МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«ТЮМЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ НАУК
Кафедра программного обеспечения

РЕКОМЕНДОВАНО К ЗАЩИТЕ В ГЭК
И ПРОВЕРЕНО НА ОБЪЕМ
ЗАИМСТВОВАНИЯ
Заведующий кафедрой
д.п.н., профессор
_____И.Г. Захарова
27.06 _____2018 г.

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

РАЗРАБОТКА МОДУЛЯ, РЕАЛИЗУЮЩЕГО ПРЕДОБРАБОТКУ ДАННЫХ КАПИЛЛЯРНОГО ДАВЛЕНИЯ И ОТНОСИТЕЛЬНОЙ ФАЗОВОЙ ПРОНИЦАЕМОСТИ

02.04.03. Математическое обеспечение и администрирование информационных систем
Магистерская программа «Высокопроизводительные вычислительные системы»

Выполнил работу
Студент 2 курса
очной формы обучения
Бобырев
Сергей
Викторович

Научный руководитель
к.пед.наук, доцент
Кафедра ПО
Плотоненко
Юрий
Анатольевич

Рецензент
к.физ-мат.н., доцент
Кафедра ПО
Ступников
Андрей
Анатольевич

Тюмень 2018

Оглавление

ВВЕДЕНИЕ

Основная цель современной разработки месторождений углеводородов направлена на наиболее полное извлечение их извлекаемых запасов при максимальной экономической рентабельности. Для достижения наиболее полного извлечения нефти используются передовые технологии. Одним из ключевых направлений является компьютерное моделирование. Адаптация истории разработки и ее прогноз, позволяют оптимально, и, с наименьшими затратами, разрабатывать месторождения углеводородов.

Одними из данных, которые необходимы для построения модели являются данные капиллярного давления и относительной фазовой проницаемости.

На начальном этапе выполняется сбор информации, при этом зачастую первичные данные нуждаются в предварительной подготовке и в введении определенных поправок (приведение к пластовым условиям, нормализация, осреднение, моделирование и т.д.). Поскольку нередко при исследовании близких по свойствам горных пород получаются кривые разной формы, есть необходимость в выборе наиболее адекватной кривой.

В связи с этим целью моей работы является разработка модуля, реализующего предобработку данных капиллярного давления и относительной фазовой проницаемости. Для реализации данной цели были выявлены следующие задачи:

- Изучить процесс предобработки данных капиллярного давления и относительной фазовой проницаемости;
- Определить инциденты при обработке данных;
- Изучить программный комплекс Techlog и API для создания модуля;
- Реализовать модуль для программного комплекса Techlog, выполняющий алгоритмы предобработки данных капиллярного давления и относительной фазовой проницаемости.

ГЛАВА 1. ПРОЦЕСС ПРЕДОБРАБОТКИ ДАННЫХ КАПИЛЛЯРНОГО ДАВЛЕНИЯ И ОТНОСИТЕЛЬНОЙ ФАЗОВОЙ ПРОНИЦАЕМОСТИ.

1.1 Значимость данных капиллярного давления и относительной фазовой проницаемости для гидродинамического моделирования.

Гидродинамическое моделирование процесса разработки – это эффективный инструмент, который позволяет определить основные моменты добычи нефти: структуру остаточных и извлекаемых запасов, определение потенциала разработки [5]. Это способствует развитию нефтедобывающей отрасли и просчета эффективности отдельных мест добычи в частности.

Для моделирования гидродинамических процессов важны функции фазовых проницаемостей. Это данные, которые зависят от коэффициента насыщенности флюидами [5]. С помощью этих функций определяется точная модель двухфазного течения, которая по большей части представляет собой процесс вытеснения нефти водой.

Создание гидродинамических моделей является важным элементом при создании проектов новых разработок. Для этого используются модели различных размерностей (2D, 3D), что позволяет получить важные данные о подземных процессах. Доступные сегодня программные комплексы позволяют прогнозировать показатели разработки, используя в основе интегрируемые системы нелинейных дифференциальных уравнений в частных производных [5]. Для подобной математической модели необходимы точные данные, касающиеся функций капиллярного давления и фазовых проницаемостей. Эти данные получаются экспериментально, производясь из расчетов на примере конкретных объектов [3].

Причиной необходимости определения ОФП является то, что эти данные незаменимы для создания рабочих моделей, ведь являются исходным материалов для определения показателей разработки. Поэтому результаты

моделирования, которые предельно точно описывают процессы в пласте, позволяют добиться отличных результатов при проектировании, регулировании и анализе разработки различных месторождений, а также гидродинамических расчетах и прогнозе потенциала разработки.

В настоящее время при проектировании разработки нефтяных месторождений по регламенту необходимо использовать постоянно действующие модели (ПДМ). Широко применяемые ПДМ (Eclipse, TempestMore, ТРИАС) включают два основных блока программ – блок программ для геологического моделирования и блок программ гидродинамического моделирования [12]. Применяемые ПДМ имеют как много общих элементов, так и много различий. Общее условие надежного функционирования всех ПДМ – это адаптация параметров входящих в модель.

Одним из основных законов для расчета технологических показателей является закон сохранения масс, в гидродинамических симуляторах он используется в виде уравнения материального баланса [12]. Уравнение материального баланса описывает взаимосвязь между давлением, добычей и начальными балансовыми запасами флюидов. При построении постоянно действующей гидродинамической модели (ПДГДМ) начальные балансовые запасы определяются геологической моделью, создание которой основано на стохастических законах [12]. Особенностью стохастического построения геологической модели заключается в том, что рассчитывается вероятностное распределение запасов нефти. Немаловажную роль в процессе адаптации, играет выбор метода расчета балансовых запасов объекта моделирования, так как от этого зависит точность и скорость расчета показателей разработки.

Характерной особенностью современных гидродинамических моделей является то, что они состоят из множества ячеек. Размер ячеек зависит от расстояния между скважинами. Чем больше расстояние, тем больше размеры ячеек, так как при малой изученности месторождения уменьшение размеров

ячеек не влияет на точность расчета, наоборот приводит к увеличению времени и погрешности расчета.

Начальные балансовые запасы нефти объекта моделирования рассчитываются как сумма объемов нефти в каждой ячейке. Объем нефти в ячейке рассчитывается по следующей формуле:

$$V_{нефт.\ яч.} = V_{яч.}\ K_{песч.} m S_{н.} \quad (1)$$

,где $V_{яч}$ – объем ячейки,

$K_{песч}$ – коэффициент песчаности,

m – пористость,

$S_н$ – нефтенасыщенность.

Объем ячейки определяется исходя из её геометрических размеров, коэффициент песчаностости и пористость определяют с помощью геофизических исследований. Наибольшую трудность представляет нахождение нефтенасыщенности ячейки. В гидродинамических симуляторах наиболее часто используется метод определения нефтенасыщенности с помощью кривых капиллярного давления [12].

Схема расчета насыщенности с использованием кривых капиллярного давления выглядит следующим образом.

Находится величина капиллярного давления. Из лабораторных исследований находится зависимость между водонасыщенностью и капиллярным давлением, которая имеет вид расположенные на рисунке 1.

Адаптация запасов нефти, при использовании модели гидростатического равновесия, осуществляется путем изменения характера кривой капиллярного давления [12]. Данный метод наиболее часто используется в моделировании, так как обладает высокой точностью результатов при небольшом расчетном времени.

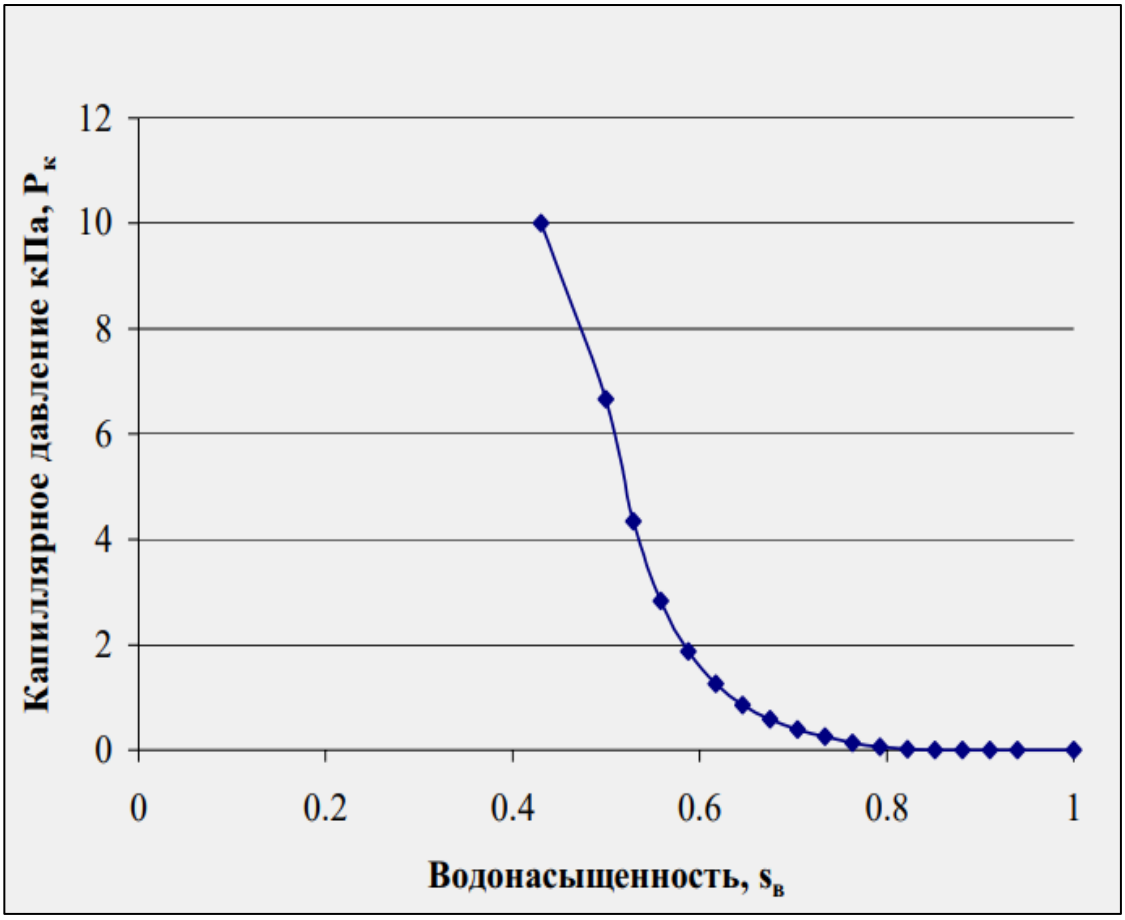

Рисунок 1. Зависимость капиллярного давления от водонасыщенности.

1.2 Процесс подготовки данных капиллярного давления и относительной фазовой проницаемости для загрузки в гидродинамический симулятор.

Так как данные капиллярного давления и относительной фазовой проницаемости получаются в ходе лабораторных исследований горных пород, то есть необходимость во введении определенных поправок. Существует также необходимость построения осредняющей кривой как для капиллярного давления, так и для относительных фазовых проницаемостей. Причиной этому является то, что нередко при исследовании горных пород со схожими свойствами получаются кривые, имеющие большое различие.

1.2.1 Обработка данных капиллярного давления типа MICP.

MICP (Mercury Injection Capillary Pressure) – это капиллярное давление, полученное в ходе исследований горных пород при помощи ртутного аппарата капиллярного давления. Данный алгоритм приближает данные каиллярного давления к реальным условиям при помощи эмпирических соотношений Хилла, Клиена и Ширли [1].

Расчет результативных значений капиллярного давления и водонасыщенности производится по следующим формулам:

$$PC_{CBW} = PC_{LAB}\left(\frac{\varphi_e}{\varphi_t}\right) \quad (2)$$

$$CSW_{CBW} = 1 - (1 - CSW_{LAB})\frac{\varphi_e}{\varphi_t} \quad (3)$$

,где PC_{LAB} - это данные капиллярного давления типа MICP;

CSW_{LAB} - это данные водонасыщенности;

φ_e - эффективная пористость;

φ_t - общая пористость.

Отношение эффективной пористости к общей пористости:

$$\frac{\varphi_e}{\varphi_t} = 1 - (0{,}084C_o^{-1/2} + 0{,}22)\,Q_v \quad (4)$$

В формуле (4) C_0 является входным параметром, представляющим значение минерализации. Минерализация - показатель количества содержащихся в воде растворённых веществ (неорганические соли, органические вещества).

Значение $Q_v = \alpha\varphi^{-b}$ (5)

,где α и b – коэффициенты, которые задаются пользователем,

φ - значение пористости. Для каждой кривой капиллярного давления используется собственное значение пористости.

1.2.2 Приведение данных капиллярного давления к пластовым условиям.

Необходимым этапом перед осреднением кривых капиллярного давление является приведение данных к пластовым условиям. В большинстве случаев входными данными являются данные капиллярного давления полученные из лабораторных исследований, но так же возможны варианты следующих входных данных:

- Высота над уровнем свободной воды [6];
- Значения функции J-Leverett. Данная функция - это средство нормализации значений капиллярного давления по пробам с различными значениями пористости и проницаемости [2].

Высота над уровнем свободной воды рассчитывается по формуле:

$$H = \frac{PC\_Res}{\Delta\rho g} \quad (6)$$

,где PC_Res – капиллярное давление в пластовых условиях;

Δρ – разность плотности воды и нефти;

g – градиент давления, константа равная 0.0980665.

Функция J-Leverett:

$$J = \frac{3.141534\ PC\_Res}{\sigma_{res}\ \ \cos_{res}} \sqrt{\frac{K}{\varphi}} \quad (7)$$

,где PC_Res – капиллярное давление в пластовых условиях;

σ_{res} – коэффициент поверхностного натяжения;

$\cos_{res}$ – величина косинуса угла смачивания для капиллярного давления в пластовых условиях;

K – проницаемость;

φ – пористость.

Расчет капиллярного давления в пластовых условиях выражается следующим образом:

$$\text{PC\_Res} = \frac{PC_{Lab}\cos_{res}\sigma_{res}}{\sigma_{lab}\cos_{lab}} \quad (8)$$

,где PC_Lab – капиллярное давление;

σ_{res} – коэффициент поверхностного натяжения в пластовых условиях;

$\cos_{res}$ – величина косинуса угла смачивания для капиллярного давления в пластовых условиях;

σ_{lab} – коэффициент поверхностного натяжения в лабораторных исследования;

$\cos_{lab}$ – величина косинуса угла смачивания для капиллярного давления в лабораторных исследованиях.

Как видно из формул (6), (7), (8) во всех случаях возможно вывести значения капиллярного давления в пластовых условиях.

Затем для кривых капиллярного давления в пластовых условиях выполняется процедура установления одинакового максимума.

Алгоритм установления одинакового максимума:

- Пользователь задает значение максимального капиллярного давления P_{max};
- Каждая кривая капиллярного давления обрабатывается в отдельности. Существует 3 возможных случая:
    - $P_{max} < Y_{max.}$ В таком случае происходит интерполяция для поиска значения X при $Y = P_{max}$.
    - $P_{max} > Y_{max}$. При таких условиях происходит экстраполяция кривой до значения по оси $Y = P_{max}$.

- $P_{max}= Y_{max}$. Кривая капиллярного давления имеет корректный максимум. Перерасчета не происходит.

P_{max} - максимальное значение капиллярного давления, заданное пользователем, Y_{max} - максимум кривой капиллярного давления по оси Y.

1.2.3 Осреднение кривых капиллярного давления.

Последний этап предобработки данных капиллярного давления - это осреднение нормализованных кривых капиллярного давления.

Нормализация заключается в масштабировании кривой по оси X между значениями 0 и 1. Формула для нормализации:

$$X_{norm\ i} = \frac{X\,i - X\,min}{1 - X\,min} \qquad (9)$$

,где $X_{norm\ i}$ - нормализованное значение точки с индексом i;

X_i - исходное значение X с индексом i;

X_{min} - минимальное значение кривой по оси X.

Перед процессом нормализации происходит сохранение X_{min} каждой кривой капиллярного давления. Это необходимо для того, чтобы была возможность вернуть нормализованные кривые в исходное состояние.

Из нормализованных кривых капиллярного давления вычисляется осредненная кривая. Осредненная может производиться как по оси X, так и по оси Y.

1.2.4 Осреднение кривых относительных фазовых проницаемостей.

Относительные фазовые проницаемости представлены в следующем виде:

- Относительная проницаемость для воды;
- Относительная проницаемость для нефти;

- Водонасыщенность.

Для каждой кривой относительной проницаемости для воды и для нефти выполняется нормализация по формулам (9) и (10).

$$Y_{norm\,i} = \frac{Y\,i - Y\,min}{1 - Y\,min} \quad (10)$$

,где $Y_{norm\,i}$ - нормализованное значение Y точки с индексом i;

Y_i - исходное значение Y с индексом i;

Y_{min} - минимальное значение кривой по оси Y.

После нормализации выполняется вычисление осредняющих кривых отдельно для данных относительной проницаемости для воды и отдельно для данных относительной проницаемости для нефти.

1.2.5 Моделирование кривых относительных проницаемостей.

Этот шаг предобработки используется, чтобы сгладить данные, что позволяет избежать проблем в среде моделирования. Этот шаг не является обязательным, поскольку данные часто получаются корректными после осреднения (когда имеется достаточно кривых). Но если кривых относительных фазовых проницаемостей немного, то рекомендуется применение моделирования.

Задача моделирования состоит в поиске коэффициентов регрессии для модели Corey. Уравнения регрессии для модели Corey:

- Для относительной проницаемости для нефти:

$$Y = (1.001 - X)^A \quad (11)$$

- Для относительной проницаемости для воды:

$$Y = X^A \quad (12)$$

После вычисления коэффициентов регрессии осуществляется перерасчет значений относительной проницаемости для нефти по формуле

(11) и для воды по формуле (12), в качестве значений X используется водонасыщенность.

1.3 Идентификация инцидентов обработки данных.

Данные кривых капиллярного давления представлены в виде двух двумерных массивов чисел.

$$P_к = \begin{pmatrix} p_{11}, p_{12}, p_{13} \dots p_{1k} \\ p_{21}, p_{22}, p_{23} \dots p_{2k} \\ \vdots \\ p_{n1}, p_{n2}, p_{n3} \dots p_{nk} \end{pmatrix}$$

$$S_в = \begin{pmatrix} s_{11}, s_{12}, s_{13} \dots s_{1k} \\ s_{21}, s_{22}, s_{23} \dots s_{2k} \\ \vdots \\ s_{n1}, s_{n2}, s_{n3} \dots s_{nk} \end{pmatrix}$$

, где

$P_к$ - значения капиллярного давления,

$S_в$ - значения водонасыщеннсоти,

k - максимальное количество точек кривой, которое может быть описано текущим набором данных,

n - значение равное максимальному количество кривых капиллярного давления, которое может быть описано текущим набором данных.

Таким образом, кривые капиллярного давления получаются из точек (s_{ni} , p_{ni}), для i = 1,2,3 .. k

,где n - номер кривой капиллярного давления,

k - количество точек в кривой,

i - индекс точки кривой капиллярного давления.

Данные кривых относительных фазовых проницаемостей представлены в следующем виде:

$$O = \begin{pmatrix} o_{11}, o_{12}, o_{13} \dots o_{1h} \\ o_{21}, o_{22}, o_{23} \dots o_{2h} \\ \vdots \\ o_{m1}, o_{m2}, o_{m3} \dots o_{mh} \end{pmatrix}$$

$$W = \begin{pmatrix} w_{11}, w_{12}, w_{13} \dots w_{1h} \\ w_{21}, w_{22}, w_{23} \dots w_{2h} \\ \vdots \\ w_{m1}, w_{m2}, w_{m3} \dots w_{mh} \end{pmatrix}$$

$$R = \begin{pmatrix} r_{11}, r_{12}, r_{13} \dots r_{1h} \\ r_{21}, r_{22}, r_{23} \dots r_{2h} \\ \vdots \\ r_{m1}, r_{m2}, r_{m3} \dots r_{mh} \end{pmatrix}$$

, где W - значения относительной проницаемости для воды,

O - значения относительной проницаемости для нефти,

R - значение водонасыщенности,

h - максимальное количество точек кривой, которое может быть описано текущим набором данных,

m - значение равное максимальному количество кривых относительных фазовых проницаемостей, которое может быть описано текущим набором данных.

Кривые относительных фазовых проницаемостей для воды образуются из точек (r_{mi} , w_{mi}), для i = 1,2,3 .. h

,где i - индекс точки кривой.

h - количество точек кривой,

m - номер кривой относительной фазовой проницаемости для воды.

Кривые относительных фазовых проницаемостей для нефти образуются из точек (r_{mi} , o_{mi}), для i = 1,2,3 .. h

,где i - индекс точки кривой.

h - количество точек кривой,

m - номер кривой относительной фазовой проницаемости для нефти.

Во время анализа алгоритмов для подготовки данных капиллярного давления и относительных фазовых проницаемостей были обнаружены инциденты обработки данных.

1.3.1 Наличие точек с пропущенными значениями.

Архитектура данных программного комплекса Techlog позволяет хранить пропущенные значения в данных [8]. Исходя из этого, существует вероятность появления пропущенных значений в исходных данных капиллярного давления и относительной фазовой проницаемости. В таком случае входные данные являются некорректными. В связи с этим существует необходимость добавления проверки на наличие пропущенных значений.

Для предотвращения ошибок, на первоначальном этапе предобработки происходит поиск точек с пустыми значениями как в кривых капиллярного давления, так и в кривых относительных фазовых проницаемостей. Если такие точки существуют, то некорректные кривые не будут учитываться в дальнейших алгоритмах предобработки.

1.3.2 Допустимые значения для входных данных.

Данные водонасыщенности и относительных фазовых проницаемостей имеют ограничения в возможных значениях.

- Для водонасыщенности значения могут находится в интервале (0, 1) [6];

- Для относительных фазовых проницаемостей значения могут находится в отрезке (0, 1) [6].

В таком случае необходимо найти максимальное и минимальное значения для данных водонасыщенности и относительных фазовых проницаемостей и проверить на вхождение в интервал (0, 1). Если крайние значения выходят за границы отрезка, то данные являются некорректными и в дальнейших расчетах не учитываются.

1.3.3 Проблемы вычисления осредняющей кривой.

Формула для расчета точки осредняющей кривой по оси X:

$$x_{avg\ i} = \frac{\sum_{k=0}^{n} x_{ki}}{n} \quad (13)$$

,где n - количество кривых,

k - индекс кривой,

i - индекс точки кривой k,

x_{ki} - значение точки по оси X с индексом i кривой с индексом k.

Формула для расчета точки осредняющей кривой по оси Y:

$$y_{avg\ i} = \frac{\sum_{k=0}^{n} y_{ki}}{n} \quad (14)$$

,где n - количество кривых,

k - индекс кривой,

i - индекс точки кривой k,

y_{ki} - значение точки по оси Y с индексом i кривой с индексом k.

Однако, для формул (13) и (14) необходимо:

- Для формулы (13) значения каждой кривой по оси Y были одинаковыми ($y_{0i} = y_{1i} = y_{2i} = .. = y_{ni}$, где i - индекс точки, n - количество кривых);

- Для формулы (14) значения каждой кривой по оси X были одинаковыми ($x_{0i} = x_{1i} = x_{2i} = .. = x_{ni}$, где i - индекс точки, n - количество кривых).

В таком случае существует необходимость перерасчета каждой кривой. Если расчет осредненяющей кривой происходит по оси X, то алгоритм перерасчета кривых следующий:

- Формируется массив значений Y, который будет одинаковым для каждой кривой. Для этого осуществляется поиск уникальных значений Y из всех кривых и сортировка этих значений.
- При помощи интерполяции рассчитывается массив значений X для каждой кривой.

Если вычисление осредняющей кривой происходит по оси Y, то алгоритм перерасчета кривых следующий:

- Формируется массив значений X, который будет одинаковым для каждой кривой. Для этого осуществляется поиск уникальных значений X из всех кривых и сортировка этих значений.
- При помощи интерполяции рассчитывается массив значений Y для каждой кривой.

1. Кривые имеют различный максимум по оси Y.

Допустим имеется кривая K_1 с максимальным значением по оси Y- y_{max1}, и кривая K_2 с максимальным значением по оси Y- y_{max2} и выполняется условие $y_{max1} > y_{max2}$. В таком случае на интервале (y_{max2}, y_{max1}) кривая K_2 не имеет значений, и, соответственно, осредняющая кривая не может быть построена на этом промежутке.

Для предотвращения таких ситуаций применяется следующий порядок действий:

- Находится значения y_{MAX}, для которого выполняются условие:

  $y_{MAX} \geq y_{max\ i}$, для $0 \leq i < n$,

где n - количество кривых,

y_{MAX}- максимальное значение Y среди всех кривых,

$y_{max\ i}$ - максимальное значение кривой с индексом i.

- Для всех кривых, у которых максимальное значение по оси Y меньше Y_{MAX}, реализовать экстраполяцию до значения Y_{MAX}.

2. Ненормализованные кривые.

Рассмотрим кривую K_1 с минимальным значением по оси X - x_{min1}. И кривую K_2 с минимальным значением по оси X- $x_{min2.}$ Причем,

$x_{min1} < x_{min2.}$

В таком случае существует полуинтервал, на котором значения кривой K_2 не существуют (x_{min1}, x_{min2}].

Для исправления необходимо применить следующий алгоритм:

- Находится значение X_{MIN}, для которого выполняются условия:

  $x_{MIN} \leq x_{min\ i}$, для $0 \leq i < n$,

  где n - количество кривых,

  X_{MIN}- минимальное значение X среди всех кривых,

  $x_{min\ i}$ - минимальное значение кривой i.

- Идентифицируются кривые, у которых $x_{min} > x_{MIN.}$
- Для всех кривых идентифицированных на предыдущем этапе производится построение отрезка [a, b],

  где a = (x_{MIN}, y_{MAX}), b = (x_{min}, y_{MAX}).

Данный алгоритм должен следовать за алгоритмом, который исправляет кривые, имеющие различный максимум по оси Y.

3. Кривая, полученная в результате осреднения, имеет избыточное количество точек.

В процессе реализации алгоритма вычисления осредняющей кривой, довольно часто результирующая кривая, имеет избыточное количество точек. Например, отрезок кривой может содержать не 2 точки, которых достаточно, а 20 точек. Такая избыточность нагружает объекты визуализации данных, а также последующие расчеты буду занимать больше времени.

Для устранения данной проблемы используется алгоритм Рамера-Дугласа-Пекера, который позволяет избавиться от избыточных точек не изменяя при этом конфигурации кривой [4]. Входными данными для алгоритмы являются массив значение X, массив значений Y, погрешность ε.

Алгоритм Рамера-Дугласа-Пекера:

- Задается значение $\varepsilon > 0$;
- Находится точка R, наиболее удаленная от отрезка, соединяющего первую и последнюю точку кривой;
- Если точка R находится на расстоянии больше чем ε, то алгоритм рекурсивно вызывается на отрезке, соединяющем первую точку кривой и точку R, и на отрезке, соединяющем точку R и последнюю точку кривой. Это означает, что максимально удаленная точка R отмечена к сохранению.
- Если точка находится на расстоянии меньше чем ε, то все точки которые еще не были отмечены к сохранению могут быть отброшены.

Пример работы данного алгоритма представлен на рис. 2.

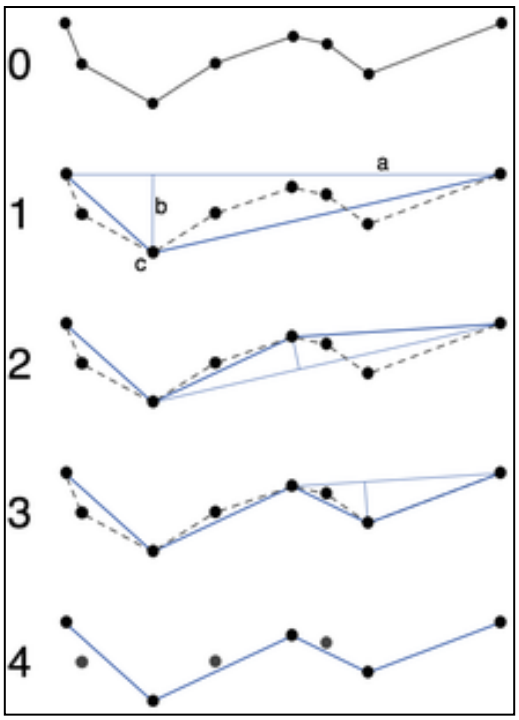

Рисунок 2. Сглаживание кусочно-линейной кривой алгоритмом Дугласа-Пекера.

1.4 Методы интерполяции и аппроксимации.

1.4.1 Интерполяция

Интерполяция – способ нахождения промежуточных значений величины по имеющемуся дискретному набору известных значений [13].

Пусть в ходе эксперимента при изменении входной величины x (x_0, x_1, x_2,..., x_n) получены значения функции y=f(x) (y_0, y_1, y_2.....y_n) (табл. 1).

Таблица 1.

Таблица экспериментальных данных.

x_0	x_1	x_2	x_3	...	x_n
y_0	y_1	y_2	y_3	...	y_n

Интерполяцию функций применяют в случае, когда требуется найти значение функции y(x) при значении аргумента x_i , принадлежащего интервалу [x_0, …, x_n], но не совпадающего по значению ни с одним значением, приведенным в таблице 1.

Данная задача, а именно интерполяция функций, часто встречается при ограниченности возможностей при проведении эксперимента.

При этом во многих случаях аналитическое выражение функции y(x) не известно и получить его по таблице ее значений (табл. 1) в большинстве случаев невозможно. Поэтому вместо нее строят другую функцию, которая имеет ту же таблицу значений (совпадает с ней в точках x_0, x_1, x_2,..., x_n), что и f(x), т. е.

$$P_n(x_0)=f(x_0)=y_0; \quad (15)$$

$$...$$

$$P_n(x_i)=f(x_i)=y_i;$$

где i = 0, 1, 2, … , n.

Нахождение приближенной функции называется интерполяцией, а точки x_0, x_1, x_2, …, x_n – узлами интерполяции.

Интерполирующую функцию ищут в виде полинома n степени.

Для каждого набора точек имеется только один интерполяционный многочлен, степени не больше n. Однозначно определенный многочлен может быть представлен в различных видах.

Графически задача интерполирования заключается в том, чтобы построить такую интерполирующую функцию, которая бы проходила через все узлы интерполирования.

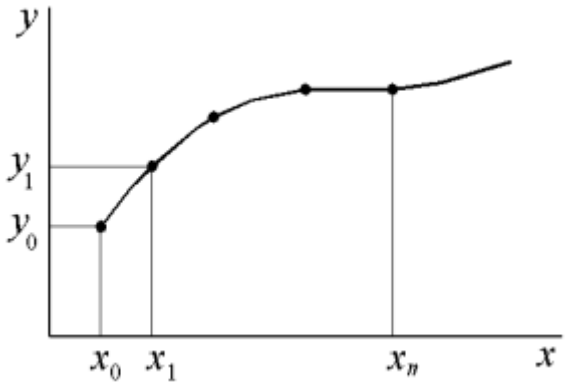

Рисунок 3. Интерполирующая функция.

1.4.1.1 Канонический полином.

Вид канонического полинома степени n:

$$P_n(x)=a_0+a_1x+a_2x^2+\ldots+a_{n-1}x^{n-1}+a_nx^n . \qquad (16)$$

Выбор многочлена степени n основан на том факте, что через n+1 точку проходит единственная кривая степени n. Подставив (16) в (15), получим систему линейных алгебраических уравнений :

$$\begin{cases} a_0 + a_1 x_0 + a_2 x_0^2 + \ldots + a_{n-1} x_0^{n-1} + a_n x_0^n = y_0 \\ a_0 + a_1 x_1 + a_2 x_1^2 + \ldots + a_{n-1} x_1^{n-1} + a_n x_1^n = y_1 \\ a_0 + a_1 x_2 + a_2 x_2^2 + \ldots + a_{n-1} x_2^{n-1} + a_n x_2^n = y_2 \\ \qquad\qquad\qquad\qquad \ldots \\ a_0 + a_1 x_n + a_2 x_n^2 + \ldots + a_{n-1} x_n^{n-1} + a_n x_n^n = y_n \end{cases} \quad (17)$$

Решая эту систему линейных алгебраических уравнений, вычисляются коэффициенты интерполяционного полинома a_0, a_1, a_2, ..., a_n [7].

1.4.1.2 Линейная интерполяция.

Линейная интерполяция – часто используемый вид интерполяции. Она состоит в том, что заданные точки с координатами x_i , y_i при i=0, 1, 2, ... n соединяются прямолинейными отрезками, а функцию y(x) можно приближенно представить в виде ломаной [7].

Уравнения каждого отрезка ломаной в общем случае разные. Поскольку имеется n интервалов (x_{i-1}, x_i), то для каждого из них в качестве уравнения интерполяционного многочлена используется уравнение прямой, проходящей через две точки: для i-го интервала можно написать уравнение прямой, проходящей через точки (x_{i-1}, y_{i-1}) и (x_i , y_i),

$$\frac{y - y_{i-1}}{y_i - y_{i-1}} = \frac{x - x_{i-1}}{x_i - x_{i-1}}. \quad (18)$$

Из (19) получается уравнение прямой:

$$y = a_i x + b_i \quad (19)$$

$$a = \frac{y_i - y_{i-1}}{x_i - x_{i-1}},\ b_i = y_{i-1} - a_i x_{i-1}. \quad (20)$$

Следовательно, при использовании линейной интерполяции сначала нужно определить интервал, в который попадает значение аргумента x, а затем подставить его в формулу (19) и найти приближенное значение функции в этой точке.

1.4.2 Аппроксимация

Аппроксимация — это способ нахождения функции, которая наиболее соответствует таблице значений. При аппроксимации выбирается вид функции и определяются параметры этой функции, таким образом, что значения аппроксимирующей функции наиболее приближены к табличным значениям [13]. С помощью аппроксимирующих функций возможно вычисление значений функции в точках отличных от табличных.

Один из наиболее широко используемых методов при решении многих задач восстановления регрессионных зависимостей - это метод наименьших квадратов [14]. Это один из методов регрессионного анализа для оценки неизвестных величин по результатам измерений, содержащих случайные ошибки. Применяется также для приближённого представления заданной функции другими (более простыми) функциями и часто оказывается полезным при обработке наблюдений.

Даны парные наблюдения (x_j, y_j) при ($1 \leq j \leq n$). Количество наблюдений n должно в 6 – 7 раз превышать количество параметров модели. Предполагается, что значения x_j и y_j зависимы: $y_j = f(x_j)$. Необходимо установить форму связи между x_j и y_j.

Предположим, что между x_j и y_j существует линейная зависимость вида $Y = ax + b$. Где a и b - искомые параметры модели. Теоретические значения Y_j можно найти подстановкой числовых значений x_j:

$$Y_j = ax_j + b \quad (21)$$

Тогда значение отклонений будет равно:

$$\varepsilon_j = Y_j - y_j = ax_j + b - y_j \quad (22)$$

Значения отклонений могут быть как положительные, так и отрицательные, поэтому чтобы оценить совокупное значение отклонений их возводят в квадрат и суммируют. Чем меньше полученная сумма, тем лучше

выбранная функция описывает зависимость между x_j и y_j. Необходимо найти такие параметры модели a и b, чтобы выполнялось условие:

$$S = \sum_{j=1}^{n} \varepsilon_j^2 \longrightarrow min \quad (23)$$

Получаем задачу исследование на экстремум функции 2-х переменных a и b.

$$\frac{\partial S}{\partial a} = 2 \sum_{j=1}^{n} \left(-y_j + ax_j + b\right) x_j = 0 \quad (24)$$

$$\frac{\partial S}{\partial b} = 2 \sum_{j=1}^{n} \left(-y_j + ax_j + b\right) = 0 \quad (25)$$

Получаем систему алгебраический уравнений:

$$a \sum_{j=1}^{n} x_j^2 + b \sum_{j=1}^{n} x_j = \sum_{j=1}^{n} y_j \; x_j \quad (26)$$

$$a \sum_{j=1}^{n} x_j + nb = \sum_{j=1}^{n} y_j \quad (27)$$

В результате решения системы находят параметры модели a и b. Аналогично можно вывести систему для другой зависимости.

1.5 Программный комплекс Techlog.

Techlog - программный продукт компании Schlumberger, содержащий набор инструментов для обработки, анализа и интерпретации данных по скважине [9]. Платформа Techlog позволяет решать задачи всех скважинных направлений. В ее функционал входит полный спектр работ - от предварительной обработки данных в режиме реального времени до комплексной детальной интерпретации [9].

Инструменты управления в Techlog позволяют осуществлять различные операции с данными, настройку и создание рабочих процессов под конкретные задачи и требования. Также есть возможность интегрирования собственных методик и алгоритмов, используя внутренний язык программирования Python [9].

Инструменты для контроля качества и управления данными:

- Импорт и экспорт данных в различных форматах;
- Создание и редактирование наборов данных (датасетов);
- Классификация наборов данных по различным типам;
- Создание собственных алгоритмов при помощи языка программирования Python;
- Работа в режиме реального времени.

1.5.1 API для создания модулей в Techlog.

Ocean for Techlog - платформа разработки приложений, дополняющих основной функционал программного комплекса Techlog. Ocean for Techlog разработан таким образом, что разрабатываемые модули не зависят от встроенных. Это позволяет улучшать стабильность работы платформы при разработке новой версии Techlog. Данный API был разработан на базе Qt фреймворка.

1.5.1.1 Qt фреймворк.

Qt - это кроссплатформенный фреймворк для разработки ПО на языке программирования C++. Разрабатывается компанией Trolltech с 1996 года. При помощи Qt можно разрабатывать приложения с графическим интерфейсом, приложения, работающие с сетью, с базами данных, а также мультимедийные приложения. Qt может работать на Linux, Mac OS, Windows [10].

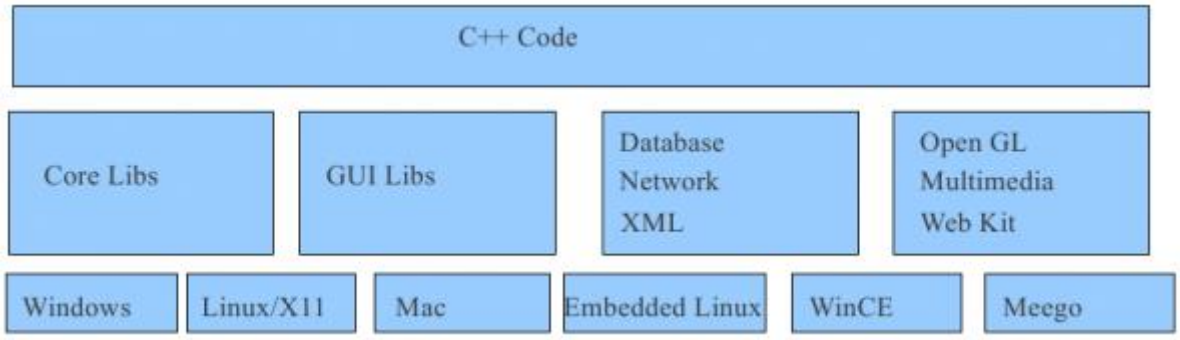

Рисунок 4. Архитектура Qt.

Как показано на рис. 4 на верхнем уровне находится код программы на языке C++. Уровнем ниже расположены классы Qt для создания графического интерфейса, взаимодействия с WebKit, работы с базами данных и.т.д. На нижнем уровне расположена поддержка различных операционных систем.

Основные модули библиотеки Qt:

- QtCore - классы ядра библиотеки Qt, используются другими модулями;
- QtGui — модуль содержит компоненты графического интерфейса;
- QtNetwork — модуль содержит классы для работы с сетью. В данный модуль входят классы для работы с протоколами FTP, HTPP, IP и другими;
- QtOpenGL — модуль содержит классы для работы с OpenGL;
- QtSql — содержит классы для работы с базами данных с использованием языка SQL;

- QtXml — классы для работы с XML [11].

Qt использует MOC (Meta Object Compiler) для предварительной компиляции программ. Исходный текст программы обрабатывается MOC, который ищет в классах программы макрос Q_OBJECT и переводит исходный код в мета-объектный код, после чего мета-объектный код компилируется C++. MOC расширяет функциональность фреймворка, благодаря ему добавляются такие понятия, как слоты и сигналы [15].

1.5.1.2 Архитектура Ocean for Techlog.

Архитектура Ocean for Techlog основана на C++ и Qt фреймворке. Каждый модуль работает в отдельном процессе, что позволяет:

- Запустить модуль в режиме отладки;
- Избегать конфликтов между библиотеками, которые используются различными модулями;
- Выполнять отладку, компиляцию модуля без необходимости перезапуска Techlog;
- Изолировать критический сбой модуля от платформы Techlog.

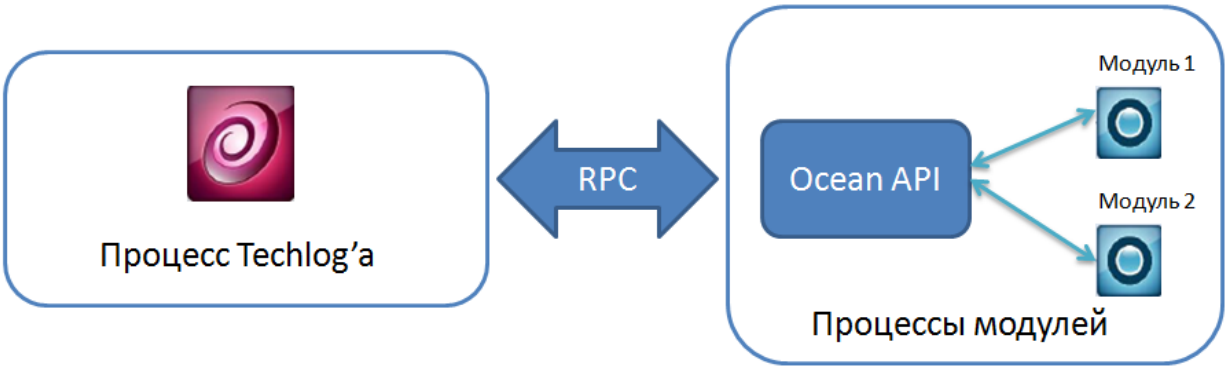

Рисунок 5. Архитектура Ocean for Techlog.

API обеспечивает основной функционал работы с объектами Techlog, а именно:

- Доменными объектами и их исходным данным;
- Графическим окружением, в котором отображаются исходные данные;

- Пользовательским интерфейсом модуля.

Ocean API позволяет получить доступ к следующим типам и свойствам модели данных Techlog:

- Скважина (Well);
- Набор данных (Dataset);
- Данные (Variable);
- Свойства данных (Data properties) [8].

Модули разрабатываются на основе Qt. Ocean-фреймворк содержит Qt версии 5.6.1, расположенных в двух папках. В папке v120 находятся Qt библиотеки совместимые с Visual Studio 2013, в папке v140 - библиотеки совместимые с Visual Studio 2015 (рис. 6). Таким образом, для разработки модуля можно использовать 2 версии Visual Stuido: 2013 и 2105. Также, в состав входят библиотеки для создания unit-тестов. Для тестирования используются Google-тесты версии 1.6.0 [8].

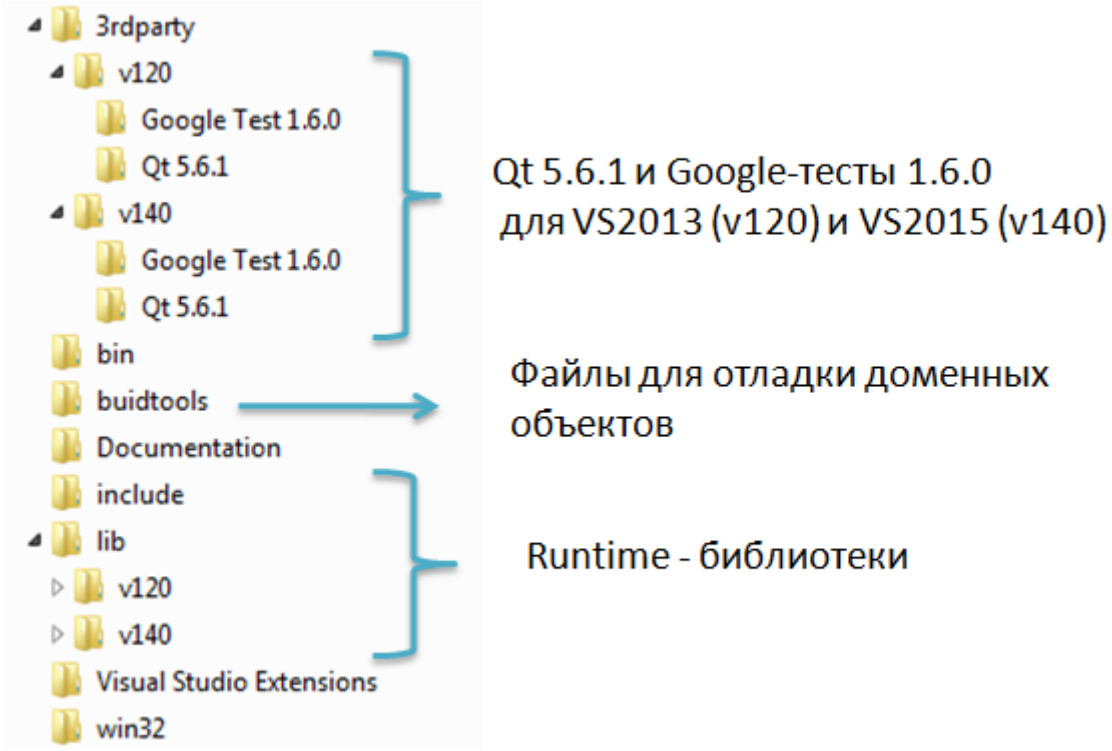

Рисунок 6. Содержание Ocean for Techlog.

1.5.2 Архитектура модуля.

После установки Ocean for Techlog в Visual Studio появится возможность создания собственного модуля для Techlog. Для этого необходимо:

- Выбрать Файл >Создать >Проект;
- В вкладке Visual C++ выбрать OceanTechlog 2017.1;
- Выбрать Ocean Plug-in.

Проект, как минимум, содержит два класса. Один класс содержит информацию о плагине: имя плагина, версия, описание, дата релиза, разработчик. Второй класс, содержит основной функционал плагина.

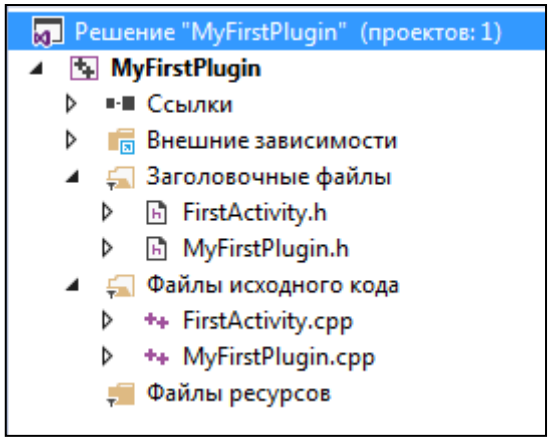

Рисунок 7. Исходные файлы плагина.

На рис. 7 класс, содержащий информацию о плагине реализован в файлах MyFirstPlugin.h и MyFirstPlugin.cpp. Данный класс наследуется от PluginIdentity (рис. 8).

```
class PluginIdentity
{
public:
    virtual void getInformation(PluginInformation &pluginInformation) const = 0;
    virtual void getActivities(PluginActivities &activities) const = 0;
    virtual void getMenu(PluginMenu &menu) const = 0;
};
```

Рисунок 8. Класс PluginIdentity.

- метод getInformation определяет основную информацию о плагине;
- метод getActivities добавляет необходимое количество пунктов в меню Techlog;
- метод getMenu описывает меню плагина, пример показан на рис. 9.

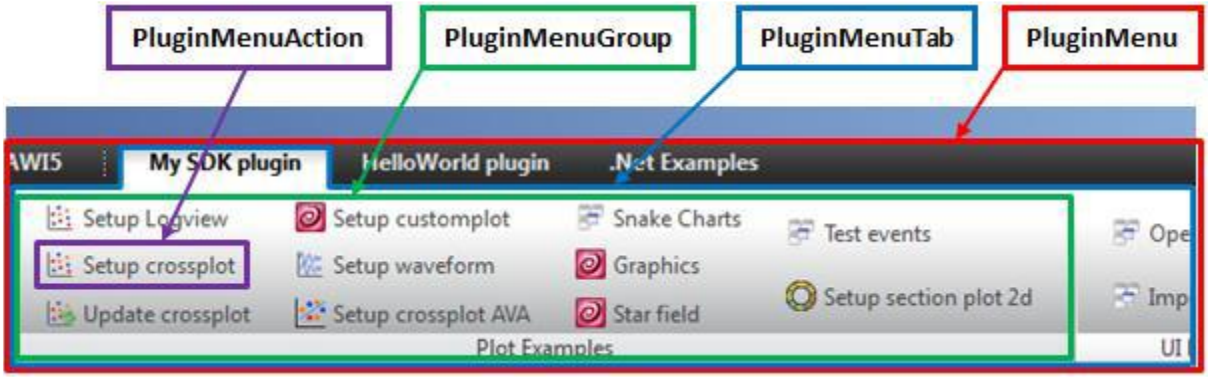

Рисунок 9.Примерменю плагина в Techlog.

- PluginMenu - основное меню платформы Techlog;
- PluginMenuTab - вкладка плагина;
- PluginMenuGroup - группы, расположенная на вкладке плагина;
- PluginMenuAction - пункт меню, выполняющий определенный функционал.

Для каждого пункта в меню, описанного в методе getActivities, необходимо создать собственный класс, который реализует основной функционал. Данный класс должен наследоваться от класса AbstractActivity.

```
class AbstractActivity
{
public:
    virtual void defineMethod(MethodDefinition &methodDefinition) const = 0;
    virtual void preliminaryExecute(MethodRealization &methodRealization) const = 0;
    virtual void refreshDisplay(MethodRealization &methodRealization) const = 0;
    virtual void execute(MethodRealization &methodRealization) const = 0;
};
```

Рисунок 10. Класс AbstractActivity.

- метод defineMethod необходим для определения типа входных и выходных данных;

- метод preliminaryExecute выполняется в случае изменения входных данных. Задача метода состоит в проверке входных данных на соответствие определенным требованиям;
- метод refreshDisplay отвечает за визуализацию. Отображение данных возможно как с использованием объектов Techlog'a, так и с созданием собственных форм на основе Qt.
- метод execute необходим для описание алгоритмов расчета выходных данных.

ГЛАВА 2. РАЗРАБОТКА МОДУЛЯ ДЛЯ ПЛАТФОРМЫ TECHLOG, РЕАЛИЗУЮЩЕГО ПРЕДОБРАБОТКУ ДАННЫХ КАПИЛЛЯРНОГО ДАВЛЕНИЯ И ОТНОСИТЕЛЬНОЙ ФАЗОВОЙ ПРОНИЦАЕМОСТИ.

Целью работы является создание модуля для программного комплекса Techlog, который реализует предобработку данных капиллярного давления и относительной фазовой проницаемости.

2.1 Постановка задачи.

Для платформы Techlog необходимо создать модуль, состоящий из нескольких подмодулей:

- Обработка данных капиллярного давления типа MICP;
- Приведение данных капиллярного давления в пластовые условия;
- Вычисление осредняющей кривой для капиллярного давления;
- Вычисление осредняющей кривой для относительной фазовой проницаемости;
- Моделирование кривых относительных фазовых проницаемостей.

Каждый подмодуль представляет из себя класс, наследованный от AbstractActivity. Данный подход позволяет сделать вышеперечисленные подмодули независимыми друг от друга.

Состав подмодуля:

- Список необходимых входных данных;
- Описание выходных данных;
- Алгоритмы проверки входных данных;
- Алгоритм расчета выходных данных;
- Визуализация входных и выходных данных;

Входными и выходными данными для каждого подмодуля являются данные проекта Techlog. Работа каждого подмодуля в общем виде представлена на рис. 12.

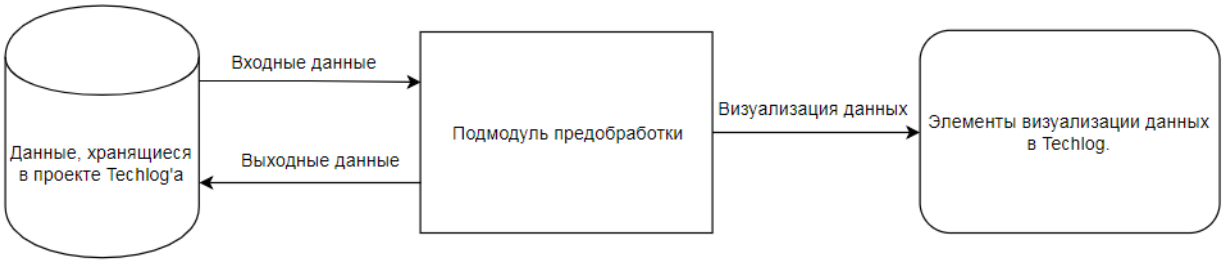

Рисунок 11. Работа подмодуля с данными.

2.2 Разработка вспомогательных классов.

2.2.1 Модель входных и выходных данных.

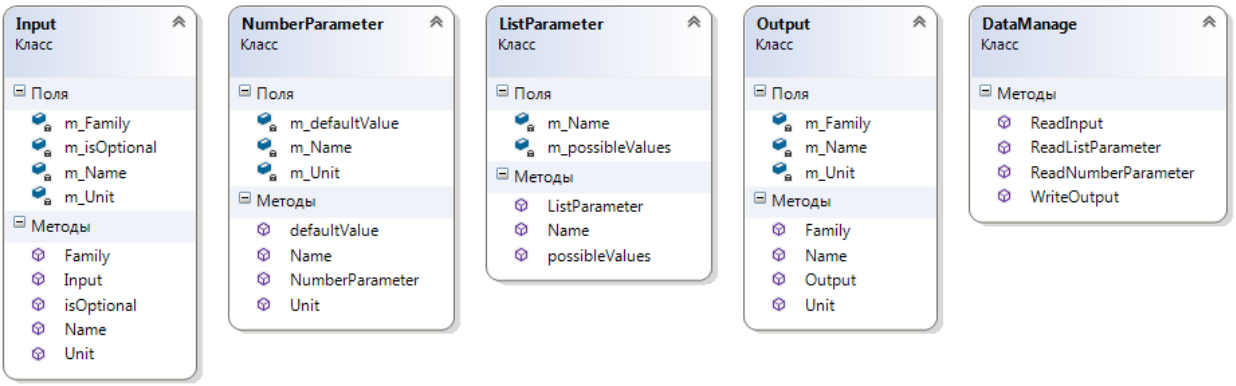

Рисунок 12. Диаграмма классов, описывающих входные и выходные данные.

Для описание входных данных, поступающих с проекта Techlog, создан класс Input. Состав класса:

Поля класса Input:

- QString m_Name - поле, которое содержит идентификатор входных данных;
- QString m_Family - поле, содержащее название типа, к которому относятся данные;
- QString m_Unit - поле, которое содержит единицу измерения входных данных;
- bool m_isOptional - поле, указывающее является ли входные данные обязательными для расчетов.

Конструкторы класса Input:

Input(const QString& Name, const QString& Family, const QString& Unit, const bool& isOptional) - конструктор, заполняющий значениями все поля класса.

Методы класса Input:

- QString Name() - возвращает поле m_Name;
- QString Family() - возвращает поле m_Family;
- QString Unit() - возвращает поле m_Unit;
- bool isOptional() - возвращает поле m_isOptional

В Ocean SDK существует возможность создания на форме подмодуля выпадающего списка и поле для ввода числовых значений. Для выпадающих списков подмодуля создан класс ListParameter, а для полей, в которых можно вводить числовые значения - класс NumberParameter.

Поля класса ListParameter:

- QString m_Name - поле, которое содержит идентификатор выпадающего списка;
- QStringList m_possibleValues - поле, содержащее список значений выпадающего списка.

Конструктор класса ListParameter:

ListParameter(const QString& Name, const QStringList& possibleValues) - конструктор, заполняющий все поля класса.

Методы класса ListParameter:

- QString Name() - возвращает значение m_Name;
- QStringList possibleValues() - возвращает список значений m_possibleValues.

Поля класса NumberParameter:

- QString m_Name - идентификатор параметра;
- double m_defaultValue - значение по умолчанию;
- QString m_Unit - единица измерения для числового поля.

Конструктор класса NumberParameter:

NumberParameter(const QString& Name, const QString& Unit, const double& defaultValue) - конструктор, который устанавливает значения для всех полей класса.

Методы класса NumberParameter:

- QString Name() - возвращает значение m_Name;
- double defaultValue() - возвращает значение m_ defaultValue;
- QString Unit() - возвращает значение m_ Unit.

Для описания выходных данных разработан класс Output. Данный класс состоит из следующих компонентов:

Поля класса Output:

- QString m_Name - имя выходных данных;
- QString m_Family - поле, описывающее тип выходных данных;
- QString m_Unit - поле, содержащее единицу измерения выходных данных.

Конструктор класса Output:

Output(const QString& Name, const QString& Family, const QString& Unit) - конструктор, который устанавливает значения для всех полей класса.

Методы класса Output:

- QString Name() - возвращает значение m_Name;
- QString Family() - возвращает значение m_Family;
- QString Unit() - возвращает значение m_Unit.

Класс DataManage содержит статические методы, при помощи которых можно получить значения входных данных и сохранить выходные данные в проект Techlog.

Методы класса DataManage:

- static QVector<double> ReadInput(MethodRealization &methodRealization, const Input& input) - возвращает массив чисел входных данных подмодуля;
- static double ReadNumberParameter(MethodRealization &methodRealization, const NumberParameter& nParam) - возвращает текущее значение числового поля nParam;
- static QString ReadListParameter(MethodRealization &methodRealization, const ListParameter& lParam) - возвращает текущее значение выпадающего списка lParam;
- static QVector<double> WriteOutput(MethodRealization &methodRealization, const Output& output, const QVector<double>& Values) - создает в проекте Techlog, выходные данные с значениями Values.

2.2.2 Модель представления кривой.

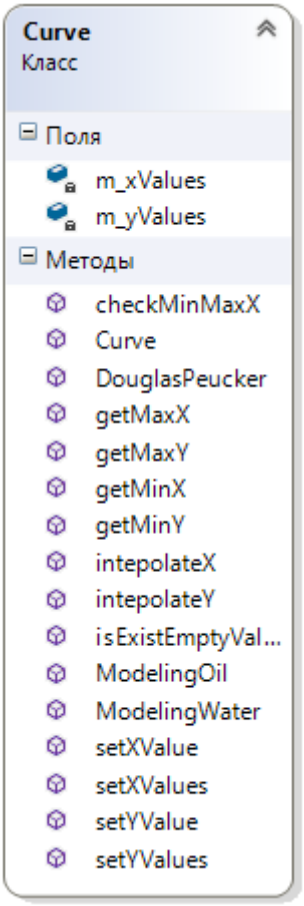

Рисунок 13. Диаграмма класса Curve.

Для представления кривой разработан класс Curve.

Поля класса Curve:

- QVector<double> m_xValues - содержит значения по оси X;
- QVector<double> m_yValues - содержит значения по оси Y.

Конструкторы класса Curve:

- Curve(const QVector<double>& xValues, const QVector<double>& yValues) - создает экземпляр класса с указанными значениями полей.

Методы класса Curve:

- double getMinX() - возвращает минимальное значение по оси X;
- double getMinY() - возвращает минимальное значение по оси Y;

- double getMaxX() - возвращает максимальное значение по оси X;
- double getMaxY() - возвращает максимальное значение по оси Y;
- void setXValues(const QVector<double>& values) - определяет новые значения по оси X;
- void setYValues(const QVector<double>& values) - определяет новые значения по оси Y;
- void setXValue(const double& value, const int& index) - изменяет значение по оси X с указанным индексом;
- void setYValue(const double& value, const int& index) - изменяет значение по оси Y с указанным индексом;
- double intepolateX(int x) - реализует линейную интерполяцию, возвращает значение Y интерполированной точки;
- double intepolateY(int y) - реализует линейную интерполяцию, возвращает значение X интерполированной точки;
- bool isExistEmptyValue() - возвращает true, если в m_ xValues или в m_yValues присутствует значение -9999, при обратной ситуации возвращает false;
- bool checkMinMaxX(double min, double max) – возвращает true, если минимальное значение m_ xValues больше или равно значению min, а максимальное значение меньше или равно значению max, при обратной ситуации - false;
- void DouglasPeucker(const int& eps) - реализация алгоритма Рамера-Дугласа-Пекера;
- double ModelingOil() - вычисляет параметр модели (11), пересчитывает значения m_ yValues, используя вычисленный параметр модели и формулу (11);
- double ModelingWater() - вычисляет параметр модели (12), пересчитывает значения m_ yValues, используя вычисленный параметр модели и формулу (12);

2.2.3 Алгоритмы предобработки.

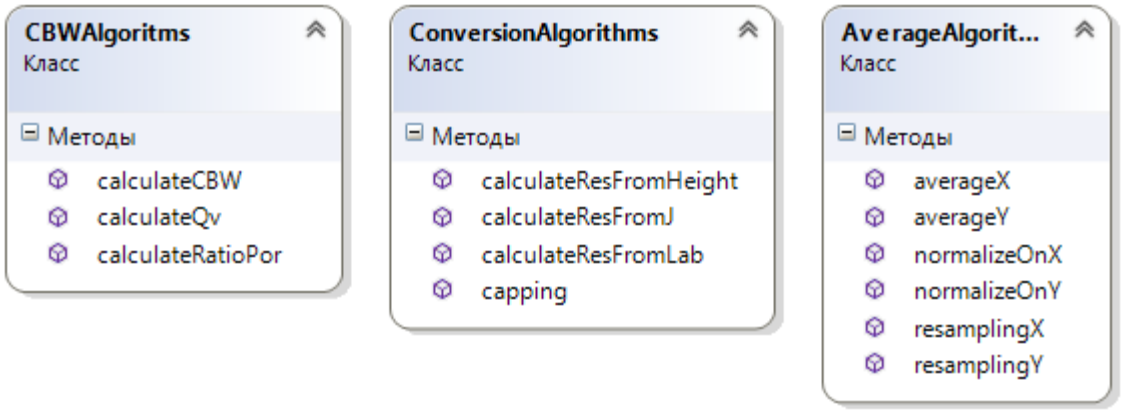

Рисунок 14. Диаграмма классов, содержащих алгоритмы предобработки.

Для обработки данных капиллярного давления типа MICP разработан класс CBWAlgoritms, включающий следующие методы:

- static QVector<double> calculateQv(QVector<double> porosity, int a, int b) - вычисления значений Qv поформуле (5);
- static QVector<double> calculateRatioPor(QVector<double> qvValues, QVector<double>salValues) - вычисления отношений эффективной пористости к общей по формуле (4);
- static Curve calculateCBW(constCurve&curve) - вычисление значений капиллярного давления и водонасыщенности по формулам (2) и (3).

Методы алгоритма приведения данных капиллярного давления к пластовым условиям описаны в классе ConversionAlgorithms:

- static QVector<double> calculateResFromLab(QVector<double> values, double theta_res, double theta_lab, double cos_res, double cos_lab) - расчет капиллярного давления в пластовых условиях по формуле (8);
- static QVector<double> calculateResFromHeight(QVector<double> values, double deltaP) - расчет капиллярного давления в пластовых условиях, используя формулу (6);
- static QVector<double> calculateResFromJ(QVector<double> values, QVector<double> por, QVector<double> perm, double theta_res, double

cos_res) - расчет капиллярного давления в пластовых условиях, используя формулу (7);

- static QVector<Curve> capping(QVector<Curve> curves, double maxValue) - установление единого максимума по оси Y с использованием линейной интерполяции и экстраполяции.

Алгоритм расчета осредняющей кривой реализован в классе AverageAlgorithms:

- static Curve normalizeOnX(const Curve& curve) – нормализация кривой по оси X;
- static Curve normalizeOnY(const Curve& curve) – нормализация кривой по оси Y;
- static QVector<Curve> resamplingX(QVector<Curve> curves, QVector<double> newXValues) – перерасчет каждой кривой при помощи интерполяции, для того чтобы у всех кривых был одинаковый набор значений по оси X.
- static QVector<Curve> resamplingY(QVector<Curve> curves, QVector<double> newYValues) – перерасчет каждой кривой при помощи интерполяции, для того чтобы у всех кривых был одинаковый набор значений по оси Y.
- static Curve averageX(QVector<Curve> curves) – вычисление осредненяющей кривой по оси X;
- static Curve averageY(QVector<Curve> curves) – вычисление осредненяющей кривой по оси Y.

2.2 Разработка модуля.

Для разработки модуля для платформы Techlog создан класс PreprocessingModule, наследуемый от класса PluginIdentity, и переопределены методы:

- void getInformation(PluginInformation&) - описание основной информации о модуле (название модуля, версия, дата релиза, разработчик модуля);
- void getActivities(PluginActivities&) - определение всех подмодулей;
- void getMenu(Menu&) - определение расположения меню запуска всех подмодулей в панели управления Techlog.

Для создания подмодулей реализованы следующие классы, наследованные от класса AbstractActivity:

- CBWCorrection;
- Conversion;
- AverageCP;
- AveragerKR;
- ModelingKR.

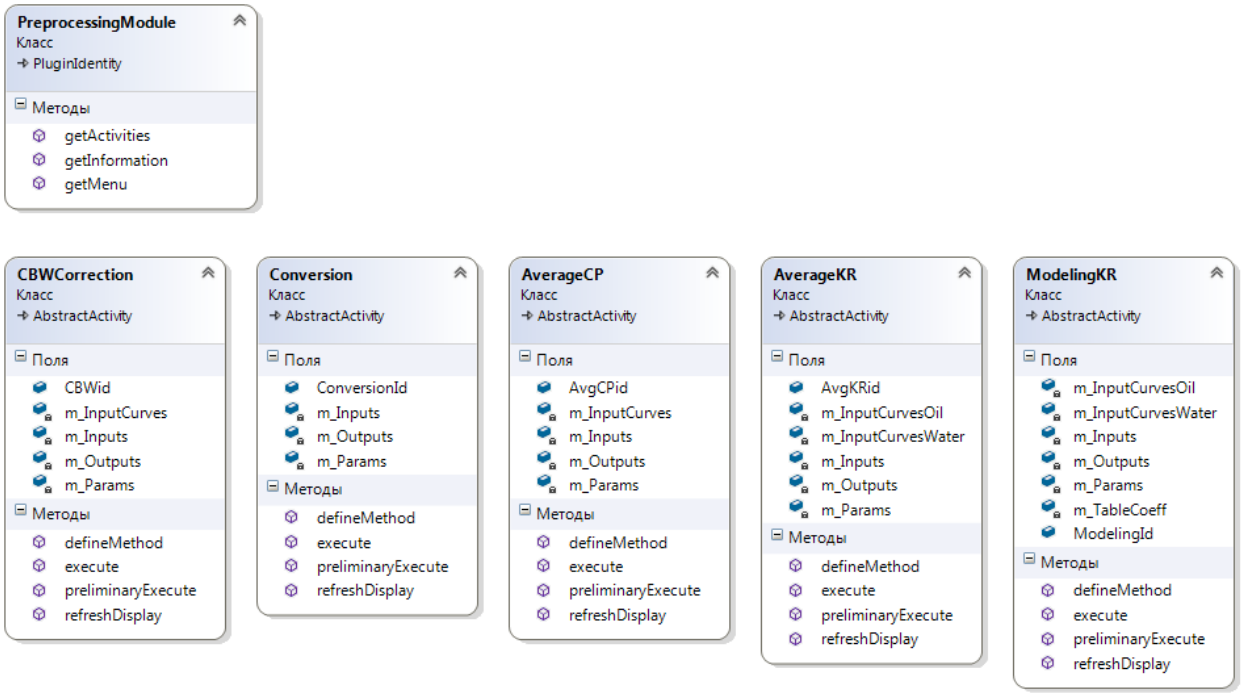

Рисунок 15. Диаграмма классов модуля.

2.2.1 Класс CBWCorrection.

Класс CBWCorrection описывает работу подмодуля, реализующего обработку данных капиллярного давления типа MICP.

Поля класса:

- QVector<Curve> m_InputCurves – входные данные капиллярного давления;
- static QString CBWid - идентификатор подмодуля;
- QVector<Input> m_Inputs - список входных данных, описанный в таблице 2;
- QVector<NumberParameter> m_Params - список числовых полей, описанный в таблице 3;
- QVector<Output> m_Outputs - список выходных данных, описанный в таблице 4.

Таблица 2.

Значения полей списка QVector<Input> m_Inputs.

Входные данные	**m_Name**	**m_Family**	**m_Unit**	**m_isOptional**
Капиллярное давление	PC_LAB	Capillaty Pressure Laboratory	Bar	False
Водонасыщенность	CSW	Water saturation	m^3/m^3	False
Пористость	Porosity	Total porosity	m^3/m^3	False

Таблица 3.

Значения полей списка QVector<NumberParameter> m_Params.

Параметр	**m_Name**	**m_defaultValue**	**m_Unit**
Минерализация	Salinity	1.711	eq/L
Коэффициент A	A	0.01	unitless
Коэффициент B	B	1	unitless

Таблица 4**.**

Значения полей списка QVector<Output> m_Outputs.

Выходные данные	**m_Name**	**m_Family**	**m_Unit**
Обработанные данные капиллярное давление	PC_CBW	Capillaty Pressure Laboratory	Bar

Обработанные данные водонасыщенности	CSW_CBW	Water saturation	m^3/m^3

Методы класса:

- void defineMethod(MethodDefinition&) - определяет входные и выходные данные, а также числовые поля для подмодуля CBWCorrection. Входные данные описаны в m_Inputs, выходные данные описаны в m_Outputs, числовые поля описаны в m_Params.
- void preliminaryExecute(MethodRealization &methodRealization) - в данном методе происходит создание экземпляров класса Curve и сохранение их в список m_InputCurves, в качестве значений по оси Y используется капиллярное давления, в качестве значений по оси X - значения водонасыщенности. Реализация проверок начальных данных на наличие:
    - Пустых значений в данных капиллярного давления;
    - Пустых значений в данных водонасыщенности;
    - Вхождения всех значений водонасыщенности в интервал (0, 1).

  Данные проверки реализованы в классе Curve.
- void execute(MethodRealization &methodRealization) - происходит основной расчет выходных значений с использованием разработанного класса CBWAlgoritms. После получения результативных значений происходит сохранение значений в проект Techlog'a с помощью класса DataManage;
- void refreshDisplay(MethodRealization &methodRealization) - визуализация входных и выходных данных капиллярного давления и водонасыщенности на графике.

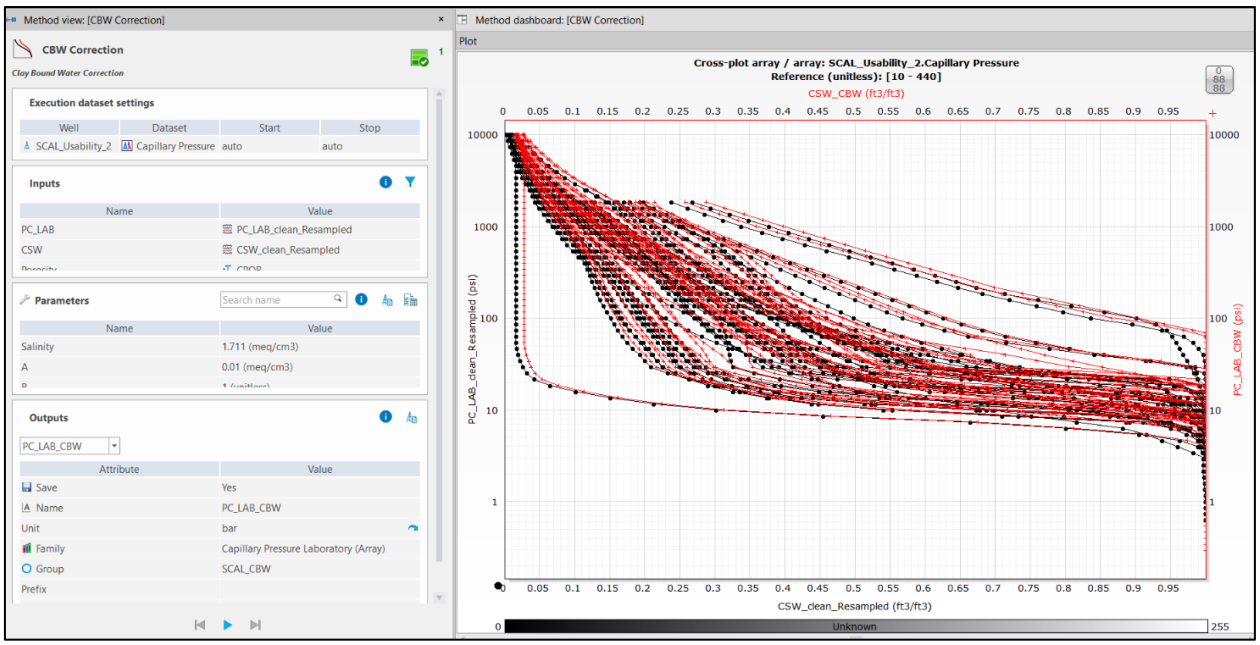

Рисунок 16. Интерфейс подмодуля CBWCorrection.

2.2.2 Класс Conversion.

Данный класс описывает подмодуль приведения капиллярного давления в пластовые условия.

Поля класса:

- static QString ConversionId - идентификатор подмодуля;
- QVector<Input> m_Inputs - список входных данных, описанный в таблице 5;
- QVector<NumberParameter> m_Params - список числовых полей, описанный в таблице 6;
- QVector<Output> m_Outputs - список выходных данных, описанный в таблице 7.

Таблица 5.

Значения полей списка QVector<Input> m_Inputs.

Входные данные	**m_Name**	**m_Family**	**m_Unit**	**m_isOptional**
Капиллярное давление	PC_LAB	Capillaty Pressure Laboratory	Bar	True
Водонасыщенность	CSW_LAB	Water saturation	m^3/ m^3	False
Значения функции J-Leverett	J	J Leverett	unitless	True
Высота над свободным уровнем воды	HAFWL	Height Above Free Water Level	M	True
Пористость	Porosity	Porosity	m^3/ m^3	True
Проницаемость	Permeability	Permeability	mD	True

Таблица 6.

Значения полей списка QVector<NumberParameter> m_Params.

Параметр	**m_Name**	**m_defaultValue**	**m_Unit**
Коэффициент поверхностного натяжения в пластовых условиях	IFT_RES	30	dyne/cm
Коэффициент поверхностного натяжения в лабораторных исследованиях	IFT_LAB	480	dyne/cm

Косинус угла смачивания для капиллярного давления из лабораторных исследований	Cos_theta_lab	0.766	Unitless
Косинус угла смачивания для капиллярного давления в пластовых условиях	Cos_theta_res	1	Unitless
Плотность воды	Water density	1	g/cm^3
Плотность нефти	Oil density	0.8	g/cm^3
Максимальное значение давления	PC_Cap	2	Bar

Таблица 7.

Значения полей списка QVector<Output> m_Outputs**.**

Выходные данные	**m_Name**	**m_Family**	**m_Unit**
Капиллярное давление в пластовых условиях	PC_RES	Capillaty Pressure Reservoir	Bar
Данные водонасыщенности	CSW_RES	Water saturation	m^3/ m^3

Методы класса:

- void defineMethod(MethodDefinition&) - определяет входные и выходные данные, а также числовые поля для подмодуля Conversion. Входные

данные описаны в списке m_Inputs, выходные данные описаны в списке m_Outputs, числовые поля описаны в m_Params.

- void preliminaryExecute(MethodRealization &methodRealization) - реализует основные проверки входных данных, а именно
    - Проверка на наличие пустых значений в входных данных;
    - Проверка значений водонасыщенности на вхождение в интервал (0, 1).
- void execute(MethodRealization &methodRealization) - реализует расчет капиллярного давления в пластовых условиях. Алгоритмы расчета реализованы в классе ConversionAlgorithms. Полученные таким образом данные капиллярного давления и водонасыщенности сохраняются в проект Techlog'а с помощью класса DataManage.
- void refreshDisplay(MethodRealization &methodRealization) - визуализация выходных данных капиллярного давления и водонасыщенности на графике.

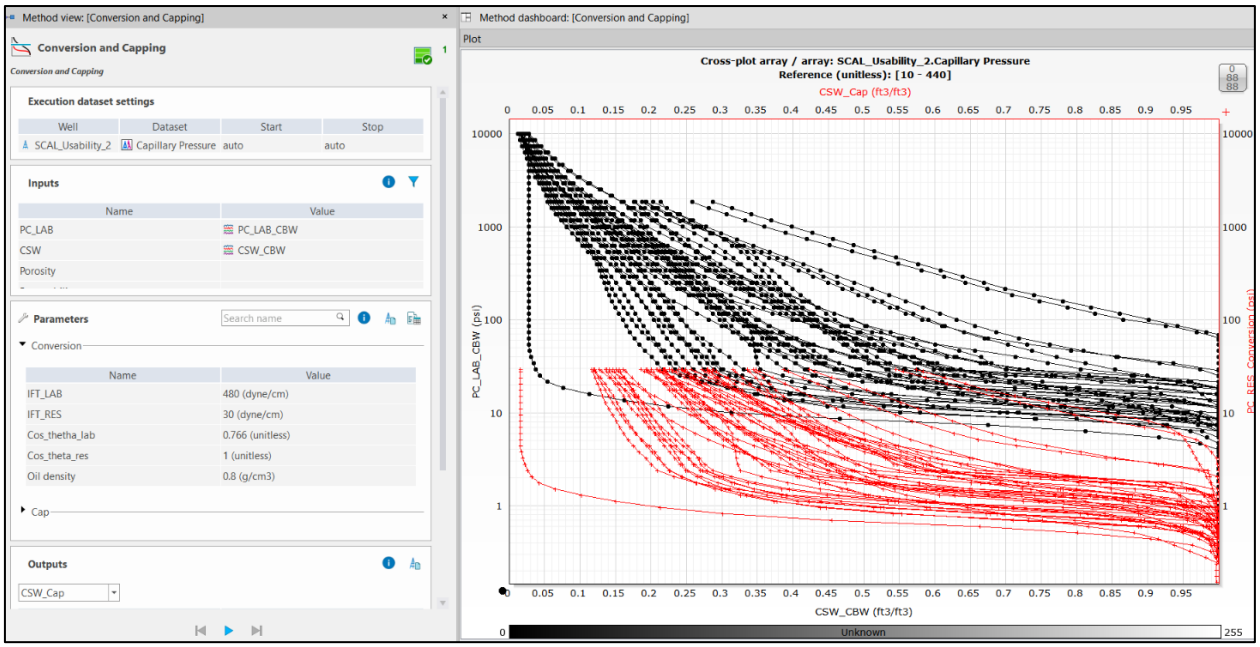

Рисунок 17. Интерфейс подмодуля Conversion.

2.2.3 Класс AverageCP.

Данный класс описывает подмодуль нормализации данных и расчета осредняющих кривых для каждой группы кривых капиллярного давления. Для

определения кривых к группам, используется входной параметр Group (табл. 8), который содержит соответствие «индекс кривой» – «группа».

Поля класса:

- QVector<Curve> m_InputCurves – начальные данные капиллярного давления;
- static QString CBWid - идентификатор подмодуля;
- QVector<Input> m_Inputs - список входных данных, описанный в таблице 8;
- QVector< ListParameter> m_Params - список параметров, описанный в таблице 9;
- QVector<Output> m_Outputs - список выходных данных, описанный в таблице 10.

Таблица 8.

Значения полей списка QVector<Input> m_Inputs.

Входные данные	**m_Name**	**m_Family**	**m_Unit**	**m_isOptional**
Капиллярное давление в пластовых условиях	PC_Res	Capillaty Pressure Reservoir	bar	False
Водонасыщенность	CSW	Water saturation	m^3/m^3	False
Группы кривых капиллярного давления	Group	Group	unitless	False

Таблица 9.

Значения полей списка QVector< ListParameter> m_Params.

Параметр	**m_Name**	**m_possibleValues**
Тип осреднения	Average	On X On Y
Нормализация	Normalization	Yes No

Таблица 10.

Значения полей списка QVector<Output> m_Outputs.

Выходные данные	**m_Name**	**m_Family**	**m_Unit**
Осредненные данные капиллярного давление в пластовых условиях	PC_RES_Avg	Capillaty Pressure Reservoir	Bar
Осредненные данные водонасыщенности	CSW_RES_Avg	Water saturation	m^3/m^3

Методы класса:

- void defineMethod(MethodDefinition&) - определяет входные и выходные данные, а также выпадающие списки для подмодуля AverageCP. Входные данные описаны в списке m_Inputs, выходные данные описаны в списке m_Outputs, выпадающие списки описаны в m_Params.
- void preliminaryExecute(MethodRealization &methodRealization) - в данном методе происходит создание экземпляров класса Curve и

сохранение их в список m_InputCurves, в качестве значений по оси Y используется капиллярное давления, в качестве значений по оси X - значения водонасыщенности. Реализация проверок начальных данных на наличие:

- Пустых значений в данных капиллярного давления;
- Пустых значений в данных водонасыщенности;
- Вхождения всех значений водонасыщенности в интервал (0, 1).

Данные проверки реализованы в классе Curve.

- void execute(MethodRealization &methodRealization) - выполняет алгоритм нормализации (если в параметре Normalization указано Yes) для кривых капиллярного давления, затем выполняется вычисление осредненяющей кривой для каждой группы кривых капиллярного давления. Данные алгоритмы реализованы в классе AverageAlgorithms. После расчета осредняющих кривых, выполняется алгоритм Рамера-Дугласа-Пекера для удаления избыточных точек с кривой.
- void refreshDisplay(MethodRealization &methodRealization) - визуализация входных данных капиллярного давления и водонасыщенности и осредняющих кривых на графике.

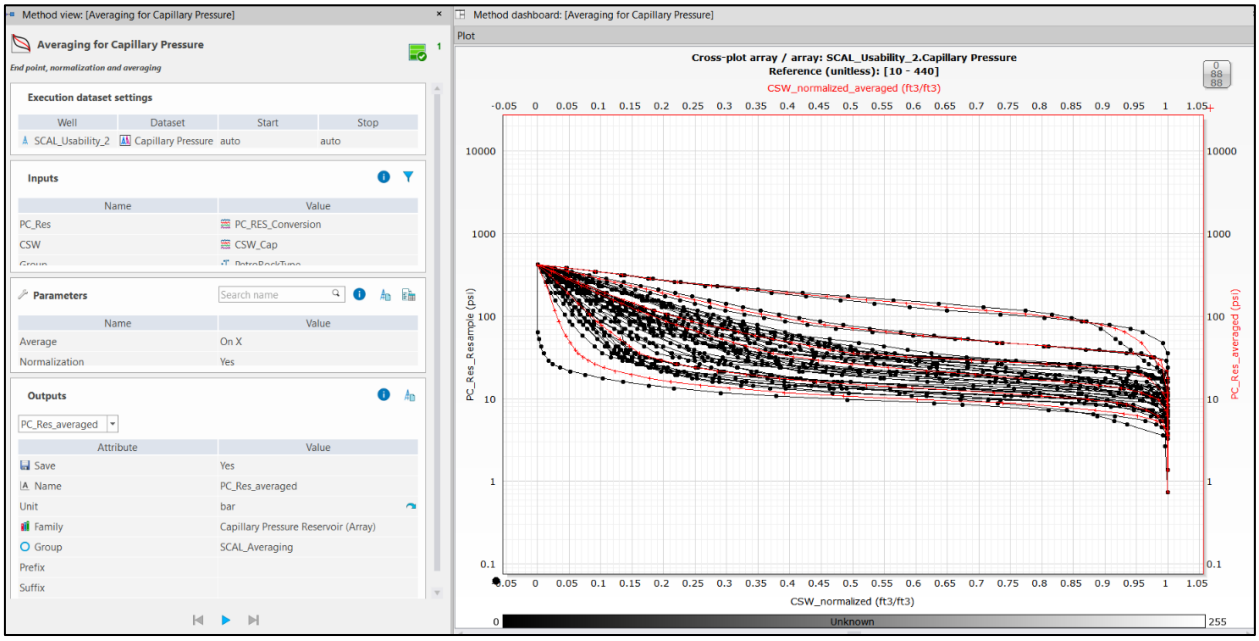

Рисунок 18. Интерфейс подмодуля AverageCP.

2.2.4 Класс AverageKR.

Данных класс описывает подмодуль осуществляющий нормализацию и расчет осредняющих кривых относительных фазовых проницаемостей в отдельности для каждой группы. Соответствие кривой относительной фазовой проницаемости к группе указано в входном параметре Group (табл. 11).

Поля класса:

- QVector<Curve> m_InputCurveOil – входные данные относительной проницаемости для нефти;
- QVector<Curve> m_InputCurveWater – входные данные относительной проницаемости для воды;
- static QString AvgKRid - идентификатор подмодуля;
- QVector<Input> m_Inputs - список входных данных, описанный в таблице 11;
- QVector<ListParameter> m_Params - список параметров, описанный в таблице 12;
- QVector<Output> m_Outputs - список выходных данных, описанный в таблице 13.

Таблица 11.

Значения полей списка QVector<Input> m_Inputs.

Входные данные	**m_Name**	**m_Family**	**m_Unit**	**m_isOptional**
Относительная проницаемость для нефти	CKRO	Relative Permeability	unitless	False
Относительная проницаемость для воды	CKRW	Relative Permeability	unitless	False

Водонасыщенность	CSW	Water saturation	v/v	False
Группы кривых относительных фазовых проницаемостей	Group	Group	unitless	False

Таблица 12.

Значения полей списка QVector< ListParameter> m_Params.

Параметр	**m_Name**	**m_possibleValues**
Тип осреднения	Average	On X On Y
Нормализация	Normalization	Yes No

Таблица 13.

Значения полей списка QVector<Output> m_Outputs.

Выходные данные	**m_Name**	**m_Family**	**m_Unit**
Осредненные относительные проницаемости по фазе нефть	CKRO_Avg	Relative permeability	Unitless
Осредненные относительные проницаемости по фазе вода	CKRW_Avg	Relative permeability	Unitless
Осредненные данные водонасыщенности	CSW_Avg	Water saturation	m^3/m^3

Методы класса:

- void defineMethod(MethodDefinition&) - определяет входные и выходные данные, а также выпадающие списки для подмодуля AverageKR. Входные данные описаны в списке m_Inputs, выходные данные описаны в списке m_Outputs, выпадающие списки описаны в m_Params.
- void preliminaryExecute(MethodRealization &methodRealization) – в данном методе происходит создание экземпляров класса Curve. Для m_InputCurveWater в качестве значений по оси Y используется относительная пронциаемость для воды, а в качестве значений по оси X используются значения водонасыщенности. Для m_InputCurveOil в качестве значений по оси Y используется относительная пронциаемость для нефти, а в качестве значений по оси X используются значения водонасыщенности. Также данный метод выполняет следующие проверки:
    - Проверка входных данных на наличие пустых значений;
    - Проверка значений водонасыщенности на вхождение в интервал (0, 1)
    - Проверка значений относительной проницаемости для нефти на вхождение в интервал (0,1);
    - Проверка значений относительной проницаемости для воды на вхождение в интервал (0,1).

  Данные проверки реализованы в классе Curve.

- void execute(MethodRealization &methodRealization) - выполняет алгоритм нормализации (если в параметре Normalization указано Yes), затем выполняется вычисление осредненяющей кривой отдельно для относительных проницаемостей для воды и отдельно для относительных проницаемостей для нефти. Данные алгоритмы реализованы в классе AverageAlgorithms. После расчета осредняющих кривых, выполняется

алгоритм Рамера-Дугласа-Пекера для удаления избыточных точек с кривой.

- void refreshDisplay(MethodRealization &methodRealization) - визуализация нормализованных входных данных относительных фазовых проницаемостей и осредненных кривых на графике.

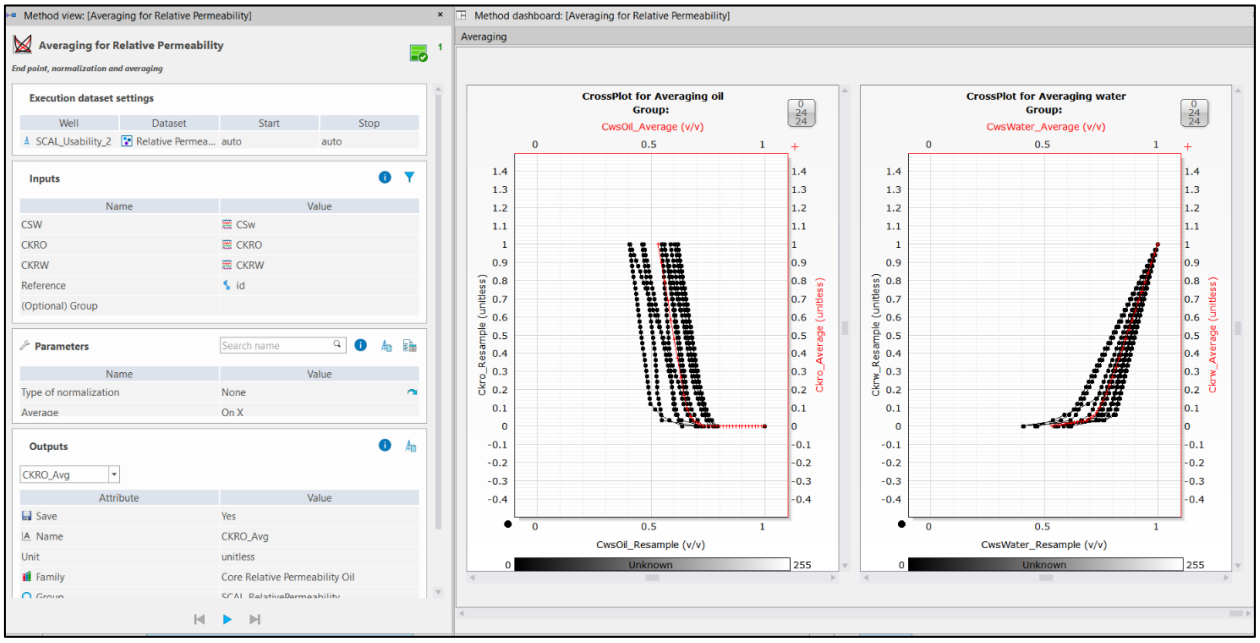

Рисунок 19. Интерфейс подмодуля AverageKR.

2.2.5 Класс ModelingKR.

Данные класс реализует подмодуль моделирования кривых относительных фазовых проницаемостей.

Поля класса:

- QVector<Curve> m_InputCurveOil – входные данные относительной проницаемости для нефти;
- QVector<Curve> m_InputCurveWater – входные данные относительной проницаемости для воды;
- static QString ModelingId - идентификатор подмодуля;
- QVector<Input> m_Inputs - список входных данных, описанный в таблице 14;

- QVector<Output> m_Outputs - список выходных данных, описанный в таблице 15;
- QTableWidget* m_ModelingCoeff – таблица содержащая параметры модели Corey для каждой кривой относительной проницаемости.

Таблица 14.

Значения полей списка QVector<Input> m_Inputs.

Входные данные	**m_Name**	**m_Family**	**m_Unit**	**m_isOptional**
Относительная проницаемость для нефти	CKRO	Relative Permeability	unitless	False
Относительная проницаемость для воды	CKRW	Relative Permeability	unitless	False
Водонасыщенность	CSW	Water saturation	v/v	False

Таблица 15.

Значения полей списка QVector<Output> m_Outputs.

Выходные данные	**m_Name**	**m_Family**	**m_Unit**
Смоделированные данные относительной проницаемости для нефти	CKRO_Modeled	Relative permeability	Unitless
Смоделированные данные относительной	CKRO_Modeled	Relative permeability	Unitless

проницаемости для воды			
Данные водонасыщенности	CSW_Modeled	Water saturation	m^3/m^3

Методы класса:

- void defineMethod(MethodDefinition&) - определяет входные и выходные данные ModelingKR. Входные данные описаны в списке m_Inputs, выходные данные описаны в списке m_Outputs.
- void preliminaryExecute(MethodRealization &methodRealization) – в данном методе происходит создание экземпляров класса Curve. Для m_InputCurveWater в качестве значений по оси Y используется относительная пронциаемость для воды, а в качестве значений по оси X используются значения водонасыщенности. Для m_InputCurveOil в качестве значений по оси Y используется относительная пронциаемость для нефти, а в качестве значений по оси X используются значения водонасыщенности. Также данный метод выполняет следующие проверки:
  - Проверка входных данных на наличие пустых значений;
  - Проверка значений водонасыщенности на вхождение в интервал (0, 1);
  - Проверка значений относительной проницаемости для нефти на вхождение в интервал (0, 1);
  - Проверка значений относительной проницаемости для воды на вхождение в интервал (0, 1).

  Данные проверки реализованы в классе Curve.
- void execute(MethodRealization &methodRealization) – выполняет расчет параметров модели (11) для кривых относительных проницаемостей для нефти и (12) для кривых относительных проницаемостей для воды.

Данные расчеты реализованы в классе Curve. Полученные параметры модели сохраняются в таблицу m_ModelingCoeff.

- void refreshDisplay(MethodRealization &methodRealization) – визуализация начальных и результативных данных относительных фазовых проницаемостей на графике. Визуализация таблицы m_ModelingCoeff.

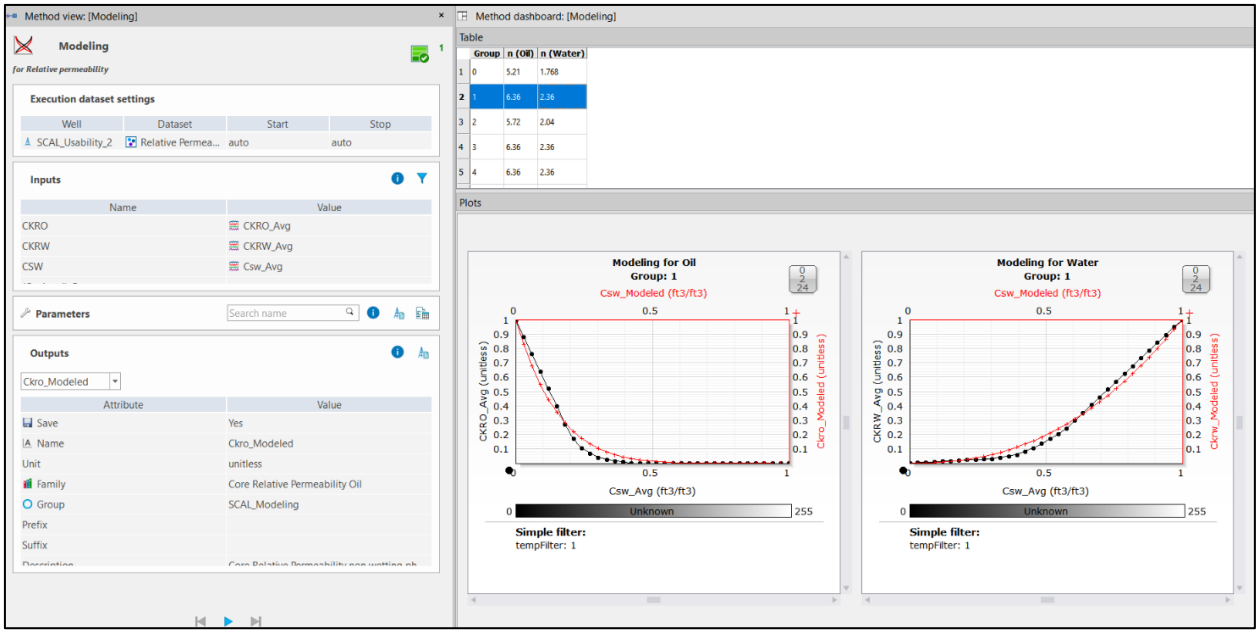

Рисунок 20. Интерфейс подмодуля ModelingKR.

ЗАКЛЮЧЕНИЕ

В ходе выполнения магистерской диссертации был изучен процесс предобработки данных капиллярного давления и относительной фазовой проницаемости, который состоит из:

- Обработка капиллярного давления типа MICP;
- Приведение данных капиллярного давления к пластовым условиям;
- Нормализация и осреднение кривых капиллярного давления;
- Нормализация и осреднение кривых относительных фазовых проницаемостей;
- Моделирование кривых относительных фазовых проницаемостей.

Во время изучения процесса предобработки был выявлен ряд инцидентов обработки данных. Входные данные могут содержать пропущенные значения. Значения водонасыщенности и относительных фазовых проницаемостей могут выходить за область допустимых значений. Чтобы избежать обработки данных с некорректными значениями, добавлены соответствующие проверки перед выполнением алгоритмов предобработки. Было установлено, что перед выполнением алгоритма вычисления осредняющей кривой необходимо провести дополнительный перерасчет кривых с использованием линейной интерполяции.

Был изучен программный комплекс Techlog, продукт компании Schlumberger, и основной API для создания модулей для Techlog'a, которые дополняют основной функционал.

Разработан модуль для платформы Techlog, реализующий предобработку данных капиллярного давления и относительной фазовой проницаемости. Реализованный модуль состоит из пяти подмодулей, каждый из которых реализует один алгоритм предобработки. Для каждого подмодуля определены входные и выходные данные, реализованы алгоритмы расчета выходных данных.

СПИСОК ЛИТЕРАТУРЫ

1. Juhasz, I. The central role of Qv and formation-water salinity in the evaluation of shaly formations. -1979.
2. А.Г. Борисов. Методы обобщения кривых капиллярного давления и их совершенствования. Каротажник. -2011. -№7. -С 43-52.
3. А.Г. Борисов. Метод обобщения кривых капиллярного давления с построением капиллярных петрофизических моделей. Геология, география и глобальная энергия. -2010. -№3. -С 100-103.
4. А.Г. Курочкин. Модифицированный алгоритм сглаживание точек маршрута. -2016.
5. Е.А. Гладков. Геологическое и гидродинамическое моделирование месторождений. -2012.
6. Н.С. Бахтий, М.В.Абдулина. Гидродинамическое моделирование с использованием программного обеспечения «Техсхема». Учебное пособие. -2016.
7. Ю.М. Волченко. Интерполяция функций. -2013.
8. Schlumberger, Ocean Framework for Techlog 2017. [Электронный ресурс]. – Режим доступа: https://www.ocean.slb.com/en/developer/techlog/getting-started/getting-started-guide, свободный. (Дата обращения: 19.10.2017 г.).
9. Schlumberger, Techlog. [Электронный ресурс]. – Режим доступа: http://sis-slb2.artusmaster.ru/upload/iblock/715/slb_booklet_techlog.pdf, свободный. (Дата обращения: 20.10.2017 г.).
10. Разработка приложений с использованием Qt.[Электронный ресурс]. / А. Панин. – Режим доступа: http://rus-linux.net/MyLDP/algol/Qt/developing-applications-qt-part-1.html, свободный. (Дата обращения: 07.02.2018 г.)
11. Краткий обзор кроссплатформенного фреймворка Qt. [Электронный ресурс] / Н. Сергейчук. – Режим доступа: https://nicknixer.ru/programmirovanie/kratkij-obzor-krossplatformennogo-frejmvorka-qt/, свободный. (Дата обращения: 10.02.2018 г.).

12. Е.А. Меркурьев, М.А. Токарев. Методические приемы адаптации параметров при прогнозе показателей разработки нефтяных месторождений с помощью постоянно действующей модели. Нефтегазовое дело. -2006. -№1. -С 20-30.
13. Методы интерполяции и аппроксимации. [Электронный ресурс]. – Режим доступа: http://portal.tpu.ru/SHARED/m/MBB/uchebnaya_rabota/%D0%9C%D0%BE%D0%B4%D0%B5%D0%BB%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5%20%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC/Tab/Interp_app.pdf, свободный. (Дата обращения: 09.03.2018 г.).
14. Метод наименьший квадратов. [Электронный ресурс]. – Режим доступа: http://help.prognoz.com/ru/mergedProjects/Lib/01_regression_models/uimodelling_linearregr_lsm.htm, свободный. (Дата обращения: 09.03.2018 г.).
15. Программирование с Qt. [Электронный ресурс] / А. Бешенов – Режим доступа - https://www.ibm.com/developerworks/ru/library/l-qt_1/index.html#N10347, свободный (Дата обращения: 15.02.2018 г.).