

# СОДЕРЖАНИЕ

Стр.

<b>ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ .....</b>	<b>3</b>
<b>ВВЕДЕНИЕ .....</b>	<b>4</b>
<b>1 Вывод уравнений мелкой воды.....</b>	<b>6</b>
<b>2 Постановка задачи .....</b>	<b>14</b>
<b>3 Триангуляция двумерной области .....</b>	<b>16</b>
3.1 алгоритм Руперта [Jim Ruppert] .....	17
3.2 алгоритм Чу [L. Paul Chew].....	20
<b>4 Метод конечных элементов в двумерной области.....</b>	<b>21</b>
4.1 Метод взвешенных невязок .....	22
4.2 Метод конечных элементов .....	25
4.3 МКЭ для рассматриваемой задачи .....	27
4.3.1 Интегрирование функции двух переменных по произ- вольному треугольнику .....	27
4.3.2 Уравнения мелкой воды в терминах МКЭ.....	28
<b>5 Решение задачи.....</b>	<b>30</b>
5.1 Изменение формы водоёма.....	30
5.1.1 Пруд.....	30
5.1.2 Реальный водоём.....	32
5.2 Учёт наличия острова (островов) внутри водоёма .....	33
5.2.1 Пруд.....	33
5.2.2 Реальный водоём.....	33
5.3 Построение графиков .....	33
5.4 Вычисления .....	34
<b>6 Разработка базы данных для хранения информации о про- ведённых экспериментах .....</b>	<b>35</b>
6.1 Описание базы данных .....	35
6.2 Формат представления данных в БД .....	37
<b>ЗАКЛЮЧЕНИЕ .....</b>	<b>40</b>

<b>СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ.....</b>	<b>41</b>
<b>ПРИЛОЖЕНИЕ А Примеры численного решения задачи .....</b>	<b>42</b>
<b>ПРИЛОЖЕНИЕ Б Исходный код реализации.....</b>	<b>43</b>
<b>ПРИЛОЖЕНИЕ В Выкладки для МКЭ .....</b>	<b>44</b>

## **ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ**

1. МКЭ - метод конечных элементов
2. КЭ - конечный элемент

## **ВВЕДЕНИЕ**

Выполненная выпускная квалификационная работа посвящена задаче течения жидкости в озере, для которой требуется составить и решить систему дифференциальных уравнений с использованием метода конечных элементов. Предполагается, что глубина озера постоянна, а сам озеро однородно. Необходимо учесть, что озеро подвержено влиянию ветра, а так же требуется рассмотреть данную задачу с различными параметрами и проанализировать полученные результаты.

На данный момент во многих случаях не оправдывается применение более сложных математических моделей для исследования течений в прибрежных водах и озерах, чем моделей, основанных на численном решении двумерных уравнений, полученных путем применения осредненных по вертикали характеристик - уравнений мелкой воды. Трехмерные решения нецелесообразны, так как они требуют намного большего количества исходной информации и машинного времени даже с учетом современных вычислительных мощностей.

Целью представленной выпускной квалификационной работы является осуществление математического моделирования течений мелкой воды с помощью метода конечных элементов, который уже давно зарекомендовал в таких областях как механика деформируемого тела, электродинамика и конечно же гидродинамика. Данный метод является оптимальным для решения поставленной задачи так как именно он дает возможность применять достаточно гибкую разбивку рассматриваемой области и при его использовании достаточно удовлетворить лишь главным граничным условиям.

Актуальность данной работы обусловлена тем, что со второй половины 20-го века метод математического моделирования стал рабочим инструментом при изучении многих научно-технических задач, в частности, и в области гидродинамики. Основной составляющей этого метода стал являющийся вычислительных эксперимент, широкое применение которого стало возможным благодаря развитию производительности ЭВМ и численных алгоритмов. Вычислительный эксперимент позволяет получить информацию о структуре течения при относительно небольших временных, трудовых и материальных затратах. конечно, при этом соответствующая математическая модель должна быть адекватной физическому процессу.

Научная новизна работы заключается в применении МКЭ конкретно к уравнениям мелкой воды и составлении программной реализации на языке программирования Python.

В данной работе будут представлены основные определения и понятия для уравнений мелкой воды, которые позволяют составить и решить поставленную задачу, которая состоит из двух частей: аналитической и численной. Аналитически будут описаны течения жидкости при пренебрежении температурными эффектами, для которых широко используются классические уравнения Сен-Венана. После этого с помощью МКЭ будет построена и рассчитана математическая модель. Для реализации такой модели была написана программа, которая выдает результаты решения системы уравнений мелкой воды, а так же генерирует графики, которые наглядным образом отображают полученные результаты.

## 1 Вывод уравнений мелкой воды

Запишем два основных уравнения для жидкости. Это уравнение количества движения и уравнение неразрывности [1]:

$$-\frac{\partial p}{\partial x_k} + \frac{\partial \tau_{ik}}{\partial x_i} + \rho b_k = \frac{\partial}{\partial x_i}(\rho v_i v_k) + \frac{\partial}{\partial t}(\rho v_k); \quad (1.1)$$

$$\frac{D\rho}{Dt} + \rho \frac{\partial v_i}{\partial x_i} = 0 \quad (1.2)$$

Если в 1.1 и 1.2 пренебречь температурными эффектами, получим:

$$-\frac{\partial p}{\partial x_k} + \frac{\partial \tau_{ik}}{\partial x_i} + \rho b_k = \frac{D(\rho v_k)}{Dt} \quad (1.3)$$

$$\frac{\partial(\rho v_i)}{\partial t} + \frac{\partial \rho}{\partial t} = 0, \quad (1.4)$$

где  $p$  - давление, оказываемое на единицу площади поверхности воды,  $b_k$  - массовые силы, приходящиеся на единицу массы,  $v_k$  - скорость частицы воды в направлении оси  $x_k$ ,  $\tau_{ik}$  - вязкостные составляющие вектора напряжения,  $\rho$  - массовая плотность воды.

В задачах циркуляции воды сложно применять данные уравнения из-за наличия свободной поверхности, изменения границ во время приливов и отливов и вследствие большого количества переменных.

Данные трудности могут быть упрощены, в результате чего получается уравнения мелкой воды. Первое упрощение состоит в том, что уравнение количества движения в проекции на ось  $x_3$  записывается в виде:

$$-\frac{\partial p}{\partial x_3} = \rho g, \quad (1.5)$$

где массовые силы отрицательны, поскольку действуют в направлении противоположном оси  $x_3$ . При выводе формулы 1.5 пренебрегаем всеми членами

ми, которые характеризуют ускорение и соответствующими им напряжениям. Проинтегрируем выражение 1.5, и в результате получим:

$$p = \int_{x_3}^{\eta} \rho g dx_3 = \rho g(\eta - x_3) + p_a, \quad (1.6)$$

где  $p_a$  атмосферное давление на поверхности воды  $\eta$  - возвышение свободной поверхности в соответствии с рисунком 1.1.

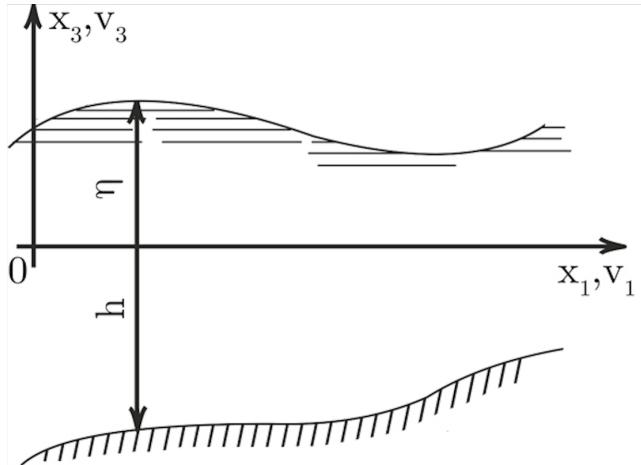


Рисунок 1.1

Оставшиеся два уравнения количества движения по направлениям  $x_1$  и  $x_2$  останутся без изменений:

$$-\frac{\partial p}{\partial x_k} + \frac{\partial \tau_{ik}}{\partial x_i} + \rho b_k = \frac{D(\rho v_k)}{Dt}. \quad (1.7)$$

При  $k = 1$  выражение 1.7 дает проекцию на ось  $x_1$ , при  $k = 2$  - на ось  $x_2$ . В выражении 1.7 величина  $v$  есть средняя скорость,  $\rho$  - переменная массовая плотность и  $\tau$  - сумма вязкостных и турбулентных напряжений.

Проинтегрируем выражения 1.4 и 1.7 по  $x_3$ . Для уравнения неразрывности это даст:

$$\int_{-h}^{\eta} \left( \frac{\partial(\rho v_i)}{\partial x_i} + \frac{\partial \rho}{\partial t} \right) dx_3 = 0, \quad (1.8)$$

где  $h$  - это глубина измеряемая от базовой поверхности (в общем случае не горизонтальной).

Определим поток  $q_k$  количества жидкости (массу жидкости, приходящуюся на единицу длины и времени):

$$q_i = \int_{-h}^{\eta} \rho v_i dx_3 = \rho \int_{-h}^{\eta} v_i dx_3, \quad (1.9)$$

Предполагается, что  $\rho(x_1, x_2)$  не зависит от  $x_3$ .

При интегрировании уравнения 1.8 необходимо использовать кинематическое условие и правило Лейбница [?] для вычисления частной производной интеграла с переменными пределами. Согласно этому правилу:

$$\frac{\partial}{\partial x_1} \int_{h_1(x_1, x_2)}^{h_2(x_1, x_2)} f(x_1, x_2, x_3) dx_3 = \int_{h_1(x_1, x_2)}^{h_2(x_1, x_2)} \frac{\partial f}{\partial x_1} dx_3 + f \Big|_{h_2} \frac{\partial h_2}{\partial x_1} + f \Big|_{h_1} \frac{\partial h_2}{\partial x_1}; \quad (1.10)$$

(аналогично для производной по  $x_2$ ).

Кинематическое соотношение для свободной поверхности можно записать как:

$$v_3 \Big|_{x_2=\eta} = \frac{D\eta}{Dt} = \frac{\partial \eta}{\partial t} + v_1 \Big|_{\eta} \frac{\partial \eta}{\partial x_1} + v_2 \Big|_{\eta} \frac{\partial \eta}{\partial x_2} \quad (1.11)$$

Применим формулы 1.9-1.11 к уравнению 1.8, и получим:

$$\frac{\partial q_i}{\partial x_i} + \frac{\partial(\rho H)}{\partial t} = 0, \quad (1.12)$$

где  $H = \eta + h$ .

Чтобы проинтегрировать уравнение количества движения 1.7 по  $x_3$ , определим мгновенные скорости  $v_1, v_2$ :

$$\begin{aligned} v_1 &= \bar{v}_1(x_1, x_2, t) + v'_1(x_1, x_2, x_3, t); \\ v_2 &= \bar{v}_2(x_1, x_2, t) + v'_2(x_1, x_2, x_3, t), \end{aligned} \quad (1.13)$$

где величина  $\bar{v}$  означает средние по вертикали скорости, а  $v'$  - отклонение от этих средних значений при различных значениях  $x_3$ .

Следовательно:

$$\langle v_k \rangle = \int_{-h}^{\eta} v_k dx_3 = \frac{1}{\rho} q_k, \quad v_k = \frac{1}{H} \langle v_k \rangle, \quad (1.14)$$

Так как  $\langle v'_k \rangle = 0$ , где знак  $\langle \rangle$  означает среднее значение стоящий внутри величины.

Будем предполагать, что массовые силы обусловлены только эффектом Кориолиса. Таким образом:

$$b_1 = \rho f v_2, \quad b_2 = -\rho f v_1. \quad (1.15)$$

Предположем, что наклоны поверхности и дна малы по сравнению с единицей, тогда составляющие внутреннего напряжения можно аппроксимировать следующим образом (в соответствии с рисунком 1.2):

$$\begin{aligned} \tau_1|_s &\approx \left\{ -\tau_{11} \frac{\partial \theta}{\partial x_1} - \tau_{12} \frac{\partial \theta}{\partial x_2} + \tau_{13} \right\}, \\ \tau_1|_b &\approx \left\{ \tau_{11} \frac{\partial h}{\partial x_1} + \tau_{12} \frac{\partial h}{\partial x_2} - \tau_{13} \right\}, \end{aligned} \quad (1.16)$$

Аналогичные значения можно выписать для величин  $\tau_2|_s$  и  $\tau_2|_b$ .

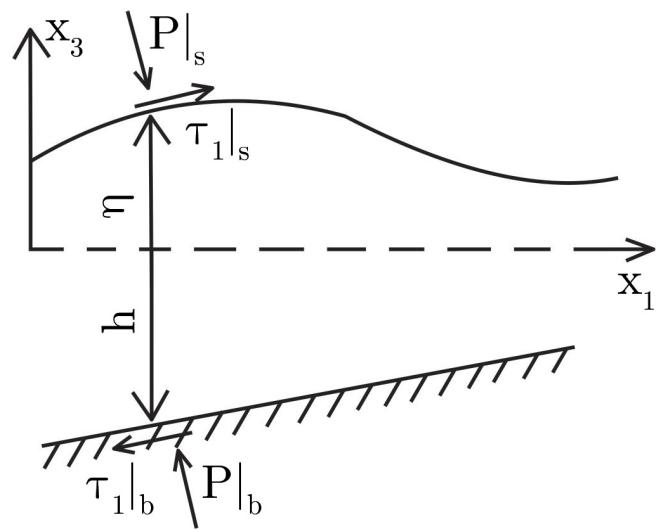


Рисунок 1.2

Величины  $\tau_2|_s$  и  $\tau_2|_b$  можно интерпретировать как компоненты внешней силы, приложенные к поверхности и ко дну.

Подставим теперь соотношения 1.13 - 1.15 в уравнение количества движения, проинтегрированные по  $x_3$  и дополнительно используем правило Лейбница и кинематическое условие. Тогда:

$$\begin{aligned} \frac{\partial q_1}{\partial t} + \frac{\partial}{\partial x_1} \left( \frac{q_1^2}{H} \right) + \frac{\partial}{\partial x_2} \left( \frac{q_1 q_2}{H} \right) &= -\frac{\partial N_p}{\partial x_2} + \frac{\partial N_{11}}{\partial x_1} + \frac{\partial N_{12}}{\partial x_2} \\ &\quad + f q_2 + p \left| \frac{\partial \eta}{\partial x_1} + \tau_1 \right|_s + p \left| \frac{\partial h}{\partial x_1} - \tau_1 \right|_b; \\ \frac{\partial q_2}{\partial t} + \frac{\partial}{\partial x_1} \left( \frac{q_1 q_2}{H} \right) + \frac{\partial}{\partial x_2} \left( \frac{q_2^2}{H} \right) &= -\frac{\partial N_p}{\partial x_1} + \frac{\partial N_{22}}{\partial x_2} + \frac{\partial N_{21}}{\partial x_1} \\ &\quad + f q_1 + p \left| \frac{\partial \eta}{\partial x_2} + \tau_2 \right|_s + p \left| \frac{\partial h}{\partial x_2} - \tau_2 \right|_b, \end{aligned} \quad (1.17)$$

где

$$\begin{aligned} N_p &= \langle p \rangle = \int_{-h}^{\eta} pdx_3 = \rho g \frac{H^2}{2} + Hp_a; \\ N_{11} &= \langle \tau_{11} \rangle - \langle pv'_1 v'_1 \rangle; \\ N_{22} &= \langle \tau_{22} \rangle - \langle pv'_2 v'_2 \rangle; \\ N_{12} &= \langle \tau_{12} \rangle - \langle pv'_1 v'_2 \rangle; \end{aligned} \quad (1.18)$$

Более того, элементы  $N_{ik}$  могут быть аппроксимированы следующими выражениями:

$$\begin{aligned} N_{11} &\approx 2\varepsilon_{11} \frac{\partial q_1}{\partial x_1}; \\ N_{22} &\approx 2\varepsilon_{22} \frac{\partial q_2}{\partial x_2}; \\ N_{12} &\approx \varepsilon_{12} \left( \frac{\partial q_2}{\partial x_1} + \frac{\partial q_1}{\partial x_2} \right); \end{aligned} \quad (1.19)$$

где  $\varepsilon_{ik}$  обобщенные коэффициенты вихревой вязкости. Для изотропного характера течения  $\varepsilon_{11} = \varepsilon_{22} = \varepsilon_{12} = \varepsilon$ .

Касательные напряжения на дне обычно определяются соотношениями:

$$\begin{aligned}\tau_1 \Big|_b &= \frac{g}{c^2} \frac{1}{\rho} \frac{q_1 \sqrt{(q_1^2 + q_2^2)}}{H^2}; \\ \tau_2 \Big|_b &= \frac{g}{c^2} \frac{1}{\rho} \frac{q_2 \sqrt{(q_1^2 + q_2^2)}}{H^2};\end{aligned}\quad (1.20)$$

где  $g$  - ускорение силы тяжести,  $\gamma$  - коэффициент трения,  $\rho$  - плотность воды.

Составляющие напряжения трения на поверхности воды обычно обусловлены действием ветра и могут быть найдены по формулам:

$$\begin{aligned}\tau_1 \Big|_s &= \gamma^2 \rho_a W^2 \cos(\Theta); \\ \tau_2 \Big|_s &= \gamma^2 \rho_a W^2 \sin(\Theta);\end{aligned}\quad (1.21)$$

где  $\gamma^2$  - коэффициент ветрового напряжения,  $\rho_a$  - плотность воздуха,  $W$  - скорость ветра,  $\theta$  - угол между осью  $x_1$  и направлением ветра.

Уравнения 1.17 перепишем в виде:

$$\begin{aligned}\frac{\partial q_1}{\partial t} + \frac{\partial}{\partial x_1} \left( \frac{q_1^2}{H} \right) + \frac{\partial}{\partial x_2} \left( \frac{q_1 q_2}{H} \right) &= \frac{\partial}{\partial x_1} (N_{11} - N_p) + \frac{\partial N_{12}}{\partial x_2} + B_1; \\ \frac{\partial q_2}{\partial t} + \frac{\partial}{\partial x_1} \left( \frac{q_1 q_2}{H} \right) + \frac{\partial}{\partial x_2} \left( \frac{q_2^2}{H} \right) &= \frac{\partial}{\partial x_2} (N_{22} - N_p) + \frac{\partial N_{12}}{\partial x_1} + B_2,\end{aligned}\quad (1.22)$$

где

$$\begin{aligned}B_1 &= f q_2 + \gamma^2 \rho_a W^2 \cos(\Theta) - \left( \frac{g}{c^2} \right) \frac{1}{\rho} \frac{q_1 \sqrt{(q_1^2 + q_2^2)}}{H^2} + p_a \frac{\partial H}{\partial x_1} + \rho g H \frac{\partial h}{\partial x_1}; \\ B_2 &= -f q_1 + \gamma^2 \rho_a W^2 \sin(\Theta) - \left( \frac{g}{c^2} \right) \frac{1}{\rho} \frac{q_2 \sqrt{(q_1^2 + q_2^2)}}{H^2} + p_a \frac{\partial H}{\partial x_2} + \rho g H \frac{\partial h}{\partial x_2}.\end{aligned}\quad (1.23)$$

Для решения окончательной системы уравнений 1.22, дополненных условием 1.12 необходимо установить требуемые граничные условия. Будем считать, что граница  $S$  состоит из двух частей: твердой границы  $S_1$  и жидкой

$S_2$ , представляющей границу рассматриваемого водоема с открытым морем в соответствии с рисунком 1.3.

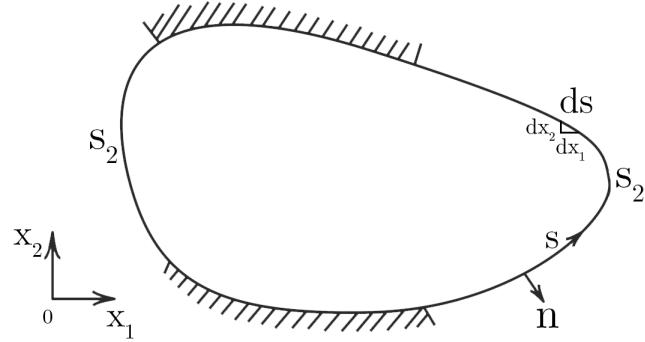


Рисунок 1.3

В системе координат  $s - n$  связанной с границей течения, расход массы жидкости можно записать через  $q_s$  и  $q_n$ :

$$\begin{aligned} q_n &= \int_{-h}^{\eta} \rho v_n dx_3 = \alpha_{n1} q_1 + \alpha_{n2} q_2; \\ q_s &= \int_{-h}^{\eta} \rho v_s dx_3 = -\alpha_{n2} q_1 + \alpha_{n1} q_2; \end{aligned} \quad (1.24)$$

где  $\alpha_{n1} = \cos(\bar{n}, x_1); \alpha_{n2} = \cos(\bar{n}, x_2)$ .

Для установления результирующих сил можно воспользоваться формулами:

$$\begin{aligned} N_{n1} &= \alpha_{n1}(N_{11} - N_p) + \alpha_{n2}N_{12}; \\ N_{n2} &= \alpha_{n1}N_{12} + \alpha_{n2}(N_{22} - N_p). \end{aligned} \quad (1.25)$$

По значениями  $N_{n1}$  и  $N_{n2}$  определяется нормальная и касательная составляющая результирующей силы для наклонной площадки:

$$\begin{aligned} N_{nn} &= \alpha_{n1}N_{n1} + \alpha_{n2}N_{n2}; \\ N_{ns} &= \alpha_{n2}N_{n1} + \alpha_{n1}N_{n2}. \end{aligned} \quad (1.26)$$

Если же в исследуемую акваторию впадает река, то на  $S_1$ :

$$\begin{aligned} q_n &= \overline{q_n} = |q|; \\ q_s &= 0. \end{aligned} \quad (1.27)$$

где  $|q|$  - поток втекающей реки.

На жидкой границе  $S_2$  необходимо задать нормальные и касательные силы:

$$\begin{aligned} N_{nn} &= \overline{N_{nn}}; \\ N_{ns} &= \overline{N_{ns}}. \end{aligned} \quad (1.28)$$

Но так как слагаемыми, учитывающими вихревую вязкость в уравнении 1.22 можно пренебречь, то касательные силы или скорости не могут быть заданы. Таким образом граничные условия сводятся к следующим:

$$\begin{aligned} q_n &= 0 \text{ или } q_n = \overline{q_n} \text{ на } S_1 \\ N_{nn} &= \overline{N_{nn}} = -N_p \text{ на } S_2. \end{aligned} \quad (1.29)$$

## 2 Постановка задачи

Пусть дана система состоящая из уравнений мелкой воды и уравнения неразрывности

$$\begin{cases} \frac{\partial q_i}{\partial x_i} + \frac{\partial(\rho H)}{\partial t} = 0 \\ \frac{\partial q_1}{\partial t} + \frac{\partial}{\partial x_1} \left( \frac{q_1^2}{H} \right) + \frac{\partial}{\partial x_2} \left( \frac{q_1 q_2}{H} \right) = \frac{\partial}{\partial x_1} (N_{11} - N_p) + \frac{\partial N_{12}}{\partial x_2} + B_1 \\ \frac{\partial q_2}{\partial t} + \frac{\partial}{\partial x_1} \left( \frac{q_1 q_2}{H} \right) + \frac{\partial}{\partial x_2} \left( \frac{q_2^2}{H} \right) = \frac{\partial}{\partial x_2} (N_{22} - N_p) + \frac{\partial N_{12}}{\partial x_1} + B_2 \end{cases}$$

где

$$B_1 = f q_2 + \gamma^2 \rho_a W^2 \cos(\Theta) - \left( \frac{g}{c^2} \right) \frac{1}{\rho} \frac{q_1 \sqrt{(q_1^2 + q_2^2)}}{H^2} + p_a \frac{\partial H}{\partial x_1} + \rho g H \frac{\partial h}{\partial x_1}; \quad (2.1)$$

$$B_2 = -f q_1 + \gamma^2 \rho_a W^2 \sin(\Theta) - \left( \frac{g}{c^2} \right) \frac{1}{\rho} \frac{q_2 \sqrt{(q_1^2 + q_2^2)}}{H^2} + p_a \frac{\partial H}{\partial x_2} + \rho g H \frac{\partial h}{\partial x_2}. \quad (2.2)$$

В системе задействованы следующие переменные физические величины:

$\rho$ - плотность воды	1000 [кг/м <sup>3</sup> ]
$c$ - коэффициент трения Шэзи	10 [м <sup>1/2</sup> с <sup>-1</sup> ]
$g$ - ускорение свободного падения	9,832 [м/с <sup>2</sup> ]
$\gamma^2$ - коэффициент ветрового напряжения	0.002
$p_a$ - атмосферное давление на поверхности воды	10 <sup>5</sup> [Па]
$\rho_a$ - плотность воздуха	1,2754 [кг/м <sup>3</sup> ]
$f$ - сила Кориолиса	0.973 · 10 <sup>-4</sup> [1/c]

Так же в системе используются следующие переменные

$W$ - скорость ветра	[м/с]
$\theta$ - угол между $x_1$ и направлением ветра	
$h$ - возвышение свободной поверхности	[м]

Для данной системы требуется написать программную реализацию решения с помощью метода конечных элементов. Реализация должна работать с любой произвольно заданной областью. Результаты должны быть получены в том числе и для реально существующих озер.

### 3 Триангуляция двумерной области

В геометрии триангуляция дискретного множества точек  $P \subset \mathbb{R}^{n+1}$  в наиболее общем значении — это разбиение выпуклой оболочки некоторого набора точек, которое представляет собой планарный граф. Одна из фигур разбиения является выпуклой оболочкой разбиваемого множества, а остальные — симплексами — геометрическими фигурами, которые являются  $n$ -мерным обобщением треугольника. В любой триангуляции ( $T$ ) формально должны выполняться следующие свойства:

1. любые два симплекса в  $T$  пересекаются в общей грани ребра или вершины, или вообще не пересекаются;
2. множество точек, являющихся вершинами симплексов разбиения, совпадает с множеством  $P$ ;
3. нельзя добавить ни одного нового ребра в граф без нарушения планарности.

Одно и то же множество можно триангулировать разными способами. Триангуляция дает тем лучшую аппроксимацию, чем больше её минимальный угол, при этом формируемые симплексы стремятся к равноугольности. Очень важна максимизация минимального угла в вычислительных задачах, когда точность производимых вычислений очень сильно зависит от размера минимального угла триангуляции. Наилучшей в этом смысле триангуляцией является триангуляция Делоне. Простейшим способом её построения является инкрементальный алгоритм, работающий за  $o(n^2)$  операций, но он не поддерживает вырожденные случаи, когда 4 точки из множества лежат на одной окружности: в этом случае триангуляция Делоне не уникальна, и способов разбиения существует несколько, но минимальные углы этих триангуляций равны. Иногда даже минимальный угол триангуляции Делоне оказывается слишком малым для устойчивой работы использующего её алгоритма, и тогда можно произвести улучшение, используя метод Рапперта [J. Ruppert]. При этом будут добавлены новые вершины триангуляции, а так же образованы дополнительные треугольники. Стабильность численного алгоритма (метода конечных элементов, к примеру) может возрасти многократно за счет появления нижней границы для углов.

### 3.1 алгоритм Руперта [Jim Ruppert]

Алгоритм Руперта [J. Ruppert], или же усовершенствованный алгоритм Делоне довольно новый, он был опубликован в 1994 году. Данный алгоритм адаптирован для МКЭ, а это значит, что он удовлетворяет всем требованиям, которые предъявляются к конечно-элементным сеткам, а именно:

1. треугольники не должны быть сильно вытянутыми, так как наличие таких треугольников отрицательно сказывается на точности результатов расчета;
2. должна быть учтена геометрия рассматриваемой области и в местах со сложной геометрией сетка должна сгущаться.

Если говорить точнее, алгоритм Рапперта [J. Ruppert] не является алгоритмом генерации сеток. Он является лишь алгоритмом улучшения качества сетки, от чего и произошло название. В общем случае входными данными для него будут являться геометрия конструкции в виде замкнутых полигонов и первичное разбиение конструкции на треугольники.

При описании алгоритма Рапперта [J. Ruppert] следует ввести следующую терминологию:

- Элемент по смыслу совпадает с конечным элементом. Это элементарная часть пространства на множество которых мы хотим разбить более сложную область этого пространства.
- Узел — точка в которой сходятся грани и соприкасаются несколько элементов. В отличие от вершин элементов узел общий для всех соприкасающихся в одной точке элементов.
- Сегмент — это отрезок, соединяющий две соседние точки лежащие на границе области разбиения, другими словами этот отрезок принадлежит контурам области.
- Грань — это отрезок по которому граничат два соприкасающихся треугольника.
- Включенная точка — любая вершина текущей сетки, которая находится внутри окружности, радиусом которой является любой сегмент.
- "Неправильный треугольник" — треугольник неудовлетворяющий глобальным условиям, которые накладываются на генерируемую сетку.

- Вытянутый треугольник — треугольник, имеющий одну сторону, намного отличающуюся от других двух. То есть треугольник слишком вытянут вдоль некоторой прямой.

Алгоритм опирается на две базовые процедуры:

1. Разбиение неправильного треугольника с введением нового узла
  - (a) Вычисляются координаты центра окружности, описанной вокруг треугольника, подлежащего разбиению
  - (b) В найденную точку добавляется новый узел
  - (c) Удаляется треугольник, подлежащий разбиению и прилегающие и добавляются новые в соответствии с рисунком 3.1
2. Разбиение сегмента, принадлежащего границе области разбиения с введением нового узла.
  - (a) Если в окружность, диаметром которой является сегмент, принадлежащий границе области разбиения попадает точка, не принадлежащая этому сегменту, то сегмент делится на две части;
  - (b) Удаляется треугольник, которому принадлежал первоначальный сегмент и добавляются два новых треугольника в соответствии с рисунком 3.2.

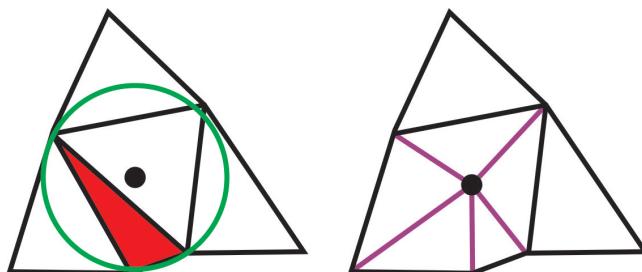


Рисунок 3.1 — Разбиение "неправильного" треугольника

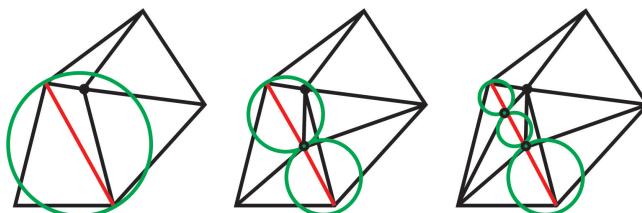


Рисунок 3.2 — Разбиение "неправильного" сегмента посередине и добавление двух новых треугольников

Принцип работы алгоритма состоит в цикле определения 'неправильных' треугольников. В каждом треугольнике сетки определяется минимальный угол и, затем, сравниваются минимальные углы для всех треугольников с целью нахождения минимального угла сетки. Другими словами, минимальный угол в сетке равен минимально возможному углу, образованному двумя сегментами, либо двумя гранями, либо сегментом и гранью, имеющими общий узел.

Математически доказано, что критерий минимального угла автоматически обеспечивает сгущение сетки вблизи мелких подробностей: отверстий, углов и вырезов.

Все описанные процедуры выполняются в определенной последовательности, они имеют разный приоритет. Общая схема работы алгоритма следующая:

1. Производится поиск включенной точки перебором всех точек и сегментов (только сегментов, но не граней).
2. Если такая точка найдена то над сегментом, который включает ее выполняется процедура деления сегмента пополам и производится возврат на шаг 1. Если включенных точек не было найдено, то алгоритм переходит на следующий шаг.
3. Производится поиск треугольника с минимальным углом.
4. Если минимальный угол меньше заданного параметра, то над треугольником, его содержащем производится процедура деления и производится возврат на шаг 1. В противном случае происходит переход на следующий шаг.
5. . Производится поиск треугольника максимальной площадью.
6. Если максимальная площадь больше заданного параметра, то над таким треугольником производится процедура деления и производится возврат на шаг 1. В противном случае происходит переход на следующий шаг.
7. Все условия удовлетворены. Конец работы алгоритма

Автор в своей работе [2] приводит строгие математические доказательства, гарантирующие правильную работу алгоритма при минимальном угле  $\alpha < 20^\circ$ . Данный алгоритм имеет ряд теоретических и практических пре-

имуществ для генерации сетки. Для не острого ввода и минимального угла порога около 20.70 градусов алгоритм гарантированно завершает работу и создает сетку оптимального размера с постоянным коэффициентом, а также дает теоретически подтвержденные гарантии качества получаемой с его помощью сетки.

### 3.2 алгоритм Чу [L. Paul Chew]

Для МКЭ очень хорошо иметь как можно более мелкие сетки конечных элементов и в некоторых случаях, в алгоритме Рапперда [J. Ruppert] получаются слишком большие сегменты. Для того, чтобы этого избежать был разработан алгоритм Чу [L. Chew], в котором сетка получается мельче за счет того, что он более консервативен в отношении разбиения на сегменты, когда граница угла меньше 30 градусов. Главное преимущество данного алгоритма по сравнению с алгоритмом Рапперда [J. Ruppert] состоит в том, что КЭ в сетке получаются с углом до 28.6 градусов, что является несомненным плюсом, так как элементы в сетке получаются приблизительно одинаковые. Алгоритм построен на более сложной теоретической базе [3] и главное его отличие от алгоритма Рапперта в том, что Рапперт только гарантирует хорошее разбиение и угол меньший, чем 20.7, а Чу обязательно даст хорошее разбиение, правда с углом до 26.5. На практике данные алгоритмы дают примерно одинаковые разбиения, но алгоритм Чу является более популярным среди разработчиков математических пакетов.

## 4 Метод конечных элементов в двумерной области

Метод конечных элементов [4,5] - численный метод решения дифференциальных уравнений с частными производными, возникающими при решении задач прикладной физики. В его основе лежат две главные идеи: дискретизация исследуемого объекта и кусочно-элементная аппроксимация исследуемых функций(в физической интерпретации - температуры, давления, перемещения и т.д.).

Первая идея состоит в том, что область, в которой ищется решение дифференциальных уравнений, разбивается на конечное количество элементов. В каждом из элементов произвольно выбирается вид аппроксимирующей функции. Вне своего элемента аппроксимирующая функция равна нулю. Значения функций в узлах являются решением задачи и заранее неизвестны. Коэффициенты аппроксимирующих функций обычно ищутся из условия равенства значения соседних функций на границах между элементами (в узлах). Затем эти коэффициенты выражаются через значения функций в узлах элементов. Составляется система линейных алгебраических уравнений. Количество уравнений равно количеству неизвестных значений в узлах, на которых ищется решение исходной системы, прямо пропорционально количеству элементов. Так как каждый из элементов связан с ограниченным количеством соседних, система линейных алгебраических уравнений имеет разрежённый вид, что существенно упрощает её решение.

Если говорить в матричных терминах, то собираются так называемые матрицы жёсткости и масс. Далее на эти матрицы накладываются граничные условия (например, при условиях Неймана в матрицах не меняется ничего, а при условиях Дирихле из матриц вычёркиваются строки и столбцы, соответствующие граничным узлам, так как в силу краевых условий значение соответствующих компонент решения известно). Затем собирается СЛАУ и решается.

Основное отличие МКЭ от классических алгоритмов вариационных принципов и методов невязок заключается в выборе базисных функций. Они берутся в виде кусочно-непрерывных функций, которые обращаются в ноль всюду, кроме ограниченных подобластей, являющихся конечными элемента-

ми. Это в свою очередь ведет к разреженной структуре матрицы коэффициентов разрешающей системы уравнений.

Главные достоинства МКЭ состоят в следующем:

1. исследуемые объекты могут иметь любую форму и различную физическую природу – твёрдые тела, жидкости, газы, электромагнитные среды
2. конечные элементы могут иметь различную криволинейную форму и различные размеры
3. можно исследовать однородные и неоднородные, изотропные и анизотропные тела с линейными и нелинейными свойствами
4. можно решать как стационарные, так и нестационарные задачи
5. можно моделировать любые граничные условия;
6. вычислительный алгоритм, представленный в матричной форме, формально единообразен для различных физических задач и для задач различной размерности. Это удобно для вычисления на ЭВМ, так как методология не меняется и фактически используется единая программа численного решения
7. на одной и той же сетке конечных элементов можно решать различные физические задачи, что облегчает анализ связанных между собой задач
8. разрешающая система уравнений имеет разреженную симметричную ленточную матрицу жёсткости, что ускоряет вычислительный процесс на ЭВМ

Перед тем как рассматривать метод конечных элементов, рассмотрим метод взвешенных невязок, который является частным случаем метода конечных элементов с одним элементом.

#### **4.1 Метод взвешенных невязок**

Рассмотрим дифференциальное уравнение вида

$$\mathcal{A} = \mathcal{L}\phi - p = 0 \quad (4.1)$$

в некоторой области  $\Omega$ .

Здесь  $\mathcal{L}$  - некоторый линейный дифференциальный оператор, а  $r$  не зависит от неизвестной функции  $\phi$ .

Решение уравнения (2.1) должно удовлетворять условию

$$\mathcal{B} = \mathcal{M}\phi + r = 0 \quad (4.2)$$

На замкнутой кривой  $\Gamma$ , ограничивающей область  $\Omega$ .

Здесь  $\mathcal{M}$  - соответствующий линейный дифференциальный оператор, а  $r$  не зависит от неизвестной функции  $\phi$ .

Построим аппроксимацию для решения  $\phi$ , которая на граничной кривой  $\Gamma$  принимает те же значения, что и  $\phi$ . Если найти некоторую функцию  $\psi$ , принимающую одинаковые с  $\phi$  значения на  $\Gamma$ , т.е.  $\psi|_{\Gamma} = \phi|_{\Gamma}$ , и ввести систему линейно независимых базисных функций  $\{N_m; m = 1, 2, \dots, N\}$ , то на  $\Omega$  можно предложить следующую аппроксимацию  $\hat{\phi}$  для  $\phi$ :

$$\phi \approx \hat{\phi} = \psi + \sum_{m=1}^M a_m N_m, \quad (4.3)$$

где  $a_m, m = \overline{1, M}$  - некоторые параметры, вычисляемые так, чтобы получить хорошее приближение, а функция  $\psi$  и базисные функции  $N_m$  выбраны таким образом, что

$$\mathcal{M}\psi = -r, \quad \mathcal{M}N_m = 0, \quad m = \overline{1, M} \text{ на } \Gamma, \quad (4.4)$$

и поэтому  $\phi$  автоматически удовлетворяет краевым условиям (2.2) при произвольных коэффициентах  $a_m$ . Отметим, что система базисных функций (которые также называют функциями формы) должна быть выбрана так, чтобы гарантировать улучшение аппроксимации при возрастании числа базисных функций для  $M$ .

Для того, чтобы найти аппроксимации производных от  $\phi$ , продифференцируем (2.3), получим:

$$\begin{aligned}\phi &\approx \hat{\phi} = \psi + \sum_{m=1}^M a_m N_m, \\ \frac{\partial \phi}{\partial x} &\approx \frac{\partial \hat{\phi}}{\partial x} = \frac{\partial \psi}{\partial x} + \sum_{m=1}^M a_m \frac{\partial N_m}{\partial x}, \\ \frac{\partial^2 \phi}{\partial x^2} &\approx \frac{\partial^2 \hat{\phi}}{\partial x^2} = \frac{\partial^2 \psi}{\partial x^2} + \sum_{m=1}^M a_m \frac{\partial^2 N_m}{\partial x^2},\end{aligned}$$

и т.д.

Так как построенное разложение (2.3) удовлетворяет краевым условиям (2.2), то для получения аппроксимации искомой функции  $\phi$  осталось гарантировать, что  $\hat{\phi}$  - приближённое решение уравнения (2.1). Для этого вначале введём погрешность или невязку  $R_\Omega$  в аппроксимации, определяемую по формуле

$$R_\Omega = A(\hat{\phi}) = \mathcal{L}\hat{\phi} + p = \mathcal{L}\psi + \left( \sum_{m=1}^M a_m \mathcal{L}N_m \right) + p. \quad (4.5)$$

Чтобы уменьшить эту невязку, потребуем равенства нулю соответствующего числа интегралов от погрешности [6], взятых с различными весами, т.е.

$$\int_{\Omega} W_l R_\Omega d\Omega = \int_{\Omega} W_l \left\{ \mathcal{L}\psi + \left( \sum_{m=1}^M a_m \mathcal{L}N_m \right) + p \right\} d\Omega = 0; \quad l = \overline{1, M}; \quad (4.6)$$

где  $\{W_l : l = 1, 2, \dots, M\}$  - это множество линейно независимых весовых (или пробных) функций.

Таким образом, система уравнений метода взвешенных невязок (2.6) сводится к системе линейных алгебраических уравнений для неизвестных коэффициентов  $a_m$ , которую можно записать как:

$$Ka = f, \quad (4.7)$$

где

$$a = (a_1, a_2, \dots, a_M)^T, \quad (4.8)$$

$$K_{lm} = \int_{\Omega} W_l \mathcal{L} N_m d\Omega, \quad l, m = \overline{1, M}; \quad (4.9)$$

$$f_l = - \int_{\Omega} W_l p d\Omega - \int_{\Omega} W_l \mathcal{L} \psi d\Omega \quad l, m = \overline{1, M}; \quad (4.10)$$

Вычислив элементы матрицы  $K$  и столбца свободных членов  $f$  и решив затем полученную систему, мы найдем  $a_m, m = \overline{1, M}$ ; и тем самым закончим процесс построения приближенного решения (2.1).

## 4.2 Метод конечных элементов

В методе взвешенных невязок предполагалось, что базисные функции  $N_m$  определены одним выражением на всей области  $\Omega$ . При этом интегралы в аппроксимирующем уравнении (2.5) вычисляются по всей области.

С другой стороны, область  $\Omega$  можно разбить на несколько непересекающихся подобластей (конечных элементов  $\Omega^e$ ), которые могут иметь различную форму и размеры. В результате разбивки создается сетка из границ элементов:

$$\Omega = \bigcup_{e=1}^E \Omega^e$$

Пересечение границ областей  $\Omega^e$  образуют узлы. Выбор типа и размера КЭ напрямую зависит от рассматриваемой задачи.

После того, как мы разбили рассматриваемую область  $\Omega$  сеткой конечных элементов можно аппроксимировать решение уравнения отдельно для каж-

дой подобласти  $\Omega^e$ , где базисные функции могут быть определены различным образом для каждой из подобластей. В этом случае определённые интегралы, входящие в аппроксимирующие уравнения, можно посчитать просуммировав вклады по каждому элементу:

$$\begin{aligned} \int_{\Omega} W_l R_{\Omega} d\Omega &= \sum_{e=1}^E \int_{\Omega^e} W_l R_{\Omega} d\Omega^e, \\ \int_{\Gamma} \overline{W}_l R_{\Gamma} d\Gamma &= \sum_{e=1}^E \int_{\Gamma^e} W_l R_{\Gamma} d\Gamma^e; \end{aligned} \quad (4.11)$$

В данных интегралах:

$$\sum_{e=1}^E \Omega^e = \Omega, \quad (4.12)$$

$$\sum_{e=1}^E \Gamma^e = \Gamma; \quad (4.13)$$

где  $E$  - число подобластей, на которые разбивается область  $\Omega$ ;  $\Gamma^e$  - часть границы  $\Omega^e$ , лежащая на  $\Gamma$ .

Система уравнений метода конечных элементов сводится в аналогичной системе метода взвешенных невязок (), но за исключением того, что:

$$K_{lm} = \sum_{e=1}^E K_{lm}^e = \sum_{e=1}^E \int_{\Omega^e} W_l^e \mathcal{L} N_m^e d\Omega^e, \quad l, m = \overline{1, M}; \quad (4.14)$$

$$f_l = \sum_{e=1}^E f_l^e = \sum_{e=1}^E \left( - \int_e W_l^e p d\Omega^e - \int_{\Omega^e} W_l^e \mathcal{L} \psi d\Omega^e \right) \quad l, m = \overline{1, M}; \quad (4.15)$$

Отметим так же, что для матрицы  $K^e$  не нужно вычислять каждую компонентку, так как её компонента  $K_{lm}^e$  равна нулю, если узлы  $l$  и  $m$  не принадлежат элементу  $e$ .

Идея метода конечных элементов заключается в следующем: если подобласти имеют простую форму и базисные функции на этих подобластях

вычисляются однотипно, то решать задачу указанным выше способом становится очень просто.

## 4.3 МКЭ для рассматриваемой задачи

### 4.3.1 Интегрирование функции двух переменных по произвольному треугольнику

Интегрирование по конечному элементу является очень важной составляющей МКЭ. Так как в общем случае симплексы в разбиении являются различными, то необходимо уметь интегрировать функцию двух переменных по произвольному треугольнику. В декартовой системе координат это является достаточно сложной задачей и именно по-этому нужно осуществить переход от нее к барицентрической системе координат.

Барицентрическая система координат представляет собой систему координат, в которой расположение точки симплекса определяется как центр массы или барицентр, как правило, неравных масс, размещенных в его вершинах.

Мы можем записать декартовы координаты точки  $\mathbf{r}$  через декартовы компоненты треугольных вершин  $\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3$  где  $\mathbf{r}_i = (x_i, y_i)$  и в терминах барицентрической системы координат переход будет выглядеть следующим образом:

$$x = \lambda_1 x_1 + \lambda_2 x_2 + \lambda_3 x_3 \quad (4.16)$$

$$y = \lambda_1 y_1 + \lambda_2 y_2 + \lambda_3 y_3 \quad (4.17)$$

$$\lambda_3 = 1 - \lambda_1 - \lambda_2 \quad (4.18)$$

Вычисление двойного интеграла в барицентрических координатах тогда осуществляется посредством следующей формулы:

$$\int_{\Delta} f(\mathbf{r}) d\mathbf{r} = 2S_{\Delta} \int_0^1 \int_0^{1-\lambda_2} f(\lambda_1 \mathbf{r}_1 + \lambda_2 \mathbf{r}_2 + (1 - \lambda_1 - \lambda_2) \mathbf{r}_3) d\lambda_1 d\lambda_2 \quad (4.19)$$

### 4.3.2 Уравнения мелкой воды в терминах МКЭ

В системе можно пренебречь членами  $-fq_1$  и  $fq_2$  так как коэффициент Кориолиса очень мал и конвективными членами.

TODO: Символьные преобразования системы посредством sympy и jupyter notebooks. Отсюда вообще говоря можно взять формулы прям в формате latex, но они огромные.

Перейдем в уравнениях (1.12), (1.22)-(1.23) к конечно-элементной формулировке. Для этого запишем функции  $q_1, q_2, H$  как разложение:

$$q_1(x_1, x_2, t) = \sum_{m=1}^M a_m(x, y, t) N_m(x, y) \quad (4.20)$$

$$q_2(x_1, x_2, t) = \sum_{m=1}^M a_{m+k}(x, y, t) N_m(x, y) \quad (4.21)$$

$$H(x_1, x_2, t) = \sum_{m=1}^M a_{2m+k}(x, y, t) N_m(x, y) \quad (4.22)$$

и подставим их в исходные уравнения. После этого каждое из уравнений необходимо умножить на весовую функцию  $W_l(x_1, x_2)$  и проинтегрировать во треугольнику. Каждое из уравнений нужно разрешить относительно производной чтобы получить систему обыкновенных дифференциальных уравнений вида:

$$\left\{ \begin{array}{l} b_1 \dot{a}_1 = c_{11}a_1 + c_{12}a_2 + \cdots + c_{1m}a_m + f_1 \\ \dots \\ b_m \dot{a}_m = c_{m1}a_1 + c_{m2}a_2 + \cdots + c_{mm}a_m + f_m \\ b_{m+1} \dot{a}_{m+1} = c_{m+11}a_1 + c_{m+2}a_2 + \cdots + c_{mm}a_m + f_{m+1} \\ \dots \\ b_{m+k} \dot{a}_{m+k} = c_{m+11}a_1 + c_{m+2}a_2 + \cdots + c_{mm}a_m \\ b_{m+k+1} \dot{a}_{m+k+1} = c_{m+11}a_1 + c_{m+2}a_2 + \cdots + c_{mm}a_m \\ \dots \\ b_{2 \cdot m + k} \dot{a}_{2 \cdot m + k} = c_{m+11}a_1 + c_{m+2}a_2 + \cdots + c_{mm}a_m \end{array} \right. \quad (4.23)$$

Подробные выкладки в приложении

## 5 Решение задачи

### 5.1 Изменение формы водоёма

Для программной реализации триангулирования произвольной области был выбран Python модуль triangle (<https://www.cs.cmu.edu/~quake/triangle.html>), который реализует триангulationю как с помощью алгоритма Рапперта, так и с помощью алгоритма Чу.

Данный модуль получает на вход геометрию в формате POLY и возвращает список вершин, сегментов, пустых областей, треугольников, маркеров вершин и маркеров сегментов, которые представляют нерегулярную сетку для исходной области. Для структуризации этой информации была написана своя реализация для объекта "точка" и "треугольник". Экземпляр класса точка хранит в себе координаты  $(x, y)$  и номер точки. Экземпляр класса треугольник хранит в себе список из трех точек, а так же имеет методы для вычисления площади, базисных функций и интегрирования по данному треугольнику. Таким образом в реализации результат триангulationии представляет из себя массив экземпляров класса "треугольник".

#### 5.1.1 Пруд

В качестве самой простой геометрии был выбран квадратный пруд в соответствии с рисунком 5.1

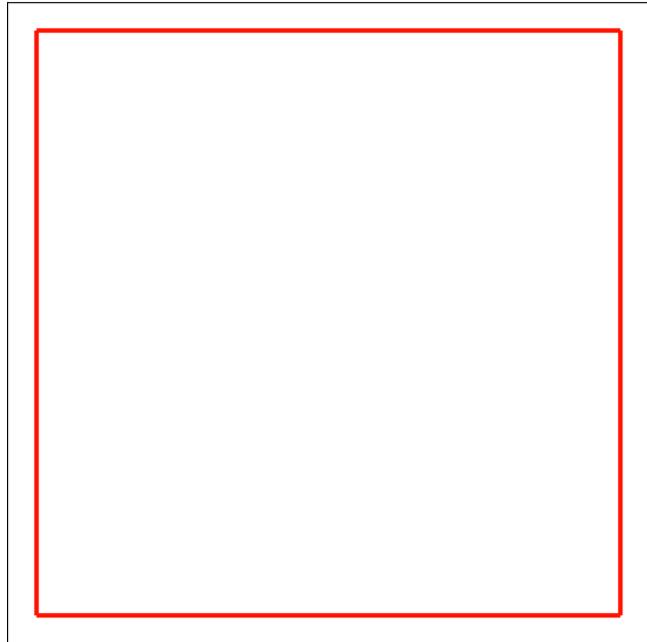


Рисунок 5.1

Для него было построено разбиение в соответствии с рисунком 5.2.

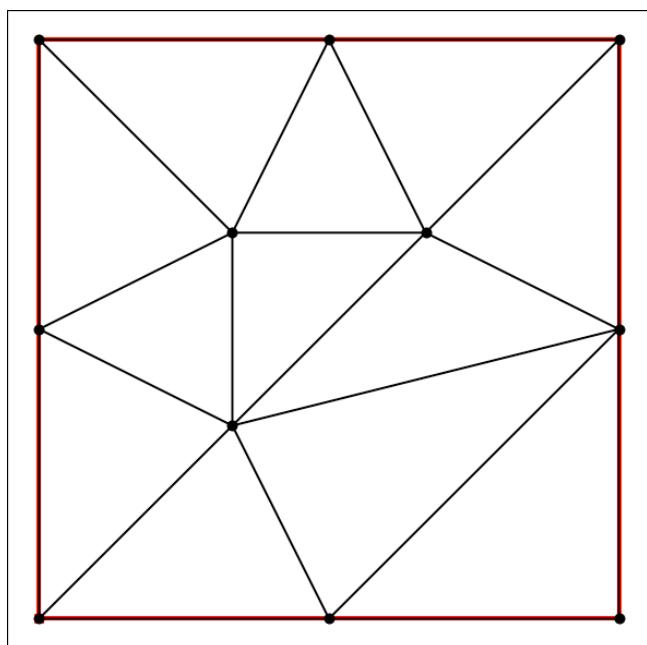


Рисунок 5.2

Как видно, сетка состоит из 12 элементов, что позволяет использовать этот пруд в качестве базового примера для отладки алгоритма МКЭ.

ВСТАВИТЬ РЕЗУЛЬТАТЫ

### 5.1.2 Реальный водоём

Данные для расчета задачи на реальной геометрии были получены на основании данных из карт OpenStreetMap. Для этого был использован сервис overpass-turbo.eu, для которого был составлен следующий запрос:

```
[out:json][timeout:25];
(
  node["name"="Elton"]({{bbox}});
  way["name"="Elton"]({{bbox}});
  relation["name"="Elton"]({{bbox}});
);
out body;
>;
out skel qt;
```

После выполнения запроса в сайта *overpass – turbo* можно скачать *geojson* файл, в котором содержатся точки запрашиваемого географического объекта и некоторая мета-информация. Так как для триангуляции требуется файл с геометрией в формате *POLY*, то была написана соответствующая функция на языке *Python*, которая конвертирует формат *geojson* в формат *POLY*.

В качестве реального водоема у которого нет островов было выбрано озеро Эльтон - солёное бессточное самосадочное озеро на севере Прикаспийской низменности. Данное озеро имеет площадь в 152 квадратных километра и наибольшую глубину до 1.5 метров. В среднем, глубина данного озера составляет 0.05 – 0.07 метра, а это значит, что для данного озера будут справедливыми уравнения мелкой воды.

ВСТАВИТЬ КАРТИНКИ

## 5.2 Учёт наличия острова (островов) внутри водоёма

### 5.2.1 Пруд

тут про пруд с двумя дырками

### 5.2.2 Реальный водоём

В качестве реального водоема у которого есть острова было выбрано озеро Верхнее. Оно находится в Северной Америке и является самым крупным и глубоким в системе Великих озёр. С физической точки зрения глубина этого озера составляет максимум 406м, а его размеры - 563 x 257 км. Получается что для можно использовать уравнения мелкой воды.

## 5.3 Построение графиков

Для численных экспериментов были построены графики для функций  $q_1$ ,  $q_2$  и  $H$  в различные моменты времени. Для лучшей визуализации был написан алгоритм, который собирает данные графики в GIF изображение. Во время написания алгоритма пришлось столкнуться с тем, что количество изображений превышало размер буфера, в котором они хранились и в итоге с буфером пришлось работать напрямую.

Так как графики строятся по изначальным данным - точкам разбиения, то для более детальной картинки полученные значения по которым строились графики были интерполированы линейно (ИСПРАВИТЬ НА КУБИЧЕСКУЮ ИНТЕРПОЛЯЦИЮ).

Еще, в качестве дополнительных результатов эксперимента были построены графики для функции тока, которая определяется следующим образом:

$$\Psi = \int q_1 dx_2$$

Визуализация данной функции дает лучшее понимание о том, каким именно образом распространяются волны в водоеме.

## 5.4 Вычисления

МКЭ является очень ресурсоемким численным методом и именно из-за этого он долго не применялся, а существовал только в теории. Так и сейчас для расчетов на сетке с более чем 100 элементами требуются достаточные мощности. Для того, чтобы быстрее получать результаты написанная программа была упакована в Docker - программное обеспечение для автоматизации развёртывания и управления приложениями в среде виртуализации на уровне операционной системы. Docker позволяет упаковать приложение со всем его окружением и зависимостями в контейнер, который может быть перенесён на любую систему. Это очень важно, так как в процессе написания работы было использовано очень много внешних python модулей, которые можно запаковать один раз и не переживать по поводу них в дальнейшем. Далее эксперименты производились на сервере с конфигурацией: TODO.

## **6 Разработка базы данных для хранения информации о проведённых экспериментах**

Данные полученные в результате численных экспериментов являются специфичными, так как они представляют из себя GIF анимацию и некоторую мета информацию, которая представлена в виде JSON файла. Из-за этого помимо общепринятых требований к БД следует, основываясь на приведенной выше специфике задачи ввести следующие требования, которой должна удовлетворять БД и СУБД управляющая ей:

1. Запись об эксперименте должна позволять хранить любую мета информацию, которая может различаться в зависимости от проведенного эксперимента
2. БД должна позволять добавлять поля к уже существующим данным. Это, к примеру, нужно для добавления в запись данных, которые получены в результате постобработки эксперимента человеком
3. БД должна легко масштабироваться, так как данных может быть много
4. Должно быть отсутствие необходимости сопоставления объекта в базе и программе для упрощения разработки
5. Хранение графической информации должно быть реализовано максимально просто и эффективно

### **6.1 Описание базы данных**

Основываясь на приведенных выше требованиях, в качестве СУБД решено выбрать MongoDB, которая классифицируется как NoSQL. MongoDB — документоориентированная система управления базами данных с открытым исходным кодом, не требующая описания схемы таблиц. Т.е. каждая запись — это документ без жестко заданной схемы, который может содержать вложенные документы.

Так как документо-ориентированные базы данных еще не так распространены как реляционные, но их популярность с каждым годом все увеличивается, целесообразно будет разъяснить некоторые нюансы терминологии документо-ориентированных баз данных. В учебнике по

MongoDB The Little MongoDB Book от автора Karl Seguin приведены шесть основных концепций MongoDB:

1. MongoDB — концептуально является тем же самым, что обычная, привычная нам база данных (или в терминологии Oracle — схема). Внутри MongoDB может быть ноль или более баз данных, каждая из которых является контейнером для прочих сущностей.
2. База данных может иметь ноль или более «коллекций». Коллекция настолько похожа на традиционную «таблицу», что можно смело считать их одним и тем же.
3. Коллекции состоят из нуля или более «документов». Опять же, документ можно рассматривать как «строку».
4. Документ состоит из одного или более «полей», которые — как можно догадаться — подобны «колонкам».
5. «Индексы» в MongoDB почти идентичны таковым в реляционных базах данных.
6. «Курсыры» отличаются от предыдущих пяти концепций, но они очень важны (хотя порой их обходят вниманием) и заслуживают отдельного обсуждения. Важно понимать, что когда мы запрашиваем у MongoDB какие-либо данные, то она возвращает курсор, с которыми мы можем делать все что угодно — подсчитывать, пропускать определённое число предшествующих записей — при этом не загружая сами данные.

Если резюмировать вышесказанное, то MongoDB состоит из «баз данных», которые состоят из «коллекций». «Коллекции» состоят из «документов». Каждый «документ» состоит из «полей». «Коллекции» могут быть проиндексированы, что улучшает производительность выборки и сортировки. И наконец, получение данных из MongoDB сводится к получению «курсора», который отдаёт эти данные по мере надобности.

Так же стоит привести основные достоинства данной базы данных:

1. Документо-ориентированное хранилище (простая и мощная JSON подобная схема данных)
2. Достаточно гибкий язык для формирования запросов
3. Динамические запросы
4. Полная поддержка индексов

5. Профилирование запросов
6. Эффективное хранение двоичных данных больших объёмов, например фото и видео
7. Журналирование операций, модифицирующих данные в БД
8. Поддержка отказоустойчивости и масштабируемости: асинхронная репликация, набор реплик и шардинг

Основными плюсами MongoDB помимо ее документо-ориентированности для данной ВКР, стали:

1. схожий с реляционными СУБД подход к хранению данных, который построен на следующих вещах: база данных-коллекция-документ-поле
2. формат хранения данных. Данные в MongoDB хранятся в формате BSON. BSON это бинарный формат, используемый для хранения документов и осуществления удаленного вызова процедур в MongoDB. Тот факт, что данные хранятся именно в бинарном виде, значительно ускоряет время их обработки.
3. Простота разработки. В отличии от конкурентов MongoDB достаточно просто разворачивается и администрируется, а так же имеет хорошее API
4. GridFS - файловая система MongoDB для хранения больших файлов

## 6.2 Формат представления данных в БД

Так как в качестве СУБД была выбрана документо-ориентированная MongoDB, то и структура базы данных будет приводится в ее терминологии.

Основной и единственной базой данных является база "эксперименты"("experiments"). В ней находится коллекция "данные"("data"). Данная коллекция содержит в себе документы, каждый из которых является проведенным экспериментом. Каждый из документов хранит в себе следующие поля:

1. дата проведения эксперимента
2. время выполнения эксперимента
3. построенные графики

4. матрица решения системы ДУ
5. вектор временных точек, в которых была решена ОДУ
6. тип и файл сетки
7. количество базисных функций
8. id изображений полученных в результате эксперимента

Приведем пример структуры документа в формате JSON (в подобном виде она хранится в MongoDB):

```
{
  "date": <some_data>,
  "images": {
    "frame": {
      "q1": frame_q1_id,
      "q2": frame_q2_id,
      "H": frame_H_id
    },
    "surf": {
      "q1": surf_q1_id,
      "q2": surf_q2_id,
      "H": surf_H_id
    },
    "wave": {
      "psi1": wave_psi1_id,
      "psi2": wave_psi2_id
    }
  },
  "solution": {
    "matrix": <some_data>,
    "time": <some_data>
  },
  "meta": {
    "mesh": {
      "type": mesh_type,
      "file": mesh_file_id
    }
  }
}
```

```
    } ,  
    "time": time ,  
    "quantity_of_basis_functions": quantity_of_fe  
}  
}
```

Поля с постфиксами *id* являются уникальными идентификаторами для GIF анимаций, которые сохранены на файловой системе *GridFS*. Благодаря этому изображения результатов экспериментов не пропадут в случае отказа одной из нод кластера *MongoDB* и получается, что таким образом обеспечивается надежная сохранность данных.

## **ЗАКЛЮЧЕНИЕ**

TODO В представленной бакалаврской работе удалось разработать программу для решения уравнений мелкой воды с помощью метода конечных элементов.

Реализована возможность переносимости разработанного алгоритма, а также возможность удаленного использования.

Анализ решения уравнений позволил выявить особенности поведения волн XXX

## **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Коннор, Дж., Бреббиа, К. Метод конечных элементов в механике жидкости. — Л.: Судостроение, 1979. — С. 264.
- 2.
3. Rand, A. Where and How Chew's Second Delaunay Refinement Algorithm Works. — URL: <http://2011.cccg.ca/PDFschedule/papers/paper91.pdf> (дата обращения: 20.04.2017).
4. Панкратов, И.А. Введение в методы взвешенных невязок. — URL: [http://www.sgu.ru/sites/default/files/textdocsfiles/2013/12/10/fem\\_introduction.pdf](http://www.sgu.ru/sites/default/files/textdocsfiles/2013/12/10/fem_introduction.pdf) (дата обращения: 20.04.2017).
5. Зенкевич, О., Морган, К. Конечные элементы и аппроксимация. — М.: Мир, 1986. — С. 318.
6. Зорич, В. А. Математический анализ. — М.: МЦНМО, 2002. — Т. 1. — С. 664.

## ПРИЛОЖЕНИЕ А

### **Примеры численного решения задачи**

тут различные примеры для различной геометрии и различных начальных параметров

ПРИЛОЖЕНИЕ Б  
**Исходный код реализации**

ПИТОН

## ПРИЛОЖЕНИЕ В

### Выкладки для МКЭ

Запишем уравнение неразрывности:

$$\frac{d}{dx_1}q_1 + \frac{d}{dx_2}q_2 + \frac{\partial}{\partial t}(H\rho) = 0 \quad (\text{B.1})$$

Подставим в него разложение для  $H$ :

$$\frac{\partial}{\partial t} \left( \rho \left( H_0(x_1, x_2) + \sum_{k=1}^M N_k(x_1, x_2) a_{2m+k}(t) \right) \right) + \quad (\text{B.2})$$

$$\frac{\partial}{\partial x_1} \sum_{k=1}^M N_k(x_1, x_2) a_k(t) + \frac{\partial}{\partial x_2} \sum_{k=1}^M N_k(x_1, x_2) a_{m+k}(t) = 0 \quad (\text{B.3})$$

Умножим уравнение на  $\sum_{l=1}^M W_l$ , где  $W_l$  весовая функция и проинтегрируем по треугольнику:

$$\sum_{l=1}^M \iint W_l(x_1, x_2) a_k(t) \sum_{k=1}^M \frac{\partial}{\partial x_1} N_k(x_1, x_2) dx_1 dx_2 + \sum_{l=1}^M \iint W_l(x_1, x_2) a_{m+k}(t) \sum_{k=1}^M \frac{\partial}{\partial x_2} N_k(x_1, x_2) dx_1 dx_2$$

$$\sum_{l=1}^M \iint \rho W_l(x_1, x_2) \frac{d}{dt} a_{2m+k}(t) \sum_{k=1}^M N_k(x_1, x_2) dx_1 dx_2 = \left[ - \sum_{l=1}^M \left( \iint W_l(x_1, x_2) a_k(t) \right) \right]$$