



**Direct-Inverse-Solver
(DiInSo)
32-bit version**

Quick (and easy) Guide

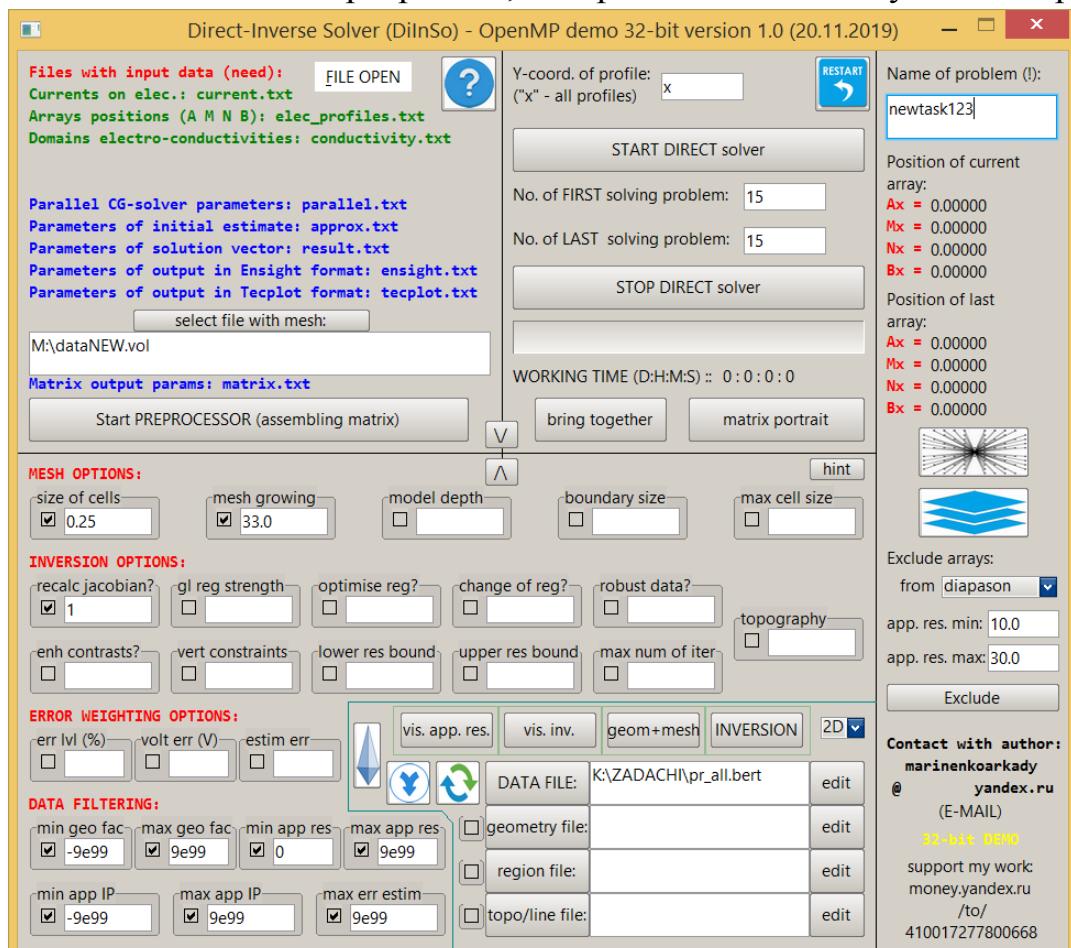
v. 1.0 (05.02.2020) — first public beta version

Вас приветствует программный комплекс DiInSo(x32), предназначенный для академических исследований, цель которого решение прямых задач и задач инверсии электротомографии на постоянном токе. Прочитав данный короткий мануал, вы освоите основные возможности программы и сможете решать прямые задачи и задачи инверсии малой и средней сложности. Для решения более сложных задач вам потребуется немного больше времени на изучение и понимание теории, а также 64-битная версия DiInSo. Текст рассчитан на его последовательное прочтение, поэтому нежелательно пропускать какие-то главы, это может затруднить понимание написанного. Итак, начнём...

О программном комплексе.

Программный комплекс DiInSo(x32) написан на C/C++, откомпилирован в 32-разрядной среде, использует большое количество OpenSource проектов, о которых будет упомянуто ниже, и не имеет никаких ограничений, как и гарантий, на распространение в сети Интернет и должен быть использован исключительно в академических целях.

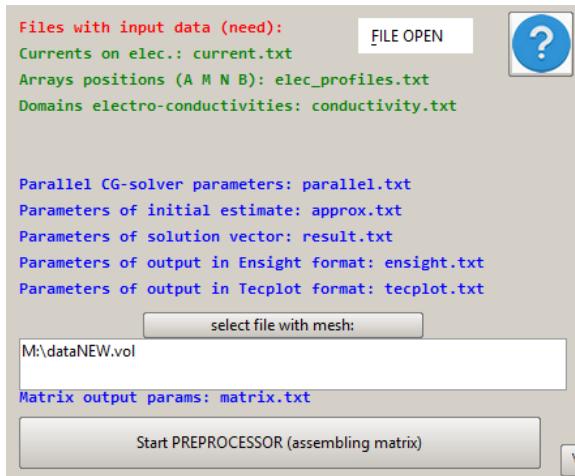
После запуска программы с помощью исполняемого файла Solver.exe, вы увидите главное окно программы, которое выглядит следующим образом:



Главное окно программы разделено на несколько частей, каждая из которых выполняет определённую функцию. Рассмотрим последовательно каждую из них.

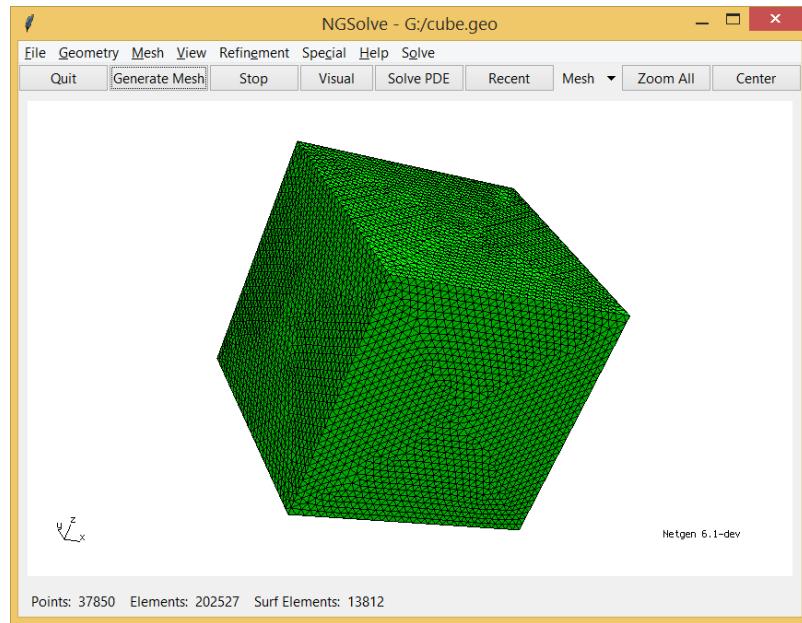
Препроцессор и входные файлы.

Левая верхняя часть главного окна программы содержит информацию о входных файлах, возможность редактирования их, выбор файла с сеткой, кнопки сборки матрицы и кнопку справки:

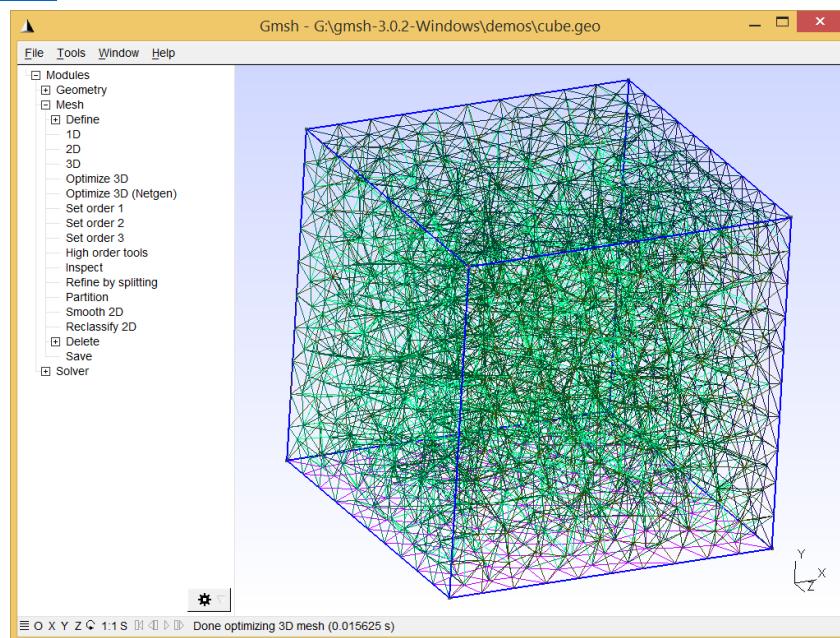


Кнопка «select file with mesh» позволяет выбрать файл с конечно-элементной сеткой, используя которую будет решаться прямая задача на постоянном токе. Решение прямой задачи проводится исключительно в трёхмерной области, заданной с помощью тетраэдральной сетки. Для генерации сетки необходимо использовать стороннее программное обеспечение, часть из которого является бесплатным (то есть вам необходимо будет изучить хотя бы один генератор тетраэдральных сеток из списка, представленного ниже). Разработанный программный комплекс поддерживает форматы следующих генераторов сеток:

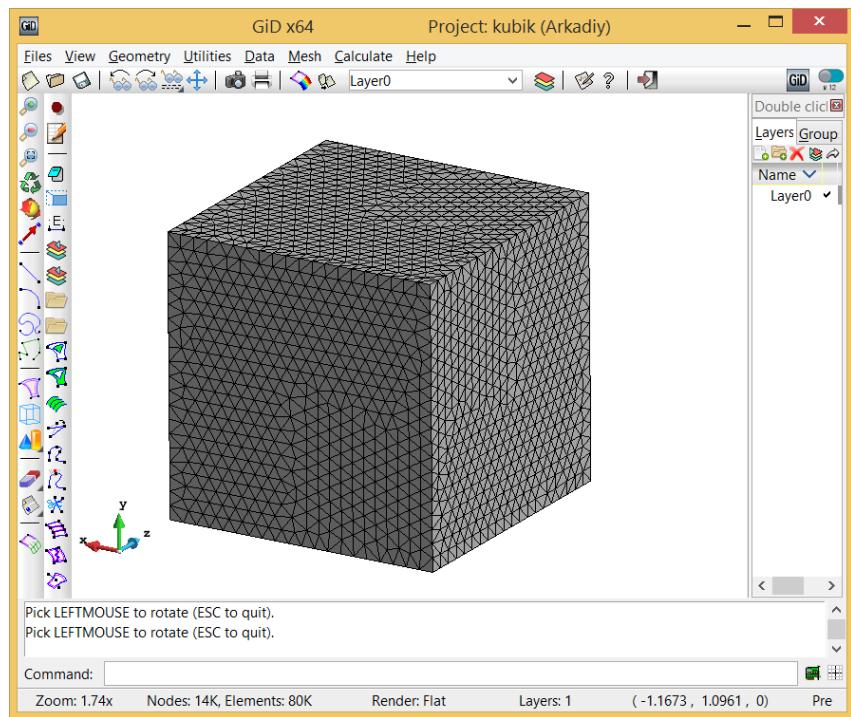
Netgen Mesh Generator (MS Windows и Linux) — бесплатная программа с открытым исходным кодом, распространяемая по лицензии LGPLv2. Сайт программы: <https://ngsolve.org/>



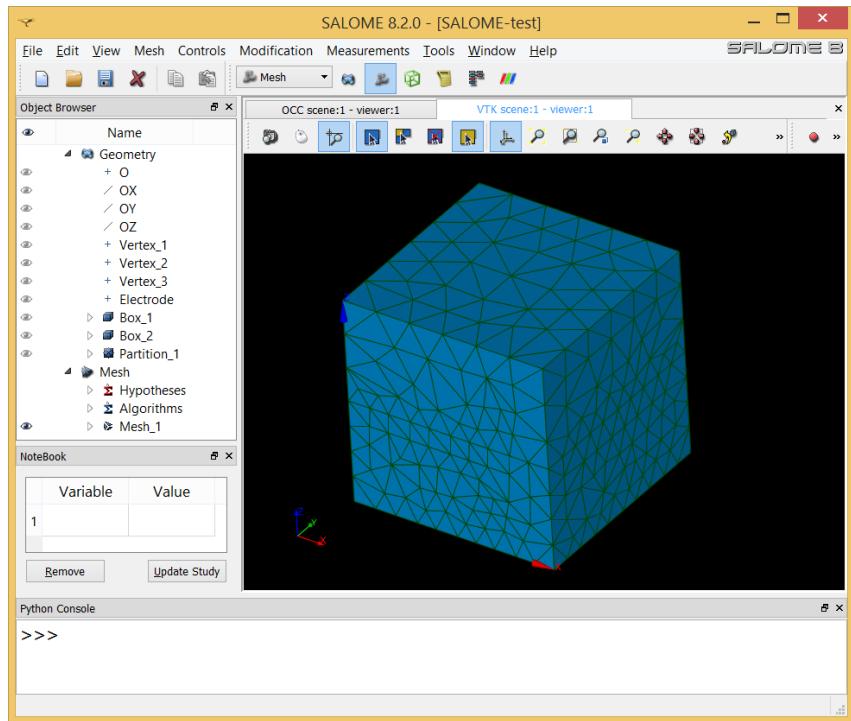
Gmsh (MS Windows и Linux) — бесплатная программа с открытым исходном кодом, распространяемая по лицензии GPL. Сайт программы: <http://gmsh.info/>



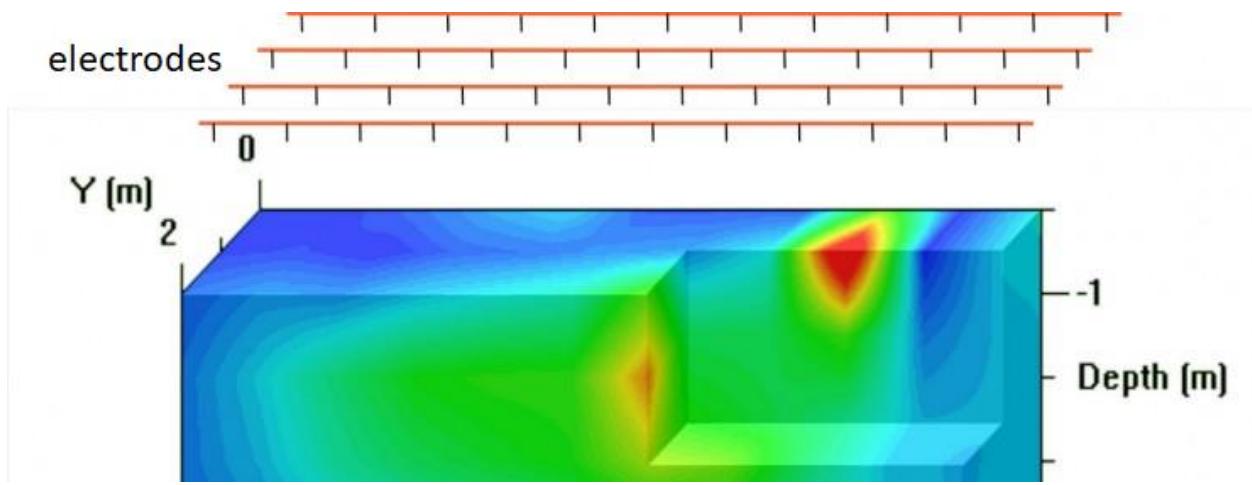
GiD (MS Windows и Linux) — коммерческая программа с закрытым исходном кодом. Сайт программы: <https://www.gidhome.com/>



SALOME (MS Windows и Linux) — бесплатная программа с открытым исходном кодом, распространяемая по лицензии LGPL. Сайт программы: <http://www.salome-platform.org/>



Разберёмся с тонкостями генерации сетки. Одной из главных сложностей генерации сетки для задач электротомографии является необходимость задания большого количества электродов, расположенных на поверхности земли и иногда с небольшим их заглублением:



Дело в том, что далеко не все алгоритмы генерации сетки поддерживают работу с таким примитивом геометрии, как «точка». Именно поэтому, в ходе задания электродов часто приходится идти на ухищрения, как, например, добавление в геометрию множества квадратов, вершины которых совпадают с точками расположения электродов. Итак, ВАЖНО ЗАПОМНИТЬ, точки расположения электродов должны совпадать с узлами построенной сетки!

Следующий момент связан с границами области моделирования. Возможны два случая: если вы планируете моделировать задачу выше границы «воздух-земля» (например, в том случае, когда выше границы «воздух-земля» находятся некоторые помехи в виде объектов) и, если вы НЕ планируете этого делать. В первом случае все границы области должны быть отнесены на такое расстояние от источников тока, где получаемые значения электрического поля будут пренебрежительно малы, а электроды должны быть немного заглублены в землю, чтобы избежать неточностей моделирования из-за влияния сильно контрастного слоя воздуха. Во втором случае делаете то же самое со всеми границами, кроме границы «воздух-земля» и электроды в данном случае заглублять необязательно (нет воздуха — нет влияния сильно контрастной среды). На всех удалённых на большое расстояние от источников границах задаются так называемые «нулевые краевые условия Дирихле» (запомните это название), что для генерирования сетки означает одно — вы должны пометить эти границы каким-то одним числом (по умолчанию «76» — порядок гептамино  из занимательной математики). Итак, ВАЖНО ЗАПОМНИТЬ, если мы планируем моделировать воздушную среду, то все границы удаляем туда, где электрическое поле очень мало, а электроды заглубляем в землю, если же мы не планируем моделировать воздушную среду, то все границы, кроме границы «воздух-земля», удаляем так же далеко, а электроды не заглубляем в

землю (при этом не забываем пометить удалённые границы одинаковым числом — по умолчанию «76»).

Ну и, конечно, не забывайте, что размеры сетки так же имеют большое значение. Обычно правило здесь такое — мелкие тетраэдры должны быть там, где поле претерпевает наибольшие изменения (например, вблизи источников), крупные же тетраэдры могут быть там, где поле не претерпевает больших изменений (например, вдали от источников).

Теперь остановимся на каждом из генераторов сетки отдельно, чтобы облегчить вашу работу с ними:

Netgen Mesh Generator (MS Windows и Linux) — вы можете задать узлы (те самые, где находятся электроды) с помощью следующей строки:

```
point (X, Y, Z);
```

где X, Y и Z — координаты узла.

Gmsh (MS Windows и Linux) — вы можете задать узлы (те самые, где находятся электроды) с помощью следующей строки:

```
Point (N) = {X, Y, Z, S};
```

где N — номер узла, X, Y и Z — координаты узла, необязательный параметр S — максимальный размер рёбер элементов, примыкающих к данному узлу.

GiD (MS Windows и Linux) — для правильной работы программы необходимо определить «тип задачи». Делается это с помощью двух файлов, в одном из которых задаются материалы (среды), а в другом граничные условия (те самые границы, которые мы отнесли подальше от источников).

Пример файла с граничными условиями с номером 76:

```
NUMBER: 76 CONDITION: Big_Bak
CONDTYPE: over surfaces
CONDMESHTYPE: over face elements
QUESTION: Zero
VALUE: 76
END CONDITION
```

Пример файла с материалами с номерами 1, 2 и 3:

```
NUMBER: 1 MATERIAL: Air
QUESTION: Number
VALUE: 1
END MATERIAL
NUMBER: 2 MATERIAL: Earth
QUESTION: Number
VALUE: 2
END MATERIAL
NUMBER: 3 MATERIAL: Ferrum
QUESTION: Number
VALUE: 3
END MATERIAL
```

Поскольку **GiD** не имеет как такого «своего» формата сеточных данных, для вывода сетки в файл должен использоваться специально сгенерированный «файл формата», который для представленного программного комплекса имеется в папке с программой DiInSo и называется

GID.bas. В программе **GiD** данный файл должен быть помещён в директорию «templates».

SALOME (MS Windows и Linux) — генератор сеток **SALOME** не умеет сохранять сетку с привязкой к материалам (средам) и граничным условиям, однако умеет сохранять отдельные части сетки в отдельные файлы. Подобная процедура, например, рассмотрена в следующем мануале: http://www.elmerfem.org/elmerwiki/index.php?title=Multiple_bodies_from_Salome_to_Elmer

Именно поэтому результатом работы генератора **SALOME** должны стать несколько файлов — для каждой отдельной среды и для граничных условий. Все эти файлы, в свою очередь, должны быть перечислены в файле, имя которого **ОБЯЗАТЕЛЬНО ДОЛЖНО** содержать слово «**SALOME**» заглавными или «**salome**» строчными буквами. Этот файл будет подаваться на вход как файл с сеткой, его пример указан ниже:

```
SALOME-CUBE.dat
2
SALOME-AIR.dat 0
SALOME-EARTH.dat 1
1
SALOME-BOUNDARY.dat 76

// ПЕРВАЯ СТРОКА - файл общей сетки
// ВТОРАЯ СТРОКА - число сред и, соответственно, количество файлов со средами
// СЛЕДУЮЩИЕ СТРОКИ -
<файл_среды> <номер> электропроводности (нумерация в соответствии с файлом
conductivity.txt)>
<файл_среды> <номер> электропроводности (нумерация в соответствии с файлом
conductivity.txt)>
...
<файл_среды> <номер> электропроводности (нумерация в соответствии с файлом
conductivity.txt)>
// СЛЕДУЮЩАЯ СТРОКА - число файлов с краевыми условиями (в нашем случае всегда 1)
// СЛЕДУЮЩИЕ СТРОКИ -
<файл_с_краевым_условием> <номер_краевого_условия_ (76_означает_нулевое_краевое_условие)>
<файл_с_краевым_условием> <номер_краевого_условия_ (76_означает_нулевое_краевое_условие)>
...
<файл_с_краевым_условием> <номер_краевого_условия_ (76_означает_нулевое_краевое_условие)>
```

Также **ВАЖНО ОТМЕТИТЬ**, что перечисленные файлы **ДОЛЖНЫ** находиться в той же директории, что и файл, в котором они перечислены.

Итак, мы воспользовались одним из генераторов, успешно получили сетку с нужными средами, с заданными нулевыми граничными условиями Дирихле и с электродами, которые совпадают с узлами сетки. Самое время обратиться к остальным входным файлам. Перечисленные в меню «FILE OPEN» 10 файлов: current.txt, elec_profiles.txt, conductivity.txt, parallel.txt, approx.txt, result.txt, ensight.txt tecplot.txt matrix.txt и null.txt — все, кроме последнего, должны находиться в одной папке с программой DiInSo и могут быть отредактированы при желании и необходимости. Пойдём по порядку:

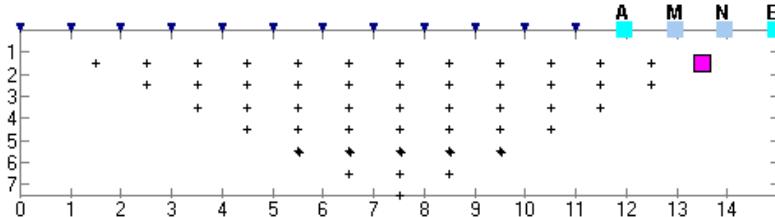
CURRENT.TXT — файл, содержащий величины токов в Амперах на точечных электродах установки, например:

```

2
-1.0
1.0
// ФОРМАТ ФАЙЛА: <количество токов> <токи>
// Р.С.: Если ток задан один,
// то его значение будет задано на всех электродах одинаковое.

```

ELEC PROFILES.TXT — файл, содержащий координаты множества положений установок по всем профилям:



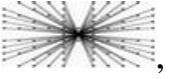
ВАЖНО: под А и В электродами ВСЕГДА подразумеваются токовые электроды, а под М и Н — ВСЕГДА приёмные.

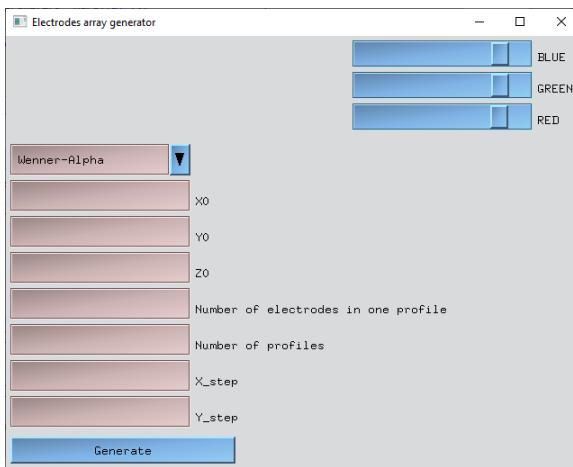
Формат файла следующий:

```

<число установок>
<A (X) > <A (Y) > <A (Z) > <M (X) > <M (Y) > <M (Z) > <N (X) > <N (Y) > <N (Z) > <B (X) > <B (Y) > <B (Z) >
<A (X) > <A (Y) > <A (Z) > <M (X) > <M (Y) > <M (Z) > <N (X) > <N (Y) > <N (Z) > <B (X) > <B (Y) > <B (Z) >
...
<A (X) > <A (Y) > <A (Z) > <M (X) > <M (Y) > <M (Z) > <N (X) > <N (Y) > <N (Z) > <B (X) > <B (Y) > <B (Z) >

```

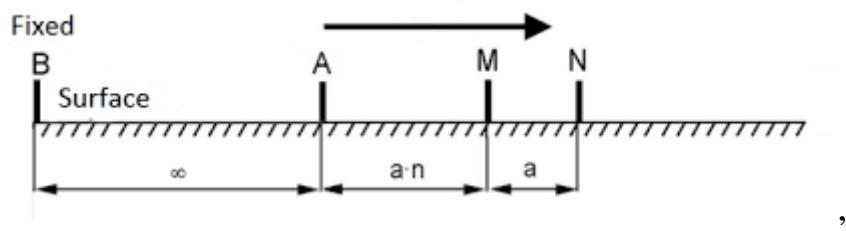
ВАЖНО: данный файл необязательно генерировать вручную, для автоматической генерации такого файла для нескольких типов установок в правой части главного окна программы DiInSo есть кнопка , которая вызывает окно вида:



Бегунки «BLUE», «GREEN», «RED» в данном и ему подобном окнах служат для смены фона окна и больше ни на что не влияют (попробуйте, поэкспериментируйте). Принцип задания положений установок предельно прост: выбираете тип установки (Wenner-Alpha, Wenner-Beta, Schlumberger, Dipole-Dipole, Pole-Dipole) —> задаёте положение (координаты) первого

электрода (X_0 , Y_0 , Z_0) —> задаёте число электродов в одном профиле (Number of electrodes in one profile) —> задаёте число профилей (Number of profiles) —> задаёте шаг между электродами в одном профиле (X_step) —> задаёте шаг между профилями (Y_step) —> нажимаете кнопку «Generate» и ваш файл elec_profiles.txt готов (он появится в той же папке, что и программа DiInSo).

ВАЖНОЕ ЗАМЕЧАНИЕ: если электрод **B** находится на большом расстоянии от остальных электродов (так называемые, трёх-электродные установки):



то в файле elec_profiles.txt необходимо задать координату X этого электрода как большое число (больше 1.000000e+100) по модулю (например, -1.000000e+300 или 1.000000e+300).

CONDUCTIVITY.TXT — файл, содержащий электропроводности соответствующих материалов (сред) в См/м, например:

```

4
0.0
1.0E-12
0.016666666666
7690000.0
// Электропроводность задаётся в следующем виде:
<число электропроводностей>
<нулевая (для нулевой среды) 0 электропроводность> - НЕ ЗАБЫВАТЬ, что нумерация идёт с нуля
<электропроводность 1 среды>
<электропроводность 2 среды>
...
<электропроводность n среды>
```

ВАЖНОЕ ЗАМЕЧАНИЕ: Если каких-то из номеров сред НЕТ, то можно в этих позициях задать любое значение, но игнорировать эти значения нельзя!

PARALLEL.TXT — файл, содержащий параметры параллельного решателя, который будет использоваться для решения прямых задач. Он содержит четыре числа в следующей последовательности: критерий сходимости по абсолютной невязке, критерий сходимости по относительной невязке, критерий расходимости (дивергенция), максимальное число итераций. Обычно не имеет смысла менять те значения, что выставлены по умолчанию (1.0e-8 1.0e-6 1.0e+8 100000). В качестве алгоритма распараллизации используется OpenMP подход, в качестве решателя метод сопряженных градиентов (CG) с так называемым LU-предобусловливателем.

APPROX.TXT — файл, в котором можно прописать необходимость использования начального приближения для решателя, отличного от нулевого начального приближения, например:

```
1  
result415.sol  
//  
ПЕРВАЯ СТРОКА:  
0 или меньше - не использовать начальное приближение  
(нулевое начальное приближение);  
1 или больше - использовать начальное приближение из файла;  
ВТОРАЯ СТРОКА:  
файл начального приближения.
```

Файл начального приближения в этом случае должен иметь формат:

<номер элемента вектора, начиная с НУЛЯ>	<элемент вектора>
<номер элемента вектора, начиная с НУЛЯ>	<элемент вектора>
...	
<номер элемента вектора, начиная с НУЛЯ>	<элемент вектора>

В файле начального приближения не обязательно должны содержаться все элементы вектора. Не содержащиеся элементы будут считаться нулевыми.

RESULT.TXT — файл, указывающий нужно ли выводить каждый полный вектор-решение в отдельный файл (например, для анализа полученного решения). 1 — нужно, 0 — не нужно.

ENSIGHT.TXT и **TECPLOT.TXT** — данные файлы указывают, нужно ли выводить результаты решения (ПЕРВОГО во множестве решений, потому что вывод всех файлов множества решений занимает слишком много времени и слишком много дискового пространства) в отдельные файлы, совместимые с визуализаторами Ensight (<https://www.ensight.com/>) и Tecplot (<http://www.tecplot.com/>), соответственно (например, для анализа решения). Формат файлов одинаков и имеет вид:

```
0  
filename.mod  
//  
ПЕРВАЯ СТРОКА - Нужно ли выводить данные в формат EnSight/Tecplot? 1 - нужно, 0 - не нужно.  
ВТОРАЯ СТРОКА - Имя файла с сеткой (то есть это сетка, которая подается на вход).
```

MATRIX.TXT — файл, указывающий нужно ли выводить сгенерированную матрицу в специальный CSR-подобный (Compact Sparse Row) формат с одномерными массивами каждый в отдельном файле (как и ранее, 1 — нужно, 0 — не нужно), что может быть необходимо для анализа данных и/или для решения задачи в стороннем решателе:

di.txt — файл с диагональными элементами;

ggl.txt — ненулевые элементы матрицы ниже главной диагонали (в так называемом «нижнем треугольнике»);

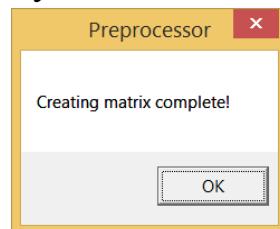
igc.txt — igc[0] = 0, igc[i] = igc[i-1] + (число ненулевых элементов в i-й строке нижнего треугольника), строки нумеруются с НУЛЯ;

jgc.txt — содержит индексы столбцов в нижнем треугольнике каждого ненулевого элемента нижнего треугольника, столбцы нумеруются с НУЛЯ.

ВАЖНО: По умолчанию (и всегда) матрица выводится в два формата — бинарный (файл matr.dat), который невозможно прочитать обычными текстовыми редакторами (он нужен для программы DiInSo), и формат Matrix Market (matr mtx), который может быть прочитан обычными текстовыми редакторами и о котором подробнее можно узнать, например, здесь: <http://networkrepository.com mtx-matrix-market-format.html>

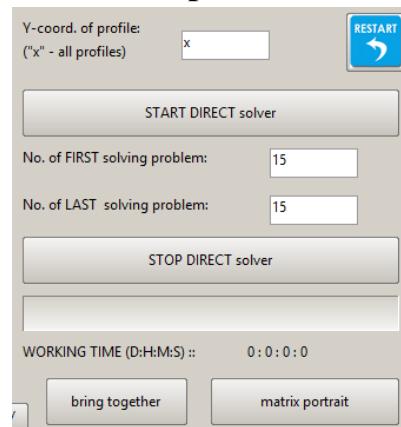
NULL.TXT — необязательный файл, который меняет номер нулевых краевых условий Дирихле (те самые границы, которые мы относим подальше от источников тока). Он содержит единственное число — новый номер нулевых краевых условий Дирихле (которые по умолчанию задаются числом 76).

Осталось лишь две кнопки в данной части главного окна, которые мы не рассмотрели. Кнопка со знаком вопроса открывает файл-справку, которую вы сейчас читаете. И, наконец, главная здесь кнопка — «Start PREPROCESSOR (assembling matrix)». После её нажатия начнётся процесс генерации матрицы для решения прямых задач. Как уже было сказано выше, по умолчанию матрица будет храниться в файлах matr.dat и matr mtx. Как только процесс завершится, вы увидите сообщение вида:



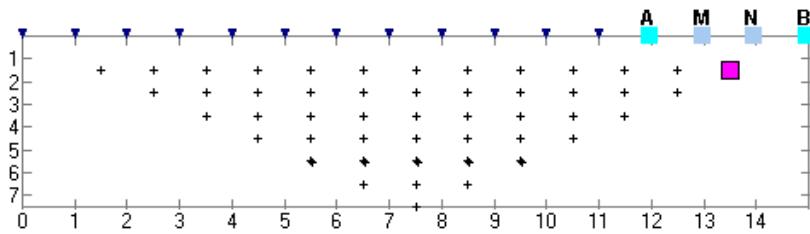
Нажимаем «Ok» и переходим к следующей части главного окна программы, которая отвечает за решение прямых задач.

Решение прямых задач.



Данная часть программы содержит поля выбора Y-координаты профиля, первой и последней решаемой задачи, кнопки старта решателя, останова решателя, компоновки (сборки) полученных результатов для дальнейшего решения задачи инверсии, запуска визуализатора портрета матрицы, перезапуска всей программы и прогресс-бар со счётчиком времени. А теперь подробнее...

В самой верхней части вы можете наблюдать поле для ввода Y-координаты профиля (либо можно ввести букву «x» без кавычек для решения прямых задач на всех профилях). Обратите внимание, что при решении прямых задач различные профили электротомографии «выстроены» в линии с $Y = \text{const}$. Поскольку решение прямой задачи подразумевает перемещения положений установки:



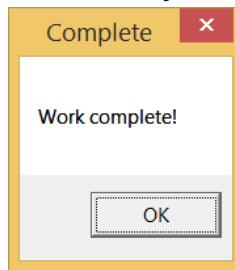
мы вынуждены решать не одну, а множество задач, которые, к слову, могут иметь одинаковые математические решения в случае совпадения положений питающих электродов (в данном случае программа автоматически пропустит цикл решателя, воспользовавшись уже имеющимся решением). Иногда этот процесс требует очень много времени, поэтому мы можем не решать все задачи на всех профилях сразу, а решить прямую задачу на одном из профилей, чтобы сэкономить время и подумать о целесообразности решения задачи на других профилях. Именно такую цель преследует поле ввода координаты профиля.

Ниже есть ещё два поля ввода — первая решаемая задача и последняя решаемая задача. Возникает вопрос, откуда берется понятие «первой» и «последней» задачи, ведь мы можем выбирать произвольный профиль? Вернёмся к файлу `elec_profiles.txt`, в котором перечислены множества положений установки на разных профилях. Если мы будем последовательно просматривать этот файл, то первое положение установки, которое встретилось на данном профиле и будет «первой» задачей на данном профиле. Соответственно, при решении задачи на всех профилях сразу, «первая» задача совпадает с первой по списку в файле `elec_profiles.txt`. Возникает ещё один вопрос, нужно ли знать сколько всего задач на профиле (или на всех профилях), чтобы правильно задать номер последней задачи. Ответ — нет, этого не требуется. В поле «последней» задачи вы можете

задать произвольное большое число, и программа после старта решателя сама исправит его на правильное.

Итак, мы определились на каком профиле будем решать задачу (или на всех профилях сразу), а также с какой по какую задачи. Можно начинать процесс решения. Нажимаем кнопку «START DIRECT solver» и ... процесс пошёл. Счётчик задач в поле «первой» задачи начнёт сдвигаться вперёд, снизу будет заполняться прогресс-бар, а время будет отсчитываться. Во время решения в папке с программой будет создаваться большое количество файлов вида <номер>.sol. Это файлы, содержащие информацию о решённой задачи с определённым <номером>. Внутри этих файлов номера токовых и приёмных электродов в глобальной нумерации узлов сетки и измеренные на приёмных электродах значения электрического потенциала. Не переживайте, позже мы «очистим» от них папку с программой DiInSo.

Как уже говорилось выше, процесс решения может затянуться на многие часы и даже дни (но чаще всё же часы), а время компьютера тоже бывает ценным. Поэтому существует возможность остановить решение на любом его этапе с помощью кнопки «STOP DIRECT solver». В этом случае завершается решение текущей задачи с текущим номером и общий процесс решения прервётся, о чём засвидетельствует информационное сообщение:



Аналогичное сообщение появится и при обычном (не принудительном) завершении решения задач. Возобновить решение с того же места, на котором остановили процесс, можно повторным нажатием кнопки «START DIRECT solver».

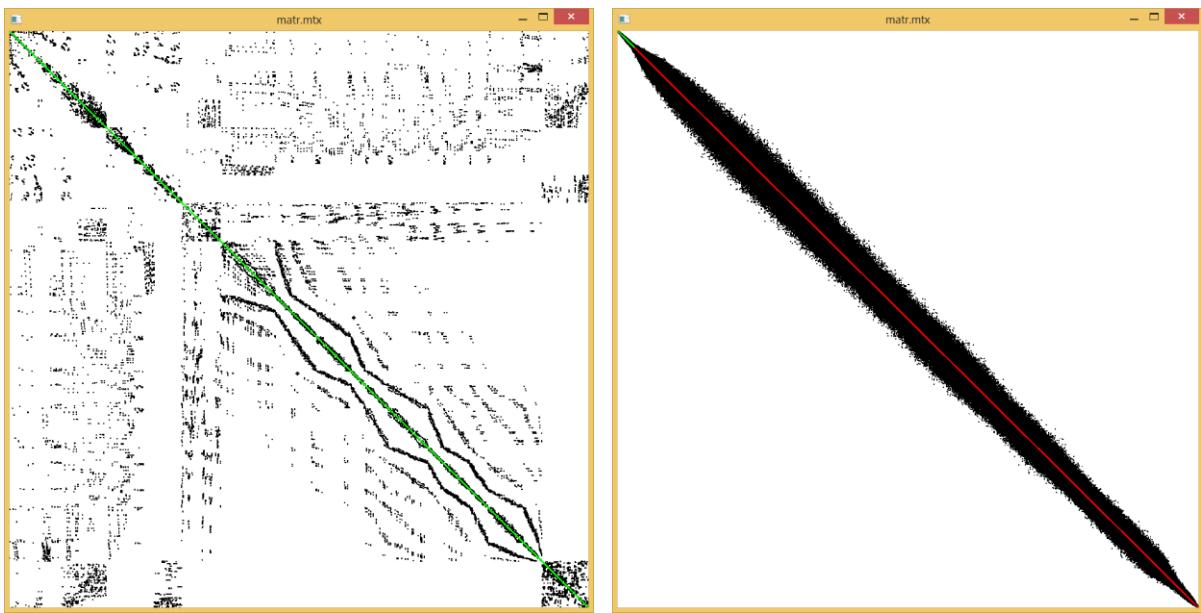
Итак, мы решили все необходимые задачи на всех нужных нам профилях, папка с программой забита файлами вида <номер>.sol, что делать дальше? Далее необходимо все эти файлы «скомпоновать», то есть собрать из них файл для решения задачи инверсии (напомним, что цель решения прямой задачи электротомографии — смоделировать ситуацию измерений на некоторой области, где решается задача инверсии). Делается это с помощью кнопки «bring together». О завершении процесса компоновки будет проинформировано специальным сообщением:



В результате в папке с файлом с сеткой (!) будут созданы файлы для проведения 2D инверсии по отдельным профилям с именами profile<номер>.bert, а также файл для проведения 3D инверсии с именем reverse_data.bert. Также будут созданы файлы для проведения 2D инверсий в сторонних программах: IP4DI (<https://github.com/cageo/Karaoulis-2013>) — файлы вида profile<номер>.d и ZondRes2D (<http://zondgeo.ru/software/resistivity-imaging-ves/zondres2d/>) — файлы вида profile<номер>.z2d, а также 3D инверсий в сторонних программах: IP4DI (<https://github.com/cageo/Karaoulis-2013>) — файл reverse_data.d и RES3DINV (<http://www.geotomosoft.com/>) — файл reverse_data.dat. Однако полная совместимость полученных файлов со сторонними программами НЕ гарантирована.

ВАЖНО ОТМЕТИТЬ, если каких-то результатов расчётов будет не хватать (не посчитаны задачи с определёнными номерами), то полученные файлы для инверсий могут оказаться не полностью заполненными и даже иметь некорректные данные, следите за этим.

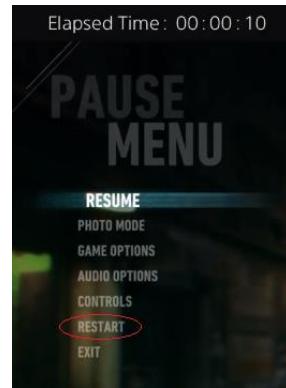
Осталось рассмотреть ещё две кнопки в данной части главного окна программы DiInSo — «matrix portrait» и «RESTART». Первая позволяет графически оценить так называемый «портрет» матрицы, то есть расположение ненулевых элементов внутри матрицы. Почему это важно? Дело в том, что скорость работы итерационных решателей в том числе зависит от «разброса» ненулевых элементов внутри матрицы. Чем этот «разброс» более хаотичный, тем медленнее будет решаться задача. Рассмотрим два портрета:



Диагональные элементы выделяются зелёным и/или красным цветом. В случае красного цвета точек — диагональные элементы близки к нулю, что также плохо сказывается на скорости сходимости итерационных решателей. В представленном примере на левом портрете все диагональные элементы помечены зелёным цветом (то есть они НЕ близки к нулю), однако ненулевые элементы вне диагонали имеют слишком большой разнос. Правый портрет имеет малый разнос ненулевых элементов, однако на диагонали находится очень много элементов, близких к нулю. Устранить большой разнос ненулевых элементов можно сделав перенумерацию узлов конечных элементов (например, выбрав другой генератор сеток, где этот момент учитывается). УстраниТЬ близкие к нулю значения на диагонали, как правило, невозможно, поскольку они появляются из-за сильного контраста сред модели. Итак, зачем вам всё это нужно? Допустим, вы столкнулись с медленной сходимостью решений прямых задач. Посмотрите портрет, может быть вам нужно воспользоваться другим генератором сеток или подумать над упрощением модели задачи.

Наконец, кнопка «RESTART»:  ... Поговорим немного о программировании. Одной из самых главных ошибок программирования на таких прекрасных, но в то же время непростых языках, как C/C++, являются ошибки с так называемой «утечкой памяти». Эти ошибки представляют собой наличие занятой оперативной памяти, которая не была вовремя освобождена от уже ненужных данных, короче говоря — память занята ненужными данными. Из-за подобных ошибок программа начинает занимать всё больше и больше места в памяти компьютера до тех пор, пока не займёт всю доступную (обычно это заканчивается «вылетом» программы). От таких

ошибок не застрахована ни одна программа, написанная на C/C++. Возможно, вы даже видели кнопку или опцию «перезапуска» в других программах (возможно, вы узнаете, откуда этот скриншот):

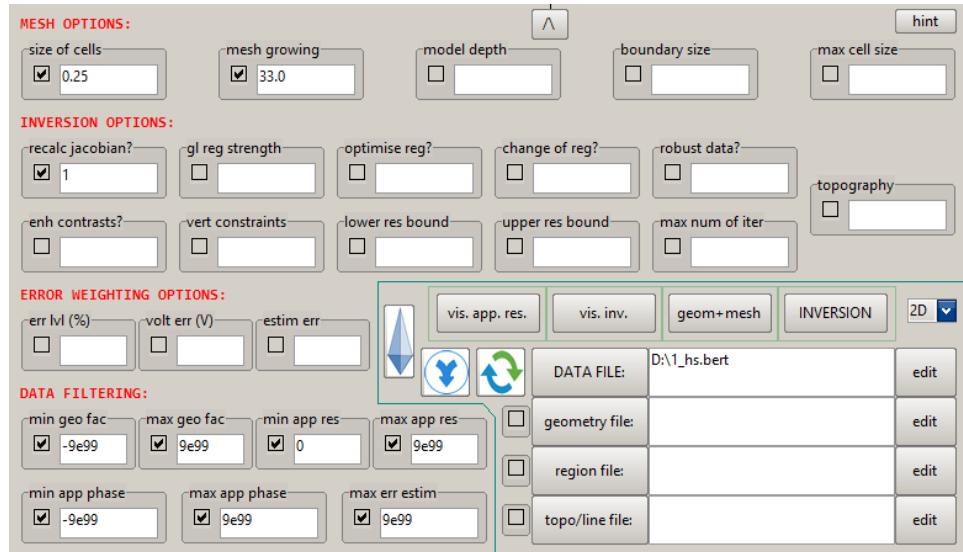


(нет, это не перезапуск миссии — это перезапуск программы)

Цель этой кнопки одна — принудительно очистить память от возможных «утечек». Слишком часто использовать эту кнопку смысла не имеет, однако иногда она может вам помочь (например, после очень долгой работы с программой).

Переходим к следующей части программы, расположившейся в самом низу. Она отвечает за решение задач инверсии.

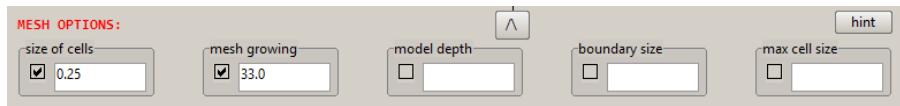
Решение задач инверсии.



Несмотря на устрашающий вид данной части программы большинство функций и настроек здесь выполняются в автоматическом режиме и понадобятся вам лишь в том случае, когда не удаётся получить приемлемый результат в автоматическом режиме. Самой главной кнопкой данной части программы является кнопка выбора пространства — 2D или 3D. Двумерная

инверсия требует меньше памяти, выполняется быстрее, но иногда может быть недостаточной для оценки восстановленной среды. Трёхмерная инверсия сложнее, медленнее и более требовательна к объёму свободной оперативной памяти. Однако выбирать трёхмерную инверсию, когда у вас есть только один профиль смысла не имеет. Да и по нескольким двумерным профилям иногда можно с достаточной степенью точности восстановить трёхмерную картину. Так или иначе, ВАЖНО ЗАПОМНИТЬ, настройки для 2D и для 3D инверсий несколько отличаются, а поэтому начнём с настроек для 2D инверсии, а потом рассмотрим те настройки 3D инверсии, которые имеют отличия, упомянув и те настройки, которые имеют сходства.

Пять опций 2D, связанных с генерируемой сеткой для задачи инверсии (не путайте с сеткой, которая генерируется для прямой задачи):



Первая опция «size of cells» — размер ячеек–ребер (в расстояниях между электродами) на поверхности от больше 0.0 до 0.5. Например, значение 0.25 даст вам примерно 4 ячейки (потому что $1/0.25 = 4$) между соседними электродами. Почему «примерно»? Потому что алгоритм генерации сетки не всегда может подобрать такую сетку, которая удовлетворяет определённым условиям (но он постарается).

Вторая опция «mesh growing» — скорость увеличения размеров ячеек–треугольников с глубиной (33 — быстрая, 35 — медленная). Здесь рекомендуется выбирать опцию опытным путём. Сначала попробовать решать задачу на грубой сетке (то есть когда ячейки быстро увеличиваются с глубиной), а затем увеличивать параметр «mesh growing», увеличивая при этом точность инверсии до необходимой (однако при этом возрастают требования к объёму оперативной памяти).

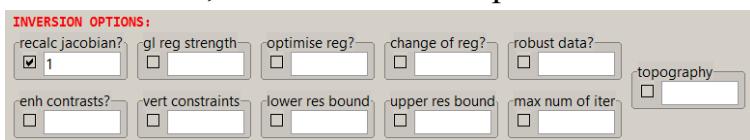
Третья опция «model depth» — глубина исследований в единицах измерения модели (обычно это метры), которую вы можете задать. Она может быть любой, хоть несколько километров. Но помните, что если расположение электродов таково, что томография «добивает» максимум, например, на 10 метров, то на глубине 100 метров вы будете получать некорректный результат, который, кроме того, будет «портить» результат на тех глубинах, где он должен быть корректным. Так что работайте с этой опцией аккуратно и не забывайте, что программа может определить глубинность метода автоматически (если вы не включите данную опцию).

Четвёртая опция «boundary size» — удалённость границ от электродов в процентах. В ходе инверсии мы решаем задачу в той или иной степени похожую на прямую задачу и правила в ней такие же — границы должны быть достаточно удалены от электродов. Пусть цепочка электродов представляет собой линию длиной 100 метров, тогда удалённость границ на 500% подразумевает расстояние от границы до ближайшего электрода 500 метров. Если удалить границу недостаточно, то решение может быть некорректным, как и в случае с прямой задачей.

Пятая опция «max cell size» — максимальный размер ячейки-треугольника в смысле площади ячейки. Если вы не хотите, чтобы ваши ячейки росли слишком быстро с глубиной, вы можете ограничить их максимальный размер и этой опцией. Однако, не забывайте правило — меньший размер ячеек означает большие требования к объёму оперативной памяти.

Наконец, маленькая кнопка «hint» позволяет вам получить краткую справку по всем опциям в данной части окна (это бывает полезно, поскольку их очень много, и они легко забываются).

Одннадцать опций 2D, связанных непосредственно с инверсией:



Опция «recalc jacobian?» — пересчитывать ли Якобиан (определитель матрицы Якоби) на каждом шаге итерации. 1 — да, 0 — нет. Что хорошего в пересчёте Якобиана на каждом шаге? Скорость сходимости растёт в смысле числа итераций. Что плохого в пересчёте Якобиана на каждом шаге? Сам пересчёт Якобиана занимает много времени в случае большой задачи.

Опция «gl reg strength» — значение параметра регуляризации. Когда вы проводите процедуру инверсии, то вы решаете задачу, которая может иметь множество «правильных» с математической точки зрения решений. Поэтому вводятся некоторые дополнительные параметры, которые ограничивают конечное решение в некоторых «рамках». Параметр регуляризации может иметь как очень маленькие значения (например, 1), так и очень большие (например, 1000). В чём разница? Одни значения (обычно малые значения) в результате инверсии дают достаточно точные геометрии сред, но при этом сопротивления этих сред могут быть далеки от реальных. Другие значения (обычно большие значения) сильно размывают геометрии сред, но при этом дают достаточно точные данные по сопротивлениям за счёт меньшего разброса значений. Можно найти и компромисс между этими двумя

случаями. Очень часто под компромиссом понимается число 20 (так уж сложилось), но это не значит, что оно всегда даст вам нужный результат. Экспериментируйте!

Опция «`optimise reg?`» — подбор параметра регуляризации, о котором говорится выше, с помощью метода так называемых L-кривых. 1 — подбирать, 0 — не подбирать. Данный метод постараётся подобрать вам параметр регуляризации, который будет компромиссным в плане точности геометрии и значений сопротивлений. Однако стоит отметить, что данный метод довольно часто ошибается с подбором, поэтому будьте внимательны при его использовании.

Опция «`change of reg?`» — изменять параметр регуляризации, о котором говорится выше, на каждом новом шаге итерации в некоторое число раз. Например, если в опции «`gl reg strength`» установлен параметр регуляризации 100, то опция «`change of reg?`» в виде значения 0.5 уменьшит параметр регуляризации на второй итерации в два раза (до 50), на третьей итерации ещё в два раза (до 25) и так далее. Зачем это нужно? Некоторые сложные модели позволяют добиться приемлемых результатов инверсии только в результате подобных манипуляций.

Опция «`robust data?`» — флаг, указывающий необходимость оценки данных на наличие помех: 1 — данные необходимо оценить на наличие помех в них, 0 — данные не нужно оценивать на наличие помех в них. Что это означает? Допустим, измеренные вами данные для решения задачи инверсии были сильно зашумлены и вы знаете об этом. Включив данную опцию, вы включаете алгоритм оценивания данных, который автоматически отбросит или изменит вес тех значений, которые явно выбиваются из общей тенденции. Если же вы уверены, что измеренные данные имеют высокую степень точности (например, если они получены путём решения прямой задачи), то включать данную опцию НЕ рекомендуется.

Опция «`enh contrasts?`» — флаг, указывающий необходимость «усилить контрастность» между отдельными подобластями (средами): 1 — усилить контрастность, 0 — не усиливать контрастность. Что это значит? Решение задачи инверсии является неразрывным, то есть вы всегда будете наблюдать плавные переходы (в смысле значений сопротивлений) между разными подобластями (средами). Реалии же жизни немного другие — в землю может быть закопан, например, огромный металлический блок, вот вам и резкий переход между подобластями. Включение данной опции иногда позволяет более чётко выделить какие-то подобласти (среды), переход к которым осуществляется скачком значения сопротивления. Однако не нужно думать,

что эта опция является панацеей, так как она не отменяет неразрывность результата решения задачи инверсии.

Опция «vert constraints» — данная опция чем-то похожа на предыдущую и служит для сообщений алгоритму инверсии данных касательно вертикальных границ подобластей. 1 указывает на изотропность (то есть непрерывность) среды, 0.1 указывает на то, что мы имеем вертикальные границы среды, которые несколько размыты, а 0.01 — на то, что вертикальные границы среды должны выделяться достаточно точно, то есть чем дальше значение от 1, тем сильнее должны выделяться вертикальные слои. Главное, не забывайте, что термин «слоистая среда» в геофизике, как правило, подразумевает горизонтально-слоистую среду, а вертикальные слои, как правило, присутствуют там, где в слоистой среде есть какие-то включения (аномалии). При этом не нужно думать, что в строго горизонтально-слоистой среде данная опция должна равняться 1, поскольку «поиск» вертикальных границ (в случае значений меньше 1) означает лишь уменьшение размытости результатов инверсии и более чёткое выделение возможных аномалий. Часто это наоборот «помогает» лучше выделить и чисто горизонтально-слоистую среду, хоть и потеряв в точности значений сопротивлений (так как уменьшается размытость).

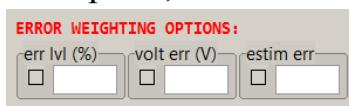
Опции «lower res bound» и «upper res bound» — указывают на минимальное и максимальное значения сопротивления в решаемой задачи инверсии. То есть, если вам точно известен диапазон возможных значений сопротивлений, то вы можете его задать, в противном случае он будет от 0 до очень большого значения. Однако не следует ожидать, что в результате вы получите решение именно в том диапазоне, который будет указан. Может случиться и так, что метод не позволяет «уместить» решение в ваш диапазон, хотя, конечно, решение будет максимально близким к вашему диапазону.

Опция «max num of iter» — вы можете ограничить максимальное число итераций решения задачи инверсии с помощью этой опции, однако делать этого НЕ рекомендуется, лучше дождаться выхода из цикла по критериям сходимости, так решение задачи инверсии будет точнее. Смысл использовать данную опцию есть только в одном случае — вы хотите посмотреть решение на определённой итерации, чтобы понаблюдать как изменяется решение с каждой итерацией.

Опция «topography» — указывает на то, что необходимо учитывать «топографический эффект», а именно — учитывать то, что поверхность земли, на которой проводились измерения, НЕ ровная: 1 — учитывать топографию, 0 — не учитывать топографию. Для каждой установки электротомографии есть такое понятие как «коэффициент установки». Этот

коэффициент высчитывается исходя из расстояний между приёмными и питающими электродами. При неровной поверхности данный коэффициент нежелательно рассчитывать традиционным методом, потому что электроды находятся на разных высотах друг относительно друга. Чем больше неровности земли, тем сильнее может отличаться реальный коэффициент установки от посчитанного. Включив эту опцию, коэффициенты установок и сетка модели будут пересчитаны исходя из особенностей геометрии. С другой стороны, включение данной опции достаточно сильно усложняет алгоритм решения и требование к объёму оперативной памяти. С третьей стороны, не включение данной опции при наличии топографического эффекта может привести к тому, что задача не сможет быть решена вовсе. С четвёртой стороны, если коэффициент установки изначально был посчитан с учётом эффекта топографии и, соответственно, с этим учётом были посчитаны кажущиеся сопротивления, то новый пересчёт этих параметров только испортит результат. Поэтому вы всегда должны понимать какие именно данные подаёте на вход.

Три опции для 2D и 3D инверсий, связанные с ошибками:



Опция «`err lvl (%)`» — здесь вы можете указать примерную среднюю точность в процентах проведенных измерений. Например, вы уверены, что из-за помех во время измерений полученные вами данные могут отклоняться от реальных в среднем на 10%, тогда указывайте в этом поле «10» без кавычек.

Опция «`volt err (V)`» — аналогично предыдущему случаю, только для измеренных электрических потенциалов (на измерительных электродах) в Вольтах. Если вы знаете, что ваши измерения отклоняются от реальных в среднем на 0.1 В, то указывайте в этом поле «0.1» без кавычек.

Опция «`estim err`» — опция-флаг, которая указывает, нужно ли оценивать ошибки измерений алгоритмически: 1 — нужно оценивать ошибки, 0 — не нужно оценивать ошибки. Заметим, что ошибки измерений не оцениваются только в том случае, если информация о них точно известна заранее и содержится во входном файле `*.bert` (в колонке «`err`»).

Семь опций 2D и 3D инверсий, связанные с фильтрацией имеющихся и получаемых данных:



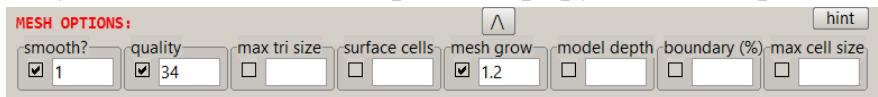
Опции «min geo fac» и «max geo fac» — установки, которые имеют коэффициент, не входящий в данный диапазон, будут исключены из расчётов.

Опции «min app res» и «max app res» — кажущиеся сопротивления, которые не находят в данный диапазон, будут исключены из расчётов.

Опции «min app IP» и «max app IP» — кажущиеся значения вызванной поляризации, которые не находят в данный диапазон, будут исключены из расчётов.

Опция «max err estim» — данные, ошибки которых оценены выше этого значения, будут исключены из расчётов.

Восемь опций 3D, связанных с генерируемой сеткой для задачи инверсии (не путайте с сеткой, которая генерируется для прямой задачи):



Опция «smooth?» — опция-флаг, которая указывает на необходимость сглаживать поверхностную сетку, то есть ту сетку, где заданы питающие и приёмные электроды. 1 — сглаживать, 0 — не сглаживать. Фактически данная опция имеет значение только при наличии топографического эффекта, так как позволяет получить более гладкую поверхность без «угловатостей», которые могут присутствовать если данную опцию не включать.

Опция «quality» — качество поверхностной сетки (не всей сетки, а только поверхностной). 34 — хорошее, 30 — плохое. Обратите внимание, что качество поверхностной сетки во многом влияет на качество остальной сетки, поскольку изначально выстраивается поверхностная (треугольная) сетка, а затем из неё получают объёмную (тетраэдральную) сетку.

Опция «max tri size» — максимальный размер ячейки-треугольника поверхностной сетки в смысле площади ячейки.

Опция «surface cells» — данная опция в какой-то степени дублирует предыдущую опцию, задавая размер ячейки поверхностной сетки, но только в смысле размера ребра. Значения задаются в диапазоне от 0.0 до 0.5. В чём смысл? Во время построения сетки (в том числе и поверхностной) генератор сетки проходит несколько этапов — от грубой сетки, построенной

по исходным точкам, которая непригодна для расчётов, до более-менее адекватной сетки с точки зрения расчётов. Перестроение сетки происходит за счёт разбиения грубой сетки. Данное значение укажет шаг этого разбиения в тех единицах измерения, которые используются (обычно это метры). Максимальное значение в 0.5 выбрано исходя из логики расчётов разбиения, так как 0.5 — это разбиение грубого шага «на два». Во многом эта опция схожа с опцией «size of cells» для 2D разбиения. Но не один в один, так как здесь мы имеем дело с треугольниками на поверхности со всеми вытекающими.

Опция «mesh grow» — скорость увеличения размеров ячеек–тетраэдров с глубиной (2 — быстрая, 1.1 — медленная). Здесь рекомендуется выбирать опцию опытным путём. Сначала попробовать решать задачу на грубой сетке (то есть когда ячейки быстро увеличиваются с глубиной), а затем увеличивать параметр «mesh growing», увеличивая при этом точность инверсии до необходимой (но и увеличивая требования к объёму оперативной памяти). Почему значения для данной опции отличаются от аналогичных для случая 2D инверсии? Это связано с тем, что для построения сеток используются разные генераторы для 2D и 3D случаев со своими особенностями и, соответственно, своими параметрами.

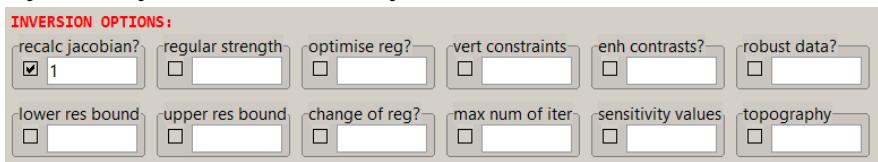
Опция «model depth» — глубина исследований в единицах измерения модели (обычно это метры), которую вы можете задать. Она может быть любой, хоть несколько километров. Но помните, что если расположение электродов такого, что томография «добивает» максимум на 10 метров, то на глубине 100 метров вы будете получать некорректный результат, который, кроме того, будет «портить» результат на тех глубинах, где он должен быть корректным. Так что работайте с этой опцией аккуратно и не забывайте, что программа может определить глубинность метода автоматически (если вы не включите данную опцию).

Опция «boundary (%)» — удаленность границ от электродов в процентах. В ходе инверсии мы решаем задачу в той или иной степени похожую на прямую задачу и правила в ней такие же — границы должны быть достаточно удалены от электродов. Пусть цепочки электродов (профили) образуют квадрат со стороной 100 метров, тогда удалённость границ на 500% подразумевает расстояние от границы до ближайшего электрода 500 метров. Если удалить границу недостаточно, то решение может быть некорректным, как и в случае с прямой задачей.

Опция «max cell size» — максимальный размер ячейки–тетраэдра в смысле объёма ячейки. Если вы не хотите, чтобы ваши ячейки росли слишком быстро с глубиной, вы можете ограничить их максимальный размер

и этой опцией. Однако, не забывайте правило — меньший размер ячеек означает большие требования к объёму оперативной памяти.

Двенадцать опций 3D, связанных непосредственно с инверсией (обратите внимание, что много опций в 2D и 3D случаях схожи, однако здесь они будут продублированы для удобства чтения мануала; кроме того, схожие опции могут иметь разные алгоритмы реализации в 2D и 3D случаях, поэтому не нужно думать, что это суть одно и то же):



Опция «`recalc jacobian?`» (аналогична опции «`recalc jacobian?`» для 2D) — пересчитывать ли Якобиан (определитель матрицы Якоби) на каждом шаге итерации. 1 — да, 0 — нет. Что хорошего в пересчёте Якобиана на каждом шаге? Скорость сходимости растёт в смысле числа итераций. Что плохого в пересчёте Якобиана на каждом шаге? Сам пересчёт Якобиана занимает много времени в случае большой задачи.

Опция «`regular strength`» (аналогична опции «`gl reg strength`» для 2D) — значение параметра регуляризации. Когда вы проводите процедуру инверсии, то вы решаете задачу, которая может иметь множество «правильных» с математической точки зрения решений. Поэтому вводятся некоторые дополнительные параметры, которые ограничивают конечное решение в некоторых «рамках». Параметр регуляризации может иметь как очень маленькие значения (например, 1), так и очень большие (например, 1000). В чём разница? Одни значения (обычно малые значения) в результате инверсии дают достаточно точные геометрии сред, но при этом сопротивления этих сред могут быть далеки от реальных. Другие значения (обычно большие значения) сильно размывают геометрии сред, но при этом дают достаточно точные данные по сопротивлениям за счёт меньшего разброса значений. Можно найти и компромисс между этими двумя проблемами. Очень часто под компромиссом понимается число 20 (так уж сложилось), но это не значит, что оно всегда даст вам нужный результат. Экспериментируйте!

Опция «`optimise reg?`» (аналогична опции «`optimise reg?`» для 2D) — подбор параметра регуляризации, о котором говорится выше, с помощью метода так называемых L-кривых. 1 — подбирать, 0 — не подбирать. Данный метод постарается подобрать вам параметр регуляризации, который будет компромиссным в плане точности геометрии и значений сопротивлений.

Однако стоит отметить, что данный метод довольно часто ошибается с подбором, поэтому будьте внимательны при его использовании.

Опция «vert constraints» (аналогична опции «vert constraints» для 2D) — данная опция чем-то похожа на опцию «enh contrasts?», которая описана сразу следом за этой опцией, и служит для сообщений алгоритму инверсии данных касательно вертикальных границ подобластей. 1 указывает на изотропность (то есть непрерывность) среды, 0.1 указывает на то, что мы имеем вертикальные границы среды, которые несколько размыты, а 0.01 — на то, что вертикальные границы среды должны выделяться достаточно точно, то есть чем дальше значение от 1, тем сильнее должны выделяться вертикальные слои. Главное, не забывайте, что термин «слоистая среда» в геофизике, как правило, подразумевает горизонтально-слоистую среду, а вертикальные слои, как правило, присутствуют там, где в слоистой среде есть какие-то включения (аномалии). При этом не нужно думать, что в строго горизонтально-слоистой среде данная опция должна равняться 1, поскольку «поиск» вертикальных границ (в случае значений меньше 1) означает лишь уменьшение размытости результатов инверсии и более чёткое выделение возможных аномалий. Часто это наоборот «помогает» лучше выделить и чисто горизонтально-слоистую среду, хоть и потеряв в точности значений сопротивлений (так как уменьшается размытость).

Опция «enh contrasts?» (аналогична опции «enh contrasts?» для 2D) — флаг, указывающий необходимость «усилить контрастность» между отдельными подобластями (средами): 1 — усилить контрастность, 0 — не усиливать контрастность. Что это значит? Решение задачи инверсии является неразрывным, то есть вы всегда будете наблюдать плавные переходы между разными подобластями (средами). Реалии же жизни немного другие — в землю может быть закопан, например, огромный металлический блок, вот вам и резкий переход между подобластями. Включение данной опции иногда позволяет более чётко выделить какие-то подобласти (среды), переход к которым осуществляется скачком значения сопротивления. Однако не нужно думать, что эта опция является панацеей, так как она не отменяет неразрывность результата решения задачи инверсии.

Опция «robust data?» (аналогична опции «robust data?» для 2D) — флаг, указывающий необходимость оценки данных на наличие помех: 1 — данные необходимо оценить на наличие помех в них, 0 — данные не нужно оценивать на наличие помех в них. Что это означает? Допустим, измеренные вами данные для решения задачи инверсии были сильно зашумлены и вы знаете об этом. Включив данную опцию, вы включаете алгоритм оценивания данных, который автоматически отбросит или изменит вес тех значений,

которые явно выбиваются из общей тенденции. Если же вы уверены, что измеренные данные имеют высокую степень точности (например, если они получены путём решения прямой задачи), то включать данную опцию НЕ рекомендуется.

Опции «lower res bound» и «upper res bound» (аналогичны опциям «lower res bound» и «upper res bound» для 2D) — указывают на минимальное и максимальное значения сопротивления в решаемой задаче инверсии. То есть, если вам точно известен диапазон возможных значений сопротивлений, то вы можете его задать, в противном случае он будет от 0 до очень большого значения. Однако не следует ожидать, что в результате вы получите решение именно в том диапазоне, который будет указан. Может случиться и так, что метод не позволяет «уместить» решение в ваш диапазон, хотя, конечно, решение будет максимально близким к вашему диапазону.

Опция «change of reg?» (аналогична опции «change of reg?» для 2D) — изменять параметр регуляризации, о котором говорится выше, на каждом новом шаге итерации в некоторое число раз. Например, если в опции «regular strength» установлен параметр регуляризации 100, то опция «change of reg?» в виде значения 0.5 уменьшит параметр регуляризации на второй итерации в два раза (до 50), на третьей итерации ещё в два раза (до 25) и так далее. Зачем это нужно? Некоторые сложные модели позволяют добиться приемлемых результатов инверсии только в результате подобных манипуляций.

Опция «max num of iter» (аналогична опции «max num of iter» для 2D) — вы можете ограничить максимальное число итераций решения задачи инверсии с помощью этой опции, однако делать этого НЕ рекомендуется, лучше дождаться выхода из цикла по критериям сходимости, так решение задачи инверсии будет точнее. Смысл использовать данную опцию есть только в одном случае — вы хотите посмотреть решение на определенной итерации, чтобы понаблюдать как изменяется решение с каждой итерацией.

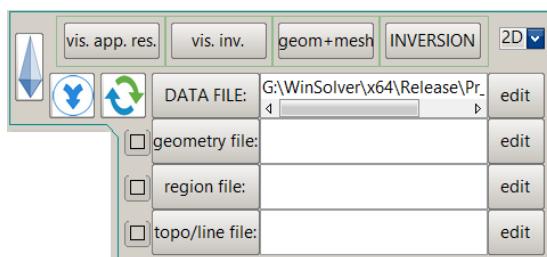
Опция «sensitivity values» — исключить из алгоритма инверсии измерения, чувствительность которых ниже заданного значения. Зачем это нужно? Во многом для экономии памяти. В 3D случае, где расход памяти очень большой, это актуально. Дело в том, что измерения со слабой чувствительностью мало влияют на конечный результат, но участвуют в алгоритме инверсии, занимая лишнюю память. Исключив эти значения часто можно сэкономить память, не сильно потеряв в точности решения.

Опция «topography» (аналогична опции «topography» для 2D) — указывает на то, что необходимо учитывать «топографический эффект», а именно — учитывать то, что поверхность земли, на которой проводились измерения, НЕ ровная: 1 — учитывать топографию, 0 — не учитывать

топографию. Для каждой установки электротомографии есть такое понятие как «коэффициент установки». Этот коэффициент высчитывается исходя из расстояний между приёмными и питающими электродами. При неровной поверхности данный коэффициент нежелательно рассчитывать традиционным методом, потому что электроды находятся на разных высотах друг относительно друга. Чем больше неровности земли, тем сильнее может отличаться реальный коэффициент установки от посчитанного. Включив эту опцию, коэффициенты установок и сетка модели будут пересчитаны исходя из особенностей геометрии. С другой стороны, включение данной опции достаточно сильно усложняет алгоритм решения и требование к объёму оперативной памяти. С третьей стороны, не включение данной опции при наличии топографического эффекта может привести к тому, что задача не сможет быть решена вовсе. С четвёртой стороны, если коэффициент установки изначально был посчитан с учётом эффекта топографии и, соответственно, с этим учётом были посчитаны кажущиеся сопротивления, то новый пересчёт этих параметров только испортит результат. Поэтому вы всегда должны понимать какие именно данные подаёте на вход.

Как уже сказано выше, опции, связанные с ошибками и с фильтрацией имеющихся и получаемых данных, полностью аналогичны для 2D и 3D случаев.

Перейдём к главному элементу этой части окна программы — к кнопкам выбора файлов, инверсии, построения геометрии и сетки, визуализации и манипуляций с входными данными:



Ещё раз обратим внимание на кнопку выбора размерности задачи: «2D/3D». Ещё раз напомним, что эта кнопка не только указывает программе какой алгоритм инверсии использовать, но и заменяет некоторые поля ввода параметров на нужные, а также меняет функционал некоторых кнопок.

Выбрав правильно размерность, переходим к выбору файла входных данных для инверсии — кнопка «DATA FILE». Файл с входными данными должен иметь расширение *.bert и иметь следующий формат:

```

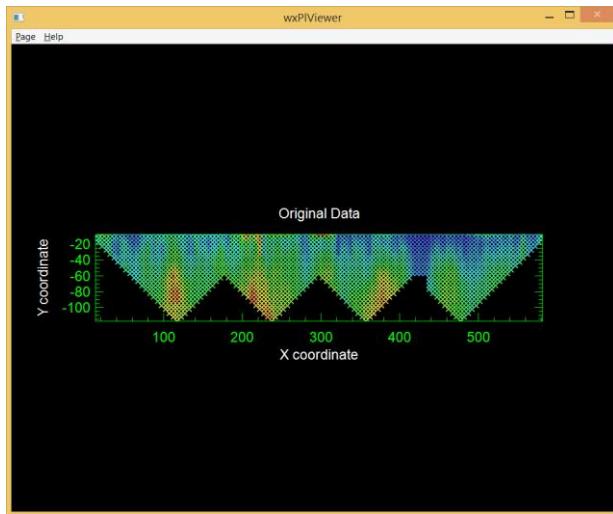
<число электродов, либо электродов + точек, задающих топографию>
<#x z> (без кавычек) - в 2D случае, либо <# x y z> (без кавычек) - в 3D случае
<x-координата электрода (точки)> <y-координата электрода (точки)>, которая есть только в 3D
случае <z-координата электрода (точки)>
<число положений установки>
  
```

«#a b m n rhoa» (без кавычек), либо «#a b m n rhoa ip» (без кавычек), если есть данные вызванной поляризации (ВП), либо «#a b m n rhoa err» (без кавычек), если есть данные по ошибкам измерений, либо «#a b m n rhoa ip err» (без кавычек), если есть данные как вызванной поляризации (ВП), так и ошибок измерений

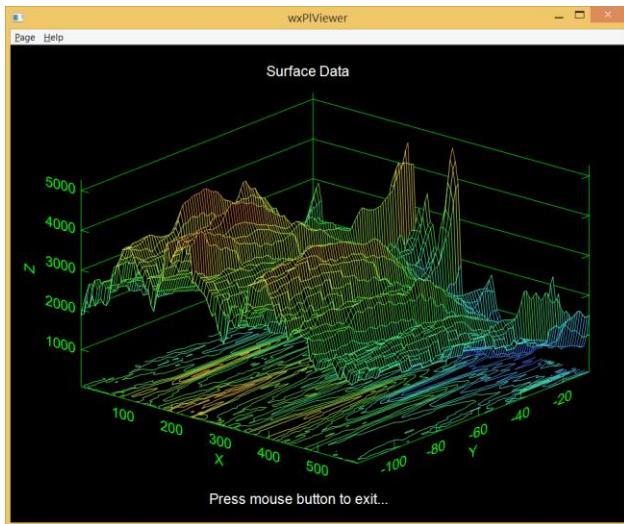
<номер токового электрода A> <номер токового электрода B> <номер приемного электрода M> <номер приёмного электрода N> <значение кажущегося сопротивления> <значение вызванной поляризации, если оно есть> <значение ошибки измерения, если оно есть>

ПОЯСНЕНИЯ: Откуда берутся номера электродов? После поля «#x z» или «# x y z» идут электроды. Считается, что первый электрод в этом списке имеет номер 1, второй — номер 2 и так далее. Какой номер у электродов, которые находятся «на бесконечности», то есть в случае так называемой «трёх-электродной установки»? У таких электродов номер ВСЕГДА нулевой — 0. Заголовки, начинающиеся со знака # — это не просто комментарии, поэтому важно их задавать правильно.

Выбрав входной файл, вы можете посмотреть распределение кажущихся сопротивлений в различной форме, нажав кнопку «vis. app. res.» (на данный момент данная функция доступна только в 2D случае):



Переключаться между страницами с разными формами представления данных можно либо с помощью меню «Page», либо с помощью кнопок «Enter» — вперёд, «Page Up» — назад. Для того, чтобы выйти из окна, необходимо дойти до последней страницы (например, нажимая «Enter»), а затем нажать левую кнопку мыши в любой области последней страницы (последняя страница имеет надпись снизу «Press mouse button to exit...»):

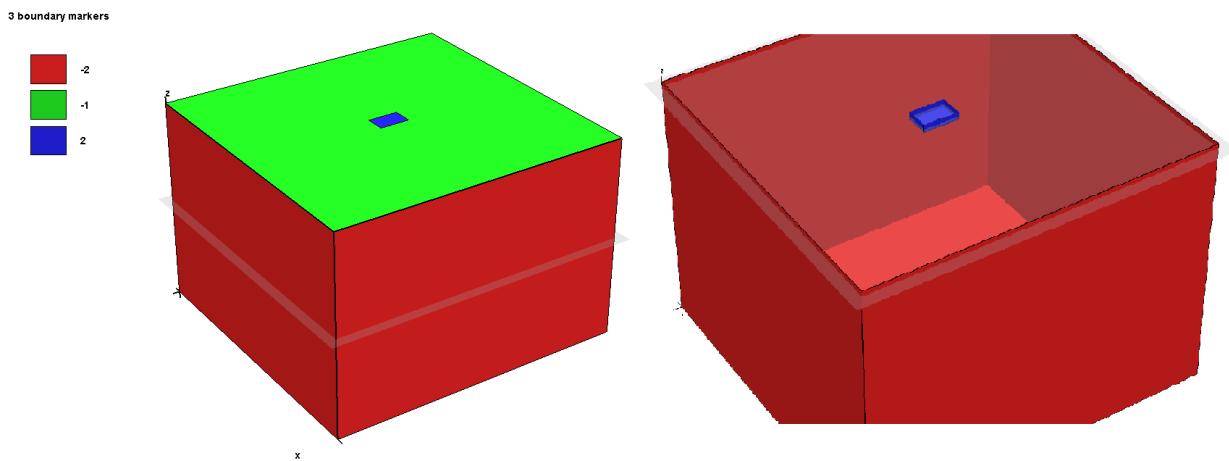


Обратите внимание, что все просмотренные графики будут сохранены в той же папке, что и входной файл *.bert (!), в файле с расширением *.png и с именем, совпадающим с именем входного файла.

Нажав кнопку «geom+mesh» можно получить файлы с геометрией, которые при необходимости будут содержать и элементы поверхностной сетки, по которым будет строиться окончательная сетка для решения задачи инверсии. Нужные нам файлы будут иметь расширение *.poly, имена файлов будут содержать имя исходного файла плюс постфикс «GEOM» (располагаться выходные файлы будут в той же папке, что и входной файл *.bert (!)). Кроме того, на выходе вы найдёте файлы с расширением *.vtk и постфиксом «MESH» в 2D случае или постфиксом «MESHPOLY» в 3D случае. Эти файлы позволяют вам посмотреть полученные геометрии/сетки с помощью любого визуализатора VTK-файлов (<https://vtk.org/>). Но вернёмся к POLY-файлам. Для случаев 2D и 3D они несколько отличаются.

В 2D случае POLY-файлы имеют формат, разработанный для триангулятора Triangle, который описан, например, здесь: <https://www.cs.cmu.edu/~quake/triangle.poly.html> Изучить данный формат крайне рекомендуется. Также необходимо изучить те особенности, которые присутствуют в созданном POLY-файле. !(Далее обратите внимание на разницу понятий «узлы», «границы» и «области»)! Например, узлы, которые содержат электроды, имеют маркер «-99» (мы можем запомнить это как правило — «самое маленькое отрицательное число» ДЛЯ УЗЛОВ), а остальные узлы — маркер «0». Удалённые от электродов границы имеют маркер «-2» (мы можем запомнить это как правило — «самое маленькое отрицательное число» ДЛЯ ГРАНИЦ). Внутренние (не верхние) границы области, где нас интересует решение, имеют маркер «1». Границы внутри этой области решения могут иметь любой маркер, начиная с «-1» и больше.

Все остальные границы (то есть в данном случае верхние границы) имеют маркер « -1 ». Это обычно граница, где расположены электроды. Область, где мы ищем решение, имеет маркер « 2 ». Область, где мы не ищем решение (область, связанная с сильно удалёнными от электродов границами) имеет маркер « 1 » (мы можем запомнить это как правило — «самое маленькое положительное число» ДЛЯ ОБЛАСТЕЙ).



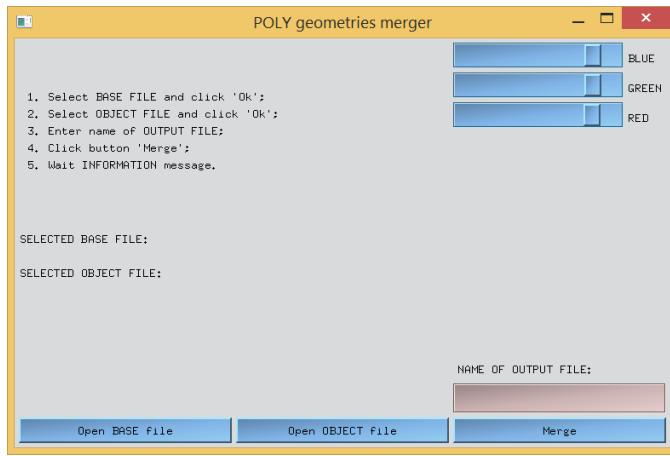
В 3D случае POLY-файлы имеют формат, разработанный для генератора тетраэдральных сеток Tetgen, который описан, например, здесь: <http://wias-berlin.de/software/tetgen/fformats.poly.html> Изучить данный формат крайне рекомендуется. Также необходимо изучить те особенности, которые присутствуют в созданном POLY-файле. !(Далее обратите внимание на разницу понятий «узлы», «границы» и «области»)! Например, граничные сильно удалённые от электродов узлы имеют маркер « -2 ». Граничные узлы, которые задают область, где нас интересует решение, имеют маркер « 2 ». Узлы, которые содержат электроды, имеют маркер « -99 » (мы можем запомнить это как правило — «самое маленькое отрицательное число» ДЛЯ УЗЛОВ). Остальные узлы имеют маркер « 0 » (это вспомогательные узлы, которые лежат под узлами-электродами на некоторой глубине). Верхняя граница области БЕЗ той границы, где находится интересующее нас решение (то есть БЕЗ той области, где находятся электроды) имеет маркер « -1 ». Внешняя сильно удалённая от электродов граница, БЕЗ верхней границы, имеет маркер « -2 » (мы можем запомнить это как правило — «самое маленькое отрицательное число» ДЛЯ ГРАНИЦ). Граница ВСЕЙ области, где нас интересует решение, имеет маркер « 2 ». Внутренние объекты тоже должны иметь границу с маркером « 2 » или больше. Область, где мы ищем решение, имеет маркер « 2 ». Область, где мы НЕ ищем решение (область, связанная с удалёнными от электродов границами) имеет маркер « 1 » (мы

можем запомнить это как правило — «самое маленькое положительное число» ДЛЯ ОБЛАСТЕЙ).

Может показаться, что правила задания границ и областей в 2D и 3D случае несколько отличаются, но это вынужденная мера, связанная с разным множеством примитивов. То, что в 2D случае является объектом, в 3D случае является границей. То, что в 3D случае является объектом, в 2D случае вообще не существует. На самом деле, самое важное ЗАПОМНИТЬ правила для внутренних объектов, и оно очень простое — внутренние точки всегда имеют маркер «0», внутренние линии в 2D случае начинаются со значения «-1» и больше, а внутренние полигоны в 3D случае — «2» и больше, внутреннему же объекту просто дайте маркер, следующий за последним имеющимся.

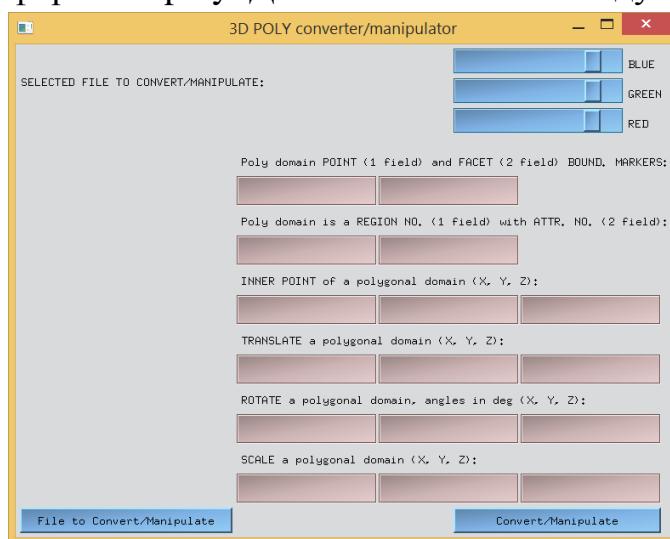
Зачем нужны файлы с геометрией? Исключительно для одной цели — чтобы проводить манипуляции с ними. На вход программы можно подавать вновь созданный файл с геометрией и получать решение на созданной геометрии, а не на сгенерированной автоматически. Выбирается файл с геометрией с помощью кнопки «geometry file:». Если геометрия выбрана, то именно она будет использована для решения задачи инверсии.

Что ещё можно делать с геометрией? Объединять ДВА файла с геометриями с помощью кнопки . Зачем это нужно? Вы можете добавлять в геометрию объекты, физические характеристики которых известны заранее. Кроме того, вы можете добавлять границы объектов, наличие которых известно заранее и без знаний о физических характеристиках объектов. При этом важно отметить, что крайне нежелательно, чтобы добавляемый объект пересекал внутренние и внешние границы той геометрии, куда он добавляется, так как в этом случае итоговая модель может оказаться некорректной. Если вам всё-таки нужно пересечение границ, то рекомендуется вручную проверить корректность полученных результатов и, в случае необходимости, внести исправления (в частности, для этого необходимо понимание форматов *.poly). Окно объединения двух файлов имеет интуитивно понятный интерфейс:



С помощью кнопки «Open BASE file» выбираете базовую геометрию (к которой будет добавляться объект), с помощью кнопки «Open OBJECT file» выбираете объект, который будете добавлять к базовой геометрии. В поле «NAME OF OUTPUT FILE:» вписываете имя нового файла с геометрией (вместе с расширением файла). Нажимаете кнопку «Merge» и ждёте сообщения о завершении операции объединения. В верхней правой области окна вы можете наблюдать «бегунки» со словами «RED», «GREEN» и «BLUE» — двигая их, вы можете менять цвет фона окна для лучшей читаемости текста на вашем мониторе.

Допустим ситуацию, что вам необходимо добавить в геометрию трёхмерный объект довольно сложной формы, который не так просто задать вручную даже при знании форматов *.poly. В этом случае на помощь приходит кнопка , которая поможет вам конвертировать объект, созданный в 3D-редакторе, в формат *.poly. Данное окно имеет следующий вид:



Для конвертирования поддерживаются следующие форматы, расположенные в алфавитном порядке:

3D, 3DS, 3MF, AC, AC3D, ACC, AMJ, ASE, ASK, B3D, BLEND, BVH, CMS, COB, DAE/Collada, DXF, ENFF, FBX, glTF 1.0 + GLB, glTF 2.0, HMB, IFC-STEP, IRR / IRRMESH, LWO, LWS, LXO, MD2, MD3, MD5, MDC, MDL, MESH / MESH.XML, MOT, MS3D, NDO, NFF, OBJ, OFF, OGEX, PLY, PMX, PRJ, Q3O, Q3S, RAW, SCN, SIB, SMD, STP, STL, TER, UC, VTA, X, X3D, XGL, ZGL.

Информацию по каждому из этих форматов вы легко найдёте в Интернете, поэтому нет смысла останавливаться на каждом из них подробно. Рассмотрим принципы работы в этом окне. Итак, выбираете файл для конвертирования с помощью кнопки «File to Convert/Manipulate». При этом выбранным файлом может быть и файл *.poly, конвертирование которого не требуется. В полях «Poly domain POINT (1 field) and FACET (2 field) BOUND. MARKERS:» можно вписать маркеры ВСЕХ узлов и граней, которые будут использованы в выходном файле *.poly. В полях «Poly domain is a REGION NO. (1 field) with ATTR. NO. (2 field):» можно вписать номер области и номер атрибута области, который будет использоваться для вашей 3D-модели. В полях «INNER POINT of a polygonal domain (X, Y, Z):» можно вписать X, Y и Z – координаты внутренней точки вашей 3D-модели. В полях «TRANSLATE a polygonal domain (X, Y, Z):» вы можете указать X, Y и Z – координаты вектора смещения, на который будет сдвинута ваша 3D-модель. В полях «ROTATE a polygonal domain, angles in deg (X, Y, Z):» вы можете указать углы поворота ПО ЧАСОВОЙ СТРЕЛКЕ в градусах по каждой из осей X, Y и Z, соответственно, на которые будет повернута ваша 3D-модель. В полях «SCALE a polygonal domain (X, Y, Z)» вы можете указать растяжение/сжатие по каждой из осей X, Y и Z, соответственно, которому будет подвержена ваша 3D-модель (например, «2» без кавычек означает растянуть в два раза по этой оси, а «0.5» без кавычек — сжать в два раза по этой оси). После нажатия кнопки «Convert/Manipulate» вы получите файл с расширением *.poly, имеющий то же самое имя, что и входной файл (если изначально был выбран файл *.poly, то манипуляциям подвергнется он). Если вы не поняли, о чём идет речь выше, то вы, скорее всего, плохо освоили формат *.poly. Что будет, если не указывать внутреннюю точку? Тогда будет считаться, что ваша 3D-модель «пустая». Это актуально, например, для задания 3D-поверхности, у которой никогда не бывает внутренних точек. Есть ли поддержка 2D моделей для данного окна? Нет, но они здесь и не нужны. 2D модели не представляют из себя ничего сложного и всегда могут быть заданы вручную, с помощью множества линий. В верхней правой области окна вы можете наблюдать «бегунки» со словами «RED», «GREEN» и «BLUE» — двигая их, вы можете менять цвет фона окна для лучшей читаемости текста на вашем мониторе.

Итак, вы получили новую геометрию с какими-то известными заранее объектами. Теперь вы можете использовать её для решения задачи инверсии на вашей геометрии. Сначала выбираете геометрию с помощью кнопки «geometry file:», затем можете уточнить для алгоритма инверсии характеристики тех объектов, которые вам заранее известны. Делается это с помощью кнопки «region file:», которая выбирает файл с характеристиками «регионов», имеющий расширение *.control. Данный файл имеет вид, который лучше продемонстрировать наглядно:

```
#no start Ctype MC zWeight Trans lBound uBound
0 100 1 1 0.2 log 50 1000
1 30 0 0.2 1 log 10 200
#no single start
2 1 100
#no fix
3 22.5
#no background
-1 1
#inter-region
0 2 0.2
```

Номер «#no» означает номер области файла *.poly, но может быть и «*» без кавычек, что означает все области, или даже «1*0», что означает все области, номера которых начинаются на 1 и заканчиваются на 0. Значение «start» задаёт стартовую электропроводность для заданной области. Значение «Ctype» аналогично опции «enh contrasts?», о которой написано выше. Значение «MC» задаёт значение параметра регуляризации относительно глобального значения регуляризации. Например, 1 означает, что параметр регуляризации для данной области следует брать таким же, как и для остальной задачи, а значение 0.1, что параметр регуляризации для данной области следует брать в 10 раз меньше. Значение «zWeight» аналогично опции «vert constraints», о которой написано выше. Значение «trans» указывает алгоритму инверсии так называемое правило «трансформации модели» и может иметь значения «log» (логарифмическая), «lin» (линейная) или «tan» (тангенциальная) без кавычек. По сути данный параметр влияет прежде всего на процесс сходимости решения и редко когда выбирается отличным от «log», который принят по умолчанию. Значения «lBound» и «uBound» аналогичны опциям «lower res bound» и «upper res bound», о которых написано выше. Значение «single» является флагом, указывающим на то, что данную область необходимо считать областью с постоянным сопротивлением, стартовое значение которого указано в поле «start».

Значение «background» является флагом, который указывает, что данную область необходимо считать областью, где решение нас не интересует, то есть это область, которая служит для удаления внешних границ решения задачи инверсии от электродов (о ней писалось выше). Значение «fix» указывает на фиксированное значение сопротивления для данной области. Однако при использовании такого подхода («fix») нужно понимать, что он подразумевает исключение области из алгоритма инверсии методом «заполнения» данной области фиксированным значением. Другими словами, данная область просто «сообщит» алгоритму инверсии своё значение, но расчёты будут проходить без расчётов внутри этой области (значения строго зафиксированы и не меняются). Обычно вариант «fix» используют в том случае, если хотят уменьшить сложность алгоритма (не будет «лишней» сетки, плюс решение не будет «подгоняться» под значения электрического сопротивления в известных областях). Наконец, значения «inter-region» связывают два указанных региона друг с другом в том же смысле, как это делается с помощью опции «vert constraints», о которой написано выше. Если параметры для каких-то областей в файле *.control не указаны, то эти параметры выбираются исходя из глобальных опций.

Перейдём к самой нижней кнопке, которая называется «topo/line file». Эта кнопка имеет разный смысл для 2D и для 3D инверсий и это важно помнить. В 2D случае необходимо выбирать файл с расширением *.xz, который имеет простой формат:

```
<X-координата точки> <Z-координата точки>
<X-координата точки> <Z-координата точки>
...
<X-координата точки> <Z-координата точки>
```

В данном файле заданы точки, которые будут добавлены в 2D геометрию области и затем последовательно соединены в линию. Зачем это нужно? С помощью подобной манипуляции вы можете задавать границы областей внутри области моделирования, которые вам заранее известны. Конечно, то же самое вы могли бы сделать с помощью изменения файла геометрии, как это описано выше. Но иногда указанный здесь путь оказывается намного проще. Если вы хотите добавить несколько линий, то необходимо разделить в файле эти линии с помощью пустых строк. **ВАЖНО ПОМНИТЬ**, что точки должны находиться строго внутри геометрии и не выходить за её пределы.

В 3D случае необходимо выбирать файл с расширением *.xyz, который также имеет простой формат:

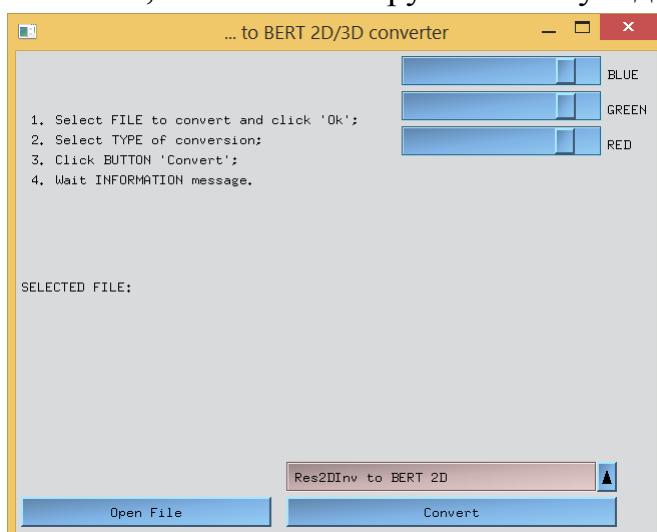
```
<X-координата точки> <Y-координата точки> <Z-координата точки>
<X-координата точки> <Y-координата точки> <Z-координата точки>
...
<X-координата точки> <Y-координата точки> <Z-координата точки>
```

В данном файле заданы точки, которые будут добавлены в 3D геометрию области в качестве точек, изменяющих топографию поверхности, на которой находятся питающие и измерительные электроды. Другими словами, эти точки ВСЕГДА будут находиться на границе «воздух–земля» после перестройки геометрической модели. Конечно, то же самое вы могли бы сделать, задав в этих точках фиктивные «электроды», которые не используются в дальнейшем, но иногда проще сделать так, как описано здесь. Если же вы желаете задать поверхности-границы внутри 3D области, то сделать это можно только путём работы с файлом-геометрией, как это описано выше.

Итак, ПОДЫТОЖИМ: 2D случай — это файлы с расширением *.xz, которые задают границы-линии областей ВНУТРИ геометрии модели; 3D случай — это файлы с расширением *.xuz, которые уточняют топографию ПОВЕРХНОСТИ, задавая точки на границе «воздух–земля».

Справа от выбираемых файлов с помощью кнопок «DATA FILE:», «geometry file:», «region file:» и «topo/line file:» можно наблюдать кнопки «edit». Их предназначение достаточно простое — нажав на такую кнопку, открывается Notepad-подобный редактор текстовых файлов, который позволяет просмотреть и/или отредактировать соответствующий файл.

Ещё одна кнопка с картинкой, функционал которой ещё не был рассмотрен, это кнопка , нажав на которую можно увидеть окно:



Это окно конвертера из форматов, используемых в сторонних программах решения задач инверсии, в формат данной программы, то есть в *.bert файлы. На данный момент поддерживается конвертирование из форматов *.dat типа «general array data» программ Res2DInv/Res3DInv (<https://www.geotomosoft.com>), однако эта поддержка не полная из-за большого разнообразия структур входных файлов данных двух программ,

которые, судя по всему, добавлялись по мере развития программных комплексов, которое насчитывает уже много лет. Поэтому используйте этот конвертер аккуратно — проверяя то, что получается на выходе, и понимая, что некоторые файлы могут НЕ конвертироваться вовсе. С помощью кнопки «Open File» выбирается файл, который будет конвертироваться. Выпадающее меню справа снизу имеет три варианта: «Res2DInv to BERT 2D» — конвертирование между 2D данными, «Res3DInv to BERT 3D» — конвертирование между 3D данными и, наконец, «List of Res2DInv's to BERT 3D» — конвертирование нескольких 2D данных в 3D данные. На последнем случае необходимо остановиться подробнее. Имея несколько *.dat файлов Res2DInv, вы имеете несколько двумерных профилей. Очевидно, что несколько двумерных профилей можно объединить в один трёхмерный. Делать это можно в том числе с помощью внутренних конвертеров Res2DInv / Res3DInv, на которых останавливаться не будем. Сделать это можно и в программе DiInSo. Для этого вручную создаётся специальный файл, который подаётся на вход конвертера, и имеет следующий формат:

```
<имя файла Res2DInv с ПЕРВЫМ профилем>
<сдвиг по X СЛЕДУЮЩЕГО профиля> <сдвиг по Y следующего профиля относительно ПЕРВОГО профиля
(то есть относительно нуля)> <поворот на угол (в градусах) СЛЕДУЮЩЕГО профиля по часовой стрелке>
    <имя файла Res2DInv со СЛЕДУЮЩИМ профилем>
    <сдвиг по X СЛЕДУЮЩЕГО профиля> <сдвиг по Y следующего профиля относительно ПЕРВОГО профиля
(то есть относительно нуля)> <поворот на угол (в градусах) СЛЕДУЮЩЕГО профиля по часовой стрелке>
        <имя файла Res2DInv со СЛЕДУЮЩИМ профилем>
...

```

То есть строчек с именами файлов всегда на одну больше, чем строчек с данными о смещении / повороте.

Итак, базовым является первый профиль, который ВСЕГДА расположен по оси Y в нулевой точке и ВСЕГДА неподвижен. Все остальные профили сдвигаются и поворачиваются на заданные значения. Подытожим:

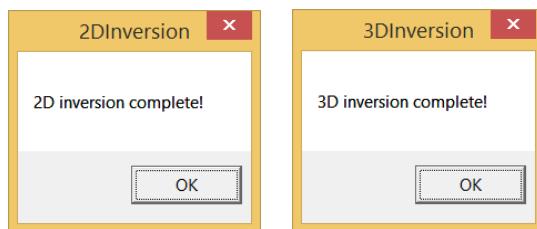
- к каждой координате X всех профилей, кроме базового (первого), будет прибавлено соответствующее значение сдвига;
- все профили, кроме базового (первого), будут смещены по Y относительно нуля (0) на соответствующее значение;
- после смещения все профили, кроме базового (первого), будут повернуты по часовой стрелке на соответствующий указанный градус, где под поворотом понимается поворот при фиксированной (неподвижной) первой точке профиля;
- если вам не нужно смещать профиль по какой-то из координат, то указывайте соответствующее значение нуль (0), если же вам не нужно поворачивать профиль, то также указывайте соответствующее значение нуль (0).

Также ВАЖНО ОТМЕТИТЬ, что перечисленные файлы ДОЛЖНЫ находиться в той же директории, что и файл, в котором они перечислены.

Ну и, наконец, кнопка «Convert» служит для запуска процесса конвертирования файла, который закончится сообщением в области окна. На выходе в папке, где расположен входной файл (!), будет получен файл с тем же именем, что и входной файл, но с расширением *.bert.

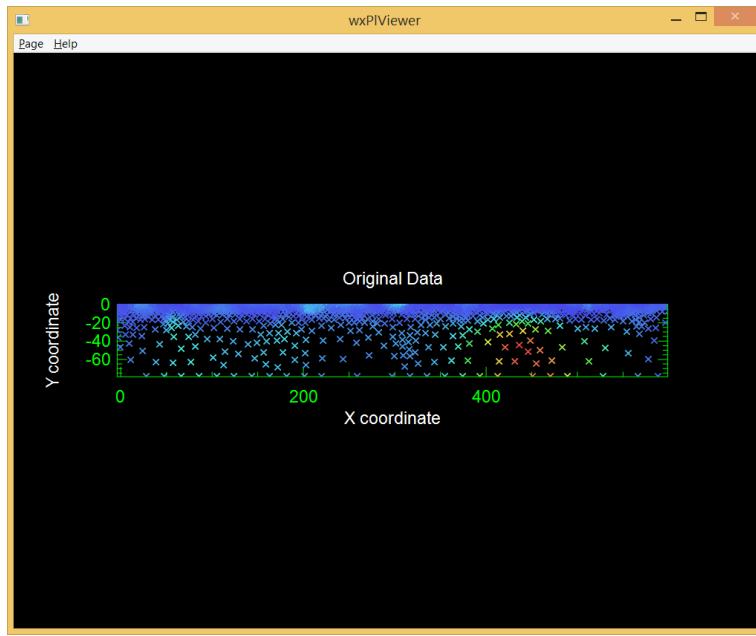
В верхней правой области окна вы можете наблюдать «бегунки» со словами «RED», «GREEN» и «BLUE» — двигая их, вы можете менять цвет фона окна для лучшей читаемости текста на вашем мониторе.

Теперь можно переходить к основной кнопке данной части главного окна программы — кнопке инверсии «INVERSION». После того, как был выбран входной файл, при необходимости (напомним, что это совсем необязательно) выбраны файлы с геометрией, свойств областей, топографии или границы между областями, настроены все нужные пользователю опции (что также необязательно, опции могут быть выставлены автоматически, однако это не гарантирует точного решения), а, ГЛАВНОЕ, правильно указана задача — 2D или 3D... МОЖНО нажимать кнопку «INVERSION» и активировать процесс инверсии, который завершится информационным сообщением вида:



Проведя инверсию, можно посмотреть результаты (значения сопротивлений) в графическом представлении, нажав на кнопку «vis. inv.», при этом визуализаторы для 2D и для 3D случаев будут отличаться.

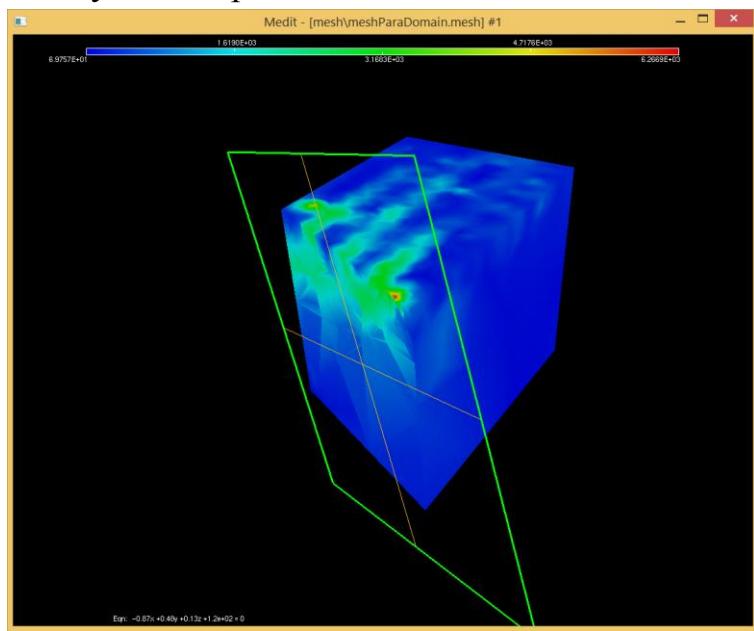
В 2D случае визуализатор аналогичен тому, что вы видели при нажатии кнопки «vis. app. res.»:



Правила работы с визуализатором такие же — переключение между страницами осуществляется с помощью меню «Page», либо с помощью кнопок «Enter» — вперед и «PageUp» — назад. Выход из визуализатора производится путём достижения последней страницы графиков и нажатия на любой области окна левой кнопки мыши (последняя страница имеет надпись снизу «Press mouse button to exit...»). При этом обратите внимание, что среди графиков вы увидите как логарифмические результаты, так и обычные. После просмотра графиков они также сохраняются в графических файлах формата *.png и имеют имя, начало которого совпадает с именем входного файла данных для инверсии *.bert, а конец имеет постфикс «dcinv» в случае обычных графиков и «dcinvLOG» в случае логарифмических графиков. Кроме того, после решения 2D задачи инверсии будут получены файлы, которые могут быть загружены в других визуализаторах. Прежде всего, это файлы с расширением *.vtk и *.vtu (<https://vtk.org/>). ОБРАТИТЕ ВНИМАНИЕ, что файлы типа *.vtk и *.vtu содержат сразу несколько информационных полей, то есть в них есть как информация о сетке, на которой решалась задача инверсии, так и полученные сопротивления, логарифмические сопротивления, чувствительность, инверсия вызванной поляризации (если данные вызванной поляризации были во входном файле, в противном случае такой информации в выходных файлах *.vtk и *.vtu не будет). Имена выходных файлов будут совпадать с именем входного файла данных для инверсии, а некоторые из них иметь постфикс «P». Данный постфикс означает то, что в этих файлах данные связаны с узлами сетки, в то время как в файлах без постфикса «P» данные связаны с ячейками сетки. В чём разница? Прежде всего в визуальном представлении. Данные, связанные

с узлами сетки, лучше воспринимаются глазом. Однако при этом ВАЖНО ПОМНИТЬ, что данные, связанные с ячейками сетки, более «правильные» с точки зрения алгоритма инверсии, так как результаты инверсии получаются в ячейках, а не в узлах. Кроме того, будут получены файлы с расширениями *.surf и *.bln. Это файлы для коммерческого визуализатора Surfer (<https://www.goldensoftware.com/products/surfer>). Имена этих файлов имеют постфикс «P» — обычные данные с сопротивлениями, связанные с узлами сетки; «PLOG» — логарифмические данные сопротивлений, связанные с узлами сетки; «P_IP» — результаты инверсии вызванной поляризации, связанные с узлами сетки (если данные вызванной поляризации были во входном файле, в противном случае файлов с постфиксом «P_IP» не будет). Файлы *.bln описывают внешнюю границу соответствующей области и необходимы в том случае, если вы работаете с данными со сложной топографией. Для их открытия воспользуйтесь функцией «Assign NoData to Grid» программы Surfer.

В 3D случае визуализатор имеет вид:

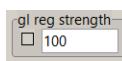


Данный визуализатор основан на визуализаторе Medit (<https://hal.inria.fr/inria-00069921/document>) и имеет схожий интерфейс. В данном мануале не будем останавливаться детально на каждой функции этого визуализатора. Можно лишь порекомендовать — нажать кнопку «h», находясь в окне визуализатора, и посмотреть в консольном окне DiInSo — какие кнопки за что отвечают. Работает в визуализаторе и мышка: левая кнопка мыши отвечает за вращение/перемещение, правая вызывает выпадающее меню, которое, в частности, дублирует функции, вызываемые с помощью нажатия клавиш на клавиатуре. ВАЖНО ПОМНИТЬ, что

раскладка клавиатуры при работе с визуализатором должна быть выбрана как «английская». Кроме того, после решения 3D задачи инверсии будут получены файлы, которые могут быть загружены в других визуализаторах. Прежде всего, это файлы с расширением *.vtk и *.vtu (<https://vtk.org/>). ОБРАТИТЕ ВНИМАНИЕ, что файлы типа *.vtk и *.vtu содержат сразу несколько информационных полей, то есть в них есть как информация о сетке, на которой решалась задача инверсии, так и полученные сопротивления, логарифмические сопротивления, чувствительность, инверсия вызванной поляризации (если данные вызванной поляризации были во входном файле, в противном случае такой информации в выходных файлах *.vtk и *.vtu не будет). Имена выходных файлов будут совпадать с именем входного файла данных для инверсии, а некоторые из них иметь постфикс «P». Данный постфикс означает то, что в этих файлах данные связаны с узлами сетки, в то время как в файлах без постфикса «P» данные связаны с ячейками сетки. В чём разница? Прежде всего в визуальном представлении. Данные, связанные с узлами сетки лучше воспринимаются глазом. Однако при этом ВАЖНО ПОМНИТЬ, что данные, связанные с ячейками сетки, более «правильные» с точки зрения алгоритма инверсии, так как результаты инверсии получаются в ячейках, а не в узлах. Также в 3D случае вы увидите особенный файл формата *.vtk с постфиксом «3D». Этот файл предназначен преимущественно для одного свободно распространяемого визуализатора — VolView (<https://www.kitware.com/volview/>), который отличается простотой и удобством в работе с подобными нашим данными. Кроме того, будут получены файлы с расширениями *.vox. Это файлы для коммерческого визуализатора Voxler (<https://www.goldensoftware.com/products/voxler>). Имена этих файлов имеют постфикс «P» — обычные данные с сопротивлениями, связанные с узлами сетки; «PLOG» — логарифмические данные сопротивлений, связанные с узлами сетки; «P_IP» — результаты инверсии вызванной поляризации, связанные с узлами сетки (если данные вызванной поляризации были во входном файле, в противном случае файлов с постфиксом «P_IP» не будет).

Итак, мы почти закончили рассмотрение той части окна, которая отвечает за решение задач инверсии. Но перед тем, как двинуться дальше, ОЧЕНЬ ВАЖНОЕ ЗАМЕЧАНИЕ! Рядом с некоторыми из опций (включая выбор файлов) можно наблюдать квадратик, где можно поставить галочку, вот такую: . Если эта галочка поставлена, то параметры для данной опции будут считываться из окна рядом. Если же она не проставлена, то параметры

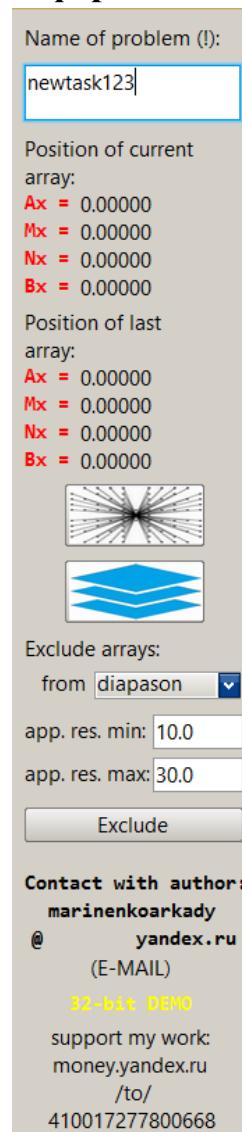
для данной опции будут браться из значений «по умолчанию». Например, для 2D опции «значения параметра регуляризации» — «gl reg strength»:

- пусть «галочка» активна, а в поле вписано значение «100» без кавычек:  , тогда алгоритм инверсии будет использовать параметр регуляризации равный 100;
- пусть «галочка» НЕ активна, а в поле всё ещё вписано значение «100» без кавычек:  , тогда алгоритм инверсии будет использовать параметр регуляризации «по умолчанию» (то есть 20).

Сделано это исключительно для удобства, чтобы не было необходимости каждый раз стирать набранные значения — просто уберите галочку!

Наконец, переходим к последней части главного окна программы, которая расположена справа...

Вспомогательные и информационные функции программы.

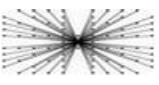


Данная часть программы преимущественно вспомогательная и информационная, но имеет и некоторые очень важные функции.

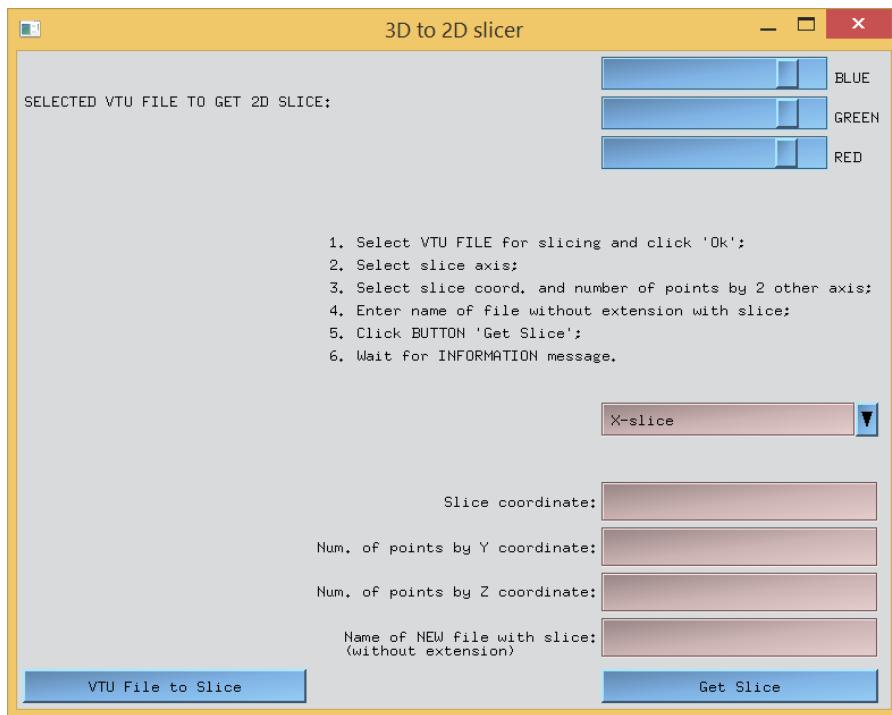
В самом верху этой части программы вы можете видеть поле для ввода имени задачи, которое называется «Name of problem (!):». Восклицательный знак здесь совсем не случайно. Имя задачи имеет очень важное значение для решения ПРЯМЫХ ЗАДАЧ. Как вы уже читали выше, процесс решения прямых задач разделён на множество этапов, где есть, в частности: генерация матрицы системы линейных алгебраических уравнений; установка соответствий между номерами узлов сетки и электродами установки; сборка всех полученных решений в файлы *.bert (кнопка «bring together») и так далее. На каждом из этих этапов мы получаем какие-то файлы, которые привязываются к имени задачи, которое задано в поле «Name of problem (!):». Когда полученные решения собираются в файлы *.bert (кнопка «bring together»), то файлы с расширением *.sol не удаляются, а перемещаются в папку с именем, совпадающим с именем задачи, которая создаётся в корневой директории программы DiInSo. Сделано это для того, чтобы не возникло путаницы между разными решаемыми прямыми задачами, и чтобы избавить алгоритм решения от повторяющихся действий, если они уже были совершены ранее. Поэтому, как только вы начинаете решать новую прямую задачу — задавайте новое имя в поле «Name of problem (!):». При этом рекомендации и требования к имени задачи такие: желательно не использовать пробелы в имени задачи и НЕЛЬЗЯ использовать служебные символы в имени задачи: тильда (~), знак номера (#), процент (%), амперсанд (&), звездочка (*), фигурные скобки ({}), обратная косая черта (\), двоеточие (:), угловые скобки (<>), вопросительный знак (?), косая черта (/), знак плюс (+), вертикальная черта (|), знак кавычек ("'). При решении задачи инверсии имя задачи не играет никакой роли, так как решение задачи инверсии не содержит столько зависящих от пользователя этапов как решение прямой задачи.

Ниже имени задачи вы можете наблюдать НЕ редактируемые поля с именами «Position of current array:», «Position of last array:» и «Ax =», «Mx =», «Nx =», «Bx =» за ними. Эти поля также связаны с прямой задачей. Во время решения множества прямых задач происходит «перемещение» установки из одних положений в другие. Это перемещение, а именно X-координаты питающих (A, B) и приемных (M, N) электродов будут отображаться ниже слов «Position of current array:». Ниже слов «Position of last array:» будут отображаться X-координаты электродов установки для той прямой задачи, которая будет решаться последней в нашем списке задач, то есть это

положение установки, к которому мы последовательно «приближаемся», решая множество прямых задач.

Функцию кнопки  мы уже рассматривали выше — это автоматический генератор координат множества положений установок по всем профилям, которые сохраняются в файле elec_profiles.txt.

Кнопка  вызывает окно вида:



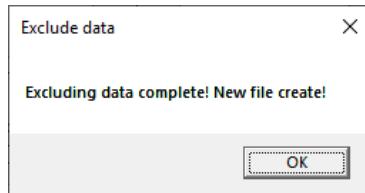
С помощью данного окна вы можете получить двумерный разрез, перпендикулярный оси X, Y или Z, из вашей трёхмерной модели, сохранённой в файле формата VTU (VTK формат не поддерживается!). Процедура достаточно проста: 1. Выбираете VTU-файл с помощью кнопки «VTU File to Slice»; 2. Выбираете какой оси будет перпендикулярен разрез с помощью выпадающего меню; 3. Вводите координаты разреза в поле «Slice coordinate:»; 4. Вводите число точек данных по соответствующим координатам в двух полях ниже (чем больше точек, тем подробнее будет информация о разрезе, однако не следует забывать, что точность данных определяет числом точек во входном файле); 5. Вписываем имя выходного файла без расширения; 6. Нажав кнопку «Get Slice» получаем выходные файлы с разрезом в форматах VTK, VTU и Surfer.

Конечно, всё это вы можете проделать и в соответствующем 3D визуализаторе, однако практика показывает, что люди, долго работающие с 2D визуализаторами, с трудом переучиваются для работы с 3D

визуализаторами (например, из Surfer в Voxler). Эта функция была реализована для них.

Ниже слов «Exclude arrays:» находится функционал программы, который призван помочь в корректировке входного файла для ЗАДАЧИ ИНВЕРСИИ. Выше уже рассматривался вариант, когда возникает необходимость исключить некоторые данные из решения задачи инверсии. Мы это можем сделать с помощью опций раздела «DATA FILTERING:». Однако бывает необходимость изменить сам входной файл, навсегда убрав из него те данные, которые нас не устраивают по той или иной причине. После слова «from:» можно наблюдать выпадающее меню с тремя значениями: «diapason», «file», «graphic». Пойдём по порядку...

В случае «diapason» укажите диапазон значений кажущихся сопротивлений в полях «app. res. min:» и «app. res. max:», нажмите кнопку «Exclude», дождитесь информационного сообщения:

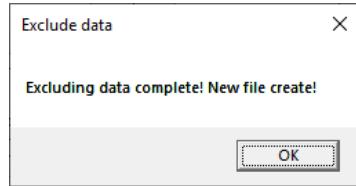


И в той же директории, что и входной файл (указанный в поле «DATA FILE:»), вы получите новый файл задачи инверсии *.bert, имя которого совпадает со входным файлом плюс постфикс «newdiap». В этом файле все данные, входящие в диапазон кажущихся сопротивлений от «app. res. min:» до «app. res. max:», окажутся исключёнными.

В случае «file» укажите имя файла с номерами положений установок, которые должны быть исключены. Формат этого файла такой:

```
<число исключаемых положений установок>
<номер первого исключаемого положения установки>
<номер второго исключаемого положения установки>
...
<номер последнего исключаемого положения установки>
```

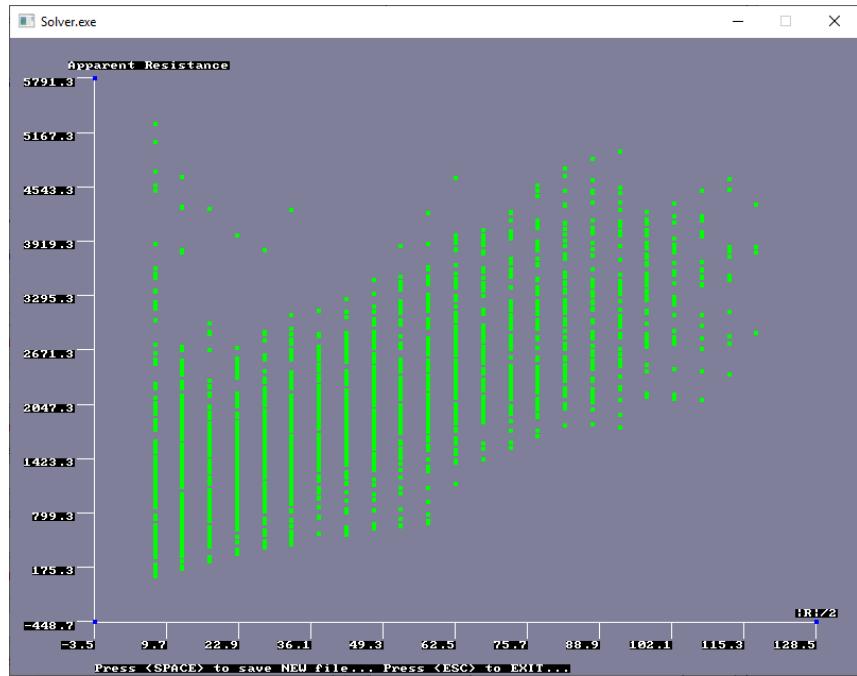
Нажмите кнопку «Exclude», дождитесь информационного сообщения:



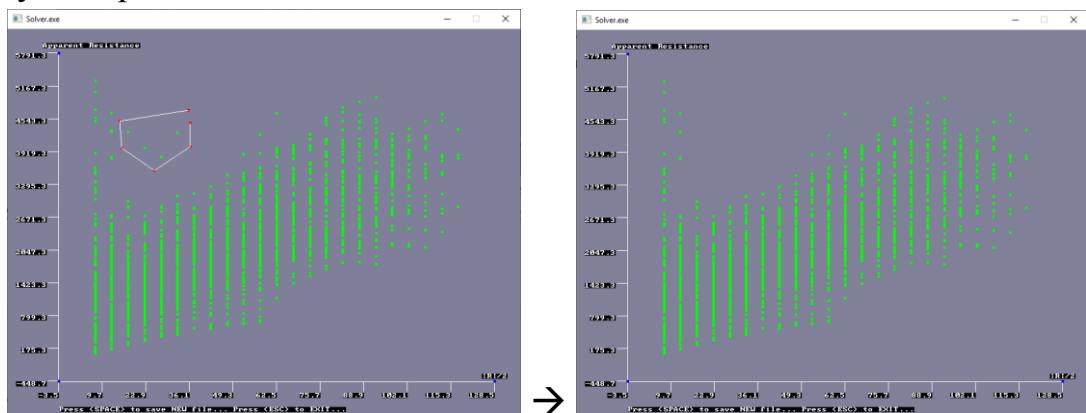
И в той же директории, что и входной файл (указанный в поле «DATA FILE:»), вы получите новый файл задачи инверсии *.bert, имя которого совпадает со входным файлом плюс постфикс «newarr». Здесь все установки,

перечисленные в выбранном файле, указанном в поле «select file:», окажутся исключенными.

В случае «graphic», после нажатия кнопки «Exclude» откроется окно интерактивного удаления «ненужных» данных:



По оси X в этом окне отложены значения половины разноса установок R/2, а по оси Y значения кажущихся сопротивлений, полученных на установках. С помощью левой кнопки мыши «окружите» ненужные значения замкнутой кривой:



... и эти значения будут исключены. Повторяйте процедуру «удаления точек» столько, сколько вам нужно. Как только закончите, можно сохранить НОВЫЙ файл с исключенными положениями установок, нажав кнопку пробел на клавиатуре. Новый файл данных для инверсии *.bert будет создан в той же директории, что и входной файл (указанный в поле «DATA FILE:»), а его имя будет совпадать с именем входного файла плюс постфикс «newgraph». Чтобы выйти из окна «исключения точек» нажмите ESC.

Наконец, в самом низу данной части главного окна программы вы увидите информацию о главном разработчике, его e-mail и способе поддержать проект.

Осталось рассказать про две кнопки, которые изменяют интерфейс



главного окна: . Кнопка « \wedge » скрывает верхнюю (относящуюся к прямым задачам), а кнопка « \vee » — нижнюю (относящуюся к задачам инверсии) области главного окна программы DiInSo. Зачем это нужно? Например, вы не планируете решать задачи инверсии, а вас интересуют только прямые задачи. В этом случае вы нажимаете кнопку « \vee » и главное окно программы не занимает лишнее пространство на экране монитора.

Кроме основного окна программы, есть также вспомогательное консольное окно, которое всегда открыто вместе с основным окном и которое НЕЛЬЗЯ ЗАКРЫВАТЬ, потому что закрытие этого окна приводит к аварийному закрытию всей программы:

```
D:\WinSolver\Release\Solver.exe
constraint matrix of size(nBounds x nModel) 5365 x 3751
calculating jacobian matrix ..... 18.351 s
min data = 17.2636 max data = 5271.28 (1868)
min error = 0.030006 max error = 0.0314972 (1868)
min response = 2036.46 max response = 2036.46 (1868)
min reference model = 2036.46 max reference model = 2036.46 (3751)
0: rms/rrms(data, response) = 972.14/171.984%
0: chi^2(data, response, error, log) = 374.103
0: Phi = 698824 + 0 * 20 = 698824
Iter: 0
recalculating jacobian matrix ...14.26 s
solve CGLSCDMtrans with lambda = 20
1: LS newModel: min = 29.8081; max = 46364.5
1: LS newResponse: min = 64.0072; max = 5263.95
1: rms/rrms(data, LS newResponse) = 447.424/20.9534%
1: chi^2(data, LS newResponse, error, log) = 73.2343
1: Phi = 136802+284.283*20=142487
Linesearch tau = 0.8
1: Model: min = 69.3814; max = 24815.5
1: Response: min = 131.089; max = 4374.16
1: rms/rrms(data, Response) = 398.481/17.7212%
1: chi^2(data, Response, error, log) = 37.321
1: Phi = 69715.6+181.941*20=73354.4
Iter: 1
recalculating jacobian matrix ...
```

Подобная «двуоконная» структура программы встречается во многих научных пакетах, например, в том же NetGen. В консольном окне содержится так называемый ЛОГ работы программы, то есть множество подробной информации как программистского характера, так и алгоритмического. Здесь вы можете, например, наблюдать за ходом сходимости прямой задачи и задачи инверсии, отслеживать часть информации, прочитанной в файлах, получать сведения о расходуемой памяти и многое другое. Обычно к этому окну обращаются в двух случаях — проверить не «застопорилось» ли решение задачи, либо найти причину ошибки. Например, возможен вариант, когда сеточный алгоритм не может правильно построить сетку для решения задачи инверсии и уходит в «бесконечный цикл». В этом случае в консольном окне можно будет наблюдать повторяющиеся однотипные сообщения. Продолжать работу программы в этом случае не имеет смысла и

её нужно завершить аварийным способом. Другим примером может быть случай, когда решается очень большая задача и программа пытается занять всё свободное пространство оперативной памяти. Операционная система будет пытаться выгружать все процессы, кроме критических, в так называемый файл подкачки (swap), чтобы освободить больше оперативной памяти. Всё это вызывает неприятные «зависания» в работе и может даже казаться, что компьютер перестал отвечать на действия пользователя. В этом случае иногда лучше завершить работу программы аварийным способом — нажав «**CTRL+C**» в консольном окне (данний метод работает в большинстве консольных приложений). В консольном окне могут встречаться незнакомые вам слова (если вы читаете данный мануал на английском языке) на русском языке. Неплохой стимул выучить русский язык, ну или просто не обращайте на них внимание.

ВАЖНЫЕ вопросы, которые встречаются чаще всего, и ответы на них.

1. Для чего создавалась данная программа?

Программа создавалась исключительно в академических целях, то есть с целью помочь научному сообществу, занятому в электротомографических исследованиях.

2. Сколько человек участвовали в разработке программы?

Мариненко Аркадий Вадимович — главный разработчик DiInSo, Оленченко Владимир Владимирович, Эпов Михаил Иванович и лаборатория геоэлектрики ИНГГ СО РАН — помочь в тестировании программы и интересные идеи, OpenSource-сообщество — люди, без которых данный проект мог бы быть намного хуже и без которых развитие программирования практически невозможно.

3. Кем является главный разработчик программы?

Математиком, программистом и геофизиком.

4. Я идеально настроил параметры программы и теперь боюсь их потерять!

Не бойтесь, если вы корректно завершили работу программы (путем нажатия крестика в главном окне программы DiInSo), то все введённые вами параметры будут сохранены и восстановлены после перезапуска программы (если вы снимите «галочку» с параметра, то он не сохранится).

5. Я так и не понял, что означает этот параметр при решении задачи инверсии и очень боюсь выставить его неправильно!

Просто ничего не выставляйте, программа DiInSo использует значение «по умолчанию», которое часто оказывается не таким уж плохим.

6. Мне кажется, что программа «застопорилась»/«зависла» и её нужно завершить. Возможно ли такое и как лучше закрыть DiInSo?

*Да, к сожалению, такое возможно и причин этому множество — от ошибки в самой программе до частных случаев, вроде невозможности построить сетку с указанными параметрами для данной конкретной модели при решении задачи инверсии. В таком случае лучше всего закрыть DiInSo нажатием сочетания клавиш «**CTRL+C**» в консольном окне программы.*

7. Я точно нашёл ошибку в программе! Такое возможно? Что мне делать?

*Конечно. Данная программа не лишена ошибок, возможно даже критических. Напишите на e-mail главного разработчика (marinenkoarkady@yandex.ru) информацию об ошибке, последовательность действий, которая приводит к ошибке и прикрепите log-файл, который находится в папке вместе с программой DiInSo и имеет имя *logging.log*.*

8. Я уже много раз прочитал мануал, но никак не могу понять ЭТО! ЛИБО: У меня есть отличная идея для программы! ЛИБО: У вас в мануале неточность! Можно ли мне написать автору?

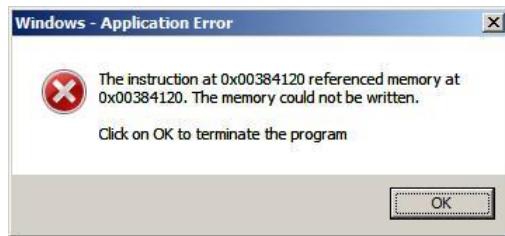
Конечно, не стесняйтесь, пишите главному разработчику на e-mail (marinenkoarkady@yandex.ru) с подробным описанием вашей проблемы, мои ошибки в мануале (которые не исключены, ведь в программе использован не только собственноручно написанный код, но и коды других учёных-разработчиков) или предложения по улучшению программы и не обижайтесь, если получите ответ не сразу.

9. На каком языке можно писать автору программы?

Русский или английский.

10. Как понять, что программе не хватает памяти?

*Обычно ошибка нехватки памяти ничем не отличается от других ошибок программы, то есть программа просто некорректно завершается. Часто это завершение сопровождается надписью «*Not enough memory*» в консольном окне программы или отдельным окном с ошибкой «*The memory could not be written*»:*



11. У меня ОЧЕНЬ МНОГО оперативной памяти, но программе не хватает памяти для решения задачи. Как такое может быть?

Вероятно, вы используете 32-битную версию программы, которая имеет естественное ограничение в 2 гигабайта оперативной памяти.

12. Чем отличается 64-битная версия DiInSo от 32-битной?

Прежде всего и самое главное, отсутствует ограничение в 2 гигабайта используемой оперативной памяти, которое свойственно всем 32-битным приложениям. Кроме того, в 64-битной версии программы есть возможность выбора метода решения прямых задач; прямые задачи могут решаться на видеокартах (как правило, значительно быстрее решения задач на процессорах) с помощью технологий CUDA и/или OpenCL; имеются альтернативные визуализаторы данных, встроенные в программу; обеспечивается более высокая точность решения за счёт использования 64-битных типов данных.

13. Как я могу получить 64-битную версию программы?

Свяжитесь с главным разработчиком программы Мариненко Аркадием Вадимовичем, либо с руководителем лаборатории геоэлектрики ИНГГ СО РАН Оленченко Владимиром Владимировичем (а лучше с обоими сразу) через e-mail (marinenkoarkady@yandex.ru и olenchenkovv@ipgg.sbras.ru).

14. Я хочу поддержать автора проекта DiInSo? Как я могу это сделать?

С помощью спонсорского пожертвования на один из указанных адресов:

Яндекс деньги: <https://money.yandex.ru/to/410017277800668>

Qiwi: <https://qiwi.com/n/ARKADIY1983>

PayPal: <https://www.paypal.me/arkadiy1983>

Patreon: <https://www.patreon.com/arkadiy1983>

Steady: <https://steadyhq.com/en/arkadiy1983>

LP: <https://liberapay.com/arkadiy1983>

15. Сколько нужно времени на усовершенствование проекта до состояния, близкого к идеальному?

Чётких рамок нет, как и у любой подобной программы, но если заинтересованность проектом будет, то его развитие обязательно продолжится.

16. Я освоил 32-битную версию программы. Мне нужно будет заново осваивать 64-битную версию?

Конечно же нет, 64-битная версия имеет очень мало отличий от 32-битной. Однако при этом 64-битная версия позволяет решать большие и сложные задачи, которые не умещаются в 2 гигабайта оперативной памяти.

17. Я не признаю Windows, только на Linux создаются серьёзные научные программы! Есть ли версия программы для Linux?

Вы правы, Linux куда лучше подходит для научных программ, но далеко не каждый пользователь способен освоить Linux. Версия под Linux (Ubuntu) существует, однако её развитие в данный момент приостановлено из-за нехватки времени. Одновременно с этим, спешу заверить, что если интерес к программе будет большим, то эта версия быстро увидит свет, поскольку программа DiInSo изначально написана без жёсткой привязки к операционной системе и по сути мультиплатформенна.

18. Возможно ли, что проект DiInSo будет полностью OpenSource?

Возможно, в будущем, но сейчас это не имеет смысла. Компиляция DiInSo на данном этапе — процедура не из лёгких, требует учёта многих тонкостей и пока не имеет автоматических «сборщиков», вроде CMAKE (<https://cmake.org/>).

19. Я раньше использовал другую программу инверсии, там было проще...

Чем проще использовать программу, тем больше программа делает «автоматически». Чем больше автоматики в программе, тем меньше возможностей получить адекватный научно выверенный результат. В данной программе была сделана попытка создать баланс между простотой и научностью подхода решения задач.

20. Использует ли программа параллельные подходы?

Да, причём как в решении прямых задач, так и в решении задач инверсии. Чем больше у вас процессоров/ядер и оперативной памяти, тем быстрее будет решаться задача и тем сложнее задачи вы сможете решить (хотя последнее, скорее, актуально для 64-битной версии).

21.Могу ли я свободно распространять программу DiInSo, скачанную с сайта sourceforge.net / github.com / gitlab.com / форума / Яндекс-диска / Google-диска / флэшки друга?

Да, и это даже приветствуется. Чем больше распространится программа DiInSo, тем больше будет заинтересованность в ней и тем больше будет желание развивать проект.

22.Чем лучше просматривать VTK-файлы?

Лучшими программами для просмотра VTK-файлов по мнению многих являются ParaView — <https://www.paraview.org/>, VisIt — <https://wci.llnl.gov/simulation/computer-codes/visit> и Cassandra — <http://dev.artenum.com/projects/cassandra>

23.Можно ли как-то посмотреть геометрии/сетки 2D файлов *.poly программы Triangle и геометрии/сетки 3D файлов *.ply программы Tetgen?

Да, в 2D случае вы можете использовать программу Triangle.NET — <https://archive.codeplex.com/?p=triangle> или <https://github.com/eppz/Triangle.NET> или <https://github.com/garykac/triangle.net>, а в 3D случае программу TetView — <http://wias-berlin.de/software/tetgen/tetview.html>

Основные OpenSource проекты, без которых создание DiInSo затянулось бы на многие годы.

Интерфейс основного окна программы: <http://nanapro.org/en-us/>

Интерфейс дополнительных окон программы: <http://plib.sourceforge.net/> и <https://www.wxwidgets.org/>

Рисование 2D графиков: <http://plplot.sourceforge.net/>

Рисование 3D графиков: <https://www.ljll.math.upmc.fr/frey/software.html>

Рисование 3D поверхностей равного уровня (только 64-битная версия): <http://visualizationlibrary.org/docs/2.0/html/index.html>

Рисование портрета матрицы: <https://github.com/INMOST-DEV/INMOST>

Конвертирование 3D форматов: <http://www.assimp.org/>

Ведение логов: <https://sourceforge.net/projects/getset/> или <https://github.com/aaichert/LibGetSet>

Работа с PDF-файлами: <https://mupdf.com/>

Интерактивная работа с графикой: <https://liballeg.org/>

Поддержка OpenGL: <http://freeglut.sourceforge.net/>

Поддержка OpenCL (только 64-битная версия): <https://www.khronos.org/opencl/>

Поддержка CUDA (только 64-битная версия):
<https://developer.nvidia.com/cuda-zone>

Решение прямых задач: <http://www.paralution.com/>

Решение задач инверсии, работа с геометриями для задач инверсии:
<https://www.pygimli.org/> и <https://gitlab.com/resistivity-net/bert>

Автоматическая генерация треугольной сетки при решении 2D задач инверсии: <https://www.cs.cmu.edu/~quake/triangle.html>

Автоматическая генерация тетраэдralной сетки при решении 3D задач инверсии: <http://wias-berlin.de/software/index.jsp?id=TetGen>

Поддержка формата VTK: <https://vtk.org/>

Генераторы тетраэдralных сеток для решения прямых задач электротомографии (не включены в проект, но результаты работы этих программ используются как входные данные): <http://gmsh.info/> и <https://ngsolve.org/> и <https://www.salome-platform.org/>

А также другие проекты, которые связаны с вышенназванными проектами.

Пример решения прямой задачи электротомографии с последующим решением задачи инверсии без сложных объектов с помощью генератора сеток NetGen.

Рассмотрим простейшую прямую задачу электротомографии, где нет никаких сложных объектов, а есть только двухслойная среда, граница которой находится на глубине 10 метров. Сразу приступим к созданию файла с геометрией для генератора сеток NetGen. Начинаются такие файлы с ключевого слова:

algebraic3d

Далее задаём 6 плоскостей, которые будут ограничивать нашу область моделирования. Одна из этих плоскостей будет пересекать точки, где заданы электроды, а остальные 5 плоскостей удалены от электродов на большое расстояние:

```
# Нижняя граница
solid plane1= plane(    100.0,      100.0,     -200.0;0,0,-1) -bc=76;
# Передняя граница
solid plane2= plane(    100.0,     -50.0,      50.0;0,-1,0) -bc=76;
# Левая граница
solid plane3= plane(   -50.0,      100.0,      50.0;-1,0,0) -bc=76;
# Верхняя граница
solid plane4= plane(    100.0,      100.0,       0.0;0,0,1);
# Задняя граница
solid plane5= plane(    100.0,     200.0,      50.0;0,1,0) -bc=76;
# Правая граница
solid plane6= plane(   250.0,      100.0,      50.0;1,0,0) -bc=76;
```

На удалённых от электродов плоскостях задаём нулевые граничные условия Дирихле. Делается это с помощью параметра «-bc=76», то есть мы

будем считать, что нулевые граничные условия Дирихле задаются с помощью значения «76». Объединяя плоскости в единое целое и получаем область в форме «бака»:

```
solid bak= planev1 and planev2 and planev3 and planev4 and planev5 and planev6;
```

Далее разделяем наш «бак» на несколько слоёв, два из которых будут представлять из себя естественные слои с разными значениями электропроводности, а ещё два — фиктивные слои, которые будут служить для настройки параметров измельчения сетки. Для этого сначала задаём три плоскости-разделителя:

```
solid cutplane1= plane(    100.0,      100.0,      -1.0;0,0,1);
solid cutplane2= plane(    100.0,      100.0,      -4.0;0,0,1);
solid cutplane3= plane(    100.0,      100.0,      -10.0;0,0,1);
```

А затем четыре слоя:

```
solid slice1= bak and not cutplane1;
solid slice2= bak and cutplane1 and not cutplane2;
solid slice3= bak and cutplane2 and not cutplane3;
solid slice4= bak and cutplane3;
```

Теперь перечисляем эти слои в качестве объектов-сред (собственно, других объектов-сред в нашей области нет), делается это с помощью ключевого слова «tlo»:

```
tlo slice1 -col=[0,1,0] -transparent -material=slice1 -maxh=      1.0;
tlo slice2 -col=[0,1,0] -transparent -material=slice2 -maxh=      2.0;
tlo slice3 -col=[0,1,0] -transparent -material=slice3 -maxh=      5.0;
tlo slice4 -col=[0,1,0] -transparent -material=slice4 -maxh= 20.0;
```

С помощью ключевого слова «-col» задаётся цвет объекта-среды в генераторе NetGen, «-transparent» указывает на прозрачность объекта-среды, а «-material» указывает сеточному генератору, что данный объект-среда относится к соответствующему «материалу». Наконец, «-maxh» задаёт параметры сетки, а именно максимальный размер ребра тетраэдра в этом объекте-среде. Как уже говорилось выше, первые три слоя — это по сути один слой с точки зрения физики, однако мы их разделили по разным материалам, потому что привычка задавать каждый объект (даже фиктивный) отдельными параметрами является хорошей привычкой.

Остался последний шаг — перечислить все точки, где находятся электроды:

```
point(   50.0,      50.0,      0.0);
point(   55.0,      50.0,      0.0);
point(   60.0,      50.0,      0.0);
point(   65.0,      50.0,      0.0);
point(   70.0,      50.0,      0.0);
point(   75.0,      50.0,      0.0);
point(   80.0,      50.0,      0.0);
point(   85.0,      50.0,      0.0);
point(   90.0,      50.0,      0.0);
point(   95.0,      50.0,      0.0);
point(  100.0,      50.0,      0.0);
point(  105.0,      50.0,      0.0);
point(  110.0,      50.0,      0.0);
point(  115.0,      50.0,      0.0);
point(  120.0,      50.0,      0.0);
point(  125.0,      50.0,      0.0);
point(  130.0,      50.0,      0.0);
point(  135.0,      50.0,      0.0);
point(  140.0,      50.0,      0.0);
point(  145.0,      50.0,      0.0);
point(  150.0,      50.0,      0.0);
point(   50.0,      55.0,      0.0);
```

```

point( 55.0,      55.0,      0.0);
point( 60.0,      55.0,      0.0);
point( 65.0,      55.0,      0.0);
point( 70.0,      55.0,      0.0);
point( 75.0,      55.0,      0.0);
point( 80.0,      55.0,      0.0);
point( 85.0,      55.0,      0.0);
point( 90.0,      55.0,      0.0);
point( 95.0,      55.0,      0.0);
point(100.0,      55.0,      0.0);
point(105.0,      55.0,      0.0);
point(110.0,      55.0,      0.0);
point(115.0,      55.0,      0.0);
point(120.0,      55.0,      0.0);
point(125.0,      55.0,      0.0);
point(130.0,      55.0,      0.0);
point(135.0,      55.0,      0.0);
point(140.0,      55.0,      0.0);
point(145.0,      55.0,      0.0);
point(150.0,      55.0,      0.0);
point( 50.0,      60.0,      0.0);
point( 55.0,      60.0,      0.0);
point( 60.0,      60.0,      0.0);
point( 65.0,      60.0,      0.0);
point( 70.0,      60.0,      0.0);
point( 75.0,      60.0,      0.0);
point( 80.0,      60.0,      0.0);
point( 85.0,      60.0,      0.0);
point( 90.0,      60.0,      0.0);
point( 95.0,      60.0,      0.0);
point(100.0,      60.0,      0.0);
point(105.0,      60.0,      0.0);
point(110.0,      60.0,      0.0);
point(115.0,      60.0,      0.0);
point(120.0,      60.0,      0.0);
point(125.0,      60.0,      0.0);
point(130.0,      60.0,      0.0);
point(135.0,      60.0,      0.0);
point(140.0,      60.0,      0.0);
point(145.0,      60.0,      0.0);
point(150.0,      60.0,      0.0);
point( 50.0,      65.0,      0.0);
point( 55.0,      65.0,      0.0);
point( 60.0,      65.0,      0.0);
point( 65.0,      65.0,      0.0);
point( 70.0,      65.0,      0.0);
point( 75.0,      65.0,      0.0);
point( 80.0,      65.0,      0.0);
point( 85.0,      65.0,      0.0);
point( 90.0,      65.0,      0.0);
point( 95.0,      65.0,      0.0);
point(100.0,      65.0,      0.0);
point(105.0,      65.0,      0.0);
point(110.0,      65.0,      0.0);
point(115.0,      65.0,      0.0);
point(120.0,      65.0,      0.0);
point(125.0,      65.0,      0.0);
point(130.0,      65.0,      0.0);
point(135.0,      65.0,      0.0);
point(140.0,      65.0,      0.0);
point(145.0,      65.0,      0.0);
point(150.0,      65.0,      0.0);
point( 50.0,      70.0,      0.0);
point( 55.0,      70.0,      0.0);
point( 60.0,      70.0,      0.0);
point( 65.0,      70.0,      0.0);
point( 70.0,      70.0,      0.0);
point( 75.0,      70.0,      0.0);
point( 80.0,      70.0,      0.0);
point( 85.0,      70.0,      0.0);
point( 90.0,      70.0,      0.0);
point( 95.0,      70.0,      0.0);
point(100.0,      70.0,      0.0);
point(105.0,      70.0,      0.0);
point(110.0,      70.0,      0.0);
point(115.0,      70.0,      0.0);
point(120.0,      70.0,      0.0);
point(125.0,      70.0,      0.0);
point(130.0,      70.0,      0.0);
point(135.0,      70.0,      0.0);
point(140.0,      70.0,      0.0);
point(145.0,      70.0,      0.0);
point(150.0,      70.0,      0.0);
point( 50.0,      75.0,      0.0);
point( 55.0,      75.0,      0.0);
point( 60.0,      75.0,      0.0);
point( 65.0,      75.0,      0.0);
point( 70.0,      75.0,      0.0);
point( 75.0,      75.0,      0.0);
point( 80.0,      75.0,      0.0);
point( 85.0,      75.0,      0.0);
point( 90.0,      75.0,      0.0);

```

```

point(    95.0,      75.0,      0.0);
point(   100.0,      75.0,      0.0);
point(   105.0,      75.0,      0.0);
point(   110.0,      75.0,      0.0);
point(   115.0,      75.0,      0.0);
point(   120.0,      75.0,      0.0);
point(   125.0,      75.0,      0.0);
point(   130.0,      75.0,      0.0);
point(   135.0,      75.0,      0.0);
point(   140.0,      75.0,      0.0);
point(   145.0,      75.0,      0.0);
point(   150.0,      75.0,      0.0);
point(    50.0,      80.0,      0.0);
point(    55.0,      80.0,      0.0);
point(    60.0,      80.0,      0.0);
point(    65.0,      80.0,      0.0);
point(    70.0,      80.0,      0.0);
point(    75.0,      80.0,      0.0);
point(    80.0,      80.0,      0.0);
point(    85.0,      80.0,      0.0);
point(    90.0,      80.0,      0.0);
point(   95.0,      80.0,      0.0);
point(  100.0,      80.0,      0.0);
point(  105.0,      80.0,      0.0);
point(  110.0,      80.0,      0.0);
point(  115.0,      80.0,      0.0);
point(  120.0,      80.0,      0.0);
point(  125.0,      80.0,      0.0);
point(  130.0,      80.0,      0.0);
point(  135.0,      80.0,      0.0);
point(  140.0,      80.0,      0.0);
point(  145.0,      80.0,      0.0);
point(  150.0,      80.0,      0.0);
point(    50.0,      85.0,      0.0);
point(    55.0,      85.0,      0.0);
point(    60.0,      85.0,      0.0);
point(    65.0,      85.0,      0.0);
point(    70.0,      85.0,      0.0);
point(    75.0,      85.0,      0.0);
point(    80.0,      85.0,      0.0);
point(    85.0,      85.0,      0.0);
point(    90.0,      85.0,      0.0);
point(   95.0,      85.0,      0.0);
point(  100.0,      85.0,      0.0);
point(  105.0,      85.0,      0.0);
point(  110.0,      85.0,      0.0);
point(  115.0,      85.0,      0.0);
point(  120.0,      85.0,      0.0);
point(  125.0,      85.0,      0.0);
point(  130.0,      85.0,      0.0);
point(  135.0,      85.0,      0.0);
point(  140.0,      85.0,      0.0);
point(  145.0,      85.0,      0.0);
point(  150.0,      85.0,      0.0);
point(    50.0,      90.0,      0.0);
point(    55.0,      90.0,      0.0);
point(    60.0,      90.0,      0.0);
point(    65.0,      90.0,      0.0);
point(    70.0,      90.0,      0.0);
point(    75.0,      90.0,      0.0);
point(    80.0,      90.0,      0.0);
point(    85.0,      90.0,      0.0);
point(    90.0,      90.0,      0.0);
point(   95.0,      90.0,      0.0);
point(  100.0,      90.0,      0.0);
point(  105.0,      90.0,      0.0);
point(  110.0,      90.0,      0.0);
point(  115.0,      90.0,      0.0);
point(  120.0,      90.0,      0.0);
point(  125.0,      90.0,      0.0);
point(  130.0,      90.0,      0.0);
point(  135.0,      90.0,      0.0);
point(  140.0,      90.0,      0.0);
point(  145.0,      90.0,      0.0);
point(  150.0,      90.0,      0.0);
point(    50.0,      95.0,      0.0);
point(    55.0,      95.0,      0.0);
point(    60.0,      95.0,      0.0);
point(    65.0,      95.0,      0.0);
point(    70.0,      95.0,      0.0);
point(    75.0,      95.0,      0.0);
point(    80.0,      95.0,      0.0);
point(    85.0,      95.0,      0.0);
point(    90.0,      95.0,      0.0);
point(   95.0,      95.0,      0.0);
point(  100.0,      95.0,      0.0);
point(  105.0,      95.0,      0.0);
point(  110.0,      95.0,      0.0);
point(  115.0,      95.0,      0.0);
point(  120.0,      95.0,      0.0);
point(  125.0,      95.0,      0.0);
point(  130.0,      95.0,      0.0);

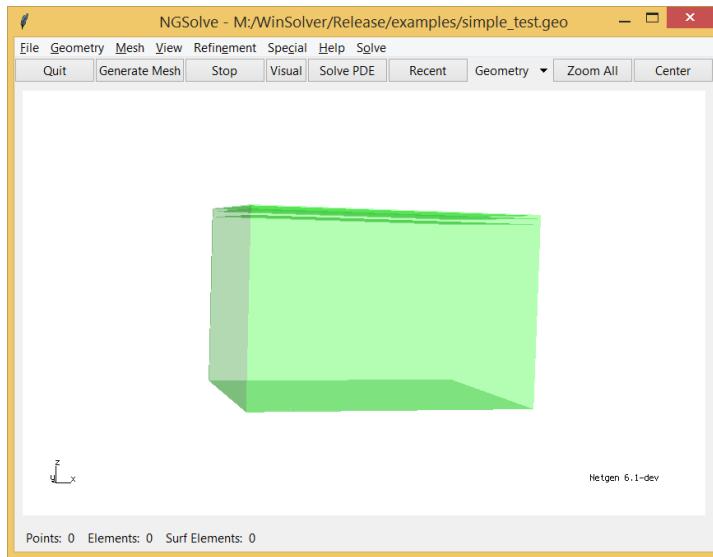
```

```

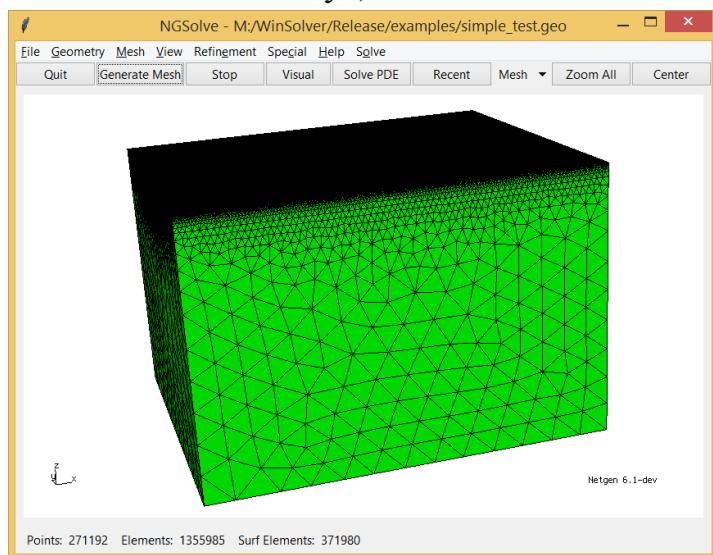
point( 135.0,      95.0,      0.0);
point( 140.0,      95.0,      0.0);
point( 145.0,      95.0,      0.0);
point( 150.0,      95.0,      0.0);
point( 50.0,       100.0,     0.0);
point( 55.0,       100.0,     0.0);
point( 60.0,       100.0,     0.0);
point( 65.0,       100.0,     0.0);
point( 70.0,       100.0,     0.0);
point( 75.0,       100.0,     0.0);
point( 80.0,       100.0,     0.0);
point( 85.0,       100.0,     0.0);
point( 90.0,       100.0,     0.0);
point( 95.0,       100.0,     0.0);
point( 100.0,      100.0,     0.0);
point( 105.0,      100.0,     0.0);
point( 110.0,      100.0,     0.0);
point( 115.0,      100.0,     0.0);
point( 120.0,      100.0,     0.0);
point( 125.0,      100.0,     0.0);
point( 130.0,      100.0,     0.0);
point( 135.0,      100.0,     0.0);
point( 140.0,      100.0,     0.0);
point( 145.0,      100.0,     0.0);
point( 150.0,      100.0,     0.0);

```

Полученный файл (есть в папке «examples» программы DiInSo, называется simple_test.geo) можно открыть в генераторе сеток NetGen:



Осталось только создать сетку, нажав кнопку «Generate Mesh» (на создание сетки уйдёт несколько минут):



Хорошая мелкая подробная сетка вблизи электродов — один из залогов успеха решения прямой задачи. Однако, не забываем, что слишком подробная сетка требует много памяти, поэтому переусердствовать с мелкостью сетки тоже не нужно. Сохраняем сетку в файл simple_test.vol через выпадающее меню File -> Save Mesh...

Открываем программу DiInSo. В поле «Name of problem (!):» придумываем имя нашей задачи. Пусть это будет, например, «my_simple_test». Нажимаем кнопку «select file with mesh:» и выбираем файл с полученной сеткой simple_test.vol. Теперь можно подумать о типе установки, которая будет использована для решения. Пусть это будет

установка поль-диполь. Нажимаем кнопку  в выпадающем меню выбираем «Pole-Dipole». Далее вспоминаем, как у нас располагаются электроды и сколько их: X0 = 50, Y0 = 50, Z0 = 0, Number of electrodes in one profile — 21, Number of profiles — 11, X_step — 5, Y_step — 5. Жмём кнопку «Generate» и новый файл elec_profiles.txt создан. Ради интереса, можно открыть его (через выпадающее меню «FILE OPEN») и узнать, что общее число задач, которое нам придётся решать, равняется 4290. Убеждаемся, что нулевые краевые условия Дирихле заданы тем числом, которое нам нужно, а именно 76. Для этого открываем файл null.txt (через выпадающее меню «FILE OPEN»). Теперь проверяем настройки токов, открыв файл current.txt (через выпадающее меню «FILE OPEN»). Его первые три строчки должны быть такими:

```
2  
-1.0  
1.0
```

Казалось бы, данная установка имеет один из питающих электродов на бесконечности, но мы всё равно должны задать токи на обоих электродах, так как в так называемых «трёх-электродных» установках оба питающих электрода фактически существуют и значит на обоих электродах должен быть задан ток.

Чтобы понять, как заполнить файл conductivity.txt, нужно вспомнить в каком порядке мы перечисляли объекты-среды («tlo») во входном файле для генератора сеток NetGen и понимать, что NetGen нумерует объекты-среды, начиная с 1. А порядок был таков:

1. верхний слой (фиктивный),
2. верхний слой (фиктивный),
3. верхний слой,
4. нижний слой,

Таким образом, файл conductivity.txt будет содержать 5 сред (4 нужных нам сред плюс «нулевая» среда):

```
5  
0.0  
0.1  
0.1  
0.1  
0.01
```

В «нулевой» среде задаем произвольное значение (так как её нет), пусть это будет 0.0. Верхний слой пусть имеет значение электропроводности равное 0.1 См/м, а нижний слой 0.01 См/м.

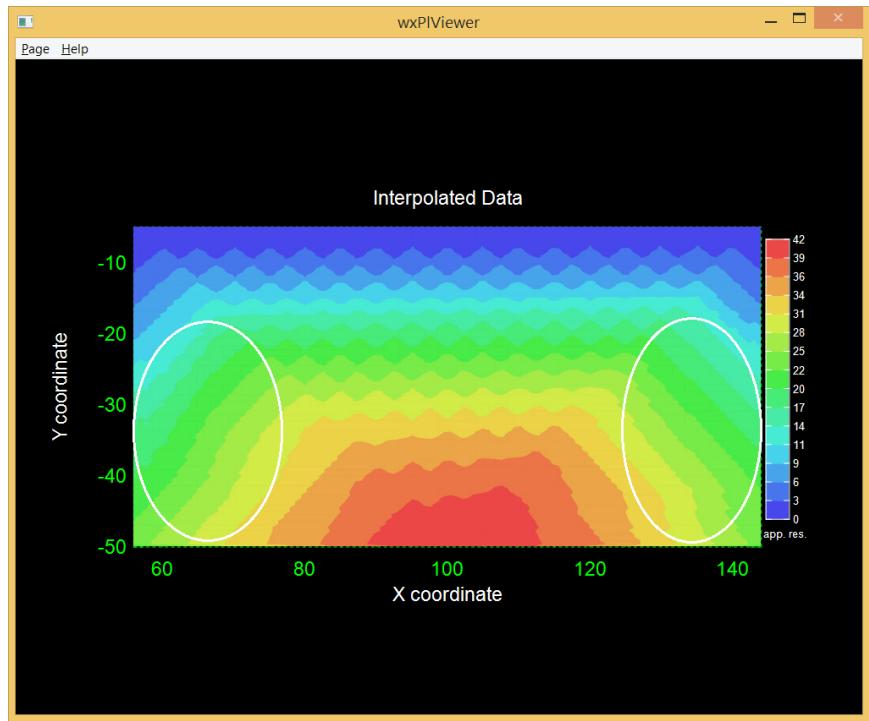
Теперь мы готовы к генерации матрицы системы линейных алгебраических уравнений. Нажимаем кнопку «Start PREPROCESSOR (assembling matrix)» для этого.

Матрица сгенерирована, можно приступать к решению прямой задачи, смотреть портрет матрицы для такой простой задачи совсем необязательно. Пусть мы хотим решать сразу все задачи сразу на всех профилях. Вводим в поле «Y-coord. of profile:» значение «x» без кавычек, в качестве первой задачи — 1, а в качестве последней задачи — произвольное большое число, например, 100000. Нажимаем кнопку «START DIRECT solver». Процесс решения прямых задач пошёл.

После окончания решения прямых задач (на это уйдёт несколько минут), нажимаем кнопку компоновки решений «bring together», после чего все файлы с решениями окажутся в директории «my_simple_test» (так мы назвали нашу задачу), которая в свою очередь расположена в директории программы DiInSo. В папке с сеткой для прямой задачи появятся новые файлы — для решения задачи инверсии в 2D и 3D случаях. Перейдём к этим задачам...

Давайте решим двумерную задачу инверсии, например, на шестом профиле, для чего выберем файл profile6.bert, нажав кнопку «DATA FILE:». Не забываем выбрать тип задачи «2D». Другие настройки выставим позднее.

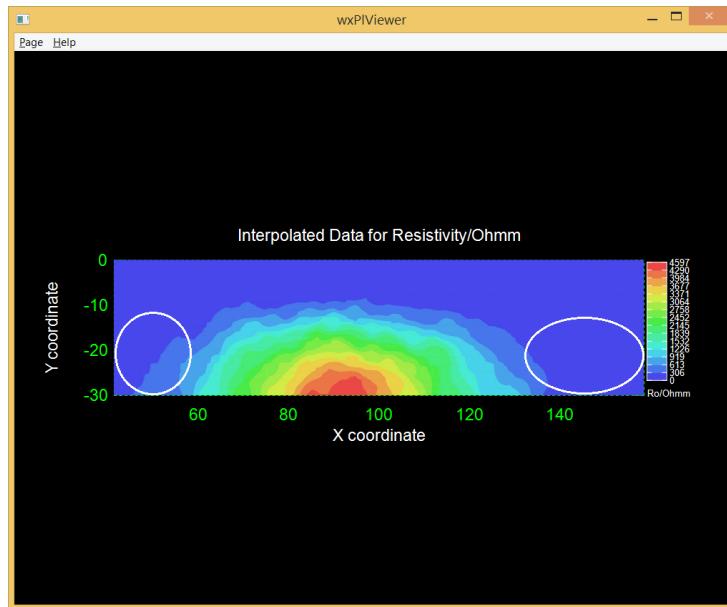
Можно посмотреть кажущиеся сопротивления, нажав кнопку «vis. app. res.»:



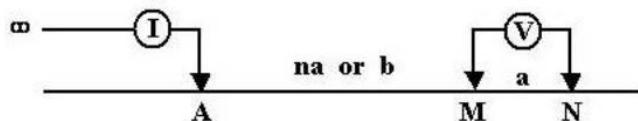
Выделенные овалами искривления связаны с эффектом нехватки данных, то есть в этих точках фактически измерений нет, а данные на картинке получены там путём интерполяции.

Попробуем решить задачу инверсии в двух вариантах — для малого значения параметра регуляризации и для большого значения параметра регуляризации.

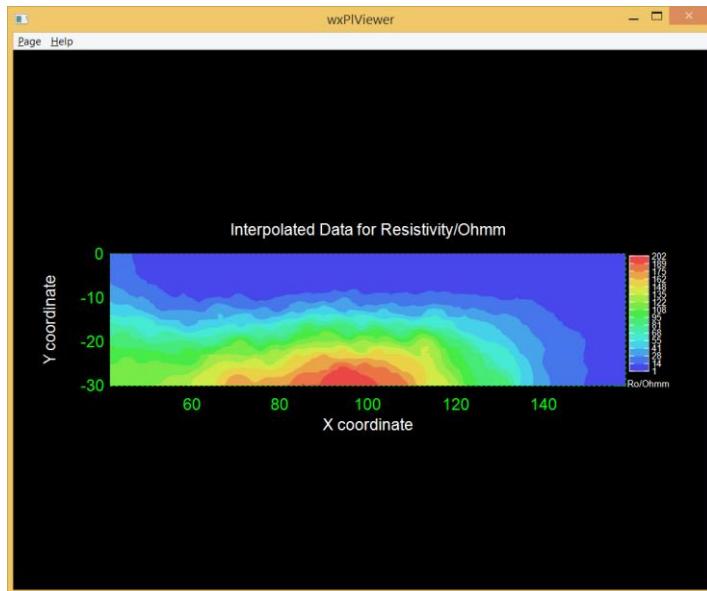
Сначала выставляем такие параметры: «size of cells» — 0.25, «mesh growing» — 34.0 (нам нужна сетка средней мелкости), «model depth» — 30 (глубже 30 метров нас результаты не интересуют), «recalc jacobian?» — 1, «gl reg strength» — 1. Жмём кнопку «INVERSION» и видим в окне консоли, что программа совершила 6 итераций, сходимость при этом более-менее нормальная. Смотрим результаты инверсии, нажав кнопку «vis. inv.»:



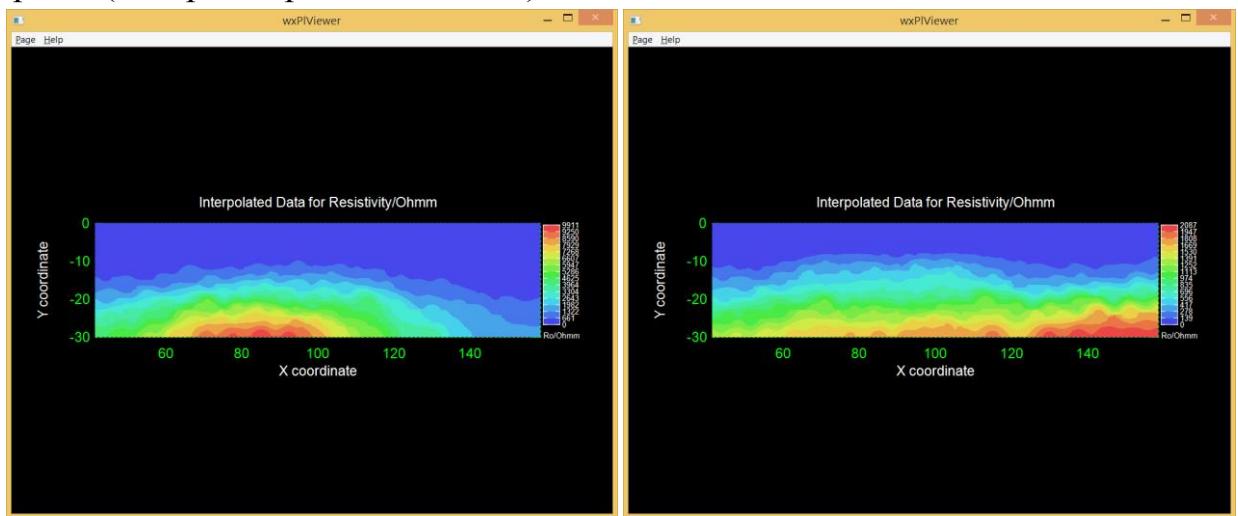
Верхний слой, мощностью 10 метров, выделяется достаточно отчётливо. Нижний слой выделяется уже не так хорошо, а по бокам нижнего слоя не хватает данных (выделено овалами), причем в данном случае справа нехватка данных чувствуется более остро. Почему так? Для понимания вспомним, как выглядит установка поль-диполь:



Этой установке явно «не хватает» питающего электрода справа, отсюда и большая область некорректных данных справа. Также заметим, что при малом значении параметра регуляризации часто получаются довольно грубые значения сопротивлений областей. Чтобы получить более точные значения сопротивлений, попробуем увеличить параметр регуляризации. Не меняя остальные настройки, выставим «gl reg strength» равным 50. Нажав кнопку «INVERSION», получим решение за 4 итерации при хорошей сходимости. Смотрим результаты инверсии, нажав кнопку «vis. inv.»:



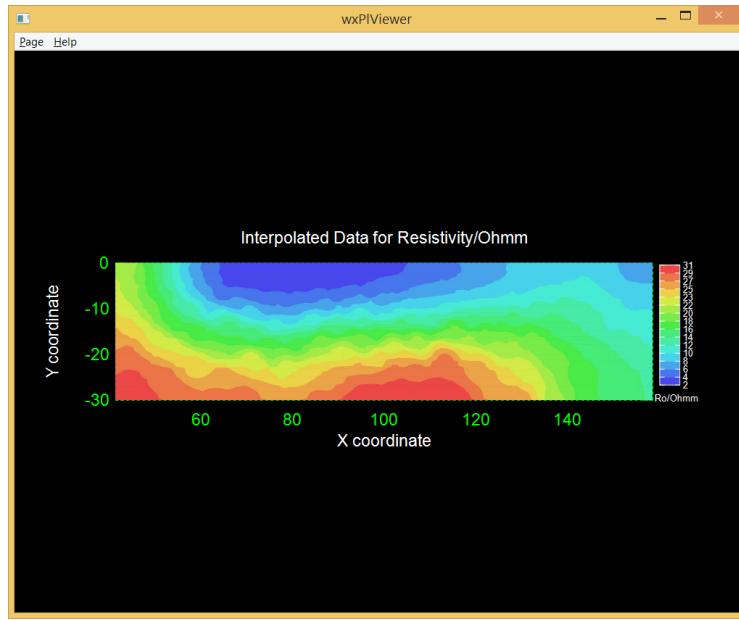
Результат получился более адекватный во всех смыслах этого слова. Верхний слой выделяется достаточно чётко, имеется переход между верхним и нижним слоями, избавиться от которого всегда сложно. При этом значения сопротивлений намного ближе к реальным. Можно попытаться устранить искажения слоистой среды, предложив алгоритму инверсии поиск вертикальных границ, уменьшив тем самым размытость результатов. Напомним, что для этого настраивается параметр «vert constraints». Зададим два значения для «vert constraints» — 0.1 (слева) и 0.01 (справа), то есть «размытые вертикальные границы» среды и «четкие вертикальные границы» среды (которых в реальности нет):



Как мы видим, такой подход действительно позволяет устраниć неточности геометрии, которые создаёт установка поль-диполь, но с потерей точности значений сопротивлений. Однако заметим, что иногда лучше просто «обрезать» часть области, где не хватает измеренных данных, при

демонстрации результатов инверсии. Если бы внутри области присутствовали какие-то объекты, расположенные внутри той части среды, где не хватает измеренных данных, они бы не были обнаружены, но визуально «необрезанная область» может создавать мнимое ощущение правильности результатов инверсии во всей области.

Наконец, попробуем очень большое значение параметра регуляризации. Не меняя остальные настройки, выставим «gl reg strength» равным 500. Нажав кнопку «INVERSION», получим решение за 3 итерации при не очень хорошей сходимости. Смотрим результаты инверсии, нажав кнопку «vis. inv.»:

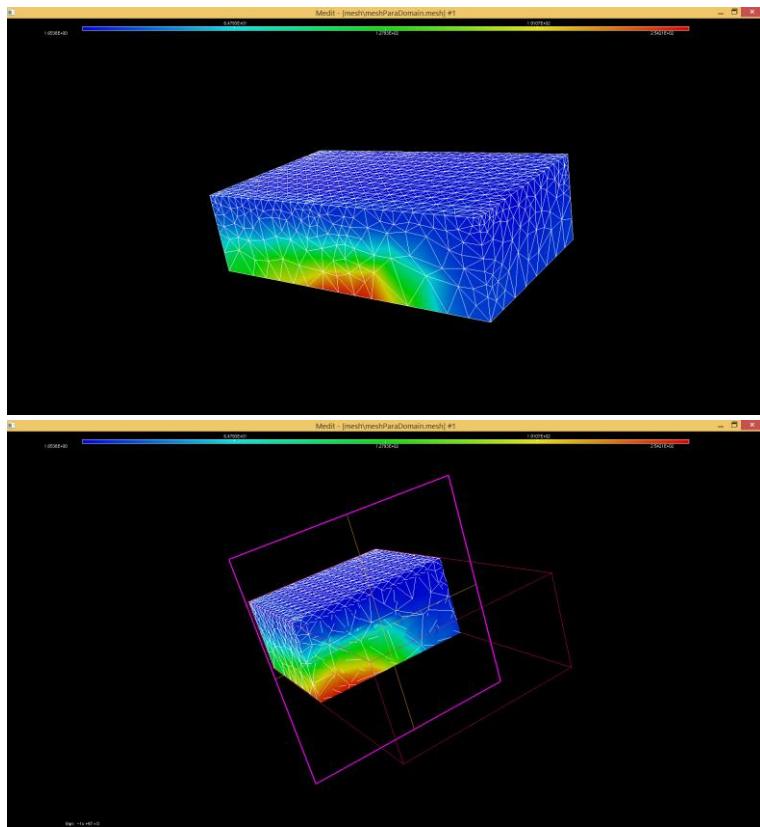


Такой результат нас вряд ли может удовлетворить. При нужной степени фантазии можно понять, что мы имеем дело с двумя слоями, однако значения сопротивлений недостаточно точны, да и сами слои неприятно искривлены и создают ложное ощущение НЕ горизонтальных границ между слоями.

Таким образом, мы показали, насколько важным является правильный подбор значения параметра регуляризации.

Решим аналогичную трёхмерную задачу, выбрав в качестве параметра регуляризации то значение, которое нам подошло больше всего в двумерном случае — 50. Выставляем тип инверсии «3D» и параметры «quality» — 33, «mesh grow» — 1.2 (нам нужна относительно хорошая сетка), «recalc jacobian?» — 1, «regular strength» — 50. Глубину исследований также ограничим 30 метрами, выставив «model depth» — 30. Выбираем созданный файл reverse_data.bert, нажав кнопку «DATA FILE:». Проводим инверсию, нажав кнопку «INVERSION». Решение сходится за 4 итерации при

относительно хорошей сходимости. Смотрим результаты инверсии, нажав кнопку «vis. inv.»:

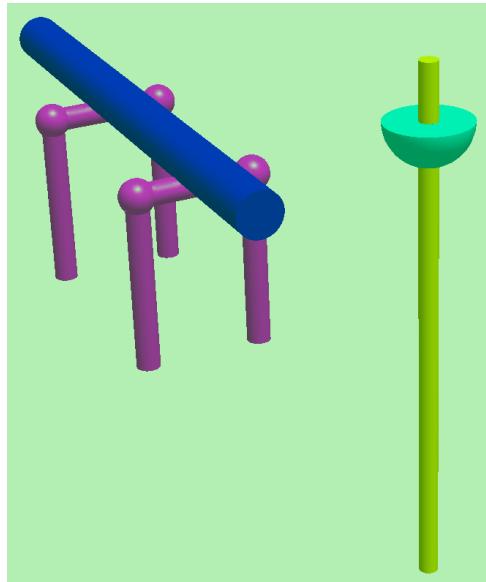


Здесь мы наблюдаем интересный эффект — «нехватка» токового электрода справа в установке поль-диполь в трёхмерном случае «обрезает» нижний слой не только справа по оси X, но и справа по оси Y. Вообще говоря, это довольно известный недостаток установки поль-диполь. Чтобы «побороть» его часто используют «прямой» и «обратный» ход установки, то есть сначала пускают установку поль-диполь слева направо, у которой питающий электрод находится слева, а затем установку поль-диполь справа налево, у которой питающий электрод находится справа. Таким образом получается более целостная картина с меньшим размером области, где не хватает данных.

Пример решения прямой задачи электротомографии с последующим решением задачи инверсии со сложными объектами с помощью генератора сеток NetGen.

Смоделируем электротомографическое исследование в области, в которой присутствуют металлические конструкции, создающие помехи. Пусть этими конструкциями будут две П-образные металлические опоры, вбитые в землю, на которых лежит металлическая труба, а также заглубленная скважина с металлической обсадкой, в приповерхностной зоне

которой наблюдается зона растяжения (область с более высокой температурой). Как известно, при изменении температуры среды могут изменяться и её физические свойства, в том числе значение электропроводности, поэтому зоны растяжения мы будем выделять как отдельный объект. Схематично описанная конструкция имеет вид:



Для простоты будем считать, что все металлические трубы НЕ являются полыми, то есть полностью состоят из металла. Поскольку металлические конструкции находятся как над землёй, так и под землёй, нам необходимо будет задавать такую среду, как воздух, а электроды будем заглублять в землю на некоторую небольшую глубину (2 метра), чтобы уменьшить влияние на численный результат сильно контрастной среды — воздуха. Также будем считать, что ниже границы «воздух-земля» мы имеем двухслойную среду, верхний слой которой имеет мощность 22 метра или 20 метров ниже электродов. Пусть электроды выстроены в 11 линий (то есть 11 профилей), каждая из линий состоит из 21 электрода, шаг между линиями (профилями) — 5 метров. В качестве установки будем рассматривать установку Веннера-Альфа, которая не имеет тех недостатков, которые есть у установки поль-диполь, описанные в прошлом примере (хотя это не значит, что данная установка лучше, просто каждая из установок подбирается под определённую задачу).

Первым делом необходимо задать область со всеми объектами-средами и электродами, чтобы затем построить тетраэдральную сетку. Делать мы это будем с помощью генератора сеток NetGen, но вы можете выбрать и другой генератор сеток — Gmsh, GiD или Salome. Файл с геометрией в NetGen задаётся с помощью примитивов. В первой строке задаём ключевое слово:

`algebraic3d`

Далее задаём 6 плоскостей, которые ограничивают нашу область (они должны быть достаточно удалены от электродов — источников тока):

```
# Нижняя граница
solid planev1= plane(    100.0,      100.0,     -200.0;0,0,-1) -bc=76;
# Передняя граница
solid planev2= plane(    100.0,     -50.0,       50.0;0,-1,0) -bc=76;
# Левая граница
solid planev3= plane(   -50.0,      100.0,      50.0;-1,0,0) -bc=76;
# Верхняя граница
solid planev4= plane(    100.0,      100.0,      100.0;0,0,1) -bc=76;
# Задняя граница
solid planev5= plane(    100.0,      200.0,      50.0;0,1,0) -bc=76;
# Правая граница
solid planev6= plane(    250.0,      100.0,      50.0;1,0,0) -bc=76;
```

Поскольку это внешние границы области, которые все сильно удалены от электродов, то на них необходимо задать нулевые граничные условия Дирихле. Делается это с помощью параметра «`-bc=76`», то есть мы будем считать, что нулевые граничные условия Дирихле задаются с помощью значения «76». Объединяем плоскости в единое целое и получаем область в форме «бака»:

```
solid bak= planev1 and planev2 and planev3 and planev4 and planev5 and planev6;
```

Далее все объекты будут создаваться внутри «бака». Задаём части нижней П-образной опоры с помощью трёх цилиндров (пока это НЕ сама П-образная опора, это лишь её части):

```
solid truba1= cylinder(  67.5,    62.5,      6.0;    67.5,    62.5,     -10.0;    1.0)
and plane( 67.5,    62.5,      6.0;0,0,0,1)
and plane( 67.5,    62.5,     -10.0;0,0,-1);
solid truba2= cylinder(  77.5,    62.5,      6.0;    77.5,    62.5,     -10.0;    1.0)
and plane( 77.5,    62.5,      6.0;0,0,0,1)
and plane( 77.5,    62.5,     -10.0;0,0,-1);
solid truba3= cylinder(  67.5,    62.5,      5.0;    77.5,    62.5,      5.0;    1.0)
and plane( 67.5,    62.5,      5.0;-1,0,0)
and plane( 77.5,    62.5,      5.0;1,0,0);
```

На двух углах П-образной опоры задаём два шара:

```
solid shar1= sphere(  67.5,    62.5,      5.0;    1.5);
solid shar2= sphere(  77.5,    62.5,      5.0;    1.5);
```

Шары нужны для того, чтобы победить известную ошибку генератора NetGen, которая на момент написания данного мануала всё ещё не исправлена. Состоит она в том, что пересечённые под прямым углом цилиндры нарушают работу алгоритма построения сетки и процесс создания сетки уходит в бесконечный цикл.

Повторяем процедуру для верхней П-образной опоры:

```
solid truba4= cylinder(  67.5,    82.5,      6.0;    67.5,    82.5,     -10.0;    1.0)
and plane( 67.5,    82.5,      6.0;0,0,0,1)
and plane( 67.5,    82.5,     -10.0;0,0,-1);
solid truba5= cylinder(  77.5,    82.5,      6.0;    77.5,    82.5,     -10.0;    1.0)
and plane( 77.5,    82.5,      6.0;0,0,0,1)
and plane( 77.5,    82.5,     -10.0;0,0,-1);
solid truba6= cylinder(  67.5,    82.5,      5.0;    77.5,    82.5,      5.0;    1.0)
and plane( 67.5,    82.5,      5.0;-1,0,0)
and plane( 77.5,    82.5,      5.0;1,0,0);
solid shar3= sphere(  67.5,    82.5,      5.0;    1.5);
solid shar4= sphere(  77.5,    82.5,      5.0;    1.5);
```

С помощью цилиндра задаём металлическую трубу, лежащую на опорах:

```
solid bigtruba= cylinder( 72.5, 50.0, 7.0; 72.5, 100.0, 7.0; 2.0)
and plane( 72.5, 50.0, 7.0;0,-1,0)
and plane( 72.5, 100.0, 7.0;0,1,0);
```

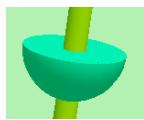
С помощью полу-шара задаём зону растяжения:

```
solid polusharl= sphere( 102.5, 77.5, 0.0; 4.5) and plane( 102.5,
77.5, 0.0;0,0,1);
```

С помощью цилиндра задаём скважину:

```
solid trubafull1 = cylinder( 102.5, 77.5, -50.0; 102.5, 77.5, 6.0; 1.0)
and plane( 102.5, 77.5, 6.0;0,0,1)
and plane( 102.5, 77.5, -50.0;0,0,-1);
```

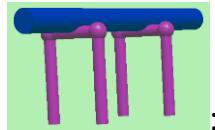
Далее нужно сообщить алгоритму какие объекты пересекаются, а какие объединяются, иначе мы получим несколько сеток внутри одних и тех же областей. Начнём с самого простого — зона растяжения НЕ включает



область скважины :

```
solid zatepl1 = polusharl and not trubafull1;
```

Каждая из П-образных опор объединяет три трубы опоры, два шара на углах П-образной опоры и исключает трубу, которая лежит сверху на опорах



```
solid oporaniz= (truba1 or truba2 or truba3 or shar1 or shar2) and not bigtruba;
solid oporaverh= (truba4 or truba5 or truba6 or shar3 or shar4) and not bigtruba;
```

Две П-образные опоры и труба, лежащая на них — это единый металлический объект с точки зрения нашей модели, поэтому мы их объединяем (этот объект не будет использован в дальнейшем, поэтому этот шаг можно пропустить, но для понимания структуры объекта он оставлен):

```
solid vsetruba= oporaniz or oporaverh or bigtruba;
```

Как мы помним, наш «бак» имеет несколько слоёв — воздух, верхний слой земли и нижний слой земли. Однако, мы зададим ещё один слой, который находится внутри верхнего слоя земли. На границе этого слоя будут лежать заглубленные на 2 метра электроды. Таким образом, суммарно имеем четыре слоя, границы между которыми разделяют три плоскости:

```
solid cutplane1= plane( 100.0, 100.0, 2.0;0,0,1);
solid cutplane2= plane( 100.0, 100.0, 0.0;0,0,1);
solid cutplane3= plane( 100.0, 100.0, -20.0;0,0,1);
```

Теперь выделяем каждый из четырёх слоев, помня, что объекты внутри слоя необходимо исключить из слоя:

```
# Воздух
solid slice1= bak and not cutplane1 and not vsetruba and not trubafull1;
# Слой земли, созданный для электродов
solid slice2= bak and cutplane1 and not cutplane2 and not vsetruba and not zatepl1 and not
trubafull1;
# Верхний слой земли
solid slice3= bak and cutplane2 and not cutplane3 and not vsetruba and not zatepl1 and not
trubafull1;
# Нижний слой земли
```

```
solid slice4= bak and cutplane3 and not trubafull1;
```

Теперь перечислим все объекты-среды («tlo»), выделив отдельно материалы объектов-сред («-material»), в том числе и цветом («-col») с прозрачностью («-transparent») и ограничив максимальные размеры рёбер тетраэдров внутри объектов-сред («-maxh»):

```
tlo slice1 -col=[0,1,0] -transparent -material=slice1 -maxh= 20.0;
tlo slice2 -col=[0,1,0] -transparent -material=slice2 -maxh= 1.5;
tlo slice3 -col=[0,1,0] -transparent -material=slice3 -maxh= 10.0;
tlo slice4 -col=[0,1,0] -transparent -material=slice4 -maxh= 20.0;
tlo oporaniz -col=[1,0,1] -material=oporaniz -maxh= 1.0;
tlo oporaverh -col=[1,0,1] -material=oporaverh -maxh= 1.0;
tlo bigtruba -col=[0,0,1] -material=bigtruba -maxh= 2.0;
tlo zatepl1 -col=[0,1,1] -material=polushar1 -maxh= 3.0;
tlo trubafull1 -col=[1,1,0] -material=trubafull1 -maxh= 1.0;
```

Заметим, что мы вновь «разделили» наши П-образные опоры и трубу, лежащую на них, в отдельные объекты. Сделано это было для того, чтобы задать разные правила генерации сетки для подобъектов единого объекта (разные максимальные размеры рёбер тетраэдров). Остался последний шаг — перечислить все точки, где находятся электроды:

```
point( 50.0,      50.0,      0.0);
point( 55.0,      50.0,      0.0);
point( 60.0,      50.0,      0.0);
point( 65.0,      50.0,      0.0);
point( 70.0,      50.0,      0.0);
point( 75.0,      50.0,      0.0);
point( 80.0,      50.0,      0.0);
point( 85.0,      50.0,      0.0);
point( 90.0,      50.0,      0.0);
point( 95.0,      50.0,      0.0);
point(100.0,      50.0,      0.0);
point(105.0,      50.0,      0.0);
point(110.0,      50.0,      0.0);
point(115.0,      50.0,      0.0);
point(120.0,      50.0,      0.0);
point(125.0,      50.0,      0.0);
point(130.0,      50.0,      0.0);
point(135.0,      50.0,      0.0);
point(140.0,      50.0,      0.0);
point(145.0,      50.0,      0.0);
point(150.0,      50.0,      0.0);
point( 50.0,      55.0,      0.0);
point( 55.0,      55.0,      0.0);
point( 60.0,      55.0,      0.0);
point( 65.0,      55.0,      0.0);
point( 70.0,      55.0,      0.0);
point( 75.0,      55.0,      0.0);
point( 80.0,      55.0,      0.0);
point( 85.0,      55.0,      0.0);
point( 90.0,      55.0,      0.0);
point( 95.0,      55.0,      0.0);
point(100.0,      55.0,      0.0);
point(105.0,      55.0,      0.0);
point(110.0,      55.0,      0.0);
point(115.0,      55.0,      0.0);
point(120.0,      55.0,      0.0);
point(125.0,      55.0,      0.0);
point(130.0,      55.0,      0.0);
point(135.0,      55.0,      0.0);
point(140.0,      55.0,      0.0);
point(145.0,      55.0,      0.0);
point(150.0,      55.0,      0.0);
point( 50.0,      60.0,      0.0);
point( 55.0,      60.0,      0.0);
point( 60.0,      60.0,      0.0);
point( 65.0,      60.0,      0.0);
point( 70.0,      60.0,      0.0);
point( 75.0,      60.0,      0.0);
point( 80.0,      60.0,      0.0);
point( 85.0,      60.0,      0.0);
point( 90.0,      60.0,      0.0);
point( 95.0,      60.0,      0.0);
point(100.0,      60.0,      0.0);
point(105.0,      60.0,      0.0);
point(110.0,      60.0,      0.0);
point(115.0,      60.0,      0.0);
point(120.0,      60.0,      0.0);
```

```

point( 125.0,      60.0,      0.0);
point( 130.0,      60.0,      0.0);
point( 135.0,      60.0,      0.0);
point( 140.0,      60.0,      0.0);
point( 145.0,      60.0,      0.0);
point( 150.0,      60.0,      0.0);
point( 50.0,       65.0,      0.0);
point( 55.0,       65.0,      0.0);
point( 60.0,       65.0,      0.0);
point( 65.0,       65.0,      0.0);
point( 70.0,       65.0,      0.0);
point( 75.0,       65.0,      0.0);
point( 80.0,       65.0,      0.0);
point( 85.0,       65.0,      0.0);
point( 90.0,       65.0,      0.0);
point( 95.0,       65.0,      0.0);
point( 100.0,      65.0,      0.0);
point( 105.0,      65.0,      0.0);
point( 110.0,      65.0,      0.0);
point( 115.0,      65.0,      0.0);
point( 120.0,      65.0,      0.0);
point( 125.0,      65.0,      0.0);
point( 130.0,      65.0,      0.0);
point( 135.0,      65.0,      0.0);
point( 140.0,      65.0,      0.0);
point( 145.0,      65.0,      0.0);
point( 150.0,      65.0,      0.0);
point( 50.0,       70.0,      0.0);
point( 55.0,       70.0,      0.0);
point( 60.0,       70.0,      0.0);
point( 65.0,       70.0,      0.0);
point( 70.0,       70.0,      0.0);
point( 75.0,       70.0,      0.0);
point( 80.0,       70.0,      0.0);
point( 85.0,       70.0,      0.0);
point( 90.0,       70.0,      0.0);
point( 95.0,       70.0,      0.0);
point( 100.0,      70.0,      0.0);
point( 105.0,      70.0,      0.0);
point( 110.0,      70.0,      0.0);
point( 115.0,      70.0,      0.0);
point( 120.0,      70.0,      0.0);
point( 125.0,      70.0,      0.0);
point( 130.0,      70.0,      0.0);
point( 135.0,      70.0,      0.0);
point( 140.0,      70.0,      0.0);
point( 145.0,      70.0,      0.0);
point( 150.0,      70.0,      0.0);
point( 50.0,       75.0,      0.0);
point( 55.0,       75.0,      0.0);
point( 60.0,       75.0,      0.0);
point( 65.0,       75.0,      0.0);
point( 70.0,       75.0,      0.0);
point( 75.0,       75.0,      0.0);
point( 80.0,       75.0,      0.0);
point( 85.0,       75.0,      0.0);
point( 90.0,       75.0,      0.0);
point( 95.0,       75.0,      0.0);
point( 100.0,      75.0,      0.0);
point( 105.0,      75.0,      0.0);
point( 110.0,      75.0,      0.0);
point( 115.0,      75.0,      0.0);
point( 120.0,      75.0,      0.0);
point( 125.0,      75.0,      0.0);
point( 130.0,      75.0,      0.0);
point( 135.0,      75.0,      0.0);
point( 140.0,      75.0,      0.0);
point( 145.0,      75.0,      0.0);
point( 150.0,      75.0,      0.0);
point( 50.0,       80.0,      0.0);
point( 55.0,       80.0,      0.0);
point( 60.0,       80.0,      0.0);
point( 65.0,       80.0,      0.0);
point( 70.0,       80.0,      0.0);
point( 75.0,       80.0,      0.0);
point( 80.0,       80.0,      0.0);
point( 85.0,       80.0,      0.0);
point( 90.0,       80.0,      0.0);
point( 95.0,       80.0,      0.0);
point( 100.0,      80.0,      0.0);
point( 105.0,      80.0,      0.0);
point( 110.0,      80.0,      0.0);
point( 115.0,      80.0,      0.0);
point( 120.0,      80.0,      0.0);
point( 125.0,      80.0,      0.0);
point( 130.0,      80.0,      0.0);
point( 135.0,      80.0,      0.0);
point( 140.0,      80.0,      0.0);
point( 145.0,      80.0,      0.0);
point( 150.0,      80.0,      0.0);
point( 50.0,       85.0,      0.0);
point( 55.0,       85.0,      0.0);

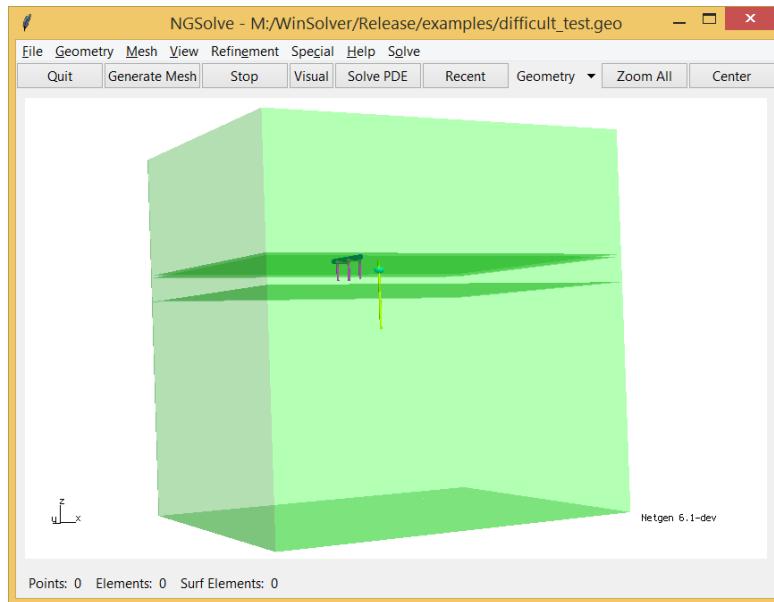
```

```

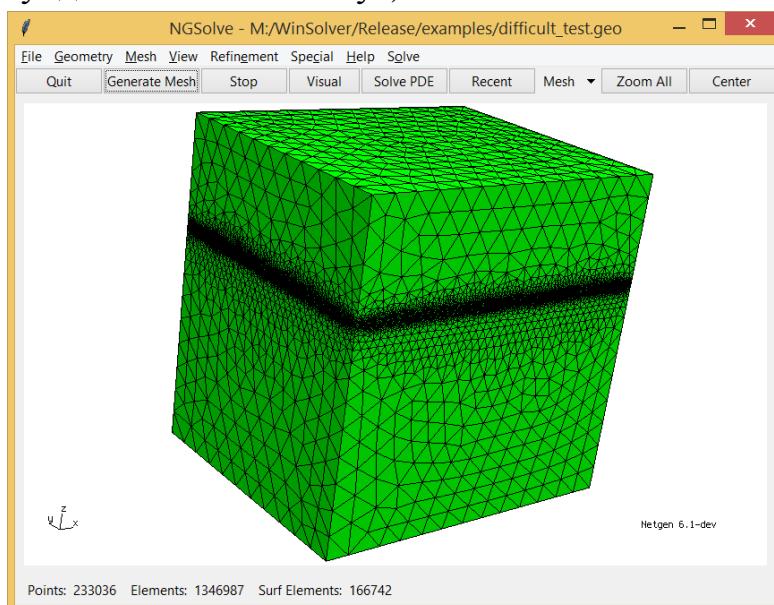
point(    60.0,     85.0,      0.0);
point(    65.0,     85.0,      0.0);
point(    70.0,     85.0,      0.0);
point(    75.0,     85.0,      0.0);
point(    80.0,     85.0,      0.0);
point(    85.0,     85.0,      0.0);
point(    90.0,     85.0,      0.0);
point(    95.0,     85.0,      0.0);
point(   100.0,     85.0,      0.0);
point(   105.0,     85.0,      0.0);
point(   110.0,     85.0,      0.0);
point(   115.0,     85.0,      0.0);
point(   120.0,     85.0,      0.0);
point(   125.0,     85.0,      0.0);
point(   130.0,     85.0,      0.0);
point(   135.0,     85.0,      0.0);
point(   140.0,     85.0,      0.0);
point(   145.0,     85.0,      0.0);
point(   150.0,     85.0,      0.0);
point(    50.0,     90.0,      0.0);
point(    55.0,     90.0,      0.0);
point(    60.0,     90.0,      0.0);
point(    65.0,     90.0,      0.0);
point(    70.0,     90.0,      0.0);
point(    75.0,     90.0,      0.0);
point(    80.0,     90.0,      0.0);
point(    85.0,     90.0,      0.0);
point(    90.0,     90.0,      0.0);
point(    95.0,     90.0,      0.0);
point(   100.0,     90.0,      0.0);
point(   105.0,     90.0,      0.0);
point(   110.0,     90.0,      0.0);
point(   115.0,     90.0,      0.0);
point(   120.0,     90.0,      0.0);
point(   125.0,     90.0,      0.0);
point(   130.0,     90.0,      0.0);
point(   135.0,     90.0,      0.0);
point(   140.0,     90.0,      0.0);
point(   145.0,     90.0,      0.0);
point(   150.0,     90.0,      0.0);
point(    50.0,     95.0,      0.0);
point(    55.0,     95.0,      0.0);
point(    60.0,     95.0,      0.0);
point(    65.0,     95.0,      0.0);
point(    70.0,     95.0,      0.0);
point(    75.0,     95.0,      0.0);
point(    80.0,     95.0,      0.0);
point(    85.0,     95.0,      0.0);
point(    90.0,     95.0,      0.0);
point(    95.0,     95.0,      0.0);
point(   100.0,     95.0,      0.0);
point(   105.0,     95.0,      0.0);
point(   110.0,     95.0,      0.0);
point(   115.0,     95.0,      0.0);
point(   120.0,     95.0,      0.0);
point(   125.0,     95.0,      0.0);
point(   130.0,     95.0,      0.0);
point(   135.0,     95.0,      0.0);
point(   140.0,     95.0,      0.0);
point(   145.0,     95.0,      0.0);
point(   150.0,     95.0,      0.0);
point(    50.0,    100.0,      0.0);
point(    55.0,    100.0,      0.0);
point(    60.0,    100.0,      0.0);
point(    65.0,    100.0,      0.0);
point(    70.0,    100.0,      0.0);
point(    75.0,    100.0,      0.0);
point(    80.0,    100.0,      0.0);
point(    85.0,    100.0,      0.0);
point(    90.0,    100.0,      0.0);
point(    95.0,    100.0,      0.0);
point(   100.0,    100.0,      0.0);
point(   105.0,    100.0,      0.0);
point(   110.0,    100.0,      0.0);
point(   115.0,    100.0,      0.0);
point(   120.0,    100.0,      0.0);
point(   125.0,    100.0,      0.0);
point(   130.0,    100.0,      0.0);
point(   135.0,    100.0,      0.0);
point(   140.0,    100.0,      0.0);
point(   145.0,    100.0,      0.0);
point(   150.0,    100.0,      0.0);

```

Полученный файл (есть в папке «examples» программы DiInSo, называется difficult_test.geo) можно открыть в генераторе сеток NetGen:



Осталось только создать сетку, нажав кнопку «Generate Mesh» (на создание сетки уйдёт несколько минут):



Сохраняем сетку в файл difficult_test.vol через выпадающее меню File -> Save Mesh...

Открываем программу DiInSo. В поле «Name of problem (!):» прописываем имя нашей задачи. Пусть это будет, например, «my_difficult_test» без кавычек. Нажимаем кнопку «select file with mesh:» и выбираем полученный файл с сеткой difficult_test.vol. Настраиваем дополнительные входные файлы через выпадающее меню «FILE OPEN». Первые три строчки файла current.txt должны иметь вид:

```
2
-1.0
1.0
```

Чтобы понять, как задавать файл conductivity.txt, нужно вспомнить в каком порядке мы перечисляли объекты-среды («tlo») во входном файле для генератора сеток NetGen и понимать, что NetGen нумерует объекты-среды, начиная с 1. А порядок был таков:

1. воздух,
2. слой земли, созданный для электродов,
3. верхний слой земли,
4. нижний слой земли,
5. нижняя П-образная опора,
6. верхняя П-образная опора,
7. труба, лежащая на П-образных опорах,
8. зона растяжения,
9. скважина.

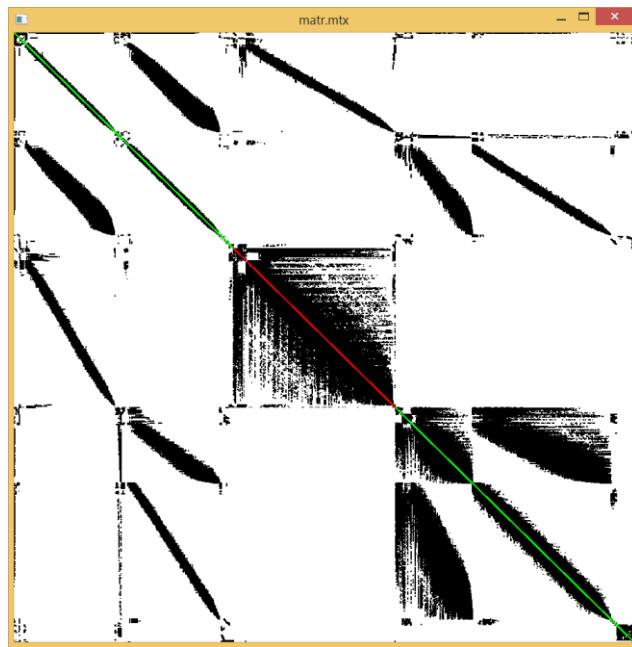
Таким образом, файл conductivity.txt будет содержать 10 сред (9 нужных нам сред плюс «нулевая» среда):

```
10
0.0
1.0E-12
0.001
0.001
0.1
7690000.0
7690000.0
7690000.0
0.01
7690000.0
```

В «нулевой» среде задаём произвольное значение электропроводности (так как этой среды нет), пусть это будет 0.0. Воздух имеет очень малое значение электропроводности. Просто задать 0.0 мы не имеем права (см. систему уравнений Максвелла), поэтому зададим очень малое число: 1.0E-12 См/м. Затем идут два слоя, которые фактически совпадают по своим значениям электропроводности, так как один из этих слоёв был задан для электродов. Пусть значения электропроводности для них равно 0.001 См/м. Следующая объект-среда — нижний слой земли с электропроводностью 0.1 См/м. Далее идут три металлических объекта — две П-образные опоры и труба на них. Электропроводность металла равна 7690000.0 См/м. Затем идёт зона растяжения. Пусть её электропроводность составляет 0.01 См/м. Далее идёт металлическая скважина, поэтому электропроводности здесь вновь 7690000.0 См/м.

Наконец, создаём новый файл elec_profiles.txt, нажав кнопку  . Выбрав нужную установку (в нашем случае это Веннер-Альфа), вспоминаем, как у нас располагаются электроды и сколько их: X0 = 50, Y0 = 50, Z0 = 0, Number of electrodes in one profile — 21, Number of profiles — 11, X_step — 5,

Y_step — 5. Жмём кнопку «Generate» и новый файл elec_profiles.txt будет создан. Ради интереса, можно открыть его (через выпадающее меню «FILE OPEN») и узнать, что общее число задач, которое нам придётся решать, равняется 693. Убеждаемся, что нулевые краевые условия Дирихле заданы тем числом, которое нам нужно, а именно 76. Для этого открываем файл null.txt (через выпадающее меню «FILE OPEN»). Теперь мы готовы к генерации матрицы системы линейных алгебраических уравнений. Нажимаем кнопку «Start PREPROCESSOR (assembling matrix)». После того как процесс создания матрицы завершится, можно приступать к решению прямых задач, но сначала посмотрим портрет матрицы, нажав на кнопку «matrix portrait»:



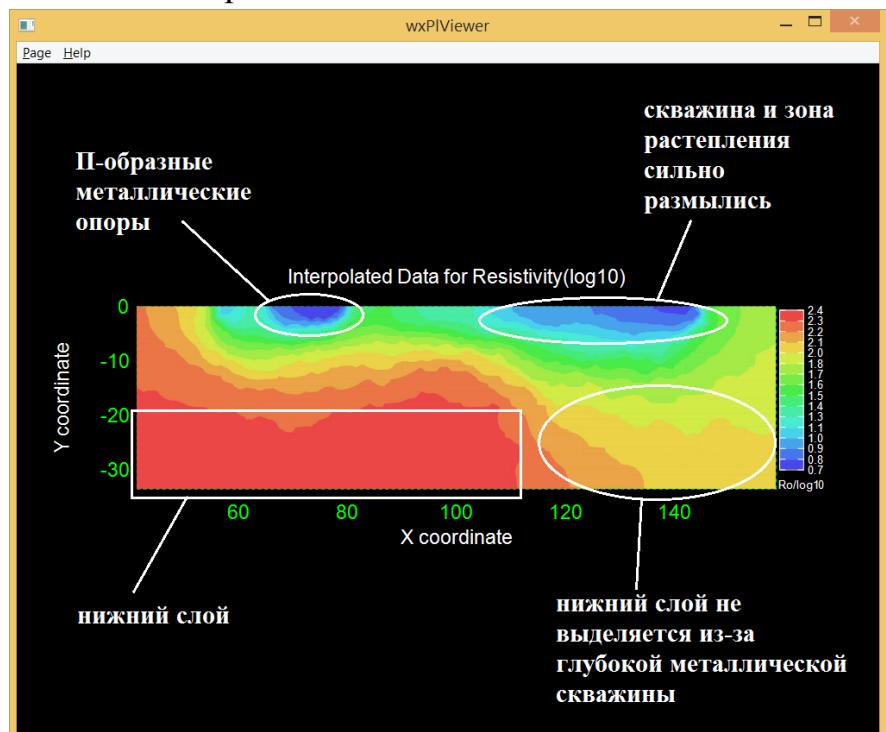
Итак, нумерация узлов сетки в NetGen оставляет желать лучшего. Но и ничего ужасного в таком портрете нет. Куда страшнее наличие близких к нулю элементов на диагонали (они обозначены красным). Их наличие связано с тем, что имеются сильно контрастные среды. Однако, как бы плохо не выглядел портрет матрицы, и такая задача должна быть решена.

Итак, пусть мы хотим решать сразу все задачи сразу на всех профилях. Вводим в поле «Y-coord. of profile:» значение «x» без кавычек, в качестве первой задачи — 1, а в качестве последней задачи — произвольное большое число, например, 100000. Нажимаем кнопку «START DIRECT solver». Процесс решения прямых задач пошёл.

После окончания решения прямых задач (на это уйдёт несколько минут), нажимаем кнопку компоновки решений «bring together», после чего все файлы с решениями окажутся в директории «my_difficult_test» (так мы назвали нашу задачу), которая в свою очередь расположена в директории

программы DiInSo. В папке с сеткой для прямой задачи появятся новые файлы — для решения задачи инверсии в 2D и 3D случаях. Перейдём к этим задачам...

Давайте решим двумерную задачу инверсии, например, на восьмом профиле, для чего выберем файл profile8.bert, нажав кнопку «DATA FILE:». Не забываем выбрать тип задачи «2D». Другие настройки выставим следующим образом: «size of cells» — 0.25, «mesh growing» — 35.0 (нам нужна подробная сетка), «recalc jacobian?» — 1, «optimise reg?» — 1 (пусть параметр регуляризации подбирается с помощью метода L-кривых), «enh contrasts?» — 1 (выделяем контрастные объекты), «min app res» — 0, «max app res» — 1000 (исключим ненужные значения кажущихся сопротивлений, если они есть). Жмём кнопку «INVERSION» и видим в окне консоли, что программа совершила 4 итерации, сходимость при этом достаточно плохая. Причина в малом количестве электродов на поверхности (для такой задачи) и в сложной модели, где присутствует множество помех в виде металлических объектов. Но наша цель и была в том, чтобы оценить влияние металлических помех на результат. Жмём кнопку «vis. inv.» и смотрим результаты... Лучше смотреть логарифмические результаты, так как у нас очень сильно контрастные объекты и среды:



Анализ результатов приведён на рисунке. Мы бы не смогли сделать подобный анализ, не проведя моделирование прямой задачи с известными металлическими конструкциями. Такое моделирование на практике помогло

бы нам избежать ошибок и неверных выводов при решении задачи инверсии «в поле».

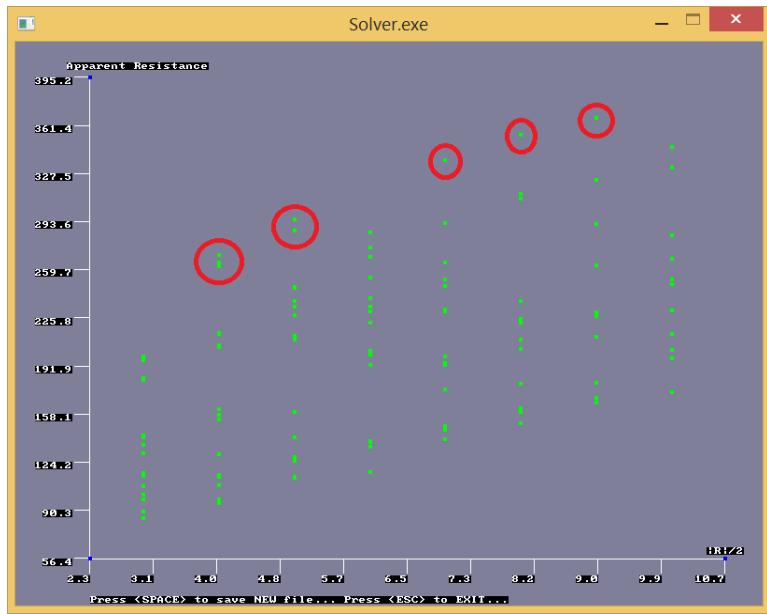
Далее можете самостоятельно посмотреть результаты на остальных профилях и в 3D случае.

Пример решения простой 2D задачи инверсии.

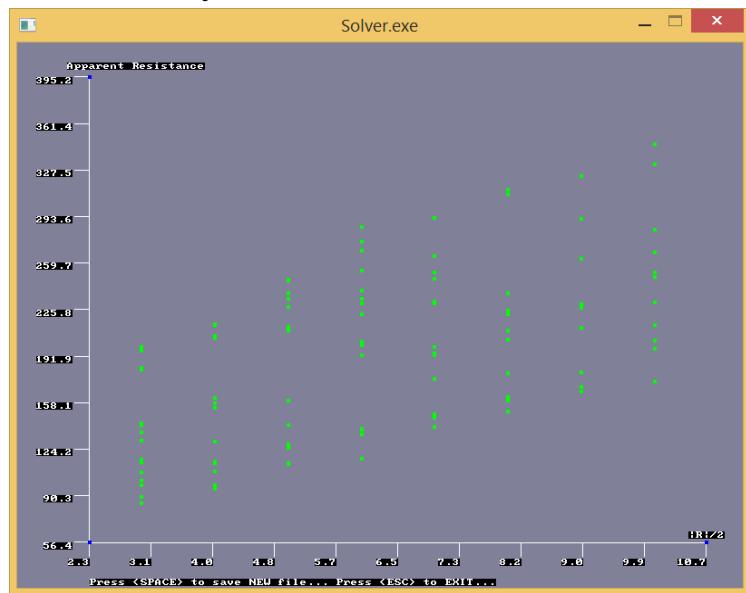
Данные примеры были взяты из набора примеров программного комплекса <https://gitlab.com/resistivity-net/bert>, при этом в файл входных данных были внесены небольшие несущественные изменения для лучшей совместимости со всеми элементами реализованной программы DiInSo. Входные файлы есть в папке «examples» программы DiInSo и называются simple_test.bert, simple_testERR.bert, simple_testIP.bert, simple_testIPERR.bert. Все эти файлы предназначены для 2D инверсии. Файл simple_test.bert содержит простые данные для инверсии, файл simple_testERR.bert содержит данные для инверсии, где были оценены ошибки измерений, файл simple_testIP.bert содержит дополнительно данные вызванной поляризации и, наконец, simple_testIPERR.bert содержит дополнительно как данные вызванной поляризации, так и оценки ошибок измерений. Поскольку мы рассматриваем простой пример, не будем выставлять вообще никаких дополнительных параметров для 2D инверсии. Поэтому выставляем только тип инверсии «2D» и последовательно рассматриваем каждый из входных файлов, выбирая их с помощью кнопки «DATA FILE:». Итак, начнём...

Файл simple_test.bert:

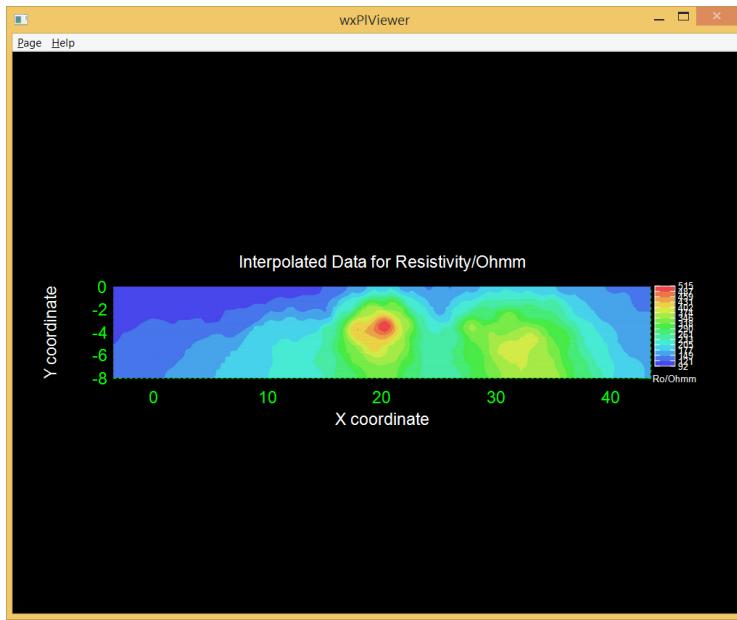
Чтобы было интереснее решать задачу инверсии, давайте поработаем с входными данными. Попробуем, например, исключить «ненужные» данные. Выбираем в поле «Exclude arrays from:» значение «graphic», жмём кнопку «Exclude» и видим следующую картину:



Допустим, у нас есть некоторые научные предположения, которые позволяют сделать вывод, что выделенные красными кружками точки выбиваются из общей картины измерений. Исключаем их, окружая замкнутыми кривыми и получаем:



Жмём <SPACE> на клавиатуре и получаем новый файл simple_testnewgraph.bert, выходим из окна исключения результатов, нажав <ESC>. Открываем новый файл simple_testnewgraph.bert с помощью кнопки «DATA FILE:». Жмём кнопку «INVERSION» и достаточно быстро получаем результат при очень хорошей сходимости. Смотрим результаты инверсии, нажимая кнопку «vis. inv.»:



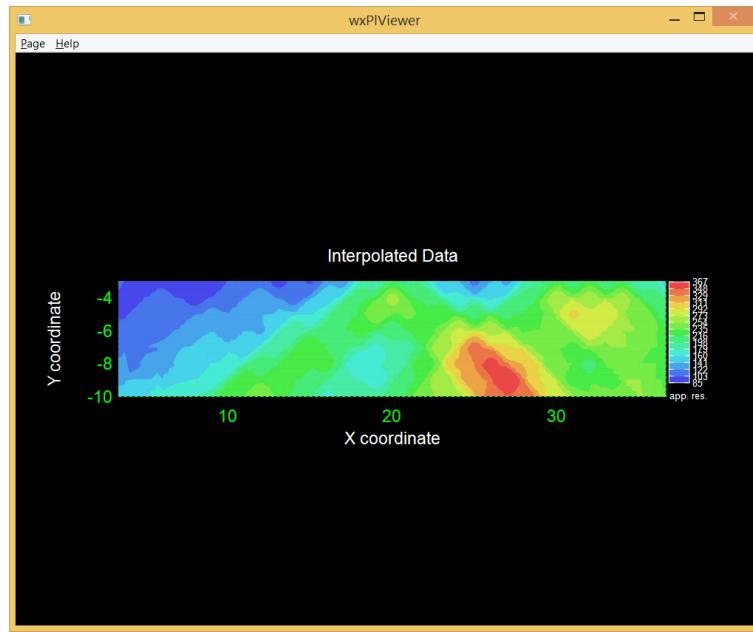
Далее можете поэкспериментировать самостоятельно — убирать точки с данными, открывать полученные *.vtk файлы с результатами, вводить какие-то параметры для инверсии и так далее.

Файл simple_testERR.bert:

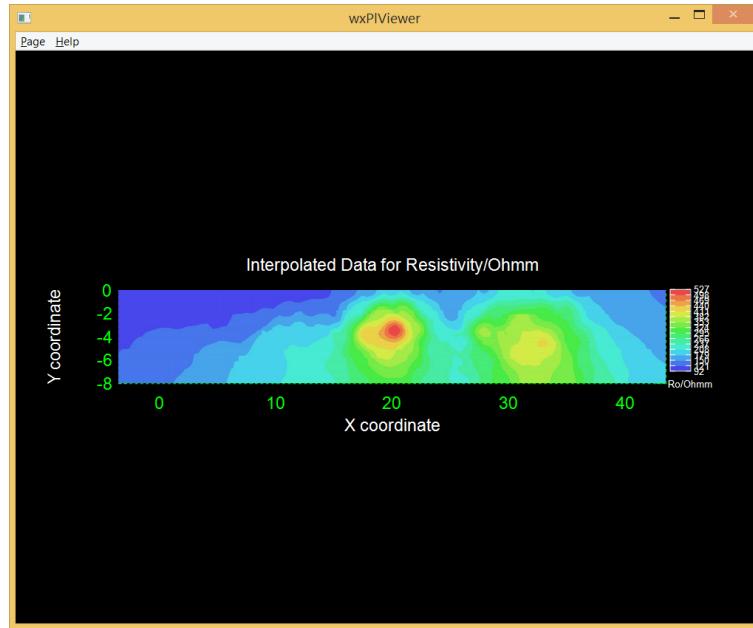
Данный файл мало чем отличается от предыдущего, просто в нём есть дополнительные данные по ошибкам измерений. Давайте вспомним, что нужно делать, если нам эти данные по ошибкам измерений не нужны и мы хотим сделать новые «автоматические» оценки ошибок измерений. Правильно, нужно выставить значение параметра «estim err» равным 1 и тогда записанные во входном файле данные по ошибкам измерений будут пересчитаны в ходе работы алгоритма инверсии.

Файл simple_testIP.bert:

Давайте посмотрим графический вид входных данных, нажав кнопку «vis. app. res.»:

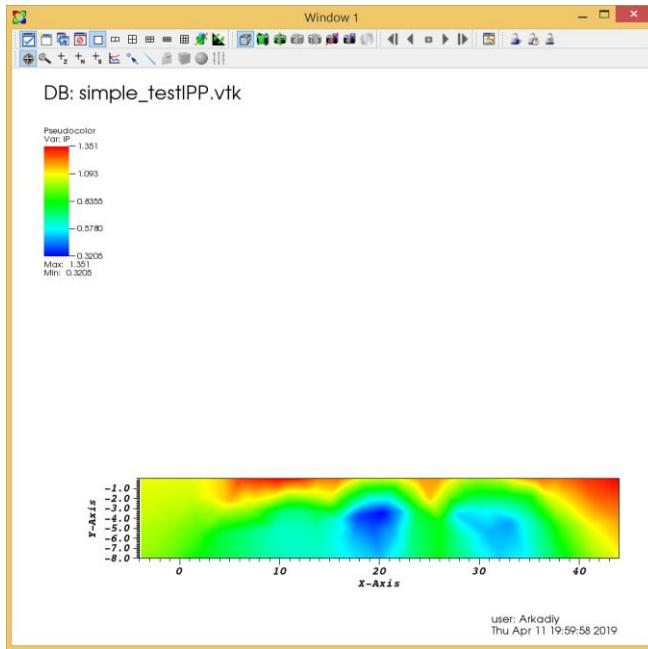


Обратите внимание, что несмотря на то, что во входном файле есть данные вызванной поляризации (ВП), на картинке вы видите данные по кажущимся сопротивлениям. Причина в том, что данные ВП мало интересны в таком представлении, они нас будут интересовать в качестве результата инверсии. Проведем инверсию, нажав кнопку «INVERSION», и посмотрим результаты, нажав кнопку «vis. inv.»:



И вновь вы видите результаты инверсии с данными сопротивлений, а не вызванной поляризации (ВП). Чтобы увидеть результаты инверсии ВП, вам нужно открыть один из получившихся файлов-результатов в стороннем визуализаторе. Например, будем работать с файлами *.vtk, как с самым удобным форматом, по мнению многих учёных. Воспользуемся программой

VisIt, ссылка на которую есть выше по тексту. Открываем файл simple_testIPP.vtk, выбираем в VisIt меню кнопки «Add» -> Pseudocolor -> IP, нажимаем кнопку «Draw» и видим результат:



Справедливо ради, наши данные ВП были забиты во входной файл НЕ реальными значениями, поэтому не нужно пытаться как-то интерпретировать полученный результат.

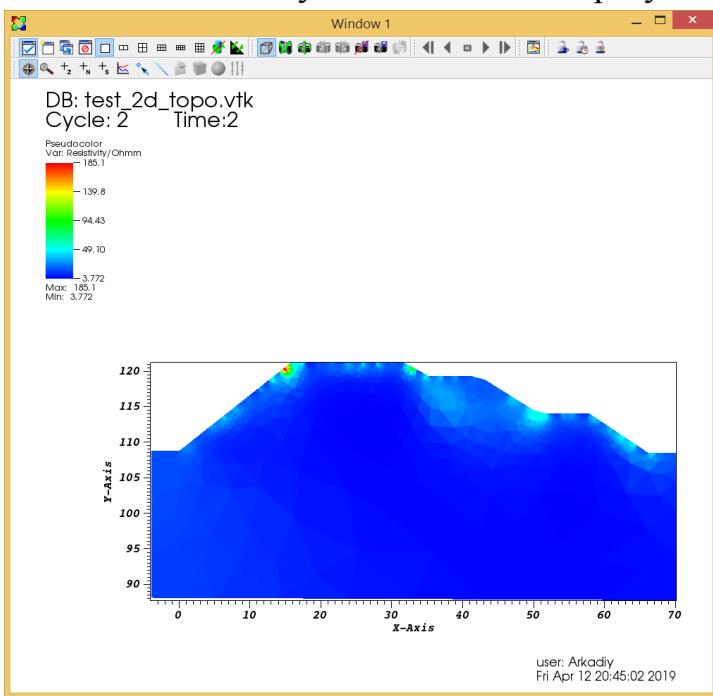
Файл simple_testIPERR.bert:

Данный файл представляет собой пример самого «богатого» набора данных — здесь вам и вызванная поляризация (ВП), и ошибки измерений в качестве дополнительных данных. Впрочем, на этом уникальность этого файла заканчивается. Поработайте с ним самостоятельно.

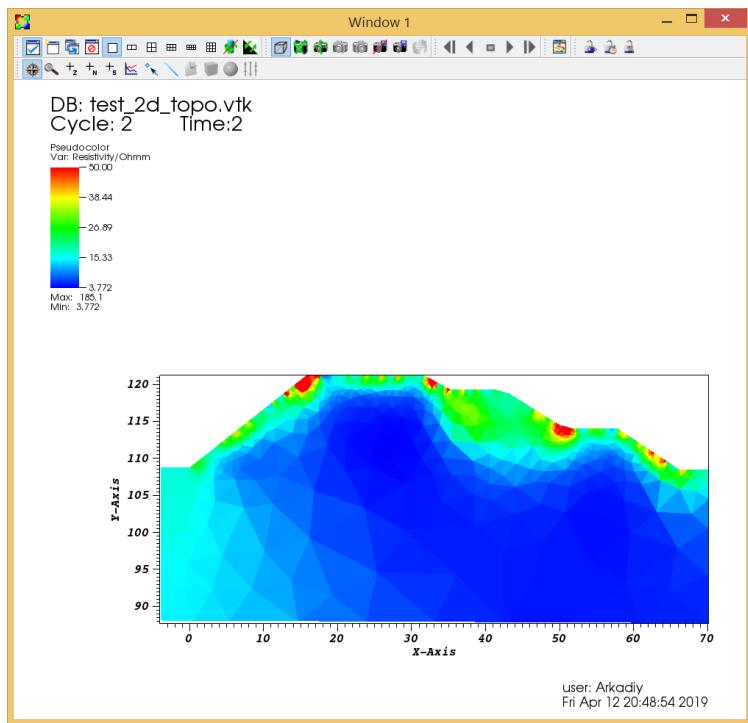
Пример решения 2D задачи инверсии с топографическим эффектом.

Данный пример был взят из набора примеров программного комплекса <https://gitlab.com/resistivity-net/bert>, при этом в файл входных данных были внесены небольшие изменения для лучшей совместимости со всеми элементами реализованной программы DiInSo. Входной файл есть в папке «examples» программы DiInSo и называется test_2d_topo.bert. Этот файл предназначен для 2D инверсии. Сначала выставим параметры 2D инверсии. Пусть «size of cells» — 0.25, «mesh growing» — 34 (нам нужна довольно подробная сетка), «recalc jacobian?» — 1 (будем пересчитывать Якобиан на каждом шаге итерации). Далее выбираем входной файл с помощью кнопки «DATA FILE:». Внимательный читатель заметил, что мы не выставили параметр топографии «topography». Да, мы действительно не сделали этого и хотим посмотреть результат без учета эффекта топографии. Итак, нажимаем

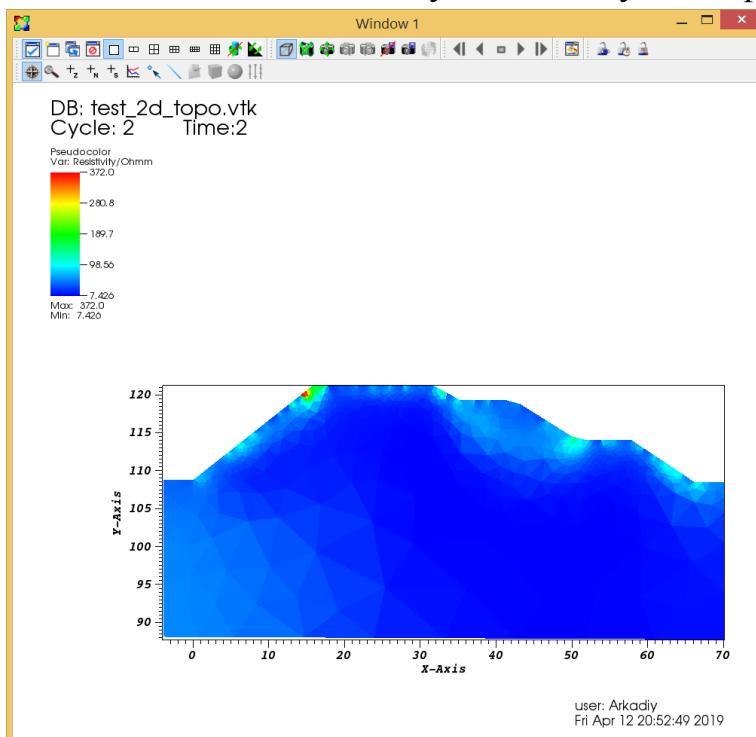
кнопку «INVERSION» и ждём окончания процесса инверсии. В консольном окне, кроме всего прочего, будет присутствовать надпись «TOPOGRAPHY FOUND BUT TOPOGRAPHY FLAG DISABLED!». Вот почему иногда важно читать информацию в консоли. Программа сообщает о возможной ошибке пользователя, но процесс инверсии при этом продолжается, необходимость учёта топографии в данном случае не является критически необходимым. Далее мы можем посмотреть результаты, нажав кнопку «vis. inv.», но лучше этого не делать и вот почему. Встроенный в программу визуализатор ВСЕГДА изображает прямоугольную область, поэтому данные будут интерполированы и в те части прямоугольника, где их фактически нет из-за эффекта топографии. Поэтому, когда мы имеем дело с топографией, лучше воспользоваться более профессиональными пакетами визуализации, например, всё тем же VisIt, о котором уже говорилось выше. Открываем файл test_2d_topo.vtk, выбираем в VisIt меню кнопки «Add» -> Pseudocolor -> Resistivity/Ohmm, нажимаем кнопку «Draw» и видим результат:



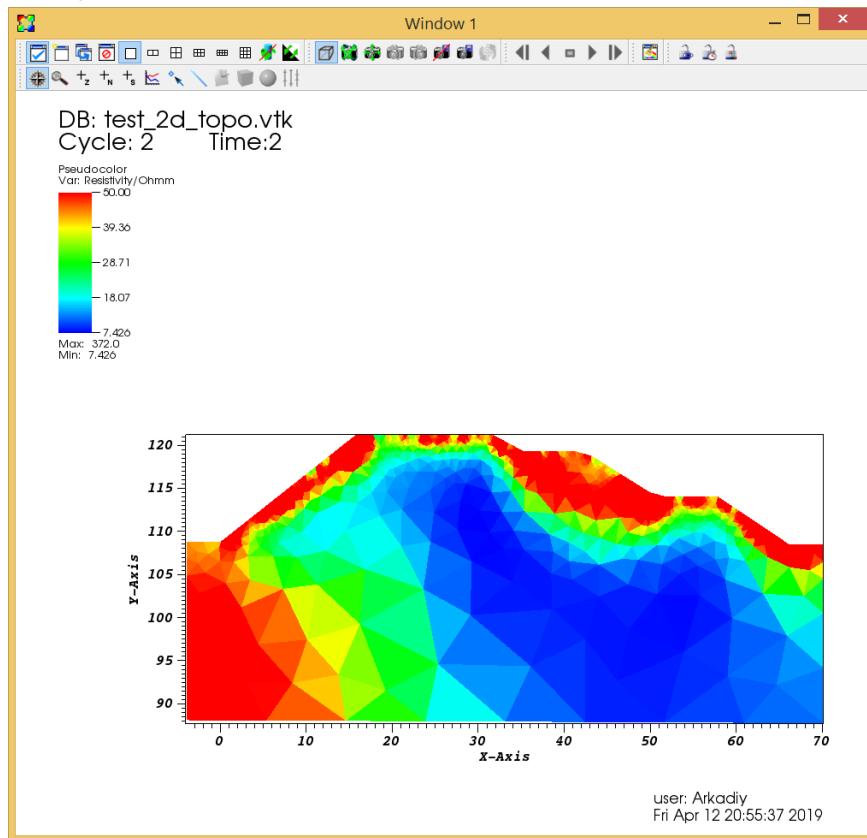
Не самое лучшее представление результатов из-за маленькой области с большим сопротивлением. Раскроем меню «Pseudocolor — Resistivity/Ohmm», нажав на стрелку слева и два раза нажмем на появившемся слове «Pseudocolor». Теперь ограничим верхний лимит значений, например, числом 50. Теперь картинка стала гораздо информативнее:



Однако, повторимся, в этом результате эффект топографии НЕ был учтён, даже несмотря на то, что область выглядит так, как область с топографией. Давайте проверим, можно ли доверять этим результатам. Для этого решим задачу с топографическим эффектом, выставив параметр «topography» равным 1. Жмём «INVERSION» и далее повторяем все те процедуры, что описаны выше. У нас получился следующий результат:



Вновь не очень хорошее представление из-за маленькой области с большим сопротивлением, однако максимальное значение сопротивлений теперь значительно больше. Ограничим верхний лимит значений всё тем же числом 50 и получим:

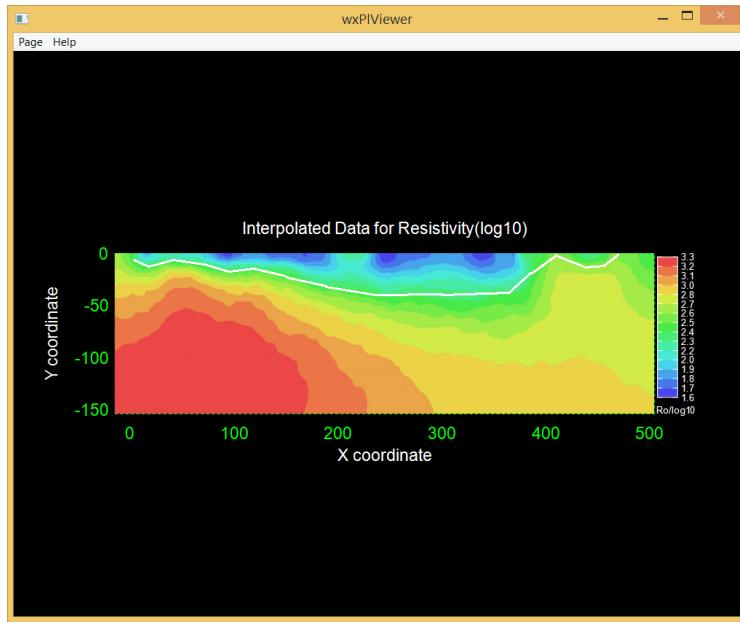


Мы видим, что с точки зрения геометрии подобластей наши результаты получились схожими в случае учёта и НЕ учёта эффекта топографии, а вот с точки зрения значений сопротивлений отличия довольно существенны, даже несмотря на то, что мы использовали одинаковые настройки для инверсии (кроме настройки, отвечающей за эффект топографии). На самом деле, можно найти и другие примеры, которые показывают, что не учёт эффекта топографии приводит к заметному искривлению подобластей, но, как мы показали, бывают и иные случаи. Например, в задаче выше: если вам больше важна геометрия, а не значения сопротивлений, то забыв про топографию вы многое не потеряете. Однако, ещё раз повторим, это актуально не всегда!

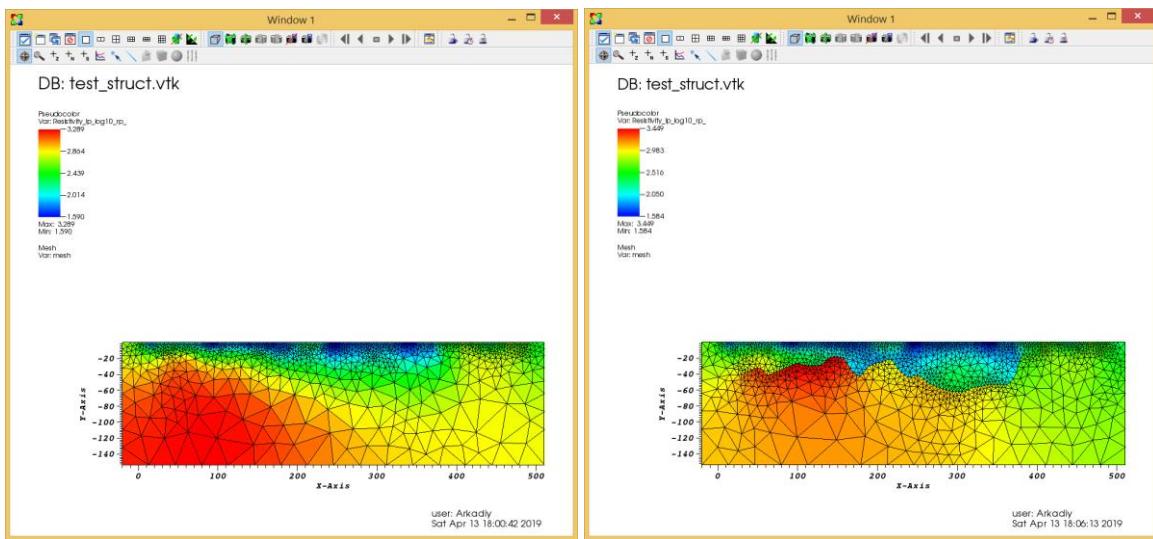
Пример решения 2D задачи инверсии с выделением границ областей с разными электрическими сопротивлениями

Данный пример был взят из набора примеров программного комплекса <https://gitlab.com/resistivity-net/bert>, при этом в файл входных данных были внесены небольшие изменения для лучшей совместимости со всеми элементами реализованной программы DiInSo. Входной файл есть в папке

«examples» программы DiInSo и называется test_struct.bert. Этот файл предназначен для 2D инверсии. Сначала выставим параметры 2D инверсии. Пусть «size of cells» — 0.25, «mesh growing» — 34 (нам нужна сетка средней подробности), «recalc jacobian?» — 1 (будем пересчитывать Якобиан на каждом шаге итерации). Далее выбираем входной файл с помощью кнопки «DATA FILE:». Данный тест отличается от предыдущих тем, что нам известны сложные границы областей (в данном случае имеются в виду границы материковой породы), которые содержатся в файле test_struct.xz. Выберем данный файл с помощью кнопки «topo/line file:». Сначала не будем выставлять галочку слева от кнопки «topo/line file:» и посмотрим результат инверсии без информации о границах областей. Жмём кнопку «INVERSION», жмём кнопку «vis. inv», смотрим результат (будем смотреть логарифмические значения, чтобы было лучше видно границы областей):



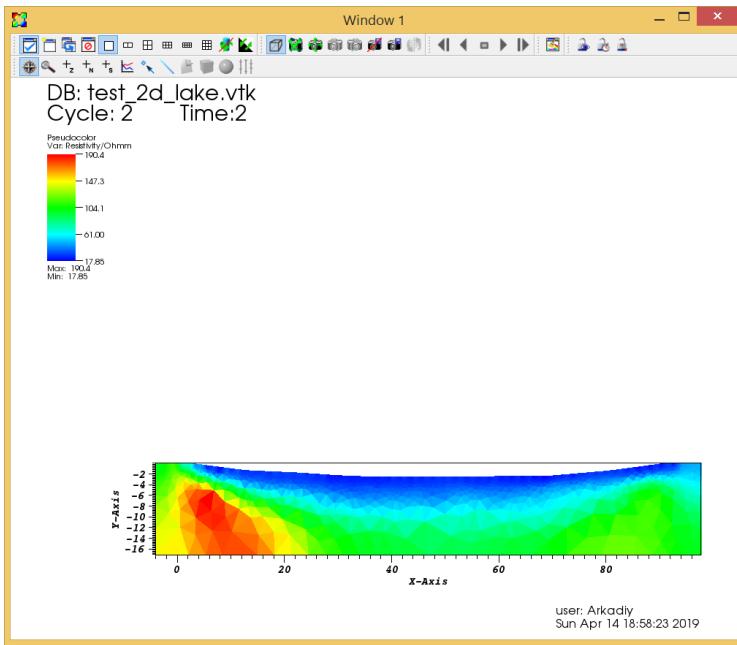
На картинке с результатами можно примерно выделить две явные подобласти даже без применения знаний о границах (мы выделили примерную границу между областями белой линией). Теперь выполним инверсию с информацией о границах областей и сравним два результата, посмотрев их в визуализаторе VisIt (в VisIt сравнивать удобнее, так как в нём можно одновременно просматривать и результат: Add -> Pseudocolor -> Resistivity_lp_log10_rp_, и сетку: Add -> Mesh -> mesh). Итак, слева — логарифмические значения без информации о границах, справа — логарифмические значения с информацией о границах (открывать лучше файл БЕЗ постфикса «P», чтобы увидеть значения по ячейкам):



Мы видим, что выделение границ подобластей в случае использования информации о границах выглядит не только намного более явно, но и помогает практически избавиться от плавных переходов между этими границами. Фактически мы имеем результат с относительно резкими переходами значений сопротивлений, чего практически невозможно достичь, если не использовать предварительные данные о границах подобластей.

Пример решения 2D задачи инверсии с известной информацией о подобласти модели

Данный пример был взят из набора примеров программного комплекса <https://gitlab.com/resistivity-net/bert>, при этом в файл входных данных были внесены небольшие изменения для лучшей совместимости со всеми элементами реализованной программы DiInSo. Входной файл есть в папке «examples» программы DiInSo и называется test_2d_lake.bert. Этот файл предназначен для 2D инверсии. Сначала выставим параметры 2D инверсии. Пусть «size of cells» — 0.25, «mesh growing» — 34 (нам нужна довольно подробная сетка), «recalc jacobian?» — 1 (будем пересчитывать Якобиан на каждом шаге итерации). Далее выбираем входной файл с помощью кнопки «DATA FILE:». В этом примере измерения проведены на не ровном дне озера. Поэтому нам необходимо выставить параметр топографии «topography» как 1. Сначала решим задачу без учёта того момента, что над электродами находится слой воды. Жмём кнопку «INVERSION» и смотрим результаты в программе VisIt (напомним, что данные с топографией лучше смотреть в VisIt):



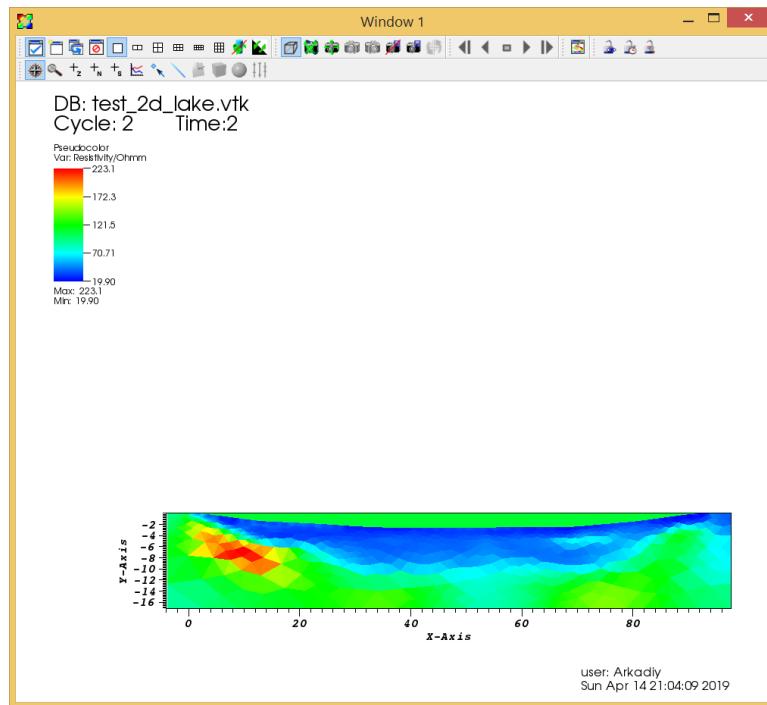
Теперь учтём тот момент, что над электродами находится слой воды. Для этого нам нужно внести небольшие изменения в геометрию области, но сначала нужно получить саму геометрию. Нажимаем кнопку «geom+mesh» и получаем файл `test_2d_lakeGEO.ply`. Открываем его в текстовом редакторе. Чтобы задать верхний слой воды, нужно соединить крайние точки с глубиной ноль, в нашем случае это точки «`4 2.0000000000000e+00 0.0000000000000e+00 -99`» и «`184 9.1745200000000e+01 0.0000000000000e+00 -99`». Изучив формат `*.poly` мы знаем, что соединить точки мы можем добавив одно новое ребро. Таким образом, увеличиваем число ребер со 198 до 199, а в конце списка рёбер добавляем строчку «`198 4 184 -1`» без кавычек. Мы использовали маркер ребра «`-1`», потому что мы добавили верхнюю границу. Менять маркеры у остальных рёбер со значением «`-1`» при этом не нужно, так как это граница подобласти внутри области решения (вспоминаем формат `*.poly`). Теперь отметим нашу новую подобласть. Увеличим список областей на 1, то есть теперь их будет 3 (область где интересует решение, область для удаления внешних границ где решение нас НЕ интересует и слой воды). Как мы помним, область помечается внутренней точкой, поэтому добавим следующую запись в список областей: «`2 50 -1 3 0.0`». Маркер новой области будет 3, так как маркеры 1 и 2 уже заняты. Новый файл с геометрией готов, и мы будем далее использовать его. Выбираем созданный нами файл с геометрией, нажав кнопку «geometry file:», отмечаем галочку слева от этой кнопки.

Теперь определим файл с информацией об областях, имеющий расширение `*.control`. Создадим его вручную с помощью любого текстового

редактора и дадим любое удобное имя с расширением *.control. Запишем в файл следующие параметры:

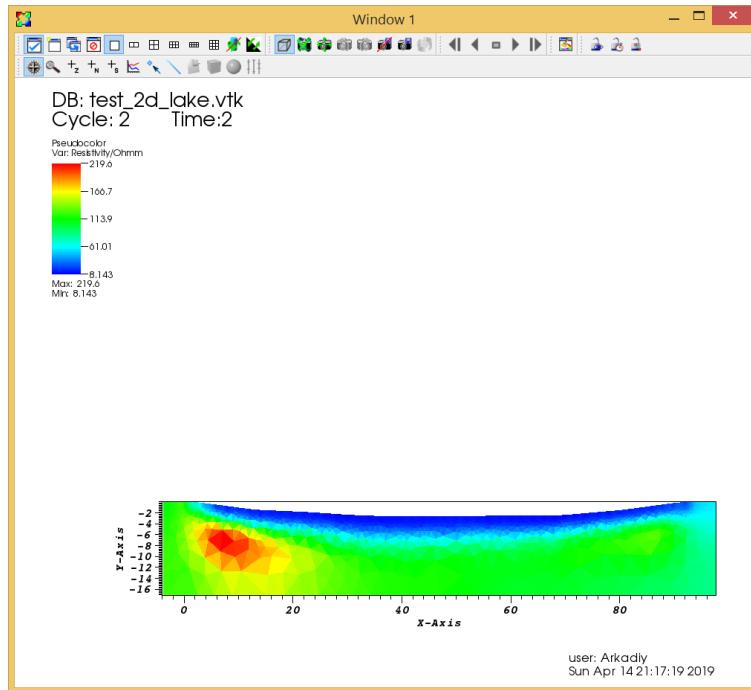
```
#No start Trans zWeight Ctype lBound uBound
2 100 log 0.1 1 10 1000
#No single start
3 1 22.5
#No background
1 1
```

То есть мы задаём основную область как «умеренно-горизонтальную», но при этом содержащую некоторые объекты-включения. Границы сопротивлений области простираются от 10 до 1000 Ом^м со стартовым значением 100 Ом^м. Область воды мы помечаем как область с постоянным значением сопротивления, которое начинается от значения 22.5 Ом^м (истинное значение сопротивления воды). Область номер 1 — это та область, где решение нас не интересует. Выбираем созданный нами файл с областями, нажав кнопку «region file:», отмечаем галочку слева от этой кнопки. Производим инверсию, нажав кнопку «INVERSION», смотрим результат в VisIt:

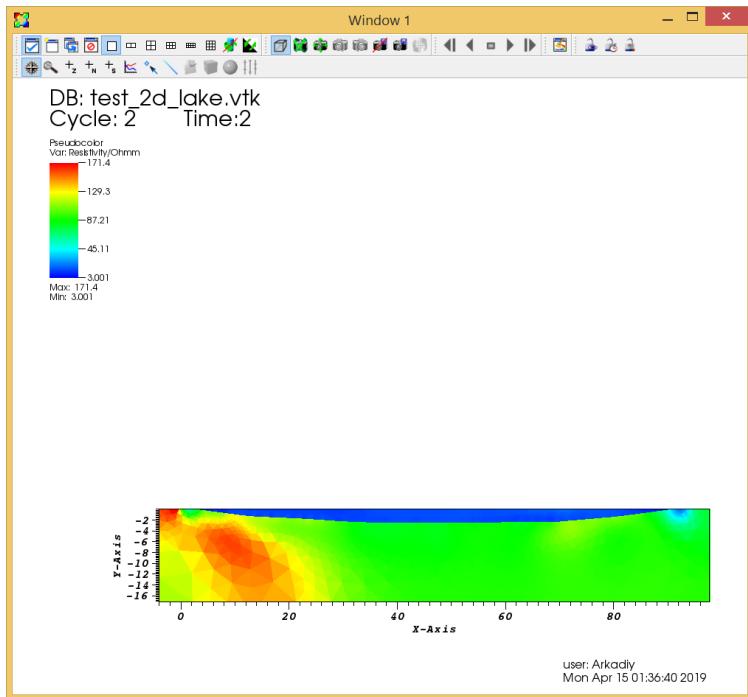


Мы видим, что алгоритм инверсии дал несколько другое распределение сопротивлений ниже дна озера, но значение сопротивления самого озера оказалось далеко от реального (реальное — 22.5 Ом^м). В чем же может быть причина? Как ни странно, причина здесь в том, что мы не отключили опцию топографии. Выше мы обсуждали, что опция включения топографии

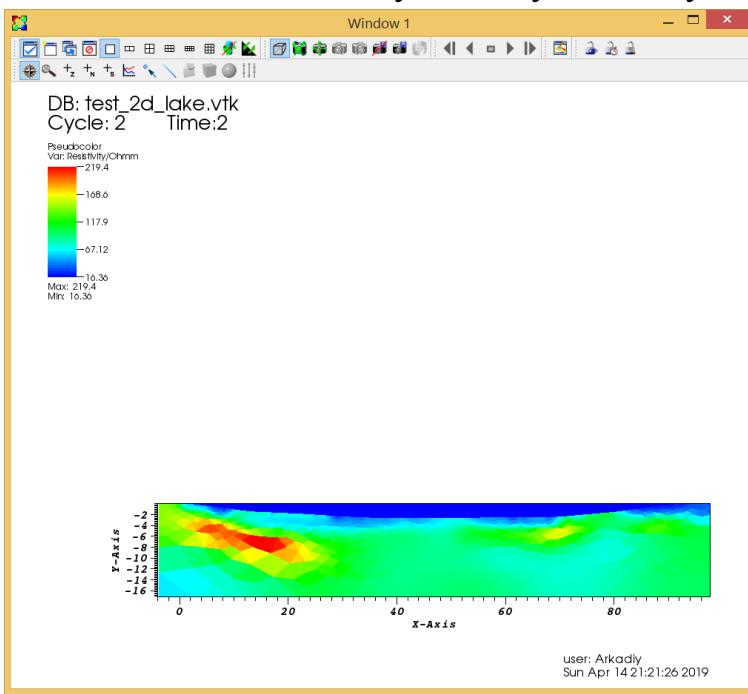
пересчитывает коэффициенты установок и, соответственно, кажущиеся сопротивления. Однако бывает и такое, что это сделали исследователи, поскольку топография была не сложной и на вход подаются уже корректные данные, которым не требуется пересчёт. Таким образом, включение опции топографии только испортило нам результат. Давайте вернёмся к изначальной задаче, когда мы не учитывали слой воды и посчитаем всё заново без эффекта топографии («topography» — 0):



Результат отличается заметно. А что будет, если провести инверсию без знаний об областях, но зная границу между водой и остальной средой. Получим следующий результат:



Алгоритм инверсии правильно определил однородность верхней водной среды хоть и занизил значения сопротивления воды, но достаточно ли точны результаты ниже дна? Чтобы понять это нужно попробовать решить задачу инверсии с известными данными о слое воды. После того, как мы учтём слой воды так, как это было описано выше, будет получен следующий результат:



Прежде всего, мы видим, что сопротивление воды стало куда ближе к реальности. Кроме того, учёт слоя воды позволил нам получить заметно изменённую (в данном случае, более адекватную) модель ниже дна. Таким образом, мы увидели пример, когда наличие одной области с известными

характеристиками не только уточняет всю модель, но и позволяет понять, что входные данные неадекватны, либо что мы неправильно с ними работаем. Вообще, чем больше мы знаем о той области, в которой работаем, тем больше шансов правильно провести инверсию. Казалось бы, очевидная мысль, но часто забываемая исследователями.

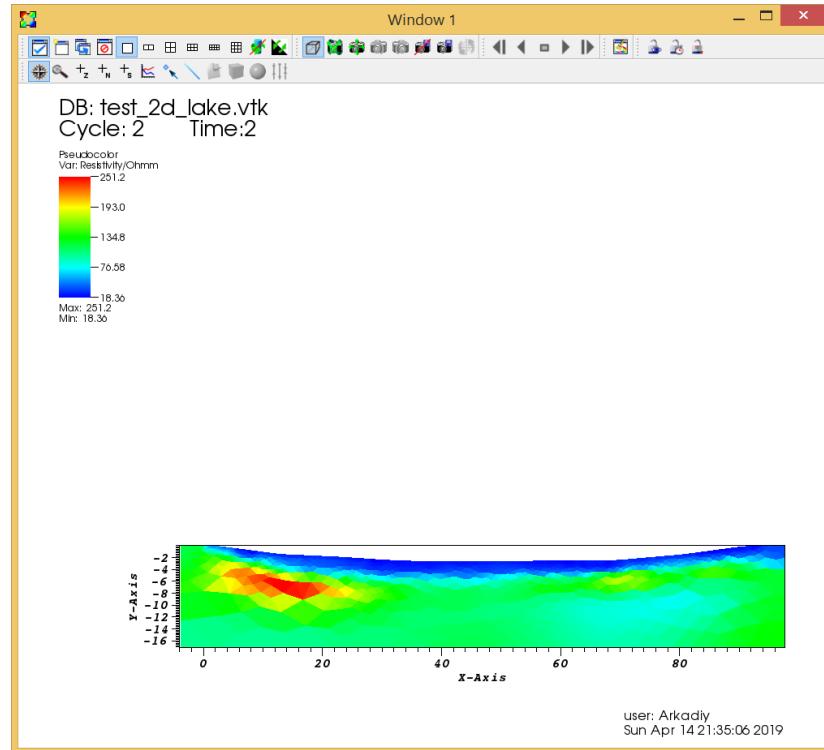
Какие ещё есть варианты задания области воды с помощью файла *.control? Например, такие:

```
#No single start Trans lBound uBound
3 1      22.5 log 22      23
```

То есть мы задаём область в чётких границах сопротивлений, которые нас интересует. Или такие:

```
#No fix
3 22.5
```

Напомним, что при данном подходе алгоритму инверсии будет «сообщено» значение сопротивления области ($22.5 \text{ Ом} \cdot \text{м}$), а сами расчёты будут проходить без расчётов внутри этой области (значения строго зафиксированы и не меняются). Поскольку вода исключена из расчётов, то и в результатах области воды не будет:



Результат оказался несколько похожим на тот, когда мы проводили расчёты с исключённым водным слоем, однако имеет ряд важных отличий в виде размеров и форм аномалий.

Наконец, можно задать правила перехода между слоем воды и основным слоем, ведь этот переход обычно имеет место (вода —> влажная подстилающая среда —> сухая подстилающая среда):

```
#No single start
3 1      22.5
#Inter-region
2 3 0.5
```

Кроме того, мы можем предположить, что слой воды имеет не совсем однородную структуру (например, за счёт слоя ила или переменной температуры) и задать этот слой наподобие того, как мы задаём основной слой, да ещё и использовать для расчётов внутри слоя воды другой параметр регуляризации, а именно:

```
#No start Trans zWeight Ctype lBound uBound MC
2 25 log 0.1 1 10 1000 1
3 22.5 log 0.01 1 10 30 3
#Inter-region
2 3 0.5
```

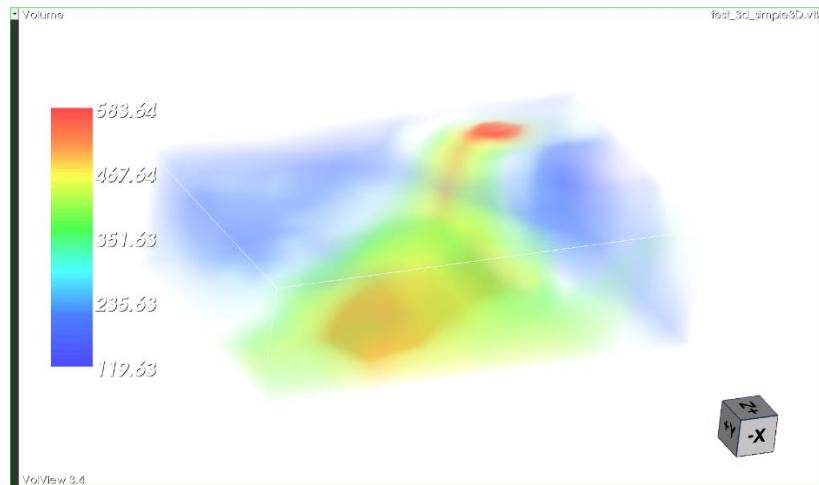
Попробуйте эти варианты самостоятельно.

Другими словами, идея для файла *.control может быть великое множество, главное понимать, что мы хотим получить и иметь некоторые представления о регионе, где проводились измерения.

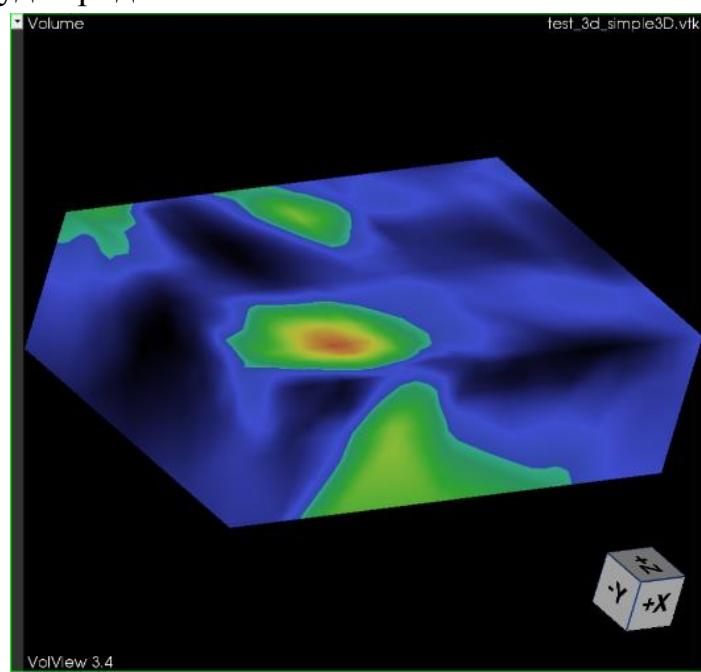
Пример решения простой задачи 3D инверсии и задачи 3D инверсии с известной информацией о границах подобластей.

Данный пример был взят из набора примеров программного комплекса <https://gitlab.com/resistivity-net/bert>, при этом в файл входных данных были внесены небольшие изменения для лучшей совместимости со всеми элементами реализованной программы DiInSo. Входной файл есть в папке «examples» программы DiInSo и называется test_3d_simple.bert. Этот файл предназначен для 3D инверсии. Сначала выставим параметры 3D инверсии. Пусть «quality» — 30, «mesh grow» — 2 (не будем задавать слишком подробную сетку с целью экономии памяти), «model depth» — 10 (определим глубину исследований самостоятельно), «recalc jacobian?» — 1 (будем пересчитывать Якобиан на каждом шаге итерации), «regular strength» — 5 (параметр регуляризации определим самостоятельно). Далее выбираем входной файл с помощью кнопки «DATA FILE:». Число профилей в данной трёхмерной задаче равно 9, в каждом профиле 14 электродов, число положений установки — 753. Можно приступать к инверсии. Жмём кнопку «INVERSION», решение будет получено за 5 итераций при достаточно

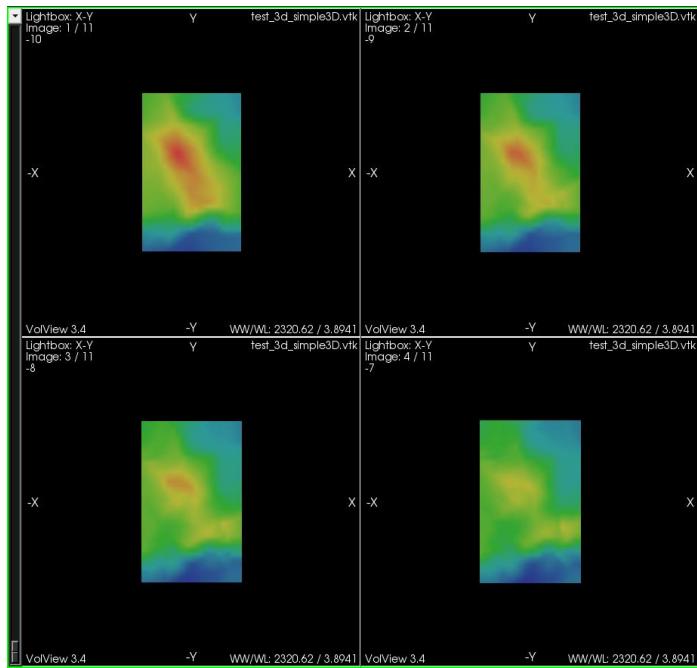
неплохой сходимости. Предлагаю для разнообразия посмотреть результаты решения в программе VolView, о которой шла речь выше, открыв файл с результатами `test_3d_simple3D.vtk`. В VolView выбираем тип данных «Scientific/General data». Регулируя параметры «Scalar Opacity Mapping» и «Scalar Color Mapping», можно выделять нужные нам объекты и, благодаря настройкам прозрачности, а также вращению модели, видеть полную картину распределения слоёв/объектов:



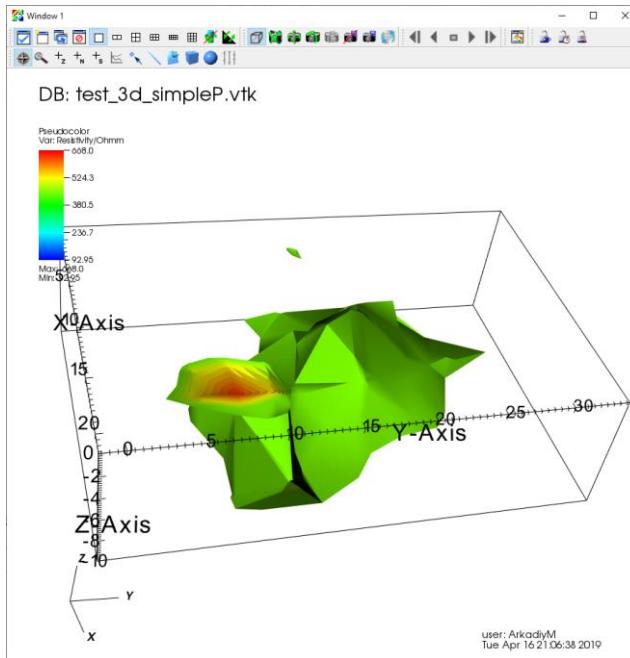
Если вам не удаётся настроить прозрачность, то, скорее всего, вы используете старую или встроенную (простую) видеокарту, в этом случае вы увидите что-нибудь вроде:



Можно просматривать результаты и в разных разрезах:



Открыв результаты моделирования в программе VisIt, можно посмотреть поверхности равного уровня (Operators -> Slicing -> Isosurface). Например, если нас интересуют поверхности равного уровня выше значения 450 Ом^{*}м, то они будут выглядеть вот так:



Теперь давайте предположим, что у нас есть информация о границе некоторой внутренней области и нам нужно добавить эту границу в модель. Допустим эта граница достаточно сложная и мы её построили в каком-нибудь трёхмерном редакторе, например, в свободно распространяемом редакторе с открытым исходным кодом Blender (<https://www.blender.org/>). Мы не будем заниматься практикой создания трёхмерных моделей, а просто

воспользуемся уже готовым объектом из обучающей инструкции Blender, расположенной по адресу:
https://en.wikibooks.org/wiki/Blender_3D:_Noob_to_Pro/Creating_a_Simple_Hat

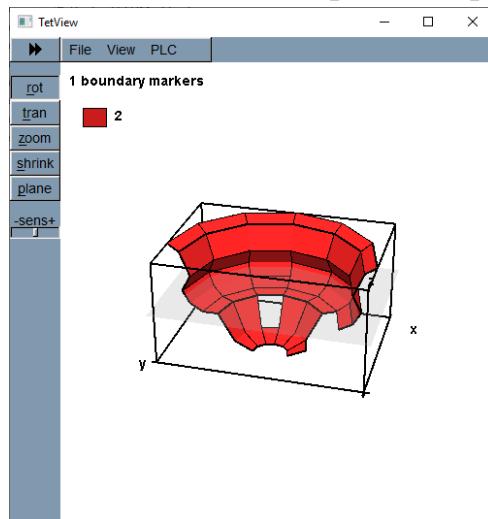
Используя этот пример, можно создать похожий объект и, сохранив его, например, в формате *.obj, конвертировать в нужный нам формат *.poly,

используя кнопку . Если положение, размеры и поворот объекта не были учтены в программе трёхмерного моделирования, то все эти манипуляции можно будет сделать здесь в программе DiInSo. Маркеры узлов и граней (первые два поля) нашей «границы области» выставляем как 0 (номер внутреннего узла) и 2 (номер, совпадающий с номером границы области, где ищется решение), поскольку таковы правила работы с форматом *.poly для TetGen и эти правила были описаны выше.

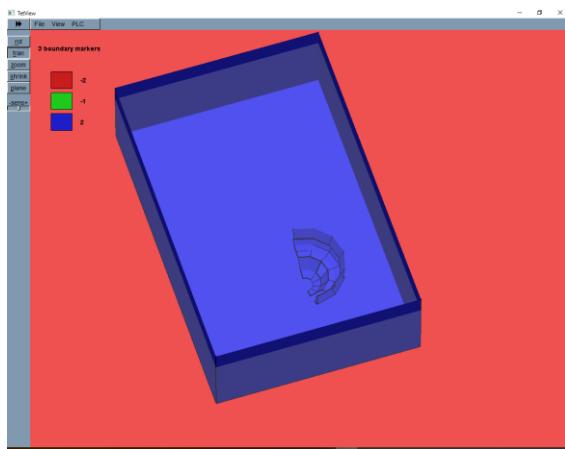
Далее создадим геометрию основной области моделирования, нажав кнопку «geom+mesh» и получив файл с геометрией test_3d_simpleGEOM.poly. Казалось бы, теперь можно объединять наши геометрии, нажав кнопку , но, к сожалению, не всё так просто. Проблема в том, что многие генераторы трёхмерных объектов не обращают внимание на такую проблему, как согласованность сетки. Другими словами, некоторые полигоны содержат, например, совпадающие точки или, что ещё хуже, точки, которые лежат очень близко друг к другу и почти совпадают. В таком случае может возникнуть пересечение полигонов и, как следствие, ещё большая несогласованность сетки. Вообще говоря, такие полигональные модели желательно исправлять даже с точки зрения компьютерной графики (в случае компьютерной графики для уменьшения размеров модели и экономии памяти ПК), но это делают далеко не всегда. Одной из программ для работы с уже готовыми сетками и в том числе их исправления является свободно распространяемая программа MeshLab: <http://www.meshlab.net/>. Обычно (но не всегда) для исправления сетки достаточно провести несколько простых действий в программе MeshLab. Открываем сетку кнопкой «Import Mesh» (либо сочетанием клавиш Ctrl+I), выбираем раздел меню Filters -> Clearing and Repairing -> Remove Duplicate Faces — удаляем совпадающие грани, затем Filters -> Clearing and Repairing -> Remove Duplicate Vertices — удаляем совпадающие узлы, затем Filters -> Clearing and Repairing -> Merge Close Vertices — объединяем близко расположенные узлы (можно оставить настройки по умолчанию, а можно и поработать с настройками) и, наконец, Filters -> Clearing and Repairing -> Remove Unreferenced Vertices — удаляем узлы, НЕ относящиеся к объекту. Можно проверить созданный объект на

корректность, выбрав раздел меню Filters -> Clearing and Repairing -> Select Self Intersecting Faces — выделить пересекающиеся грани. Очевидно, что таких граней быть НЕ должно, иначе объект НЕ является корректным в нашем смысле. Затем, если всё нормально, сохраняем сетку, например, в формате *.ply с помощью кнопки «Export Mesh As» (либо сочетанием клавиш Ctrl+Shift+E). И уже новый полученный файл *.ply будем конвертировать в

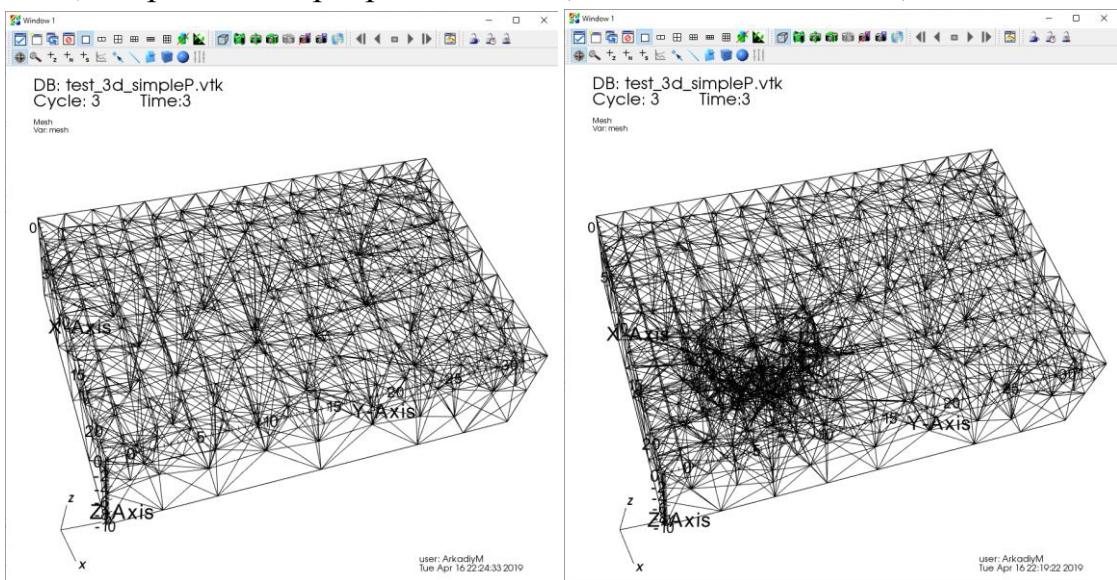
нужный нам формат *.poly с помощью кнопки  . Как уже было сказано выше, мы здесь не будем рассматривать тонкости 3D-моделирования, вам их придётся осваивать самостоятельно. Но факт остаётся фактом: нет согласованной сетки — нет рабочей итоговой модели. Исправленная вручную (не идеальным образом) модель объекта *.poly есть в уже известной нам папке «examples» и имеет имя test_3d_object.poly. Этот объект можно посмотреть, например, в программе TetView (<http://wias-berlin.de/software/tetgen/tetview.html>), о которой шла речь выше:



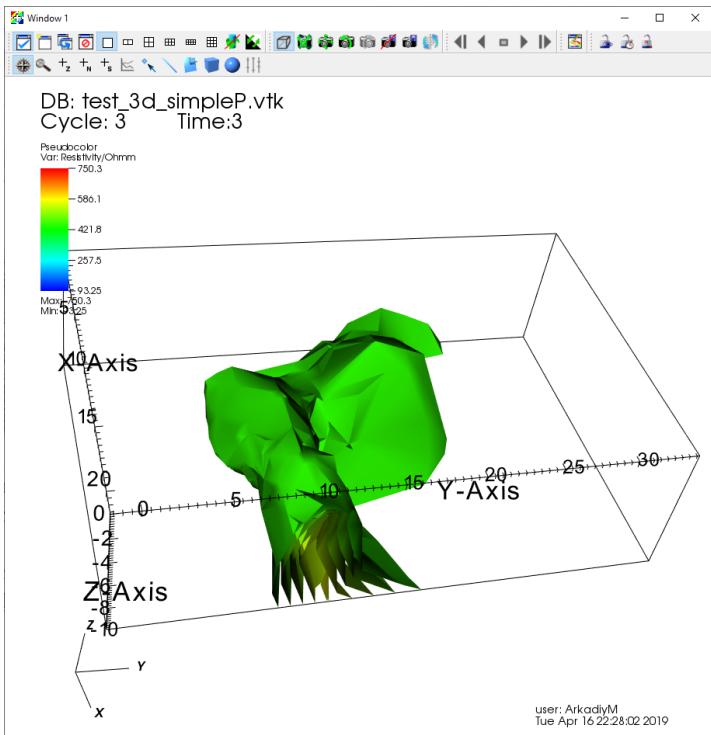
Объединим две геометрии — test_3d_simpleGEOM.poly и test_3d_object.poly с помощью кнопки  . Новый файл назовём, например, my_new_geom.poly. Посмотрим с помощью TetView, что получилось, увеличив область с включённым объектом:



Итак, мы включили наш объект—«границу» внутрь области моделирования. Теперь выбираем файл `my_new_geom.poly` с помощью кнопки «`geometry file:`», не забывая поставить галочку слева от кнопки. Проведём моделирование с помощью новой геометрии. Жмём кнопку «`INVERSION`», ждём окончания процесса инверсии (замечаем при этом, что сходимость стала лучше). Сначала сравним две сетки результирующего решения, открыв их в программе VisIt (Add -> Mesh -> mesh):



Мы видим, как на сетке справа отчётливо выделяется включённый нами объект—«граница». Значит включение границы области прошло корректно. Теперь посмотрим поверхности равного уровня выше значения сопротивления 450 Ом*м, которые мы наблюдали ранее (Operators -> Slicing -> Isosurface):



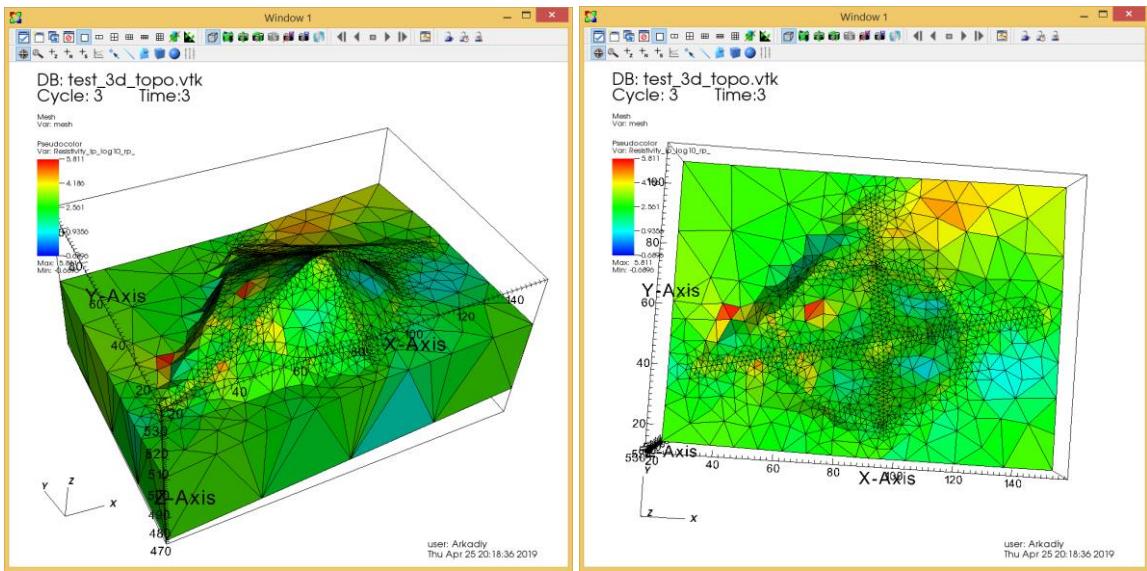
Как мы видим, результат в месте включения нашего объекта—«границы» несколько отличается. Связано это прежде всего с тем, что в этой области была сгенерирована более мелкая, а значит более подробная сетка, как видно на рисунке сетки выше (именно поэтому сходимость в случае включённого объекта—«границы» стала лучше — без объекта сетка была слишком грубой). Иного влияния в данном случае наш объект—«граница» не оказывает, ведь он подобран совершенно произвольным образом и включён в произвольное место.

В том случае, если бы нам захотелось включить в область моделирования не объект—«границу», а целостный (заполненный) трёхмерный объект, мы бы действовали точно так же, с той лишь разницей, что в случае заполненного трёхмерного объекта нам бы понадобилась замкнутая полигональная модель, нам бы понадобилось знать координаты любой точки внутри заполненного объекта, чтобы задать её при нажатии

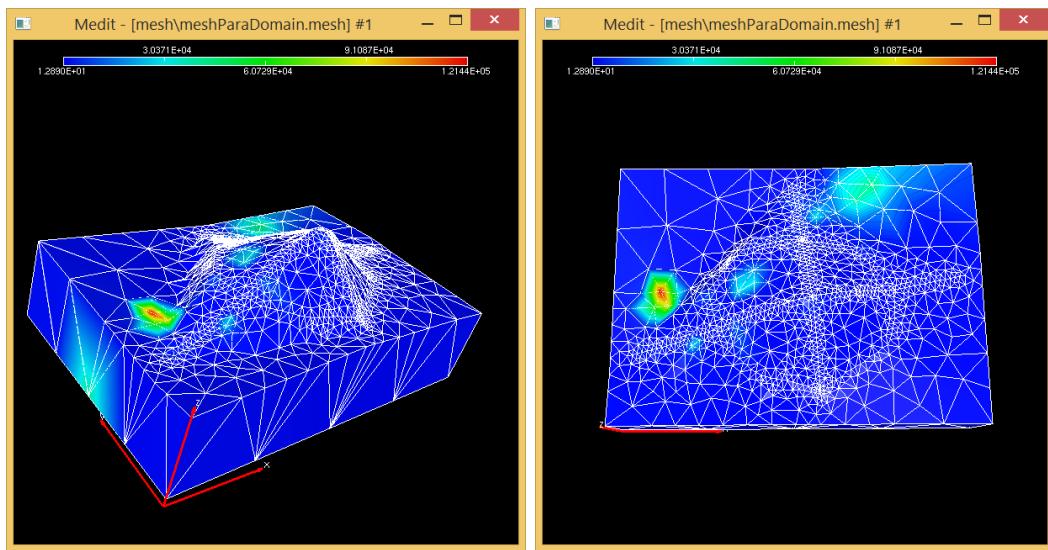
↗
или чтобы вписать её в файл *.poly (вспоминаем формат *.poly для TetGen) и, наконец, нам бы понадобился файл *.control, аналогичный тому, который использовался в 2D примере. В остальном процедура моделирования идентична.

Пример решения задачи 3D инверсии с топографией в исходном и в дополнительном файлах.

Данный пример был взят из набора примеров программного комплекса <https://gitlab.com/resistivity-net/bert>, при этом в файл входных данных были внесены небольшие изменения для лучшей совместимости со всеми элементами реализованной программы DiInSo. Входной файл есть в папке «examples» программы DiInSo и называется test_3d_topo.bert. Этот файл предназначен для 3D инверсии, электроды в файле располагаются на разных высотах, поэтому мы будем иметь дело с топографическим эффектом. Сначала выставим параметры 3D инверсии. Пусть «smooth?» — 1 (нам нужна сглаженная сетка на поверхности области моделирования), «quality» — 30, «mesh grow» — 2 (не будем задавать слишком подробную сетку с целью экономии памяти компьютера), «recalc jacobian?» — 1 (будем пересчитывать Якобиан на каждом шаге итерации), «topography» — 1 (будем учитывать топографический эффект). ВАЖНО ПОМНИТЬ, что учёт топографического эффекта в трёхмерном случае представляет из себя куда более сложную задачу, чем в двумерном случае, поэтому при наличии сложной поверхности моделирования пользователь обязательно ДОЛЖЕН прописывать опцию «topography» — 1, иначе построенная сетка может оказаться несогласованной с положениями электродов и программа DiInSo завершится аварийным образом. Далее выбираем входной файл с помощью кнопки «DATA FILE:». Число положений установки для данной задачи довольно велико — 2424, поэтому процесс решения может немного затянуться. Теперь мы готовы приступить к инверсии. Жмём кнопку «INVERSION», решение будет получено за 7 итераций при не очень хорошей сходимости (у нас слишком грубая сетка). Не будем в данном примере уделять большое внимание результатам инверсии внутри области моделирования, тем более у нас довольно грубая сетка, а обратим свой взор на результаты на поверхности области моделирования и на саму поверхность, которая имеет топографический эффект. Откроем файл с результатами моделирования test_3d_topo.vtk в программе VisIt и внимательно рассмотрим неровную поверхность (Add → Mesh → mesh, Add → Pseudocolor → Resistivity_lp_log10_rp_, Draw):



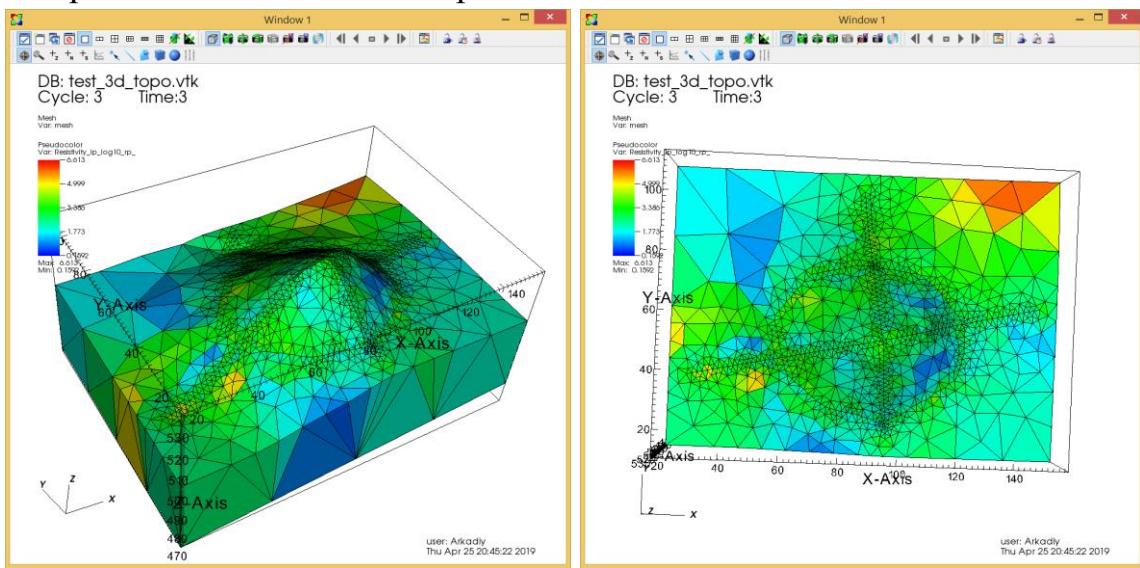
Либо можно посмотреть результаты с помощью встроенного визуализатора, нажав кнопку «vis. inv.» (хотя в данном случае представление о поверхности будет несколько хуже из-за иной реализации затенения во встроенным визуализаторе), а затем, например, кнопки клавиатуры «т» (данные), «о» (изолинии), «A» (оси координат) и «с» («добавить цвет поверхности»):



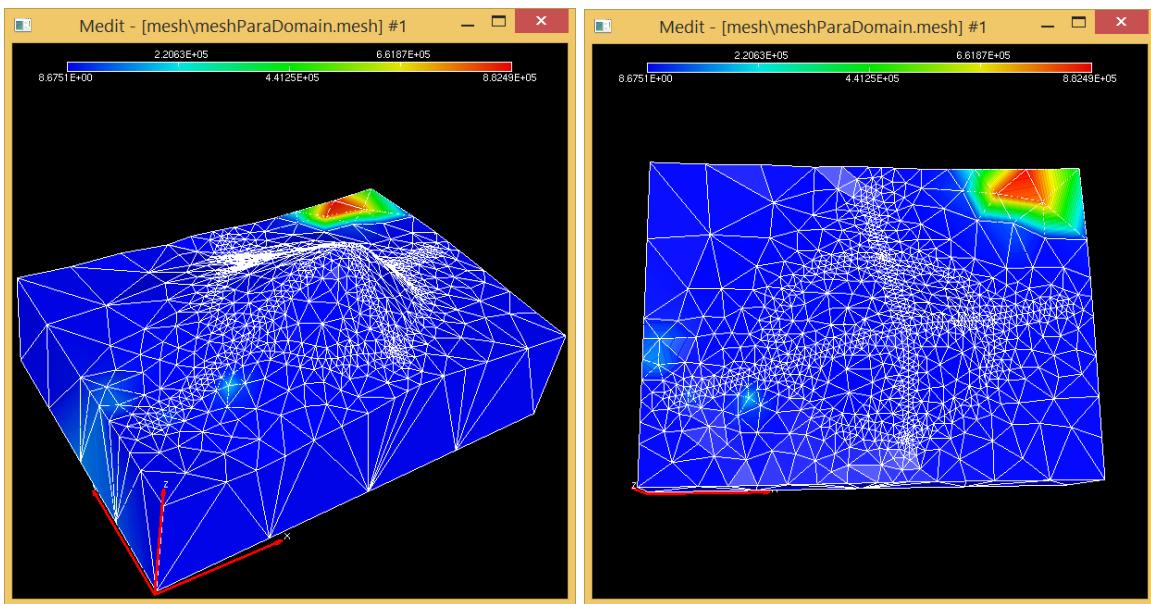
Кажущаяся разница в распределении сопротивлений при открытии результатов инверсии в программе VisIt и во встроенном визуализаторе связана с тем, что в VisIt мы открыли файл, у которого результаты связаны с ячейками сетки (тетраэдрами), а встроенный визуализатор показывает результаты, связанные с узлами сетки. Напомним, что первое представление более естественное и «правильное» с точки зрения работы алгоритма инверсии, а второе более наглядное. Чем мельче сетка, тем меньше различаются представления, связанные с ячейками и с узлами. К слову, если

бы мы открыли в VisIt файл test_3d_topoP.vtk, то увидели бы ту же картину распределения сопротивлений, что и во встроенным визуализаторе.

На поверхности мы можем наблюдать довольно резкие переходы по высоте и часто несвойственные реальным геологическим поверхностям острые углы, которые особенно заметны при приближении к «холму» и к линиям электродов, расположенным крест-накрест. Очевидно, что ошибки при задании сложной поверхности могут приводить к ошибкам в результатах инверсии. Именно для этого необходимо такие поверхности «уточнить». Делают это с помощью дополнительных точек, заданных в отдельном файле. В нашем случае такой файл есть в папке «examples» программы DiInSo и называется test_3d_topo.xyz. Оставляем все предыдущие настройки неизменными, но добавляем файл test_3d_topo.xyz, выбрав его с помощью кнопки «topo/line file:» и поставив «галочку» слева от этой кнопки. Делаем инверсию по новой, нажимая кнопку «INVERSION». На этот раз решение получается за 8 итераций, сходимость по-прежнему не очень хорошая (но лучше, чем в предыдущем случае), поскольку внутренняя сетка не была улучшена. Открываем вновь полученный файл с результатами test_3d_topo.vtk и, как и в прошлый раз, обращаем внимание исключительно на поверхность области моделирования:



Либо можно посмотреть результаты с помощью встроенного визуализатора, нажав кнопку «vis. inv.» (хотя в данном случае представление о поверхности будет несколько хуже из-за иной реализации затенения во встроенном визуализаторе), а затем, например, кнопки «m» (данные), «o» (изолинии), «A» (оси координат) и «c» («добавить цвет поверхности»):



О кажущейся разнице в распределении сопротивлений при открытии результатов инверсии в программе VisIt и во встроенным визуализаторе мы уже поясняли выше.

Мы видим, что поверхность стала выглядеть более естественно, резкие переходы и острые углы сведены к минимуму. При желании можно было бы ещё точнее задать поверхность, добавив дополнительные точки. Правило здесь предельно простое: больше опорных точек — качественнее модель. Однако переусердствовать здесь тоже не нужно, так как наша цель не в моделировании идеальной поверхности, а в устраниении грубых огехов, которые могут повлиять на результаты инверсии. Кстати, о результатах. Мы видим, что после уточнения поверхности они несколько изменились, хотя мы не меняли другие настройки инверсии. Это ярко демонстрирует, что сложная поверхность моделирования требует особого внимания от исследователя. Бывает недостаточно просто провести электротомографические измерения, запомнив точки расположения электродов. Нам нужны ещё и дополнительные опорные точки для точного моделирования поверхности, где проводились измерения.

Пример решения задачи 3D инверсии по данным, полученным для другой программы решения задач инверсии (Res2DInv / Res3DInv).

Для того, чтобы воспользоваться входными данными для следующего примера, нам понадобится установленная версия программы Res2DInv. Для этого можно скачать демонстрационную версию программы Res2DInv x32 или Res2DInv x64 с официального сайта: <https://www.geotomosoft.com/downloads.php>

После установки программы Res2DInv, её файлы с примерами располагаются в той же папке, что и сама программа. Наша цель — взять один из файлов-примеров *.dat, конвертировать его в формат *.bert и решить задачу инверсии. Напомним, что конвертер DiInSo работает только с форматом «general array data» программ Res2DInv / Res3DInv и полная поддержка не гарантируется, поэтому любой файл *.dat из папки нам не подойдёт. Узнать больше о формате «general array data» программ Res2DInv / Res3DInv можно из официальной документации, которая устанавливается вместе с программами. Остановимся на примере с именем MIXED.DAT. Простое конвертирование из Res2DInv в *.bert для 2D инверсии и из Res3DInv в *.bert для 3D инверсии не представляет из себя ничего сложного и интересного (выполните его самостоятельно), поэтому мы остановимся на конвертировании из нескольких Res2DInv-файлов в один *.bert файл для 3D инверсии. В качестве «нескольких» файлов будет выступать один файл MIXED.DAT, копии которого создавать совсем не обязательно. Вспоминаем, что для такого конвертирования нужно создать в любом текстовом редакторе файл в следующем, описанном выше, формате:

```
<имя файла Res2DInv с ПЕРВЫМ профилем>
<сдвиг по X СЛЕДУЮЩЕГО профиля> <сдвиг по Y следующего профиля относительно ПЕРВОГО профиля
(то есть относительно нуля)> <поворот на угол (в градусах) СЛЕДУЮЩЕГО профиля по ЧАСОВОЙ стрелке>
<имя файла Res2DInv со СЛЕДУЮЩИМ профилем>
<сдвиг по X СЛЕДУЮЩЕГО профиля> <сдвиг по Y СЛЕДУЮЩЕГО профиля относительно ПЕРВОГО профиля
(то есть относительно нуля)> <поворот на угол (в градусах) СЛЕДУЮЩЕГО профиля по ЧАСОВОЙ стрелке>
<имя файла Res2DInv со СЛЕДУЮЩИМ профилем>
...

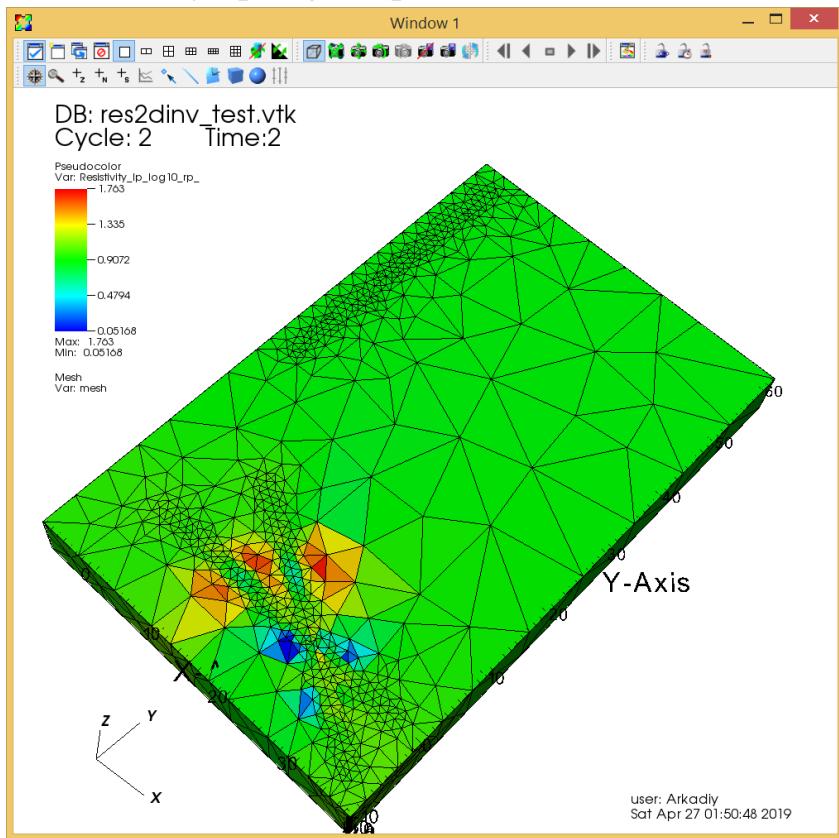
```

Создаём файл с любым именем, допустим, res2dinv_test.txt и записываем в него, например, такие строчки:

```
MIXED.DAT
5 10 30
MIXED.DAT
-5 25 270
MIXED.DAT
```

Таким образом, в результирующем файле *.bert мы хотим видеть три одинаковых профиля из файла MIXED.DAT, второй из которых сдвинут относительно первого на 5 метров по X, на 10 метров по Y и повернут на 30 градусов по часовой стрелке вокруг первой точки профиля, а третий профиль сдвинут относительно первого на -5 метров по X, на 25 метров по Y (или на 15 метров относительно второго профиля) и повернут на 270 градусов по часовой стрелке вокруг первой точки профиля. Сохраняем файл в той же директории, что и файл MIXED.DAT и открываем его с помощью конвертера DiInSo, вызываемого кнопкой  . Тип конвертирования выбираем «List of Res2DInv's to BERT 3D». Жмём кнопку «Convert» и получаем *.bert файл-результат для 3D инверсии с именем res2dinv_test.bert. В этом файле число положений установок равняется 1221, что в три раза больше, чем в файле

MIXED.DAT и это логично. Выберем файл res2dinv_test.bert, нажав на кнопку «DATA FILE:». Сначала выставим параметры 3D инверсии. Пусть «quality» — 34, «mesh grow» — 1.5 (будем задавать сетку средней подробности), «recalc jacobian?» — 1 (будем пересчитывать Якобиан на каждом шаге итерации). Теперь мы готовы приступить к инверсии. Жмем кнопку «INVERSION», решение будет получено за 4 итерации при очень хорошей сходимости. Не будем в данном примере уделять большое внимание результатам инверсии внутри области моделирования, тем более что наш тест является вымышленным, а обратим свой взор на результаты на поверхности области моделирования и на саму поверхность, где профили размещены нами довольно необычным образом. Откроем файл с результатами моделирования res2dinv_test.vtk в программе VisIt и внимательно рассмотрим поверхность (Add -> Mesh -> mesh, Add -> Pseudocolor -> Resistivity_lp_log10_rp_, Draw):



На поверхности отчётливо выделяются три профиля, два из которых пересекаются в форме буквы «Х», а третий сильно отодвинут от остальных (так как был повернут на 270 градусов по часовой стрелке). Конечно, в реальных задачах подобное расположение профилей встречается достаточно редко, но цель данного примера была показать, что мы можем собрать из набора 2D профилей сколь угодно сложную конфигурацию для 3D инверсии.

Можно приводить ещё много самых разных по сложности примеров, однако мы остановимся на этом. Если у вас есть интересный пример и вы бы хотели увидеть его в обновлённом файле справки, пришлите его на мой адрес электронной почты с подробным описанием: marinenkoarkady@yandex.ru

На этом всё! Приятного пользования программой!