Department of Informatics
King's College London
WC2R 2LS London
United Kingdom

Controlling Chaos in Memristor based chaotic circuits

Ilya Manyakin
Student Number: 1576125
Course: Intelligent Systems MSc

Supervisor: Dr Hak-Keung Lam

Thesis submitted as part of the requirements for the award of the MSc in Intelligent Systems.
7CCSMPRJ - MSc Individual Project - 2016

Abstract

This dissertation investigates the application of robust nonlinear controllers, derived based on Lyapunov stability theory, to unidirectional master-slave synchronization of Chua circuits utilizing a novel component known as a "Memristor". Convex optimization is used to solve Linear Matrix Inequalities and Sum of Squares programs to synthesize simulated controllers which stabilize the origin of the state error system, exhibiting a cubic nonlinearity, to synchronize the systems. Full state feedback methods based on Fuzzy Logic as well as Polynomial error feedback are implemented. An extension of full state feedback fuzzy logic to output state feedback is proposed and evaluated against an alternative fuzzy logic based control strategy using delayed as well as instantaneous output error. Additionally, a sliding mode control strategy using a proportional-integral sliding surface, applying control to a single variable is demonstrated to be capable of synchronizing the master and slave systems.

Keywords: Robust control, H_∞ methods, Memristor, LMI, Fuzzy logic

Contents

List of Symbols

$>, \geq$	Inequality relations
$\dot{x}(t)$	Time derivatives
$\mathbb{C}$	Complex Numbers
$\mathbb{R}$	Real Numbers
$\succ$	Matrix inequality relations
A	Matrix variable
$x(t)$	Time dependent state vector variable

List of Figures

Acknowledgements

I would like to express gratitude to my supervisor, Dr.Hak-Keung Lam, for his guidance during this project.

I also owe a big thank you to Xiaozhan Yang for the numerous discussions and useful suggestions.

1 Introduction

Chaos, in popular science known as the “Butterfly effect” is characterized by high sensitivity to initial conditions and arises naturally in many real world systems, ranging from biological and chemical reactions to weather systems and computer traffic networks. Taking its roots in the study of dynamical systems, it can be argued that the investigations of chaotic behaviour only began with the advent of the digital computer where the ability to visualize the complex behaviour of seemingly simple models under controlled conditions lead to the development of what we now call Chaos theory. The early 1990s saw breakthroughs in regard to the control of chaotic systems, namely with the work of Pecora and Carroll (PC) [1], demonstrating the possibility of synchronizing two chaotic systems, as well as that of Ott, Grebogi and Yorke [2] showing the feasibility of using small well-timed perturbations to stabilize unstable periodic orbits (UPOs) in the system state space. Recent approaches have developed these ideas, in particular the application of control theoretic techniques based on Lyapunov stability theory has allowed in the development of controllers yielding certain performance and disturbance rejection guarantees, perhaps making the control of chaotic systems feasible in a more practical and widespread setting. In parallel, the rapid pace of Moores law has seen the rise of neural networks and the subsequent activity in the development of so called “neuromorphic” architectures aimed at efficiently supporting this new paradigm at the hardware level. In this regard, a long hypothesized fourth fundamental component, relating magnetic flux linkage ϕ to electric charge q, termed “Memristor” [3] has recently seen physical implementation [4]. The memristor constitutes a passive, two-terminal device the properties of which are characterized by a nonlinear “memristance” $M(q) = d\phi/dq$ [5], analogous to electrical resistance, and which allows the memristor to retain a history of previous current flow through the component. This controllable nonlinear behaviour is also of interest circuits generating chaotic signals, whose spectral properties are desirable in a communication systems for ensuring secure and reliable signal transmission. In this context, numerous schemes based on the underlying synchronization of the transmitter and receiver chaotic systems have been proposed. It is this synchronization of chaotic systems that will be investigated within this project, based on the Chua circuit where the standard nonlinear Chua diode is replaced by a memristor component.

This project is focused on exploring the issues relating to the reduction in the number of system variables that must be measured and forced by the controller while still guaranteeing robust synchronization. This is motivated by noting that in real systems, the assumption of full state accessibility may not hold or lead to unnecessary increases in complexity and corresponding reduced fault tolerance. In secure communications this also increases the difficulty of compromising the system through limiting the information available to the attacker. Robust controllers with disturbance attenuation will be synthesized in simulation to demonstrate synchronization in cases of both full and output state measurements. An extension of a full state measurement controller based on [6] is proposed and evaluated against alternative variant [7]. Finally a graphical user interface

will be implemented to aid in demonstrations.
The report organization is as follows. In the next section, an introduction to the broader field of chaos control, synchronization and memristive systems is presented and the memristor based Chua circuit is introduced. Section three introduces formal concepts used throughout this disseration, and concludes with the statement of project objectives. In sections four and five, the main contributions of the project, consisting of implemented controllers and graphical user interface as well as the proposed extension of the full state fuzzy controller are presented. Section six concludes with a brief overview, conclusions and suggestions for future work.

2 Literature Review

2.1 Chaos in the real world

Since the proposal of the OGY method in 1990, chaos control has been applied in numerous fields, ranging from physics and chemistry where control of lasers [8], plasma [9], the Belousov-Zhabotinsky reaction [10] have been investigated. Further up the complexity scale, control of heart fibrillation [11] was an early success of this technique. In the medical field, chaos, and more specifically synchronization between chaotic systems has also broadly been implicated in its relation to physiological rhythms [12], such as those arising near the end of an epileptic seizure which sees synchronization of region of neural tissue, and is proposed as a mechanism to end its termination[13]. Chaotic phenomena in engineering have been found occur in numerous types of systems ranging from mechanical [14] and electronic and information systems [15], where the goal is often to reduce the negative impact of chaotic oscillations on system performance. In electronic systems, due to greater accessibility of the system, classical chaotic models such as Rössler, Chua and Lorenz systems have seen extensive study and extensions due to having straightforward physical implementations. A broad review on the applications of chaos control is presented in [16] to which the interested reader is referred.

2.2 Chaos control

The goal of 'control' can broadly be grouped into three categories: stabilization, excitation and synchronization. As noted by Fradkov and Andrievskii[17] these problems can be considered to be variations of the control theoretic 'tracking' problem, whereby the system state $x(t)$ is required to follow some pre-determined trajectory $x_*(t)$ in the state space, in the common case of asymptotic tracking 3.5 formulated as: $\lim_{t\to\infty}(x(t) - x_*(t)) = 0$. With the distinction lying in the properties of the reference signal $x_*(t)$. In the case of stabilization, the trajectory is periodic and chaotic otherwise. Excitation is concerned with following an arbitrary chaotic trajectory, which requires that the signal generated by the controlled system satisfy a formal "chaoticity" criterion. While in synchronization the reference signal is generated by some other chaotic system.

2.2.1 Classic chaos control methods

Similarly to stabilization of systems from a control theoretic perspective, chaotic stabilization is concerned with suppressing the chaotic oscillations that may arise from high order nonlinear terms, leading to poor system performance. As such, high sensitivity of the oscillations to perturbations makes it desirable to design controllers which utilize use 'small' perturbations to achieve the desirable performance, either fully suppressing the chaos or reducing it to a more predictable periodic form, a UPO. This smallness of control criterion is considered the distinguishing feature related chaos control, separating it from the broader control theoretic notion, where this is typically not considered.

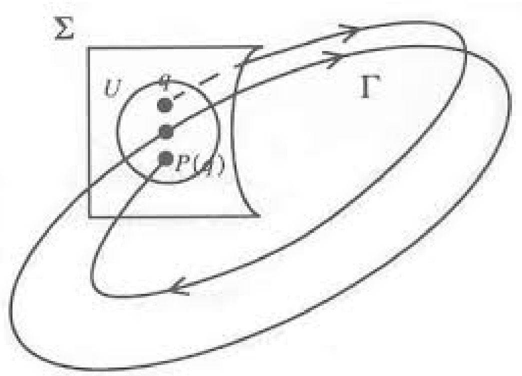

Figure 1: Illustration of the OGY method stabilizing a UPO [18]

One of the first methods proposed for stabilizing of UPOs is the OGY method [2]. The core idea utilizes the concept of recurrence of chaotic trajectories summarized as the property of chaotic systems that guarantees a trajectory will visit any arbitrarily small region of the attractor phase space, given a suitably long time. After the suitable UPOs are extracted from the system dynamics using techniques such as the Newton-Raphson-Mees algorithm [19] or a stabilizing transformation [20], a Poincare section ('$\sum$' in Fig. 1) is introduced, perpendicular to the UPO ('Γ' in Fig. 1) at the point of intersection, which is then termed a 'fixed point' denoted x_0. A first recurrence plot termed Poincare map ('$P(q)$' in Fig.1) is constructed by monitoring the intersection of subsequent system trajectory with the Poincare section and constitutes a conversion of the original continuous time system to a discrete time map. Considering a small ball around the fixed point $|x - x_0| < r$ ('U' in Fig.1) with a suitably small radius, a linear expansion of the discrete time system is obtained as $x_{k+1} = Ax_k + u_k$, where the discrete time control signal $u_k = Bx_k$ is introduced to stabilize the desired fixed point, constituting a piecewise constant control for the continuous time system [17]. For trajectories lying outside the control radius r, the corresponding control signal is set to $u_k = 0$, thus bounding the maximum control action. This method owes its success in part to the ability of applying it in systems with only a single accessible variable through a technique known as "delay coordinate embedding" [21]. However, as a consequence of requiring the trajectory to intersect the Poincare section within the control region, the time of achieve control is often unsatisfactory and additional introduction of disturbances inherent in physical system often results in long intermittent periods where control is lost.

Another classic method of stabilizing UPOs of chaotic systems, proposed by Pyragas [22] and similarly aimed at stabilizing UPOs, utilizes the idea of delayed state feedback, with the control law taking the form $u(t) = K(x(t) - x(t - \tau))$ where $K \in \mathbb{R}$ represents a constant feedback while $x(t) \in \mathbb{R}$ represents a state variable used for feedback and τ is a constant delay, equal to period of the UPO to be stabilized thus vanishes when stabilization is achieved, thus ensuring the required "smallness" of control, at the same time ensuring the system retains its unmodified dynamics and original UPO. An advantage of this technique is partly due to its transparency related to low number of parameters to be set as well as its ability to, like the OGY method, be applied to systems with only a single accessible variable.

2.3 Synchronization and Communication

Synchronization phenomena arise when, due to coupling, two or more subsystems adjust a property of their motion to a common value [23], determined by individual dynamics and well as the coupling strength and configuration. Whilst extensively studied for cases involving harmonic oscillators, chaotic synchronization is a relatively recent development beginning the in 1990s where the ability of chaotic systems to synchronize in whole or in part was discovered to be possible given suitably designed coupling or external forcing. Here we will only be concerned with unidirectional synchronization, commonly known as a "Master-Slave" system in a communications system context, where our goal will be to achieve 'identical synchronization', the exact tracking of the drive by the response. Applications of such systems in secure signal transmission have been considered, utilizing the master-slave configuration owing to chaotic carriers exhibiting the desirable properties of:

Broadbandedness: a term describing signals with a wide signal frequency spectrum, where the amplitude in the Fourier domain has a broad peak. The benefits of utilizing such signals lies in their increased resistance to channel imperfections [24]. Modulation of the message signal onto a chaotic carrier having these characteristics results increases the bandwidth over which the signal is transmitted, and is termed "spread spectrum" technology. Additionally making the signal harder to detect and intercept by monitoring only a small part of the spectrum, also increasing resistance to jamming [25].

Orthogonality: Aperiodicity of chaotic signals results in the rapid decay of the autocorrelation function of a chaotic signal, allowing for "signals generated by different generators to be assumed uncorrelated (orthogonal)" [24]. This can be exploited in multi-user channel access communication, where orthogonality is used to enable the designated receiver to extract the senders message. A classical example multi-user channel access are Code Division Multiple Access (CDMA) techniques.

Additionally, the inherent complexity and long term unpredictability of chaotic signals is appealing from the viewpoint of cryptography, where the pseudo-random nature of the signal has been suggested as a mechanism of masking of messages as they are sent over a channel. Whilst still subject to debate relating to the true cryptographic properties of chaotic signals [26], the main characteristics already make chaos a desirable phenomena from an engineering viewpoint. A simple example initially proposed known as "chaotic masking" [24] sums the message to be transmitted with the chaotic carrier signal prior to transmission, subtracting the carrier from the received signal to recover the message. Whilst too simple for real implementations, this helps to illustrate the practical challenge of implementing chaos based communications systems, specifically how does one go about synchronizing two non-identical physically separate chaotic systems given that both systems are implemented using non-ideal components and are subject to physical effects such as noise and disturbance?

2.3.1 Master-Slave synchronization

The landmark method proposed by Pecora and Caroll [1] considers decomposing the set of differential equations modelling the communication system into two identical master and slave systems, connected via coupling that links the identical states of both. As shown in their paper, asymptotic synchronization can then be achieved if the Lyapunov exponents (see 3.3) of the uncoupled slave subsystem, termed "Conditional Lyapunov exponents" (CLEs) are negative. With regard to initial conditions this can be viewed as the unforced slave subsystem 'forgetting' its initial state. The attractive property of this design relates to its simplicity as well as the availability of Lyapunov exponent algorithms [21] for this calculation, which are capable of computing the entire Lyapunov spectrum from a single variable time series, using the previously mentioned 'delay embedding'. Other methods also considering Lyapunov exponents based on alternative ways of decomposing the overall system have also been proposed, an example of such known as 'Active-Passive Decomposition' [27] instead connects the master and slave systems to a third time dependent perturbation, the 'active' component and considers the signs of the CLEs in the corresponding subsystems to demonstrate stability. In this case the autonomous system of form $\dot{z}(t) = F(z(t))$ is rewritten to include a time dependent 'active component' $s(t)$ with the master subsystem $\dot{x}(t) = f(x(t), s(t))$ and identical slave system $\dot{y}(t) = f(y(t), s(t))$. Recently the idea of controlling chaos through Lyapunov exponents has also been investigated using computational intelligence techniques such as neural networks [28][29] where the training objective function is set as the size of the maximal Lyapunov exponent (MLE). Whilst such methods are capable of stabilizing the chaos and thus theoretically synchronize systems and are often slow to converge, in part due to the high computational complexity of evaluating the MLE.

Whilst the above class of techniques based on the Lyapunov exponent is suitable for many applications, from an engineering perspective on is often interested in rigorous performance guarantees that can be derived for the controllers, allowing designs which fail in known or predictable ways or by undergoing ‘graceful degradation’, dealt with by an area of control theory known as ‘robust control’. Before considering this we first introduce the memristor and memristive systems.

2.4 Memristive systems

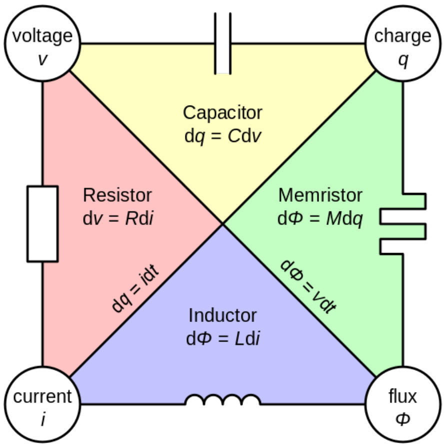

Figure 2: Relation between fundamental circuit quantities of charge, voltage, current and magnetic flux [30]

In the original derivation of the memristor, Chua utilized the idea of symmetry arguments relating to the basic circuit parameters of voltage v, charge q, current i and magnetic flux ϕ to derive an extension of the standard set of electric components: Capacitor ($q = Cv$), Inductor ($\phi = Li$) and Resistor ($v = Ri$), where the formulae in brackets are the relations for the linear components of each type . The “Memristor” then relates to quantities of electric charge q to magnetic flux ϕ, in the differential form given by the equations [3]:

$$\begin{aligned} M(q)dq &= d\phi \\ W(\phi)d\phi &= dq \end{aligned}$$

Which were used to define two types of memristors, namely charge-controlled and flux-controlled [5], where the standard relations between the above quantities can be used to rewrite the above into the form:

$$\begin{aligned} v(t) &= M(q(t))i(t) \\ i(t) &= W(\phi(t))v(t) \end{aligned} \tag{1}$$

Here the quantities $M(q)$ and $W(\phi)$ are known as "memristance" and "memductance", analogous to electrical resistance and conductance, and as noted by Chua "at low frequencies, memristive systems are indistinguishable from nonlinear resistors". One should also note, that the key attractive characteristic of such devices is that they are "passive" components, that is it cannot introduce energy into the system, similarly to the inductor, capacitor and resistor. Following the original postulation, Chua defined a broader class of "memristive" systems [31], which take the form:

$$\begin{aligned}\dot{x}(t) &= f(x(t), u(t), t)\\ y(t) &= g(x(t), u(t), t)u(t)\end{aligned}$$

forming a subclass of dynamical systems. Here $u(t)$ and $y(t)$ represent the inputs and outputs of the system while $x \in \mathbb{R}^n$ is the system state space vector. The function $f(x, u, t)$ is a vector function producing mapping inputs to $\mathbb{R}^n$ whilst $g(x, u, t)$ is a scalar valued function.

2.4.1 Memristor as Chua diode

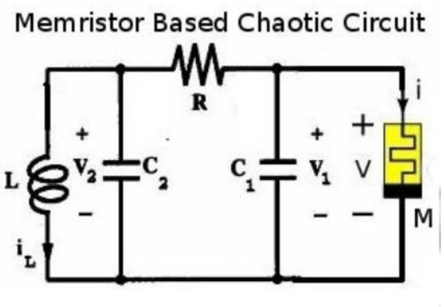

Figure 3: Chua circuit diagram. Here the memristor is labelled by 'M' [32]

Instead of considering the more general definition above, the focus will be on a Chua based memristor circuit described in [32] where the memductance $W(\phi)$ takes a quadratic form given by

$$W(\phi) = a + 3b\phi^2$$

with the system state equations being

$$
\begin{aligned}
\frac{d\phi(t)}{dt} &= \frac{-v_1(t)}{\xi} \\
\frac{dv_1(t)}{dt} &= \frac{1}{C_1}\left(\frac{v_2(t)-v_1(t)}{R} - W(\phi(t))v_1(t)\right) \\
\frac{dv_2(t)}{dt} &= \frac{1}{C_2}\left(\frac{v_1(t)-v_2(t)}{R} - i_L(t)\right) \\
\frac{i_L(t)}{dt} &= \frac{v_2(t)}{L}
\end{aligned} \tag{2}
$$

where the constants correspond to the labelled physical components properties of 3. Whilst physical memristors are still under development and thus not widely available for constructing such a circuit, an implementation using active operational amplifier components in the paper above allows for the practical realization of this system. From a theoretical perspective this system poses interesting complications to control law design, namely the resultant the cubic nonlinearity prevents the applications of controllers that assume a global Lipschitz continuous functions [33] while the 'hiding' of $\phi(t)$ behind the nonlinearity in $v_1(t)$, appears to prevent the application of nonlinear controllers such as [34] which use the controllable matrix canonical forms for their analysis.

3 Theoretic background

In this section we aim to present more rigorous definitions relating to chaotic phenomena in general as well as provide a brief overview of Lyapunov stability theory and introduce the concept of robust control, used in the following sections.

3.1 Deterministic Chaos

Given a continuous dynamic system of form where $x(t) \in \mathbb{R}^n$ represents the system state vector, $u(t) \in \mathbb{R}^m$ represents the control signal and $y(t)$ represents the measured output state, with functions $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ and $g : \mathbb{R}^n \to \mathbb{R}^k$

$$
\begin{aligned}
\dot{x}(t) &= f(x(t), u(t)) \\
y(t) &= h(x(t))
\end{aligned} \tag{3}
$$

allow the definition of the following basic concepts relating to chaotic systems [35].

Definition 3.1. Attractor
The closed set $\Omega \subset \mathbb{R}^n$ is termed an attractor if a) There exists an open set Ω_0 such that $\Omega \subset \Omega_0$ for which all trajectories $x(t)$ of the system with initial conditions in Ω_0 are defined $\forall t \geq 0$ and tend to Ω as $t \to \infty$

That is every for initial condition Ω_0, the system trajectory will eventually converge to the attractor.

Definition 3.2. Chaotic Attractor
An attractor is 'chaotic' if it is bounded and any trajectory beginning on it is a Lyapunov-unstable trajectory.

Remark 3.3. In practical terms this means that for any two initial conditions which have an initial separation in one dimension given by $\delta x(0)$, the trajectories of the time evolved states diverge exponentially in the phase space as $\delta x(t) = \delta x(0)2^{\lambda t}$ where λ and is termed a "Lyapunov exponent". The exponents measure the infinitesimal rate of divergence of state space trajectories. In order to remain bounded, the set of system Lyapunov exponents, termed "Lyapunov spectrum", is required to be negative [21].

Definition 3.4. Chaotic system
The system is chaotic if it has at least one chaotic attractor

Definition 3.5. Asymptotic Tracking
For a given desirable phase space trajectory $x_*(t)$, the system $x(t)$, controlled by a a signal $u(t)$ is said to asymptotically track the desirable trajectory if, given an initial value $x(0) \in \Omega$, where Ω is the set of initial values, it satisfies: $lim_{t\to\infty}(x(t) - x_*(t)) = 0$
.

Remark 3.6. A similar definition can be constructed for the system output $y(t)$ tracking a desired output $y_*(t)$. Here one should note that whilst application of the control $u(t)$ can be performed directly to the state variables, known as 'parametric control', in this discussion we will assume the form 3, known as 'coordinate control' which modifies the dynamical equation of the system.

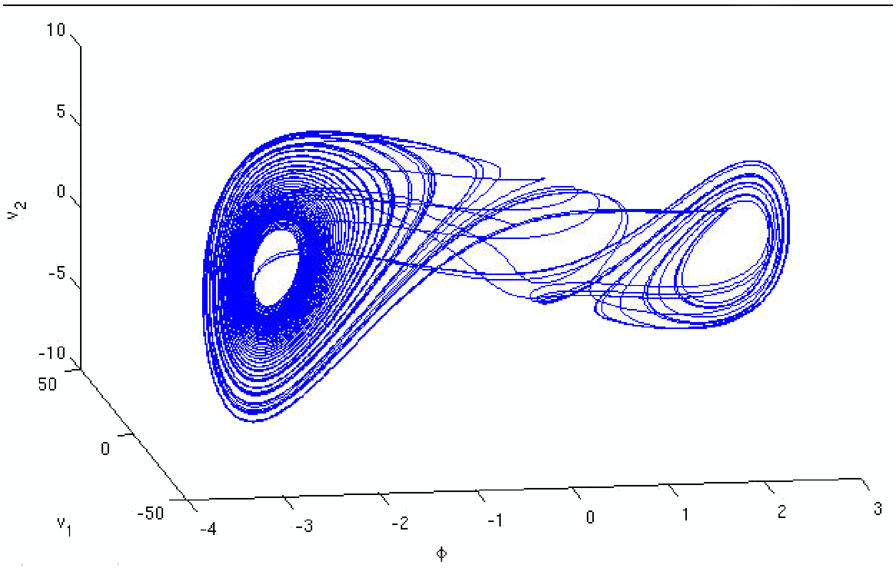

Figure 4: The chaotic attractor of the memristor based Chua circuit

3.2 Lyapunov stability theory

Based on the work of Aleksadr Lyapunov at the end of the 19th century, stability of nonlinear systems near equilibrium points can be analysed using Lyapunov stability theory. For the case of an autonomous system of the form: $\dot{x} = f(x)$ (as opposed to an non-autonomous system: $\dot{x} = f(x,t)$, for which an extension of the definitions below is possible, but omitted) where $x \in \mathbb{R}^n$ and $f : D \to \mathbb{R}^n$ one can formally define "stability" of a system, as well as two methods for proving system stability termed the "direct" and "indirect" Lyapunov methods, the former of which will be used extensively to derive control laws, while the latter is introduced for completeness and helps to suggest methods for constructing controllers in practice.

Definition 3.7. Stability[17]
For an equilibrium point x^* given a continuous, autonomous system $\dot{x} = f(x)$, defined by $f(x^*) = 0$, given the initial condition $x(0) = x_0$

1) The equilibrium termed Lyapunov stable if for any $\epsilon > 0$ there exists a $\delta(\epsilon) > 0$ such that, if $||x_0 - x^*|| < \delta$ then $\forall t \geq 0. ||x(t) - x^*|| < \epsilon$
2) The equilibrium is asymptotically stable if it is Lyapunov stable and there exists $\delta > 0$ such that if $||x_0 - x^*|| < \delta$ then $lim_{t \to 0} ||x(t) - x^*|| = 0s$.
3) An equilibrium is exponentially stable if it is asymptotically stable and there exist $\alpha > 0, \beta > 0, \delta > 0$ such that $\forall t \geq 0. ||x_0 - x^*|| < \delta$ then $||x(t) - x^*|| \leq \alpha ||x_0 - x^*|| e^{-\beta t}$

Without loss of generality the above can be translated to have the equilibrium point about $x^* = 0$.

Theorem 3.8. *Lyapunovs direct method[36]*
Let the origin $x = 0 \in D \subset \mathbb{R}^n$ be an equilibrium point for $\dot{x} = f(x)$ and $V : D \to \mathbb{R}$ be a continuously differentiable function such that:

$$\begin{aligned} &V(0) = 0, \\ &V(x) > 0, \forall x \in D - \{0\} \\ &\dot{V}(x) \leq 0, \forall x \in D \end{aligned} \tag{4}$$

Then $x = 0$ is stable. In the case when $\dot{V}(x) < 0$, $\forall x \in D - \{0\}$ the equilibrium $x = 0$ is asymptotically stable.

This is the central result utilized in the construction of the control laws that can both stabilize individual chaotic systems or synchronize two distinct systems. In effect the theorem states that given a control law $u(t)$ which modifies the original system $\dot{x} = f(x(t))$ into the form 3, for an initially unstable point $x_* = 0$, the stability of the new system around $x_* = 0$ can be shown by finding a suitable Lyapunov function which decreases everywhere along the trajectory, as the system is allowed to evolve in time and can be viewed as an analogue of the energy in the system. Thus the goal of proving stability is reduced to finding a suitable Lyapunov function. As the stability of the system itself depends on the applied control $u(t)$ one can then use this in the definition of the Lyapunov function itself, and thus by constructing a suitable Lyapunov function, we also automatically construct the control signal. For completeness now we introduce Lyapunovs direct method, as well as some of the definitions required (taken from [17]) which will also play a part in later sections.

Definition 3.9. Positive definite function
Let $V : D \to \mathbb{R}$ on a neighbourhood D including the origin. V is positive definite if $V(0) = 0$ and $\forall x \in D.V(x) > 0$.

Definition 3.10. Positive semidefinite function
Let $V : D \to \mathbb{R}$ on a neighbourhood D including the origin. V is positive definite if $V(0) = 0$ and $\forall x \in D.V(x) >= 0$.

Remark 3.11. A special case of these definitions relating to matrices, where a matrix $A \in \mathbb{R}^{n\times n}$ is termed 'positive definite' if $x^T Ax > 0$ for all non-zero $x \in \mathbb{R}^n$. Extensions for semidefinite as well as negative definite cases are similarly performed, with the latter consisting of a change in the direction of the inequality, $x^T Ax < 0$.

Definition 3.12. Hurwitz matrix
A matrix $A \in \mathbb{R}^{n\times n}$ is termed Hurwitz (asymptotically stable) if and only if $Re(\lambda_i) < 0, \forall i = 1, 2, ..., n$. Where λ_i represent the eigenvalues of the matrix A.

Theorem 3.13. *Lyapunov equation*
For $A \in \mathbb{R}^{n\times n}$ the following statements are equivalent:
1) A is Hurwitz
2) For all $Q = Q^T > 0$ there exists a unique $P = P^T > 0$ satisfying the Lyapunov matrix equation: $A^T P + PA = -Q$

Here the Lyapunov matrix equation arises as a consequence of choosing the Lyapunov function to have quadratic form: $V(x(t)) = x^T(t)Px(t)$ and for a linear system $\dot{x}(t) = Ax(t)$ this yields for following rate of change: $\dot{V}(x(t)) = x^T(t)(A^T P + PA)x(t) = -x^T(t)Qx(t)$, proving system stability in the case when $Q = Q^T$ exists. As mentioned previously the Lyapunov function $V(x)$ acts as an analogue of system energy. The time derivative being negative is then analogous to the rate of energy dissipation at the given point x in the phase space.

Theorem 3.14. *Lyapunovs indirect method[36]*
Let $x = 0$ be an equilibrium point of $\dot{x}(t) = f(x(t))$ where f is a continuously differentiable function such that $A = \frac{\partial f}{\partial x}|_{x=0}$ then:
1) The origin is asymptotically stable if $Re(\lambda_i) < 0$ for all eigenvalues of A.
2) The origin is unstable if $Re(\lambda_i) > 0$ for one or more of the eigenvalues of A.

It should be noted that this the indirect method is not applicable in the case when $Re(\lambda_i) = 0$ for some i, and hence in this case an alternative stability analysis method is required.

3.3 Robust synchronization

During the discussion about synchronization of systems in the previous section, consideration was mainly focused on the case of ideal identical systems. In a practical setting additional complications arise from external disturbances, parameter uncertainty and mismatch as well as other physical effects, which may result in the theoretically viable synchronization strategy performing badly on a physical system ('plant'). To illustrate further, consider the standard three step process involved in designing controllers. The first step, consists of constructing a plant model of the process to be controlled from measurements of the system. Here, uncertainty and offset error in the measurements as well as the modelling error, introduces disparities between the plant and its model. The second step, design of the control law may make additional simplifying assumptions to the plant model in order to allow the application of the selected synthesis procedure to derive the control law, such as neglecting high order terms. Finally, the derived control law is used to construct a physical controller, which is applied back to the plant. Thus one effectively ends up with an approximation of the control law, derived on the approximated plant attempting to stabilize the original system. Whilst this process can result in a controller capable of good performance under standard operating conditions, for the example of a safety critical setting it is often desirable to consider the controllers performance in the 'worst-case' scenario. For the case of synchronization, this 'worst-case' scenario corresponds to a disturbance to the system, explicit modelling of which is studied in 'robust control theory'. Introduced in the following subsection is a broad class of techniques known as H_∞ control, characterized by providing upper bounds on the disturbance induced response on the controlled system, used in the following section to perform the synthesis of the controllers.

3.3.1 H_∞ control

In order to move towards defining the H_∞ norm it let us review some basic definitions first, which we will build upon and will help understand what follows. The interested reader is directed to [37] for further discussion.

Definition 3.15. Norms[38]
Euclidean norm (2-norm):

$$||x||_2 = \left(\int_{-\infty}^{\infty} |x(t)|^2 dt \right)^{1/2}$$

p-norm:

$$||x||_p = \left(\int_{-\infty}^{\infty} |x(t)|^p dt \right)^{1/p}, \text{ for } 1 < p < \infty$$

∞-norm:

$$||x||_\infty = \sup_{t \in \mathbb{R}} |x(t)|, \text{ for } p = \infty$$

[‘sup’ denotes the ’supremum’ or peak of the value of function $|x(t)|$]

The purpose of defining the above is that now, we can use these to define more complex ‘induced’ norms, which can measure the effect of operators, of which the H_∞ is an example.

Definition 3.16. Matrix Norm[39]
If vector norms on $\mathbb{R}^n$ and $\mathbb{R}^m$ are given then one can define an matrix operator norm of the space of $m \times n$ matrices as the largest value, given by

$$||A|| = \sup \frac{||Ax||}{||x||} \text{ where } A \in \mathbb{R}^{m \times n}, x \in \mathbb{R}^n \text{ with } x \neq 0$$

This matrix norm is a measure of how the matrix A scales the x to which it is applied, and is measured by what are known as ‘singular values’ of A. Whilst this is useful, one may then want to generalize this description further to consider more general operator types, such as those mapping complex numbers $s \in \mathbb{C}$ from scalar to matrix values in the form: $f(s) : \mathbb{C} \to \mathbb{C}^{n \times m}$. The space of such functions which are analytic (broadly described as being infinitely differentiable) on the right hand side of the complex plane (that is $Re(s) > 0$) is then known as the H_∞ or ‘Hardy’ space. Here the operator norm can be defined as being[39]

$$||f||_\infty = \sup_{Re(s)>0} \bar{\sigma}(f(s))$$

where $\bar{\sigma}$ denotes the function returning the singular values of $f(s)$.

In order to finally reach the definition of H_∞ we now consider the following example [37] of a proper stable Linear Time Invariant (LTI) system, in its state space given by equations:

$$\dot{x}(t) = Ax(t) + Bw(t)$$
$$y(t) = Cx(t) + Dw(t)$$

Recast into the Laplace domain to obtain the system transfer function is $G(s) = C(sI - A)^{-1}B + D$, and the corresponding H_∞ norm:

$$||G||_\infty = \sup_{||w||_2=1} ||Gw||_2 = \sup_{||w||_2=1} ||z||_2 = \sup_{\omega} \bar{\sigma}(G(j\omega))$$

Here the transfer function $G(s)$ gives the ratio between the input signal $w(t)$ and the output signal $z(t)$, where the former assumed to have unit norm. Thus, the H_∞ norm gives an energy to energy induced norm, giving an upper bound on the energy of the output $z(t)$ of the system given the energy of the disturbance signal $w(t)$. For the more general case of a nonlinear system [40] (Note: we have switching from the Laplace domain back to a state space model)

$$\begin{aligned} \dot{x} &= f(x, u, w) \\ z &= h(x, u, w) \end{aligned}$$

The H_∞ norm is expressed as $||z||_2^2 \leq \eta ||w||_2^2$ where η is a preassigned tolerance level (the H_∞ bound). The authors note that this corresponds to the closed-loop system being dissipative , that is the energy supply rate for the above H_∞ bound corresponds to $\eta ||w||_2^2 - ||z||_2^2$.

Definition 3.17. H_∞ performance
Given a system of the form [40] a control law $u(t)$, applied for all $t \geq 0$ is a H_∞ controller if, for a given level of attenuation η, a norm-bounded disturbance $w(t)$ and output error function $z(t)$ the following relation holds:

$$\left(\int_0^\infty z(t)^2 dt \right)^{1/2} \leq \eta \left(\int_0^\infty w(t)^2 dt \right)^{1/2}$$

3.4 Convex Optimization

In the previous two subsections we have considered the type of constraints that we would like to impose on the control law, namely via the Lyapunov function $V(x)$, to ensure asymptotic stability of the stabilized equilibrium and H_∞ bound, to ensure the error of the controlled system remains bounded for any finite energy disturbance. To find feasible solutions to these constraints we will utilize a technique known as convex optimization, which is an extension of the well known linear programming problem.

Definition 3.18. Optimization problem [41]
Minimize $f_0(x)$
Subject to $f_i(x) \leq b_i$, $i = 1, ..., m$

Where $f_0(x) : \mathbb{R}^n \to \mathbb{R}$ is termed the 'objective' functions and $f_i(x) : \mathbb{R}_n \to \mathbb{R}$ are 'constraint' functions. Problems of this type are typically characterized by the types of objective and constraint functions considered, in linear programming both f_0 and f_i are linear functions. Similarly, in convex optimization the objective function and constraints are convex functions.

Definition 3.19. Convex function [41]
A function $f(x) : \mathbb{R}^n \to \mathbb{R}$ is convex if $\forall x, y \in \mathbb{R}^n$ and for any $a, b \in \mathbb{R}$ where $a + b = 1$ and $a, b \geq 0$ it satisfies:

$$f(ax + by) \leq af(x) + bf(y)$$

Definition 3.20. Convex set
A set C is convex if the line segment between any two points in C lies in C, that is, $\forall x_1, x_2 \in C$ and any α for which $0 \leq \alpha \leq 1$ we have: $\alpha x_1 + (1 - \alpha)x_2 \in C$.

Each of the convex constraints has a set of solutions that forms a convex set, the intersection of which in turn forms the set of feasible solutions to the optimization problem, which is also convex. A useful example a convex set is the set of positive semidefinite cones.

Definition 3.21. Positive semidefinite cone [41]
Let $S^n = \{X \in \mathbb{R}^{n \times n} | X = X^T\}$ be the set of symmetric $n \times n$ matrices. The set of symmetric positive semidefinite definite matrices is then defined as

$$S^n_+ = \{X \in S^n | X \geq 0\}$$

The reason for utilizing this technique in the design of the controller is that, once formulated as a convex optimization problem, algorithms known as 'interior point methods' can be utilized to automate the process of finding a solution. This simplifies the design process by removing the need to perform symbolic manipulations and also scales well with problem size in comparison to older simplex algorithms [42], with the distinction between the two lying in the way in which the two traverse the set of potential solutions.

3.4.1 Linear matrix inequalities

To introduce the concept of Linear Matrix Inequalities (LMIs) and give a preliminary simple illustration of how they can be applied to derive control laws, consider a LTI system, with a linear full state feedback law:

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$u(t) = Kx(t)$$

Combining the two equations yields: $\dot{x}(t) = (A + BK)x(t)$ where A,B,K are constant matrices. Whilst one may attempt to prove stability in this case by considering the condition for negative eigenvalues of the transfer matrix $A + BK$, instead to relate this to the previous discussions now we will introduce a function $V(x(t)) = x^T(t)Px(t)$, where $P = P^T$, a symmetric matrix. Taking the time derivative yields

$$\frac{dV}{dt} = \dot{x}^T(t)Px(t) + x^T(t)P\dot{x}(t) = x^T(t)[(A+BK)^TP + P(A+BK)]x(t)$$

In order to apply Lyapunovs direct method 3.8, $V(x(t))$ must satisfy constraints: $V(x) > 0, \forall x \neq 0$ and $\dot{V}(x) < 0, \forall x \neq 0$, which leads to the pair of matrix definiteness constraints

$$\begin{gathered} P > 0 \\ A^TP + PA + K^TB^TP + PBK < 0 \end{gathered}$$

This last matrix inequality termed Bilinear Matrix Inequality (BMI) due to being linear in both variables P and K, a non-convex constraint. To reduce this to a convex constraint apply a linearising variable change [37], denoting $X = P^{-1}$ and using it to pre- and post-multiply the BMI obtain: $XA^T + AX + XK^TB^T + BKX < 0$. Setting $N = KX = KP^{-1}$ results in the final form where each constraint is a LMI as each term contains only one of the matrix variable

$$\begin{gathered} X > 0 \\ XA^T + AX + N^TB^T + BN < 0 \end{gathered}$$

Once in this form, this problem becomes solvable using standard solvers such as those in MATLAB® LMI toolbox.

3.5 Fuzzy logic

In the above example, a LTI system with a linear feedback gain controller was considered to demonstrate the application of LMIs, and at first may appear to be restricted to the above class of problems. To allow the extension of this method to the class of nonlinear systems we now turn to Fuzzy logic, which can be used to decompose the nonlinear system into a finite number of linear subsystems, used to exactly model the original system within a finite region of the state space due to the universal approximation property of fuzzy models [43]. In such systems, rules are represented in the IF-THEN format where the distinction between different types of fuzzy inference systems is the form of the consequent term following the ELSE. In nonlinear systems control the most commonly used is the Takagi-Sugeno (TS) model [44] where the i^{th} rule takes the form[6]:

$$\begin{gathered} \text{IF } f_1(x(t)) \text{ is } M_1^i \text{ AND... AND } f_p(x(t)) \text{ is } M_p^i \text{ THEN} \\ \dot{x}(t) = A_i x(t) + B_i u(t), \\ y(t) = C_i x(t) + D_i u(t) \end{gathered} \tag{5}$$

Here $x(t) \in \mathbb{R}^n$ is the state vector, $y(t) \in \mathbb{R}^m$ is the measured state output and $A_i \in$

$\mathbb{R}^{n\times n}$, $B_i \in \mathbb{R}^{n\times k}$ and $C_i \in \mathbb{R}^{m\times n}$ and $D_i \in \mathbb{R}^{m\times k}$ are constant matrices.The label M_k^i a fuzzy set, with a corresponding membership function $\mu_{M_k^i}$ which we denote as μ_{ik} with the indices ranges $i = 1, ...r$, $k = 1, ..., p$. Here the functions $f_k(x(t))$ are 'premise' variables, constituting some known functions of system state. The degree of membership of each premise variable $f_k(x(t))$ in the fuzzy set M_k^i is then given by $\mu_{ik}(f_k(x(t))$. The final form, giving the relative size of firing of the i^th rule is then:

$$w_i(x(t)) = \frac{\prod_{k=1}^{p} \mu_{ik}(f_k(x(t)))}{\sum_{i=1}^{r} \prod_{k=1}^{p} \mu_{ik}(f_k(x(t)))}$$

Where the above defined fuzzy 'AND' operator is assigned be arithmetic multiplication, with $w_i(x(t))$ denoting the grade of membership of $x(t)$ in the fuzzy set formed by the intersection of fuzzy sets M_k^i for $k = 1, ...p$. The above formulation then results in the defining property of membership functions:

$$\begin{array}{c} 0 \leq w_i(x(t)) \leq 1, \forall i \\ \sum_{i=1}^{r} w_i(x(t)) = 1 \end{array} \tag{6}$$

Thus allowing the original nonlinear system to be written as a weighted average of linear subsystems, defined in the consequent of the above fuzzy rule:

$$\begin{aligned} \dot{x}(t) &= \sum_{i=1}^{r} w_i(x(t))[A_i x(t) + B_i u(t)] \\ y(t) &= \sum_{i=1}^{r} w_i(x(t))[C_i x(t) + D_i u(t)] \end{aligned} \tag{7}$$

Note that he nonlinearity is hidden inside the membership functions $w_i(x(t))$. This allows linear control techniques to be applied to each of the linear subsystems, and a fuzzy nonlinear controller is synthesized by computing a weighted average of the linear controllers, in the same as performed above for approximating a nonlinear system. This method is termed Parallel Distributed Compensation (PDC) [45]. Whilst in a practical setting one would need to infer the model from data, due to the availability of an nonlinear state space model for the memristor Chua circuit 2 we can utilize the 'Sector Nonlinearity' obtain an exact representation of the original model in the form 7.

3.5.1 Sector Nonlinearity

To create a TS fuzzy model of a nonlinear system, the sector nonlinearity method [46] approach uses the idea computing linear bounds on the function and performing nonlinear interpolating between those to reproduce the behaviour of the original system. To bound a nonlinearity of order > 1 it is required that the model is constructed on a finite interval $x \in [a, b]$, limiting the operating range. Whilst important from a theoretical perspective, physical variables are often are bounded in practical systems this issue can be overcome by suitably tuning the operating range, as is the case for the memristor system. Once defined, the linear bounds are then combined to construct the linear subsystems while the interpolation functions are composed to make the 'membership grades'.

To illustrate, consider the initial system $\dot{\mathbf{x}}(t) = f(\mathbf{x}(t))$, where $x(t) \in \mathbb{R}^n$ and $f : \mathbb{R}^n \to \mathbb{R}^n$. Rewriting this into a matrix form $\dot{\mathbf{x}}(t) = A(\mathbf{x}(t))\mathbf{x}(t)$ where $A : \mathbb{R}^n \to \mathbb{R}^{n\times n}$ is a matrix of polynomial terms, for example

$$A(\mathbf{x}) = \begin{pmatrix} 2 & 1 \\ x_1x_2 & 3 \end{pmatrix}$$

Now assume $x_1 \in [-a, a]$, $x_2 \in [-b, b]$, for $a, b > 0$, defining $z_1 = x_1x_2$ as the fuzzy premise variable we obtain $\max(z_1) = ab$ and $\min(z_1) = -ab$. Substituting these limits back into $A(\mathbf{x})$ we obtain two linear subsystem matrices, bounding the original from above and below:

$$A_1 = \begin{pmatrix} 2 & 1 \\ ab & 3 \end{pmatrix}$$

$$A_2 = \begin{pmatrix} 2 & 1 \\ -ab & 3 \end{pmatrix}$$

Now consider the premise variable z_1 and two fuzzy sets M_1 and M_2, with corresponding membership functions $w_1(z_1)$ and $w_2(z_1)$ which equate to 1 in cases when $z_1 = \max(z_1) = ab$ and $z_1 = \min(z_1) = -ab$ respectively termed 'fuzzy partitioning'.

$$z_1 = \max(z_1)w_1(z_1) + \min(z_1)w_2(z_1) = ab(w_1(z_1) - w_2(z_1))$$

Within the given ranges $x_1 \in [-a, a]$, $x_2 \in [-b, b]$ these functions also satisfy 6, thus by rearranging the above and using the property of fuzzy membership grades obtain

$$\begin{aligned} w_1(z_1) &= \frac{1}{2}(1 + \frac{z_1}{ab}) \\ w_2(z_1) &= \frac{1}{2}(1 - \frac{z_1}{ab}) \end{aligned}$$

exactly modelling the initial system within the given operating range. This method can be extended further to involve more premise variables, with additional bounds and corresponding pairs of fuzzy sets. In this more general case it one considers each of the state variables separately, introducing the additional step of constructing the overall subsystem membership grade from the individual state variable membership grades. To illustrate, consider the same system, but now with premise variables $x_1 = z_1, x_2 = z_2$ where fuzzy sets are represented by membership functions $M^0(z_1), M^1(z_1)$ and $N^0(z_2), N^1(z_2)$. In a similar way, these functions have the form:

$$M^{0,1}(z_1) = \frac{1}{2}(1 \pm \frac{z_1}{a})$$
$$N^{0,1}(z_2) = \frac{1}{2}(1 \pm \frac{z_2}{b})$$

Noting that now, the fuzzy antecedent includes two premise variables and thus the form:

IF $z_1$is M^i AND $z_2$is N^j THEN ...

By using, as noted previously, multiplication as arithmetic equivalent of the fuzzy 'AND' operator we obtain the following fuzzy membership functions for each of the rules:

$$h_1(z_1, z_2) = M^0(z_1)N^0(z_2)$$
$$h_2(z_1, z_2) = M^0(z_1)N^1(z_2)$$
$$h_3(z_1, z_2) = M^1(z_1)N^0(z_2)$$
$$h_4(z_1, z_2) = M^1(z_1)N^1(z_2)$$

Here one can note that, whilst the number of fuzzy rules grows exponentially in the number of premise variables, the inherent pattern of indices (specifically in the progression $[0, 0], [0, 1], [1, 0], [1, 1]$) indicates how one may automate this step of constructing fuzzy membership functions from the nonlinear system.

3.6 Project Objectives

The aim of this project is to apply robust nonlinear control techniques based on H_∞ and Lyapunov stability theory to design continuous-time control laws for unidirectional identical synchronization between Chua circuits subject to disturbance. The drive and response circuits are taken to have identical structure 2 based the physical implementation [32]. The goal of the project is to act as a preliminary step to the implementation of a physical chaotic synchronization system using the above as the drive and response circuits. First full state feedback controllers will be implemented in simulation and compared, to demonstrate the feasibility of identical synchronization using both full and partial response state forcing. The latter is investigated is used feasibility of the approach in this case. Next, output state feedback methods based on fuzzy logic will be applied to overcome the strong assumptions on the availability of all state variables for measurement. Finally, for purposes of demonstration the developed software will be augmented with a graphical user interface for demonstration purposes.

4 Full state feedback control

In this section we consider the synthesis of full state feedback controllers that are characterized by using the full state of both drive and response systems for the generation of a control signal. A Takagi-Sugeno fuzzy logic controller based [6] is generated using the MATLAB LMI toolbox. Next, a polynomial error feedback controller based on [47] is constructed using the SOSTOOLS [48] extension to MATLAB. The final controller [49] considered falls into the 'Sliding mode' class, is analyzed due to its relaxed assumption on the rank of the control transfer matrix, achieving synchronization by controlling only a single responses state. The section concludes with a discussion and comparison of these methods.

For the purposes of simulation, parameter values of 2 taken from [32], for the nominal model are set to:

$$
\begin{aligned}
a &= -6.67 \times 10^{-4}\Omega^{-1} \\
b &= 2.9 \times 10^{-5}\Omega^{-1}Wb^{-2} \\
C_1 &= 6.8 \times 10^{-9}F \\
C_2 &= 6.8 \times 10^{-8}F \\
R &= 2 \times 10^{3}\Omega \\
L &= 1.8 \times 10^{-2}H \\
\tau &= 1/\xi = 2.5947 \times 10^{3}
\end{aligned} \tag{8}
$$

Their use is motivated by the availability of a direct physical realization of a circuit with these settings, experimentally found to produce chaos. An additional benefit is expected similarity of physical system response time scales with that of the simulation.

4.1 Fuzzy logic based controller

The first controller to be implemented was a full state switching controller proposed in [6]. The drive and response systems are special cases of 7 and take the form:

$$\dot{\hat{x}}(t) = \sum_{i=1}^{r} \hat{w}_i(\hat{x}(t))[\hat{A}_i\hat{x}(t)] \tag{9}$$

$$\dot{x}(t) = \sum_{i=1}^{r} w_i(x(t))[A_i x(t) + Bu(t)] \tag{10}$$

Due to considering full state feedback control, the output term $y(t) = x(t)$ and consequently the error vector is defined as $e(t) = x(t) - \hat{\mathbf{x}}(t)$. To prove that the two systems are identically synchronized, instead of utilizing the CLEs as in the case of PC synchronization, instead here one considers the error system, from the above definitions given as

$$\dot{e}(t) = \dot{x}(t) - \dot{\hat{x}}(t) \tag{11}$$

By showing that the control stabilizes this error system at the origin we obtain an identical synchronization of the drive-response system. Note that in this case the constant $B_i = B$ matrices are required to be invertible and thus are full rank, resulting in a controller that forces all response system variables, with the control law:

$$u = B^{-1} \sum_{i=1}^{r} m_i(x(t)) G_i e(t) \tag{12}$$

Here one should note that use of $m_i(x(t))$ error membership function. Whilst in a theoretical setting one may initially assume the the system membership functions are known, in a practical setting these may be complex and thus, while the system membership function can be learned from data [50], in practice one often opts for a simpler design of the controller. This motivates the use of a switching controller where the switching law [6]

$$m_i(e(t)) = \frac{1}{2}[1 - sign(e(t)^T P G_i e(t))] \tag{13}$$

with $P = P^T \succ 0$ being a matrix of constants. By first rewriting 11 using the given forms 9 and 10 one can group mismatched terms into a disturbance vector $m_e(t)$ yielding the simplified system form

$$\dot{e}(t) = \sum_{i=1}^{r} [A_i e(t) + Bu(t) + m_e(t)] \tag{14}$$

At which point, substituting the above control law and using the Lyapunov function $V(t) = e^T(t) P e(t)$ (similar to the form discussed in section 3.4.1). By then rewriting the system in matrix form and imposing a negative definiteness constraint on the main matrix term one obtains the following theorem, after applying the a similar linearizing transformation of variables as discussed in 3.4.1

Theorem 4.1. *Fuzzy Full State Error Feedback controller [6]*
The error system 11 with the drive and response systems represented by fuzzy models if form 9 and 10 respective, resulting in the error system

$$\dot{e}(t) = \sum_{i=1}^{r} w_i(x(t))(A_i + G_i)e(t) + \sum_{i=1}^{r} (m_i(e(t)) - w_i(x(t)))G_i e(t) + m_e(x(t), \hat{x}(t))$$

where $m_e(x(t), \hat{x}(t)$ represent a disturbance signal. Is stabilized by a PDC control law of the form 12 where the controller membership functions take the form 13. If there *exists a scalar η, symmetric matrix $X \in \mathbb{R}^{n\times n}$ (where $P = X^{-1}$) and variable matrices $N_i = G_i X \in \mathbb{R}^{n\times n}$ which are solutions of*

minimize η subject to:

$$\begin{aligned}
&\eta > 0,\\
&X \succ 0,\\
&\begin{bmatrix} XA_i^T + A_iX + N_i^T + N_i + X & X \\ X & -\eta X \end{bmatrix} \prec 0
\end{aligned}$$

With the resulting controller satisfying the H_∞ disturbance rejection criterion of 3.17 with the bound given by η.

In order to solve this within the MATLAB LMI Toolbox using the 'gevp' solver used for this type of problem, first one must convert the above into an alternate required form used by the solver [51]

$$\begin{aligned}
C(z) &< D(z)\\
0 &< B(z)\\
A(z) &< \eta B(z)
\end{aligned}$$

Here the terms $A(z), B(z), C(z), D(z)$ represent linear matrix terms and where instead of considering unknown matrices X, N_i, with each element constituting a decision variable, the set decision variables for the whole problem is restructured into a single vector 'z'. In order to obtain the above form, one uses the Schur complement.

Lemma 4.2. *Schur Complement [52]*
Let F be an affine function represented in the form:

$$F(x) = \begin{pmatrix} F_{11}(x) & F_{12}(x) \\ F_{21}(x) & F_{22}(x) \end{pmatrix}$$

Where $F_{11}(x)$ is a square matrix. Then $F(x) \prec 0$ if and only if

$$\begin{aligned}
&F_{22}(x) \prec 0\\
&F_{11}(x) - F_{12}(x)[F_{22}(x)]^{-1}F_{21}(x) \prec 0
\end{aligned}$$

Where the first transformed inequality is redunant due to being previously specified in the problem. Thus obtaining the final form of the matrix inequality constraint for each subsystems that can be directly solved using the MATLAB LMI 'gevp' solver.

$$X \prec \eta(-XA_i^T - A_iX - N_i^T - N_i - X)$$

For the linear terms of the memristor chua circuit model 2 obtain the linear matrix:

$$A = \begin{pmatrix} 0 & -\tau & 0 & 0 \\ 0 & -\frac{1+aR}{C_1 R} & \frac{1}{C_1 R} & 0 \\ 0 & \frac{1}{C_2 R}, & -\frac{1}{C_2 R} & -\frac{1}{C_2} \\ 0 & 0 & \frac{1}{L} & 0 \end{pmatrix} \tag{15}$$

The additive contributions from the nonlinear terms arising from applying the sector nonlinearity technique 3.5.1 and labelled as B_1, B_2, with the corresponding membership functions w_1, w_2, for the premise variable $z = \phi v_1$ are then given by:

$$B_1 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ \frac{3b}{C_1}\max(z) & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{16}$$

$$B_2 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ -\frac{3b}{C_1}\max(z) & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{17}$$

$$w_1(\mathbf{x}) = \frac{1}{2}[1 + z/\max(z)] \tag{18}$$

$$w_2(\mathbf{x}) = \frac{1}{2}[1 - z/\max(z)] \tag{19}$$

$$\tag{20}$$

Here $max(z)$ is a constant determined by the operating range of ϕ and v_1, which for this simulation are set to: $\phi \in [-3, 3]$, $v_1 \in [-100, 100]$. Applying the 'gevp' solver to the resulting fuzzy system, with the parametrization given given by 8 then yields the feedback gains (with $\eta = 3.262467e - 7$)

$$G_1 = \begin{pmatrix} 3.3539 & 0.1924 & -0.0011 & 0.0034 \\ 0.1925 & -3.3508 & -0.0060 & 0.0018 \\ -0.0011 & -0.0060 & -3.3958 & 0.7643 \\ 0.0034 & 0.0018 & 0.7630 & -3.4872 \end{pmatrix} \times 10^7$$

$$G_2 = \begin{pmatrix} 3.3551 & -0.1910 & 0.0011 & -0.0034 \\ -0.1910 & -3.3521 & -0.0059 & 0.0018 \\ 0.0011 & -0.0060 & -3.3958 & 0.7643 \\ -0.0034 & 0.0018 & 0.7631 & -3.4872 \end{pmatrix} \times 10^7$$

At this point, before moving on to the second type of controller, it is useful to note one of the challenges relating to control theoretic techniques, as mentioned briefly in the introduction and in other reviews [17], namely the 'smallness' of control requirement.

As one can see above, the synthesized feedback gains are of the order 10^7, which whilst ensuring fast stabilization of the error system, thus achieving synchronization, also may not be feasible in physical implementations. Before discussing this and related issues further, let's consider an alternative controller design method based on global polynomial error feedback.

4.2 Polynomial error feedback controller

At its core, a nonlinear controller can be considered to approximate the inverse dynamics of the system, which when applied results in a stable system. A simple example of this is an active control approach, where a control law directly subtracts a known nonlinearity and introduces a linear feedback gain to asymptotically stabilize the resulting system [53]. Whilst simple to for the case of synchronizing nominal models with identical parameters, the issue becomes more complicated with the introduction of disturbances. Here, unlike the case of fuzzy logic method, by modelling directly the inverse dynamics one may avoid the potential issues arising from the finite operating range of the fuzzy controller, as well as reduce circuit complexity through replacing the individual linear subsystem feedback gains, with a single global polynomial feedback. With this motivation before considering the central theorem taken from [47] used to construct the controller it is useful to introduce the sum of squares (SOS) techniques as a subset of the already familiar convex optimization problems.

A real polynomial $f(x(t))$ is of SOS form if there exist other real polynomials $g_i(x(t))$ for which $f(x(t)) = \sum_i g_i^2(x(t))$. Due to the polynomials being real this implies $f(x(t)) \geq 0$. Given a suitable vector of monomials $z(x)$ the above form can then be rewritten in the matrix form familiar from convex optimization [48]

$$f(x) = z^T(x)Pz(x)$$

with P being a constant positive semi-definite matrix, allowing the reformulation of the search for a SOS function as a semi-definite program, which once found can be used to construct the corresponding SOS polynomial. In the context of chaotic synchronization, a full state output model of the drive and response systems can be written in polynomial matrix form given by

$$\dot{\hat{x}}(t) = \hat{A}(\hat{x}(t))\hat{x}(t) \tag{21}$$

$$\dot{x}(t) = A(x(t))x(t) + Bu(t) \tag{22}$$

with $\hat{x}(t)$ denoting drive state and $x(t)$ denoting response state. The original free memristor model 2 rewritten in the above form 21, is then

$$\begin{pmatrix} \dot{\phi}(t) \\ \dot{v_1}(t) \\ \dot{v_2}(t) \\ \dot{i_L}(t) \end{pmatrix} = \begin{pmatrix} 0 & -\tau & 0 & 0 \\ -\frac{3b}{C_1}\phi(t)v_1(t) & -\frac{1+aR}{C_1R} & \frac{1}{C_1R} & 0 \\ 0 & \frac{1}{C_2R}, & \frac{-1}{C_2R} & -\frac{1}{C_2} \\ 0 & 0 & \frac{1}{L} & 0 \end{pmatrix} \begin{pmatrix} \phi(t) \\ v_1(t) \\ v_2(t) \\ i_L(t) \end{pmatrix}$$

By considering two such systems the overall error, the stabilization of which at the origin is the goal of synchronization, can be obtained by using the relation $\dot{e} = \dot{x} - \dot{\hat{x}}$ [obtained from the time derivative of the error $e(t) = x(t) - \hat{x}(t)$]. In the same way as for the case of fuzzy logic, parameter differences or nonlinear cross-terms resulting from this operation can then be grouped into a disturbance vector $m_e(t)$ to allow the corresponding error system to be written in the form[47]:

$$\dot{e}(t) = A(e(t))e(t) + Bu(t) + m_e(t)$$

where the polynomial matrix $A(e(t))$ takes the same form as the response system polynomial matrix $A(x(t))$ but error vector components instead. To then derive a controller that results in a Lyapunov stable system, one can then introduce a polynomial control law similar to that seen for the case of fuzzy logic subsystems.

$$u(t) = G(e(t))e(t) \tag{23}$$

Here, the constant coefficients matrices of the fuzzy controller are replaced a single matrix valued function of the state error $G(e(t))$. This allows the resulting controller to be a global control strategy as opposed to a local, as is the case for the fuzzy subsystems. By again defining the Lyapunov function to this unknown control matrix $G(e(t))$ and converting the definiteness constraints 3.8 into the corresponding SOS conditions, by finding a suitable Lyapunov function one is then able to construct the corresponding control matrix $G(e(t))$, the method of which is summarized next, the dynamic variables $e(t), m_e(t), z(t), \tilde{e}(t)$ are written as $e, m_e, z, \tilde{e}$.

Theorem 4.3. *Full state polynomial feedback gain controller [47]*
A full state error feedback control law, given in terms of dummy variable vector $e(t)$ as $G(e) = N(e)X^{-1}(\tilde{e})$ for the above system synchronizes the drive and response systems, subject to a H_∞ performance, with attenuation level σ:

$$\frac{\int_0^{t_f} z^T U(e) z dt - V(0)}{\int_0^{t_f} m_e^T m_e dt} \leq \sigma^2 \tag{24}$$

Where z is some affine transformation of the full error state vector $\hat{e}$, $V(0)$ represents the initial Lyapunov function value at the point when control is activated and with m_e

modelling a vector valued disturbance. The existence of the above control law is subject to the existence of polynomial matrices $U(e) = U(e)^T \in \mathbb{R}^{N\times N}$, $X(\tilde{e}) = X(\tilde{e})^T \in \mathbb{R}^{N\times N}$ and $N(e) \in \mathbb{R}^{m\times N}$ given the predefined polynomial functions $\epsilon_1(e), \epsilon_2(e), \epsilon_3(e)$ such that:

$$\begin{aligned}
&v^T(X(\tilde{e}) - \epsilon_1(e)I)v \textit{ is SOS} \\
&v^T(U - \epsilon_2(e)I)v \textit{ is SOS} \\
&-\begin{bmatrix} r \\ s \end{bmatrix}^T \left(\begin{bmatrix} \hat{R}(e) & T(e) \\ T(e)^T & -\sigma^2 I \end{bmatrix} + \epsilon_3(e)I \right) \begin{bmatrix} r \\ s \end{bmatrix} \textit{ is SOS}
\end{aligned}$$

Where r, s and $v \in \mathbb{R}^N$ denote arbitrary vectors. With the matrix $X(\tilde{e})$ taking the block diagonal form:

$$X = \begin{bmatrix} X_{11} & 0 \\ 0 & X_{22} \end{bmatrix}$$

With $X_{11} \in \mathbb{R}^{l\times l}$ and $X_{22} \in \mathbb{R}^{(N-l)\times(N-l)}$ and where the matrix $T(e) \in \mathbb{R}^{N\times n}$ (typically taken as an identify matrix), describing the relation between state error components $\hat{e}_i$ and dummy error variables e_j as:

$$T_{ij}(e) = \frac{\partial \hat{e}_i(e)}{\partial e_j} \tag{25}$$

It should be noted that the above formulation considers only partial forcing of the response system state, where $\tilde{e}$ is composed of state error variables $\hat{e}$ for which the corresponding row in the system 'forcing' matrix B contains no non-zero terms. In the case when B is a full rank matrix, the last matrix variable term $\hat{R}(e)$ is given by the form :

$$\begin{aligned}
&\hat{R}(e) = R(e) + U(e) \\
&R(e) = T(e)A(\hat{e})X + XA(\hat{e})^T T(e)^T + T(e)BN(e) + N(e)^T B^T T(e)^T
\end{aligned}$$

The reason for restating the theorem for the special case where B is full rank relates to the additional terms which are eliminated in this case, specifically for the general case where B may include rows of zeros, additional terms in the definition of $\hat{R}$ include require *a priori* knowledge of parameter mismatch. Thus using this special case allows the design of a more generic controller with guaranteed stability, independent of mismatches, at the cost of forcing all response system state variables.

The synthesis is performed using the SDPT3-4.0 solver and SOSTOOLS extension of the MATLAB LMI. The model parameter values 8 are chosen, the dummy variable derivative matrix is set to the identity: $T(e) = I \in \mathbb{R}^{4\times 4}$ resulting in the unforced error system being denoted as (where $e(t) = [e_\phi(t), e_{v_1}(t), e_{v_2}(t), e_{i_L}(t)]^T$)

$$\dot{e}(t) = \begin{pmatrix} 0 & -\tau & 0 & 0 \\ -\frac{3b}{C_1}e_\phi(t)e_{v_1}(t) & -\frac{1+aR}{C_1R} & \frac{1}{C_1R} & 0 \\ 0 & \frac{1}{C_2R}, & \frac{-1}{C_2R} & -\frac{1}{C_2} \\ 0 & 0 & \frac{1}{L} & 0 \end{pmatrix} \begin{pmatrix} e_\phi(t) \\ e_{v_1}(t) \\ e_{v_2}(t) \\ e_{i_L}(t) \end{pmatrix} + m_e(t)$$

Yielding the polynomial error system matrix

$$A(e(t)) = \begin{pmatrix} 0 & -\tau & 0 & 0 \\ -\frac{3b}{C_1}e_\phi(t)e_{v_1}(t) & -\frac{1+aR}{C_1R} & \frac{1}{C_1R} & 0 \\ 0 & \frac{1}{C_2R}, & \frac{-1}{C_2R} & -\frac{1}{C_2} \\ 0 & 0 & \frac{1}{L} & 0 \end{pmatrix}$$

Finally, the polynomial value functions are set to constant values: $\epsilon_1(e) = 0.001$,$\epsilon_2(e) = 0.002$,$\epsilon_3(e) = 0.003$ with the H_∞ attenuation set to $\sigma = 0.1$. For this configuration, the maximum order of polynomial matrix $N(e)$ is set to 2, this yielding a 3^{rd} order control law, the minimum order controller required to bound the cubic system nonlinearity in 2. In order to reduce computational complexity, the order of the H_∞ bound matrix $U(e)$ is set to 0. For the implementation code see Appendix **??**.

4.3 Sliding mode controller

In the above techniques using fuzzy logic and polynomial control techniques, we have utilized full state measurements as well as full state forcing of the response system. As discussed previously in practice this may not be feasible due to inaccessibility of the variables. In the particular case of the memristor based chaotic circuit this may be particularly challenging for the flux variable ϕ. Thus in the ideal scenario, we wish to apply the controlling signal to only a single response state variable. In this subsection we consider an technique known as 'sliding mode control' where the response dynamical system is extended with additional dynamic subsystem 's', defining a 'sliding surface' $s = 0$ for which the error dynamical system, representing the synchronization error between master and slave systems, is asymptotically stable. The first of two steps in the synthesis of this type of controller consists of defining a suitable form for the 's' subsystem and using this form to derive an idealizes control law, applicable when the system is on the sliding surface to demonstrate that asymptotic synchronization is achieved. The second step involves constructing the discontinuous control law, designed to ensure that origin of the sliding mode subsystem $s = 0$ is asymptotically stable and thus reachable using the control action, termed 'reaching condition'. The discontinuity is introduced in order to compensate for the unknown disturbances and unmodelled dynamics of systems, where unlike the previous cases, the bound on the disturbance is used directly in the control law. This bound overcompensates the destabilizing disturbance, with the discontinuous switching changing the direction in which this is applied and results in the system remaining in a region around the $s = 0$ point.The following method based on the technique described in [49] uses a proportional integral (PI) sliding surface, where the model of error dynamics is taken to have the form

$$\dot{e}(t) = Ae(t) + Bu(t) + Bf(t) + B\Delta f(t) \tag{26}$$

where, A represents the linear system matrix 15, $e(t)$ represents the state error vector and $u(t)$ the control signal, with corresponding forcing matrix B. As the model is nonlinear and subject to disturbance, the penultimate term f is the vector of nonlinear components and Δf represents the disturbance signal. The above can be directly constructing by considering error system 11, for which a sliding surface is selected as [49]

$$s = Ce - \int_0^t (CA + CBK)e(\xi)d\xi \tag{27}$$

Where C, K are constant matrices chosen to make $CB \neq 0$ and $(A + BK)$ is Hurwitz 3.12.By the defining property of 'sliding mode' operation $s = 0$, one can then set the above equation and its time derivative to zero and substitute 26 to derive the resulting 'equivalent control' law the form of which will later be used in the construction of the discontinuous control.

$$\begin{aligned}
\dot{s} &= C\dot{e} + (CA + CBK)e = C(Ae + Bu_{eq} + Bf + B\Delta f) + C(A + BK)e = 0 \\
u_{eq} &= Ke - f - \Delta f \\
\dot{e} &= (A + BK)e
\end{aligned}$$

The last equation, represents an asymptotically stable linear system resulting from the application of the u_{eq} to the original model. Note that in practical scenarios u_{eq} cannot be known *a priori* due to not knowing the exact form of the disturbances Δf, which here we assume to be bounded, in a specific way discussed later. The second step now consists of designing the discontinuous control law that, by quickly switching across the surface $s = 0$ will attempt to approximate the above u_{eq}, whilst also ensuring convergence towards this surface if it is initially far away. Here a commonly used Lyapunov candidate function is $V(s(t)) = \frac{1}{2}s(t)^T s(t)$, whose time derivative is used as the 'reaching condition'

$$\dot{V}(t) = s^T \dot{s} < 0$$

corresponding to the Lyapunov criterion for asymptotic stability of the sliding surface dynamics. For the switching control law of form [49]

$$u(t) = Ke - \gamma(CB)^{-1}[||CB||(||f|| + \beta_1||x(t)|| + \beta_2)]sign(s) \tag{28}$$

where $\gamma > 1$ and under assumption that the disturbance vector bound is of form

$$||\Delta f|| \leq \beta_1||x(t)|| + \beta_2 \tag{29}$$

with $x(t)$ denoting response system state, the above reaching condition can be shown to be satisfied for suitable bounds $\beta_1, \beta_2 > 0$.

At this point consider designing the matrix K to ensure stability of $A + BK$, where A represents the linear part of the modelled system. As noted above, the requirement of negative eigenvalues is equivalent to the matrix satisfying the Hurwitz property and consequently allows the application of 3.13, solvable by the MATLAB 'feasp' solver after applying a linearizing variable transformations 3.4.1 producing:

$$\begin{aligned}
&AX + XA^T + BN + N^TB^T + Q \prec 0 \\
&X \succ 0 \\
&Q \succ 0
\end{aligned}$$

here matrices X, N are used to construct feedback gains as $K = NX^{-1}$. For the memristor system: $A \in \mathbb{R}^{4\times4}$ 15, the forcing matrix $B \in \mathbb{R}^{4\times k}$ for control input dimension $u \in \mathbb{R}^k$ and $N \in R^{k\times4}$, $Q \in \mathbb{R}^{4\times4}$. At this point it is useful to remark that unlike the case of the fuzzy logic controller, no assumptions on the rank of the forcing matrix 'B' are required, thus allowing the design of controllers capable of stabilizing the error systems by forcing only a subset of the response system state space.

For the nominal drive and response memristor models the free (no control) idealized (no disturbance) error dynamics in the form 26

$$\dot{e}(t) = \begin{pmatrix} 0 & -\tau & 0 & 0 \\ 0 & -\frac{1+aR}{C_1R} & \frac{1}{C_1R} & 0 \\ 0 & \frac{1}{C_2R}, & \frac{-1}{C_2R} & -\frac{1}{C_2} \\ 0 & 0 & \frac{1}{L} & 0 \end{pmatrix} \begin{pmatrix} e_\phi(t) \\ e_{v_1}(t) \\ e_{v_2}(t) \\ e_{i_L}(t) \end{pmatrix} + B \begin{pmatrix} 0 \\ \frac{3b}{C_1}[\phi^2 v_1 - (\phi - e_\phi(t))^2(v_1 - e_{v_1}(t))] \\ 0 \\ 0 \end{pmatrix} \tag{30}$$

where $\phi = \phi(t), v_1 = v_1(t)$ denote response system state variables, with the second matrix term above being the nonlinear error 'f'.

4.4 Results and discussion

For the above cases of H_∞ fuzzy logic and polynomial feedback controllers, in order to measure performance a metric suggested in [54] is utilized:

$$J = \int_0^\infty \begin{bmatrix} e(t) \\ u(t) \end{bmatrix}^T \begin{bmatrix} J_1 & 0 \\ 0 & J_2 \end{bmatrix} \begin{bmatrix} e(t) \\ u(t) \end{bmatrix} dt \tag{31}$$

This metric can viewed as the sum of error $e(t)$ and control signal $u(t)$ energies, where the constant weight matrices $J_1, J_2 \succ 0$ may be adjusted to suitably weight each of the terms. In order to perform this computation during the simulation, the fixed step size Runge-Kutta (RK) [38] routine is augmented with a accumulator variable storing the parameter J as the sum of total scaled error and control signals, incremented in each step of the RK call. Due to the system being fully deterministic for nominal master and nominal slave models, a single evaluation is sufficient for the computation of the performance of the controller. Matched parameter disturbance can then be introduced into the response system to model similar effects arising in physical implementations where components value ranges are typically bounded by a pre-set tolerance level, often $\pm 10\%$ of nominal value, resulting in the response system dynamics differing from that of the drive system. In simulation such properties can be modelled by an additive disturbance Δp, the form of which is given by

$$p_{new} = p + \Delta p = p(1 + r\sigma)$$

where $r \in [-1, 1]$ is random number, with $\sigma \geq 0$ being a pre-specified tolerance level and $p \in \mathbb{R}$ is the nominal parameter value. This allows the evaluation of controller performance for the case when parameters are mismatched. Whilst more complex models such as generalized time varying disturbances and time delayed dynamics, can also be introduced, the simulations here are implemented for the above simpler case, with the above suggested as possible future extensions and in the latter case may require a corresponding modification to the controller analysis. Evaluating this performance metric for the fuzzy and polynomial controller for nominal drive and response models, with RK step size $h = 1 \times 10^{-8}s$ for 10^4 steps, given initial values of (ϕ, v_1, v_2, i_L):

$$\begin{aligned} &\text{Drive: } [\phi, v_1, v_2, i_L] = [0, -1, 2, 0] \\ &\text{Response: } [\phi, v_1, v_2, i_L] = [0, 1, 0.5, 0] \end{aligned} \tag{32}$$

we obtain the corresponding J values

$$\begin{aligned} &\text{Fuzzy}: \ J_{gevp} = 1.9206 \\ &\text{Polynomial}: \ J_{poly} = 1.3762 \times 10^{-6} \end{aligned}$$

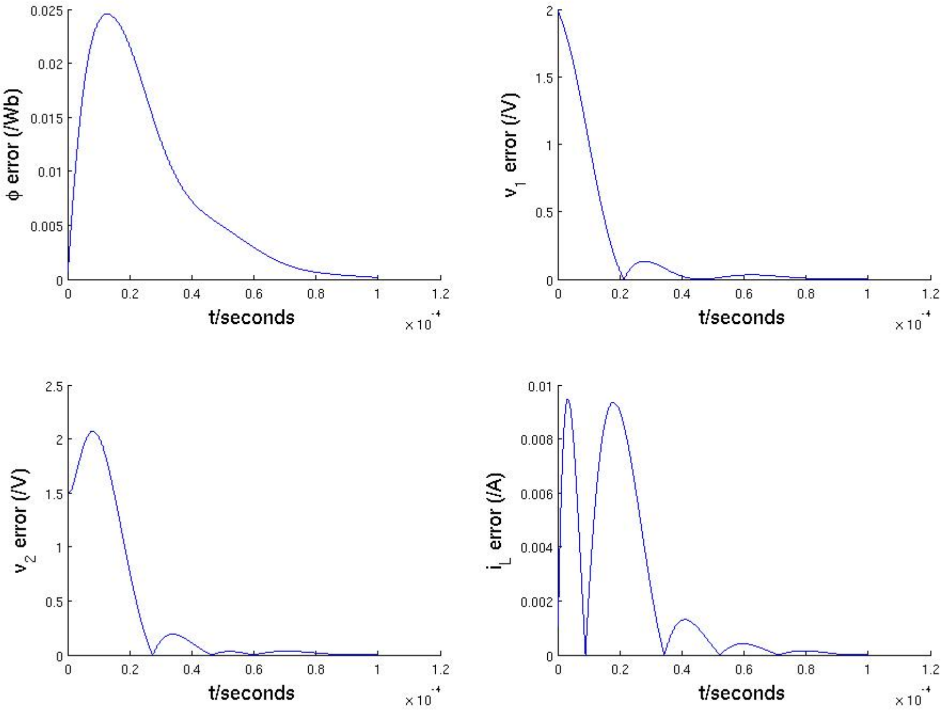

Figure 5: Sum of squares polynomial controller for attenuation level $\eta = 0.1$

for $J_1 = J_2 = I \in \mathbb{R}^{4\times 4}$. The vast order of magnitudes in difference between the control strategies arises as a result of discontinuous switching and minimization of the attenuation level η. The latter, determined by the MATLAB LMI 'gevp' to be $\eta = 3.26 \times 10^{-7}$ produces an almost instant error attenuation effect on the response system, at the cost of utilizing a unphysical control signal amplitude and producing the above poor performance. Replacing the 'gevp' minimization of η with the 'feasp' solver and manually setting attenuation level to $\eta = 0.1$, a improved performance controller is obtained with corresponding gain matrices:

$$G_1 = \begin{pmatrix} -0.0000 & -1.9178 & 0.0000 & 0.0000 \\ -1.9178 & -0.0246 & -0.0404 & 0.0000 \\ -0.0000 & -0.0404 & 0.0073 & 7.3529 \\ 0.0000 & 0.0000 & 7.3498 & -0.0000 \end{pmatrix} \times 10^6$$

$$G_2 = \begin{pmatrix} -0.0000 & 1.9204 & 0.0000 & -0.0000 \\ 1.9204 & -0.0246 & -0.0404 & 0.0000 \\ -0.0000 & -0.0404 & 0.0073 & 7.3529 \\ -0.0000 & 0.0000 & 7.3498 & -0.0000 \end{pmatrix} \times 10^6$$

The new performance metric $J_{feasp} = 1.2097$ for which the time evolution of the error is plotting in the Fig. 6, with the control applied for $t \geq 0$. The predicted chattering effect is now observed and the saturation function proposed in [6] is then utilized to achieve an factor of 10 performance improvement: $J_{sat} = 0.1215$, with the value of $T = 5 \times 10^5$ determined experimentally. This yields a significant improvement to reducing the size of the control signal, the introduction of control strategy is good relative to the initial solution obtain by the 'gevp' solver through only minimizing attenuation, its performance is still significantly worse than that of the polynomial controller which does not suffer for the above effect as seen in Fig. 5.

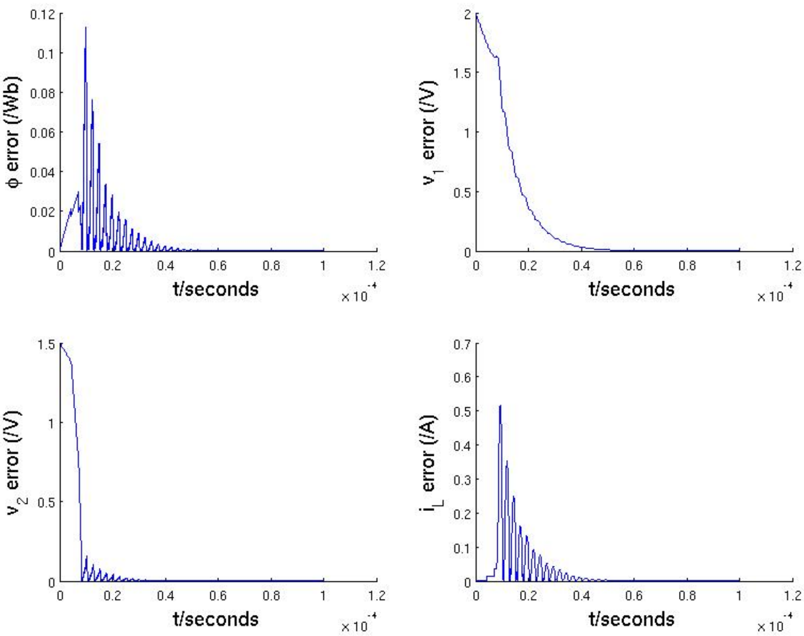

Figure 6: Switching fuzzy controller for attenuation level $\eta = 0.1$

Similarly to the fuzzy logic controller the sliding mode control strategy, based on discontinuous switching around $s = 0$ suffers from the same limitations. However, the motivation behind its implementation is to demonstrate synchronization using only single state variable forcing, selected for the memristor model to apply v_1 state variable, achieved by setting the forcing matrix B and corresponding sliding mode matrix C to

$$C = \begin{pmatrix} 0 & 1 & 0 & 0 \end{pmatrix}$$
$$B = \begin{pmatrix} 0 & 1 & 0 & 0 \end{pmatrix}^T$$

with corresponding gain K

$$K = \begin{pmatrix} 0.0026 & -0.0262 & -0.0746 & 2.9814 \end{pmatrix} \times 10^6$$

The feasibility of this method for asymptotic synchronization of the master and slave systems is demonstrated in Fig. 7. The primary limitation is tied to the affine bound on the system response state 29 and its use in 28. The difficulty arises from the conflict of requirements for smallness of control and suitable bounding of disturbance, where a large bound results in large amplitude chattering, degrading performance. For the purposes of simulation in Fig. 7 for synchronization between nominal drive and response models, the parameters of the controller are set to $\beta_1 = 2 \times 10^4$, $\beta_2 = 2 \times 10^3$, gradual stabilization of the error system. To accelerate convergence and improve performance, the modification of introducing a saturation function to replace $\text{sign}(s)$ for the case of nominal-nominal synchronization with step size $h = 1 \times 10^{-7}s$ for 2×10^4 steps, for $J_1 = J_2 = I \in \mathbb{R}^{4\times4}$ yields J values

$$\begin{aligned} \text{Switching law : sign(s)} &= 2.5004 \\ \text{Saturation law : sat(s,T)} &= 0.5427 \end{aligned}$$

where $T = 0.5$ represents region around the origin for where $\text{sat}(s,T) = s/T$ for $|s| < T$, with $\text{sat}(s,T) = \text{sign}(s)$ for $|s| \geq T$ [47].

From the above results it is apparent that the main issue arising in the case of both fuzzy logic control and sliding mode control is the discontinuous switching with fixed amplitude gains degrading performance in comparison to the smooth feedback of the polynomial controller. In practical applications numerous challenges are posed by the theoretically infinite frequency of switching such as the limit imposed in digital controllers with finite sampling rates, parasitic dynamics induced from high frequency effects and effects of system delayed dynamics arising from non-ideal system properties [55]. To this extent alternative techniques such as those based on higher order sliding mode controllers [55] have been proposed but will not be discussed further here. Instead focusing on the polynomial controller it is useful to note that for generalization to partial state forcing this technique requires *a priori* disturbance information. Additionally, the modification of the synthesis procedure for the case of output state control introduces additional complexity into the design procedure, which for the case of the memristor circuit is found infeasible by the SOSTOOLS optimization procedure.

5 Output state feedback control

5.1 Fuzzy Output State feedback controller

By referring back to the more general form of fuzzy models of dynamic systems 7 it should be apparent that the fuzzy controller derived for the full state feedback 4.1 is

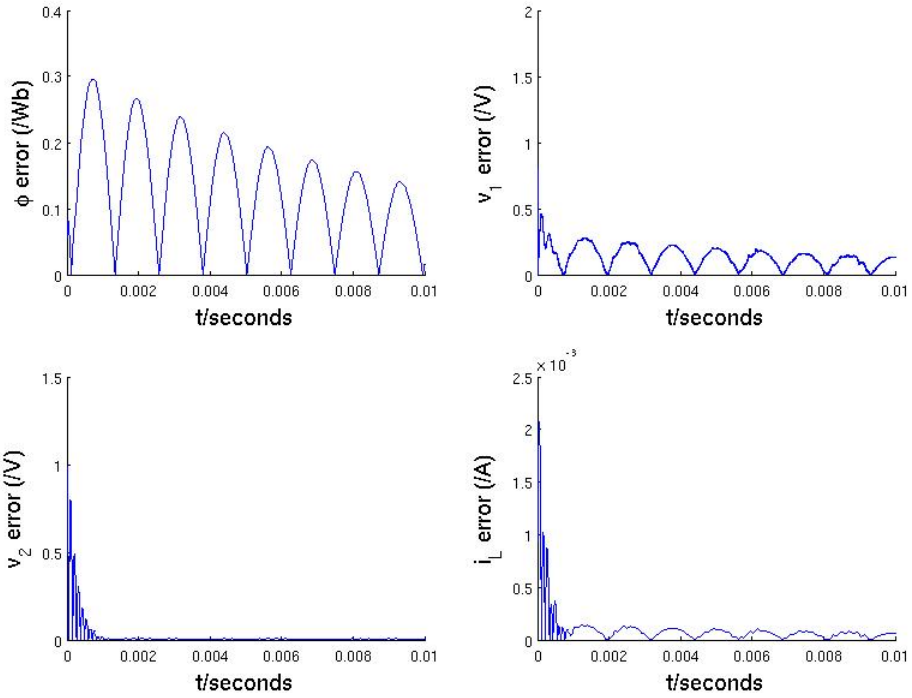

Figure 7: Sliding mode control based on single variable forcing of v_1 using full state measurement

a special case of the more general output state feedback fuzzy controller. In order to simplify the design the physical controller one considers using the output state vectors $y(t), \hat{y}(t)$ in the synthesis of the control law. To do so, consider the drive system to have form

$$\begin{aligned} \dot{\hat{x}}(t) &= \sum_{i=1}^{r} \hat{h}_i(\hat{x}_i(t))A_i\hat{x}(t) \\ \hat{y}(t) &= \sum_{i=1}^{r} \hat{h}_i(\hat{x}(t))C_i\hat{x}(t) \end{aligned} \tag{33}$$

with the corresponding response system as

$$\begin{aligned}\dot{x}(t) &= \sum_{i=1}^{r} h_i(x(t))(A_i + \Delta A_i)x(t) + Bu(t) \\ \hat{y}(t) &= \sum_{i=1}^{r} h_i(x(t))(C_i + \Delta C_i)x(t)\end{aligned} \tag{34}$$

where dimensions of both systems are still that of 7. The disturbance of the response system relative to the master system is represented both by the different membership functions for the master and slave $\hat{h}_i(\hat{x}_i(t)), h_i(x(t))$ as well as additive constant matrix terms $\Delta A_i \in \mathbb{R}^{n\times n}, \Delta C_i \in \mathbb{R}^{m\times n}$. From this point, to simplify the notation, we will denote controller and system membership functions $h_i(x(t)), \hat{h}_i(\hat{x}(t)), m_i(x(t))$ as $h_i, \hat{h}_i, m_i$. By now defining a control law, denoting $u(t)$ as u, using output state feedback as

$$u = B^{-1} \sum_i h_i(G_i(y - \hat{y}) \tag{35}$$

we can proceed similarly to the full state case, first expanding the above control law using the definitions of output states above and grouping mismatched terms into a disturbance vector $w_e(t)$ (for a full derivation see Appendix A) to obtain

$$u(t) = B^{-1} \sum_{i=1}^{r} \sum_{j=1}^{r} m_i h_j [G_i C_j e(t) + G_i w_e(t)]$$

By similarly grouping mismatched terms in the error dynamical system into the disturbance $m_e(t)$ and substituting the above control law form (with the assumption $m_i = h_i$)

$$\dot{e}(t) = \sum_i h_i[A_i e(t) + Bu(t) + m_e(t)] = \sum_i h_i h_j([A_i + G_i C_j]e(t) + G_i w_e(t) + m_e(t)) \tag{36}$$

At this point note the similarity between the above error system and 14, suggesting that we can proceed in a similar way to [6] by employing a Lyapunov candidate function $V(t) = e^T(t)Pe(t)$ and using this to define the corresponding time derivative, which we restructure into matrix form and impose the negative definiteness constraint on to derive the following form

$$\dot{V}(t) = \sum_{ij} h_i h_j \begin{pmatrix} e(t) \\ w_e(t) \\ m_e(t) \end{pmatrix}^T \begin{pmatrix} (P[A_i + G_i C_j] + [A_i^T + G_i^T C_j^T]P) & PG_i & P \\ G_i^T P & 0 & 0 \\ P & 0 & 0 \end{pmatrix} \begin{pmatrix} e(t) \\ w_e(t) \\ m_e(t) \end{pmatrix} \tag{37}$$

which yields the following theorem and correspond H_∞ bound.

Theorem 5.1. *For a output state drive and response systems of the form 33, 34, a controller given by 35 the synchronization error system 36 is stabilized subject to the H_∞ disturbance rejection criterion if there exist terms $P = P^T \succ 0, Q = Q^T \succ 0, M = M^T \succ 0, N_{ij} \in \mathbb{R}^{n \times n}, \eta > 0$ for a given $R_j = C_j^T (C_j C_j^T)^{-1}$ which for $i, j = 1, ..., r$ satisfy*

$$\begin{pmatrix} A_i^T P + PA_i + N_{ij}^T + N_{ij} + Q & N_{ij} R_j & P \\ R_j^T N_{ij}^T & -\eta M & 0 \\ P & 0 & -\eta P \end{pmatrix} \prec 0$$

The H_∞ criterion 3.17 then takes the form

$$\int_0^\infty e^T(t) Q e(t) dt \leq V(0) + \eta \int_0^\infty w_e^T(t) M w_e(t) dt + \eta \int_0^\infty m_e^T(t) P m_e(t) dt$$

Considering the special case when subsystem output matrices are equal $C_i = C$ we can then obtain a controller utilizing output state feedback to construct a H_∞ controller for the system. For the case of the memristor system, due to the difficulty of measurement of magnetic flux ϕ we first consider the output matrix to have the form

$$C = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

which eliminates the measurement of ϕ from the synthesis of the control signal. Simulating the synchronization of nominal drive and response systems for $1 \times 10^{-3} s$, using step size $h = 1 \times 10^{-7} s$ with initial values given by 32 and with control actived at $t = 5 \times 10^{-4} s$ the evolution of the error system as shown in Fig. 8 can be seen to ideally synchronize the three state variables used for the synthesis of the control signal while the ϕ is forced to remain at a fixed offset. A similar effect occurs when fixed error parameter mismatch is introduced, with the error variable e_ϕ being stabilized to oscillate around a constant offset value. The fundamental reason is attributable to the form of the control law, namely the proportional nature of the feedback, which unlike the sliding mode controller does not introduce an additional integral term introducing overshooting

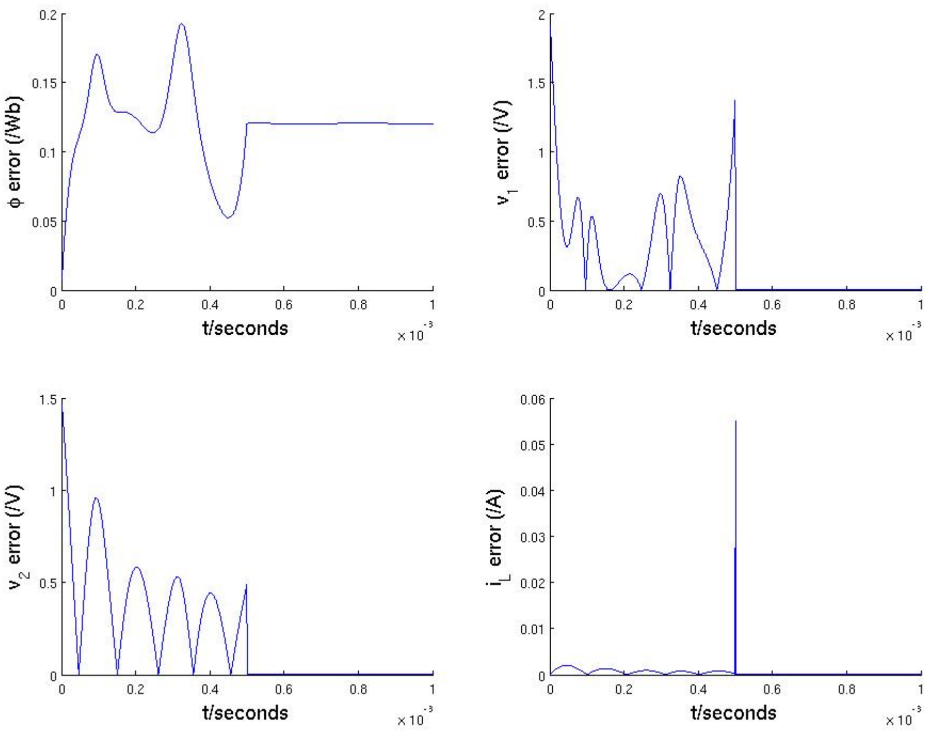

Figure 8: Fuzzy output state controller utilizing v_1, v_2, i_L as measured variables

behaviour to produce oscillatory behaviour to stabilize the entire system. Thus at this point it may be suggested that the above output fuzzy control law can be augmented with an additional dynamic subsystems which acts as an 'inertial' term for this purpose to introduce a delayed control component, a simple variant of which is considered in the following section. Instead, here we will stop to consider the possible ways in which the output states can be selected for measurement in order to guarantee that the error between unmeasured states is also stabilized. To do this we now look back to the PC synchronization scheme where synchronization of chaotic systems via a suitably selected direct coupling between system states, under the condition of the largest CLEs being negative was a necessary and sufficient condition to stabilize the system. Here, as one can see from Fig. 8 the controller is capable of achieving the first requirement, namely it is capable of identically stabilizing measured state variables under H_∞. Consider the subsystem (v_2, i_L) when the system output states are ϕ, v_1. In this case, instead of considering the CLEs of the system one can consider the error dynamics of this subsystem in the limit of identical output state synchronization. Consider the error subsystem $(e_{v_2}(t), e_{i_L}(t))$ by taking the lowest two rows of 15

$$\begin{pmatrix} \dot{e}_{v_2}(t) \\ \dot{e}_{i_L}(t) \end{pmatrix} = \begin{pmatrix} 0 & \frac{1}{C_2R}, & \frac{-1}{C_2R} & -\frac{1}{C_2} \\ 0 & 0 & \frac{1}{L} & 0 \end{pmatrix} \begin{pmatrix} e_\phi(t) \\ e_{v_1}(t) \\ e_{v_2}(t) \\ e_{i_L}(t) \end{pmatrix} + \begin{pmatrix} u_3(t) \\ u_4(t) \end{pmatrix} \tag{38}$$

Now consider the limiting case, when $e_\phi(t) \to 0$, $e_{v_1}(t) \to 0$ for the case when the system output matrix C is given as

$$C = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \tag{39}$$

Here now, the control signal $u(t)$ and it's corresponding components $u_i(t) \to 0$ for $i = 1, ...4$ giving

$$\begin{pmatrix} \dot{e}_{v_2}(t) \\ \dot{e}_{i_L}(t) \end{pmatrix} = \begin{pmatrix} \frac{-1}{C_2R} & -\frac{1}{C_2} \\ \frac{1}{L} & 0 \end{pmatrix} \begin{pmatrix} e_{v_2}(t) \\ e_{i_L}(t) \end{pmatrix}$$

with corresponding eigenvalues

$$\lambda_{1,2} = \frac{-L \pm \sqrt{L(L - 4C_2R^2)}}{2RLC_2}$$

yielding an asymptotically stable subsystem, the necessary and sufficient condition for synchronization [56], for the selected set of parameters 8 used to produce chaotic behaviour in the memristor system. Notice only measuring ϕ and v_1 we are also then able to construct the membership functions 16, or approximations to them, due to their definition of the fuzzy premise variable as $z = \phi v_1$, thus again simplifying the construction of the controller. Finally note that by extending the system dimensionality by one through introducing an new dynamic variable similar the sliding surface 's', coupling only to the flux ϕ and considering $e_{v_1}(t) \to 0$, where v_1 is the only output variable, we obtain

$$\begin{pmatrix} \dot{s}(t) \\ \dot{e}_\phi(t) \end{pmatrix} = \begin{pmatrix} a_1 & a_2 \\ a_3 & 0 \end{pmatrix} \begin{pmatrix} s(t) \\ e_\phi(t) \end{pmatrix}$$

Then by setting a_1, a_2, a_3 to make as asymptotically stable subsystem and thus allows the reduction required output variables to one, v_1. To then design a fuzzy controller one can then apply the same theorem above to this new extended system, which can be shown to stabilize the error and thus synchronize the drive and response. However, it is now more challenging to relate system performance to the H_∞ bound set during fuzzy controller design, which as noted for the above examples is required to be large for good performance. Additionally, it should be noted that this analysis applies only in the limit and thus convergence in a practical scenario is not guaranteed. An alternative to the above is the introduction of an intermediate state 'observer' system used to estimate unmeasurable states [57] from a drive system variable. As mentioned previously, such systems have been demonstrated effective not only for synchronization but for cryptanalytic attacks on secure communications [26] utilizing chaotic masking, here however issues relating to quality of the transmitted signal affect synchronization.

5.2 Fuzzy Delayed Output State feedback controller

The final controller introduced here and implemented within the software is based on the earlier mentioned idea of introducing a 'inertial' error term, for the case of the sliding mode controller given by the integral in 27, where if suitably selected it will ensure the control forced all state variables to zero, as opposed to deactivating when output state synchronization is achieved. Here this 'inertial' term will consist of a time delayed output error, modifying the proportional feedback control law 35 to have the form suggested in [7] where $C_i = C, \forall i$ is assumed.

$$u = B^{-1} \sum_i h_i G_i[(y(t) - \hat{y}(t)) - \xi(y(t - \tau_d) - \hat{y}(t - \tau_d))] = B^{-1} \sum_i h_i G_i C[e(t) - \xi e(t - \tau_d)] \tag{40}$$

where $0 < \xi < 1$ and $0 < \tau_d$ represents the time delay, in physical circuits achieved by analog delay lines [58]. To deal with the delay, the method proposed in [7] extends the quadratic form of the Lyapunov function in the previous subsection to

$$\begin{aligned} V(e(t)) &= V_1(e(t)) + V_2(e(t)) + V_3(e(t)) \\ V_1(e(t)) &= e^T(t) P e(t) \\ V_2(e(t)) &= \int_{-\tau}^{0} \int_{t+\beta}^{t} e^T(\alpha) Q e(\alpha) d\alpha d\beta \\ V_3(e(t)) &= \int_{t-\tau}^{t} e^T(\sigma) R e(\sigma) d\sigma \end{aligned} \tag{41}$$

where $P = P^T \succ 0, R = R^T \succ 0, Q = Q^T \succ 0$. Due to the more involved nature of this proof, its discussion here is omitted, it suffices to say that by taking the time derivative of each of the above terms and finding a suitable bound the form of $\dot{V}(e(t))$ is then similar to 37

$$\dot{V}(e(t)) \leq \sum_i \sum_j h_i h_j \begin{bmatrix} e(t) \\ e(t-\tau_d) \\ \int_{t-\tau_d}^{t} e(\sigma)d\sigma \end{bmatrix} \begin{bmatrix} (1,1) & -\xi P G_j C & 0 \\ -\xi C^T G_j^T P & -R & 0 \\ 0 & 0 & -\frac{1}{\tau_d} Q \end{bmatrix} \begin{bmatrix} e(t) \\ e(t-\tau_d) \\ \int_{t-\tau_d}^{t} e(\sigma)d\sigma \end{bmatrix}$$
$$- e(t)^T S e(t) + \eta^2 d^T(t) d(t)$$

where $(1,1) = A_i^T P + PA_i + PG_j C + C^T G_j P + \frac{1}{\eta^2} P W_i W_i^T P + \tau_d Q + R + S$, $W_i \in \mathbb{R}^n \times k$ represent disturbance matrices applied $d(t) \in \mathbb{R}^k$, response disturbance. Using a variant of Schur complement 4.2 this yields

Theorem 5.2. *[7] For given $0 < \eta$, $0 < \xi < 1$, $S = S^T \succ 0$ if there exist matrices $P = P^T \succ 0, R = R^T \succ 0, Q = Q^T \succ 0$ for which*

$$\begin{bmatrix} [1,1] & -\xi P G_j C & 0 & PW_i & I & I \\ -\xi C^T G_j^T P & -R & 0 & 0 & 0 & 0 \\ W_i^T P & 0 & 0 & -\eta^2 I & 0 & 0 \\ I & 0 & 0 & 0 & -\frac{1}{\tau} Q^{-1} & 0 \\ I & 0 & 0 & 0 & 0 & -S^{-1} \end{bmatrix} < 0$$

where $[1,1] = A_i^T P + PA_i + PG_j C + C^T G_j^T P + R$.

Note that in the above, the form is given as bilinear constraints, thus some of the terms require grouping before applying the convex optimization. It should be noted here that similarly to the previous section the assumption about exact knowledge of the fuzzy membership functions is assumed. Thus, when comparing the two controllers we will set C to the form 39.

5.3 Results and discussion

Simulating the two fuzzy controllers for the nominal drive and response systems with initial values

$$\text{Drive: } [\phi, v_1, v_2, i_L] = [0, -1, 2, 0] \tag{42}$$
$$\text{Response: } [\phi, v_1, v_2, i_L] = [0, 1, 0.5, 0] \tag{43}$$

for step size $h = 1 \times 10^{-7} s$, for 10^4 steps, where the H_∞ bounds for both are given by $\eta = 1 \times 10^5$, determined experimentally to give good performance for the proportional controller 5.1. Using the output matrix 39 and $J_1 = J_2 = I \in \mathbb{R}^{4\times 4}$ we obtain performance metrics:

$$J_{proportional} = 0.3005$$
$$J_{delayed} = 0.2411$$

Note that the delayed fuzzy controller 5.2 uses a $\tau_d = 1 \times 10^{-5}s$, with a delay scaling term $\xi = 0.5$. Consequently, for the purposes of simulation the control is activated at $t = 1 \times 10^{-5}s$ in both cases. For the case of the proportional output fuzzy controller the feedback gains G_1, G_2 are

$$G1 = \begin{pmatrix} -0.0492 & -0.0418 \\ -2.0953 & -2.4795 \\ -0.0021 & -0.0025 \\ 0.0000 & 0.0000 \end{pmatrix} \times 10^6$$
$$G2 = \begin{pmatrix} -0.0495 & 0.0451 \\ 2.2244 & -2.5335 \\ 0.0024 & -0.0027 \\ -0.0000 & 0.0000 \end{pmatrix} \times 10^6$$

while the delayed error feedback controller gains are

$$G1 = \begin{pmatrix} -0.0007 & 0.0002 \\ 0.5611 & -1.0861 \\ 0.6267 & -1.2354 \\ -0.0000 & 0.0000 \end{pmatrix} \times 10^7$$
$$G2 = \begin{pmatrix} -0.0007 & 0.0002 \\ 0.5611 & -1.0861 \\ 0.6267 & -1.2354 \\ -0.0000 & 0.0000 \end{pmatrix} \times 10^7$$

As demonstrated in Fig. 9 and Fig. 10 both stabilize the error. It should be noted that LMI system of the delayed controller is found infeasible by MATLAB LMI whenever the state variable ϕ is not a measured output. Additionally the design of the proportional controller and the selection of ϕ, v_1 as output variables to be driven to zero relates to the PC coupling described previously 2.3.1, thus it is suggested that in physical implementations of synchronization a preliminary test can be conducted by directly coupling ϕ, v_1 of the response to the drive, which is also verified to synchronize the resulting systems in simulation and is implemented in the code under 'Impulsive', a generalization of PC based on [59].

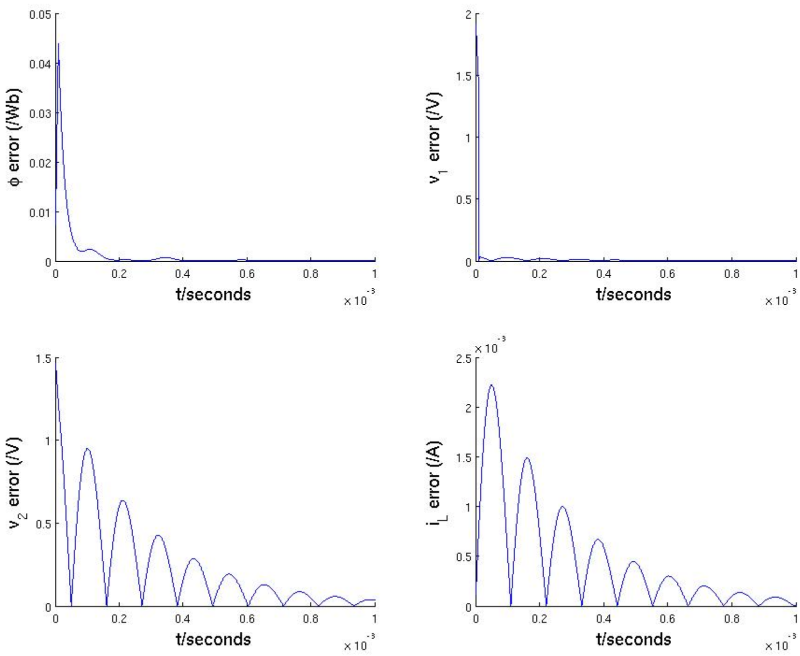

Figure 9: Fuzzy output state controller utilizing ϕ, v_1 as measured with asymptotically stable error subsystem e_{v_2}, e_{i_L} in limit of output state synchronization

5.4 Software implementation considerations

The language used for the implementation, MATLAB, is selected due to the availability of the LMI solvers as well as the SOSTOOLS package required for controller design. The simple fixed step size fourth order Runge-Kutta method [38] is implemented instead of using the more advanced built in solvers due to the addition complexity of the underlying code. In particular, the use of adaptive step size adjustment in the available solvers combined with the property of chaos leads to unpredictable simulation times as initially nearby trajectories move into different parts of the phase space. Thus the use of fixed step size allow for predictable execution times as well as increase transparency of code. It should also be noted that in simulation, as opposed to the differential equation form, the system is converted into a discrete time approximation of the continuous time model. Thus, when performing a simulation it is suggested that the user use step sizes of $1 \times 10^{-7} s$ or below, and use the reduction of step size as a first point of call when simulating controllers. Specifically it is found that the controllers based of discontinuous switching are more unstable with increasing steps size. In this regard, 'assert' statements are included in the code to attempt to catch some of the issues that may arise. Extensions of the simulation code specifically may attempt to compute and maintain certain physical invariants, rescale the system in a particular way or implement saturation features seen in

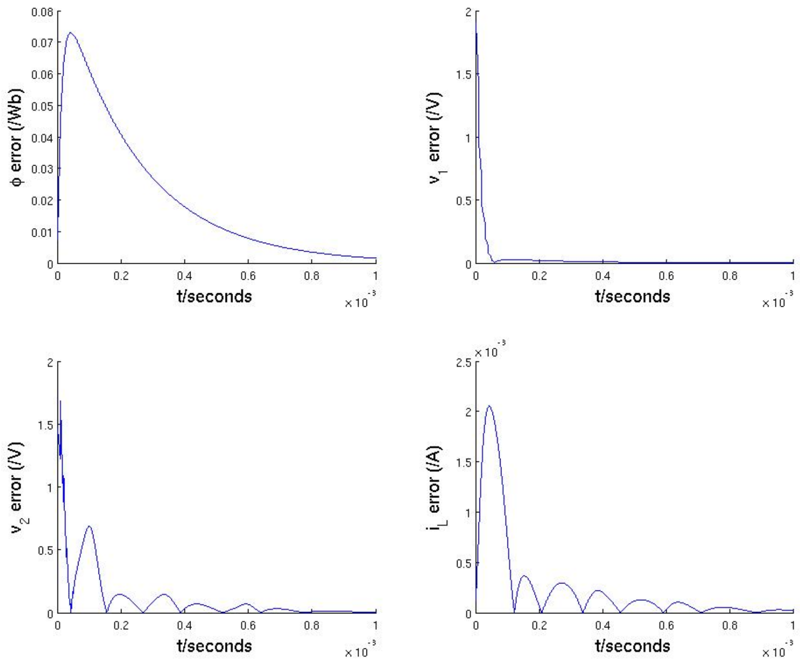

Figure 10: Fuzzy delayed output state feedback controller, delay $\tau_d = 10\mu s$, delayed error attenuation $\xi = 0.5$, output state variables ϕ, v_1

physical circuits, however the benefits of this need further scrutiny. For an introduction to the GUI implemented and example commands see Appendix B.

6 Conclusions and future work

In the above we have considered the identical synchronization of chaotic systems for two memristor based Chua circuits. Control laws based on full state feedback utilizing fuzzy logic and polyonomial feedback were demonstrated to be capable of synchronizing systems, where it was found that chattering phenomena and the fixed feedback gains of the fuzzy controller present performance challenges when compared to the smooth polynomial controller. A similar effect occurs in the sliding mode controller presented as a technique enabling synchronization by applying control to only a single variable, where the magnitude of the disturbance bound must be balanced with the need for using small control action. It was demonstrated that by suitable selection of measured output variables, a extension of the original fuzzy controller to use output state error can be combined with asymptotic analysis of unmeasured subsystem states to achieve identical synchronization. However, it is recognized that in order to do this for the more general case, the proportional subsystem feedback laws must be extended to include additional non-vanishing terms that judiciously overshoot the control and attenuate unmeasured state errors.

Numerous possible theoretical extensions dealing with these and other issues here have been investigated in the literature, such as the extensions to delayed dynamical systems [60] applying the control signal intermittently, combining sliding mode control with fuzzy logic [61] or even adaptively tuning sliding mode gain [62] to eliminate the issue of using the fixed bound on tolerable disturbance. Additionally, whilst here we have worked in continuous time, the extension of techniques to discrete time and digital controllers [63] opens the possibility of easier deployment and reconfiguration. From a practical perspective, the next step of this work would consist of physically implementing the system to demonstrate the feasibility of synchronization, perhaps using digital controller for ease of reprogramming.

References

[1] L. M. Pecora and T. L. Carroll, "Synchronization in chaotic systems," *Physical review letters*, vol. 64, no. 8, p. 821, 1990.

[2] E. Ott, C. Grebogi, and J. A. Yorke, "Controlling chaos," *Physical review letters*, vol. 64, no. 11, p. 1196, 1990.

[3] L. Chua, "Memristor-the missing circuit element," *IEEE Transactions on circuit theory*, vol. 18, no. 5, pp. 507–519, 1971.

[4] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, "The missing memristor found," *nature*, vol. 453, no. 7191, pp. 80–83, 2008.

[5] A. G. Radwan and M. E. Fouda, *On the mathematical modeling of memristor, memcapacitor, and meminductor*, vol. 26. Springer, 2015.

[6] H. Lam and F. F. Leung, "Synchronization of uncertain chaotic systems based on the fuzzy-model-based approach," *International Journal of Bifurcation and Chaos*, vol. 16, no. 05, pp. 1435–1444, 2006.

[7] C. K. Ahn, "T–s fuzzy synchronization for chaotic systems via delayed output feedback control," *Nonlinear Dynamics*, vol. 59, no. 4, pp. 535–543, 2010.

[8] R. Roy, T. Murphy Jr, T. Maier, Z. Gills, and E. Hunt, "Dynamical control of a chaotic laser: Experimental stabilization of a globally coupled system," *Physical Review Letters*, vol. 68, no. 9, p. 1259, 1992.

[9] T. Klinger, C. Schröder, D. Block, F. Greiner, A. Piel, G. Bonhomme, and V. Naulin, "Chaos control and taming of turbulence in plasma devices," *Physics of Plasmas (1994-present)*, vol. 8, no. 5, pp. 1961–1968, 2001.

[10] V. Petrov, V. Gaspar, J. Masere, and K. Showalter, "Controlling chaos in the belousov zhabotinsky reaction," *Nature*, vol. 361, no. 6409, pp. 240–243, 1993.

[11] W. Ditto, M. Spano, V. In, J. Neff, B. Meadows, J. Langberg, A. Bolmann, and K. McTeague, "Control of human atrial fibrillation," *International Journal of Bifurcation and Chaos*, vol. 10, no. 03, pp. 593–601, 2000.

[12] L. Glass, "Synchronization and rhythmic processes in physiology," *Nature*, vol. 410, no. 6825, pp. 277–284, 2001.

[13] P. Jiruska, M. de Curtis, J. G. Jefferys, C. A. Schevon, S. J. Schiff, and K. Schindler, "Synchronization and desynchronization in epilepsy: controversies and hypotheses," *The Journal of physiology*, vol. 591, no. 4, pp. 787–797, 2013.

[14] M. Ashhab, M. Salapaka, M. Dahleh, and I. Mezic, "Control of chaos in atomic force microscopes," in *American Control Conference, 1997. Proceedings of the 1997*, vol. 1, pp. 196–202, IEEE, 1997.

[15] E. Nikulchev and E. Pluzhnik, "Study of chaos in the traffic of computer networks," *Editorial Preface*, vol. 5, no. 9, 2014.

[16] B. R. Andrievskii and A. L. Fradkov, "Control of chaos: Methods and applications. ii. applications," *Automation and Remote Control*, vol. 65, no. 4, pp. 505–533, 2004.

[17] B. R. Andrievskii and A. L. Fradkov, "Control of chaos: Methods and applications. i. methods," *Automation and Remote Control*, vol. 64, no. 5, pp. 673–713, 2003.

[18] G. Witvoet, "Control of chaotic dynamical systems using ogy," 2005.

[19] T. S. Parker and L. Chua, *Practical numerical algorithms for chaotic systems*. Springer Science & Business Media, 2012.

[20] P. So, E. Ott, T. Sauer, B. J. Gluckman, C. Grebogi, and S. J. Schiff, "Extracting unstable periodic orbits from chaotic time series data," *Physical Review E*, vol. 55, no. 5, p. 5398, 1997.

[21] A. Wolf, J. B. Swift, H. L. Swinney, and J. A. Vastano, "Determining lyapunov exponents from a time series," *Physica D: Nonlinear Phenomena*, vol. 16, no. 3, pp. 285–317, 1985.

[22] K. Pyragas, "Continuous control of chaos by self-controlling feedback," *Physics letters A*, vol. 170, no. 6, pp. 421–428, 1992.

[23] S. Boccaletti, J. Kurths, G. Osipov, D. Valladares, and C. Zhou, "The synchronization of chaotic systems," *Physics reports*, vol. 366, no. 1, pp. 1–101, 2002.

[24] A. Abel and W. Schwarz, "Chaos communications-principles, schemes, and system analysis," *Proceedings of the IEEE*, vol. 90, no. 5, pp. 691–710, 2002.

[25] K. M. Cuomo, A. V. Oppenheim, and S. H. Isabelle, "Spread spectrum modulation and signal masking using synchronized chaotic systems," tech. rep., DTIC Document, 1992.

[26] Y. Liu and W. K. Tang, "Cryptanalysis of chaotic masking secure communication systems using an adaptive observer," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 55, no. 11, pp. 1183–1187, 2008.

[27] L. Kocarev and U. Parlitz, "General approach for chaotic synchronization with applications to communication," *Physical review letters*, vol. 74, no. 25, p. 5028, 1995.

[28] E. R. Weeks and J. M. Burgess, "Evolving artificial neural networks to control chaotic systems," *physical review E*, vol. 56, no. 2, p. 1531, 1997.

[29] K. Hirasawa, X. Wang, J. Murata, J. Hu, and C. Jin, "Universal learning network and its application to chaos control," *Neural Networks*, vol. 13, no. 2, pp. 239–253, 2000.

[30] P. Taxel, "Wikipedia,"

[31] L. O. Chua and S. M. Kang, "Memristive devices and systems," *Proceedings of the IEEE*, vol. 64, no. 2, pp. 209–223, 1976.

[32] B. Muthuswamy, "Implementing memristor based chaotic circuits," *International Journal of Bifurcation and Chaos*, vol. 20, no. 05, pp. 1335–1350, 2010.

[33] A. S. . H. . L. condition and L. continuity, "Lipschitz condition and lipschitz continuity," 2006.

[34] M. Feki, "An adaptive feedback control of linearizable chaotic systems," *Chaos, Solitons & Fractals*, vol. 15, no. 5, pp. 883–890, 2003.

[35] S. Boccaletti, C. Grebogi, Y.-C. Lai, H. Mancini, and D. Maza, "The control of chaos: theory and applications," *Physics reports*, vol. 329, no. 3, pp. 103–197, 2000.

[36] R. M. Murray, Z. Li, S. S. Sastry, and S. S. Sastry, *A mathematical introduction to robotic manipulation.* CRC press, 1994.

[37] D. Arzelier, "Lmis in systems control state-space method, performance analysis and synthesis,," 2008.

[38] K. F. Riley, M. P. Hobson, and S. J. Bence, *Mathematical methods for physics and engineering: a comprehensive guide.* Cambridge University Press, 2006.

[39] A. A. Stoorvogel, *The H infinity Control Problem: A State Space Approach.* Citeseer, 1992.

[40] J. A. Ball, J. W. Helton, and M. L. Walker, "H control for nonlinear systems with output feedback," *IEEE Transactions on Automatic Control*, vol. 38, no. 4, pp. 546–559, 1993.

[41] S. Boyd and L. Vandenberghe, *Convex optimization.* Cambridge university press, 2004.

[42] J. A. Nelder and R. Mead, "A simplex method for function minimization," *The computer journal*, vol. 7, no. 4, pp. 308–313, 1965.

[43] D. Tikk, L. T. Kóczy, and T. D. Gedeon, "A survey on universal approximation and its limits in soft computing techniques," *International Journal of Approximate Reasoning*, vol. 33, no. 2, pp. 185–202, 2003.

[44] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE transactions on systems, man, and cybernetics*, no. 1, pp. 116–132, 1985.

[45] H. O. Wang, K. Tanaka, and M. F. Griffin, "An approach to fuzzy control of nonlinear systems: stability and design issues," *IEEE transactions on fuzzy systems*, vol. 4, no. 1, pp. 14–23, 1996.

[46] K. Mehran, "Takagi-sugeno fuzzy modeling for process control," *Industrial Automation, Robotics and Artificial Intelligence (EEE8005); Newcastle University: Newcastle upon Tyne, UK*, 2008.

[47] H. Lam, "Chaotic synchronisation using output/full state-feedback polynomial controller," *IET Control Theory & Applications*, vol. 4, no. 11, pp. 2285–2292, 2010.

[48] S. Prajna, A. Papachristodoulou, and P. A. Parrilo, "Introducing sostools: A general purpose sum of squares programming solver," in *Decision and Control, 2002, Proceedings of the 41st IEEE Conference on*, vol. 1, pp. 741–746, IEEE, 2002.

[49] J.-J. Yan, Y.-S. Yang, T.-Y. Chiang, and C.-Y. Chen, "Robust synchronization of unified chaotic systems via sliding mode control," *Chaos, Solitons & Fractals*, vol. 34, no. 3, pp. 947–954, 2007.

[50] H.-K. Lam and L. D. Seneviratne, "Synchronization of chaotic systems using neural-network-based controller," *International Journal of Bifurcation and Chaos*, vol. 17, no. 06, pp. 2117–2125, 2007.

[51] P. Gahinet, A. Nemirovskii, A. J. Laub, and M. Chilali, "The lmi control toolbox," in *IEEE conference on decision and control*, vol. 2, pp. 2038–2038, Citeseer, 1994.

[52] C. Scherer and S. Weiland, "Linear matrix inequalities in control," *Lecture Notes, Dutch Institute for Systems and Control, Delft, The Netherlands*, vol. 3, 2000.

[53] S. Banerjee and Ş. Ş. Erçetin, *Chaos, complexity and leadership 2012.* Springer, 2014.

[54] H.-K. Lam and M. Narimani, “Stability analysis and performance design for fuzzy-model-based control system under imperfect premise matching,” *IEEE Transactions on Fuzzy Systems*, vol. 17, no. 4, pp. 949–961, 2009.

[55] K. D. Young, V. I. Utkin, and U. Ozguner, “A control engineer’s guide to sliding mode control,” *IEEE transactions on control systems technology*, vol. 7, no. 3, pp. 328–342, 1999.

[56] R. He and P. Vaidya, “Analysis and synthesis of synchronous periodic and chaotic systems,” *Physical Review A*, vol. 46, no. 12, p. 7387, 1992.

[57] K.-Y. Lian, P. Liu, and W.-C. Lin, “Robust chaotic fuzzy output feedback tracking control,” in *Fuzzy Systems, 2001. The 10th IEEE International Conference on*, vol. 3, pp. 1577–1580, IEEE, 2001.

[58] R. industries Inc., “Delay lines,” 1998.

[59] M. Itoh, T. Yang, and L. O. Chua, “Conditions for impulsive synchronization of chaotic and hyperchaotic systems,” *International Journal of Bifurcation and Chaos*, vol. 11, no. 02, pp. 551–560, 2001.

[60] T. Huang, C. Li, W. Yu, and G. Chen, “Synchronization of delayed chaotic systems with parameter mismatches by using intermittent linear state feedback,” *Nonlinearity*, vol. 22, no. 3, p. 569, 2009.

[61] H.-T. Yau, “Chaos synchronization of two uncertain chaotic nonlinear gyros using fuzzy sliding mode control,” *Mechanical Systems and Signal Processing*, vol. 22, no. 2, pp. 408–418, 2008.

[62] M. Roopaei, M. Z. Jahromi, and S. Jafari, “Adaptive gain fuzzy sliding mode control for the synchronization of nonlinear chaotic gyros,” *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 19, no. 1, p. 013125, 2009.

[63] H. Lam and F. H. Leung, “Sampled-data fuzzy controller for time-delay nonlinear systems: fuzzy-model-based lmi approach,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 37, no. 3, pp. 617–629, 2007.

A Output State Fuzzy Controller Derivation

In the below, for purposes of clarity denote $x(t), y(t), e(t), u(t)$ and similar variables as x, y, e, u. Fuzzy membership functions, such as $h_i(x(t), \hat{x}(t))$ are similarly denoted h_i. Dot notation $\dot{x}$ then represents the time derivative. Capital symbols represent matrix variables. We start by considering the TS fuzzy logic models of the master and slave systems, where the amplitude of the drive signal generated by the master is assumed to be bounded from the property of chaotic systems 3.1.

Master:

$\dot{\hat{x}} = \sum_i \hat{h}_i(\hat{x}_i) A_i \hat{x}$
$\hat{y} = \sum_i \hat{h}_i(\hat{x}) C_i \hat{x}$

Slave:

$\dot{x} = \sum_i h_i(x)(A_i + \Delta A_i)x + Bu$
$\hat{y} = \sum_i h_i(x)(C_i + \Delta C_i)x$

Let:

$e = x - \hat{x}$ then $\dot{e} = \dot{x} - \dot{\hat{x}} =$
$= \sum_i [h_i(x)A_i x - \hat{h}_i(\hat{x})A_i\hat{x} + h_i(x)\Delta A_i x] + Bu =$
$= \sum_i [h_i(x)A_i(x - \hat{x}) + h_i(x)A_i\hat{x} - \hat{h}_i(\hat{x})A_i\hat{x} + h_i(x)\Delta A_i x] + Bu$

Let $m_e(x, \hat{x}) = \sum_i (h_i(x) - \hat{h}_i)A_i\hat{x} + h_i(x)\Delta A_i x$ then the above becomes:

$\sum_i [h_i(x)A_i e] + m_e(x, \hat{x}) + Bu$

Now let the control signal be: $u = B^{-1}\sum_i m_i G_i(y - \hat{y})$. Here m_i represents the controller fuzzy function that aims to approximate that fuzzy function h_i, which in the case of an uncertain system is unknown. This can be done in the process of system identification. Now use the above output state definitions to expand this expression to get:

$u = B^{-1}\sum_i m_i G_i \sum_j [h_j(x)(C_j + \Delta C_j)x - \hat{h}_j(\hat{x})C_j\hat{x})] =$
$= B^{-1}\sum_i m_i G_i \sum_j [h_j(x)C_j x - \hat{h}_j(\hat{x})C_j\hat{x}) + h_j(x)\Delta C_j x] =$
$= B^{-1}\sum_i m_i G_i \sum_j [h_j(x)C_j e + h_j(x)C_j\hat{x} - \hat{h}_j(\hat{x})C_j\hat{x}) + h_j(x)\Delta C_j x]$

Performing the same manipulation as above by defining: $w_e(x, \hat{x}) = \sum_j [h_j(x)C_j\hat{x} -$

$\hat{h}_j(\hat{x})C_j\hat{x}) + h_j(x)\Delta C_j x]$ obtain:

$$u = B^{-1}\sum_i m_i G_i[(\sum_j h_j(x)C_j e) + w_e(x,\hat{x})]$$

Pushing the summation out into a single sum obtain:

$u = B^{-1}\sum_i\sum_j m_i(x)h_j(x)G_i[C_j e + w_e(x,\hat{x})] =$
$= B^{-1}\sum_{ij} m_i(x)h_j(x)[G_i C_j e + G_i w_e(x,\hat{x})]$

Substituting back into the error system $\dot{e}$ obtain:

$\dot{e} = \sum_i h_i(x)A_i e + B(B^{-1}\sum_{ij} m_i(x)h_j(x)[G_i C_j e + G_i w_e(x,\hat{x})]) + m_e(x,\hat{x}) =$

$= \sum_i(h_i(x)A_i e + m_e(x,\hat{x})) + (\sum_{ij} m_i(x)h_j(x)[G_i C_j e + G_i w_e(x,\hat{x})]) =$
$= \sum_{ij} h_i h_j[([A_i + G_i C_j]e + G_i w_e + m_e] + (\sum_{ij}(m_i - h_i)h_j[G_i C_j e + G_i w_e])$
For the purpose of this analysis, assume we can approximate the system accurately, such that: $m_i = h_i$ giving:

$\dot{e} = \sum_i h_i h_j[([A_i + G_i C_j]e + G_i w_e + m_e]$

Using the error form of the Lyapunov function: $V(e) = e^T P e$ where $P = P^T > 0$ is a symmetric positive definite matrix, when differentiating with respect to time: $\dot{V} = \dot{e}^T P e + e^T P \dot{e}$ using the above error we can rewrite $\dot{V}$ in the form:

$\dot{V} = \sum_{ij} h_i h_j[(e^T P[A_i + G_i C_j]e + e^T P G_i w_e + e^T P m_e) + e^T[A_i^T + G_i^T C_j^T]Pe + w_e^T G_i^T Pe + m_e^T Pe]$

When written in matrix form this becomes:

$$\dot{V} = \sum_{ij} h_i h_j (e^T w_e^T m_e^T)\begin{pmatrix} (P[A_i + G_i C_j] + [A_i^T + G_i^T C_j^T]P) & PG_i & P \\ G_i^T P & 0 & 0 \\ P & 0 & 0 \end{pmatrix}\begin{pmatrix} e \\ w_e \\ m_e \end{pmatrix}$$

Now, rewriting the above by pulling out terms, where $Q = Q^T > 0, M = M^T > 0$ are positive symmetric matrices:

$$\dot{V} = \sum_{ij} h_i h_j (e^T w_e^T m_e^T) \begin{pmatrix} (P[A_i + G_i C_j] + [A_i^T + G_i^T C_j^T]P) + Q & PG_i & P \\ G_i^T P & -\eta M & 0 \\ P & 0 & -\eta P \end{pmatrix} \begin{pmatrix} e \\ w_e \\ m_e \end{pmatrix}$$
$$- e^T Q e + \eta w_e^T M w_e + \eta m_e^T P m_e \tag{44}$$

Under the negative definiteness constraint:

$$\begin{pmatrix} (P[A_i + G_i C_j] + [A_i^T + G_i^T C_j^T]P) + Q & PG_i & P \\ G_i^T P & -\eta M & 0 \\ P & 0 & -\eta P \end{pmatrix} < 0$$

The resulting inequality is:
$\dot{V} < -e^T Q e + \eta w_e^T M w_e + \eta m_e^T P m_e$

Integrating up, this yields:
$\int_0^T e^T Q e < V(0) - V(T) + \int_0^T w_e^T M w_e + \eta m_e^T P m_e$

where $V(T) > 0$ and is dropped from the inequality. In order to allow the above negative definiteness constraint to be solvable using MATLAB LMI, the constraint must be rewritten to be a linear functions of variables and constant terms. Consider again:

$$\begin{pmatrix} (P[A_i + G_i C_j] + [A_i^T + G_i^T C_j^T]P) + Q & PG_i & P \\ G_i^T P & -\eta M & 0 \\ P & 0 & -\eta P \end{pmatrix} < 0$$

Let $K_{ij} = G_i C_j$
Then $G_i = G_i C_j C_j^T (C_j C_j^T)^{-1} = K_{ij} C_j^T (C_j C_j^T)^{-1}$
Then letting $R_j = C_j^T (C_j C_j^T)^{-1}$ one obtains:

$$\begin{pmatrix} A_i^T P + PA_i + K_{ij}^T P + PK_{ij} + Q & PK_{ij} R_j & P \\ R_j^T K_{ij}^T P & -\eta M & 0 \\ P & 0 & -\eta P \end{pmatrix} < 0$$

Now set $N_{ij} = PK_{ij}$ obtain the final form, directly solvable by MATLAB LMI 'feasp' solver:

$$\begin{pmatrix} A_i^T P + P A_i + N_{ij}^T + N_{ij} + Q & N_{ij} R_j & P \\ R_j^T N_{ij}^T & -\eta M & 0 \\ P & 0 & -\eta P \end{pmatrix} < 0$$

What are the limitations of this method? The first, as is the case for the original full state output, is that fact that it required the control transfer matrix B to be invertible, therefore meaning it has rank equal to the dimensionality of the system that is, every system variable is perturbed. This may not be feasible for many systems where certain state variables are not accessible. Here this issue will be noted, but not dealt with.

One should note that now, instead of the control being proportional to the full state, it is instead proportional to the output state. A consequence of this is that in the case when the output state error is synchronized, the control is deactivated, which does not guarantee that the the full state error is zero when not all state variables are coupled to the output state. One possible method for resolving this is to utilize a dynamic control term, added into the system via an extension of the original system defining an extension system $\tilde{x}^T = (x^T, z^T)$, with the corresponding time derivative term: $\dot{\tilde{x}} = (\dot{x}^T, \dot{z}^T)$. In the case when $z \in \mathbb{R}^1$ the new equations for the state of the system can be easily obtained by simply extending the dimensionality of the system transfer matrices: $\tilde{A}_i \in \mathbb{R}^{(n+1)\times(n+1)}$ where $A_i \in \mathbb{R}^{n\times n}$.

As such the same design technique can then be applied to synthesize a controller. The added new freedom comes from the ability to set the lowest row to any desirable value and the right-most column to a combination of values that determines which variables will be affected by this new forcing term, for example setting the last column to: $(0, 1, 0, 0, 0)^T$ means that only x_2 is forced by the new dynamic control allowing for increased flexibility. For fixed parameter mismatch in the case of output variables v_1, v_2, i_L being for the memristor it is found that the offset of ϕ is stabilized at around a fixed offset value determined by the mismatch in drive and response parameters.

B Graphical User Interface

B.1 Graphical User Interface

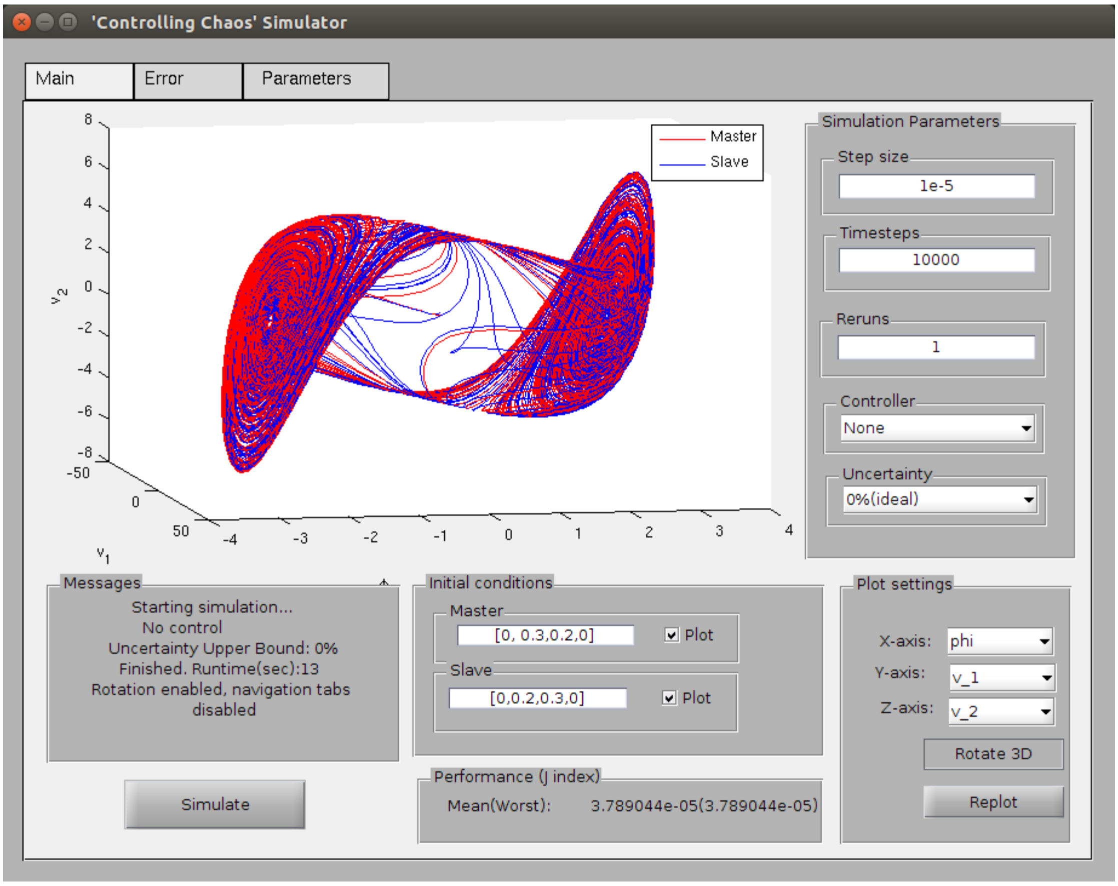

Figure 11: Main tab of graphical user interface

In the above Fig. 11 the main section of the GUI is presented. The goal here was to allow simple user interaction with the system and thus hides the majority of the parameters set in the code for each of the different controllers other than basic parameters that require modification in the process of experimentation. The tabs template is borrowed from 'Simple Optimized GUI Tabs' created by the WFAToolbox Team and is available via Mathworks File Exchange. The 'Error' tab contains the four state variable error traces, while the 'Parameters' enables the user to change the nominal parameter values and displays the differential equation model, Chua circuit diagram and memductance function, for ease of reference.

Simulation Parameters

- Step size - Runge-Kutta integration step size 'h'.

- Timesteps - Number of integration steps to perform for a given run.

- Reruns - Total number of runs to perform for this level of uncertainty, introduced to increase reliability of J index estimate (bottom-center).

- Uncertainty - Maximum bound on parameter offset of response system from nominal model values.

- Controller - type of controller selected for given simulation

  - None : No coupling between systems.

  - Fuzzy : Full state fuzzy logic controller
    [section: 4.1]

  - FuzzyOutput : Output state fuzzy logic control (output variables: ϕ, v_1)
    [section: 5]

  - FuzzyOutputDelay : Output state fuzzy logic control with delayed error term
    [section: 5.2]

  - SlidingMode : Sliding mode controller, applying control to v_1
    [section: 4.3]

  - Sos : Full state polynomial controller [Note: To make the program independent of SOSTOOLS, polynomial controller synthesis must be done separately]
    [section: 4.2]

  - Impulsive : Intermittent direct coupling controller based on [59], constitutes an extension of PC type control. Applied to variables ϕ and v_1.

Plot settings

- X,Y,Z axes - Select variable to be plotted on corresponding axis

- Rotate 3D - Toggle on to enable plot rotation [Note: This disables tab navigation, toggle off to enable it]

- Replot - Replot data using selected axes variables

Initial Conditions

- Initial conditions format: [ϕ,v_1,v_2,i_L]
- Plot - Toggle to plot corresponding system state phase space trace

B.2 Code examples

Instead of using the GUI, direct calls to the simulation can be performed, allowing greater flexibility in the simulations one can run and produce alternative more detailed plots of performance.

Basic call syntax

```
chaosSim(reruns,steps,stepSize, fracErrs,master_init,slave_init)
```

Similar to the above GUI parameters, but where instead 'fracErrs' may be an array of tolerance bounds, as opposed to a single value, thus allowing evaluation of multiple levels of uncertainty within a single call. A runnable example is included in the main directory of the project. Note that the default controller is set to 'None' and default control start time is 0. To run a fuzzy logic controller from the 100^{th} time step of the system:

```
global controller;
controller = controllerType.fuzzy;
global controlStart;
controlStart = 100;
chaosSim(1,10000,1e-8,[0],[0,-1,2,0],[0,1,0.5,0]);
```