GST

Geologic Mapping and Reserves Appraisal Program

The user's guide

GeoSpline Technology

Version 6.9

Tyumen
2015

CONTENTS

New functionalities in GST 6.1-6.6

- New **Print** - object has been added that allows printing of GST project content.
- When linking to another project, coordinate projections and coordinate units transference functions have been added.
- A system that recalls saves and cleans-up the temporary directory has been added.
- "Bug fixes" have been applied to the UTM projection and projection Pulkovo-42 has been added.
- New option, "**Optimize lines**", in grid tracing has been added, and a relevant editing function has been added to the "**Cover**" object.
- When printing, an area selection function via the contextual menu and the tab button has been added. Area can be deleted with the keyboard as well.
- When printing, a line has been added to mark the wells, which is indicated by a symbol and non-transparent background. A new feature for drawing the seismic survey points in a defined number and at a defined point has been added.
- In the print layout the user can set their own page size and track its fit to the size of the selected printer.
- When printing, there is a word sign in the legend, which doesn't need a prototype.
- Reading and recording to the DBF file option for the table has been added.
- A new object, **Rectangle,** has been added which serves as a mapping area, import/export area, display, etc.
- A new build method for the **Table**-object has been added – "**Get values from grid**", which allows the user to calculate the values and derivative values from one or several grids inside the polygon.
- New functions, **SetWeightMask** and **SetWeightMaskCrew,** have been added to the local weights set up script. They allow setting weight according to the polygon mask.
- A new feature providing a link to the script file has been added (script is updated automatically when passing the build-up steps).
- Now the user can imply a grid based on another grid object, when using the **Map-builder** method. It can be implied fully or according to the defined polygon. See option "**Use as fragment**" in the reference property dialog.
- A new object type "**Faults"** has been added for building grids with faults.
- An "**Intermediate object**" feature has been added to the **Table, Cover** and **Grid** objects. This feature optimizes the memory used in the project. A new object ready state has been added - "**Temporary cleaned**".
- Structure analysis is now performed for all the grid structures (both closed and unclosed which are closed only at the grid border).
- A new column copy option has been added to the table editing functions.
- The color of the selected table row can be set in the application settings dialogue box.
- The grid can be saved in *xyz* format.
- There is a new feature that helps to edit part of a line within a spline or a polyline.
- The user can put a checkpoint (when the category mode is enabled) to any line point.
- All file and project links (import and export) can be disabled with menu commands.
- The option of saving a project as a template has been removed.
- Additional options for **Faults** object: importing fault geometry and displacements from CPS files format; calculating fault amplitudes using point data; triangulation manual adjustment.
- Accelerated Map-builder method. The user can calculate grids up to 700x700 cells at once (without splitting it into bands).

- Users cannot define volumetrics parameters inside the references any longer.
- Additional features in structure analysis: structures can be found with the filter and the grid definition area; the structures are approximated with ellipses.
- When the cover is built with the use of a grid tracing procedure (build by **references** method) a new option appears: "**Trace grid region**". This allows the user to build a polygon out of grid levels.
- In the resource zones a new build method has been applied: "**Merging resource zones**", which allows users to merge several objects of this type into one.
- During resource zone building (by mouse clicking), the inner polygon parts are built automatically. This means that the user may start clicking in any order.
- If polygons cross themselves while being built, these points are marked with special markers.
- Different cover types (simple, polygon, resource zones) are displayed in a hierarchy tree by different icons. There are also special buttons to add them to the project.
- Now, it is possible to delete secondary lines without using a clipboard and to delete certain line nodes when working with the lines of the edited field of the cover.
- It is now possible to add references by mouse dragging the object from the hierarchy tree to the working window of another object.
- Importing/exporting data with sub folders is now possible.
- A new "**Make 'base' object**" option has been added. It helps to change the object build method to "**Load from file**" or "**Create with editor**" without data reset.
- Net pay thickness contour cover build method "**Tracing of a composite grid**" now doesn't require the "delta-value" to connect isolines at the boundary of the definition area.
- New grid build method - "**Composite grid by polygons**" has been added. It allows composing a net pay grid out of several grids, built for each pay zone.
- **Calculator** dialogue box display has been updated.
- There is a new cover type: "**Deposit model**", which allows calculating pay zone polygons for the layer-uplifted deposits based on the reservoir top and bottom maps and well data.
- Based on "**Deposit model**" the process of assembling resource zones is simplified.
- In the table of reserves a column with category name is generated. It helps to "tie" each resource zone to a corresponding category.
- A new "**Rectangle mode**" is added in grid and cover windows. You can copy the coordinates of the selected rectangle and paste them into the "rectangle" object via the clipboard.
- In the filling polygon dialog (drawing by the gradient or template) transparent color fill is added.
- When building a grid based on integrated data, the value of the integral can be set by the numeric attribute of the polygon lines.
- Displaying data in **Cross-section** is fully updated (see Chapter 7).
- "**Merging tables**" method now allows the sequential and parallel connections. Under the parallel connection, rows can be connected by names (numbers), or by the coordinates.
- A new option for data tables – "**Special sorting**" is introduced; it provides sorting by multiple table columns, sorting the selected rows, sorting groups of rows, etc.

New functionalities in GST 6.7-6.8

- The "**Deposit model**" object can now be used not only for tracing saturation zones, but to build grids of oil- and gas-bearing net thicknesses. For these purposes, a new building method is offered: "**Net-to-gross ratio for saturated thickness**", which can calculate the proportion of the effective saturated thickness on the basis of "**Deposit model**", with regard to the total hydrocarbon saturated reservoir thickness. The "**Deposit model**" is used as input data in

building the effective saturated thickness grids, to coordinate further the resulting grid and geometry of the collector.

- A new option is added to the "**Cross-section**": it displays the intersection points between profile lines and cover lines (faults, boundaries, lines, etc.), as well as visualizes the "**Deposit model**" in cross-section: oil- and gas-bearing zones and contacts.
- Creating covers by the references "optimized" the lines copied from other covers or created from tables. This option is enabled via references to the corresponding objects.
- Visualization of the project's tree made possible to "**Highlight**" objects, i.e. mark them by a contrasting background. You can mark all references to the specified object, the build chain, links to files, other projects, output linking, etc.
- A new **IF** function is developed for the **Calculator**, which performs multiple actions, depending on conditions.
- "**Intermediate object**" option can be enabled or disabled – for all objects in a folder, or several objects selected in the object tree from the context menu.
- Select points mode is supplemented by deselection mode that allows unselecting the desired table rows via grid and cover windows.
- "**Spline-smoothing**" method for tables received an option to reduce weight coefficients in points with high gradients. This option improves the quality of maps that use directional derivatives along the seismic profile.
- When converting coordinates from one projection into another, you can now save the specific set of parameters for a given projection as a template.
- "**Coordinates transference**" is excluded from the "**Tools**" menu, since its functionality is fully covered by the embedded **GST** functions.
- In parallel table connections, a new feature is added that allows linking data in the table to attributes stored in the other tables, upon assigning a unique number ("**Attributes**" type).
- The new **Calculator** option performs operations on string table columns (see Section 9.2.3).
- "**Post-build**" option is implemented in tables; it ensures a user-friendly format and allows sorting rows, renaming columns, etc. in the already built tables.
- The context menu of the hierarchy tree features a new option: **Sort child objects**.
- Now it is possible to enable or disable an option "**Use relative path**" for all links to other projects. It may be done via "**Options**" menu commands: "**Make all links relative**" and "**Make all links direct**".
- Importing or exporting information can use "**Coordinate keys**" – shift ("indent") the coordinate data along the respective axes.
- A more compact table storage format is implemented.
- The **Calculator** received a new feature – **Trace**, to produce under a given parameters grid isolines or a polygon bounded by the isoline.
- Cover editor introduces a new function "**Cut polygon by lines**": an arbitrary set of lines "cuts" a polygon into several polygons.
- A new method "**Automatic assembling of resource zones**" is introduced: **Resource zones** are assembled in a fully automatic mode.
- You can copy (in a standard way) the selected table rows into the **Windows** clipboard and then paste them into **Excel** or other program.
- **Chapter 9**, User manual, section Calculator, is fully revised.
- A new grid import and export format – **ASCII Irap RMS** – is added.

New functionalities in GST 6.9

- A new building method – "**Copy of other object**" is developed for objects of the basic types.
- A new building method – "**Import from the project**" is implemented in objects of basic types. Unlike "**Linking to the project**", it ensures that the imported data are saved and stored in the project and are not lost when the project is moved to another PC.
- An object of "**Grid**" type received a new, additional analytic method "**Search channels and watersheds**".
- **Map-builder**: variable anisotropy can be set in the grid references via the indirect grid.
- Cover lines can be linked to an object of "**Table**" type by a unique identifier and assigned attributes containing in the table (DBF). The DBF table can be retrieved from the "cover" into a separate object.
- Cover lines can be selected by the DBF table.
- A new logic function «**IO_P**» is added to the **Calculator;** it determines capturing points inside the polygon.
- New option of grid bilinear interpolation has appeared; it allows avoiding the false extremes in high-gradient regions between grid nodes.
- "Table" has a new option that allows you to read values from the grid without **Calculator** in edit mode and put them into the specified column.
- "Category" has an option to mark the **selected** points.
- "Grid" has a new build method "**Stability map**".
- Points creation" mode has a new option to add new point(s) to the specified table.
- You can now directly import "**Cover**" from ASCII file in tabular form, without creating an intermediate table.
- There is a new option to take into account a delimiter string when importing a table from ASCII file, i.e. string consisting of equal numbers, dividing the table on information blocks.
- New option "**Line smoothing**" based on simplified covering lines by "moving average" has been added to "**Cover**".
- You can now copy several selected objects in the hierarchy tree to the clipboard. Before, it was possible to copy one object or one folder with its contents.
- It is now possible to set the same parameters simultaneously for several selected similar references.
- Function of the "batch" references redefinition for objects and folders in the context menu of hierarchy tree has been added.
- Toolbar in the tree hierarchy now has a new button, which can give brief information of the marked object to the display field.
- There is a post-building procedure for grid mainly connected with the transformation of the grid basis from bicubic to bilinear and back.
- There is a post-building procedure for lines and polygons: filters, transformations, etc.
- ASCII CPS-3 LINES format has been added to a list of import-export formats.
- There is a new procedure for integration of grid with the bilinear basis.
- Build method "**Composite grid by polygons**" has been functionally expanded and renamed into method "**Merge grids**".

1. Getting acquainted with GST

1.1. General terms and concepts

1.1.1. Object types in GST

While solving mapping tasks and estimating reserves, geologists have to handle different types of information such as: well and seismic data, key horizons, faults, pinchout and fault lines, saturation-zone boundaries etc. Depending on the task, this data set can be reduced or considerably enlarged. Due to this, all variety of the geological information available for the program-solved tasks was grouped into several formal mathematical primitives called further as *Program Objects*. The **GST** program contains the following **object types**:

- "**Grid**" is a networked surface model. The objects of this type are represented by geological surfaces such as tops, bottoms, seismic horizons, thicknesses etc. Grids can define the fields of physical properties – temperatures, pressures, porosity, oil saturation and others.
- "**Cover**" is a set of lines and contours (closed lines) that are characterized by two spatial coordinates and attribute information. The following are related: isolines maps, fault lines, different polygons – net pay boundaries, category & license-area boundaries etc.
- "**Table**" is a set of data that is structured by rows and columns. The columns may contain numeric and string data. The table can contain well data, seismic data interpretations, arbitrary point information, reserves data, etc.
- **"Faults"** is an object, which serves for creating grids with faults. It is a numeric grid on triangles, in which nodes of field of displacements is set. This object is used as a source in the grid-building algorithm. This object is an alternative to the old method, where the faults are represented as a simple linear cover. This is discussed further in Chapter 5 "**Grid building with Map Builder**".
- "**Statistics**" allows defining a statistical relationship between columns of one or more tables, and computing linear-regression coefficients, correlation matrix, dispersion, etc.
- **"Rectangle"** is an object, which looks like a table of 4 columns and one row. It is used to build other objects by importing/exporting project data. It is discussed further in Section 2.5.
- "**Cross Section**" is an object that inherits its properties from "**Cover**" and allows creation of grid and well-data section displays.
- "**Picture**" allows users to load graphics files and use them as a graphic layer.
- **"Print"** is an object that manages the layout process (assembly and design) and makes paper copies of GST objects.
- **"Folder"** is an auxiliary object that allows grouping of other objects by their function. Separate folders can contain initial data, interim results and final results of the computations performed. Various folders can correspond to several variants of computations made with different parameters, etc. The objects of this type can contain text information.

1.1.2. Object building methods

GST objects are used to display and store geological and geophysical data in table, grid or vector modes. This data itself can be received in different ways. Below is an example based on an arbitrary gross thickness grid. These are 3 ways of getting this object:

- Import of an existing grid from a file to **GST;**
- Building grid in the **Calculator** as the difference between the top and bottom;
- Calculation grid with the **Map-builder** algorithms using the well data or digitized contours.

Different build methods for the same type of an object are grouped in the Object **build methods**. In general, each type of object is characterized by an individual set of methods, which can be divided into two main groups:

- **Object import** involves loading from a file, database or other **GST** project or building an object manually by the user drawing cover lines in the editing mode or filling in a table.
- **Build based on references –** building object upon another project objects.

The methods in the first group are called basic methods, as no additional objects are needed for their work.

1.1.3. Hierarchy tree, build chain, references

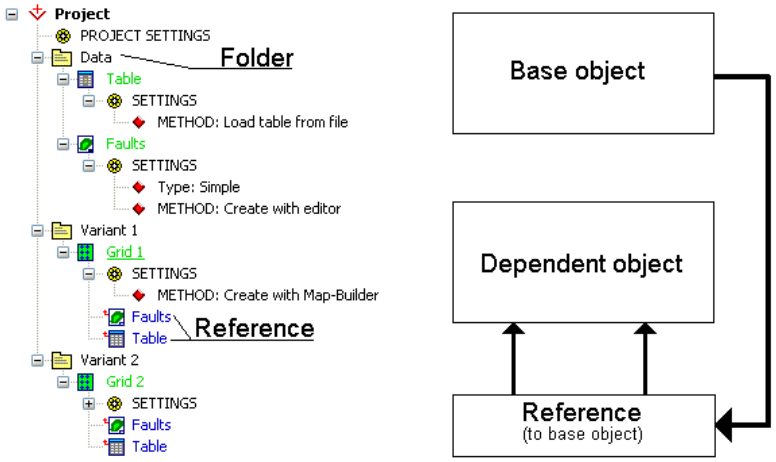
GST works with interrelated objects that form a hierarchy tree, rather than with an ordinary set of graphic layers. Let's demonstrate this with a simple example. Assume that a file contains reservoir top and bottom data. Our task is to build top and bottom maps, as well as a thickness map. A hierarchy tree will contain, at least, four objects: well-data table and three grids (top, bottom and thickness grids). A sequence of steps at solving this task will be as follows:

- Loading well data to the "**Table**" object.
- Building top reservoir grid using well data.
- Building bottom reservoir grid using well data.
- Building a thickness grid (by subtracting the top and bottom grids).

Such a sequence is defined as a **build chain**. Steps "2" and "3" can be place-traded; however, the other operations should be done according to this sequence. Since the thickness grid is calculated based on previously constructed top and bottom grids, the latter are considered to be the **basic objects** relative to it. In their turn, top and bottom grids are constructed based on well data; that is why the "**Table**" object, containing well data, is considered basic relative to all other elements of the hierarchy.

A typical example of a hierarchy tree is shown in the left figure below. The figure illustrates construction of two variants of a grid, using point data, contained in the object with the name and type "**Table**", and fault lines from the object with the name "**Faults**" and type "**Cover**". "**Table**" and "**Faults**" are the basic objects of a hierarchy since the objects "**Grid**" and "**Grid1**" are computed on their basis. Initial data and computational results are folder-grouped for convenience.

Tree elements contained "inside" of other elements will be named as **child objects** (or children). The objects with children are called **parent objects**. The folders can contain objects of any types (including other folders as well). All other objects can contain only references and settings within them.



Building the object, based on the other one, is made using a hierarchy-tree element called a **reference. References** establish a relationship between the objects. They contain instructions on how a basic object is used to build a dependent one. Schematically, it is shown in the figure to the right. The dependent object contains a reference to a basic object; here, the reference is a child. This reference is to the lowest level of a hierarchy tree it cannot have children. During the build process, the data from a basic object 'passes' through a reference in order to be used in calculating algorithms. There are two types of references:

- **Significant references** – if basic-object data is used in calculating algorithms while building dependent object. By default, these references are shown by a blue color in a hierarchy tree.
- **Insignificant references** – if basic-object data is used only as a graphic layer (for visualization), but not for calculations. By default, these references are shown by a grey color in a hierarchy tree.

Only a significant reference makes the hierarchy-tree objects dependent. From this follows a regulation on inadmissibility of **significant cross-references**. Hence, the object "**Faults**" (rf. to the figure beyond) cannot contain a reference to "**Grid 1**", since it will make these objects interdependent, however, it is impossible under the program ideology.

1.1.4. Copy of the object

A new method "**Copy of another object**" was developed in **GST** Version 6.9 for building objects of basic types (tables, grids, covers). It makes easier working with a complex hierarchy tree and implies copying data from one object into another.

Select "**Copy of another object**" and add a reference link to the copied object. For objects with graphics windows (i.e., grids, covers, etc.) check "**Copy grid, cover, etc**." option in the reference parameters.

This method can be explained by the following example. Assume that there are two options of well data stored in two different tables, "A" and "B". A set of structural maps is built on the basis of one option (e.g., "A"). Our task is to run a test calculation of these maps on the basis of option "B" and be able to return promptly to the first option, if necessary. This is possible in several ways:

1. Replace the “A” content by the “B” content and recalculate the modified objects. Significant drawbacks of this method are: the threat of losing the original information contained in the “A”, or getting confused between the options.

2. Redefine all references from “A” object to “B” object. This method is more preferable, as compared with the first approach, since the above threats are reduced significantly. The disadvantage is the need for a very careful override of *all* references from “A” to “B”, which is quite tedious in case of numerous links.

3. Create a “C” object by *copying the data* from “A” or “B”, and perform all further calculations on its basis. In this case, swift replacement from “A” to “B” requires overriding to “C” a single reference only.

1.1.5. Intermediate Object

Starting with **GST v. 6.5**, objects of **Table**, **Grid**, and **Cover** may be imparted the property of an ”**Intermediate object**“ (via the context menu of the hierarchy tree). This property might be useful when applied to objects that have no value of their own within a given project, but are rather used as input data in the construction of other objects.

A simple example: input data include a t_0 grid, *a grid of average velocities* and *stratigraphic markers* of the horizon to be mapped. The task is to build a structural surface. A typical method of resolving this task in GST would be:

- Load *input data* into project to obtain three objects in the hierarchy tree: two grids and a table of markers;
- Perform calculator procedure (multiply grids) *to build a structure map at a first approximation* (a new object on the hierarchy tree);
- Match the first-approximation map to well markers, i.e. use the *map built as a first approximation* and wells as input data for the computation of a *structural surface*.

The project contains five objects, one of which (structural map at a first approximation) is only needed to build a structural surface adjusted to well data. In case this «approximate» grid is given the property of an «**Intermediate object**», and once the final structural map has been built, the object data will be meticulously de-initialized, while the object itself is in the mode of «**Intermediate cleaned**». As a result, the project would take less RAM or disk memory, and saving and loading would take less time. Intermediate objects are subject to the following rules:

- Enabling/Disabling this property is done via the context menu of the hierarchy tree (by right-clicking the mouse).
- The intermediate object is «cleared» automatically upon the completion of any task, provided all other objects dependent on this object have been constructed. The term «directly dependent» means objects that contain a *significant reference* to a specific intermediate object.
- The “*intermediate cleaned*” object is shown in the hierarchy tree by a specific color (light green by default).
- The work window of an *intermediate cleaned object* is closed and can be opened only after the cleared object has been restored.
- If not all objects directly dependent on the «intermediate object» have been constructed, the object is not cleared automatically.
- The property of «**Intermediate object**» cannot be installed on objects loaded from file or data base, or created manually (in Editor), since these objects are basic input data.

- Automatic clearing of intermediate objects can be initiated by executing menu command «**Options → Clean intermediate objects**».

1.1.6. Object preparedness states

To build any **GST** object the user should undertake several steps:

- add the object to a project;
- select a build method;
- if necessary, add references to other objects;
- set parameters of the references;
- set build parameters;
- start a build task.

Each of these steps corresponds to a certain state called an object **preparedness state** (there are default colors used in the hierarchy tree for each state):

- **Undefined object (red).** The object is added to a hierarchy tree, but neither a build method, nor a minimal set of needed parameters have been selected yet. In such a state the object cannot participate in a build process.
- **Prepared object (light yellow).** A build method and all needed (but not necessarily sufficient) parameters and references are set. The program will allow sending this object for building.
- **Currently building or editing object (bright yellow, bold).** It implies that currently either a calculating scheme is in progress, or the user performs is manually editing.
- **Object is built (green).** All computations are successfully completed, and the user (if necessary) has confirmed that a build process is over. Only a built object can be used as a basis for further computations.
- **The object has been temporarily cleaned (light green).** The object, marked as **Intermediate** transfers to this state of readiness after it and all relative objects have been built. In order to save memory the object data is cleared and can be restored during the next rebuilding.

1.1.7. Advantages of the selected approach

The **GST** authors refused to apply technologies of a "Red Button" type, which allow solving the standard geological tasks using a developer-predetermined scenario. A basic drawback of such approaches is that it is impossible, in principle, to foresee all the peculiar features of real-world tasks a geologist deals with while performing his/her job. For example, one can't suggest a common algorithm of a structural map based on both well and seismic data. We have developed a more flexible method, under which the user himself adjusts a technological task-solution chain – from out-of-file loading of initial data up to delivery of final results. The following **principles** underlie the method proposed:

Flexibility. Having a hierarchy tree constructed, the user gets a capability to realize any task-solution scenario, including the most optimal one. The user makes a decision, based on the amount and

quality of initial data (which is often heterogeneous) and then defines operation columns of the tables, features of missing information, etc.

Repeatability. Repeatability is a major benefit of using GST, since the program saves not only initial data and final results, but also all intermediate steps, including the settings made by the user. When approaching a project that was worked previously in GST, a geologist can easily reconstruct a sequence of the operations performed.

Actuality of a hierarchy tree is provided by a correspondence of final results to the initial data and all parameters of tree derivation. Doing his/her job, a geologist frequently encounters a situation, when all initial data (well marks, seismic-data interpretations etc.) suddenly changes from that used previously in a project. At that, a hierarchy tree can be so intricate, containing lots of intermediate task-solution steps and variants, that the user could not remember all interrelations that were used. **GST** independently traces all the changes in initial data and specific settings, and informs which of the dependent objects should be updated.

Automation of task solution. This issue directly relates to all prior paragraphs. Since initial data and a final result, as a rule, are separated by a great number of interim objects, automated problem solving is very helpful. **GST** successfully solves this problem. Having all the interrelations and settings established, and believing that they are correct, the user can send the object of his interest to build, and **GST** will automatically create a build chain and successively pass through all intermediate steps of a build process. At that, the user keeps the right to control the entire process – he/she can stop a build process; start it up again with other parameters, etc.

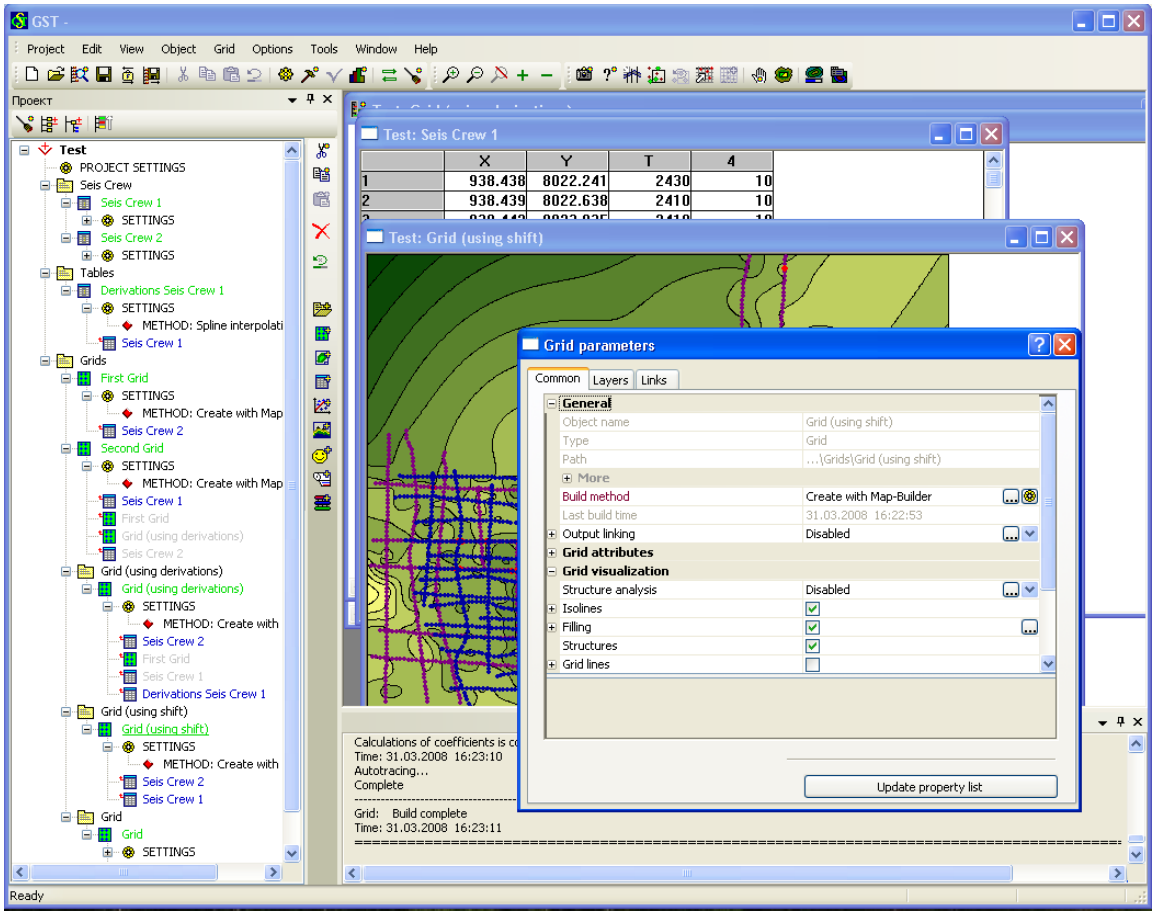
So, a hierarchy tree can contain a lot of objects (limited only by a computer resource), both interrelated and independent. No limits are imposed on a number of independent "branches" of a tree. Prior to computations of any object, the program automatically creates *a build chain*, i.e. it analyses all object referencing the basic objects of the hierarchy. Then, **GST** successively builds all elements of a chain, at readiness-level: "*prepared to build*". If one of the chain elements is *"undefined"*, a build process cannot be started up, and the chain automatically "*breaks off*". The elements which were built up previously (which have the status "object is built") are omitted, so the program doesn't waste time on their rebuilding. After the final element has been built the calculation stops and the chain breaks off.

Each object can be run concurrently through only one build-chain. **GST** traces this independently.

1.2. Interface and basic operations

1.2.1. Program layout

Having any of **GST** projects opened, you will find a picture that is identical to the one in the figure below. There is a child window at the left side of the main window of the application, which is called an *objects' hierarchy tree* (or a hierarchy window). It shows an interrelation of all data used in the project (initial, interim and computational results). Below there is a output window, which shows the task-solution procedure.



The other child windows are called object's work windows. Each object of the project relates only to one work window, where it is graphically displayed. With each work window other interface-control elements (menu, toolbars, property dialogs, etc.) are related.

1.2.2. Hierarchy window

By default, a hierarchy window is found at the left side of the application main-window; however, it can be shifted to any other application. If it is a double-monitor computer, a hierarchy window (and the property dialogue box) is worth having on one of the screens with the application main-window on the other. There is a vertical tool box in the right part of the hierarchy window that helps add and delete objects from the project. The horizontal tool box in the top part of the window serves to display the hierarchy tree.

The objects' hierarchy window, in the form of a tree, clearly illustrates the links between all objects of the project. Their names, type (icon), as well as a current state (color) are marked with different colors. The elements' color adjustment can be user-performed (via a dialog called by menu buttons "**Options→ Applications Settings**"). The default colors are as follows:

- **Red** –the object is undefined.
- **Yellow** - the object is ready to be built.
- **Light yellow** (solid italics) – build or edit in progress.
- **Green** – the object is built.
- **Blue** is for a significant reference.
- **Grey** is for an insignificant reference.

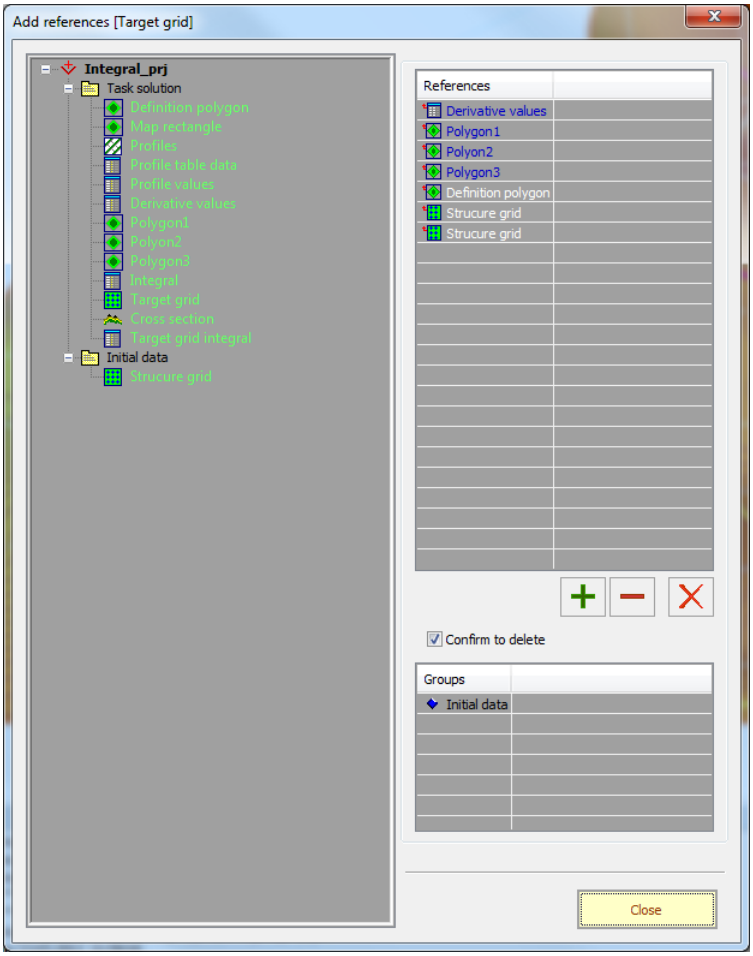
The object-build parameters dialog-box shortcuts and group shortcuts are displayed in the hierarchy window starting from **GST 5.0**. Now the user can change the build method or parameters without opening the object's child window.

Navigation using a hierarchy tree. Double left-button clicking on any of the objects shown in the tree activates a work window of this object. A double click at a reference invokes a reference property dialog that allows the user to set all parameters' object-to-object link. The user can move from an object reference to the work window of the object with the use of the contextual menu **"Move to object"**. The name of the active object is underlined in the hierarchy window.

Reference creation. To create a reference, it is necessary to left click at the object (to which a reference is being created), and then, not releasing it, move this object to the one that will contain the reference. Starting from **GST 6.6** the reference can be also added by dragging & dropping it from the hierarchy tree to the object's work window. The program does not allow reference creation in the following cases: the object cannot actually contain any reference inside it (for example, there is no reference to a reference); the object cannot contain reference of this type.

In a case where a hierarchy tree is rather intricate and bulky and it is difficult to create a reference using a mouse, a reference can be created in a different way. Activate a work window of the object that should contain a reference, and perform the main menu's command "**Object→ Add Reference**". An "**Add references**" dialog pops-up below the figure.

The left window-list enumerates all objects of a project. To add a reference, it is necessary to select the object needed from the list and click the button "+" (Add). A name of the reference added pops-up in the list on the right. To delete a reference, the user should mark out it in the list and click the button "-".



Moving objects within a tree. To place an object in any of the folders, it is necessary, as it is done for a reference, to drag an object with the mouse to the icon that shows the folder needed. The same can be performed through the menu command "**Object→ Move Object"** or the contextual (right-click) menu "**Move to folder**". A dialog with a list of folder pops-up.

If it is necessary to move the object within one folder (for example, from first to the last place), this can be done with the left button of a mouse and the "**Shift**" key, pressed simultaneously or by turning on the "**Move object mode**" (toolbox on the top of the hierarchy tree).

Multiple select. In the previous **GST** versions only one object in the tree could be chosen at a time. Starting in version 5.0, multiple choices can be made. The user can choose more than one object by:

- Pressing the Shift button several successive objects can be chosen at once;
- Pressing the Ctrl button several objects can be selected.

The selected objects can be deleted, moved, cleared, and sent to build.

Note: Only objects of the same level of the hierarchy tree (same folder, within the same object) can be selected simultaneously.

1.2.3. Contextual (right-click) menu of an hierarchy tree

Contextual menu of a hierarchy tree is invoked by clicking of the right button of a mouse at any of the tree elements. The menu contents depend on the type of an element chosen (object, reference, properties shortcut) and on the number of selected elements.

Object information. This option displays basic information about the object in the output window: build method, location in the tree, ID, number of references, etc.

Add object. Adding new objects to a hierarchy tree of a project is possible through either a contextual menu, or by pressing one of the keys in a vertical toolbar. An initial position of the created object in a hierarchy depends on what object of the tree has been selected. In case of selecting a root object (the project itself), the new object is added directly to it. If a folder (or any child object of this folder) is selected, then the created object appears within it.

Add reference (group). This feature allows adding a reference or a group of references to an object. If there is a huge hierarchy tree, this feature can be very useful.

Rename. One can set actually any names of hierarchy elements, in Russian or in English, consisting of several words, etc. To rename an element, one should left click at the element selected (or press down the "**Ente**r" key). In the opened editing window, type new name and then click by the mouse button at any point of the hierarchy window (or press **Enter**). Elements can be renamed multiple times.

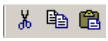
Move to folder. The user can move one or several objects from one folder to the other.

Highlight: i.e., mark objects in the hierarchy tree by contrasting background, for easy navigation. All references to the selected object (significant and non-significant) and the building chain – basic and dependent objects can be highlighted. In the selected folder or the root element of the tree, you can highlight all linkages to files and projects, as well as the "**Output links**". You can cancel 'highlight objects' with  button on the toolbar at the top of hierarchy window.

Sort child objects. With this option you can sort objects by type and name within the folder or in the entire project. References to objects can be sorted likewise.

Clean. The user can clear all the build results for any object, folder of project.

Delete object. While deleting a folder, you delete all its content, and deleting an object, you delete all references to this object.

Copy, cut, paste. An object can be copied or cut to a clipboard and pasted to the hierarchy tree. Execution of these operations is identical to their execution under other Window-programs, being performed using the buttons  of the hierarchy tree toolbar or via a contextual menu of a hierarchy tree. The tree element should be selected before these operations are being performed. Copying and pasting an object includes all its "children" and setup parameters. Copying procedures are very useful while repeating operations of the same type, for example, when studying various variants of building, which differ in details. Starting with version 6.9.2, you can copy several objects to the clipboard.

Protect Object. This is used for the selected object and its "children". If this option is built-in, the program asks for the user-confirmation each time he/she tries to delete or change the object.

Intermediate object. The option sets or clears the "Intermediate object" feature. (See details above).

Build. The contextual-menu command "**Build**" or button  in the toolbar generates a *build chain* for the given object and consequential computation of all of its references.

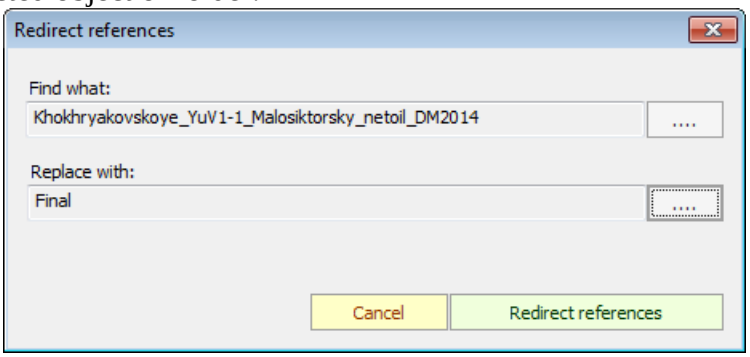
Make 'base' object. This feature allows the user to change the build method to "Load from file" or "Create with editor" without resetting the object data or that of referenced objects.

A contextual menu for a reference differs slightly from an object's menu. It allows the following additional operations to be performed.

Reference properties. Similarly to double-click on the reference: shows its properties dialog. Starting from GST 6.9.2, you can set the properties for several selected similar references at a time (without creating a group of references).

Move to object. This allows a work window of the base object to be activated.

Redirect reference. This command can redefine a reference from one base object to any other object of the same type. In this case, all reference parameters remain the same. References redefinition option is in the context menu of the object and folders. In this case, you must specify two objects in the dialog shown below: "**Find what**" and "**Replace with**". The operation will be applied to all references contained within the selected object or folder.



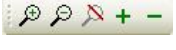
This command can be very handy when you have to carry out a number of similar calculations. For example: to build ten maps by ten different tables. In this case, you can build one map, duplicate it ten times in the hierarchy (option "copy"), and then redirect references to the relevant table and recalculate.

1.2.4. Object's work window

Each object has its own work window where visualization, editing and other operations are executed. They can be subdivided into two groups by a type of the information displayed:

- *Textual visualization (text windows).* These are the folders' work windows, where the user can enter any text information to, and table windows with accessible string and numerical information in a textual mode.
- *Graphic visualization (graphic windows).* These windows display maps, cross-sections, graphs, etc. These are the work windows of all other types of the objects.

Only relevant object data can be displayed in the text windows. In the graphic windows additional object layers (references) can be displayed. Further on this paragraph will deal with graphic windows.

A visualization **Scale** is selected by default in such a way that window client-area will include all object's data. A scale can be changed with *zooming* functions that are standardized for graphic windows of all types. To zoom in on an image, the user, while holding the **Shift** button and pressing down the left mouse button, select the zoom rectangle. It is possible to perform this operation sequentially. To return to the previous zoom-position, user should click the right button of a mouse with simultaneous pressing down the "**Shift**" button. The more accurate scale value sets with menu commands: "**Cover→ Scale**", "**Grid→ Scale**", etc. The zoom can also be changed with the toolbox buttons  at the top part of the main window.

By enabling the "**Prompt mode**" (button in the toolbar) user can receive information about the graphic layer objects displayed in the graphic window.

The commands "**Move to layer**" or "**Compare point**" allow moving from one object's work window to another object's work window.

Saving of a work-window **image** into a graphics **bmp**-file or a vector **emf**-file is possible using the "**Photo-mode**" (the button in the toolbar). If this mode switched on, the user should double-click the mouse left button at client area of the object's work window. To save a picture within a rectangle, user must (pressing a left mouse button) select a needed area.

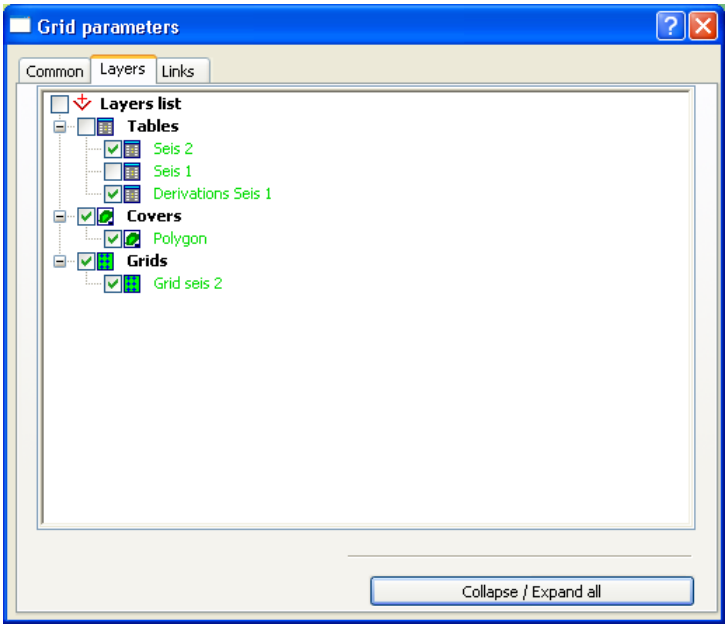
Clicking the right mouse button on the work window invokes a contextual menu that partially duplicates the basic menu and toolbars.

1.2.5. Object parameters dialog

Dialog of properties (or dialog of parameters) of an object is opened by double-clicking on the work window or by executing the menu command «**Object → Object Parameters**». Objects of all types, with the exception of **Folder** have property dialogs. Depending on the object type and build method, the dialog may have a different number of tabs. Let us review tabs that are typical for the majority of occasions.

The tab **«Common»** contains practically all object properties. The tab contents depend on object type and build method. The category «**General**» stores all main information (name, path in hierarchy tree, status etc.), a box identifying the build method, and object export/import parameters. The presence and contents of other categories depend on object type. Boxes of exclusively informative property are colored grey, and changeable boxes, light green. The tab «**Common**» will be discussed in greater detail in subsequent sections.

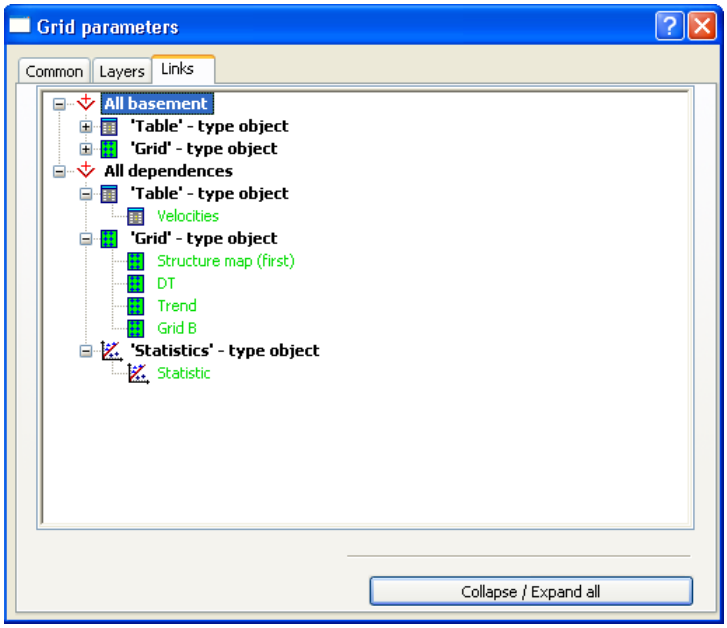
The tab **«Layers»** pops up in the property dialog of those objects whose work windows can show information on additional underlying layers. The rule is: «**Object 1**» can show in the graphic window of «**Object 2**» only if «**Object 2**» has a reference to «**Object 1**».



The list of layers accessible to view is placed in the tab as a tree where user can turn on/off drawing and change the sequence of showing layers by moving the mouse. The list is automatically supplemented depending on the presence of references of a type.

The tab **«Links»** shows the place of an object in the project hierarchy. It shows a list of basic objects (meaning those objects that were directly or indirectly involved to build a given object), as well as of all directly or indirectly dependent objects.

This tab is worth opening before deleting or clearing an object in order to make sure that the move will not corrupt valuable information. By double-clicking on an element of the tree, the user can activate the work window of the relevant object.



1.2.6. Menu

Menu items correspond to the object, the working window of which is currently active. Let's consider the menu items that refer to the project and the program, as well as general menu-items for all the objects.

- **Project.** It provides an access to the functions of: creating new-project, loading and saving of an entire project, adding projects', etc.
- **Edit.** Depending on a type of the active object, it provides different capabilities of dealing with clipboards and use of an undo operation.
- **View.** An interface language of the program is being selected here (Russian or English). It switches the toolbars on and off and states, as well as a hierarchy window. An automatic redrawing property is set here.
- **Object.** This menu item combines all general operations made with the objects, such as adding, deleting, startup and termination of a building process, calling a parameter dialog, etc. Use the function "**Break build process**" in cases when due to some reasons an object building process cannot be terminated, while this object is still in the active chain (shown with solid italic print).
- **Options.** Combines all functions that concern general project settings and applications, as well as memory-work functions.

Right clicking of the mouse at an object's work window, user can invoke a contextual menu whose commands will be described later.

1.2.7. Toolbars

Starting in **GST 5.0** the toolbar configurations have been changed a little. All the functions connected with a hierarchy tree are now situated in special toolbars linked to the hierarchy tree. There is the toolbar with the visualization parameters and the mouse cursor modes at the top of the hierarchy window; the toolbar with object's add/delete/restore/copy functions is on the right.

The toolbar configuration depends on the type of an object which work window is active. On the left, there are buttons  responsible for creation, loading and saving of a current project as well as for the data import and export. Then there are buttons that allow working with a clipboard  - they are active for the objects supporting these operations. For example, if a **Cover** object's working window is activated, one can copy a cover line into the clipboard and insert it to the working window of this or any other cover. The **Undo** operation (4th button) is for returning to the initial state, while editing data of some objects (rf. to **Table** and **Cover**). The buttons  serve for object build process management (left to right):

- Object build settings;
- Start build;
- Break build;
- Complete build (edit);
- View build statistics (if this option is supported).

The first of the buttons  turns on/off the automatic redrawing mode, the second carries out the command "redraw the contents of the active window". The scaling toolbox  is active for graphic windows. Other toolbox instruments deal with specific functions and will be described in the following chapters.

1.3. Working with Projects

A set of initial, interim and final data displayed in a hierarchy tree, as well as all interrelations between the objects are combined into a notion "**Project**". A project contains all parameters and settings needed for running through a build chain of any object. Any **GST**-operation is performed under one or another project. Even if the user wishes to review file-table contents using the program software, it can be done only within the project that contains, at least one of the **Table** objects.

1.3.1. Loading and creation of a new project

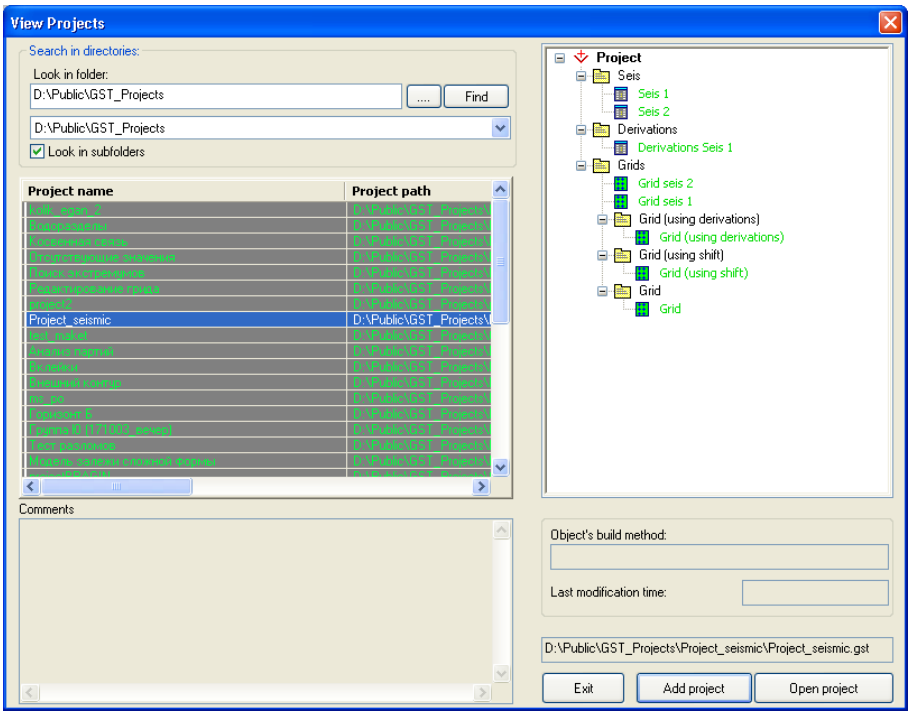
Launch the **GST** program. In the "**Project**" menu execute the command "**New**", if it is necessary to create a new project, or command "**Open**", if you want to open an existing project.

To load the existing project, user should select a file with "**.gst**" extension. If loading is successful, user can find a project's tree in a hierarchy window, otherwise the program can give an error message that means the following: 1) a selected file is not a **GST** - project's file, 2) you are using the old version of the program, 3) the project's directory content is invalid.

While creating a new project, the only element that appears in a hierarchy window has default name "Project". Next, user must add all objects needed for further work, set the references between the objects, etc.

1.3.2. To search and view projects

By pressing button on a toolbar or by performing a command "**Project → View Projects**", user can invoke a dialogue box shown in the picture below.



There is an input window in the "**Search in directories**" control group where the initial directory should be set and a "**Find**" button – to start the search. There is a list of frequently used directories in the drop-down list situated under the input window. The search begins by pressing the **"Find"** button or when changing the selection in the list of directories. The search begins with the initial directory taking into account all the subfolders.

The search results are displayed in a list below. The first column of the list contains the name of a project (this is the name set in the root of the hierarchy tree). If the user hasn't given a unique name to a project and left the default one (e.g. "Project"), then a name of a gst-file is displayed in the first column. The second column contains the whole path to the project main file (*.gst - file).

Information, contained in the work window of the root element of a hierarchy tree is displayed in the "**Comments**" window. For the projects saved in version 4.3 and later there is a hierarchy tree itself as it is displayed in the main program. The method and the date of the latest build can be found here when selecting the objects of a tree with a mouse.

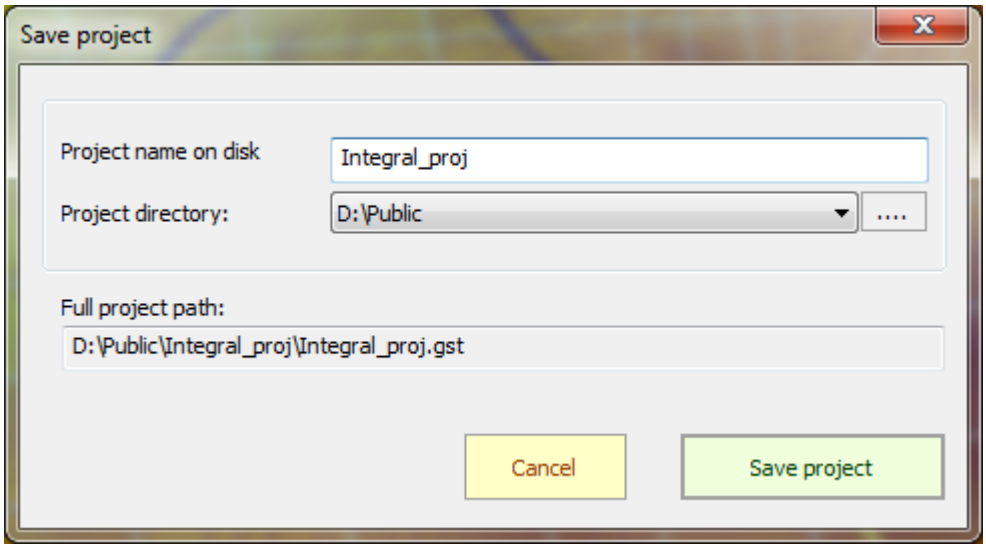
Thus, having selected a necessary project from a list the user can view its contents and open it by pressing **"Open project**" button or add to the existing project by pressing **"Add project"**.

1.3.3. Saving a project, saving under new name

To save a project, use menu commands "**Save**" and "**Save as**". If project saved under a new name, or if it is the first saving, the dialog appears where user should type the name of a **GST** - project's file.

Under this file name the program creates a directory on the hard disk, where all other information is written to. The project directory contains a start file with "**.gst**" extension, as well as files and folders for storing the objects. Each object is saved in a separate folder, with its hierarchy nesting being

inherited. By default, the program suggests to save the project in a directory standing first in a drop-down list of a dialogue box, which is refilled with the last opened or saved paths. Once the project's name and path have been defined, press **"Save".**



ATTENTION! The project is saved in a binary format, which can be read only by GST. *Don't try to change the contents of the project directory (rename files, modify their contents) as this may lead to errors during the load.* The user can change *only* project's directory name or a name of a *gst* load file. All other file and directory names are generated by the program and can't be changed.

There are no limitations for moving a project within a hard disc or from one computer to another (excluding the cases when there are links to files or other projects), since a project contains all information needed for the work. A project should be moved together with the folder that contains a load file ("**.gst**").

1.3.4. Uniting projects

In case user needs to use a once saved branch in another project, it can be done by executing menu command «**Project → Add project**», and then choosing the saved file of interest. This command can integrate any two or several projects. The project (or template) that is added is copied to the new project and appears on the hierarchy tree as a separate folder containing all data.

Upon adding a project, user can «build» it in the task resolution technique and add references it needs by the reference re-setting function.

1.4. Data Import/Export

There are three ways of importing data into GST:

- Load data from file;
- Read in a database (realized by reading tables in the **Oracle** data base);
- Link to data of another project.

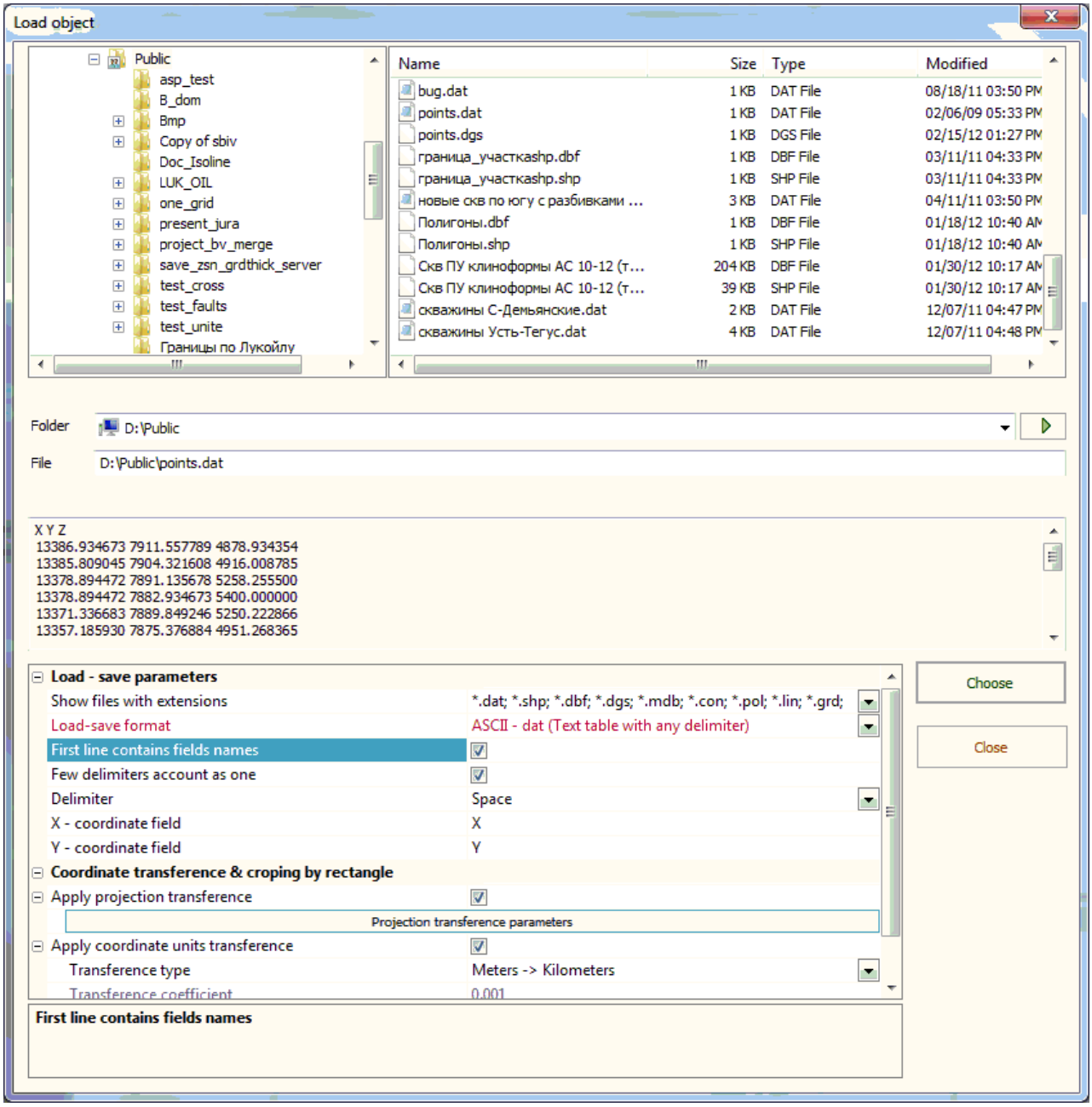
1.4.1. Importing data from file

To load data from file users have to select the build method «**Load from file**» or by execute menu commands: «**Table → Load table from file**», «**Cover → Load cover from file**», etc. The execution of these commands automatically sets the build method to «**Load from file**» for the object. The interface of importing objects of the main types is standardized and realized in the load-save dialog (see below).

The dropdown list «**Folder**» contains the path of the last previous import/export, the window below shows the file contents in a textual format (the first 100 кВ), and the parameter window in the lower part of the screen contains specific parameters of file reading: format, delimiter, parameters of projections and coordinates transference, etc. This list of properties is renewed automatically, depending on the chosen loading format.

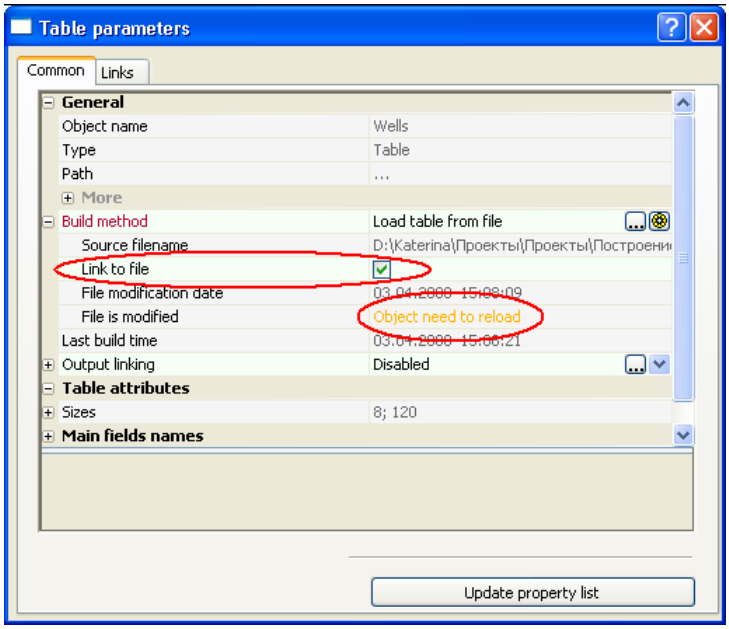
Linking to file. Objects loaded from file store the path of data transfer, but data is copied to the project directory into an internal **GST** format and exist independently of their source. In other words, the project will exist and operate appropriately even if all files from which objects are loaded are deleted, or the referencing project gets transferred on another machine which has never had and does not have these files. This property of **GST** is very important for the transferability of projects from one work station to another.

But in a number of cases, especially where the software is a link in a technological chain, there is the acute need to track all changes in files that contain input data. For instance, a table is a basic object in a project and contains well marks, and many maps are based on this table. The input file for the table is result of different working software and is regularly renewed. In this case, **GST** needs to be able to track these changes and advise the user to re-build all objects that are dependent on the table.



In **GST** this object is realized as «**Link to file**». It can only be applied to those objects, which have the build method «**Load from file**» set up; by default this option is disabled. It can be turned on/off in the tab «**Common**» in the object property dialog.

When the "**Link to file**" is enabled, while loading a project, the software checks the date of file modification against the date of its latest loading. If the dates differ the relevant message is sent to the display field and the dialog shows a note indicating that the file has been modified, and that the object needs to be re-built.



Modifications are checked and loading is performed automatically (in case there have been modifications) when an object dependent on a linked-up object is about to rebuild. If, for instance, the user wants to rebuild a structure grid, **GST** will first check the timing of modifications done to the well marks table, reload it from file as needed, and perform all intermediate operations needed to rebuild a structure grid. To check the status of all objects linked up to a project, user executes the menu command «**Options → Update links**» or clicks on the toolbar. In this case, all linked-up objects (as well as all objects dependent on them) whose saved files have been modified will be cleared and the user will have to recalculate them.

Linking to files can be turned off for all objects simultaneously by executing the main menu command «**Options → Disable file links**».

1.4.2. Data export

The objects data is saved by executing the commands: «**Table → Save table to file**» etc. At that, a dialog pops up that is analogous to the data import dialog where the user specifies the path and parameters of saving the object data to file.

Output linking. In some cases where the number of objects to be saved is relatively big (for export to other programs, for instance), while the procedure is often performed (for reason of regular project amendment), the impact of human factor is inevitable (saving to a wrong file, failing to save, etc.). For this eventuality **GST** features an option named «**Output linking**». The term means that user saves selected objects simultaneously along preset paths.

The **Output linking** option is disabled by default, but it can be enabled in the «**Common**» tab of the parameters dialog. In this section user may by specify all export parameters.

Objects are saved to export files when the user executes menu command «**Options → Write output linked objects**» or clicks on the toolbar. To automatically save data of one specific object, user executes «**Object → Write output linked**».

Output linking is only possible when the object has the relevant option enabled and when the object is built. Starting from **GST** version 6.6, auto-upload of all selected Project objects saves only those modified later than the modified export file, i.e., if the object is not rebuilt, the export file is not overwritten. The export process of output linking objects is reflected in the output window.

The **Output linking** option can be disabled for all objects in the project simultaneously by executing the main menu command «**Options → Disable output links** ».

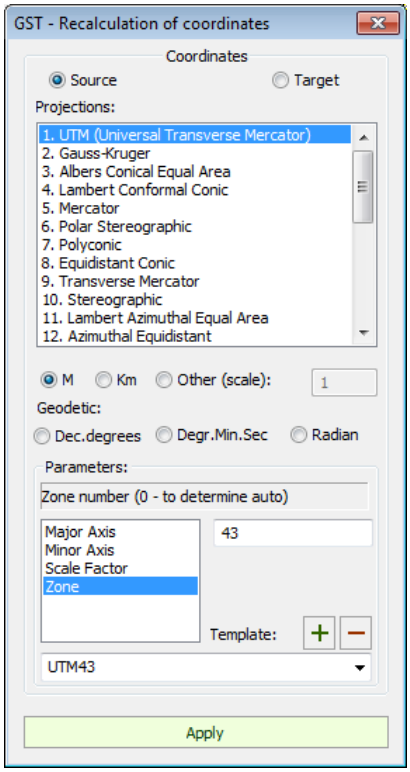
1.4.3. Projections and coordinate units transference

While loading objects from file and for the **output linking** option **GST** will remember all loading and saving parameters (file format, delimiters, flags, etc.). Beginning with **GST v. 4.1**, the coordinate transference from one projection to another, as well as coordinate units transference is included in to the loading / saving process. Users can:

- Use input data saved in differing projections; projection transference performs automatically during loading process;
- Choose user-friendly coordinates units. Thus, solving gridding tasks is easier when coordinates are measured in kilometers, rather than meters (setting weights on derivatives);
- Choose the projection and units for saving in to output file.

In the load/save dialog (category «**Coordinate transference & Cropping by rectangle**») features options «**Apply coordinate keys**», «**Apply projection transference**», «**Apply coordinate units transference**» and «**Load / Save data by rectangle**» are accessible.

The first option allows user to recalculate input / output data from one projection to another. Clicking «**Projection transference parameters**» activates a dialog where user sets up parameters of recalculations from the input projection to the target projection.



If one of the projections is geographic coordinates, the user needs to set up the format for their input/output to radians, decimal degrees or degrees-minutes-seconds. If the input data is in a rectangular projection, the user needs to specify units of measurement – meters, kilometers, other.

After a projection is recalculated (into a rectangular system, the result will always be output in meters, and the coordinate units will have to be recalculated, as needed, in this case.

Recalculation of coordinates is supported in 27 projections. Each projection with specific parameter values can be saved as a template. The number of templates is unlimited. Thus, the user can "bind" the program to a set of different geographic areas, with the optimal set of parameters for each area. The default parameters in all projections refer to Western Siberia. In this case, the template is "*Not defined*". Creating templates is not required by the program, just select the desired projection for the source and target coordinates and change some parameters. The program will save the recent changes. Templates allow creating and saving several parameter options in one projection.

The template is created with "+" button (Add) and it receives the selected projection with given parameters. The template name can be changed. Projection parameters can be corrected but this template can not be assigned another projection (any other projection is linked to another template (or the set of templates), or to the string "*Not defined*". A template selected from the list can be deleted ("-" button), the string "*Not defined*" can not be deleted. By activating the option «**Apply coordinate units transference**», the user actually multiplies coordinates by a certain scale multiplier, which in the case of translating «**Kilometers to Meters**» is 1000. Another multiplier needs to be input for other units. If the option is disabled, the units remain unchanged.

The option «**Load / Save data by rectangle**» allows user to load/save data within the specified rectangular area. Coordinates of the rectangle are input in the target projection and measurement units. A **Rectangle** object can be used for a rectangle of import/export, if it is added as a reference. Objects of different types are managed by different crop methods (conditions). If the rectangle and a *grid* do not cross, the latter is input with no changes. Every line of *cover* may be either input/saved in full, if at least one of its points gets into the rectangle (by default), or it may be cut off with only its internal fragments saved. In the latter case, user needs to additionally activate an option «**Cut lines by rectangle boundaries**». The data of *tables* may be cut off either «point-by-point», or grouped together in a column. In the latter case, the data of the whole group is input/saved, if at least one point in the group belongs to the rectangle.

If any coordinate transference option is applied for a table, user will need to specify the names of fields containing coordinates (to load, the name is input from the keyboard, and to save, chosen from a combined list).

Coordinates may be recalculated after having been loaded. To do this, the user executes the command «**Object->Transfer coordinates**», following which specifies recalculation parameters in the dialog. The command may be applied to the whole folder, when coordinates are recalculated for all objects in it.

Note! After reading from file or database and before saving to file, the process of coordinate conversion is done as follows (in case all relevant options are on):

- First coordinate keys are applied: the corresponding shifts are added to the source coordinates.
- Next – a projection conversion where user specifies measurement units for the input coordinates (meters, kilometers, other);
- When the projections have been recalculated (always into metric system), the converting coordinates measurement units process starts;
- Once coordinates have been recalculated, the data crops by chosen rectangle boundary.

In data export process, the coordinate keys are applied after conversion of coordinates and are specified in the same measurement units as those in the exported file. The sequence of actions is as follows: Recalculate the projection, convert measurement units, apply coordinate keys, cut along the rectangle.

1.4.4. Loading data groups

There are tasks that demand massive data loading (for example, hundreds of tables). To load each table individually is time-consuming and can lead to errors. To simplify loading, the **GST** provides data-group loading. A data group is a set of objects of the same type (tables, grids, covers etc.) that have an identical structure (the same table-field names, etc.). A data group can be input by executing the relevant command in the context menu of the hierarchy window «**Add object->Load Group**» or by clicking on the vertical tools bar.

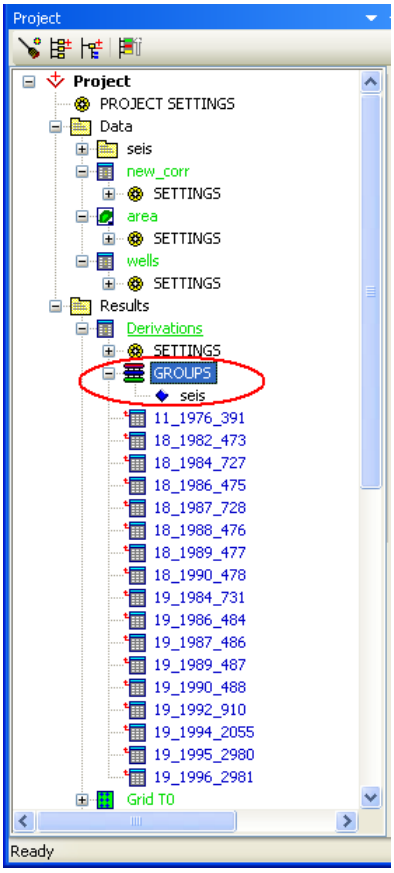
In the standard **GST** objects import-export dialog user specifies the load directory. All other saving parameters are set up as is done in order to import files.

When all parameters have been set up, the user may initiate loading. The software will try to load all files included in the chosen directory (and sub directories) into the relevant objects. If there is a file of a different type, a reading error may occur. It is therefore necessary to make sure that a directory contains files of one type.

If the loading is successful, the hierarchy tree displays a **Folder** object together with the name of the directory being loaded and containing objects along with the names of the relevant files. The group loading process is displayed in the output window.

1.4.5. Operating data groups

Suppose there is a directory named «Seis», which contains table files of seismic data. Having made the steps described above, the user obtains a folder «Seis» in the project tree, which contains **Table** objects with the same names as the files.



In **GST v. 5.0** and later versions adding a data group as a reference is easy: move the folder to the object as when adding a reference and get inside the object, under the tag «**Settings**», a tag named «**Groups**», and inside it, a tag bearing the folder name. Also, there will be a reference created for each object. «**Groups**» may contain any number of tags. References to objects stored in the sub folders are not added.

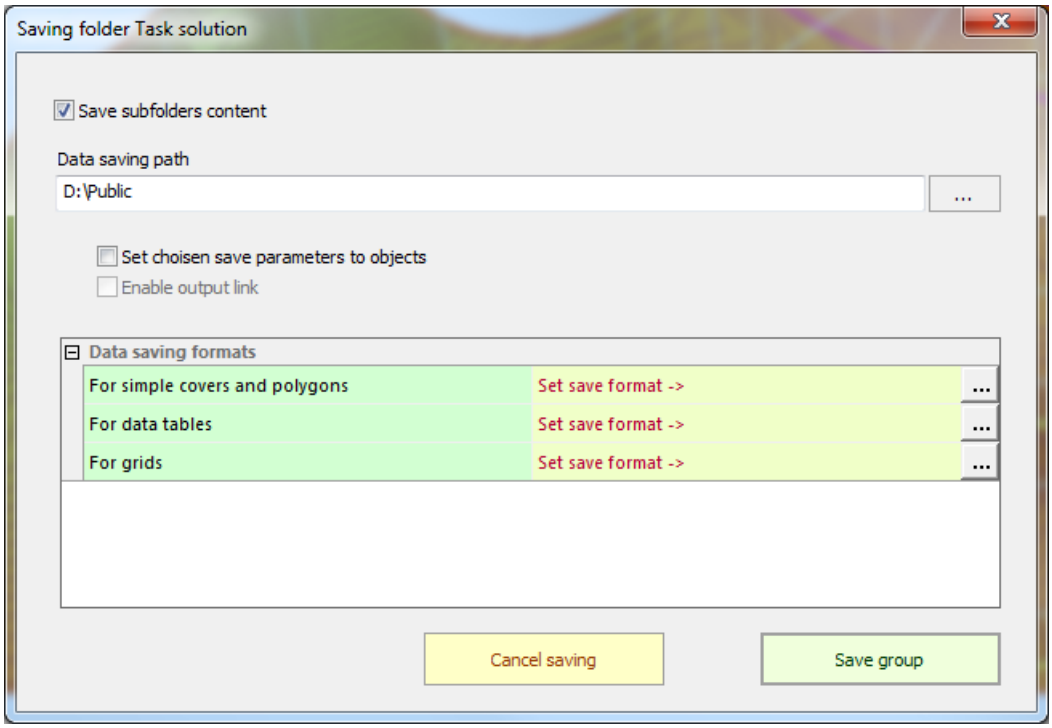
Double-click on a reference to activate a reference property dialog; double-click on the tag of a specific group to also activate a property dialog, but the parameters set in it will apply to all elements of the group. The group elements are all objects stored in the relevant folder references to which are located inside a specific object.

Executing context menu commands performs all delete/add operations with groups. Deleting a group deletes all its elements, i.e. references whose objects are stored in a certain folder.

In case an object contains a data group, the group is shown on the tree in the tab «**Layers**» of the object property dialog where user may turn on/off the drawing of underlying layer groups.

1.4.6. Saving data groups

A data group can be saved by executing a context menu command («**Save group**») in the hierarchy window by right-clicking on the folder. A dialog like the one below pops up:

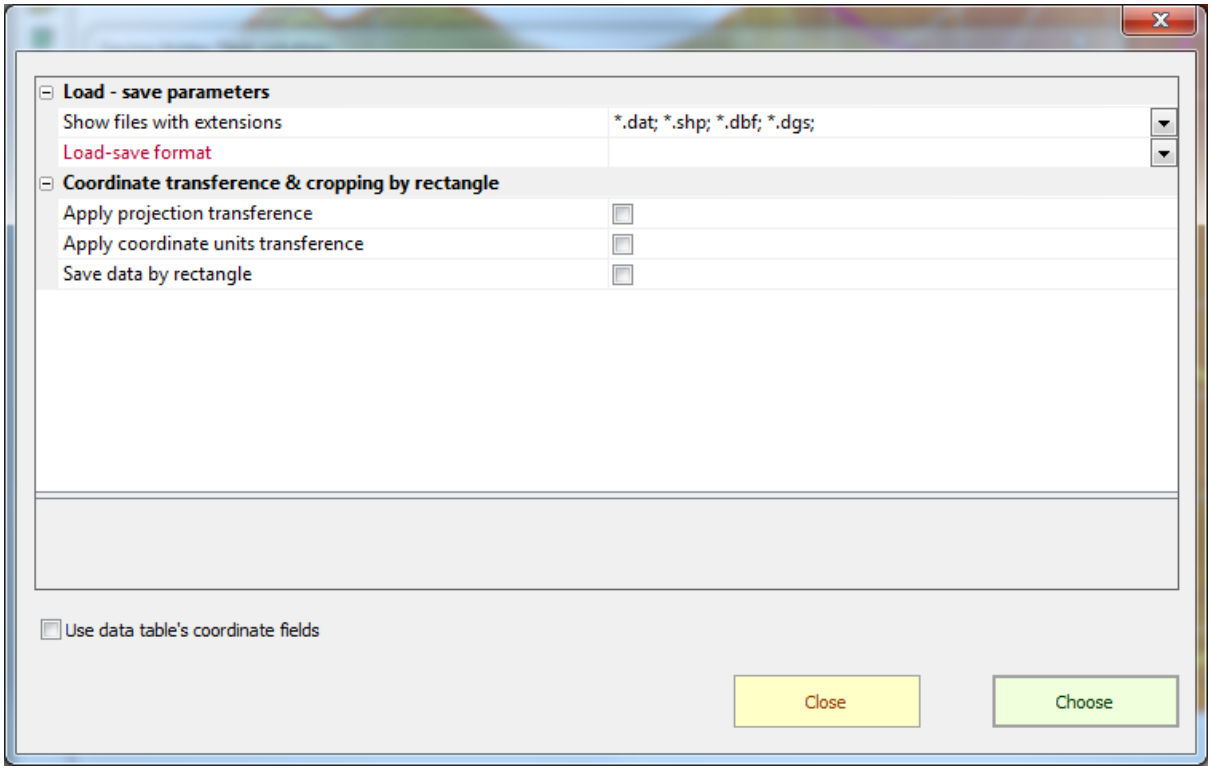


In this dialog user needs to specify the export directory name and other parameters. An option is enabled by default that allows saving subfolders contents. The user can activate an option that sets up saving paths and formats selected to be on by default, as well as activate object automatic output.

In the lower part of the dialog, the user specifies objects export formats. Formats are set up in the dialog below. User sets not only export format, but also coordinates transference parameters that are established the same for all objects in a group as needed. In case a folder to be saved contains tables and coordinates transference is enabled, user will have to specify the names of coordinate fields. They can either be specified the same for all tables in a dialog or the option «**Use data table's coordinate**

fields» can be activated. In the latter case the coordinates will be converted in those fields which are specified in the property dialog of the relevant table.

User can export data of the entire project by executing the command «**Save group**» for the tree top in the object hierarchy. When saving is successful, **GST** creates a directory with the name of the exported GST folder, as well as all sub-directories, if the relevant option is on. All files are named as objects.



Attention! If there are objects with the same names in the saving directory, they will be written over with no warning!

1.4.7. Import/Export Formats

All objects of the main types in **GST** (Tables, Covers, Grids) can be loaded from files, as well as saved to them. Listed below are read/save formats of objects of these types.

Table:

- **ASCII – dat (read/save)**. – A text data file with any delimiter. The first line in the file may content the field names; the number of names should correspond to the columns number.
- **Shape – file (*.shp) (read/save)**. Can contain both tabular data and lines (polygons).
- **Dbf – file (*.dbf) (read/save)**. Each record in a file converted into a table row.
- **Internal GST** format **(*.dgs) (read/save)**. A binary file readable only in **GST** and is intended for fast exchange of data between projects.
- **Files MS Access (*.mdb) (read)**. Enables reading tables from Access base (so far in Office 97).

- **Files RMS Internal Points (read)**. Files of storing point data used in Roxar.
- **Line files ASCII Irap Classic (read)**. Storage files for lines used in Roxar. The line data are converted to a tabular format (X, Y coordinates, values).
- **Line files ASCII Arc Info (*.con) (read)**. Line data are converted to a tabular format (X, Y coordinates, values).
- **Grid files ASCII Surfer (*.grd) (read)**. Grid data are converted to a tabular format (X, Y node coordinates, values).
- **Grid files ASCII Zmap+ (read)**. Grid data are converted to a tabular format (X, Y node coordinates, values).
- **Grid files ASCII CPS-3 (read)**. Grid data are converted to a tabular format (X, Y node coordinates, values).
- **Grid files ASCII ESRI (read)**. Grid data are converted to a tabular format (X, Y node coordinates, values).
- Tables of volumes, reserves of oil, reserves of gas (*.vt, *.ot, *.gt) (save). Needed to export reserves estimation results to printer layout.

Cover:

- **Shape – file (*.shp) (read/save)**. Can only store lines or polygons.
- Line files ASCII Irap Classic (read/save). Storage files for Roxar lines.
- Line files ASCII Arc Info (*.con) (read/save).
- Line files Surfer (*.bln) (read/save).
- **Internal GST format (*.mgs) (read/save)**. A binary file readable in **GST** only and intended for fast data exchange between projects.
- Grid files Surfer (*.grd) (save). (See more detailed discussion in the «Cover Object» Chapter, in Section «**Tracing composite grid**»).
- **ASCII table (read)**. Files of table structure containing coordinate columns, column with line identifier or delimiter string (composed of equal numbers), separating one line from another.
- Line files ASCII CPS-3 LINES **(read/write)**.

Grid:

- Grid files ASCII Surfer (*.grd) (read/save).
- Grid files ASCII Zmap+ (read/save).
- Grid files ASCII CPS-3 (read/save).
- Grid files ASCII ESRI (read/save).
- Grid files ASCII IRAP RMS (reading/writing).
- **Internal GST format (*.ggs) (read/save)**. A binary file readable in **GST** only and intended for fast data exchange between projects. Stores color settings and tracking results.
- **Shape – file (*.shp) (save)**. Stores grid contours as polygons.
- **ASCII – table of points in xyz format (*.dat) (save).** Stores grids in tabular format, as well as node coordinates and values.

1.4.8. Linking to other project data

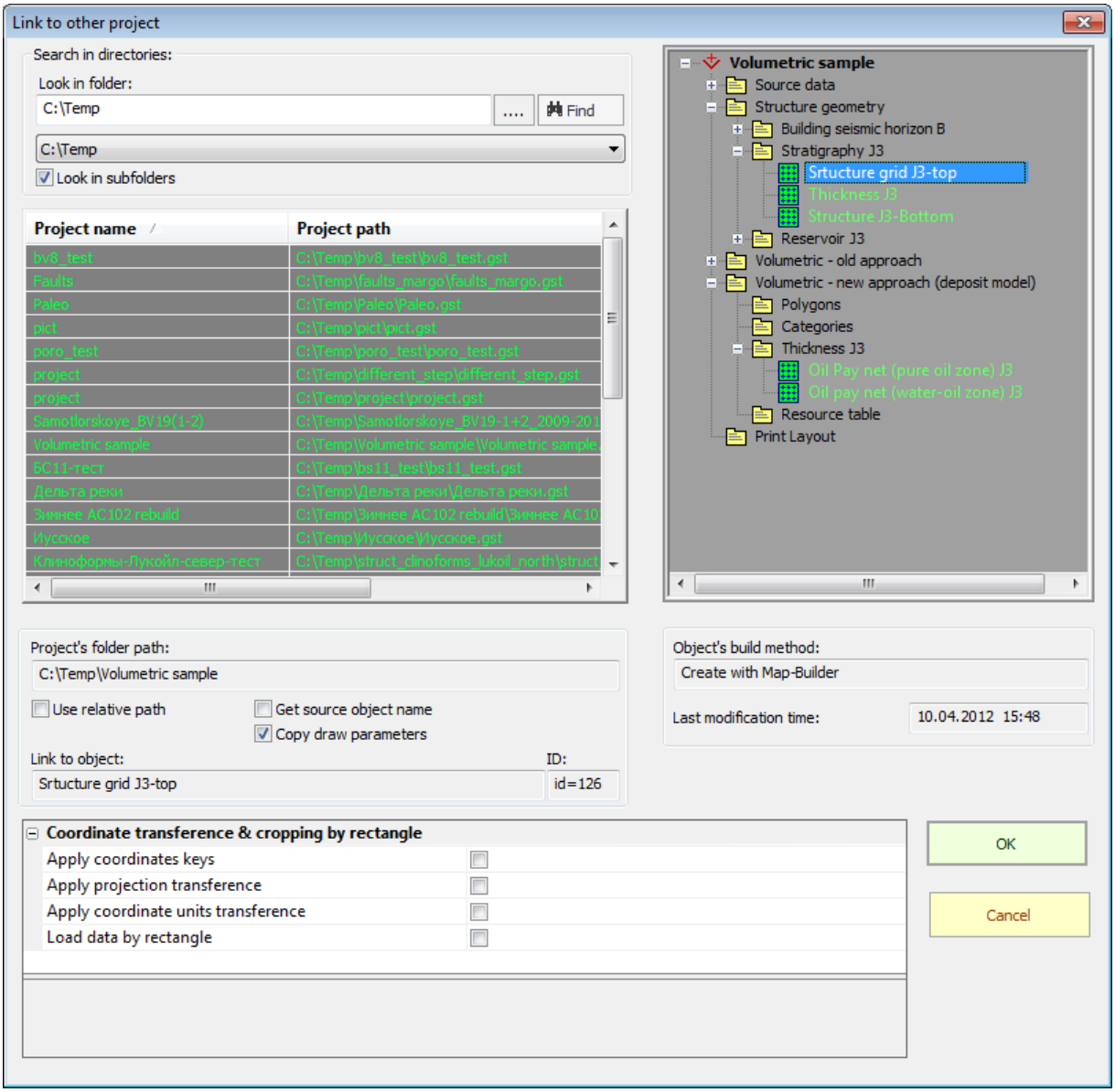
Starting with **GST v. 4.7**, the software realizes a new build method for objects of the main types (tables, covers, grids) - «**Link to other project**», which loads data without intermediate file usage. In

effect, the method boils down to copying data from a project saved on a disk to a work project, which provides the following capacities to the user:

- Add data of one or several objects of a project to another project without loading the source project (which may be time-consuming) or saving the data in an intermediate file;
- To tie up projects more closely: all saved changes in the source project are automatically tracked (while updating links);
- Save time on project saving, as well as memory space on hard disk (all data are stored in the source project, while linked-up object only contain references to them).

To link an object to an object in another project, user needs to do as follows:

- Set the build method «**Link to other project**»;
- Click  (standard build parameters button) on the toolbar to activate dialog that allows to choose a project and object:
- Select a project from list (same method as that in the dialog of viewing projects) and indicate object in a tree.



By default, the software seeks the source project following its direct path, e.g. '**C:\dir1\root\source_prj\source_prj.gst**'. In case the option «**Use relative path**» is enabled, then the starting search directory is the one that stores the work project. If the work project is stored in '**C:\dir1\root\work_prj\work_prj.gst**', then by using the option «**Use relative path**» the search starts from the directory '**C:\dir1\root**' and proceeds down the tree.

Thus, while linking long the direct path data can be loaded from any project on the local disk or from a network resource, but the search may run into problems while moving projects from one machine to another. While linking along the relative path, the directory storing the source project can be in the same folder (or lower) as the directory of the work project. When linking along the relative path user can transfer projects from one machine to another together with the root directory.

When data are imported from another project, you can apply the coordinate keys, recalculate data from one projection into another, convert coordinate units, and cut along the rectangle, same as you do while importing data from a file. Remember that the program will spend extra time on recalculation at every startup of the project and every update of a linked object.

With the option «**Get source object name**» enabled, the object name will be renewed each time along with the data updating, otherwise it gets renewed only the first time, if the object name is given by default.

In case the user wants to load data from one project to another by using the linkup interface but does not want to continue being linked to the source project, he must execute the context menu command "**Make 'base' object**". The main menu command «**Options → Disable project links** » applies this operation to all objects of a project at a time.

If, for any reason, the source project (or object) is not found at startup of the current project, data of the current project are not reset but further calculations based on a linked object are impossible.

Attention! Linking the object is possible if the source project is saved in **GST** Version 4.7 or later. Since the data of a linked object are stored physically in the source project only, linking to this object is impossible: such objects are not displayed in the dialog tree.

1.4.9. Import data from project

A new method "**Import from project**" is implemented in **GST** Version 6.9 for building objects of basic types (tables, grids, covers). Import means direct data loading from one project into another, bypassing the intermediate files and using the same interface as in "linking". But unlike "**Linking to project**" method, the imported data are saved in the project and are not lost when transferred to another computer. The objects built with this method can be linked from other projects.

When the data of the source object are modified, the imported object is not updated automatically but is highlighted in the hierarchy tree to indicate changes. The user can update the data of a particular object simply by rebuilding, or update all imported objects by selecting the menu command "**Options -> Update imported objects**".

In case there is no connection to the source project, regardless of its modifications, the imported object is displayed in the hierarchy tree as "Built" i.e., "green".

1.4.10. Rules of moving and Renaming Projects

Once again, follow rules that need to be observed if the user wants to rename projects or move them from one machine to another (or from one directory to another).

Renaming. The project name featured on the tree may be any – it does not affect the names of directories and files generated by **GST**. User has the right to change the name of the project directory and the name of the main project (***.gst**) file. User cannot change the names of other files and folders in the project directory to avoid corrupting information. Changing the name of a directory may lead to loss of other projects' links to the linked-up data.

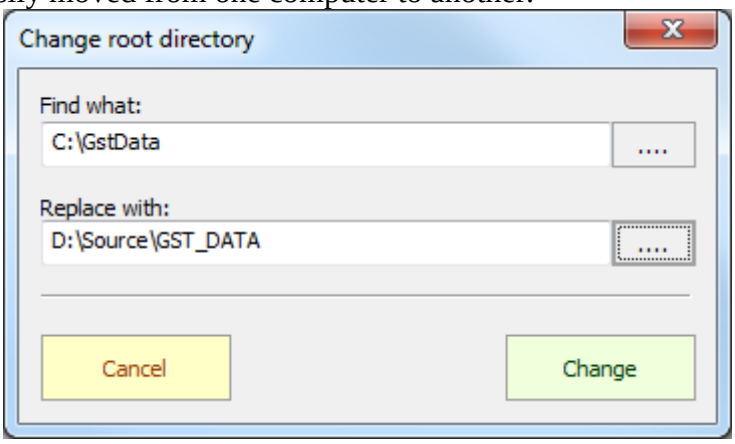
Move projects. Moving a project from one machine to another (or from one directory to another) may lead to the loss of links and output linking paths.

In the case where object data are loaded from file, the project stores the entire path to file (whether local, or network). Thus, in the course of moving a project to another machine or if the network connection failed, connection with file will be lost and the object will not be renewed. But the earlier loaded data will still be stored in the project structure and are usable for work. In case an object's option «**Link to file**» is on, every renewal of project links will emit the warning that the file is not to be found, but all subsequent build operations will be possible as if the input file of data has not been changed.

For objects with the option «**Output linking**» on, the output linking along the previous paths is impossible. In this case the user either needs to create a new directory structure on the other machine, or modify the paths of output linking.

If the link paths to file and the path of output linking is no longer actual, the menu commands «**Options->Disable file links** » and «**Options → Disable output links** » will disable the relevant options for all objects in a project.

To move project properly, if there are some links to other projects data, is only possible along with the all source projects. If user cannot move from one computer to another the whole projects set, some linked-up objects can be made «local» by context menu command "**Make 'base' object**". Executing main menu command «**Options → Disable project links**» user can apply this operation to all objects in a project, at a single time. Following this, the data will be saved in the project directory and the project itself is easily moved from one computer to another.



In case the entire project structure with its input data files and output linking directories has been transferred from one source to another (e.g. from Disk C to Disc D), all linkup paths and output links paths can be updated by executing command «**Options → Change root directory**. In the dialog above, user sets a fragment of the old path to be converted and a fragment of the new path where the whole structure is going to be stored. The procedure will check all project paths (file links, project links, output linking) for all objects and will replace the old path for a new one.

1.5. General settings

1.5.1. Application settings

To call the **GST-**application property dialog, users have to execute the menu command "**Options → Program settings**". In the property dialog pop up users can define objects display colors in hierarchy-tree, project saving work paths, etc.

Hierarchy tree colors. At the program's properties dialog page "**Hierarchy tree colors**" the user selects friendly colors for objects' annotation in a hierarchy tree, as well as colors for this window's background. Use the button "**Back to Default Colors**" in order to restore color settings recommended by software developers.

"**Enable memory safe mode**" is set in the **"Optimization"** category. This option is needed when the user handles large projects requiring a lot of RAM memory. In this case, the program creates the temporary directory in a system directory to store the objects which are not currently in use. **GST** automatically restores the data or stores it back to a hard disc as necessary.

Object storing provides when:

- The object is not currently under build process;
- The object's working window is closed;
- The object is not displayed as a graphic layer in other object's working window.

To make memory saving more efficient the working windows of only those objects which are currently in work should be kept open.

When the memory saving mode is enabled, the program reserves additional space on a hard disc, where temporary files are stored. Temporary files are automatically deleted when the project is about to be closed.

It should be kept in mind that with memory saving mode enabled, the program spends additional time to perform object storing-restoring.

"**Save memory while using Calculator".** This additional option (turned off by default) performs additional archiving of the objects during operation of the **Calculator**. One should keep in mind that turning on of this option will result in an increase of calculations time.

Reserve copying. Saving large, memory consuming projects (several hundreds of Mb) might take a lot of time, as long as several minutes. The project content directory may be corrupted if there is a system failure or a computer power supply failure. Starting **GST 4.4** version, a reserve copy of the project is created when it is loaded from hard disc, (the entire file structure is copied to the folder "**res_copy"**, which is automatically created in the project directory).

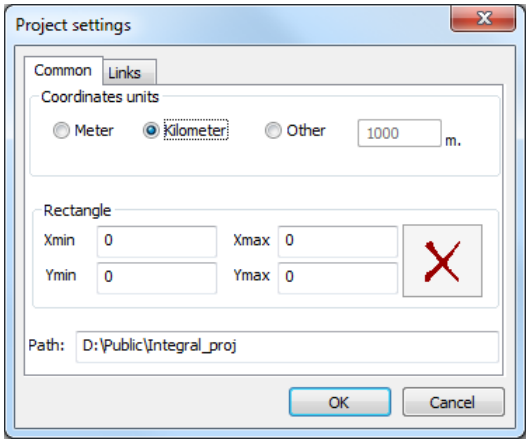
Settings for reserve copying are set in the **"Reserve copy"** category. Reserve copy option **can be turned off** (it is enabled by default), and the copy can be updated every time the project is saved (option **"Update after saving"**). Option **"Ask user's confirmation"** allows the user either to accept or reject a creation (or an update) of a project reserve copy.

If there was a system or a power supply failure during project saving, and the project content was corrupted, the user should copy the content of "**res_copy**»" directory into the main project folder and repeat the load procedure. If there are no errors during project loading and saving, the content of the reserve directory will be automatically deleted after closing a project.

Creation and updating of the reserve copy increases the project loading and saving time and it is a good guaranty of data and results safety. The **GST** developers strongly recommend making use of the back-up option, especially when working on big and important projects.

1.5.2. Project settings

Coordinate units of measurement. A project can contain an arbitrary number of objects different in their meaning. The only thing they must have in common is the coordinates units of measurement.



Coordinates measurement units are set on the tab "**Common**" of the "**Project Settings**" dialog (menu command "**Options → Project settings**"). This value is needed for proper use of a scale size (for grid tracing, output to print, etc.)

Starting with **GST** version **4.8** users can set a rectangle for the entire project, which can be used to set the grid boundaries or in data loading.

2. Table object

A **Table** object is a combination of digital and string data structured into rows and columns. A table may contain both point information, and any data of two types: strings and values. The data in one column can only be of the same type.

Information contained in a table is shown in the work window of an object in a text format and can be directly edited. It can be shown as points, etc. in the graphic windows of other objects as a significant or insignificant reference (graphic layer).

2.1. Selecting rows and data

2.1.1. Selecting rows in a table

Starting with **v. 5.0**, users can select table rows and perform different mathematical and logical operations. A selected row is highlighted (green by default). User can change the color in the dialog «**Program settings**» (menu «**Options**») in the category «**Other settings**».

To highlight one or several rows, the user needs to enable the **Row selection mode**, by clicking  on the toolbar. A row can be selected by left-clicking on the row-number column. By holding **Shift** key, the user can select a series of rows. The highlighting (selection of rows) is canceled by executing the command «**Selection->Deselect Rows**» or clicking  on the tools bar.

2.1.2. Selecting rows via work windows of other objects

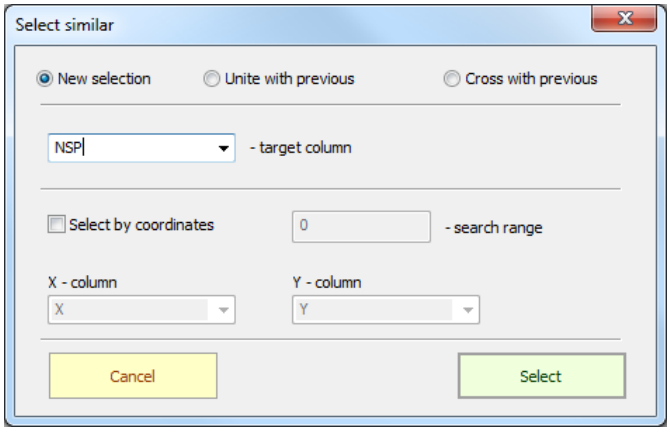
User can select and deselect rows in a table via the work windows of other objects (grid, cover), where the table is shown as a graphic layer. The selected point is shown on the plane and colored as a selected table row.

Selecting one point can be done by executing a command in the context menu of a grid or cover work window (right-mouse click). The command to be executed is «**Select/Deselect**» by moving the cursor to the point. *Selecting a group of points* can be done by going to the **Select point's mode** (click  on toolbar). With this mode enabled user left-clicks to mark the points or define the selection rectangle. Click  on the toolbar to cancel the latest selection. Clicking on the selected points will deselect that point.

2.1.3. Selecting similar rows

The command «**Selection->Select similar rows**» is used to analyze table data for repetition of information. In a dialog the user defines a selection condition.

The software searches in the target column for equal values and selects the relevant rows. In the case where the option «**Select by coordinates**» is on, **GST** compares the point's coordinates and selects rows, if the points are closer than defined in the «**Search range**».

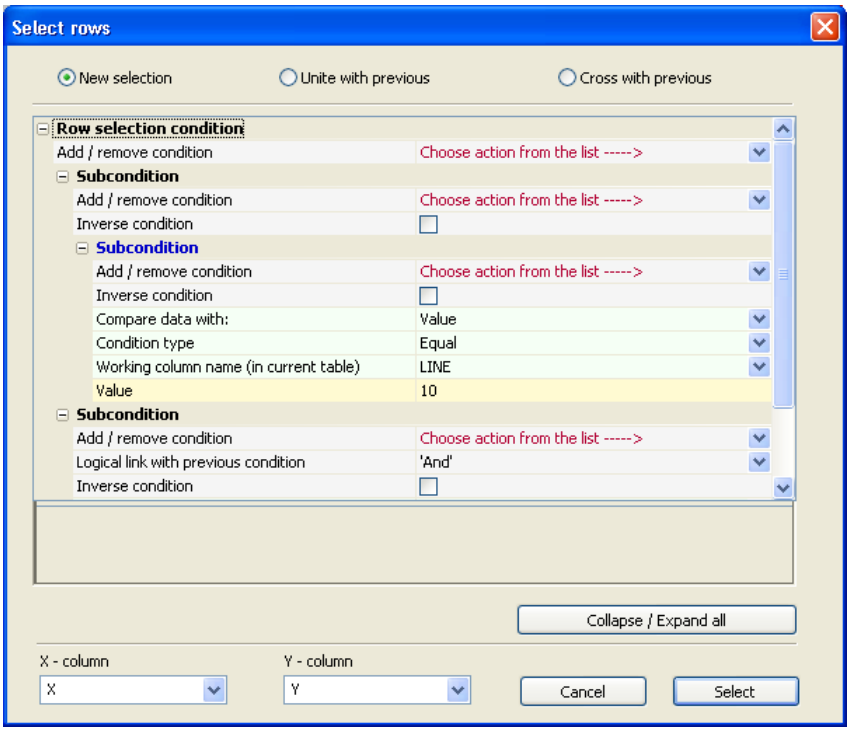


Switches in the upper part of the dialog enable the user to do a new selection, unite with a previous selection or select inside a previous selection.

2.1.4. Selecting rows by special condition

The menu command «**Selection->Select special**» or the icon on the tools bar selects rows according to any user- defined complex condition.

The condition and other parameters are defined in the dialog below. Switches in the upper part of dialog enable the user to do a new selection, unite with previous selection or make a selection inside previous. The coordinate columns (in the lower part of dialog) are applicable when table rows are seen as point data. The condition is formulated in the dialog as a hierarchy tree.



Selection can consist of one or several conditions (**simple** or **complex)**. A complex condition contains one or several **sub-conditions**, each of which can also be simple and complex. A simple condition contains only operator and comparison parameters which can be either «**True**» or «**False**».

Example:

$$A = 0 \; and \; B > 1 \; and \; \{C \neq 0 \; or \; C \leq A\},$$

where A, B, C – are table column names. In this case, A =0 and B > 1 are simple conditions, and the expression in brackets is a complex condition consisting of two simple sub-conditions.

Selection is formulated using the functional element «**Add/Remove condition**» (in the first line inside any condition). The element drop-down menu in the title of selection suggests two options:

- «**Add condition**», meaning to add a condition to the main selection list;
- «**Reset**» - delete all but the first condition.

The dropdown menu of the analogous element inside every condition enables user to:

- «**Add sub-condition**» - change a simple condition into a complex one;
- «**Remove all sub-conditions**» - change a complex condition into a simple one;
- «**Delete condition**» - delete the condition with all its sub-conditions.

Let us consider parameters which may be set in a condition. The two most typical conditions of both simple and complex conditions are:

Inverse condition. Having this option on actually negates the condition.

Logical link with previous condition. This option is accessible for more the one condition. The parameter can mean either «AND», or «OR».

The rest of the parameters are accessible for a simple condition only.

Working column name. This parameter accepts the column name. These column data are compared with values, rows and data of other objects in **GST**.

After the work column has been chosen the user defines the comparison type. This option is installed in the field «**Compare data with**». There are:

- **Compare with a value** (simplest and by-default variant of a simple condition);
- Compare with a string;
- **Compare with a column in table**, which means that the data in the work column are compared with data in another column of same table;
- Compare with a polygon;
- Compare with a grid;
- Compare with another table;

In case of one of the three last variants, the user needs to choose an object to compare in «**Compare with object**» element. The software prompts the user to define the condition type and set some additional parameters.

To **compare with a value** the user chooses one of the following condition types: «Equal», «More», «Less», «Within range», «Outside range» and «Approximately equal». In the first three cases, the user sets a value for comparison, in the next two cases, a range of values, and in the last case, apart from a value, user specifies «**Delta for approximate comparison**».

The **comparison with a string** can be of three types: «Equal to string», «Contains string», and «Beginning with string». User sets the string itself as a parameter.

To compare **with data in another column of the same table the** user specifies the name of the column. **GST** will then suggest the choice of four conditions: «Equal», «Approximately equal», «More» and «Less».

Comparison with a polygon provides for checking whether a point is inside or outside a polygon. To do this, the user specifies coordinates fields in the current table in the lower part of the dialog.

Comparison with a grid is done by comparing the data in the work column with grid values measured in the points of the relevant rows. **GST** offers the following condition types: «Approximately equal», «More», «Less», «Inside grid rectangle», and «Outside grid rectangle». The user has to specify coordinates fields in the lower part of the dialog.

The **comparison with another table** is done between a row in the current table and a row in the chosen table. **Searching rows in tables** can be done in two ways: «**Compare rows by unique ID**» and «**By coordinates**». In the first case, the user specifies columns with unique ID in both the current and the chosen table. In the latter case, coordinates fields in both tables, as well as a "delta" for approximate comparison. User has a choice of: «Equal», «More», «Less», «Approximately equal» and «Selected». For the first four conditions in both tables, the user specifies names of work columns. The last condition becomes «True», when the row in the chosen table that corresponds (either by ID or by coordinates) to the row in the current table, is selected.

2.1.5. Handling selected rows

If at least one row is selected in the command «**Selection->Deselect rows**» (icon  on the tools bar) becomes accessible. By clicking  on the toolbar user can jump from one selected row to another.

Other commands to operate selected rows are also found in the menu item «**Selection**». The user can undo the line selection, copy selected rows in a new table, copy or cut out the selected rows into the buffer, delete selected rows, and move selected rows either to the beginning or end of the table. The command «**Set value to selected rows**» will show a dialog where the user needs to specify the column name and the value to be set. The command «**Calculator operations using selected rows**» (icon  on the tools bar) allows mathematical and logical operations to the table data only in the selected rows. More detailed discussion is found in the paragraph «**Mathematical operations with columns**».

2.2. Handling tables

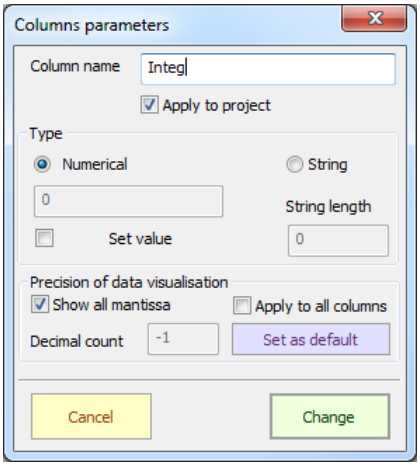
2.2.1. Editing data

To access the table editing mode left-click on the data window. To exit the table editing mode, click outside the data field. If a cell seems to be inaccessible for editing, scroll a bit and repeat the operation.

2.2.2. Changing column type and name

To change a data type in a column, select a column and execute the menu command "**Table→ Column Parameters**" or click  on the toolbar.

In the dialog window that pops up, user sets new parameters for the column: type (string or values) and field name, as well as specify a certain numerical value for the whole column. If a column is of string-type, it is necessary to indicate the maximum string length (max number of symbols). If it is a numerical column, you can specify a number of digits after the decimal sign.



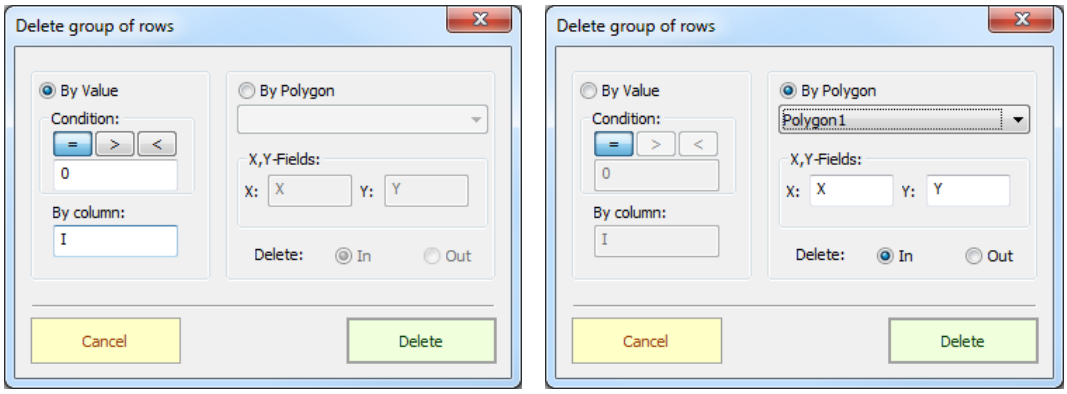
Attention! Please remember that **GST** works with column names, not column numbers; therefore, if there are digits in the table title, they will be treated as column names.

2.2.3. Adding, deleting and moving rows and columns, sorting table

To add a column, click on the toolbar, or execute command «**Table->Insert Column**». The dialog window «**Column parameters**» pops up. The new column is inserted in front of the pre-selected column. Moving a column right or left is done by buttons. To delete a column, select it by clicking the mouse and then press or execute the menu command «**Table->Delete column**». Clicking buttons or executing the relevant command in the menu «**Table**» the user can sort rows in a descending or ascending order. By executing the context (right click) menu command «**Copy column**» user duplicates a selected column. The suffix «(Copy)» is then added to the column name.

To add a row click button on the toolbar or execute the command «**Table → Insert row**». Delete a row by clicking button or executing the command «**Table->Delete row**».

Deleting a group of rows. **GST** features the capacity of deleting a group of rows from a table by using a certain condition. The command «**Table->Delete rows group**» shows the following dialog:

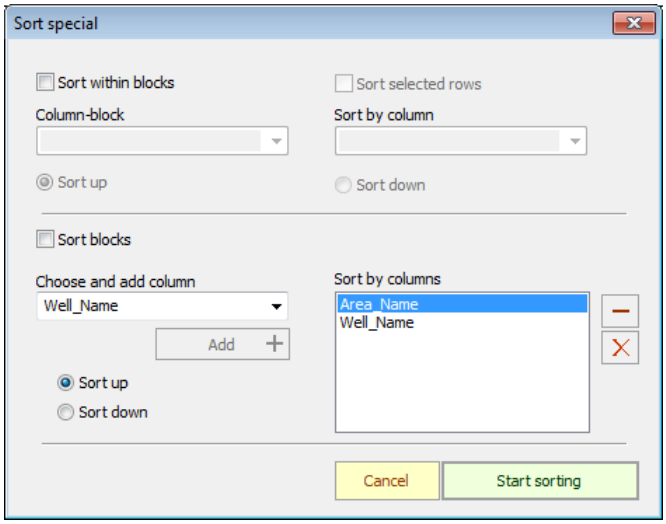


Using the option «**By value**», the user can delete rows with the application of a simple logical expression. The example cited in the dialog deletes from the table all rows with values of zero in column I. Having the option «**By polygon**» on and specified coordinate fields user can delete rows that belong or do not belong to a certain polygon (to be picked from a drop-down list).

2.2.4. Sorting the table

To sort the table content either in ascending or descending order, according to values in a column, select this column and use the toolbar buttons  or the corresponding commands in the "**Table**" menu.

A more advanced sorting option is provided by the menu command "**Table - > Sort special**" (see the dialog in the figure below).



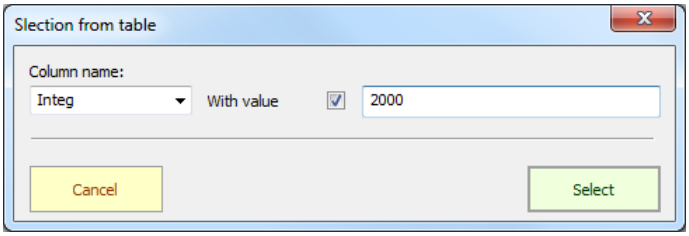
"**Sort within blocks**" option allows the user to sort the table data according to some feature inside the automatically selected row blocks. In this case, a "block" refers to a group of rows arranged sequentially in a table, with the identical value in one column. For example, for seismic data downloaded in a tabular format, stations within each profile can be sorted (by number or cdp). For this purpose, specify a column with the profile number in the drop-down list "**Column-block**" and in the right-side list – the column with the station number. When the option "**Sort selected rows**" is enabled, splitting into blocks is performed by the selected rows instead of the specified column.

The user can specify sorting parameters across multiple columns in the lower part of the dialog. Sorting sequence is displayed in the right-side list, which is formed by the drop-down list on the left. Once the user selects a new field, it appears immediately in the list. If the field name is typed from the keyboard, press the "**Add**" button. Each element in the list can be assigned the direction of sorting; you can delete the element or clear list using the buttons on the right side of the dialog. In this example, sorting is first performed by the area name and then (within this set) – by the well number.

"**Sort blocks**" option provides sorting table data by blocks without changing the sequence of rows within these blocks. This enables, for example, sorting the seismic table by profile number (name) without changing the topography sequence of stations.

2.2.5. Data selection from table

This function enables the user to create one or several new tables consisting of rows that contain a certain numerical property, on the basis of a current table. Thus, if the table contains seismic line numbers, the selection function can be used to sort out seismic lines into separate tables. Execute the command «**Table->Selection**».



Specify the name of the column that contains the property to be placed at the basis of the selection. To make the selection more definite, activate the option «**With value**» and insert a number in the textual field (e.g. 10, the seismic line number). A new table (or tables) will be added to the **GST** hierarchy tree.

If the user sets the column name, but does not specify a value, the software will create a folder with the name of the table that will contain a set of tables all differing by the numerical property in the specific column. If the user does not specify either a name or a value, the newly created folder will contain a set of one-row tables whose names will be the row names in the initial table.

2.2.6. Mathematical operations with columns

To handle a table, **GST** has the capability to perform various mathematical and logical operations with numerical columns. Left-click on the column, where the result will be entered, and execute the command "**Table → Column Calculator**" or click on the tools bar. The **Calculator** dialog will pop up on the screen (see chapter 9 "*Creating objects with Calculator*").

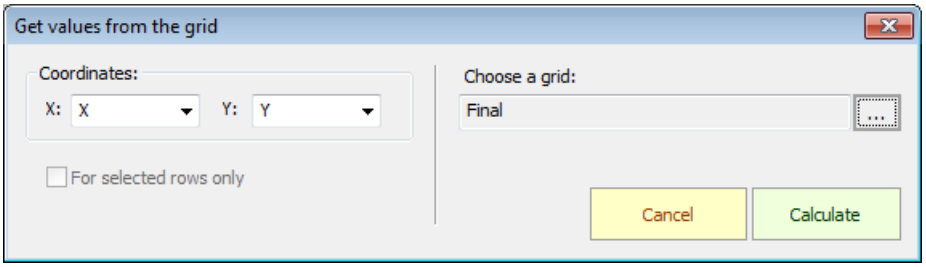
In the lower list the variables D1, D2, …, Dn identify all numerical columns of a table. In the upper window the user writes a mathematical or logical expression (see **Section 9** for details).

Click "**Apply changes**" to start calculations, or "**Cancel**" to cancel. The results are entered into the column originally selected.

The command «**Column calculator with selected rows**» (icon on the toolbar) allows mathematical and logical operations done to table data only with the selected rows.

2.2.7. Reading values from the grid

It is possible to read values from the grid without **Calculator** in specified columns and put them into the specified column of the table. To do this, click the right column and click the toolbar button .

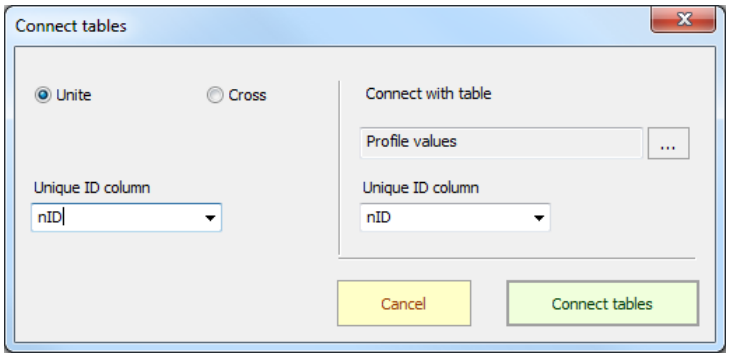


You should specify the coordinate columns and choose grid to read values in the dialog box shown above. This action can also be performed only for the selected lines of the table.

2.2.8. Connecting tables

The menu command «**Table–>Connect tables**» enables the user to add columns from another table in the project to the current table according to a certain congruent property. All parameters are set up in the dialog below.

Select the table to connect in the right part of the dialog, then establish in both tables columns containing unique (non-repeating) IDs.



If the user activates the option «**Cross**», the resultant table will only retain rows with coinciding identifiers. In case the option «**Unite**» is activated, the resultant table will contain, apart from the extension, the rest of the rows of the selected table at the bottom of the initial table. The names of the added columns will have the prefix «new».

2.2.9. Undo

When editing tables, the program memorizes the user's actions and, if necessary, helps to roll back to one of the previous states through the undo function. Click button on the toolbar or execute the menu command "**Edit → Undo".**

Note: When the table work window closes, the undo buffer content is deleted.

2.3. Table build methods

2.3.1. Creating a table manually

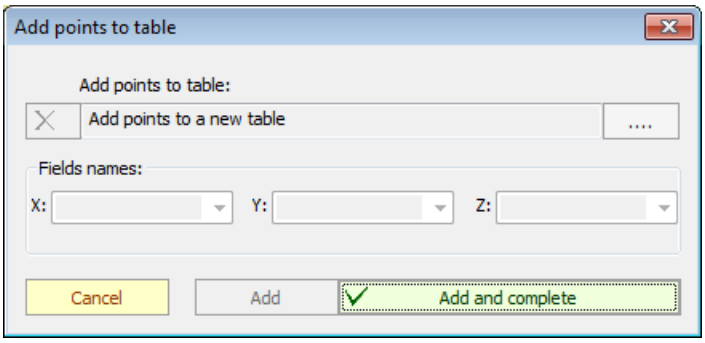
Select a build method «**Create table manually**». In the *build settings* dialog specify the expected number of rows and columns, and start build process. **GST** will create a table of the relevant size and fill it with zeroes, which the user can edit it at will. The table sizes may be changed later.

2.3.2. Creating a table of points

This method cannot be found in the list of build methods. The table is created so that it contains the coordinates and the values of the points specified by the user by the mouse in the main window of the grid or coverage. The method "Creating a table manually" will be set for such table.

You should press button or run the command "**Grid> Manual editing-> Creation points mode**" on the toolbar of grid or coverage.
Mark the location of points (points are displayed in orange by default) with the left mouse button.

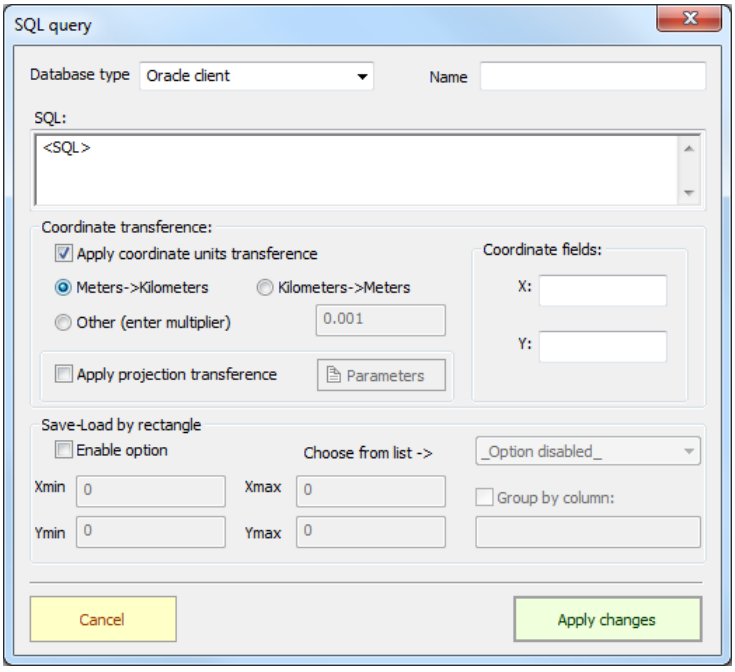
Press the button on the toolbar or run the command "**Grid-> Manual editing**-> **Add points to the table**"; you can specify in the dialog box whether to create a new table or add points to the existing one.



The hierarchy window will show a new object with the default name of «**Table**» that contains three columns: point coordinates and values taken from map (grid). In case a point is outside the grid area, the table will contain a nan-value. When the points are created through the cover's work window, the table will contain only the coordinate's column. The user can edit the data and use it to build other objects.

2.3.3. Loading from database

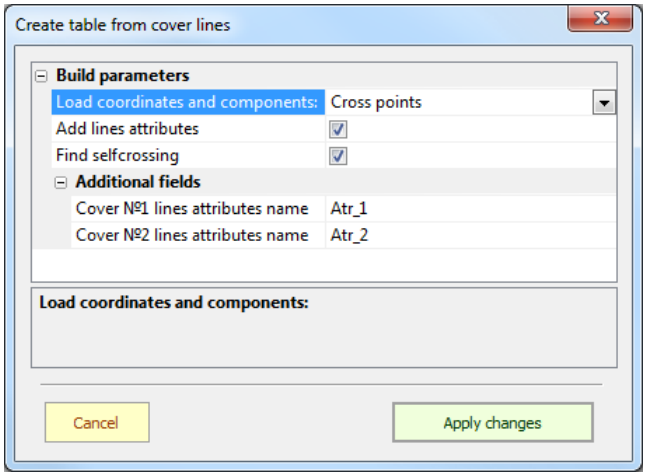
Choose the build method «**Load table from database»,** and the build settings dialog will be as follows:



In the dropdown list select the database type (the current **GST** version realizes access to the **Oracle** base). In the input window on the right specify the name of a database and, below it, make the **SQL** request. Then user may specify the coordinate transferring parameters.

2.3.4. Creating table from cover lines

To translate line coordinates and attributes into a tabular form, there is a special build method named "**Create table from cover lines**". All parameters are defined in the following build settings dialog:



Changing selection in a "**Load coordinates and components**" drop-list, the user can load into the table:

- **Cover points** (X and Y-coordinates are placed in two different columns). If the option "**Add Attributes**" is selected, then a numerical attribute (for example, a level value) sets into a third column.
- **Cross-points**. Point coordinates are entered in the first two columns. The option "**Add Attributes**" selected, the third and fourth columns include attributes of lines intersected in a given point. With the option "**Find self-crossing**" off, the cross points are searched for only between the lines of different covers; when this option is enabled, all cross points are sought.
- **Tangent components**. Line point's coordinates are entered into the first two columns of a table (GST takes a middle point between two nodes). The normalized tangent vector components are entered into the next two columns.
- **Normal components**. A table is populated in the same manner as in item above, except the third and fourth columns are filled with normal vector's components.
- Names of additional fields of a table (for line attributes, tangents and normal coordinates) are set in the lower part of the dialog.

2.3.5. Correction of data table

This build method relates to the task of tying up conflicting data-groups, e.g. seismic crews or seismic lines/profiles. If the data is not properly matched, the mapping task is frequently solved with vertical-offset optimization, i.e. the resultant map and data can differ by some fixed offset or by a constant factor, the value of which also can be unknown.

The method provides for the use of tables involved in the construction of a certain grid, the value column of which is multiplied by a constant factor and added to the fixed offset (e.g. to the parameters P and Q on the tab "L -parameters" of a table-reference property dialog).

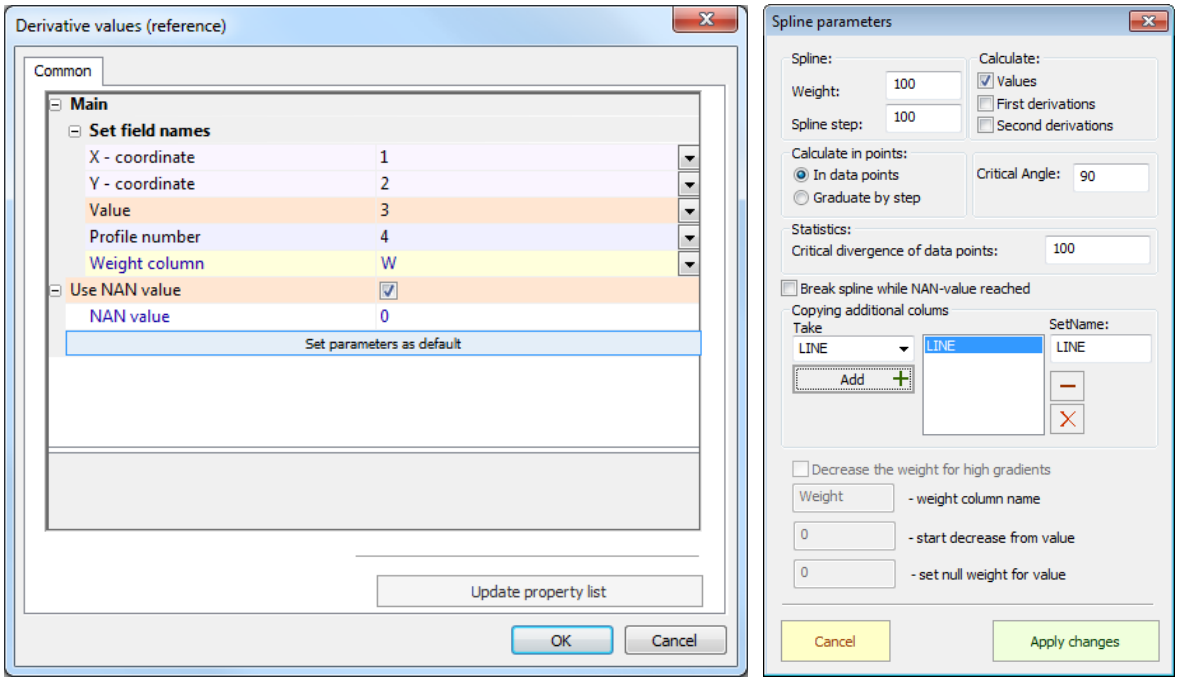
A table is built on the basis of one grid, the computation of which involved an offset or constant-factor optimization, as well as on the basis of tables used to build this grid. It is thus necessary to make a reference to a grid, without which an object could not be constructed, and a reference to the tables based on which the object was constructed. If references relate to one grid only, the amended table is made out of all the tables used to build this grid. Importantly, these tables should have the same structure (same number of columns, etc.), or the build is impossible. If an object has table-references added, the object is built on the basis of the grid and the tables references had been made to.

The user copies all the basic-table columns to a new table, amends the value column and adds a column with an offset value (bearing the name DZ). If there is more than one basic table used, the user adds a column where each point is assigned a proper table name.

2.3.6. Spline interpolation

GST features the capability of table data spline smoothing. This mode can be used to process seismic line and seismic crew data. A seismic data table should have columns with X- and Y-coordinates, Z (T0)-values, as well as a column with a seismic line number or name. This build method provides the user with a table having smoothed values of a function, and allows calculating the first and second derivatives along a line direction.

To build a new table, indicate build method as "**Spline interpolation**", and make a reference to a table or table group containing seismic data. In the table reference dialog (see table below on the left) indicate parameters – X-and Y-coordinates, Z-values, line numbers and nan-values. If there is a column containing weights, the software will omit the rows where the weight is zero and will take into account only rows with weights that differ from zero. Other build parameters are specified in the build settings dialog (see below on the right).



Weight and **Spline step**. An optimal approximation step, as a rule, should nearly equal the average distance between data points (may be more).

Calculate in points: – there are two variants of calculation:

- Calculation at data points;
- Graduate by step – the program, whenever possible, graduates step and perform calculations not only in data points, but for intermediate points as well.

Calculate – there are several variants of calculation:

- Calculate by **Values**. Result obtained is **X**- and **Y**-coordinates accounted for a smoothing spline; **Z**-parameter; a line number (**LINE**) and name of a seismic crew (**NAME**), if there is more than one table reference;
- Calculate **First Derivatives**. Result obtained is three additional columns: first derivatives (**DZ**) and direction cosines (**DX**) and (**DY**);
- Calculate **Second Derivatives**. Result obtained is four more additional columns: second derivatives (**D2Z**), squared direction cosines (**DX2**) and (**DY2**), as well as direction cosines 2DX multiplied by DY (**DXY**).

Critical angle – since some profiles can curve in excess of critical angle (90^o by default) the software automatically splits (cuts up) the line into parts.

Break spline while NAN-value reached – there are two variants:

- If this mode is off, the software will draw a smoothed curve through these points with no account for their values.
- With this mode on, the software will break a spline when it arrives at the point with a NAN-value and resume it from the next existing value.

Critical divergence of data points: The user can set a critical value of smoothing-result deviation from initial data, which when exceeded will be found in the dialog "**Smoothing Spline Statistics**".

Copying additional columns. Users can copy any other basic-table fields to the resultant table of derivations. To do this, select the needed field in a drop-down dialog list box, and then click "**Add-→**" and assign a name to a new field.

Decrease the weight for high gradients. This option is implemented in **GST** Version 6.7.1 or later and is available when the calculation of the first derivatives is enabled. It generates a weight column, where the point weight decreases against the increase in the absolute value of the gradient (the first derivative along the profile). Using this weight function in map-building excludes mapping flaws (false extremes) in high-gradient zones. Weight is generated by the following rules:

- If the absolute gradient value is lower than that specified in the field "**start decrease from value**", weight equals to 1;
- If the absolute gradient value is greater than that specified in the field "**set null weight for value**", weight equals to 0;
- Between the two specified values weight decreases from 1 to 0.

The calculated weight function is entered into the column with the name specified in the field "**Weight-column name**". If a column of the same name already exists in the table (appears when copying additional columns), the generated weight is multiplied by this column.

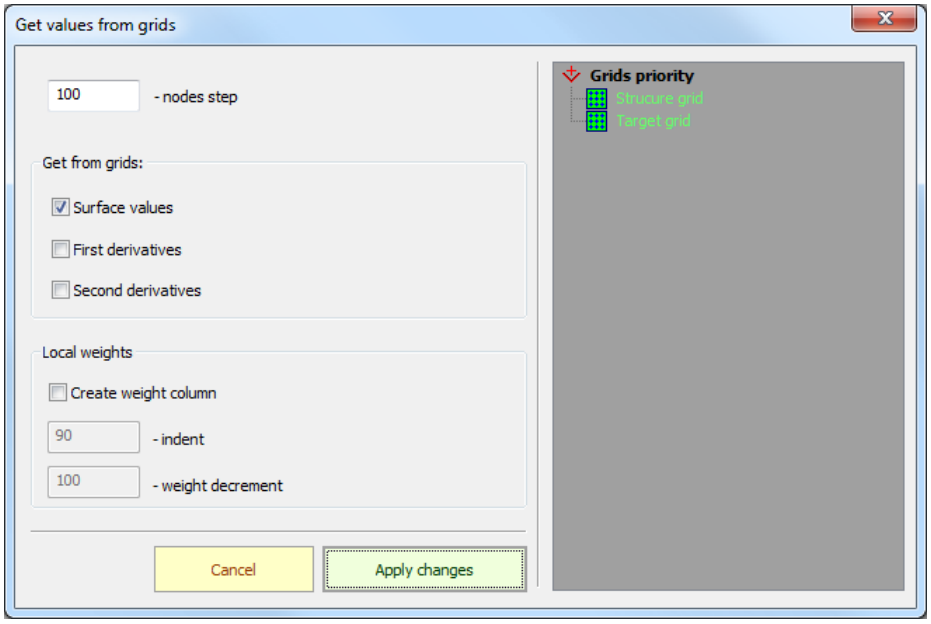
Having input all necessary parameters, build a table by clicking  on the toolbar or by executing the command «**Object->Start build**». One the table has been built, click  on the toolbar to show the dialog «**Statistics**», which reflects in detail the deviations from the initial data (maximum deviation, average absolute deviation and quadratic deviation).

2.3.7. Reading data from grids

The method allows reading data values, as well as first and second derivatives, from one or several grids according to a set network within a definite polygon. To build a table, the user needs to:

- Choose build method «**Get values from grids**»;
- Add reference to grids;
- Add reference to the polygon within which the table will be computed (the cover type should be «**Polygon**»);
- Enter build parameters.

The parameters dialog is shown below. **Grid step** is entered in a project's measurement units. Nodes are generated within the definite polygon at the established grid step. User makes the choice of what will be computed at the grid nodes – the first or the second derivative or grid value. The option «**Create weight column**» allows assigning local weight to each node. Initially, the weight is assigned as 1. The closer to the polygon boundary, if is distance smaller than the «**Indent**»-parameter, the weight decreases. The maximal weight decrement value is entered in a «**weight decrement**» control. Use «**weight decrement**» option to minimize boundary effects that appear while mapping dense point data.



The grids priority is identified in the right part of the dialog. This is important when two or several grids share an area – in this case, data is read from the grid whose priority is the highest.

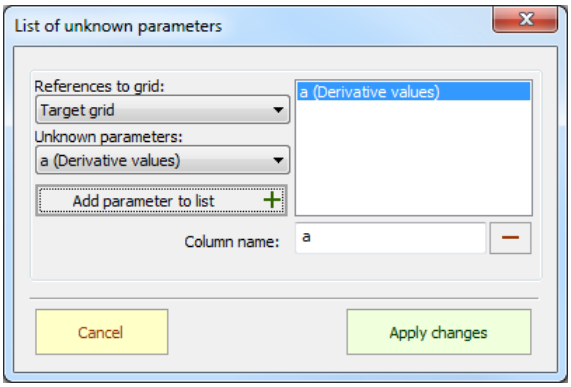
2.3.8. Load unknown coefficients

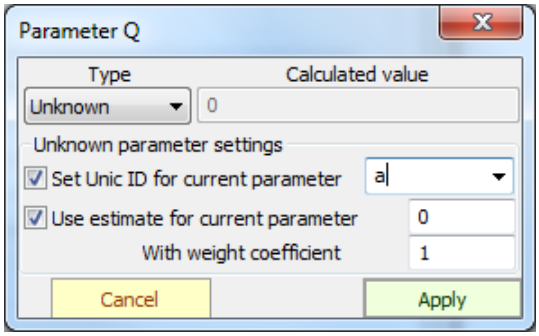
This build method is applied for value-table input of unknown factors that are used at *generalized mapping-task setting* (in global and local equations). A table being built has an arbitrary number of columns in a row/line. It can be used in subsequent computations, e.g., for **Calculator**-based object build. To build a table, follow these steps:

- Set build method to "**Load unknown coefficients**";
- Add references to the grids built by **Map Builder** with the use of an unknown parameter;

- In the build parameter dialog (see below) enter the required options.

In the right part of the dialog there is a box listing unknown parameters. Each parameter is loaded to a column with the certain name which is by default taken by the unique name of the unknown assumed. The list is added to by clicking "**Add parameter to list**". The value of an unknown parameter is taken from a grid indicated in the upper dropdown list box. Find the unique name of the unknown parameter in the list below.





To table-load the unknown factor computed via a mapping task, the corresponding parameter should have the option "**Set Unique ID for current parameter**" specified (in right dialog).

2.3.9. Merging tables

This method allows parallel or sequential merging of multiple tables into one. For this purpose, set "**Merge tables**" method and add references to the merged tables.

Sequential merging is set by default and is very simple: the first columns are combined with the first, the second – with second, etc. The field names are taken from the first table in the queue. If the merged tables have different number of columns, the number of columns in the resultant table is taken at maximum and the empty cells are filled with null-values .

For parallel merging, select "**Merge tables: Parallel**" in the 'build table parameters' dialog. Next, select the settings for the parallel merging. **GST** offers three modes: merge table rows as point data, as seismic data, or link to attribute tables. These modes are set by selecting "**Merge as: (Points, Seismic profiles, Attributes)**".

Merging in the "**Points**" mode means that for each row of one table a correspondence is sought in the rows of the other tables. This correspondence is established: by a unique name (number) of a point, by a double name (name of a well and area), or by coordinates. When merged in the "**Seismic**" mode, the tables are sought by identical profiles (by name or double name) and then binding of shot-points is performed for each profile. The "**Attributes**" mode implies that the table indicated in the reference properties as the "**Base table**" adds attribute columns from the other referenced tables. Merging is performed by matching the unique column number in the base and attribute tables.

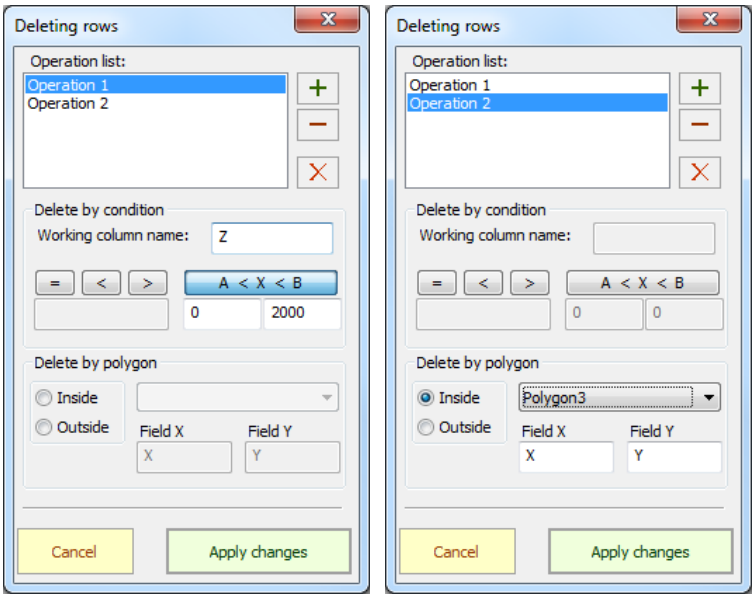
If a **sequential merging** is selected, the reference parameters are not required; for **parallel merging** you must specify fields, names (well, profiles), coordinates, as well as indicate null-value, if applicable, in the reference properties dialog. In "**Seismic**" mode you must specify in the reference parameters a column with the **profile number** and **shot-point number** or **cdp**. **Important**: The data inside the profile should be initially sorted in ascending order by SP number or by cdp. In the "**Attributes**" mode only one reference should be indicated as "**Base table**".

In parallel connection columns with the same name from different tables are not duplicated and merged into one; therefore, if the columns in the resultant table are considered different, they must be assigned a unique name in the source tables, i.e., prior to this procedure.

2.3.10. Deleting rows by condition

Set the build method as «**Deleting rows by condition**» to "filter" data in one or several tables and place the data into the current table. Do the following:

- Set the build method "Deleting rows by condition";
- Add references of tables to be filtered (if there are several, they should have the same structure and field names);
- Add references to polygons for polygon condition (the cover type should be set as «**Polygon**»!);
- Enter deleting rows conditions.



Rows can be deleted step-by-step, the steps being of two types: **deleting by condition** and **deleting by polygon**. For the first type, enter name of work column (e.g. Z, as in the window on the left) and specify condition of the required type. In this particular example the software will delete all rows where the value entered into the Z column is over zero but under 2,000. For the second type, enter the option of «**Inside**» or «**Outside**» the polygon, select the polygon from the list and specify the names of coordinate fields.

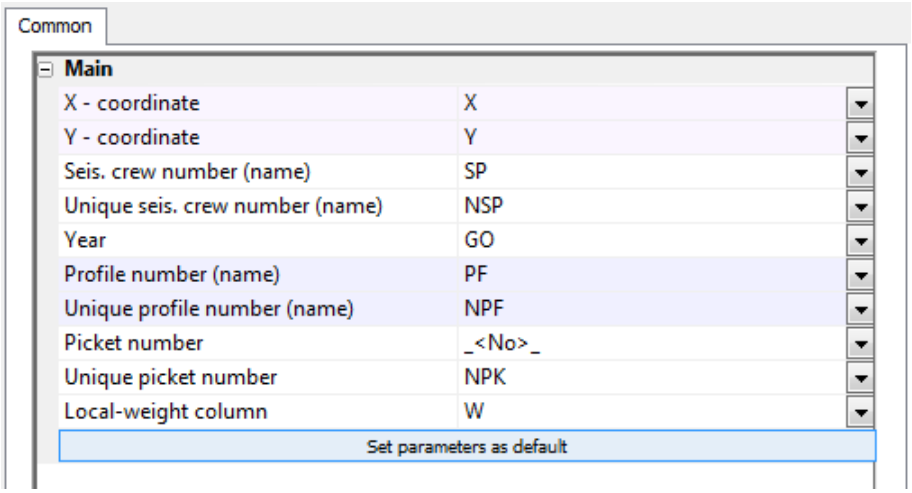
Click «**Add**», «**Delete**», «**Clear**» to add to or to clear the list of operations.

2.3.11. Local weight setting

This method is designed to process seismic (or other) data represented in a tabular format. It boils down to adding a number column containing weight ratios on a specific point. The purpose is to provide the logged weight-setting procedure to a specific seismic crew number, seismic line, point, etc. By setting the local weight at zero, the user can reject some data.

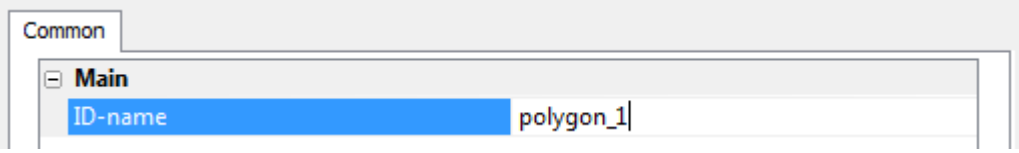
Set this build method and add reference to a table (tables), with the seismic data. In the reference properties dialog user specifies the names of the following fields:

- X, Y coordinates – necessarily;
- Seismic crew number (name);
- Year of shooting seismic;
- Unique seismic crew ID;
- Seismic line number (name);
- Unique seismic line ID;
- Point (picket) number;
- Unique picket ID;
- Local weights column (specify it if column exists in source, and do not specify if it is not).



Fields identifying seismic crew numbers are needed to establish local weights for certain seismic crews. If a table contains a column with a seismic crew number which is unique for the entire set, it should be entered as «**Unique seis. crew number**» (regular number and year of seismic shooting need not be set); if the crew number is not unique, the column must be identified as «**Seis. crew number (name)**» and additionally enter the year column. Line number and picket number can equally be stated or not.

Local weights can be entered for the seismic data inside (and outside of) polygons. In this case, the user needs to make references to the polygons (the cover type should be «**Polygon**»!) and in the reference properties dialog every polygon should be assigned an ID, see below:



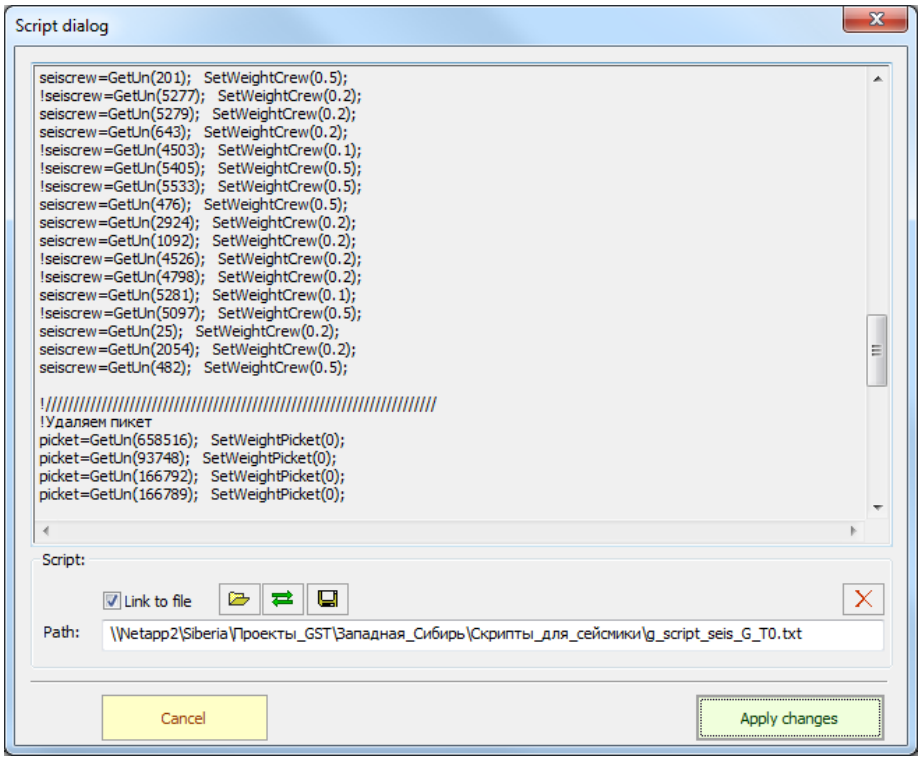
The name (ID) should be unique and should contain no blanks, commas, dots, etc.

Local weights are generated with a special tool "**Weight Script**", which can be reviewed in a build settings dialog.

The text window is empty initially; the text of the task script is typed in a text editor (**Notepad**) e and is loaded into software by clicking «**Load from file**» - following which it is saved in the project. The task can be edited in an external text editor. If the task file is lost, the text can first be loaded into the file by executing command «**Save to file**», then edited and loaded back into **GST**. Click «**Update**

script», for fast loading of edited script to the software. Use icon buttons  to perform these operations. With the option «**Link to file**» enabled, the script is renewed automatically while rebuilding an object.

Once table fields, polygon IDs and task script have been loaded, the user can commence building the table. **GST** processes the task and identifies local weights to accord with it.



Script description. Seismic data processing script is subject to the following rules:

- It is always typed in Latin alphabet;
- Software executes it step-by-step from beginning of file to the end;
- Each command in the script is divided from the next one by '**;**';
- A line that starts with an '**!**' is ignored by **GST** which sees it as commentary (in any language).

Seismic crew, seismic line, picket. To establish local weight for a definite seismic crew the user needs to specify **current (work)** crew by using the keyword «**seiscrew**» and functions «**GetUn(NSP)**» and «**Get(SP, YEAR)**». Example:

seiscrew = GetUn(1515); or **seiscrew = Get(2, 1999);**

In the former case, work crew is identified by its unique number or name, and in the latter case, its regular number and year. Depending on data availability, either function is applicable.

Keyword **profile,** and functions «**GetUn(NPR)**» and «**Get(PR)**» allow identifying the work seismic line, i.e.:

profile = GetUn(12536); or **profile = Get(10);**

In the former case, the seismic line is identified by its unique number or name, and in the latter case, the search is done by its regular number in the work seismic crew (needs to be identified in advance). In case of executing the command «**profile = GetUn**», the work seismic crew needs not be identified.

The keyword **picket,** and functions «**GetUn(NPK)**» and «**Get(PK)**» allow identifying the work picket, as follows:

picket = GetUn(1253678); or **picket = Get(1000);**

Identification by the unique number can be done without preliminary identification of seismic crew number or seismic line number: identification by the regular number can only be done with the work line (profile) initiated.

Apart from keyword **picket** there are keywords **first_ picket, last_ picket**; which are intended to identify a segment of the seismic line and get initiated the same way as **picket**.

Key word «**end**» is intended to reset seismic crew, seismic line, stake, etc.

The local weights setting functions are divided into several types: functions, working with the whole data set, and those working with current seismic crew, seismic line and picket.

Functions working with entire data set:

AddWeight(Col, wei); - adds weights column, the first parameter being the column name, the second being the clearly stated weight ratio.

SetWeight(wei); - sets established weight into the weights column.

SetWeight(wei, polID, io); - sets weight (first parameter) by polygon, identified by its ID (second parameter). Third parameter can be either «**in**» or «**out**», meaning that established weight can be set either in, or outside, the polygon.

SetWeightMask(polID, wei, N, dist); - sets weight (second parameter) inside polygon identified by its ID (first parameter). Being close to the polygon boundary less than the distance «**dist**» the weight gradually diminishes by **N** times. The procedure minimizes peripheral effects while mapping dense point data, e.g. 3D seismic data.

SetWeightByYear(wei, year1, year2); -sets weight (first parameter) by year of shooting seismic (the two next parameters). Weight is set for seismic crews and years from **year1** through **year2**.

CutEndings(d); - sets local weight at **0** at distance **d** from the beginning and from end of all seismic lines.

Disperse(d); - disperses too dense seismic data where **d** is the minimum allowable distance between pickets.

Weights setting functions for work seismic crew:

SetWeightCrew(wei); - sets established weight on crew identified as **seiscrew**.

SetWeightCrew(wei, polID, io); - sets weight on work seismic crew (first parameter) by definite polygon (second parameter). Third parameter can have «**in**» and «**out**» views.

SetWeightMaskCrew(polID, wei, N, dist); - sets weight by rules identified in **SetWeightMask**, but only for one seismic crew.

CutEndingsCrew(d); - sets local weight on work seismic crew at **0** at distance **d** from beginning and end of its seismic lines.

DisperseCrew(d); - disperses too dense seismic data within one seismic crew field where **d** is the minimum allowable distance between pickets.

Weights setting functions for work seismic line:

SetWeightProf(wei); - sets established weight for seismic line identified as **profile**.

SetWeightProf(wei, polID, io); - sets weight for work line (first parameter) by definite polygon (second parameter). Third parameter can have both «**in**» and «**out**» views.

CutEndingsProf(d); - sets local weight on work line at **0** at distance **d** from its beginning and its end.

SetWeightProfStart(wei, d); - sets local weight (first parameter) at distance **d** from beginning of work line.

SetWeightProfEnd(wei, d); - sets local weight (first parameter) at distance **d** from end of work line.

DisperseProf(d); - disperses too dense seismic data within work line where **d** is the minimum allowable distance between pickets.

For pickets:

SetWeightPicket(wei); - Sets established weight on work picket identified as **picket**.

SetWeightProfRange(wei); - sets local weight on fragment of seismic line between pickets identified as **first_picket** and **last_picket**.

2.3.12. Loading structure statistics

This build method is entitled «**Structure statistics**» in the list of table build methods. The table only allows referencing to grids, with at least one of the grids featuring structures in it. As a result each row in the table will correspond to one reference grid.

The first two columns contain geometric center coordinates of the grid definition area (polygon in general, rectangle in particular). The content of other columns depends what statistics of structures user is interested in. In the build parameters dialog specify the value type (positive or negative), property (amplitude, area, volume) and nature of calculated units (amount or value).

The table will show distribution of structures by selected property. The rules of identifying interval boundaries for all properties are the same: 0-10, 10-20,…, 90-100, 100-200,…, 900-1000, 1000-2000, etc. The range of interval in which the property value of at least one structure is found, is recorded in the name of a table column. For instance, if one were to select amplitude of positive structures and the units of amount as properties, a certain interval of amplitude will correspond to a number – the relationship between the number of structures having amplitudes with the selected range and the total number of positive structures. If the units of value were to be selected, it is the relationship of the sum total of amplitudes with the selected interval to the sum total of all amplitudes of all structures. Clearly, the sum total of units in each row of the table is 1. The table will feature only those intervals in which at least one structure selected by a certain property from the grid references. The user needs to remember that statistics are calculated with allowance for the filter of structure analysis in a given grid (see «**Structure analysis**» for more detailed discussion).

2.3.13. Create with calculator, loading results of Statistics

These two methods will be discussed later, in the sections «**Create with calculator**» and «**Statistics Objects**», respectively.

2.3.14. Net-to-gross ratio for saturated thickness

This method involves construction of a table based on the "**Deposit model**" object. The resultant table represents a copy (duplicate) of the table with wells data, which was used in building the "**Deposit model**", with additional columns containing values of net pay-to-gross ratio of the reservoir. For example, for the wells located in **Pure-oil zone**, this is just gross sand ratio and for the wells in the Water-oil-zone – the ratio between the effective net pay thickness and the distance from the collector top to the contact surface.

These coefficients are entered into two columns with the predefined names - «CFO» and «CFG», for oil- and gas- components of the deposit, respectively.

The NTG table is based on the "**Deposit model**" object built in **GST** Version 6.6.7 or later. If an object was created in the earlier versions, it must be rebuilt.

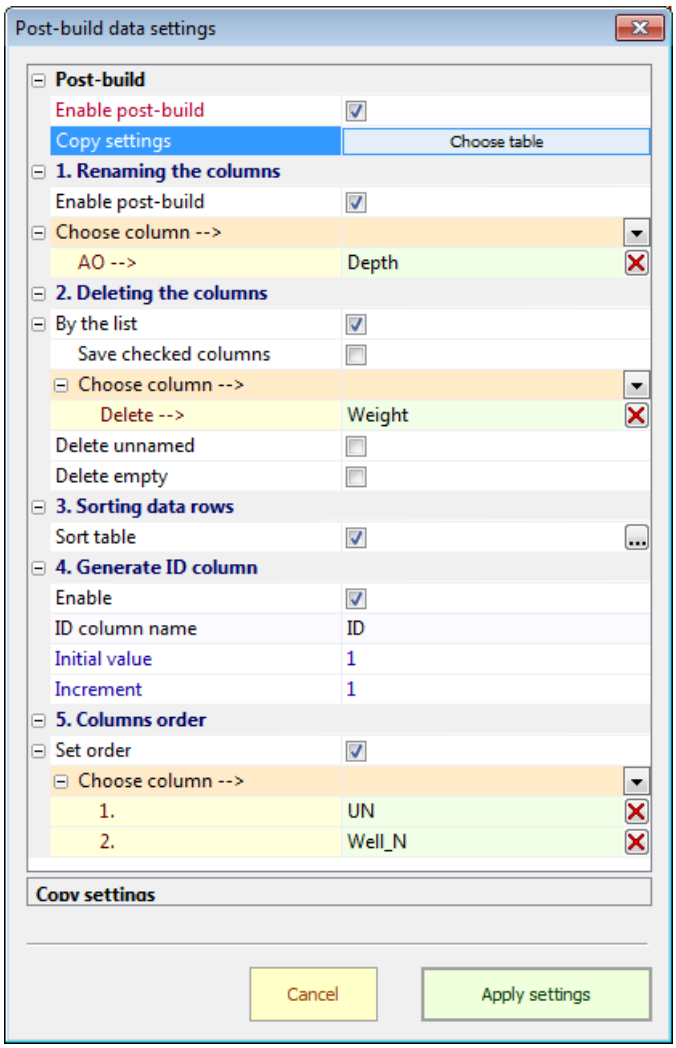
2.3.15. Post-Build operations

Post-build means a set of operations on the table performed in the automatic mode, after the object building is completed. Post-build allows you to:

- Rename the column names in the table;
- Disregard unnecessary columns or keep only those desired (in the list);
- Sort columns and rows in the table, etc.

Post-build can be enabled or disabled in the table properties dialog or in the hierarchy tree, as demonstrated in the figure below.

Double-clicking on the corresponding icon in the hierarchy tree, or pressing the button (yellow star) in the properties dialog opens a 'parameters post-build' dialog.



Post-build option is disabled by default; if activated, a set of items becomes available.

1. Renaming the columns. When this option is enabled, the user can create a list intended of the table fields to be renamed. For this purpose, select the previous name (e.g. «**AO**») from the drop-down list "**Choose column**" → and assign it a new name (e.g., «**Depth**») in the line below. If the drop-down list with the field names is empty, the column name can be typed directly into the input field of the control element. If needed, delete a row from the list by clicking the cross-sign button.

2. **Deleting the columns**. Columns can be deleted upon the list compiled by the user. It is created according to the same rules, as the renaming list. "**Save checked columns**" option either deletes the listed columns from the table, or leaves them and deletes all the rest. You can also delete from the table all unnamed or "empty" columns.

3. **Sorting rows**. This option allows you to apply automatically all the possibilities provided by "**Special sorting**" in tables. Sorting parameters are set in the sorting properties dialog.

4. **Generate a column ID**. With this option, you can assign a unique number to each row of the table and place it in a separate column. For this purpose, specify the name for the new column, the initial value of this number, and the increment. Numbers can be either integer or fractional, and the increment can also be negative.

5. **Columns order**. Using this option, you can position the columns conveniently. For this purpose, create a list of sequence of columns, according to the general rules. If the list does not include

all the fields in the table, the column sequence will be first formed according to the list, and then – according to the original sequence.

Post-build settings can be copied from another table: press the button "**Choose table**" next to the corresponding option.

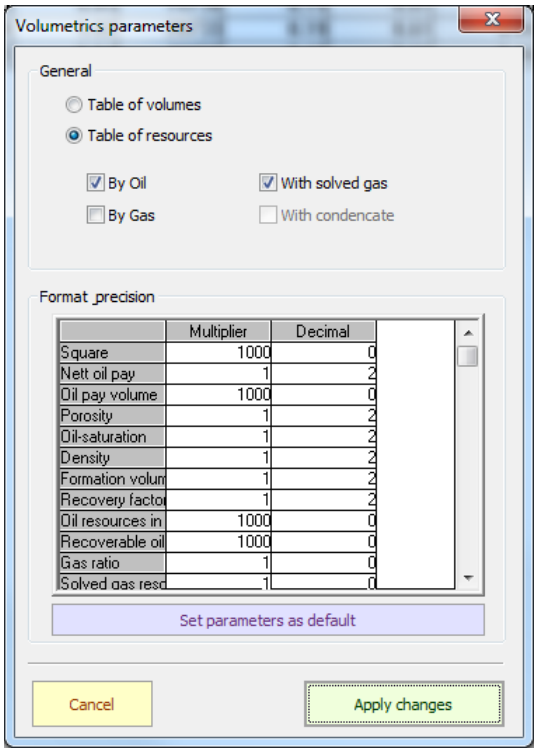
2.4. Building a resources table

Volume calculation as well as oil and gas resources calculation are released in **GST** as table build methods. Thus, one can get a table that contains volumes, acreages and weighted average values of estimate parameters distributed by areas, saturation zones and resource categories as a result of calculations performed.

2.4.1. Preliminary steps

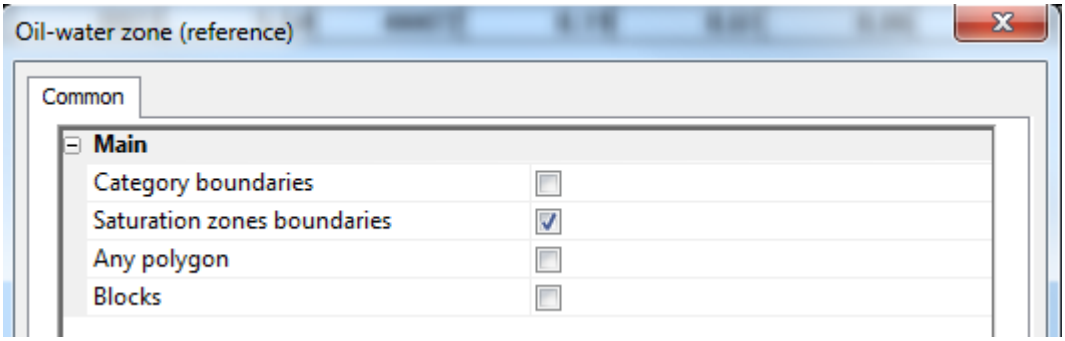
Add a new table to the project tree and set a build method as "**Creating table of resources**". Then, in the build settings dialog (below) specify what exactly should be calculated: volumes or resources, oil and/or gas, with or without condensate and solution gas.

In the lower part of the dialog box under "Format precision," specify measurement units multipliers for each estimation parameter and the decimals number. Measurement units multipliers allow the conversion of the computation results into the required units of measurement.



Then add the necessary references to the table. The set of added references may change depending upon the build details selected by the user. However the major elements of the set are as follows:

Resource zones. User should add a reference of a cover of **"Resource zones"** type. This reference is obligatory. The table cannot contain more than one reference to resource zones (otherwise estimation will be performed by the first reference encountered).

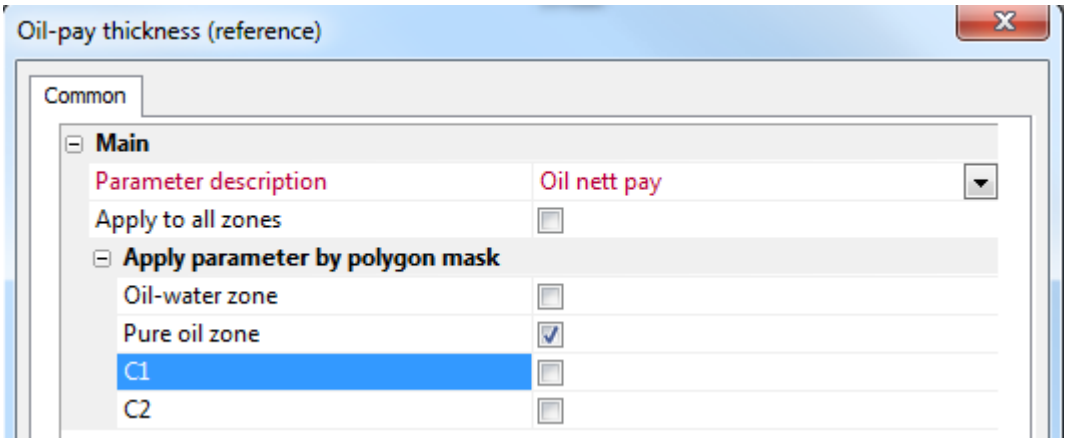


Saturation zones and resource categories. Saturation zone and resource category boundaries should be included in **Cover** objects (of a "**Polygon"** type). References to these objects are added when the resources estimated are to be summed up by the categories and zones. To do this, in the reference property dialog box (above) it is necessary to indicate what exactly the given boundary refers to. If an **"Any polygon"** flag is set, resources will also be calculated separately within the said polygon (with account of saturation zones and categories).

Blocks. By setting this flag we make the program include total resources contained in each block (with account for saturation zones and categories) into the resulting table.

Thickness contours. A reference to gross thickness contours is obligatory, if the user wants to add the results of calculations by pay intervals to a volume table, then it is necessary to include the option "**Thickness Grid**". A reference to thickness contours is to be added through setting of the option "**Thickness contours**" in a reference properties dialog box.

Volumetric parameters. Calculation parameters can mean any quantitative characteristics, such as: gross thickness, oil and gas saturated thickness, porosity, permeability, recovery factor, etc. Calculation parameters can be represented both by grids and numbers. One or several **GST** objects correspond to each type of parameter. If a calculation parameter is a number, then its value is specified in a separate table consisting of one row and one column. References to all volumetric parameters should be added to a resources (volumes) table.



Starting with version **GST 6.6** all required settings can be set in a reference properties dialog (to calculation parameters) which makes subsequent filling of the parameter table much easier (see Fig. above). The user shall specify in the **"Parameter description"** area to which parameter the reference corresponds: oil thickness, porosity, etc. Below are the options controlling application of the said parameter. A parameter can be used both for all resource zones and within the selected polygons only. The settings made in the references during table building will be used automatically.

GST also determines parameter description and application automatically to download any projects saved in previous versions.

2.4.2. Resources and volumes estimation

Oil, gas and volume table building procedures differ with a number of volumetric parameters specified.

Having a build method and details specified and needed references added, and initialized, the user should start the build process (clicking the button  in the toolbar). Then, a table will pop up in the object's working window as below:

Zone	Cat A	Cat B	Cat C2	Oil-pool	Water_oil zone	g_h	g_Por	g_Sat
Zone №1	-	-	+	+	-	Thickness (Oil-pool)	porosity	saturation
Zone №2	+	-	-	+	-	Thickness (Oil-pool)	porosity	saturation
Zone №3	-	+	-	-	+	Thickness (water-oil)	porosity	saturation
Zone №4	-	+	-	+	-	Thickness (Oil-pool)	porosity	saturation
Zone №5	-	-	+	+	-	Thickness (Oil-pool)	porosity	saturation

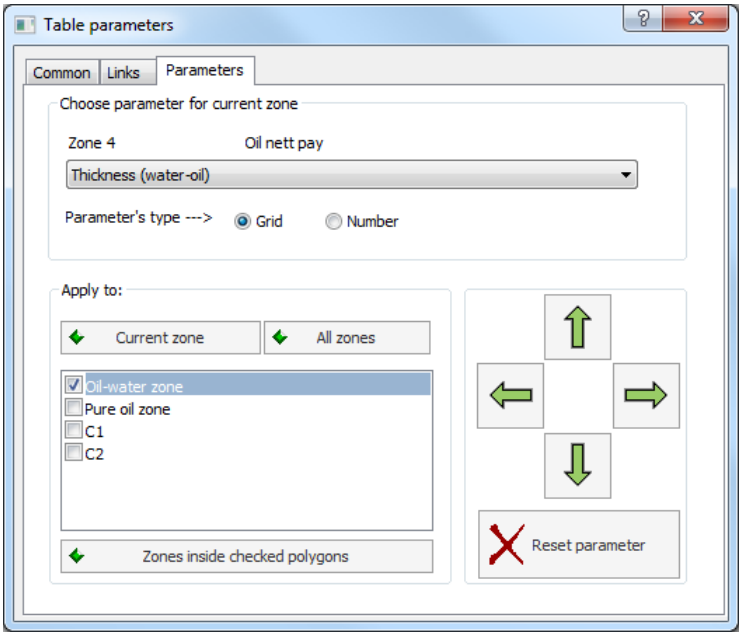
The first column in the table contains resource zone numbering information, while the other six show what "theme" the given zone is belongs to. "Themes" are the polygons that define boundaries of categories, saturation zones, license blocks, etc.

A sign «+» indicates the zones inside a theme, while a sign «-» marks those which did not. The last columns contain the information about oil thickness, porosity coefficient, saturation and so on.

In general, each zone can be assigned its own volumetric parameter, as one can tell from the table. At that, a parameter type can differ for various areas as well. For example, for resource calculation, a porosity ratio for the first zone can be specified with a numerical value, for the second zone with a grid, for the third one with an absolutely different numerical value or a grid. Thus, in the table shown above a net pay for the area #3 is represented by a grid calculated within a water-oil zone, while the remaining four areas are represented by a grid defined within purely oil zone.

User sets the volumetric parameters in page "**Parameters**" in the table properties dialog (see below).

- Activate a property dialog and open the page "**Parameters"**. Left click on a table cell in a column with the parameter being specified, which is assigned to a certain zone. The page will show the information needed: zone number, name of the estimate parameter. If a parameter has been specified earlier, you will see its type (number or grid) there as well; and a drop box will show a name of the object that contains this parameter.
- If a parameter has not been specified yet or it should be replaced, then specify its type by clicking the "**Grid**" or "**Number**" buttons, and select the object needed in the drop-down list.
- The settings made can be applied to a **current selected zone**, **all zones**, or to the **zones inside checked polygons**. In this last case it is necessary to check polygons in the list. Click the button **"Reset parameter"** if you need to clear a parameter column.
- Use arrow-buttons for convenient parameter table navigation.



If both the parameter description and application polygons are set in references, **GST** will fill in the parameters table automatically in accordance of the settings made. In case the parameters are not set, the user should set them "manually" as shown above. If application polygons are not set for any parameter, but the table contains only one reference to a parameter with such description, it will be assigned the flag **"Apply to all resource zones"** automatically.

When all volumetric parameters are assigned, user can start the resource calculation process. Press the button in the toolbar, and then you will get the calculation results will pop-up on a screen. Switching over between resource table and parameter table is performed by the buttons and , respectively.

2.5. Rectangle object

It is a table of four columns and one row containing vertices coordinates. Rectangle is an auxiliary object designed to determine the object load and save, grid mapping area and so on.

2.5.1. *R*ectangle basic build methods

- **Building rectangle "manually"**. This is a default method. An empty table with **Xmin, Xmax, Ymin, Ymax** fields displays after the menu command **"Object → Start build"**. The user sets the vertices coordinates and completes the build.
- **Linking to other project**. This method works in a similar way to the corresponding methods for other objects.
- **Use other rectangle.** Different rectangle data are copied to the rectangle by reference. This is done by setting the appropriate method, adding a reference to the relevant rectangle, and starting the build process.

Coordinates of the rectangle can be easily placed or copied from the clipboard using commands "**Edit-> Copy**" and "**Edit - > Paste**". You can create a buffer rectangle in the window of any grid or

cover by enabling "**create a rectangle**" mode ( button on the toolbar). If a rectangle is created in the grid window, its coordinates are multiples of the grid step.

2.5.2. Use of a Rectangle object

A **"Rectangle"** object is used for building and visualizing other **GST** objects by adding it as a reference.

- **Import-export area**. There is an option to select **"Rectangle"** as a limiting area in an import-export dialog, link dialog and load from the database dialog. Meanwhile, the reference to a rectangle becomes significant when it is used for loading data, however, and it is insignificant while saving data.
- **Grid mapping area**. A rectangle can be selected as a grid mapping area. So, the area updates automatically as the rectangle changes. This option is set under tab **"Grid"** in the **"Grid builder"** dialog.
- **Visualization covers area**. The rectangle of cover visualization is extended automatically by all applied references by default. It is possible to limit this area by Rectangle dimensions. Add reference to rectangle and choose option **"Use as cover drawing rectangle"** in the reference parameters dialog.
- **Convert to cover lines**. Rectangle can be added to edit cover layer as one of the lines. User have to choose option **"Add to edit layer"** in the reference parameters dialog box.

3. Cover object

There are three cover types in the **GST:**

- **Simple:** an arbitrary set of lines and contours (closed lines), which has certain attributive information. It can include maps shown as contours, fault lines, etc.
- **Polygon:** a closed, most possibly multilinked area (with "gaps" and "isles"). A polygon consists of non-overlapping contours, and is used, for example, for mapping tasks inside of a definition area.
- **Resource zones**: a numbered set of polygons.
- **Deposit boundaries**: a special type of cover created for automatic finding of saturated zones for flat-dome shaped deposits. The object contains a set of unedited polygons and lines (it does not have an edited layer) and can be used for building other objects.

To change the cover type, choose the corresponding option in the category **"Cover attributes"** under the tab "**Common**" in the object properties dialog box or under the tag **"BUILD SETTINGS - Type"** in the hierarchy tree. A simple cover is described below. Methods used for the other two types are discussed at the end of this chapter.

3.1. Basic cover building methods

3.1.1. Create with editor

Set the build method as "**Create with editor**" and then specify a cover rectangle. There are two ways of setting a rectangle: either by creating a reference to any other object, and in this case a cover rectangle will be automatically extended to the size of the object, or manually – in the category **"Cover attributes"** under the tab "**Common**" in object properties dialog box.

After this procedure the editing modes become accessible (see "**Basic cover operations**" for details). In the end user must press  button on the toolbar to complete build process.

3.1.2. Create by references

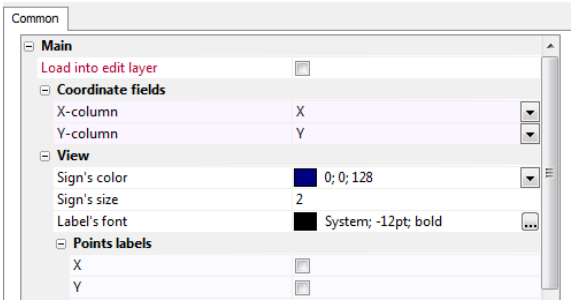
This method converts rectangle, grids and table objects into cover lines, copying the contents of other covers and place the results into edit layer. To build the reference-based cover, the user should specify the proper build method and add required references to the object and initialize them.

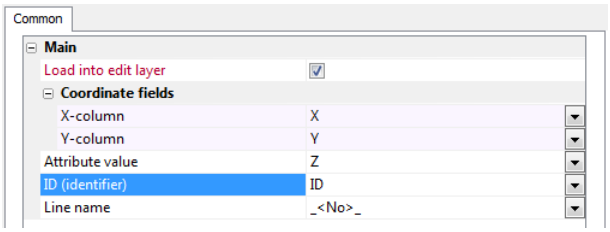
Reference to a table. Table data can be used either as a graphic layer (for visualization), or added to the edited layer, being transformed into cover-line coordinates.

To show data table contents in a cover's work window as points, indicate coordinate fields in the reference parameter dialog. Also set a color and point-sign size.

Check desired positions in the list **"Labels"** to display point parameters on the screen. Use the button **"Fonts"** to select label fonts.

When the option "**Load into edit layer**" is enabled, the values from coordinate columns are converted in to cover lines. At that, a drop-down list box becomes accessible, in order to define the column with a line attribute (**Attribute value**), unique identifier (**ID**), and line information **(Line name)**.

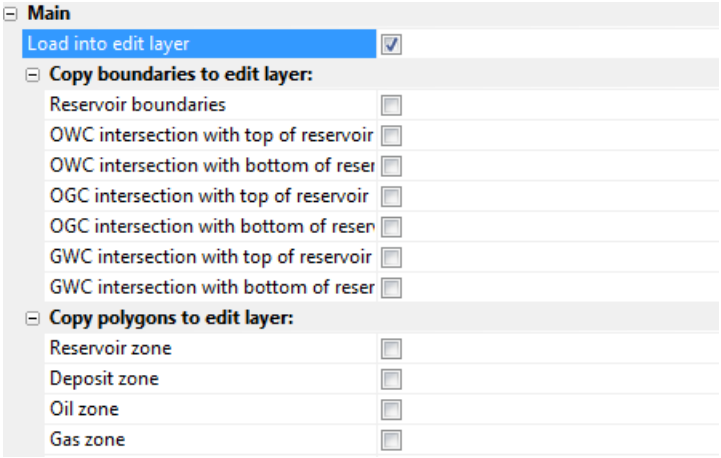




Reference to a cover. In the reference-parameter dialog user should specify, whether a reference cover is to be loaded into the edit layer or simply used as a graphical layer. If the first option is selected, the cover lines are loaded to the edit layer with all their attribute information. In the second case, the cover participates only as a graphical layer for object visualization, but it is not accessible for editing.

The option **"Cut lines by polygon"** becomes accessible with a polygon reference which allows keeping the lines copied into the edited layer within the given polygon only.

If reference cover type is **"Deposit model",** all deposit boundaries and polygons (Fig. below) can be transferred to the edited cover layer.



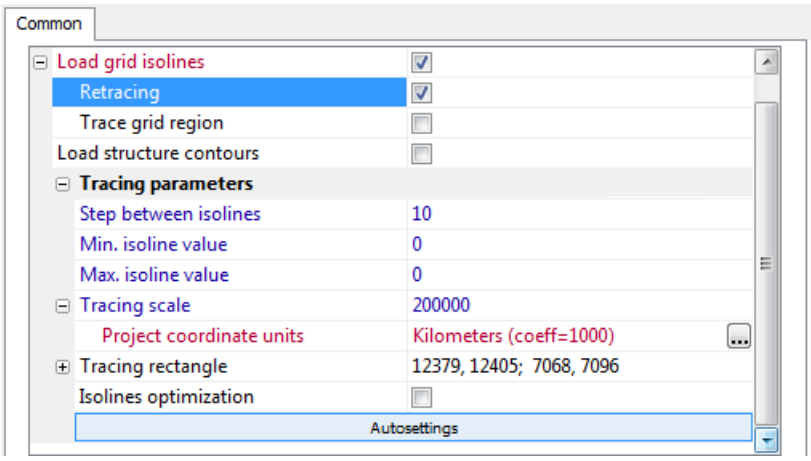
Reference to grid. The grid data can be either used as a graphical layer or loaded into the edit layer (button "**Load grid isolines**"). In last case GST extracts grid level lines (isolines) and put them into edit layer.

The option **"Retracing"** becomes available in the second case. If this option is disabled, the isolines traced in the grid work window are loaded into the edited layer. One should keep in mind that the isolines approaching the boundary of the mapping area are closed at the grid rectangle.

It is recommended to enable the "**Retracing**" option and set the specific tracing parameters to get a high quality contour map. The button "**Auto settings**" sets tracing parameters according to a grid rectangle and tracing step grid.

The parameter "**Tracing scale**" is responsible for tracing accuracy (smoothness). The larger the scale is (the lower is the number) the higher grid accuracy is, however there is more time required for tracing. Thus, one should keep in mind the purpose of the tracing procedure when selecting a value for this parameter. A scale can be selected rather small, if it is a preliminary stage of construction. When selecting the scale, one should make sure that the measurement units of the traced grid are the same as the measurement units set forth in the project properties.

Option **“Grid area tracing”** became available in the grid reference starting with version **GST 6.6,** which allows creating a polygon which includes an area where the grid can have a higher or lower value versus the one preset. In order to do that, one should set the parameter **“Tracing scale”** and the level value of the traced contour.



Option **“Isolines optimization”** allows to get a contour containing a minimum number of points required to represent the surface levels with a given smoothness. It makes sense to use this option when the contours are used for representation, however if these lines are used as input data for map building, this option should be turned off.

“Load structure contours” switch allows copying a cover, consisting of the grid structure contours, into the edited layer according to the filter selected in the grid. Loading of the grid contours is impossible during the loading of the structure contours, and vice a versa. Two references to one grid should be arranged if there is a need to have both covers loaded simultaneously.

Reference to rectangle. A rectangle is converted into a closed line and is added to the cover edit layer, if option **“Add to edit layer”** is on in the properties reference dialog box.

A cover created with this method may contain several references of various types. Lines obtained from the corresponding objects should be added to the edited layer in succession.

3.1.3. Create with Calculator

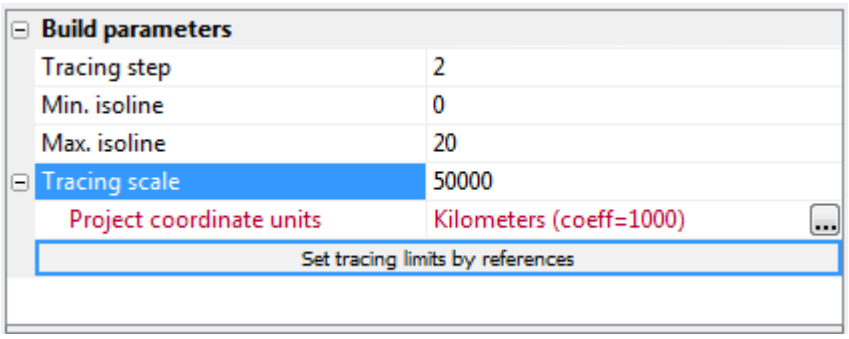
This build method in the first place refers to “**Polygon**” covers, when a result is presented by polygons merging, overlapping or subtraction. If the user deals with a simple cover, he/she can, for example, perform operations with line’s attributes. For more details of **Calculator** operations refer to Section **“Creating objects with Calculator”**.

3.1.4. Tracing composite grid

This build method is very efficient for tracing of net pay map contours. Under the method several grids can be traced within their **definition polygons**, with further contours merging at adjacent boundaries of these areas.

The **GST** technology assumes separate construction of thickness grids for purely oil-bearing zones and water-oil zones. A net pay map is created through net-pay grid tracing within a purely oil zone and then within a water-oil zone with further merging of the contours at a WOC contour.

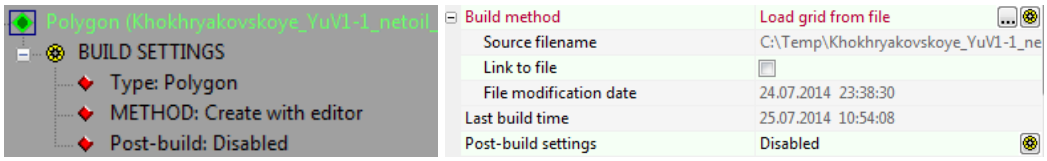
To perform this procedure, it is necessary to define a proper build method for a cover assumed and add references to the traceable grids. Each of these grids should have a **definition polygon**. The build parameter dialog will look as follows:



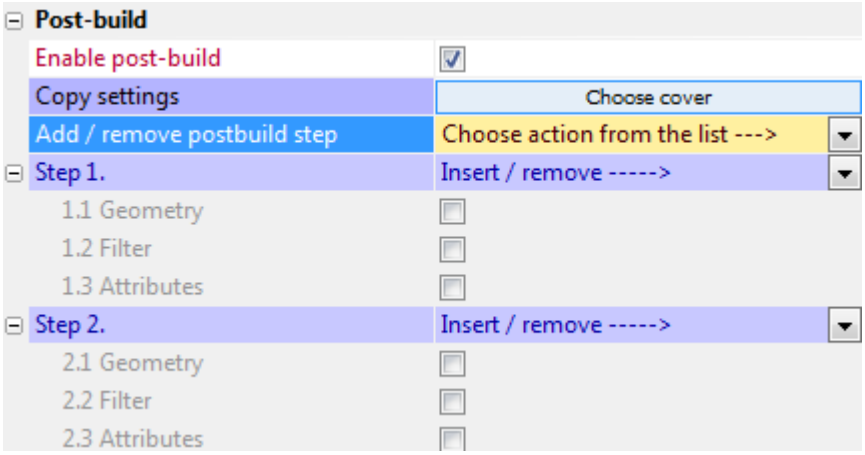
Spacing, tracing limits and scale (lines smoothness) common for all grids are to be set in this dialog.

3.1.5. Cover post-build

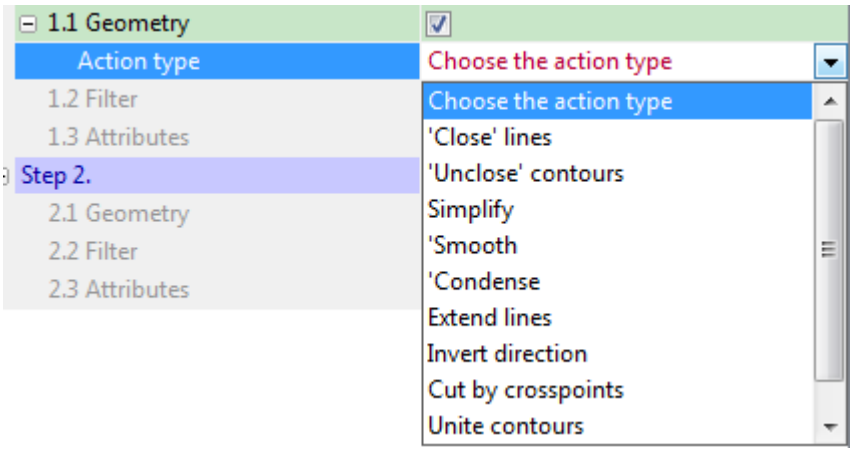
Cover post-build – is a set of operations on lines and polygons that are performed after the procedure of object building. I.e. post-building can replace some operations performed in the object edit mode. You can enable or disable the post-build in the properties dialog or in the hierarchy tree, as shown in the figure below.



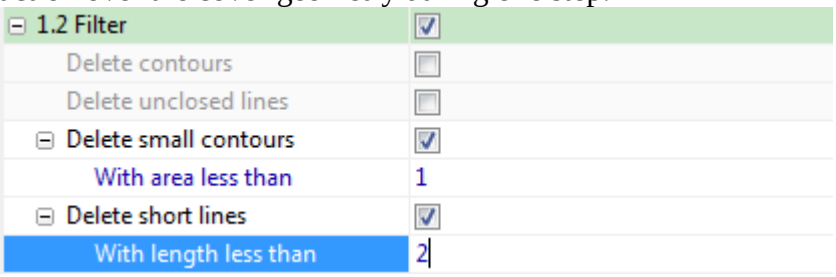
Double-click on the corresponding icon in the hierarchy tree or press the button (yellow star) in the properties dialog to call post-build parameters dialog. Post-building is disabled by default: when this option is activated set of several items is available.



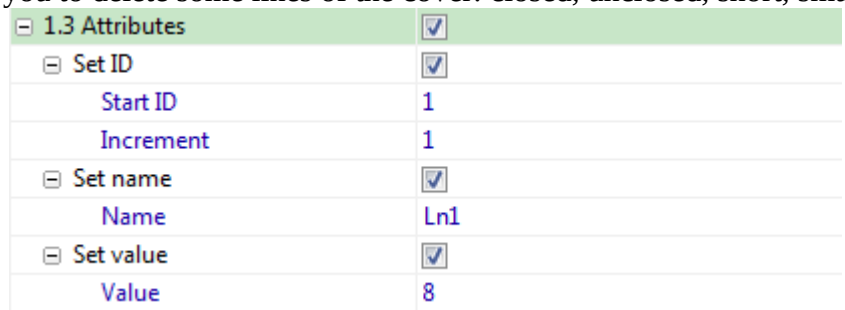
The post-building process consists of a set of steps, number and sequence of which is determined by the user. Each step can contain actions related to the line and polygon geometry, their "filtering", working with numeric attributes, names, identifiers. Each step is connected with user menu that allows you to delete a step, change the sequence, insert a new step, etc.



Option "**Geometry**" offers a set of actions with lines and polygons. The user must select one of them and, if necessary, set additional parameters: condensation, optimization, smoothing step, etc. You can only perform one action over the cover geometry during one step.



"**Filter**" allows you to delete some lines of the cover: closed, unclosed, short, small contours.



The third option "**Attributes**" allows you to set a numeric attribute for all lines, specify a name, and generate unique identifiers.

3.2. Cover visualization

3.2.1. Line visualization

The menu command "**Mode → Show ending points**" (the button on the toolbar) activates/deactivates a mode of line boundary-point display. The command "**Mode → Show inner points**" (the button on the toolbar) enables/disables a mode of line inner-point display.

The display of cover points can be activated/deactivated in the category **"Edit layer's lines drawing"** on the main page of the object properties dialog. The boundary and inner point's size can be also set there. Such visualization parameters as: color, width, line style are also set in the same

category. There are five standard line styles available for a **"Cover"** object, however the lines will be always shown as solid if the weight is more than 1.

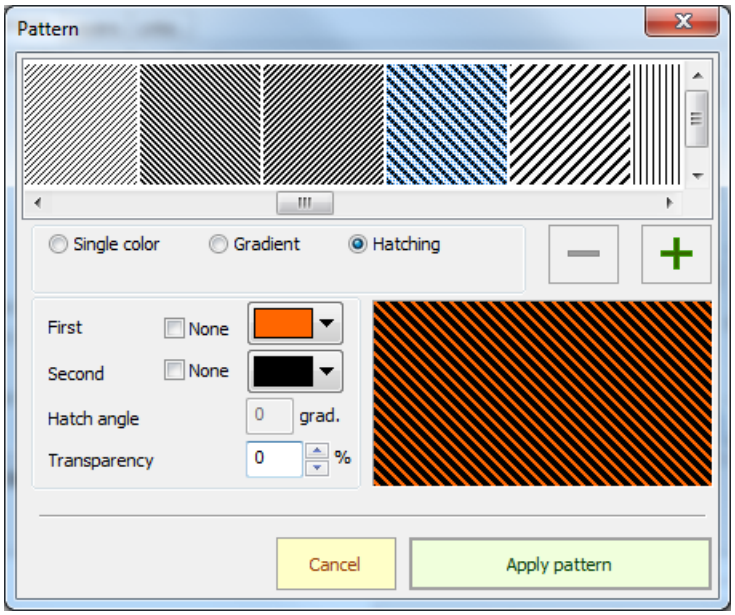
It is also possible to show/hide line labels, set their fonts and color. Both numerical attributes and the line names which are set for each cover line in the **"Line attribute"** dialog box can be used to show/hide line labels.

3.2.2. Polygon fill

Option **"Draw filling"** is preset as "No filling" by default. In this case the cover is displayed as a set of closed and unclosed lines. The user can set a polygonal fill to display a cover of a **"Polygon"** type.

The user can call a fill dialog (below) pressing expand button "…". There are three ways fill polygon:

- **Single color**. User selects a color and transparency level (from 0 to 100 percent – the range from nontransparent to completely transparent).
- **Gradient**. Polygon is filled with smooth transition from color #1 to color #2. The user selects colors, transparency level, and the fill angle of tilt (angle is taken from the horizontal line downward to the right). .
- **Hatching**. Polygon is filled by means of one of the templates shown in the upper list. The user selects a hatch lines and background color, transparency and template itself. Any picture in .jpg, .bmp, and .emf format can be added (using the button **"+"**), and it can be deleted from the list by **"-"** button in addition to standard templates (which cannot be deleted). Templates (graphic files) are stored in the service directory "**…/Graphics/Hatch"**, which is installed together with **GST** program.



If the cover grid level lines (closed lines), the user can set **"Fill levels"** mode and display a cover similar to the grid in accordance with the values of the line numerical attributes.

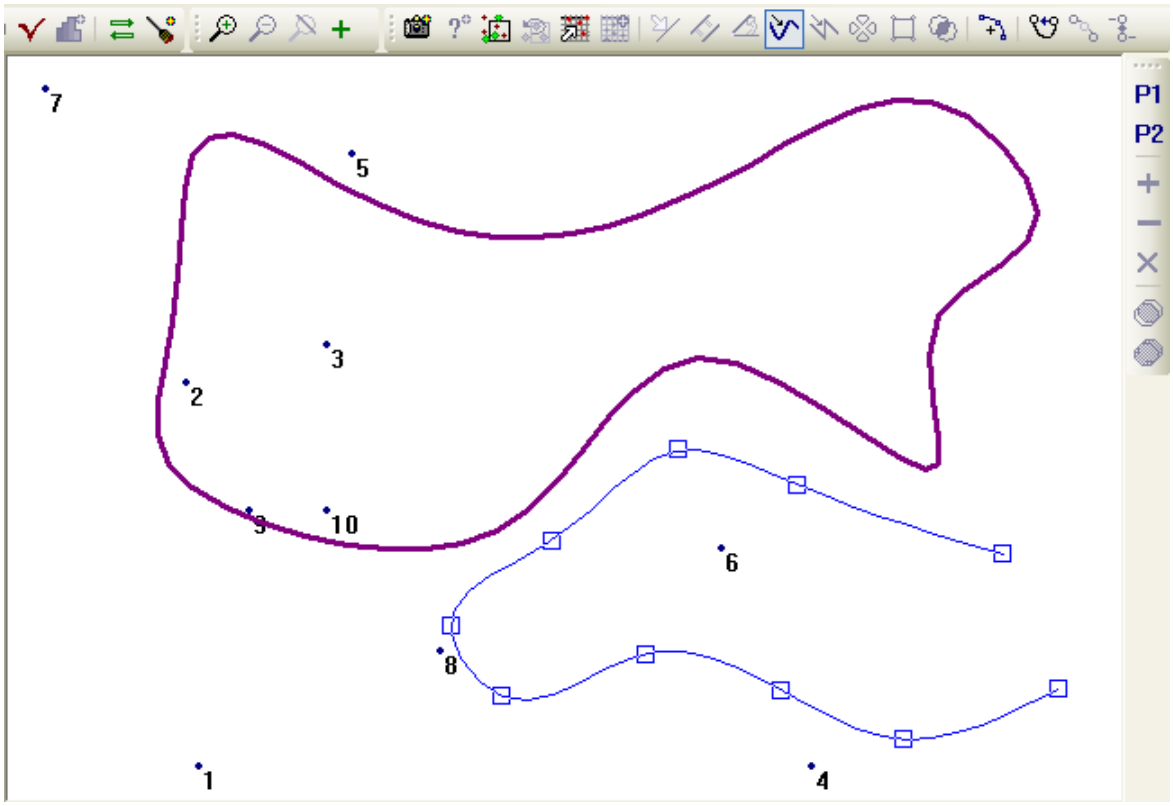
3.3. Edit modes

3.3.1. Spline-mode, polyline-mode

The spline mode is designed for adding new lines to the cover. To enable\disable the spline mode choose menu command "**Mode → Spline-mode**" or by press button on the toolbar. Line location is defined by the check point's location for a new line to run through. A check point is defined by pressing the left button of the mouse and its subsequent fixation with the right button (holding the left button pressed down). Gaps between the check points are automatically filled in by parametrical splines.

A check point location can be adjusted at any time by placing the cursor to the box of a check point, pressing down the left button of a mouse, holding it, shifting the point and fixing it by pressing down the right button. The new check point can be placed either at the end of the line created or at any other part of the line. If the last check point coincides with the first one, then the spline closes smoothly. Spline closing or opening can be performed by pressing **Ctrl + mouse right click,** as well as by the button on the toolbar.

A check point can be deleted by the command **"Delete spline node"** from the context (right click) menu.



Under **"Polyline mode"** check points are connected with straight lines. The mode is activated\deactivated by command **«Mode → Polyline Mode"** or by button on the toolbar.

Adding of a new line (spline or polyline) to a cover is performed by command **"Cover → Add object"** or by button on the toolbar.

Select "**Mode → Use control points**" if the added line need to pass exactly through the data points shown in a work window.

If only one line is selected and the user enables the spline mode (or polyline mode), the selected line points will be automatically converted into check points and become accessible for editing. In this case the check point frequency may become too high. Use command **"Disperse spline"** from the context (right button) menu to reduce the number of the spline nodes. This command can be used several times until satisfactory result is obtained.

New mode **"Select line's part"** became available starting the **GST version 6.5**, (button on the toolbar), which allows user to edit only a portion of the line. Point the cursor at the line subject to editing and select the required section by clicking the left mouse button. If the line to be edited is closed, one needs to click for the third time to determine which of the two fragments is subject to edit. After the fragment is defined, spline and polyline modes become available (,). The user performs editing under one of these modes and then applies it by using button on the toolbar.

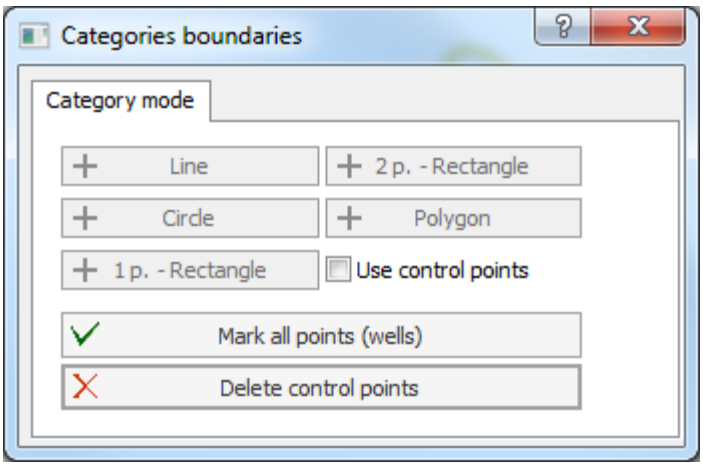
3.3.2. Line cutting

A line cutting mode is activated or deactivated by the command "**Mode → Cut line mode**" or button on the toolbar. Cutting of a line takes place at the cursor location with a left mouse button click. A new boundary point will be displayed.

If the additional mode "**Cut by internal points**" is invoked, then the line inner point closest to a cursor becomes a new boundary point. The mode is activated/deactivated by the menu command "**Mode → Cut by internal points**".

3.3.3. Category boundaries mode

Under this mode the user may add simple geometrical primitives to the cover (circles, lines, rectangles and polygons) and set their required dimensions (e.g., circle radius), and to "tie" them to some check/reference points (wells). The mode is activated/deactivated by the command "**Mode → Category mode**" or button on the toolbar. You will get the following dialog box popped up when this mode is activated:



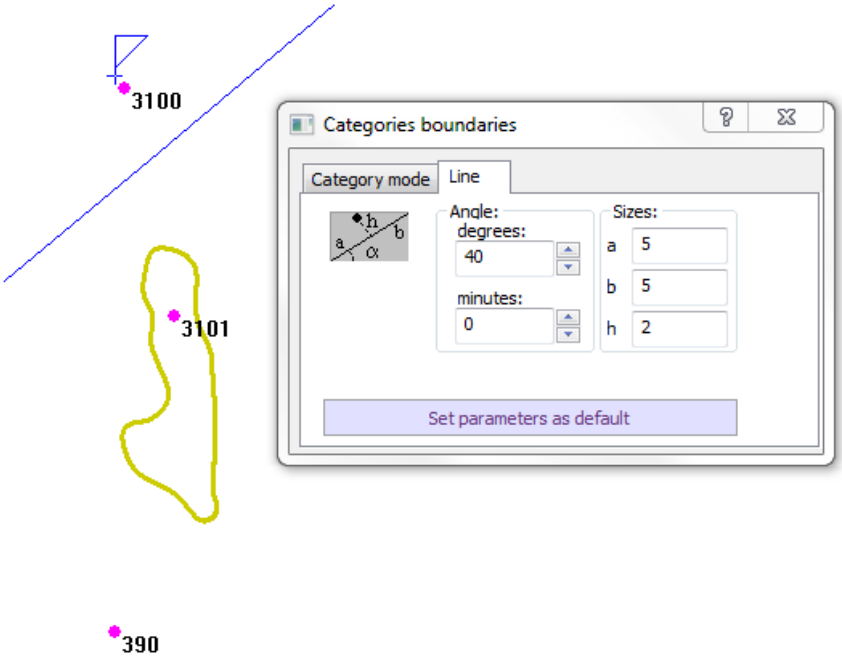
Procedure for the user to follow: 1) set reference points in a graphic window and some buttons become active; 2) click the button of the required primitive and set its parameters (dimensions should correspond to the units of measurements of the project coordinates); 3) add object to the cover. The

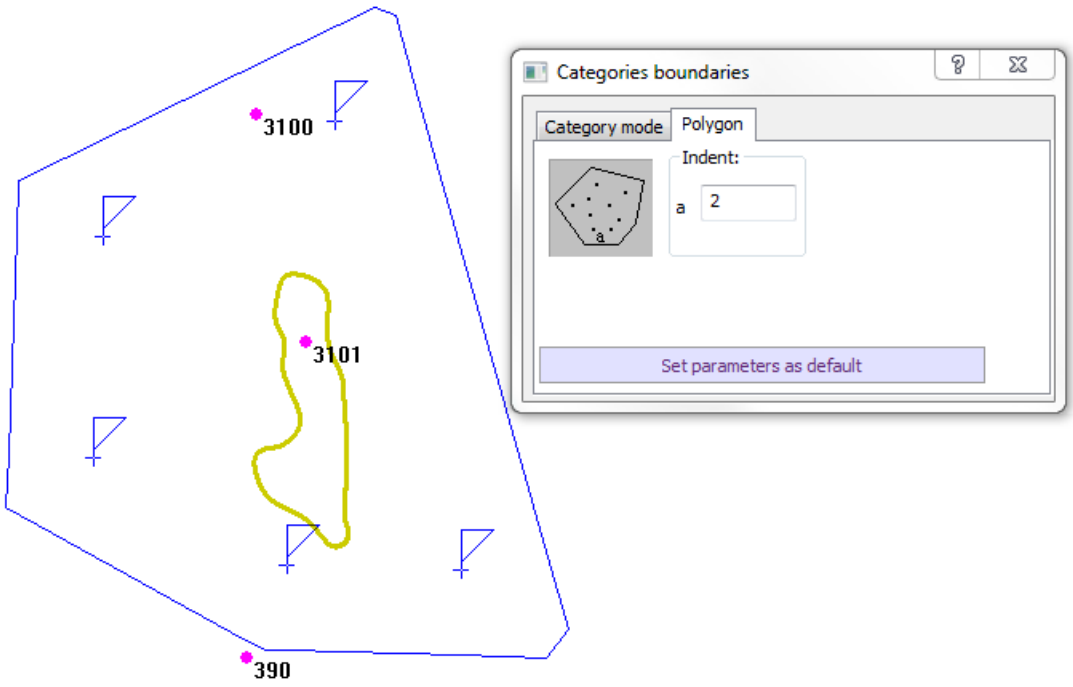
check points are selected by the left button click of the mouse and marked on screen with a special flag.

Point referencing. The command **"Use control points"** activates/deactivates a mode of referencing to certain specific points. When this mode is on, one of the cover graphic layer points is taken as a check point (a point corresponding to a line in a table, line node of the edited layer or graphic layer); therefore the object should have a reference to a table or other cover. If this mode is off, a check point is placed directly at the spot pointed out by the user.

Mark all points (wells). Check points are set for all table data (wells) displayed in the cover working window.

Line. The command (button) "**Line**" is accessible with one or several check points. This command represents the added object in a form of a straight line. Meanwhile, it activates the page "**Line**" in the dialog "**Category boundaries**", where the user can set the default size of the line and its location in relative to a check point (see fig. below). If there are several check points marked, the program will create an equal number of lines of the same size and direction.





Circle. The command "**Circle**" (accessible with one or several check points) creates circle object. It activates the page "**Circle**" in the dialog "**Category boundaries**", where the user can set the circle radius and points number (circle "smoothness"). If there are several check points marked, the program will create equal number of circles of the same radius and detail.

Rectangle by one point. Command **"1-p Rectangle"** creates a rectangle object and activates a page in the dialog to set default rectangle dimensions and its location in relation to the check point. If there are several check points marked, the program will create the same number of equal rectangles.

Rectangle by two points. Command **"2-p Rectangle"** also creates a rectangle object with reference to two check points. This command is available when two check points are defined, accordingly.

Polygon. Polygon building function becomes available when there more than two check points, which is then executed by the command **"Polygon"** (see fig. above)**.**

A convex polygon is created upon execution of this command which contains all check points in it. Just like in the previous cases, the corresponding page is activated in the dialog box, where the user can adjust the distance of the polygon sides from the check points.

Adding objects. The command "**Cover→ Add cover object**" (the button on the toolbar) translates the objects created under category boundaries construction mode to a basic (edited) layer.

3.3.4. Line moving and rotation mode

Buttons  on the toolbar for **line relocation mode** and **line turning mode**, respectively, allows user to change location of one or several cover lines with a mouse. One needs to select the required lines and switch over to one of the above modes. Line moving and rotating is performed by a left mouse click; click the right button to fix the line in terminal position. Rotating is performed relatively to the common geometrical center of the lines selected.

3.4. Basic cover operations

3.4.1. Selecting lines

To select a line press both **Ctrl** + **left mouse button**. To select several lines in a rectangle select this rectangle with the mouse, keep holding **Ctrl** button. **Ctrl** button may not be pressed, if the "**Select mode**" is enabled (the button  on the toolbar). Selected-line visualization parameters are set at the parameter dialog page **"Common"** under **"Selected lines**".

Besides mouse-aided selection operations, under the menu item "**Edit**" the following additional operations are accessible:

- **Select all.** All cover lines are selected.
- **Deselect all.** All cover lines selected are canceled.
- **Select all polygons.** All closed lines are selected.
- **Select all lines.** All unclosed lines are selected.
- **Invert selection.** A selection state of each line is changed to the opposite. .
- **Select by square.** Contours with the areas larger, smaller than or equal to the given one are selected.
- **Select by attributes.** Lines and contours with attribute values larger, smaller than or equal to the given one are selected.
- **Select by DBF**. If the cover is loaded from a shape–file with a non-empty dbf table, lines can be selected upon the relevant attribute information contained in dbf. This command starts a screen dialog similar to the table row dialogue.

3.4.2. Line operations

The line operations are accessible, if, at least, one line is selected. Commands described below are executable under the submenu "**Line operations**" (menu item "**Cover**").

"**Cutting line by length.** This operation is accessible, if one line is selected. While executing the command "**Cutting Line by Length**", the following dialog pops up:

In this dialog box the user can select a line cutting mode: at the given distance/length from an initial point (initially a total length is shown in the editing window) or by a given segment of the total length of a line assumed.

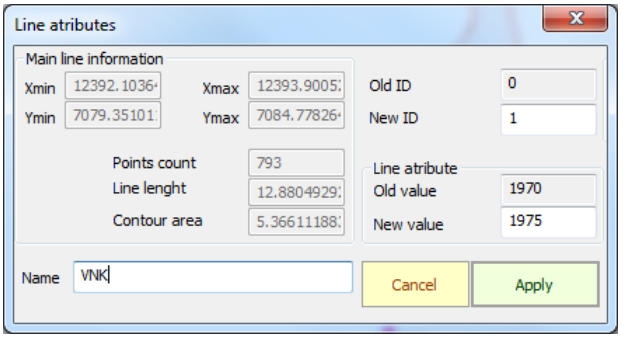
Line closing. The command **"Close/Unclose lines"** (the button on the toolbar) closes (adds the first point to the end of a line) all unclosed lines and uncloses (deletes the last point of a line) all closed lines. The command is performed for all selected lines.

Join lines. The command "**Join by same points**" (the button on the toolbar) views all the lines selected and combines those of them, which have boundary points with same coordinates. The command is accessible, if more than one line is selected.

Line merging. The command "**Merging by nearest points**" (the button on the toolbar) is accessible, if two lines are selected. This command merges two lines in one, connecting two nearest points.

Invert direction. The command "**Invert direction**" (the button on the toolbar) modifies a sequence of points in the lines selected. In a dialog window with cover parameters you can select a mode of initial points' selection to establish a visual control over line direction (**"Direction markers"**).

Line attributes. The command "**Line parameters**" (the button on the toolbar) is accessible, if one line is selected. The command activates a dialog, where the user can set a new value of line attribute.



Besides, it is possible to change the value of line ID and to introduce some line information (name), which can be shown as a label.

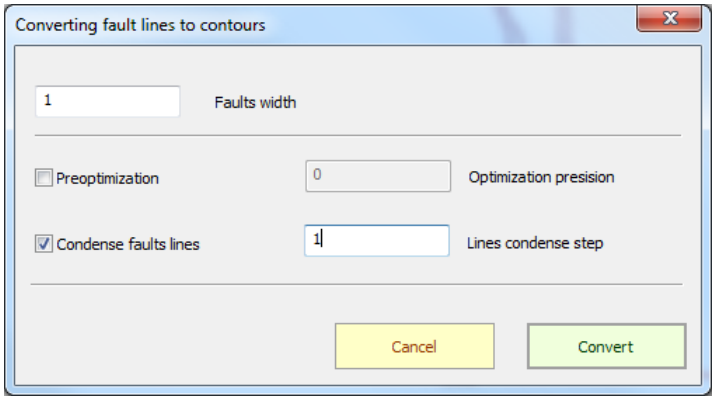
If user needs to assign names and numerical attributes for several selected lines at a time, it can be done from the main menu by the command **"Cover → Line operations → Set lines attributes"** and **"Cover → Line operations → Set lines names"**.

Line extension. The command "**Cover → Line operations → Extend lines**" becomes accessible with one or several lines selected. This operation allows to extend an unclosed line from its end points by a given increment (in project coordinate units). Selecting the indicated menu item in a popped-up dialog box and set an incremental value.

Line extension can be helpful during assembling of resource zones, in case some of the lines do not "stretch" up to the boundary, which they should essentially intersect.

Condense Lines. The command "**Cover → Line operations → Condense lines**" becomes accessible with one or several lines selected. This operation allows for enlarging a number of interim points of the lines assumed, and dense them to a spacing set by the user in the popped up dialog box. A condense spacing is set in project coordinate units.

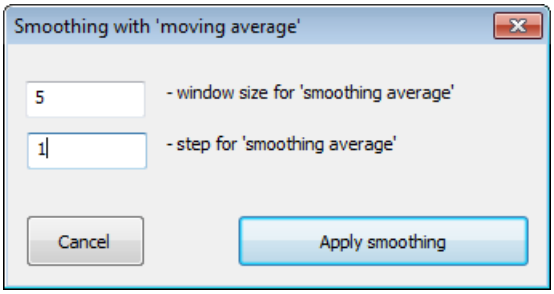
"Line → Contour" transference. This operation is required primarily to transform a fault from the "central line" into a contour; however it can be helpful to accomplish some other tasks as well.



Lines can be easily transformed into contours by selecting and executing command from the menu **"Edited layer → Line operations → "Line → Contour" transference**. The user sets a distance in a popped-up dialog box between the fault sides (**h**), line condensing step (**d**), and options of preliminary line optimization. Optimization is required if the fault lines are set by the excessive number of nodes. This procedure deletes interim nodes which do not have any impact on the line geometry. Parameter **"Optimization precision"** sets the distance limit for the optimized line to deviate from the original line.

Line optimization. Command **"Cover → Line operations → Optimize lines"** becomes available with one or several lines selected. This operation allows to obtain the most simple line out of the line selected (which has the least number of points) deviating from the original to the distance not exceeding the set value.

Smoothing lines. Command "**Coverage-> Operations with lines-> Smoothing lines**" is available when one or more lines are highlighted. The operation performs the smoothing by moving average. The dialog shown below sets the size of "smoothing window" and pitch. It should be noted that the pitch should not exceed the window size.



Deleting of selected lines without moving them into the clipboard (like during **"Cut"** operation) is performed by the corresponding command from the context menu when at least one line is selected.

It is possible to **delete nearest node** by the corresponding command from the context (right click) menu. The closest to the cursor node will be deleted.

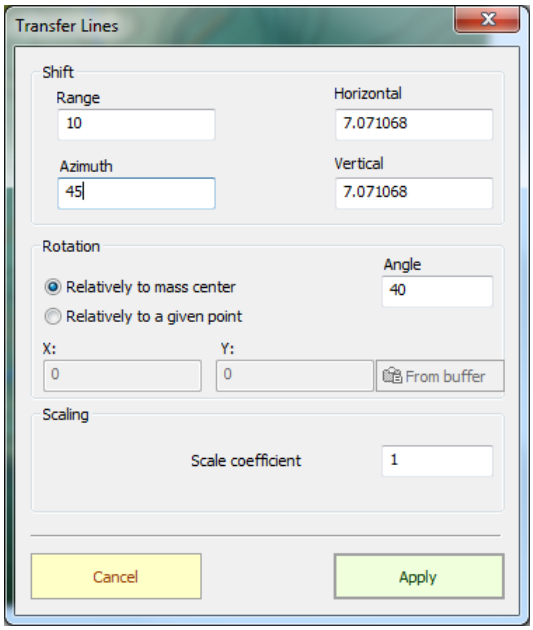
3.4.3. Cover operations

Cross points. The command "**Cut by cross points**" finds cross points and cuts the cover intersecting lines (regardless of the selection state). The command can be called by pressing on the toolbar.

Delete inside and outside contour. The command "**Delete inside contour**" (the button on the toolbar) is accessible, if one closed line is selected. At executing this command the **GST** finds cross-points and deletes all lines that have been found inside of a contour.

The command "**Delete outside contour**" (the button ⊗ on the toolbar) is identical to the command above.

Line transference. Menu command **"Cover → Cover operations → Transfer lines"** calls a dialog below. The user can specify parameters for relocation and turning of the selected lines.



Shifting is set either in Cartesian or polar coordinates, and a rotation (clockwise) is set in degrees and can be performed both in relation to common geometrical center and in relation to a given point, which can be taken from a GST clipboard (buffer). It is possible to place a point into a clipboard (buffer) by command from a context menu of cover working window by pointing a cursor at any point in object's work window.

3.4.4. Operations with polygons

If a simple cover is an arbitrary set of lines, a polygon describes a certain area contained inside contours (closed lines). Polygons can be presented by multilink areas, i.e. it can contain "gaps" and "isles", etc. Polygon type cover can be used as grid defining area, in Calculator operations and estimation of reserves when forming resource zones.

If an edited layer contains any contours' set, user may construct contour-based polygons, and perform multiple theoretical operations with them: unite, subtract and multiply (cross). The following steps should be taken:

- Select one or several contours that do not cross each other and name them as a polygon #1, (pressing P1 on a right hand side of the toolbar). Those contours included into the polygon will be marked by the orange color.
- Select contours for polygon #2 (the button P2). This polygon will be marked by the blue color.
- When both polygons area initialized, buttons + − × that correspond to polygons' conjunction, subtraction and crossing, respectively, become accessible.
- Press one of this buttons to perform the operation needed.

After the operation performed, the polygon-making contours are changed based on the results of calculation obtained. Other (not selected) contours or lines do not change.

Uniting or crossing of several polygons. You can find the area of uniting or crossing for the entire set of selected contours by selecting several closed lines in the cover window with and toolbar buttons on the right.

Cutting a polygon with lines. This option allows you to "cut" a polygon into several polygons using an arbitrary set of lines. You must do the following:

- Select one or more contours and declare them a polygon # 1, press P1 toolbar button on the right. Contours included in the polygon will be marked in orange.
- Select a set of lines or contours to cut the polygon.
- When the polygon **P1** is initialized and the lines are marked, perform the menu command "**Cover -> Polygon operations -> Cut Polygon**" or press the corresponding toolbar button on the right.

3.4.5. Editing a DBF table

The attribute information contained in the dbf – table is linked to a specific object of "**Cover**" type and is available for viewing in the "**Prompt**" mode. The **GST** has no option to edit dbf data directly in the cover. To do this, the following steps should be taken:

- Extract dbf from the cover and place these data into a new object of "**Table**" type. Perform the menu command "**Cover → DBF → Extract DBF**".
- Edit the content of the table that appears in the hierarchy tree.
- Upload the edited table into the cover using the command "**Cover → DBF → Set DBF**". Specify the table with edited attributes in the pop-up dialog box.
- Once the edited attribute data are uploaded in the cover, delete the temporary table from the hierarchy tree.

The cover line is linked to a row in the dbf-table via the line identifier; its value can be viewed and set in the 'line parameters' dialog (A toolbar button). The ID value is compared with values contained in the first column of the dbf-table: when the identifier matches the n-th value in the first column, the n-th row is selected as the attributes for this given line.

If the cover does not contain a dbf-table, the attribute information can be assigned from any project table. For this purpose, you must set line identifiers and the appropriate ID values in the first column of this table; then perform the menu command "**Cover → DBF → Set DBF**".

3.5. Resource Zones

Resource zones cover is a numbered set of polygons used for calculation of volume and reserves tables.

3.5.1. Building and assembling of resource zones

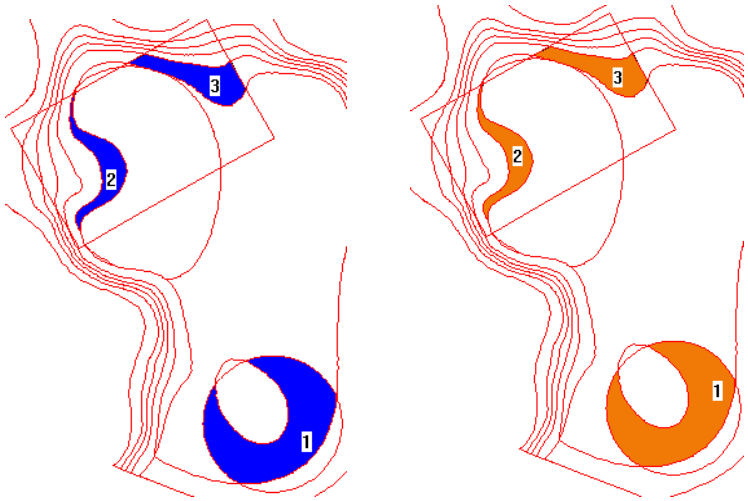
Add a new cover to a project tree, and specify a cover type as **"Resource Zones"**. Select one of the build methods listed above (e.g. build by references), specify parameters needed, add and activate references; and then execute the command "**Object → Start build**", or press the button on the toolbar.

The lines obtained from the references to other objects will be added to a cover edited layer. The user is building resource zones exactly based on the lines from the edited layer. Select needed lines and contours of the edited layer and translate them to the mode **"Zone mode**" by pressing the button on the toolbar or executing the command "**Mode → Zone mode**".

The program will perform required line preparation (handles crossings, etc.) when transferred to the indicated mode, and then the user can start area assembling.

Lines, transferred to the assembling mode, create a working cover, shown, by default, in blue color and by one unit wider than the width of edited layer lines. By working cover appearance user can check a quality of the available data (line closeness, boundary adjoining, etc.). It is appropriate to double check data quality and perform editing required (extend or close a line, etc.), if some lines visually forming a closed contour cannot be translated to the zone mode.

Assembling is executed by a simple left button click of a mouse at the spot of the anticipated area. The program assembles from of the cover lines selected a minimum contour containing the indicated point, and then fills it with the color of the added lines (color selection is in the cover parameters dialog box). The last area assembled is assigned a next number and highlighted by a contrast color.

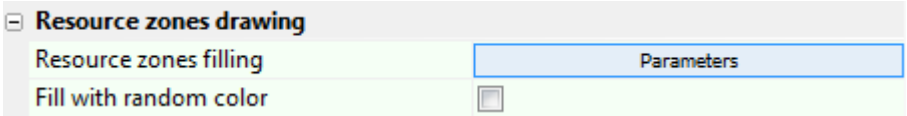


At the beginning, the zones are considered to be preassembled (Fig. above, left) and are stored in a temporary buffer. To add them to a set of the assembled zones, the user should execute the command "**Cover → Add object**" or press the button  on the toolbar (Fig. above, right).

One can "switch off" the already assembled zone by the second click of a mouse until the zones have been added to a set. In this case, the numbering of the zones will be adjusted automatically.

A name can be assigned to a zone which will be used afterwards in the reserves table. Just right click on the assembled zone and execute command "**Change zone name**". The assembled zone can be deleted from the set by a different command from the context menu **"Delete zone"**.

Display zones parameters are set in the cover properties dialog (see Fig. below). There is an option of the resource zone "random" color fill.



Note: To assemble a zone accurately, lines translated to the zone mode should form a closed contour. Sometimes, the end of one line can very closely approach the other; however in reality these lines do not cross each other. It can lead to problems at automated zones build. Thus, user should adjust the lines using editor functions and then translate them back to the resource zone build mode.

User can perform any operations with the edited cover layer (add lines out of a clip board, use all editor functions), since the zones are contained in a separate layer (set), while the cover lines are used just as material, or the basis, for their construction.

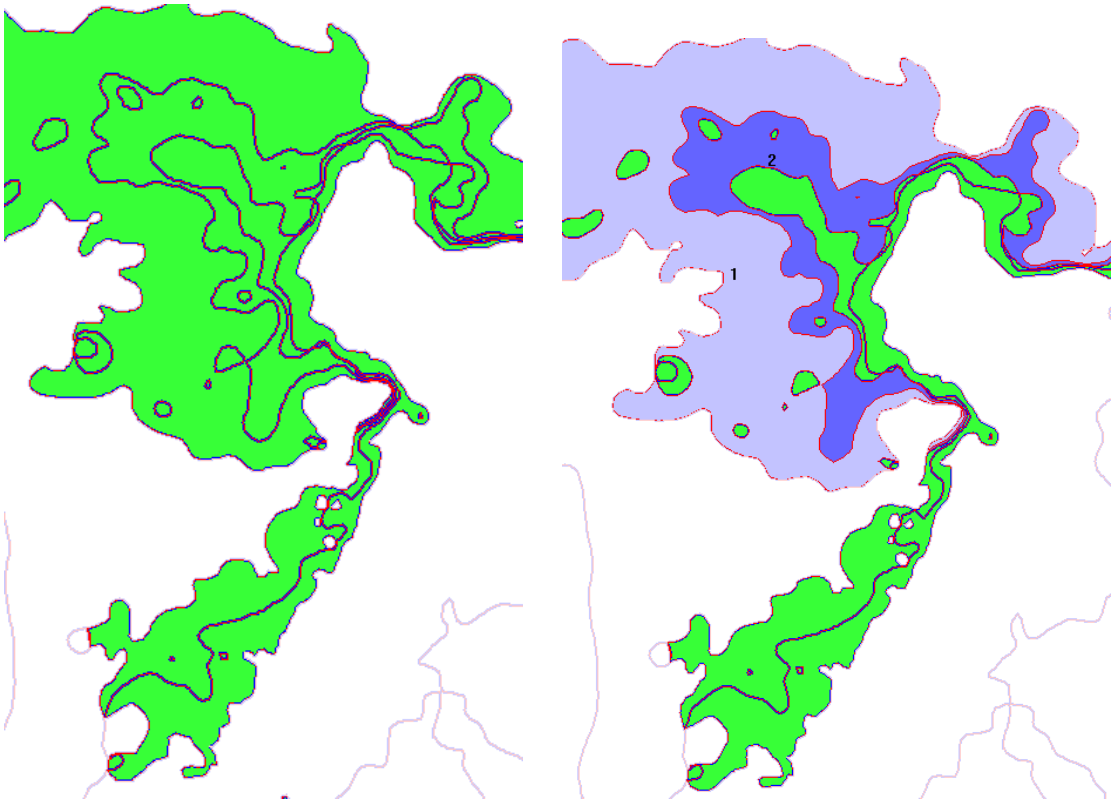
3.5.2. Assembling of resource zones using deposit model

Assembling of resource zones can be significantly simplified if it is performed using the **"Deposit model"** object. The following steps should be taken:

- Set "**Building by references**" build method for the object "Resource zones";
- Add reference to **"Deposit model"** and select option **"Load into edited layer"** in the reference properties dialog;
- Add reference to the covers containing boundaries of categories, license areas, etc., selecting also option **"Load into edited layer"**;
- Start build process.

Once building starts, **GST** transfers all lines required to perform assembling into the edited layer. It is necessary to select all lines and transfer them into "zone mode" by pressing button on the toolbar. The program would suggest to assemble zones in an automatic mode: the user can either accept it or reject it. **GST** will assemble simple contours in an automatic mode, as well as the zones with wells (from "Deposit model"). A picture similar to the one shown below should appear on the screen (left below Fig.).

The deposit polygon (i.e. where the zones assembling should be performed) is highlighted by a contrast color, while all the lines outside of it will be shown semitransparent. The left-click closes the "green" zone (see Fig. on the right hand); clicking outside the deposit polygon produces no effect.



To add the assembled zones in to a set execute command **"Cover → Add object"** or press button  on the toolbar.

3.5.3. Automatic assembling of resource zones

This method enables automatic assembling of resource zones, without navigation to the respective edit mode. This method implies that a certain "**Initial**" polygon is split into several polygons (i.e., sites) with a set of **unique** lines. "Unique" means that the lines and the initial polygon have no shared or overlapping fragments. It includes the following steps:

- Set building method "Automatic assembling of resource zones";
- Add reference to deposit polygon (or any other polygon), check options "**Load into editable layer**" and "**Initial**" in the 'reference properties' dialog box;
- Add a set of references to other "unique" boundaries: contacts, categories, etc. Check option "**Load into editable layer**" in the references;
- Start building.

The program will automatically split the initial polygon into a set of resource zones. If the user does not add references to supplementary lines, the zones will be created only on the basis of the initial polygon.

If a "**Deposit model**" object is already built and computed in the project, the automatic assembling goes as follows:

- Set building method "Automatic assembling of resource zones";
- Add reference to "**Deposit model**" and check option "**Load into editable layer**" in the 'reference properties' dialog box;
- Add a set of references to other "unique" boundaries: contacts, categories, etc. Check option "**Load into editable layer**" in the references;
- Start building.

The program automatically retrieves the deposit polygon and contact lines from the "**Deposit model**" and splits it into resource zones under supplementary lines assumptions.

3.5.4. Split resource zones by development blocks

This method is designed to partially automate zones assembling process and reduce the amount of "manual" work.

By **blocks** in **GST** is meant **Cover** of **"Simple"** type consisting of closed lines forming the blocks boundaries. Block boundaries can be unilateral only, the blocks should not overlap (however adjacency is allowed). A name can be assigned to each block (line) (in a dialog **"Line attributes"**) which will be later used for filling in the tables with reserves.

The following steps should be taken to use this method:

- Add a new **Cover** with the **"Resource zones"** type to the project and set "**Split of resource zones by blocks**" build method for it;
- Add reference to the resource zones, which were created without accounting blocks (option **"Load into edited layer"** for reference should be on);
- Add references to one or several covers containing blocks (options **"Load into edited layer"** and **"Development blocks"** should be turned on in a dialog box);
- Start building.

In the course of building the program will automatically "slice" previously assembled zones for each block leaving only those that fall within their zone.

3.5.5. Merging of resource zones

This method allows to combine in one object the data of several covers of resource zone type. To perform building both the object and the reference should have the same type – Resource zones.

3.5.6. Operations with resource zones

The following operations can be performed with a set of assembled resource zones:

- Deleting of assembled zones can be performed under a corresponding editing mode (command (**"Mode → Zones delete mode"**). After accessing this mode the user can delete certain zones from the set clicking the left button of a mouse. Command **"Cover → Zones operations → Delete all zones"** deletes all zones from the set.
- Zones can be moved to the edited layer by command **"Cover → Zones operations → Move zones into edited layer".** The zones are also deleted from the set and their contours become accessible for editing.
- To expedite the zones assembling process it is possible to add all simple closed lines (those which do not contain any other lines within them) to the assembled zones by one single command **"Cover → Zones operations → Add simple contours to zones"**.

3.6. Deposit model

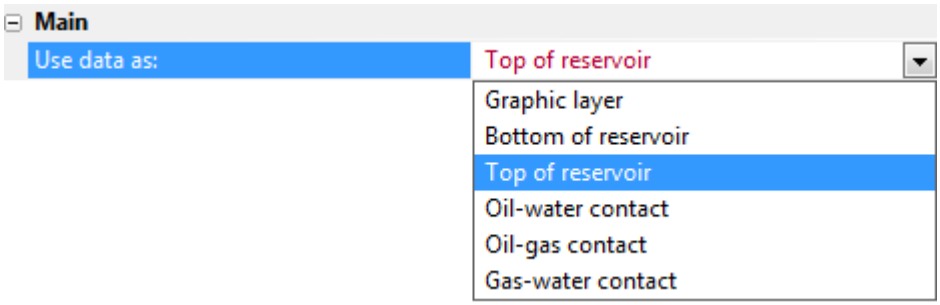
Deposit model is a special type of cover which does not have edited layers and contains polygons of saturated zones, as well as contact lines and lithological boundaries. The objects of this type are used to solve the tasks pertaining to estimation of hydrocarbon reserves.

3.6.1. Building deposit model using reservoir geometry and well data

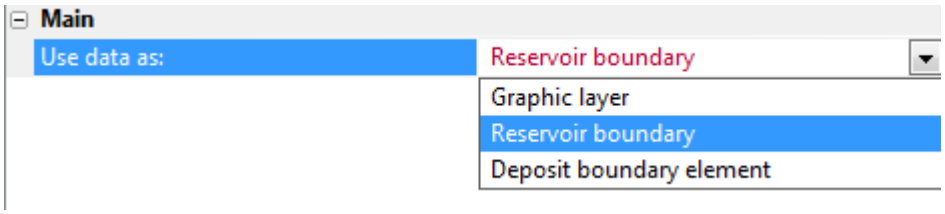
This section deals with building of a deposit model using reservoir top and bottom grids, lithological and tectonic boundaries and well data. Add object **"Deposit model"** to the project, set **"Build deposit boundaries"** method and add references to the data required to perform computations:

- Reference to reservoir top and bottom grids;
- Reference to the table (tables) with well data, containing information about saturation type;
- References to the contact data (it can be both numbers shown in the tables describing horizontal contacts, and grids for tilted contacts);
- Additionally – references to the covers containing data on lithological boundaries, screening faults, etc.

In the grid reference properties dialog the user should select from the list (see below) in what form the data would be used during model building. Note, that the object cannot contain more than one reference to the top and bottom grids or contact surface. It makes sense to set the surface by grids only in case of tilted contacts, otherwise they will be set by numbers in the tables.

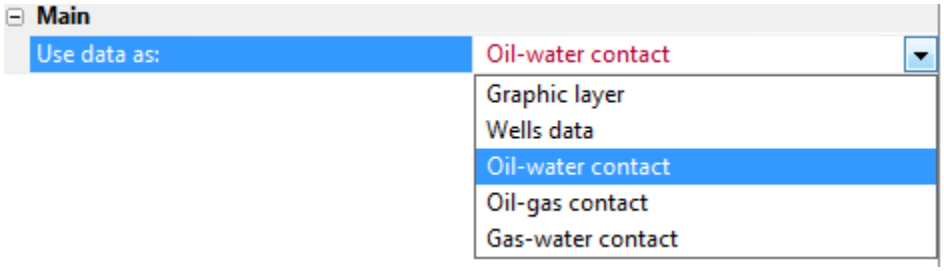


In the dialog box of cover reference properties user should select from the items shown in the Fig. below.

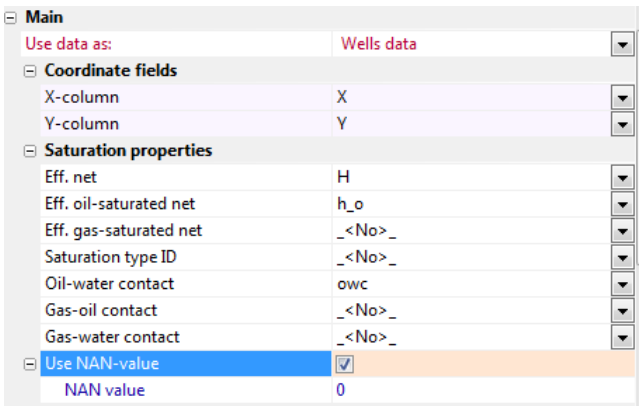


One can set **"Reservoir boundary"** in case the reference cover is a boundary of a reservoir extension; however if it is any other kind of boundary screening the accumulation the **"Deposit boundary element"** type should be set. Usually the reservoir polygon is not required to accomplish tasks pertaining to estimation of reserves (it is sufficient to have a deposit polygon): in this case the **"Deposit boundary element"** type can be set for all screening boundaries.

Reference to a table can be used as a graphic layer, contact or well data. If the contact is flat and has only one value for the entire deposit, its depth should be set in a separate table, consisting of a single column and a single row.



The following options should be selected by user under well data reference:



- Coordinate fields;
- Column with effective net pays;
- Column with saturation data from the wells;
- Contact depths – when necessary;
- NAN-value (if needed);

There are two ways to determine saturation type in the wells: either by setting the tables with effective oil or gas net pays, or by means of a column containing a saturation type identifier. In the first case, for instance, to define the well as "oil – gas" it is necessary that the sum of gas saturated thickness and oil saturated thickness was exactly (to the last decimal) equal to the net pay. In the second case, the program will be considering only the value of the saturation type identifier. If in the references parameters are set both - thickness and identifier, the priority will be given to the identifier column.

Identifier of saturation type may have the following values:

- **0** – wells tested water;
- **1** – pure oil;
- **2** – oil with water;
- **3** – gas;
- **4** – gas and oil;
- **5** – gas, oil, and water;
- **6** – gas and water.

It is possible to set in well table's reference parameters the columns containing the depths of horizontal contacts. It allows to build in one object **"Deposit model"** the boundaries of several isolated deposits, each having its own contact. It is also possible to set a contact of one type by a separate reference, and another type via the column in the well table. For instance, in case of a gas-oil deposit with a single-connected boundary containing several gas caps (each having its own GOC), OWC can be set as a reference to a separate table or grid, and take GOC from the corresponding column of the well table.

Building process consists of contact boundaries tracing and automatic polygons assembling using all this data. Smoothness and tracing detailing is insured by a single build parameter **"Tracing scale"** which is set in an object building parameters dialog box (see below).

This parameter ensures the visual smoothness of the contour lines when viewed in the preset scale. In order to use it correctly, it is necessary that the object measurement units are the same as the measurement units in the **project settings**.

One can start building process after all required parameters are set. Building is performed in fully automatic mode, but it can be divided into several stages.

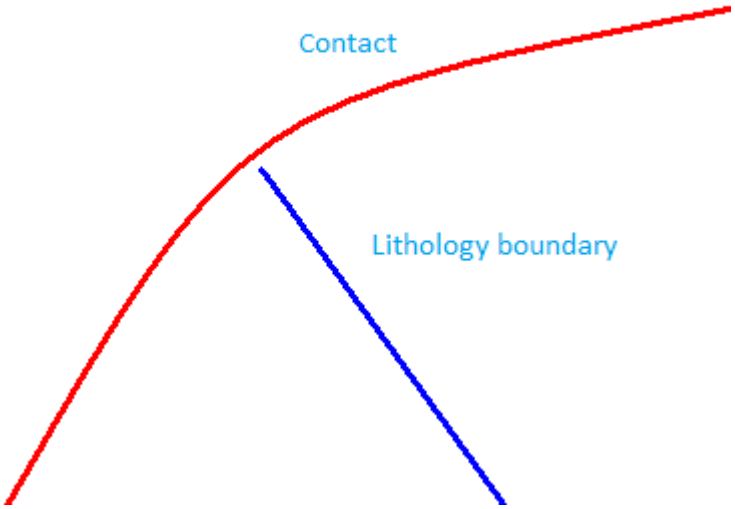
Building of reservoir polygon. If a reference object, marked as **"Reservoir boundary"** is a polygon, it means that this particular polygon is accepted by the program as a reservoir polygon without any additional computations. If reservoir boundary is set by a simple cover, then **GST** first traces a cross line of reservoir top and bottom, and after that builds a reservoir polygon around wells with positive net pays based on that line and the reservoir boundary elements set via references. All

closed lines without wells are disregarded. A joint top and bottom grids rectangle is taken as a reservoir polygon, if the top and bottom cross line and the reservoir boundary elements do not form a closed structure.

Building deposit, oil and gas polygons. Contact lines tracing, using top and bottom grids; assembling polygons using contact lines and reference covers marked as **"Deposit boundary element"**. Closed structures formed by the contact lines which do not have wells tested hydrocarbons are excluded.

Building other polygons is performed within the oil and gas zones on the basis of all obtained contact lines and lithological boundaries.

If for some reason a polygon cannot be built, the output window will inform about a corresponding error and the fragments of the boundaries will be moved to a special layer **"Errors during assembling"**, which can be visualized by user.



The most typical error during polygon assembling can be the case described in the Fig above, when, for instance, lithological boundary screening the deposit does not extend all the way to the contact line, and they do not form a closed structure. The way to fix it is to edit one of the lines.

As the result of all computations a layer of polygons is formed which determines all zones of saturation: deposit polygon, zones with one, two or three phases, "pure" hydrocarbons zones, etc. All lithological boundaries of accumulation and contact lines are preserved in a separate layer.

Recommendations and notes. One should keep in mind the following when making computations of the **"Deposit model"** using references:

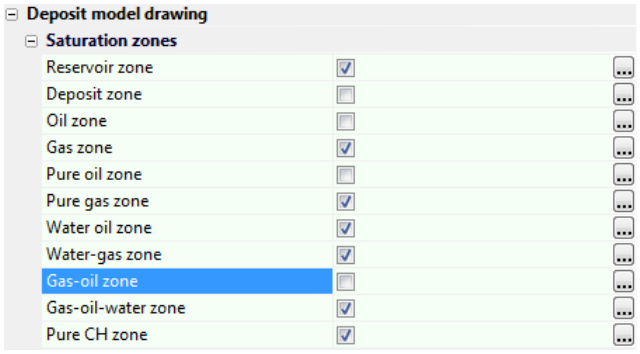
- To set correctly a tracing scale it is necessary to have the coordinates measurement units of original input data be the same as the units installed in the project settings;
- Usually there is no need to assembly a "reservoir polygon" for the purpose of reserves estimation, therefor all lithological boundaries in the references can be marked as **"Deposit boundaries element"**;
- If there is any boundary (e.g. a fault) splits a deposit into two or more isolated deposits, then this boundary should be set as a thin fissure; the line can be converted into a fissure (fault) using lines editing operations.

3.6.2. Import and export of deposit model

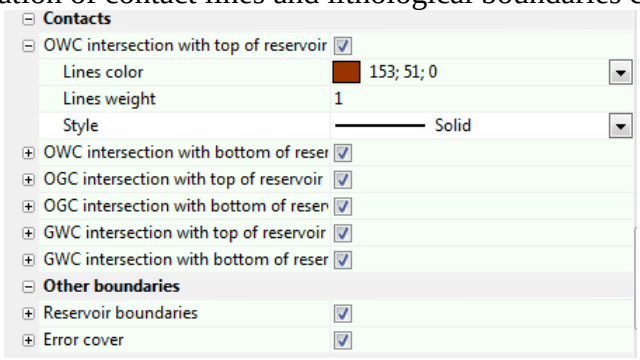
“Deposit model” being an “internal” object of **GST** can be imported directly from a different **GST** project (it is necessary to set **“Link to other project”** build method) or loaded from the ***.mgs** internal binary file. The object of this type can be exported only in a format of internal representation of the objects ***.mgs**.

3.6.3. Representation of deposit model

All options of reflection of deposit model layers are set in a corresponding of the main page of object properties dialog. (see Fig. below). User can set a color and fill type for each polygon (right-hand buttons).



Parameters of representation of contact lines and lithological boundaries can be set below.



If the object is not built the control elements of representation of all possible saturation zones and lines will be shown in a dialog page. If the object is built, only those polygons and contacts which are present in deposit will be accessible.

3.6.4. Using deposit model to build other objects

Deposit model contains two work layers: polygon layer (all possible zones of saturation) and boundary layer (lines of contacts and lithological boundaries). One or several elements of the said layers can be moved to the cover edited layer during building of cover by references (see section 3.1.2), and used for further computations.

Building of resource zones on the basis of “Deposit model” (building of resource zones by references with adding reference to deposit model) makes their assembling much easier.

4. Grid object

4.1. Basic grid-build methods

User should specify one of the following methods:

- Load grid from file;
- Create with Map-Builder;
- Create with Calculator;
- Recalculation with tabulated relation;
- Composite grid with inserts;
- Insert;
- Grid Generator;

Methods - Map-builder, Calculator, Composite grid with inserts and **Insert** should be given a special attention that is why they will be considered in separate chapters.

4.1.1. Recalculation with tabulated relation

This procedure envisages recalculation of one grid to the other based on an arbitrary, tabular-form specified relation. The table should have at least two columns: argument value (with which the values in the initial grid will be compared) and relevant function values (which will correspond to the final grid). To apply this build method, it is necessary:

- To specify a build method as "Recalculation with tabulated relation";
- To add a reference to the grid being recalculated (enable the option "**Use to build grid**" in a reference-property dialog);
- To add a reference to the table that contains the relation assumed; Users must enable the option "**Use to build grid**" and specify the fields with an argument and function, set a nan-value, etc.

At grid calculation, the values between tabular data are interpolated linearly (accounting for a nan-value); linear extrapolation takes place in the case when grid values overstep a dependence range.

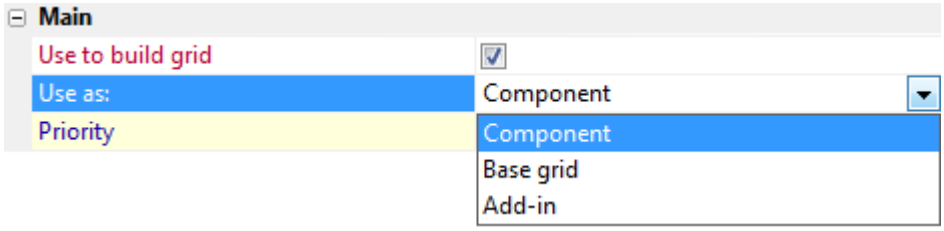
4.1.2. Grid Generator

When using this grid building method the **Map-builder** (which will be considered in a separate chapter) dialogue window will have a tab **Generator** which can be used to build a grid of a special type without using the objects of another type (e.g. references). After setting up the grid's rectangle and step user can generate one of the two following grids:

- Grid is a plane (in the particular case, grid is a constant, that is a horizontal plane). The overview of the inclined plane in the Generator is set by the following equation: **Z = a*(x-dx) + b*(y-dy) + c**, in which the user needs to set the values of 5 parameters.
- A random grid, which contains random values (from 0 till 1) in the grid note.
- These grids can be useful during the solving of different mathematical physics tasks using the features of the **GST Map-builder** and **Calculator**.

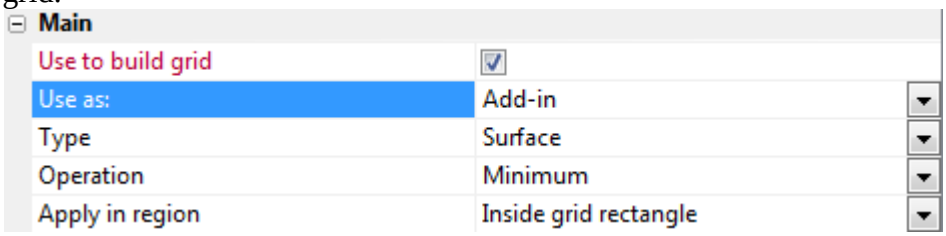
4.1.3. Merge grids

This method allows you to build a single grid from a set of grids and (if necessary) polygons. All building parameters are specified in the references to the grid and polygon. A reference properties dialog is as follows:



Checking the option "**Use to build grid**", you select the desired item in the list "Use as".

Component. This option must be selected if the resulting grid is building "piece by piece" from the reference grids. In case of several grids overlay, one is chosen with higher "**Priority**" parameter. The parameter "**Priority**" can be set as a float number as well. Rectangle of the resulting grid is defined by reference to the object "**Rectangle**". If all reference grids have definition areas, the grid merging takes place within the polygon definition (as in previous versions).
Base grid. In this case, the resulting grid is built by adding other grids to "base grid". Rectangle of the resulting grid is defined by basic grid area or a reference to the object "Rectangle". Reference to the basic grid should be only one; the other references should be of "**Add-in**" type.
Add-in. The additive to the basic grid may have "**Surface**" or "**Thickness**" type. If you choose the first option, value corresponding to the minimum or maximum of two surfaces is selected in the overlap area of the base grid and additives. If you select "**Thickness**", this thickness is added to or subtracted from the basic grid.

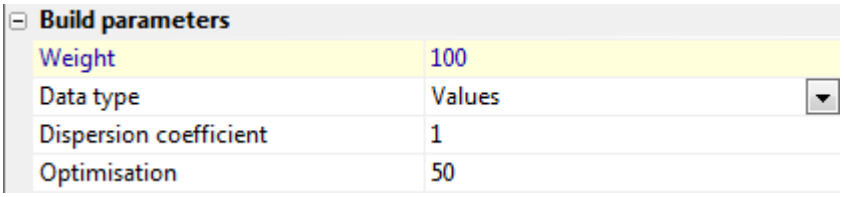


In this case, the negative values of the grid thicknesses are ignored. The additive can be applied to the entire grid rectangle or within a given polygon.

4.1.4. Grid of stability

This method allows you to assess the reliability of the previously built grid using the "Map-Builder". This building is based on the calculation and mapping of stability coefficient - parameter characterizing the effect of adding additional test data at any point in the mapping field on results of the grid building.

It was theoretically proved that the stability factor - the ratio of the approximation error of the forecast error $\nu^{+} = \Delta z_a / \Delta z_p$ - is independent of properties of the mapping indicator set in the test point. Here Δz_a and Δz_p is the difference between the exact value of the parameter and values of approximating map at this point (built based on test data) and the forecast map (built without test data). *Stability factor value* varies from 0 to 1.



The user must set the method "Stability map", add a link to the grid created with the **Grid-Builder** and activate option in the link "**Use in the grid calculation**." Four values are set in "Building parameters" dialog:

Weight. It should be considered in test data during the calculations. Typically, this value is assumed to be equal to the weighting factor of data, on which target grid is built.

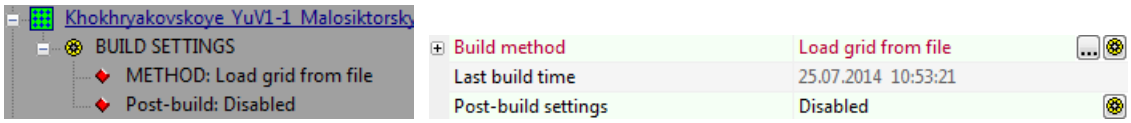
Data type, characterizing type of the considered test data (values of the mapped field, its derivatives). As a rule, calculation of the values of stability of the mapped indicator is carried out.

Dispersion coefficient of the original grid, taking positive integer values. In case of equality of this parameter to 1, the calculations are carried out for each node point of the target grid. If the parameter is set equal to 2, the calculation is carried out for each second node of the grid, etc. In this case, the calculation speed is increased by reducing the precision of the resulting stability map.

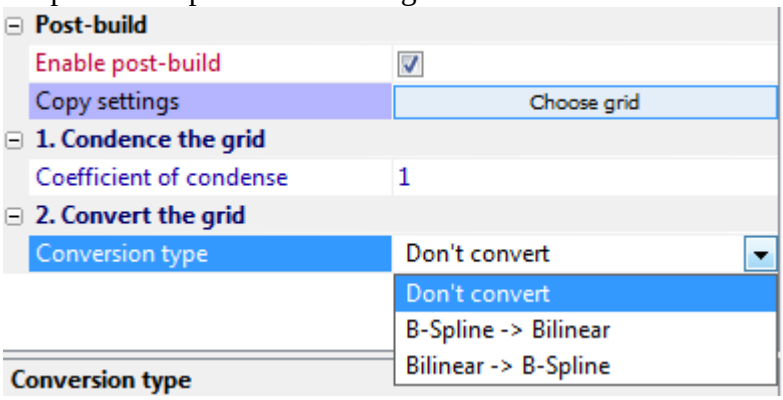
Optimization parameter is used to implement the parallel calculation capabilities and determines the number of points for which the stability factor calculations should be carried out at the same time. The optimum value of this parameter depends essentially on the dimensions of the target grid and on the computer's RAM. The minimum parameter value is 1.

4.1.5. Grid post-build

You can enable or disable post-building properties dialog of the grid or in the hierarchy tree, as shown in the figure below.



Double-click on the corresponding icon in the hierarchy tree or press the button (yellow star) in the properties dialog to call post-build parameters dialog.



Only two operations are offered: condensing of grid and grid conversion. By conversion we mean changing the method of calculating between grid points from bicubic spline to bilinear and vice versa.

4.2. Grid display options

4.2.1. Grid tracing

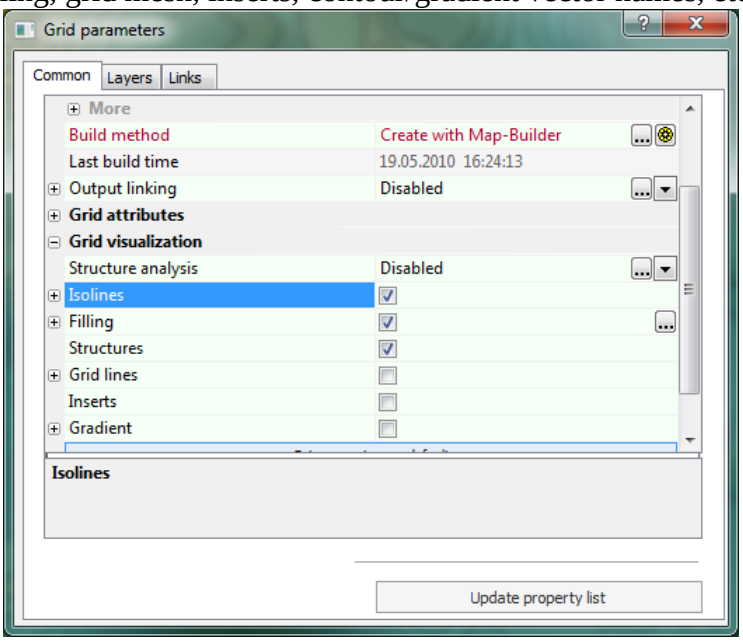
Tracing means a process of transferring a networked surface model to a set of contours. To perform tracing (after grid loading\or building), press button , or execute the command "**Tracing**" from the "**Grid**" menu. In the popped up dialog specify the necessary parameters: tracing limits, interval (the interval between the tracing levels), grid graduation ratio, which defines the contour smoothness.

By default, the program suggests automatically selected parameters of tracing. At the grid-property dialog's page "**Tracing of a grid**", tab "**Common"** the user can set the tracing parameters.

In order to accelerate the grid display and to economize the memory starting from 6.1 version the GST automatically optimizes the contours after tracing (deletes extra points without visual detail loss). By default this option is activated, the user can deactivate it at the grid-property dialog's page "**Tracing of a grid**", tab **"General"**.

4.2.2. Display Parameters

In the "**Grid visualization**" the user can specify all grid representation options (enable/disable isolines drawing, color filling, grid mesh, inserts, contour/gradient vector names, etc.).



A control-item group "**Gradient**" is designed to display surface-gradient vectors. A vector direction in each point is marked by an arrow, which length characterizes a gradient value. The "**Invert direction**" option helps to change arrow direction to the opposite one, while "**graduation**" helps to increase or decrease the arrow density in relation to a grid.

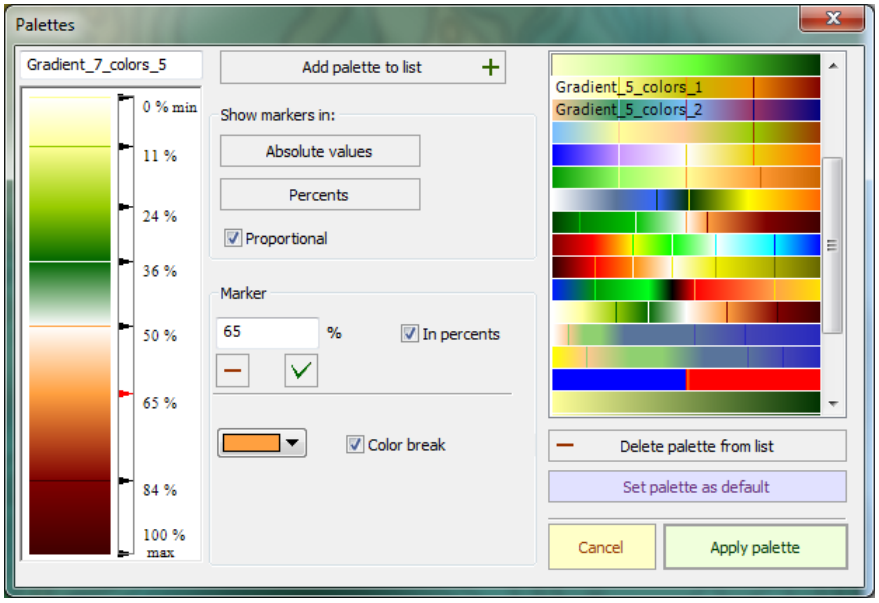
Using control item group "**Isolines**" user can set isolines specifications: set their color, thickness, style and font. The option "**Highlight isolines"** helps to mark the isolines following in a certain interval with a thicker line.

Display grid inside the polygon. In the **GST**, grid always has a rectangular area; in each point of this area a mapping-parameter value can be calculated. If a mapping property/feature does not apply to the entire grid rectangular, the **GST** uses a notion "**Grid definition area (polygon)**".

A definition area is a **Polygon** "**Cover**" that should be added to a grid as a reference. The option "**Use as grid region**" is set in the reference-property dialog box. With this option enabled, grid visualization takes place inside the selected polygon.

4.2.3. Grid Levels Filling

A new scheme of level filling is implied starting from GST 5.0. It helps to build a necessary number of attribute value levels and to mark them with a certain color. The filling options are set in the **Grid visualization** category (tab "**Common**", control-item **Filling** or  toolbar button).



There is a number of preset palettes in the list on the right which are kept in the service directory «**.../Graphics/Palettes/**», in the **GST** installation folder. By double clicking at the element in the list the user can choose a necessary palette and then it will appear in the left part of the dialog (palette control). Clicking at the **Apply palette** button the user sets the pallet on the left to fill the grid levels.

The chosen palette can be modified. Palette modification consists of adding/deleting markers and setting colors and values for them. New level is added to the palette by clicking at the narrow vertical line located near the palette. A new marker shows up (or the old one is marked). It is highlighted by a red triangle. A control-item group of the chosen marker is in the **Marker** group. The marker values can be set either in the absolute values of the grid levels or in percent to the interval of the grid values. This feature is set in the **"Percent"** option. The value can be changed by pressing  button. "**Color break"** option helps to change colors in a leap. A chosen marker can be deleted by pressing a "-" button.

A corresponding color can be modified when the marker is selected. The marker can be vertically moved inside the palette control with a mouse. Also, once a marker is selected with a mouse, the user can set its accurate value and move it to a definite place pressing the  button.

Values in percent or in absolute values for all the markers can be set with relevant buttons in the top part of the dialog. After the modification user can add it to the preset list (directory) palettes under a new or an old name. The user can remove the palette from the service directory with a button "**Delete palette from the list**".

Attention! The present palette can be modified, but the changes are not saved until the palette is added to the list. Which means that if the user modifies the palette and then switches to another one from the list, all the changes will be lost.

4.3. Structure analysis

4.3.1. General issues

Structure analysis assumes the presence of some surface (not specifically geologic), which looks like a cellular model (grid) inside of its (not specifically rectangular) definition area. It includes:

- Positive and negative structures search (mark up);
- Structure visualization with contours and/or filling;
- Structure parameters display when marked by the cursor in the grid window;
- Filter apply for each parameter of the structure;
- Structure parameters bar chart build up and statistics saving in the text file;
- Structure contours and attributes saving in the **shape**-file;
- Structure attributes saving in the table with preliminary sorting.

Inside, the positive structure is counts as a trap, the negative – as downfold. The surface values increase from top to bottom. Structure search is performed inside of the grid definition area, which can be any set of simply connected regions. Further on in this chapter let consider the grid boundary as the boundary of the definition area.

Structure search is based on grid horizontal slicing with a certain interval. Then GST automatically traces the levels, transferring a grid data to a set of closed and non-closed lines.

Closed lines form the boundary of so called "closed" structures which characteristics can be identified with accuracy. Closed structures cover only a part of the grid definition area.

Non-closed contours (those which exceed the grid boundary in two points) become closed at the boundary, forming closed lines which are used for so called "contingent" (non-closed) structures build up. These objects are part of real structures which true characteristics are unknown. Contingent structures parameters give us information about real structures only within the grid definition area. Non-closed structures include close structures, but not vice versa.

The final structure sign (positive or negative) is defined by the grid contour's integral sign. The absolute integral value characterizes the structure volume. The area of a figure situated inside of the border corresponds with the structure area.

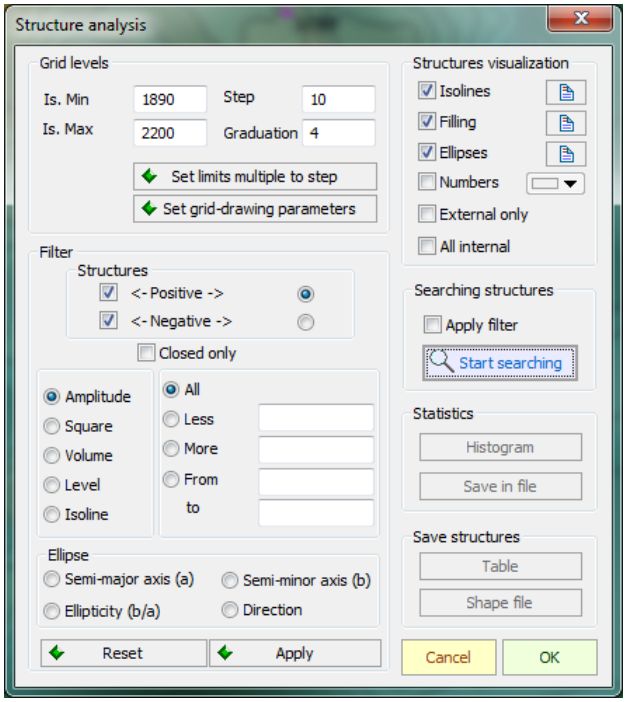
There are two possible variants. The first one assumes that each closed line forms a structure (all so called "internal structures" are found). For this case the user needs to turn on the flag "**Apply filter**" and option "**All internal**" before the search. In the second case internal structures should be ignored if they are included in another one, that doesn't consist of any other (but this one) structure.

The structures that don't include other structures are called elementary and have a class 1. The class of a composed (complex) structure is defined as the highest class of the included structures plus 1. The maximal difference between structure boundary level and grid value inside the structure contour defines structure amplitude. If the structure contour does not contain any grid nodes (small structure), the amplitude is calculated as a volume to area ratio. The height of the compound structure is found as a sum between the heights of the first (according to the sections) included elementary structure and the difference between the section heights of these structures.

The structures calculation can be automatically started after the object build up. In order to do this the **Structure analysis** option should be enabled on in the **Common** tab of the grid properties dialog.

4.3.2. Structure analysis settings

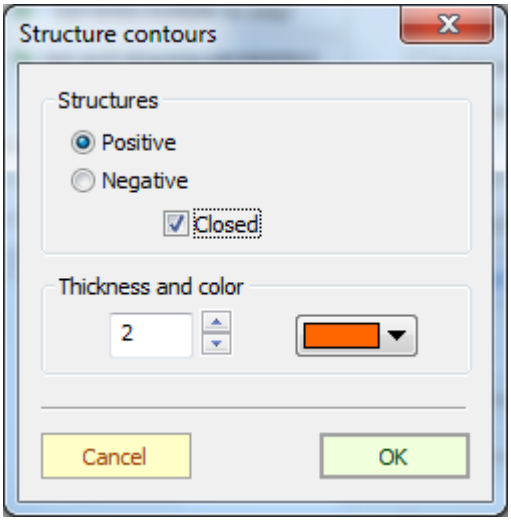
The parameters for the structure analysis should be set by pressing "…" button in the "**Structures analysis**" control (grid properties dialog, tab **Common**) or pressing the button in the toolbar. The following dialog will pop up:

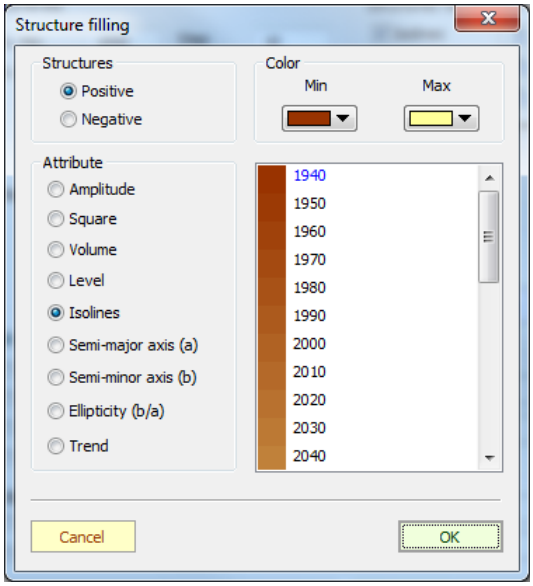


The grid tracing parameters are indicated in the "**Grid levels**" group. By pressing «**OK**» user can save the settings that will be used for automatic structure analysis (if the option "**Structure analysis**" is enabled). To perform structure analysis for the ready user should press "**Start searching**" button.

The selected structures are characterized by the following set of parameters: amplitude, square, volume, level (class), isoline value and the ellipsis parameters. The structure shape can be approximated by the equivalent ellipsis. 4 parameters of the ellipsis are used in the analysis: major and minor semi-axis, ellipticity (axis ratio), structure direction (major axis direction – deviation from the vertical line, range between -90 to +90 degrees). The structure search, visualization, filter, calculation and results saving (statistics, attributes) are set in accordance with 9 characters.

In the "**Structure visualization**" group the user can manage the display mode. The contour parameters setting, filling and structure ellipsis dialogs are also there. User can modify thickness, color of the boundary line (closed are modified separately) in the "**Structure contours**" dialog. The color gradations can be modified in the "**Structure filling**" dialog, the ellipsis drawing parameters – in the "**Structure ellipsis**". The settings are made separately for positive and negative structures. This dialog pops up when the user presses button near the proper checkboxes. The color can be selected and the structure numbers can be displayed on the map after enabling the "**Numbers**" option.





The filter helps to examine only those structures that satisfy the selected limits for each attribute. Filter also may be applied to visualization, statistics data processing, statistics and attributes saving. Moreover, the structure analysis may be applied with the account of filter. When the option "**External only**" is enabled, only structures not included in other structures are being analyzed.

Structure analysis without filtering has the same interface as the dialog after pressing "**Reset**" button. All the closed and non-closed, positive and negative structures are found without any limits.

When pressing the "**Histogram**" button an additional window "**Structure statistics**" pops up, where the interval structure distribution for the attribute chosen in the filter at the present moment. The statistics calculation results (for all attributes at once) can be saved in a text file. The structures themselves and the attribute information can be saved in a table or in a shape-file. Before saving the structure attributes in a table the sorting by one of the attributes takes place. If the option "**Structures content**" is selected, a column with structure numbers will be added to the table.

Specific structure attributes in the grid visualizing window can be seen when the user points a cursor inside of the selected structure and clicks with a right mouse button at the "**Structure Parameters**" tab in the pop up menu.

5. Creating Grid with Map-Builder

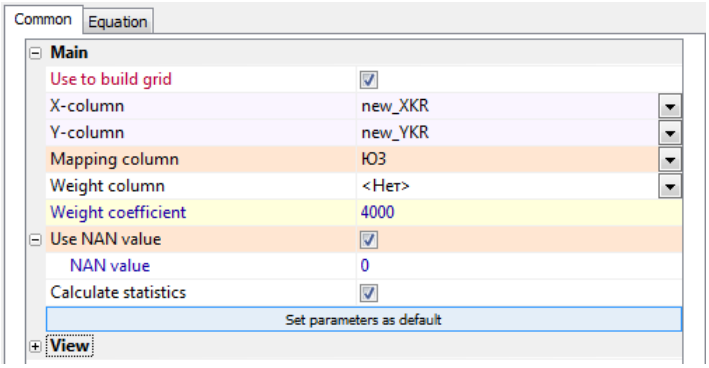
To build a grid using the **Map-Builder,** do the following:

- Specify build method as "**Create with Map-Builder**" in the grid-object-parameter dialog;
- Add object-references to grid, needed for its creation;
- Specify parameters of each reference to be used to create grid ;
- Specify parameters in the "**Grid-Builder**" dialog - mapping rectangle, grid step, weights on stabilizers, etc.

5.1. References Initialization

5.1.1. Reference to Table

To apply table data to mapping, enable at reference property-dialog page "**Common"** the option "**Use to build grid**". If this option is disabled, the data will be used only as a graphical layer. In this case the dialog makes only X- and Y-coordinate windows accessible. In the former case, specify a column name corresponding to mapping parameter, and, if necessary, column that contains local points-weights**.** Specify a general weight coefficient for point data and, if necessary, set a NAN-value. In the category **«View»** user can select point drawing parameters: color, size, font of signs, etc.



The option «**Calculate statistics**» is enabled by default – grid has been built, it reflects points in excess of the appointed value and computes other statistical characteristics such as average deviation, dispersion, etc. To save memory, the option can be disabled.

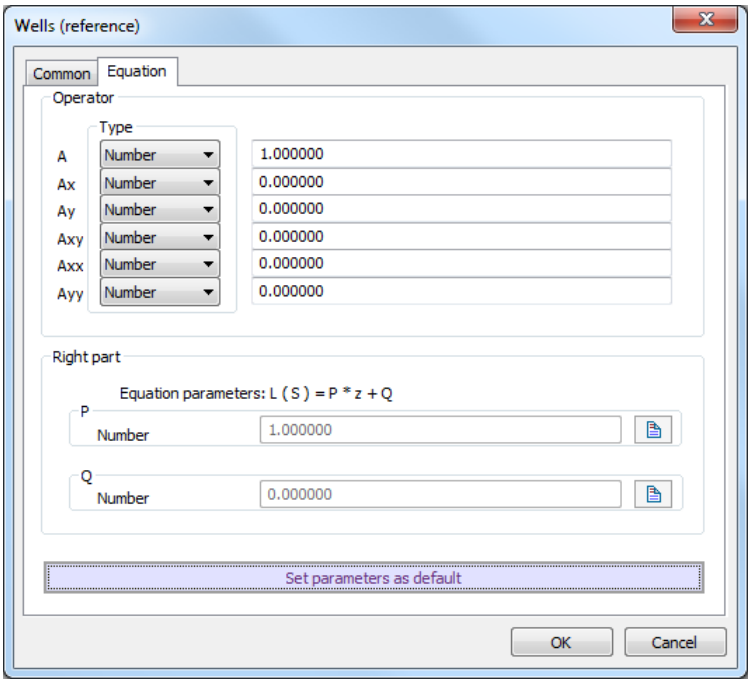
If reference is used to build a grid, the tab «**Equation**» becomes accessible. In this tab user specifies additional conditions – how to use point data in calculations.

The mapping surface should satisfy certain conditions at given data points. These conditions can be applied for values of a desirable function, as well as on its first, second and mixed derivatives. They are written as follows:

$$A \cdot F + A_x \frac{\partial F}{\partial x} + A_y \frac{\partial F}{\partial y} + A_{xy} \frac{\partial^2 F}{\partial x \partial y} + A_{xx} \frac{\partial^2 F}{\partial x^2} + A_{yy} \frac{\partial^2 F}{\partial y^2} = P \cdot Z + Q.$$

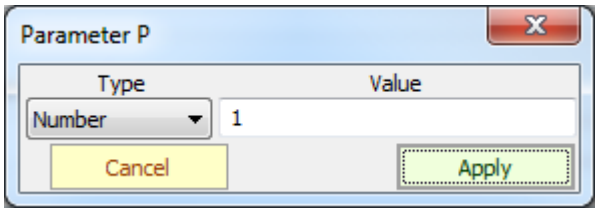
Where F=F(x, y) – a desirable surface, Z=Z(x, y) – in-point observations (values contained in a mapping column), $A, A_x, A_y \ldots$, as well as P, Q - coefficients that in a general case depend on

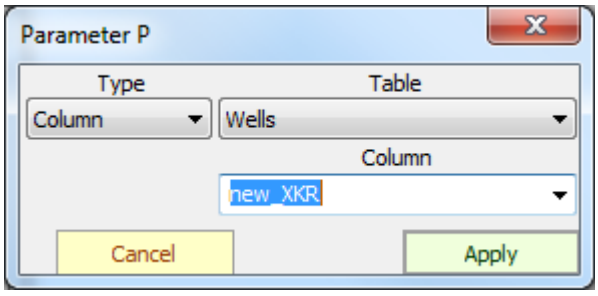
coordinates. The left expression is called a local operator (**L-operator**), while the right one is a local parameter (**L - parameter)**.



In a general case the operator's coefficients depend on coordinates. For that, it is necessary to specify their type as "**Grid**" or "**Column**". In the former case to the right a drop-down grid-list box appears (grids from references). A value calculated from a grid or taken from the table is used as a coefficient. In the latter case, a coefficient is a value from the indicated column of the table under review.

Coefficients *P, Q* are an amplitude factor and vertical offset, respectively. If, for example, Q=100 m, then the resultant map will have values 100 m deeper than in observation points. To change a parameter type and value press the button  .

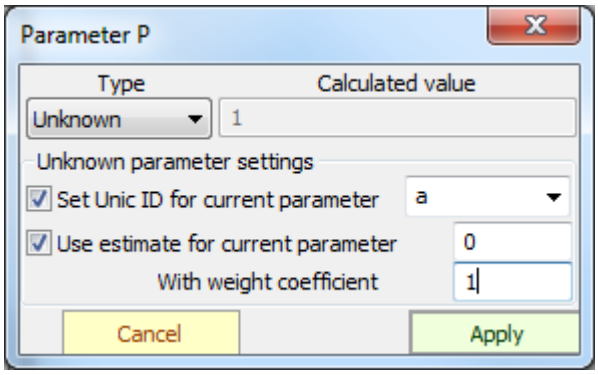




If a parameter is of a "**Number**" type, just change its value in the input field. In the "**Column**"-type case, it is necessary to select a name of column that contains values of current parameter.

Coefficients *P* and *Q* may be established as a certain grid. For this, the grid should be among the significant references (the option «**Use to build grid**» is on) while all connected weights may be established at zero.

In some cases (while using conflicting seismic data) a desirable surface should go with some shift from picket data, when this shift is unknown. To make it, specify the type "**Unknown**".

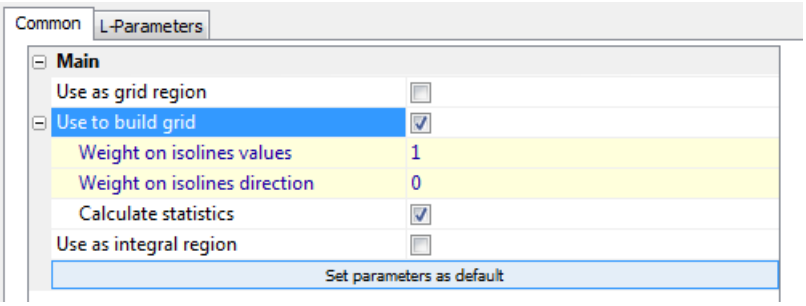


Here, a shift value (or amplitude-factor value) is calculated, while solving a mapping task proceeding from other established conditions. Users can set an approximate value with a certain weight coefficient. By setting a unique ID for current grid, it is possible to combine several data tables (with identical identifiers) in such a way that their desirable values will be the same.

5.1.2. Reference to Cover

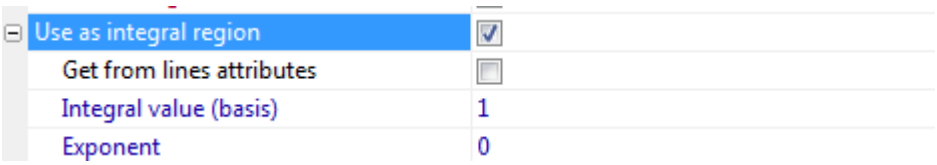
Cover can be used:

- As an insignificant reference (graphic layer only);
- As a source data for a re-gridding task (restoring grid from contours);
- As a tectonic fault (only the line geometry is important here);
- As a grid-definition area (should be a polygon cover);
- As integration region (should be a polygon cover).



For a re-gridding task the user specifies weight coefficients for lines attributes and lines directions. If a weight on the lines directions is larger than zero, then additional conditions are applied to surface, which relate to a value of its first derivatives along contour's direction (zeroes of first derivatives). Enabling option "**Use to build grid**" opens a reference-property dialog page «**L-Parameters**» that is identical to the one considered in the previous paragraph.

In case of a «**Polygon cover**», property dialog shows the option «**Use as grid region**», as well as the option «**Use as integral region**». The latter allows users to establish directly the value of the surface integral by polygon. For instance, it allows users to build a thicknesses map with a set rock volume within a definite polygon. The map will fully correspond to the point data and other established conditions. The integral value is established in two edit fields: in the former, («**Integral value (basis)**») the base of a number is established, in the latter, the **exponent**.



Example: **6.5** x 10^{8}. In the former area, the number 6.5 is established, in the latter, 8. Apply option "**Get from lines attributes**" to initialize integral value with polygon's attribute value.

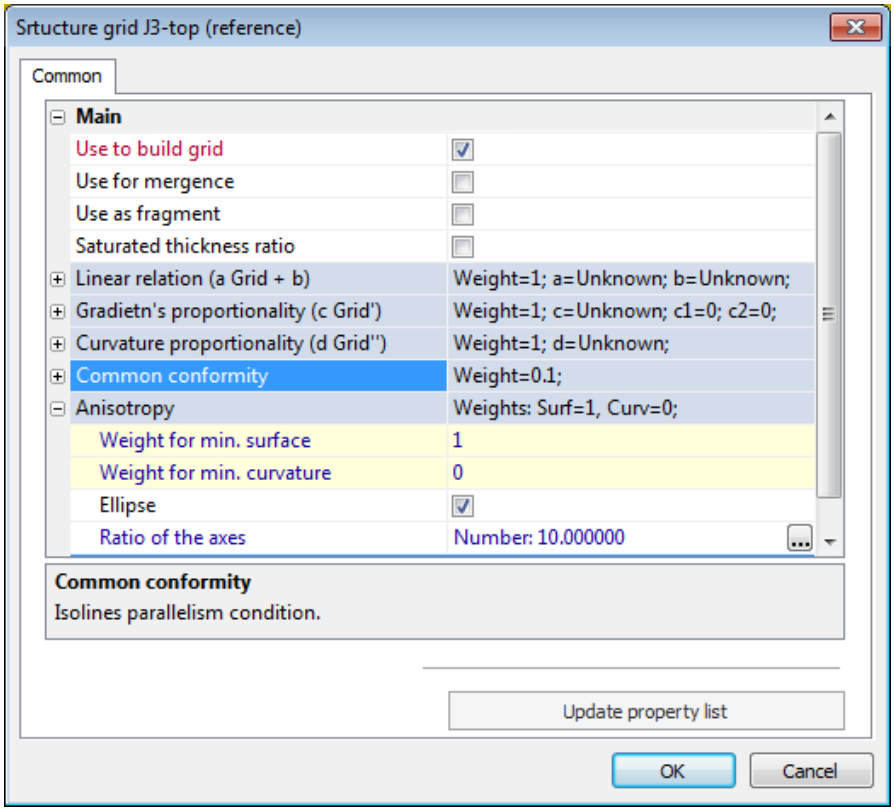
A grid may contain references to several polygons simultaneously, each of which will have its own integral value. Polygons may be multi-connected or overlapping.

The option "**Use as significant grid region**" makes significant the reference to the polygon grid. This means that the polygon is calculated automatically in the chain build, prior to grid calculation. This option is recommended in building grids of effective saturated thicknesses across the zones of saturation.

5.1.3. Reference to Grid

As in table and cover references, a grid reference can be used to build a grid or as a graphical layer.

With option "**Use to build grid**" enabled additional controls appear in the reference properties dialog. Use this controls to establish the relations between currently mapping surface and the reference grid.



Four types of equations are available: linear dependence, first and second derivatives proportionality and common conformity. If a weight set opposite the equation is zero, then this equation is not used.

The first equation is a simple linear dependence: $U = a\,G + b$,

Second -

$$\frac{\partial U}{\partial x} = c\frac{\partial G}{\partial x} + c_1, \quad \frac{\partial U}{\partial y} = c\frac{\partial G}{\partial y} + c_2,$$

Third -

$$\frac{\partial^2 U}{\partial x^2} = d\frac{\partial^2 G}{\partial x^2}, \quad \frac{\partial^2 U}{\partial y^2} = d\frac{\partial^2 G}{\partial y^2}, \quad \frac{\partial^2 U}{\partial x \partial y} = d\frac{\partial^2 G}{\partial x \partial y},$$

Fourth –

$$\frac{\partial U}{\partial x} \cdot \frac{\partial G}{\partial y} - \frac{\partial U}{\partial y} \cdot \frac{\partial G}{\partial x} = 0,$$

Where U is currently mapping surface; G - a reference grid; a, b, c, d, c_1, c_2 are coefficients. Coefficients for these equations can be user-defined or be set as unknowns and computed at solving the mapping task. In the latter case, after grid build, user can find the obtained values in the coefficient fields.

The first equation establishes the most rigid relation between the mapped surface and the indirect surface.

The second equation sets rigid conditions for the mapped surface gradient (via conformity coefficient ***c***), yet leaves free its absolute location (vertical shift). Coefficients c_1, c_2 are zeroes by defaults and establish general inclination of the mapped surface to the reference surface along the relevant coordinates.

The third equation describes an even more flexible dependence. It demands that domes, troughs and bends should coincide, but leaves free the gradient value and the absolute shift.

The Fourth equation (common conformity) represents the condition of contour parallelism applied to the mapped and indirect surfaces. Starting from **GST** Version 6.8.3, this condition is formulated both in the “strong” and “soft” forms. In the first case, the functionality requires strict parallelism of gradients (or isolines) and in the second arbitrary angle indexing is allowed. Along with the parallelism condition, there is a new option: setting the perpendicularity condition. For this purpose, you must select “**Normality condition**” option from the “common conformity” category.

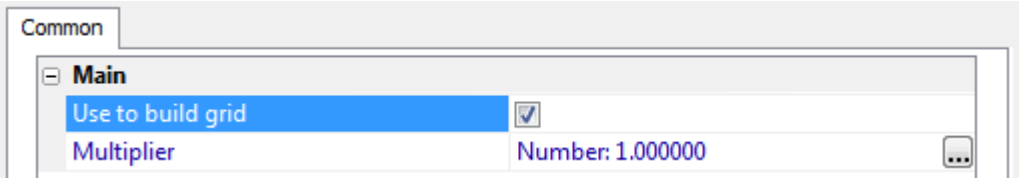
The Fifth equation introduces variable anisotropy, (as defined by the geometry of indirect grid) into the map-building tasks. This anisotropic setting can use the minimum suface and minimum curvature functional, as well as parameterized ellipse anisotropy (axes ratio). The **axes ratio** regulates the elongation of the mapped grid in directions determined by the geometry of indirect surface. When the axes ratio equals to 1, the resultant grid is equivalent to a grid built with a conventional method, using a minimum curvature or surface. Parameters of the ellipse can be specified either by a number or by grid. In the latter case, a significant reference must be added to this grid. If the “**Ellipse**” option is disabled, the axes ratio of the anisotropy ellipse is determined by the value and direction of the gradient across the indirect surface.

The option «**Use for mergence**» allows to tightly patch together the rims of the current grid with the reference grid. The patching by values and the first derivatives is done along the meet line of the rectangle of the grid under construction and the reference grid area.

The option «**Use as fragment**» allows to imply the reference grid onto the currently mapping surface. The grid may be implied either as a whole or according to the polygon defined below. In the latter case the grid should contain a reference to the «**Polygon cover**». As a result, within the defined area the resultant grid will be an absolute copy of the reference grid (deviations are possible in area immediately adjacent to border), while other data in that area will be ignored.

5.1.4. Reference to Faults

To build grids with faults will be discussed in detail below. In the reference property dialog enable option «**Use to build grid**» and define the multiplier when necessary, which by default equals 1. The multiplier may be a number, or defined by the grid (in the latter case, significant reference to that rid must be made).



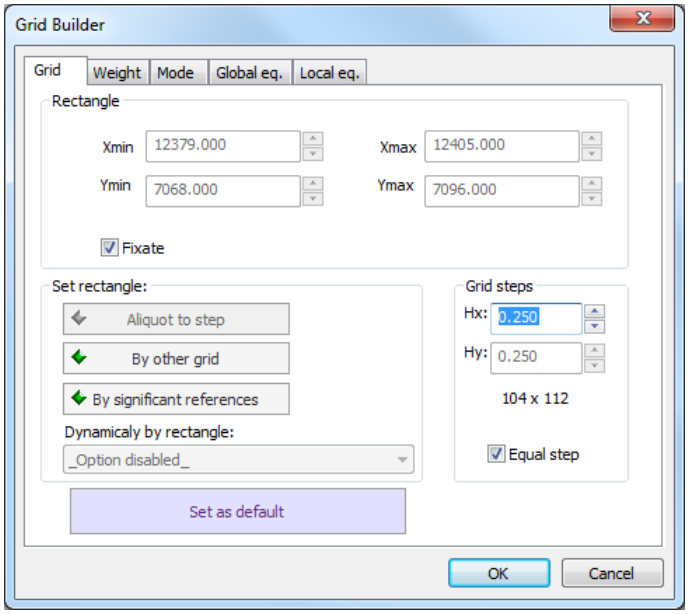
5.2. Grid-Builder

Having all reference parameters specified, set grid-build parameters using the dialog «**Grid-Builder**» (shortcut «**Build settings**» in hierarchy tree or toolbar button). The «**Grid-builder**» establishes:

- Grid rectangle and gridding step;
- Smoothing parameters (weight coefficients and anisotropy settings);
- Faults-gridding parameters (if necessary);
- Optimization parameters (to build large grids);
- Fine-tuning for indirect and point information (Global and Local equations).

5.2.1. Setting a grid rectangle

In the tab «**Grid**» user specifies a grid rectangle and gridding step as well as a number of other properties.



Firstly, specify grid steps (in project units). If it is necessary to use different steps along the axes, disable the option "**Equal**" and set a step along the vertical axis.

Then define a grid rectangle. Grid coordinates can be input manually – arrows will change them by a step. Option "**By significant references**" allows the user to set the grid boundaries to account for all available information. Grid parameters can be copied from another grid – to perform this, click "**by other grid**" and select the source grid in the list. If the grid contains references to **Rectangle** objects**,** user can select gridding area from "**Dynamically by rectangle**". In this case the grid rectangle will change as the object **Rectangle** changes. Option "**Aliquot to step**" easily changes rectangle boundaries to make its coordinates divisible by the step selected. The option "**Fixate**" makes input field inaccessible to avoid accidental coordinate change.

5.2.2. Smoothing parameters, anisotropy

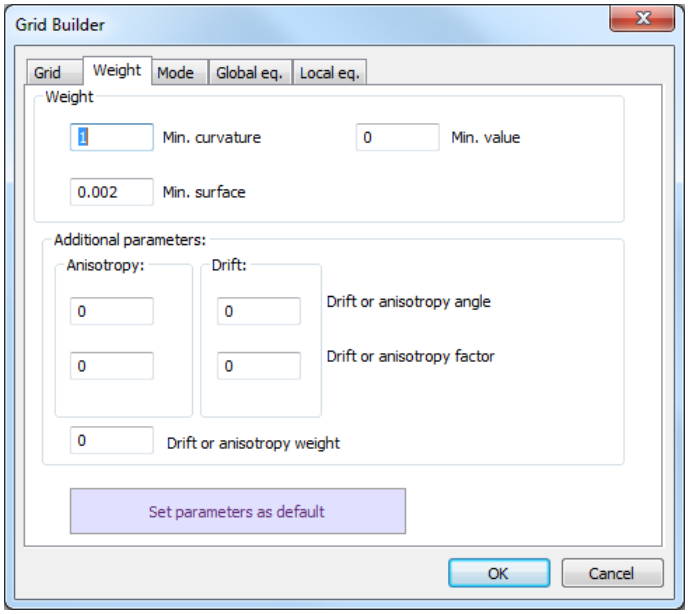
Smoothing parameters (stabilizers) are defined in the **«Weight»** tab. Users can also establish additional parameters to model spatial anisotropy.

Min. curvature. If a weight coefficient is larger than zero, then a minimum-curvature condition (minimum of second derivatives) is applied to a surface being mapped. This stabilizer is preferable for structural surface mapping.

Min. surface. This stabilizer corresponds to a minimum-surface condition (minimum of first derivatives).

Values. This stabilizer is applied very rarely and corresponds to a minimum-value principle.

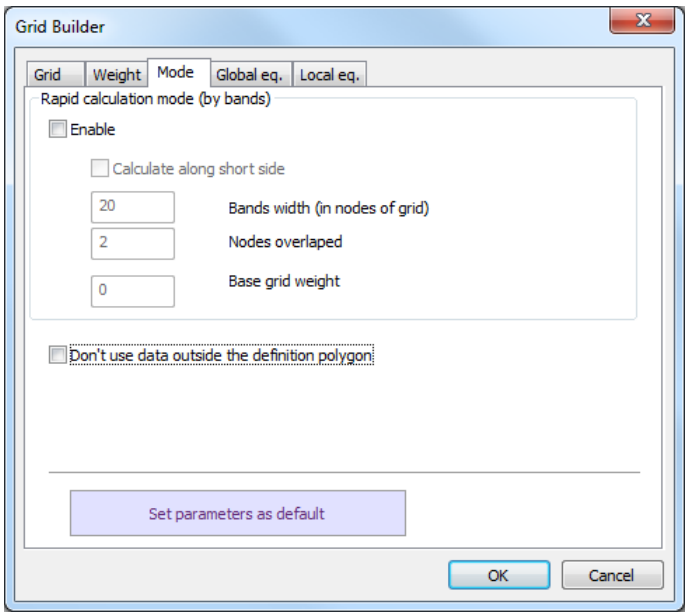
An optimal ratio between weights on stabilizers at structural map construction is: **Curvatures – 1, Surfaces – 0.1-0.2**. Here, the first coefficient provides the map smoothness needed, while the second one does not give the surface values in extrapolation areas to go far beyond the initial-data values.



Anisotropy coefficients (**angle** and **factor**) are responsible for a scale in the given direction. If a factor is 1, then a task is isotropic. A physical meaning of a factor in "drift" coefficients is velocity. If a factor is zero – there is no drift. A weighting factor defines a degree of anisotropy-consideration impact at mapping. However, for majority of tasks it is enough to specify weights on curvatures and areas.

5.2.3. Grid computation speed-mode setup

At the tab "**Mode**" the user can set up grid calculation rapid mode. In this mode the grid calculation is made by bands, rather than for the entire area. Since the grid-build time cubically (non-linearly) depends on the number of grid nodes, the gridding process in this mode requires less time than what is needed for full-task solution. The rapid mode takes less computer resources also.



To set up the grid computation speed mode: A grid is divided along its longest side into a certain number of bands. Bands width (in grid nodes) is specified by the user. To avoid input data conflict at bands boundaries, the bands are overlapped (next parameter); this overlap, however, being not more than ½ of a band's width. Grid computation starts from the central band. To stabilize the process in sparse-data grid zones, the software, before dividing the grid into bands, constructs a sparse-grid, with automatically defined steps. A sparse grid is used as the indirect information with a weight coefficient set in the lower input box. If this coefficient is zero, then gridding is done without the prior sparse-grid map build.

At sufficient density and uniformity of data distribution, the weight on a sparse grid can be set equal to zero. In any case developers recommend to first build a separate sparse grid instead of using this factor, and only then apply this sparse grid as the indirect, or referenced information.

The larger a zone width and overlap, the higher the quality of the rapid-mode constructed grid.

5.2.4. Global and local equations

While constructing a grid, the user can specify conditions in a form of generalized differential second-degree equations. These conditions are specified at two pages of **Grid-Builder** dialog: **Global Equations** and **Local Equations**. Global equations are applied for the entire grid-build area, while local equations describe a surface at given points. More details on global and local equations are discussed in Section **«Common tasks»**.

5.2.5. Mapping statistics

To control the quality of grid build, **GST** has the capability to examine discrepancies between a built map and the input data. The dialog "**Statistics**" is opened when the user selects a command with the same name out of the menu "**Grid**", or clicks  in the instruments bar.

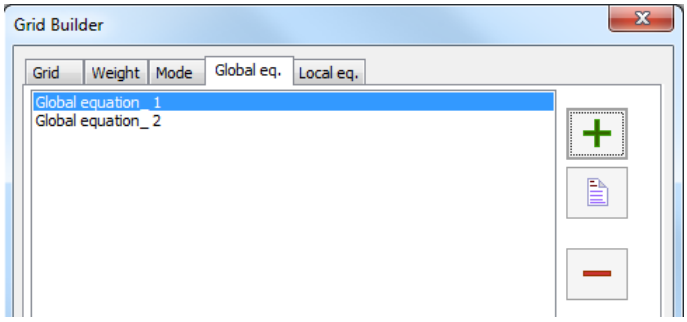
The list box contains all data (names of references) involved in the build process; in the lower part the user will find information on discrepancies (for each reference separately). Here, the user can activate a mode that displays data with large deviation from factual values of the mapping parameter (the option "**Show if error more:**", and specify a limiting value of an absolute error (points, at which an error exceeds the limiting value, are shown in the grid). To display such points, the user can specify size and color of the sign (circle) for each reference individually. The button "**Apply to All**" sets up identical color and size, as well as the same limiting-error value for all the references.

In case a data set has the option of showing a more critical error set up for it, the key  is activated in the toolbar, which highlights tabular lines where the mapped surface deviates from the data to a greater extent than the established one.

5.3. Common Tasks

5.3.1. Global equations

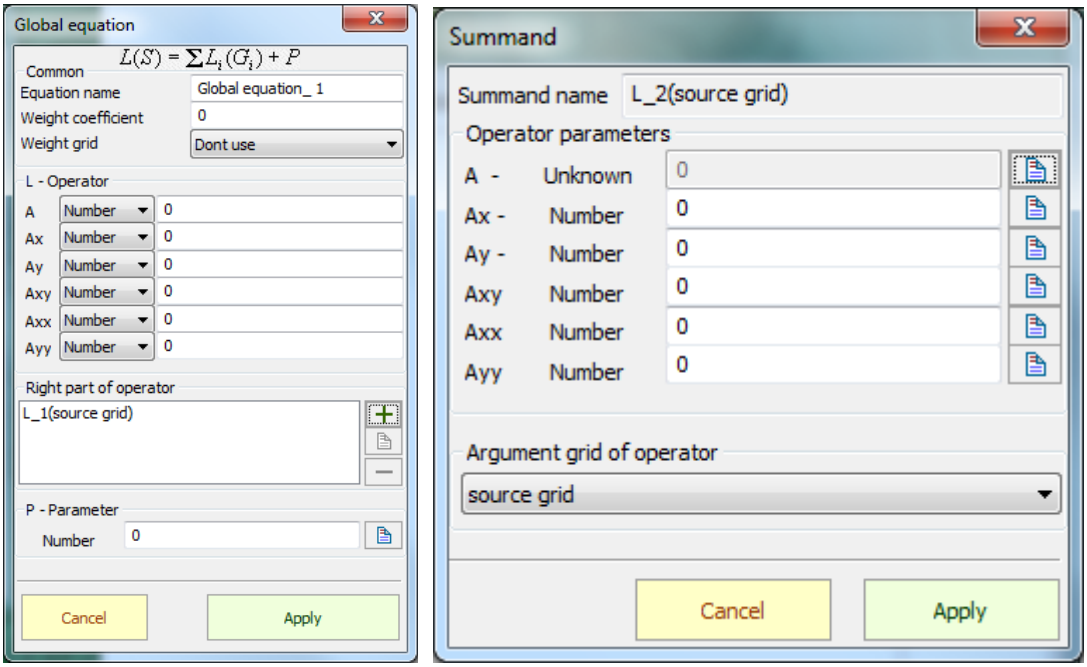
The tabs of global and local equations in "**Grid-Builder**" dialog look the same. To the left is the list box that contains names of user-created equations. Initially, this list box is empty. Using the button "**+**" (add) user can create a new equation. This button is always accessible. The buttons "**Edit"** and "**-**" (delete) are accessible only if one of the equations has been selected in the list.



The «**Add**» command opens global equation properties dialog. The «**Edit**» command opens the same dialog with parameters already set up.

In this dialog users can specify:

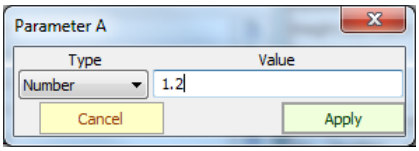
- Equation name;
- An equation weight coefficient;
- Parameters of the operator (L-operator) that affects a desired surface. The operator's parameters can be set as a number or grid, if the grid under construction has corresponding references to the grids needed. If the user selects "**Grid**" type for operator parameters, then a window for this parameter definition will look like a drop-down list box.

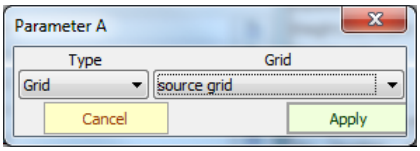


The right side of a global equation is specified with a sum of one of several operator's transforms of existing grids, which can be added, edited or deleted using the buttons "+" (add), "**Edit**" and "-" (delete). User can set a simple parameter (p-parameter) to the right part of equation.

The command "**Add**" opens a property dialog for a new right-part summand of the global equation being set up. The command "**Edit**" opens the same dialog with parameters that were specified previously.

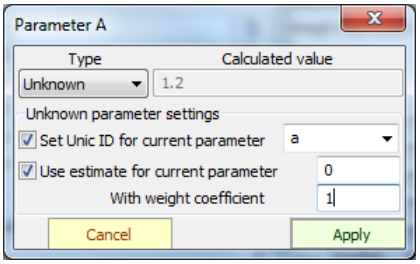
In this dialog the user can set operator's parameters and specify a grid which this operator affects. There are three types of right-side operator parameters: "**Number**", "**Grid**" and "**Unknown**". To change a parameter type, the user should press the button (change) which opens a parameter property dialog.





If the parameter is of the "**Number**"-type, then in the window "**Number**" the user can place a number needed (a "**Number**"-type parameter's value can be set in the previous dialog "**Summand**" as well). If it is a "**Grid**"-type parameter, in the drop down list the user can select a grid.

If a parameter type is set as "**Unknown**", the dialog takes the following form:



After grid computation a value calculated for this unknown parameter is entered into the window "**Calculated Value**". It is possible to specify more detailed information for an unknown parameter. The option "**Set Unique ID for current parameter**" helps to apply one unknown in different equations or in different parameters of operators related to one equation. When selected, this option makes accessible a field of an edited list box where the user should type in or select a unique name identifier needed.

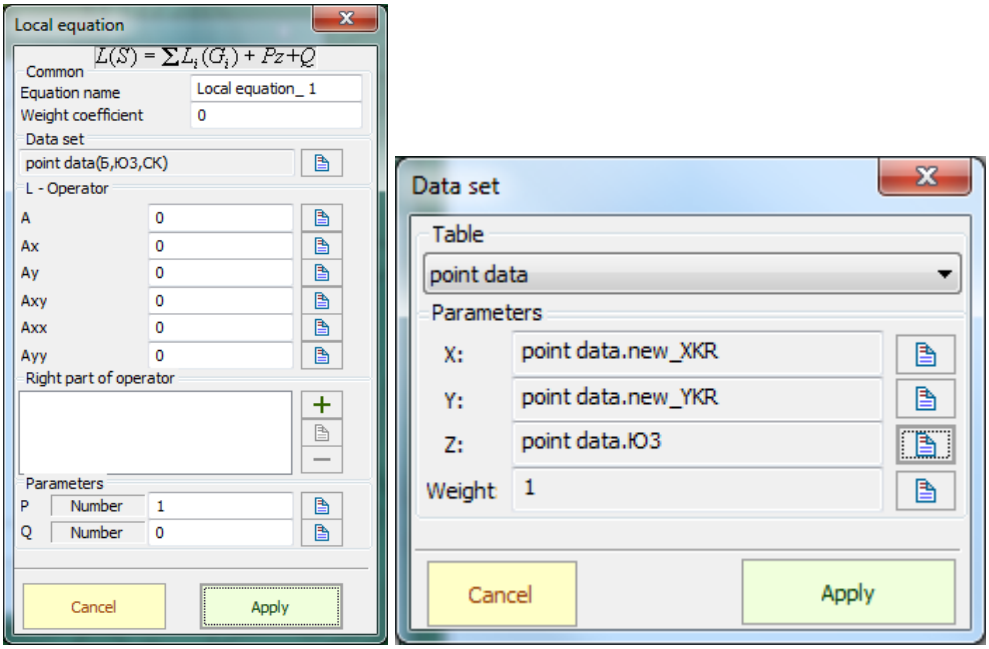
Users can specify an approximate value of the unknown parameter and manage the nearness of the estimate and approximate values using a weight coefficient.

This dialog window is also used to define a type and settings of the **P**-parameter in the right-side of a global equation.

5.3.2. Local equations

A local-equation-invoked interface is identical to the one for the global-equation case. However, a dialog invoked (edited) under a local equation has a new capability – it helps to define a data set (parameter data tables) and enter a second parameter to its right-hand side (**Q**).

The command "**Change**" in a group "**Data Set**" opens a dialog window where the user should select a table which parameters are used by this local equation. This command also defines the table coordinate-columns X and Y; the Z-column contains mapping values (i.e. a value included into a local equation in a form of the P-parameter multiplier) and local weight coefficients applied at a corresponding observation point.



5.4. Gridding with faults

Beginning with **GST 6.5,** fault gridding is done in two formats:

- Gridding the surface in the faulted area together with the folded surface. Fault gridding method uses the triangular-based bicubic splines (this has been the method used in earlier versions of the software);
- Separate construction of the folded and faulted parts and the presentation of the resultant surface as a sum of the two fields.

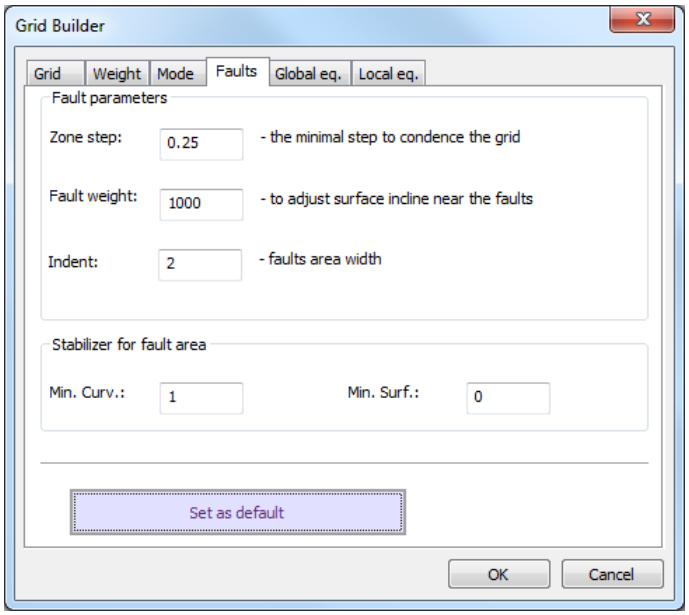
5.4.1. Joint construction of the faulted and the folded parts

- Faults are set up as a simple linear cover, and each fault is represented by a central line (not contour);
- Reference to faults is initialized as per Section 5.1.2; the grid can contain no more than one significant reference to faults.

The method boils down to constructing in the area of the fault line, and as an addition to the main grid, a package of inserted grids with the use of triangular-based splines. This way, the desired function indeed suffers a rupture and acts independently on either fault bank. The grid build is done as follows:

- Build a folded grid using all information available in the task.
- Mark grid cells in the vicinity of the faults. These are the cells that contain the fault and the cells removed from the fault line to a distance smaller than the one specified by the parameter «**Indent**». Its value depends on the number of steps in the grid and by default equals 2.
- Approximate fault line as lines drawn through the edges and diagonals of the marked cells.
- Triangulate the marked pre-fault area.

- Solve the gridding task in the fault area using all point information in it. The values and derivatives on the outer edges of the triangles coincide with the values and derivatives of the folded part, while on the edges of the triangles that coincide with the faults, the function gets ruptured to an extent resulting from task solution.
- Mark cells containing the fault. These are used as the basis for a new triangulation with a grid step twice smaller and the task of mapping pre-fault area is resoled again.
- The grid density in the pre-fault area reaches the value established by the parameter «**Zone step**».



All gridding parameters for the fault area are set up in the tab **«Faults»**. Users can specify stabilizer weights in the fault area (by default, they are the same as in the folded grid). In case the accuracy of the mapped surface in the fault area passing through the point data is insufficient, it makes sense to reduce stabilizer weights. The value of the parameter **«Zone step»** commands the number of inserted grids in the package and, hence, the time of computation. The zone step should be smaller than or equal to the grid step, but should not be too small to prevent high computation time requirements. The zone step of a fault is usually 4-6 times smaller than the grid step.

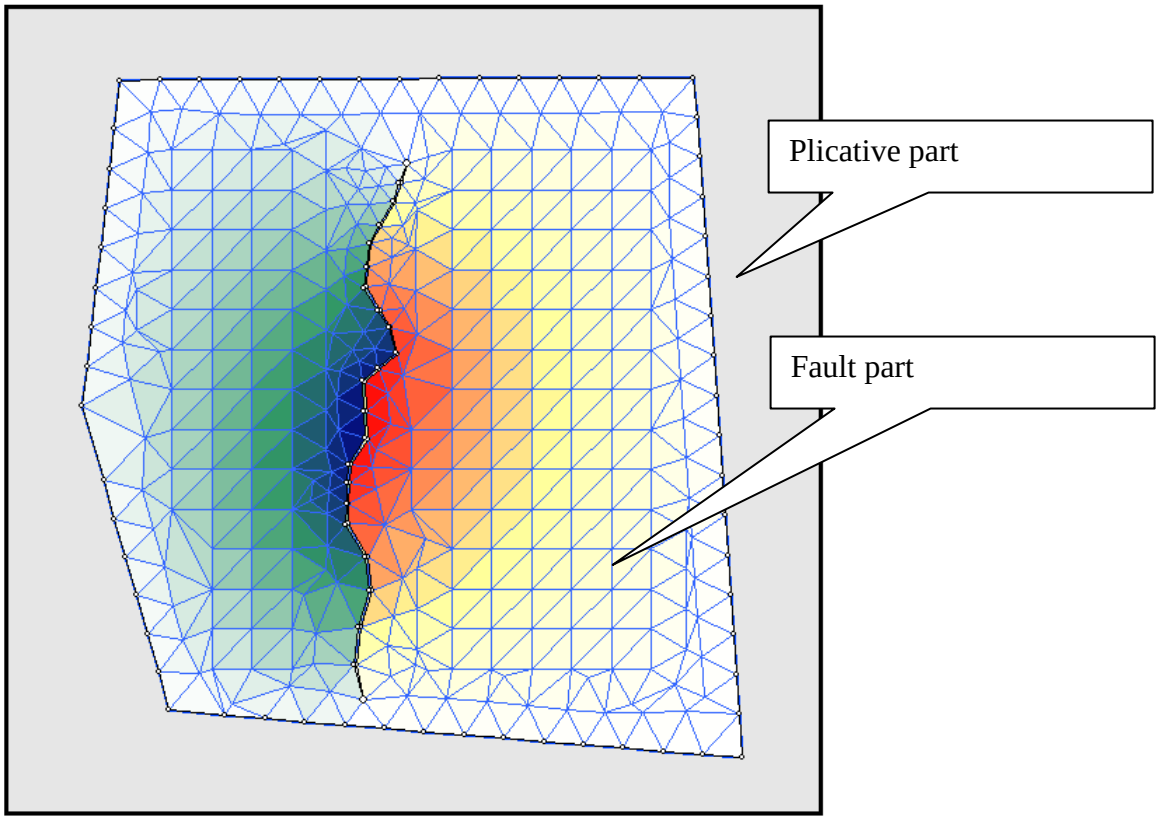
If need be, establish «**Fault weight**». Increasing this weight alleviates derivatives on the opposite banks of the fault, which corresponds to a clear vertical shift.

5.4.2. Separate construction of the faulted and the folded parts

- The displacements field is set up in the form as a separate (one or several) **GST** «**Fault**» objects.
- Each fault is represented by a contour, i.e. has a right and a left bank.
- The grid under construction can have one or several significant references to «**Faults**» which are initialized as per Section **5.1.4**.

The main idea is to present the mapped surface (**S**) as a sum of the folded part (**P**) and fault part (**T**). The folded part can be viewed as an un-faulted surface, and the fault part, as tectonic displacements inside the fault area. The mapping task is divided into two:

- Construct a fault model in a separate «**Faults**» object.
- Construct a folded part with the use of all capabilities of the **Grid-Builder** and fault objects.



The fault part is a grid composed of irregular triangular elements (see figure above). Nodes contain displacements values while the fault bank lines coincide with the edges of the triangles. The external contour can have any shape – it can either be inside the mapped area or extend beyond its confines. Inside the contour the faults can form blocks of any degree of inclusion. The fault part may consist of several *non-overlapping* «patches», each of which is a separate «**Fault**» object. Inside the contour the fault field **T** is non- zero, and equals zero outside.

The folded field **P** is estimated on the basis of the displacements field duly adjusted to the used point and indirect information (adjustments are done automatically). The resultant surface is as follows:

S(x, y) = P(x, y) + A(x, y) T(x, y),

Where **A(x, y)** – is amplitude multiplier that can be established as a constant or a grid (see Section **5.1.4**). If, for instance, the fault model is based on seismic data in the scale t_0, and S is the structural surface, the amplitude multiplier has the meaning of a field of velocities.

5.5. Fault Objects

An object of this type is a **GST** object reflected in the tree and partially inheriting the properties of «**Cover**». The building of faults is going through three main stages:

- Create fault geometry;
- Create triangulation grid;
- Establish displacements at faults banks and calculate displacements field.

Several methods to build faults are suggested:

- **Load geometry and displacements from file**. Geometry and displacements are either loaded from file, or adjusted by user manually as needed.
- **Create geometry and displacements with Editor**. Both geometry and displacements are defined by **editor** functions.
- **Create geometry and displacements by references**. Both geometry and displacements are constructed using point data.
- **Geometry from file, displacements by references**. Geometry is loaded from file; displacements are computed based on point data.
- **Geometry with Editor, displacements by references.** Geometry is created using **editor** functions; displacements are computed as based on point data.
- **Link to other project**. The object is fully copied from another project.

The object «**Faults**» includes two main structural elements:

1. **Editable layer**. Layer in which user can prepare faults geometry using **editor** functions.
2. **Data layer**. The basic element created with editable layer data and containing the faults geometry, triangulation and displacements field.

5.5.1. Loading faults from file

Importing faults from file corresponds to two methods of grid build. In the former case (**Load geometry and displacements from file**) both fault lines and displacements may be imported from file. If there is no displacements information in file, the user may establish the displacements using **faults-editor** functions. In the latter case, only geometry is imported from file, while displacements are computed from references to different point data. Loading faults from file if done by standard import-export **GST** procedure.

Object «**Faults**» can be loaded from an «***.fgs**» file, a binary internal **GST** format. The file in this format contains both the lines of the edited layer, and the main data layer, i.e. geometry, triangulation and displacements field.

While being imported from files of other formats (*.con, *.shp, *.cps, etc.) the fault lines and blocks are loaded into the **edited layer** where user can edit them and add to the **data layer** as needed.

By importing fault files in **CPS** user loads both the fault lines and their amplitudes which are then used to compute the displacements field.

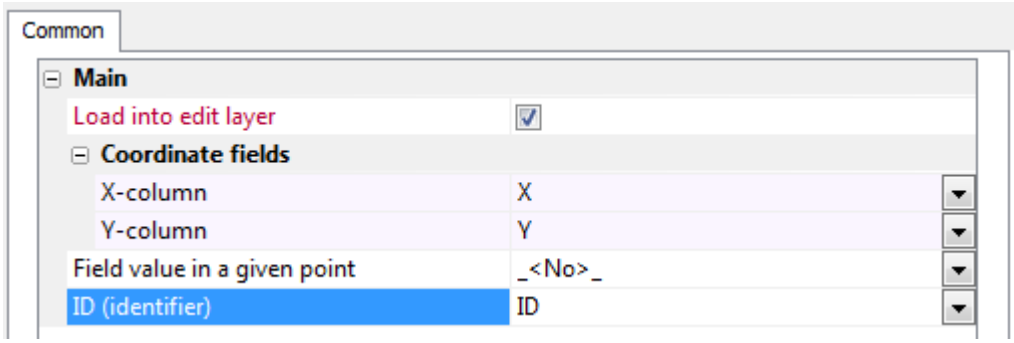
5.5.2. Creating faults in Editor

At stage 1, the user creates fault geometry in the edited layer using the **editor** functions. Working with edited layer of faults is fully analogous to working with the relevant layer of simple cover.

At stage 2, if the method «**Create geometry and displacements with Editor**» is chosen, the user manually indicates displacements in the fault banks and computes the displacements field. In case the method «**Geometry with Editor, displacements by references**» is picked, the displacements field is computed based on references to point data.

5.5.3. Creating geometry and displacements by references

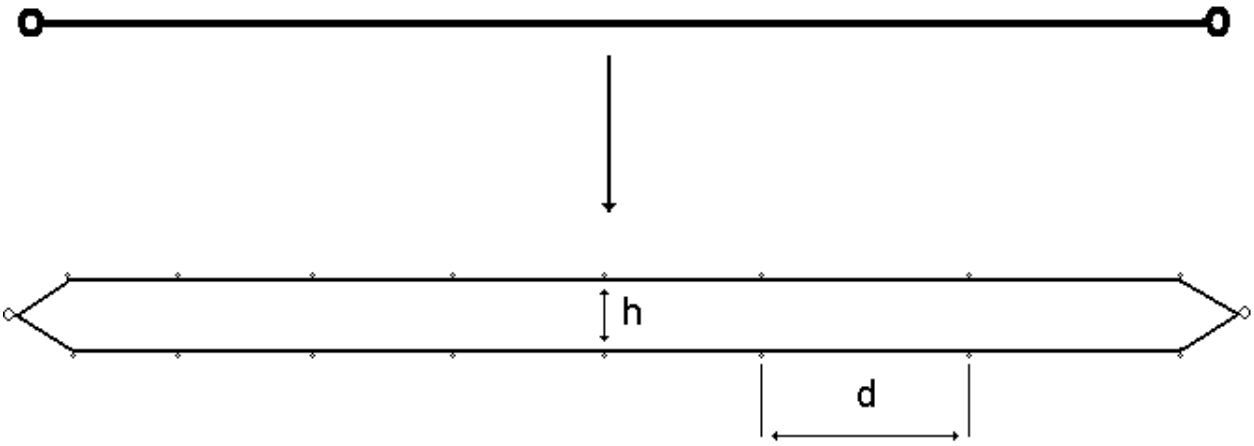
In case fault geometry contains in a table with the format of : **coordinate X**, **coordinate Y**, fault line number, displacements at fault (field not necessary), then this method is quite usable for object construction. Users need to add reference to the table and indicate fields as shown below.



Point data is transformed into lines and loaded into edited layer. In case the table contains a column with fault amplitude, their values are specified for each fault node. In case the displacements column is not included, these values may be established by user manually or computed as based on point data (see one of sections below).

5.5.4. Creating fault geometry

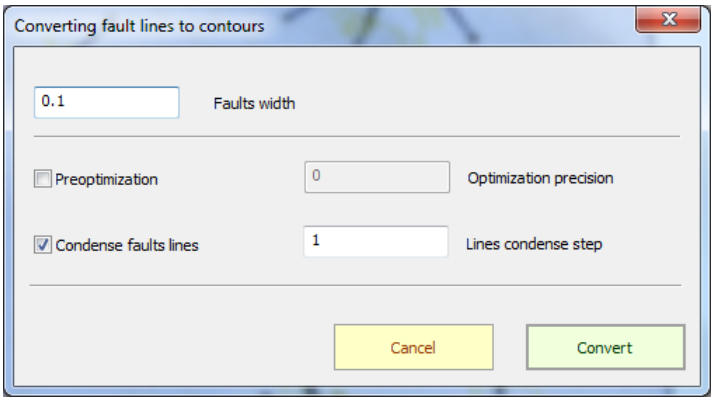
When faults are imported from file (with the exception of the internal format **fgs**) or are created by reference, the fault lines and blocks are loaded into the **edited layer of object**, which features all functions of editing the linear cover. Fault geometry is subjected to preliminary preparation in the edited layer.



A fault is shown by **GST** as a crack of certain width h (see figure above), i.e. a closed line. Width "**h**" may be small but always larger than zero.

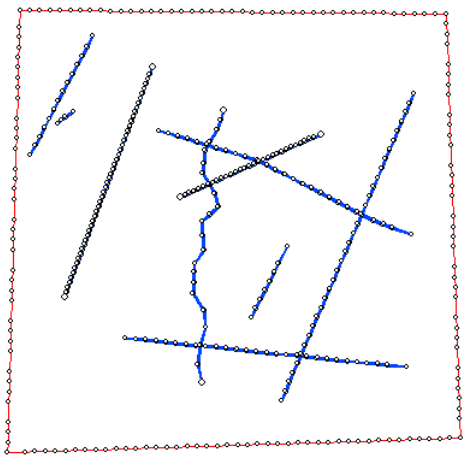
In case faults are set as central lines, they can be easily transformed into contours by menu command «**Edited layer → Operations with lines → 'Line → Contour' transference**» or by clicking  on the toolbar. In the dialog that pops up the user defines distance between fault banks (**h**), line density step (**d**), and options of preliminary line optimization. The optimization is needed in case the fault lines are established with an excessive number of nodes. The procedure removes

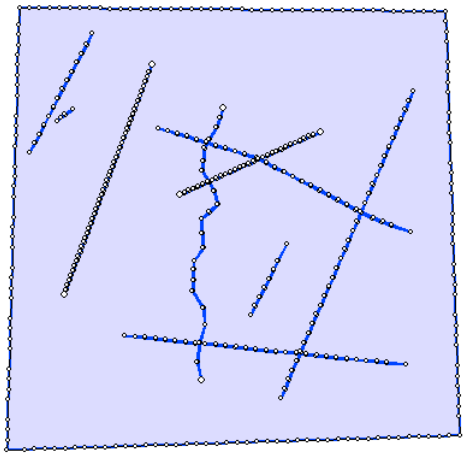
intermediate nodes that do not impact the line geometry. The parameter «**Optimization precision**» specifies the cutoff distance to which the optimized line can diverge from the source line.



While creating fault geometry with editor, it is most convenient to draw a fault as a central line and then translate it into a contour.

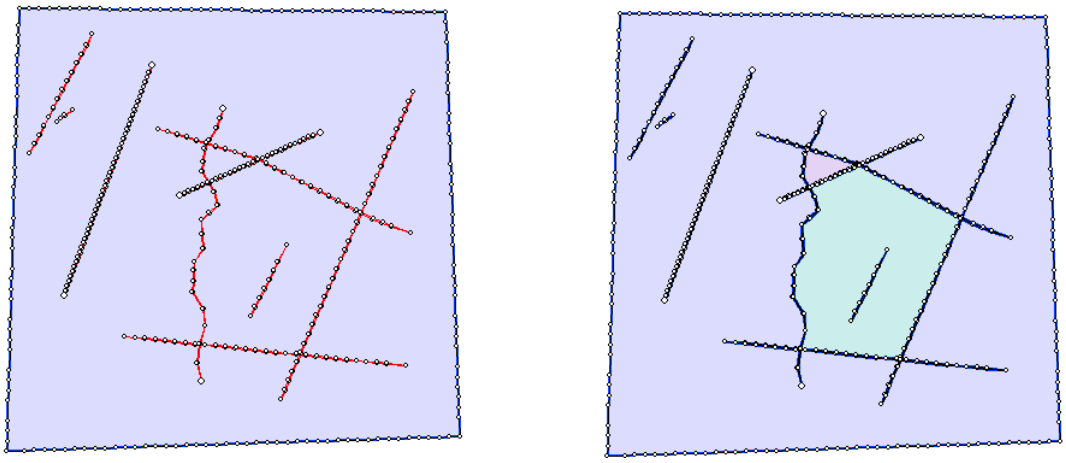
Forming the geometry of faults begins with the addition of an **external contour**. The external contour may have any shape, but it's always unilaterally connected, and may include, fully or partially, the mapping area. Outside the external contour the value of tectonic additive to grid equals zero.



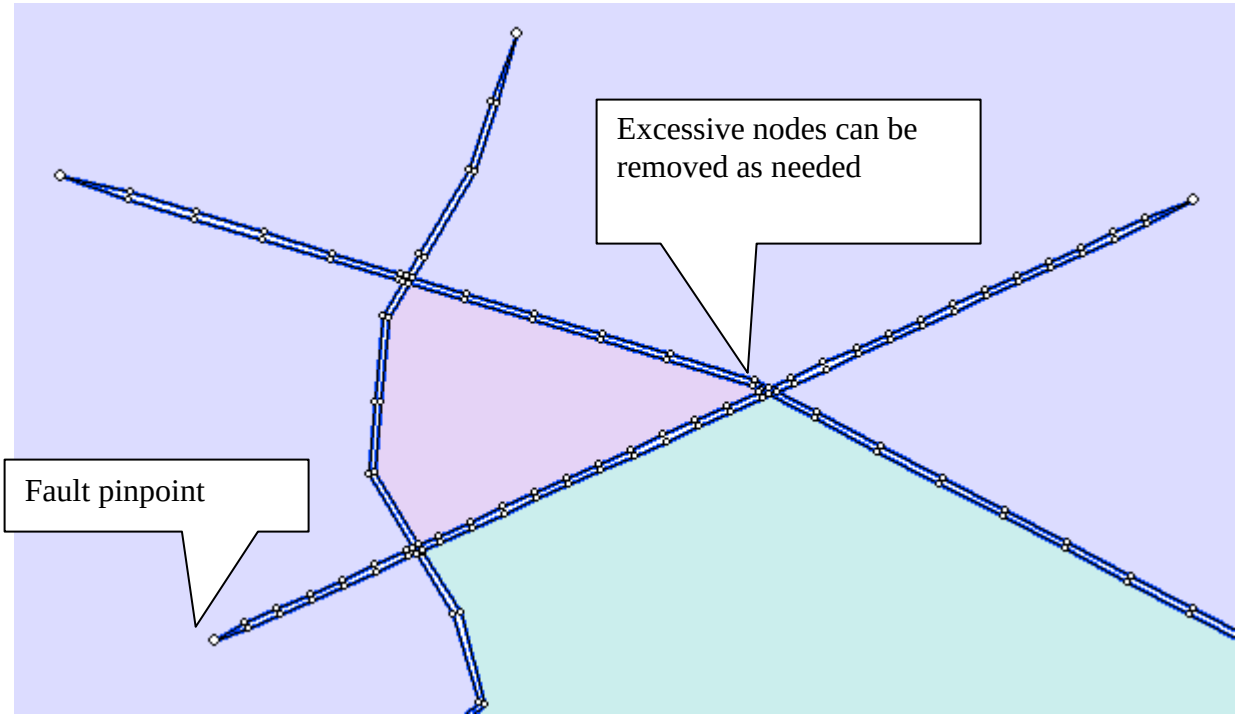


To add external contour, user marks one closed line (see figure above) and execute menu command «**Faults → Add external contour**» or click on the toolbar on the right. If successfully performed, the faults area will be filled with color. When geometry is imported from file or created by reference, the external contour is generated and added automatically. User then marks faults translated into contours (see figure below) and executes menu command «**Faults → Add faults**» or clicks on the toolbar.

In case faults are successfully added, the area inside the faults contour gets filled with white color, and blocks that have formed get filled with other colors. Faults can be added all together or individually: **GST** will automatically unite the contours and form blocks. A fault that needs to be added can exceed the boundaries of the external contour – in this case it is cut off at the boundary.



Following the addition of faults, the geometry can be viewed as complete. The nodes in the external contour line and the tectonic fault borders are shown as circles (see figure below); the fault pinpoints are shown as slightly larger circles.



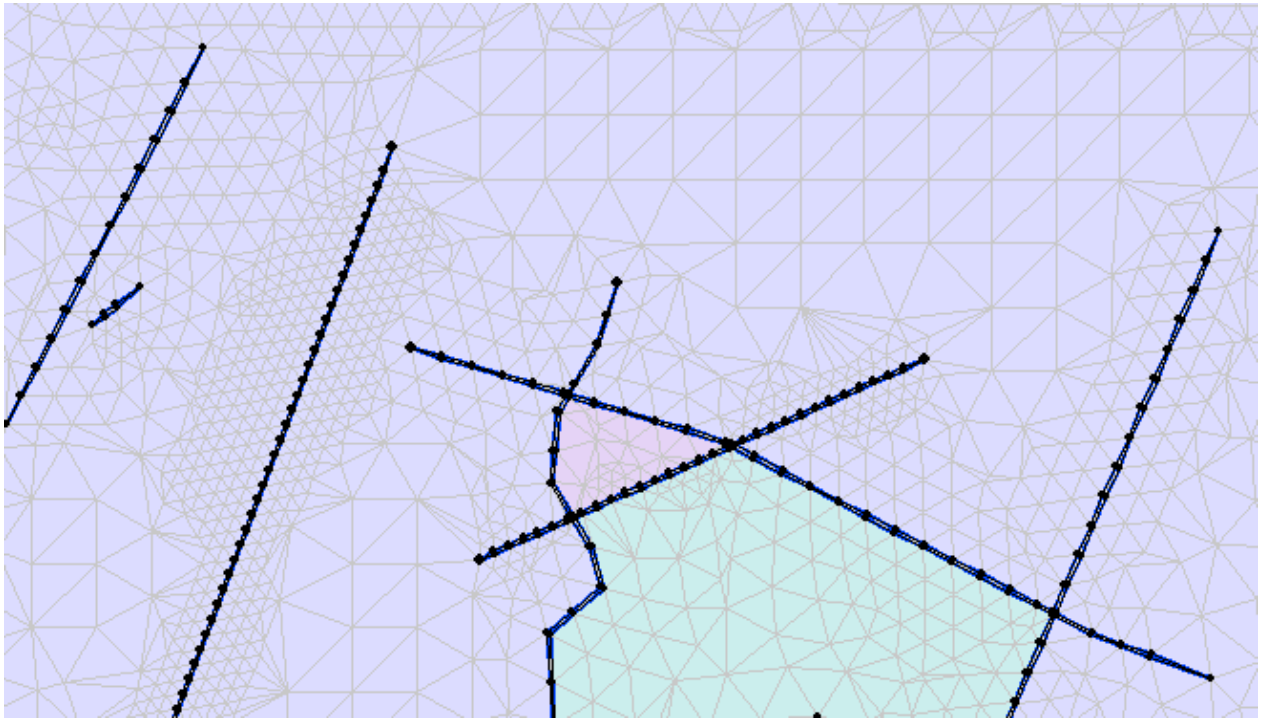
The nodes form boundary segments along which the area triangulation will be done. In case faults cross to form an excessive number of closely packed nodes, the geometry can be updated by switching on the **Faults editing mode –** by clicking on the toolbar. The editing mode gives access to the context menu (by right-clicking the mouse) that contains the main functions of editing fault geometry and node parameters. With the context menu functions user can **remove an element** of a boundary (nearest fault or block), **remove** a fault, **add** a new faults, etc.

5.5.5. Triangulation

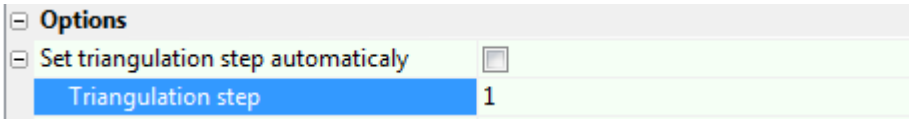
Tectonic additive to the grid is a network of irregular triangles. The triangles edges coincide with segments of the external contour, faults and blocks. Hence, triangulation is only possible after the

faults geometry is complete. To begin triangulation, execute menu command «**Faults → Triangulate**» or click  on the toolbar.

If triangulation is successful, a picture very similar to the one in the figure below will appear. To make calculations faster, in areas far removed from faults the grid of triangles is regular, and the triangulation step is defined by the user or automatically. In the vicinity of faults or the external contour the triangulation is as detailed as required to make it fit the boundary.



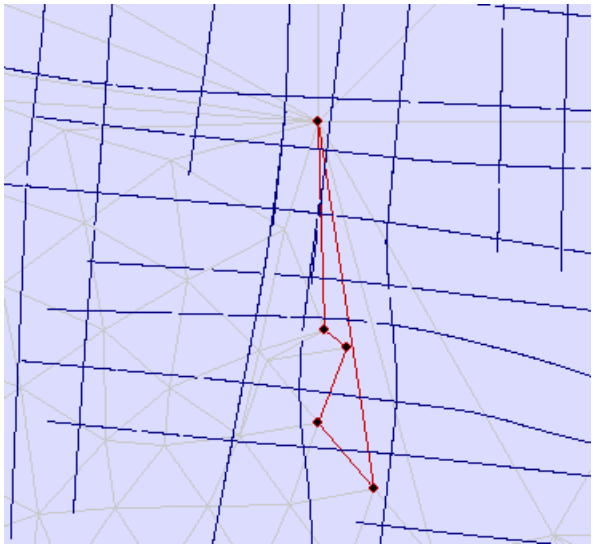
Users can establish triangulation parameters in the object's properties dialog (category «**Options**»). A step of initial regular triangulation is defined in "**Triangulation step**" property. If the option «**Set triangulation step automatically**» is enabled the step is established according to the largest boundary segment.



The software signals if triangulation cannot be completed (in view of boundary complexity, as a rule), and the problem edges get marked red.

In this case, there is a choice of measures to be used:

- Remove excessively close nodes (which results in the appearance of a large number of small or narrow triangles);
- Include additional nodes (in case a boundary element is not detailed enough);
- Change the step of the initial regular triangulation.
- Switch over to the mode «**Faults → Manual triangulation editing**» (click  on the toolbar on the right).



In the manual mode the user left-clicks the mouse to connect "free" nodes and the software automatically completes the triangulation.

5.5.6. Defining conditions on the external contour and faults

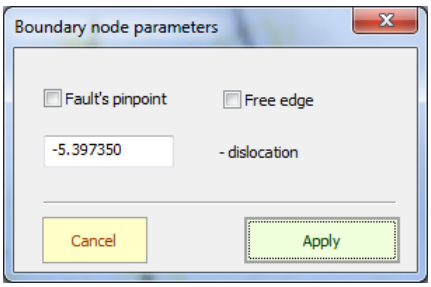
The field of tectonic fault *T(x, y)* in **GST** is identified as solution of the following differential equation:

$$\frac{\partial^2 T(x,\ y)}{\partial x^2} + \frac{\partial^2 T(x,\ y)}{\partial y^2} = 0,$$

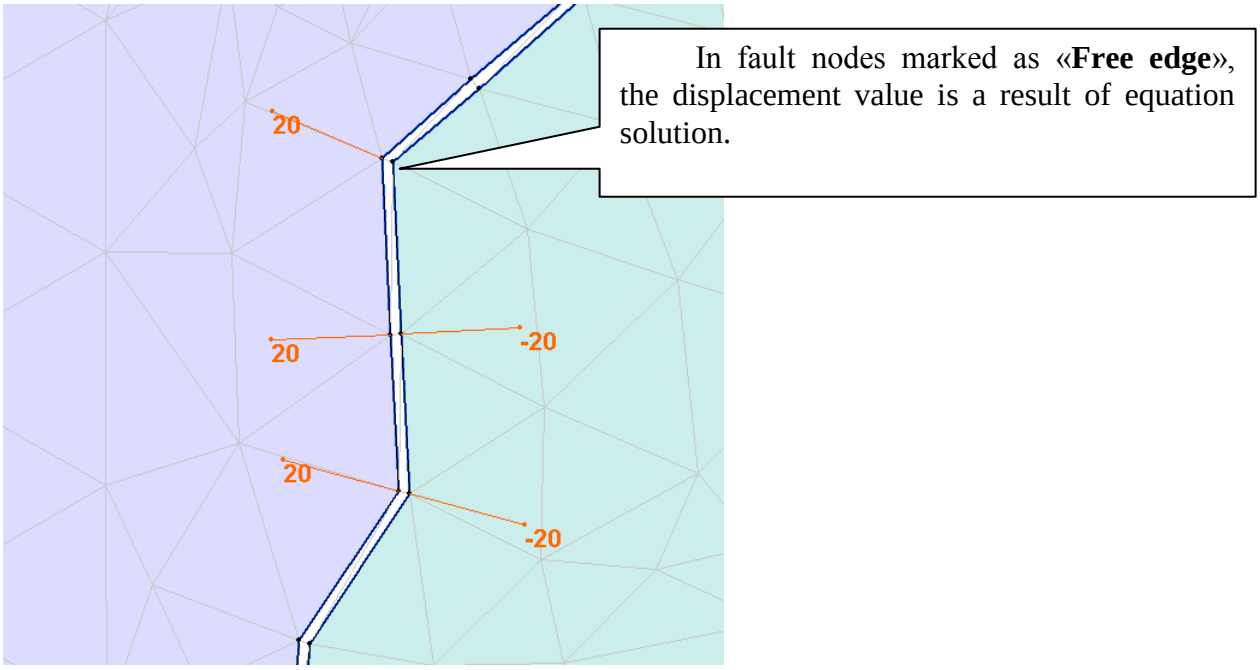
which corresponds to the physics of vertical fault. To solve the equation, the user should define boundary conditions in the nodes of the external contour and all faults. **GST** uses two types of boundary conditions:

- **Fault amplitude** – the node contains a definite displacement of a fault bank element against its central axis;
- **Free boundary** – the node contains the condition of the first derivative function *T(x, y)* being equal to zero in the direction of the surface normal towards a boundary element. The displacement value in this case is a result of the equation solution.

To set a boundary condition users have to enable the **Fault edit mode** (click  on the toolbar) and execute context menu command «**Node parameters**» (right-click the mouse). The following dialog will appear.



In this dialog user establishes the type of boundary element – fault pinpoint, free edge, or displacement (dislocation) value. The node marked as «**Fault pinpoint**», is shown as a large circle, while the fault amplitude is defined as equal to zero. If a node contains a displacement, its value is shown on the screen as a signature (see figure below), and nodes marked as «**Free edge**», are unsigned.



To establish an amplitude of a certain value (e.g. 40, see figure above) at a certain segment the user needs to define in the nearest opposite pair of nodes a set of displacements so that the difference between them equals the desired number. Since at the fault pinpoint the amplitude is zero, the value defined in the node describes the vertical displacement relatively the pinpoint. The displacements ratio in the nearest opposite nodes defines which fault bank is "higher" and which is "lower".

Users need not set specific displacements in all nodes of the fault; it suffices to set them for some of the nodes and mark the rest of them as «**Free edge**». **GST** then automatically computes the amplitudes proceeding from the Laplace equation. Users can set the same parameters to all nodes of an individual fault or block by executing context menu command «**Edge element parameters**». Also, one can set the "**Free edge**" or "**Fault pinpoint**" property to a specific node using the relevant commands of the context menu.

GST features a handy method of setting up and editing faults and blocks displacements when user operates the mouse to deform a fault bank or move a separate block. By choosing in the context menu the modes «**Positive displacement**» or «**Negative displacement** » (on the toolbar) and left-clicking the mouse next to a boundary element, the user defines displacement for the boundary element. If the nearest element is a fault bank, then all nodes located between the two pinpoints (fixed nodes) are given an additional displacement value that equals "**incr**" for the nearest node and goes down to zero as one moves closer to the fault pinpoints or the fixed points. If a fault does not have at least two nodes marked as "pinpoint", the software gives an error warning. If the nearest element is a block formed by faults and it does not have at least one fixed point, then additional displacement "**incr**" will be set to all nodes. With the additional mode «**Dislocate one node**» enabled ( ) changes will apply only to one nearest node, not the element as a whole. Users can set the increment value by executing the context menu command «**Set increment**» (should only be a positive number).

If all fault nodes have zero displacements, this fault will not be involved in the computation of the tectonic displacement field. If all fault nodes of are marked as "**Free edge**", the software emits an error warning and marks the boundary element.

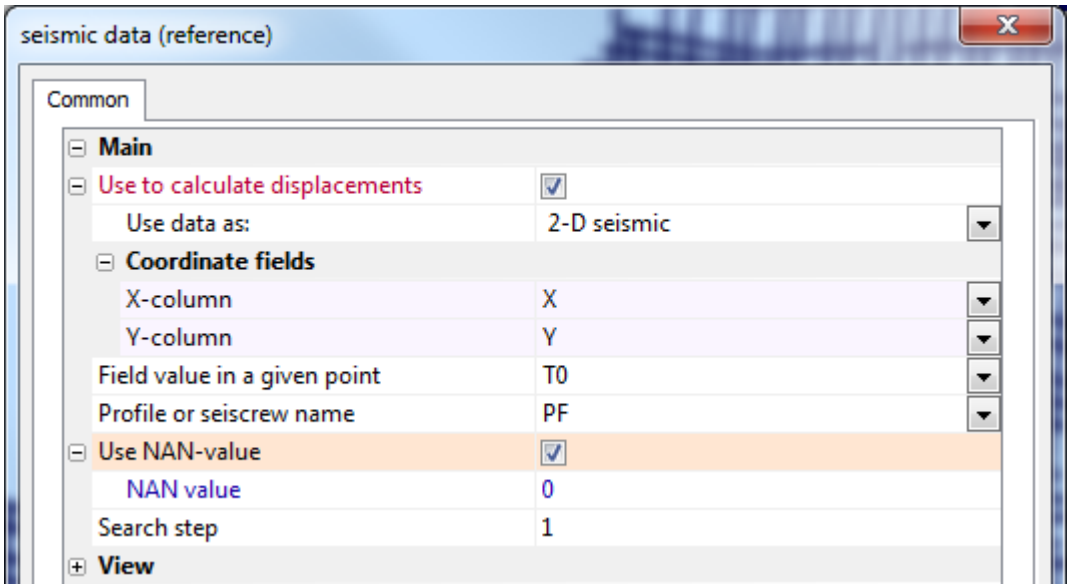
To set boundary conditions for the external contour user have to apply the same steps as for faults and blocks. If external boundary nodes are inside the gridding rectangle, user must set zero displacements for these nodes. If external boundary nodes are outside the gridding rectangle, user may set any boundary conditions for these nodes – non-zero displacements or "**Free edge**".

After the triangulation has been complete and all boundary conditions are set, user may start the displacements calculations by menu command «**Faults → Calculate dislocations**».

5.5.7. To define faults amplitudes automatically using point data

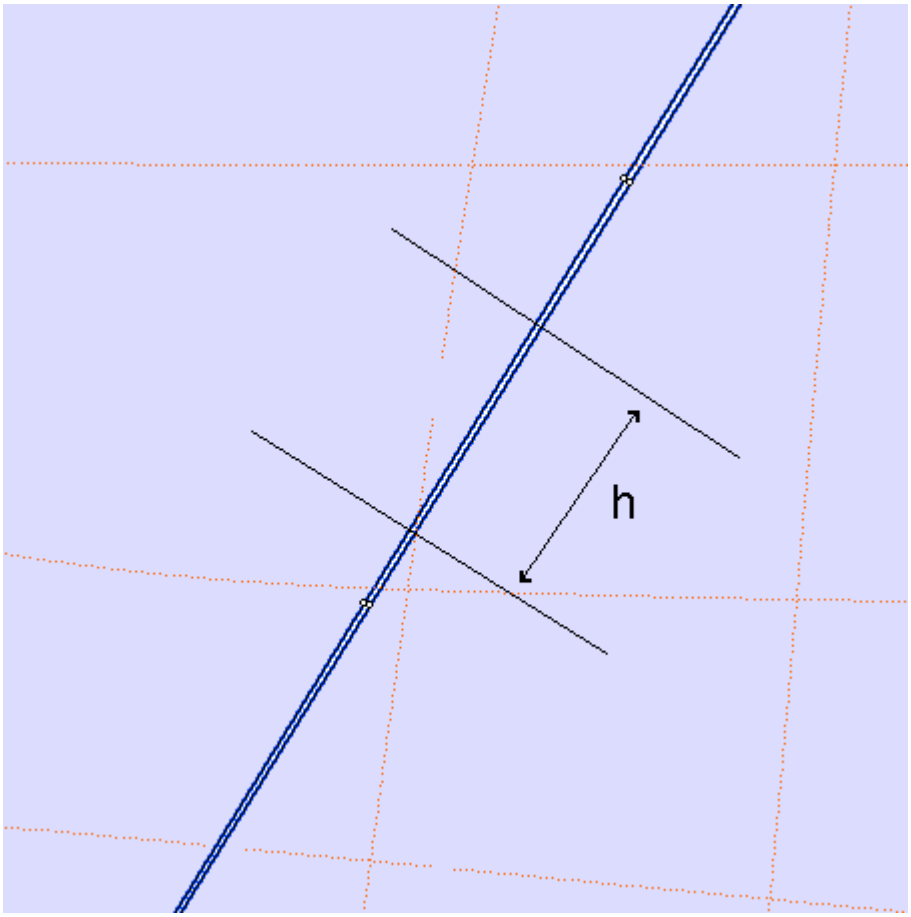
If the faults are confirmed by 2D or 3D seismic data, the data can well be used to automatically define fault amplitudes.

First user needs to choose a corresponding build method, and then add a reference or references to the seismic data.



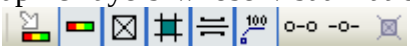
In the reference parameters dialog choose option «**Use to calculate displacements**», and in the list below choose data type to be used (2D or 3D seismic data are available). For all data types specify coordinates columns names, the value column (t_0 or depth) and the NAN-value if needed. For 2D seismic, specify a profile number field, and for 3D seismic (if a table contains data of more than one seismic crew) specify the field with seismic crew id number.

The idea boils down to constructing along the fault line a series of cross sections which define the nearest points on the two fault banks. The sections are constructed one step **h** apart (fig. below), the step having been set in the reference property dialog «**Search step**». The software automatically picks a section closest to the input data and computes displacements on the fault banks.



After all parameters have been specified in references and the fault geometry has been constructed, the displacements field can be computed by executing the relevant command.

5.5.8. Faults drawing

Object of this type has several graphic layers whose visualization can be regulated from the toolbar by pressing one of the following: 

- Choose palette to fill the tectonic displacements field;
- Displacements field filling On/Off;
- Triangulation lines On/Off;
- Fault blocks On/Off;
- Edited layer lines On/Off;
- Node signatures On/Off;
- End-point and internal points of the edited layer lines On/Off;
- De-selection of marked nodes.

Other Faults-drawing functions are set up in the object properties dialog (categories «**Edit layer's lines drawing**» and «**Faults drawing**»).

5.6. Grid and source data editing

GST has a set of convenient tools that help the user to edit both a result-grid and the data used for its computation. It makes it possible:

- To edit grid's isolines;
- To input additional points (described in section **Creating a points table** in Chapter **Table-object**);
- To create smooth inserts;
- To edit source data (points and cover lines) using the working grid-window.

All grid-editing stages (except the last item that concerns source data editing) are saved in the current project, and the user can always return to the initial variant.

5.6.1. Editing grid's isolines

Editing isolines is the same as entry of additional source lines into build process. To make this, **GST** suggests to perform the following operations:

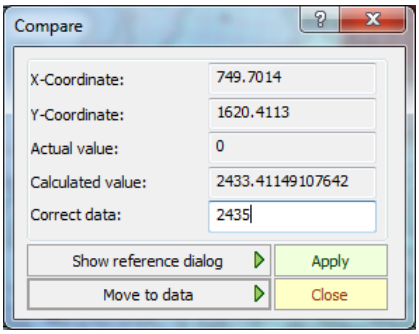
- Press on the toolbar or execute the command "**Grid → Manual Edit → Edit Isolines**". At that, a folder "**Edit** (Grid Name)" is automatically created in a hierarchy tree window, having a "**Cover**" object in it, called, for example, "**Edit 1**". The program assigns the "**Create with editor**" method to the cover; all references contained in the grid being edited are added as insignificant references; at that a cover window is activated.
- In the cover working window opened, the user should draw (using spline-mode) one or several isolines and assign them an attribute needed (isoline value – depth, thickness, etc.).
- To complete cover-build process click .
- Activate a grid's working window and press («**Start Build Grid**»). It shows a dialog where user should input a weight on the current edit. After this, the program will automatically create a cover reference inside of the grid, specify all parameters, and recalculate a grid taking into account the edit performed.

Users can create a lot of edits using the same sequence of operations. However, user ought to keep in mind that the edit entered should not be in conflict with the initial data applied.

5.6.2. Editing source data

Following the grid-build process the initial-data errors could be discovered. **GST** can easily edit such information in the grid working window.

Calling of a compare-dialog. Holding **Ctrl** key pressed right-click on the mouse at the point of interest (well, seismic pick or contour). You will get the following dialog on your screen:



The dialog shows an actual value set up in the corresponding table row, or a cover-line attribute, as well as a calculated grid value. For tabular data the edit-window "**Correct data**" is accessible, where the user can edit the value of a mapping parameter. Buttons "**Show reference dialog**" and "**Move to data**" activate a property dialog of the corresponding reference or a working window of the table or cover assumed. At that, in a table and cover's working windows the edited line (row), respectively, marked and selected.

5.6.3. Creating smooth inserts

In some cases it is necessary to edit a map in some of areas, leaving all the remaining data unchanged. It is in particularly topical when:

- due to complexity of the mapping area it is impossible to establish common smoothing parameters (weights, stabilizers);
- when the user must add or remove some data in a small area of the map considered;
- Irregular data distribution within the mapping area frequently leads to necessity to make an estimate grid denser. To solve such a task, **GST** is capable of making smooth inserts in a grid.

To make an insert, press  on the toolbar (or execute command "**Grid → Manual Edit → Make Insert"**), and, holding the left button of the mouse, delineate a paste-in rectangle in the grid's working window. Two "**Grid**" objects called "**Insert** (grid name**)_1**" and "**Summary** (grid name**)**" will automatically appear in a hierarchy window.

Let us name the grid we are editing Basic Grid. All references contained in this basic grid are automatically added (with the same parameters) to the "Insert"-grid and build method sets as "**Insert**". Besides, one more reference is added, a reference to the basic-grid.

In the insert-grid activate Grid-Builder dialog to set up build parameters. Moreover, user can change, delete and add new reference parameters. In other words, insert realizes all capabilities of the "**Grid-Builder**"; however, final grid values and derivatives at an insert/basic-grid boundary should be equal, and the insert grid step is to be even (less or equal) to a basic-grid step.

A summary grid is built as based on the basic grid and insert(s) made. Insert-grid and basic-grid references are automatically added into this summary grid. With the user-specified grid step **GST** gets grid values: if the point under consideration falls in one of inserts, then its value is taken from this insert; otherwise, it is obtained from the basic grid. The total-grid step is specified in the Grid-Builder on the corresponding dialog page.

Users can create any number of inserts that are partially or fully overlapping each other (priority is given to the last one); meanwhile, the program automatically adds needed references and traces object relationships.

Note: In this version the "**Global Equations**" and "**Local Equations**" pages content is not automatically copied from the basic grid to inserts.

User can activate the insert working window by executing the menu command "**Grid → Manual Edit → Move to Insert**" and then left-clicking the mouse at the insert rectangle needed.

5.6.4. Visualizing cross-sections along seismic lines

Version **5.0 GST** featured the capacity of easy visualizing cross-sections along seismic lines directly from the grid working window. The cross-section can reflect the grid itself, all reference grids, as well as source data from seismic lines. This new option is very handy in visually assessing the results of one's gridding based on seismic data, especially with the use of direction derivatives. To view a cross-section, the user does as follows:

- Add to grid a reference to table containing seismic data (when necessary);
- In dialog of reference properties, in the category «**View**», enable flag «**Use to view cross-section**», and in the drop-down list specify field that contains the line number (or name);
- In the list below user can specify other table fields which contain data user wants to be visualized on the cross-section.

After all setups have been done, the user points the cursor in the grid working window on the seismic line and in the context menu (right-click) choose the option «**View cross-section**». Users can also enter the cross-section visualization mode by clicking  on the toolbar. In this case the viewing is exercised by left-clicking the mouse on the seismic line. The line is viewed in the same window in which user works with the «**Cross-section**» object.

6. Statistics object

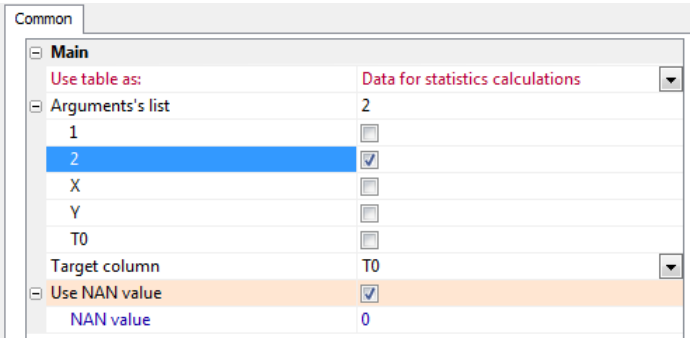
An object of this type is designed to perform calculations and under-project saving of the following table-data statistical characteristics:

- Vectors of table-column average values;
- Vectors of table-column variances;
- Correlation matrices for table columns;
- Linear-regression coefficients.

6.1. Statistics generation

There is one method for statistics generation; it is based on "**Table**"- data references. To calculate statistics, you need to add and initialize the references to the tables needed and to set the necessary parameters in the build-settings dialog. Press the button  or execute the menu command "**Object → Start Build**".

Reference to a table. In the table reference property dialog you need to assign names to argument columns and a name to the column with a target property. There may be one or several argument columns.



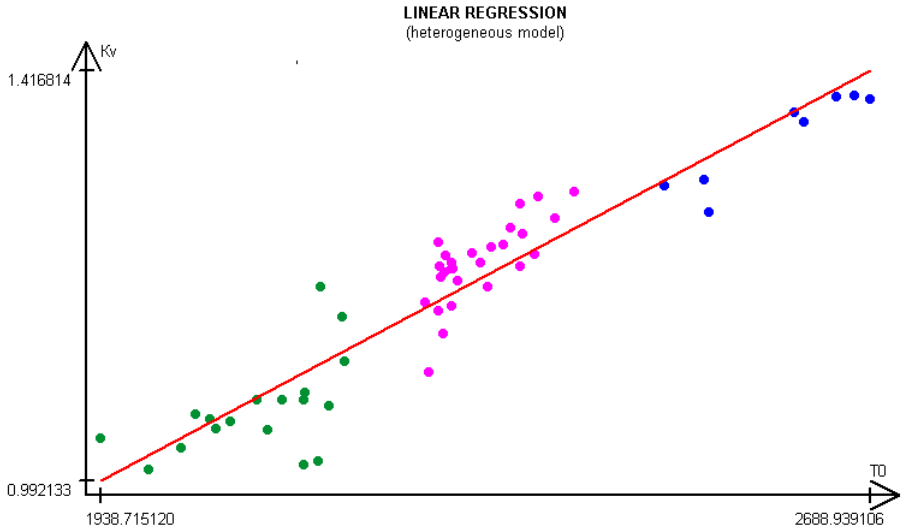
To add a new argument column, it is necessary to check it in the list of arguments. To have table data get involved in statistics calculation, you should select the option "**Use a table as"** and then choose **"Data for statistics calculations",** otherwise columns' data will be used only as a graphical layer. At that moment the control elements become accessible, which help to specify a color, form and size of data-display signs. The option "**Use NAN-value**" provides filtering of data with NAN-values and their disuse in further calculations.

Statistics are generated by data from one or several tables. Tables' merging can be sequential or parallel. At sequential tables merging it is necessary that the tables have the same number of columns. The user sets up argument columns and a target-property column in a property dialog of each reference (a number of argument columns for all references should be the same). At the parallel merging the tables being combined should have the same number of data rows with a target-property-name column be set only for one reference.

A table-merging mode, regression model type (homogeneous/heterogeneous) is selected in the build settings dialog.

6.2. Statistics imaging

When calculations are over, the results (correlation matrix, average values & variances' vectors, regression coefficients), as well as a statistical-dependence graphic display appear at the user's screen.



Maximal divergence from model = 0.101314
Residual dispersion = 0.001304
Dispersion = 0.036104

Correlation matrix:

1.000000	0.940072
0.940072	1.000000

Lineal regression coefficients:
(0.000566, -0.105322)

Multiple correlation coefficient:
0.940072

Vector of the average values:
(1.184870, 2279.197283)

Number of the points:
53

Vector of the dispersions:
(0.011212, 30920.394851)

Each point of a graph corresponds to one table row. A vertical axis is for target-property values, while a horizontal axis is for values of an argument column. If there are several arguments, one should place the complex-parameter values along X-axis:

$$\xi_i = \sum_{k=1}^{m} b_k v_i^k \,,\; b_k = \frac{a_k m}{\sqrt{a_1^2 + \ldots + a_m^2}},$$

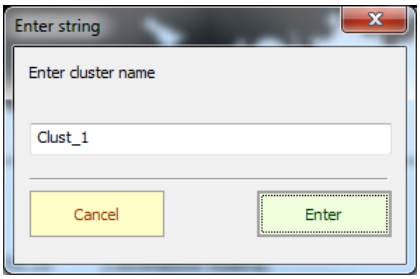
where m – a number of argument columns; v_i^k - argument-column values; a_k - coefficients of linear regression.

Visualization parameters of the data used in calculations are set at the object-properties dialog (category "**Statistics visualization**").

6.3. Cluster generation

The points in a display obtained in the result of regression-model calculations can be grouped in separate clusters and then used for creation of new tables. To activate the cluster-choice mode you need to press the button in the toolbar, or execute the menu command "**Statistics → Cluster Mode**". Then, holding the left button pressed, generate one or several rectangular areas (choose the points for a cluster). To generate a cluster press the button in the toolbar or

execute the menu command "**Statistics → Create Cluster**". In a dialog popped up, enter a name of the cluster delineated.



The points that have fallen in the area chosen will get painted with other color. It is possible to cancel cluster-generation using the "**Undo**"- button in the toolbar. The color of the cluster formed can be changed at the object-property dialog's category "**Clusters**". Cluster generation is performed under two modes: **exclusive** and **shared**. Under the **exclusive** mode, set up by default, the points included into one cluster, cannot be included to the other one. Under the **shared** mode the points can be present in several different clusters.

To generate a cluster that includes all the points fallen outside any of the existing clusters, press the button in the toolbar or execute the menu command "**Statistics → Add the rest points to Cluster**".

6.4. Statistics saving

Statistical calculation results can be saved in files of the ***.dat** type. To do this, select the item "**Save Statistics To File…"** in the menu **"Statistics**", and then highlight a file name. At saving the program creates several *.dat files:

- *_Disp.dat file – contains a max deviation from a regression model, remaining variance and root mean square deviation from the model created;
- *_Koef.dat file – contains regression-model coefficients;
- *_Matr.dat file – contains a correlation matrix of columns;
- *_Stat.dat file – contains average values and columns' variances.

Additionally, clusters (if they are generated) can be saved in separate files (of the *_ClusterN.dat file template). At saving, instead of a star use a name of the file chosen.

6.5. Loading statistics parameters to table

The results obtained through statistical processing can be loaded to a "Table" object and then used for further calculations. Here, it is necessary to perform the following steps:

- To create an empty "**Table**" object;
- To select the build method as "Load Statistics parameters";
- To add a Statistics link to a table;
- To set up needed build parameters in a link-property dialog.

If the user wants to enter regression coefficients, a correlation matrix or some other statistical characteristics to a table, **"Statistics parameters"** should be selected in the "**Load to table**" box and a necessary feature should be selected in a drop-down list box. "Cluster data" should be selected in order to insert the cluster data to a table and then a name of a cluster should be chosen in a cluster list.

7. Cross Section object

An object of this type is used for visualization of geological cross-sections and their editing. In fact, cross-section has every property of a conventional linear cover but unlike the latter it can be displayed in two windows. The first – standard **GST** window – displays cross-section lines in plan view and the second independent window (can be placed upon the second screen) is for visualizing traces of geological surfaces along the profile line.

7.1. Building a profile

First, add to your project a new object of "**Cross-section"** type using the command "**Add object -> Cross-section"** in the tree-view context menu or click the button on the toolbar. The program will add an empty object in the hierarchy tree and set automatically "**Create in editor**" method. The user draws the profile lines manually, or they can be uploaded from the file as a conventional cover.

Next, all the necessary references are added: grids (their traces will be displayed on the cross-section), wells and seismic (to draw the profile line), etc. References to these specified objects are listed as "insignificant", i.e., as graphic layer.

7.1.1. Create profile lines in the editor

The profile line is created in the same way as any other polyline – in "**Polyline**" mode. "**Use Control Points**" mode is enabled when all mapped wells must be captured precisely (i.e., included) by profile line. Editing functions applied to the conventional cover (except those specific) are available to profiles – cutting, condensation-optimization, displacement, etc.

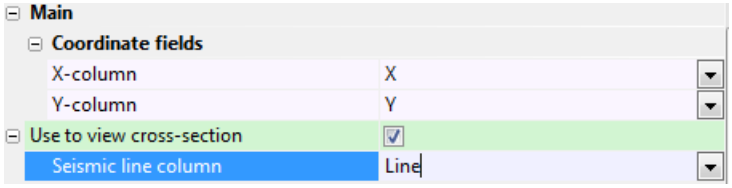
The user can create multiple profiles but only one can be visualized in the cross-section window. For this purpose, select this line and click button on the toolbar. The cross-section will appear in a separate visualization window.

"**View section mode**" is convenient for viewing a large number of profiles on a PC with two monitors. Left-click the line in the plan, cross-section will appear on the second screen.

7.1.2. Viewing cross-sections along the seismic profiles

Viewing cross-sections along seismic profiles can be performed in the **Cross-section** or **Grid** window.

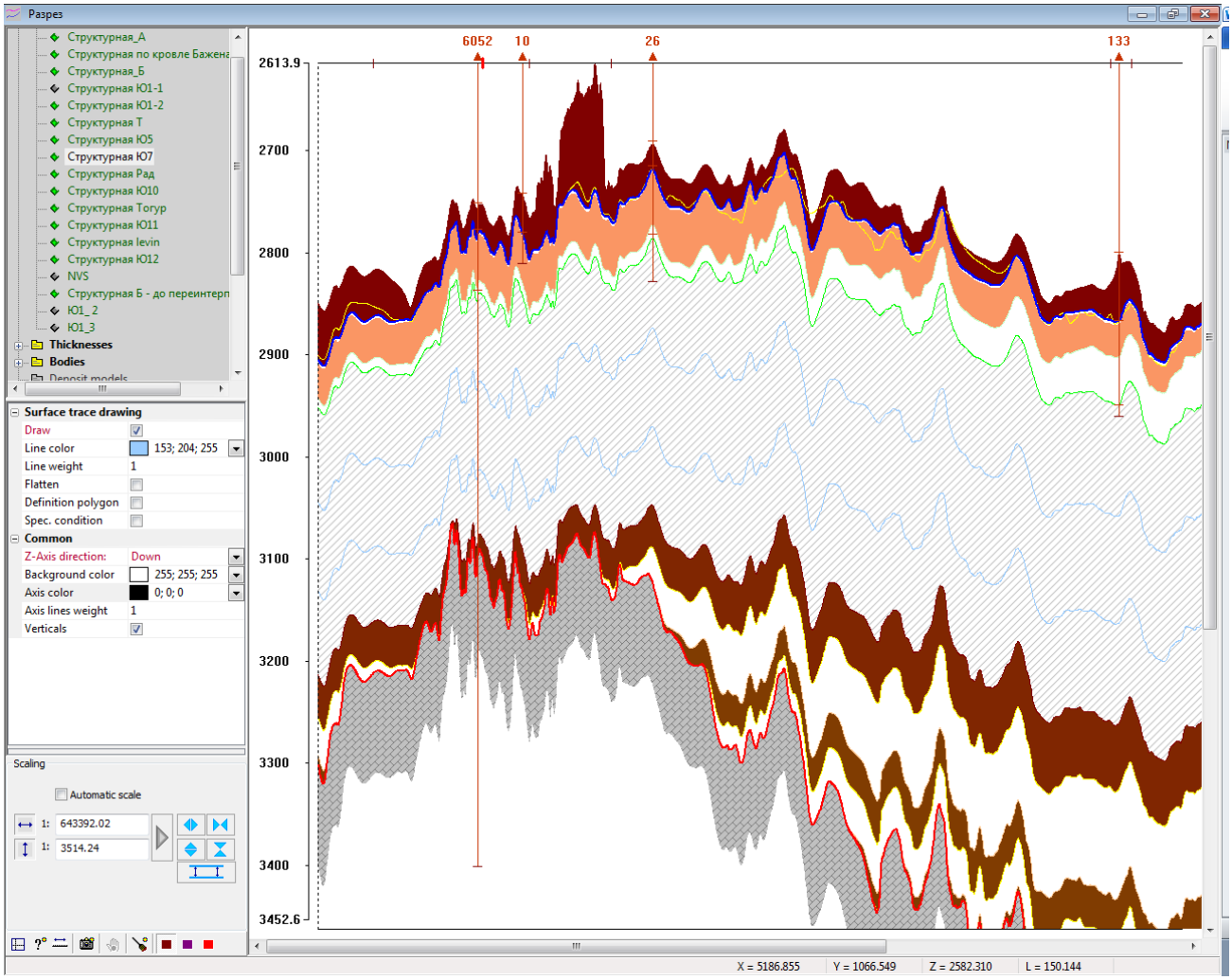
- Add a reference to the table that contains seismic data.
- Set the field coordinates and the field with profile # in the reference parameter dialog (see Fig. below).
- Go to "**View section mode**" and left-click on the desired profile.



When viewing the cross-section along the seismic profile, surface traces, t_0 values or depth in the shot points are displayed. More information is provided in the next paragraph.

7.2. Visualization of cross-section

Cross-section visualization module is presented as a separate pop-up window called by a relevant user command, see Figure below.



The visualization window consists of a cross-section drawing area and the control panel located on the left side of the window. The Panel, in turn, contains four main elements:

- Cross-section content tree (at the top), where all the data used are stored;
- List of drawing properties (in the middle), to set the drawing parameters for cross-section elements;
- Scale control panel;
- Toolbar (at the bottom) with cursor mode buttons, etc.

Button  on the toolbar provides forced redrawing of the cross-section content and buttons  set the detail level (low, medium, high). Higher detail level increases drawing time but helps to eliminate some inaccuracies in filling bodies and the deposit model.

7.2.1. The Cross-section content tree

The cross-section tree contains all the uploaded reference data in a structured form. If the object-cross-section does not contain references, the cross-section tree is empty. The root item contains several folders that sort the cross-section content by its meaning:

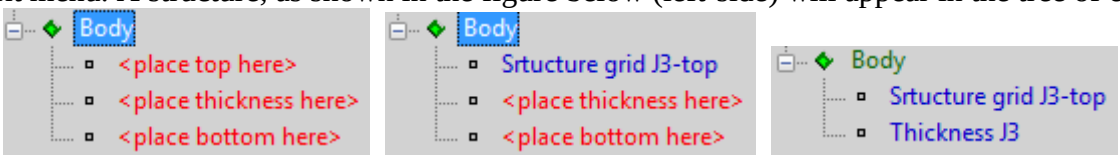
- **Surfaces**. This folder contains references to the grids (structural, isochronal, contact surfaces, etc.) that appear in the drawing area as surface traces;
- **Thickness**. This folder contains references to thickness grids. These grids are not displayed on the cross-section but contribute to the construction of "**bodies**" (see below), as well as to conditions of displaying surfaces and bodies.
- **Other grids**. This folder contains references to grids that are neither surface nor thickness, in their meaning. These grids are not displayed on the cross-section but contribute to conditions of displaying surfaces and bodies.
- **Bodies**. "Bodies" are constructed directly in the cross-section tree from surface and thickness grids and are displayed as polygons in the drawing area.
- **Point data**. This folder contains references to the tables with data that are displayed in the drawing area of a cross-section.
- **Boundaries**. This folder contains references to lines and polygons. Points of intersection of these objects with a profile line are displayed in the cross-section window.
- **Deposit models**. This folder contains references to objects of "**Deposit model**" type that are displayed as oil and gas bodies, as well as contact lines.

Sorting grids. When a new grid reference is added, it is placed by default in the "**Surface**" folder and is displayed as a "trace". If the grid bears another physical meaning, it should be moved to "**Thickness**" or "**Other grids**" folder, to avoid inconvenient displaying: **GST** will remember the folder assignment for each grid, i.e., its membership.

Moving elements within the tree. To move an element within the tree, you must "capture" it with left mouse button, hold down the button and move the element to a new place. **GST** prohibits mixed content in the folder: i.e., a grid or a body cannot be moved into the "**Point data**" folder. Grids can be moved between "**Surface**", "**Thickness**" and "**Other grids**" folders, as well as inside these folders. If the tree is "long" enough, grids can be moved between folders using the context (right-click) menu commands: "**Move to 'Thickness'**", "**Move to 'Other grids'**", "**Move to 'Surfaces'**". Position in the folder reflects the sequence of drawing: elements that open the list are displayed above, i.e., the last.

Creating bodies. The updated version of the cross-section has a new option – displaying the body of the layer or reservoir (etc.) as a filled polygon. The body is a complex tree element constructed from one or two grids.

To add a new "body", right-click on the "**Bodies**" folder and select "**Add**" command in the context menu. A structure, as shown in the figure below (left-side) will appear in the tree of elements.



The body can be specified by top and bottom grids, either top or bottom and the total thickness. Initially, three "holders" are available for all the elements of the body. The user "moves" the corresponding grids into these "holders" using the mouse (see figures 2 and 3 above). Grids from the "**Surface**" folder can be placed only into the top and bottom "holders" and from the "**Thickness**" – into the thickness "holder". A body can be created from one surface (top or bottom) and constant

thickness. For this purpose, right-click the thickness holder and select the menu command "**Offset top or bottom**". The thickness parameter of the body is set in the list of display properties below.

If moving grids with a mouse is uncomfortable, due to the complexity of the tree, the top, bottom or thickness can be set using "**Set grid**" command in the context menu.

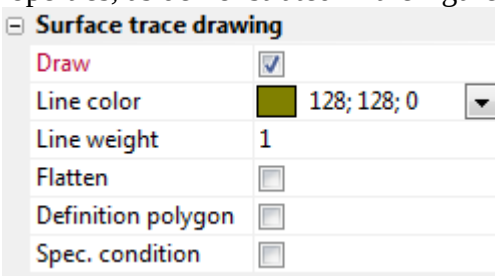
When two components of the body element are set, the third holder is closed automatically. You can release the grid from the body using the command "**Release grid**" the context menu (right-click the mouse) and replace it by another grid, or unpin all grids. You can quickly "reverse" the body with the command "**Swap top and bottom**".

Visible and invisible elements of the tree. Elements of a tree that can be displayed in drawing area are marked by diamond-shaped icons (like the "body", see Fig. above). Green color means that the element is "on" for drawing, and gray – that the element is temporarily "off". You can enable or disable drawing the element by double-clicking it with the left mouse button or via "**Draw**" option in the list of display properties. You can enable or disable drawing all elements in a folder with commands "**Draw all**" and "**Don't draw all**". Elements of the tree, which are not visible directly, are marked with small gray icons (see Fig. above).

7.2.2. List of display properties for the cross-section elements

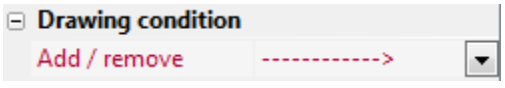
The list of display properties associated with the selected element of the tree is placed below the cross-section content tree. By default, such general properties, as the color of background and the coordinate axes, as well as the direction of the vertical axis are available to the user. By using "**Verticals**" option, the user can enable or disable drawing the intermediate points of the cross-section. When any element is selected in the tree, the relevant display options appear in the list of properties.

Displaying the surface trace. When any surface is selected in the tree, the relevant category with options appears in the list of properties, as demonstrated in the figure below.

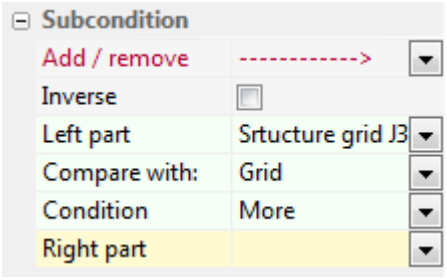


The user can enable or disable drawing the surface, set the colour and line thickness, and align the cross-section on this surface. The "**Definition polygon**" option defines scoping (under the relevant reference properties) and restricts grid display to the polygon boundaries. "**Spec. condition**" option allows specifying a more complex display condition for the surface trace.

Special display conditions. If the "**Spec. condition**" option is enabled, an additional category appears in the list of properties (see fig. below).

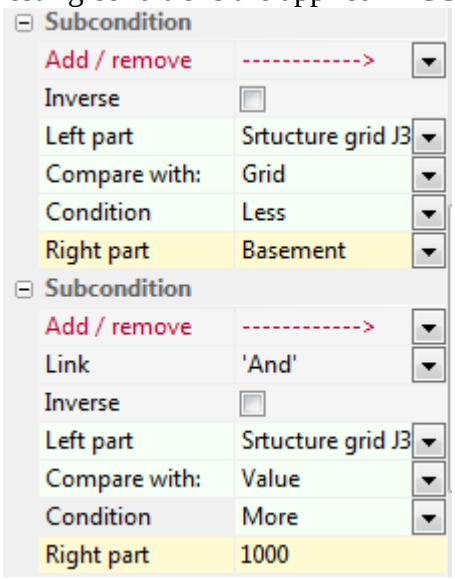


Initially, list of conditions is empty: the user must add a condition by selecting the relevant command from the drop-down list. A category containing parameters of the condition will appear (see fig. below).



Logically, condition consists of the left-hand side, which is always a grid, the right-hand side (a grid or a number), and the type of condition (greater than, less than and approximately equal to). The type in the right-hand side is determined by the option "**Compare with:**", where the user selects a grid or a number from the list. The "**Inverse**" option is equivalent to the logical "Not".

The condition can be complex. An example of a complex condition (the fig. below) is drawing the trace of the structured surface "*Structure grid J3*" – above the surface "*Basement*" (outside the stratigraphic pinch-out) and below the 1,000 m stratigraphic mark. The logical link between the first and second conditions is set by "**Link**" option, which can take values "*And*", "*Or*". Each condition in the list can also consist of several encapsulated sub-conditions, which is equivalent to the expression in parentheses. To add a sub-condition, a relevant command from the "**Add/delete condition**" is executed. No special restrictions on nesting conditions are applied in **GST**.



Displaying tree elements under the special condition follows a simple rule: if the condition is "*true*" at a given point, the element is displayed and if not – it is invisible.

Displaying the body. The following options are available in the list of display properties for the body:

- Enable-disable drawing the body.
- Parameters of body filling are set via the standard 'fill polygon' dialog.
- Body thickness option is available only for bodies with constant thickness.
- Special display condition is set by the same rules as for the surfaces.

Displaying the deposit model. The cross-section visualization window can display deposit models built in **GST** Versions 6.7 or later. Otherwise, "**Deposit model**" must be rebuilt for correct visualization. The visualization window displays:

- **Saturated oil-gas parts of the deposit**. The user can set parameters for filling cross-sections in oil- and gas-bearing areas.

- **Contact boundaries, i**.e., traces of the surface contacts (gas-oil, gas-water, water-oil). The user can specify the color, thickness and depiction style.

Displaying point data. Point data can be displayed in several modes:

- **Coordinate marks**. Points are projected onto the cross-section line and displayed as a mark in the upper part of the drawing area.
- **Wells**. Points are projected onto the cross-section line and displayed as a vertical well.
- **Seismic profile trace**. Data are displayed as a trace of an isochronous or structural surface with values specified in seismic stations.

The "**Seismic profile trace"** mode is set automatically when viewing a cross-section along the seismic profile. To set other modes the user must change "**Show as**:" option. Depending on the display properties mode selected, various options are available, as follows:

Coordinate marks: "**Draw -not draw**", mark color and tag label field settings, as well as the target distance ("**Delta**") for projecting points onto the profile line. "**Delta"** is specified in the current units of measurement of the object coordinates.

Wells: in addition to the above, available options include line weight option (showing the well bore), as well as select well bottom field and depth markers. If a field with the well bottom mark is not selected or absent, the wellbore is displayed vertically across the entire drawing area, otherwise it is limited to a well bottom. "**Depth markers**" category: table fields containing stratigraphic markers (to be displayed on the wellbore as horizontal strokes). "**Marker color**" category: setting the color for the selected markers. If some information is lacking in the wells table, a NAN-value can be set.

Seismic profile trace. This type of displaying data is set automatically at the software level and cannot be modified by the user. In the "**Depth markers**" category, the table field containing values t_0 or depths at shot points can be selected. In the option "**Use NAN-value**" the user can also set the line break mode at null information nodes.

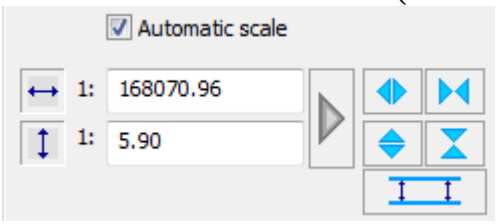
To make a program running faster, newly added table references are not displayed by default. If necessary, the user "turns them on".

Displaying points of intersection with cover lines. In the cross-section visualization window the points of intersection with cover lines can be displayed. The points of intersection are displayed in two ways:

- **Coordinate marks** – small strokes in the top drawing area. The user can set the color and weight of the coordinate tag.
- **Vertical lines** with the specified color, weight and depiction style. By default, lines are displayed throughout the cross-section but the drawing interval (e.g. between two structural surfaces) can be set by "**Spec. condition**" option. The rules specifying conditions of displaying lines are similar to those for the surface trace.

7.2.3. Scale control panel for displaying the cross-section

Scale control elements for displaying the cross-section are demonstrated in the figure below. The selected default scale ensures that the cross-section fits the screen (automatic zoom option is enabled).



The user can set any other scale in the input boxes below and press the activated right-side button after that. The vertical and horizontal scale can be modified with help of a group of buttons on the right side of the panel (150% per one hit) and the button at the bottom sets the vertical scale in a way that the cross-section fits the screen vertically.

The scale can be changed also by a standard **GST** zooming mechanism (Shift + left mouse button). Press right mouse button + Shift to exit the zoom.

7.2.4. Operation modes for cross-section

Several cursor modes are implemented for cross-section. They can be switched by using commands (click right mouse button on the drawing area) from the shortcut menu or duplicate menu buttons at the bottom of the control panel.

- The **Basic mode** is enabled when all others are disabled. In this mode, the status bar at the bottom displays the cursor coordinates: coordinates in plain view, the distance from the profile starting point, and the depth.
- "**Cursor marker**" – the mouse cursor is at the point of intersection of vertical and horizontal lines, which can be used to analyze the relative position of elements in the cross-section.
- "**Prompt**" displays information about an element of the cross-section in the pop-up window. For this purpose, left-click the mouse button on the desired element.
- "**Distance measurer**" allows measuring the distance between two points of the cross-section. The starting point is fixed by pressing the left mouse button. The current distance is displayed in the status bar by five variables: DL – along the profile length, DX – along the X-axis, DY – along the Y-axis, DZ – along the depth, 3D – the actual spatial distance.
- "**Edit grid**" – makes possible introducing ghost points to build a grid.

"**Photo-mode**", similar to **GST**, converts the content (or its part) of the visualization window in the **Bitmap**-format and saves it in a *.b**mp** file.

"**Edit grid**" can be applied only to structural surface grids built with the **Map Builder**. To start editing, execute "**Edit grid**" command to a desired grid in the "**Surface**" folder, then 'edit grid' mode turns active – the corresponding button on the control panel is activated. When the edit mode is enabled, spline mode is enabled automatically and the context menu for the drawing area is modified.

In spline mode, the user must draw a correcting line in surface trace, in the same manner as in drawing cover lines. To introduce changes in grid building, perform sequentially "**Add spline**" and "**Apply grid edit**" commands from the context menu. To exit the edit mode without saving changes, release the relevant button on the control panel.

"**Move to GST layer**" is performed by the same name item in the pop-up menu by pressing the right mouse button; a layer (grid, well, editing) closest to the cursor is selected and activated in the **GST** window.

8. Image object

8.1. Loading and display of graphics files

In a project each graphics file is represented by an independent "**Image**" object, and displayed in an individual working window. To load a graphic image you need to execute the following operations:

- To create a "**Image**" object using the button  in the toolbar or execute the command "**Add Object → Image**".
- The object's working window being opened, execute the menu command "**Image → Load Image From File**".

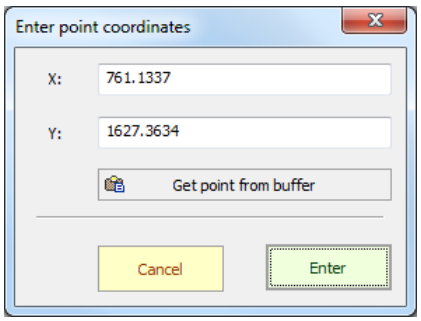
When a picture is loaded, the corresponding object can be copied in a tree as any other object. At project saving, the graphics is saved in the internal **GST** format, in a project directory. To download a picture into an external file under one of the supported formats, use the command "**Image → Save Image To File**".

An object's visualization can be scaled using the same methods as applied at other objects' imaging. While a mouse cursor moves across the object-drawing area, you can see "pixel" coordinates of the picture in a main window's status line. To use a picture as a graphic layer, you need to tie pixel coordinates to the real coordinates.

8.2. Coordinates referencing

At the main object-property dialog's page the user can select one of two modes of coordinate referencing (tie-in): by two or three points. Two-point tying stipulates coordinate-system rotation with a certain angle and uniform scaling by both axes. Three-point tying stipulates some rotation and axes' nonuniform scaling.

Select one of these modes and press the button  in the toolbar (coordinate referencing mode). With this mode enabled, click the left button of a mouse at a point in the picture assumed, which physical coordinates are known. The following dialog will pop up:



Enter physical coordinates of the indicated point to the dialog's fields. By default, coordinates of the point are getting from the **clipboard** (see above). The coordinate entry confirmed, click the second and, if necessary, third time – it makes the button  accessible (make a tie-in), and pixel coordinates of a picture will be converted into physical coordinates. The referencing points are fixed points or markers are displayed by cross signs. For an easier search they can be displayed as a circle by executing a command "**Image → Highlight markers**".

The user can change the physical coordinates of one of the tie-up points by executing a contextual menu command (press the right button of a mouse at the closest marker) “Change the physical marker coordinates” and setting new coordinate values in a dialogue box (see above). In this case the program will switch to coordinates tie-up mode.

In order to change the screen (pixel) marker coordinates the user needs to turn on the coordinates tie-up mode, then drag the mouse cursor to the marker and move it to a needed place.

After the physical and pixel marker coordinates modifying the linkage should be performed by pressing the button  at the control panel. To reset tying (to get back to pixel coordinates), use the button  .

Saving point coordinates to clip-board. If a project contains fixed points coordinates (wells, seismic events picked, etc.), then coordinate tying can be performed as follows:

1. Switch the coordinate-mode for a picture assumed;
2. To any of the project’s covers add a reference to the table that contains fixed points (these points be displayed in a cover’s work box).
3. Getting a mouse-cursor (in the cover’s working box) at the first control point and pressing the right button of a mouse, execute the command “**Point to buffer**”.
4. Switch to a image work window, and by the left button click at a point that corresponds to a fixed point (its physical coordinates will appear in a dialog);
5. Repeat Steps 2-4 two or three times and press the button  .

When coordinate tie-in is done, image can be used as a graphical layer (for example, for contour digitizing). Links to a picture can contain “**Grid**”, “**Cover**”, “**Section**” and **“Printer layout”**.

9. Calculator

The **Calculator** is a program module that performs various operations (mathematical, logical, etc.) with **numbers** and objects, such as "**Table**", "**Cover**", and "**Grid**". The calculation output is expressed as an object of one of the above types.

The Calculator represents an integrated program module, i.e., objects generated with its help are included in the chaining technology. Essentially, the Calculator brings programming elements to **GST**, since it allows specifying operations between the Project objects prior to building these objects.

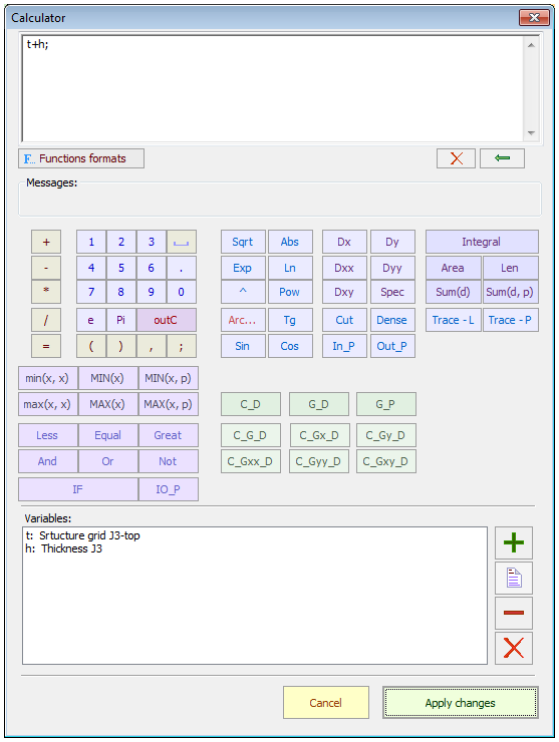
9.1. Calculator rules

All objects referring to Table, Cover and Grid types have build-method called "**Create with Calculator**". By choosing this method we assume that an object is created on the basis of other objects (through reference links) using mathematical and logical operations. Thus, the user must perform the following steps:

- Specify the build method as "**Create with Calculator**";
- Add references to the objects used at build process;
- Set the calculator's command (one or more).

9.1.1. Interface elements

The figure below presents the main dialog for **Calculator** command input. It consists of three main elements: a command input window, numeric and function keys that allow faster and more convenient way to type a command, and a list of variables used for calculations.



Entering the command entry into the display can be performed both by using the keyboard, and with the function buttons. All Calculator buttons have pop-up tips, which appear by holding the mouse

cursor over the button for about two seconds. You can also see rules of writing functions by clicking "**Functions format**" buttons.

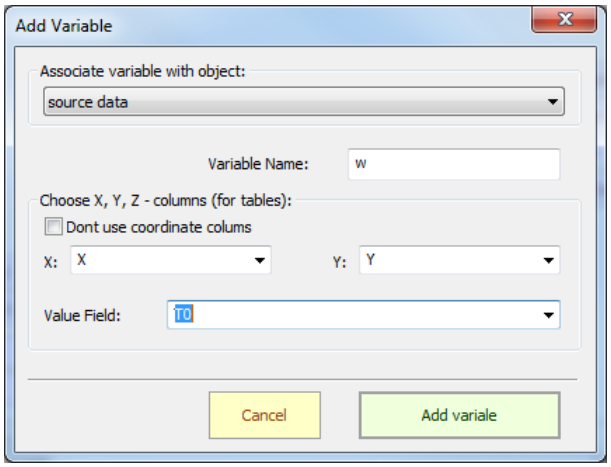
9.1.2. Declaration of variables

The Calculator operates with variables of the following types:

- **Grid** – i.e., grid function. Variables of this type are initialized from **Grid** objects;
- **Lines** – variables of this type represent, as a rule, the map of isolines characterized by some numerical attribute. They are initialized from **Cover** objects;
- **Polygons** are initialized from a polygonal cover only;
- **Data** – variables of this type represent a numerical column of the table (vector), or three columns (two are for coordinates and the third gives a parameter value in a certain point);
- **Number** – represents a table column with size 1.

Variable names are assigned in Latin code register with no spaces, periods, commas, or other characters. Both uppercase and lowercase characters can be used (case-sensitive); therefore, variables «**a**» and «**A**» are considered as different variables. Variable names can consist of multiple characters and contain numbers and underscores but these must not be placed at the beginning of the name. The name must not match the names of standard Calculator functions. *Valid variable names*: **a, B, Cx, T_1, Var**, etc. *Invalid* - **ж1**, **1_B**, **Var 1**, **_xx**, **#66G, Abs, min, IF**, etc.

Declaring new variables is performed directly in the **Calculator** command line window: "*c=a+b;*". The type of the new variable «*c*» is determined by the type of operation result "*a+b*"; the variables a and b are declared earlier. The entry "*a=a+b;*" means that the result of addition is placed back into the variable a, and the latter can subsequently change both its value and the type.



The primary declaration of variables, i.e., initialization is made in the list at the bottom of the **Calculator** dialog box. Buttons **Add** (+) and **Delete** (-) variables and **Clear** list are on the right side of the list. Initially, the list is empty. When adding a new variable, the dialog box appears, as shown above. The user selects an object to be associated with the added variable from the drop-down list. This list is generated from references contained in the calculated object. The variable name is set by the above rules. If an object of the "**Table**" type is selected from the drop-down list, additional parameters should be specified: the name of the column containing the vector of values and names of coordinate columns, if necessary.

By clicking the "**Add variable**" button, we place the initialized variable in the list. The user can edit parameters of any variable in the list by selecting it and clicking the corresponding button on the right.

9.1.3. Rules for writing commands in Calculator

The **Calculator command** consists of one or more mathematical **expressions**. Each expression consists of one or more **operations** or **functions** and produces a final output of a type used in the Calculator (Grid, Number, Polygon, etc.). Operations and functions also produce the result of a standard type and are discussed in the next section. Each mathematical expression should end with ";" (semicolon). The Calculator command output is expressed as the result of calculations of the last expression. Below is an example of a correctly specified command:

p = min(Abs(G1), G2 + 220); Integral(p + max(G1, 2745.5), P_1);

In this case, **G1, G2** and **P_1** are initialized variables associated with **GST** objects. **G1, G2** are of Grid type and **P_1** is a polygon. This command consists of two complex expressions: the result of the first is placed in the variable "*p*", which is used in the second expression. **Abs, min, max**, and **Integral** – are the standard Calculator functions. The calculation output is the number returned by **Integral** function.

As demonstrated by the example, function arguments may be represented by numeric values, variables, functions, and expressions. In terms of running sequence, the rules for writing expressions in Calculator are consistent with the general rules for writing algebraic expressions. The program applies no constraints on the number of expressions and nested functions: the equivalent notations for the above-described command are as follows:

Integral(min(Abs(G1), G2 + 220) + max(G1, 2745.5), P_1);

or

a= Abs(G1); p = min(a, G2 + 220); i = p + max(G1, 2745.5); Integral(i, P_1);

Here is an example of meaningless command, where the last expression which "cancels" all calculations:

p = min(Abs(G1), G2 + 220); Integral(p + max(G1, 2745.5), P_1); 300;

The command may be entered in the display directly from the keyboard or by using functional buttons. When pressing the corresponding button, the function is added together with parentheses and parameter prototypes, which are replaced by the desired correct variables entered by the user. For example, after clicking the "**Sin**" button, **Sin(X)** is displayed, where **X** can be substituted by a variable of any type (number, column, or grid). If the function parameters accept only one type of variables, the prototypes are marked by other symbols: G – Grid, P – polygon, D – tabular data, N – number. The name of this variable is entered by double-click on the respective row in the list of variables.

9.1.4. Numbers and constants

Entering numbers in the Calculator command is defined by the following format rules:

- Decimal sign is denoted only by dot: ".";
- Large or small numbers can be recorded by means of the exponential power (e.g., 1e6, 5e+7, 1.478e-9);

Primarily, the following constants are initialized the in Calculator:

- **Pi** = 3.1415926 (i.e., π number);
- **e** = 2.718282 (i.e., base exponent);
- **outC** = 0 – null character indicating lack of information.

In other words, the allowed entry is: L=2*Pi*R, where specific declaration of Pi variable is not requested.

OutC constant determines which data values should be perceived by the program as a null value (NAN). The Calculator includes a rule that any operation with numbers equal to outC provides a result equal to outC. Initially, outC=0 but if we intend using zero as a number, the outC constant should be overridden by any other number. For example: outC =-1e38; c=a+b; null character can be repeatedly overridden in the same command:

outC=-1e38; I=Great(a, b); a=a*I; outC=0; a+b;

Here the values for data vectors a and b are summarized and the summation takes place if only a> b, otherwise the result is 0.

9.2. Calculator functions

9.2.1. Algebraic operations

The **Calculator** supports standard binary algebraic operations: summation-subtraction, multiplication- division, and exponentiation. These operations are recorded in a standard way: c=a+b; c=a-b; c=a*b; c=a/b; c=a^b; a more complex record is also possible, for example: (a+a1+a2+(b-c)/(d+v^3))^s; all these actions are performed pairwise, according to the general mathematical rules. The arguments of binary operations can be either the same type or of different types.

Argument type №1	Argument type №2	Operation result
Number	Number	Number
Number	Grid	Grid
Number	Data	Data
Number	Lines / Polygon	Lines / Polygon
Data	Grid	Data
Data	Data	Data
Data	Lines / Polygon	Data
Polygon	Polygon	Polygon
Grid	Grid	Grid

The above table demonstrates the type of calculation outputs for the corresponding types of the first and the second arguments. Obviously, relocation of arguments does not change the output type. So:

- If any argument is a *number*, the result will have the type of the other argument, i.e., values at the grid points are multiplied, divided, etc. by a predetermined value; the table column is added to a number or is exponentiated. If the other argument is a line or a polygon, the actions are performed with numeric attributes, which denote the isoline value.
- When both arguments are *data*, operations are applied to the values in the n-th row of the first and second argument. Thus, the data dimensionality (number of rows) must be equal in both arguments; otherwise the **Calculator** would generate an error. In data-data operations, arguments can either contain or not contain the coordinate columns.
- In binary operations, such as *line-line, line-data, polygon-data*, lines and polygons are converted to point data and the operation follows the rules, if both arguments are data. However, it is unlikely that two different linear covers would have an equal number of points; therefore, such operations are possible just theoretically.

- When one argument is a *grid* and the other refers to the *data*, *line*, or *polygon* type, the calculations will always result in *data*. Values in data points (lines) are taken from the grid and the operation will be performed between these values and values of the column content, or the line attributes. If the point falls outside the grid, the operation result is the **outC** value.
- Operations between the two *grids* also result in *grid* output. Rectangles and grids steps may mismatch; in this case, the resultant grid will take the lower of two grids steps and its rectangle derives from the intersection of argument grid rectangles.
- If both arguments are of *polygon* type, the operations of adding, subtracting, and multiplying have the set-theoretical meaning: adding means merging of polygons, subtracting – cutting from the polygon an area belonging to another polygon, multiplying – selecting the shared area. Exponentiation and division of polygons are meaningless operations.

9.2.2. Logical operations

Logical operations in **Calculator** introduce programming elements, since they can channelize the calculations depending on conditions. For example, the Boolean "if" (**IF** function) enables one or another action on the argument. This function is written by the following rule:

IF (**condition**, *expression 1*, *expression 2*);

The first argument is a number or a function that returns 1 or 0. The second argument is an expression, if the condition is met, and the third argument – the expression, if the condition is not met. Example:

IF(Great(n, 10), a+b, a-b);

This means that if the number **n** is greater than 10, the variables **a** and **b** are added, otherwise they are subtracted.

However, the **IF** function has a strict restriction, as the first argument can be expressed by a number only; i.e., it cannot be run "in the loop" for the vector of values. In the Calculator, this operation is performed by other logical (Boolean) functions, which return "boolean" grids or columns that contain values 0 or 1 (false-true).

Less(X, X) means a logical "less than"; returns 1 if the first argument is less than the second, and 0 – if otherwise.

Great(X, X) means a logical "greater than"; returns 1 if the first argument is greater than the second, and 0 – if otherwise.

Equal(X, X) means a logical "equal to"; returns 1 if the first argument is equal to the second, and 0 – if otherwise.

And(X, X) means a logical "and"; returns 1 if the first and the second arguments are equal to 1 (true), and 0 – if otherwise.

Or(X, X) means a logical "or"; returns 1 if either argument is equal to 1 (true), and 0 – if otherwise.

The expression "the value of X lies between A and B" is written as follows: **And**(**Great(X, A), Less(X, B)**);

Not(X) means a logical "not". It changes 1 to 0 in the argument and vice versa. The logical "greater than or equal" is written as **Not(Less(A, B));**

IO_P(X, P) means that a given point is inside the polygon. The first argument can be expressed by point data, lines, polygons, or grids. The second argument is unconditionally a polygon. The function output are data: two columns with the coordinates of the argument and the value column (1 – the point is inside the polygon and 0 – outside).

Be aware that by default the number 0 is a NAN-value. To operate Boolean functions correctly, you must override the **outC** constant.

Logical functions also include functions for calculating the minimum and maximum of two arguments.

min(X, X), **max**(X, X) compare the values of the first and second arguments, and the minimum or maximum values are placed in the output.

Binary logic function, same as the algebraic operations, can work with different types of arguments an under the same mixing rules applied to the arguments.

9.2.3. Standard mathematical functions

Algebraic and trigonometric functions of any argument return a value of the same type. *Grid* operations are performed with values in the nodes, *data* operations – with each element of the vector, for *lines* or *polygons* – with numeric attributes.

Sqrt(X) means a square root of the argument. This function works with all types of data and generates an error if the argument is less than 0.

Abs(X) means calculation of the absolute value.

Exp(X) means exponent e^x.

Ln(X) means natural logarithm. In case of zero- and negative-value arguments, it generates an error.

Pow(X, X) – exponentiation of an argument. Mixing of types is allowed by the general rules, as applied to arguments. Instead of the «Pow» function, "^" (X^Y) command can be used.

Sin(X), **Cos**(X), **Tg**(X) – sine, cosine and tangent, respectively. The measurement unit for arguments is radian. If the argument value is given in degrees, it must be written, as follows: **Sin**(Pi/180*X);

ArcSin(X), **ArcCos**(X), **ArcTg**(X) – inverse trigonometric functions. To enter them with the functional buttons, first press «**Arc**», then «**Sin**» or other. These functions provide output also in radians.

9.2.4. Additional functions

For practical purposes, a number of functions are introduced in the **Calculator –** to convert variables from one type to another, edit data, etc.

Data type conversion

C_D(X) – conversion of cover lines (polygon) into point data. Values of line attributes are entered into the value column. The argument can be of *line, polygon*, and *data* type. In the latter case, the function returns a value equal to the argument.

G_D(G) – conversion of the argument-grid into point data. In coordinate columns contain coordinates of the nodes, the value column – grid values.

G_P(G) - conversion of the argument-grid into polygon. Polygon is created upon a grid rectangle.

Trace(G, min, max, step, coef) – tracing of grid, or conversion of grid into lines (function key "**Trace-L"**). The first argument is a grid, *min, max* – values of tracing limits (from-to), step – tracing step, coef – grid condensation coefficient for smooth lines. To trace a single isoline, the limits are set, as follows: min = max.

Trace(G, value, coef, index) – selecting from grid a polygon containing values greater or less than a specified *value* parameter (function key "**Trace-P"**); *coef* – grid condensation coefficient. The *index* parameter takes the values 0 or 1 and defines values greater than or less than the specified for the traced polygon. For example, to select a polygon with positive grid values: p = **Trace**(G, 0, 1, 1);

Taking grid values at given points

C_G_D(X, G) – calculation of point values contained in the argument X over grid G. The first argument can be of *data, line*, and *polygon* type. The function outputs are: data containing two coordinate columns and the column with values taken from the grid.

C_Gx_D(X, G), **C_Gy_D**(X, G), **C_Gxx_D**(X, G), **C_Gyy_D**(X, G), **C_Gxy_D**(X, G) – calculation of derivative values at points contained in the argument X over grid G. The first argument can be of *data, line*, and *polygon* type. Each of the five functions calculates: the first derivatives with respect to x (**Gx**) and y (**Gy**), the second derivatives with respect to x and y (**Gxx, Gyy**), and the second mixed derivative (**Gxy**).

Data truncation functions in a polygon and rectangle

In_P(X, P), **Out_P**(X, P) are data truncation functions polygon (the second argument). The first function **In_P** deletes (clears) data outside the polygon, i.e., retains those inside, whereas the second function keeps data outside the polygon, i.e., deletes those inside. The first argument can be point *data, line, polygon, and grid*. For *data*, the function deletes points inside or outside the polygon; *lines* are cut by the polygon boundary and the respective fragments are deleted. If the first argument is a *polygon*, the functions act as set-theoretic operations: **In_P** – polygons' intersection, **Out_P** – subtraction. If the first argument is a grid, the respective nodes are assigned the **outC** value.

Cut(G, Xmin, Xmax, Ymin, Ymax, index) means cutting a fragment of the grid (G) in rectangular pattern, with coordinates *Xmin, Xmax, Ymin, Ymax*. The *index* parameter takes the values 1 or 0. If *index* = 1, the grid fragment expands to the nearest grid nodes; if *index* = 0, it converges to the nearest nodes.

Cut(G, Nmin, Nmax, Mmin, Mmax) means cutting a fragment of the grid (G) in rectangular pattern defined by the integer numbers of nodes.

Additional functions to work with grids

Dense(G, N) means condensation of grid (G) nodes by (N) times. If $N > 1$, the grid is condensed, if $N < 1$, the grid is attenuated. **Dense** function can be useful for specifying binary operations between grids. Since all binary operations occur as "node to node", such expression as Dense(G2, 2)-Dense(G1, 2) would produce a more accurate result than Dense(G2-G1, 2).

Merge(G1, G2) means merging of two grids along their rectangles. This function is not supported by the function key but can be entered directly into the panel. The resultant grid step is the lower of two; grid values in the areas of intersection are taken as the arithmetic mean, the **outC** value is set outside grid rectangles. This function makes a formal connection of grids, using **Map-Builder** is recommended for more accurate results.

9.2.5. Summation and integration functions

Sum(D), **Sum**(D, P) returns a number equal to the sum of elements in the data column (in the first case), or the sum of elements in the data column for the points contained within the polygon *P*. In the first case, the variable D can lack coordinate columns, whereas in the second they must be present. *Important*: if any value in the data column equals **outC**, these rows are not summed up.

Area(X) – returns the number equal to the argument area. The argument can be a *polygon* or a *grid;* in the latter case the area of a grid rectangle is calculated.

Len(X) is a function of length. If the argument is a *line* or *polygon*, it returns the total length of lines; if a grid – its rectangle perimeter; if data – the number of rows. *Important*: if a value in the data column equals **outC**, these rows are not used in length calculation.

Integral(G, P) calculates values of a grid integral across polygon P. For correct calculation, the polygon must lie within a grid rectangle.

MIN(X), **MAX**(X), **MIN**(X, P), **MAX**(X, P) are functions defining the minimum or maximum argument values, as well as the minimum or maximum within the polygon *P*. If the first argument is of data type, rows with the outC value are ignored. For *line* or *polygon* types, the minimum or maximum attribute is selected; for *grids* – the minimum or maximum value of the node.

9.2.6. Differentiation function

Dx(G), **Dy**(G), **Dxx**(G), **Dyy**(G), **Dxy**(G) – calculating partial derivatives of the grid function **G**(x, y) (grid) in the coordinates. Left to right: the first x-derivative, the first y-derivative, the second x- and y-derivatives, the second mixed derivative. The functions output is a grid with a set value of the corresponding derivative in each node.

Spec(G, index) is a special function for calculating differential characteristics of the grid (G), such as slope ratio, curvature, etc. The argument index takes values (1-17) and specifies the type of calculated characteristics. The set of characteristics (specifications) for function **Spec** is provided below. Notations are as follows:

Notations:

$$p = \frac{\partial G}{\partial x},\ q = \frac{\partial G}{\partial y}, r = \frac{\partial^2 G}{\partial x^2},\ s = \frac{\partial^2 G}{\partial x \partial y},\ t = \frac{\partial^2 G}{\partial y^2}$$

sign(x) = 1 at *x>0*, *sign(x) = 0* at *x=0*, *sign(x) = -1* at *x<0*

$$T = \sqrt{(1+p^2+q^2)^3},\ M = \sqrt{(1+q^2)/(1+p^2)},\ N = \sqrt{(1+q^2)(1+p^2)}$$

1. $\sqrt{p^2+q^2}$ - slope (norm of the gradient);
2. $r^2+2s^2+t^2$ - strain energy density;
3. $\frac{180\cdot}{\pi} arctg\left(\sqrt{p^2+q^2}\right)$ - slope gradient, degree;
4. $\sqrt{\frac{p^2+q^2}{1+p^2+q^2}}$ - steepness factor (dimensionless);
5. $180\cdot(1+sign(p))-90\cdot(1-sign(q))\cdot(1-|sign(p)|)-\frac{180}{\pi}\cdot sign(p)\cdot\arccos\left(-\frac{q}{\sqrt{p^2+q^2}}\right)$ - slope orientation, degree;
6. $H=-\frac{(r(1+q^2)-2pqs+t(1+p^2))}{2T}$ - mean curvature (1*m^{-1});
7. $A=\sqrt{rM-t/M}\cdot\frac{rM-t/M}{1+p^2+q^2}+\frac{(pqrM-2Ns+pqrT/M)^2}{2T}$ - non-sphericity (1*m^{-1});
8. $E=\frac{q^2r-2pqs+p^2t}{(p^2+q^2)\sqrt{1+p^2+q^2}}-\frac{(1+q^2)r+2pqs+(1+p^2)t}{2T}$ - differential curvature (1*m^{-1});

9. $-\frac{q^2 r - 2pqs + p^2 t}{(p^2 + q^2)\sqrt{1 + p^2 + q^2}}$ - horizontal curvature (1*m^{-1});
10. $-\frac{q^2 r - 2pqs + p^2 t}{T}$ - planned curvature (1/м);
11. $-\frac{p^2 r + 2pqs + q^2 t}{(p^2 + q^2)T}$ - vertical (cross-sectional) curvature (1*m^{-1});
12. $\frac{(p^2 - q^2)s - pq(r - t)}{T}$ - rotor (1*m^{-1});
13. *A-E* – horizontal excessive curvature (1*m^{-1});
14. *A+E* - vertical excessive curvature (1*m^{-1});
15. *H*-A – minimum curvature (1*m^{-1});
16. *H+A* – maximum curvature (1*m^{-1});
17. $\frac{rt - s^2}{(1 + p^2 + q^2)^2}$ - Gaussian curvature (1*m^{-2}).

9.3. Creating objects using Calculator

9.3.1. Creating grids

Building objects of **Grid** type with the Calculator means operations with other Project grids and numbers: functions of one argument (for example, **Sqrt**, **Abs**, **Sin**), finding the minimum or maximum values, cutting the fragment, merging grids, etc. When creating objects of other types (converting grid into polygon, taking grid values at points, etc.), grid can be used as an argument in many functions. To construct a grid using Calculator, the user must perform the following steps:

- Add an object of **Grid** type to the project;
- Set building method "**Create with Calculator**" to the grid;
- Add references to objects (grids) to be used in calculations;
- Set the Calculator command;
- Start building object.

Upon the end of calculations, the resultant grid is traced and the building is completed automatically.

9.3.2. Creating covers

The Calculator creates covers and performs two basic types of operations with covers: *operations with isoline attributes* and *operations with polygons*. Cover-polygon is used as an argument in some complex and very significant functions, such as integration and truncation of arguments of other types.

The sequence of user actions when building cover with the Calculator is the same as that for the grid.

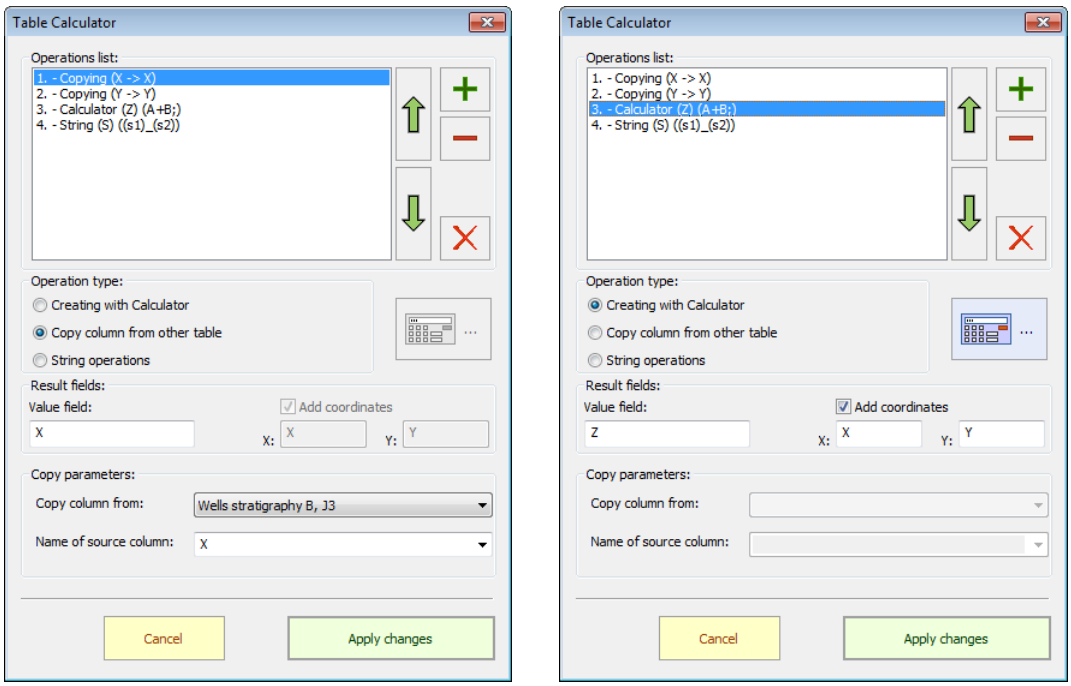
The user must be aware, what type of cover (**Simple** or **Polygon**) should be used in a particular function (as an argument or an output). Building cannot be completed if the created cover is of polygon type and the calculation outputs are unclosed lines.

9.3.3. Creating tables

Unlike covers and grids, creating tables has certain specifics. The reason is that the **GST** table can contain an arbitrary number of both numeric and text columns, while the **Calculator** can operate no more than three numeric columns – two coordinate and one with values. Therefore, construction of the table in **GST** can be performed through several consecutive Calculator operations, where each produces three (or one) columns. Construction can fail for the following reasons:

- Error in initializing variables;
- The Calculator output is not an object of "**Table**" type;
- Syntax error when setting a command;
- Derived columns are of unequal length.

The main dialog for building a table with Calculator is called with "**Building parameters**" button, as demonstrated in the figures below.



The dialogue consists of two control parts: a list of operations (top group) and control elements that specify parameters of each operation. Buttons located to the right of the list are to add or delete operation, or to clear the list completely. Table operations are performed in the same order as they appear in the list, with the subsequent addition of columns. The sequence of operations can be changed using the buttons marked with arrows. There are three types of operations: creating with Calculator, copying column from another table, and string operations.

If "**Creating with Calculator**" mode is selected (the right-side figure), the Calculator button becomes available to call the above-mentioned main dialogue. The user specifies variables (according to the general rules) and writes the command in the panel. This operation produces a set of three columns (sometimes one column), which are added to the resultant table. Columns are added with names that are specified in the text boxes united in "**Value field**" group. In this operation, the user can enable or disable Add coordinate columns option.

For example: the user's task is to take grid values at given points, as well as the first derivatives of the coordinates and place the results into a single table. This table is built in three steps. First, the

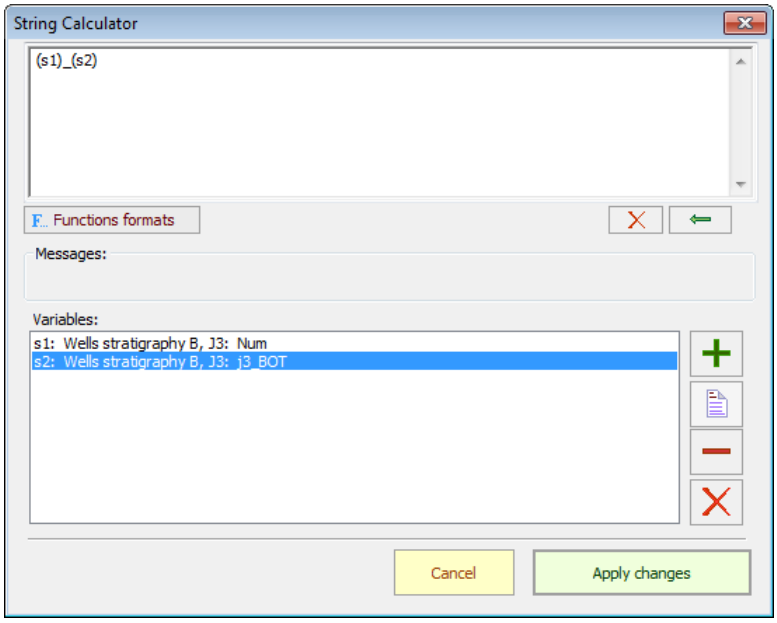
grid values are taken and three columns (coordinates and values) are entered into the table; the next two operations produce derivatives for **x, y** ("**Add coordinates**" option can be disabled, since the same locations are computed).

In sequential computing of the table columns, previous outputs can be used in computing the subsequent columns. The name of the object under construction appears in '**add variable**' dialog in the list of objects. The user just needs to specify the name of the previously computed column.

Note. When a table has one row only (without coordinate fields), the Calculator perceives its field content as a number.

When "**Copy column from another table**" mode is selected (the left-side figure), the user specifies in the drop-down list box the source table, name of the column to be copied and the name of the resultant column. Copying is useful in cases when the unmodified numeric column should be moved (to avoid to be multiplied by 1 and added to 0), and there is a need of copying text columns.

String columns require the third type of operations, i.e. "**String Operations**". By selecting this type of operation, the user must specify the name of the resultant column and the command – in the string dialogue box.

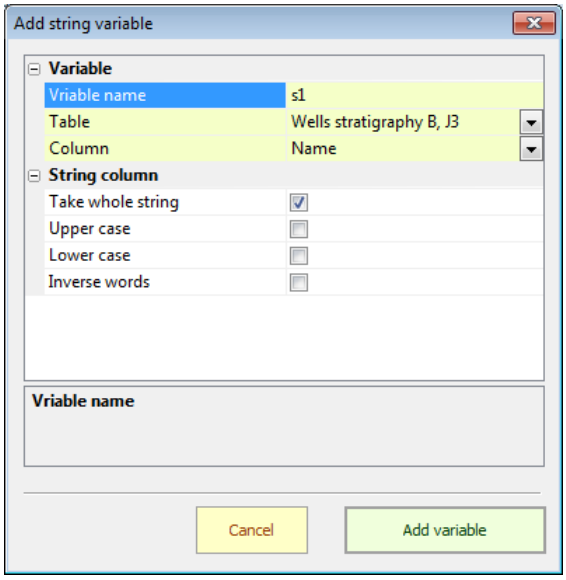


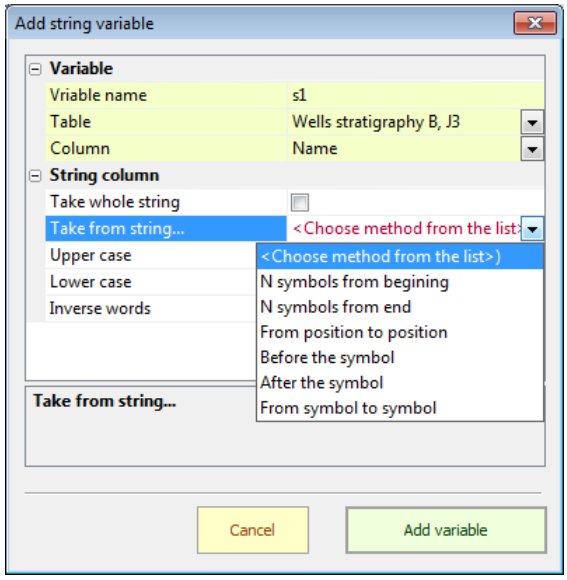
Similar to the mathematical calculator, the list of string variables is located at the bottom dialog box: buttons to the right add a variable, set the variable parameters, delete the selected variable from the list, or clear the entire list.

The string calculator command is written in the panel on top of the dialog. Variable names are entered in the display in parentheses (otherwise they will be perceived as ordinary strings). Additionally, the command can contain arbitrary text, i.e., the resultant string is generated from the variables' content and arbitrary text. In the above example, the variable «s1» denotes the table column with the well number and «s2» - the horizon depth. This operation results in a new string column containing the text in the format "**number_depth"**.

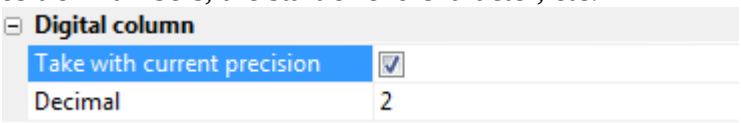
A string of any complexity can be created under this pattern. Sign '**(**' or '**)**' is a special symbol to denote a variable. To add sign '**(**' to the string, '**((**'' must be entered in the panel, i.e., if the operation produces a string in "**number_(depth)"** format, enter the command: «**(s1)_(((s2))**».

String variable parameters are set in the dialog called by adding a new variable, or by pressing the variable parameters button.





The user must specify the name of the variable in the dialog box and link it to a column in a specific table. The user specifies below parameters of text conversion related to the string column variable: change all characters to uppercase or lowercase, invert words. By default, the entire string is taken from the column; however, sampling from the string (right-side figure) can be selected in the drop-down list by disabling "**Take whole string**" option. Depending on the method selected, the user must set the start or end position numbers, the start or end character, etc.



If the variable-associated column has numeric type, the user is offered another set of parameters. By default, numeric column data are converted to a string with all significant digits, otherwise the user can specify decimal places in the "**Decimal**" option.

10. Print Object

10.1. General

Print layout (referred to as **Layout** or **Print**) – object controlling the process of laying out (assembly, formatting) and creating paper copies of **GST** objects. It is loaded into the hierarchy tree using the context menu **"Add Object → Print"** or icon  on the toolbar.

Initially the visualization window for this object includes one "clean" portrait-oriented sheet. In the process of creating the **Print Layout**, the user can:

- Set format, size, margins and number of ***pages*** required for the print;
- Create the ***print areas*** (set positions for print areas, scales, etc.);
- Load ***external (reference)*** print ***objects***, i.e., the objects located in the **GST** project tree. In this case the **Print Layout** receives them as a reference.
- Create *internal (custom)* Printer Layout *objects* (frames, lines, text, etc.);
- Adjust ***Print Settings*** of the selected printer (plotter);
- Use ***Preview*** before printing.

Laying-out is controlled by the main menu items "**Create**", "**Print Layout**", toolbar buttons 

and the pop-up context menu (right-click) in the working window. These commands allow creating print areas and internal objects, calling up the **Layout** parameter dialog, and managing the paper size.

Loading external print objects is performed in the **GST** hierarchy tree by creating references, usually by dragging the specific object (**grid, cover, table, picture**) into the **Print** object. Double clicking the reference will call up the dialog "**Print Parameters**", where printing attributes of a given object are specified. The "**Print Parameters**" dialog allows managing the print areas and page parameters.

There are *two methods* of **re-scaling** in the **Printing** visualization window. First, standard for **GST**, is holding the **Shift** button and outlining a rectangle using the mouse to increase the scale. Second – using buttons on the right side of the toolbar (as shown above).

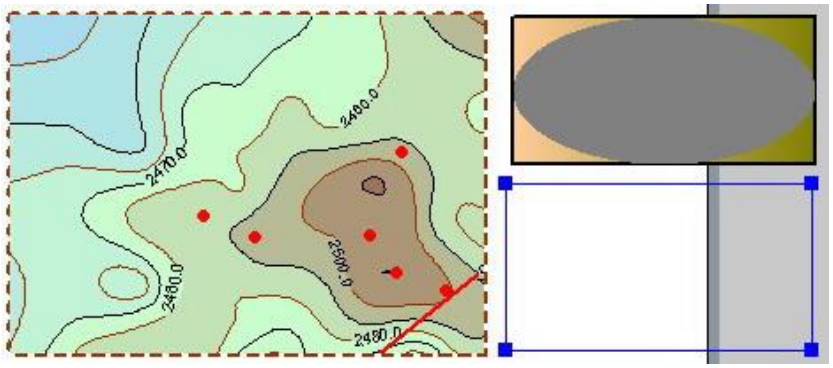
10.2. Print areas

Print area (or **area**) is a basic element of the **Print Layout**. Each object going to be printed is assigned to a certain *rectangular* print area. The area may be empty or contain either one internal print object or number of external (reference) objects. The areas may be located arbitrarily in the working window of the **Layout,** independently from each other and from the paper position. The order of object overlaying (when it happens) and the order of sequence in each area determine the sequence of object printing on paper. The required hierarchy is established in the "**Print Parameters**" dialog.

Each print area is represented by a blue rectangle and the selected area additionally is marked by small colored rectangles at the corners. The selected area boundaries are visible through all the layers of the print objects. The print area content is shown even outside the paper boundaries, but only the part that coincides with the paper (considering the margins) will be actually printed. This can be monitored by turning on the **Preview mode**. The area will be printed if its visualization parameters are set: boundary (with defined weight, color and style) and background (with selected color fill template; discussed later).

Users may hide print area boundaries with the option **Hide Frames** in the **Print Layout** menu or on the **Paper** tab in the "**Print Parameters**" dialog.

In the **Active Area mode** only the content of the selected area is shown, the rest are designated by frames with the labels. This mode is used only for viewing on the screen.

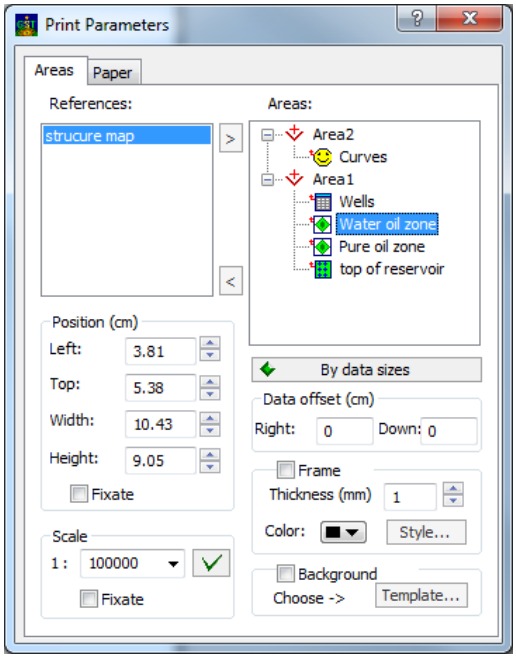


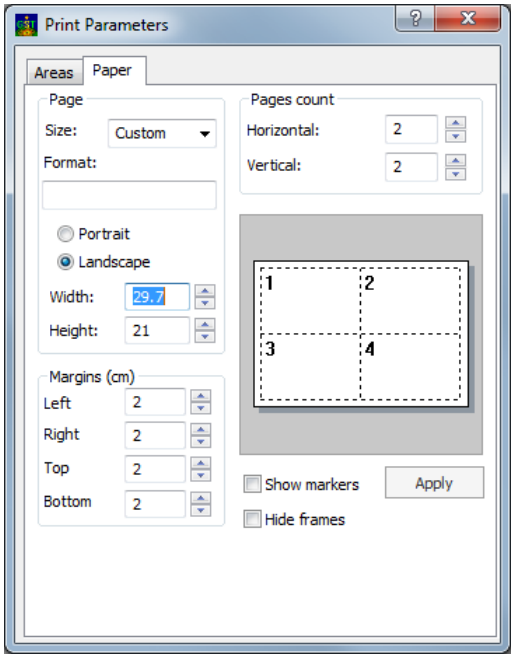
On the figure above three print areas are shown. The first (left) one is bounded with dashed line and contains three printing objects: a grid represented by contours and filled with color, a table (well symbols) and a cover (a line in the bottom right). The second area (top right) has a contour and its own gradient fill, and contains the elliptical shape. The third area is empty and is currently selected.

To load external objects (references) in **Layout**, the user has to create at least one print area. To do this, it is necessary to enable **"Create area"** mode in the main menu or in the toolbar, then select using a mouse a rectangle area. When creating the internal print object, the print area is created automatically.

10.3. Layout Parameters

"**Print Parameters**" dialog may be called by using the option «**Print object parameters**» in the main menu or in the pop-up menu, or by double clicking the mouse in the **Print** working window (outside the print areas of the internal print objects). The parameter dialog has two tabs: **"Areas"** and **"Paper"** as shown in the figure below.





The first tab is provided for working with print areas that have been previously created in **Layout**. Every new area is added in the beginning of **the areas tree** – this means that its contents will be printed last (as an upper layer). The print area name, assigned the next sequence number, for external print objects begins with the word "Area", for internal print objects – with the word respective to the object type ("Frame", "Legend", etc.). Nevertheless, any area can be renamed. Areas can be moved within the tree, resulting in a change in the print order.

There are three ways to send external **GST** objects (reference) to **Print**:

1) Create the references in the **Print** object; in the ***references list*** (in "**Print Parameters**") new elements will appear. To assign print area inside which this object will be drawn, user must select the reference in the list and an area in the area tree, then press the button **«>»**. The reference will be moved from the list to the tree, inside the selected area. To remove a reference from the print area, the user has to select the reference in the area tree and press the button «<». One can re-assign another print area for the reference both by these two buttons or moving the reference in the area tree.

2) By dragging the object (or a group of objects) with the mouse from the GST hierarchy tree to the working window of the Print object into the desired print area. In this case the print area size remains the same.

Notes:

- If the mouse button is released in the selected area, then dragged GST objects will be moved into this area;
- If the mouse button is released outside the selected area, the top area where the mouse has stopped will become the print area;
- If the mouse is released outside all areas, then objects will be moved to the references list.

3) Using the menu command «**Object → Send to Print**». In this case a new **Print** object will be created where the only one print area will appear. The desired object with all references (graphic layers) will be moved to this area with its visualization parameters. The print area is adjusted to the object's size.

In the print area, objects will have the same drawing parameters as they had in the project. They can be changed in "**Print object parameters**" dialog (double click on the reference in the hierarchy tree).

If the reference contained in the reference list ("**Print Parameters**" left), the reference is insignificant. Upon being moved to some print area, the reference becomes significant, and its color in the project tree is changed.

Order of the object's printing exactly matches the order of the object's (layer's) position in the area tree: from the bottom to the top. First, the lowest area is printed (boundaries and background if available), then, objects contained in inside (also from the bottom to the top). The upper-most layer is printed last.

Moving along the tree of references and area by mouse is done as per the following rules:

1) The area to be moved (together with its contents) takes the place above the area, where it was released. There will be no reordering if the area is released on the reference.
2) If the reference, when being moved, is released on the area icon, then it will become the top (last for output) layer in this area. If the reference is released on another reference, then the moving reference will be set one position higher than the one where from it was released.

It must be noted, that the area of internal print object (Frame, Text, Legend etc.) can not contain any references.

The print area can be deleted – the user has to select the area and press **"Delete area"** button  in the toolbar or keyboard button **Delete**. All references previously contained in this area, will be moved to the list of references. The user may delete an area with the corresponding right-click menu command in the print area's tree ("**Print Parameters**" dialog).

The pop-up menu items (as well as toolbar buttons) allow executing the standard operations with the print areas: «**Cut**», «**Copy**», «**Paste**», that is convenient when duplicating the objects created in this project or borrowed from another project.

For instance, in the project P1 in the **Print** object some rather complicated stamp was created. In the new project P2 among the other print objects it is necessary to include the same stamp. Recommended actions: add to "P2" project, project P1. Activate the working window of **Print** object, containing the required stamp. Select the stamp print area (in a work window or in an areas tree) and copy it to the clipboard. Move to the newly created object **Print** of the Project P2 and paste in it the stamp as a new print element.

It should be noted, that:

1) Internal print objects (section, legend) referring to the external objects, while being copied into another object of **Print** type, keep their print parameters (attributes), and references to external objects should be redefined.
2) The area of external print objects is copied without references contained in it.

When executing the operation **Clear** for the object **Print** (context menu in the **GST** directory) all print areas are deleted but the external objects are saved as references.

Area selection can be done using one of the following ways:

- by mouse click in the working window (when several areas intersect, the top area is selected);
- by using the keyboard button **Tab –** for moving through the areas, according to the directory, from the top to the bottom;
- by **"Select area"** option in the context (right click) menu;
- by mouse click in the area's tree ("**Print Parameters**" dialog).

Multiple area selection is allowed - by clicking the mouse in the visualization window while pressing the **Ctrl** button. The functions move, delete, cut, copy, and paste are available for selected areas.

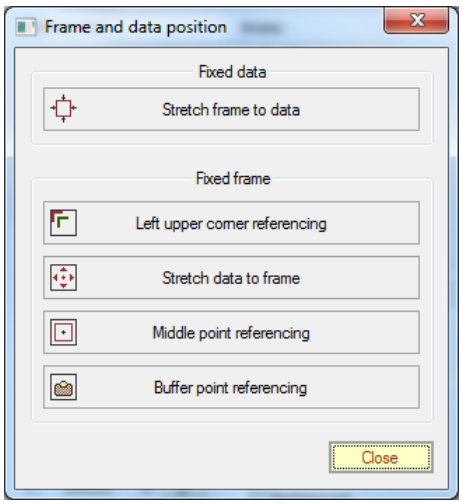
The **following parameters of the selected area** may be changed in the "**Print Parameters**" dialog:

- *size* (width, height) and *location* (area's left upper corner position). Another way to change these parameters is by moving the mouse while pushing the left button (inside the area – for moving, in the area corners – for changing the size). These parameters can be fixed;
- *scale* – drawing scale must be set for all external (reference) objects contained in the print area;
- changing the area size exactly to the size of contained objects and on the opposite – by pressing the button "**By data sizes**";
- *offset along the coordinate axis* of the exterior print objects respective of the print area; (by default, offset equals 0, when the data are tied in to the left upper corner of the area); negative offset is also possible;
- *frame* drawing parameters (thickness, color and line style);
- *background filling* parameters (fill template).

Notes:

1) If the toolbar buttons and are switched off the user may move with the mouse, area frame with all content. If button is turned on, only the area frame is moving, and its contents position in relation to the paper is not changed. If button is turned on, the area contents are moved while the frame position is not changed.

2) When pressing the button **"By data sizes"** a dialog is opened, which provides for the following actions:



When data position is fixed:

a) Stretch frame to data;

When the area position is fixed:

b) Left upper corner referencing;
c) Stretch data to frame;
d) Middle point referencing;
e) Buffer point referencing.

In case a) data are fixed respective to the paper, the area moves over the display and its size changes. In other cases, the data move respective to the fixed area. **In case b)** left top corners of the frame and data rectangle are superimposed. **Case c)** is similar to the previous one, but the data scale is changed so that the data would exactly fit the area size. For **case e**) one point should be previously copied in the buffer.

3) The area move is accompanied with the appearance on the screen of horizontal and vertical dash lines that prove the superposition of the given area boundaries with the respective boundaries of other areas. This option is helpful when aligning the areas.

On the "**Paper**" tab the **paper page parameters** can be set:

- *Page size;*
  Before starting to print or preview, if the page size is different than the selected printer parameters, the program will send a message that invalid printing is possible. For concurrence of the paper size, the user should either select the proper print device (for instance, a plotter), or select in Layout the page size supported by the printer.
- *Page orientation* («Portrait» or «Landscape»);
- *fields* on the page;
- *number of pages* (horizontal or vertical), required for this **Print Layout**;
- the mode for *Page layout preview* for convenient layout*;*

- *Hide frames* mode on the screen (to hide area frames is necessary, for instance, when using photo-mode);
- The *active area* mode, when all areas except of the selected ones are displayed without their contents.

For basic operations to create print Layout, the **Undo** option is available. These operations are:

- Print areas moving;
- Changing the print area scale;
- Changing the object's display scale;
- Creating, selecting, moving and changing the area size;
- Creating the lines and polygons nodes;
- Lines and points labels editing;
- Preparing a stamp in the table dialog.

10.4. External print objects parameters

External (reference) print objects (**grid, cover, table, image)** parameters dialog pops-up when the user double clicks at the reference shortcut in the **GST** hierarchy tree. Print object parameters are divided into **basic** (type, name and print area) and **display attributes**. Attributes (print settings) may be copied from the other objects of this type, by using the button **"Import**". Let's consider the attributes of various object types in more detail.

10.4.1. Grid

When printing the object **Grid, the** user can operate with the following parameters:

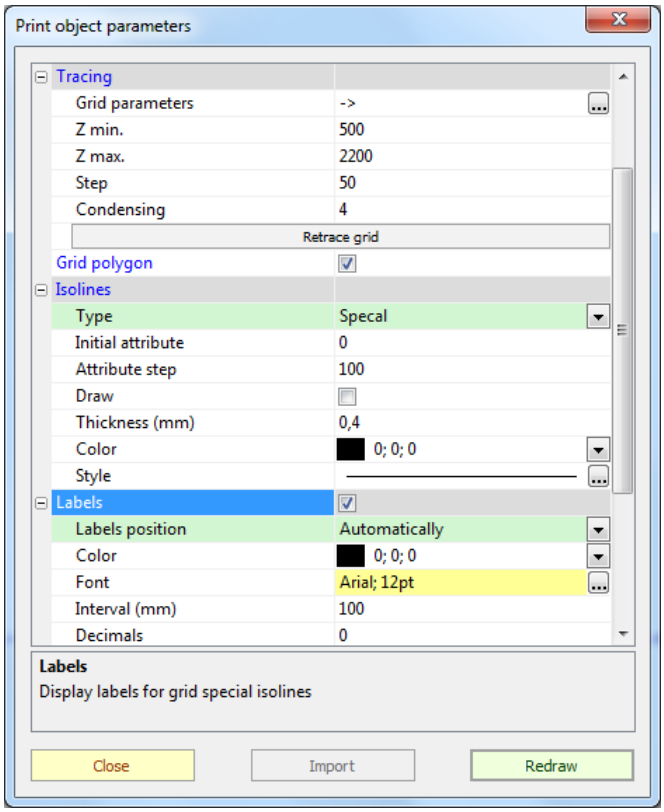
- ***tracing*** the grid – the grid sent to **Print Layout** may be re-traced with other parameters (minimum and maximum values, level step, condensing);
- ***drawing inside the definition polygon*** or drawing inside the grid rectangle;
- ***isolines drawing*** (thickness, color and line style);
- ***level filling*** with different palettes;
- ***isoline labels*** (color, font, interval, precision); labels position estimation is done either automatically or manually;
- ***structure drawing***, based on the results of the structure's analysis (Chapter 4.3);
- ***grid cells drawing*** (thickness and line color);
- ***insert cells drawing***;
- ***gradient vectors drawing*** (color, etc.);
- ***small structures filter*** – an option that allows to hide small structures; the critical structures size is defined by the user.

All grid **isolines** in **Print Layout** are divided into two types: **special** and **ordinary**. Special lines are those which attribute differs from **initial attribute** parameter by user defined step. Ordinary lines are all the rest. To assign special lines, it is necessary to assign the initial attribute and the attribute step. For special and ordinary lines, the user may set different drawing parameters (color, style, etc.).

Option "**Min. diameter**" allows to not display (print) small closed isolines (structures). The size is set in mm and if this parameter is zero, all isolines are displayed. "**Min. diameter**" value is set equal to all grids for current "**Print Layout**" object.

By default isoline's (levels) **label positions** are defined automatically. User may change automatic mode to manual one and edit the position of each label. A toolbar button  is active when

even one external print object (**Grid** or **Cover)** is using label positioning in **manual** mode. This button enables "**Lines labels edit mode**", where the user may set a certain position for each label.



Attention! When "**Lines labels edit mode**" is enabled, all operations with print areas by mouse are blocked!

In this mode new context (right click) menu options are available: **"Move label or name", "Create label"** and **"Delete all labels"**.

To create a label with a given position the user should enable the **"Create label"** option and mark with a mouse click the desired point. To delete a label the user should enable "**Delete label or name**" option and click on a desired label. To move a label or line name the **"Move label or name"** option should be enabled. Users must drag a label from its current position to a new one. These operations can be performed to grid's isolines and cover lines as well.

Menu command "**Delete all labels**" deletes only manually defined labels for the upper layer reference object (the upper in an areas tree in "**Print Parameters**" dialog). When the user switches label position mode from manual to automatic all manually established positions will be lost.

The option "**Minimal size**" in a "**Labels**" category will not display labels with height less than a given value (in mm). By default this value is equal 1 mm, and this option is valid only for drawing on a screen, but not for printing.

10.4.2. Cover

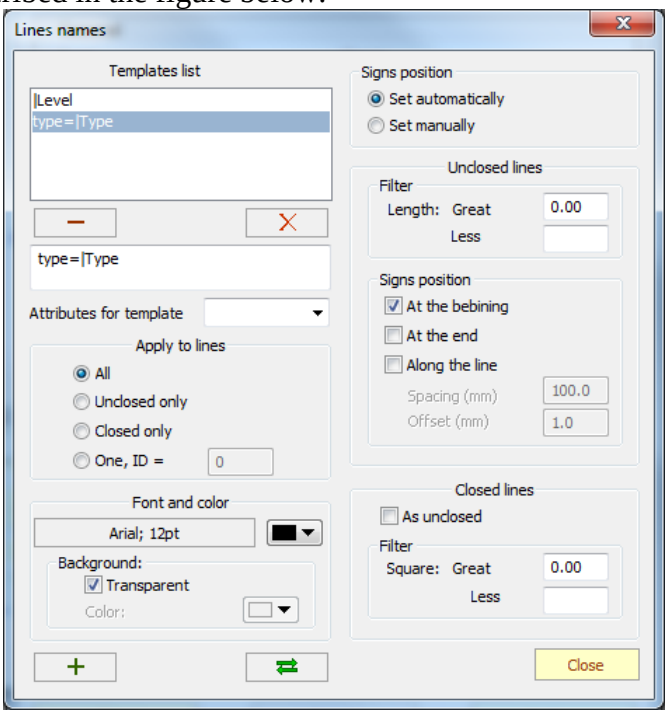
"**Cover**" drawing parameters are similar to "**Grid**" drawing parameters: the user may set line's color, style, labels positions etc. Additional "**Cover**" drawing features are as follows:

1) The "**Polygon**" filling mode is available. The polygon may be filled: with one color, gradual transition from one color to another and using the selected template; the transparency option may be also applied.

2) Optimization (simplification) of the lines is allowed ("**Optimize lines**" option), when excessive points are deleted. The new line may deflect of the initial one to the distance not exceeding the set value. Line optimization is recalculated when the print area scale is changed.

3) When using **"Cover"** loading from **shape files**, lines can be signed with **names** which are formed from the **dbf**-fields using the set templates. The process of creating and editing the line names is discussed below.

For covers, loaded from shape files, the "**Line names**" option is available. With this option the user may call the dialog, described in the figure below.



For this cover, any number of templates can be created. Using this template, **GST** generates line labels. Every template is represented by one or several lines including **dbf**-fields names and any text. Before the **dbf**-field name, the '|' character should be set. For instance, the template line "type=|Type" indicates that **GST** will generate a label text "*type=**current value***".

The selected template can be applied: to all lines, closed or unclosed lines or to a single line with the given ID. When generating the line names, filters may be applied: "length" filter for unclosed lines and "square" filter for closed lines. Label positions for unclosed and closed lines are calculated differently. For unclosed lines, labels are placed along the line (at the beginning, at the end or along the line with user defined interval). For closed lines, the labels are positioned horizontally in the polygon's rectangle center. Closed lines can be treated as unclosed as well.

If label positions are defined manually, the "**Line labels edit mode"**  becomes active. In this mode, the following context (right click) menu options are available: "**Move label or name**", «**Rotate name**», «**Create name by a template**», «**Delete label or name**».

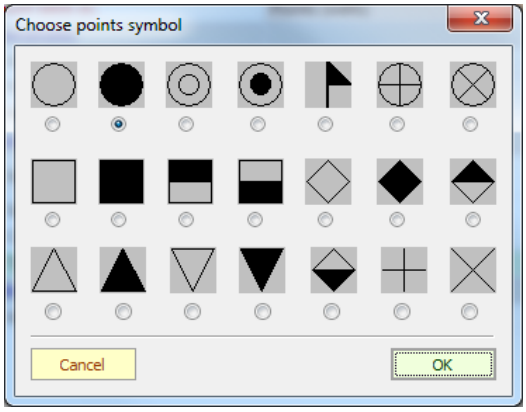
Labels can be moved without changing the angle. Rotation is done by moving the mouse with the pressed left button around the top left corner of the signature rectangle, the upper side of the rectangle is marked by red dashed line. Upon selecting the template, one should note the position of the top left angle of the new label. To delete the label, it should be just clicked by the mouse.

10.4.3. Table

Point data tables can be displayed in a **Print Layout** as: ***wells*** or ***seismic lines (profiles)***. For both data types the following shall be defined:

- Coordinate fields name (wells or shot points);
- the NAN – value (if necessary).

For the table data that should be displayed as points (wells) the following **parameters** shall be set: size, color and type ("Any symbol" or "Well"). If "any symbol" is selected, the user may choose the point symbol in the dialog below ("**View**" option in a reference properties dialog):



«Well» shall refer to a certain well category (wildcat, exploration, etc.) and have a certain condition (rigging up, testing, producing water, etc.). There are 4 categories and 24 conditions. Well symbol may be the same for all points or taken from certain table columns (with categories and conditions);

There are five possible label positions around the point symbol (User should enable "**Labels**" option). Label text may contain information from user defined columns. First, the user must define the label position (Top left, Top Right etc.) than choose a source column for the label ("**Label text column name**"). If, for instance, column name "Num" is chosen, a string "|Num" will appear in the "**Text**" field. The user may edit this text, for example, like this: "N - |Num". In this case the label text will be: "N – *current number*", where "current number" is a value from "Num" column. If a numeric value is taken from the table, the precision is set by the "**Decimals**" option. For each of the labels, an **exception** is provided, the label is not displayed when it matches. The label's background may have color or be transparent. Also an underline for each is provided.

Label positioning is done automatically and may be fine-tuned manually. In the first case, labels are placed by the template in the following positions: top left, top right, bottom right, bottom left.

To change the label position, in the toolbar **Print Layout** (and in «**Layout**» menu) a button for turning on the mode "**Move point labels**" is provided. The button is active when the label positions are estimated manually. With "**Move point labels**" mode enabled, drag with the mouse a label from previous position to new one.

Attention! Upon turning on this mode, mouse operations with the print areas are blocked!

For the table containing information on **seismic profiles**, the following **parameters** can be set:

- Indicate the table columns containing the profile number (name), seismic crew number (name), the final year of seismic acquisition; - the first parameter is necessary, the second parameter is used to display different seismic crew data with different color; all these parameters are displayed as: “crew/year/profile” (see figure below);
- Seismic picket drawing parameters (color, size, style, labels etc.);
- Profile label properties (at the beginning, at the end, indent etc.);
- Seismic line color and thickness; seismic lines may be displayed with certain user defined color or with random color.

10.4.4. Image

Image size and position depend on the following parameters:

- Coordinate referencing – performed in the **GST project**. If the image has coordinate referencing, it is scaled like all other objects in the print area. If there is no coordinate referencing, the top left corner of the image is tied to the top left corner of the print area.

The size of an “unreferenced” image is defined by the options “**Change size**” and “**Lock ratio**”:

- If the “**Change size**” option is enabled, the image stretches to fit the current print area;
- If the “**Lock ratio**” option is enabled, the image stretches with a locked horizontal and vertical scale ratio.

10.5. Internal print object parameters

Internal (or native) **Print Layout** objects are: **frame, line, polygon, text, stamp, special tables, scale ruler, geographic grid, legend, cross section**.

To create these, in the main menu, “**Create**” select a required menu item (for instance, **Create → Frame → Ellipse**). Then, in the working window, with the mouse select a rectangle to define the object’s position. A dialog “**Print object parameters**” immediately pops-up. To create a polyline and a polygon, a different sequence of actions is required (see below).

Object parameter dialog is called by double clicking the mouse in the relevant object (in a working window or in the area tree). Below, **visualization parameters** of the different objects are discussed.

10.5.1. Frame

Three types of frames exist: **rectangle, rounded rectangle and ellipse**. The frame drawing attributes are:

- Thickness, color, style of the frame line;
- Frame’s filling parameters; if filling is not used, the frame is transparent.

10.5.2. Line and polygon

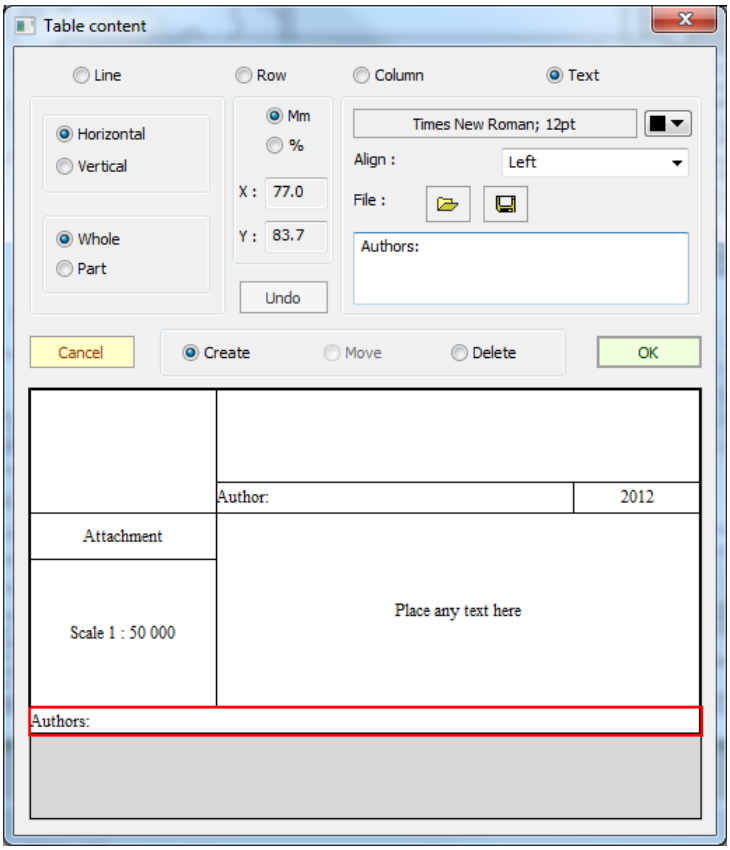
The following types of line are possible: **line, polyline, spline curve and arrow**. Simple lines (line, arrow) are created after selecting their rectangle. Their direction may be changed by changing area size and position. To create a polyline (curve), the user has to perform the same steps as when creating a polyline or spline curve in a "**Cover**" working window. A **polygon** (or **spline** polygon) is created in the same way as a polyline or a spline curve.

10.5.3. Text

The print parameter dialog of this object contains a multi-line window for the text input. Text may be entered not only from a keyboard, but also from a file (saved in a file). Text attributes are: font, color, angle, aligning options (left, center, right).

10.5.4. Stamp

Stamp is a special rectangle object divided with horizontal and vertical lines into a number of text cells. Line number and position are determined by the user. The **stamp** may be considered as a table with arbitrary content.



In the print object **parameter** dialog the following features can be set:

- Stamp rectangle size;
- Stamp scaling;
- Thickness and color of the frame lines;
- General (default) font and color; specific font and color can be assigned for each cell;
- Caption name and font.

Stamp is developed in the "**Table content**" dialog (shown above) which is called from the print object parameter dialog.

When working with lines or text it is necessary to turn on button (lines or text). There are three modes of operation with stamp elements: **Create, Move** (for lines only) and **Delete**. **Work with lines** is performed by mouse in the display window. The type (**Horizontal, Vertical**) and size (**Whole, Fragment)** should be defined by switching the respective buttons. Creating and deleting performed by mouse click, and moving – with left mouse button pressed. Whole line is limited by the stamp frame, fragment – by the nearest intersecting cross lines. When the line is moved, text elements in the adjacent cells are scaled.

For text input and edit functions, the mode **Create** is selected, and cursor points to one of the stamp cells (it is selected by red color). Then in multi-line edit box the text must be set. Actions may be performed in the different order: first, insert the text and then point to the target cell by the mouse. Changing each text block parameters (font, color, alignment) maybe performed at any stage of the text entry. Text can be downloaded from a file or saved as a file. It is recommended first to prepare the linear part of the stamp and then create all necessary cells and then insert text.

The function of deleting the current and creating new **rows** and **columns** is provided. The user has to turn on the respective mode and point the row (column) with the mouse. In **"Delete"** mode the row or column is deleted, and in **"Create"** mode a new element is created below (or to the right) of the specified element. For all actions in the table dialog, the option **"Undo"** is provided.

10.5.5. Tables of special format

These are the tables required for **preparing the volumetrics plan**:

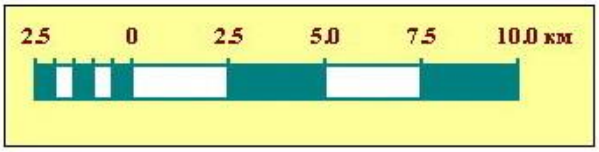
- table of the volumes of the fluid containing rocks;
- table of oil reserves;
- table of gas reserves;
- table of test results;
- table of the investigation maturity.

While creating the table of special format, Print Layout function forms a respective table heading. The table content is created by the user (with the stamp creating dialog, as shown above).

For tables of three types (tables of volumes and reserves), **GST** creates their contents in the native format based on the volumetrics results. The table may be loaded either from file (with extensions **.vt, .ot, .gt**, respectively), or from the "**Table**" object (through reference), which is a volumetrics result.

10.5.6. Scale ruler

When developing a **scale ruler**, user can operate the following **parameters**:



- Width of one big interval of the ruler (in centimeters of the paper and real units – **m**, **km**);
- Number of big intervals (while the number of small intervals to the left of "0" mark always equals 5);
- Ruler height and line thickness;
- Color of the ruler and label font.

10.5.7. Geographic grid

Geographic grid represents a set of meridian and parallel lines, displayed in the print area. Main condition for creating the geographic grid is to define a referenced print object and coordinate projection for data in the print area. In the "**Coordinate conversion**" dialog, the respective initial projection shall be selected.

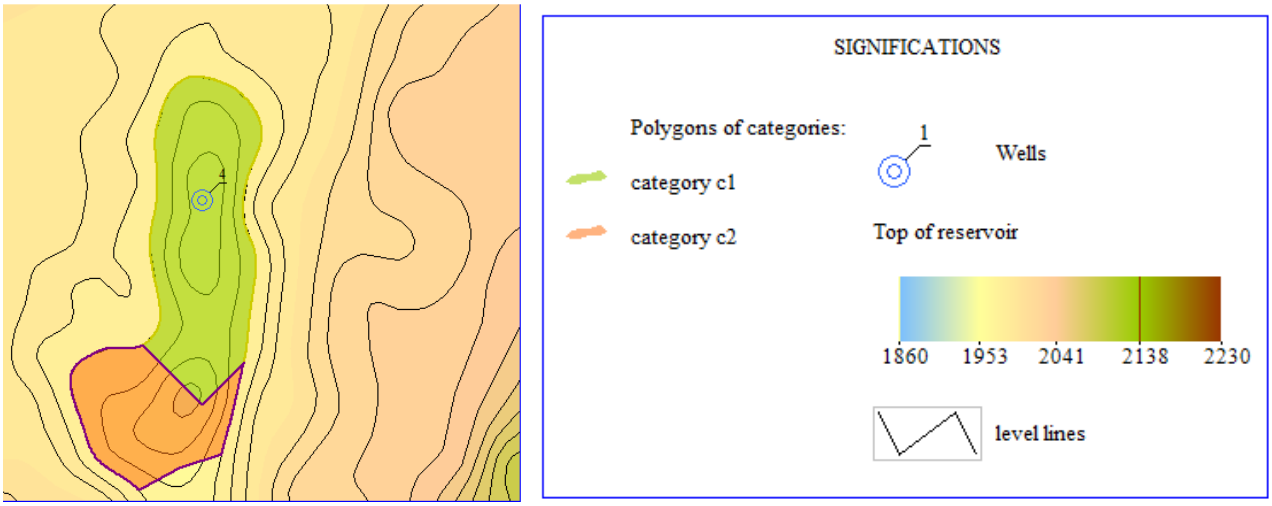
When printing out the geographic grid, the following **parameters** can be changed:

- Thickness and color of the rectangular frame, coinciding in size with the print area;
- Thickness and color of the grid lines;
- Grid step (in degrees) along horizontal and vertical axis – distances between adjacent meridians and parallels, respectively;
- Attributes of the grid labels located outside the print area. They are: label step (in degrees), indent (in mm), font, position (horizontal, left, right) from the left and right side of the frame. Signatures are output in the "degrees-minutes-seconds" format.

In the category "**Referencing**" select an option "**Overlay areas**" to overlay the geographic grid print area and referenced object's print area.

10.5.8. Legend

Legend internal object represents a set of rows and columns, defining the list and contents of the printed objects. Number, size and order of the legend lines are defined by the user. The legend heading, by default, is "**Legend**".



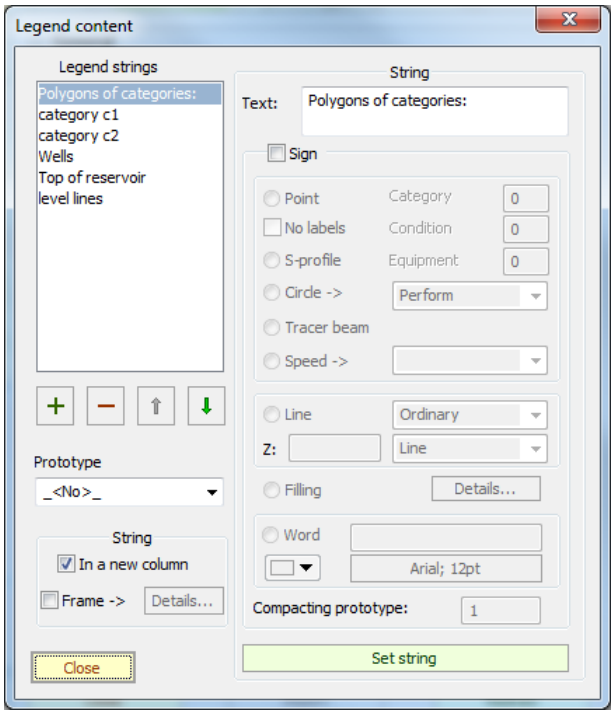
Legend is created in the "**Legend content**" dialog, which is called from the print object parameter dialog. Legend "strings" are placed in the list that determines their printing. Created strings may be deleted and moved in the list. To place current string to the new legend column, it is enough to select the option "**In a new column**" for the respective string.

Each "string" represents the combination of the **symbol** and **notation** text. Also it is possible to use text without a symbol and a symbol without text. A symbol is always located within a frame, that may be shown by the option "**Frame** →", or hidden. In the latter case, the frame height is limited by the string height, and the width is twice as big as the height.

The symbol size (for instance, well symbol) corresponds to the real size of the print object. To adjust the symbol size (bigger or smaller) the factor "**Compacting prototype**" (>1 for compression or <1 for stretching) is provided.

For the **frame,** the following **parameters** can be set:

- Shape (rectangle, rounded rectangle, ellipse);
- Size (width and height);
- Line drawing options – color, style, thickness;
- Background filling options.



Notation text for the string is entered in the multi-line text field. Color, font and aligning options of the text (same for all the lines) are set in the print object parameter dialog.

The "string" is usually linked to some print object (**prototype**), for which the legend is created. In this case, symbol drawing parameters are determined by the print properties of the particular prototype. For the legend string not related to the prototype, there is a possibility to assign a symbol as "**word**" (i.e. text or number) with adjusted color and font.

Symbol of the "**Table**" object, depending on the type of contained data, is either a point, or seismic profile. For the point's symbol, the fields **"Category"** and **"Condition"** are important if the well symbols taken from table columns serve as well symbols. It is possible to derive the symbol without a label even when the prototype has labels.

For "**Grids**" and "**Covers,**" option "**Line**" is accessible. For a grid, the user may display ordinary and special lines. The line's symbol appearance may be: Line, Polyline and Curve.

There are two ways to fill a symbol with color: **Polygon** filling and **gradient type** filling. In the first case, the symbol has the polygon shape, which is filled as the print object. In the second case, there is a possibility to adjust **filling details** in an additional dialog. The following options can be set:

- **Filling type**: *discrete*, when each cell has particular color and matches some attribute of the contour, or *gradient*, when the palette, accepted for the print object, is displayed;
- **Position** (vertical, horizontal);
- **Label position** (left, right or top, bottom);
- Rectangle size for the filling symbol (when the frame is available, the frame size).

The filling symbol and labels are fitted to the frame size. Also, the font size corresponds to the font of notation text. If filling is not printed out, then the frame size is too small for the font. To change the font size of the labels, it is recommended to use the prototype compression.

10.5.9. Cross section

This print object is created as an analog to the **GST** object of the same name. To construct a cross section for a print object, the user must:

- Create a "**Cross section**" object (menu command "**Create → Cross section**");
- Create a print area and add to this area all reference data required for the cross section: grids, data tables and a cover, containing profile line;
- Set all drawing properties in the parameters dialog.

The **Parameter** dialog contains the following attributes:

- Scale (horizontal and vertical) that can be fixed. Otherwise the cross section is scaled to the print area;
- Profile line (thickness, color, display of nodes);
- Vertical axis and label drawing parameters;
- Grid line drawing parameters (vertical and horizontal step, color, style, etc.);
- Delta – minimal distance on the map, needed for placing well on the profile;
- Background filling parameters.

To define a profile line the user has to place a reference to a single-line cover into any print area and then select this cover in a drop-list "**Line**" (category "**Profile**" in a *print object parameters* dialog). A seismic profile, contained in a data table, may be also considered as profile line. In this case, an additional list, the user has to choose a profile number (seismic crew number, if needed).

The next steps are performed in the **"Cross section content"** dialog that is called from the print object parameter dialog. Object names (grids and well data) are displayed in the respective lists.

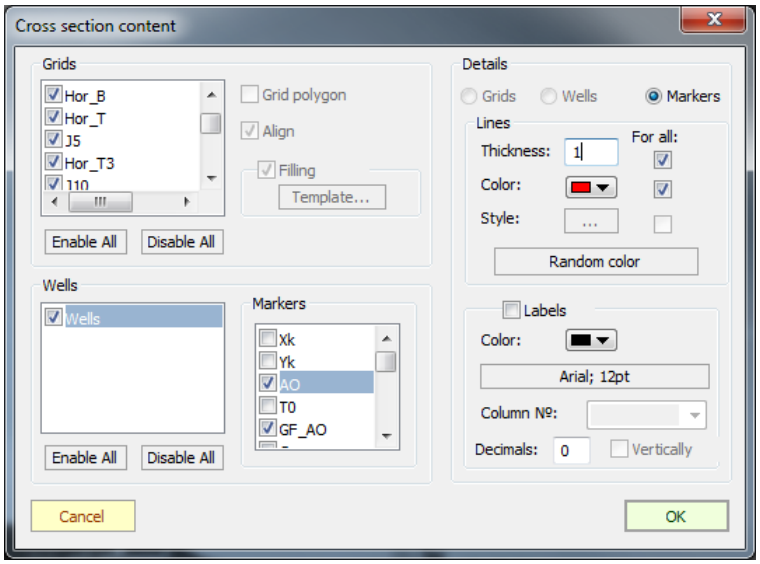
All grids downloaded in the **print Layout** appear in the list of **"Grids"**. Every element of the list has a button (flag) for turning its visualization on/off. When selecting the grid, its display parameters can be set:

- Thickness, color and line style;
- Labels color and font;
- Filling parameters (polygon filling);
- To account for the grid definition area; in this case the grid section is shown only inside this area;
- To assign this grid as the "*level grid*" for vertical alignment of the rest data: level grid will be displayed as the zero horizontal line, and the rest – as the difference in relation to this one.

The order for grid display in the cross section corresponds to the order of their position is as follows – from top to bottom: the first grid in the list is drawn first, each next one overlays the previous one. Grid filling area adjoins the bottom boundary of the section rectangle, if the depth axis (Z) is directed downward, and to the top boundary, if otherwise. The order of grid display can be changed by moving them in the list by the mouse.

In the list **"Wells,"** only those objects (tables) are included that have at least one well near the profile line. Parameter **Delta** (which equals 0 by default) regulates points hitting the profile. Every point (well) at the distance less or more than Delta value from the profile is considered located on the profile. Delta unit corresponds with the coordinate measurement unit accepted in the current project.

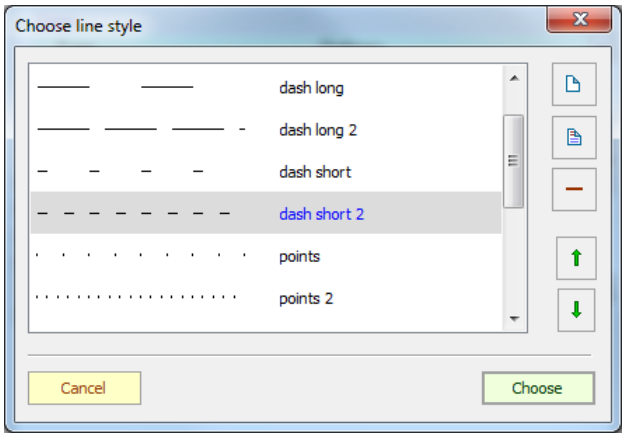
When the cover line is taken as the profile, only the tables containing wells (not seismic profiles) can be displayed. When seismic profile serves as the cross section line, the tables containing wells and seismic pickets of the given profile are considered (in this case Delta is not accounted for).



Wells are displayed as vertical lines with top triangles all over the section area. Seismic pickets are connected by lines over selected marks. Seismic line nodes are not displayed. When the element in the well list is selected, the list **"Marks"** is populated with the data-table field names. The selected mark is displayed on the cross section as follows: for a well – by horizontal line, for pickets in a line style (represents t_0 along the profile).

10.6. Line style – choose and create

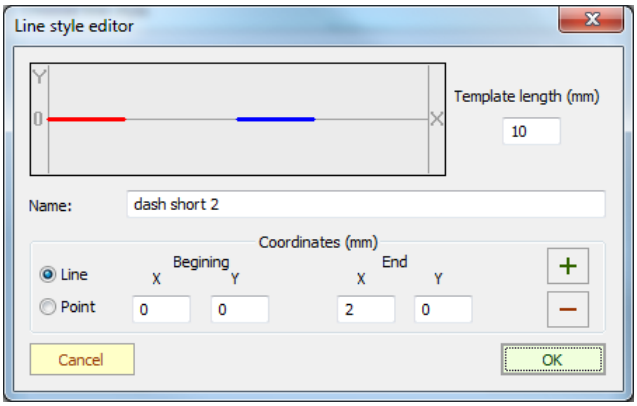
To select, edit or create a new line style the dialog, «**Choose line style**» is called. To call this dialog, press the button «**Style…**» in the print object (reference) parameters dialog.



The list contains a set of line styles for this PC (they are stored in the text file **LineStyle.txt**). Some styles are installed with **GST**, the rest are created by the user as needed. When saving the **GST** project, all parameters of the styles are saved, therefore, when moving the project to another PC, the styles are not lost but if needed (when missing) they are included in the service text file.

Every style on the list has graphic representation and name. The first on the list (default style – "solid line") cannot be edited, moved or deleted. To choose the style, it is selected in the list then press the **"Choose"** button.

To create a new style or edit the selected style on the list, press the button **"Create new"** or **"Edit",** respectively. The dialog **"Line style editor"** will open.



Every style is characterized by the template of user-defined length (in **mm**). The template consists of number of lines and points. Multiplication of the template drawing gives as a result an image of the whole line. Template lines and points in the *style editor* are displayed in the dialog in the orthogonal X, Y coordinate system. Coordinates of points and lines (start and end) are measured in millimeters. Using text fields to enter of the coordinates and the button **"+"**, the required number of fragments (lines and points) is entered in the template. To delete any fragment, the user has to select it with the mouse and press "-" button. The selected fragment is colored red. To edit a fragment, the user must delete the old one and then create a new fragment with new parameters, or create new and then delete the old one. To add new (or edited) line style to a list, "OK" must be pressed.

Creation (editing) of the line styles can be done in **LineStyle.txt** (all GST instances must be closed). Line styles in the service file are saved in the **following format**:

N
kL kP -1 x1 y1 x2 y2 ... -2 xp yp ... L Name
. . .
where: (all integer numbers)
N - number of templates (styles) in the file
kL - number of lines in the template
kP - number of points in the template
-1 – indicator that next follows the line (four numbers – coordinates of start and end)
-2 – indicator that next follows the point (two numbers – point coordinates)
L – template length (mm)
Name – name of the style (one or several words), can be missing.

Notes:

- The order of the lines and points in the record does not matter;
- Line orientation does not matter.