Санкт-Петербургский государственный университет

Прикладная математика и информатика

Динамические системы, эволюционные уравнения, экстремальные задачи
и математическая кибернетика

Семенов Данила Михайлович

Синхронизация в спайковых нейронных сетях

Магистерская диссертация

Научный руководитель:
д. т. н., профессор Фрадков А. Л.

Рецензент:
к. ф.-м. н., Усик Е. В.

Санкт-Петербург
2019

SAINT PETERSBURG STATE UNIVERSITY

Applied Mathematics and Computer Science

Dynamical Systems, Evolution Equations, Extremal Problems

and Mathematical Cybernetics

Danila Semenov

Synchronization in Spiking Neural Networks

Master's Thesis

Scientific supervisor:
D.Sc. in Engineering,
Professor Alexander L. Fradkov.

Reviewer:
Cand. Sc. in Physics and Mathematics,
Egor V. Usik

Saint Petersburg
2019

Оглавление

Введение

Многочисленные исследования синхронизации в системах различной природы позволяют заключить, что явление синхронизации может рассматриваться с общих и единых позиций [2, 10, 27]. Установление этого факта привело к созданию целой междисциплинарной области, основу которой заложил ряд научных дисциплин. В частности, к таким дисциплинам относятся биология и медицина.

В природе существует большое количество биологических систем, демонстрирующих различные синхронные режимы функционирования. В качестве примеров таких систем можно привести скоординированную активность сердечных клеток, позволяющую сердцу сокращаться; птиц, летящих стаей; рой светлячков, синхронно вспыхивающий в ночном лесу [13, 26]. Кроме того, важнейшим примером таких систем являются популяции нейронов в мозге человека или животного. Хорошо известно, что синхронизация большого количества нейронов нервной системы играет ключевую роль в формировании макроколебаний в мозге [27, 38]. Также было установлено, что ряд заболеваний центральной нервной системы (например: эссенциальный тремор, эпилепсия, болезни Паркинсона и Альцгеймера, а также различные когнитивные расстройства) напрямую связан с аномальной синхронизацией некоторых нейронных популяций [33, 39]. Сегодня в терапии данных заболеваний активно внедряются методы, основанные на использовании электрических импульсов для подавления патологической синхронизации в определенных зонах мозга [15]. Кроме того, такие методы могут включать в себя принцип биологической обратной связи, благодаря которому эффективность такой терапии может возрасти [8, 36]. Однако такие методы пока находятся на раннем этапе развития и требуют качественного математического описания. Такое описание возможно получить, если построить математическую модель популяции биологических нейронов и исследовать условия их синхронизации внутри данной популяции.

Одним из подходов к построению математических моделей популяции биологических нейронов является представление популяции в виде сложной динамической сети, узлами которой являются модели единичных нейронов. Первой единичной моделью является модель Ходжкина-Хаксли [21],

представляющая собой динамическую систему и описывающая динамику распространения спайков[1] вдоль мембраны клетки. На настоящий момент существуют модели, близкие к модели Ходжкина-Хаксли. К таким моделям относятся: модель Моррис-Лекара, система ФитцХью-Нагумо, система Хиндмарш-Роуз и другие. Все эти модели также способны демонстрировать различные режимы активности нейрона, в том числе и режимы спайковой активности. [18, 22]. Таким образом, спайковая нейронная сеть — это динамическая сеть, узлы которой представляют собой математические модели нейронов, способные демонстрировать спайковую активность нервной клетки.

На сегодняшний день существует множество научных работ, посвященных теме синхронизации в спайковых нейронных сетях. Однако до недавнего времени практически все исследования ограничивались случаем идентичных узлов в сети [14, 37]. На практике же сети биологических нейронов гетерогенны, поскольку нейроны сети могут обладать различными физиологическими особенностями, т. е. являются неидентичными между собой. Этот факт побудил проведение ряда исследований, в которых были получены условия синхронизации в гетерогенных сетях ФитцХью-Нагумо [28–31]. Кроме проблемы гетерогенности в нейронных популяциях, существует проблема определения характеристик нейронов, необходимых для управления ими в их популяции. В настоящей работе предпринята попытка решения приведенных выше проблем методами адаптивной теории управления [1, 3, 4, 23]. В качестве исследуемой модели нейронной популяции выбрана гетерогенная спайковая нейронная сеть Хиндмарш-Роуз.

Работа состоит из четырех частей. В первой части работы приводятся необходимые сведения для решения перечисленных выше проблем. Вторая часть посвящена синхронизации сети, состоящей из двух связанных неидентичных систем Хиндмарш-Роуз при наличии возмущений, действующих на каждый узел. В третьей части исследуется случай адаптивной синхронизации для сети с аналогичной топологией. Четвертая часть посвящена адаптивной синхронизации в гетерогенной сети Хиндмарш-Роуз. Все полученные теоретические результаты согласуются с результатами компьютерного моделирования, проведенного в среде MATLAB.

[1]Спайк (англ. *spike*) — одна из фаз потенциала действия биологического нейрона, включающая себя процессы деполяризации и реполяризации мембранного потенциала клетки.

1 Предварительные сведения

1.1 Устойчивость нелинейных систем

Рассмотрим систему однородных дифференциальных уравнений:

$$\dot{x} = f(t, x), \tag{1.1}$$

где $f\colon \mathbb{R}_+ \times \mathbb{R}^n \to \mathbb{R}^n$ непрерывна и локально липшицева по аргументу x. Далее будем предполагать, что все рассматриваемые решения $x(t, t_0, x_0)$ с начальными данными $x(t_0, t_0, x_0) = x_0$ определены на интервале $[t_0, +\infty)$.

Определение 1.1. *Будем говорить, что решение $x(t, t_0, x_0)$ системы (1.1) устойчиво по Ляпунову, если $\forall \varepsilon > 0$ $\exists \delta_\varepsilon = \delta_\varepsilon(\varepsilon, t_0) > 0$ такое, что для всех x_0', удовлетворяющих неравенству $|x_0 - x_0'| \leqslant \delta_\varepsilon$, выполняется соотношение*

$$|x(t, t_0, x_0) - x(t, t_0, x_0')| \leqslant \varepsilon \quad \forall t \geqslant t_0.$$

Определение 1.2. *Если решение $x(t, t_0, x_0)$ устойчиво по Ляпунову и $\exists \delta = \delta(t_0) > 0$ такое, что для всех x_0', удовлетворяющих неравенству $|x_0 - x_0'| \leqslant \delta$ выполняется соотношение*

$$\lim_{t \to +\infty} |x(t, t_0, x_0) - x(t, t_0, x_0')| = 0,$$

то говорят, что решение $x(t, t_0, x_0)$ асимптотически устойчиво.

Приведем одно замечание, которое является важным для случая, когда система (1.1) нестационарна.

Замечание 1.1. *Если в определениях 1.1 и 1.2 величины δ_ε и δ не зависят от t_0, то говорят, что решение $x(t, t_0, x_0)$ равномерно устойчиво по Ляпунову и равномерно асимптотически устойчиво.*

Наиболее распространенным средством анализа устойчивости нелинейных систем является прямой (второй) метод Ляпунова. Метод основан на использовании скалярных функций, обладающих на решениях динамической системы некоторыми специальными свойствами и получивших название функций Ляпунова.

Далее будем предполагать, что рассматриваемое решение $x(t, t_0, x_0)$ системы (1.1) является нулевым, а функциями Ляпунова будем называть непрерывно дифференцируемые скалярные функции $V(x)$ и $V(t, x)$, которые в некоторой окрестности точки $x = 0$ являются положительно определенными. Ознакомиться с определением положительной определенности для функций $V(x)$ и $V(t, x)$ можно в работе [4].

Перейдем к формулировкам теорем прямого метода Ляпунова, предлагающих достаточные условия устойчивости и асимптотической устойчивости нелинейных систем.

Теорема 1.1 (об устойчивости)**.** *Нулевое решение системы* (1.1) *устойчиво по Ляпунову, если в некоторой окрестности точки* $x = 0$ *существует функция Ляпунова* $V(t, x)$ *такая, что* $\forall t \geqslant t_0$, $t_0 \in \mathbb{R}_+$ *выполняется неравенство*

$$\dot{V}(t, x) \leqslant 0.$$

Теорема 1.2 (об асимптотической устойчивости)**.** *Нулевое решение системы* (1.1) *асимптотически устойчиво, если в некоторой окрестности точки* $x = 0$ *существует функция Ляпунова* $V(t, x)$ *такая, что* $\forall t \geqslant t_0$, $t_0 \in \mathbb{R}_+$ *выполняются следующие условия:*

1) $V(t, x) \leqslant W_0(x)$,

2) $\dot{V}(t, x) < -W(x)$,

где $W_0(x)$ *и* $W(x)$ *— положительно определенные функции.*

Приведем важное замечание, которое позволяет ослабить условия теоремы 1.2 на случай, когда система (1.1) стационарна.

Замечание 1.2. *Если система* (1.1) *стационарна и в качестве функции Ляпунова рассматривается функция* $V(x)$*, то условие 1 теоремы 1.2 выполняется по определению.*

Доказательства теорем 1.1 и 1.2 приведены в [3, 4].

1.2 Определение и некоторые виды синхронизации

В данном разделе приводится общее определение синхронизации, а также рассматриваются некоторые ее виды, такие как координатная синхронизация и обобщенная (частичная) координатная синхронизация.

Под синхронизацией принято понимать согласованное во времени функционирование двух или нескольких процессов или объектов [5, 10]. В частности, это может быть согласованное изменение некоторых количественных характеристик двух или нескольких систем, или сближение переменных состояния рассматриваемых систем.

В некоторых случаях синхронизация может проявляться в силу естественных свойств взаимодействующих систем. Тогда говорят о самосинхронизации. В других случаях для согласованного поведения объектов необходимо вводить в систему дополнительные связи и воздействия. Такая синхронизация называется принудительной или управляемой синхронизацией.

Для рассмотрения различных вопросов синхронизации с единых позиций приведем общее определение синхронизации процессов и объектов различной природы.

Определение 1.3. *Будем говорить, что имеет место синхронизация процессов (объектов)* $x_i(t) \in X$, $i = 1, 2, \ldots, N$, *относительно характеристики* C_t *и функций сравнения* F_i, *если существуют числа* $\tau_i \in \mathbb{R}$, $i = 1, 2, \ldots, N$ *такие, что* $\forall t \geqslant 0$ *выполняются соотношения*

$$F_1\left(C_{t+\tau_1}[x_1]\right) = F_2\left(C_{t+\tau_2}[x_2]\right) = \ldots = F_N\left(C_{t+\tau_N}[x_N]\right). \tag{1.2}$$

В определении 1.3 под X понимается некоторое функциональное пространство. Характеристика C_t называется показателем синхронизации или индексом синхронизации и определяется как $C_t \colon X \to C$, где C есть множество возможных значений C_t. Важно, что характеристика C_t предполагается одинаковой для всех процессов или объектов. Значение характеристики C_t может быть скаляром, вектором, матрицей, а также функцией. Для сравнения значений характеристики различных процессов вводится набор, не зависящих от времени, вектор-функций $F_i \colon C \to \mathbb{R}^m$, $i = 1, 2, \ldots, N$, называемых функциями сравнения.

Определение 1.4. *Приближенной синхронизацией (ε-синхронизацией) называется случай, когда соотношения (1.2) выполняются лишь приближенно, с точностью до ε:*

$$\left\| F_i\left(C_{t+\tau_i}[x_i]\right) - F_j\left(C_{t+\tau_j}[x_j]\right) \right\| \leqslant \varepsilon \qquad \forall i, j, \quad t \geqslant t^*.$$

Определение 1.5. *Асимптотической синхронизацией называется случай, когда погрешность выполнения соотношений (1.2) со временем исчезает:*

$$\lim_{t \to +\infty} \left\| F_i\left(C_{t+\tau_i}[x_i]\right) - F_j\left(C_{t+\tau_j}[x_j]\right) \right\| = 0 \qquad \forall i, j.$$

В определениях 1.4 и 1.5 под $\|\cdot\|$ следует понимать евклидову норму в пространстве $\mathbb{R}^m$.

Общее определение синхронизации 1.3 охватывает большое число видов синхронного поведения процессов или объектов. Однако в настоящей работе далее будут рассмотрены только два: координатная синхронизация и обобщенная (частичная) координатная синхронизация. Рассмотрим данные виды синхронизации более подробно.

Определение 1.6. *Координатной синхронизацией процессов или объектов $x_i(t)$, $i = 1, 2, \ldots, N$, называется вид синхронизации, при котором координаты векторов состояний рассматриваемых процессов (объектов) совпадают.*

Очевидно, что координатная синхронизация укладывается в общее определение 1.3, если ввести показатель синхронизации $C_t(x_i) = x_i(t)$, а функции сравнения взять тождественными: $F_i(x) = x$, $i = 1, 2, \ldots, N$.

Определение 1.7. *Обобщенной (частистной) координатной синхронизацией называют координатную синхронизацию, при которой совпадает лишь часть фазовых координат $y_i = h(x_i)$, $i = 1, 2, \ldots, N$.*

Обобщенная координатная синхронизация также укладывается в определение 1.3, если взять $C_t(x_i) = x_i(t)$ и $F_i(x) = h(x)$, $i = 1, 2, \ldots, N$.

1.3 Вспомогательные сведения из теории графов

Графом (ориентированным) называется упорядоченная пара $G = (N, E)$ из двух конечных множеств $N = \{1, 2, \ldots, n\}$ (множество узлов или вер-

шин графа) и $E \subset N \times N$ (множество дуг графа). Узел i соединен с узлом j в графе G, если $(i, j) \in E$. Граф называется простым, если отсутствуют петли и между узлами может быть максимум одна дуга. Пара (N, E'), где $E' \subset E$, называется частичным графом графа (N, E).

Дуга (i, j) графа G представляется в виде стрелки с началом в i и концом в j. Говорят, что дуга (i, j) — исходящая по отношению к узлу i и входящая по отношению к узлу j; узел i называется родительским, а узел j — дочерним.

Полустепенью захода вершины i называется количество входящих в нее дуг, а полустепенью исхода вершины i — количество выходящих дуг.

Множеством соседей узла i называется множество $N^i = \{j : (j, i) \in E\}$, т. е. множество узлов с дугами, входящими в i. Число соседей $|N^i|$ узла i равно его полустепени захода.

Если $(i, j) \in E \Rightarrow (j, i) \in E \; \forall i, j$, то граф называется двунаправленным (неориентированным), иначе граф называется ориентированным (или орграфом). Для неориентированных графов дуги обычно называют ребрами. Если полустепень захода совпадает с полустепенью исхода $\forall i \in N$, то граф называется сбалансированным.

Направленный путь из узла i_1 в узел i_s состоит из последовательности узлов $i_1, \ldots, i_s$, $s \geqslant 2$ таких, что $(i_k, i_{k+1}) \in E$, $k \in \{1, 2, \ldots, s-1\}$.

Говорят, что узел i связан с узлом j, если есть направленный путь из i в j. Расстояние от i до j — это длина кратчайшего пути из i в j. Граф называется сильно связным, если i и j связаны для всех различных узлов $i, j \in N$. Для двунаправленных графов, если есть направленный путь из i в j, тогда есть направленный путь из j в i. Такой граф называется связным.

Сопоставим каждому ребру $(i, j) \in E$ вес α_{ij}. Предположим, что все веса положительны. Граф может быть представлен матрицей смежности (связности) $A = [\alpha_{ij}]$ с весами $\alpha_{ij} > 0$, если $(i, j) \in E$, и $\alpha_{ij} = 0$ в противном случае. Также отметим, что $\alpha_{ii} = 0$, т. е. в графе отсутствуют петли.

Определим взвешенную полустепень захода вершины i как сумму i-го столбца матрицы A: $d^i_{in}(A) = \sum_{j=1}^{N} \alpha_{ji}$ и взвешенную полустепень исхода вершины i как сумму i-ой строки матрицы : $d^i_{out}(A) = \sum_{j=1}^{N} \alpha_{ij}$.

Если $\alpha_{ij} = \alpha_{ji} \; \forall i, j \in N$, т. е. веса ребер (i, j) и (j, i) совпадают, то граф является неориентированным (двунаправленным). При этом матрица его

смежности симметрична.

Граф называется сбалансированным по весам, если взвешенная полустепень захода совпадает с взвешенной полустепенью исхода для всех узлов i. Если все ненулевые веса ребер равны 1, это то же самое, что и определение сбалансированного графа. Неориентированный граф является сбалансированным, так как $A = A^T$ и сумма i-ой строки равна сумме по i-му столбцу. В дальнейшем, говоря о полустепени захода или исхода, а также о сбалансированности графа, будем подразумевать взвешенные полустепени захода или исхода и сбалансированность по весам.

Для графа G определим диагональную матрицу степеней вершин графа $D(A) = \operatorname{diag}\left\{d^i_{out}(A)\right\}$ из полустепеней исхода. Тогда матрицей Лапласа (лапласианом) графа будем называть матрицу вида $L(A) = D(A) - A$, т. е.

$$L(A) = \begin{bmatrix} \sum_{j=1}^{N} \alpha_{1\,j} & -\alpha_{1\,2} & \dots & -\alpha_{1\,N} \\ -\alpha_{2\,1} & \sum_{j=1}^{N} \alpha_{2\,j} & \dots & -\alpha_{2\,N} \\ \vdots & \vdots & \ddots & \vdots \\ -\alpha_{N\,1} & -\alpha_{N\,2} & \dots & \sum_{j=1}^{N} \alpha_{N,j} \end{bmatrix}.$$

Заметим, что суммы по строкам элементов матрицы Лапласа равны нулю. Следовательно, любой вектор, составленный из одинаковых констант, является правым собственным вектором, соответствующим нулевому собственному значению $L(A)$.

Обозначим через $\mathbf{1}_N$ вектор-столбец размерности N, состоящий из единиц. Известно, что матрица $L(A)$ обладает следующими свойствами [5]:

1) матрица $L(A)$ имеет нулевое собственное число, которому соответствует правый собственный вектор $\mathbf{1}_N$: $L(A)\mathbf{1}_N = 0$;

2) кратность нулевого собственного числа $L(A)$ неориентированного графа равна количеству компонент связности;

3) нулевое собственное число лапласовской матрицы $L(A)$ имеет единичную кратность, если соответствующий орграф сильно связен;

4) все собственные числа лапласовской матрицы имеют неотрицательные вещественные части;

5) для сбалансированного графа вектор $\mathbf{1}_N$ является левым собственным вектором, соответствующим нулевому собственному числу: $\mathbf{1}_N^T L(A) = 0$.

Второе собственное число λ_2 матрицы $L(A)$ называют числом Фидлера (Fiedler) или алгебраической связностью графа G [16]. Алгебраическая связность графа λ_2 обладает одним важным свойством: она положительна в том и только в том случае, когда соответствующий ей неориентированный граф G связен [12, 16, 24, 32].

1.4 Метод скоростного градиента

Кратко опишем метод скоростного градиента, который мы в дальнейшем будем использовать для решения задач адаптивной синхронизации. Метод скоростного градиента впервые был предложен в конце 70-х годов прошлого века для решения задач адаптивного управления [11]. Метод опирается на использование функций Ляпунова и требует определения цели управления как минимизации некоторой скалярной целевой функции (функционала) до заданной величины.

Рассмотрим следующую динамическую систему:

$$\dot{x} = f(t, x, \theta), \tag{1.3}$$

где $x \in \mathbb{R}^n$ — вектор состояния системы; $\theta \in \mathbb{R}^m$ — вектор настраиваемых параметров; $f(t, x, \theta)$ — вектор-функция, которая является кусочно-непрерывной по t и непрерывно дифференцируемой по x и θ. Цель управления может быть сформулирована одним из двух следующих способов:

$$Q_t \to 0, \text{ при } t \to \infty \quad (Q_t \leqslant \Delta, \forall t > T), \tag{1.4}$$

где $Q_t \geqslant 0$ — целевой функционал; Δ, T — некоторые постоянные, значения которых зависят от конкретной задачи.

Далее будем рассматривать только случай, когда Q_t является локальным функционалом, т. е. $Q_t = Q(t, x(t))$, где $Q(t, x(t))$ — скалярная функция $n+1$ переменных. В качестве такого функционала для решения задачи синхронизации можно выбрать квадратичную форму

$$Q(x) = (x_1 - x_2)^T P (x_1 - x_2),$$

где $P = P^T > 0$ и $x = \{x_1, x_2\}$ — расширенный вектор состояний обобщенной системы.

Для построения алгоритма управления найдем скалярную функцию $\dot{Q}_t = \omega(t, x, \theta)$, которая представляет собой скорость изменения Q_t в силу системы (1.3):

$$\omega(t, x, \theta) = \frac{\partial Q(t, x)}{\partial t} + \left[\nabla_x Q(t, x)\right]^T f(t, x, \theta). \tag{1.5}$$

Затем вычислим градиент $\omega(t, x, \theta)$ по входным переменным θ:

$$\nabla_\theta \omega(t, x, \theta) = \left[\frac{\partial \omega}{\partial \theta}\right]^T = \left[\frac{\partial f}{\partial \theta}\right]^T \nabla_x Q(t, x). \tag{1.6}$$

Наконец, определим алгоритм изменения θ в дифференциальной форме:

$$\dot{u} = -\Gamma \nabla_\theta \omega(t, x, \theta), \tag{1.7}$$

где $\Gamma = \Gamma^T > 0$ — симметричная положительно определенная матрица, например, $\Gamma = \operatorname{diag}\{\gamma_1, \ldots, \gamma_m\}$, $\gamma_i > 0$.

Основную идею алгоритма (1.7) можно объяснить следующим образом. Для того, чтобы достигнуть цели управления (1.4), нужно изменять θ в направлении убывания Q_t. Однако это проблематично, поскольку Q_t не зависит явно от θ. Вместо этого, мы можем попробовать уменьшать $\dot{Q}_t$, стремясь к выполнению неравенства $\dot{Q}_t < 0$, которое, в свою очередь, приводит к уменьшению Q_t. Теперь функция $\dot{Q}_t = \omega(t, x, \theta)$ явно зависит от θ, что и позволяет записать алгоритм (1.7).

Сходимость метода скоростного градиента зависит от выполнения ряда условий. Теорему о достаточных условиях сходимости метода скоростного градиента можно найти в работе [17].

1.5 Спайковая модель Хиндмарш-Роуз

Система Хиндмарш-Роуз представляет собой спайковую модель биологического нейрона и описывается нелинейной системой дифференциальных

уравнений 3-го порядка:

$$\begin{aligned}
\dot{x}(t) &= y(t) - ax^3(t) + bx^2(t) - z(t) + u(t),\\
\dot{y}(t) &= c - dx^2(t) - y(t),\\
\dot{z}(t) &= r\big[s\,(x(t) - x_{\text{rest}}) - z(t)\big].
\end{aligned} \tag{1.8}$$

Здесь переменная $x(t)$ описывает динамику мембранного потенциала нейрона, а переменные $y(t)$ и $z(t)$ описывают работу натрий-калиевого насоса. Поскольку скорость изменения переменной $z(t)$ определяется параметром r таким, что $0 < r < 1$, то $z(t)$ описывает динамику медленного ионного тока, а $y(t)$ – быстрого ионного тока [19, 20, 35]. Внешнее воздействие на нейрон определяется переменной $u(t)$. Все параметры системы являются положительными, кроме параметра x_{rest}, отвечающего за потенциал покоя нервной клетки.

Изначально система Хиндмарш–Роуз описывалась нелинейной системой дифференциальных уравнений 2-го порядка и являлась упрощенным вариантом модели Ходжкина–Хаксли [19]. Однако дальнейшие эксперименты, проводимые над нервными клетками пресноводной улитки Lymnaea stagnalis, позволили обнаружить новые режимы функционирования нейронов, которые не учитывались в модели [20]. Потому авторы модели, Джеймс Хиндмарш и Мальком Роуз, приняли решение о включении в модель третьего уравнения. Это уравнение позволило учесть большинство режимов поведения биологического нейрона. В частности, модель (1.8) способна демонстрировать пачечную активность и посттормозную отдачу, а также адаптироваться к воздействию внешними стимулами [20, 35].

В дальнейшем будем предполагать, что решения системы (1.8) являются глобально предельно ограниченными по переменной t. С определением глобальной предельной ограниченности можно ознакомиться в работах [23, 25].

2 Синхронизация двух связанных неидентичных моделей Хиндмарш-Роуз при наличии возмущений

2.1 Постановка задачи

Рассмотрим гетерогенную динамическую сеть с возмущениями, состоящую из двух связанных моделей Хиндмарш-Роуз:

$$\begin{aligned}
&\dot{\mathbf{x}}_1(t) = f_1(\mathbf{x}_1(t)) + \varphi(\mathbf{x}_1(t), \mathbf{x}_2(t)) + \xi_1(t) + Bu(t),\\
&\dot{\mathbf{x}}_2(t) = f_2(\mathbf{x}_2(t)) + \varphi(\mathbf{x}_2(t), \mathbf{x}_1(t)) + \xi_2(t),\\
&f_i(\mathbf{x}_i(t)) = \begin{bmatrix} y_i(t) - ax_i^3(t) + bx_i^2(t) - z_i(t) \\ c - dx_i^2(t) - y_i(t) \\ r\left[s(x_i(t) - x_{\mathrm{rest}\,i}) - z_i(t)\right] \end{bmatrix},\\
&\varphi(\mathbf{x}_i(t), \mathbf{x}_j(t)) = \begin{bmatrix} -\sigma\,(x_i(t) - x_j(t)) & 0 & 0 \end{bmatrix}^T,
\end{aligned} \tag{2.1}$$

где $i, j = 1, 2$. Здесь $\mathbf{x}_i(t) = \mathrm{col}\,(x_i(t), y_i(t), z_i(t))$ — вектор состояния i-ой модели; $B = \mathrm{col}\,(1, 0, 0)$; $u(t)$ — управление; $\sigma \geqslant 0$ — сила связи между моделями; $\xi_i(t) = \mathrm{col}\,(\xi_{x\,i}(t), \xi_{y\,i}(t), \xi_{z\,i}(t))$ — вектор-функция, описывающая возмущения i-ой модели; a, b, c, d, r, s — параметры, принимающие положительные значения; $x_{\mathrm{rest}\,i}$, $i = 1, 2$ — параметры, значения которых отрицательны.

Далее будем предполагать, что параметры моделей (2.1) известны и могут быть использованы при построении управления $u(t)$. Также предполагается, что вектор-функции $\xi_1(t)$ и $\xi_2(t)$, описывающие возмущения, являются равномерно-ограниченными функциями, т. е. функциями вида

$$|\xi_{x\,i}(t)| \leqslant \frac{\Delta_x}{2}, \quad |\xi_{y\,i}(t)| \leqslant \frac{\Delta_y}{2}, \quad |\xi_{z\,i}(t)| \leqslant \frac{\Delta_z}{2}, \quad i = 1, 2, \quad \forall t \geqslant 0, \tag{2.2}$$

где Δ_x, Δ_y, Δ_z — некоторые положительные постоянные.

Теперь остановимся подробнее на постановке задачи управляемой координатной ε-синхронизации моделей (2.1). Для этого введем в рассмотрение некоторые обозначения, которые в дальнейшем будут использованы в ходе

решения данной задачи:

$$\delta(t) = \mathbf{x}_1(t) - \mathbf{x}_2(t), \quad \eta(t) = \xi_1(t) - \xi_2(t), \quad \Delta_{\text{rest}} = x_{\text{rest}\,1} - x_{\text{rest}\,2},$$
$$\psi_1(t) = x_1^2(t) + x_1(t)x_2(t) + x_2^2(t), \quad \psi_2(t) = x_1(t) + x_2(t),$$

где $\delta(t) = \operatorname{col}(\delta_x(t), \delta_y(t), \delta_z(t))$ — вектор отклонений между координатами состояний систем (2.1); $\eta(t) = \operatorname{col}(\eta_x(t), \eta_y(t), \eta_z(t))$ — вектор разности функций $\xi_1(t)$ и $\xi_2(t)$, причем из неравенств (2.2) следует, что

$$|\eta_x(t)| \leqslant \Delta_x, \quad |\eta_y(t)| \leqslant \Delta_y, \quad |\eta_z(t)| \leqslant \Delta_z, \quad \forall t \geqslant 0. \tag{2.3}$$

Основываясь на приведенных выше обозначениях, сформулируем задачу управляемой координатной ε-синхронизации моделей (2.1) в виде следующей цели управления:

$$|\delta_x(t)| \leqslant \varepsilon, \quad |\delta_y(t)| \leqslant \varepsilon, \quad |\delta_z(t) + s\Delta_{\text{rest}}| \leqslant \varepsilon, \quad \forall t \geqslant t^*. \tag{2.4}$$

Таким образом, решение задачи управляемой координатной ε-синхронизации сводится к поиску такого управления $u(t)$, которое бы обеспечивало достижение цели управления (2.4).

2.2 Основной результат

Для решения поставленной задачи, в динамической сети (2.1) последовательно вычтем уравнения второй модели из уравнений первой и запишем следующую систему нелинейных дифференциальных уравнений:

$$\dot{\delta}(t) = F(\delta(t), \psi(t)) + 2\varphi(\mathbf{x}_1(t), \mathbf{x}_2(t)) + \eta(t) + Bu(t),$$
$$F(\delta(t), \psi(t)) = \begin{bmatrix} -\left(a\psi_1(t) - b\psi_2(t)\right)\delta_x(t) + \delta_y(t) - \delta_z(t) \\ -d\psi_2(t)\delta_x(t) - \delta_y(t) \\ r\left(s\delta_x(t) - \delta_z(t) - s\Delta_{\text{rest}}\right) \end{bmatrix}. \tag{2.5}$$

Здесь $\psi(t) = \operatorname{col}(\psi_1(t), \psi_2(t))$ — вектор-функция, описывающая нелинейную часть системы.

Перейдем к координатам $e_x(t) = \delta_x(t)$, $e_y(t) = \delta_y(t)$, $e_z(t) = \delta_z(t) + s\Delta_{\text{rest}}$.

Тогда цель управления (2.4) примет вид

$$|e_x(t)| \leqslant \varepsilon, \quad |e_y(t)| \leqslant \varepsilon, \quad |e_z(t)| \leqslant \varepsilon, \quad \forall t \geqslant 0, \tag{2.6}$$

а система (2.5) запишется как

$$\begin{gathered}\dot{e}(t) = F(e(t), \psi(t)) + 2\varphi(\mathbf{x}_1(t), \mathbf{x}_2(t)) + \eta(t) + Bu(t), \\ F(e(t), \psi(t)) = \begin{bmatrix} -\left(a\psi_1(t) - b\psi_2(t)\right) e_x(t) + e_y(t) - e_z(t) + s\Delta_{\text{rest}} \\ -d\psi_2(t)e_x(t) - e_y(t) \\ r\big(se_x(t) - e_z(t)\big) \end{bmatrix}.\end{gathered} \tag{2.7}$$

Здесь $e(t) = \operatorname{col}\left(e_x(t), e_y(t), e_z(t)\right)$ — вектор состояния системы (2.7).

Опираясь на цель управления (2.6), выполним переход от задачи управления синхронизацией в системах (2.1) к задаче поиска управления $u(t)$, гарантирующего устойчивость нулевого решения системы (2.7). Для поиска такого управления рассмотрим функцию Ляпунова в виде квадратичной формы:

$$V\left(e(t)\right) = \frac{1}{2}\left(e_x^2(t) + e_y^2(t) + \frac{1}{rs}e_z^2(t)\right). \tag{2.8}$$

Тогда производная $\dot{V}(e(t))$ вдоль траекторий системы (2.7) примет следующий вид:

$$\begin{aligned}\dot{V}(e(t)) = &-(a\psi_1(t) - b\psi_2(t) + 2\sigma)e_x^2(t) - e_y^2(t) - \frac{1}{s}e_z^2(t) + \\ &+ (1 - d\psi_2(t))e_x(t)e_y(t) + s\Delta_{\text{rest}}e_x(t) + u(t)e_x(t) + \\ &+ \eta_x(t)e_x(t) + \eta_y(t)e_y(t) + \frac{1}{rs}\eta_z(t)e(t).\end{aligned} \tag{2.9}$$

Введем следующий закон управления:

$$u(t) = -(\gamma_0 + b\psi_2(t))\delta_x(t) + d\psi_2(t)\delta_y(t) - s\Delta_{\text{rest}}, \tag{2.10}$$

где γ_0 — коэффициент усиления. В новых координатах $e_x(t)$, $e_y(t)$ и $e_z(t)$, управление (2.10) примет вид:

$$u(t) = -(\gamma_0 + b\psi_2(t))\delta_x(t) + d\psi_2(t)\delta_y(t) - s\Delta_{\text{rest}}. \tag{2.11}$$

Подставляя закон управления (2.11) в производную (2.9), получаем следующее выражение:

$$\begin{aligned}\dot{V}(e(t)) = &-(\gamma_0 + a\psi_1(t) + 2\sigma)e_x^2(t) + e_x(t)e_y(t) - e_y^2(t) - \frac{1}{s}e_z^2(t) \\ &+ \eta_x(t)e_x(t) + \eta_y(t)e_y(t) + \frac{1}{rs}\eta_z(t)e(t).\end{aligned} \tag{2.12}$$

Теперь полученное выражение (2.12) будем оценивать сверху. Во-первых, исключим из рассмотрения слагаемое, содержащее функцию $\psi_1(t)$, поскольку оно неположительно $\forall t \geqslant 0$. Тогда имеет место неравенство:

$$\begin{aligned}\dot{V}(e(t)) \leqslant &-(\gamma_0 + 2\sigma)e_x^2(t) + e_x(t)e_y(t) - e_y^2(t) - \frac{1}{s}e_z^2(t) \\ &+ \eta_x(t)e_x(t) + \eta_y(t)e_y(t) + \frac{1}{rs}\eta_z(t)e(t).\end{aligned} \tag{2.13}$$

Во-вторых, слагаемые, содержащие функции $\eta_x(t)$, $\eta_y(t)$ и $\eta_z(t)$, можно оценить сверху модулями от этих слагаемых. Таким образом, получим следующую оценку сверху для правой части неравенства (2.13):

$$\begin{aligned}\dot{V}(e(t)) \leqslant &-(\gamma_0 + 2\sigma)e_x^2(t) + e_x(t)e_y(t) - e_y^2(t) - \frac{1}{s}e_z^2(t) \\ &+ |\eta_x(t)e_x(t)| + |\eta_y(t)e_y(t)| + \frac{1}{rs}|\eta_z(t)e(t)|.\end{aligned} \tag{2.14}$$

Наконец, рассмотрим вспомогательное неравенство $\left(\sqrt{\nu}\alpha + \frac{1}{\sqrt{\nu}}\beta\right)^2 \geqslant 0$. Данное неравенство равносильно неравенству $|\alpha\beta| \leqslant \frac{\nu}{2}\alpha^2 + \frac{1}{2\nu}\beta^2$, а значит имеет место выражение:

$$\begin{aligned}\dot{V}(e(t)) \leqslant &-(\gamma_0 + 2\sigma)\, e_x^2(t) + e_x(t)e_y(t) - e_y^2(t) - \frac{1}{s}e_z^2(t) + \frac{\nu_x}{2}e_x^2(t) + \\ &+ \frac{1}{2\nu_x}\eta_x^2(t) + \frac{\nu_y}{2}e_y^2(t) + \frac{1}{2\nu_y}\eta_y^2(t) + \frac{\nu_z}{2rs}e_z^2(t) + \frac{1}{2rs\nu_z}\eta_z^2(t),\end{aligned} \tag{2.15}$$

где ν_x, ν_y, ν_z — некоторые положительные постоянные.

Далее представим правую часть неравенства (2.15) в следующем виде:

$$\begin{aligned}\dot{V}(e(t)) \leqslant &-\left(\gamma_0 + 2\sigma - \frac{1}{2}\right) e_x^2(t) + e_x(t)e_y(t) - \frac{1}{2}e_y^2(t) - \\ &- \frac{1}{2s}e_z^2(t) - \frac{1}{2}\left[\left(e_x^2(t) + e_y^2(t) + \frac{1}{s}e_z^2(t)\right) + \nu_x e_x^2(t) + \right. \\ &\left. + \frac{1}{\nu_x}\eta_x^2(t) + \nu_y e_y^2(t) + \frac{1}{\nu_y}\eta_y^2(t) + \frac{\nu_z}{rs}e_z^2(t) + \frac{1}{rs\nu_z}\eta_z^2(t)\right].\end{aligned} \tag{2.16}$$

В неравенстве (2.16) выделим квадратичную форму, а ν_x, ν_y, ν_z положим равными $1/2$, $1/2$ и $r/2$ соответственно. Отсюда получаем неравенство:

$$\dot{V}(e(t)) \leqslant e^T(t)We(t) - \frac{r}{2}V(e(t)) + \left(\eta_x^2(t) + \eta_y^2(t) + \frac{1}{r^2 s}\eta_z^2(t)\right), \tag{2.17}$$

где

$$W = \begin{bmatrix} -(\gamma_0 + 2\sigma - 1/2) & 1/2 & 0 \\ 1/2 & -1/2 & 0 \\ 0 & 0 & -1/2s \end{bmatrix}.$$

Теперь найдем такие γ_0 и σ, при которых квадратичная форма $e^T(t)We(t)$ в неравенстве (2.17) является отрицательно определенной. Используя критерий Сильвестра, получим следующее условие:

$$\gamma_0 + 2\sigma - 1 > 0. \tag{2.18}$$

Если условие (2.18) выполнено, тогда квадратичная форма в неравенстве (2.17) является отрицательно определенной. Кроме того, функции $\eta_x(t)$, $\eta_y(t)$ и $\eta_z(t)$ являются равномерно ограниченными. Поэтому правую часть неравенства (2.17) можно вновь оценить сверху:

$$\dot{V}(e(t)) \leqslant -\frac{r}{2}V(e(t)) + h, \tag{2.19}$$

где

$$h = \left(\Delta_x^2 + \Delta_y^2 + \frac{1}{r^2 s}\Delta_z^2\right).$$

Наконец, применяя известные теоремы сравнения дифференциальных неравенств [9] к неравенству (2.19) и устремляя t к бесконечности, получа-

ем верхний предел:

$$\overline{\lim_{t\to\infty}} V(e(t)) = \frac{2h}{r}. \tag{2.20}$$

Следовательно, траектории систем (2.5) и (2.7) ограничены эллипсоидом:

$$e(t)^T H^{-1} e(t) = 1, \tag{2.21}$$

где

$$H = \begin{bmatrix} 2\sqrt{h/r} & 0 & 0 \\ 0 & 2\sqrt{h/r} & 0 \\ 0 & 0 & 2\sqrt{sh} \end{bmatrix}.$$

Ограниченность траекторий в системах (2.5) и (2.7) свидетельствует о достижении цели управления (2.4), а значит и ε-синхронизации между связанными системами (2.1). Следует отметить, что точность такой синхронизации прямо пропорциональна величинам Δ_x, Δ_y и Δ_z. Полученный результат сформулируем в виде следующей теоремы.

Теорема 2.1. *Пусть условие* (2.18) *выполнено. Тогда закон управление* $u(t)$ *в форме* (2.10) *гарантирует достижение цели управления* (2.4) *при* ε *равном* $2\sqrt{h/r}$ *и* $\forall \mathbf{x}_1(0), \mathbf{x}_2(0)$ *систем* (2.1).

Таким образом, теорема 2.1 является достаточным условием для достижения ε-синхронизации связанных систем (2.1). Данный результат был опубликован в работах [6, 7].

2.3 Компьютерное моделирование

Проведем компьютерное моделирование с целью подтверждения результатов, полученных в разделе 2.2. Для этого предварительно определим параметры моделирования динамической сети (2.1). Их значения будут следующие:

$$a = 1, \quad b = 3, \quad c = 1, \quad d = 5, \quad r = 3 \cdot 10^{-3},$$
$$s = 4, \quad x_{\text{rest}\,1} = -1, \quad x_{\text{rest}\,2} = -0.99, \quad \sigma = 1 \cdot 10^{-3}, \quad \gamma_0 = 10.$$

Кроме того, предполагается, что в динамической сети (2.1) присутствуют возмущения, описываемые равномерно ограниченными функциями от

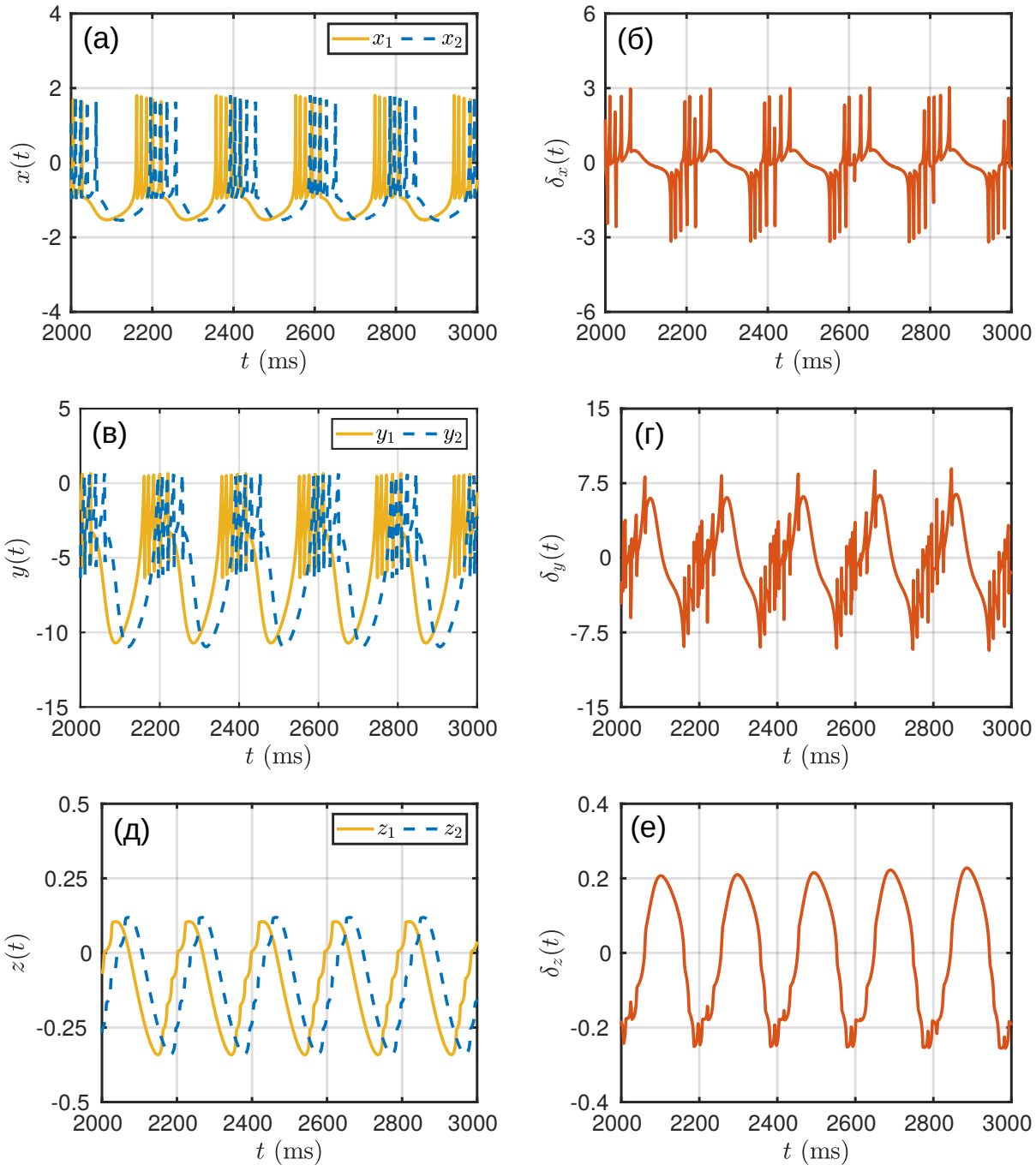

Рис. 2.1: Поведение двух связанных неидентичных моделей Хиндмарш-Роуз при наличии возмущений без управления: (а) динамика переменных $x_1(t)$ и $x_2(t)$; (б) разница между переменными $x_1(t)$ и $x_2(t)$; (в) динамика переменных y_1 и y_2; (г) разница между переменными y_1 и y_2; (д) динамика переменных z_1 и z_2; (е) разница между переменными z_1 и z_2

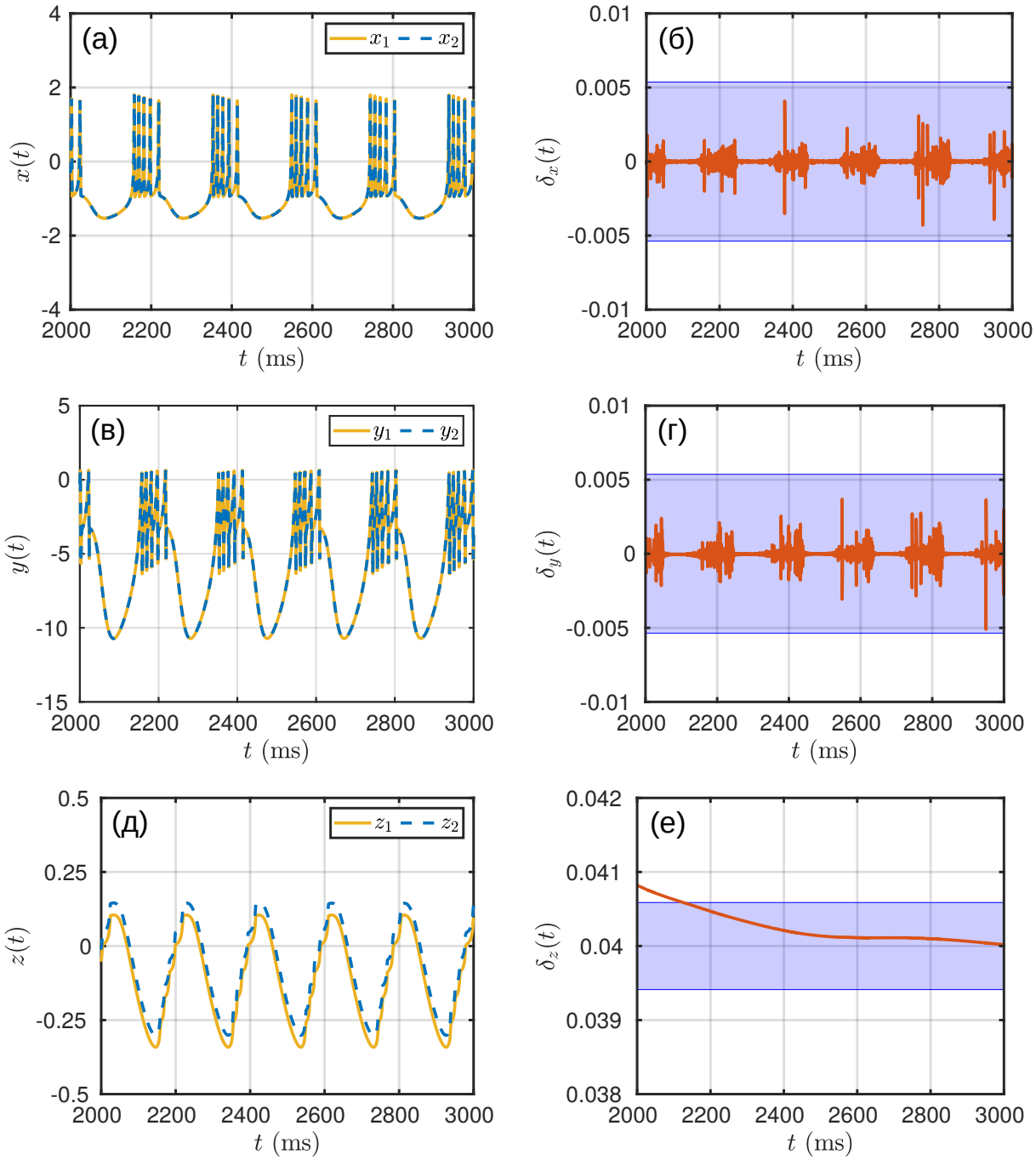

Рис. 2.2: Управляемая синхронизация двух связанных неидентичных моделей Хиндмарш-Роуз при наличии возмущений: (а) динамика переменных $x_1(t)$ и $x_2(t)$; (б) разница между переменными $x_1(t)$ и $x_2(t)$; (в) динамика переменных y_1 и y_2; (г) разница между переменными y_1 и y_2; (д) динамика переменных z_1 и z_2; (е) разница между переменными z_1 и z_2

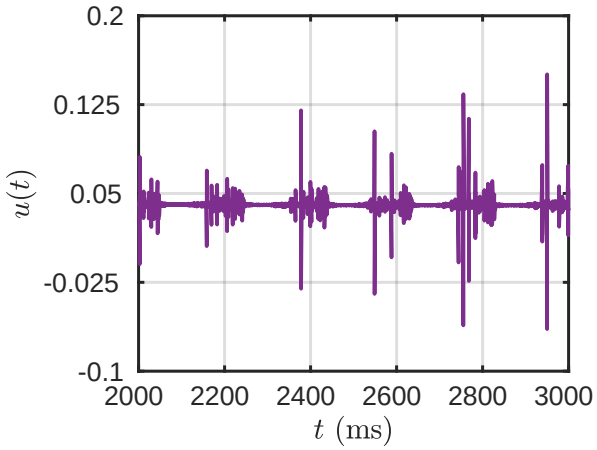

Рис. 2.3: Динамика закона управления $u(t)$

времени. Для моделирования таких возмущений были выбраны следующие функции:

$$\begin{aligned} \xi_{x,1} &= 0.01r\sin(100t), & \xi_{x,2} &= -0.01r\sin(100t), \\ \xi_{y,1} &= 0.01r\sin(t), & \xi_{y,2} &= -0.01r\sin(t), \\ \xi_{z,1} &= 0.02r^2\sin(0.01t), & \xi_{z,2} &= -0.02r^2\sin(0.01t) \end{aligned}$$

Теперь перейдем непосредственно к результатам моделирования. Для начала рассмотрим случай, когда на сеть (2.1) не оказывается никакого управляющего воздействия (рис. 2.1). Можно видеть, что в данном случае узлы динамической сети ведут себя асинхронно и в их динамике наблюдается некоторый временной сдвиг друг относительно друга.

Далее рассмотрим случай, когда динамическая сеть (2.1) находится под управлением (2.10). Поскольку значения параметров γ_0 и σ удовлетворяют условию теоремы 2.1, то закон управления (2.10) гарантирует достижение ε-синхронизации между узлами сети (2.1), что и проиллюстрировано на рис. 2.2. Динамика же самого закона управления изображена на рис. 2.3.

Таким образом, можно заключить, что результаты проведенного компьютерного моделирования полностью согласуются с полученными теоретическими результатами.

3 Адаптивная синхронизация двух связанных неидентичных моделей Хиндмарш-Роуз

3.1 Постановка задачи

Рассмотрим гетерогенную динамическую сеть, состоящую из двух связанных моделей Хиндмарш-Роуз:

$$\begin{aligned}
\dot{\mathbf{x}}_1(t) &= f_1(\mathbf{x}_1(t)) + \varphi(\mathbf{x}_1(t), \mathbf{x}_2(t)) + Bu(t), \\
\dot{\mathbf{x}}_2(t) &= f_2(\mathbf{x}_2(t)) + \varphi(\mathbf{x}_2(t), \mathbf{x}_1(t)), \\
f_i(\mathbf{x}_i(t)) &= \begin{bmatrix} y_i(t) - ax_i^3(t) + bx_i^2(t) - z_i(t) \\ c - dx_i^2(t) - y_i(t) \\ r\left[s(x_i(t) - x_{\mathrm{rest}\,i}) - z_i(t)\right] \end{bmatrix}, \\
\varphi(\mathbf{x}_i(t), \mathbf{x}_j(t)) &= \begin{bmatrix} -\sigma\left(x_i(t) - x_j(t)\right) & 0 & 0 \end{bmatrix}^T,
\end{aligned} \tag{3.1}$$

где $i, j = 1, 2$. Здесь $\mathbf{x}_i(t) = \operatorname{col}\left(x_i(t), y_i(t), z_i(t)\right)$ — вектор состояния i-ой модели; $B = \operatorname{col}\left(1, 0, 0\right)$; $u(t)$ — управление; $\sigma \geqslant 0$ — сила связи между моделями; a, b, c, d, r, s — параметры, принимающие положительные значения; $x_{\mathrm{rest}\,i}$, $i = 1, 2$ — параметры, значения которых отрицательны.

Далее будем предполагать, что параметры моделей (3.1) неизвестны и не могут быть использованы при построении управления $u(t)$. Следовательно, управление $u(t)$ должно настраиваться самостоятельно, т. е. быть адаптивным.

Управляемую синхронизацию будем назвать адаптивной синхронизацией, если она обеспечивается управляющим воздействием $u(t)$, которое, в свою очередь, является адаптивным.

Остановимся подробнее на постановке задачи адаптивной координатной синхронизации моделей (3.1). Для этого введем в рассмотрение некоторые обозначения, которые в дальнейшем будут использованы в ходе решения

данной задачи:

$$\delta(t) = \mathbf{x}_1(t) - \mathbf{x}_2(t), \quad \Delta_{\text{rest}} = x_{\text{rest}\,1} - x_{\text{rest}\,2},$$
$$\psi_1(t) = x_1^2(t) + x_1(t)x_2(t) + x_2^2(t), \quad \psi_2(t) = x_1(t) + x_2(t),$$

где $\delta(t) = \operatorname{col}(\delta_x(t), \delta_y(t), \delta_z(t))$ — вектор отклонений между координатами состояний систем (2.1).

Основываясь на приведенных выше обозначениях, сформулируем задачу адаптивной координатной синхронизации моделей (3.1) в виде следующей цели управления:

$$\lim_{t\to\infty} \delta_x(t) = 0, \quad \lim_{t\to\infty} \delta_y(t) = 0, \quad \lim_{t\to\infty} \delta_z(t) = -s\Delta_{\text{rest}}. \tag{3.2}$$

Таким образом, решение задачи адаптивной координатной синхронизации сводится к поиску такого адаптивного управления $u(t)$, которое бы обеспечивало достижение цели управления (3.2).

3.2 Основной результат

Для решения поставленной задачи, в динамической сети (3.1) последовательно вычтем уравнения второй модели из уравнений первой и запишем следующую систему нелинейных дифференциальных уравнений:

$$\dot{\delta}(t) = F(\delta(t), \psi(t)) + 2\varphi(\mathbf{x}_1(t), \mathbf{x}_2(t)) + Bu(t),$$
$$F(\delta(t), \psi(t)) = \begin{bmatrix} -\left(a\psi_1(t) - b\psi_2(t)\right)\delta_x(t) + \delta_y(t) - \delta_z(t) \\ -d\psi_2(t)\delta_x(t) - \delta_y(t) \\ r\left(s\delta_x(t) - \delta_z(t) - s\Delta_{\text{rest}}\right) \end{bmatrix}. \tag{3.3}$$

Здесь $\psi(t) = \operatorname{col}(\psi_1(t), \psi_2(t))$ — вектор-функция, описывающая нелинейную часть системы.

Перейдем к координатам $e_x(t) = \delta_x(t)$, $e_y(t) = \delta_y(t)$, $e_z(t) = \delta_z(t) + s\Delta_{\text{rest}}$. Тогда цель управления (3.2) примет вид

$$\lim_{t\to\infty} e_x(t) = 0, \quad \lim_{t\to\infty} e_y(t) = 0, \quad \lim_{t\to\infty} e_z(t) = 0. \tag{3.4}$$

а система (3.3) запишется как

$$\dot{e}(t) = F(e(t), \psi(t)) + 2\varphi(\mathbf{x}_1(t), \mathbf{x}_2(t)) + Bu(t),$$
$$F(e(t), \psi(t)) = \begin{bmatrix} -\left(a\psi_1(t) - b\psi_2(t)\right) e_x(t) + e_y(t) - e_z(t) + s\Delta_{\text{rest}} \\ -d\psi_2(t)e_x(t) - e_y(t) \\ r\big(se_x(t) - e_z(t)\big) \end{bmatrix}. \tag{3.5}$$

Здесь $e(t) = \operatorname{col}\left(e_x(t), e_y(t), e_z(t)\right)$ — вектор состояния системы (3.5).

Опираясь на цель управления (3.4), выполним переход от задачи адаптивного управления синхронизацией в системах (3.1) к задаче поиска адаптивного управления $u(t)$, гарантирующего устойчивость нулевого решения системы (3.5). Для поиска такого управления рассмотрим следующий функционал:

$$Q\left(e(t)\right) = \frac{1}{2}\left(e_x^2(t) + e_y^2(t) + \frac{1}{rs}e_z^2(t)\right). \tag{3.6}$$

Тогда производная $\dot{Q}(e(t))$ вдоль траекторий системы (3.5) примет следующий вид:

$$\begin{aligned} \dot{Q}(e(t)) = -(a\psi_1(t) - b\psi_2(t) + 2\sigma)e_x^2(t) - e_y^2(t) - \frac{1}{s}e_z^2(t) + \\ + (1 - d\psi_2(t))e_x(t)e_y(t) + s\Delta_{\text{rest}}e_x(t) + u(t)e_x(t). \end{aligned} \tag{3.7}$$

Введем следующий закон управления:

$$u(t) = -(\gamma_0 + \theta_1(t)\psi_2(t))\delta_x(t) + \theta_2(t)\psi_2(t)\delta_y(t) + \theta_3(t), \tag{3.8}$$

где γ_0 — коэффициент усиления; $\theta_1(t)$, $\theta_2(t)$ и $\theta_3(t)$ — настраиваемые параметры. В новых координатах $e_x(t)$, $e_y(t)$ и $e_z(t)$, управление (3.8) примет вид:

$$u(t) = -(\gamma_0 + \theta_1(t)\psi_2(t))e_x(t) + \theta_2(t)\psi_2(t)e_y(t) + \theta_3(t). \tag{3.9}$$

Подставляя закон управления (3.9) в производную (3.7), получаем следу-

ющее выражение:

$$\begin{aligned}\dot{Q}(e(t)) = &-(\gamma_0 + a\psi_1(t) + 2\sigma)e_x^2(t) + e_x(t)e_y(t) - e_y^2(t) - \frac{1}{s}e_z^2(t) + \\ &+ (\theta_1(t) + b)\psi_2(t)e_x^2(t) + (\theta_2(t) - d)\psi_2(t)e_x(t)e_y(t) + \\ &+ (\theta_3(t) + s\Delta_{\text{rest}}).\end{aligned} \tag{3.10}$$

Теперь полученное выражение (3.10) будем оценивать сверху. Заметим, что $\psi_1(t) \geqslant 0$ и $\frac{r}{2}Q(e(t)) \leqslant \frac{1}{2}\left(e_x^2(t) + e_y^2(t) + e_z^2(t)/s\right)$ $\forall t \geqslant 0$. А значит имеет место неравенство (3.11).

$$\begin{aligned}\dot{Q}(e(t)) \leqslant &-\frac{r}{2}Q(e(t)) + e^T(t)We(t) + (\theta_1(t) + b)\psi_2(t)e_x^2(t) + \\ &+ (\theta_2(t) - d)\psi_2(t)e_x(t)e_y(t) + (\theta_3(t) + s\Delta_{\text{rest}}),\end{aligned} \tag{3.11}$$

где

$$W = \begin{bmatrix} -(\gamma_0 + 2\sigma - 1/2) & 1/2 & 0 \\ 1/2 & -1/2 & 0 \\ 0 & 0 & -1/2s \end{bmatrix}.$$

Теперь найдем такие γ_0 и σ, при которых квадратичная форма $e^T(t)We(t)$ в неравенстве (3.11) является отрицательно определенной. Используя критерий Сильвестра, получим следующее условие:

$$\gamma_0 + 2\sigma - 1 > 0. \tag{3.12}$$

Если условие (3.12) выполнено, то неравенство (3.11) примет вид:

$$\dot{Q}(e(t)) \leqslant -\frac{r}{2}Q(e(t)) + \omega\left(e(t), \theta(t)\right), \tag{3.13}$$

где $\theta(t) = \text{col}\left(\theta_1(t), \theta_2(t), \theta_3(t)\right)$ и

$$\begin{aligned}\omega\left(e(t), \theta(t)\right) = &(\theta_1(t) + b)\psi_2(t)e_x^2(t) + \\ &+ (\theta_2(t) - d)\psi_2(t)e_x(t)e_y(t) + (\theta_3(t) + s\Delta_{\text{rest}}).\end{aligned}$$

Настройку параметров $\theta_1(t)$, $\theta_2(t)$ и $\theta_3(t)$ будем осуществлять при помощи метода скоростного градиента. Для этого сначала найдем градиент

функции $\omega\left(e(t), \theta(t)\right)$ по переменным вектора $\theta(t)$:

$$\nabla_\theta \omega\left(e(t), \theta(t)\right) = \begin{bmatrix} \psi_2(t) e_x^2(t) & \psi_2(t) e_x(t) e_y(t) & e_x(t) \end{bmatrix}^T, \tag{3.14}$$

а затем зададим алгоритм изменения $\theta(t)$ в форме следующего дифференциального уравнения:

$$\dot{\theta}(t) = -\gamma \nabla_\theta \omega\left(e(t), \theta(t)\right), \tag{3.15}$$

где γ — коэффициент усиления. Алгоритм настройки параметров $\theta_1(t)$, $\theta_2(t)$ и $\theta_3(t)$ обеспечивает неположительность функции $\omega\left(e(t), \theta(t)\right)$, а значит правую часть неравенства (3.13) можно оценить сверху:

$$\dot{Q}(e(t)) \leqslant -\frac{r}{2} Q(e(t)). \tag{3.16}$$

Наконец, применяя известные теоремы сравнения дифференциальных неравенств [9] к неравенству (3.16) и устремляя t к бесконечности, получаем предел:

$$\lim_{t \to \infty} Q(e(t)) = 0. \tag{3.17}$$

Из последнего выражения следует, что цель управления (3.2) достигается, что, в свою очередь, свидетельствует о достижении синхронизации между связанными системами (3.1). Полученный результат сформулируем в виде следующей теоремы.

Теорема 3.1. *Пусть условие* (3.12) *выполнено. Тогда закон управления* $u(t)$ *в форме* (3.8) *гарантирует достижение цели управления* (3.2) $\forall \mathbf{x}_1(0), \mathbf{x}_2(0)$ *и* $\forall \theta(0)$ *систем* (3.1), (3.15).

Следовательно, теорема 3.1 является достаточным условием для достижения синхронизации связанных систем (3.1). Данный результат был опубликован в статье [34].

3.3 Компьютерное моделирование

Проведем компьютерное моделирование с целью подтверждения результатов, полученных в разделе 3.2. Для этого предварительно определим параметры моделирования для рассматриваемой динамической сети (3.1).

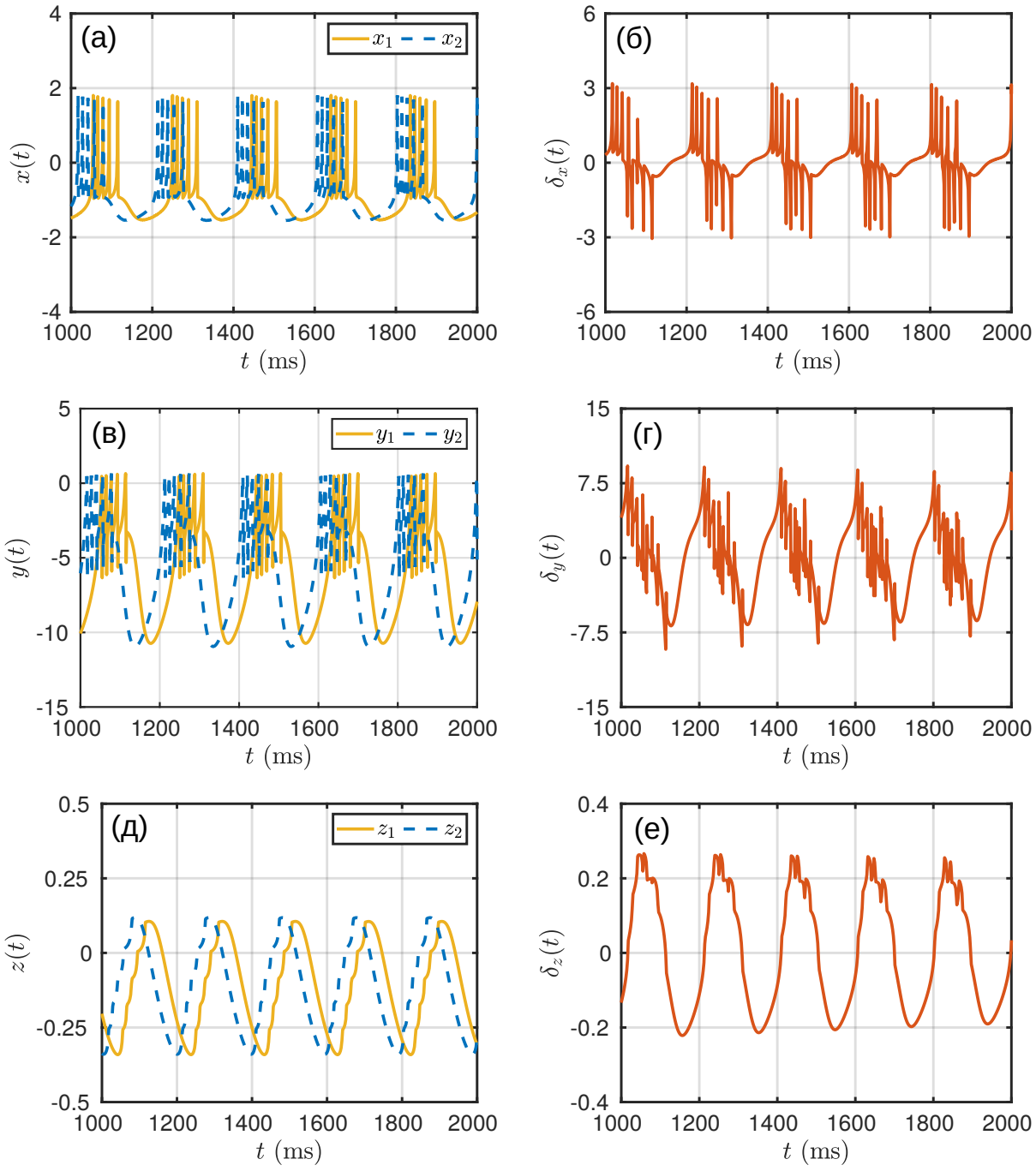

Рис. 3.1: Поведение двух связанных неидентичных моделей Хиндмарш-Роуз без управления: (а) динамика переменных $x_1(t)$ и $x_2(t)$; (б) разница между переменными $x_1(t)$ и $x_2(t)$; (в) динамика переменных y_1 и y_2; (г) разница между переменными y_1 и y_2; (д) динамика переменных z_1 и z_2; (е) разница между переменными z_1 и z_2

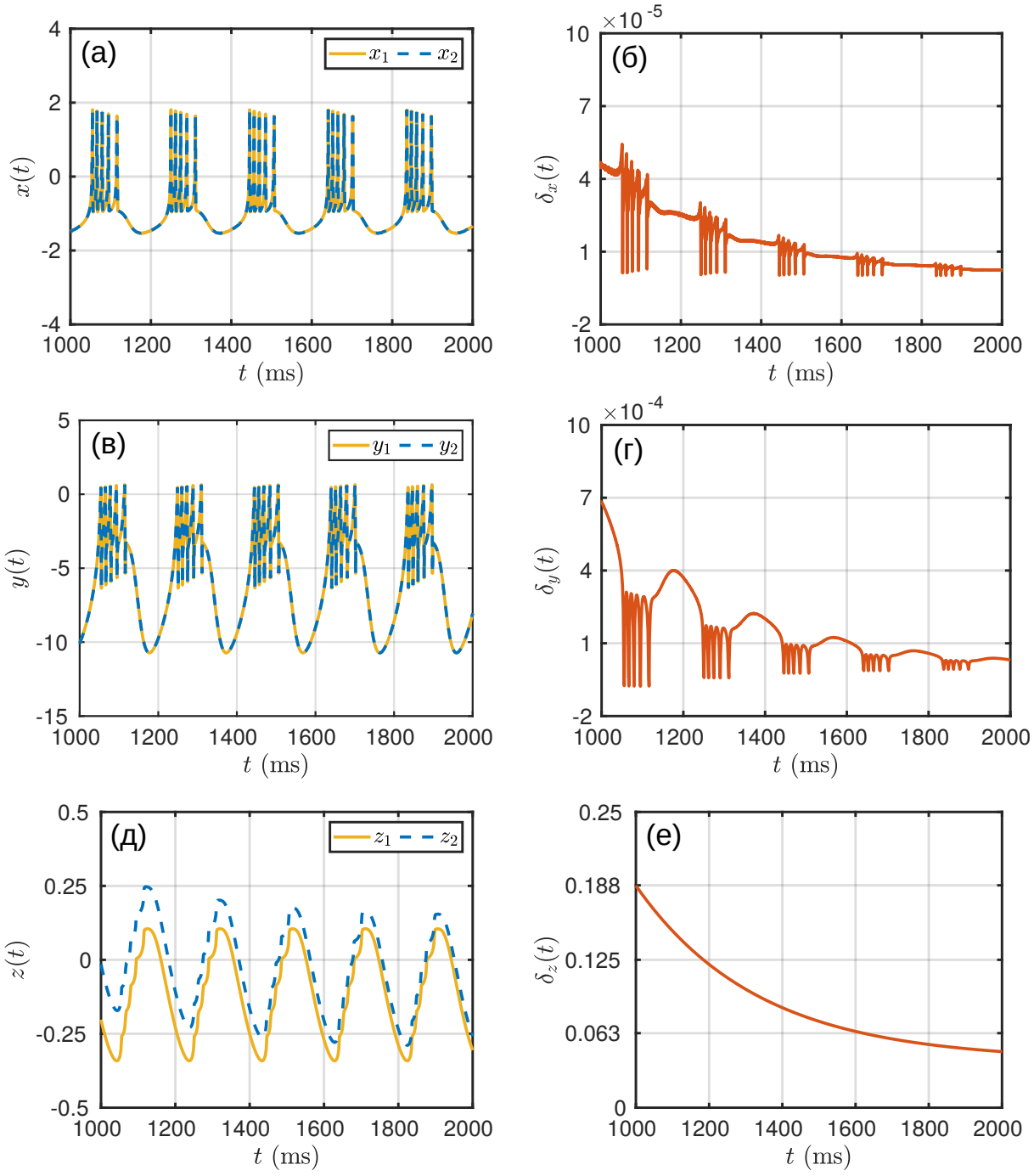

Рис. 3.2: Адаптивная синхронизация двух связанных неидентичных моделей Хиндмарш-Роуз: (а) динамика переменных $x_1(t)$ и $x_2(t)$; (б) разница между переменными $x_1(t)$ и $x_2(t)$; (в) динамика переменных y_1 и y_2; (г) разница между переменными y_1 и y_2; (д) динамика переменных z_1 и z_2; (е) разница между переменными z_1 и z_2

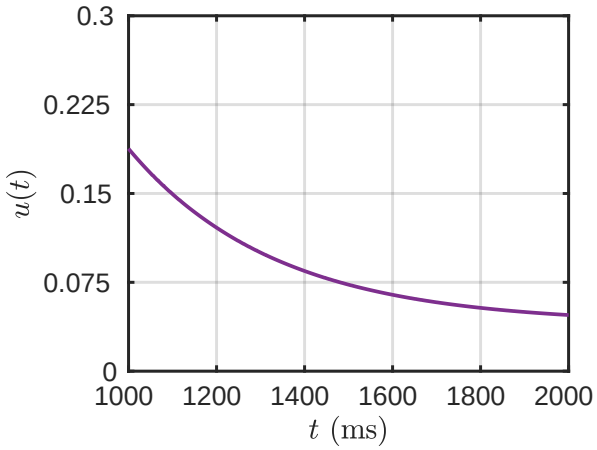

Рис. 3.3: Динамика закона управления $u(t)$

Значения данных параметров будут следующие:

$$a = 1, \quad b = 3, \quad c = 1, \quad d = 5, \quad r = 3 \cdot 10^{-3}, \quad s = 4,$$
$$x_{\text{rest}\,1} = -1, \quad x_{\text{rest}\,2} = -0.99, \quad \sigma = 1 \cdot 10^{-3}, \quad \gamma_0 = 5, \quad \gamma = 10.$$

Теперь перейдем непосредственно к результатам моделирования. Для начала рассмотрим случай, когда на сеть (3.1) не оказывается никакого управляющего воздействия (рис. 3.1). Можно видеть, что в данном случае узлы динамической сети ведут себя асинхронно и в их динамике наблюдается некоторый временной сдвиг друг относительно друга.

Далее рассмотрим случай, когда динамическая сеть (3.1) находится под управлением (3.8). Поскольку значения параметров γ_0 и σ удовлетворяют условию теоремы 3.1, то закон управления (3.8) гарантирует достижение синхронизации между узлами сети (3.1), что и проиллюстрировано на рис. 3.2. Динамика же самого закона управления изображена на рис. 3.3.

Таким образом, можно заключить, что результаты проведенного компьютерного моделирования полностью согласуются с полученными теоретическими результатами.

4 Адаптивная синхронизация гетерогенной спайковой нейронной сети Хиндмарш-Роуз

4.1 Постановка задачи

Рассмотрим гетерогенную динамическую сеть, состоящую из N связанных моделей Хиндмарш-Роуз:

$$\begin{aligned}
\dot{\mathbf{x}}_i(t) &= f_i(\mathbf{x}_i(t)) + \sum_{j=1}^{N} \alpha_{i\,j}\varphi(\mathbf{x}_i(t), \mathbf{x}_j(t)) + Bu_i(t), \\
f_i(\mathbf{x}_i(t)) &= \begin{bmatrix} y_i(t) - ax_i^3(t) + bx_i^2(t) - z_i(t) \\ c - dx_i^2(t) - y_i(t) \\ r\left[s(x_i(t) - x_{\mathrm{rest}\,i}) - z_i(t)\right] \end{bmatrix}, \\
\varphi(\mathbf{x}_i(t), \mathbf{x}_j(t)) &= \begin{bmatrix} -\sigma\,(x_i(t) - x_j(t)) & 0 & 0 \end{bmatrix}^T,
\end{aligned} \tag{4.1}$$

где $i, j = 1, \ldots, N$. Здесь $\mathbf{x}_i(t) = \mathrm{col}\,(x_i(t), y_i(t), z_i(t))$ — вектор состояния i-го узла в сети; $A = [\alpha_{i\,j}]$ — матрица смежности графа G, задающего топологию сети; $B = \mathrm{col}\,(1, 0, 0)$; $u_i(t)$ — управляющее воздействие, действующее на i-ый узел сети; $\sigma \geqslant 0$ — сила связи между узлами сети; a, b, c, d, r, s — параметры , принимающие положительные значения; $x_{\mathrm{rest}\,i}$, $i = 1, \ldots, N$ — параметры, значения которых отрицательны.

Теперь определим среднюю динамику сети (4.1). Для этого сложим все, соответствующие между собой, уравнения динамической сети и разделим на N. Кроме того, в силу произвольности выбора управляющего воздействия $u_i(t)$ для каждого узла, будем считать, что $\frac{1}{N}\sum_{j=1}^{N} u_i(t) = 0$. Таким образом, получим следующую систему дифференциальных уравнений:

$$\begin{aligned}
\dot{\bar{\mathbf{x}}}(t) &= f_{\mathrm{m}}(\bar{\mathbf{x}}(t), \psi_{\mathrm{m}}(t)), \\
f_{\mathrm{m}}(\bar{\mathbf{x}}(t), \psi_{\mathrm{m}}(t)) &= \begin{bmatrix} \bar{y}(t) - a\psi_{1\,\mathrm{m}}(t) + b\psi_{2\,\mathrm{m}}(t) - \bar{z}(t) \\ c - d\psi_{2\,\mathrm{m}}(t) - \bar{y}(t) \\ r\left[s(\bar{x}(t) - \bar{x}_{\mathrm{rest}}) - \bar{z}(t)\right] \end{bmatrix},
\end{aligned} \tag{4.2}$$

где

$$\bar{x}(t) = \frac{1}{N}\sum_{j=1}^{N} x_j(t), \quad \bar{y}(t) = \frac{1}{N}\sum_{j=1}^{N} y_j(t), \quad \bar{z}(t) = \frac{1}{N}\sum_{j=1}^{N} z_j(t),$$

$$\bar{x}_{\mathrm{rest}} = \frac{1}{N}\sum_{j=1}^{N} x_{\mathrm{rest}\,j}, \quad \psi_{1\,\mathrm{m}}(t) = \frac{1}{N}\sum_{j=1}^{N} x_j^3(t), \quad \psi_{2\,\mathrm{m}}(t) = \frac{1}{N}\sum_{j=1}^{N} x_j^2(t).$$

Здесь $\bar{\mathbf{x}}(t) = \mathrm{col}\,(\bar{x}(t), \bar{y}(t), \bar{z}(t))$ — вектор состояния системы средней динамики; $\psi_{\mathrm{m}}(t) = \mathrm{col}\,(\psi_{1\,\mathrm{m}}(t), \psi_{2\,\mathrm{m}}(t))$ — вектор-функция, описывающая нелинейную часть системы средней динамики.

Далее сформулируем некоторые предположения относительно рассматриваемой динамической сети. Во-первых, будем предполагать, что граф G, определяющий топологию сети (4.1), является неориентированным. Во-вторых, предполагается, что значения параметров динамической сети (4.1) неизвестны и не могут быть использованы при построении законов управления $u_i(t)$, $i = 1,\ldots,N$. Следовательно, каждый закон управления $u_i(t)$ должен настраиваться самостоятельно, т. е. быть адаптивным. И наконец, будем предполагать, что значения всех параметров $x_{\mathrm{rest}\,i}$ принадлежат некоторому интервалу, т. е. $|x_{\mathrm{rest}\,i} - x_{\mathrm{rest}\,j}| \leqslant \Delta$, $i, j = 1,\ldots,N$.

Управляемую синхронизацию будем назвать адаптивной синхронизацией, если она обеспечивается управляющим воздействием $u(t)$, которое, в свою очередь, является адаптивным.

Остановимся подробнее на постановке задачи адаптивной координатной синхронизации динамической сети (4.1). Для этого введем в рассмотрение некоторые обозначения, которые в дальнейшем будут использованы в ходе решения данной задачи:

$$\delta_i(t) = \mathbf{x}_i(t) - \bar{\mathbf{x}}(t), \quad \Delta_{\mathrm{rest}\,i} = x_{\mathrm{rest}\,i} - \bar{x}_{\mathrm{rest}\,i},$$
$$\psi_{1\,i}(t) = x_i^3(t) - \psi_{1\,\mathrm{m}}(t), \quad \psi_{2\,i}(t) = x_i^2(t) - \psi_{2\,\mathrm{m}}(t),$$

где $\delta_i(t) = \mathrm{col}\,(\delta_{x\,i}(t), \delta_{y\,i}(t), \delta_{z\,i}(t))$ — вектор отклонений координат узлов сети (4.1) от координат средней динамики, $i = 1,\ldots,N$.

Основываясь на приведенных выше обозначениях, сформулируем задачу адаптивной координатной синхронизации динамической (4.1) в виде

следующей цели управления:

$$\lim_{t\to\infty}\delta_{x\,i}(t)=0,\quad \lim_{t\to\infty}\delta_{y\,i}(t)=0,\quad \lim_{t\to\infty}\delta_{z\,i}(t)=-s\Delta_{\mathrm{rest}\,i},\quad i=1,\ldots,N. \tag{4.3}$$

Таким образом, решение задачи адаптивной координатной синхронизации сводится к поиску такого адаптивного управления $u(t)$, которое бы обеспечивало достижение цели управления (4.3).

4.2 Основной результат

Для решения поставленной задачи последовательно вычтем уравнения средней динамики (4.2) из уравнений динамической сети (4.1) и запишем следующую систему нелинейных дифференциальных уравнений:

$$\begin{aligned}\dot{\delta}_i(t)&=F_i(\delta_i(t),\psi_i(t))+\sum_{j=1}^{N}\varphi(\mathbf{x}_i(t),\mathbf{x}_j(t))+Bu_i(t),\\ F_i(\delta_i(t),\psi_i(t))&=\begin{bmatrix}\delta_{y,i}(t)-a\psi_{1\,i}(t)+b\psi_{2\,i}(t)-\delta_{z\,i}(t)\\ -d\psi_{2\,i}(t)-\delta_{y\,i}(t)\\ r\big(s\delta_{x\,i}(t)-\delta_{z\,i}(t)-s\Delta_{\mathrm{rest}\,i}\big)\end{bmatrix},\end{aligned} \tag{4.4}$$

где $i=1,\ldots,N$. Здесь $\psi_i(t)=\mathrm{col}\,(\psi_{1\,i}(t),\psi_{2\,i}(t))$ — вектор-функция, описывающая нелинейную часть i-го узла системы (4.4).

Перейдем к новой системе координат $e_{x\,i}(t)=\delta_{x\,i}(t)$, $e_{y\,i}(t)=\delta_{y\,i}(t)$, $e_{z\,i}(t)=\delta_{z\,i}(t)+s\Delta_{\mathrm{rest}\,i}$. Тогда цель управления (4.3) примет вид

$$\lim_{t\to\infty}e_{x\,i}(t)=0,\quad \lim_{t\to\infty}e_{y\,i}(t)=0,\quad \lim_{t\to\infty}e_{z\,i}(t)=0,\quad i=1,\ldots,N. \tag{4.5}$$

а система (4.4) запишется как

$$\begin{aligned}\dot{e}_i(t)&=F_i(e_i(t),\psi_i(t))+\sum_{j=1}^{N}\varphi(\mathbf{x}_i(t),\mathbf{x}_j(t))+Bu_i(t),\\ F_i(e_i(t),\psi_i(t))&=\begin{bmatrix}e_{y\,i}(t)-a\psi_{1\,i}(t)+b\psi_{2\,i}(t)-e_{z\,i}(t)+s\Delta_{\mathrm{rest}\,i}\\ -d\psi_{2\,i}(t)-e_{y\,i}(t)\\ r(se_{x\,i}(t)-e_{z\,i}(t))\end{bmatrix},\end{aligned} \tag{4.6}$$

где $i=1,\ldots,N$. Здесь $e_i(t)=\mathrm{col}\,(e_{x\,i}(t),e_{y\,i}(t),e_{z\,i}(t))$ — вектор состояния

i-го узла в системе (4.6).

Опираясь на цель управления (4.5), выполним переход от задачи адаптивного управления синхронизацией в сети (4.1) к задаче поиска адаптивных законов управления $u_i(t)$, $i = 1, \ldots, N$, гарантирующих устойчивость нулевого решения для каждого узла системы (4.6). Для поиска таких законов управления рассмотрим следующий функционал:

$$Q\left(\mathbf{e}(t)\right) = \frac{1}{2}\sum_{i=1}^{N}\left(e_{x\,i}^2(t) + e_{y\,i}^2(t) + \frac{1}{rs}e_{z\,i}^2(t)\right), \tag{4.7}$$

где $\mathbf{e}(t) = \operatorname{col}\left(e_1^T(t), \ldots, e_N^T(t)\right)$. Тогда производная $\dot{Q}\left(\mathbf{e}(t)\right)$ вдоль траекторий системы (4.6) примет следующий вид:

$$\begin{aligned}\dot{Q}(\mathbf{e}(t)) = \sum_{i=1}^{N}\Bigg[\Bigg(&e_{y\,i}(t) - a\psi_{1\,i}(t) + b\psi_{2\,i}(t) - e_{z\,i}(t) + s\Delta_{\mathrm{rest}\,i} + \\ &+ \sigma\sum_{j=1}^{N}\alpha_{i\,j}(x_j(t) - x_i(t)) + u_i(t)\Bigg)e_{x\,i}(t) - \\ &- \left(d\psi_{2\,i}(t) + e_{y\,i}(t)\right)e_{y\,i}(t) + \left(e_{x\,i}(t) - \frac{1}{s}e_{z\,i}(t)\right)e_{z\,i}(t)\Bigg]\end{aligned} \tag{4.8}$$

Выполним некоторые преобразования над слагаемыми, находящимися в правой части равенства (4.8). Во-первых, основываясь на определении матрицы Лапласа, представим сумму $\sum_{i=1}^{N} e_{x\,i}(t)\sum_{j=1}^{N}\alpha_{i\,j}(x_j(t) - x_i(t))$ как

$$\begin{aligned}&\sum_{i=1}^{N}e_{x\,i}(t)\sum_{j=1}^{N}\alpha_{i\,j}(x_j(t) - x_i(t)) = \sum_{i=1}^{N}(x_i(t) - \bar{x}(t))\sum_{j=1}^{N}\alpha_{i\,j}(x_j(t) - x_i(t)) = \\ &= \sum_{i=1}^{N}\left[x_i(t)\sum_{j=1}^{N}\alpha_{i\,j}(x_j(t) - x_i(t)) - \bar{x}(t)\sum_{j=1}^{N}\alpha_{i\,j}(x_j(t) - x_i(t))\right] = \\ &= -\sum_{i=1}^{N}n_i x_i^2(t) + \sum_{i,j=1}^{N}\alpha_{i\,j}x_i(t)x_j(t) = -x^T(t)Lx(t),\end{aligned} \tag{4.9}$$

где $x(t) = \operatorname{col}\left(x_1(t), \ldots, x_2(t)\right)$; L — матрица Лапласа; n_i — степень i-ой вершины графа G. Во-вторых, сумму $\sum_{i=1}^{N}\psi_{2\,i}(t)e_{x\,i}(t)$ можно представить

в следующем виде:

$$\begin{aligned}
&\sum_{i=1}^{N}\psi_{2\,i}(t)e_{x\,i}(t)=\sum_{i=1}^{N}(x_i^2(t)-\psi_{2\,\mathrm{m}}(t))(x_i(t)-\bar{x}(t))=\\
&=\sum_{i=1}^{N}x_i^2(t)(x_i(t)-\bar{x}(t))-\psi_{2\,\mathrm{m}}(t)\underbrace{\sum_{i=1}^{N}(x_i(t)-\bar{x}(t))}_{=0\text{ по определению}}=\\
&=\sum_{i=1}^{N}x_i^2(t)(x_i(t)-\bar{x}(t))-\bar{x}(t)\sum_{i=1}^{N}(x_i(t)-\bar{x}(t))=\\
&=\sum_{i=1}^{N}(x_i^2(t)-\bar{x}^2(t))(x_i(t)-\bar{x}(t))=\sum_{i=1}^{N}(x_i(t)+\bar{x}(t))e_{x\,i}^2(t).
\end{aligned}\tag{4.10}$$

Наконец, выполняя преобразования над суммой $\sum_{i=1}^{N}\psi_{2\,i}(t)e_{y\,i}(t)$, получим:

$$\begin{aligned}
&\sum_{i=1}^{N}\psi_{2\,i}(t)e_{y\,i}(t)=\sum_{i=1}^{N}(x_i^2(t)-\psi_{2\,\mathrm{m}}(t))(y_i(t)-\bar{y}(t))=\\
&=\sum_{i=1}^{N}x_i^2(t)(y_i(t)-\bar{y}(t))-\psi_{2\,\mathrm{m}}(t)\underbrace{\sum_{i=1}^{N}(y_i(t)-\bar{y}(t))}_{=0\text{ по определению}}=\\
&=\sum_{i=1}^{N}x_i^2(t)(y_i(t)-\bar{y}(t))-\bar{x}^2(t)\sum_{i=1}^{N}(x_i(t)-\bar{x}(t))=\\
&=\sum_{i=1}^{N}(x_i^2(t)-\bar{x}^2(t))(y_i(t)-\bar{y}(t))=\sum_{i=1}^{N}(y_i(t)+\bar{y}(t))e_{x\,i}(t)e_{y\,i}(t).
\end{aligned}\tag{4.11}$$

Теперь перепишем выражение (4.8) с учетом полученных выше преобразований. Получим следующее:

$$\begin{aligned}
\dot{Q}(\mathbf{e}(t))=\sum_{i=1}^{N}\Bigg[&\Big(e_{y\,i}(t)-a\psi_{1\,i}(t)+b(x_i(t)+\bar{x}(t))e_{x\,i}(t)-e_{z\,i}(t)+\\
&+s\Delta_{\mathrm{rest}\,i}+u_i(t)\Big)e_{x\,i}(t)-\Big(d(x_i(t)+\bar{x}(t))e_{x\,i}+e_{y\,i}(t)\Big)e_{y\,i}(t)+\\
&+\Big(e_{x\,i}(t)-\frac{1}{s}e_{z\,i}(t)\Big)e_{z\,i}(t)\Bigg]-\sigma x^T(t)Lx(t).
\end{aligned}\tag{4.12}$$

Введем управление в виде следующих управляющих воздействий:

$$\begin{aligned} u_i(t) = &-\big[\gamma_0 - \theta_{1\,i}(t)\big(x_i(t) + \bar{x}(t)\big)\big]\delta_{x\,i}(t) + \\ &+ \theta_{2\,i}(t)\big(x_i(t) + \bar{x}(t)\big)\delta_{y\,i}(t) + \theta_{3\,i}(t), \quad i = 1, \ldots, N, \end{aligned} \tag{4.13}$$

где γ_0 — коэффициент усиления; $\theta_{1\,i}(t)$, $\theta_{2\,i}(t)$ и $\theta_{3\,i}(t)$, $i = 1, \ldots, N$ — настраиваемые параметры. В новой системе координат $e_{x\,i}(t)$, $e_{y\,i}(t)$ и $e_{z\,i}(t)$, управляющие воздействия (4.13) примут вид:

$$\begin{aligned} u_i(t) = &-\big[\gamma_0 - \theta_{1\,i}(t)\big(x_i(t) + \bar{x}(t)\big)\big]e_{x\,i}(t) + \\ &+ \theta_{2\,i}(t)\big(x_i(t) + \bar{x}(t)\big)e_{y\,i}(t) + \theta_{3\,i}(t), \quad i = 1, \ldots, N. \end{aligned} \tag{4.14}$$

Подставляя управляющие воздействия (4.14) в производную (4.12), получаем следующее выражение:

$$\begin{aligned} \dot{Q}(\mathbf{e}(t)) = &\sum_{i=1}^{N} \Bigg[- \bigg(\gamma_0 e_{x\,i}^2(t) + e_{x\,i}(t)e_{y\,i}(t) + \frac{1}{s}e_{z\,i}^2(t)\bigg) - \\ &- a\psi_{1\,i}(t)e_{x\,i}(t) + \omega(e_i(t), \theta_i(t))\Bigg] - \sigma x^T(t)Lx(t), \\ \omega(e_i(t), \theta_i(t)) = &(\theta_{1\,i}(t) + b)(x_i(t) + \bar{x})e_{x\,i}^2(t) + \\ &+ (\theta_{2\,i}(t) - d)(x_i(t) + \bar{x})e_{x\,i}(t)e_{y\,i}(t) + \\ &+ (\theta_{3\,i}(t) - s\Delta_{\mathrm{rest}\,i})e_{x\,i}(t), \end{aligned} \tag{4.15}$$

где $\theta_i(t) = \mathrm{col}\,(\theta_{1\,i}(t), \theta_{2\,i}(t), \theta_{3\,i}(t))$ и $i = 1, \ldots, N$.

Теперь правую часть уравнения (4.15) будем оценивать сверху. Во-первых, заметим, что $\frac{r}{2}Q(\mathbf{e}(t)) \leqslant \frac{1}{2}\sum_{i=1}^{N}\left(e_{x\,i}^2(t) + e_{y\,i}^2(t) + e_{z\,i}^2(t)/s\right)$ $\forall t \geqslant 0$, а значит имеет место неравенство:

$$\begin{aligned} \dot{Q}(\mathbf{e}(t)) \leqslant &-\frac{r}{2}Q(\mathbf{e}(t)) - \sigma x^T(t)Lx(t) + \\ &+ \sum_{i=1}^{N}\bigg(-a\psi_{1\,i}(t)e_{x\,i}(t) + e_i^T(t)We_i(t) + \omega(e_i(t), \theta_i(t))\bigg), \end{aligned} \tag{4.16}$$

где

$$W = \begin{bmatrix} -(\gamma_0 - 1/2) & 1/2 & 0 \\ 1/2 & -1/2 & 0 \\ 0 & 0 & -1/2s \end{bmatrix}.$$

Во-вторых, покажем, что сумма $-\sum_{i=1}^{N}\psi_{1\,i}(t)e_{x\,i}(t)$ не является положительной $\forall t \geqslant 0$. Для этого данную сумму запишем как

$$
\begin{aligned}
&-\sum_{i=1}^{N}\psi_{1\,i}(t)e_{x\,i}(t) = \sum_{i=1}^{N}(x_i^3(t) - \psi_{1\,\mathrm{m}}(t))(x_i(t) - \bar{x}(t)) = \\
&= -\sum_{i=1}^{N}x_i^3(t)(x_i(t) - \bar{x}(t)) + \psi_{1\,\mathrm{m}}(t)\underbrace{\sum_{i=1}^{N}(x_i(t) - \bar{x}(t))}_{=0\ \text{по определению}} = \\
&= -\sum_{i=1}^{N}x_i^3(t)(x_i(t) - \bar{x}(t)) + \bar{x}^3(t)\sum_{i=1}^{N}(x_i(t) - \bar{x}(t)) = \\
&= \sum_{i=1}^{N}(x_i(t) - \bar{x}(t))^2(x_i^2(t) + x_i(t)\bar{x}(t) + \bar{x}^2(t)).
\end{aligned}
\tag{4.17}
$$

Далее найдем такое $\mu > 0$, для которого верно неравенство

$$x_i^2(t) + x_i(t)\bar{x}(t) + \bar{x}^2(t) \geqslant \mu(x_i(t) - \bar{x}(t))^2.$$

Выполнив простейшие преобразования, получим

$$(1-\mu)x_i^2(t) + (1+2\mu)x_i(t)\bar{x}(t) + (1-\mu)\bar{x}^2(t) \geqslant 0.$$

Данное неравенство выполняется, когда соответствующее ему квадратное уравнение не имеет корней, т. е. когда дискриминант $(1+2\mu)^2 - 4(1-\mu)^2 < 0$. Это имеет место $\forall \mu \leqslant 1/4$. Таким образом, выбрав $\mu = 1/4$, приходим к следующему неравенству:

$$-\sum_{i=1}^{N}\psi_{1\,i}(t)e_{x\,i}(t) \leqslant -\frac{1}{4}\sum_{i=1}^{N}(x_i(t) - \bar{x}(t))^4.$$

Следовательно, рассматриваемая сумма $-\sum_{i=1}^{N}\psi_{1\,i}(t)e_{x\,i}(t)$ не является положительной $\forall t \geqslant 0$. А значит правую часть неравенства (4.16) можно

оценить сверху:

$$\dot{Q}(\mathbf{e}(t)) \leqslant -\frac{r}{2}Q(\mathbf{e}(t)) - \sigma x^T(t)Lx(t) + \\ + \sum_{i=1}^{N}\left(e_i^T(t)We_i(t) + \omega(e_i(t),\theta_i(t))\right). \tag{4.18}$$

В-третьих, покажем, что квадратичная форма $x^T(t)Lx(t)$ положительно полуопределена, т. е. $x^T(t)Lx(t) \geqslant 0$. В самом деле, поскольку L является матрицей Лапласа графа G, то L имеет нулевое собственное число λ_1. Кроме того, все собственные числа имеют неотрицательную вещественную часть, а значит могут быть упорядочены следующим образом:

$$0 = \lambda_1 \leqslant \lambda_2 \leqslant \ldots \leqslant \lambda_N.$$

Из приведенных выше отношений порядка для собственных чисел матрицы L следует положительная полуопределенность квадратичной формы $x^T(t)Lx(t)$, а значит имеет место следующая оценка:

$$\dot{Q}(\mathbf{e}(t)) \leqslant -\frac{r}{2}Q(\mathbf{e}(t)) + \sum_{i=1}^{N}\left(e_i^T(t)We_i(t) + \omega(e_i(t),\theta_i(t))\right). \tag{4.19}$$

И наконец, найдем такое γ_0, при котором сумма $\sum_{i=1}^{N} e_i^T(t)We_i(t)$ в неравенстве (4.19) отрицательно определена, а именно когда каждая i-я квадратичная форма $e_i^T(t)We_i(t)$ является отрицательно определенной. Используя критерий Сильвестра, получим следующее условие:

$$\gamma_0 - 1 > 0. \tag{4.20}$$

Если условие (4.20) выполнено, то неравенство (4.19) примет вид:

$$\dot{Q}(\mathbf{e}(t)) \leqslant -\frac{r}{2}Q(\mathbf{e}(t)) + \sum_{i=1}^{N}\omega(e_i(t),\theta_i(t)). \tag{4.21}$$

Настройку параметров $\theta_{1\,i}(t)$, $\theta_{2\,i}(t)$, $\theta_{3\,i}(t)$, $i = 1, \ldots, N$ будем осуществлять при помощи метода скоростного градиента. Для этого сначала найдем

градиент функции $\omega\left(e_i(t), \theta_i(t)\right)$ по переменным вектора $\theta_i(t)$:

$$\nabla_\theta \omega\left(e_i(t), \theta_i(t)\right) = \left(x_i(t) + \bar{x}(t)\right) \begin{bmatrix} e_{x\,i}^2(t) & e_{x\,i}(t) e_{y\,i}(t) & e_{x\,i}(t) \end{bmatrix}^T, \tag{4.22}$$

Далее зададим алгоритм изменения для каждого $\theta_i(t)$ в форме следующего дифференциального уравнения:

$$\dot{\theta}_i(t) = -\gamma_i \nabla_\theta \omega\left(e_i(t), \theta_i(t)\right), \tag{4.23}$$

где γ_i — коэффициент усиления в алгоритме настройки параметров i-го управляющего воздействия, $i = 1, \ldots, N$. Алгоритм настройки параметров $\theta_{1\,i}(t)$, $\theta_{2\,i}(t)$ и $\theta_{3\,i}(t)$ для управляющего воздействия $u_i(t)$ обеспечивает неположительность функции $\omega\left(e_i(t), \theta_i(t)\right)$, а значит правую часть неравенства (4.21) можно оценить сверху:

$$\dot{Q}(\mathbf{e}(t)) \leqslant -\frac{r}{2} Q(\mathbf{e}(t)). \tag{4.24}$$

Теперь, применяя известные теоремы сравнения дифференциальных неравенств [9] к неравенству (4.24) и устремляя t к бесконечности, получаем предел:

$$\lim_{t \to \infty} Q(e(t)) = 0. \tag{4.25}$$

Из последнего выражения следует, что цель управления (4.3) достигается, что, в свою очередь, свидетельствует о достижении синхронизации в сети (4.1). Полученный результат сформулируем в виде теоремы.

Теорема 4.1. *Пусть условие* (4.20) *выполнено. Тогда управляющие воздействия* $u_i(t)$, $i = 1, \ldots, N$ *в форме* (4.13) *гарантируют достижение цели управления* (4.3) $\forall \mathbf{x}_1(0), \ldots \mathbf{x}_N(0)$ *и* $\forall \theta_1(0), \ldots, \theta_N(0)$ *систем* (4.1), (4.23).

Следовательно, теорема 4.1 является достаточным условием для достижения синхронизации в гетерогенной динамической сети (4.1).

4.3 Компьютерное моделирование

Проведем компьютерное моделирование с целью подтверждения результатов, полученных в разделе 4.2. Для этого предварительно определим параметры моделирования для динамической сети (4.1). Значения данных

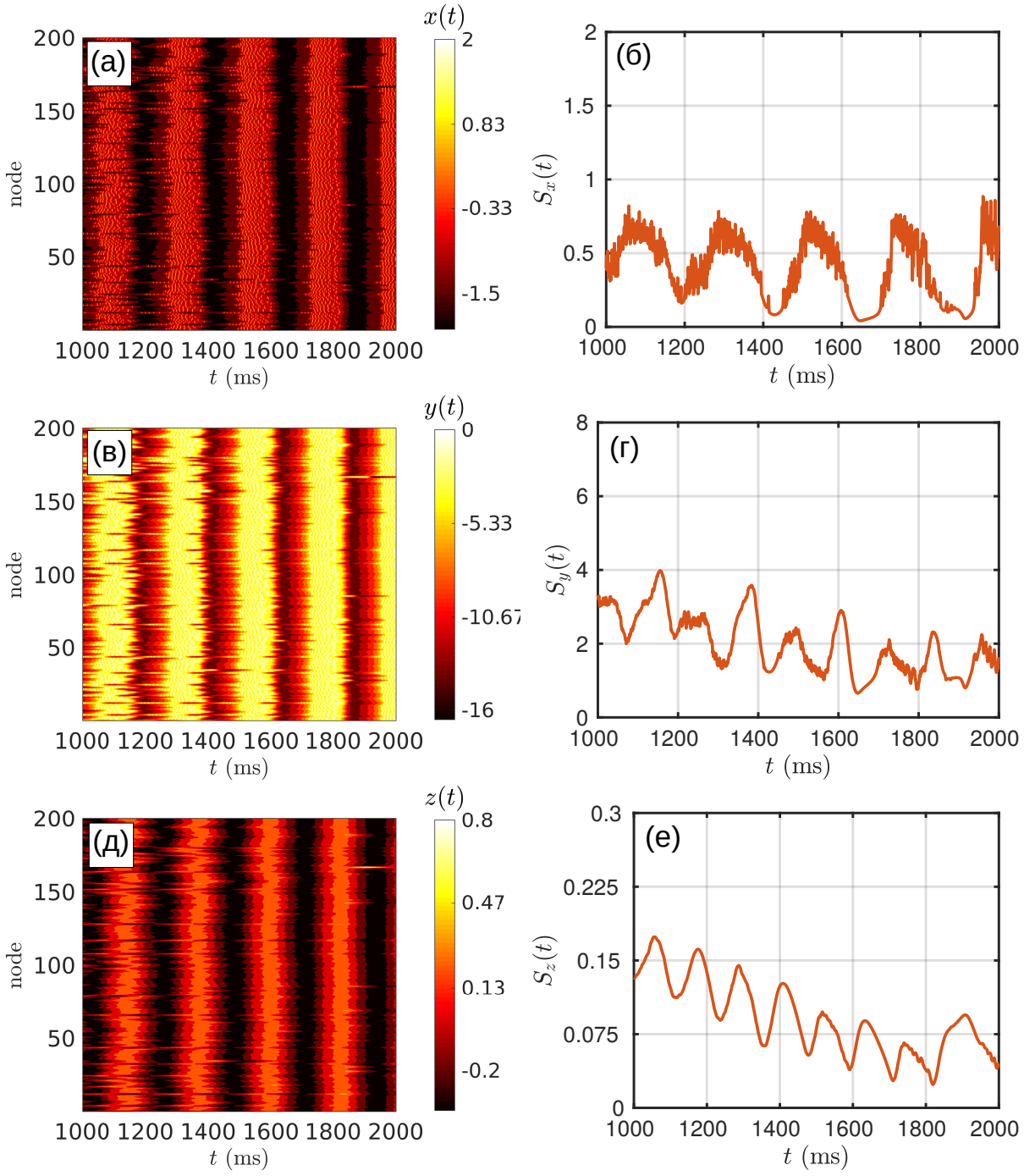

Рис. 4.1: Поведение гетерогенной динамической сети из 200 моделей Хиндмарш-Роуз без управления: (а), (в) и (д) динамика узлов сети по соответствующим переменным; (б), (г) и (е) среднеквадратическое отклонение динамики узлов сети от средней динамики по соответствующим переменным

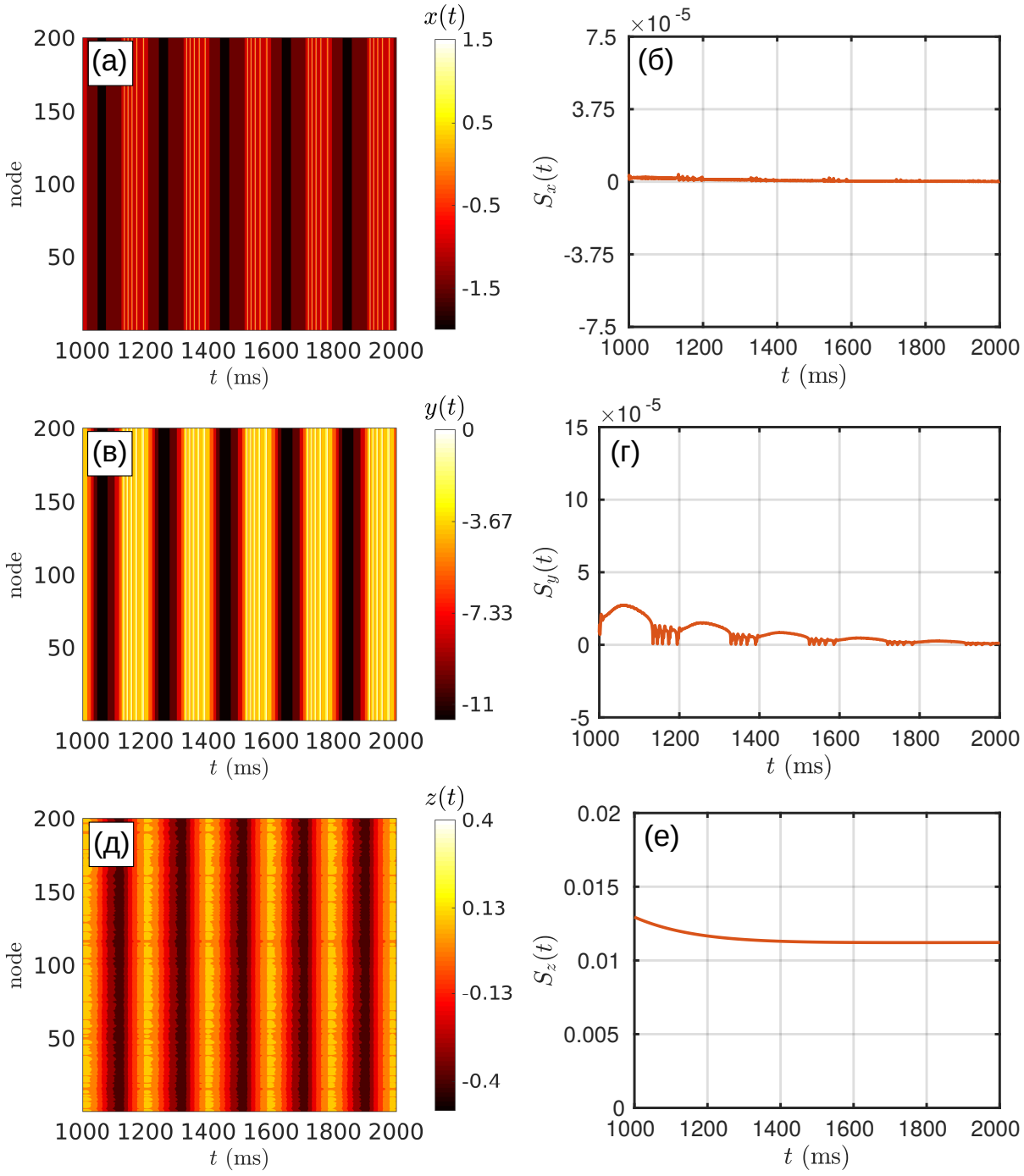

Рис. 4.2: Адаптивная синхронизация гетерогенной динамической сети из 200 моделей Хиндмарш-Роуз: (а), (в) и (д) динамика узлов сети по соответствующим переменным; (б), (г) и (е) среднеквадратическое отклонение динамики узлов сети от средней динамики по соответствующим переменным

параметров будут следующие:

$$a = 1,\ b = 3,\ c = 1,\ d = 5,\ r = 3 \cdot 10^{-3},\ s = 4,\ \sigma = 1 \cdot 10^{-3}, N = 200,$$
$$\gamma_0 = 5,\ \gamma_1 = \ldots = \gamma_N = 10,\ x_{\mathrm{rest}\,i} \sim U[-1, -0.99],\ i = 1, \ldots, N.$$

Теперь перейдем непосредственно к результатам моделирования. Для начала рассмотрим случай, когда на сеть (4.1) не оказывается никакого управляющего воздействия (рис. 4.1). Можно видеть, что в данном случае узлы динамической сети ведут себя асинхронно и в их динамике наблюдается некоторый временной сдвиг друг относительно друга.

Далее рассмотрим случай, когда динамическая сеть (4.1) находится под управлением (4.13). Поскольку значение параметра γ_0 удовлетворяет условию теоремы 4.1, то управляющие воздействия (4.13) гарантируют достижение синхронизации между узлами динамической сети (4.1), что и проиллюстрировано на рис. 4.2.

Таким образом, можно заключить, что результаты проведенного компьютерного моделирования полностью согласуются с полученными теоретическими результатами.

Заключение

В настоящей работе рассмотрена задача управления синхронизацией в спайковых нейронных сетях Хиндмарш-Роуз для трех случаев: при наличии ограниченных возмущений в сети; при отсутствии априорной информации о значениях параметров сети; при отсутствии априорной информации как о значениях параметров сети, так и о топологии сети. В первых двух случаях топология сети является фиксированной и представляет собой две связанные модели Хиндмарш-Роуз. В третьем случае топология сети является произвольной. Во всех трех случаях предполагается неидентичность узлов сети по параметрам, т. е. сети являются гетерогенными.

В ходе работы для каждого случая был предложен свой закон управления и математически доказана достижимость синхронизации в рассматриваемой сети, посредством предложенного закона. Для каждого из рассматриваемых в работе случаев было проведено компьютерное моделирование, результаты которого полностью соответствуют полученным теоретическим выводам.

Направлением дальнейших исследований, в первую очередь, является введение нелинейных связей между узлами сети, т. е. между нейронами. Эта необходимость вызвана, прежде всего, физиологическими особенностями механизма взаимодействия нервных клеток между собой. Кроме того, в дальнейшем необходимо разработать алгоритмы управления, обеспечивающие условия синхронизации сети Хиндмарш-Роуз даже при наличии временных задержек при передаче сигналов между нейронами.

Список литературы

1. Андриевский Б. Р., Бобцов А. А., Фрадков А. Л. Методы анализа и синтеза нелинейных систем управления. М.–Ижевск: Институт компьютерных исследований, 2018. 336 с.

2. Блехман И. И. Синхронизация в природе и технике. М.: Наука, 1981. 352 с.

3. Леонов Г. А. Теория управления. СПб.: Изд-во СПбГУ, 2006. 233 с.

4. Мирошник И. В., Никифоров В. О., Фрадков А. Л. Нелинейное и адаптивное управление сложными динамическими системами. СПб.: Наука, 2000. 549 с.

5. Проблемы сетевого управления / Н. О. Амелина [и др.] / под ред. д. т. н., проф. А. Л. Фрадкова. М.–Ижевск: Институт компьютерных исследований, 2015. 392 с.

6. Семенов Д. М. Управление синхронизацией двух связанных неидентичных систем Хиндмарш-Роуз. // Управление большими системами. 2018. № 75. С. 30–49.

7. Семенов Д. М. Управляемая синхронизация двух связанных нейронных моделей Хиндмарш-Роуз при наличии возмущений // XVIII научная школа «Нелинейные волны–2018»: тезисы докладов молодых ученых. Н. Новгород: Изд-во ИПФ РАН, 2018. С. 168–170.

8. Степаненко Д. А., Семенов Д. М., Плоткников С. А., Фрадков А. Л. Программный комплекс для реализации нейрообратной связи: модуль обработки сигнала ЭЭГ // Свидетельство о регистрации программы для ЭВМ. № 2019610034, 09.01.2019.

9. Тихонов А. Н., Васильева А. Б. Свешников А. Г. Дифференциальные уравнения. М.: Наука, 1985. 232 с.

10. Фрадков А. Л. Кибернетическая физика: принципы и примеры. СПб.: Наука, 2003. 208 с.

11. Фрадков А. Л. Схема скоростного градиента в задачах адаптации и управления // Автоматика и телемеханика. 1979. № 9. С. 90–101.

12. Чеботарев П. Ю., Агаев Р. П. Согласование характеристик в многоагентных системах и спектры лапласовских матриц орграфов // Автоматика и телемеханика. 2009. № 3. С. 136–151.

13. Buck J., Buck E. Mechanism of Rhythmic Synchronous Flashing of Fireflies: Fireflies of Southeast Asia may use anticipatory time measuring in synchronizing their flashing // Science. 1968. Vol. 159, Issue 3821. P. 1319–1327

14. Castanedo-Guerra I. T., Steur E., Nijmeijer H. Synchronization of coupled Hindmarsh-Rose neurons: effects of an exogenous parameter. IFAC PapersOnLine. 2016. Vol. 49, Issue 14. P. 84–89.

15. Chkhenkeli S. A. Direct Deep-Brain Stimulation: First Steps Towards the Feedback Control of Seizures // Epilepsy as a Dynamic Disease / ed. by J. Milton, P. Jung. 2003. P. 249–261

16. Fiedler M. Algebraic connectivity of graphs // Czechoslovak Mathematical Journal. 1973. Vol. 23. P. 298–305.

17. Fradkov A. L., Pogromsky A. Yu. Speed gradient control of chaotic continuous-time systems // EEE Transactions on Circuits and Systems I: Fundamental Theory and Applications. 1996. Vol. 43, Issue 11. P. 907–913.

18. Gerstner W., Kistler W. M. Spiking Neuron Models: Single neurons, Populations, Plasticity. Cambridge University Press, 2002.

19. Hindmarsh J. L., Rose R. M. A model of the nerve impulse using two first-order differential equations // Nature. 1982. Vol. 296, Issue 5853. P. 162–164.

20. Hindmarsh J. L., Rose R. M. A Model of Neuronal Bursting Using Three Coupled First Order Differential Equations // Proceedings of the Royal Society of London. Series B, Biological Sciences. 1984. Vol. 221, Issue 1222. P. 87–102.

21. Hodgkin A. L., Huxley A. F. A Quantitative Description Of Membrane Current And Its Application To Conduction And Excitation In Nerve // The Journal of Physiology. 1952. Vol. 117, Issue 4. P. 500–544.

22. Izhikevich E. M. Dynamical Systems in Neuroscience: The Geometry of Excitability and Bursting. MIT press, 2007.

23. Khalil H. K. Nonlinear Systems. Third Edition. Prentice Hall, 2002.

24. Olfati-Saber R., Murray R. M. Consensus problems in networks of agents with switching topology and time-delays // IEEE Transactions on Automatic Control. 2004. Vol. 49, Issue 9. P. 1520–1533.

25. Panteley E., Loria A. Synchronization and Dynamic Consensus of Heterogeneous Networked Systems // IEEE Transactions on Automatic Control. 2017. Vol. 62, Issue 8. P. 3758–3773.

26. Peskin C. S. Mathematical aspects of heart physiology. Courant Inst. Math, New York University, 1975.

27. Pikovsky A., Rosenblum M., Kurths J. Synchronization: A Universal Concept In Nonlinear Sciences. Cambridge University Press, 2003.

28. Plotnikov S. A., Fradkov A. L. On synchronization in heterogeneous FitzHugh-Nagumo networks // Chaos, Solitons & Fractals. 2019. Vol. 121. P. 85–91.

29. Plotnikov S. A., Fradkov A. L. On Synchronization in FitzHugh-Nagumo Networks with Small Delays // 2018 European Control Conference (ECC). IEEE, 2018. P. 2052–2056.

30. Plotnikov S. A., Lehnert J., Fradkov A. L., and Schöll E. Synchronization in heterogeneous FitzHugh-Nagumo networks with hierachical architecture // Physical Review E. 2016. Vol. 94, Issue 1. 012203.

31. Plotnikov S. Controlled synchronization in two FitzHugh-Nagumo systems with slowly-varying delays // Cybernetics and Physics. 2015. Vol. 4, Issue 1. P. 21–25.

32. Ren. W., Beard R. W. Distributed Consensus in Multi-vehicle Cooperative Control. – London : Springer London, 2008.

33. Rosenblum, M., Tass, P., Kurths, J., Volkmann, J., Schnitzler, A., and Freund, H.J. Detection of phase locking from noisy data: application to magnetoencephalography // Chaos In Brain?. 2000. P. 34–51.

34. Semenov D. M., Fradkov A. L. Adaptive synchronization of two coupled non-identical Hindmarsh-Rose systems by the Speed Gradient method // IFAC PapersOnLine. 2018. Vol. 51, Issue 33. P. 12–14.

35. Shilnikov A., Kolomiets M. Methods of the Qualitative Theory for the Hindmarsh-Rose Model: a Case Study - a Tutorial // International Journal of Bifurcation and Chaos. 2008. Vol. 18, Issue 8. P. 2141–2168.

36. Smetanin N., Volkova K., Zabodaev S., Lebedev M., Ossadtchi A. NFBLab — A Versatile Software for Neurofeedback and Brain-Computer Interface Research // Frontiers in neuroinformatics. 2018. Vol. 12. 100.

37. Steur E., Tyukin I., Nijmeijer H. Semi-passivity and synchronization of diffusively coupled neuronal oscillators // Physica D: Nonlinear Phenomena. 2009. Vol. 238, Issue 21. P. 2119–2128.

38. Strogatz S. H., Stewart I. Coupled oscillators and biological synchronization // Scientific American. 1993. Vol. 269, Issue 6. P. 102–109.

39. Uhlhaas P. J., Pipa G., Lima B., Melloni, L., Neuenschwander, S., Nikolić, D., Singer, W. Neural synchrony in cortical networks: History, concept and current status // Frontiers in Integrative Neuroscience. 2009. Vol. 3. 17.

Приложение

Код MATLAB программы для моделирования сети Хиндмарш-Роуз

main.m

```
clear all; close all;
%% ========================================================================


%
%     Adaptive synchronization of Hindmarsh–Rose networks
%                written by Danila Semenov, 2019
%
%% ========================================================================


%% Declaration
node = @(x) 6*(x - 1) + 1; % Node pointer


%% Definition
global callNetHR_Plotting;
callNetHR_Plotting = 0;

numNodes = 201; % number of network's nodes

% Network Generation
% graph adjacency matrix
param.Adj = zeros(numNodes);
for i = 1 : numNodes
    for j = i+1 : numNodes
```

```
24            param.Adj(i,j) = randi([0 1]);
25        end
26    end
27    param.Adj = param.Adj + param.Adj';
28
29
30    % Common parameters
31    param.sigma = 1e-3;
32    param.gamma0 = 5;
33    param.gamma1 = 10*ones(numNodes,1);
34
35    for i = 1 : numNodes
36        % System parameters
37        sysParam(i,1).a=1; sysParam(i,1).b=3;
38        sysParam(i,1).c=1; sysParam(i,1).d=5;
39        sysParam(i,1).eps1=3e-3; sysParam(i,1).s=4;
40        sysParam(i,1).r= -random('Uniform',0.99, 1.0);
41
42        %Initial system states
43        % x0_i variable
44        x0(node(i),1) = random('Uniform',-2, 2);
45        % y0_i variable
46        x0(node(i)+1,1) = random('Uniform',-10, 1);
47        % z0_i variable
48        x0(node(i)+2,1) = random('Uniform',-0.25, 0.25);
49        % theta0_1i variable
50        x0(node(i)+3,1) = random('Uniform',-0.1, 0.1);
51        % theta0_2i variable
52        x0(node(i)+4,1) = random('Uniform',-0.1, 0.1);
53        % theta0_3i variable
54        x0(node(i)+5,1) = random('Uniform',-0.1, 0.1);
55    end
56
57    disp('Calculations in progress, please wait...');
```

```
58 disp('');
59
60 %% Main body
61 % The systems without control
62
63 control = false;
64 [t1,x1] = ode45(@(t1,x1) HRNetwork(t1, x1, param,
     sysParam,...
65           numNodes, control),[0 2005], x0);
66
67 % The systems under control
68 control = true;
69 [t2,x2] = ode45(@(t2,x2) HRNetwork(t2, x2, param,
     sysParam,...
70       numNodes, control),[0 2005], x0);
71
72 disp('Calculations completed!...
73           Data processing has started...');
74 %% Plotting
75 % Plot of the system dynamics without control
76 NetHR_Plotting(t1, x1, param, numNodes);
77 clear t1 x1;
78
79 % Plot of the system dynamics under control
80 NetHR_Plotting(t2, x2, param, numNodes);
81 disp('Data processing completed.');
82 clear all;
```

HRsystem.m

```
function F = HRsystem(x, param)
%%
```

```
%
%     Adaptive synchronization of Hindmarsh-Rose networks
%                written by Danila Semenov, 2019
%
%%
```

```
% Description:
%                              Hindmarsh-Rose system
%
% x is a vector states
% param is a struture of the parameters
% link is a connection between systems
% u is the controller
%
%%
```

```
%% Parameters
a = param.a; b = param.b; c = param.c; d = param.d;
eps1 = param.eps1; s = param.s; r = param.r;

%% Body of the function

if(nargin == 2)
    F(1,1) = x(2,1)-a*x(1,1)^3+b*x(1,1)^2-x(3,1);
    F(2,1) = c-d*x(1,1)^2-x(2,1);
    F(3,1) = eps1*(s*(x(1,1)-r)-x(3,1));
else
```

```
28      error('Incorrect number of the arguments in...
29      "HR system" func.');
30  end
31
32  end
```

HRnetwork.m

```
1  function F = HRNetwork(t, x, param, sysParam, numNodes,
       control)
2  %%
       ================================================================


3  %
4  %     Adaptive synchronization of Hindmarsh-Rose networks
5  %               written by Danila Semenov, 2019
6  %
7  %%
       ================================================================


8
9  %% Description
10 %                                  Overall system
11 %
12 %% Declaration
13  link = zeros(numNodes,1);
14 u = zeros(numNodes,1);
15
16 node = @(x) 6*(x - 1) + 1; % Node pointer
17
18 %% Definition
19 B = [1; 0 ; 0];
20
21 % Common parameters
22 sigma = param.sigma;
```

```
23 A = param.Adj;
24 gamma0 = param.gamma0;
25 gamma1 = param.gamma1;
26
27 % Mean-field dynamics
28 x_mean = mean(x(node(1:numNodes), 1));
29 y_mean = mean(x(node(1:numNodes)+1, 1));
30
31 phi(1:numNodes,1) = x(node(1:numNodes),1) + x_mean;
32 delta_x(1:numNodes,1) = x(node(1:numNodes),1) - x_mean;
33 delta_y(1:numNodes,1) = x(node(1:numNodes)+1,1) - y_mean;
34
35 %% Main body
36
37 % Connection between nodes
38 for i = 1 : numNodes
39     for j = 1 : numNodes
40         link(i,1) = link(i,1) +...
41          sigma*A(i,j)*(x(node(j),1) - x(node(i),1));
42     end
43 end
44 % Controller
45 if (control == true)
46     u(1:numNodes,1) = ControlFcn(x, param, numNodes);
47 end
48
49 % Nodes
50 for i = 1 : numNodes
51     F(node(i):node(i)+2,1) = ...
52         HRsystem(x(node(i):node(i)+2,1),...
53          sysParam(i,1))+B*link(i,1)+B*u(i,1);
54
55     if (control == true)
56         F(node(i)+3:node(i)+5,1) = ...
```

```
57            -gamma1(i,1)*delta_x(i,1)...
58             *[phi(i,1)*delta_x(i,1);...
59             phi(i,1)*delta_y(i,1);  1];
60        else
61            F(node(i)+3:node(i)+5,1) = [0;  0;  0];
62        end
63    end
64
65    % End of the Function
66    end
```

ControlFcn.m

```
1  function u = ControlFcn(x, netParam, numNodes)
2  %%
   ================================================================

3  %
4  %     Adaptive synchronization of Hindmarsh-Rose networks
5  %                written by Danila Semenov, 2019
6  %
7  %%
   ================================================================

8  % Description:
9  %                              Control Law
10 %
11 %%
   ================================================================

12
13 gamma0 = netParam.gamma0;
14
15 node = @(x) 6*(x - 1) + 1; % Node pointer
16
```

```
17 % Mean-field dynamics
18 x_mean = mean(x(node(1:numNodes), 1));
19 y_mean = mean(x(node(1:numNodes)+1, 1));
20
21 phi(1:numNodes,1) = x(node(1:numNodes),1) + x_mean;
22 delta_x(1:numNodes,1) = x(node(1:numNodes),1) - x_mean;
23 delta_y(1:numNodes,1) = x(node(1:numNodes)+1,1) - y_mean;
24
25 u = -(gamma0-x(node(1:numNodes)+3,1).*...
26 phi(1:numNodes,1)).*delta_x(1:numNodes,1)...
27 +x(node(1:numNodes)+4,1).*phi(1:numNodes,1).*...
28 delta_y(1:numNodes,1)+x(node(1:numNodes)+5,1);
29
30 end
```

netHR_Plotting.m

```
1 function NetHR_Plotting(t, x, param, numNodes)
2
3 %%
```

```
4 %
5 %    Adaptive synchronization of Hindmarsh-Rose networks
6 %              written by Danila Semenov, 2019
7 %
8 %%
```

```
9 % Description:
10 %                                  Plotting
11 %
12 %%
```

```
13
14 %% Preparation
15 global callNetHR_Plotting; %number of calling "Plotting".
16 N = length(t); % length of time series
17
18 node = @(x) 6*(x - 1) + 1; % Node pointer
19
20
21 index = 1;
22 while(index <= N)
23 % Standard Deviation
24 e_x(index,1) = std(x(index, node(1:numNodes)));
25 e_y(index,1) = std(x(index, node(1:numNodes)+1));
26 e_z(index,1) = std(x(index, node(1:numNodes)+2));
27
28 %Mean control
29 u(:,index) = ControlFcn(x(index,:)', param, numNodes);
30 mean_u(index,1) = mean(u(:,index));
31 index = index + 1;
32 end
33
34 node = @(x) 6*(x - 1) + 1; % Node pointer
35
36
37 %% Main body
38 set(0,'defaulttextinterpreter','none');
39 [T,X] = meshgrid(t,1:numNodes);
40 % Plot x1 x2
41 fig1 = figure;
42 set(gcf,'units','inches');
43 plotPosition = get(gcf, 'Position');
44 plotWidth = plotPosition(3);
45 plotHeight = plotPosition(4);
46 set(fig1,'PaperUnits','inches',...
```

```
47 'PaperSize',[plotWidth plotHeight],...
48 'PaperPosition',[0 0 plotWidth plotHeight]);
49
50 contourf(T,X,x(:,node(1:numNodes))','LineStyle','none');
51 title(' ', 'FontSize', 36);
52 colormap hot;
53 cbar=colorbar;
54 title(cbar,'$x(t)$','Interpreter','latex','FontSize',20);
55 cbarMaxTick = max(cbar.YTick);
56 cbarMinTick = min(cbar.YTick);
57 stepTick = (cbarMaxTick - cbarMinTick)/3;
58 cbar.YTick=round(cbar.YTick : stepTick : cbarMaxTick, 2);
59 ax1 = gca;
60 set(ax1,'ticklabelinterpreter','none','FontSize',18);
61 set(ax1,'box','on','TickLength',[0 0]);
62 XTickMin = 1000;
63 XTickMax = 2000;
64 stepXTick = (XTickMax - XTickMin)/5;
65 ax1.XTick = XTickMin : stepXTick : XTickMax;
66 ax1.YTick = 2 : numNodes;
67 YTickLabel = ax1.YTickLabel;
68 temp = str2double(string(YTickLabel(:,1))) - 1;
69 index = 1; base = 50;
70 while(index <= numNodes-1)
71     if(mod(index, base) == 0)
72         ax1.YTickLabel(index,1) = num2cell(temp(index,1));
73     else
74         ax1.YTickLabel(index,1) = {''};
75     end
76     index = index + 1;
77 end
78 axYTickLabel = ax1.YTickLabel;
79 xlabel('$t$ (ms)','FontSize',18, 'interpreter','latex');
80 ylabel('node','FontSize',18,'interpreter','latex');
```

```
81 xlim([XTickMin XTickMax]); ylim([1 numNodes]);
82 print('-dpdf','-r500',...
83 ['x' num2str(callNetHR_Plotting + 1) '_variable.pdf']);
84 close(fig1);
85 clear fig1 ax1;
86
87 % Plot y1 y2
88 fig2 = figure;
89 set(gcf,'units','inches');
90 plotPosition = get(gcf, 'Position');
91 plotWidth = plotPosition(3);
92 plotHeight = plotPosition(4);
93 set(fig2,'PaperUnits','inches',...
94 'PaperSize',[plotWidth plotHeight],...
95 'PaperPosition',[0 0 plotWidth plotHeight]);
96
97 contourf(T,X,x(:,node(1:numNodes+1)','LineStyle','none');
98 title(' ', 'FontSize', 36);
99 colormap hot;
100 cbar=colorbar;
101 title(cbar,'$y(t)$','Interpreter','latex','FontSize',20);
102 cbarMaxTick = max(cbar.YTick);
103 cbarMinTick = min(cbar.YTick);
104 stepTick = (cbarMaxTick - cbarMinTick)/3;
105 cbar.YTick=round(cbar.YTick : stepTick : cbarMaxTick, 2);
106
107 ax2 = gca;
108 set(ax2,'ticklabelinterpreter','none','FontSize',18);
109 set(ax2,'box','on','TickLength',[0 0]);
110 XTickMin = 1000;
111 XTickMax = 2000;
112 stepXTick = (XTickMax - XTickMin)/5;
113 ax2.XTick = XTickMin : stepXTick : XTickMax;
114 ax2.YTick = 2 : numNodes;
```

```
115
116 ax2.YTickLabel = axYTickLabel;
117
118 xlabel('$t$ (ms)','FontSize',18, 'interpreter','latex');
119 ylabel('node','FontSize',18,'interpreter','latex');
120 xlim([XTickMin XTickMax]); ylim([1 numNodes]);
121 print('-dpdf','-r500',...
122 ['y' num2str(callNetHR_Plotting + 1) '_variable.pdf']);
123 close(fig2);
124 clear fig2 ax2;
125
126 % Plot z1 z2
127 fig3 = figure;
128 set(gcf,'units','inches');
129 plotPosition = get(gcf, 'Position');
130 plotWidth = plotPosition(3);
131 plotHeight = plotPosition(4);
132 set(fig3,'PaperUnits','inches',...
133 'PaperSize',[plotWidth plotHeight],...
134 'PaperPosition',[0 0 plotWidth plotHeight]);
135
136 contourf(T,X,x(:,node(1:numNodes+2)','LineStyle','none');
137 title(' ', 'FontSize', 36);
138 colormap hot;
139 cbar=colorbar;
140 title(cbar,'$z(t)$','Interpreter','latex','FontSize',20);
141 cbarMaxTick = max(cbar.YTick);
142 cbarMinTick = min(cbar.YTick);
143 stepTick = (cbarMaxTick - cbarMinTick)/3;
144 cbar.YTick=round(cbar.YTick : stepTick : cbarMaxTick, 2);
145
146 ax3 = gca;
147 set(ax3,'ticklabelinterpreter','none','FontSize',18);
148 set(ax3,'box','on','TickLength',[0 0]);
```

```
149 XTickMin = 1000;
150 XTickMax = 2000;
151 stepXTick = (XTickMax - XTickMin)/5;
152 ax3.XTick = XTickMin : stepXTick : XTickMax;
153 ax3.YTick = 2 : numNodes;
154 ax3.YTickLabel = axYTickLabel;
155
156 xlabel('$t$ (ms)','FontSize',18, 'interpreter','latex');
157 ylabel('node','FontSize',18,'interpreter','latex');
158 xlim([XTickMin XTickMax]); ylim([1 numNodes]);
159 print('-dpdf','-r500',...
160 ['z' num2str(callNetHR_Plotting + 1) '_variable.pdf']);
161 close(fig3);
162 clear fig3 ax3;
163
164 if(callNetHR_Plotting == 0)
165
166 % Plot x1 - x2
167 fig4 = figure;
168 set(gcf,'units','inches');
169 plotPosition = get(gcf, 'Position');
170 plotWidth = plotPosition(3);
171 plotHeight = plotPosition(4);
172 set(fig4,'PaperUnits',...
173 'inches','PaperSize',[plotWidth plotHeight],...
174 'PaperPosition',[0 0 plotWidth plotHeight]);
175
176 plot(t, e_x(:,1),'LineWidth',2.5,...
177                  'Color', [0.8500, 0.3250, 0.0980]);
178 hold on;
179 grid on;
180 ax4 = gca;
181 set(ax4,'ticklabelinterpreter','none','FontSize',18);
182 set(ax4,'box','on','TickDir','in','LineWidth', 2);
```

```
YTickMax = 2;
YTickMin = 0;
stepTick = (YTickMax - YTickMin)/4;
ax4.YTick = YTickMin : stepTick : YTickMax;
XTickMin = 1000;
XTickMax = 2000;
stepXTick = (XTickMax - XTickMin)/5;
ax4.XTick = XTickMin : stepXTick : XTickMax;
xlim([XTickMin XTickMax]); ylim([YTickMin YTickMax]);
xlabel('$t$ (ms)','FontSize',20, 'interpreter','latex');
ylabel('$S_x(t)$','FontSize',20, 'interpreter','latex');
print('-dpdf','-r500',...
['x' num2str(callNetHR_Plotting + 1) ' Error.pdf']);
close(fig4);
clear fig4 ax4;


% Plot y1 - y2
fig5 = figure;
set(gcf,'units','inches');
plotPosition = get(gcf, 'Position');
plotWidth = plotPosition(3);
plotHeight = plotPosition(4);
set(fig5,'PaperUnits','inches',...
'PaperSize',[plotWidth plotHeight],...
'PaperPosition',[0 0 plotWidth plotHeight]);
hold on;
plot(t, e_y(:,1),'LineWidth',2.5,...
 'Color', [0.8500, 0.3250, 0.0980]);
grid on;
ax5 = gca;
set(ax5,'ticklabelinterpreter','none','FontSize',18);
set(ax5,'box','on','TickDir','in','LineWidth', 2);
YTickMax = 8;
```

```
YTickMin = 0;
stepTick = (YTickMax - YTickMin)/4;
ax5.YTick = YTickMin : stepTick : YTickMax;
XTickMin = 1000;
XTickMax = 2000;
stepXTick = (XTickMax - XTickMin)/5;
ax5.XTick = XTickMin : stepXTick : XTickMax;
xlim([XTickMin XTickMax]); ylim([YTickMin YTickMax]);
xlabel('$t$ (ms)','FontSize',20, 'interpreter','latex');
ylabel('$S_y(t)$','FontSize',20,'interpreter','latex');
print('-dpdf','-r500',...
['y' num2str(callNetHR_Plotting + 1) ' Error.pdf']);
close(fig5);
clear fig5 ax5;


% Plot z1 - z2
fig6 = figure;
set(gcf,'units','inches');
plotPosition = get(gcf, 'Position');
plotWidth = plotPosition(3);
plotHeight = plotPosition(4);
set(fig6,'PaperUnits',...
'inches','PaperSize',[plotWidth plotHeight],...
'PaperPosition',[0 0 plotWidth plotHeight]);

hold on;
plot(t, e_z(:,1),'LineWidth',2.5,...
 'Color', [0.8500, 0.3250, 0.0980]);
grid on;
ax6 = gca;
set(ax6,'ticklabelinterpreter','none','FontSize',18);
set(ax6,'box','on','TickDir','in','LineWidth', 2);
YTickMax = 0.3;
```

```
YTickMin = 0;
stepTick = (YTickMax - YTickMin)/4;
ax6.YTick = YTickMin : stepTick : YTickMax;
XTickMin = 1000;
XTickMax = 2000;
stepXTick = (XTickMax - XTickMin)/5;
ax6.XTick = XTickMin : stepXTick : XTickMax;
xlim([XTickMin XTickMax]); ylim([YTickMin YTickMax]);
xlabel('$t$ (ms)','FontSize',20, 'interpreter','latex');
ylabel('$S_z(t)$','FontSize',20,'interpreter','latex');
print('-dpdf','-r500',...
['z' num2str(callNetHR_Plotting + 1) ' Error.pdf']);
close(fig6);
clear fig6 ax6;

else

% Plot x1 - x2
fig4 = figure;
set(gcf,'units','inches');
plotPosition = get(gcf, 'Position');
plotWidth = plotPosition(3);
plotHeight = plotPosition(4);
set(fig4,'PaperUnits','inches',...
'PaperSize',[plotWidth plotHeight],...
'PaperPosition',[0 0 plotWidth plotHeight]);

plot(t, e_x(:,1),'LineWidth',2.5,...
'Color', [0.8500, 0.3250, 0.0980]);
hold on;
grid on;
ax4 = gca;
set(ax4,'ticklabelinterpreter','none','FontSize',18);
set(ax4,'box','on','TickDir','in','LineWidth', 2);
```

```
285 YTickMax = 7.5e-5;
286 YTickMin = -7.5e-5;
287 stepTick = (YTickMax - YTickMin)/4;
288 ax4.YTick = YTickMin : stepTick : YTickMax;
289 XTickMin = 1000;
290 XTickMax = 2000;
291 stepXTick = (XTickMax - XTickMin)/5;
292 ax4.XTick = XTickMin : stepXTick : XTickMax;
293 xlim([XTickMin XTickMax]); ylim([YTickMin YTickMax]);
294 xlabel('$t$ (ms)','FontSize',20, 'interpreter','latex');
295 ylabel('$S_x(t)$','FontSize',20,'interpreter','latex');
296 print('-dpdf','-r500',...
297 ['x' num2str(callNetHR_Plotting + 1) ' Error.pdf']);
298 close(fig4);
299 clear fig4 ax4;
300
301
302 % Plot y1 - y2
303 fig5 = figure;
304 set(gcf,'units','inches');
305 plotPosition = get(gcf, 'Position');
306 plotWidth = plotPosition(3);
307 plotHeight = plotPosition(4);
308 set(fig5,'PaperUnits','inches',...
309 'PaperSize',[plotWidth plotHeight],...
310 'PaperPosition',[0 0 plotWidth plotHeight]);
311 hold on;
312 plot(t, e_y(:,1),'LineWidth',2.5,...
313  'Color', [0.8500, 0.3250, 0.0980]);
314 grid on;
315 ax5 = gca;
316 set(ax5,'ticklabelinterpreter','none','FontSize',18);
317 set(ax5,'box','on','TickDir','in','LineWidth', 2);
318 YTickMax = 1.5e-4;
```

```
YTickMin = -0.5e-4;
stepTick = (YTickMax - YTickMin)/4;
ax5.YTick = YTickMin : stepTick : YTickMax;
XTickMin = 1000;
XTickMax = 2000;
stepXTick = (XTickMax - XTickMin)/5;
ax5.XTick = XTickMin : stepXTick : XTickMax;
xlim([XTickMin XTickMax]); ylim([YTickMin YTickMax]);
xlabel('$t$ (ms)','FontSize',20, 'interpreter','latex');
ylabel('$S_y(t)$','FontSize',20,'interpreter','latex');
print('-dpdf','-r500',...
['y' num2str(callNetHR_Plotting + 1) ' Error.pdf']);
close(fig5);
clear fig5 ax5;


% Plot z1 - z2
fig6 = figure;
set(gcf,'units','inches');
plotPosition = get(gcf, 'Position');
plotWidth = plotPosition(3);
plotHeight = plotPosition(4);

set(fig6,'PaperUnits','inches',...
'PaperSize',[plotWidth plotHeight],...
'PaperPosition',[0 0 plotWidth plotHeight]);

hold on;
plot(t, e_z(:,1),'LineWidth',2.5,...
'Color', [0.8500, 0.3250, 0.0980]);
grid on;
ax6 = gca;
set(ax6,'ticklabelinterpreter','none','FontSize',18);
set(ax6,'box','on','TickDir','in','LineWidth', 2);
```

```
353 YTickMax = 20e-3;
354 YTickMin = 0;
355 stepTick = (YTickMax - YTickMin)/4;
356 ax6.YTick=round(YTickMin : stepTick : YTickMax, 3);
357 XTickMin = 1000;
358 XTickMax = 2000;
359 stepXTick = (XTickMax - XTickMin)/5;
360 ax6.XTick = XTickMin : stepXTick : XTickMax;
361 xlim([XTickMin XTickMax]); ylim([YTickMin YTickMax]);
362 xlabel('$t$ (ms)','FontSize',20, 'interpreter','latex');
363 ylabel('$S_z(t)$','FontSize',20,'interpreter','latex');
364 print('-dpdf','-r500',...
365 ['z' num2str(callNetHR_Plotting + 1) ' Error.pdf']);
366 close(fig6);
367 clear fig6 ax6;
368 end
369
370 callNetHR_Plotting = callNetHR_Plotting + 1;
371 end
```