performance. Problems using random graphs are generally tougher to solve than practical problems, since practical problems are subjected to physical constraints and can be simpler. Problems with only positive weights tend to be trivially easy, for which we have considered graphs with ± 1 edge weights. At first, we demonstrate how OPM can solve a special case of the Vector Potts problem with 2 states per spin, *i.e.*, the Ising problem, with the help of a simple example, in Sec. IV-A. In this case, the Oscillator-Based Potts Machine serves as the Oscillator-Based Ising Machine [18]. Next, we show that OPM can solve a Potts problem with 3 states in Sec. IV-B. Also, the significance of ramping SYNC up and down is explained in this subsection. In Sec. IV-C, we use OPM to find the global minimum Hamiltonian of a 4-state Potts problem and observe the role of noise as n increases. After that, we increase n further to verify the importance of SYNC ramping and noise to reach the global minimum Hamiltonian in Sec. IV-D. Finally, in Sec. IV-E, we summarize the results shown in the previous subsections.

A. Small ($n = 2$, $N = 2$) illustrative example

To explain how the method works, we first take a very small and simple example of a random graph, with just 2 nodes and 1 edge, with the edge weight set to -1 (Figure 1) and the value of N set to 2, *i.e.*, the Ising problem.

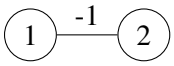

Fig. 1: A random graph with 2 nodes and 1 edge, with edge weight -1.

The intent of this experiment is to confirm that simulation of the Kuramoto model, in the presence of 2-SHIL, can successfully solve the problem (Figure 1) *i.e.*, find the global minimum Hamiltonian of the Potts Problem, considering each node as a Potts spin and the coupling between the spins as -1. Each spin can take 2 possible states ($N = 2$), uniformly distributed in 2π, and given by $\{\frac{\pi}{2}, \frac{3\pi}{2}\}$. To show that OPM can find a solution of this problem, the Kuramoto model in the presence of N-SHIL (35) was simulated, keeping the modulating factor over the coupling strength of the network $K = 0.5$, and the SYNC coupling $K_s = 3$, both constant with time. The Forward Euler method was used to solve (35) with simulation time step[2] set to 10^{-5}. Results from the simulation with $K_n = 0$ (*i.e.*, in the absence of noise) are shown in Figure 2.

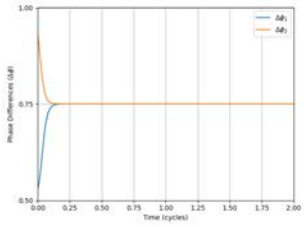

Fig. 2: 2-SHIL OPM simulation on the $n = 2$ system in the absence of noise with random initialization.

The initial phase differences of the oscillators were randomly initialized to values in the range [0, 1). In Sec. III-D, it has been shown that there should be N equally-spaced stable solutions to the Kuramoto equation in the presence of N-SHIL, given by $\Delta\phi_k^* = \frac{k}{2N}$ $(k = 1, 3, \cdots, 2N-1)$. Therefore, for 2-SHIL, phase differences should converge to $\frac{1}{4}$ or $\frac{3}{4}$, *i.e.*, 0.25 or 0.75. As expected, the phase differences converge to 0.75 in this example (Figure 2).

It is readily verified that the settled phase differences $\{\Delta\phi_1 = 0.75, \Delta\phi_2 = 0.75\}$ globally minimize the Hamiltonian (2):

$$\begin{aligned} H &= -1 \cdot (\cos(2\pi(0.75 - 0.75))) = -1 \cdot (\cos(2\pi \cdot 0)) \\ &= -1. \end{aligned} \tag{63}$$

Hence, OPM can be used for solving Ising combinatorial problems such as MAX-CUT, taking the value of N as 2.

It is to be noted that due to random initializations, the solutions converge to different values on every run, thereby generating different plots, and hence convergence to the global minimum solution cannot be ensured on every run. For *e.g.*, in Figure 3, the initial phase differences of the oscillators are such that the settled phase differences $\{\Delta\phi_1 = 0.25, \Delta\phi_2 = 0.75\}$ do not globally minimize the Hamiltonian (2):

$$\begin{aligned} H &= -1 \cdot (\cos(2\pi(0.25 - 0.75))) \\ &= -1 \cdot (\cos(2\pi \cdot (-0.5))) = (-1) \cdot (-1) = 1. \end{aligned} \tag{64}$$

For this $n = 2, N = 2$ problem, using 10 different combinations of K and K_s (both constant with time), with 10 random initial conditions per combination, the global minimum Hamiltonian value was achieved on an average of 6 out of 10 times. Hence, for this small problem, ramping the SYNC signal up and down, as well as noise was not necessary. The necessity arises when the size

[2]The Forward Euler method was also used for the other problems, which have been explained later on in this section, keeping the simulation time step fixed at 10^{-5}. Accuracy of the Forward Euler increases as step size is reduced, but the computational cost also increases due to increase in the number of time steps. Hence an optimum value of 10^{-5} was chosen.

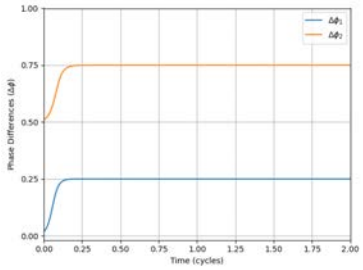

Fig. 3: 2-SHIL OPM simulation on the $n = 2$ system in the absence of noise with random initialization such that global minimum Hamiltonian is not obtained.

of the problem increases, as we will be seeing in the subsequent subsections.

B. $n = 4$, $N = 3$ example

The purpose of this example is to confirm that simulation of the Kuramoto model, under the influence of 3-SHIL, can find the global minimum Hamiltonian of a Potts problem, with each Potts spin taking up one out of 3 possible states ($N = 3$) distributed evenly in 2π, given by $\{\frac{\pi}{3}, \pi, \frac{5\pi}{3}\}$. Moreover, the importance of ramping the SYNC signal up and down will be demonstrated using this example.

For this problem, a random graph with 4 nodes and 3 edges was generated using **./rudy -rnd_graph 4 50 10001**, with edge weights in $\{-1, +1\}$ (Figure 4). The fraction of edges with negative weights is $\frac{2}{3}$.

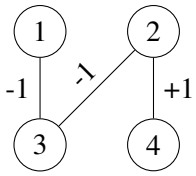

Fig. 4: A random graph with 4 nodes and 3 edges, with edge weights in $\{-1, +1\}$.

All different possible values of the Hamiltonian for this problem are summarized in Table I.

TABLE I: Possible values of the Hamiltonian for the problem in Figure 4.

No.	Hamiltonian Value	An example of settled phase differences $(\Delta\phi_1, \Delta\phi_2, \Delta\phi_3, \Delta\phi_4)$
1	-2.5	$(\frac{5}{6}, \frac{5}{6}, \frac{5}{6}, \frac{1}{6})$
2	-1.0	$(\frac{1}{2}, \frac{1}{6}, \frac{1}{2}, \frac{5}{6})$
3	0.5	$(\frac{1}{2}, \frac{1}{2}, \frac{1}{6}, \frac{5}{6})$
4	2.0	$(\frac{1}{2}, \frac{1}{6}, \frac{5}{6}, \frac{1}{6})$

For the simulation, the values of K and K_s were kept fixed (with time) at 0.5 and 3 respectively. The simulation was run in the absence of noise. The evolution of phase differences with time are shown in Figure 5. It is seen from Figure 5 that the phase differences converge to an odd multiple of $\frac{1}{6}$. Moreover, using the settled phase differences $\{\Delta\phi_1 = \frac{1}{2}, \Delta\phi_2 = \frac{1}{2}, \Delta\phi_3 = \frac{1}{2}, \Delta\phi_4 = \frac{1}{6}\}$, the obtained value of the corresponding Hamiltonian is -2.5, which is the global minimum, as verified from Table I.

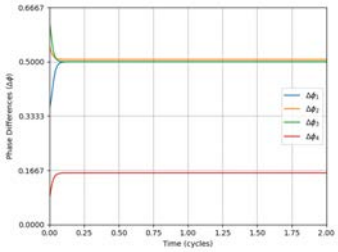

Fig. 5: 3-SHIL OPM simulation on the $n = 4$ system in the absence of noise with random initialization.

For this example, it was observed that the global minimum Hamiltonian value was either not obtained, or obtained in very few random trials for certain combinations of K and K_s (constant with time), due to the local minimization of the associated Lyapunov function. Running the simulation keeping K and K_s fixed with time at 0.3 and 2 respectively, Figure 6 was obtained.

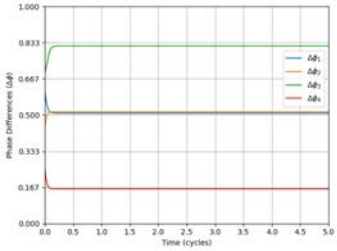

Fig. 6: 3-SHIL OPM simulation on the $n = 4$ system with constant $K_s = 2$, for which the global minimum Hamiltonian is not reached.

It is readily seen that the approximate settled phase differences in Figure 6 are $\{\Delta\phi_1 = \frac{1}{2}, \Delta\phi_2 = \frac{1}{2}, \Delta\phi_3 = \frac{5}{6}, \Delta\phi_4 = \frac{1}{6}\}$, for which the Hamiltonian value is 0.5. This is not the global minimum, as seen from Table I.

To move out of the local minimum, ramping the SYNC signal up and down is very useful. Hence, the K_s parameter, kept fixed at 2 for obtaining Figure 6, was varied with time using a triangular waveform with periodicity 1 (Figure 7), taking the peak value as 2 (previously constant with time) and the minimum value as 0.

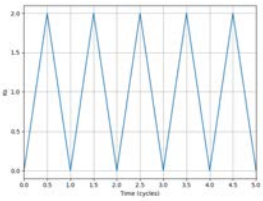

Fig. 7: Triangular waveform for K_s with minimum value = 0, maximum value = 2 and period = 1.

The results from this simulation, considering the same random initial condition used for obtaining Figure 6, is shown in Figure 8.

The settled values in Figure 8 (in the presence of SHIL) are $\{\Delta\phi_1 = \frac{1}{2}, \Delta\phi_2 = \frac{1}{2}, \Delta\phi_3 = \frac{1}{2}, \Delta\phi_4 = \frac{1}{6}\}$, for which the Hamiltonian value is -2.5. This illustrates how ramping SYNC up and down can drive the system to find the global minimum In fact, for 100 random choices of initial condition, simulation using fixed $K = 0.3$ and

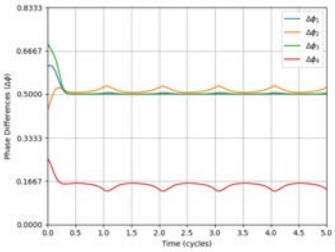

Fig. 8: 3-SHIL OPM simulation on the $n = 4$ system with time-varying K_s (Figure 7) and the same random seed as Figure 6, for which the global minimum Hamiltonian is reached.

$K_s = 2$ globally minimized the Hamiltonian only 9 times, whereas simulation using time-varying K_s (Figure 7) globally minimized the Hamiltonian 26 times. For this problem, noise was not essential; but noise plays an important role as the size of the problem increases, as we will see in the following subsections.

C. $n = 20$, $N = 4$, $Sparsity = \frac{137}{190}$ example

The point of this example is to show that the Hamiltonian of a Potts problem, with 4 possible states ($N = 4$) per spin, divided equally in 2π, can be globally minimized using the simulation of OPM, in the presence of 4-SHIL. Also, the role of noise is shown using this example.

We now consider a random graph with 20 nodes and 53 edges, generated using **./rudy -rnd_graph 20 28 1000**, with edge weights in $\{-1, +1\}$. The fraction of negative edge weights is $\frac{32}{53}$. As discussed in Sec. IV-B, with the increase in the size of the problem, global minimum Hamiltonian may not be obtained using constant K_s due to the local minimization of the associated Lyapunov function, and ramping SYNC up and down becomes necessary. Hence, a triangular waveform for K_s, shown in Figure 9, was used for the problem.

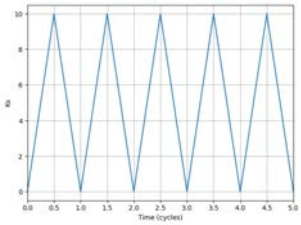

Fig. 9: Triangular waveform for K_s with minimum value = 0, maximum value = 10 and period = 1.

The value of K was kept fixed at 2.5 during the simulation, with $K_n = 0$. The evolution of phase differences and the corresponding Hamiltonian is shown in Figures 10 and 11, respectively. It is observed from Figure 10 that the phase differences converge approximately to odd multiples of $\frac{1}{8}$, in the presence of SHIL. It is also seen that the discretization goes away as SHIL is gradually removed from the system. Yet another observation is that most of the values converge approximately to some odd multiple of $\frac{1}{2N}$ and in the range [0, 1), as expected. However, some of the values are outside that range, but can be mapped to some odd multiple of $\frac{1}{2N}$ and in the range [0, 1). The Kuramoto equation (35) contains $\sin(2\pi(\cdot))$ terms which are 1-periodic. So, if $\Delta\phi$ is an odd multiple of $\frac{1}{2N}$ and lies in the range [-1, 0), it can be mapped to $(\Delta\phi + 1)$, which is also an odd multiple of $\frac{1}{2N}$ and lies in the range [0, 1). Similarly, if $\Delta\phi$ lies in the range [1, 2), it can be mapped to $(\Delta\phi - 1)$, which lies in the range [0, 1).

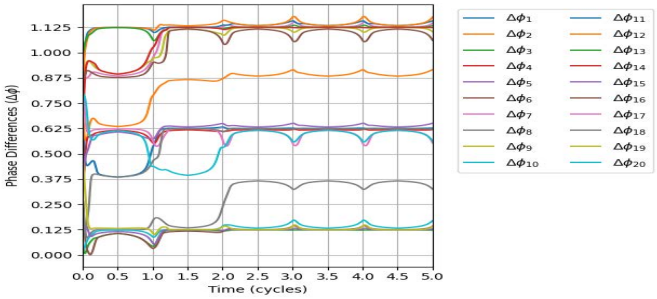

Fig. 10: 4-SHIL OPM simulation on the $n = 20$ system in the absence of noise with random initialization.

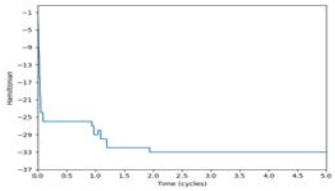

Fig. 11: The evolution of the corresponding Hamiltonian over time for the $n = 20$ system.

Figure 11 has been generated by thresholding phase differences at each time step to the nearest odd multiples of $\frac{1}{8}$ to correspond to the discretized Potts states. It is observed that the Hamiltonian value, obtained using the settled phase difference, is -33.

To see if this value is the global minimum, a separate program was written that follows a brute-force approach for generating the minimum value of the Vector Potts Hamiltonian for a particular problem. For this program, Backtracking[3] [22] was used to generate all possible permutations of the states of the Potts spins, thereby computing the Hamiltonian for each permutation, and extracting the minimum value. The Python implementation of this is given in Appendix A. Using this program, the minimum Hamiltonian value for this problem was found out to be -33, thereby confirming that the Hamiltonian value obtained using OPM is indeed the global minimum.

Moreover, it was observed that the introduction of an appropriate amount of noise improved the performance by effectively globally minimizing the associated Lyapunov function. Keeping K fixed at 2.5 and using the time-varying K_s shown in Figure 9, in the absence of noise, the number of times that the global minimum Hamiltonian value was reached, out of 100 random choices of initial conditions, was 36, whereas with $K_n = 0.2$ (constant with time), the number of times that the global minimum was obtained was 55. Also, in the presence of noise, the number of times that Hamiltonian values $< 90\%$ of the global minimum was obtained, was 94 out of the 100 random instances. This indicates that OPM successfully found solutions of the problem, which are

[3]Backtracking is a general method used to solve a computational problem. It recursively builds the solution in an incremental manner with time by searching the entire solution space and removing the solutions that do not satisfy the constraints of the problem.

close to the global minimum. Considering one such random initialization such that the global minimum value was obtained with noise and not in the absence of noise, the progress of phase differences with time, in the absence and the presence of noise, are shown in Figures 12 and 13 respectively.

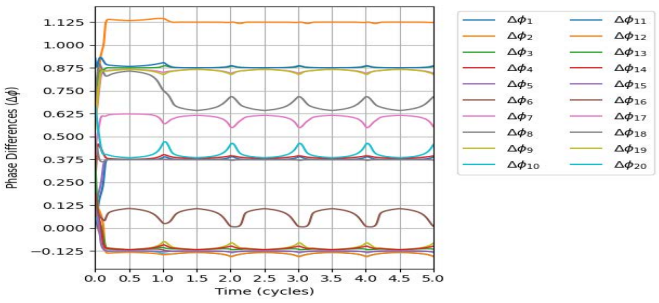

Fig. 12: 4-SHIL OPM simulation on the $n = 20$ system with $K_n = 0$, for which the global minimum Hamiltonian is not reached.

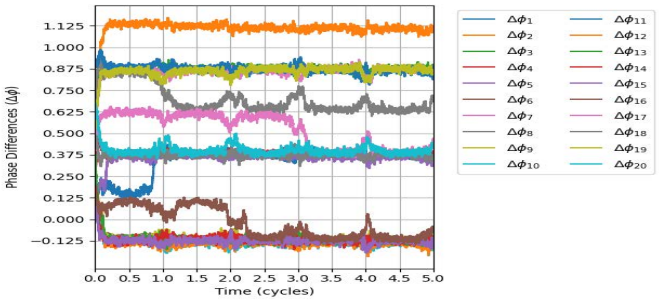

Fig. 13: 4-SHIL OPM simulation on the $n = 20$ system with $K_n = 0.2$ and the same random seed as Figure 12, for which the global minimum Hamiltonian is reached.

The change in the corresponding Hamiltonian values with time, in the absence and presence of noise, are shown in Figures 14 and 15, respectively. In the absence of noise, the Hamiltonian value reached is -32 (Figure 14). But, with noise, the global minimum Hamiltonian value of -33 is achieved (Figure 15). Hence, noise plays an important role in the global minimization of the Lyapunov function, resulting in the global minimization of the corresponding Hamiltonian.

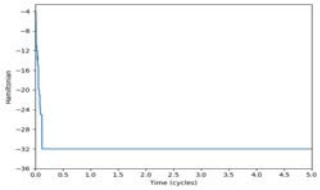

Fig. 14: The evolution of the Hamiltonian over time corresponding to Figure 12, for which the global minimum Hamiltonian is not reached.

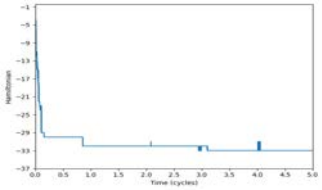

Fig. 15: The evolution of the Hamiltonian over time corresponding to Figure 13, for which the global minimum Hamiltonian is reached.

Simulation was also run using 10 different combinations of K and K_s, keeping K_n fixed at 0.2. For each combination, 10 random initializations were considered, adding to a total of 100 random instances. The histogram obtained from this simulation is shown in Figure 16.

The global minimum Hamiltonian was noted to be found at least once in 7 out of the 10 combinations, and in 36 out of the 100 random instances. The mean and standard deviation of the minimum Hamiltonian value achieved

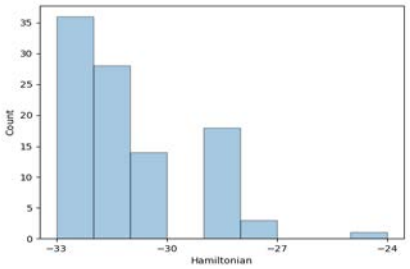

Fig. 16: Histogram of the minimum Hamiltonian obtained using different combinations of K and K_s, keeping $K_n = 0.2$, for the $n = 20$ system.

in each simulation were -32.1 and 1.578, respectively.

D. $n = 32$, $N = 4$, $Sparsity = \frac{109}{124}$ example

We consider another example, with 32 nodes and 60 edges, generated using **./rudy -rnd_graph 32 12 10001**, with the edge weights in $\{-1, +1\}$ and N set to 4. The fraction of edges with negative weights is $\frac{7}{15}$. The aim of this example is to confirm the usefulness of ramping SYNC up and down for globally minimizing the Hamiltonian, as discussed in Sec. IV-B. Also, the value of noise, already discussed in Sec. IV-C, is verified.

K and K_n were kept constant at 5 and 0.2, respectively, over the simulation. K_s was time-varying as in Figure 9. Figures 17 and 18 show how phase differences and the corresponding Hamiltonian advance with time, respectively.

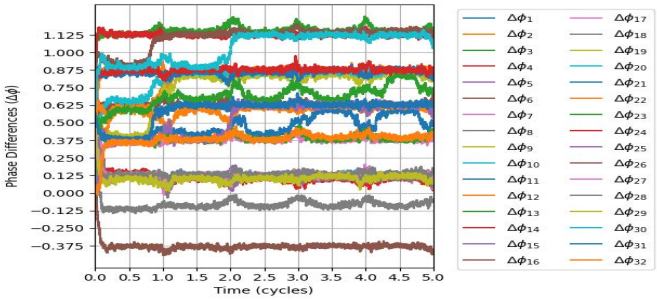

Fig. 17: 4-SHIL OPM simulation on the $n = 32$ system in the presence of noise with random initialization.

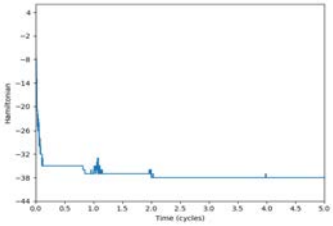

Fig. 18: The evolution of the corresponding Hamiltonian over time for the $n = 32$ system.

It is seen from Figure 17 that the phase differences converge to odd multiples of $\frac{1}{8}$ in the presence of SHIL, indicating that the phase differences get discretized, which correspond to the discretized Potts states. Also, it is observed from Figure 18 that the value of the Hamiltonian, computed using the settled phase differences, is -38. The brute force program, mentioned in Sec. IV-C (Python implementation in Appendix A), was used to verify that the Hamiltonian value of -38 is the global minimum value.

Furthermore, simulation was run using 10 different combinations of K and K_s, over 10 random initializations per combination, *i.e.*, a total of 100 random instances, with $K_n = 0.2$. The obtained histogram is shown in Figure 19.

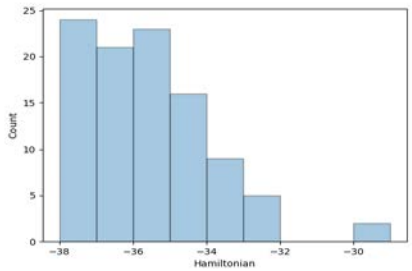

Fig. 19: Histogram of the minimum Hamiltonian obtained using different combinations of the parameters K and K_s, keeping $K_n = 0.2$, for the $n = 32$ system.

It was observed that the global minimum Hamiltonian was obtained at least once in 5 out of the 10 combinations. Moreover, it can be seen from Figure 19 that the global minimum value was reached in 24 out of the 100 random instances. The mean and standard deviation of the minimum Hamiltonian obtained in each simulation were -37.1 and 1.044, respectively.

It is important to note that, with the increase in the size of the problem, the proper choice of the parameters become crucial in achieving good results. For this problem, the importance of the choice has been shown using histograms (Figure 20), with the help of three different combinations of the simulation parameters. The combinations include constant K_s, SYNC ramping in the absence of noise and SYNC ramping in the presence of noise. K was kept fixed for all the three combinations at 5. For the first combination, K_s was taken to be 10 (fixed with time). For the second and third combinations in which SYNC ramping was performed, time-varying K_s as in Figure 9 was considered. K_n was kept constant with time at 0.2 for the third combination. 100 random choices of initial conditions were considered for each of the three combinations.

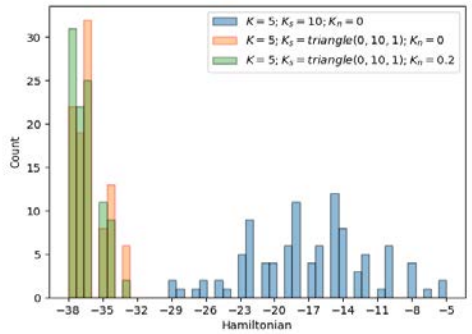

Fig. 20: Histograms of the minimum Hamiltonian achieved by different combinations of the parameters K, K_s and K_n, for the $n = 32$ system. triangle(0, 10, 1) refers to the triangular waveform in Figure 9.

The results from the histograms (Figure 20) have been summarized in Table II.

TABLE II: Different combinations of the simulation parameters to observe the number of times the global minimum is obtained, out of 100 random instances. triangle(0, 10, 1) refers to the triangular waveform in Figure 9 with minimum value = 0, peak value = 10 and period = 1.

No.	Simulation Parameters	Number of times the global minimum is obtained
1	K = 5; K_s = 10; K_n = 0	0
2	K = 5; K_s = triangle(0, 10, 1); K_n = 0	22
3	K = 5; K_s = triangle(0, 10, 1); K_n = 0.2	31

Hence, it is readily seen that both SYNC ramping and noise play a crucial role in the global minimization of the Hamiltonian. Moreover, for the 3^{rd} combination in Table II, *i.e.*, in the presence of time-varying K_s and noise, Hamiltonian values $< 90\%$ of the global minimum were obtained in 89 out of the 100 random instances, as observed from Figure 20. This signifies that OPM was successful in finding solutions of the problem, which are near the global minimum.

Python code for reproducing these outcomes is given in Appendix B. To generate the results reported here, we used a C++ implementation which is much faster than Python.

E. Summary of the results

The results of the already-mentioned examples, along with the results of two other examples (*i.e.*, $n = 20$, $N = 3$ example, for which the same graph as in Sec. IV-C was used, and $n = 32$, $N = 3$ example, for which the graph used in Sec. IV-D was considered), have been summarized in Table III.

Note that, as expected, OPM is much faster than the Brute Force approach as n increases, which is observed from Table IV. The examples in Table IV correspond to the ones discussed in the previous subsections. For very small values of n, the difference in computation time between the two approaches is not significant. However, as n increases, due to the N^n possible permutations of the states of the Potts spins, the Brute Force approach can take several days to complete, whereas the OPM approach can be completed within a very short period of time, as seen in Table IV.

TABLE III: Examples for validating OPM

Test No.	(n,N)	-ve w_{ij} [4]	Brute Force: min H value[5]	OPM: min H value[6]	Parameters[7]	OPM: (Mean, STD)[8]
1	(2, 2)	1	-1	-1	K = 0.5; K_s = 3; K_n = 0 ; T_{stop} = 2; Edges = 1.	(-1, 0)
2	(4, 3)	$\frac{2}{3}$	-2.5	-2.5	K = 0.5; K_s = 3; K_n = 0 ; T_{stop} = 2; Edges = 3.	(-2.5, 0)
3	(20, 3)	$\frac{32}{53}$	-27.5	-27.5	K = 1.5; K_s = triangle(0, 10, 1)[9]; K_n = 0.2 ; T_{stop} = 5; Edges = 53.	(-27.2, 0.9)
4	(20, 4)	$\frac{32}{53}$	-33	-33	K = 2.5; K_s = triangle(0, 10, 1)[9]; K_n = 0.2 ; T_{stop} = 5; Edges = 53.	(-32.1, 1.578)
5	(32, 3)	$\frac{7}{15}$	-33.5	-33.5	K = 5; K_s = triangle(0, 10, 1)[9]; K_n = 0.2 ; T_{stop} = 5; Edges = 60.	(-32.9, 0.995)
6	(32, 4)	$\frac{7}{15}$	-38	-38	K = 5; K_s = triangle(0, 10, 1)[9]; K_n = 0.2 ; T_{stop} = 5; Edges = 60.	(-37.1, 1.044)

V. Conclusion

In conclusion, we have demonstrated a method for implementing the Vector Potts Model using a network of coupled nonlinear oscillators in this technical report. We have shown that the phase dynamics of the oscillator networks, under the influence of N-SHIL, evolve naturally to minimize the associated Lyapunov function, signifying the minimization of the Vector Potts Hamiltonian of the corresponding network coupling graph. The effectiveness and validity of our proposed scheme have been demonstrated using simulation, with the help of several randomly generated graphs of varying sizes. The proposed OPM method can be used in the implementation of the Vector Potts Model in a practical manner.

[4]Fraction of negative edge weights.

[5]Minimum Hamiltonian Value found by Brute Force.

[6]Minimum Hamiltonian Value found by OPM (over 10 random runs).

[7]Edge weights $\in \{-1,+1\}$, $T_{step} = 10^{-5}$.

[8]Mean and Standard Deviation of the Minimum Hamiltonian value achieved in each simulation, obtained using 10 different combinations of K and K_s, over 10 random initial conditions per combination.

[9]triangle(K_s_low, K_s_high, period)

[10]For these examples, the C++ implementation of the Brute Force approach was run parallelly using 40 processors (2.50 GHz) to obtain the global minimum value. For all the remaining entries in Table IV, the C++ implementation was run on a single processor (2.30 GHz) machine.

TABLE IV: Comparison of computation times taken by the Brute Force approach and OPM with the increase in n.

n	N	Brute Force: Computation Time	OPM: Computation Time
2	2	1.675×10^{-5} s	0.0507 s
4	3	3.843×10^{-5} s	0.1642 s
20	4	28 hrs (approximately)[10]	9.214 s
32	4	96.3 hrs (approximately)[10]	27.074s

References

[1] Wu, F. Y., "The Potts model," *Reviews of Modern Physics*, vol. 54, pp. 235–268, Jan 1982. DOI link.

[2] E. Ising, "Beitrag zur theorie des ferromagnetismus," *Zeitschrift für Physik A Hadrons and Nuclei*, vol. 31, no. 1, pp. 253–258, 1925.

[3] S. G. Brush, "History of the Lenz-Ising Model," *Rev. Mod. Phys.*, vol. 39, pp. 883–893, Oct 1967.

[4] G. Rinaldi, "Rudy Graph Generator code." Source Code link.

[5] A. Demir, A. Mehrotra, and J. Roychowdhury, "Phase Noise in Oscillators: a Unifying Theory and Numerical Methods for Characterization," *IEEE Trans. Ckts. Syst. – I: Fund. Th. Appl.*, vol. 47, pp. 655–674, May 2000. DOI link.

[6] A. Demir and J. Roychowdhury, "A Reliable and Efficient Procedure for Oscillator PPV Computation, with Phase Noise Macromodelling Applications," *IEEE Trans. CAD*, vol. 22, pp. 188–197, February 2003.

[7] Y. Kuramoto, "Self-entrainment of a population of coupled non-linear oscillators," in *International symposium on mathematical problems in theoretical physics*, pp. 420–422, Springer, 1975.

[8] Y. Kuramoto, *Chemical Oscillations, Waves and Turbulence*. Dover, 2003.

[9] J. A. Acebrón, L. L. Bonilla, C. J. P. Vicente, F. Ritort and R. Spigler, "The Kuramoto Model: A Simple Paradigm for Synchronization Phenomena," *Reviews of Modern Physics*, vol. 77, no. 1, p. 137, 2005.

[10] A. Neogy and J. Roychowdhury, "Analysis and Design of Sub-harmonically Injection Locked Oscillators," in *Proc. IEEE DATE*, Mar 2012. DOI

link.

[11] A. M. Lyapunov, "The General Problem of the Stability of Motion," *International Journal of Control*, vol. 55, no. 3, pp. 531–534, 1992.

[12] M. W. Johnson, M. H. S. Amin, S. Gildert, T. Lanting, F. Hamze, N. Dickson, R. Harris, A. J. Berkley, J. Johansson, P. Bunyk and others, "Quantum Annealing with Manufactured Spins," *Nature*, vol. 473, no. 7346, p. 194, 2011.

[13] Z. Bian, F. Chudak, R. Israel, B. Lackey, W. G. Macready and A. Roy, "Discrete optimization using quantum annealing on sparse Ising models," *Frontiers in Physics*, vol. 2, p. 56, 2014.

[14] A. Marandi, Z. Wang, K. Takata, R. L. Byer and Y. Yamamoto, "Network of time-multiplexed optical parametric oscillators as a coherent Ising machine," *Nature Photonics*, vol. 8, no. 12, pp. 937–942, 2014.

[15] P. L. McMahon, A. Marandi, Y. Haribara, R. Hamerly, C. Langrock, S. Tamate and T. Inagaki, H. Takesue, S. Utsunomiya, K. Aihara and others, "A fully-programmable 100-spin coherent Ising machine with all-to-all connections," *Science*, p. 5178, 2016.

[16] T. Inagaki, Y. Haribara, K. Igarashi, T. Sonobe, S. Tamate, T. Honjo, A. Marandi, P. L. McMahon, T. Umeki, K. Enbutsu and others, "A Coherent Ising machine for 2000-node Optimization Problems," *Science*, vol. 354, no. 6312, pp. 603–606, 2016.

[17] M. Yamaoka, C. Yoshimura, M. Hayashi, T. Okuyama, H. Aoki and H. Mizuno, "A 20k-spin Ising Chip to Solve Combinatorial Optimization Problems with CMOS Annealing," *IEEE Journal of Solid-State Circuits*, vol. 51, no. 1, pp. 303–309, 2016.

[18] T. Wang and J. Roychowdhury, "OIM: Oscillator-based Ising Machines for Solving Combinatorial Optimisation Problems," *arXiv preprint arXiv:1903.07163*, 2019.

[19] M. Honari-Latifpour and M. A. Miri, "Optical Potts machine through networks of three-photon down-conversion oscillators," *Nanophotonics*, vol. 9, no. 13, pp. 4199–4205, 2020. DOI link.

[20] J. Roychowdhury, "Numerical simulation and modelling of electronic and biochemical systems," *Foundations and Trends in Electronic Design Automation*, vol. 3, pp. 97–303, December 2009.

[21] A. Winfree, "Biological Rhythms and the Behavior of Populations of Coupled Oscillators," *Theoretical Biology*, vol. 16, pp. 15–42, 1967.

[22] R.J. Walker, "An enumerative technique for a class of combinatorial problems," in *Combinatorial Analysis, Proc. Sympos. Appl. Math*, vol. 10, pp. 91–94, 1960.

Appendix A. Python Code for the Brute Force Method for Computing the Global Minimum Hamiltonian

Listing 1: Brute_Force.py

```
import numpy as np
import math

# Calculate and Compare the Hamiltonian
def calc_hamil(n, N, J, min_ham, permutation):
    func_val = 0;

    for i in range(0, n):
        for j in range(i + 1, n):
            func_val += J[i][j] * math.cos((2 * \
                    math.pi * (permutation[i] - \
                    permutation[j])) / N);

    if (func_val < min_ham[0]):
        min_ham[0] = func_val;

# Backtracking Approach for Computing the
# Global Minimum Hamiltonian
def getMinHamil(n, N, J, min_ham, permutation, i):
    if (i == n):
        calc_hamil(n, N, J, min_ham, permutation);
        return;

    for j in range(N):
        permutation[i] = j;
        getMinHamil(n, N, J, min_ham, \
        permutation, i + 1);

if __name__ == "__main__":
    test_file = open("path_to_rudy_file", "r");
    file_content = test_file.read().split();
    file_content = [int(content) for content \
                    in file_content];

    n = file_content[0];

    num_edges = file_content[1];

    w = np.zeros((n,n));
    for i in range(2, 3 * num_edges, 3):
        w[file_content[i] - 1, file_content[i \
                + 1] - 1] = file_content[i + 2];

    test_file.close();

    N = 4;

    J = w + w.T;

    permutation = np.zeros(n);

    min_ham = [];
    min_ham.append(float("inf"));

    getMinHamil(n, N, J, min_ham, permutation, 0);

    print("Minimum Hamiltonian: %.2f" % \
            (min_ham[0]))
```

Appendix B. Python Code for OPM

Listing 2: Oscillator_Potts_Machine.py

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

# Thresholding the phase difference values
def thresholding(delphi, thresholded_delphi, \
        n, N, possible_stable_sols):
    for i in range(n):
        delphi_val = delphi[i];
        if (delphi_val >= 1):
            delphi_val -= 1;
        elif (delphi_val < 0):
            delphi_val += 1;
```

```
        min_idx = (np.abs(possible_stable_sols \
                - delphi_val)).argmin();
        thresholded_delphi[i] = \
                possible_stable_sols[min_idx];

# Calculate the Hamiltonian
def calc_hamil(J, delphi, n, N, \
        possible_stable_sols):
    thresholded_delphi = np.zeros(n);
    thresholding(delphi, thresholded_delphi, \
            n, N, possible_stable_sols);

    func_val = 0.0;
    for i in range(n):
        for j in range(i + 1, n):
            func_val += J[i][j] * math.cos(2 * \
                    math.pi * \
                    (thresholded_delphi[i] \
                    - thresholded_delphi[j]));
    del thresholded_delphi;
    return func_val;

# N-SHIL Kuramoto Model
def NSHIL_Kuramoto(K, Ks, J, i, n, delphi, N):
    func_val = 0;
    for j in range(n):
        if j != i:
            func_val += J[i][j] * math.sin(2 * \
                    math.pi * (delphi[i] - \
                    delphi[j]));
    func_val = (K * func_val) + (Ks * \
            math.sin(2 * math.pi * N * delphi[i]));
    return func_val;

# Forward Euler method
def ForwardEuler(t, delphi, h, steps, n, J, K, \
        Ks, Kn, N, ham, possible_stable_sols):
    for step in range(1, steps):
        for osc in range(n):
            delphi[osc, step] = delphi[osc, \
                    step - 1] + ((h * \
                    NSHIL_Kuramoto(K, \
                    Ks[step - 1], J, osc, \
                    n, delphi[:, step - 1], \
                    N)) + (Kn * np.random.normal \
                    (loc = 0.0, scale = \
                    math.sqrt(h))));
        ham[step - 1] = calc_hamil(J, delphi[:, \
                step - 1], n, N, \
                possible_stable_sols);

    # Hamiltonian for the last timestep
    ham[steps - 1] = calc_hamil(J, delphi[:, \
                steps - 1], n, N, \
                possible_stable_sols);

# Square Waveform
def square_Ks(t, Ks, steps, Ks_max):
    for i in range(steps):
        if (int(t[i])%2 == 0):
            Ks[i] = 0;
        else:
            Ks[i] = Ks_max;

# Triangular Waveform
def triangle_Ks(t, Ks, steps, Ks_max):
    j = 0;
    for i in range (1, round(t[steps - 1]) + 1):
        while ((j < steps) and (t[j] <= (0.5 * (2 \
                * i - 1)))):
            Ks[j] = 2 * Ks_max * t[j] - \
                    2 * Ks_max * (i - 1);
            j += 1;
        while ((j < steps) and (t[j] > \
                (0.5 * (2 * i - 1))) and \
                (t[j] < (0.5 * 2 * i))):
            Ks[j] = - 2 * Ks_max * t[j] + \
                    2 * Ks_max * i;
            j += 1;

if __name__ == "__main__":
    test_file = open("path_to_rudy_file", "r");
    file_content = test_file.read().split();
    file_content = [int(content) for content \
            in file_content];
    n = file_content[0];
    num_edges = file_content[1];

    w = np.zeros((n,n));
    for i in range(2, 3 * num_edges, 3):
        w[file_content[i] - 1, file_content[i \
                + 1] - 1] = file_content[i + 2];

    test_file.close();

    N = 4;
    t_start = 0;
    t_stop = 5;
    h = 1e-5;
    steps = int((t_stop - t_start) / h) + 2;
    t = np.zeros(steps);
    for i in range(1, steps):
        t[i] = t[i - 1] + h;

    delphi = np.zeros((n, steps));
    for i in range(n):
        delphi[i, 0] = np.random.rand();

    J = w + w.T;

    K = 5;
    Ks_max = 10;
    Kn = 0.2;

    Ks = np.zeros(steps);
    triangle_Ks(t, Ks, steps, Ks_max);
    #square_Ks(t, Ks, steps, Ks_max);

    ham = np.zeros(steps);

    possible_stable_sols = np.zeros(N);
    for i in range(N):
        possible_stable_sols[i] = (2 * i + \
                1) / (2 * N);

    ForwardEuler(t, delphi, h, steps, n, J, \
            K, Ks, Kn, N, ham, \
            possible_stable_sols);

    del J, possible_stable_sols;

    print("Minimum Hamiltonian: %.2f" % \
            (np.min(ham)))

    plt.figure(0);
    for i in range(n):
        plt.plot(t, delphi[i]);
    plt.xlim(t[0], t[-1]);
    plt.xlabel('Time (cycles)');
    plt.ylabel('Phase Differences ($\Delta\phi$)');
    plt.xticks(np.linspace(t[0], round(t[-1]), 11));
    plt.title("Phase Differences ($\Delta\phi$)) \
                vs Time (cycles)")
    plt.grid(True);
    plt.tight_layout();

    plt.figure(1);
    plt.plot(t, ham);
    plt.xlim(t[0], t[-1]);
    plt.xlabel('Time (cycles)');
    plt.ylabel('Hamiltonian');
    plt.xticks(np.linspace(t[0], round(t[-1]), 11));
    plt.title("Hamiltonian vs Time (cycles)")
    plt.tight_layout();

    plt.figure(2);
    plt.plot(t, Ks);
    plt.xlim(t[0], t[-1]);
    plt.xlabel('Time (cycles)');
    plt.ylabel('Ks');
    plt.xticks(np.linspace(t[0], round(t[-1]), 11));
    plt.title("Ks vs Time (cycles)")
    plt.grid(True);
    plt.tight_layout();

    plt.show()

    del t, delphi, ham;
```