# A Functional Subnetwork Approach to Multistate Central Pattern Generator Phase Difference Control *

Cody Scharzenberger[1] and Alexander Hunt[1]

Portland State University, Portland OR 97207, USA
cscharz2@pdx.edu

**Abstract.** Central pattern generators (CPGs) are ubiquitous neural circuits that contribute to an eclectic collection of rhythmic behaviors across an equally diverse assortment of animal species. Due to their prominent role in many neuromechanical phenomena, numerous bioinspired robots have been designed to both investigate and exploit the operation of these neural oscillators. In order to serve as effective tools for these robotics applications, however, it is often necessary to be able to adjust the phase alignment of multiple CPGs during operation. To achieve this goal, we present the design of our phase difference control (PDC) network using a functional subnetwork approach (FSA) wherein subnetworks that perform basic mathematical operations are assembled such that they serve to control the relative phase lead/lag of target CPGs. Our PDC network operates by first estimating the phase difference between two CPGs, then comparing this phase difference to a reference signal that encodes the desired phase difference, and finally eliminating any error by emulating a proportional controller that adjusts the CPG oscillation frequencies. The architecture of our PDC network, as well as its various parameters, are all determined via analytical design rules that allow for direct interpretability of the network behavior. Simulation results for both the complete PDC network and a selection of its various functional subnetworks are provided to demonstrate the efficacy of our methodology.

**Keywords:** Multistate Central Pattern Generators · Functional Subnetwork Approach · Phase Difference Control

## 1 Introduction

As one of the fundamental neural units that contributes to a myriad of rhythmic behaviors throughout the animal kingdom, central pattern generators (CPGs) have seen a plethora of academic research over the past several decades. Thanks to these efforts, there is strong evidence for the existence of CPGs in a wide variety of animals, including but not limited to insects [10], fish [9], mollusks [15],

* Supported by Portland State University and NSF DBI 2015317 as part of the NSF/CIHR/DFG/FRQ/UKRI-MRC Next Generation Networks for Neuroscience Program.

amphibians [8], and mammals [6], as well as numerous studies detailing their role in various animal behaviors from respiration [11] to swimming [15], flying [12], and ambulation [3]. At the same time, we have seen numerous examples of biologically inspired robots that incorporate CPG elements into their broader control schemes in attempts to study the underlying neuromechanical basis for locomotion [7]. Given their wide applicability to disparate fields of academic research from biology to robotics, it is difficult to overstate the importance of these ostensibly elementary neural circuits; yet, despite the abundance of research in this field, several fundamental areas of inquiry still remain open. Of particular importance to locomotor robotics applications is the question of how best multiple CPGs may be coordinated via descending commands to achieve desired activation patterns. More specifically, since many practical applications require the ability to modify the phase relationship between several interacting CPGs, such as in legged locomotion where the relative timing of individual legs segments varies by gait, it is necessary to be able to design systems of coupled CPGs whose phase difference may be adjusted by simple descending commands. This is precisely the feat that we set out to accomplish in this paper.

### 1.1 Our Contribution

We present a novel method for controlling the phase difference between several multistate CPGs using a non-spiking functional subnetwork approach (FSA). These CPGs are "multistate" in that they permit arbitrarily many neurons to become maximally excited in a specific, predetermined order. Similarly, the "functional subnetwork approach" refers to the analytical non-spiking network design techniques developed in [14], which we extend and apply extensively in this work to the design of our phase difference control (PDC) network. Our PDC network operates by: (1) computing the existing phase lead/lag between two multistate CPGs, (2) computing the difference between the computed phase lead/lag and the desired phase lead/lag represented by a single descending command, and (3) adjusting the relative excitation of the two CPGs to modify their oscillation frequency and thus adjust their phase difference based on a simple proportional control scheme. Compared to existing techniques such as [1], this method is novel not just in the analytical approach that we take to designing our PDC network, but also in the fact that we are able to control the relative phase difference between two multistate CPGs with a *single* descending command, as opposed to having a specific, pre-determined desired phase difference built into the network parameters themselves. Furthermore, the technique that we present here is applicable to controlling the relative phase of multistate CPGs arranged in arbitrary patterns, including both sequential chains of CPGs such as those found in lamprey [9] and salamander [8] applications, as well as configurations where multiple CPGs utilize a single reference CPG such as when pattern generating CPG layers connect to higher level rhythm generating CPG layers in walking applications [2].

## 2 Background

The two fields of information that are required to understand our PDC network formulation are those pertaining to central pattern generators (CPGs) and the non-spiking functional subnetwork approach (FSA).

### 2.1 Central Pattern Generators (CPGs)

As has perhaps already been made clear by the abundant selection of aforementioned studies, a thorough treatment of the history of CPGs and their development is beyond the scope of this work. For a summary, refer to works such as [6] and [10] that provide animal specific overviews. To meet our ends, it is sufficient to instead focus on the mathematical description of these dynamical systems.

Consider a CPG comprised of $n \in \mathbb{N}$ neurons. Let $\mathbb{N}_{\leq n} = \{1, \ldots, n\}$. Then $\forall i \in \mathbb{N}_{\leq n}$ the membrane voltage $U_i$ of the $i$th neuron with respect to its resting potential $E_{r,i}$ with leak, synaptic, sodium channel, and applied currents satisfies

$$C_{m,i}\dot{U}_i = I_{leak,i} + I_{syn,i} + I_{Na,i} + I_{app,i} \tag{1}$$

where $C_{m,i}$ is the membrane capacitance and the constituent currents are defined as

$$I_{leak,i} = -G_{m,i}U_i, \tag{2}$$

$$I_{syn,i} = \sum_{j=1}^{n} G_{syn,ij}(\Delta E_{syn,ij} - U_i), \tag{3}$$

$$I_{Na,i} = G_{Na,i}m_{\infty,i}h_i(\Delta E_{Na,i} - U_i) \tag{4}$$

with membrane conductance $G_{m,i}$, synaptic reversal potential $\Delta E_{syn,ij}$, sodium channel conductance $G_{Na,i}$, and sodium channel reversal potential $\Delta E_{Na,i}$. Likewise, the synaptic conductance of the $ij$th synapse $G_{syn,ij}$ is defined by

$$G_{syn,ij} = g_{syn,ij} \min\left(\max\left(\frac{U_j}{R_j}, 0\right), 1\right) \tag{5}$$

where $g_{syn,ij}$ is the maximum synaptic conductance of the $ij$th synapse and $R_j$ is the operating voltage domain of the $j$th pre-synaptic neuron. Note that $h_i$ is a second dynamical variable that describes the sodium channel deactivation of neuron $i$ and satisfies

$$\dot{h}_i = \frac{h_{\infty,i} - h_i}{\tau_{h,i}} \tag{6}$$

where the sodium channel deactivation time constant is

$$\tau_{h,i} = \tau_{h,max,i}h_{\infty,i}\sqrt{A_{h,i}e^{-S_{h,i}(\Delta E_{h,i} - U_i)}} \tag{7}$$

with maximum sodium channel deactivation time constant $\tau_{h,max,i}$, and sodium channel deactivation parameters $A_{h,i}$, $S_{h,i}$, and $\Delta E_{h,i}$. The final remaining variables of these equations are the steady state sodium channel activation and deactivation parameters defined by

$$m_{\infty,i} = \frac{1}{1 + A_{m,i} e^{-S_{m,i}(\Delta E_{m,i} - U_i)}}, \tag{8}$$

$$h_{\infty,i} = \frac{1}{1 + A_{h,i} e^{-S_{h,i}(\Delta E_{h,i} - U_i)}} \tag{9}$$

where $A_{m,i}$, $S_{m,i}$, and $\Delta E_{m,i}$ are sodium channel activation parameters.

### 2.2 Functional Subnetwork Approach (FSA)

The functional subnetwork approach (FSA) refers to the analytical methods developed by [14] for designing subnetworks of non-spiking neurons to perform basic tasks, including: (1) signal transfer such as transmission and modulation; (2) arithmetic operations such as addition, subtraction, multiplication, and division; and (3) calculus operations such as differentiation and integration. One of the main attractions of this work lies in the fact that it combines *simple* neural architectures with *analytical* design rules constrained by biological limitations, a combination of features that ensures that the resulting subnetworks are both meaningful and easily interpretable. This is in contrast to the often "black-box" nature of networks whose architectures and other hyper-parameters are tuned via genetic algorithms [4] or Bayesian optimization [5], and whose lower level parameters are determined via any number of popular optimization schemes. In order to construct our phase difference controller subnetwork, we apply many slight variations of the different FSA subnetworks. For a thorough explanation of their original FSA design rules refer to Tables 1 and 2 in [14].

## 3 Methodology

Having established the context for this work, we now present the computational algorithm that our PDC network performs, as well as the mathematical formulations necessary to design each of its constituent subnetworks. Since the same PDC network architecture is used to control the relative phase between any pair of multistate CPGs, we can assume that we have $N = 2$ multistate CPGs with $n \in \mathbb{N}$ neurons each without loss of generality. Let the subscripts $A$ and $B$ refer to the properties of the first and second multistate CPGs, respectively. In this case, we wish to construct a PDC network that controls the phase difference $\Delta\phi_{AB} = \phi_A - \phi_B$ according to the desired phase difference encoded as an applied current. To accomplish this task, we: (1) create the two multistate CPGs whose relative phase we will control; (2) design a subnetwork that estimates the phase difference between these two CPGs; (3) compute the phase error relative to the desired phase reference; and (4), use a simple proportional control scheme to eliminate the error.

### 3.1 Designing Driven Multistate CPGs

The first step in designing our PDC subnetwork is to actually create the two multistate CPGs whose relative phase we will be controlling. To accomplish this, we draw significant inspiration from [13], extending their methodology to work for CPGs comprised of arbitrarily many neurons.

**Sodium Channel Conductances $G_{Na,i}$** While the sodium channel reversal potentials $\Delta E_{Na,i}$ are determined based on biological constraints, the sodium channel conductances $G_{Na,i}$ are CPG exclusive parameters that must be set to ensure adequate intrinsic excitation of the CPG neurons. Fortunately, the use of arbitrarily many CPG neurons does not complicate [13]'s design rule beyond additional indexing, so no additional derivation is required.

**Maximum Synaptic Conductances $g_{syn,ij}$** To compute the required maximum synaptic conductances $g_{syn,ij}$ for each synapse in our multistate CPG, we once again take inspiration from [13]. Szczecinski et. al point out that the oscillatory behavior of a two neuron CPG depends strongly on the bifurcation parameter $\delta$, which represents the steady state membrane voltage of the inhibited neuron when the other neuron is excited. If $\delta \leq 0$, the system operates in a bistable mode. If instead $0 < \delta < \delta_{max}$, the system oscillates with increasing frequency as $\delta \to \delta_{max}$. Finally, once $\delta \geq \delta_{max}$, the system settles into a single stable equilibrium point.

For the purposes of our multistate CPG comprised of $n \in \mathbb{N}$ neurons, we will have $n(n-1)$ total $\delta_{ij}$ values to set, where each $\delta_{ij}$ represents the desired steady state membrane voltage $U_i^\star$ of neuron $i$ when neuron $j$ is maximally excited $\forall i, j \in \mathbb{N}_{\leq n}$ such that $i \neq j$. In other words, when $U_j^\star = R_j$ we want $U_i^\star = \delta_{ij}$ $\forall i \neq j$. Let $(k_j)_{j=1}^n = (k_1, \ldots, k_n)$ be a sequence of $n$ indexes $k_j \in \mathbb{N}_{\leq n}$ such that $k_j \neq j$ that defines the order in which we would like the neurons of our multistate CPG to become maximally excited. To ensure that only a single neuron becomes excited at a time and that the system oscillates in the desired order, we require that $\forall j \in \mathbb{N}_{\leq n}$ with $i \neq j$

$$\begin{cases} 0 < \delta_{ij} < \delta_{max}, & \text{if } i = k_j \\ \delta_{ij} < 0, & \text{otherwise} \end{cases} \tag{10}$$

In other words, we require that when each CPG neuron becomes excited, only one other neuron in the network attains a positive (but not too large) steady state membrane voltage $0 < \delta_{ij} < \delta_{max,ij}$ and that this neuron be the one that we desire to become maximally excited next. All other neurons in the CPG subnetwork should attain negative steady state membrane voltages $\delta_{ij} < 0$.

With an appropriate selection of bifurcation parameters $\delta_{ij}$, we can write a system of equations that describes the steady state behavior of the multistate CPG system when each of its neurons is maximally excited one at a time.

Specifically, $\forall i, j, k \in \mathbb{N}_{\leq n}$ with $i \neq j$, $j \neq k$, and $k \neq i$ we have

$$\begin{aligned} 0 = &-G_{m,i}\delta_{ik} + \sum_{j=1, j\neq k}^{n} g_{syn,ij} \min\left(\max\left(\frac{\delta_{jk}}{R_j}, 0\right), 1\right)(\Delta E_{syn,ij} - \delta_{ik}) \\ &+ G_{Na,i} m_{\infty,i}(\delta_{ik})\, h_{\infty,i}(\delta_{ik})\,(\Delta E_{Na,i} - \delta_{ik}) + I_{app,i} \end{aligned} \tag{11}$$

where $i$ indicates the post-synaptic neuron of interest, $j$ indicates the pre-synaptic neuron of interest, and $k$ indicates the neuron that is maximally excited. The previous expression is actually a system of $n(n-1)$ equations with the $n(n-1)$ maximum synaptic conductances $g_{syn,ij}$ being the only unknowns (we do not use self-connections, so $g_{syn,ij} = 0\ \forall i = j$). Solving this system for the unknown $g_{syn,ij}$ allows us to design a multistate CPG that oscillates in the order determined by our choice of $\delta_{ij}$ values.

**Designing Drive Synapses** In order to control the phase difference between two of our multistate CPGs, it is necessary that we design a mechanism through which we can adjust their relative oscillation frequency. Since each $\delta_{ij}$ represents the steady state membrane voltage $U_i^\star$ of neuron $i$ when neuron $j$ is maximally excited, we can temporarily adjust the steady state membrane voltage $U_i^\star$ without changing the associated $\delta_{ij}$ for which we designed our network by applying an external drive current $I_{dr,i}$ to the $i$th neuron. When this drive current has a magnitude of zero, the transition between neurons $j$ and $i$ occurs at the frequency determined by $\delta_{ij}$. However, as we increase this drive current up to some maximum value $I_{dr,id}^{max}$, the transition between neurons $j$ and $i$ occurs with an increased frequency beyond that associated with $\delta_{ij}$. Since it is our intention to control the *aggregate* phase difference between these CPGs, we use a single drive current per CPG that increases the transition frequency of all pairs of neurons.

In order to control the CPG phase difference via feedback, this drive current must be created via synaptic connections from a single drive neuron (one per multistate CPG). Let $d \in \mathbb{N}$ be the index associated with the drive neuron. Consider the synaptic currents $I_{syn,id}$ generated by the drive neuron $\forall i \in \mathbb{N}_{\leq n}$

$$I_{syn,id} = g_{syn,id} \min\left(\max\left(\frac{U_d}{R_d}, 0\right), 1\right)(\Delta E_{syn,id} - U_i). \tag{12}$$

We want to choose $g_{syn,id}$ such that when $U_d = R_d$ and $U_i = \max_{j\in\mathbb{N}}(\delta_{ij})$ then $I_{syn,id} = I_{dr,id}^{max}$. Making these substitutions into the above equation and solving for $g_{syn,id}$ yields our drive maximum synaptic conductance design requirement

$$g_{syn,id} = \frac{I_{dr,id}^{max}}{\Delta E_{syn,id} - \max_{j\in\mathbb{N}}(\delta_{ij})}. \tag{13}$$

### 3.2 Estimating Phase Lead/Lag

Estimating the phase lead/lag of the two CPGs is the most involved part of controlling their phase difference. This procedure adheres to the following logic: (1)

subtract the signals from the corresponding neurons in the CPGs, (2) integrate and post-process this difference to get a proxy for the phase difference, and (3) combine the phase difference estimates for each CPG neuron pair into a single phase difference estimate.

**Double Subtraction Subnetwork** To begin this process, we first subtract the membrane voltages of the corresponding CPG neurons such that $\Delta V_{AB,i} = V_{A,i} - V_{B,i}\ \forall i \in \mathbb{N}_{\leq n}$. In order to ensure that we can represent both lead and lag conditions, we perform this subtraction in both directions, resulting in two signals (e.g., $\Delta V_{AB,i}$ and $\Delta V_{BA,i}$). The first of these signals, $\Delta V_{AB,i}$, is maximized when neuron $i$ of CPG $A$ is maximally excited and neuron $i$ of CPG $B$ is inactive, while the second, $\Delta V_{BA,i}$, exhibits the opposite behavior.

While these two signals are not direct proxies for the phase difference between the two CPG neurons, they do provide information about the phase of the system. Fortunately, no additional design rules are required here because our double subtraction subnetwork is merely a combination of two of the single subtraction subnetworks from [14].

**Voltage-Based Integration Subnetwork** In order to determine the extent to which one CPG leads/lags the other, we must integrate the result of our double subtraction subnetwork using a modification of the integration subnetwork from [14]. Integrating $\Delta V_{AB,i}$ and $\Delta V_{BA,i}$ over time indicates how long the $i$th neuron of each CPG was held high while the other was inactive. Thus by integrating these signals over the oscillation period $T$ of the CPGs, we can compute a proxy $T_{AB,i}$ for the temporal lead/lag between the $i$th neurons by

$$T_{AB,i} = \int_0^T \Delta V_{AB,i} - \Delta V_{BA,i} dt. \tag{14}$$

Note that here we are assuming that the two CPGs have the same oscillation period $T$, which will not be the case once we start driving them at different oscillation frequencies to control their phase difference. This is not a problem in practice so long as the $i$th neuron of the lagging CPG does not become inactive while the $i$th neuron of the leading CPG remains active. Practically speaking, this limits how quickly we can adjust the phase difference of the two CPGs.

**Post-Processing Neuron Phase Difference Estimates** Having computed a rough proxy for the temporal lead/lag between the $i$th neurons of two CPGs, it is prudent to clean up this result before proceeding. In particular, the integration subnetwork result tends to change linearly over time as it accumulates incoming currents. Although such behavior is expected of an integrator, it is not ideal for achieving an accurate phase difference estimate because the temporal lead/lag estimate is only accurate after the integrator has reached a steady state result, not during the transient period wherein it accumulates currents.

To address this problem, we first split the centered integration result into two signals via a second double subtraction subnetwork. The first of these signals is produced by a neuron that is excited when the integration output is above its equilibrium value and the second when it is below its equilibrium value. Once the integration result is split into two signals, the linear ramping portions of these signals are removed via a modulation subnetwork [14] that is setup to suppress the results when either $V_{AB,i}$ or $V_{BA,i}$ are high. We call the signals from this modulatory subnetwork $V_{lead,i}$ and $V_{lag,i}$.

**Aggregating Phase Difference Estimates** With clean temporal lead/lag estimates $V_{lead,i}$ and $V_{lag,i}$ for each of the neuron pairs in our two CPGs, we can now aggregate these results into single CPG-based lead/lag estimates $V_{lead}$ and $V_{lag}$. Due to the fact that only one neuron in each CPG is active at a time and that we remove the integration transients, the act of combining the various neuron-based temporal lead and lag estimates is a simple matter of addition. This means that we can use an addition subnetwork from [14] to perform the calculations $V_{lead} = \sum_{i=1}^{n} V_{lead,i}$ and $V_{lag} = \sum_{i=1}^{n} V_{lag,i}$.

Although we now have neural estimates of the temporal lead/lag for our two CPGs, it is necessary to consolidate these estimates into a single value for the purpose of later computing phase error. To accomplish this, we use a combination of transmission, addition, and subtraction subnetworks that we refer to as a "centering subnetwork." First, the $V_{lead}$ and $V_{lag}$ signals are scaled via transmission synapses, then they are shifted upward using a tonic signal of $\frac{R_{center}}{2}$, and finally the opposing scaled signal is subtracted to produce a single, centered lead/lag estimate. Since all of the subnetworks that comprise our centering subnetwork are discussed in [13], no new design rules are necessary.

### 3.3 Eliminating Phase Error

Given a reasonable proxy for the phase difference between the two CPGs, the next step is to compare this estimate with the desired phase difference and then to take corrective actions to eliminate any error. To begin, we directly compute the phase error via a third double subtraction subnetwork. This time the first output of the double subtraction subnetwork is high when CPG $A$ leads CPG $B$ by too great of a margin, and visa versa for the second output. Despite the fact that these error signals contain the information that the control system needs, the fact that they are split among two neurons is inconvenient. As such, we implement a second centering subnetwork using the same strategy as discussed in Sec. 3.2. Like before the first output of the centering subnetwork is high when CPG $A$ leads CPG $B$ too significantly, low when CPG $A$ lags CPG $B$ too significantly, and exactly in the middle of its representational domain when there is no error. The second output of the centering subnetwork has an exactly inverse interpretation. Given this representation of the error, we can now implement any number of control schemes to eliminate the error by adjusting the currents to the drive neurons for our two multistate CPGs. This work uses a

simple proportional control scheme wherein the greater the phase error the more substantial the adjustment in the current applied to each drive neuron.

## 4 Results

Given the analytical design rules for our various functional subnetworks in Sec. 3, we now turn our attention to demonstrating that these design rules do in fact yield networks that produce the desired results. Toward this end, Fig. 1 summarizes the architectures and open loop simulation results of each of the main functional subnetworks components in their context as part of the PDC network. Neuron C1 serves as the drive neuron in the CPG shown in Fig. 1 and is set to have a small tonic current in order to achieve slow oscillation. Fig. 2 demonstrates the impact that our PDC network has on the CPGs it controls as the desired lead/lag reference is varied.

## 5 Discussion

The results presented in Sec. 4 indicate that both our individual subnetworks and the fully assembled PDC network accomplish their stated goals. More specifically, as shown in Fig. 1, our multistate CPGs produce robust oscillations that follow their specified neuron oscillation order and respond well to changes in the drive neuron state. Similarly, our basic subnetworks, including our addition, subtraction, integration, and centering subnetworks all perform their associated mathematical operations in Fig. 1 with similar quality to that reported by [14]. Despite the relative slowness of our PDC network, the results generated by this network 2 do demonstrate its ability to adjust the phase difference of two multistate CPGs using a single descending command.

Although our novel PDC network is successful at controlling phase differences, there are several potential areas for improvement, including: (1) improving the robustness of our phase difference estimate; (2) incorporating numerical optimization techniques; and (3) utilizing a more sophisticated control scheme. The first of these improvements concerns the main limiting factor of our current approach. Since our computational strategy for estimating phase difference requires not only that each pair of associated CPG neurons have over lapping signals, but also that the oscillation period of the two CPGs remain relatively similar, there are significant limitations on how quickly we can adjust the phase of our CPGs. One potential solution to these limitations is to estimate phase differences by detecting the rising and falling edges of our CPG neurons, rather than directly subtracting their voltage signals. Once the phase difference estimate it more robust to changing oscillation frequencies, it should be possible to further refine the accuracy of our functional subnetworks by employing numerical optimization techniques. While each of our individual functional subnetworks have been optimized analytically for their specific operation, it is intractable to perform this same type of analysis on the complete network; hence the value in employing a numerical approach. Finally, a more sophisticated control law, such

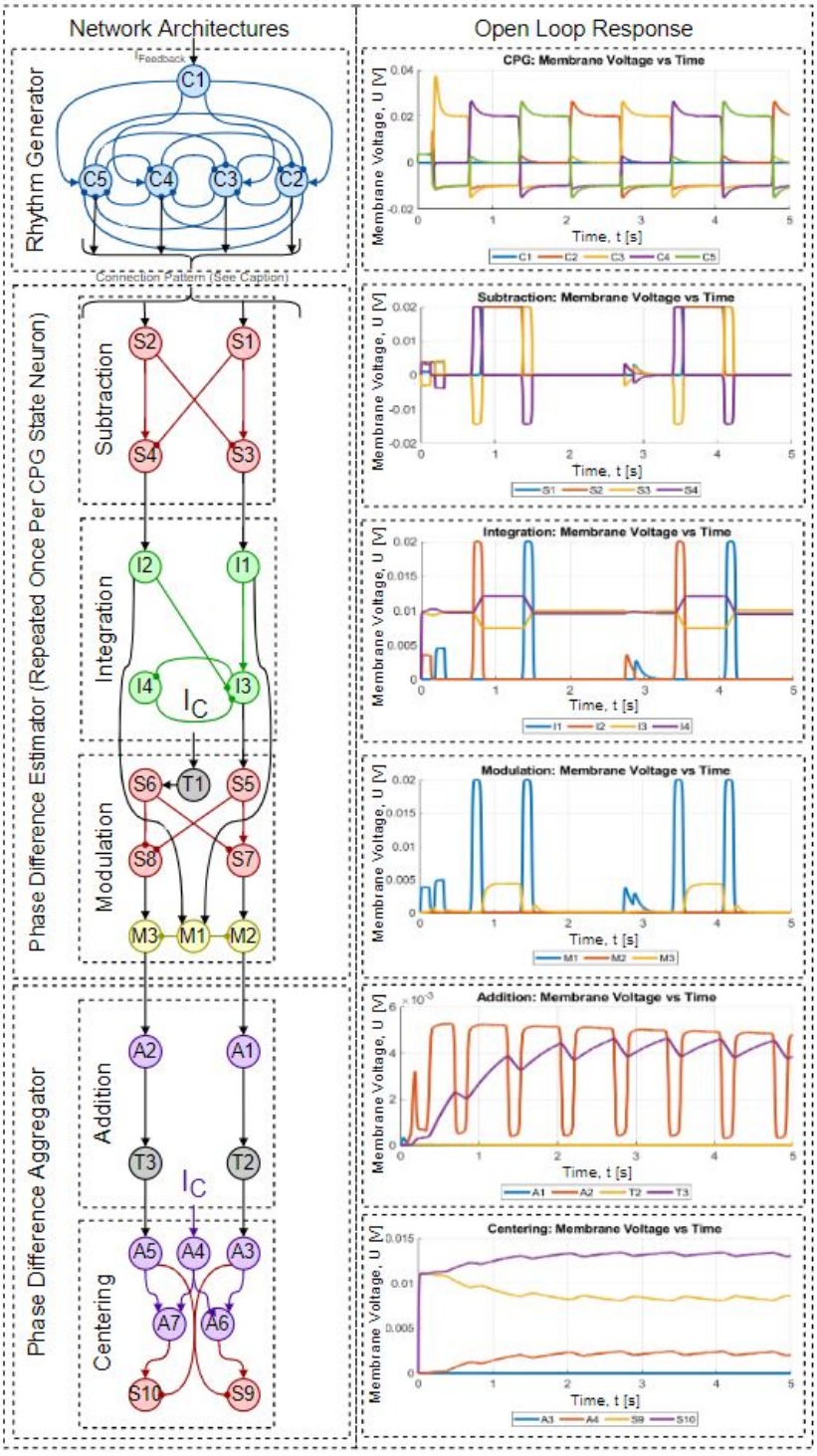


**Fig. 1.** Selected functional subnetwork architectures and their associated open loop responses in the context of a PDC network. While only a single CPG is shown in the rhythm generator, the other subnetwork responses are selected from a simulation with two multistate CPGs wherein CPG $A$ lags CPG $B$. Neurons C2-C5 from CPG A connect to neuron S1 from their associated phase difference estimator, while neurons C2-C5 from CPG B connect to S2 from their associated phase difference estimator.

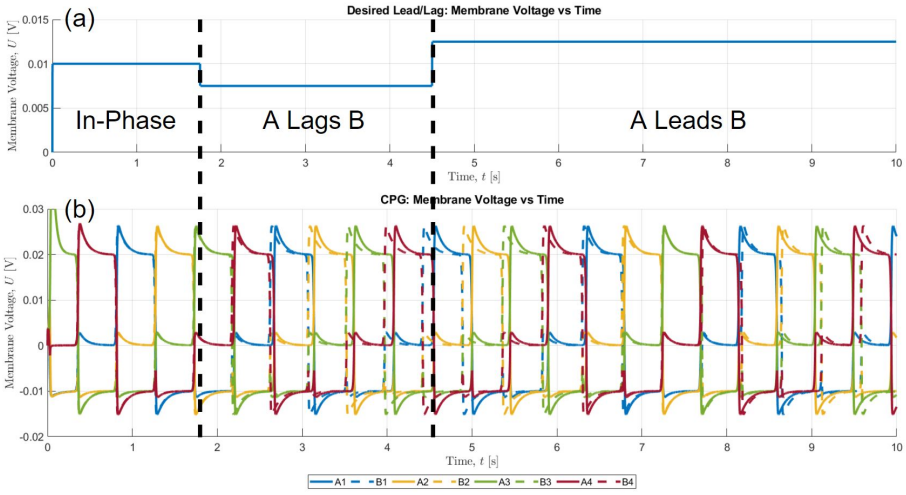


**Fig. 2.** Simulation results for a PDC network that adjusts the phase difference between two multistate CPGs comprised of four neurons each. (a) Desired CPG phase difference. (b) Membrane voltage response of each multistate CPG.

as PID or state-space control, would be able to more quickly and accurately reduce phase error given high quality feedback.

## 6 Conclusions

The functional subnetwork approach (FSA) provides a suite of analytical design tools that can be leveraged to build biological relevant subnetworks that perform basic mathematical operations. By extending the principles of the FSA, we have successfully designed a phase difference control (PDC) network that adjusts the phase difference of pairs of multistate CPGs through the use of a single descending command. As evidenced by the large body of research on the subject, CPGs are ubiquitous neural circuits that are fundamental to the proper functioning of many oscillatory animal behaviors and have therefore also been explored in a wide variety of robotics applications. For many of these applications that utilize a complex system of coupled CPGs, the PDC network approach could provide a useful tool for their coordination. In future work, we intend to improve the existing PDC methodology established here by incorporating the aforementioned areas of improvement.

## 7 Acknowledgements

The authors acknowledge support by Portland State University and NSF DBI 2015317 as part of the NSF/CIHR/DFG/FRQ/UKRI-MRC Next Generation Networks for Neuroscience Program.

## References


1. Cohen, A.H., Bard Ermentrout, G., Kiemel, T., Kopell, N., Sigvardt, K.A., Williams, T.L.: Modelling of intersegmental coordination in the lamprey central pattern generator for locomotion. Trends in Neurosciences 15(11), 434–438 (Nov 1992), https://linkinghub.elsevier.com/retrieve/pii/016622369290006T
2. Deng, K., Szczecinski, N.S., Arnold, D., Andrada, E., Fischer, M.S., Quinn, R.D., Hunt, A.J.: Neuromechanical Model of Rat Hindlimb Walking with Two-Layer CPGs. Biomimetics 4(1), 21 (Mar 2019), https://www.mdpi.com/2313-7673/4/1/21
3. Duysens, J., Van de Crommert, H.W.: Neural control of locomotion; Part 1: The central pattern generator from cats to humans. Gait & Posture 7(2), 131–141 (Mar 1998), https://linkinghub.elsevier.com/retrieve/pii/S0966636297000428
4. Forrest, S.: Genetic Algorithms. ACM Computing Surveys 28(1), 4 (Mar 1996)
5. Frazier, P.I.: A Tutorial on Bayesian Optimization. arXiv:1807.02811 [cs, math, stat] (Jul 2018), http://arxiv.org/abs/1807.02811, arXiv: 1807.02811
6. Guertin, P.A.: The mammalian central pattern generator for locomotion. Brain Research Reviews 62(1), 45–56 (Dec 2009), https://linkinghub.elsevier.com/retrieve/pii/S0165017309000812
7. Hunt, A., Szczecinski, N., Quinn, R.: Development and Training of a Neural Controller for Hind Leg Walking in a Dog Robot. Frontiers in Neurorobotics 11 (Apr 2017), http://journal.frontiersin.org/article/10.3389/fnbot.2017.00018/full
8. Ijspeert, A.J.: A connectionist central pattern generator for the aquatic and terrestrial gaits of a simulated salamander. Biological Cybernetics 84(5), 331–348 (Apr 2001), http://link.springer.com/10.1007/s004220000211
9. Ijspeert, A.J., Kodjabachian, J.: Evolution and Development of a Central Pattern Generator for the Swimming of a Lamprey. Artificial Life 5(3), 247–269 (Jul 1999), https://direct.mit.edu/artl/article/5/3/247-269/2322
10. Mantziaris, C., Bockemühl, T., Büschges, A.: Central pattern generating networks in insect locomotion. Developmental Neurobiology 80(1-2), 16–30 (Jan 2020), https://onlinelibrary.wiley.com/doi/10.1002/dneu.22738
11. Rubin, J.E., Shevtsova, N.A., Ermentrout, G.B., Smith, J.C., Rybak, I.A.: Multiple Rhythmic States in a Model of the Respiratory Central Pattern Generator. Journal of Neurophysiology 101(4), 2146–2165 (Apr 2009), https://www.physiology.org/doi/10.1152/jn.90958.2008
12. Stevenson, P.A., Kutsch, W.: A reconsideration of the central pattern generator concept for locust flight. Journal of Comparative Physiology A 161(1), 115–129 (1987), http://link.springer.com/10.1007/BF00609460
13. Szczecinski, N.S., Hunt, A.J., Quinn, R.D.: Design process and tools for dynamic neuromechanical models and robot controllers. Biological Cybernetics 111(1), 105–127 (Feb 2017), http://link.springer.com/10.1007/s00422-017-0711-4
14. Szczecinski, N.S., Hunt, A.J., Quinn, R.D.: A Functional Subnetwork Approach to Designing Synthetic Nervous Systems That Control Legged Robot Locomotion. Frontiers in Neurorobotics 11, 37 (Aug 2017), http://journal.frontiersin.org/article/10.3389/fnbot.2017.00037/full
15. Thompson, S., Watson, W.H.: Central pattern generator for swimming in *Melibe*. Journal of Experimental Biology 208(7), 1347–1361 (Apr 2005), https://journals.biologists.com/jeb/article/208/7/1347/16006/Central-pattern-generator-for-swimming-in-Melibe