Московский Государственный Университет
им. М. В. Ломоносова
Механико-математический факультет
Кафедра дифференциальной геометрии и приложений

Дипломная работа

студента 507 группы
Блинова Вениамина Николаевича.

Компьютерное моделирование топологических резонансных дефектов молекулы ДНК

Научный руководитель:

д. ф.-м. н., профессор Голо Войслав Любомирович

Москва, 2010

1 Введение

Молекула ДНК представляет собой спираль из двух нитей. Каждая нить представляет собой последовательность сахаров и нуклеотидов. Пары нуклеотидов на разных нитях соединены водородными связями. Такие водородные связи и топология молекулы удерживают нити вместе. Совокупность водородных связей несомненно играет важную роль в биофизике ДНК. В отличие от более прочных ковалентных связей, они в большей степени подвержены деформациям и разрывам. Известно также, что межнитевые степени свободы в меньшей степени подвержены затуханию, а потому они представляют особый интерес при изучении резонансных эффектов.

В работах *G.Manning, Mandel, J.Leroy, Guéron*, [16, 17, 19, 20], приводятся данные ЯМР, свидетельствующие о разрывах единичной водородной связи в паре нуклеотидов. Время жизни такого разрыва составляет порядка 10^{-8} sec.

Другая группа *G.Altan-Bonnet et al.* [21, 22] на основе люминесцентных меток приводит экспериментальное наблюдение разрыва межнитевых водородных связей на некотором участке (порядка 10 оснований) с образованием долго ($10^{-3} - 10^{-5}$ sec) живущих дефектов, которые авторы назвали *пузырями* (*bubbles*). При таких деформациях топология ДНК претерпевает изменение фундаментальной группы, если рассматривать её как двумерное многообразие. Интересен и тот факт, что до сих пор неизвестна конформация разошедшихся при образовании пузыря участков ДНК. На этих участках молекула, по-видимому, более уязвима, поскольку в таком 'открытом' состоянии нуклеозиды в большей степени подвержены химическим изменениям.

В качестве вероятного примера такого изменения можно привести спонтанные мутации, предложенные *Watson*'ом и *Crick*'ом, [1, 2, 3, 4]. Они полагали, что азотистые основания могут претерпевать изменения, которые влекут ошибку в его распознавании ДНК-полимеразой при репликации. Это приводит к нарушению правила комплементарности, то есть к мутации. Одним из таких изменений может являться кето-енольная таутомерия для G и T, и амино-иминый переход для A и C. При нормальных условиях равновесие смещено в сторону амино-формы аденина и гуанина, и в сторону кето-форм тимина и цитозина. Тем не менее, в естественных условиях с малой вероятностью могут образовываться имино- и енольная-формы соответственно (их концентрация составляет примерно 10^{-4} - 10^{-5} mol/l). Как результат при репликации вместо обычных амино- и кето-пар $Adenine - Thymine$, $Guanine - Cytosine$ могут образовываться пары $A_{imino} - C$, $A - C_{imino}$, $G_{enol} - T$ или $G - T_{enol}$. В связи с этим, мутации — одна из причин необходимости исследования дефектов в ДНК, связанных с водородными связями и их свойств.

Ряд недавних работ посвящён объяснению возникновения пузырей, см.[21, 22] и ссылки. Их формирование в ДНК связывается, в частности, с нелинейной динамикой молекул ДНК, см. [5]. Однако нас интересует

идея, предложенная в статьях [6], [7], [8]. Авторы предлагают учитывать в моделях нерегулярность в структуре ДНК, в частности, случайное распределение азотистых оснований внутри молекулы, которое, как известно, влияет на конформацию молекулы и другие её физические свойства. ДНК больше не рассматривается в качестве идеальной B-ДНК. Возникает вопрос, к каким последствиям приводит подобная нерегулярность и как она может быть связана с формированием дефектов в ДНК.

Чтобы ответить на этот вопрос воспользуемся теорией, разработанной *И.М.Лифшицем* и соавторами, см.[9]-[13] (также [14] и [15]). Согласно этой теории, неупорядоченная среда имеет ряд характерных свойств. Так, неупорядоченность является причиной формирования локализованных возбуждений (см. [9]-[13]), то есть колебаний, почти вся энергии которых всегда остаётся в ограниченной области. Введение неупорядоченности также меняет вид спектра собственных частот системы, что было численно показано *Dean*'ом [26]. Он построил спектры и провёл их подробный анализ для различных случайных распределений параметров достаточно длинных цепочек. Сегодня можно синтезировать молекулы ДНК с заданной нуклеотидной последовательностью, получив, таким образом, большое разнообразие упругих систем с различными спектрами собственных частот. Так, ДНК может служить объектом экспериментов по динамике хаотических упругих систем. В этом смысле можно предположить, что формирование пузырей в молекуле ДНК обусловлено межнитевыми колебаниями достаточной для разрыва водородных связей амплитуды. Несомненно, в реальной ДНК нельзя пренебрегать нелинейными эффектами. Однако нелинейность подразумевает сложности, которые пока обойти крайне сложно. Тем не менее, можно предположить, что возникновение межнитевых колебаний большой амплитуды, например вследствие резонанса, может способствовать возникновению пузырей. По этой причине, имеет смысл искать локализованные межнитевые моды в неупорядоченных цепочках, соответствующих молекулами ДНК.

В этом отношении, нас интересуют колебания на частотах по порядку близких к макромолекулярным: гиперзвуковые. Известно, что ультразвук малой мощности широко используется в медицине и, как сегодня верят люди, сильно не вредит здоровью человека. Повышение интенсивности приводит к нагреванию раствора, что в случае ДНК может привести к денатурации. Наконец, взяв достаточно мощный источник ультразвука, можно столкнуться с эффектом кавитации, при котором в веществе образуются пузырьки, которые, схлопываясь, выделяют энергию, которой хватит, чтобы без труда разорвать молекулу ДНК. Пороги кавитации для ультразвука составляют порядка 0.03 $\mathrm{W/cm^2}$ и 3 $\mathrm{W/cm^2}$ для частот 0.25 MHz и 5 MHz [27].

Говоря о внешнем воздействии на ДНК, следует обратить внимание на следующие факты. В реальной жизни ДНК находится в растворах, потому окружение будет сильно влиять на характер любого внешнего воздействия. Механизм такого изменения весьма сложен. В частности,

остро стоит вопрос о степени демпфированности любых колебаний в водном растворе в связи с диссипацией. Важным моментом здесь является то, что на гиперзвуковых частотах уравнения Навье-Стокса перестают быть верными и её движение подчиняется другим законам (гидродинамика Мандельштама-Леонтовича). Тепловое движение воды происходит также на гиперзвуковых частотах, а потому вода может, в первую очередь, являться источником внешней силы, а не причиной затухания этих колебаний.
Сам по себе гиперзвук весьма интересен и заслуживает исследования. Воздействие гиперзвука на ДНК изучено значительно хуже, чем ультразвука и основная причина этого - сравнительно недавнее появление доступных источников гиперзвука. Пока не ясно, какого его действие на человека, на другие живые существа, даже на воду. Его изучение несомненно принесёт пользу в исследовании гидродинамики и динамики макромолекул.

2 Модель

Среди моделей ДНК можно выделить два основных класса. Первый составляют модели молекулярной динамики, в которых учитывается каждый атом, химические связи и производится расчёт динамики. Такие модели претендуют на реалистичность, однако крайне сложны в интерпретации результатов, а также требуют большой вычислительной мощности. Второй класс образуют более грубые, схематичные модели, содержащие лишь основные черты молекулы. Они значительно проще, хотя и не столь реалистичны, как первые. Поскольку нас интересуют лишь качественные эффекты, которые объясняются определёнными свойствами ДНК, мы будем использовать модель, которую относим ко второму классу. Рассматривая динамику молекулы ДНК будем учитывать основные черты её топологии:

1. ДНК состоит из двух нитей, образованными сахаро-фосфатными остовами с прикреплёнными к ним азотистыми основаниями;
2. молекула обладает геликоидальной симметрией;
3. последовательность пар оснований молекулы, вообще говоря, случайна.

El Hasan и *Calladine*, [18], описали схему внутренней геометрии двойной спирали ДНК, которая описывает относительное положение одного основания по отношению к другому в комплементарной паре, а также позиции самих пар оснований в модели *Watson*'а-*Crick*'а. Хотя последовательность нуклеотидов влияет на форму ДНК, вызывая девиации от идеальной B-формы, на малых длинах молекулы будем пренебрегать этим фактом, приближённо считая её B-ДНК. Таким образом, для

описания молекулы мы будем использовать модель одномерной цепочки, удовлетворяющую этим требованиям. Для качественного описания динамики молекулы выберем упрощённый набор переменных. Рассмотрим приближение, в котором плоскости пар оснований молекулы в неизогнутом состоянии параллельны и находятся на определённом расстоянии друг от друга. Это приближение оправдано на длинах порядка персистентной длины молекулы (150–1200 пар оснований в зависимости от раствора). Ввиду того, что нас прежде всего интересуют межнитевые моды молекулы, для описания положения одной пары оснований будем использовать две величины: угол φ и вектор $\vec{Y}$. Последний описывает растяжение связей в парах оснований по отношению к общему смещению пары, [28], его будем предполагать двумерным (см. рис. 1). Общее смещение обеих пар в модели описывается торсионным углом φ, обозначающим вращательное отклонение от положения равновесия всей пары оснований. При таком смещении напряжений между основаниями в паре не возникает, поскольку конформация пары не меняется, растяжения или сжатия происходят ковалентных связей цепочек спирали. Так, в состоянии равновесия все смещения нулевые, то есть, и φ, и $\vec{Y}$ равны нулю. Если приложить к разным концам молекулы в равновесном состоянии вращательные моменты, направленные по её оси в разные стороны, система деформируется и займёт новое положение. Важно, что при таком воздействии внутреннее состояние, то есть векторы $\vec{Y}$ останется невозмущённым. Межнитевые колебания, в свою очередь, получаются за счёт сил, действующих не по оси молекулы (поперечных растяжений или сжатий, к примеру). Продольное растяжение и сжатие происходит с большей упругой константой и не играет важной роли в описании межнитевых мод.

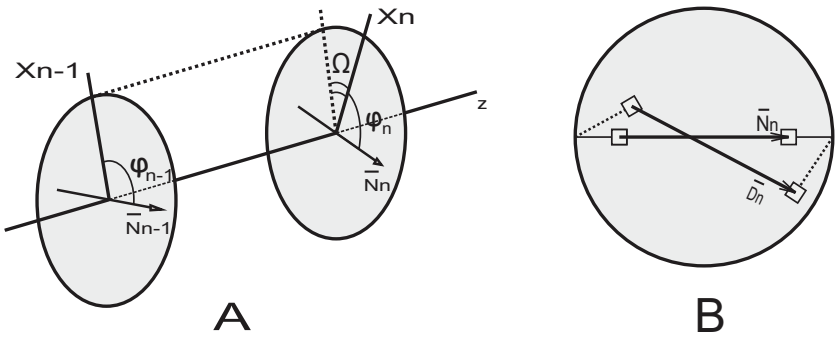

Рис. 1: A. Введение углов φ_n отклонения вектора невозмущённого состояния. Угол $\Omega = \pi/5$ соответствует повороту соседних плоскостей B-ДНК. B. Невозмущённое и возмущённое состояния $\vec{N}_n$ и $\vec{D}_n$ вектора, описывающего относительное положение азотистых оснований в комплементарной паре. Вводимый нами вектор $\vec{Y}_n = \vec{D}_n - \vec{N}_n$.

Полная энергия нашей модели имеет вид

$$E = T + U,$$

где кинетическая энергия T складывается из энергии вращения комплементарной пары и энергии относительного движения нуклеотидов

$$T = \sum_{n=0}^{N} \frac{I_n \dot{\varphi}_n^2}{2} + \sum_{n=0}^{N} \frac{m_n \dot{\vec{Y}}_n^2}{2},$$

а потенциальная содержит следующие члены, отвечающие параболическим потенциалам, позволяющим учесть жёсткие связи в молекуле

$$U = \sum_{n=0}^{N} \frac{\epsilon_n \vec{Y}_n^2}{2} + \sum_{n=0}^{N-1} \frac{\tau_n}{2a^2} |\varphi_{n+1} - \varphi_n|^2 + \sum_{n=0}^{N-1} \frac{k_n}{2a^2} |R^{-1}(\Omega + \varphi_{n+1} - \varphi_n)\vec{Y}_{n+1} - \vec{Y}_n|^2.$$

Здесь N - число пар оснований в молекуле, m_n - массовые коэффициенты, связанные с движением оснований в паре друг относительно друга, I_n - моменты инерции пар оснований, связанные с торсионным движением пары, τ_n - коэффициенты жёсткости торсионного движения пары, ϵ_n - упругий коэффициент растяжения связи между основаниями в паре и k_n - упругие константы жёсткости молекулы при относительном смещении оснований одной пары при неподвижных соседних парах. R - матрица поворота на угол, равный аргументу, написанному в скобках, Ω - угол $\pi/5$, соответствующий повороту при переходе к соседней паре оснований в B-форме. Третье слагаемое, таким образом, есть подкручивание системы координат так, чтобы векторы $\vec{Y}_n$ и $\vec{Y}_{n+1}$ лежали бы в одной и той же системе координат. Это подкручивание необходимо сделать, поскольку в равновесном состоянии соседние векторы $\vec{Y}$ уже находятся под некоторым углом друг относительно друга, а потому следуют учитывать эту закрутку, например, таким образом. Коэффициент a - расстояние между плоскостями соседних пар оснований, его в расчётах мы везде будем класть равным 1, как характерную длину (для реальной ДНК это расстояние составляет порядка 3,4 Å).

Преобразуем выписанные выражения, вводя комплексные числа z_n вместо векторов $\vec{Y}_n$ так, что $\vec{Y}_n = (Y_n^1; Y_n^2) \to z_n = Y_n^1 + iY_n^2$. Тогда выражения для энергий будут выглядеть следующим образом

$$T = \sum_{n=0}^{N} \frac{I_n \dot{\varphi}_n^2}{2} + \sum_{n=0}^{N} \frac{m_n \dot{z}_n \dot{z}_n^*}{2},$$

$$U = \sum_{n=0}^{N} \frac{\epsilon_n z_n z_n^*}{2} + \sum_{n=0}^{N-1} \frac{\tau_n}{2a^2} |\varphi_{n+1} - \varphi_n|^2 + \sum_{n=0}^{N-1} \frac{k_n}{2a^2} |z_{n+1} - \exp[i(\Omega + \varphi_{n+1} - \varphi_n)] z_n|^2.$$

Далее, замена $z_n = e^{in\Omega} y_n$ сводит эти выражения к виду

$$T = \sum_{n=0}^{N} \frac{I_n \dot{\varphi}_n^2}{2} + \sum_{n=0}^{N} \frac{m_n \dot{y}_n \dot{y}_n^*}{2},$$

$$U = \sum_{n=0}^{N} \frac{\epsilon_n y_n y_n^*}{2} + \sum_{n=0}^{N-1} \frac{\tau_n}{2a^2} |\varphi_{n+1} - \varphi_n|^2 + \sum_{n=0}^{N-1} \frac{k_n}{2a^2} |y_{n+1} - \exp[i(\varphi_{n+1} - \varphi_n)] z_n|^2.$$

Теперь сделаем предположение о том, что величина $(\varphi_{n+1} - \varphi_n)$ мала и сделаем разложение в ряд Тейлора экспоненты в последней скобке:

$$e^{i(\varphi_{n+1} - \varphi_n)} = 1 + i(\varphi_{n+1} - \varphi_n) + O((\varphi_{n+1} - \varphi_n)^2).$$

С учётом этого предположения от экспоненты останутся только первые два члена. Составим функцию Лагранжа для данной системы $L = T - U$. Уравнения Лагранжа динамики системы запишутся в виде

$$\begin{aligned} a^2 I_n \ddot{\varphi}_n = \tau_{n-1}\varphi_{n-1} - (\tau_{n-1} + \tau_n)\varphi_n + \tau_n \varphi_{n+1} - \\ - k_n(a_n b_{n+1} - a_{n+1} b_n) + k_{n-1}(a_{n-1} b_n - a_n b_{n-1}) \end{aligned} \tag{1}$$

$$\begin{aligned} a^2 m_n \ddot{a}_n = -a^2 \epsilon_n a_n + k_{n-1} a_{n-1} - (k_{n-1} + k_n) a_n + k_n a_{n+1} + \\ + k_n(\varphi_{n+1} - \varphi_n) b_{n+1} - k_{n-1}(\varphi_n - \varphi_{n-1}) b_{n-1} \end{aligned} \tag{2}$$

$$\begin{aligned} a^2 m_n \ddot{b}_n = -a^2 \epsilon_n b_n + k_{n-1} b_{n-1} - (k_{n-1} + k_n) b_n + k_n b_{n+1} - \\ - k_n(\varphi_{n+1} - \varphi_n) a_{n+1} + k_{n-1}(\varphi_n - \varphi_{n-1}) a_{n-1} \end{aligned} \tag{3}$$

Здесь введено обозначение $a_n = \texttt{Re}\ y_n$ и $b_n = \texttt{Im}\ y_n$. Таким образом, исходная модель распалась на три вещественных цепочки, связанные нелинейными соотношениями.

3 Компьютерный анализ динамики

В данной работе проводится компьютерный анализ динамики молекулы ДНК. При счёте использовалось, для надёжности, несколько схем интегрирования дифференциальных уравнений: неявный метод трапеции, Verlet, LeapFrog, а также явный и неявный методы Адамса. Метод Рунге-Кутта не использовался, поскольку быстро 'разваливается' на подобных системах. Шаг брался равным от 0.01 до 0.001 от характерного времени, обычно - периода внешнего воздействия.

В используемой модели имеется довольно большое число параметров, для каждой пары оснований это m_n , I_n, ϵ_n, k_n. Однако исходя из того, что имеется лишь два типа комплементарных пар, а нас интересуют качественный результат, множество значений каждого из параметров содержит не более двух элементов, причём они зависят друг от друга некоторым образом: по сути, всё определяется типом пары, *A–T* или *G–C*. Отметим, что в уравнениях (1), (2) и (3) члены, стоящие в первой строке образуют уравнения трёх независимых систем. Каждая из них представляет систему N масс на прямой, соединённых пружинами (Рис.2).

Первое, что нас интересует в подобных системах, особенно в макромолекулярных, это собственные частоты и моды. В случае, когда

Рис. 2: Одномерная цепочка масс, связанных пружинами.

такая цепочка симметрична, то есть параметры одинаковы для всех узлов, можно явно указать вид спектра, сделав дискретное Фурье-преобразование (см., например, [24]). Динамика таких систем в случае, когда какие-либо параметры (например, массы) выбраны с некоторой случайностью весьма интересна и сложна. Некоторые выводы её касающиеся помогает сделать теория неупорядоченных систем *И.М.Лифшица*, [9]. Важнейшими свойствами таких систем является наличие щелей в спектре собственных частот и существование долго живущих локализованных колебаний. Согласно теории Лифшица, любое возбуждение малого участка неупорядоченной цепи будет сохранять большую часть своей энергии на некотором фиксированном конечном отрезке. Однако его теория предполагает неограниченно длинные цепочки. Для ограниченных отрезков цепей локализованные колебания наблюдаются в высокочастотной области спектра. Другим результатом этой теории является наличие щелей в спектре собственных частот таких систем. Опять же, если говорить о конечных системах, в них спектр дискретный и говорить о щелях в обычном понимании не следует. Здесь стоит вспомнить работы *P.Dean*'а, [26], в которых были впервые численно получены спектры некоторых случайных конечных цепочек. Получив численно собственные значения, он строит гистограммы плотности $G(\omega^2)$, то есть графики, демонстрирующие распределение квадратов собственных частот по отрезкам числовой оси. Спектры, полученные им численно достаточно сложны и точно предсказать вид спектра исходя только из теоретических данных крайне сложно. Однако щели в спектрах, то есть участки без собственных частот, по обе стороны от которых есть собственные значения, были получены, что подтверждает теорию Лифшица. Численные алгоритмы, приведённые в его работе позволили повторить и несколько углубить его работу. Используя современные компьютеры, удалось получить спектры большого числа неупорядоченных цепочек, найти собственные колебания, а также провести численное интегрирование уравнений движения и проследить время жизни локализованных колебаний, исследовать их зависимость от свойств цепочки. В работе [25] была замечена возможность параметрического резонанса к внешнему воздействию.

Остановимся поподробнее на явлении параметрического резонанса. В простейшем случае его можно наблюдать для осциллятора. Рассмотрим уравнение

$$\ddot{u} + \chi\dot{u} + (n^2 - 2\alpha \sin 2pt)u = 0 \tag{4}$$

где χ и α предполагаются малыми. При $p = n$ оно имеет неограниченное

решение, (см. [29]), которое существует при

$$\alpha \geq \chi p \tag{5}$$

и если выполнено *условие Рэлея*:

$$p^2 = n^2 - \sqrt{\alpha^2 - \chi^2 p^2} \tag{6}$$

Возникает вопрос, сохранятся ли качественные характеристики неупорядоченных цепочек и какие новые свойства могут появиться у более сложной модели.

Основная характеристика, которая нас интересует, это наличие резонанса по отношению к внешнему возбуждению. Механизм воздействия звуковой волны на молекулу ДНК неизвестен. Чтобы иметь представление о том, что вообще происходит, применим следующие соображения. Поскольку воздействие подразумевается гиперзвуковым, то есть на частотах порядка $10^9 - 10^{12}\,Hz$, можно оценить длину волны. Скорость звука в конденсированных средах порядка $10^5\,cm/s$, длина волны гиперзвука порядка $10^{-7} - 10^{-4}\,m$. Персистентная длина (порядка 150 пар) молекулы ДНК составляет порядка $10^{-6}\,m$, диаметр порядка 20 Å или $10^{-7}\,m$. Исходя из этого, для малых частот гиперзвука длина волны на порядок превышает размеры молекулы. Поэтому, воздействие можно считать аналогичным действию плоской волны, то есть зависящим только от времени в каждой точке молекулы.

Введём в уравнения члены, отвечающие за диссипацию и внешнюю силу. Поскольку изменение вектора $\vec{Y}$ мы считаем независимым явно от воздействия среды, поскольку он находится 'внутри' молекулы, то оба новых члена будут добавлены в уравнение (1). Диссипацию добавим в самом простом виде $-\gamma\dot{\varphi}$, где γ - коэффициент диссипации. Это приближение, формально говоря, неверно, потому как в рассматриваемой модели нас интересуют частоты гиперзвука, на которых привычные уравнения динамики сплошной среды неверны. В связи с этим, с этим членом уравнения следует обращаться более тщательно. Мы чаще будет класть его нулём. Вопрос о внешней силе весьма сложен, для начала поставим её, как и в работе [25], равной $A\varphi\cos(\omega t)$, где A - амплитуда, ω - период внешнего воздействия. Это приближение имеет смысл, поскольку величина силы зависит от положения молекулы в конкретной точке. Конкретно охарактеризовать эту зависимость крайне сложно, поэтому остановимся пока на таком частном случае. Перепишем уравнение (1) в этом случае:

$$\begin{aligned} a^2 I_n \ddot{\varphi}_n = {} & \tau_{n-1}\varphi_{n-1} - (\tau_{n-1} + \tau_n)\varphi_n + \tau_n\varphi_{n+1} - \\ & - k_n(a_n b_{n+1} - a_{n+1}b_n) + k_{n-1}(a_{n-1}b_n - a_n b_{n-1}) - \\ & - \gamma\dot{\varphi} + A\varphi\cos(\omega t) \end{aligned} \tag{7}$$

Положим a и b малыми, тогда можно переписать это уравнение в виде

$$a^2 I_n \ddot{\varphi}_n = \tau_{n-1}\varphi_{n-1} - (\tau_{n-1} + \tau_n)\varphi_n + \tau_n\varphi_{n+1} - \\ - \gamma\dot{\varphi} + A\varphi\cos(\omega t) \tag{8}$$

В случае регулярной цепочки с зацикленными граничными условиями, преобразование Фурье позволяет свести эту систему уравнений её динамики (8) к системе независимых осцилляторов с воздействием как в случае параметрического резонанса (уравнение (4)). Таким образом, теория параметрического резонанса хорошо работает для регулярных цепочек, резонансными частотами будут удвоенные собственные частоты цепочки. Если же взять случайным распределение, например, параметров жёсткости τ_n, это рассуждение перестаёт быть верным. Рассмотрим частный случай, когда все $I_n = 1$, τ_n выбирается равновероятно из 1 и 1.5 (генератор - `rand()` из `C++`), $\gamma = 0$. Для каждой частоты ω из отрезка $[0; 5]$ найдём численно минимальную амплитуду A, при которой в течение 1000 периодов внешнего воздействия наступает резонанс. Формально проверяется условие возрастания начальной энергии в 100 раз. Для этого выбираем достаточно большую начальную амплитуду для каждой частоты и начинаем её уменьшать с шагом 0,01. Графики для такой цепи и регулярной ($\tau_n = 1$) имеют следующий вид (точность 0,01).

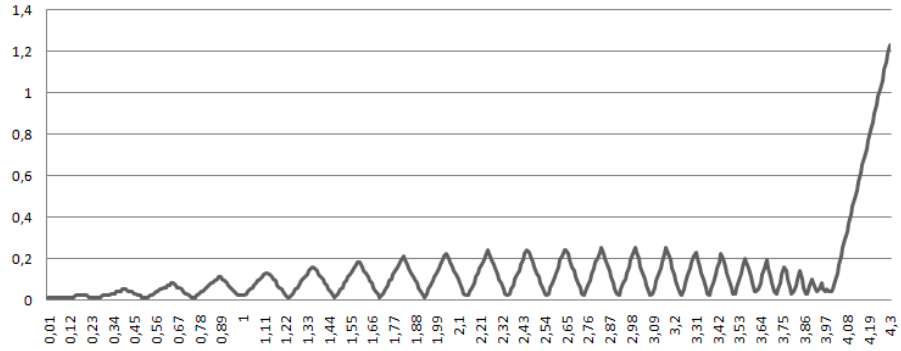

Рис. 3: Зависимость минимальной резонансной амплитуды от частоты параметрического воздействия для регулярной цепи $N = 50$, регулярная цепочка.

В этом случае, если теперь добавить нелинейные члены и цепочки a и b, можно получить поведение модели, при котором внешняя сила вызывает локализованное возбуждение φ-цепочки, которое, в свою очередь передаёт его на a- и b-цепочки. Теория Лифшица говорит, что для длинных молекул все моды локализованы. Однако, как уже было сказано, это верно на бесконечных масштабах. В случае конечной цепочки локализация будет для достаточно высоких частот, поскольку участок локализации может иметь размеры сравнимые со всей цепочкой. Приведём пример цепочки,

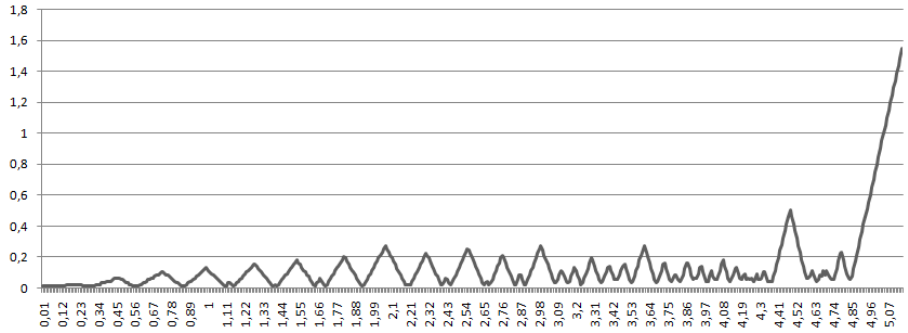

Рис. 4: Зависимость минимальной резонансной амплитуды от частоты параметрического воздействия для нерегулярной цепи $N = 50$, нерегулярная цепочка.

для которой имеет место резонанс локализованного колебания на области по порядку соответствующей размеру пузырей, полученных в опыте [22]. Начальное состояние бралось случайным, по модулю не превосходящим 0.005 для каждого параметра a_n, b_n, φ_n, все начальные скорости равными нулю.

Как видно из рисунка 5, воздействие образует локализованное колебание области порядка 40 узлов, что примерно соответствует экспериментальному размеру пузыря. Отметим, что о появлении пузыря говорить рано, поскольку растяжения ещё не достаточно для разрыва связей, однако такое колебание может служить зародышем пузыря. Поскольку график, аналогичный рис.4 для любой цепочки имеет похожий вид: несколько бугров и значительный рост при больших частотах, то высокоамплитудное воздействие на частотах, выше резонансных (ям) приведёт к похожему резонансному эффекту локализации. Таким образом, применительно к ДНК случайность нуклеотидной последовательности может стать причиной того, что при воздействии на молекулу гиперзвуком может происходить возникновение дефектов.

Роль диссипации, если её записать в виде $-\gamma\dot{\phi}$, при малых воздействиях оценивается условием Рэлея, которое при больших амплитудах перестаёт действовать (вывод этого условия использует ряд Тейлора по амплитуде, где отбрасываются члены высших порядков). Как видно из графика 6, зависимость линейна, что соответствует неравенству (5). На некоторой окрестности нуля линейность нарушается, что связано с действием условия Рэлея (6), неточным попаданием в частоту локализации и конечным временем ожидания резонанса. В идеальном случае (точная частота и очень долгое время счёта) кривая подчиняется только условию Рэлея (6). В случае, если частота внешнего воздействия не попадает точно в резонанс, будет наблюдаться порог как показано на 7. Далее, график выходит на прямую. Этот порог связан с отдалением от параметрического воздействия,

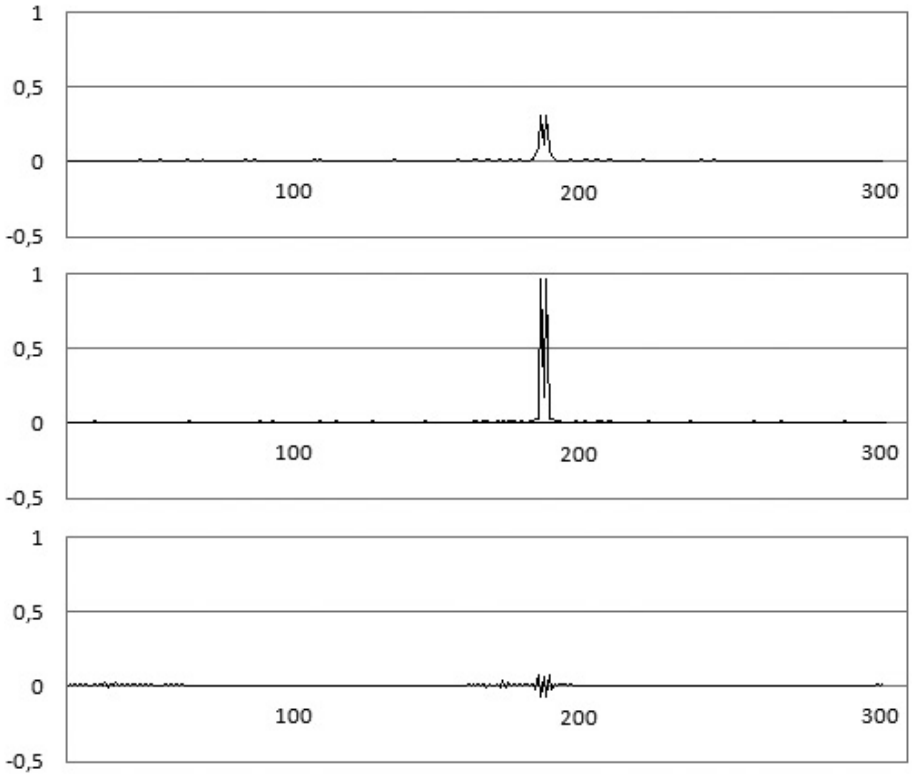

Рис. 5: Режим локализации для цепочки с $N = 300$, $m_n = 1$, $I_n = 1$, $\tau_n = k_n$ и равны 1 или 1,5 с вероятностью $\frac{1}{2}$, $\epsilon = 0.01$, воздействие на частоте 4.86 с амплитудой 0.12, коэффициент $\gamma = 0$. Состояние цепочек φ, a, b после 500 периодов воздействия возмущением $A\varphi\cos(\omega t)$ (метод - LeapFrog, шаг $\frac{1}{200}$ периода внешнего воздействия).

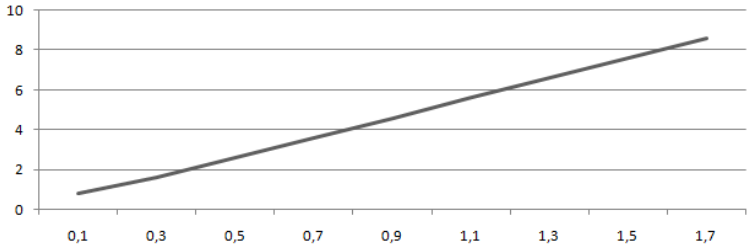

Рис. 6: Зависимость минимальной резонансной амплитуды от диссипации для цепочки с $N = 300$, $m_n = 1$, $I_n = 1$, $\tau_n = k_n$ и равны 1 или 1,5 с вероятностью $\frac{1}{2}$, $\epsilon = 0.01$, воздействие на частоте 4.86.

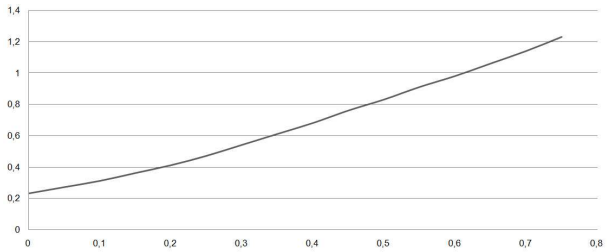

Рис. 7: Зависимость минимальной резонансной амплитуды от диссипации: порог при неточном попадании в частоту.

и его зависимость от частоты подобна той, что показана на графиках 3 и 4, выбор частоты, большей последнего локального минимума влечёт монотонное увеличение этого порога.

Рассмотрим теперь другой, более сложный случай, когда вид внешней силы задаётся выражением $F_n = A\sqrt{a_n^2 + b_n^2}\, cos(\frac{\pi n}{5} + \varphi_n)cos(\omega t)$. Такая сила включает следующие предположения: действующая сила зависит от длины вектора $\vec{Y_n}$ (чем он больше, тем шире молекула в данном месте и, аналогично парусу и ветру, сильнее действует волна), поворота молекулы относительно её оси (сила наибольшая в том случае, когда молекула расположена перпендикулярно вектору распространения волны), от амплитуды и времени воздействия. Таким образом, такая запись претендует на более подробное описание влияния волны на молекулу. Стоит однако, помнить, что точный механизм воздействия волны на ДНК неизвестен, а потому нам остаётся лишь строить догадки. Рассмотрим график, аналогичный 4, то есть для такой же нерегулярной цепочки, изменив вид внешнего воздействия и взяв теперь всю систему уравнений (1),(2),(3).

Как можно заметить, график 8 минимальных резонансных амплитуд имеет значительно более сложный осциллирующий вид, в отличие от предыдущих. Это связано со сложностью такой нелинейной системы;

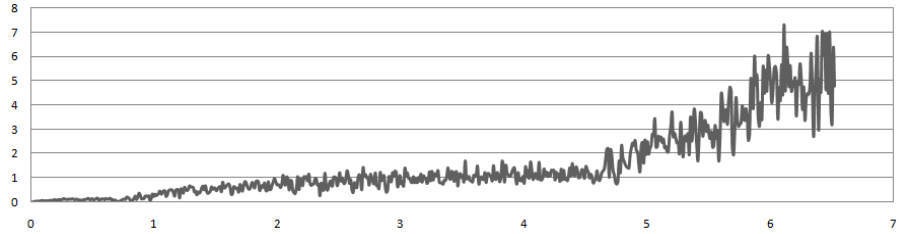

Рис. 8: Зависимость минимальной резонансной амплитуды от частоты параметрического воздействия для нерегулярной цепи $N = 50$, нерегулярная цепочка.

вероятно, от неё можно ждать большого количество различных моделей поведения, зависящих как от параметров самой системы, так и от начальных условий, что представляет сложность в её изучении. Однако и для этого случая характерна локализация, как, например на рисунке 9 (условия его счёта такие же, что и на рис. 5, кроме начальных условий. Последние взяты случайными не превосходящими 0.0005 по модулю). На графике, отражающем состояние φ-цепочки видны волны, появление которых связано с тем, что коэффициент $\sqrt{a_n^2 + b_n^2}$ не равен нулю при начальных условиях и на протяжении всего счёта. Тем не менее, на близких к резонансных частотах система ведёт себя аналогичным образом для обоих рассмотренных форм внешнего воздействия, можно отметить сходства резонансных участков, их положение.

Отметим также тот факт, что ждать появления подобных разрывов нужно в течение сравнительно долгого времени, в то время как стремительный рост амплитуды, происходит значительно быстрее, в течение 10–20 периодов, что соответствует своего рода 'рывку'.

4 Результаты

В принятой нами модели по порядку величин

1. единица длины: $3 \cdot 10^{-8} cm$;
2. единица массы: 500 дальтон, или $10^{-21} gr$;
3. единица времени:$10^{-9} sec$.

Попробуем оценить мощность источника внешнего воздействия, необходимую для появления дефектов на основе параметрического резонанса.
Рассмотрим некоторый объём жидкости, содержащий молекулы ДНК. Будем закачивать в него гиперзвук в виде монохроматической волны. Считаем при этом, что закачка приводит к возбуждению межнитевых мод в молекулах, что, в свою очередь, приводит к накоплению энергии в

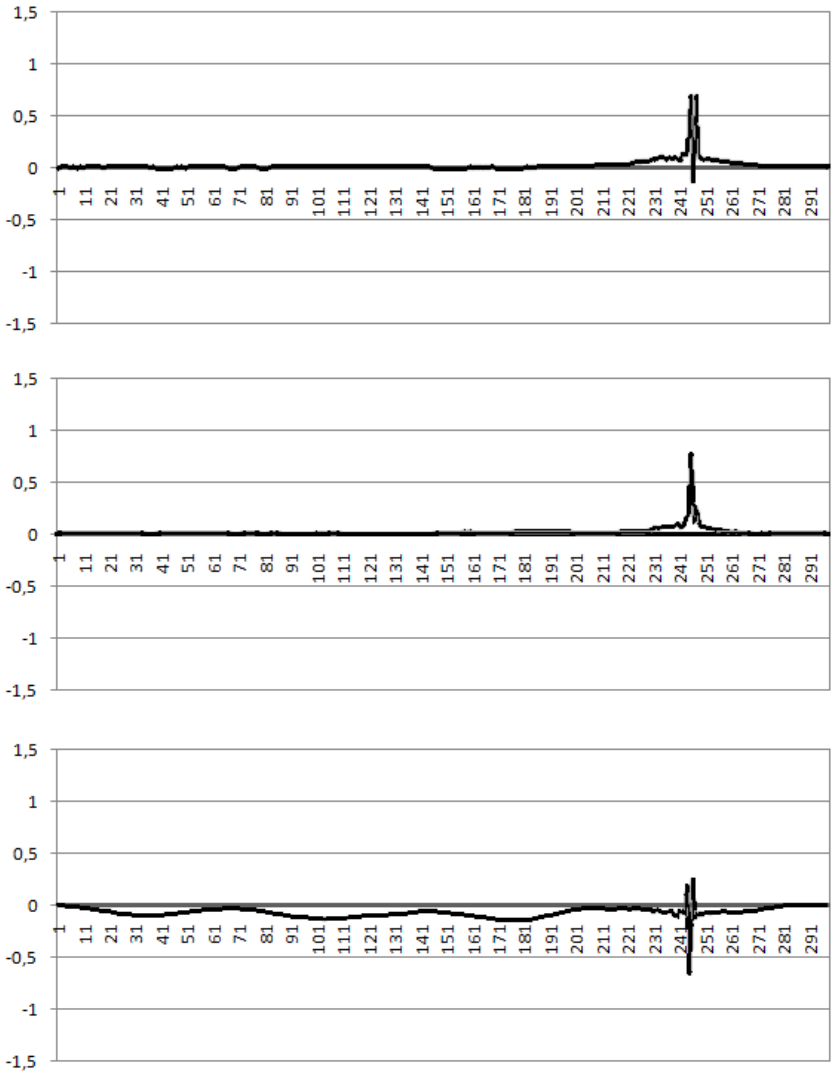

Рис. 9: Режим локализации для цепочки с $N = 300$, $m_n = 1$, $I_n = 1$, $\tau_n = k_n$ и равны 1 или 1,5 с вероятностью $\frac{1}{2}$, $\epsilon = 0.01$, воздействие на частоте 4.75 с амплитудой 1.5, коэффициент $\gamma = 0$. Состояние цепочек a, b, φ после 700 периодов воздействия возмущением $F_n = A\sqrt{a_n^2 + b_n^2}\, cos(\frac{\pi n}{5} + \varphi_n) cos(\omega t)$. График растянут по оси Y в 50 раз.

молекулах, даже если поток энергии, соответствующий закачке невелик. Наличие резонанса вызовет эффект значительного роста амплитуды.
Ещё раз оговоримся, механизм воздействия звуковой волны на двойную спираль молекулы неизвестен. Правдоподобным представляется тот факт, что оно происходит за счёт зацепления движущейся водой молекулы ДНК. Предположим, что внешние слои молекулы двигаются с той же скоростью, что и жидкость. Это точный аналог условия прилипания классической гидродинамики.

5 Выводы

В настоящее время топология реальной ДНК изучена слабо. В данной работе для простой модели ДНК численно была установлена возможность параметрического резонанса по отношению к внешнему возбуждению. Образующиеся при таком высокочастотном воздействии моды оказываются локализованными. Условиями локализации являются случайное распределение элементов A, T, G, C (с физической точки зрения) и достаточная длина самой молекулы. Размер области локализации схож с размерами участков пузырьковых дефектов в ДНК, [22]. Оценка порядка частот позволяет сказать, что воздействие происходит на гиперзвуковых частотах, а потому, вероятно, гиперзвуковой резонанс может быть начальной причиной возникновения пузырьковых дефектов в ДНК. Была рассмотрена зависимость резонансных режимов от диссипации. Она линейна вне некоторой окрестности нуля диссипации, а в самой окрестности ведёт себя степенным образом, что обусловлено условием Рэлея (6). При частоте, отличной от резонансной, наблюдается характерный порог амплитуды, ниже которого резонанс не наблюдается. Здесь, однако, следует оговориться. На гиперзвуковых частотах перестают выполняться уравнения Навье-Стокса. Потому, поднимая вопрос о диссипации в воде на таких частотах, следует быть очень осторожным. С геометрической точки зрения резонансные явления приводят к появлению дырок, что может быть причиной увеличения энтропии, а, следовательно, и более выгодного энергетически состояния молекулы. Так, в слабых соляных растворах, например, образование дырок происходит при комнатной температуре. При этом, колебания в самой воде происходят на гиперзвуковых частотах, а потому вода может являться источником внешнего воздействия в большей мере, нежели причиной затухания колебаний. Воздействие на гиперзвуковых частотах через механизм параметрического резонанса вполне вписывается в такую картину, что ДНК обычно наблюдают в растворе, в то время как раствор является хорошим проводником гиперзвука. Таким образом гиперзвуковой источник малой мощности может, при условии попадания в нужную частоту вызвать дефект в ДНК. При неточном попадании можно увеличить мощность воздействия, это должно привести к похожим результатам. При чрезмерном повышении

могут возникнуть другие эффекты, например кавитация. Однако, пока об этом судить сложно, поскольку гиперзвук исследован слабо. В любом случае, гиперзвук представляет собой крайне интересный объект исследования, который может иметь важные приложения.

Список литературы

[1] J.D. Watson and F.H.C. Crick, Nature 171, 737 (1953).

[2] F.H.C. Crick and J.D. Watson, Brookhaven Symp. Biol. No. 8 (1956).

[3] F.H.C. Crick, J.Mol. Biol., 19, 548 (1966).

[4] M.D. Topal and J.R. Fresco, Nature 263, 285 - 289 (1976).

[5] A. Hanke and R. Metzler, J.Phys.A: Math.Gen. 36, L473, (2003).

[6] D.Hennig, Eur. Phys. J. B 30, 211218 (2002)

[7] D.Hennig, Eur. Phys. J. B 37, 391397 (2004)

[8] J.A.D. Wattis, Phil.Trans.R.Soc. Lon., A 362, 1461 (2004).

[9] I.M. Lifschits, S.A.Gredeskul, and L.A.Pastur, Introduction into the theory of random media, "Nauka", (1982).

[10] I.M.Lifshits and L.N. Rosenzweig, Izv.Ac.Nauk USSR Ser.Fyz. **12**, 667 (1948).

[11] I.M.Lifshits, Journ. of Phys. (USSR)7, 211, 249 (1943).

[12] I.M.Lifshits, ZhETF, 17, 1017, 1076 (1947).

[13] I.M.Lifshits, Nuovo Cimento Suppl., 10, 3, 716 (1956).

[14] A.A. Maradudin, E.W. Montroll, and G.H. Weiss, Theory of Lattice Dynamics in the Harmonic Approximation, Academic Press, New York and London (1963).

[15] J.M. Ziman, Models of Disorder, Ch.8, Cambridge University Press, Cambridge (1979).

[16] G.S.Manning, Biopolymers, **22**, 689 (1983).

[17] M.Guéron, M.Kochoyan, and J.L.Leroy, Nature **328**, 89 (1987).

[18] M.E.El Hassan and C.R.Calladine, J.Mol.Biol. 251, 648 (1995).

[19] J.L.Leroy, X.Gao, V.Misra, M.Guéron, and D.J.Patel, Biochemistry, **31**, 1407 (1992).

[20] M.Guéron and J.L.Leroy, Methods Enzymol. **261**, 383 (1995).

[21] G.Bonnet, O.Krichevsky, and A.Libchaber, Proc.Natl.Acad.Sci. USA 95, 8602(1998).

[22] G.Altan-Bonnet, A.Libchaber, and O.Krichevsky, Phys.Rev.Lett. **90**, 138101 (2003).

[23] A.A. Maradudin, E.W. Montroll, and G.H. Weiss, Theory of Lattice Dynamics in the Harmonic Approximation, Academic Press, New York and London (1963).

[24] V.L.Golo, arXiv:q-bio/0309007v1.

[25] V.Blinov, V.L.Golo, arXiv:q-bio/0309007v1.

[26] P.Dean, Phys.Rev. **44**, №2, 127–168 (1972).

[27] C.R.Hill, G. ter Haar, Ultrasound Medical EURO SERIES 25 (chapter 6).

[28] R.J. Isaacs and H.P. Spielmann, J. Mol. Biol. 307, 525 (2001).

[29] J.W. Rayleigh, The Theory of Sound, vol. **I**, MacMillan, London (1926).

A Приложение: код программы.

```
//
// Structs.h Program Header
//


#include <fstream>
#include <iostream>
#include <stdlib.h>
#include <math.h>

#define CONSTS_FILE "consts.dat"
#define SYSTEM_INIT_FILE "sysinit.dat"
#define WARNING_FILE "warnings.txt"
#define RESONANCE_FILE "resparam.txt"

#define CONST_PI 3.14159265358979323846

#define WRITE_EVERY_DEF 100

#define DEF_LAMBDA 1
#define MAX_SYSTEM_SIZE 310
#define METHOD_DEF 3
#define RESONANCE_AMP_DEF 1.3
#define RESONANCE_FREQ_DEF 1.489
#define PHI_NONLINEAR_DEF 1
#define FIXED_EDGES_DEF 1
#define PHI_ONLY_DEF 0
#define ADAMSMOULTONPRES 4
#define TRAPPRES 4

#define SEARCH 1
#define COUNT 2
#define DISSIPCHECK 3
#define RESFREQ 4
#define EIGENSEARCH 5
#define BRUTEFORCE 6
#define LINES 7

#define MODE 2
using namespace std;


struct Coord
{
```

```
int N;

double a[MAX_SYSTEM_SIZE];
double b[MAX_SYSTEM_SIZE];
double p[MAX_SYSTEM_SIZE];

Coord operator+(Coord& Arg);
Coord operator-(Coord& Arg);
Coord operator*(double Arg);
Coord operator/(double Arg);
void operator+=(Coord& Arg);
void operator-=(Coord& Arg);
void operator*=(double Arg);
void operator/=(double Arg);
void operator=(double Arg);
void operator=(Coord& Arg);
};

struct State
{
Coord x;
Coord v;

State();
 State();
State operator+(State& Arg);
State operator-(State& Arg);
State operator*(double Arg);
State operator/(double Arg);
void operator+=(State& Arg);
void operator-=(State& Arg);
void operator*=(double Arg);
void operator/=(double Arg);
void operator=(double Arg);
void operator=(State& Arg);
};

class Parameters
{
public:
double K[MAX_SYSTEM_SIZE];
double M[MAX_SYSTEM_SIZE];
double E[MAX_SYSTEM_SIZE];
double I[MAX_SYSTEM_SIZE];
double T[MAX_SYSTEM_SIZE];
double Dissipation;
```

```
Parameters();
 Parameters();
void ReadFlie(int N);
};


class PhaseSpace
{

public:
int StepCounter,
SystemSize,
WriteStep,

FIXED_EDGES,
PHI_NONLINEAR,
PHI_ONLY;

Parameters PC;

State StartState,
CurrentState,
NextState,
tempState,
Speed[3],
CurrentSpeed;


double dT,
WorkTime,
Time;

char outfileName[50];
ofstream ofOut;

ofstream ofWarn;

int METHOD;

double Energy,
lambda;

double
RESONANCE_AMP,
```

```
RESONANCE_FREQ;

PhaseSpace();
 PhaseSpace();
void InitSystem();
void WriteCurrency();
void WriteCurrency(double Freq);
void WriteCurrency(char * outfile);

void RHS(State& Arg, State& Res, double T);

double TimeInPeriods();
void FirstStep();
void ResetEdges();
void Cycle(State & Arg);

void CountEnergy();
double PhiEnergy();

void Step();

void StepVerlet();
void StepTrapecia();
void LeapFrogStep();
void StepAdamsBashforth();
void AdamsMoultonStep();

void Refresh();
void Go();
};

//
// Structs.cpp
//

#include "structs.h"

Parameters :: Parameters()
{

int i;
for (i=0;i<MAX_SYSTEM_SIZE;i++)
{
M[i] = 0;
E[i] = 0;
T[i] = 0;
```

```
K[i] = 0;
I[i] = 0;
}
}

Parameters ::  Parameters()
{


}


PhaseSpace :: PhaseSpace()
{
InitSystem();
StepCounter = -1;
CurrentSpeed = 0;
Time = 0;
}

PhaseSpace ::  PhaseSpace()
{

ofOut.close();
ofWarn.close();
}


void PhaseSpace :: Go()
{
while (Time<WorkTime)
{
Step();
}
}



double PhaseSpace :: TimeInPeriods()
{

if (RESONANCE_FREQ>0.001)
{
return (Time*RESONANCE_FREQ)/(2*CONST_PI);
}
else
```

```
{

return Time;
}
}


void PhaseSpace :: Refresh()
{
Time = 0;
CurrentState.x = StartState.x;
CurrentState.v = 0;
NextState=CurrentState;
}

void PhaseSpace :: Cycle(State & Arg)
{
Arg.x.a[0]=Arg.x.a[SystemSize];
Arg.x.b[0]=Arg.x.b[SystemSize];
Arg.x.p[0]=Arg.x.p[SystemSize];
Arg.x.a[SystemSize+1]=Arg.x.a[1];
Arg.x.b[SystemSize+1]=Arg.x.b[1];
Arg.x.p[SystemSize+1]=Arg.x.p[1];

Arg.v.a[0]=Arg.v.a[SystemSize];
Arg.v.b[0]=Arg.v.b[SystemSize];
Arg.v.p[0]=Arg.v.p[SystemSize];
Arg.v.a[SystemSize+1]=Arg.v.a[1];
Arg.v.b[SystemSize+1]=Arg.v.b[1];
Arg.v.p[SystemSize+1]=Arg.v.p[1];
}

void PhaseSpace :: ResetEdges()
{

if (FIXED_EDGES == 1)
{
CurrentState.x.a[0]=0;
CurrentState.x.b[0]=0;
CurrentState.x.p[0]=0;
CurrentState.x.a[SystemSize+1]=0;
CurrentState.x.b[SystemSize+1]=0;
CurrentState.x.p[SystemSize+1]=0;

CurrentState.v.a[0]=0;
CurrentState.v.b[0]=0;
```

```
CurrentState.v.p[0]=0;
CurrentState.v.a[SystemSize+1]=0;
CurrentState.v.b[SystemSize+1]=0;
CurrentState.v.p[SystemSize+1]=0;

Speed[0].x.a[0]=0;
Speed[0].x.b[0]=0;
Speed[0].x.p[0]=0;
Speed[0].x.a[SystemSize+1]=0;
Speed[0].x.b[SystemSize+1]=0;
Speed[0].x.p[SystemSize+1]=0;

Speed[0].v.a[0]=0;
Speed[0].v.b[0]=0;
Speed[0].v.p[0]=0;
Speed[0].v.a[SystemSize+1]=0;
Speed[0].v.b[SystemSize+1]=0;
Speed[0].v.p[SystemSize+1]=0;
}
else
{
Cycle(CurrentState);
Cycle(Speed[0]);
}
}

//
// State.cpp – State struct overload
//

#include "structs.h"

void State::operator=(double Arg)
{
x=Arg;
v=Arg;
}
void State::operator=(State& Arg)
{
Coord A = (Arg.x);
x = A;
A = (Arg.v);
v = A;
}

State State::operator+(State& Arg)
```

```
{

State fCoord;
Coord A = (Arg.x);
fCoord.x = x;
fCoord.x += A;
A = (Arg.v);
fCoord.v=v;
fCoord.v+=A;
return fCoord;
}
State State::operator-(State& Arg)
{

State fCoord;
Coord A = (Arg.x);
fCoord.x = x;
fCoord.x -= A;
A = (Arg.v);
fCoord.v=v;
fCoord.v-=A;
return fCoord;
}
State State::operator*(double Arg)
{
State fCoord;
fCoord.x=x;
fCoord *= Arg;
fCoord.v=v;
fCoord *= Arg;
return fCoord;
}

State State::operator/(double Arg)
{
State fCoord;
if (fabs(Arg)>0.0000001)
{
fCoord.x=x;
fCoord /= Arg;
fCoord.v=v;
fCoord /= Arg;
}
return fCoord;
}
```

```
void State::operator+=(State& Arg)
{

x+=Arg.x;
v+=Arg.v;
}
void State::operator-=(State& Arg)
{
x-=Arg.x;
v-=Arg.v;
}

void State::operator*=(double Arg)
{
x*=Arg;
v*=Arg;
}

void State::operator/=(double Arg)
{
x/=Arg;
v/=Arg;
}


State::State()
{
for (int i=0;i<MAX_SYSTEM_SIZE;++i)
{
x.a[i] = 0;
x.b[i] = 0;
x.p[i] = 0;
v.a[i] = 0;
v.b[i] = 0;
v.p[i] = 0;
}
}


State:: State()
{
}

//
// Coord.spp – Coord struct overload
```

```
//

#include "structs.h"

void Coord::operator=(double Arg)
{
int i;
for (i=0;i<=N+1;i++)
{
a[i]=Arg;
b[i]=Arg;
p[i]=Arg;
}

}

void Coord::operator=(Coord& Arg)
{
for (int i=0;i<=N+1;++i)
{
a[i]=Arg.a[i];
b[i]=Arg.b[i];
p[i]=Arg.p[i];
}
}

Coord Coord::operator+(Coord& Arg)
{
Coord fCoord;
for (int i=0; i<=N+1;++i)
{
fCoord.a[i] = a[i]+Arg.a[i];
fCoord.b[i] = b[i]+Arg.b[i];
fCoord.p[i] = p[i]+Arg.p[i];
}
return fCoord;
}

Coord Coord::operator-(Coord& Arg)
{

Coord fCoord;
for (int i=0;i<=N+1;++i)
{
fCoord.a[i] = a[i]-Arg.a[i];
fCoord.b[i] = b[i]-Arg.b[i];
```

```
fCoord.p[i] = p[i]-Arg.p[i];
}
return fCoord;
}

Coord Coord::operator*(double Arg)
{

Coord fCoord;
for (int i=0;i<=N+1;++i)
{
fCoord.a[i] = a[i]*Arg;
fCoord.b[i] = b[i]*Arg;
fCoord.p[i] = p[i]*Arg;
}
return fCoord;
}

Coord Coord::operator/(double Arg)
{

Coord fCoord;
for (int i=0;i<=N+1;++i)
{
fCoord.a[i] = a[i]/Arg;
fCoord.b[i] = b[i]/Arg;
fCoord.p[i] = p[i]/Arg;
}
return fCoord;
}


void Coord::operator+=(Coord& Arg)
{
for (int i=0;i<=N+1;++i)
{
a[i]+=Arg.a[i];
b[i]+=Arg.b[i];
p[i]+=Arg.p[i];
}
}

void Coord::operator-=(Coord& Arg)
{
for (int i=0;i<=N+1;++i)
{
```

```
a[i]-=Arg.a[i];
b[i]-=Arg.b[i];
p[i]-=Arg.p[i];
}
}

void Coord::operator*=(double Arg)
{
for (int i=0;i<=N+1;++i)
{
a[i]*=Arg;
b[i]*=Arg;
p[i]*=Arg;
}
}

void Coord::operator/=(double Arg)
{
for (int i=0;i<=N+1;++i)
{
a[i]/=Arg;
b[i]/=Arg;
p[i]/=Arg;
}

}


//
// Input.cpp - I/O Streams
//

#include "structs.h"

void Parameters :: ReadFlie(int N)
{
int i=0;
double asq;
ifstream ifData(CONSTS_FILE);
ifData » Dissipation » asq;
for (i=1;i<=N;i++)
{
ifData » M[i] » E[i] » T[i] » K[i] » I[i];
}
M[0] = M[N];
E[0] = E[N];
```

```
T[0] = T[N];
K[0] = K[N];
I[0] = I[N];
M[N+1] = M[1];
E[N+1] = E[1];
T[N+1] = T[1];
K[N+1] = K[1];
I[N+1] = I[1];
ifData.close();
}

void PhaseSpace :: InitSystem()
{
int i;

if (!ofWarn.is_open())
{
ofWarn.open(WARNING_FILE);
}


ifstream ifData(SYSTEM_INIT_FILE);
ifstream ifResData(RESONANCE_FILE);

ifResData » METHOD;
ifResData » RESONANCE_AMP;
ifResData » RESONANCE_FREQ;
ifResData » PHI_NONLINEAR;
ifResData » PHI_ONLY;
ifResData » lambda;
if ((RESONANCE_FREQ<0.000001)||(RESONANCE_AMP<0.000001))
{
ofWarn « "Using Standart Resonance Parametres";

RESONANCE_AMP = RESONANCE_AMP_DEF;
RESONANCE_FREQ = RESONANCE_FREQ_DEF;
PHI_NONLINEAR = PHI_NONLINEAR_DEF;
PHI_ONLY = PHI_ONLY_DEF;
METHOD = METHOD_DEF;
lambda=DEF_LAMBDA;
}

ifResData.close();

ifData » SystemSize;
ifData » dT;
```

```
ifData » WorkTime;
ifData » WriteStep;
srand(7);

if (SystemSize>MAX_SYSTEM_SIZE)
{
ofWarn « "Warning: System Size is too large [MAX 300]" « endl;
}

StartState.v.N =SystemSize;
tempState.v.N = SystemSize;
CurrentState.v.N = SystemSize;
NextState.v.N = SystemSize;
Speed[0].v.N = SystemSize;
Speed[1].v.N = SystemSize;
Speed[2].v.N = SystemSize;
CurrentSpeed.v.N = SystemSize;


StartState.x.N =SystemSize;
tempState.x.N = SystemSize;
CurrentState.x.N = SystemSize;
NextState.x.N = SystemSize;
Speed[0].x.N = SystemSize;
Speed[1].x.N = SystemSize;
Speed[2].x.N = SystemSize;
CurrentSpeed.x.N = SystemSize;


for (i=1; i<=SystemSize; i++)
{

ifData » StartState.x.a[i] » StartState.x.b[i] » StartState.x.p[i];
}
CurrentState.x = StartState.x;
ResetEdges();

PC.ReadFlie(SystemSize);

sprintf(outfileName, "m%d(%.4lf) a(%.4lf)
diss(%.5lf).dat", METHOD, dT, RESONANCE_AMP, PC.Dissipation);
ofOut.open(outfileName);
ofOut « SystemSize « endl;

ifData.close();
}
```

```
void PhaseSpace :: WriteCurrency()
{
int i;
ofOut « TimeInPeriods() « endl;
for (i=0;i<=SystemSize+1;i++)
{
ofOut « CurrentState.x.a[i] « " " « CurrentState.x.b[i]
« " " « CurrentState.x.p[i] « endl;
}

}


void PhaseSpace :: WriteCurrency(double Freq)
{
int i;
ofOut « Freq « endl;
for (i=0;i<=SystemSize+1;i++)
{
ofOut « CurrentState.x.a[i] « " " « CurrentState.x.b[i]
« " " « CurrentState.x.p[i] « endl;
}
}

void PhaseSpace :: WriteCurrency(char * outfile)
{
int i;

ofstream outbuf(outfile);
outbuf « SystemSize « endl;
outbuf « TimeInPeriods() « endl;

for (i=0;i<=SystemSize+1;i++)
{
outbuf « CurrentState.x.a[i] « " " « CurrentState.x.b[i]
« " " « CurrentState.x.p[i] « endl;
}
outbuf.close();
}


//
// Computing.cpp Intregration
//
```

```
#define sqr(A) ((A)*(A))
#include ”structs.h”

void PhaseSpace :: Step()
{
Time+=dT;
StepCounter++;
if (StepCounter==0)
{

Refresh();
FirstStep();
}

if (MODE==COUNT)
{
if (StepCounter%WriteStep==0)
{
StepCounter=0;
if (PHI_ONLY==0)
{
CountEnergy();
printf(”Full - %.9lf /n”, Energy);
}
else
{
printf(”%.9lf /n”, PhiEnergy());
}

WriteCurrency();
}
}


ResetEdges();
switch (METHOD)
{
case 1:
StepVerlet();
break;
case 2:
StepTrapecia();
break;
case 3:
LeapFrogStep();
break;
```

```
case 4:
StepAdamsBashforth();
break;
case 5:
AdamsMoultonStep();
break;

}
}

double PhaseSpace :: PhiEnergy()
{
double res=0;
int i;
for (i=1;i<=SystemSize;i++)
{
res += PC.I[i]*sqr(CurrentState.v.p[i])/2;
res += PC.T[i]*sqr(CurrentState.x.p[i+1]-CurrentState.x.p[i])/2;
}
res += PC.T[0]*sqr(CurrentState.x.p[1]-CurrentState.x.p[0])/2;
Energy = res;
return res;
}

void PhaseSpace :: CountEnergy()
{
int i;
Energy = 0;
if (PHI_ONLY==1)
{
PhiEnergy();
}
else
{
for (i=1;i<=SystemSize;i++)
{
Energy += PC.I[i]*sqr(CurrentState.v.p[i])/2;
Energy += PC.M[i]*(sqr(CurrentState.v.a[i])+sqr(CurrentState.v.b[i]))/2;
Energy += PC.E[i]*(sqr(CurrentState.x.a[i])+sqr(CurrentState.v.b[i]))/2;
}

for (i=0;i<=SystemSize;i++)
{

Energy += PC.T[i]*sqr(CurrentState.x.p[i+1]-CurrentState.x.p[i])/2;
Energy += PC.K[i]*(sqr(CurrentState.x.a[i+1]-CurrentState.x.a[i])
```

```
+sqr(CurrentState.x.b[i+1]-CurrentState.x.b[i]))/2;
Energy += PC.K[i]*((CurrentState.x.p[i+1]-CurrentState.x.p[i])
*(CurrentState.x.a[i+1]*CurrentState.x.b[i]-CurrentState.x.a[i]*CurrentState.x.b[i+1]));
} }

}



void PhaseSpace :: StepVerlet()
{
RHS(CurrentState, CurrentSpeed, Time);

NextState.x = CurrentState.x;
NextState.x *= 2;
NextState.x -= tempState.x;

CurrentSpeed.x = CurrentSpeed.v;
CurrentSpeed.x *= dT*dT;
NextState.x += CurrentSpeed.x;
NextState.v = 0;

tempState.x = CurrentState.x;
CurrentState.x = NextState.x;
}

void PhaseSpace :: StepTrapecia()
{
NextState = CurrentState;
RHS(CurrentState, CurrentSpeed, Time);
for (int i=0;i<TRAPPRES;++i)
{
RHS(NextState, tempState, Time+dT);
NextState = CurrentSpeed;
NextState += tempState;
NextState *= dT/2;
NextState += CurrentState;
}
CurrentState=NextState;
}

void PhaseSpace :: LeapFrogStep()
{
NextState = CurrentState;
RHS(CurrentState, CurrentSpeed, Time);
```

```
tempState.x = CurrentState.v;
tempState.x *= (dT);
NextState.x += tempState.x;

tempState.x = CurrentSpeed.v;
tempState.x *= (dT*dT*0.5);
NextState.x += tempState.x;

tempState.v = CurrentSpeed.v;
tempState.v *= (dT/2);
NextState.v += tempState.v;

RHS(NextState, tempState, Time+dT);

tempState.v *= dT/2;
NextState.v += tempState.v;

CurrentState = NextState;
}

void PhaseSpace :: StepAdamsBashforth()
{
double k1=(23.)/(12.),
k2=(-4.)/(3.),
k3=(5.)/(12.);

NextState = CurrentState;

tempState = Speed[0];
tempState *= dT*k1;
NextState += tempState;

tempState = Speed[1];
tempState *= dT*k2;
NextState += tempState;

tempState = Speed[2];
tempState *= dT*k3;
NextState += tempState;



Speed[2] = Speed[1];
Speed[1] = Speed[0];

CurrentState = NextState;
```

```
RHS(CurrentState, Speed[0], Time);
}

void PhaseSpace :: AdamsMoultonStep()
{

double k1=(5.)/(12.),
k2=(2.)/(3.),
k3=(-1.)/(12.);


Speed[2] = Speed[1];
Speed[1] = Speed[0];
NextState = CurrentState;
for (int i=0;i<ADAMSMOULTONPRES;i++)
{
RHS(NextState, Speed[0], Time+dT);
NextState = CurrentState;

tempState = Speed[0];
tempState *= dT*k1;
NextState += tempState;

tempState = Speed[1];
tempState *= dT*k2;
NextState += tempState;

tempState = Speed[2];
tempState *= dT*k3;
NextState += tempState;
}

CurrentState = NextState;

}

void PhaseSpace :: RHS(State& Arg, State& Res, double T)
{
int n;
if (!FIXED_EDGES)
{
Cycle(Arg);
}
for (n=1;n<=SystemSize;n++)
{
Res.v.p[n] = PC.T[n]*Arg.x.p[n+1]+PC.T[n-1]*Arg.x.p[n-1]-(PC.T[n-1]+PC.T[n])*Arg.x.p[n];
```

```
if (PHI_NONLINEAR==1)
{
Res.v.p[n] += lambda*PC.K[n]*(Arg.x.a[n+1]*Arg.x.b[n] - Arg.x.a[n]*Arg.x.b[n+1]);
Res.v.p[n] -= lambda*PC.K[n-1]*(Arg.x.a[n]*Arg.x.b[n-1]-Arg.x.a[n-1]*Arg.x.b[n]);
}

Res.v.p[n] += RESONANCE_AMP*Arg.x.p[n]*sin(RESONANCE_FREQ * T);
Res.v.p[n] -= Arg.v.p[n]*PC.Dissipation;
Res.v.p[n] /= PC.I[n];

if (!PHI_ONLY)
{
Res.v.a[n] = PC.K[n]*Arg.x.a[n+1]+PC.K[n-1]*Arg.x.a[n-1]-(PC.K[n-1]+PC.K[n])*Arg.x.a[n];
Res.v.a[n] += lambda*PC.K[n]*Arg.x.b[n+1]*(Arg.x.p[n+1]-Arg.x.p[n]);
Res.v.a[n] -= lambda*PC.K[n-1]*Arg.x.b[n-1]*(Arg.x.p[n]-Arg.x.p[n-1]);
Res.v.a[n] -= PC.E[n]*Arg.x.a[n];
Res.v.a[n] /= PC.M[n];

Res.v.b[n] = PC.K[n]*Arg.x.b[n+1]+PC.K[n-1]*Arg.x.b[n-1]-(PC.K[n-1]+PC.K[n])*Arg.x.b[n];
Res.v.b[n] -= lambda*PC.K[n]*Arg.x.a[n+1]*(Arg.x.p[n+1]- Arg.x.p[n]);
Res.v.b[n] += lambda*PC.K[n-1]*Arg.x.a[n-1]*(Arg.x.p[n]- Arg.x.p[n-1]);
Res.v.b[n] -= PC.E[n]*Arg.x.b[n];
Res.v.b[n] /= PC.M[n];
}
}
Res.x = Arg.v;
if (!FIXED_EDGES)
{
Cycle(Res);
}
}

void PhaseSpace :: FirstStep()
{
StepTrapecia();
RHS(CurrentState, Speed[2], Time);
Time+=dT;
StepTrapecia();
RHS(CurrentState, Speed[1], Time);
Time+=dT;
tempState = CurrentState;
StepTrapecia();
RHS(CurrentState, Speed[0], Time);
}
```

```
//
// Main.cpp
//

#include "structs.h"


int main()
{
PhaseSpace DNA;
if (MODE==COUNT)
{
DNA.Go();
}
}
```