

Genome Sequencer FLX System Software Manual, version 2.3

Part C: GS De Novo Assembler – GS Reference Mapper – SFF Tools

October 2009



Genome Sequencer FLX System Software Manual

Software v. 2.3, October 2009

Part C

GS *De Novo* Assembler, GS Reference Mapper, SFF Tools

Table of Contents

1.	GS <i>De Novo</i> Assembler	5
1.1	Overview of the GS <i>De Novo</i> Assembler	5
1.2	GS <i>De Novo</i> Assembler GUI	6
1.3	Launching the GS <i>De Novo</i> Assembler GUI Application.....	7
1.3.1	The Main Buttons, Status Area and Progress Box.....	8
1.3.2	The Quick Start and Documentation Text Links	8
1.4	Opening a Project.....	9
1.4.1	Creating a New Project	9
1.4.2	Open an Existing Project.....	9
1.5	View Project Summary with the Overview Tab	10
1.6	Add/Remove Read Data with the Project Tab.....	11
1.6.1	GS Reads and FASTA Reads Sub-Tabs	12
1.6.2	Removing Read Data from the Project Sub-Tabs	15
1.6.3	Specifying Multiplex Identifiers (MIDs)	15
1.7	Customize Project with the Parameters Tab	17
1.7.1	Genomic Project Input Sub-Tab	18
1.7.2	Project Computation Sub-Tab	19
1.7.3	Genomic Project Output Sub-Tab	21
1.7.4	cDNA Project Input Sub-Tab	24
1.7.5	cDNA Project Output Sub-Tab	24
1.8	Computing the Assembly	25
1.8.1	Computing an Assembly for the First Time	25
1.8.2	Re-Computing an Assembly.....	25
1.8.3	Stopping an Assembly Computation	25
1.8.4	Information Updated when Assembly Completes.....	25
1.9	Viewing Assembly Output with the Result Files Tab	26
1.10	Viewing Contigs with the Alignment Results Tab	28
1.11	The Flowgrams Tab	31
1.12	Project Error Indicators.....	32
1.13	The GS <i>De Novo</i> Assembler Command Line Interface.....	32
1.13.1	Working with Project Folders and Data Files.....	33
1.13.2	One-Step Assembly: the runAssembly Command	34
1.13.3	Incremental Assembly: the newAssembly and Related Commands	35
1.13.3.1	The newAssembly Command.....	36
1.13.3.2	The addRun Command	36
1.13.3.3	The removeRun Command	37

1.13.3.4	The runProject Command	37
1.14	GS De Novo Assembler cDNA / Transcriptome Options	37
1.15	GS De Novo Assembler Output	39
1.15.1	Output File Specification	39
1.15.1.1	454AlignmentInfo.tsv	41
1.15.1.2	fna and qual files: 454AllContigs, 454LargeContigs, 454Scaffolds files.....	42
1.15.1.3	454ReadStatus.txt	44
1.15.1.4	454TrimStatus.txt	45
1.15.1.5	454Contigs.ace or ace/ContigName.ace or consed/...	45
1.15.1.6	454NewblerMetrics.txt	48
1.15.1.7	454NewblerProgress.txt	52
1.15.1.8	454PairAlign.txt	52
1.15.1.9	454PairStatus.txt	52
1.15.1.10	454TagPairAlign.txt	53
1.15.1.11	454Scaffolds.txt	54
1.15.1.12	454ContigGraph.txt	54
1.15.2	GS De Novo Assembler cDNA / Transcriptome Output	55
1.15.2.1	454Isotigs.fna, 454Isotigs.qual, and 454Isotigs.txt files	55
1.15.2.2	454Isotigs.ace and consed/...	56
1.15.2.3	454NewblerMetrics.txt	56
1.15.2.4	454RefLink.txt.....	57
1.15.2.5	454IsotigsLayout.txt.....	58
2.	GS Reference Mapper	60
2.1	Overview of the GS Reference Mapper	60
2.2	GS Reference Mapper GUI	61
2.3	Launching the GS Reference Mapper GUI	62
2.3.1	The Main Buttons, Status Area and Progress Box	62
2.3.2	The Quick Start and Documentation Text Links	63
2.4	Opening a Project.....	63
2.4.1	Creating a New Project	63
2.4.2	Open an Existing Project.....	64
2.5	View Project Summary with the Overview Tab	65
2.6	Add/Remove Read Data and References with the Project Tab	66
2.6.1	GS Reads, References and FASTA Reads Sub-Tabs	67
2.6.2	Removing Read or Reference Data from the project Sub-Tabs.....	70
2.6.3	Specifying Multiplex Identifiers (MIDs)	70
2.7	Customize Project with the Parameters Tab	72
2.7.1	Genomic Project Input Sub-Tab	73
2.7.2	Project Computation Sub-Tab	75
2.7.3	Project Output Sub-Tab.....	78
2.7.4	cDNA Project Input Sub-Tab	80
2.8	Computing the Mapping	80
2.8.1	Computing a Mapping for the First Time	80
2.8.2	Re-Computing a Mapping	80
2.8.3	Stopping a Mapping Computation	81
2.8.4	Information Updated when Mapping Completes	81
2.9	Viewing Mapping Output with the Result Files Tab.....	82
2.10	Using the Variants Tab.....	83
2.10.1	The HC Diff's Sub-Tab	84
2.10.2	The Structural Variations Sub-Tab	85

2.11	Profile Tab	87
2.12	Viewing Reads Mapped to the Reference with the Alignment Results Tab	89
2.13	The Flowgrams Tab	93
2.14	Project Error Indicators.....	94
2.15	GS Reference Mapper Command Line Interface	94
2.15.1	Working with Project Folders and Data Files.....	95
2.15.2	One-Step Mapping: the runMapping Command.....	96
2.15.3	Incremental Mapping: the newMapping and Related Commands.....	99
2.15.3.1	The newMapping Command	100
2.15.3.2	The setRef Command	100
2.15.3.3	The addRun Command	100
2.15.3.4	The removeRun Command	101
2.15.3.5	The runProject Command	102
2.16	GS Reference Mapper cDNA / Transcriptome Options.....	102
2.16.1	Annotation Input Files for Mapping Transcriptomes	102
2.17	GS Reference Mapper Output.....	103
2.17.1	Output File Descriptions	103
2.17.1.1	454AlignmentInfo.tsv	106
2.17.1.2	fna and qual files: 454AllContigs, 454LargeContigs	107
2.17.1.3	454ReadStatus.txt	108
2.17.1.4	454TrimStatus.txt	109
2.17.1.5	454Contigs.ace or ace/ContigName.ace or consed/...	110
2.17.1.6	NewblerMetrics.txt	112
2.17.1.7	454NewblerProgress.txt	116
2.17.1.8	454PairAlign.txt	116
2.17.1.9	454PairStatus.txt	117
2.17.1.10	454TagPairAlign.txt	118
2.17.1.11	454MappingQC.xls	118
2.17.1.12	454RefStatus.txt	123
2.17.1.13	454AllDiffs.txt.....	123
2.17.1.14	454HCDiffs.txt	125
2.17.1.15	454HCStructVars.txt and 454AllStructVars.txt	125
2.17.1.15.1	Rearrangement Points	126
2.17.1.15.2	Rearrangement Regions	127
2.17.2	GS Reference Mapper cDNA / Transcriptome Output	128
2.17.2.1	454GeneStatus.txt.....	128
2.17.2.2	454MappingQC.xls	128
3.	SFF Tools Commands	130
3.1	sfffile.....	130
3.1.1	Using sfffile for Evaluation of Individual Reads	133
3.1.2	Using sfffile to Extract Read Subsets	134
3.2	sffinfo.....	135
3.3	sff2scf.....	136
3.4	fnafile.....	138
3.5	sffrescore.....	140
4.	GS De Novo Assembler and GS Reference Mapper Appendices	142
4.1	Options Common to Mapping and Assembly	142
4.2	Options Specific to Assembly	147
4.3	Options Specific to Mapping.....	147

4.4	Options for the addRun Command.....	148
4.5	Mutually Exclusive Options for Assembly and Mapping Commands	149
4.6	Multiplex Identifiers.....	149
4.6.1	The Use of Multiplex Identifiers in the Data Analysis (MIDs).....	149
4.6.2	The MIDConfig.parse File	151
4.7	Paired End Libraries in the Genome Sequencer FLX System	153
4.7.1	Paired End Read Naming Convention.....	153
4.7.2	Ends of Paired End Reads Shorter Than 50 bp (Tags).....	154
4.7.3	Paired End Library Span Estimation.....	154
4.7.4	True Pairs and False Pairs	154
4.8	Trimming and Screening Files.....	155
4.9	Accno Renaming File	155
4.10	Assembling with Reads Obtained Using the Sanger Sequencing Method.....	156
4.10.1	Simple Sanger Reads	156
4.10.2	Sanger Reads with available Quality Scores.....	157
4.10.3	Sanger Read Annotations	157
4.10.3.1	The “template”, “dir”, and “ library” Annotations.....	157
4.10.3.2	The “trim” Annotation.....	158
4.10.3.3	The “scf” and “phd” Annotations	158
4.10.4	Recommended Method for Including Sanger Reads.....	158
4.11	Using the Flowgrams Tab	160
4.12	Project Error Indicators.....	163
5.	cDNA / Transcriptome Sequencing Appendix.....	165
5.1	Introduction to cDNA Sequencing Analysis.....	165
5.2	Transcriptome Assembly Concepts.....	165
5.2.1	Definitions.....	165
5.2.1.1	Isogroup.....	165
5.2.1.2	Isotig.....	165
5.2.2	Rules for Path Traversal of Contigs in an Isogroup.....	166
5.2.2.1	Path Initiation.....	166
5.2.2.1.1	Spike Detection	167
5.2.2.2	Path Extension	168
5.2.2.3	Path Termination	168
5.2.2.4	Other Rules that Prevent Path Traversal.....	168
5.3	Transcriptome Mapping Concepts	168
5.3.1	Mapping cDNA Reads to a Transcriptomic Reference.....	168
5.3.2	Mapping cDNA Reads to a Genomic Reference	169

1. GS *DE NOVO* ASSEMBLER

1.1 Overview of the GS *De Novo* Assembler

The GS *De Novo* Assembler application constructs *de novo* assemblies of the reads from one or more sequencing Runs, using the “read flowgrams” (SFF files) as input. The GS *De Novo* Assembler software is an interactive application used to create assembly projects, add and remove reads from the project, specify project assembly parameters, run the assembly algorithms on the project data, and view the output produced by the assembly computations.

The input data can come from one, several or all of the PicoTiterPlate regions of the Run(s) of interest. Assembly generates a consensus sequence of the sample DNA library, output as one or more contiguous sequences (contigs) in FASTA, ACE or consed files.

Input data can also come from one or more FASTA files of reads, such as reads obtained using the Sanger sequencing method (“Sanger reads”). The longer read length information in these files can be used to improve the construction of contigs and scaffolds.

The GS *De Novo* Assembler allows the inclusion of one or more Paired End Runs into the analysis, enabling the ordering and orientation of the assembled contigs from shotgun sequencing Runs into scaffolds; this requires the preparation of a separate Paired End library from the same DNA material that was used to prepare the library for shotgun sequencing. In certain cases, the presence of reads from a separate shotgun library in the project is not strictly required, as Paired End reads can themselves be assembled with each other.

The GS *De Novo* Assembler application can be accessed via a Graphical User Interface (GUI) or from a command line interface (CLI). The GUI also allows the user to view the output produced by the assembly.

During the assembly process, the software:

- Identifies pairwise overlaps between reads
- Constructs multiple alignments of overlapping reads and divides or introduces breaks into the multiple alignments in regions where consistent differences are found between different sets of reads. (This step results in a preliminary set of “contigs” that represent the assembled reads.)
- Attempts to resolve branching structures between contigs
- Generates consensus basecalls of the contigs by using quality and flow signal information for each nucleotide flow included in the contigs’ multiple alignments
- Outputs the contig consensus sequences and corresponding quality scores, along with an ACE file of the multiple alignments and assembly metrics files.

When Paired End data is available, the assembler performs these extra steps:

- Organizes the contigs into scaffolds using Paired End information to order and orient the contigs and to approximate the distance between contigs

- Outputs scaffolded consensus sequences and corresponding quality scores, along with an AGP file of the scaffolds and specific metrics Tables.

Read overlaps and multiple alignments are made in “nucleotide” space while the consensus basecalling and quality value determination for contigs are performed in “flowspace”. Work in flowspace allows the quality-weighted averaging of processed flow signals (a continuous variable) at each nucleotide flow of the sequencing Run(s) and allows the use of information from the “negative flows”, *i.e.* flows where no nucleotide incorporation is detected. The use of flowspace in determining the properties of the consensus sequence results in an improved accuracy for the final basecalls.

The assembler produces contigs from the multiple alignments of overlapping read sequences. The GUI (Graphical User Interface) provides tools that allow the contig consensus sequences, the multiple alignments of the reads that form the contig, and the flowgrams of these reads to be viewed interactively. The CLI (Command Line Interface) produces files containing the output produced by the assembly. The GUI can display the contents of these files.



- The GS *De Novo* Assembler application is not available on the Genome Sequencer FLX Instrument and must be run on a DataRig.
- Because of the inherent complexity of large genome assembly, we recommend contacting your Roche representative prior to assembling any project where the expected genome size exceeds 500Mbp.

The GS *De Novo* Assembler allows users to create, modify, and run assemblies in the form of projects. Both the GUI and command line interface (CLI) provide this functionality. Projects may be setup to assemble all reads at once. Alternatively, incremental operation allows additional reads to be added to an existing assembly. Results appear as output files using either the GUI or the CLI. The GUI provides a graphical interface to view many of the results from the assembly whether the project was assembled using the GUI or the CLI.

The GS *De Novo* Assembler application uses a folder on the file system to hold the assembly project information (whether the assembly of the reads, the computation, is carried out in project-based mode through the GUI application or through the newAssembly and related commands) and to hold the output files generated during and after the assembly computation.

The operation of the GUI is described in several of the subsequent sections. A description of the CLI is then presented followed by a discussion of transcriptome assembly. Finally, output files produced by the GS *De Novo* Assembler are described.

1.2 GS *De Novo* Assembler GUI

Assemblies can be performed or viewed using the Graphical User Interface application, gsAssembly described in the following sections. The application includes graphical interfaces to:

- create new assembly projects
- open existing assembly projects
- Add/remove reads to/from assembly projects
- Modify assembly input, computation, and output parameters

- carry out an assembly computation
- view the results of a completed assembly
- view the progress and logging information of an assembly computation

1.3 Launching the GS *De Novo* Assembler GUI Application

The following command is used to launch the GS *De Novo* Assembler GUI application (from a Linux terminal window on a DataRig where the data analysis software package is installed):

```
gsAssembler
```

Once the GUI is launched (Figure 1), a user can open an existing project or create a new project.

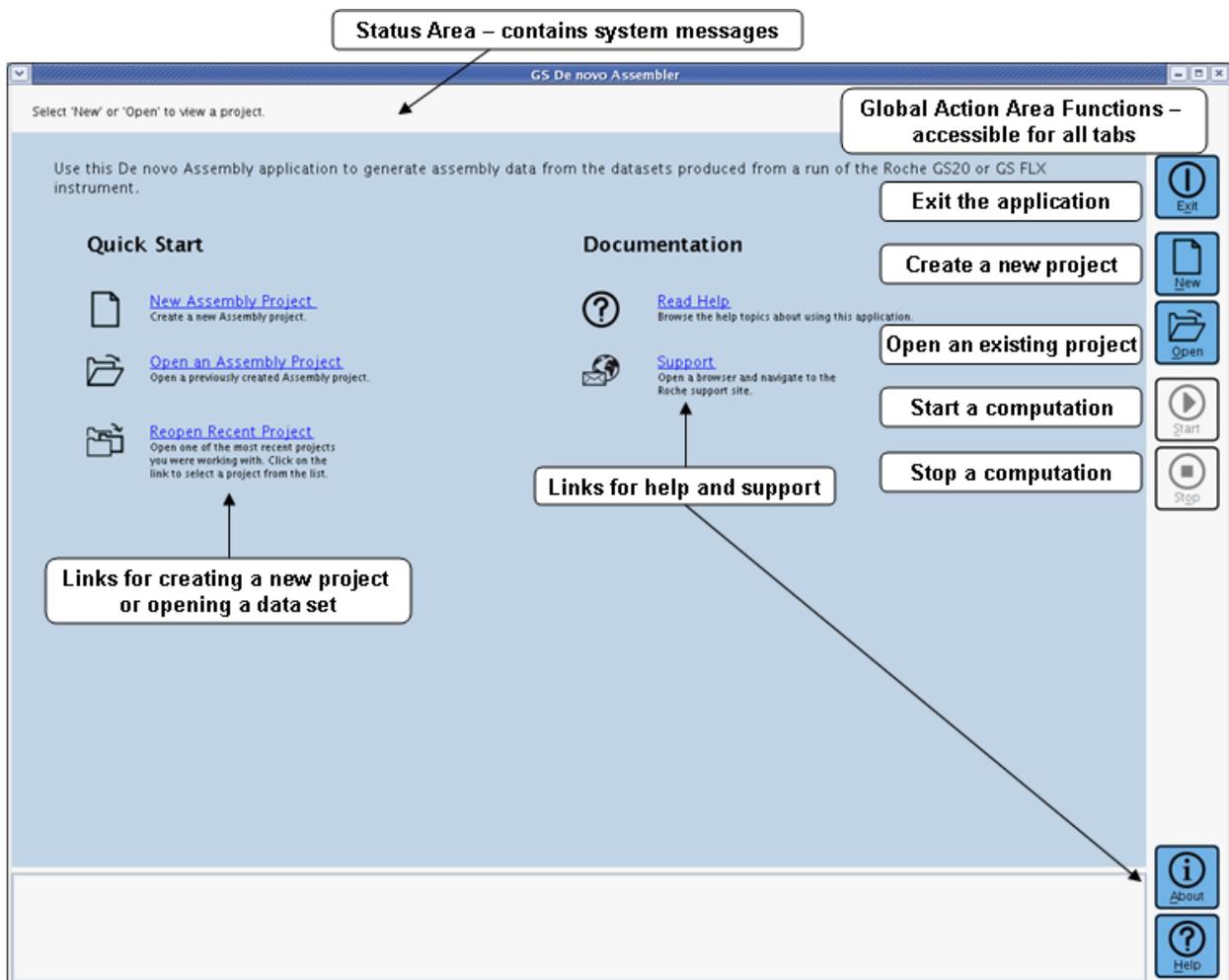


Figure 1: Initial Interface of the gsAssembler

1.3.1 The Main Buttons, Status Area and Progress Box

Seven main buttons are always visible on the toolbar, along the right-hand side of the GS *De Novo* Assembler's main window:

- The **Exit** button closes the GS *De Novo* Assembler application
- The **New** button allows one to create a new assembly project
- The **Open** button allows one to open an existing assembly project
- The **Start** button begins the assembly (computation) of an open assembly project
- The **Stop** button halts the execution of an ongoing assembly computation
- The **About** button shows the GS *De Novo* Assembler splash screen
- The **Help** button shows a help dialog (online help is not currently implemented)

The top of the main GS *De Novo* Assembler window contains a status area that displays a command hint, while the bottom panel is a progress box which remains blank until an assembly project has been opened.



The progress box at the bottom of the window can be resized by dragging its top frame separator. If the field is not visible, it is probably simply collapsed; you can view it by dragging up the edge of the collapsed frame.

1.3.2 The Quick Start and Documentation Text Links

There are 3 text links under the **Quick Start** column:

- The **New Assembly Project** text link creates a new assembly project. It performs the same action as the **New** button on the right side of the window.
- The **Open an Assembly Project** text link opens an existing project. It performs the same action as the **Open** button on the right side of the window.
- The **Reopen Recent Project** text link displays a dialog from which you can open a project on which you worked recently.

There are 2 text links under the **Documentation** column:

- The **Read Help** text link shows a help dialog. Online help is not currently implemented; please use this Software Manual, or click on the Support button (see below), or contact your Roche Representative for any information regarding the Genome Sequencer FLX System and its software package. The Read Help text button performs the same action as the **Help** button on the right side of the window.
- The **Support** text link opens the default internet browser and navigates to the Roche support site.

1.4 Opening a Project

1.4.1 Creating a New Project

To create a new assembly project, either click on the **New Project** button in the right toolbar, or click on the “New Assembly Project” text button in the **Quick Start** column. This displays a dialog (Figure 2) in which you can specify the name and directory location for a new project.

Type a name for the new project in the **Name** text field. To specify the project location, either type the full path in the **Location** text field, or click on the **Open Project** button to the right of the text field and use the “Select Project Location” window (not shown) to navigate to the directory where you want to create the new project. The **Full Path** field changes as you update the **Name** and **Location** fields. When you are satisfied with the location and name for the new project, click on the **OK** button. You can save a project by clicking on the Yes button to the **Save** prompt.



You may save the project to your hard drive either by using the Exit button to exit the GS *De Novo* Assembler application (you will be prompted to save before the application actually exits) or by adding read data and running the project by clicking the Start button (i.e., compute the assembly), in which case the project will automatically be saved prior to computation, as described in section 1.8.1, below.

The New Project dialog contains a drop-down menu (Figure 2) to specify the **Sequence type** indicating the type of DNA library sequenced, cDNA or Genomic. This choice of Sequence type determines many of the parameters the user will be allowed to modify in the project. Once a project is created the type cannot be changed.

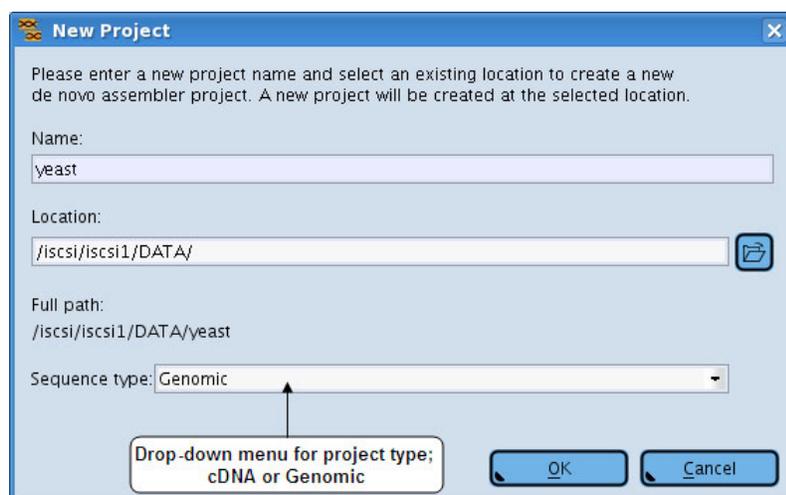


Figure 2: New Project dialog used to specify Sequence type

1.4.2 Open an Existing Project

To open an existing assembly project, either click on the **Open Project** button in the right toolbar, or click on the “Open an Assembly Project” text button in the **Quick Start** column. This displays a dialog (Figure 3) in which you can select the name and specify the directory location

of the project to be opened. By default, only 454 Assembly Projects will be displayed. You may choose to display all files by selecting “All Files” from the “Files of Type” dropdown menu.

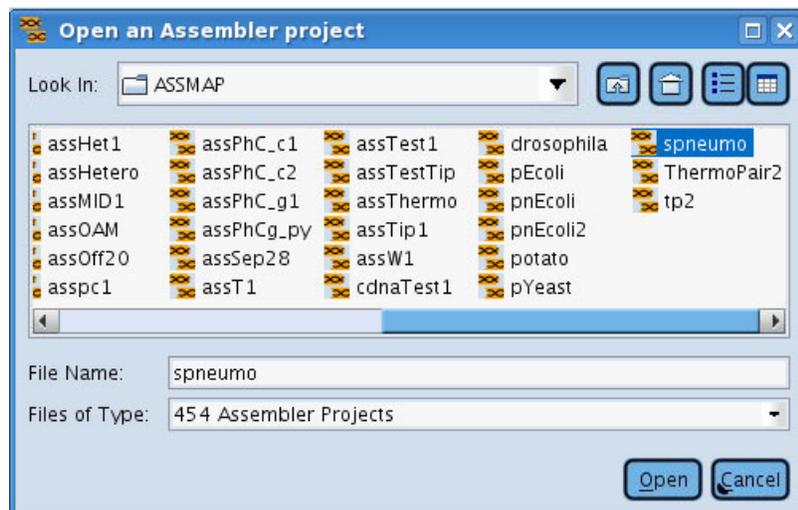


Figure 3: Open Assembly Project dialog

1.5 View Project Summary with the Overview Tab

When a project opens, the Overview Tab is displayed (Figure 4). Other tabs can be selected with the mouse. Some tabs may remain inactive (grayed-out), until the type of information they require is available for the project. The Overview Tab’s **Project Summary Information** area indicates the Sequence type along with other information that applies to the overall project. The data displayed on this tab is updated as new information becomes available, which occurs when data files are added or removed from the project and when a project computation completes.

When a project is first created, the **Computation status message** in the upper right hand corner of the application window indicates “Not ready for Analysis” and the Start button is disabled because at least one Read Data file must be added to the project before an Assembly computation can be performed. Other errors in input parameters can also cause this message to appear. Once all such errors have been corrected and at least one Read Data file is added to the project and all options have valid values, the message will change to “Ready for analysis” and the Start button will become active

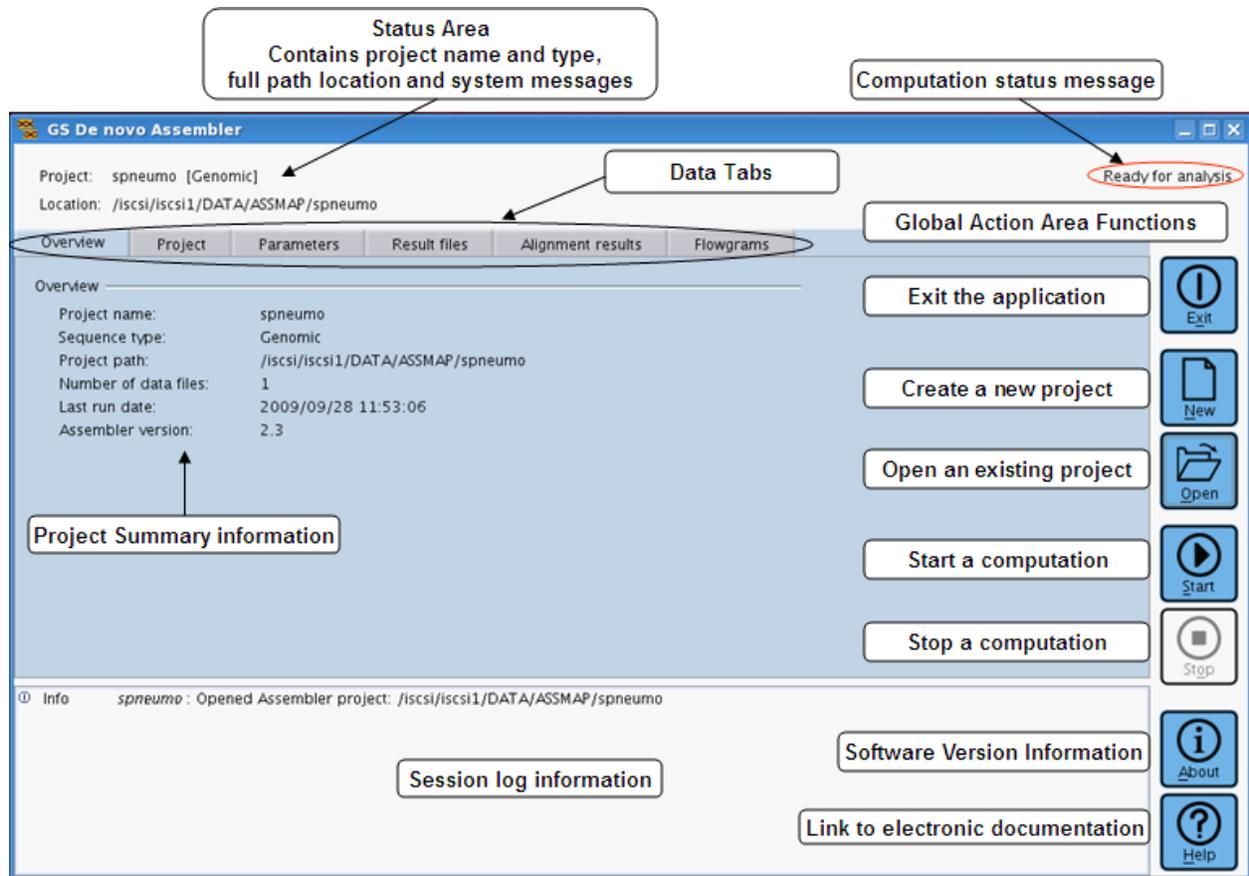


Figure 4: gsAssembler Overview Tab

1.6 Add/Remove Read Data with the Project Tab

Read Data files are added to a project on the Project tab of the main application window. There are two sub-tabs for adding read data: one for adding GS reads (GS Reads sub-tab) and one for adding non-GS reads, such as Sanger reads, in FASTA format (FASTA Reads sub-tab).



The **Start** button in the toolbar, which is used to start an assembly, remains disabled until at least one Read Data file (either GS read or FASTA read) has been added to the project.

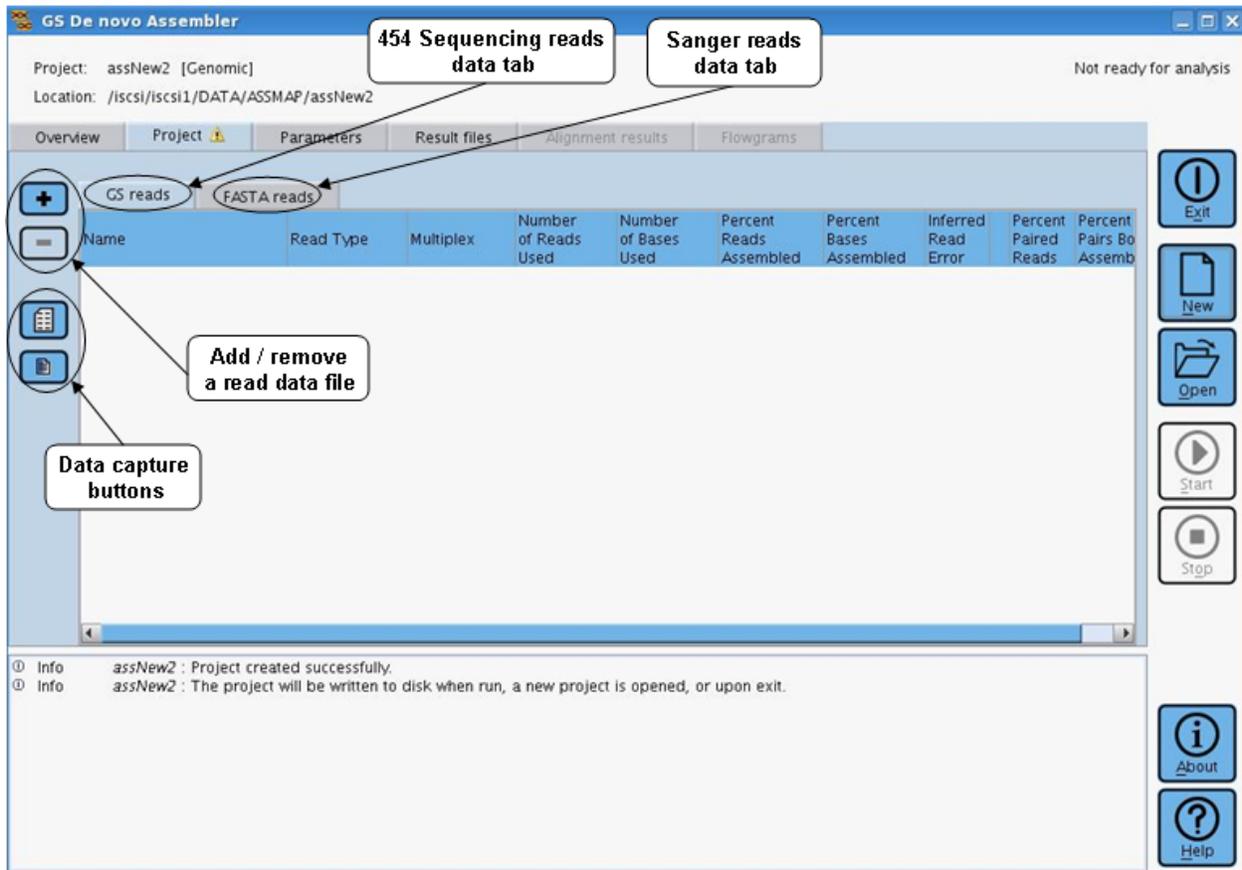


Figure 5: Project Tab of the gsAssembler (GS Reads sub-tab)

1.6.1 GS Reads and FASTA Reads Sub-Tabs

To add GS read data sets to a project, click on the GS reads sub-tab, then click the **Add** button  to open the “Select GS Read Data” window (Figure 6). The top portion of this window lists the GS Read Data files that can be added to the project; if no files exist at this location, the window displays the message: “There are no GS Read Data files in the selected directory”. More than one file can be selected for import. The GS Read files can be explicitly specified as Paired End or non-Paired End, or the user can invoke auto-detection using the **Read Type Specification** dropdown. When the read type is known, it is advisable to specify it directly rather than use auto-detect as the auto-detect feature, on rare occasion, may fail to detect a Paired End file.



When planning to run an incremental assembly, it is best to first add the shotgun reads, and then add reads from Paired End libraries with increasing insert spans. This is because longer contigs can be assembled using the longer shotgun reads. These longer contigs then have a better chance of having both ends of paired end reads aligned within them. This in turn allows more robust library span estimates to be made (see section 4.7)

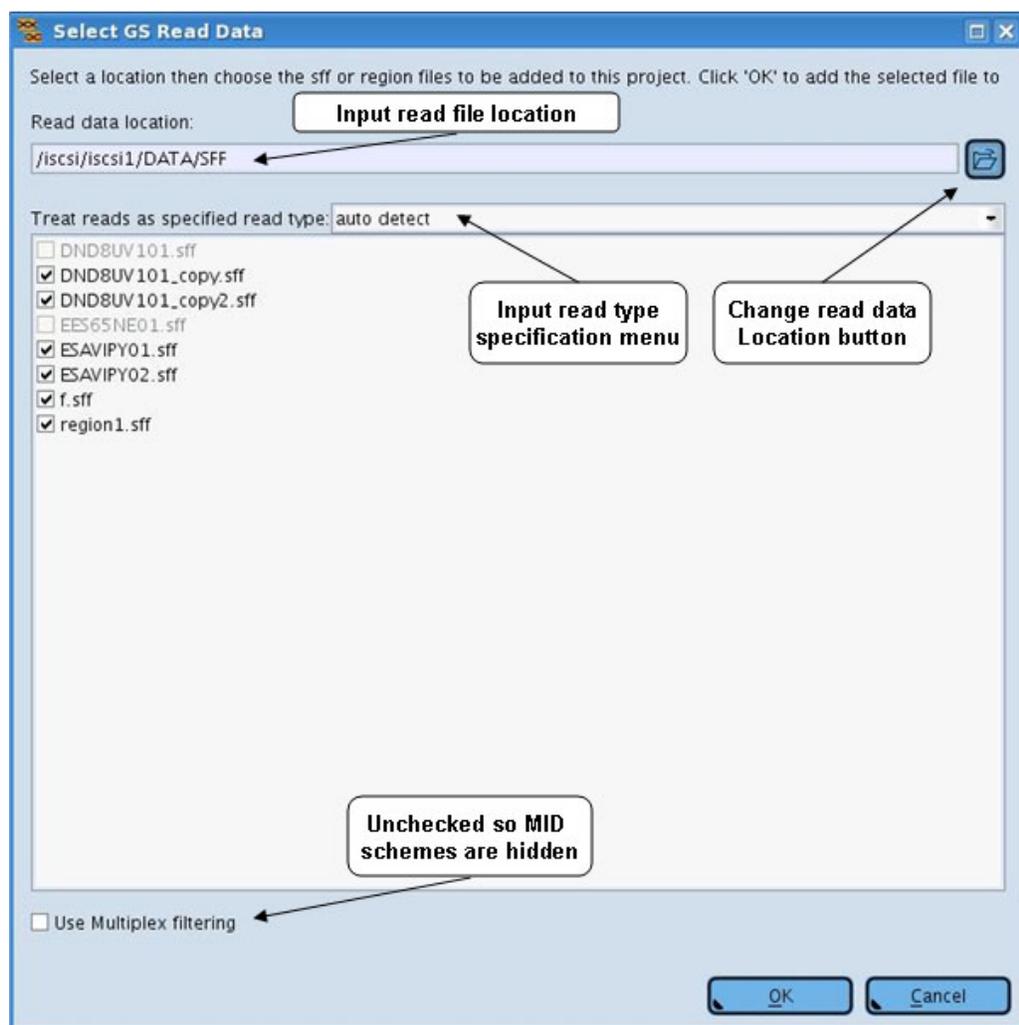


Figure 6: Select GS Read Data dialog

To change to a different location that contains Read Data files, click on the “**Change the read data location**” button (with the open-folder icon) to the right of the text field and use the “Select Read Data Directory” window (not shown) to navigate to and select the new location. You may also directly type or paste the desired path into the “File Name:” field of the navigation window, rather than browsing through the file system for the directory of interest. While using this interface, only directories will appear, not the files within those directories. When you return from the Select GS Read Data window, the list of available Read Data files found at the new location will be refreshed and appear in the field below the Read data location text area (as seen in Figure 6). Use the check boxes to select the file(s) you want to include in the assembly project.

You can add any number of Read Data files to a project using the Add button, navigating to a folder, then selecting the files needed from the folder. You can even add multiple files that share the same name into a project, as long as they have different path locations in the file system (e.g. “/dir1/path1/reads.sff” and “/dir2/otherpath2/reads.sff”). In such a case, both files will be added to the Read Data list and displayed using the same name. To see the path to a file listed in the Read Data area of the main window (and the file’s last modification date), pause the mouse over the filename of interest and a tooltip containing the file’s path will be displayed. Files that have failed validation will have a red X left of the file name as an indication of failure.

Pause the mouse cursor over the red **X** to bring up a tooltip explaining the problem encountered.

The FASTA Reads sub-tab is functionally similar to the GS Reads sub-tab for all project types. The only difference is that when a directory containing FASTA files is selected, the software examines the files in that directory to determine which files are FASTA files. This can take some time if there are many files in the directory, so a progress bar is displayed to show the progress of the search.



Order of addition of Read data may affect assembly results: In general, the reads should be added to incremental assemblies in the following order to achieve the best contigging and scaffolding results:

- Shotgun
- 3kb Paired End
- 8kb Paired End
- 20kb Paired End

Except for the Name column and the Multiplex column (which contains comma-delimited lists of the MIDs associated with the file; see Section 1.6.3, below), all columns with run statistics are initially filled with dashes. These data are updated in the table each time the project runs to completion. For a project that has already been through an assembly computation, summary statistics relating to the usage of the reads in the assembly process is also listed, as shown in Figure 15. On the left hand side of the reads table are two data capture buttons that can be used to capture the reads table data in tsv or text formats.

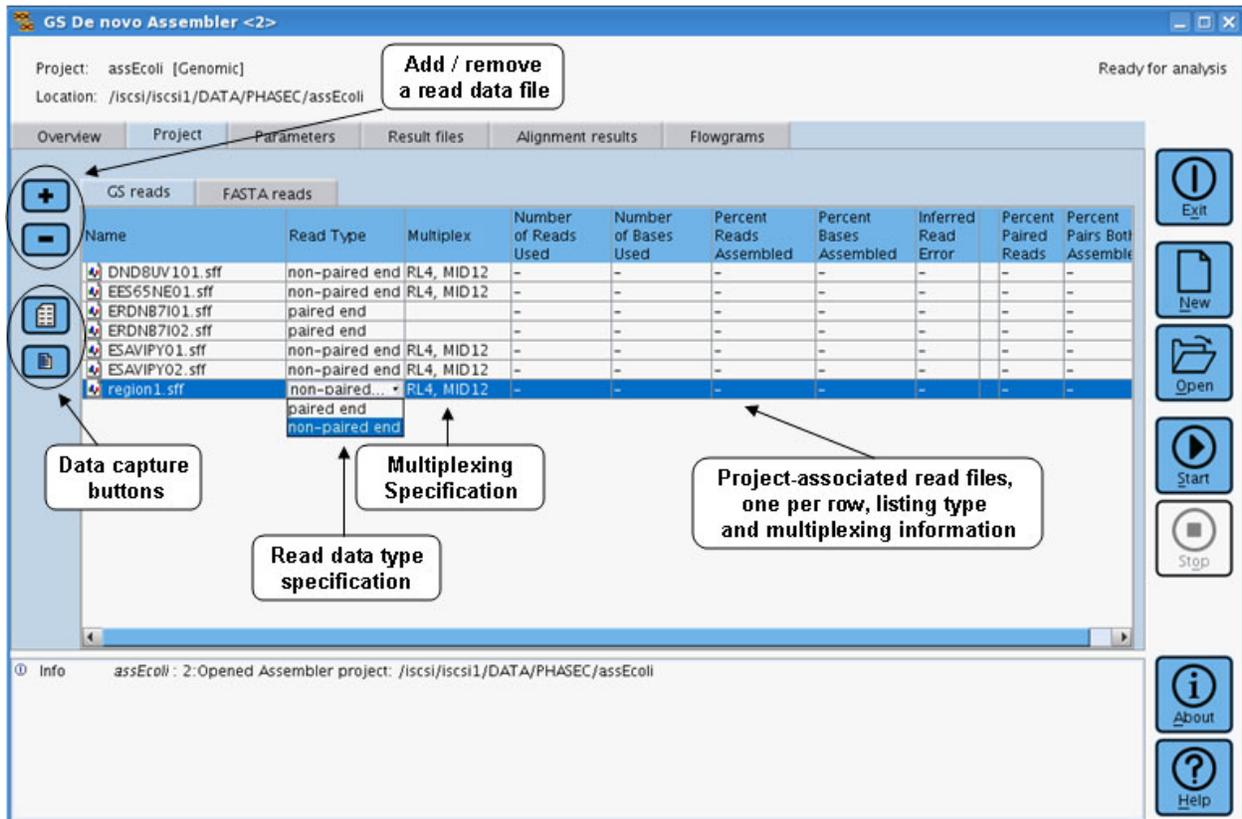


Figure 7: Project Tab of the gsAssembler (GS Reads sub-tab), after adding several read data files

1.6.2 Removing Read Data from the Project Sub-Tabs

To delete read data from a project, click on the appropriate sub-tab, then click on a read file(s), then click the **Remove** button .



The **Remove** button removes the selected Read Data files from the list, but does not immediately remove the file(s) from the project's sff sub-directory or from any computed assembly results. Actual removal of the files from the project (and of the reads they contain from an existing alignments computed therefrom), occurs only when the assembly is re-computed (see section 1.8.2)

1.6.3 Specifying Multiplex Identifiers (MIDs)

When the "Use Multiplex Filtering" checkbox is selected, special controls for MID filtering are displayed in the bottom half of the Select GS Read Data window (Figure 8). Multiplex Identifiers (MIDs) allow you to design an experiment whereby multiple libraries are prepared using distinct MID tags and sequenced together, in the same region of a PTP device. (This can greatly improve the workflow and cost effectiveness of your experiment.)

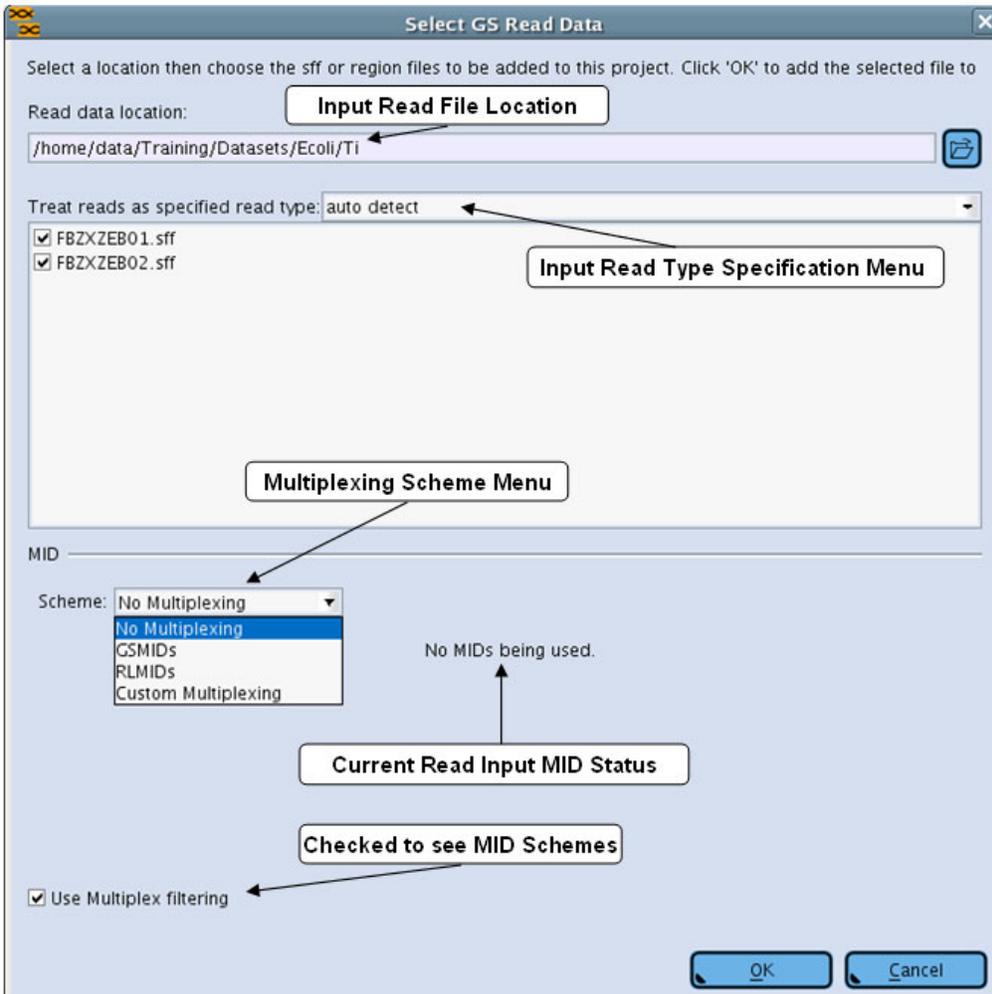


Figure 8: Select GS Read Data dialog with the MID Scheme options expanded

The MID controls on this window allow you to filter the reads in the selected Read Data files for inclusion in the Assembly project. Note, this filtering operation does not produce a different Assembly for each specified MID. Rather, the reads of the SFF files are scanned for the presence of MID's, and the reads containing the selected MID's are made eligible for assembly in the project. The selected MID filtering criteria are associated with the sff files when the OK button is clicked (Figure 8). Any reads from these file(s) without the specified MID's will be ignored. To produce independent Assemblies for different MID's (or sets of MID's), you must create independent Assembly projects for each of the desired MID subsets of the data.

MID filtering information is contained in 'MID schemes'. The choices are 'No Multiplexing', 'GSMIDs' for use of the original Genome Sequencer MID sets, 'RLMIDs' for the Rapid Library Preparation Kit MID's, and 'Custom Multiplexing' if custom MID's were used. By default, no MID/Multiplexing scheme is associated with Read Data files. To use MID's, first select the desired MID Scheme (Figure 8) using the **Scheme** drop down menu. When an MID scheme is selected, a table of the MID's present in that scheme is displayed, as shown in Figure 9. Use the checkboxes on the left to select or deselect the MID's to be included in the assembly. The names, sequences and error limits of the MID's selected will be used to filter the reads (from the

Read Data sets selected in the top part of the window). The result is that only the reads that contain the selected MIDs, within the number of errors specified in the scheme, will be included in the project.

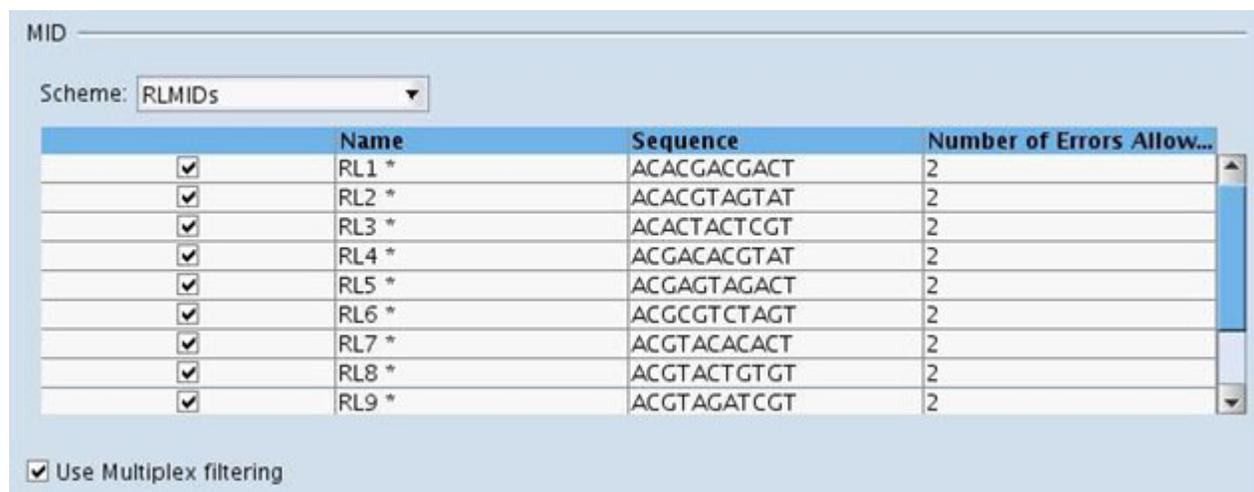


Figure 9: GS Read Import Dialog Rapid Library MID Support

For more information about MIDs, see Section 4.6.

Once the MID scheme has been specified, click the **OK** button to add the selected data files (with MID filtration information) to the list of GS read data files (see Figure 7).



In certain circumstances, you may want to apply a different MID selection to different Read Data sets, within the same project. For example, you may have multiple libraries of the same organism, made with different MIDs (or with/without MIDs), that you want to assemble together. In this case, select the Read Data set(s) that need to be filtered with any given MID (or combination of MIDs), and click **OK** to add the MID-filtered data file(s) to the project. Back on the GS Reads sub-tab of the GS *De Novo* Assembler window (with the GS Reads sub-tab active) (Figure 5), click the **Add** button (+) again, and select other Read Data sets, to be filtered with different MIDs.

1.7 Customize Project with the Parameters Tab

The Parameters Tab is organized as three sub-tabs: Input, Computation and Output (Figure 10). The input parameters and output data options that can be specified are different for the Genomic and cDNA sequence types. Each sequence type will have its own set of parameters under the project sub-tabs.

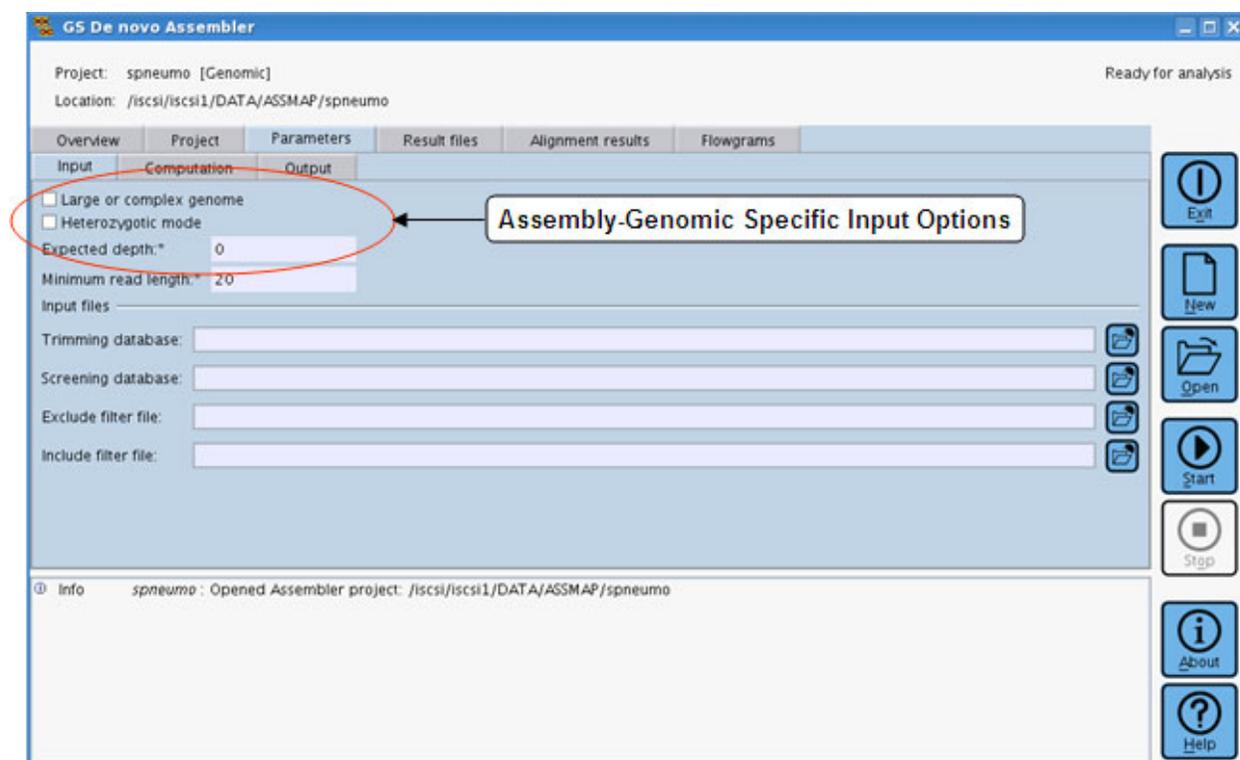


Figure 10: gsAssembler Parameters tab, Input sub-tab for Genomic projects

For a new project, the assembly parameters that take numerical values are initially set with their default values, as shown. If an invalid value is entered into any of these fields, a small red “X” appears in the lower left corner of the field and activates the “Not ready for analysis” warning in the upper right hand corner of the screen. Pausing the mouse over the red X reveals a tooltip indicating the allowed values for the parameter (see Section 1.12).

While default parameter values and settings are appropriate for most genomic and transcriptomic assemblies, you may find some options useful for specific projects.

1.7.1 Genomic Project Input Sub-Tab

The Input sub-tab is shown on Figure 10. It allows you to adjust the following settings:

- The **Large or complex genome** option should be used when large or complex datasets (*i.e.* eukaryotic genomes) are being assembled. This will invoke algorithms especially suited for such data sets, allowing successful and speedy assembly. This option:
 - Skips assembly of high-copy count (> 500) repeats
 - Sets the “seed count” to 2 during the assembly computation (see Section 1.7.2)
 - Skips the last level of detangling
 - May result in 10% more contigs than might be obtained without this option
 - Usually should not be used in cDNA assembly projects (unless a project’s computation won’t complete without using the option).

- The **Heterozygotic mode** option is used to specify that the project's read data is from a diploid or non-inbred organism. This prompts the assembler to adjust the algorithms it uses to reflect an increase in the expected variability in sequence identity.
- The **Expected depth** option allows you to specify the expected depth of coverage in the assembly. For high-depth assemblies (where the expected coverage of a position in the genome could reach hundreds or thousands of reads), this option can be used to filter out random-chance events that would be considered significant against a lower depth background.
 - Default: 0
 - Allowed values: 0 or greater, where a value of 0 tells the assembler to not use expected depth information in its computation
- The **Minimum read length** option can be used to change the minimum accepted tag/read length allowed in the assembly. For projects that use any Paired End data, this option sets the minimum length for reads or tags (see discussion of PE reads in section 4.7.2) to be used in the assembly (the default is 20 bp, the allowed value range is 15-45). In such projects, 454NewblerMetrics.txt will report the value of numberTooShort as 0 since any shotgun reads at least as long as the minimum read length will be used in the assembly. For projects with only shotgun read data available, a minimum of 50 bp is required and this cannot be changed using this option.

The **input files** section of the tab allows you to specify the locations of several files that may be used in the assembly.

- An optional **Trimming database** is used to trim the ends of input reads (for cloning vectors, primers, adapters or other end sequences). Specify the path to a FASTA file of sequences to be used for this trimming (see Section 4.8).
- To use the **Screening database** option, set the path to a FASTA file of sequences to be used to screen the input reads for contaminants. A read that almost completely aligns against a sequence in the screening database is removed so that it is not used in the computation; if at least 15 bases of a read do not align to the screening sequence, no action is taken (see Section 4.8).
- The **Include** and **exclude filter file** options can be used to specify the path to a file of sequences to specifically include or exclude in the computation. The file format and functionality are the same as the `-e` and `-i` options of `sfffile` (see Section 3.1).



Long file paths will be displayed in full when you pause the mouse cursor on the text area.

1.7.2 Project Computation Sub-Tab

The Computation sub-tab is shown on Figure 11. It allows you to adjust the settings described below.

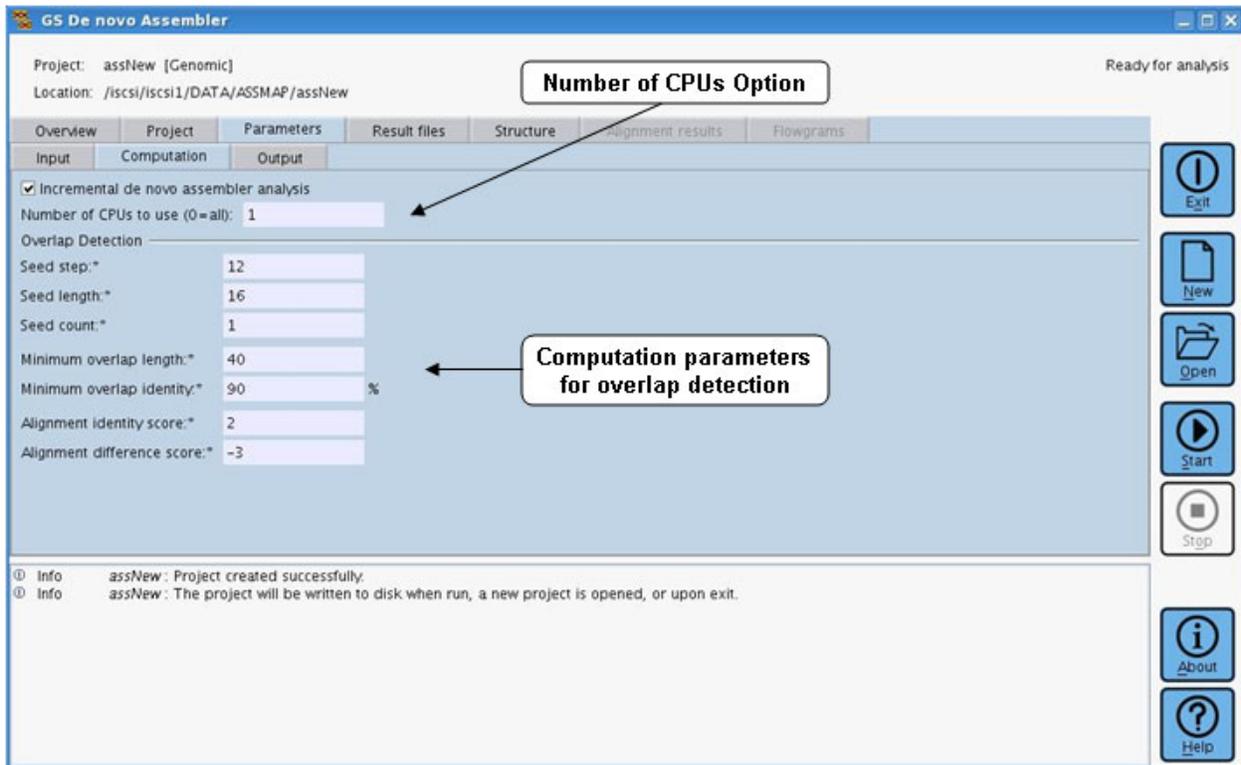


Figure 11: gsAssembler Parameters tab, Computation sub-tab for Genomic projects

- Incremental De Novo assembler analysis** – Results from computations performed with this option selected (checked) will serve as the basis for the assembly of additional reads. If a project computation is performed with this option deselected, all the project data will be assembled anew each time it is computed, i.e. forcing all alignments to be recalculated and overwriting any existing assembly results.
 - Default: selected (Note: This setting is not saved with a project, and is always reset to “selected” when a project is re-opened).
- The number of CPUs** option can be given to limit the load of the software on the computing resources. The default is 1, which specifies that one of the CPUs on the computer should be used. The computation uses the CPU setting to create an approximate load on the computer, so the actual load may vary somewhat from this number, i.e., using “-cpu 3” may cause a load from 2 to 4 on the system.



Currently, only the “Computing alignments...” and “Generating output...” phases have been parallelized (because they are the most computationally intensive phases for larger projects). Other phases of the computation will still use single threading. The parallel assembler only runs on shared memory, i.e. only CPUs with access to the same physical memory can be used.

The current memory footprint is 3 bytes per input (read) base. For example a 1GigaBase genome with a 20x coverage would need 60 GigaBytes of physical memory.

- **Overlap Detection Parameters**
 - **Seed step** – The number of bases between seed generation locations used in the exact k-mer matching part of the overlap detection
 - Default value: 12
 - Allowed values: 1 or greater
 - **Seed length** – The number of bases used for each seed in the exact k-mer matching part of the overlap detection (*i.e.* the “k” value of the k-mer matching)
 - Default value: 16
 - Allowed values: 6-16
 - **Seed count** – The number of seeds required in a window before an extension is made
 - Default value: 1
 - Allowed values: 1 or greater
 - **Minimum overlap length** – The minimum length of overlaps used by the assembler for the pairwise alignment step
 - Default value: 40
 - Allowed values: 1 or greater
 - **Minimum overlap identity** – The minimum percent identity of overlaps used by the assembler for the pairwise alignment step
 - Default value: 90
 - Allowed values: 0-100
 - **Alignment identity score** – When multiple overlaps are found, the per-overlap-column identity score used to sort the overlaps for use in the progressive alignment
 - Default value: 2
 - Allowed values: 0 or greater
 - **Alignment difference score** – When multiple overlaps are found, the per-overlap-column difference score used to sort the overlaps for use in the progressive multi-alignment
 - Default value: -3
 - Allowed values: 0 or less

For a description of the overlap detection parameters for assembly, please see Table 4: Mapping and Assembly options, in the Appendix section.

1.7.3 Genomic Project Output Sub-Tab

The Output sub-tab is shown on Figure 12. It allows you to adjust the settings described below.

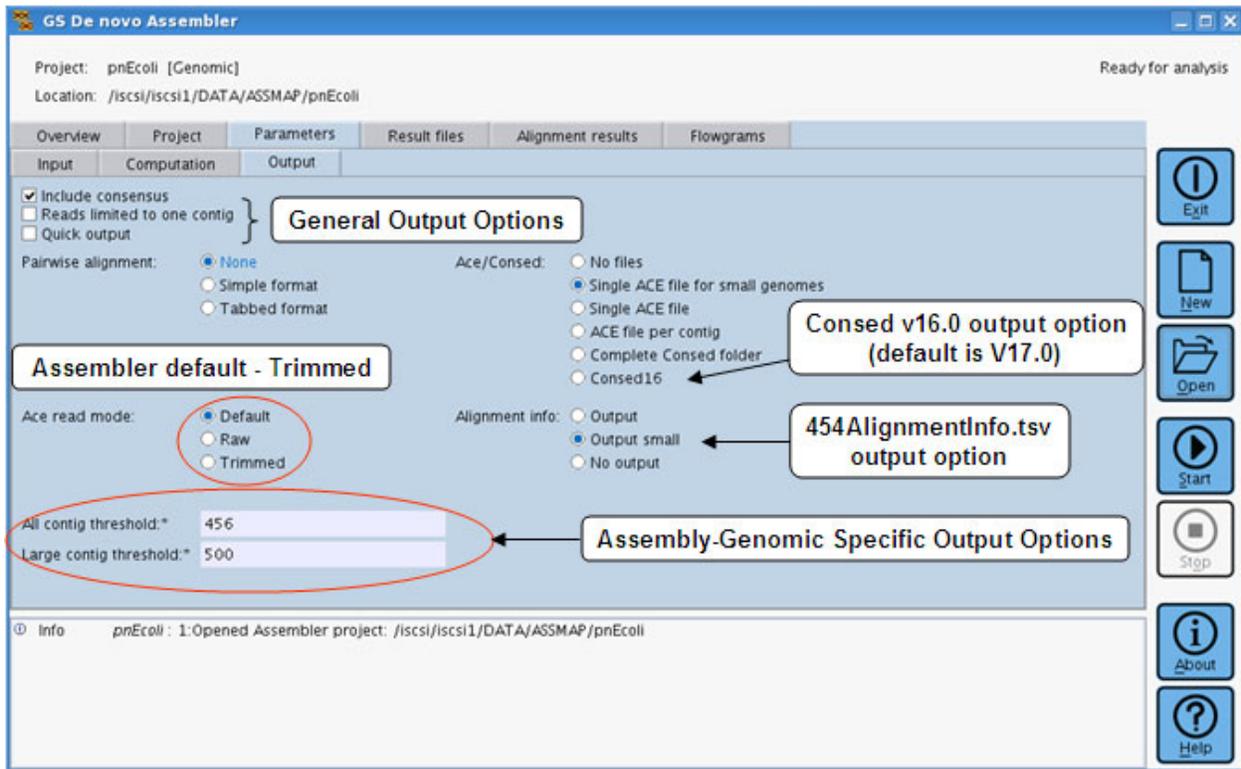


Figure 12: gsAssembler Parameters tab, Output sub-tab for Genomic projects

- **Include consensus** – If this check box is selected, the application will generate all the output files, including the output files related to the generation of contigs and consensus sequence information. If it is not selected, the application will perform the assembly and will output files associated with the creation of the multiple-alignments, but will not output files (454AllContigs.fna and .qual, 454LargeContigs.fna and .qual, scaffold files, etc.) or metrics (latter parts of 454NewblerMetrics.txt) involved with contig or consensus information.
 - Default: selected (Note: This setting is not saved with a project, and is always reset to “selected” when a project is re-opened.)
- The **Reads limited to one contig** option tells the assembler to generate output where each read occurs in one and only one contig (i.e. read alignments are not allowed to span across multiple contigs). This creates output ace and consed files suitable for 3rd party software for further data analysis. The function operates after the contigs and scaffolds are created, by attempting to remove reads from the ends of contigs that contain sets of reads that branch to multiple contigs. When a read is found that extends from such a contig to another contig end that doesn’t branch, the part of the read in the branched contig is removed and placed at the end of the part of the read in the non-branching contig.



Caution: the use of this option results in the loss of information, as connections between unique and repeat contigs are removed.

- The **Quick output** option generates output faster than using the default setting by disabling the “Computing signals...” computation. The default behavior of the assembler requires that all reads be re-read so that signal and quality information can be used to compute consensi (see Section 1.1). If this option is selected, the accuracy of the consensus may be degraded (i.e., more consensus errors may be generated) as the distribution statistics are not generated to assist in the basecalling step.
- The **Ace/Consed** option determines whether or what form of ACE file(s) are output by the application. The default is “Single ACE file for small genomes”. See section 1.15.1.5 for a description of the 454Contigs.ace file and of the ace and consed directories. By default, files consistent with version 17.0 or higher of consed are generated.
 - No files – no ACE file is generated
 - Single ACE file for small genome – a single ACE file is generated if fewer than 4 million reads are input to the assembly
 - Single Ace file – a single ACE file is generated containing the multiple alignments of all the contigs in the assembly, irrespective of the size of the genome
 - ACE file per contig – a separate ACE file is generated for each contig in the assembly
 - Complete Consed folder – a “consed” folder is generated, containing all the directories and files necessary to display the data in the consed software
 - Consed 16 – This option generates files consistent with version 16.0 or earlier
- The **Alignment Info** option has 3 possible settings.
 - **Output** – turns on generation of the 454AlignmentInfo.tsv file
 - **No output** – suppresses the generation of the 454AlignmentInfo.tsv file
 - **Output small** – the file will be generated unless there are more than 4M reads or the assembly generates contigs with a total length in excess of 40Mbp.
- **Pairwise Alignment** – Determines whether the 454PairAlign.txt file is output; this file contains the overlaps used by the assembler.
 - None – the file is not generated
 - Simple format – the file contains a human-readable view of the alignments
 - Tabbed format – the file contains tab-delimited lines of the overlaps
 - Default is “None”. See section 1.15.1.8 for a description of the 454PairAlign.txt file.
- **Ace read mode** – When the ACE file is generated, output reads using either the raw, complete basecalled read or using just the trimmed portion of the reads (after low quality, vector and key trimming)
 - Default is set to output trimmed reads
 - Raw is to output the entire sequence of reads
 - Trimmed is to output trimmed reads
- **All contig threshold** – The minimum number of bases for a contig to be output in the 454AllContigs.fna file
 - Default: 100
- **Large contig threshold** – The minimum number of bases for a contig to be output in the 454LargeContigs.fna file
 - Default: 500

For a full listing and description of the output options for assembly, please see Table 4: Mapping and Assembly options and Table 5: Options Specific to Assembly, in the Appendix section.

1.7.4 cDNA Project Input Sub-Tab

The cDNA Assembly project parameters on the Input sub-tab are the same as those of Genomic Assembly projects with the exceptions noted below.

- cDNA Assembly projects do not have checkboxes for **Large or Complex Genome** or **Heterozygotic Mode**, or a text field for entering **Expected Depth**, as these are not relevant to cDNA projects.

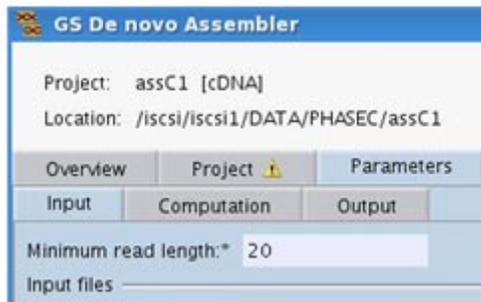


Figure 13: gsAssembler Parameters tab, Input sub-tab for cDNA projects (top-left corner)

1.7.5 cDNA Project Output Sub-Tab

The cDNA Assembly project parameters on the Output sub-tab are the same as those of Genomic Assembly projects with the exceptions noted below.

- cDNA Assembly projects have options to specify the **isotig** and **isogroup thresholds**, as well as **isotig count** and **isogroup count thresholds**. (See 1.14 for a description of these thresholds.)

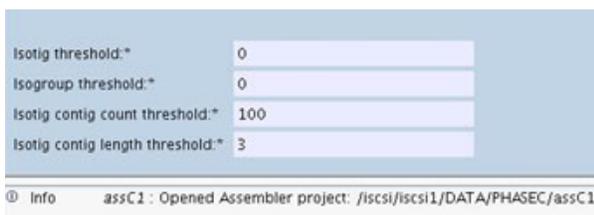


Figure 14: gsAssembler Parameters tab, Output sub-tab, bottom-left corner, showing the Isotig and Isogroup fields specific to cDNA projects

The third sub-tab of the Parameters tab – Computation – is identical for Genomic and cDNA projects.

1.8 Computing the Assembly

1.8.1 Computing an Assembly for the First Time

When at least one Read Data file has been added to the project and all parameters are within acceptable limits, the **Start** button becomes enabled and the “Ready for analysis” message appears in the upper right corner of the application window (Figure 12). The project's parameter settings are also saved when the **Start** button is clicked. Click the **Start** button to carry out the assembly computation of the current project, *i.e.* as defined in the Project and Parameters tabs. As the computation progresses, the **Start** button is disabled, periodic progress messages appear in the progress box, at the bottom of the application's main window, and the current stage of the assembly computation along with a progress bar are displayed in the upper right corner of the window. When the assembly computation is complete, the **Start** button is re-enabled and the message “Ready for analysis” appears again in the upper right corner of the application window.

1.8.2 Re-Computing an Assembly

After an assembly has been performed on a project, the assembly computation can be repeated on that project, with the same Read Data or after addition or removal of one or more file, with the same assembly parameter settings or with different ones. If the “Incremental *De Novo* assembler analysis” checkbox on the Computation sub-tab of the Parameters tab is checked, this re-analysis will take much less time than the initial analysis, since only some data will be re-computed. Changes to some parameter settings (such as those on the Input and Computation sub-tabs of the Project Tab) will have no effect on reads already included in the prior assembly computation.

1.8.3 Stopping an Assembly Computation

Click on the **Stop** button while an assembly project computation is in progress to halt the computation. The effect may not be instantaneous, however. Any delay will be a function of the complexity of the assembly being performed and of the stage of the computation at the time it was interrupted. An informational message appears in the progress box at the bottom of the application's main window, containing the text “Warning: stopRun called for this project. Stopping...” When the assembly computation has been halted, the **Stop** button is disabled and the **Start** button re-enabled.

1.8.4 Information Updated when Assembly Completes

After a successful assembly, the data for the various columns of the reads in the GS reads and FASTA reads sub-tabs will be updated (Figure 15). Placing the mouse pointer over any cell in the table brings up a tool tip with additional information about that item. A successful assembly also updates data in the Overview tab. Project information is also saved when the computation completes.

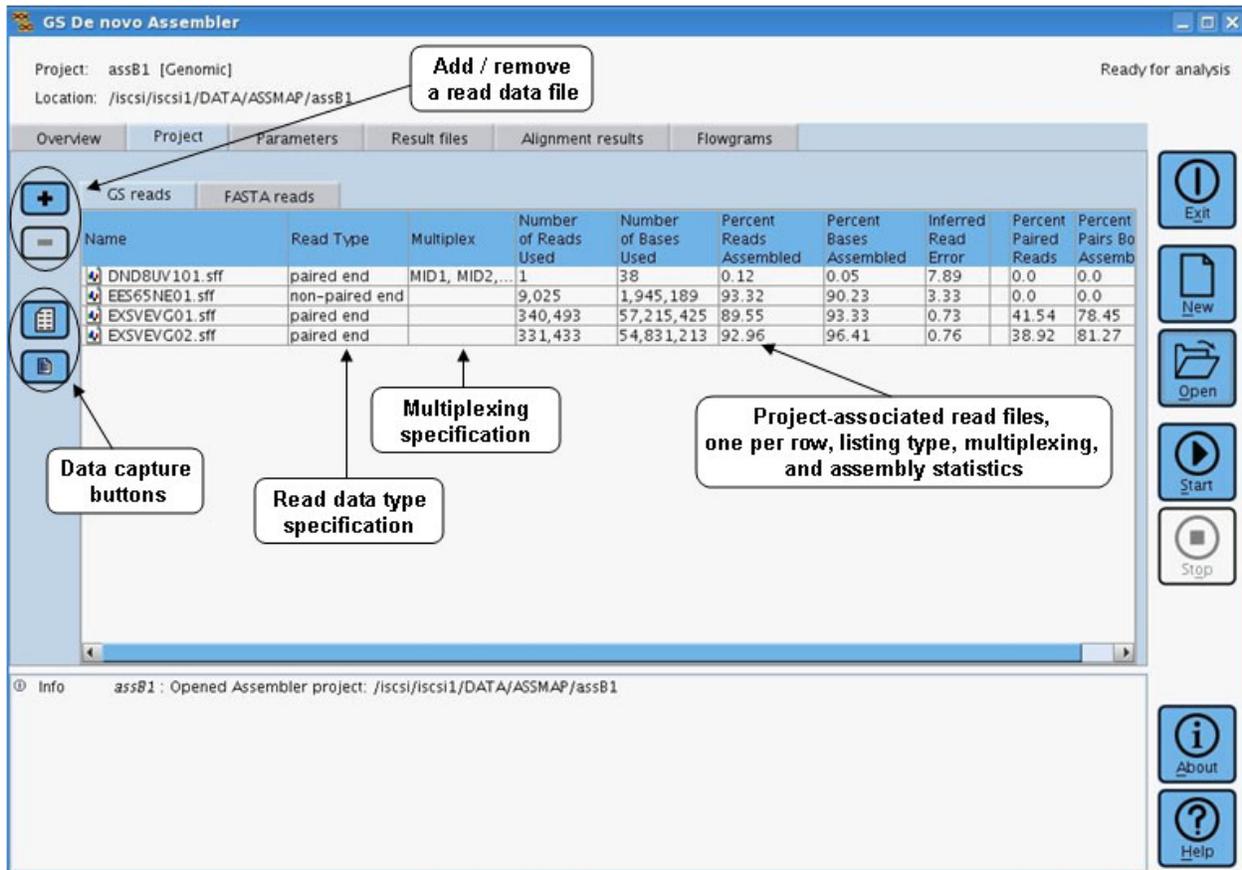


Figure 15: Project Tab of the gsAssembler (GS Reads sub-tab), after completion of an assembly

After an assembly project has been computed, the results of the assembly can be viewed on the Overview, Project, Results Files, Alignment Results, and Flowgram tabs of the GS De Novo Assembly application's main window.



In some cases where large (greater than 100 Mbp) or complex genomes are being assembled, the computation may not indicate its progress for an extended period of time. When this happens, the GUI may incorrectly indicate that the computation has stalled. The computation should be given several more hours to complete if a large or complex genome is being assembled.

1.9 Viewing Assembly Output with the Result Files Tab

The Result Files tab allows you to view the various metrics files generated by the GS De Novo Assembly software (described in detail in Section 1.15.1). Using the list in the left-hand panel of the tab, click on the name of the output file you want to examine; its content will be displayed on the right-hand panel of the tab (Figure 16).



For very long files, such as the sequences of all contigs of a large assembled genome, this view displays the file truncated to 50,000 lines. When this occurs, a note to that effect specifies that it is so both at the beginning and at the end of the display. The file itself remains intact, however.

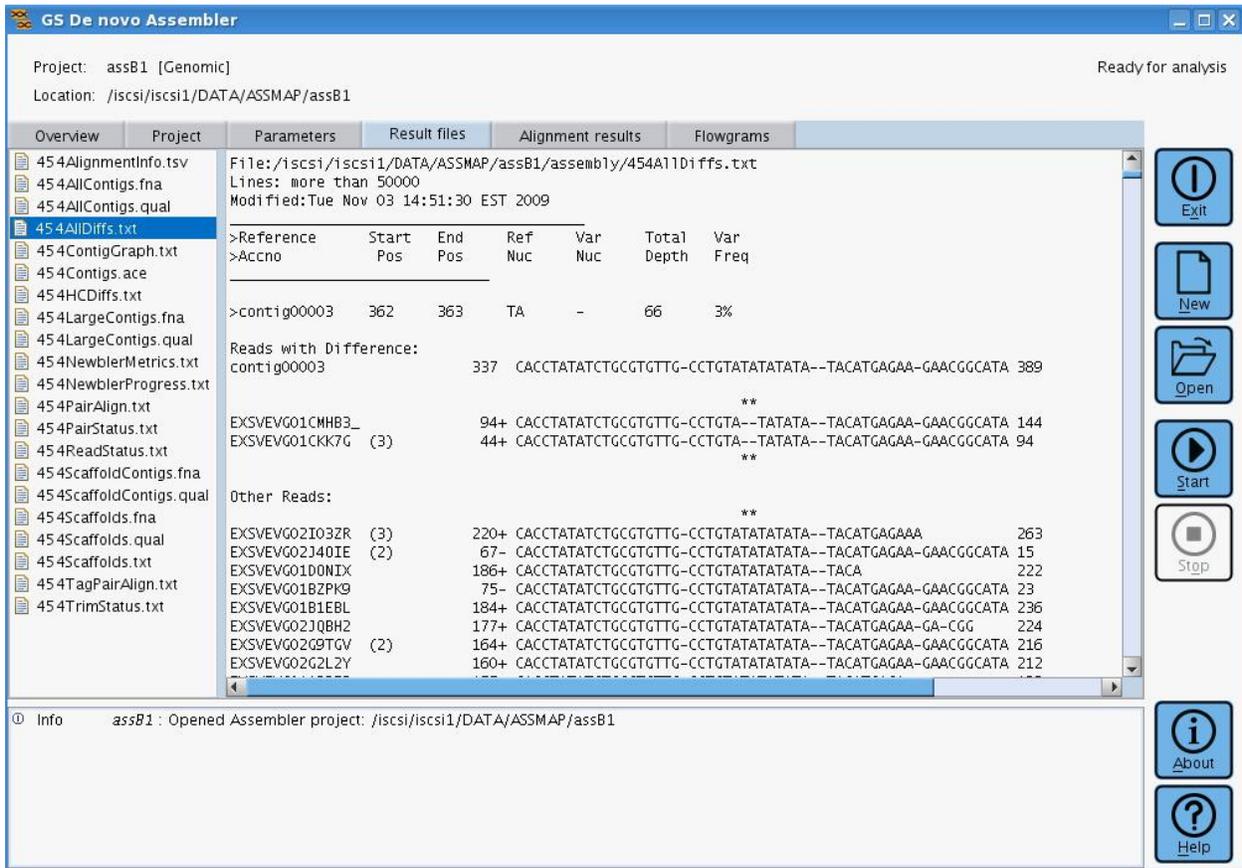


Figure 16: gsAssembler Result Files tab for Genomic projects

Figure 16 above shows the Result Files tab for a Genomic Assembly project. The result files for a cDNA project are listed below (Figure 17). The result file contents are described in detail in Section 1.15.1.

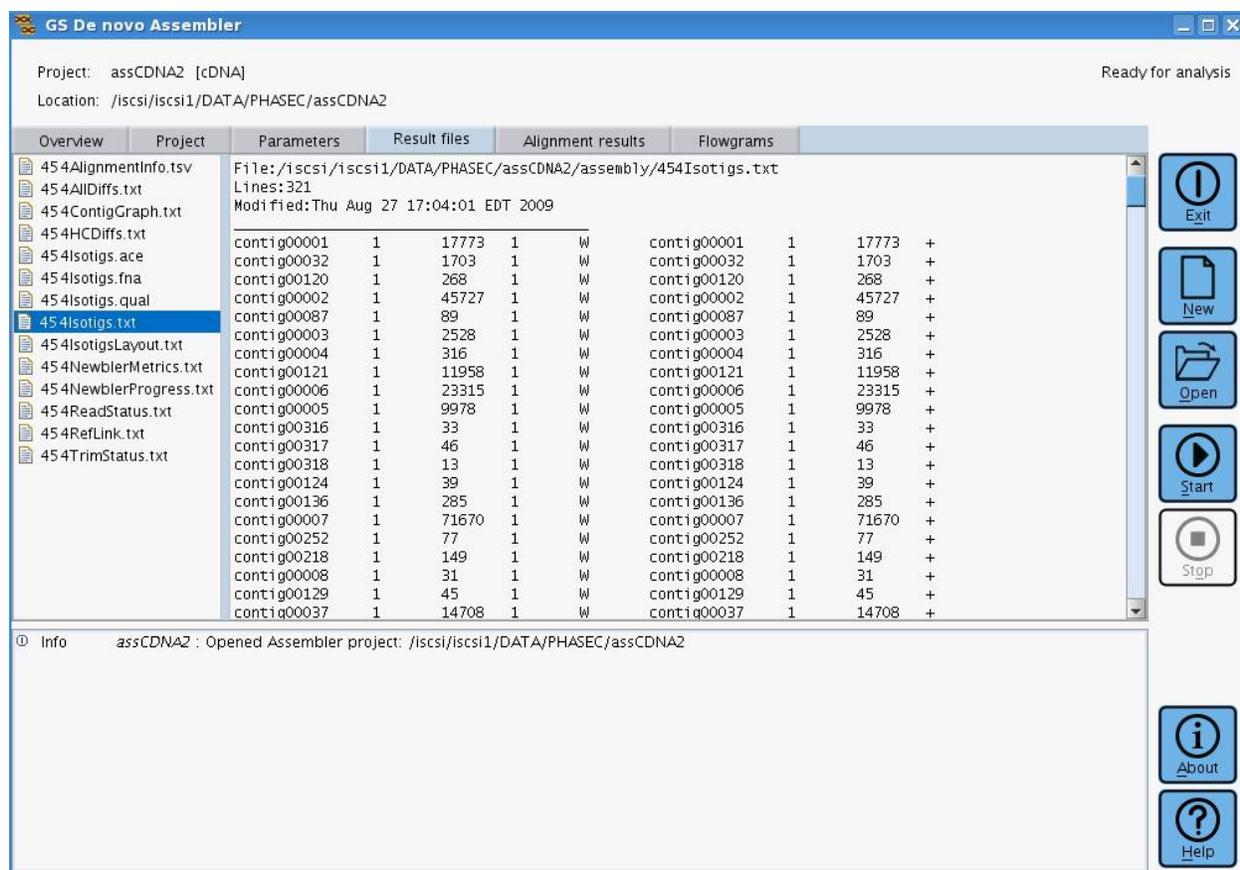


Figure 17: gsAssembler Results Files tab for cDNA projects

The result file contents are described in detail in Section 1.15.1.

1.10 Viewing Contigs with the Alignment Results Tab

Click on the Alignment results tab to view the alignment data from the assembly computation (Figure 18). This tab shows the contigs and their underlying multiple alignments.

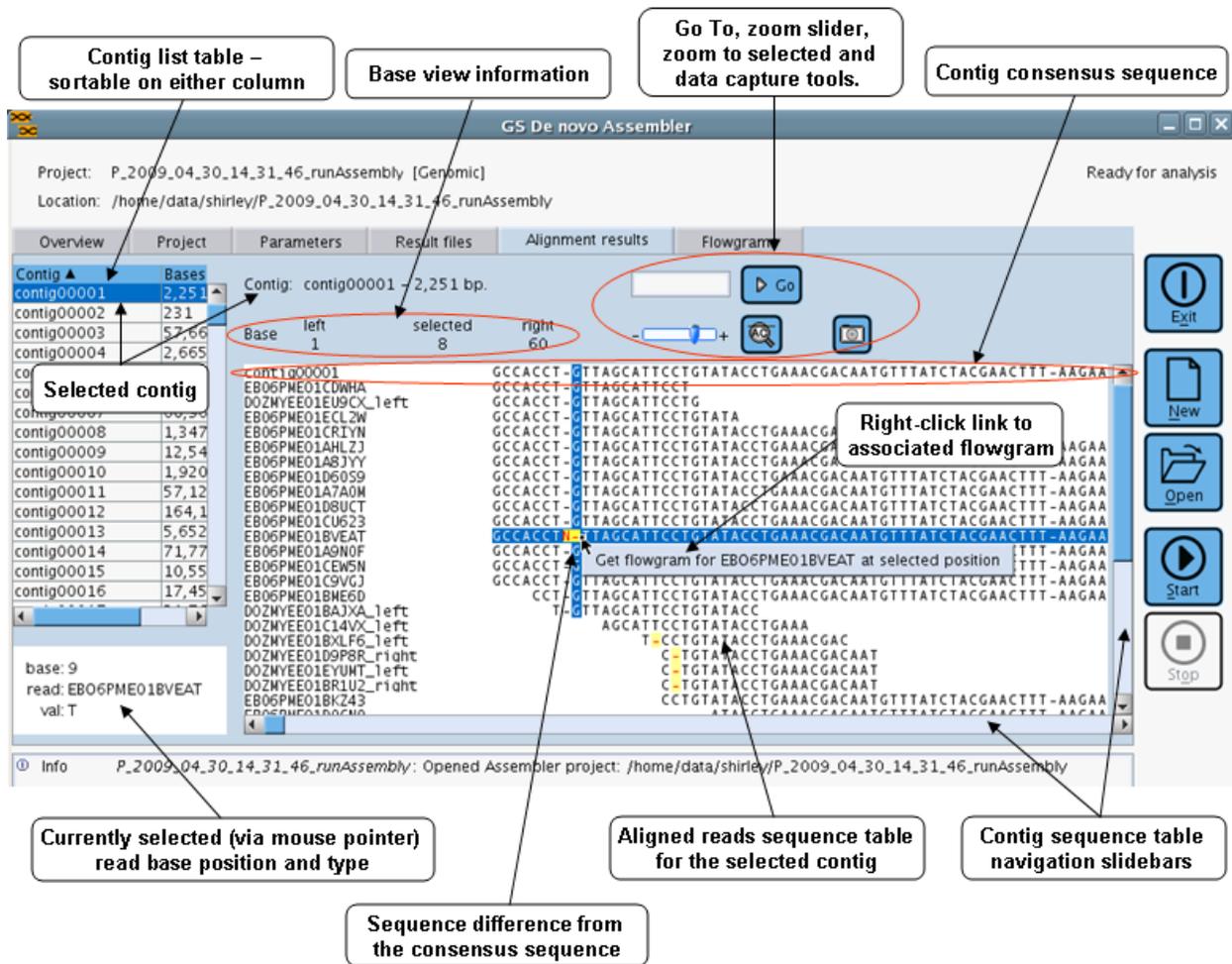


Figure 18: gsAssembler Alignment Results Tab

The left panel of the Alignment Results tab contains a list of the contigs produced by the assembly project. By default, the first contig on the list will be automatically selected.

The multi-alignment panel has the following features:

- The larger area, on white background, contains the multi-alignment proper:
 - The selected contig consensus sequence is shown on top of the panel, gapped for insertions in the aligned reads
 - The reads underlying the contig's multi-alignment, gapped and showing bases that diverge from the consensus as red dashes on yellow background
 - The names of the contig and reads in the multi-alignment appear in a column, on the left side of the multi-alignment area
 - Scroll bars allow you to navigate the alignment manually:
 - Clicking on the arrowheads on either side of a scroll bar scrolls the alignment by one base (horizontal) or one read (vertical) in the direction of the arrowhead
 - Clicking on the light-colored area on either side of the horizontal scroll bar "handle" scrolls the alignment by 40 bases in the corresponding direction

- Clicking and dragging the “handle” of a scroll bar allows you to move larger distances rapidly
- Right clicking on a GS read will produce a “Get flowgram for ... at selected position” menu item which, if selected, will activate the Flowgrams tab and display the flowgram for the read; centered on the flow corresponding to the base on which the user clicked to activate the option.



This capability is not active for FASTA reads or the contig/consensus sequences themselves, for which no flowgrams are available.

- Above that area are some informational and navigational controls:
 - **Go To:** A data entry field and a **Go To** button allow you to navigate directly to a position of interest: enter a number not larger than the length of the selected contig in the data entry field, and click the **Go To** button.
 - A **‘zoom to selected’** button can be used to zoom in, centering on the selected base.
 - The **zoom-slider** tool allows for more or fewer bases to be viewed at a time in the main window. If the slider is set to all the way zoom out (-) then the individual bases are grayed out but the shape of the read alignment is still shown along with the positions of the differences highlighted in yellow (Figure 19).
 -  **Camera icon:** saves an image of the part of the multi-alignment table currently visible on screen, to a file in .png format.
- The Alignment Results tab also features a “Mouse Tracker” area, in the left panel, below the list of reference sequences. If you pause the mouse pointer over a base in the multi-alignment, the following information is provided about that base:
 - **Base:** the position of that base relative to the selected reference sequence; gaps in the reference sequence are displayed as the following base of the reference
 - **Read:** the name of the read or contig to which this base belongs
 - **Val:** the “value” of the base under the mouse pointer, in that read, *i.e.* A, G, T, or C.

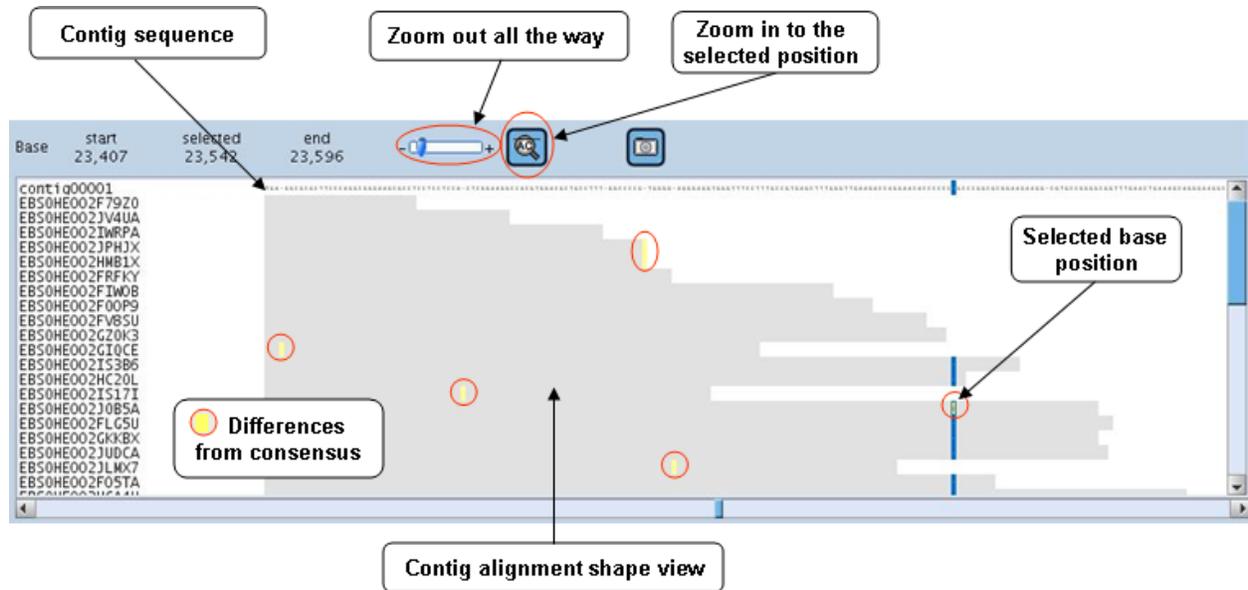


Figure 19: gsAssembler Alignment Results tab with the zoom slider all the way out

1.11 The Flowgrams Tab

When viewing a multiple alignment in the Alignment Results tab, any read in the multiple alignment can be selected in order to display its flowgram. Select the desired read, and then right-click on any base in the read of interest to open a contextual menu that contains a single item ("Get flowgram:..."; Figure 18). Selecting that item will display the flowgram for this read in the Flowgrams tab (Figure 20). For additional information about using the **Flowgrams Tab**, see section 4.11.

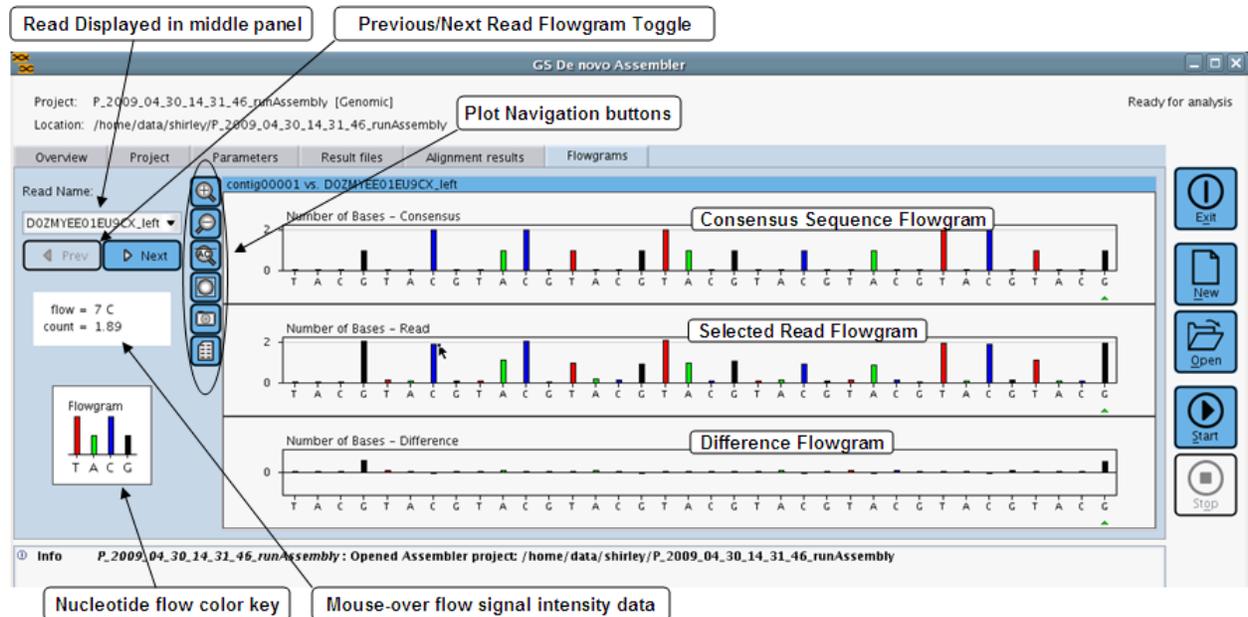


Figure 20: gsAssembler Flowgrams Tab

1.12 Project Error Indicators

To assist the user in the proper setup of assembly, expansion of the error messaging is provided on the Parameters and Project tabs. Some examples are shown in section 4.12.



When a project is first created and no read files have been added yet, a warning icon on the Project Tab header will be displayed along with the status message ‘Not ready for Analysis’.

1.13 The GS *De Novo* Assembler Command Line Interface

The GS *De Novo* Assembler CLI commands for single shot assembly of reads can be launched using the following commands:

```
runAssembly [options] [filedesc]
```

For incremental assembly via the CLI, a project directory structure is created to organize the files of the incremental build of the project data. The following commands are used for incremental assembly projects;

```
newAssembly [-cdna] projdir
```

```
addRun [options: -p, -lib, -mcf] [projdir] filedesc
```

```
removeRun [projdir] filedesc
```

```
runProject [options] [projdir]
```

For all of these commands;

- The arguments in brackets [] are optional.
- ‘options’ refers to the specific command options available.
- ‘filedesc’ is one of the following:
 - an sfffilename
 - a [regionlist:]datadir
 - a readfastafilename
 - any of the above prepended by an [MIDList@] specification where each “MIDList” is a multiplexing information string used to filter the set of file reads to be used in the assembly (If MIDs were used in the generation of the read data file, then an MIDList string must be specified in order for the assembler to properly handle the file’s reads.) For details about using MIDs, see section 4.6.
- ‘projdir’ is the path of the data analysis project directory.



Some of the command line options for Assembly are mutually exclusive, for obvious functional reasons. See section 4.5 for a list of these options.

1.13.1 Working with Project Folders and Data Files

Since the assembly computation is often performed on a pool of sequencing Runs (or Read Data files) rather than on any single Run, the result files it generates are not deposited in a Run folder. Two general cases exist.

- If the assembly is performed using the “one-step” command `runAssembly`, a folder with a ‘P_’ prefix (for ‘P’roject folder) is created in the user’s current working directory on the DataRig at the time the application is launched, or written to a directory specified by the user on the command line, to contain these files. The name structure for this folder is as follows:

```
P_yyyy_mm_dd_hh_min_sec_runAssembly
```

- For “incremental”, or “project-based” assembly, using the GUI application or using the `newAssembly` related commands, the output is placed in a “project” folder. A user can specify any name for a project folder; it will be recognized as a project folder by virtue of the “454Project.xml” file that will be automatically created within it. If a directory name is not specified using the `newAssembly` command line, the software will use the same default name as the `runAssembly` command (as above).

The incremental assembly folder contains additional folders and files that mark the folder as a project folder and that store configuration information and internal data for the GS *De Novo* Assembler application. A project folder is comprised of two sub-folders: an ‘assembly’ sub-folder which contains the project state and output files; and an ‘sff’ sub-folder containing the copies and/or symbolic links for the SFF files used as input to the project. The `454Project.xml` file identifies the folder as a 454 project folder.

- The “-o” option can be specified on the command line to change the directory where the output files should be written. If the specified directory exists, the output files will be written to that directory. If the specified directory does not exist, the program will create it, if possible.



When using the `-o` option with the `runAssembly` or `newAssembly` commands, the assembler will overwrite the contents of the specified project directory if any exist. This will occur without warning. It is good practice to make a copy of any assembly project you wish to protect before reusing the same project directory.



External Files

- GS Read Data files (SFF files) are not actually copied to the project directory. Rather, a symbolic link to the file is created and placed in the project’s sff directory. Consequently, if the original file is moved or erased from the file system, the project will not operate correctly. This applies whether the files are added via the GUI or the command line.
- FASTA files are not actually copied to the project directory. Rather, the file path is simply recorded in the project setup. Consequently, if the original file is moved or erased from the file system, the project will not operate correctly. This applies to all instances of FASTA files that can be added to an Assembly project, whether done via the GUI or the command line:
 - FASTA reads files, including Sanger reads
 - Trimming database files

- Screening database files

Unlike with the GS Read Data files, no symbolic link to the file is created for FASTA files.

1.13.2 One-Step Assembly: the runAssembly Command

If all the reads to be assembled are available at once, the GS *De Novo* Assembler application can process them with a single command, which has the following command line structure:

```
runAssembly [options] [MIDList@]filedesc...
```

...where:

- “[options]” are zero or more of the command line options (see sections 4.1 and 4.2),
- each “filedesc” is one of the following :
 - sfffilename or
 - [regionlist:]datadir or
 - readfastafilename
- and each “MIDList” is a multiplexing information string used to filter the set of file reads to be used in the assembly (see section 4.6 for the format of the MIDList information). If MIDs were used in the generation of the data file, then an MIDList string must be specified in order for the assembler to properly handle the file’s reads.



The runAssembly command is actually a wrapper program around the newAssembly and related commands used in incremental assembly (see section 1.13.3). After the incremental assembly command is completed, runAssembly then transforms the project files and folders into this one-step form (which more closely matches the output structure of older versions of the “Assembly” application). The actions taken are:

- All the assembly output files are moved up from the assembly sub-directory into the main folder
- All internal data files in the assembly sub-directory are deleted
- The 454AssemblyProject.xml and 454Project.xml files are deleted
- The assembly sub-directory is deleted

If the **-nrm** option is given, runAssembly does not perform this transformation, and the resulting folder can be used for further project-based assembly (the structure of the files/folders will match that of an incremental assembly project).



The path given for any of the data directories may be prepended with an optional list of regions, separated from the path by a colon. For example:

```
1-3,4,6-8:D_yyyy_mm_dd_hh_min_sec_testuser_SignalProcessing
```

The list of regions must have the following format:

- It must be a comma-separated list, ending with a colon.
- Each element of the list can either be a single region number or a dash-separated range of region numbers.
- Duplicate regions may be specified, and the regions may be specified in any order, but duplicates will be removed and the regions will be processed and reported in numerical order.
- No spaces or other characters are allowed in the string.

If the region list is not present for a data directory path, all regions of the Run for that directory will be used in the assembly. Also, any combination of explicit SFF files and data directory paths (with optional region lists) may be specified on the command line. For each data directory path given, the `runAssembly` command will read the existing SFF files in the “`sff`” sub-directory of the data directory. If SFF files are not present in a data directory (e.g. for a Run whose data has been processed with a version of the Genome Sequencer System software anterior to 1.0.52), the signal processing step of the GS Run Processor application must be rerun on the Data Processing folder.

The path for any filename or data directory name can also be prepended with a multiplexing information string. Section 4.6 gives further information on using MIDs.

The data analysis software installation package contains a default MID configuration file, found by default at `Installation_path/454/config/MIDConfig.parse`. This file is read by the GS *De Novo* Assembler and used to match MID set names and MID names with their multiplexing information. Users can edit this file to add their own MID sets (following the format and syntax described in the file), or can copy this file to create their own separate MID configuration file (and then use the “`-mcf`” option to specify that as the MID configuration file to be used).

1.13.3 Incremental Assembly: the `newAssembly` and Related Commands

This section describes the main commands of the GS *De Novo* Assembler application, to be used when one or more Runs are to be assembled as part of an assembly project. These commands allow you to add and remove Runs over time and incrementally update the assembly. With these commands, the execution of the assembly algorithms and generation of the results can be controlled by command line options and configuration parameters (paralleling the equivalent controls in the GUI application). Such incremental assembly can be useful when, for example, you want to see intermediate results on existing sequencing Runs to determine if you need to carry out further Runs to reach a desired depth of coverage, or just to monitor the project. Incremental assembly is also useful if you simply wish to create output using different output parameter settings.

Four commands are used in an assembly project: `newAssembly`, `addRun`, `removeRun` and `runProject`. These are described in the sub-sections below.

1.13.3.1 *The newAssembly Command*

The newAssembly command is used to initiate an assembly project and set an assembly project folder to contain the project data (see Section 1.13.1). Its command line structure is:

```
newAssembly [option] [dir]
```

...where:

- [option] -cdna may be used; see sections 4.1
- [dir] is the optional name of the project directory

1.13.3.2 *The addRun Command*

The addRun command is used to add Read Data sets to existing assembly projects. Its command line structure is:

```
addRun [options] [MIDList@]filedesc...
```

...where:

- “[options]” are zero or more of the command line options described in section 4.3
- each “filedesc” is one of the following:
 - sfffilename or
 - [regionlist:]datadir or
 - readfastafilename
- and each “MIDList” is a list of multiplexing information used to filter the set of file reads used in the assembly (see section 4.6 for the format of the MIDList information). If MIDList strings were used in the generation of the data file, then an MIDList string must be specified in order for the GS Reference Mapper to properly handle the file's reads.



- **Input read size constraints:** The reads input for the assembly computation must be shorter than 2000 bases per read (longer sequences are ignored) and longer than 50 bases (shorter sequences are ignored). When Paired End 454 Reads (SFF reads) are part of the project, reads with lengths between the value of the minlen parameter and 50 bp will be mapped onto contigs formed by assembling longer reads during a later stage in the assembly.
- **Order of addition of Read data may affect assembly results:** In general, the reads should be added to incremental assemblies in the following order to achieve the best contigging and scaffolding results:
 - Shotgun
 - 3kb Paired End
 - 8kb Paired End
 - 20kb Paired End



- The addRun command can be executed multiple times for an assembly project (in any combination with the removeRun and runProject commands). When addRun is executed, it adds (not “resets”) the given Runs/regions or SFF files to the list of sequencing data used in the project. (It does not reset the list of sequence data). The reads used in the assembly (runProject, see section 1.13.3.4 below) are the union of the data from all executions of addRun for the project.
- The commands addRun, removeRun, and runProject can be executed in any combination and in any order, in an assembly project.

1.13.3.3 *The removeRun Command*

The removeRun command is used to remove Read Data sets from existing assembly projects. Its command line structure is:

```
removeRun [dir] (sffname or readfastfilepath)...
```



- This command is more conveniently carried out from the GUI application. When called from the command line, it requires the SFF file name(s) *given in the project sff sub-directory* or the FASTA file name(s) *given in the project configuration file 454AssemblyProject.xml*. These names may not match the original names of the corresponding files or may not be known to the user, especially if the software had to rename any of them to ensure name uniqueness. (The filenames of the SFF files in the sff sub-directory are assigned by the GS *De Novo* Assembler application to ensure uniqueness for all the files, while trying to preserve the original names when possible.)
- The execution of this command does not physically remove the file(s) from the project sff sub-directory or from any existing assembly. The file(s) and the reads they contain are only marked for removal by the removeRun command, and are actually removed only the next time the project is computed (via the runProject command).
- The commands addRun, removeRun, and runProject can be executed in any combination and in any order, in an assembly project.

1.13.3.4 *The runProject Command*

The runProject command performs the actual assembly computation for a project and generates the results of the assembly. Its command line structure is:

```
runProject [options] [dir]
```

1.14 GS *De Novo* Assembler cDNA / Transcriptome Options

These descriptions are for options specific to cDNA / transcriptome assembly projects.

cDNA / Transcriptome assembly "**-cdna**": To specify a cDNA / transcriptome assembly project, use the -cdna option with the runAssembly or newAssembly commands.

Isogroup Threshold "**-ig**": Specifies the maximum number of contigs in an isogroup. If an isogroup has more contigs than this threshold, it will not be traversed for isotigs and its contigs will appear as contigs in the output files.

Isotig Threshold "**-it**": Specifies the maximum number of isotigs in an isogroup. When this threshold is reached, the isogroup traversal will stop and all of its contigs will appear as contigs in the output files.

Isotig Contig Count Threshold "**-icc**": Specifies the maximum number of contigs in one isotig. When this threshold is reached, the further traversal of a particular isotig in an isogroup will be stopped and the isotig's contigs might or might not appear as contigs in the output files, depending on whether that particular contig is part of another isotig in an isogroup; only the contigs which are not part of any isotig will be reported as contigs in the output files.

Isotig Contig Length Threshold "**-icl**" : if any contig length is shorter than this threshold, the further traversal of a particular isotig in an isogroup will be stopped. The contig shorter than the icl threshold will be marked as such and reported in the output files. Other contigs from such an isotig might or might not appear as contigs in the output files, depending on whether each particular contig is part of another isotig in an isogroup; only the contigs which are not part of any isotig will be reported as contigs in the output files.

If any of the above-mentioned thresholds is set to **zero**, it will not be taken into account during the computation. In such cases, some implicit thresholds and limits may still be applied as described below. Default thresholds for the computation are:

- ig = 500 contigs
- it = 100 isotigs
- icc = 100 contigs
- icl = 3 base pairs

Implicit thresholds and limits for the computation (imposed due to the recursive nature of traversal algorithm) are:

- Maximum recursion depth during isotig path traversal is **200**. This corresponds to the number of contigs in the isotig, i.e. the -icc threshold.
- Maximum isotig count during graph traversal is **10000**. This corresponds to the number of isotigs in the isogroup, i.e. the -it threshold.
- Cyclic graphs: recursive isotig path traversal will stop if cyclic isotig structures are detected, i.e. revisiting one contig which has already been part of the isotig being traversed. Such cyclic structures will be marked in the output files by assigning cyclic status for the first contig detected, and the cyclic isotig's contigs might or might not appear as contigs in the output files, depending on whether each particular contig is part of another non-cyclic isotig in an isogroup.



The Large or complex genome option should not normally be used in cDNA assembly projects (unless the project's computation won't complete without using the option).



For cDNA assembly, the “-a” is set to 0 regardless of the value that may be entered on the command-line.

1.15 GS *De Novo* Assembler Output

1.15.1 Output File Specification

Output Files produced by the GS *De Novo* Assembler are described briefly below. GUI settings and command line options that control the content type of a file are given when applicable.

File name	Description	Assembly – Genomic Project	Assembly – cDNA Project	GUI Conditional output option	CLI Conditional output option
454AlignmentInfo.tsv	Tab-delimited file giving position-by-position consensus base and flow signal information. Output conditionally (using –info/-noinfo options or checkbox selection on GUI Parameters Tab Output Sub tab) if there are more than 4M reads or the total length of assembled contigs exceeds 40Mbp.	✓	✓	Alignment info: Output -infoall Output small -info No output -noinfo	
454AllContigs.fna	FASTA file of all the consensus basecalled contigs longer than 100 bases. The minimum length output can be changed by using the [-a #] option in the CLI or changing the All contig threshold in the GUI Parameters Tab, Output Sub tab.	✓	✓	Parameters: All contig threshold -a #	
454AllContigs.qual	Corresponding Phred-equivalent quality scores for each base in the consensus contigs in 454AllContigs.fna.	✓	✓		
454Contigs.ace	ACE format file that can be loaded by third-party viewer programs that understand the ACE format. The output can be a single file for the entire project or a folder containing individual files for each contig in the assembly.	✓		Ace/Consed <u>selection</u>	-nobig, -ace, -consed, -noace, -ar, -at, -ad

454ContigGraph.txt	A text file giving the “contig graph” that describes the branching structure between contigs.	✓	✓	(-g is deprecated since the contig graph is now output by default)
454Isotigs.ace	ACE format file that can be loaded by third-party viewer programs that understand the ACE format. The output can be a single file for the entire project or a folder containing individual files for each isotig and large contig in the assembly (only large contigs that aren't part of any isotigs appear as a separate entry in the ACE file).		✓	Ace/Consed selection -nobig -ace/acedir -consed -noace -consed16
454Isotigs.fna	FASTA file of all the isotig sequences traversed from the multiple alignment graph structure (i.e. from the isogroup) along with any large (longer than the Large contig Threshold number of bases) contigs that are not part of any isotig.		✓	
454Isotigs.qual	Corresponding Phred-equivalent quality scores for each base in the isotigs and contigs in 454Isotigs.fna.		✓	
454Isotigs.txt	An AGP file (NCBI's format for describing scaffolds of contigs) containing information describing the path through the contigs taken by each isotig in the isogroup		✓	
454IsotigLayout.txt	A text file that gives a visual representation of how the contigs are laid along each isotig in the isogroup.		✓	
454LargeContigs.fna	FASTA file of all the “large” consensus basecalled contigs contained in 454AllContigs.fna (>500bp). This can be changed by using the [-l #] option in the CLI or changing the Large contig threshold in the GUI Parameters Tab, Output Sub tab.	✓		Parameter: Large contig threshold -l #
454LargeContigs.qual	Corresponding Phred-equivalent quality scores for each base in the “large” consensus contigs in 454LargeContigs.fna.	✓		
454NewblerMetrics.txt	File providing various assembly metrics, including the number of input Runs and reads, the number and size of the large consensus contigs as well as all consensus contigs.	✓	✓	
454NewblerProgress.txt	A text log of the messages sent to standard output during the assembly computation.	✓	✓	

454PairAlign.txt	A text file giving the pairwise alignment(s) of the overlaps used in the assembly computation (only produced when using the <code>-pair</code> option [or <code>-pairt</code> option for the tab-delimited version of the file]).	✓		Pairwise alignment <u>selection</u>	<code>-pair</code> , <code>-pairt</code> , <code>-nobig</code>
454PairStatus.txt	Tab-delimited text file providing a per-pair report of the location and status of how each Paired End pair of reads were used in the assembly.	✓			
454ReadStatus.txt	Tab-delimited text file providing a per-read report of the status of each read in the assembly. The 3' and 5' positions of each read's alignment within the contig are also reported.	✓	✓		
454Scaffolds.fna	FASTA file of the concatenated contig sequences that were scaffolded as a result of Paired End analysis. The contigs are separated by a number of 'N' corresponding to the estimated size of the gap between them (but with a minimum of 20 N's to ensure the separation of the contigs).	✓			
454Scaffolds.qual	Corresponding Phred-equivalent quality scores for each nucleotide in the scaffolded consensus contigs in 454Scaffolds.fna.	✓			
454Scaffolds.txt	An AGP file (NCBI's format for describing scaffolds of contigs) containing the scaffold layout.	✓			
454TagPairAlign.txt	A text file giving the pairwise alignments used in the assembly computation for Paired End reads shorter than 50 bases (which are not part of the overlap computation, but are mapped to the consensi in a later computation step).	✓		Pairwise alignment <u>selection</u>	<code>-pair</code> , <code>-pairt</code>
454TrimStatus.txt	Tab-delimited text file providing a per-read report of the original and revised trimpoints used in the assembly.	✓			

Table 1: GS *De Novo* Assembly Output Files



454NewblerProgress.txt: If Runs are added incrementally and multiple executions of runProject occur, the output messages are appended to this file. If the "Incremental *De Novo* assembly analysis" checkbox option is not selected in the GUI application, or the `-r` option is given on the runProject command line, the GS *De Novo* Assembler deletes intermediate data and "restarts" the assembly computation, and the NewblerProgress.txt file is deleted and restarted as well.

1.15.1.1 454AlignmentInfo.tsv

The 454AlignmentInfo.tsv file (Figure 21) contains position-by-position summary information about the consensus sequence for the contigs generated by the GS *De Novo* Assembler

application, listed one nucleotide per line (in a tab-delimited format). Output conditionally (using the -info/-infoall/-noinfo options or the selection made on the GUI Parameters Tab Output Sub tab). By default, this file is only output if there are fewer than 4 million input reads and the total length of assembled contigs is less than 40Mbp. For larger projects, -info or -infoall or the corresponding GUI **Output** selection for **Alignment Info** must be used to generate this file.

The columns of each line contain the following information:

1. **Position** – the position in the contig
2. **Consensus** – the consensus nucleotide for that position in the contig
3. **Quality Score** – the quality score of the consensus base
4. **Unique Depth** – the number of non-duplicate reads that align at that position in the alignment
5. **Align Depth** – the number of reads (including duplicates) that align at that position in the alignment
6. **Signal** – the average signal of the read flowgrams, for the flows that correspond to that position in the alignment
7. **StdDeviation** – the standard deviation of the read flowgram signals at the corresponding flows

Prior to each region of lines for each contig, a header line beginning with a '>' displays the contig name. e.g.,

Position	Consensus	Quality Score	Unique Depth	Align Depth	Signal	StdDeviation
>contig00001	1					
1	G	64	184	410	1.88	0.56
2	G	64	184	410	1.88	0.56
3	T	64	185	411	1.01	0.11
4	G	64	183	409	0.94	0.12
5	A	64	184	410	0.98	0.09
6	T	64	183	412	1.03	0.10
7	G	64	183	411	0.97	0.09
8	C	64	184	412	0.99	0.12
9	T	64	183	409	0.98	0.13
10	G	64	184	409	0.98	0.08
11	C	64	185	407	1.90	0.13
12	C	64	185	407	1.90	0.13
13	A	64	188	412	1.94	0.15
14	A	64	187	411	1.94	0.15
15	C	64	187	411	0.89	0.13
16	T	64	188	416	1.99	0.15
17	T	64	188	416	1.99	0.15
18	A	64	188	415	1.00	0.09
19	C	64	188	415	0.98	0.09
20	T	64	188	413	1.01	0.14

Figure 21: 454AlignmentInfo.tsv file portion example for an Assembly project

1.15.1.2 *fna and qual files: 454AllContigs, 454LargeContigs, 454Scaffolds files*

These files contain the nucleotide sequences of all the contigs, large contigs, or scaffolds (Figure 22) and associated nucleotide Quality Scores (Phred-equivalent; Figure 23) produced by the GS *De Novo* Assembler application. The AllContig and LargeContig output lengths are specified in the GUI or by CLI options described in Table 1, above.


```
>contig00014 length=104 numreads=0 gene=isogroup00001 st
atus=isotig
64 64 64 64 64 64 64 64 15 64 64 64 64 64 64 55 64 64 64 38
64 61 64 64 64 64 64 64 64 64 64 64 60 64 64 64 64 64 58
64 64 64 64 64 64 64 64 64 64 64 64 64 64 64 64 64 64 64
64 64 64 64 64 64 64 64 64 64 64 64 64 64 64 64 64 64
64 64 55 64 64 64 64 61 64 64 64 64 64 64 64 64 64 64 64
64 64 64 64
```

Figure 23: 454AllContigs.qual file portion example for an Assembly project

1.15.1.3 454ReadStatus.txt

The 454ReadStatus.txt file (Figure 24) contains the status identifiers for all the reads used in the assembly computation, plus the 3' and 5' positions for each assembled read's alignment within the contig results. The reads are listed one per line, in tab-delimited format. Each line contains the following information:

1. **Accno** – Accession number of the input read. If this is a Paired End read, the accno is followed by an underscore character and the mention “left” or “right”, for which half of the pair this read comes from.
2. **Read Status** – status of the read in the assembly, which can be one of the following:
 - a. **Assembled** – the read is fully incorporated into the assembly
 - b. **PartiallyAssembled** – only part of the read was included in the assembly, the rest was deemed to have diverged sufficiently to not be included
 - c. **Singleton** – the read did not overlap with any other reads in the input
 - d. **Repeat** – the read was identified by the assembler as likely coming from a repeat region, and so was excluded from the final contigs
 - e. **Outlier** – the read was identified by the GS *De Novo* Assembler as problematic, and was excluded from the final contigs (one explanation of these outliers are chimeric sequences, but sequences may be identified as outliers simply as an assembler artifact)
 - f. **TooShort** – the trimmed read was too short to be used in the computation (shorter than 50 bases and longer than the value of the minlen parameter, unless 454 Paired End Reads are included in the dataset, in which case, all reads at least “minlen” bases are used).
3. **5' Contig** – The accno of the contig in which the 5' end of the read's alignment begins.
4. **5' Position** – The position in the 5' contig where the 5' end of the read's alignment begins.
5. **5' Strand** – The orientation of the read's alignment relative to the 5' contig. A '+' indicates the alignment orientation of the read is the same as the orientation of the 5' contig. A '-' indicates the alignment orientation of the read is opposite to the orientation of the 5' contig.
6. **3' Contig** – The accno of the contig in which the 3' end of the read's alignment ends.
7. **3' Position** – The position in the 3' contig where the 3' end of the read's alignment ends.
8. **3' Strand** – The orientation of the read's alignment relative to the 3' contig. A '+' indicates the alignment orientation of the read is the same as the orientation of the 3' contig. A '-' indicates the alignment orientation of the read is opposite to the orientation of the 3' contig.

Accno	Read Status	5' Contig	5' Position	5' Strand	3' Contig	3' Position	3' Strand
FKHFISHO2TKM2B_left	Assembled	contig00022	31192	-	contig00022	30934	+
FKHFISHO2TKM2B_right	Assembled	contig00022	27207	+	contig00022	27274	-
FKHFISHO2PIIHY_left	Assembled	contig00005	60616	-	contig00005	60469	+
FKHFISHO2PIIHY_right	Assembled	contig00005	58009	+	contig00005	58181	-
FKHFISHO2P66G3_left	Assembled	contig00036	13248	-	contig00036	12984	+
FKHFISHO2P66G3_right	Repeat						
FKHFISHO2QDT26	Assembled	contig00016	22996	+	contig00016	23241	-
FKHFISHO2QE1IX_left	Assembled	contig00167	20	-	contig00118	63	-

Figure 24: 454ReadStatus.txt file portion example for an Assembly project

1.15.1.4 454TrimStatus.txt

This file contains a per-read report of the original and revised trimpoints used in the assembly, where the revised trimpoints include the effects of primer, vector and/or quality trimming of the original input reads (Figure 25). Each line contains the following information (these are the columns in the tab-delimited format):

1. **Accno** – accession number of the input read
2. **Trimpoints Used** – the final trimpoints used in the assembly, in #-# format
3. **Trimmed Length** – the final trimmed length of the read
4. **Orig. Trimpoints** – the original trimpoints of the read, found in the SFF or FASTA file resulting from signal processing
5. **Orig. Trimmed Length** – the original trimmed length of the read
6. **Raw Length** – the length of the raw read (without any trimming)

Accno	Trimpoints Used	Used	Trimmed Length	Orig	Trimpoints	Orig	Trimmed Length	Raw Length
FKHFISHO2TKM2B_left	5-256	252	5-368	364	401			
FKHFISHO2TKM2B_right	299-368	70	5-368	364	401			
FKHFISHO2PIIHY_left	5-149	145	5-363	359	401			
FKHFISHO2PIIHY_right	192-363	172	5-363	359	401			
FKHFISHO2P66G3_left	5-265	261	5-339	335	375			
FKHFISHO2P66G3_right	308-339	32	5-339	335	375			
FKHFISHO2QDT26	64-304	241	5-304	300	338			
FKHFISHO2QE1IX_left	5-94	90	5-292	288	325			

Figure 25: 454TrimStatus.txt file portion example

1.15.1.5 454Contigs.ace or ace/ContigName.ace or consed/...

This viewer-ready genome file shows all the contigs contained in 454AllContigs.fna and allows the display of how the individual reads aligned to those contigs, in an ACE format file suitable for use in various third-party sequence finishing programs (Figure 26). (The freeware “clview” application can be downloaded from: <http://compbio.dfci.harvard.edu/tgi/software/>; a full description of the .ace file format can be found at: <http://bozeman.mbt.washington.edu/consed/consed.html>.) It should be noted, however, that such third-party viewing software will not be able to make full use of the flow-space assembly information available with Genome Sequencer reads, and that conversely some of the third-party program’s functions (e.g. involving sequence chromatogram input) are not usable with Genome Sequencer datasets. Nonetheless, these programs may be useful to view and assess read characteristics and coverage depth in regions of interest.

The data software analysis applications can also output the assembly results in a complete directory structure suitable for use as input to the consed software (see the above link for a description of this structure and its files). When the appropriate option is selected, a “consed” sub-directory is produced in the output (or project) directory for the computation. This directory contains the sub-directories, the ACE file and the PHD file, so that the functions of consed (viewing traces, editing reads, autofinishing) can be performed on the assembly. In order to integrate the 454 Sequencing reads (and their SFF files) into the consed data, an extra “sff_dir” directory is created in the consed sub-directory, and a number of consed options are automatically specified in this directory (see the “consed/edit_dir/.consedrc” file for the options specified by the generated structure). One option tells consed to use the “sff2scf” command to access any trace information requested by a user. See Section 3.3 for a description of the sff2scf command, and how it can be used to generate synthetic traces from SFF data, as well as provide a pass-through access to SCF data for Sanger reads.

A

```
AS      187 1074357

CO contig00001 464 852 7 U
GG**T**GATG*C*TG*CCAACIT*AC*T*G*A*TTT*AG*TGTA*T*G
*AT*GG**T*GTTTT**G**A*GG**T**GC*T*CC*A*G*T*GGC*TT
*C*TGTTT*CT*A*T*C**AGC*T*GTCCC*TCC**T**GTT*CAG*CTA
C*TG***A*C*GGGG*T*GG*T*GCG*TAA*CGG*CAAAA*G*CACc*G*
CCGG*ACAT*CA*G*C*G*C*T*A*TCTC*TG*CTC*TCA*CT**GCCG*T
*AAAA*C*A*T**GGC*AA*CTG*C*AGTT*C*ACTT*A*CA*CC*G*C
*TT*C*T*C***AA*CCC*GG*T*ACGCA*CC*A***GAAAA*T*C*ATT
G*ATATGGCC*AT**G*AAATGGCGTT**GG*A*T*G***CC**GGG**CA
AC*AG**CCC*GCA*TT*A**T*G*GG*C*GTT*GG*CC**T*C*AACA
**CG**A*T*TT*
```

B

```
AF FKHFISHO2RVIHK_right.366-293.to131 C 346
AF FKHFISHO2RC01X_left.29-1.fm32.pr32 C -199
AF FKHFISHO2QZABD_left.1-3.to131.pr32 U 461
AF FKHFISHO2TF65J_left.246-301.fm2.pr3 U -244
AF FKHFISHO2P54QU_right.14-114.fm46.pr45 U -12
AF FKHFISHO2TBL3K_right.1-115.to131.pr124 U 280
AF FKHFISHO2PK309.211-478.fm32 U -209
AF FKHFISHO2QS2XI_right.96-1.fm63.pr62 C -14
AF FKHFISHO2RVVA3_right.pr2 C 246
AF FKHFISHO2SO6XK_left.348-356.fm66.pr67 U -346
AF FKHFISHO2PN9GI_right.243-279.fm66.pr67 U -241
AF FKHFISHO2SSH8E_left.1-280.to131 U 24
AF FKHFISHO2SVYVD_left.1-192.to131.pr32 U 169
AF FKHFISHO2PVOGQ_left.248-287.fm32.pr33 U -246

BS 1 196 FKHFISHO2PYT8Z.356-61.fm60.to12
BS 197 198 FKHFISHO2QYZGC_right.28-326.fm32.to131.pr33
BS 199 402 FKHFISHO2PYT8Z.356-61.fm60.to12
BS 403 404 FKHFISHO2QYZGC_right.28-326.fm32.to131.pr33
BS 405 462 FKHFISHO2PYT8Z.356-61.fm60.to12
BS 463 463 FKHFISHO2REOJT_right.1-296.to131.pr32
BS 464 464 FKHFISHO2PYT8Z.356-61.fm60.to12
```

C

```
RD FKHFISH02Q1TDD.1-280.to12 622 0 0
AC*T*G*A*TTT*AG*TGTA*T*G*AT*GG**T*GTTTT**G**A*GG*
*T**G*C*T*CC*A*G*T*GGC*TT*C*TGTTT*CT*A*T*C**AGC*T*G
TCCC*TCC**T**GTT* CAG*CIAC*TG***A*C*GGGG*T*GG*T*GCG
*TAA*CGG*CAAAA*G*CAC*TG*COGG*ACAT*CA*G*C*G*C*T*A*TC
TC*TG*CTC*TC*CT**GCCG*T*AAAA**C*A*T**GGC*AA*CTG*C
*AGTT*C*ACTT*A*CA*CC*G*C*TT*C*T*C***AA*CCC*GG*T*AC
GCA*CC*A***GAAAA*T*C*ATTG*ATATGGCC*AT**G*AATGGCGTT
**GG*A*T*G***CC**GGG**CAACC*G**CCC*GCA*TT*A**T*G*G
G*C*GTT**GG*CC*T*C*AACA**G**A*T*TTT*CGCCATTTAAA
AAACTCAGGCCG CAGTGGTAACCTCG CG CATA CAG COGGG CAGTGA CGT
CATCGTCTG CG CGGAAATGGACGAACAGTGGGATA CGT CGGG CTA AAT
CG CG CCAG CG CTGG CTGTTTTA CG CGTATGACAGG CT CGGAAGACGGTT
GTTGCGCAGTATTGGTGAAC

QA 1 438 1 438
DS CHROMAT_FILE: FKHFISH02Q1TDD.1-280.to12 PHD_FILE: FKHFISH02Q1TDD.1-280.to12.phd.1 TIME: Thu Jul 27 12:33:48 2000 CHEM: 454
```

Figure 26: 454Contigs.ace file portions example for a genomic DNA project with Paired End reads. (A) Contig sequence and quality information; (B) Information mapping reads to contigs; (C) Read sequence and quality information



The read information included in the ACE file produced by the GS *De Novo* Assembler application differs from traditional files in that a single read may appear in multiple contigs of the assembly. This occurs because the objective of the Genome Sequencer FLX assembler software is to first identify and partition the repeat and non-repeat regions of the genome, and then output the consensus sequences for those regions. Therefore, if a single read spans the boundary between two contigs, and only one of these contigs consists of a repeat region, the read will be displayed in both the repeat contig and the non-repeat contig on the other side of the boundary.

To ensure compatibility with ACE file viewers, as well as to assist in the post-analysis of the assembly, the identifiers for the reads that appear in multiple places, and for Paired End reads if present, are output with an additional suffix. Several suffixes may be added to the original read identifier:

- For 454 Paired End reads, the two halves of the 454 read that constitute the sequences at the two ends of the original clone (from which the Paired End read was generated) are marked by “_left” and “_right” suffixes.
- If only part of the trimmed read aligns in this contig (either because the read is only Partially Assembled, or the read is aligned with different sections in different contigs, because it spans a repeat region boundary, for example), the base position range of the region aligned in this contig is added, as in “.1-60” if the read has bases 1-60 aligned in the contig.
- If a read has sections aligning to different contigs, then a “.fm” or “.to” suffix (or both) is added to list the contig the read’s alignment is “coming from” or “going to”. The “.fm” suffix is added when the read’s alignment continues into the 5’ end of the contig, and the “.to” suffix is added for reads that lead to another contig off the 3’ end of the contig. For example, if read “C3U5GNB01C40I3” appears in contigs 2 and 45, with bases 1-60 appearing in contig 2 and bases 61-248 appearing in contig 45, the entry in contig 2 will use the identifier “C3U5GNB01C40I3.1-60.to45” and the entry in contig 45 will be “C3U5GNB01C40I3.61-248.fm2”.
- Paired End reads where both halves of the pair align into contigs will have a “.pr” suffix to list where the other half aligns in the assembly. For example, if reads “C3U5GNB01R8GSX_left” and “C3U5GNB01R8GSX_right” align in contigs 36 and 87, respectively, then the accessions will appear as “C3U5GNB01R8GSX_left.pr87” and “C3U5GNB01R8GSX_right.pr36” (assuming that their complete trimmed sequences aligned inside the two contigs, and did not span across multiple contigs).

1.15.1.6 454NewblerMetrics.txt

The 454NewblerMetrics.txt file is a 454 parser file (see the General Overview section of this manual for a description of the file formats) that reports the key input, algorithmic and output metrics for the data analysis software applications. Each application that uses the Newbler algorithm creates a 454NewblerMetrics.txt file with relevant output. Below is a description of all the sections present in this file when produced by the GS *De Novo* Assembler, with their named groups and keywords.

- **runData group** – contains information about the read data used in the analysis (both Sanger and non-Paired-End 454 Sequencing read files are reported on in this section) (not shown in Figure 27 since only Paired End data files were used in this example).

- **pairedReadData group** – contains information about the Paired End input data (Paired End only; Genome Sequencer FLX System and Sanger if any) (Figure 27)

```

/*****
**
**      454 Life Sciences Corporation
**      Newbler Metrics Results
**
**      Date of Assembly: 2009/10/19 13:00:00
**      Project Directory: /remote/rigdata/nas17/watson/data2/rwiner/rwTest2/ecoPEAssy
**      Software Release: 2.3 (091009_0919)
**
*****/

/*
** Input information.
*/

runData
{
}

pairedReadData
{
    file
    {
        path = "/remote/rigdata/nas17/watson/data2/pairedDataSets/ecoli3kb/RandD1108/FKHFISHO2.sff";

        numberOfReads = 673886, 1125098;
        numberOfBases = 263921005, 235847650;
        numWithPairedRead = 459233;
    }
}

```

Figure 27: 454NewblerMetrics file, runData and pairedReadData groups portion example

- **runMetrics group** – contains information about the assembly computation (Figure 28)
- **readAlignmentResults group** – contains information about the alignments for each input file (SFF, FASTA, or Run regions from wells file) (not shown on Figure 28 since only Paired End data files were used in this example)
- **pairedReadResults group** – contains information about the Paired End input data (Paired End only; Genome Sequencer FLX System and Sanger, if any) (Figure 28)

```
/*
** Operation metrics.
*/
runMetrics
{
    totalNumberOfReads = 1125098;
    totalNumberOfBases = 235847650;

    numberSearches    = 159652;
    seedHitsFound     = 18631881, 116.70;
    overlapsFound      = 3891408, 24.37, 20.89%;
    overlapsReported   = 3644824, 22.83, 93.66%;
    overlapsUsed       = 1034286, 6.48, 28.38%;
}

readAlignmentResults
{
}

pairedReadResults
{
    file
    {
        path = "/remote/rigdata/nas17/watson/data2/pairedDataSets/ecoli3kb/RandD1108/FKHFISH02.sff";

        numAlignedReads    = 1061189, 94.32%;
        numAlignedBases    = 232183395, 98.45%;
        inferredReadError  = 0.91%, 2111076;

        numberWithBothMapped = 402145;
        numWithOneUnmapped   = 4568;
        numWithMultiplyMapped = 50246;
        numWithBothUnmapped  = 2274;
    }
}
}
```

Figure 28: 454NewblerMetrics file, runMetrics, readAlignmentResults, and pairedReadResults groups portion example

- **consensusDistribution group** – contains information about the consensus signals and basecalling thresholds
- **alignmentDepths group** – provides a histogram of the number of alignment positions at each coverage depth
- **consensusResults group** – contains summary information about the read status and, if Paired End reads are used in the project, Paired End library statistics and scaffold statistics. This is followed by contig statistics for large contigs (longer than 'largeContigThreshold'; default is 500 bp) and all contigs (longer than 'allContigsThreshold'; default is 100 bp) (Figure 29).

```

/*
** Consensus results.
*/
consensusResults
{
    readStatus
    {
        numAlignedReads    = 1061189, 94.32%;
        numAlignedBases    = 232183395, 98.45%;
        inferredReadError = 0.91%, 2111076;

        numberAssembled = 1053788;
        numberPartial   = 7278;
        numberSingleton = 10970;
        numberRepeat    = 52724;
        numberOutlier   = 338;
        numberTooShort  = 0;
    }

    pairedReadStatus
    {
        numberWithBothMapped    = 402145;
        numberWithOneUnmapped   = 4568;
        numberMultiplyMapped    = 50246;
        numberWithBothUnmapped = 2274;

        library
        {
            libraryName    = "FKHFISHO2.sff";
            pairDistanceAvg = 2787.1;
            pairDistanceDev = 696.8;
        }
    }

    scaffoldMetrics
    {
        numberOfScaffolds = 11;
        numberOfBases     = 4606980;

        avgScaffoldSize    = 418816;
        N50ScaffoldSize    = 2496701;
        largestScaffoldSize = 2496701;
    }

    largeContigMetrics
    {
        numberOfContigs = 101;
        numberOfBases   = 4564280;

        avgContigSize    = 45190;
        N50ContigSize    = 105463;
        largestContigSize = 267914;

        Q40PlusBases     = 4553447, 99.76%;
        Q39MinusBases    = 10833, 0.24%;
    }

    allContigMetrics
    {
        numberOfContigs = 187;
        numberOfBases   = 4580076;
    }
}

```

Figure 29: 454NewblerMetrics file, consensusResults group example for a Paired End genomic project

1.15.1.7 454NewblerProgress.txt

This file represents the text log of the messages sent to standard output by the runProject command (showing the progress of the execution of the assembly computation). If Runs are added incrementally and multiple executions of runProject occur, the output messages are appended to this file. If the “Incremental *De Novo* assembler analysis” checkbox option is not selected in the GUI application, or if the “-r” option is given on the runProject command line, the GS *De Novo* Assembler deletes intermediate data and “restarts” the assembly computation, and this file is deleted and restarted as well.

1.15.1.8 454PairAlign.txt

This file contains the pairwise alignments of the overlaps that were found during the assembly computation (Figure 30). By default, this file is not generated, but if the “-p” or “-pt” options are given on the runProject command line, it will be generated either in a human-readable text format (“-p”) or in tab-delimited format (“-pt”).

Each of the displayed alignments contains the following information (these are the columns in the tab-delimited format):

1. **QueryAccno** – accession number of the read used in the overlap detection search (the “query sequence”)
2. **QueryStart** – starting position of the alignment in query sequence
3. **QueryEnd** – ending position of the alignment in query sequence
4. **QueryLength** – length of the query sequence
5. **SubjAccno** – accession number of the other read (the “subject sequence”)
6. **SubjStart** – starting position of the alignment in subject sequence
7. **SubjEnd** – ending position of the alignment in subject sequence
8. **SubjLength** – length of the subject sequence
9. **NumIdent** – number of identities in the pairwise alignment, *i.e.* where query and subject characters match
10. **AlignLength** – the length of the pairwise alignment
11. **QueryAlign** – query alignment sequence
12. **SubjAlign** – subject alignment sequence

```
>FKHERYTO1BB82F, 58..1 of 58 and gi|84626310|ref|NC_001147.5|, 305311..305371 of 1153232 (58/61 ident)
 58 ACT-AAAGCAAGG-AAATTACACACATAATAATATAAGTAATGATACGTCCATTATG-TCAA 1
 305311 ACTAAAAGCAAGGAAATTACACACATAATAATATAAGTAATGATACGTCCATTATGTTCAA 305371
>FKHERYTO1AQG4Y_left, 1..164 of 164 and gi|50593503|ref|NC_001148.3|, 446318..446483 of 886119 (162/168 ident)
 1 ATCTTCTCCAGAAGGCAGGCATTTTCTCTCTGGTTAGTAAAATACCACAGAACAGTAACATAGCAGCAAAACAAATTGGAATTAATCATAGTAAGCTGAGGGAACATCAGGAGA
CATGTACAGGATCCATAGACCATATGTGACAAA-TATAATGG-AAA-TTAAA-C 164
 446318 ATCTTCTCCAGAAGGCAGGCATTTTCTCTCTGGTT-GTAAAATACCACAGAACAGTAACATAGCAGCAAAACAAATTGGAATTAATCATAG-AAGCTGAGGGAACATCAGGAGA
CATGTACAGGATCCATAGACCATATGTGACAAAATATAATGGGAAAATTTAAAAC 446483
```

Figure 30: 454PairAlign.txt file portion example for a genomic project with Paired End reads

1.15.1.9 454PairStatus.txt

This file contains the per-pair report of the location and status of how each Paired End pair of reads were used in the assembly. Each line contains the following information (see also Figure 31):

1. **Template** – template string for the pair (this will be the original 454 accession for 454 Paired End reads, and the “template” string for Sanger reads)
2. **Status** – the status of the pair in the assembly, with the following possible values:
 - a. **BothUnmapped** – both halves of the pair were unmapped
 - b. **OneUnmapped** – one of the reads in the pair was unmapped
 - c. **MultiplyMapped** – one or both of the reads in the pair were marked as Repeat
 - d. **SameContig** – both halves of the pair were assembled into the same contig, with the correct relative orientation, and are within the expected distance of each other
 - e. **Link** – the halves were assembled to different contigs, and are near enough to the ends of those contigs that they could be used as a link in a scaffold
 - f. **FalsePair** – the halves were assembled, but the orientation of the aligned contigs is inconsistent with a Paired End pair or the distance between the halves is outside the expected distance
3. **Distance** – for “SameContig” or “Link” pairs, the distance between the halves (where the “Link” distance is simply the sum of the distances from each half to the end of its respective contig).
4. **Left Contig** – the contig where the left half was assembled, or ‘-’ if the read was Unmapped or Repeat. (If the read aligns across multiple contigs, the location of the first base in the read that is aligned is used, to denote where the end of the corresponding clone occurs.)
5. **Left Pos** – the position in the contig where the 5’ end of the left half was assembled
6. **Left Dir** – the direction (‘+’ for the forward strand of the contig and ‘-’ for reverse strand) in which the left half was assembled
7. **Right Contig** – the contig where the right half was assembled, or ‘-’ if the read was Unmapped or Repeat. (If the read aligns across multiple contigs, the location of the first base of the read that is aligned is used, to denote where the end of the corresponding clone occurs.)
8. **Right Pos** – the position in the contig where the 3’ end of the right half was assembled
9. **Right Dir** – the direction (‘+’ for the forward strand of the contig and ‘-’ for reverse strand) in which the right half was assembled
10. **Left Distance** – the distance from the Left Pos to the respective end of the contig (for forward matches, this is the distance to the 3’ end of the contig; for reverse matches, to the 5’ end)
11. **Right Distance** – the distance from the Right Pos to the respective end of the contig (for forward matches, this is the distance to the 3’ end of the contig; for reverse matches, to the 5’ end).

Template	Status	Distance	Left Contig	Left Pos	Left Dir	Right Contig	Right Pos	Right Dir	Left Distance	RightDistance
FKHFISHO2TRM2B	SameContig	3985	contig00022	31192	-	contig00022	27207	+		
FKHFISHO2PLIH	SameContig	2607	contig00005	60616	-	contig00005	58009	+		
FKHFISHO2P66G3	MultiplyMapped	-	contig00036	13248	-	Repeat				
FKHFISHO2QELIX	Link	2467	contig00167	20	-	contig00166	92854	+	20	2447
FKHFISHO2QCKK8	SameContig	4134	contig00060	8142	+	contig00060	12276	-		
FKHFISHO2QDWD0	SameContig	1781	contig00008	28178	+	contig00008	29959	-		
FKHFISHO2TQ2A5	SameContig	1529	contig00031	7905	-	contig00031	6376	+		
FKHFISHO2QLY0L	Link	2083	contig00104	1908	-	contig00018	175	-	1908	175

Figure 31: 454PairStatus.txt file portion example

1.15.1.10 454TagPairAlign.txt

This file contains the pairwise alignments of “short” reads that were found during the assembly computation. When 454 Paired End reads are included in the dataset, reads that are between 15 and 50 bases long are not included in the overlap computation, but are aligned to the contig consensi in a later step of the computation (using different alignment detection settings). This

file reports the alignments of the 15-50 bp reads against the contig consensi. By default, this file is not generated, but if the “-p” or “-pt” options are given on the runProject command line, it will be generated either in a human-readable text format (“-p”) or in tab-delimited format (“-pt”). The format of this file is identical to the 454PairAlign.txt file described in section 1.15.1.8.

1.15.1.11 454Scaffolds.txt

This file contains AGP-formatted information describing how the contigs included in the 454LargeContigs.fna file (see section 1.15.1.2) are scaffolded into the sequence scaffolds of 454Scaffolds.fna (section 1.15.1.2). A description of the AGP format can be found on NCBI’s web site (http://www.ncbi.nlm.nih.gov/genome/guide/Assembly/AGP_Specification.html).

scaffold00001	1	203169	1	W	contig00002	1	203169	+
scaffold00001	203170	203466	2	W	contig00001	1	297	+
scaffold00001	203467	203935	3	W	contig00131	1	469	+
scaffold00001	203936	207000	4	W	contig00003	1	3065	+
scaffold00001	207001	208346	5	W	contig00124	1	1346	+
scaffold00001	208347	250745	6	W	contig00004	1	42399	+
scaffold00001	250746	251084	7	N	339	fragment	yes	
scaffold00001	251085	322395	8	W	contig00005	1	71311	+
scaffold00001	322396	322718	9	N	323	fragment	yes	

Figure 32: 454Scaffolds.txt file portion example

1.15.1.12 454ContigGraph.txt

The 454ContigGraph.txt file contains a graph-based description of the branching structure of an assembly’s contigs, where nodes represent the contigs and edges between contigs give the branching structure. When Paired End reads are included in the assembly, the file also shows scaffold edges, describing how the contigs have been linked together into scaffolds.

The file is comprised of two or three sections (the third section is for scaffold information, produced only if Paired End reads are part of the project), with each line of a section describing a single node or edge of the graph. The first section describes the nodes (contigs) of the graph, and contains the following columns, separated by tab characters:

1. The number of the contig (*i.e.*, for contig “contig00002”, this column would contain the number “2”)
2. The full name of the contig (*e.g.*, “contig00002”)
3. The length of the consensus for this contig
4. The average depth of the contig’s multiple alignment

The second section describes the edges of the graph, describing how the alignment of reads branch off in different directions (or converge together) from the end of a contig. Edges connect not contigs, but the ends of contigs (each contig has a 5’ and 3’ end, corresponding to the beginning (5’) and end (3’) of the contig multiple alignment). Different contigs are not oriented relative to each other, so an edge can connect the 5’ end of contig00001 and the 5’ end of contig00002 (or the 3’ ends of two contigs). In this situation, reverse complementing one of the contigs, so that the edge is a 5’ to 3’ edge, presents the two contigs in a consistent orientation

with each other. The columns of the second section are the following (separated by tab characters):

1. 'C' – the letter 'C' to mark this line as a “contig edge”
2. The contig number on one side of the edge
3. Either “5” or “3” to denote which end of the contig
4. The contig number on the other side of the edge
5. Either “5” or “3” to denote which end of the contig
6. The depth of the multiple alignment that spans between the two contig ends (*i.e.*, the number of reads whose alignments crosses the two contigs ends).

For example, the line

```
C      1      5'      2      3'      21
```

describes an edge between the 5' end of contig00001 and the 3' end of contig00002, where the alignments of 21 reads occur partially at the 5' end of contig00001 and partially at the 3' end of contig00002. All edges are undirected edges.

The third section of the file describes the “scaffold edges” found when performing scaffolding (with Paired End reads only). Only edges used in the scaffolding itself are listed (not edges representing all paired end data). The columns of the third section are the following (separated by tab characters):

1. 'S' – the letter 'S' to mark this line as a “scaffold edge”
2. The contig number on one side of the edge
3. Either “5” or “3” to denote which end of the contig
4. The contig number on the other side of the edge
5. Either “5” or “3” to denote which end of the contig

For example, the line

```
S      6      3'      8      5'
```

describes an edge between the 3' end of contig00006 and the 5' end of contig00008. All edges are undirected edges.

1.15.2 GS *De Novo* Assembler cDNA / Transcriptome Output

1.15.2.1 *454Isotigs.fna*, *454Isotigs.qual*, and *454Isotigs.txt* files

The *454LargeContigs.fna*, *454LargeContigs.qual* (1.15.1.2) are not generated for a cDNA / transcriptome assembly project. However, the new files *454Isotigs.fna* (Figure 33) and *454Isotigs.qual* files are generated (in addition to the *454AllContigs.fna*, *454AllContigs.qual* files). Similar to contig files, they contain respectively fasta sequences and their quality scores. These files contain the isotig sequences traversed from the multiple-alignment graph structure (*i.e.* from the isogroup), but they also can contain single contigs which haven't been traversed (and therefore aren't considered isotigs) because of thresholds, limits, cycles, etc.

```
>isotig00001 gene=isogroup00001 length=779 numContigs=4
CCGCCATTtGCATTGTAAAaGCTACAGACTATCACCATGGTATATATATATATATGGTGA
TTATCTGCTATATACGACAGAAATACCATACAAACCGTTTACTCGGCAACCTTGACTtG
CTTTtGATTTCACTTTATTAATTAaaaaaTATACATAAGTTCAATGTAATTGAAAGTACA
ACACATCAAAATATAAACCTAGATATATAGAAACCAGAAGAAATGTTACAAAAGAGAAAATT
CAATTGGAGCTACGAATTCCTTAATATCTTCTTTGGTCTTTGAGTTGGGTTTCTTTCT
TGAATGTAGGTACCCTGGAACAATGTGTCTGGCAAGTTCAAGTATTCTCTCAAGTATTCA
ACACCTTCTTCAGTCAAGGTGTAGTAGTATTGCCATGAGAATTGAGTCTTGACGTAA
CCCTTAGAAGTCAAGGATGTAAAGCCTTAATGACATACAAGTTCTGGGTGTCAAATTTCT
TCGTGCTTGGCTTGGTTGAAATCCTTCTTGGCGACAACAACCTTCTTGAATAAGTAT
TGGTGGATCTTGTCTTCTTCTTGGCATCAACATCTTGGCTACTTAGTTCTGACTG
TTCTTGCTAAACTTTGCTACTCCcaCCAAGATCTGCACTAGAGGCCGTTCCGACCGACCT
TAGGGTCTAGGCTTGTCTGCTGACCTCCACGCTGCTACTCGTCAGGGCATCATATCAA
CCCTGACGGTAGAGTATAGGTAaCACGCTTGAGCGCCATCCaTTtCAGGgCTAGTtCA
```

Figure 33: Portion example of the 454Isotigs.fna file for a cDNA assembly project

The description line for a contig sequence looks like this:

```
>contig03247 length=225 numreads=11 gene=isogroup00004 status=cyclic
```

Also, a special file, 454Isotigs.txt is generated with content and format identical to the 454Scaffolds.txt file.

isotig00001	1	104	1	W	contig00014	1	104	+
isotig00001	105	528	2	W	contig00015	1	424	+
isotig00001	529	622	3	W	contig00018	1	94	+
isotig00001	623	779	4	W	contig00019	1	157	+
isotig00002	1	424	1	W	contig00015	1	424	+
isotig00002	425	518	2	W	contig00018	1	94	+
isotig00002	519	675	3	W	contig00019	1	157	+

Figure 34: Portion example of the 454Isotigs.txt file for a cDNA assembly project

1.15.2.2 454Isotigs.ace and consed/...

The cDNA assembler produces an ACE file similar to that produced by the genomic assembler (see 1.15.1.5). Isotigs and large contigs that do not appear in any isotig are represented in the ACE file along with reads used to construct their multiple alignments.

1.15.2.3 454NewblerMetrics.txt

The consensusResults group of the 454NewblerMetrics file for cDNA projects displays output similar but not identical to those output by genomic projects in that it gives metrics for isogroups and isotigs in addition to the metrics for the large contigs and all contigs (Figure 35).

```
/*  
** Consensus results.  
*/  
consensusResults  
{  
    readStatus  
    {  
        numAlignedReads    = 4993, 99.44%;  
        numAlignedBases    = 1956211, 97.86%;  
        inferredReadError  = 0.66%, 12851;  
  
        numberAssembled   = 4434;  
        numberPartial     = 559;  
        numberSingleton   = 0;  
        numberRepeat      = 0;  
        numberOutlier     = 28;  
        numberTooShort    = 0;  
    }  
  
    isogroupMetrics  
    {  
        numberOfIsogroups  = 7;  
        avgContigCnt       = 2.9;  
        largestContigCnt   = 7;  
        numberWithOneContig = 4;  
  
        avgIsotigCnt       = 2.1;  
        largestIsotigCnt   = 5;  
        numberWithOneIsotig = 4;  
    }  
  
    isotigMetrics  
    {  
        numberOfIsotigs    = 15;  
        avgContigCnt       = 2.1;  
        largestContigCnt   = 4;  
        numberWithOneConitg = 5;  
  
        numberOfBases      = 29448;  
        avgIsotigSize      = 1963;  
        N50IsotigSize     = 3267;  
        largestIsotigSize  = 4526;  
    }  
}
```

Figure 35: 454NewblerMetrics file, consensusResults group example for a cDNA project

1.15.2.4 454RefLink.txt

Another file specific to cDNA / transcriptome projects, 454RefLink.txt, is generated with a format like UCSC's refLink.txt files. This can be used as an annotation file for further mapping projects of the cDNA / transcriptome assembly products (isotigs and contigs).

```
#name product mrnaAcc protAcc geneName prodName locusLinkId omimId
isogroup00001 isotig00001
isogroup00001 isotig00002
isogroup00001 isotig00003
isogroup00001 isotig00004
isogroup00002 isotig00005
isogroup00002 isotig00006
isogroup00002 isotig00007
isogroup00002 isotig00008
isogroup00002 isotig00009
isogroup00003 isotig00010
isogroup00003 isotig00011
isogroup00004 isotig00012
isogroup00005 isotig00013
isogroup00006 isotig00014
isogroup00007 isotig00015
```

Figure 36: 454RefLink.txt file example

1.15.2.5 454IsotigsLayout.txt

Finally, the file 454IsotigsLayout.txt gives some sort of visual representation of how the contigs are laid along each isotig in the isogroup:

```
>isogroup00004 numIsotigs=16 numContigs=12
  Length : 413  426  849  368  177  90  352  88  264  245  345  178  (bp)
  Contig : 00110 03640 03164 03644 03818 03819 03592 03615 03593 00107 00108 03709 Total:
isotig00001 >>>> <<<<<< >>>> <<<<<< <<<<<< >>>> >>>> 1812
isotig00002 >>>> <<<<<< >>>> <<<<<< <<<<<< >>>> >>>> 1645
isotig00003 >>>> <<<<<< >>>> <<<<<< <<<<<< >>>> >>>> 1658
isotig00004 >>>> <<<<<< >>>> >>>> <<<<<< <<<<<< >>>> >>>> 1825
isotig00005 >>>> <<<<<< >>>> <<<<<< >>>> <<<<<< >>>> >>>> 1658
isotig00006 >>>> <<<<<< >>>> >>>> <<<<<< >>>> >>>> 1825
isotig00007 >>>> <<<<<< >>>> <<<<<< >>>> <<<<<< >>>> >>>> 1658
isotig00008 >>>> <<<<<< >>>> <<<<<< >>>> <<<<<< >>>> >>>> 1825
isotig00009 >>>> >>>> >>>> <<<<<< <<<<<< >>>> >>>> 1714
isotig00010 >>>> >>>> >>>> >>>> <<<<<< <<<<<< >>>> >>>> 1881
isotig00011 >>>> <<<<<< >>>> <<<<<< >>>> <<<<<< >>>> >>>> 1645
isotig00012 >>>> <<<<<< >>>> >>>> <<<<<< >>>> >>>> 1812
isotig00013 >>>> <<<<<< >>>> <<<<<< >>>> <<<<<< >>>> >>>> 1645
isotig00014 >>>> <<<<<< >>>> <<<<<< >>>> <<<<<< >>>> >>>> 1812
isotig00015 >>>> >>>> >>>> >>>> <<<<<< >>>> >>>> >>>> 1714
isotig00016 >>>> >>>> >>>> >>>> <<<<<< >>>> >>>> >>>> 1881
```

where ">>>>" and "<<<<<" denote whether the particular contig was traversed from the 5' to the 3' end or vice-versa.

The lengths given in the top row are contig lengths, and the lengths on the right-most column are isotig lengths.

In addition, at the end of this file there are 4 histograms:

- counts of isogroup contigs
- counts of isogroup isotigs
- counts of isotig contigs
- lengths of isotigs (binned into 100bp bins)

IsogroupContigCount	IsogroupCount
1	2366
2	9
3	247
4	17
11	5

IsogroupIsotigsCount	IsogroupCount
1	2371
2	253
3	38
4	21
5	6
6	5
7	3

IsotigContigCount	IsotigCount
1	2366
2	595
3	170
4	55
5	42
6	17
7	8

IsotigLength	IsotigCount
0-100	152
100-200	473
200-300	1172
400-500	397
500-600	256
700-800	137
800-900	62
1100-1200	24
1400-1500	8

At the very end of the 454IsotigsLayout.txt file is information about how many regular isotigs are generated and how many non-traversed contigs remain. The status column gives the reason each contig was not traversed (thresholds, limits, cycles, etc.). For some contigs, the status can be "none", meaning they weren't reached at all because the traversal stopped before.

```
Filter status:
#isotig #none #cyclyc
2821    1      12
```

2. GS REFERENCE MAPPER

2.1 Overview of the GS Reference Mapper

The GS Reference Mapper application aligns sequencing reads against a reference sequence (consisting of one or more sequences or a GoldenPath genome), with or without associated annotations. Mapping generates consensus sequences of the reads that align against the reference and also computes statistics for variations found in the reads, relative to the reference.

The input data can come from one, several or all the PicoTiterPlate regions of the Run(s) of interest. The GS Reference Mapper application also allows the inclusion of one or more FASTA files of reads, such as reads obtained using Sanger sequencing (“Sanger reads”) in the analysis, as well as the inclusion of one or more Paired End read files. The application can be accessed via a Graphical User Interface (GUI) or from a command line interface (CLI).

The GS Reference Mapper application allows a user to:

- create mapping projects for genomic or cDNA reads
- add and remove read data sets from the project
- add or remove reference sequences and annotations
- specify mapping parameters
- specify special library preparations (such as the use of MIDs)
- run the mapping algorithms on the project data
- and view the output produced by the mapping computation

When the mapping algorithms run, the software performs the following operations:

- For each read, search for its best alignment to the reference sequence(s) (a read may align to multiple positions in the reference); this is done in “nucleotide” space.
- Perform multiple alignments for the reads that align contiguously to the reference in order to form “contigs.” From the contigs’ multiple alignments, consensus basecall sequences are produced using the signals of the reads in the multiple alignments (performed in “flowspace”)
- Identify subsets of the reads that vary relative to the reference to form lists of putative variations (nucleotide differences and structural variants). For each putative variation, the reads supporting the variation will be in the subset.
- Evaluate these lists of putative variations to identify High-Confidence nucleotide differences (HCDiffs) and structural variations (HCStructVars).
- Output the following information:
 - contig consensus sequence(s) and associated quality values
 - alignments of the reads to the reference
 - position-by-position metrics of the depth and consensus accuracy (quality values) for each position in the aligned reference
 - the positions and alignments of identified differences.

Read overlaps and multiple alignments are made in “nucleotide” space while the consensus basecalling and quality value determination for contigs are performed in “flowspace”. Work in flowspace allows the averaging of processed flow signals (a continuous variable) at each nucleotide flow of the sequencing Run(s) and allows the use of information from the “negative flows”, *i.e.* flows where no nucleotide incorporation is detected. The use of flowspace in determining the properties of the consensus sequence results in an improved accuracy for the final basecalls.



The GS Reference Mapper application is not available on the Genome Sequencer FLX Instrument and must be run on a DataRig.

The GS Reference Mapper allows users to create, modify, and run mapping projects. Both the GUI and command line interface (CLI) provide this functionality. Projects may be setup to map all reads at once. Alternatively, incremental operation allows additional reads to be added to an existing mapping project. Results appear as output files using either the GUI or the CLI. The GUI provides a graphical interface to view many of the results from the mapping computation whether the mapping was performed using the GUI or the CLI.

The GS Reference Mapper application uses a folder on the file system to hold the mapping project information (whether the mapping is performed through the GUI application or through the newMapping and related commands) and to hold the output files generated during the mapping computation. The contents of this folder and the names of the files generated by the application are the same for any mapping project or output folder.

The operation of the GUI is described in several of the subsequent sections. A description of the CLI is then presented followed by a discussion of transcriptome mapping. Finally, output files produced by the GS Reference Mapper are described.

2.2 GS Reference Mapper GUI

Mapping can be performed and the results can be viewed using the Graphical User Interface application, gsMapper, described in the next few sections. The application includes graphical interfaces to:

- create new mapping projects
- open existing mapping projects
- Add/remove reads to/from mapping projects
- Add/remove references to/from mapping projects
- Associate annotations with the reference(s)
- Modify mapping project and output parameters
- carry out a mapping computation on a project
- view the results of a completed mapping
- view the progress and logging information of a mapping computation

2.3 Launching the GS Reference Mapper GUI

From a Linux terminal window on a DataRig where the data analysis software package is installed, the following command can be used to launch the GS Reference Mapper GUI application:

```
gsMapper
```

Once the GUI is launched (Figure 37), a user can open an existing project or create a new project.

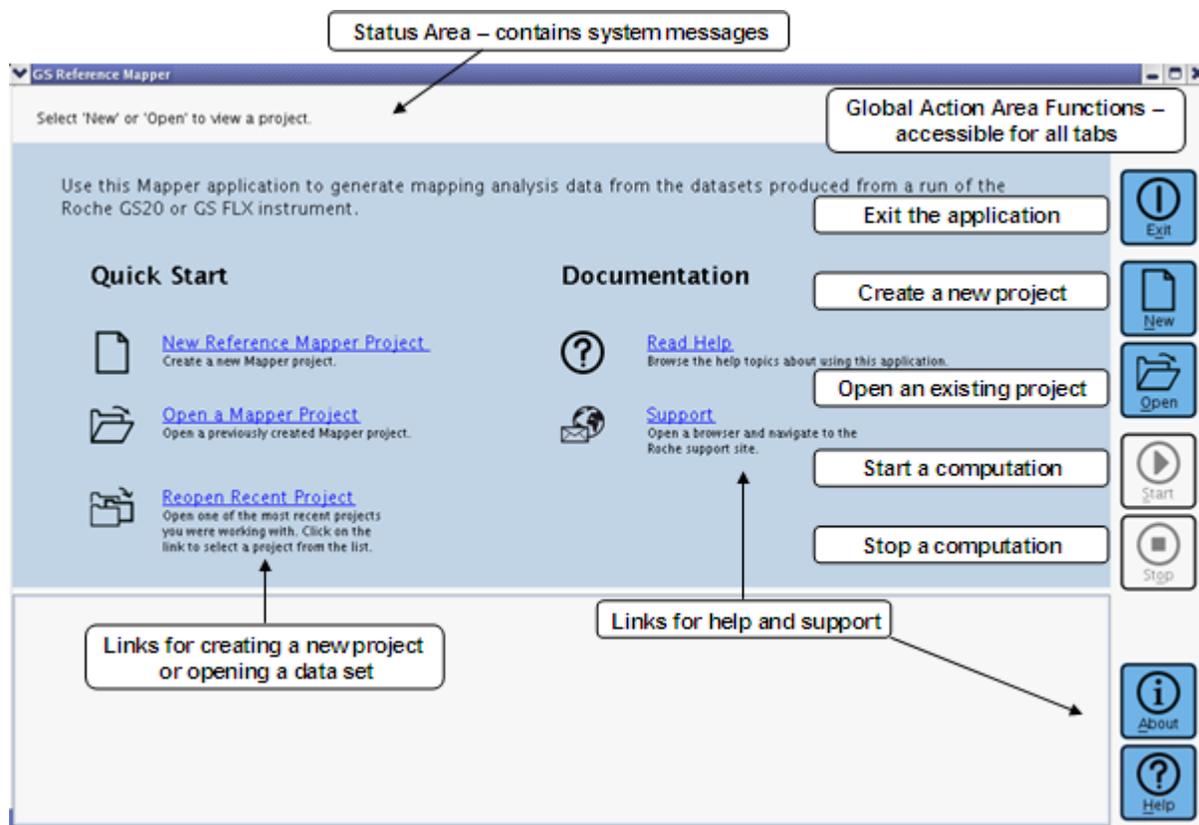


Figure 37: Initial Interface of the gsMapper

2.3.1 The Main Buttons, Status Area and Progress Box

Seven main buttons are always visible on the toolbar, along the right-hand side of the GS Reference Mapper's main window:

- The **Exit** button closes the GS Reference Mapper application
- The **New** button allows you to create a new mapping project
- The **Open** button allows you to open an existing mapping project
- The **Start** button begins the mapping (computation) of an open mapping project
- The **Stop** button halts the execution of an ongoing mapping computation
- The **About** button shows the GS Reference Mapper splash screen
- The **Help** button shows a help dialog (online help is not currently implemented)

The top of the main GS Reference Mapper window contains a status area that displays a command hint, while the bottom panel is a progress box which remains blank until a mapping project has been opened.



The progress box at the bottom of the window can be resized by dragging its top frame separator. If the field is not visible, it is probably simply collapsed; you can view it by dragging up the edge of the collapsed frame.

2.3.2 The Quick Start and Documentation Text Links

There are 3 text links under the **Quick Start** column:

- The **New Mapping Project** text link creates a new mapping project. It performs the same action as the **New** button on the right side of the window.
- The **Open a Mapping Project** text link opens an existing project. It performs the same action as the **Open** button on the right side of the window.
- The **Reopen Recent Project** text link displays a dialog from which you can open a project on which you worked recently.

There are 2 text links under the **Documentation** column:

- The **Read Help** text link shows a help dialog. Online help is not currently implemented; please use this Software Manual, or click on the Support button (see below), or contact your Roche Representative for any information regarding the Genome Sequencer FLX System and its software package. The Read Help text button performs the same action as the **Help** button on the right side of the window.
- The **Support** text link opens the default internet browser and navigates to the Roche support site.

2.4 Opening a Project

2.4.1 Creating a New Project

To create a new mapping project, either click on the **New Project** button in the right toolbar, or click on the “New Mapping Project” text button in the **Quick Start** column. This displays a dialog (Figure 38) in which you can specify the name and directory location for a new project.

Type a name for the new project in the **Name** text field. To specify the project location, either type the full path in the **Location** text field, or click on the **Open Project** button to the right of the text field and use the “Select Project Location” window (not shown) to navigate to the directory where you want to create the new project. The **Full Path** field changes as you update the **Name** and **Location** fields. When you are satisfied with the location and name for the new

project, click on the **OK** button. You can save a project by clicking on the Yes button to the **Save** prompt.



You may save the project to your hard drive either by using the Exit button to exit the GS Mapper application (you will be prompted to save before the application actually exits) or by adding read data and running the project by clicking the Start button (i.e., compute the mapping), in which case the project will automatically be saved prior to computation, as described in section 2.8.1, below.

The New Project Dialog contains a drop-down menu (Figure 38) to specify the **Sequence type** indicating the type of DNA library sequenced, cDNA or Genomic. This choice of Sequence type determines many of the parameters the user will be allowed to modify in the project. Once a project is created the type cannot be changed.

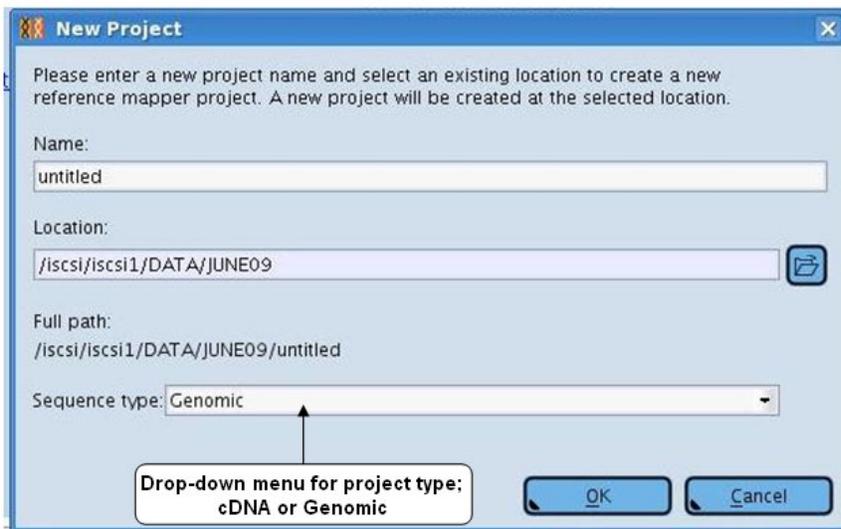


Figure 38: New Project dialog used to specify Sequence type

2.4.2 Open an Existing Project

To open an existing mapping project, either click on the **Open Project** button in the right toolbar, or click on the “Open a Mapping Project” text button in the **Quick Start** column. This displays a dialog (Figure 39) in which you can select a name and specify the directory location of the project to be opened. By default, only 454 Mapping Projects will be displayed. You may choose to display all files by selecting “All Files” from the “**Files of Type**” dropdown menu.

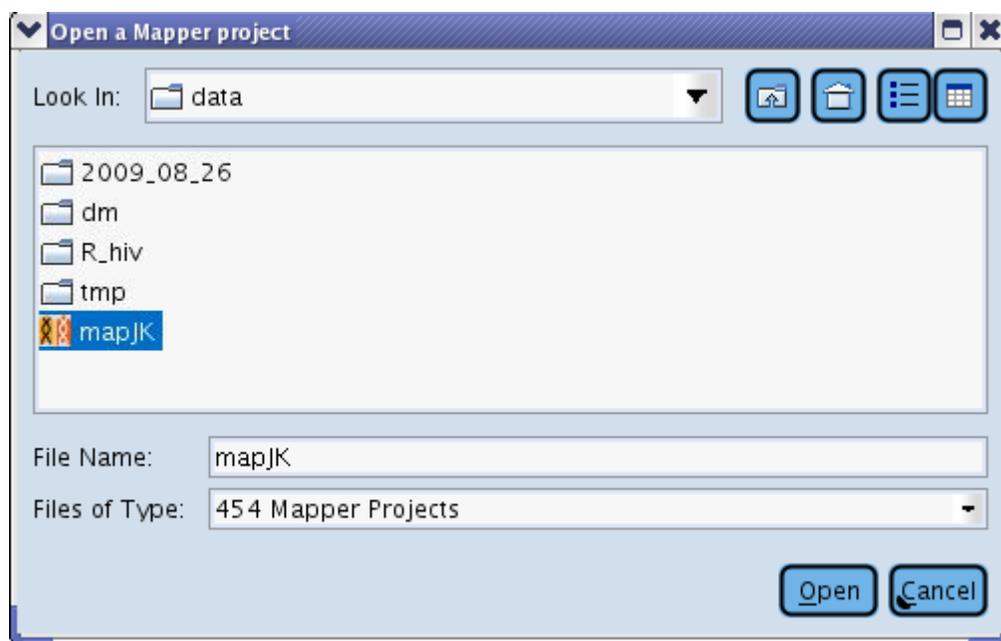


Figure 39: Open a Mapping Project dialog.

2.5 View Project Summary with the Overview Tab

When a project opens, the Overview Tab is displayed (Figure 40). Other tabs can be selected with the mouse. Some tabs may remain inactive (grayed-out), until the type of information they require is available for the project. The Overview Tab's **Project Summary Information** area indicates the Sequence type along with other information that applies to the overall project. The data displayed on this tab is updated as new information becomes available, which occurs when data and reference files are added or removed from the project and when the project computation completes.

When a project is first created, the **Computation status message** in the upper right hand corner of the application window indicates "Not ready for analysis" and the Start button is disabled because at least one Read Data file and one Reference file must be added to the project before a Mapping computation can be performed. Other errors in input parameters can also cause this message to appear. Once all such errors have been corrected and at least one Read Data file and one Reference file are added to the project, the message will change to "Ready for analysis" and the Start button will become active.

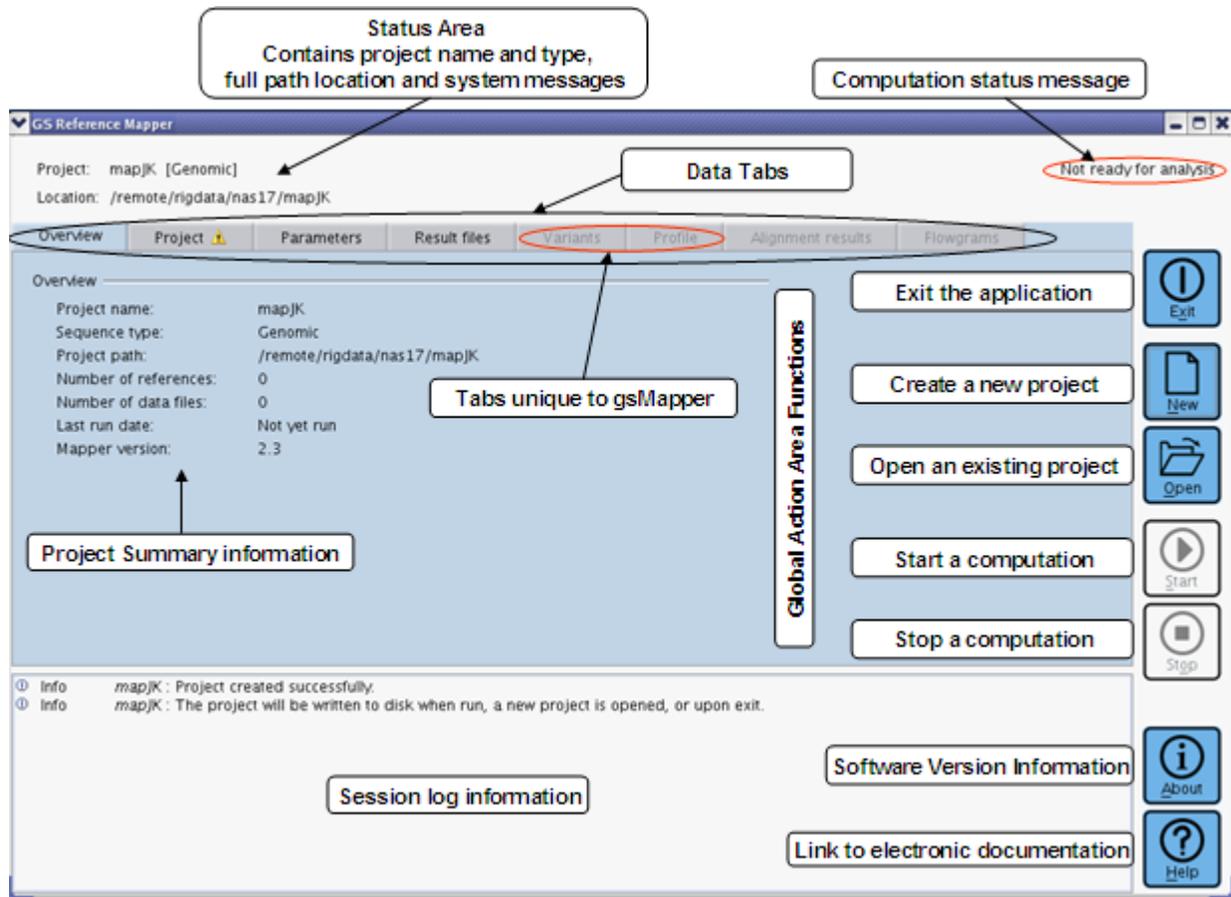


Figure 40: gsMapper Overview Tab

2.6 Add/Remove Read Data and References with the Project Tab

Read Data and reference files are added to a project on the Project tab of the main application window (Figure 41). There are two sub-tabs for adding read data: one for adding GS reads (“GS Reads” sub-tab) and one for adding non-GS reads, such as Sanger reads, in FASTA format (“FASTA Reads” sub-tab). In addition, the References sub-tab is used to add reference files to a project.



The **Start** button in the toolbar, which is used to start a mapping computation, remains disabled until at least one Read Data file (either GS read or FASTA read) and one Reference File have been added to the project.

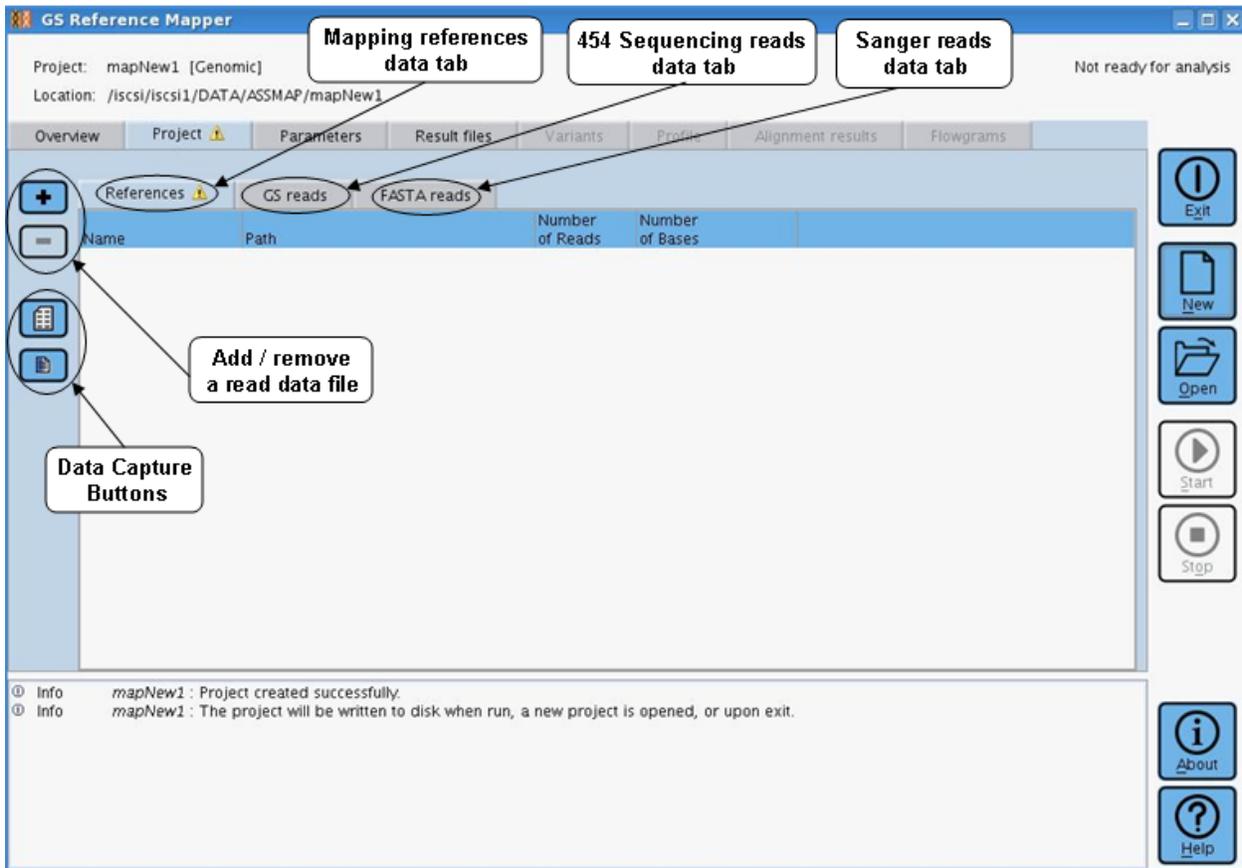


Figure 41: Project Tab of the gsMapper (GS Reads sub-tab)

2.6.1 GS Reads, References and FASTA Reads Sub-Tabs

To add GS read data sets to a project, click on the GS reads sub-tab, then click the **Add** button to open the “Select GS Read Data” window (Figure 42). The top portion of this window lists the GS Read Data files that can be added to the project; if no files exist at this location, the window displays the message: “There are no GS Read Data files in the selected directory”. More than one file can be selected for import. The GS Read files can be explicitly specified as Paired End or non-Paired End or the user can invoke auto-detection (the default setting) using the **Read Type Specification** dropdown. When the read type is known, it is advisable to specify it directly rather than use auto-detect as the auto-detect feature, on rare occasion, may fail to detect a Paired End file.

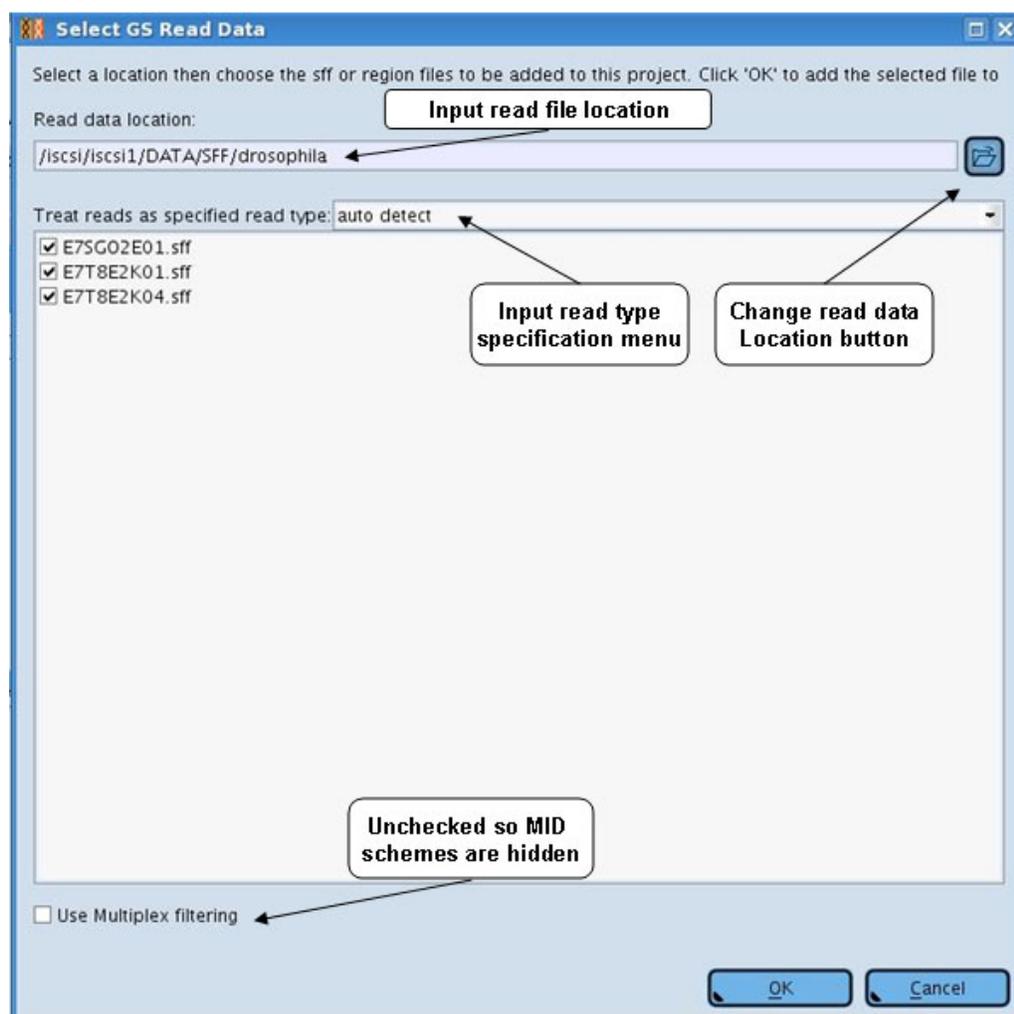


Figure 42: Select GS Read Data dialog

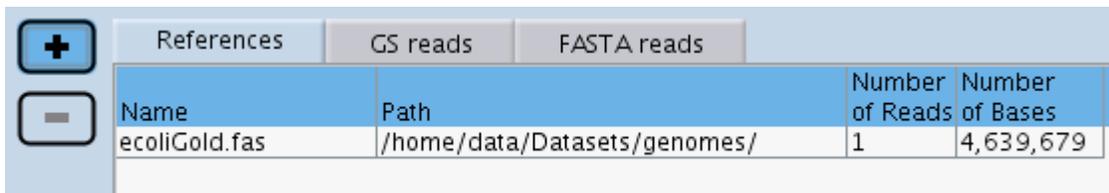
To change to a different location that contains Read Data files, click on the “**Change the read data location**” button (with the open-folder icon) to the right of the text field and use the “Select Read Data Directory” window (not shown) to navigate to and select the new location. You may also directly type or paste the desired path into the “File Name:” field of the navigation window, rather than browsing through the file system for the directory of interest. While using this interface, only directories will appear, not the files within those directories. When you return from the Select GS Read Data window, the list of available Read Data files found at the new location will be refreshed and appear in the field below the Read data location text area (as seen in Figure 42). Use the check boxes to select the file(s) you want to include in the mapping project.

You can add any number of Read Data files to a project using the Add button, navigating to a folder, then selecting the files needed from the folder. You can even add multiple files that share the same name into a project, as long as they have different path locations in the file system (e.g. “/dir1/path1/reads.sff” and “/dir2/otherpath2/reads.sff”). In such a case, both files will be added to the Read Data list and displayed using the same name. To see the path to a file listed in the Read Data area of the main window (and the file’s last modification date), pause the mouse over the filename of interest and a tooltip containing the file’s path will be displayed. Files that have failed validation will have a red X left of the file name as an indication of failure.

Pause the mouse cursor over the red **X** to bring up a tooltip explaining the problem encountered.

The FASTA Reads sub-tab is functionally similar to the GS Reads sub-tab for all project types. The only difference is that when a directory containing FASTA files is selected, the software examines the files in that directory to determine which files are FASTA files. This can take some time if there are many files in the directory, so a progress bar is displayed to show the progress of the search.

The References sub-tab is functionally similar to the Reads sub-tabs for both genomic and cDNA mapping projects. The difference is that instead of displaying all the available read data files when a directory selected, the software displays all recognized FASTA files. You may select one or more of these files. Reference files contain the DNA sequence(s) against which the reads in the Read Data files will be aligned. Once the mapping computation is run, the software determines the total number of reference sequences and bases in the file and includes this information in the two rightmost columns of the **References** tab (Figure 43).



Name	Path	Number of Reads	Number of Bases
ecoliGold.fas	/home/data/Datasets/genomes/	1	4,639,679

Figure 43: gsMapper Project Tab References sub-tab



Although in many cases the file name extension for FASTA files may be **.fna**, there is no standardized extension for such files. If you cannot see your FASTA file, try setting the “Files of Type” field to “All Files”.

Except for the Name column and the Multiplex column (which contains comma-delimited lists of the MIDs associated with the file; see Section 2.6.3, below), all columns with run statistics are initially filled with dashes. These data are updated in the table each time the project runs to completion. For a project that has already been through a mapping computation, summary statistics relating to the usage of the reads in the mapping process are also listed, as shown in Figure 44.. On the left hand side of the reads table are two data capture buttons that can be used to capture the reads table data in tsv or text formats.

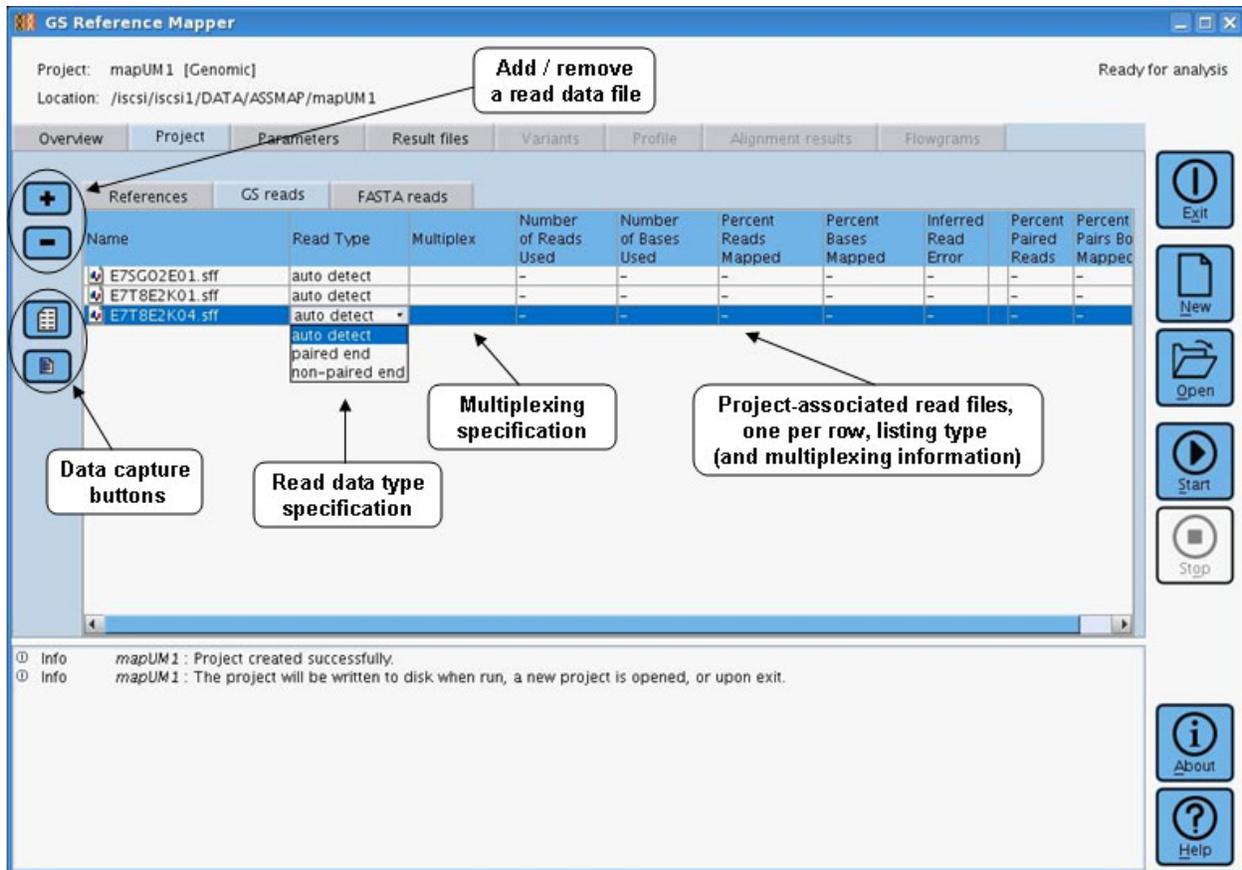


Figure 44: Project Tab of the gsMapper (GS Reads sub-tab)

2.6.2 Removing Read or Reference Data from the project Sub-Tabs

To delete read or reference data from a project, click on the sub-tab that lists the reads or reference data, select the data file(s) to be removed, then click the **Remove** button .



The **Remove** button removes the selected Reference and/or Read Data files from the list, but does not immediately remove the file(s) from the project's sff sub-directory or from any computed mapping results. Actual removal of the files from the project (and of the reads they contain from existing alignments), occurs only when the mapping is re-computed (see section 2.8)

2.6.3 Specifying Multiplex Identifiers (MIDs)

When the Use Multiplex Filtering checkbox is selected, special controls for MID filtering are displayed in the bottom half of the Select GS Read Data window (Figure 45). Multiplex Identifiers (MIDs) allow you to design an experiment whereby multiple libraries are prepared using distinct MID tags and sequenced together, in the same region of a PTP device. (This can greatly improve the workflow and cost effectiveness of your experiment.)

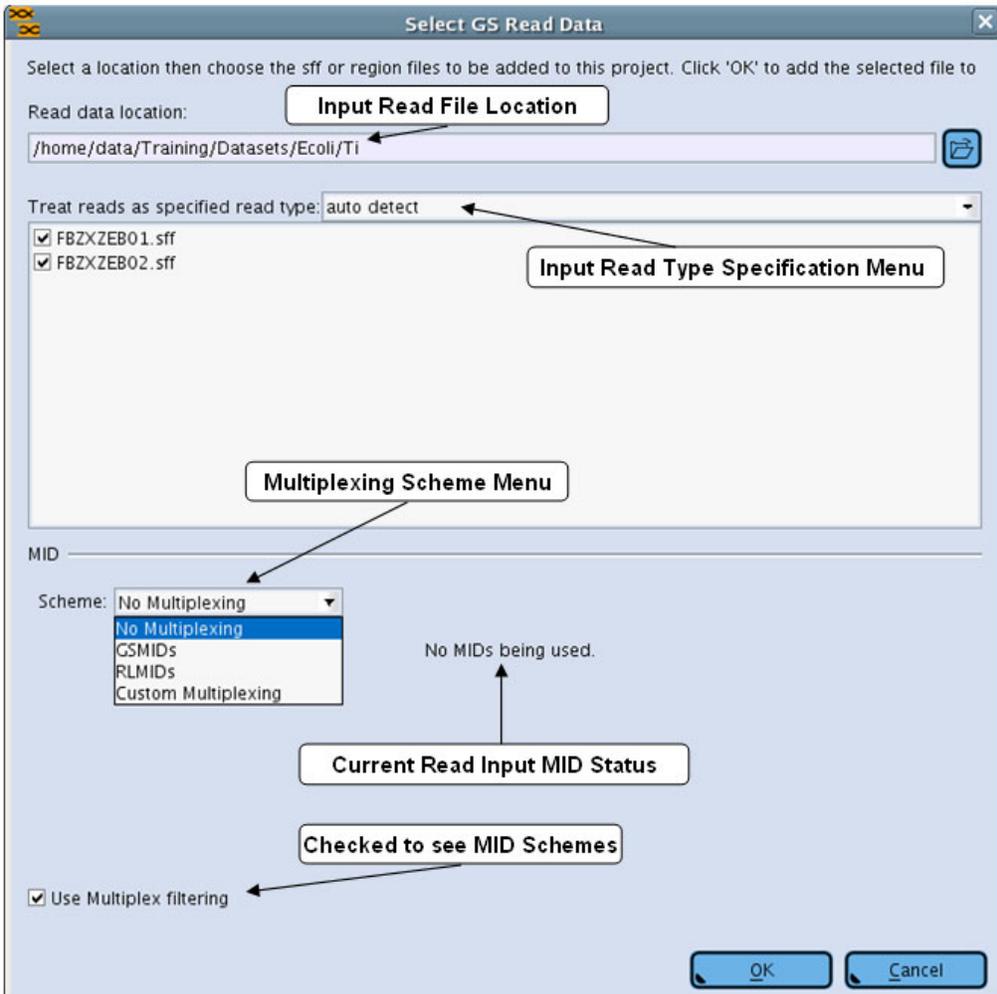


Figure 45: Select GS Read Data dialog with the MID Scheme options expanded

The MID controls on this window allow you to filter the reads in the selected Read Data files for inclusion in the Mapping project. Note, this filtering operation does not produce a different Mapping for each specified MID. Rather, the reads of the SFF files are scanned for the presence of MIDs, and the reads containing the selected MIDs are made eligible for mapping in the project. The selected MID filtering criteria are associated with the sff files when OK is pressed (Figure 45). Any reads from these file(s) without the specified MIDs will be ignored. To produce independent mappings for different MIDs (or sets of MIDs), you must create independent mapping projects for each of the desired MID subsets of the data.

MID filtering information is contained in 'MID schemes'. The choices are 'No Multiplexing', 'GSMIDs' for use of the original Genome Sequencer MID sets, 'RLMIDs' for the Rapid Library Preparation Kit MIDs, and 'Custom Multiplexing' if custom MIDs were used. By default, no MID/Multiplexing scheme is associated with Read Data files. To use MIDs, first select the desired MID Scheme (Figure 45) using the **Scheme** drop down menu. When an MID scheme is selected, a table of the MIDs present in that scheme is displayed, as shown in Figure 46. Use the checkboxes on the left to select or deselect the MIDs to be included in the mapping. The names, sequences and error limits of the MIDs selected will be used to filter the reads (from the

Read Data sets selected in the top part of the window). The result is that only the reads that contain the selected MIDs, within the number of errors specified in the scheme, will be included in the project.

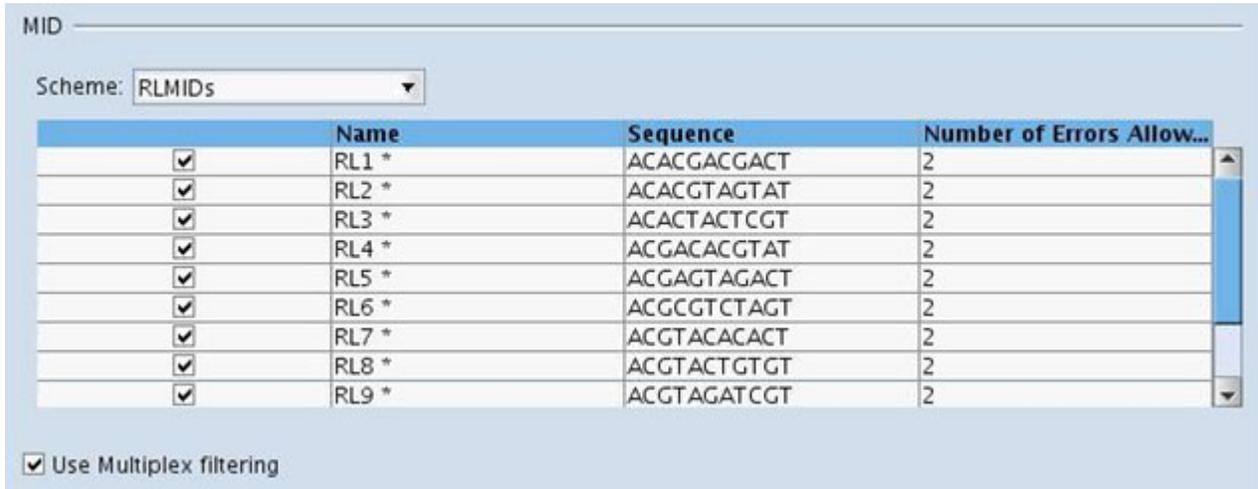


Figure 46: GS Read Import Dialog Rapid Library MID Support

For more information about MIDs, see Section 4.6.

Once the MID scheme has been specified, click the **OK** button to add the selected data files (with MID filtration information) to the list of GS read data files (see Figure 44).



In certain circumstances, you may want to apply a different MID selection to different Read Data sets, within the same project. For example, you may have multiple libraries each with a different strain of the same species, made with different MIDs but sequenced in the same run, that you want to map together. In this case, select the Read Data set(s) that need to be filtered with any given MID (or combination of MIDs), and click **OK** to add the MID-filtered data file(s) to the project. Back on the GS Reads sub-tab of the GS Reference Mapper window (with the GS Reads sub-tab active, see Figure 44), click the **Add** button (+) again, and select other Read Data sets, to be filtered with different MIDs.

2.7 Customize Project with the Parameters Tab

The Parameters Tab is organized as three sub-tabs: Input, Computation and Output (Figure 47). The input parameters and output data options that can be specified are different for the Genomic and cDNA sequence types. Each sequence type will have its own set of parameters under the project sub-tabs.

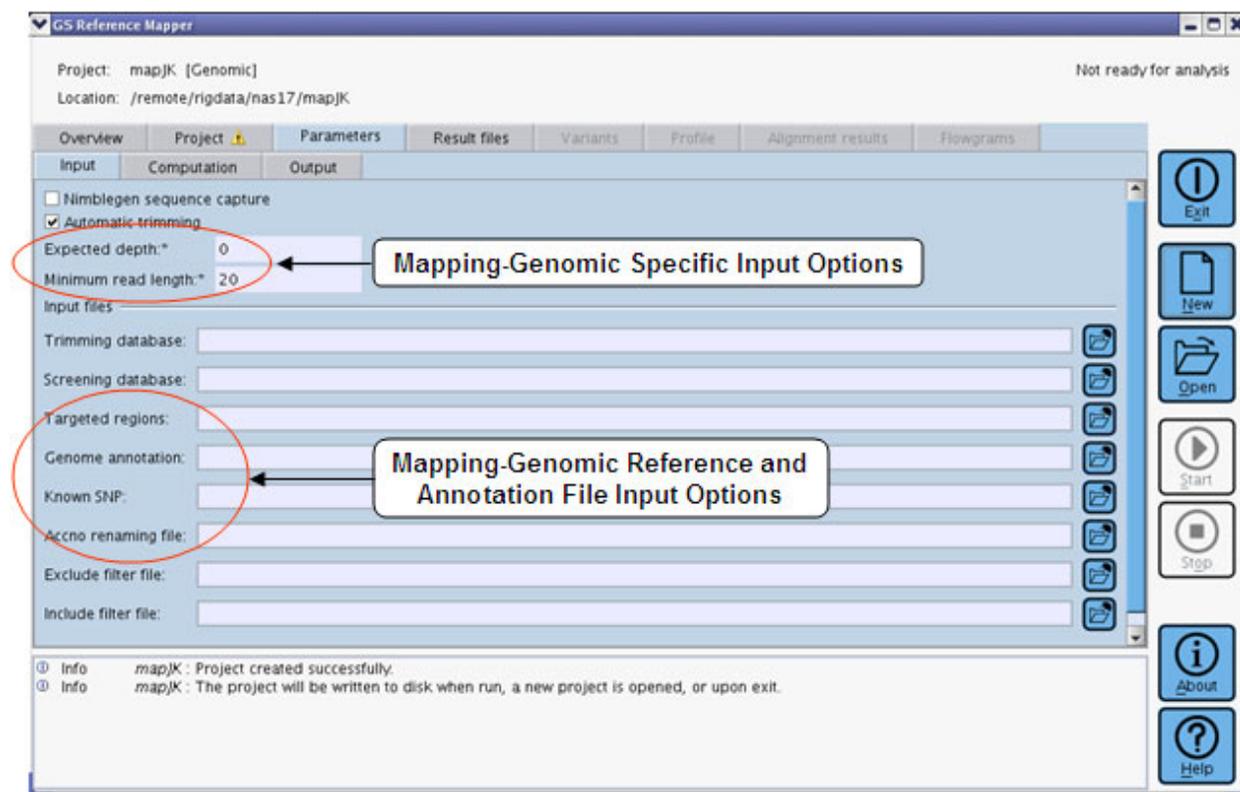


Figure 47: gsMapper Parameters tab, Input sub-tab for Genomic projects

For a new project, the mapping parameters are initially set to their defaults, as shown. If an invalid value is entered into any of these fields, an error indicator (red “X”) appears in the lower left corner of the field as well as on the tab heading, and the “Not ready for analysis” warning in the upper right hand corner of the screen. Pausing the mouse over the error indicator reveals a tooltip indicating the allowed values for the parameter(see section 2.14).

While default parameter values and settings are appropriate for most genomic and cDNA mapping projects, you may find some options useful for specific experiments.

2.7.1 Genomic Project Input Sub-Tab

The Input sub-tab is shown on Figure 47. It allows you to adjust the following settings:

- When the **Nimblegen sequence capture** checkbox is checked, the read data will automatically be primer-trimmed assuming NimbleGen Sequence Capture adapters are present at the beginning of the reads (this option defaults to unchecked). There is no need to use this option with NimbleGen’s Titanium-optimized protocol.



NOTE on trimming in general: new reads added to a project are trimmed when the mapping computation is performed. The trimming is performed using the parameter settings on the Project Input sub-tab at the time the computation is run. Reads for which the mapping computation has already been run are not re-trimmed.

- If the **Automatic trimming** checkbox is checked, the ends of new read files added to the project will be trimmed the next time a mapping computation is performed.
- **Expected depth** allows you to specify the number of reads expected to uniquely map to a single position of the reference genome. For high-depth mapping (where the expected coverage of a position in the genome could reach hundreds or thousands of reads), this option can be used to prevent low-frequency variations from being reported. If this option isn't used, many variations that are actually due to sequencing errors may be reported. The default value for this option is 0. A value of 0 or greater is allowed. If this option is set to a very low value (below 25), any variations that appear in only two reads may be output to the 454AllDiffs.txt file.
- The **Minimum Read Length** option can be used to change the minimum accepted read length to be used in the mapping (default is 20 bp, allowed value range is 15-45). This option is only applied if Paired End data is used.

The **input files** section of the tab allows you to specify the locations of several files that may be used in the mapping.

- An optional **Trimming database** is used to trim the ends of input reads (for cloning vectors, primers, adapters or other end sequences). Specify the path to a FASTA file of sequences to be used for this trimming (see Section 4.8).
- To use the **Screening database** option, set the path to a FASTA file of sequences to be used to screen the input reads for contaminants. A read that almost completely aligns against a sequence in the screening database is removed so that it is not used in the computation; if at least 15 bases of a read do not align to the screening sequence, no action is taken (see Section 4.8).
- The **Targeted regions** option can be used to direct the GS Reference Mapper to only report output for regions of the reference you specify and the reads aligned to those regions. The reads are mapped against the complete reference, but the output files will only report on the specified regions. The string value can be
 - a "accno:#-#" string, e.g. "chr9:10549-30594" specifies bases 10549 to 30594 of the chr9 reference sequence
 - the path to a file containing lines of "accno:#-#" strings (one per line)
 - the path to a GFF file created during a NimbleGen Sequence Capture experiment, describing the "primary_target_region" regions of the experiment
 This option may be useful when performing SeqCap or ChIPSeq projects.
- If you are performing a mapping against an annotated reference, you may have to specify the locations of the annotation files (unless a GoldenPath reference is being used)
 - To use the **Genome annotation** option, enter the path to an annotation file describing the gene/coding-region annotations for the reference sequences, so that the variation detection (AllDiffs, HCDiffs, and Structural Variations) can report gene names and protein translations of any identified variations in gene regions. The format of this file must match that of the GoldenPath "refGene.txt" file.

- The **Known SNP** option is used to specify the path to an annotation file describing the known SNP information for the reference sequences, so that the variation detection (HCDiffs tab) can link identified variations to the known SNP information. The format of this file must match that of the GoldenPath snp128.txt file.
- The **Accno Renaming File** option can be used to specify a renaming file to incorporate gene and transcript descriptions into the output files and/or to provide more meaningful names for genes and transcripts. For a description of the renaming file see Section 4.8



The software does three things if no Genome annotations or known SNP paths are specified:

- If there is a single reference file, the software looks for a file with the same name as the reference, but with a .refGene/.refLink/.snp* suffix as the annotations.
 - If the reference file(s) is (are all) in the same directory, the software looks in that directory for a “refGene.txt”, “refLink.txt” or “snp*.txt” file for the annotation files.
 - If the GOLDENPATH environment variable is set for a genomic mapping project, and the reference files are in a “chromosomes” directory inside the GOLDENPATH tree, the software looks for a “database” directory at the same level as the “chromosomes” directory, and then looks for the annotation files inside the database directory. As a consequence the situation may arise whereby a project gets annotated even if no annotation file has been specified in the parameters tab.
- The **Include** and **exclude filter file** options can be used to specify a file of sequences to specifically include or exclude in the computation. The file format and functionality the same as the `-e` and `-i` options of sfffile, Section 3.1.

2.7.2 Project Computation Sub-Tab

The Computation sub-tab is shown on Figure 48. It allows you to adjust the settings described below.

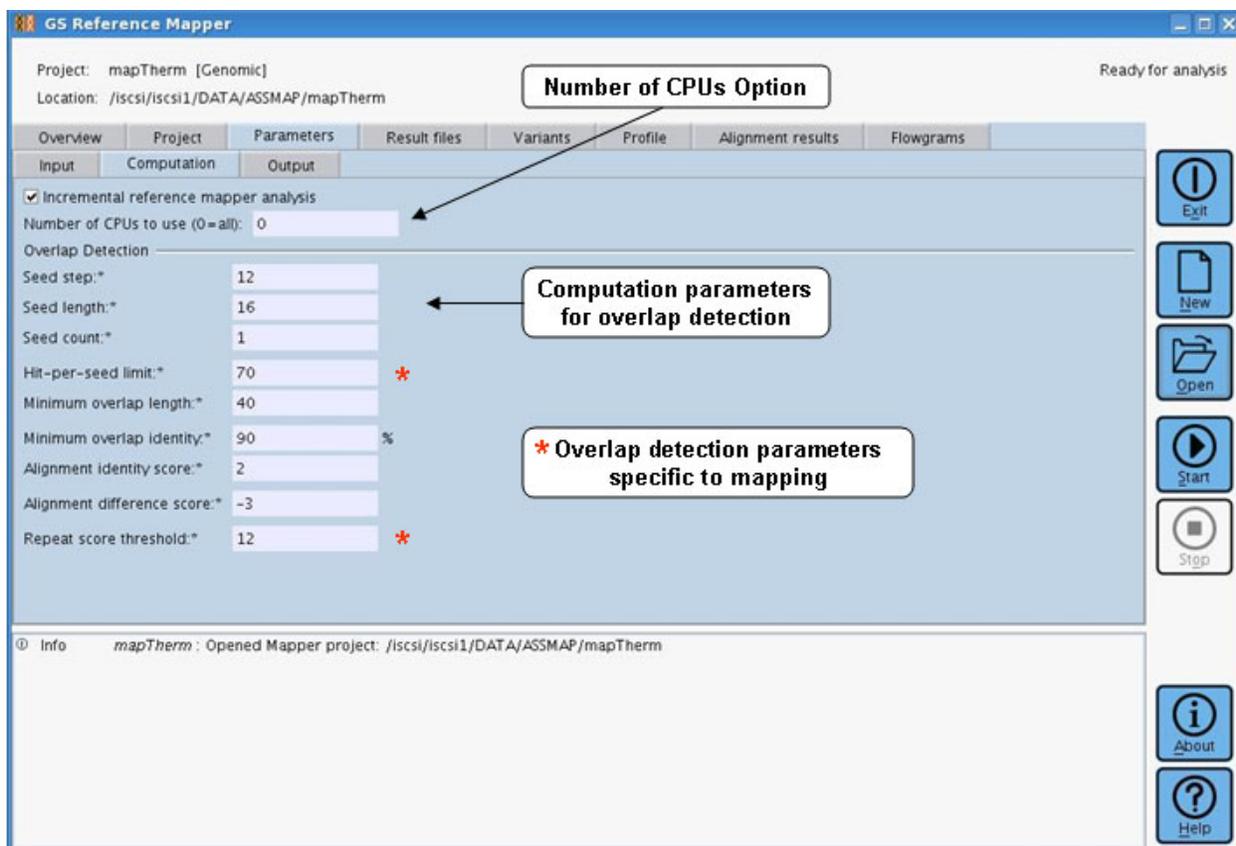


Figure 48: gsMapper Parameters tab, Computation sub-tab for Genomic projects

- **Incremental reference mapper analysis** – Results from computations performed with this option selected (checked) will serve as the basis for the mapping of additional reads. If a project computation is performed with this option deselected, all the project data will be mapped anew each time it is computed, i.e. forcing all alignments to be recalculated and overwriting any existing results.
 - Default: selected (Note: This setting is not saved with a project, and is always reset to “selected” when a project is re-opened).
- The **number of CPUs** option can be given to limit the load of the software on the computing resources. The default is 0, which specifies that all of the CPUs on the computer should be used. The computation uses the CPU setting to create an approximate load on the computer, so the actual load may vary somewhat from this number, i.e., using “-cpu 3” may cause a load from 2 to 4 on the system.



Currently, only the “Computing alignments...” and “Generating output...” phases have been parallelized (because they are the most computationally intensive phases for larger projects). Other phases of the computation will still use single threading. Also, note that only CPUs which have access to the same physical memory can be used.

The current memory footprint is roughly 4 bytes per reference base plus 2 bytes per input (read) base. For example a 1 GigaBase genome with a 20x coverage would need 44 GigaBytes of physical memory

- **Overlap Detection Parameters**
 - **Seed step** – The number of bases between seed generation locations used in the exact k-mer matching part of the overlap detection
 - Default value: 12
 - Allowed values: 1 or greater
 - **Seed length** – The number of bases used for each seed in the exact k-mer matching part of the overlap detection (i.e. the “k” value of the k-mer matching)
 - Default value: 16
 - Allowed values: 6-16
 - **Seed count** – The number of seeds required in a window before an extension is made
 - Default value: 1
 - Allowed values: 1 or greater
 - **Hit-per-seed limit** – A filtering limit on the seeds found during the exact k-mer matching (if more than this number of seed matches are found for a lookup position in the query sequence, those seed matches are not tested as candidate alignments). If more than 70% of a read’s seeds have more than this number of seed hits, the read is not used and is classified as Repeat.
 - Default value: 70
 - Allowed values: 0 or greater
 - **Minimum overlap length** – The minimum length of overlaps used by the mapper
 - Default value: 40
 - Allowed values: 1 or greater
 - **Minimum overlap identity** – The minimum percent identity of overlaps used by the mapper
 - Default value: 90
 - Allowed values: 0-100
 - **Alignment identity score** – When overlaps are extended during an alignment, this is the per-overlap-column identity score used to calculate the overall score for the alignment
 - Default value: 2
 - Allowed values: 0 or greater
 - **Alignment difference score** – When overlaps are extended during an alignment, this is the per-overlap-column difference score used to calculate the overall score for the alignment
 - Default value: -3
 - Allowed values: 0 or less
 - **Repeat score threshold** – The threshold used to declare alignments “unique” or “multiple”. If an input read aligns to more than one position in the reference sequence(s), an alignment score is calculated for each alignment by summing the identity or difference score for each alignment column; if the best scoring alignment is larger than any other by at least this threshold value, it is marked as “uniquely mapped”. Otherwise, it is “multiply mapped” or “Repeat”.
 - Default value: 12
 - Allowed values: 0 or greater



When a read is mapped, exact k-mer matches between the read and the reference yield a set of overlaps between the two. These overlaps are then extended into longer alignments. The best alignment is used to determine the mapping location of the read. In some cases, non-overlapping portions of a read may map to different loci in the reference(s). Together these mappings may form a chimeric best alignment for the read.

2.7.3 Project Output Sub-Tab

The Output sub-tab is shown on Figure 49. It allows you to adjust the settings described below.

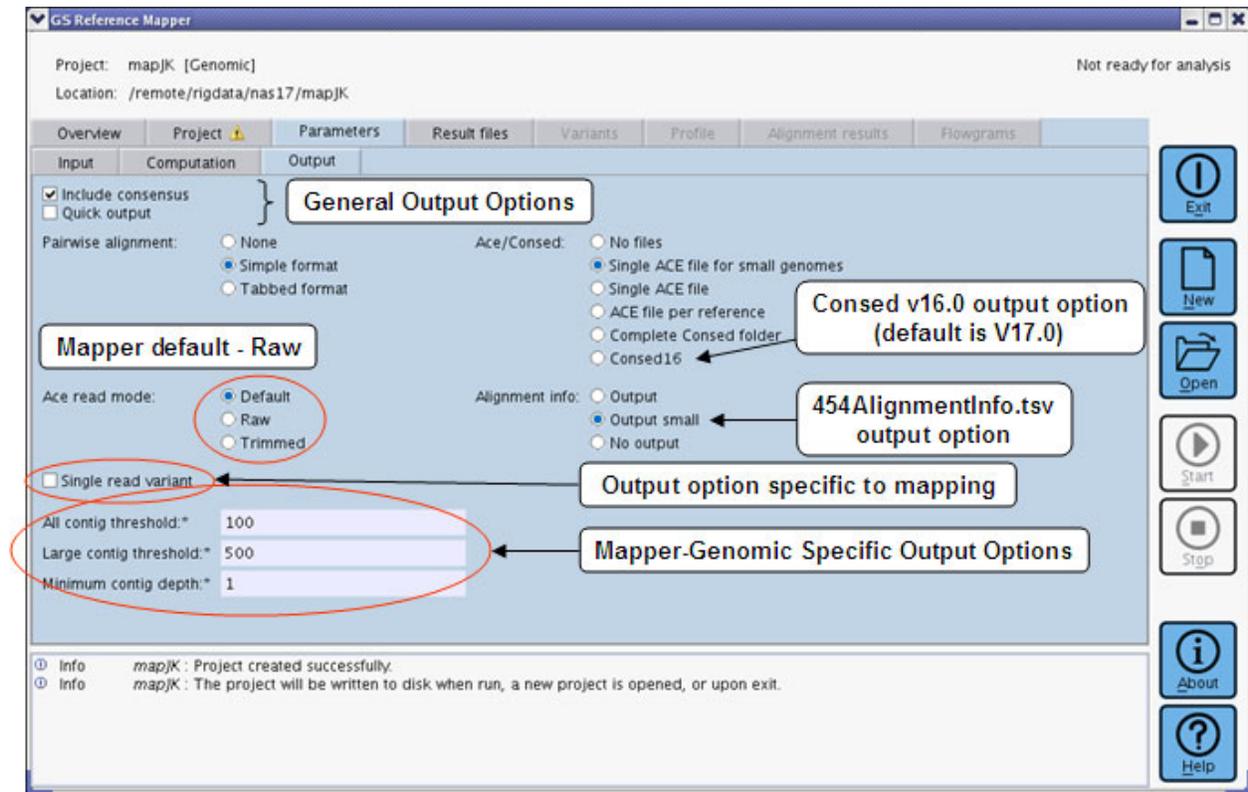


Figure 49: gsMapper Parameters tab, Output sub-tab for Genomic projects

- **Include Consensus** – If this check box is selected, the application will generate all the output files, including the output files related to the generation of contigs and consensus sequence information. If it is not selected, the application will perform the mapping and will output files associated with the creation of the multiple-alignments, but will not output files or metrics involved with contig or consensus information.
 - Default: selected (Note: This value is not saved with a project, and is always reset to “selected” when a project is re-opened).
- **Quick output** option generates output faster than using the default setting by disabling the “Computing signals...” computation. The default behavior of the assembler requires that all reads be re-read so that signal and quality information can be used to compute consensi. If this option is selected, the accuracy of the consensus may be degraded (i.e.,

more consensus errors may be generated) as the distribution statistics are not generated to assist in the basecalling step.

- The **Ace/Consed** option determines whether or what form of ACE file(s) are output by the application. The default is “Single ACE file for small genomes”. See section 2.17.1.5 for a description of the 454Contigs.ace file and of the ace and consed directories.
 - No files – no ACE file is generated
 - Single ACE file for small genome – a single ACE file is generated if fewer than 4 million reads are input to the mapping and the reference is less than 40 Mbp
 - Single Ace file – a single ACE file is generated containing the multiple alignments of all contigs
 - ACE file per contig – a separate ACE file is generated for each contig in the mapping
 - Complete Consed folder – a “consed” folder is generated, containing all the directories and files necessary to display the data in the consed software
 - Consed 16 – Ace/Consed generates files consistent with version 17.0 or higher of consed by default. This option generates files consistent with version 16.0 or earlier.
- The **Alignment Info** option has 3 possible settings.
 - **Output** – turns on generation of the 454AlignmentInfo.tsv file
 - **No output** – suppresses the generation of the 454AlignmentInfo.tsv file
 - **Output small** – the file will be generated unless there are more than 4M reads or the mapping generates contigs with a total length in excess of 40Mbp.
- **Pairwise Alignment** – Determines whether the 454PairAlign.txt file is output; this file contains the overlaps used by the mapper.
 - None – the file is not generated
 - Simple format – the file contains a human-readable view of the alignments
 - Tabbed format – the file contains tab-delimited lines of the overlaps
 - Default is “None”. See section 2.17.1.8 for a description of the 454PairAlign.txt file.
- **Ace read mode** – When the ACE file is generated, output reads using either the raw, complete basecalled read or using just the trimmed portion of the reads (after low quality, vector and key trimming)
 - Default is set to output raw
 - Raw is to output the entire sequence of reads
 - Trimmed is to output trimmed reads
- **All contig threshold** – The minimum number of bases for a contig to be output in the 454AllContigs.fna file
 - Default: 100
- **Large contig threshold** – The minimum number of bases for a contig to be output in the 454LargeContigs.fna file
 - Default: 500
- **Minimum contig depth** – regions of references covered by at least this many reads are considered contigs for output purposes

For a full listing and description of the output options for mapping, please see Table 4: Mapping and Assembly options and Table 6: Options Specific to Mapping, in the Appendix section.

2.7.4 cDNA Project Input Sub-Tab

The cDNA Mapping project parameters on the Input sub-tab are the same as for Genomic mapping projects with the exceptions noted below.

- cDNA Mapping contains a section for specifying the **Reference Type** as **cDNA**, **Genomic**, or **Auto** (auto-detect).

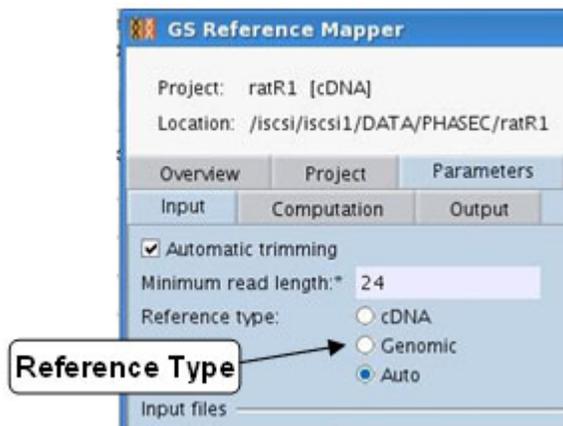


Figure 50: gsMapper Parameters tab, Input sub-tab for cDNA projects

The other two sub-tabs of the Parameters tab are identical for Genomic and cDNA projects.

2.8 Computing the Mapping

2.8.1 Computing a Mapping for the First Time

When at least one Read Data file and one Reference file have been added to the project and all parameters are within acceptable limits, the **Start** button becomes enabled and the “Ready for analysis” message appears in the upper right corner of the application window (see Figure 49). The projects parameter settings are also saved when the **Start** button is clicked. Click the **Start** button to save the project parameter settings and carry out the mapping computation of the current project, *i.e.* as defined in the Project and Parameters tabs. As the computation progresses, the **Start** button is disabled, periodic progress messages appear in the progress box, at the bottom of the application’s main window, and the current stage of the mapping computation along with a progress bar are displayed in the upper right corner of the window. When the mapping computation is complete, the **Start** button is re-enabled and the message “Ready for analysis” appears again in the upper right corner of the application window.

2.8.2 Re-Computing a Mapping

After a mapping has been performed on a project, the mapping computation can be repeated on that project - with the same Read Data or after addition or removal of one or more files, with the

same mapping parameter settings or with different ones by pressing the **Start** button. If the “Incremental reference mapper analysis” checkbox on the Parameters tab is checked and no change has been made to the Reference, this computation will use the current parameters settings when mapping any reads new to the project; this re-analysis will take much less time than the initial analysis, since previously mapped will remain aligned to their existing reference locations. Changes to some parameter settings (such as those on the Input and Computation sub-tabs of the Project Tab) will have no effect on reads already included the prior mapping computation

2.8.3 Stopping a Mapping Computation

Click on the **Stop** button while a mapping project computation is in progress to halt the computation. The effect may not be instantaneous, however. Any delay will be a function of the complexity of the mapping being performed and of the stage of the computation at the time it was interrupted. An informational message appears in the progress box at the bottom of the application’s main window, containing the text “Warning: stopRun called for this project. Stopping...” When the mapping computation has been halted, the **Stop** button is disabled and the **Start** button re-enabled.

2.8.4 Information Updated when Mapping Completes

After a successful mapping, the data for the various columns of the reads in the GS reads and FASTA reads sub-tabs will be updated (Figure 51). Placing the mouse pointer over any cell in the table brings up a tool tip with additional information about that item. A successful mapping also updates data in the Overview tab. Project information is also saved when the computation completes.

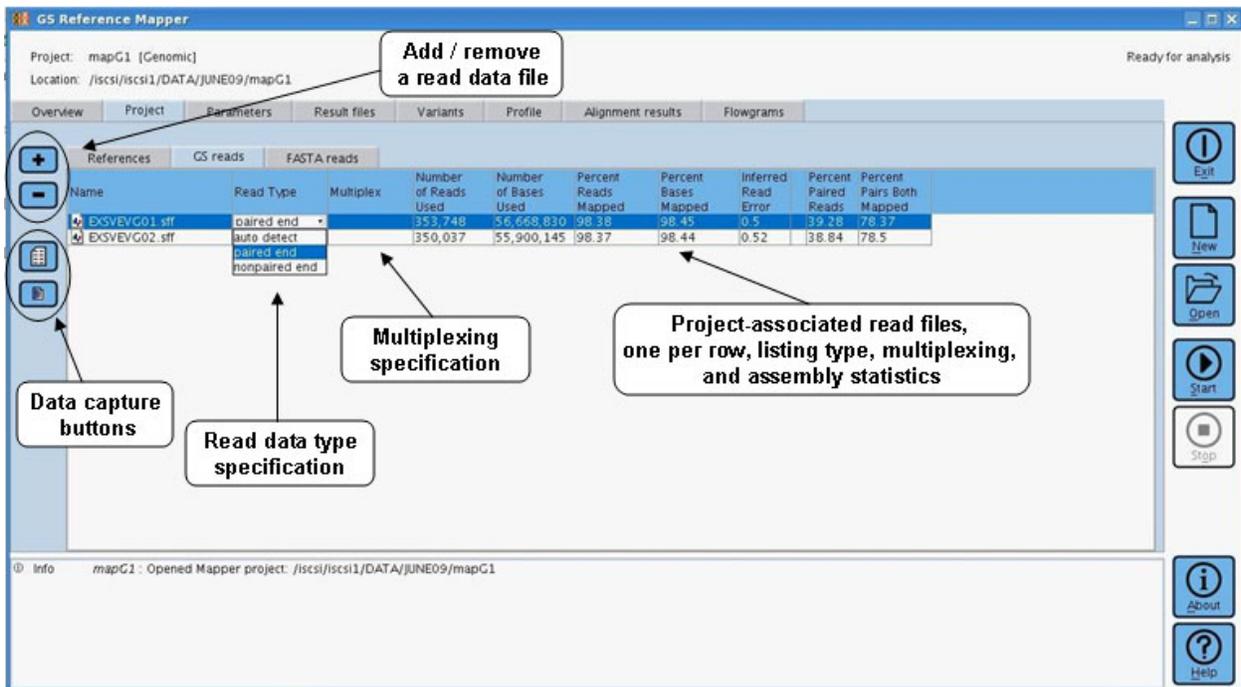


Figure 51: Project Tab of the gsMapper (GS Reads sub-tab), after completion of a mapping computation

After a mapping project has been computed, the mapping results can be viewed on the Overview, Project, Results Files, Variants, Profile, Alignment Results, and Flowgram tabs of the Reference Mapper application's main window.

2.9 Viewing Mapping Output with the Result Files Tab

The Result Files tab allows you to view the various files generated by the GS Reference Mapper software (described in detail in 2.17.1). Using the list in the left-hand panel of the tab, click on the name of the output file you want to examine; its content will be displayed on the right-hand panel of the tab (Figure 52).



For very long files, such as the sequences of all contigs mapped to a large genome, this view displays the file truncated to 50,000 lines. When this occurs, a note to that effect appears at both the beginning and at the end of the display. The file itself remains intact, however.

Project: mapGreen [Genomic] Ready for analysis
 Location: /iscsi/iscsi1/DATA/ASSMAP/mapGreen

>Reference	Start	End	Ref	Var	Total	Var		
>Accno	Pos	Pos	Nuc	Nuc	Depth	Freq		
>gi 49175990 ref NC_000913.2			246	246	T	A	12	17%
Reads with Difference:								
gi 49175990 ref			218+	CCACCATCACCATTACC-ACAGGTAACGG-TGCGGGCTGACG-CGT-ACAGGAAACACAG	273		**	
FLTF6DL01C260E			130+	CCACCATCACCATTACC-ACAGGTAACGG-TGCGGGCTG	166		**	
FLTF6DL01EGIML_			71+	CCACCATCACCATTACC-ACAGGTAACG-A-GCGG	102		**	
Other Reads:								
FLTF6DL02GBP1H			87-	CCACCATCACCATTACC-ACAGGTAACGG-TGCGGGCTGACG-CGT-ACAGGAAACACAG	32		**	
FLTF6DL01D05IR_			134+	CCACCATCACCATTACC-ACAGGTAACGG-TGCGGGCTGACG-CGT-ACAGG	181		**	
FLTF6DL01EGF15			124+	CCACCATCACCATTACC-ACAGGTAACGG-TGCGGGCTGACG-CGT-ACAGGAAACACAG	179		**	

Info mapGreen : Opened Mapper project: /iscsi/iscsi1/DATA/ASSMAP/mapGreen

Figure 52: gsMapper Result Files tab for Genomic projects

Figure 52 above shows the Result Files tab for a Genomic Mapping project. The result files for a cDNA project are shown below (Figure 53). The result file contents are described in detail in Section 2.17.1.

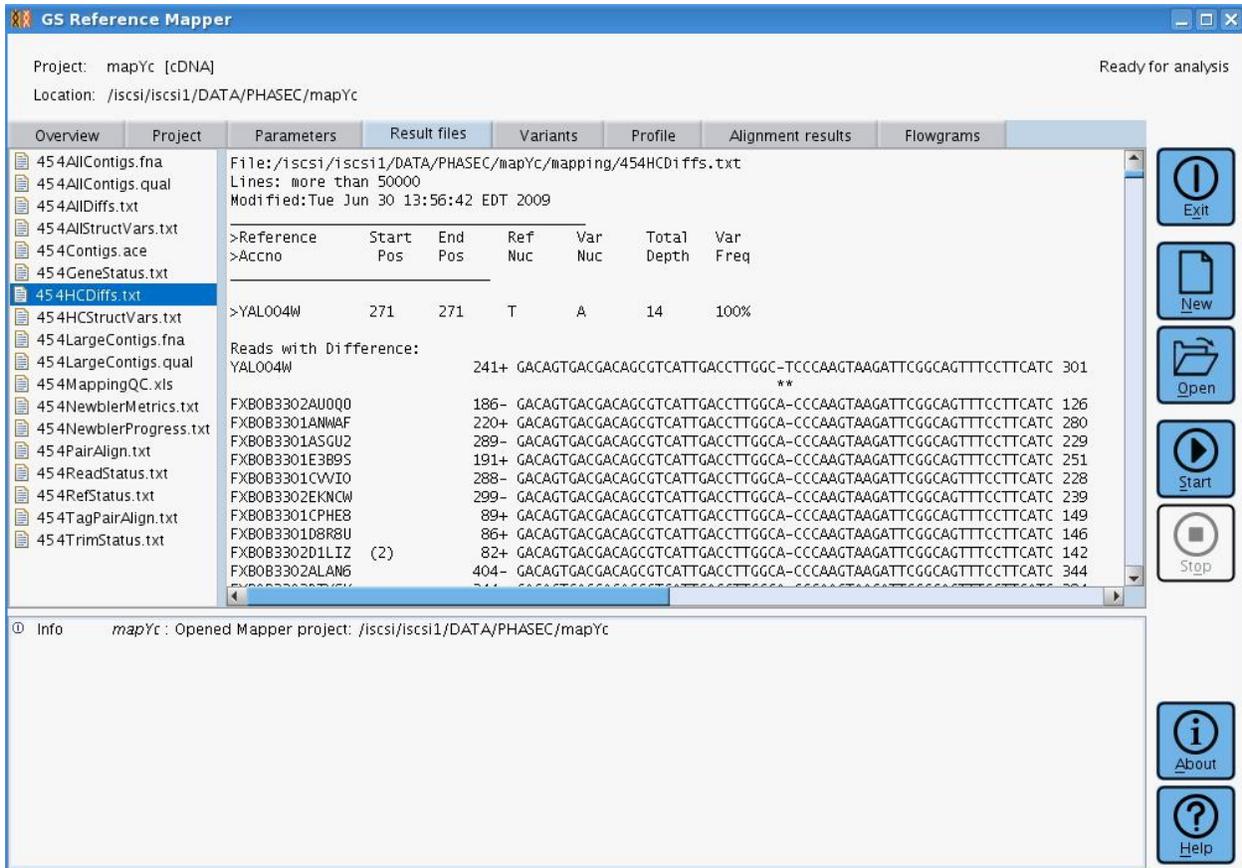


Figure 53: gsMapper Results Files tab for cDNA projects

2.10 Using the Variants Tab

There are two sub-tabs on the Variants tab that contain information about differences (relative to the reference) that are indicated by alignments of reads mapped to the reference: one reports on nucleotide differences and the other on structural variations. Both sub-tabs contain a table of variations showing statistics, loci, and (optionally) annotations. Using the mouse, you can right-click any of the entries and choose to view the alignment for the selected variation.

The two sub-tabs are described in more detail in the sections below; they share the following common features:

- **Sorting:**
 - All columns are sortable by clicking on the column header. A small triangle appears to indicate the direction of the sort (see Figure 54 and Figure 55, below). Clicking on a column header of a column that is already sorted will reverse the order of the sort.
 - If multiple rows have the same value for the sorted column, these rows stay in the order in which they were before the sort, relative to one another. This allows for “nested sorting”. For example, if you
 - sort first based on “Start Position in Ref”,
 - then sort based on “Reference Accession Number”,

- results in a table where the high confidence differences are grouped by reference sequence, and the ones that belong to the same reference are displayed in order of their position on the reference sequence.
- **Summary tooltip** (mouse-over information):
 - Pausing the mouse cursor over any row provides a tooltip that summarizes the basic statistics of the high confidence difference described in that row (see Figure 54). In particular, the bases involved in the variation coupled with the total variation percentage, the depth of sequencing, and the separate statistics for both forward and reverse reads are presented.
- **Links to read alignment data:**
 - Right-clicking in the row for a particular HCDiff or Structural Variant opens a contextual menu with a single item: “Show Alignment”. Selecting this loads the Alignment results tab with the alignment that supports the variation in question. The first base of the variation appears as the leftmost base in the view.
 - You can also select multiple variants, then right-click to load these variations into a list. The Alignment results tab will display the first variation in the list and the list itself will comprise the dropdown menu contents of the Positions of interest (Figure 59).
 - You can then scroll the alignment to examine the context of the variation.



Note on duplicate reads handling when computing Variants:

Duplicate reads of a single DNA input are a known potential artifact of the emPCR amplification process which may occur, for example, when two or more beads are amplified in a single emulsion microreactor. In an attempt to compute more accurate variation percentages, the GS Reference Mapper, by default, groups individual reads into groups of duplicates (reads that start at the same base of the reference sequence). Groups of duplicate reads count as only one read in the computation of variants (HCDiffs and Structural Variants). However, the corresponding alignments on the Alignment results tab display all the reads whether or not they were considered duplicates. When grouping occurred, the variant summary tooltip provides the statistics for both grouped and for individual reads (see Figure 54).

On the command line, the “-ud” option (see Table 4, in Appendix) instructs the GS Reference Mapper to treat all reads individually and to not perform this grouping operation. This option is not directly available through the GUI, but if a project executed from the command line were computed in that fashion and then displayed in the GUI, the values in the HCDiffs table would be based on the ungrouped, individual reads instead of the default grouped duplicate reads.

2.10.1 The HC Diffs Sub-Tab

Information about local differences between reads and the reference appear in a table on the **HC Diffs** sub-tab. Explanations for each of the columns in the table are found in the description of the 454AllDiffs.txt file, in Section 2.17.1.13. Right-clicking on a row opens a contextual menu from which one can access the alignment that underlies the variation. Pausing the mouse over a row displays a tooltip providing information about that variation.

The tab can be divided conceptually into three main functional sections (Figure 54):

- The leftmost columns (from Reference Accession Number to Total Depth) contain general information on the high confidence differences (HCDiffs) that were found in the computed data, between the reads and the reference sequences.
- The center columns (from Reference Amino Acids to Known SNP Info) show gene annotation or known SNP file information on the regions of the HCDiffs. For such information to be available, Genome Annotation and Known SNP databases must have been specified (either on the Parameters Tab or by using a Golden Path reference sequence).
- The remaining columns on the right (from Percent Forward to Total Num Reverse Reads) contain a detailed breakdown of the computation data shown on the left.

High Confidence differences listed by Reference Accession Number

Sort order toggle – appears for each column when heading is clicked on

Reference Accession Number	Start Position in Ref	End Position in Ref	Reference Bases	Variation Bases	Total Variation Percent	Total Depth	Reference Amino Acids	Variation Amino Acids	Coding Frame	Region Name	Known SNP Info	Percent Forward With Vari
chr3	16,238,924	16,238...	G	C	100.0	7						100.0
chr3	16,243,773	16,243...	C	G	47.1	17						25.0
chr3	16,243,978	16,243...	T	A	57.9	19						33.3
chr3	16,243,990	16,243...	C	A	62.5	16						42.9
chr3	16,280,678	16,280...	C	G	33.3	12						33.3
chr3	16,281,022	16,281...	C	T	100.0	13						100.0
chr3	16,287,568	16,287...	C	T	71.4	7						100.0
chr3	16,302,859	16,302...	T	G	43.5	23						38.5
chr3	16,302,913	16,302...	C	T	58.3	24						61.5
chr3	16,318,096	16,318...	T	G	100.0	5						100.0
chr3	16,318,509	16,318...	T	C	100.0	11						100.0
chr3	16,333,588	16,333...	T	C	45.5	11						87.5
chr3	16,333,906	16,333...	A	G	100.0	4						100.0
chr3	16,394,712	16,394...	G	A	80.0	5						100.0
chr3	16,425,876	16,425...	T	C	100.0	4						66.7
chr3	16,450,219	16,450...	G	T	37.5	8						66.7
chr3	16,510,101	16,510...	C	T	60.0	5						66.7
chr3	16,510,233	16,510...	C	T	45.5	11						66.7
chr3	17,025,997	17,025...	G	C	85.7	7						66.7
chr3	17,026,332	17,026...	T	C	80.0	15						66.7
chr3	17,028,503	17,028...	A	G	85.7	7						66.7
chr3	17,031,035	17,031...	G	A	100.0	4						66.7
chr3	17,106,214	17,106...	C	A	60.0	5						66.7

Info hcdiffs : Opened Mapper project: /iscsi/iscsi1/DATA/OCT/hcdiffs

Figure 54: gsMapper Variants tab, HC Diffs sub-tab

2.10.2 The Structural Variations Sub-Tab

The Structural Variations sub-tab shows locations of larger-scale changes relative to the reference that are indicated by a group of reads. Entries in the table are classified as either Rearrangement points or Rearrangement regions. See Section 2.17.1.15 for more details on

these topics. Right-clicking on a row opens a contextual menu from which one can access the alignment that underlies the variation. Pausing the mouse over a row displays a tooltip providing information about that variation.

A

Structural Variants listed by Reference Accession Number

Sort order toggle – appears for each column when heading is clicked on

(side 1) Reference Accession Number	(side 1) Reference Position	(side 1) Variator Direction	(side 1) Region Name	(side 2) Reference Accession Number	(side 2) Reference Position	(side 2) Variator Direction	(side 2) Region Name	Higher Side Variator Percent	Higher Side Depth	Deviatio Length	Variator Type	(side 1) Total with Variator Percent	(side 1) Forward with Variator Percent
gij49175990[ref]...	1	<--		gij49175990[ref]...	4,639,675	-->		100.00	8	4,63...	Point	88.9	100.0
gij49175990[ref]...	46	<--		gij49175990[ref]...	4,639,624	-->		96.90	129	4,63...	Region		
gij49175990[ref]...	257,900	<--						88.31	7		Point	85.7	100.0
gij49175990[ref]...	257,907	-->							14		Point	100.0	100.0
gij49175990[ref]...	1,206,541	-->		gij49175990[ref]...	1,208,759	-->		49.55	113	2,217	Region		
gij49175990[ref]...	1,207,013	<--		gij49175990[ref]...	1,208,842	<--		35.71	14	1,828	Point	35.7	37.5
gij49175990[ref]...	1,207,057	<--		gij49175990[ref]...	1,209,589	<--		27.18	103	2,531	Region		
gij49175990[ref]...	1,298,718	<--						36.36	11		Point	36.4	42.9
gij49175990[ref]...	1,298,721	-->						100.00	7		Point	100.0	100.0
gij49175990[ref]...	1,871,056	<--						100.00	11		Point	100.0	100.0
gij49175990[ref]...	1,871,063	-->						87.50	8		Point	87.5	100.0

B

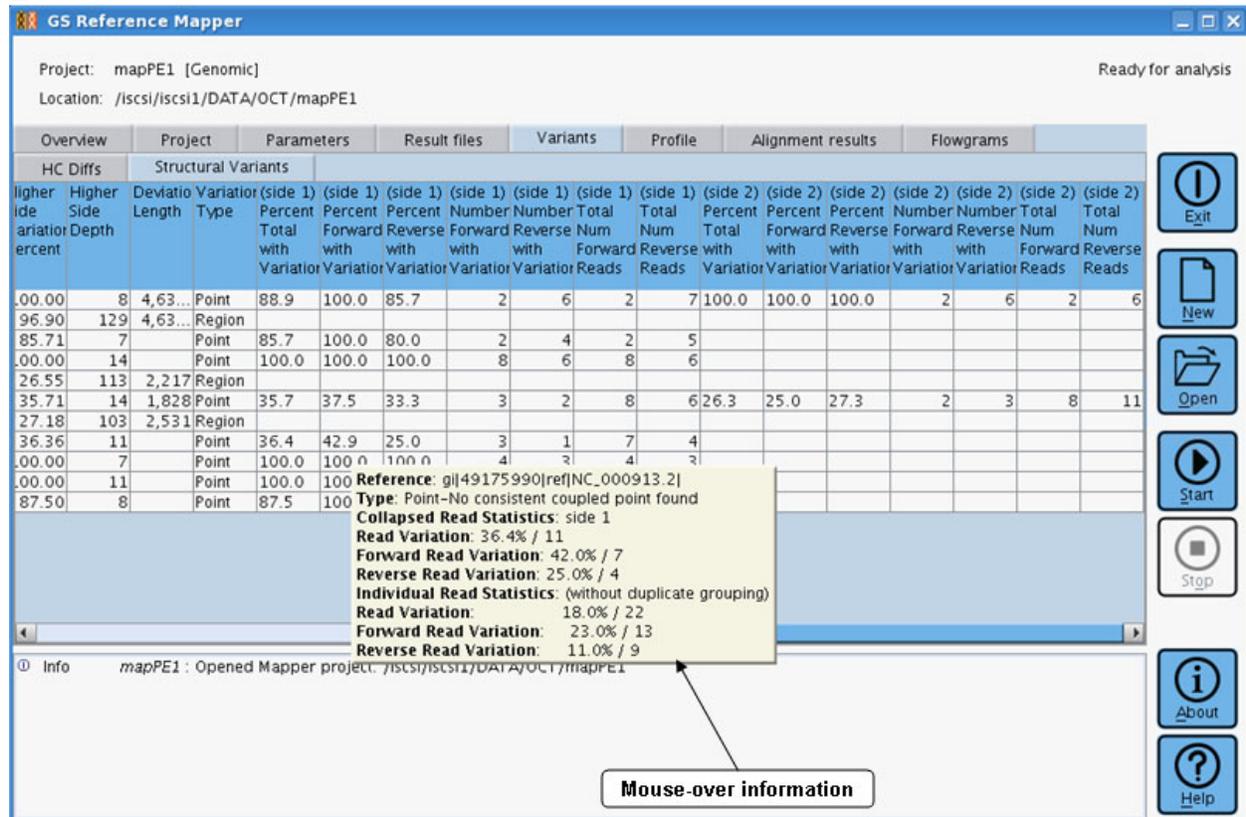


Figure 55: gsMapper Variants tab Structural Variants sub-tab. A: Left columns; B: Right columns.



Note on handling Paired End reads when computing Structural Variants:

The detection of Rearrangement Regions currently depends on finding clusters of Paired End reads considered to be False Pairs. When Paired End libraries using different span distances are used (e.g. 3kb and 8kb or 20kb libraries), more than one cluster may be found for the same structural variation.

2.11 Profile Tab

The Profile tab (Figure 56) contains sub-tabs for **Ref Status** and **Gene Status** (for cDNA Mapping projects only). These sub-tabs display statistics for reads mapping to each reference or gene. See Sections 2.17.1.12 and 2.17.2.1 for column descriptions.

Both sub-tabs contain tables that can be sorted.

**Note on duplicate reads handling when computing Variants:**

Duplicate reads of a single DNA input are a known potential artifact of the emPCR amplification process which may occur, for example, when two or more beads are amplified in a single emulsion microreactor. In an attempt to compute more accurate variation percentages, the GS Reference Mapper, by default, groups individual reads into groups of duplicates (reads that start at the same base of the reference sequence). Groups of duplicate reads count as only one read in the computation of variants (HCDiffs and Structural Variants). However, the corresponding alignments on the Alignment results tab display all the reads whether or not they were considered duplicates. When grouping occurred, the variant summary tooltip provides the statistics for both grouped and for individual reads (see Figure 54).

On the command line, the “-ud” option (see Table 4, in Appendix) instructs the GS Reference Mapper to treat all reads individually and to not perform this grouping operation. This option is not directly available through the GUI, but if a project executed from the command line were computed in that fashion and then displayed in the GUI, the values in the HCDiffs table would be based on the ungrouped, individual reads instead of the default grouped duplicate reads.

- **Sorting:**
 - All columns are sortable by clicking on the column header. A small triangle appears to indicate the direction of the sort. Clicking on a column header of a column that is already sorted will reverse the order of the sort.
 - If multiple rows have the same value for the sorted column, these rows stay in the order in which they were before the sort, relative to one another. This allows for “nested sorting”, as described in Section 2.10.

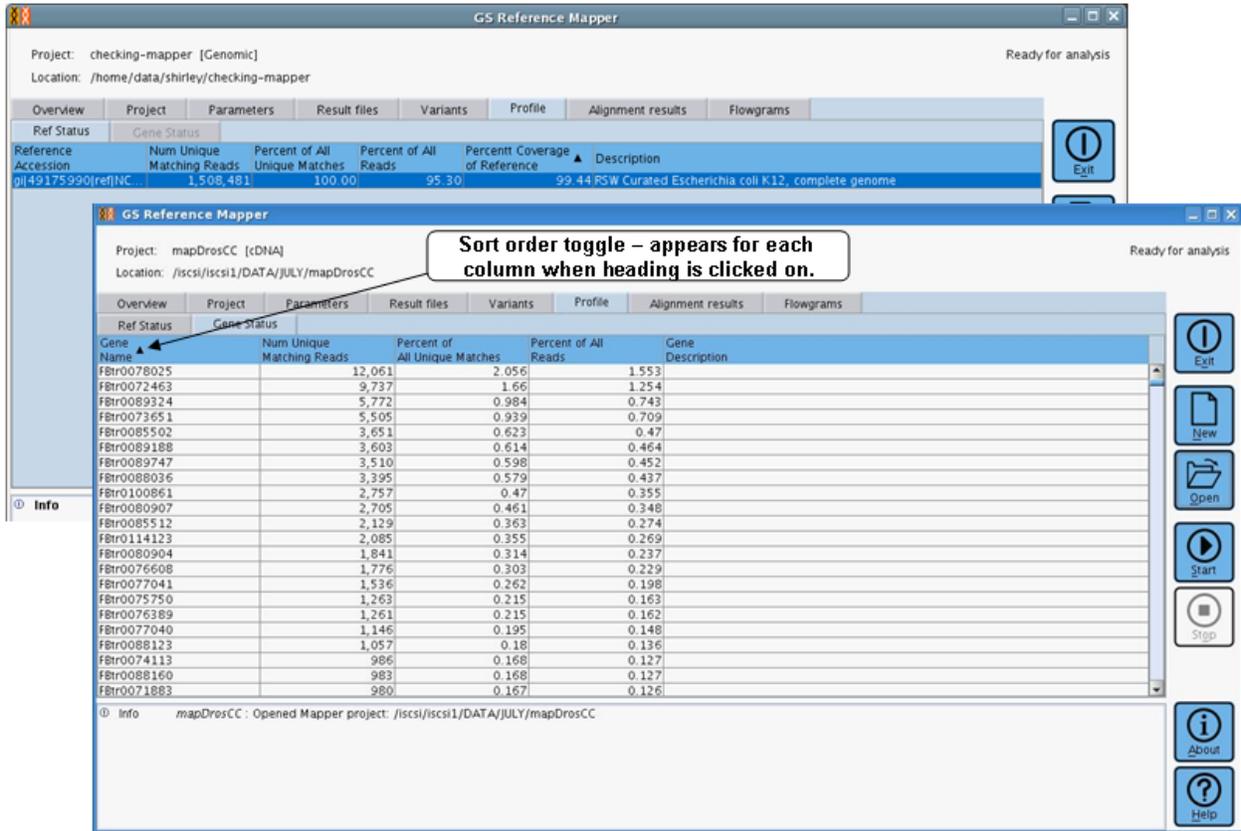


Figure 56: gsMapper Profile Tab Ref Status and Gene Status Sub-tabs

2.12 Viewing Reads Mapped to the Reference with the Alignment Results Tab

Click on the Alignment results tab to view the alignment data from the mapping computation (Figure 57). This tab shows the multiple alignments underlying the mapping of the reads to the reference.

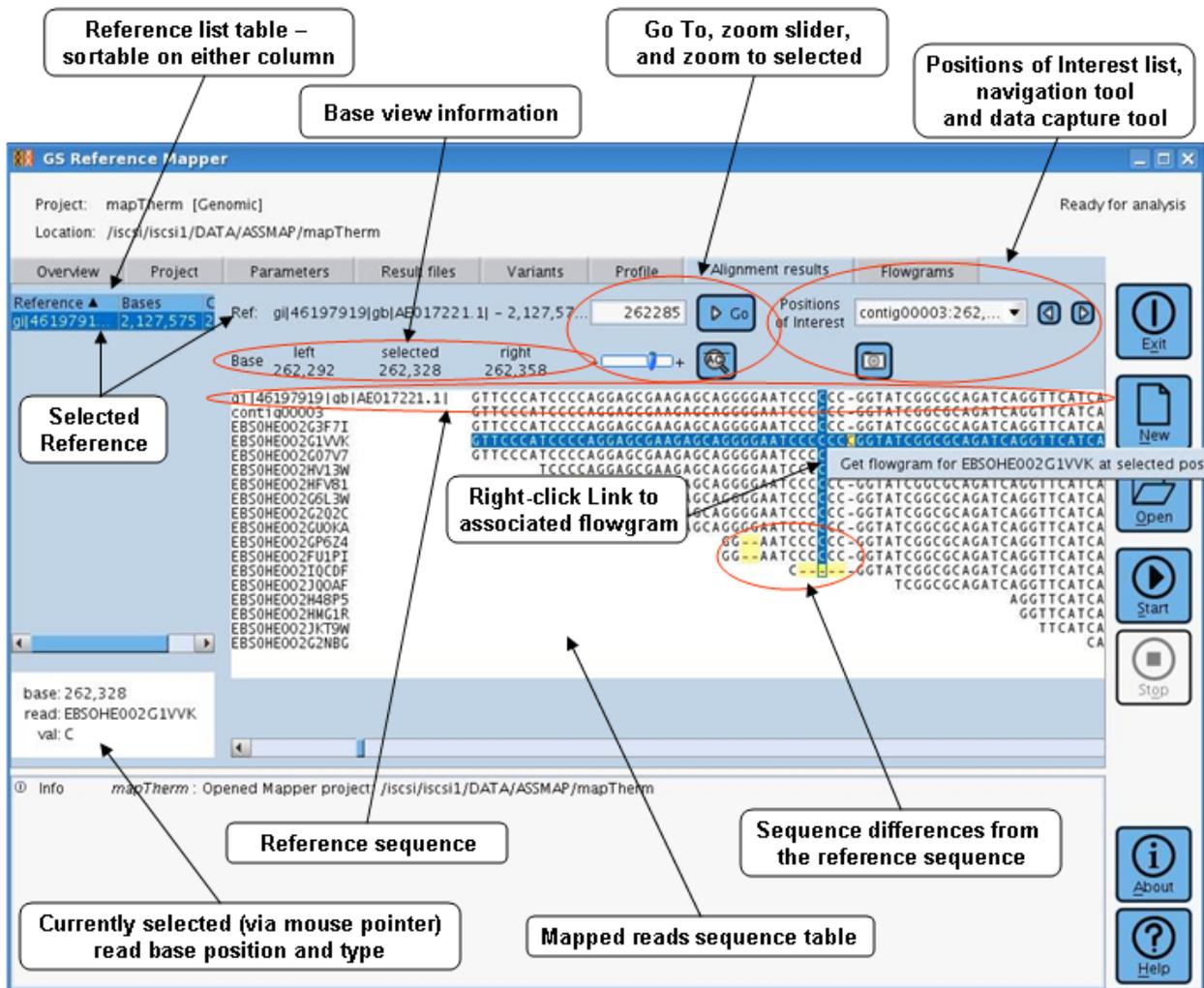


Figure 57: gsMapper Alignment results tab

- The left-hand panel contains a sortable table of References.
 - When an entry in that table is selected, the corresponding alignment results are displayed in the main window. The table columns include the accno, number of bases, and the number of contigs in each reference. Sorting by number of contigs (contiguous portions of the reference covered by at least a minimum number of reads) is helpful when the reads are expected to map to only a few of many reference sequences.
- The first row of the multiple alignment in the main window contains the reference sequence for the selected multiple alignment, gapped for insertions in the aligned reads.
 - The successive lines display the individually aligned reads, gapped and showing bases that diverge from the consensus as red dashes on yellow background.
- Scroll bars allow you to navigate the alignment manually:
 - Clicking on the arrowheads on either side of a scroll bar scrolls the alignment by one base (horizontal) or one read (vertical) in the direction of the arrowhead
 - Clicking on the light-colored area on either side of the horizontal scroll bar “handle” scrolls the alignment by 40 bases in the corresponding direction

- Clicking and dragging the “handle” of a scroll bar allows you to move larger distances rapidly
- For easy viewing, when a base position is selected, the base position column is highlighted so that individual variations may be easily identified.
- Right-clicking on a GS read will produce a “Get flowgram for ... at selected position” menu item which, if selected, will activate the Flowgrams tab and display the flowgram for the read; centered on the flow corresponding to the base on which the user clicked to activate the option.



This capability is not active for FASTA reads or the contig/consensus sequences themselves, for which no flowgrams are available.

- Above the Main window, the selected contig information is displayed. Below this is the base view information reporting the selected base and the left and right view positions in the current window.
- To the right of these information fields are tools for viewing and data capture of the main window alignments.
 - **Go To:** A data entry field and a **Go To** button allow you to navigate directly to a position of interest: enter a number not larger than the length of the selected contig in the data entry field, and click the **Go To** button.
 - The **zoom-slider** tool allows for more or fewer bases to be viewed at a time in the main window. If the slider is set to all the way zoom out (-) then the individual bases are grayed out but the shape of the read alignment is still shown along with the positions of the high quality differences highlighted in yellow. (Figure 58).
 - A **'zoom to selected'** button can be used to zoom in, centering on the selected base.
 -  **Camera icon:** saves an image of the part of the multi-alignment table currently visible on screen, to a file in .png format.
- The Alignment Results tab also features a “Mouse Tracker” area, in the left panel, below the list of reference sequences. If you pause the mouse pointer over a base in the multi-alignment, the following information is provided about that base:
 - **Base:** the position of that base relative to the selected reference sequence; gaps in the reference sequence are displayed as the following base of the reference
 - **Read:** the name of the read or contig to which this base belongs
 - **Val:** the “value” of the base under the mouse pointer, in that read, *i.e.* A, G, T, or C.

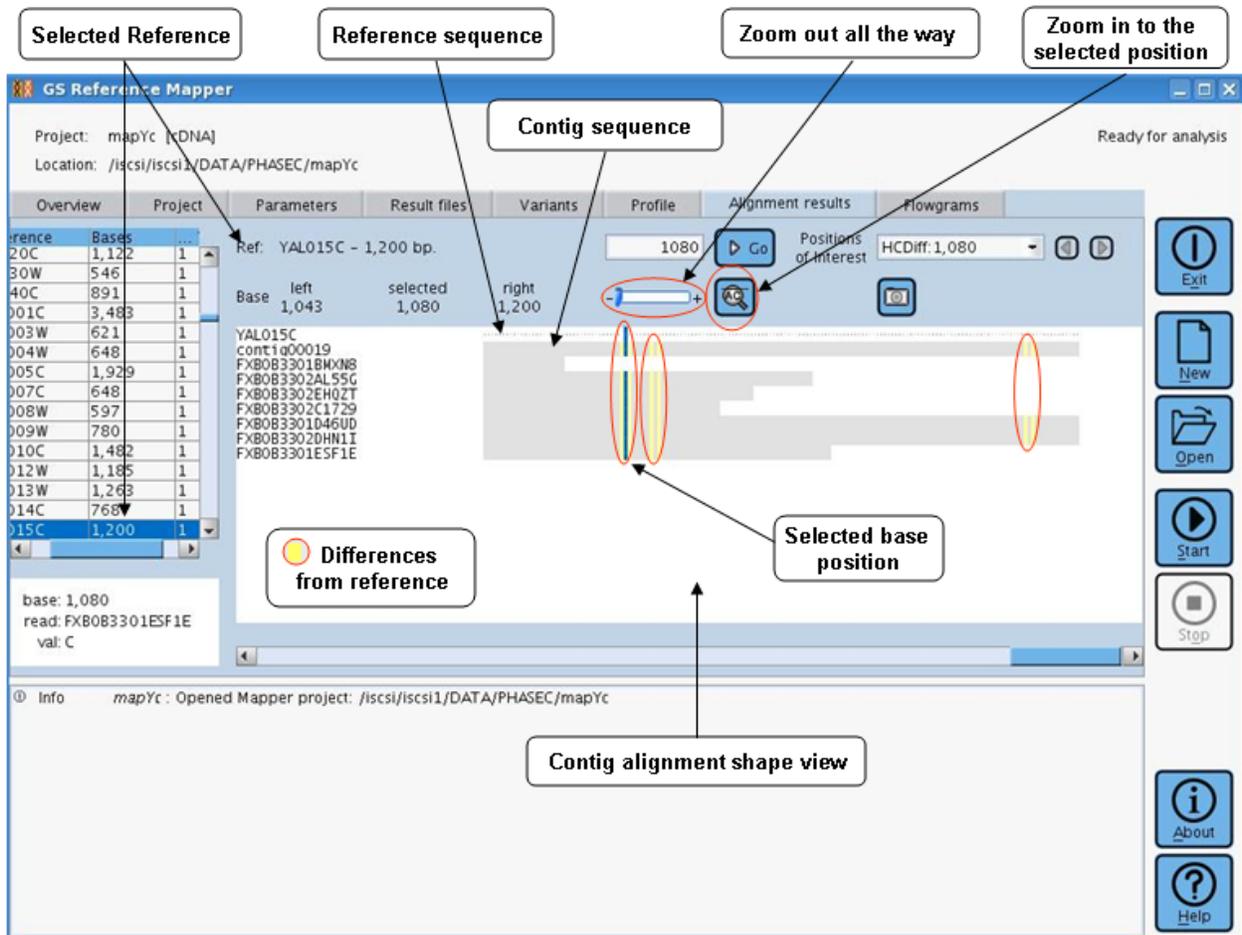


Figure 58: gsMapper Zoom Slider - all the way out.

- Finally, a drop down menu of **Positions of Interest** exists near the upper right corner of the gsMapper Alignment results tab (Figure 59). This is populated with contig positions for a chosen reference or the position(s) from the HCDiff and Structural Variants tables. The Positions of interest list allows the user to “goto” base positions without having to type in the value and using the goto button. There are next and previous buttons to move through the list.

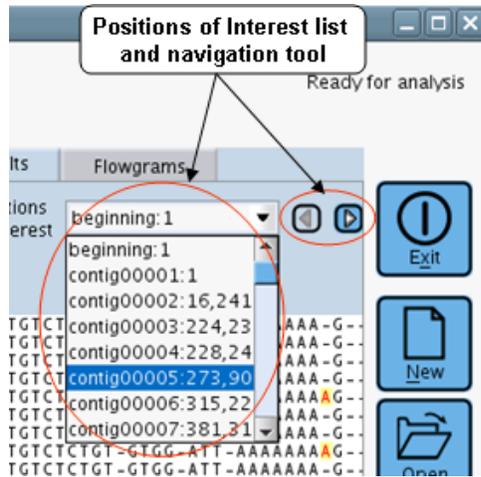


Figure 59: gsMapper Alignment Results Tab

2.13 The Flowgrams Tab

When viewing a multiple alignment in the Alignment Results tab, any read in the multiple alignment can be selected in order to display its flowgram. Select the desired read, and right-click on any base in the read of interest to open a contextual menu that contains a single item ("Get flowgram:.."; Figure 57). Selecting that item will display the flowgram for this read in the Flowgrams tab (Figure 60). For additional information about using the **Flowgrams Tab**, see section 4.11.

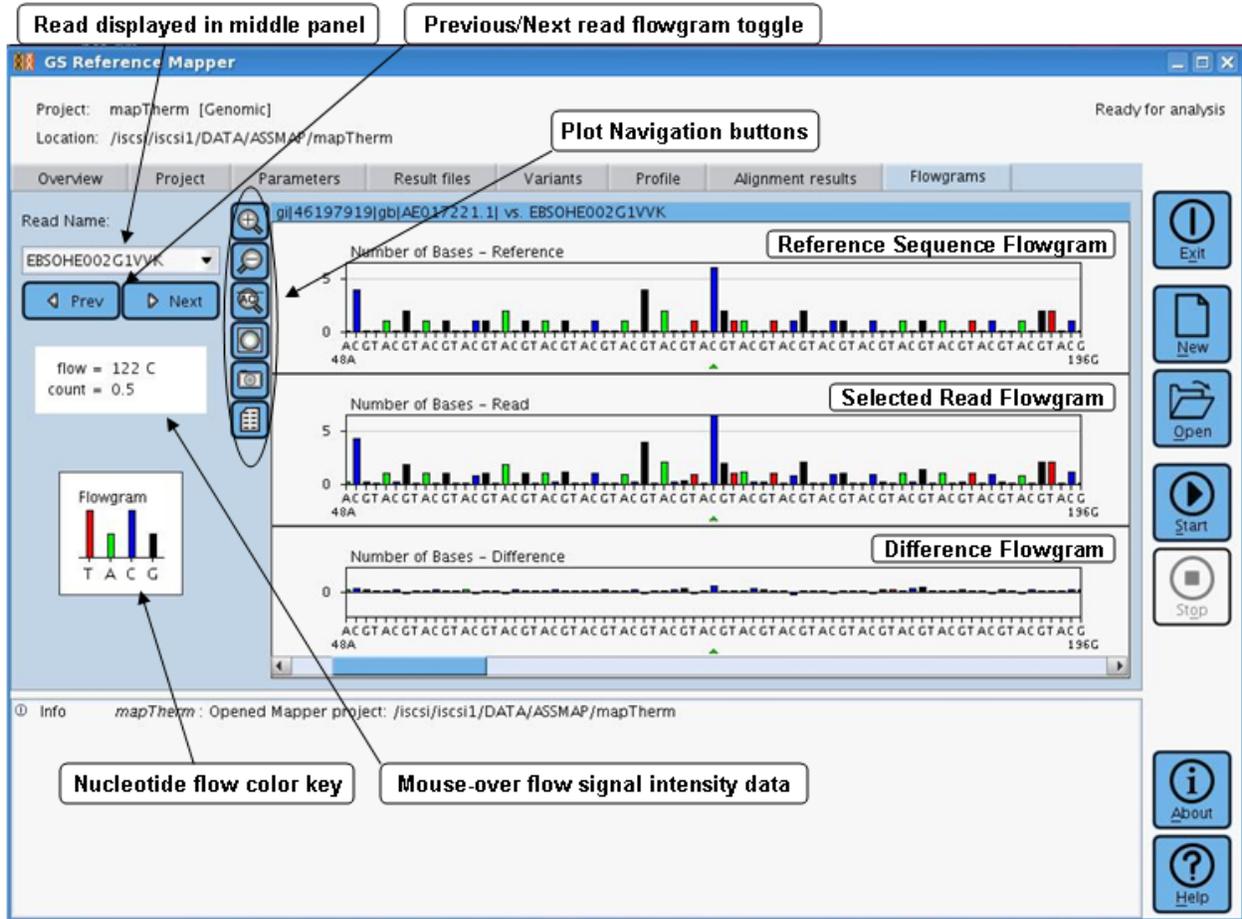


Figure 60: gsMapper Flowgrams Tab

2.14 Project Error Indicators

To assist the user in the proper setup of mapping, expansion of the error messaging is provided on the Parameters and Project tabs. Some examples are shown in the figures in section 4.12.



When a project is first created and no read and reference files have yet been added, a warning icon on the Project Tab header will be displayed along with the status message 'Not ready for analysis'.

2.15 GS Reference Mapper Command Line Interface

The GS Reference Mapper CLI command for single shot mapping of reads can be launched using the following command:

```
runMapping [options] refdesc [filedesc]
```

For incremental mapping via the CLI, a project directory structure is created to organize the files of the incremental build of the project data. The following commands are used for incremental mapping projects:

```
newMapping projdir

addRun [options: -p, -lib, -mcf] [projdir] filedesc

removeRun [projdir] filedesc

setRef [options: -gref, -cref, -random] [projdir] refdesc

runProject [options] [projdir]
```

For all of these commands;

- The arguments in brackets [] are optional.
- 'options' refers to the specific command options available.
- 'filedesc' is one of the following:
 - an sfffilename
 - a [regionlist:]datadir
 - a readfastafilename
 - any of the above prepended by an [MIDList@] specification where each "MIDList" is a multiplexing information string used to filter the set of file reads to be used in the mapping (If MIDs were used in the generation of the read data file, then an MIDList string must be specified in order for the mapper to properly handle the file's reads.) For details about using MIDs, see section 4.6.
- 'refdesc' is a path/name to a reference sequence file in FASTA format.
- 'projdir' is the path of the data analysis project directory.



Some of the command line options for Mapping are mutually exclusive, for obvious functional reasons. See section 4.5 for a list of these options.

2.15.1 Working with Project Folders and Data Files

Since the mapping computation is often performed on a pool of sequencing Runs (or Read Data files) rather than on any single Run, the result files it generates are not deposited in a Run folder. Two general cases exist.

- If the mapping is performed using the "one-step" command runMapping, a folder with a 'P_' prefix (for 'P'ost-Run Analysis) is created in the user's current working directory on the DataRig at the time the application is launched, or written to a directory specified by the user on the command line (or its GUI equivalent), to contain these files. The name structure for this folder is as follows:

```
P_yyyy_mm_dd_hh_min_sec_runMapping
```

- For "incremental", or "project-based" mapping, using the GUI application or using the newMapping and related commands, the output is placed in a "project" folder. A user can specify any name for a project folder; it will be recognized as a project folder by

virtue of the “454Project.xml” file that will be automatically created within it. If a directory name is not specified using the newMapping command line, the software will use the same default name as the runMapping command (as above).

The incremental mapping folder contains additional folders and files that mark the folder as a project folder and that store configuration information and internal data for the GS Reference Mapper application. A project folder is comprised of two sub-folders: a ‘mapping’ sub-folder which contains the project state and output files; and an ‘sff’ sub-folder containing the copies and/or symbolic links for the SFF files used as input to the project. The 454Project.xml file identifies the folder as a 454 project folder.

- The “-o” option can be specified on the command line to change the directory where the output files should be written. If the specified directory exists, the output files will be written to that directory. If the specified directory does not exist, the program will create it, if possible.



When using the `-o` option with the runMapping or newMapping command, the mapper will overwrite the contents of the specified project directory if any exist. This will occur without warning. It is good practice to make a copy of any mapping project you wish to protect before reusing the same project directory.



External Files

- GS Read Data files (SFF files) are not actually copied to the project directory. Rather, a symbolic link to the file is created and placed in the project’s sff directory. Consequently, if the original file is moved or erased from the file system, the project will not operate correctly. This applies whether the files are added via the GUI or the command line.
- FASTA files are not actually copied to the project directory. Rather, the file path is simply recorded in the project setup. Consequently, if the original file is moved or erased from the file system, the project will not operate correctly. This applies to all instances of FASTA files that can be added to a Mapping project, whether done via the GUI or the command line:
 - Reference files
 - FASTA reads files, including Sanger reads
 - Trimming database files
 - Screening database files

Unlike with the GS Read Data files, no symbolic link to the file is created for FASTA files.

2.15.2 One-Step Mapping: the runMapping Command

If all the reads to be mapped to the reference sequence are available at once, the GS Reference Mapper application can process them with a single command, which has the following command line structure:

```
runMapping [options] refdesc [MIDList@]filedesc...
```

...where:

- “[options]” are zero or more of the command line options (listed below),

- the “refdesc” is one of the following:
 - reffasta
 - a FASTA file containing the reference sequence(s),
 - refdirname
 - a name or path for a directory containing the reference sequence (all FASTA files in the directory will be used as the reference),
 - refgenome
 - a GoldenPath genome name (see the fourth Note, below),
 - -ref (reffasta | refdirname)... -read
 - the string “-ref”, followed by multiple FASTA file names or directories, followed by the string “-read” (to mark the beginning of the read data files).



These options are order-dependent: -ref must be used prior to -reads.

- each “filedesc” is one of the following:
 - sfffilename or
 - [regionlist:]datadir or
 - readfastafilename
- and each “MIDList” is a multiplexing information string used to filter the set of file reads to be used in the mapping (see section 4.6 for the format of the MIDList information). If MIDs were used in the generation of the data file, then an MIDList string must be specified in order for the GS Reference Mapper to properly handle the file’s reads.



The runMapping command is actually a wrapper program around the newMapping and related commands used in incremental mapping (see section 2.15.3). After the incremental mapping command is completed, runMapping then transforms the project files and folders into this one-step form (which more closely matches the output structure of older versions of the “Mapping” application). The actions taken are:

- All the mapping output files are moved up from the mapping sub-directory into the main folder
- All internal data files in the mapping sub-directory are deleted
- The 454MappingProject.xml and 454Project.xml files are deleted
- The mapping sub-directory is deleted

If the **-nrm** option is given, runMapping does not perform this transformation, and the resulting folder can be used for further project-based mapping (the structure of the files/folders will match that of an incremental mapping project).



The path given for any of the data directories may be prepended with an optional list of regions, separated from the path by a colon. For example:

```
1-3,4,6-8:D_yyyy_mm_dd_hh_min_sec_testuser_SignalProcessing
```

The list of regions must have the following format:

- It must be a comma-separated list, ending with a colon.
- Each element of the list can either be a single region number or a dash-separated range of region numbers.
- Duplicate regions may be specified, and the regions may be specified in any order, but duplicates will be removed and the regions will be processed and reported in numerical order.
- No spaces or other characters are allowed in the string.

If the region list is not present for a data directory path, all regions of the Run for that directory will be used in the mapping. Also, any combination of explicit SFF files and data directory paths (with optional region lists) may be specified on the command line. For each data directory path given, the runMapping command will read the existing SFF files in the “sff” sub-directory of the data directory. If SFF files are not present in a data directory (e.g. for a Run whose data has been processed with a version of the Genome Sequencer System software anterior to 1.0.52), the signal processing step of the GS Run Processor application must be rerun on the Data Processing folder.

The path for any filename or data directory name can also be prepended with a multiplexing information string. Section 4.6 gives further information on using MIDs.

The off-instrument installation contains a default MID configuration file, found by default at `Installation_path/454/config/MIDConfig.parse`. This file is read by the GS Reference Mapper and used to match MID set names and MID names with their multiplexing information. Users can edit this file to add their own MID sets (following the format and syntax described in the file), or can copy this file to create their own separate MID configuration file (and then use the “-mcf” option to specify that as the MID configuration file to be used).



Contrary to the situation with incremental mapping (see the `setRef` command, section 2.15.3.2), you can use only one reference FASTA file, directory or GoldenPath genome name with the `runMapping` command, unless the `-ref/-read` options are including to separate the reference files from the read data files. All the files can still contain multiple reference sequences.

The GS Reference Mapper is aware of the folder/file structure of the UCSC GoldenPath genome databases (most specifically the human genome and the other major eukaryotic genome databases). If one or more of those databases are downloaded onto the DataRig, and the user sets a `GOLDENPATH` environment variable to the root directory containing those databases, then the genome name can be given as the reference for `runMapping` (i.e., the command “`runMapping hg18 reads.sff`” will map `reads.sff` against the GoldenPath hg18 human genome). The GS Reference Mapper will automatically read the chromosome FASTA files, the gene annotation file and the known SNP information file.

When given a GoldenPath genome (in genome mapping mode only, not for cDNA mapping), only the main chromosome files will be used as the reference by default. The GS Reference Mapper will not read the additional “`_random`” or “`_hap`” files contained in the chromosome directory. In order to include the “`_random`” and “`_hap`” files, the “`-random`” option must be given to the “`setRef`” or “`runMapping`” command.

Specifically, the Mapping application looks for the reference FASTA files in the “`chromosomes`” sub-directory, and files named “`refGene.txt`” and “`snp###.txt`” (i.e., a file like “`snp128.txt`” that begins with “`snp`”, has three digits, then ends with “`.txt`”) in the “`database`” sub-directory. This matches the organization of the human genome database. Other GoldenPath databases (which don’t match this organization) can be used by the GS Reference Mapper application if the files are modified to match this organization. See the UCSC GoldenPath “`Downloads`” page for information on how their genome databases can be downloaded.

In addition to automatically reading the gene/coding-region annotation and known SNP files when the reference is a GoldenPath genome, the GS Reference Mapper application automatically searches for specific annotation/SNP files for any reference. If the reference files are located in a directory named “`chromosomes`”, the GS Reference Mapper will look for `refGene.txt` and `snp###.txt` files in a neighboring “`database`” directory (inside the same parent directory as “`chromosomes`”). The GS Reference Mapper also searches the same directory as the reference FASTA files for a `refGene.txt` and “`snp###.txt`”. Finally, if the reference consists of a single FASTA file, then the software will look for files with “`.refGene`” and “`.snp`” suffixes (e.g., if the reference file is “`mygenome.fna`”, then files “`mygenome.refGene`” and “`mygenome.snp`” will be tried.

To disable this automated reading of the annotation/SNP files, the `-noannot` and `-nosnp` options must be included as part of the command line options.

2.15.3 Incremental Mapping: the `newMapping` and Related Commands

This section describes the main commands of the GS Reference Mapper application, to be used when one or more Runs are to be mapped against a reference sequence or database, as part of

a mapping project. These commands allow you to add Runs over time and incrementally align them to the reference; and to change the reference sequence and restart mapping without losing the dataset of Runs. With these commands, the execution of the mapping and generation of the results can be controlled by command line options and configuration parameters (paralleling the equivalent controls in the GUI application). Such incremental mapping can be useful when, for example, you want to see intermediate mapping results on existing sequencing Runs to determine if you need to carry out further Runs to reach a desired depth of coverage, or just to monitor the project. Incremental mapping is also useful if you simply wish to create output using different output parameter settings.

Five commands are used in a mapping project: `newMapping`, `setRef`, `addRun`, `removeRun` and `runProject`. These are described in the subsections below.

2.15.3.1 *The newMapping Command*

The `newMapping` command is used to initiate a mapping project and set up a mapping project folder to contain the project data (see Section 2.15.1). Its command line structure is:

```
newMapping [option] [projdir]
```

...where:

- [option] `-cdna` may be used; see section 4.1
- [projdir] is the optional name of the project directory

2.15.3.2 *The setRef Command*

The `setRef` command is used to assign the reference sequence(s) to which the reads from the sequencing Run(s) included in the project will be mapped. Its command line structure is:

```
setRef [projdir] refdesc...
```

...where each “refdesc” is one of the following:

- `reffasta`
 - a FASTA file containing the reference sequence(s),
- `refdirname`
 - a name or path for a directory containing the reference sequence (all FASTA files in the directory will be used in the reference),
- `refgenome`
 - a GoldenPath genome name (see the third Note in the `runMapping` Section 2.15.2, above).

2.15.3.3 *The addRun Command*

The `addRun` command is used to add Read Data sets to existing mapping projects. Its command line structure is:

```
addRun [options] [MIDList@]filedesc...
```

...where:

- “[options]” are zero or more of the command line options described in section 4.3
- each “filedesc” is one of the following:
 - o sfffilename or
 - o [regionlist:]datadir or
 - o readfastafilename



Input read size constraints: The reads input for the mapping computation must be shorter than 2000 bases per read (longer sequences are ignored) and longer than 50 bases (shorter sequences are ignored). When Paired End 454 Reads (SFF reads) are part of the project, reads between the value of the minlen parameter and 50 bp long will be mapped onto contigs formed by mapping longer reads to the reference during a later stage in the mapping process.



- The addRun command can be executed multiple times for a mapping project (in any combination with the removeRun and runProject commands). When addRun is executed, it adds (not “resets”) the given Runs/regions or SFF files to the list of sequencing data used in the project. (It does not reset the list of sequence data). The reads used in the mapping (runProject, see section 2.15.3.5 below) are the union of the data from all executions of addRun for the project.
- The commands addRun, removeRun, and runProject can be executed in any combination and in any order, in a mapping project.

2.15.3.4 *The removeRun Command*

The removeRun command is used to remove Read Data sets from existing mapping projects. Its command line structure is:

```
removeRun [projdir] (sffname or readfastafilename)...
```



- This command is more conveniently carried out from the GUI application. When called from the command line, it requires the SFF file name(s) *given in the project sff sub-directory* or the FASTA file name(s) *given in the project configuration file 454MappingProject.xml*. These names may not match the original names of the corresponding files or may not be known to the user, especially if the software had to rename any of them to ensure name uniqueness. The filenames of the SFF files in the sff sub-directory are assigned by the GS Reference Mapper application to ensure uniqueness for all the files, while trying to preserve the original names when possible.
- The execution of this command does not physically remove the file(s) from the project sff sub-directory or from any existing mapping. The file(s) and the reads they contain are only marked for removal by the removeRun command, and are actually removed only the next time the project is computed (via the runProject command).
- The commands addRun, removeRun, and runProject can be executed in any combination and in any order, in a mapping project.

2.15.3.5 The runProject Command

The runProject command performs the actual mapping computation for a project and generates the results of the mapping. Its command line structure is:

```
runProject [options] [projdir]
```

2.16 GS Reference Mapper cDNA / Transcriptome Options

These descriptions are for options specific to cDNA / transcriptome mapping projects.

cDNA / transcriptome mapping option "**-cdna**": To specify a cDNA / transcriptome mapping project, use the **-cdna** option with the runMapping or newMapping commands.

cDNA reference option "**-cref**": To specify a cDNA (transcript) reference sequence for the mapping project, use the **-cref** option on the command line for the setRef command or with the runMapping, newMapping, or runProject commands.

cDNA reference option "**-gref**": To specify a gDNA (genomic DNA) reference sequence for the mapping project, use the **-gref** option on the command line for the setRef command or with the runMapping, newMapping, or runProject commands.

Renaming file option "**-accno**": To specify a tab-delimited file containing annotation data, use the **-accno** option (see more details on the use of this option in the next section).

refGene file option "**-annot**": To specify a tab-delimited file containing gene, transcript, and exon information, use the **-annot** option (see more details on the use of this option are in the next section).

2.16.1 Annotation Input Files for Mapping Transcriptomes

Annotation files specified using the '-annot' option in the CLI for mapping to cDNA / transcriptome sequence data can be of several different types. They are described in Table 2, below.

Source	Reference	Description
Golden Path annotation data	Mapping to a cDNA ref	If a project is using the GoldenPath reference file named refMrna.fa or a compatible reference file (i.e., one using the NCBI style accession numbers), Golden Path annotation data can be incorporated into the project by setting up the required directory structure (the same as used for working with SNP data) that contains the required Golden Path database files and setting the GOLDENPATH environment variable to the path culminating in the "bigZips" directory of the Golden Path directory structure. The only required Golden Path annotation file is refLink.txt. The Golden Path annotation file productName.txt, if present, can be used to incorporate gene-level descriptions into the output files.

Golden Path annotation data	Mapping to a gDNA Ref	Annotation information for genes, transcripts, and exons can be obtained from the Golden Path annotation file named refGene.txt. This gives transcript start and stop positions, coding sequence stop and start positions and exon boundaries for each gene.
Reference sequence file	all	In cases where Golden Path annotation data is unavailable, annotations may be obtained from the description lines for each reference sequence in reference file. The tag/value pair "gene= <i>geneName</i> " must be present on the description line. If "gene= <i>geneName</i> " is not present, transcript and/or gene descriptions can only be incorporated into the output by using a renaming file (see below).
renaming file	all	A renaming file can be used to either provide more meaningful names for genes and/or transcripts and/or to incorporate transcript and gene descriptions into the output files. The structure of a renaming file is as follows: OLD_ID<TAB>NEW_ID<TAB>DESCRIPTION, where OLD_ID is either the transcript identifier or gene name that currently exists in the project or incoming annotation data, and NEW_ID is the new identifier you want to map to the old identifier. DESCRIPTION is the transcript- or gene-level description you want to map to the NEW_ID. Note that it is <i>not</i> necessary to rename a gene or transcript if all you want to do is incorporate gene- or transcript-level descriptions into the output. In this scenario, the OLD_ID and NEW_ID would be identical and would correspond to the appropriate description. Note also that it is possible to specify <i>both</i> transcript- and gene-level descriptions in the same renaming file. The way to do this is to put the transcript-level descriptions on the same lines as the corresponding transcript identifiers and the gene-level descriptions on the same lines as the corresponding gene identifiers.

Table 2: Annotation input files for mapping cDNA / transcriptome data sets

2.17 GS Reference Mapper Output

2.17.1 Output File Descriptions

Output Files produced by the GS Reference Mapper are described briefly below. GUI settings and command line options that control the content type of a file are given when applicable.

File name	Description	Mapping – Genomic Project	Mapping- cDNA Project	GUI Conditional output option	CLI Conditional output option
454AlignmentInfo.tsv	Tab-delimited file giving position-by-position consensus base and flow signal information. Output conditionally (using <code>-info/-noinfo</code> options or checkbox selection on GUI Parameters Tab Output Sub tab) if there are more than 4M reads or the total length of reference sequences exceeds 40Mbp.	✓	✓	Alignment info <u>selection</u> : Output <code>-infoall</code> Output small <code>-info</code> No output <code>-noinfo</code>	
454AllContigs.fna	FASTA file of all the consensus basecalled contigs longer than 100 bases. The minimum length output can be changed by using the <code>[-a #]</code> option in the CLI or changing the All contig threshold in the GUI Parameters Tab, Output Sub tab.	✓	✓	Parameters: All contig threshold <code>-a #</code>	
454AllContigs.qual	Corresponding Phred-equivalent quality scores for each base in the consensus contigs in 454AllContigs.fna.	✓	✓		
454AllDiffs.txt	This file contains the list of variations (of at least 2 reads) relative to the reference sequence or to other reads aligned at a specific location. SNPs, insertion-deletion pairs, multi-homopolymer insertion or deletion regions, and single-base overcalls and undercalls are reported.	✓	✓	Single read variant <u>Selection</u> . If used, single read variations are also included	<code>-srv</code> Single reads variations are also included if this option is used
454AllStructVars.txt	A text file containing a section listing the rearrangement points, followed by a section listing the rearrangement regions. (The associated data describing rearrangement points and rearrangement regions output is found in Section 2.17.1.15.)	✓	✓		
454Contigs.ace	ACE format file that can be loaded by third-party viewer programs that understand the ACE format. The output can be a single file for the entire project or a folder containing individual files for each contig in the mapping.	✓		Ace/Consed <u>Selection</u> Ace read mode <u>Selection</u>	<code>-nobig,</code> <code>-ace,</code> <code>-consed,</code> <code>-noace,</code> <code>-ar, -at, -ad,</code> <code>-consed16</code>
454GeneStatus.txt	A file reporting statistics on the number of reads mapped exclusively to each gene (for cDNA mapping projects only)		✓		

454HCDiff.txt	This file contains the list of high confidence variations (of at least 3 non-duplicate reads) relative to the reference sequence or to other reads aligned at a specific location. SNPs, insertion-deletion pairs, multi-homopolymer insertion or deletion regions, and single-base overcalls and undercalls are reported. The GS Reference Mapper application uses a combination of flow signal information, quality score information and difference type information to determine if a difference is High-Confidence.	✓	✓	
454HCStructVars.txt	A text file containing a section listing the high confidence rearrangement points, followed by a section listing the high confidence rearrangement regions. (The associated data describing rearrangement points and rearrangement regions output is found in section 2.17.1.15.) The GS Reference Mapper application uses a combination of flow signal information, quality score information and variant type information to determine if a variant is High-Confidence.	✓	✓	
454LargeContigs.fna	FASTA file of all the “large” consensus basecalled contigs contained in 454AllContigs.fna (>500bp). This can be changed by using the [-l #] option in the CLI or changing the Large contig threshold in the GUI Parameters Tab, Output Sub tab.	✓		Parameter: Large contig threshold -l #
454LargeContigs.qual	Corresponding Phred-equivalent quality scores for each base in the “large” consensus contigs in 454LargeContigs.fna.	✓		
454MappingQC.xls	This file reports the cumulative errors by base position for mapping projects. (The columns reported differ for genomic and cDNA projects and are described in sections 2.17.1.11 and 2.17.2.2).	✓	✓	
454NewblerMetrics.txt	File providing various mapping metrics, including the number of input Runs and reads, the number and size of the large consensus contigs and the number of all consensus contigs. Reference metrics are provided as well.	✓	✓	
454NewblerProgress.txt	A text log of the messages sent to standard output during the mapping computation.	✓	✓	

454PairAlign.txt	A text file giving the pairwise alignment(s) of the overlaps used in the mapping computation (only produced when using the -p or -pair option [or -pt or -pairt option for the tab-delimited version of the file]).	✓	✓	Pairwise alignment <u>selection</u>	-p or -pair, -pt or -pairt,
454ReadStatus.txt	Tab-delimited text file providing a per-read report of the status of each read in the mapping. The mapped location of all uniquely mapped reads is also reported. If the -reg option is used to specify regions of a reference, such as for a NimbleGen Sequence Capture experiment, then the per-read 'in-region' and 'out-of-region' status is also given.	✓	✓		
454RefStatus.txt	A file reporting the statistical information on the number or reads mapping to each reference sequence.	✓	✓		
454TagPairAlign.txt	A text file giving the pairwise alignments used in the mapping computation for Paired End reads shorter than 50 bases (which are not part of the overlap computation, but are mapped to the consensi in a later computation step).	✓	✓	Pairwise alignment <u>selection</u>	-p or -pair, -pt or -pairt
454TrimStatus.txt	Tab-delimited text file providing a per-read report of the original and revised trimpoints used in the mapping.	✓	✓		

Table 3: GS Reference Mapper Output Files



NewblerProgress.txt: If Runs are added incrementally and multiple executions of runProject occur, the output messages are appended to this file. If the “Incremental Reference Mapper Analysis” checkbox option is not selected in the GUI application, or the “-r” option is given on the runProject command line, the GS Reference Mapper deletes intermediate data and “restarts” the mapping computation, and the NewblerProgress.txt file is deleted and restarted as well.

2.17.1.1 454AlignmentInfo.tsv

The 454AlignmentInfo.tsv file (Figure 61) contains position-by-position summary information about the consensus sequence for the contigs generated by the GS Reference Mapper application, listed one nucleotide per line (in a tab-delimited format). Lines are generated for any position of the reference for which any of the reported Depths is greater than 0. The 454AlignmentInfo.tsv file is output conditionally (depending on the **-info/-infoall/-noinfo** options or the selection made on the GUI Parameters Tab Output Sub tab). By default, this file is only output if there are fewer than 4 million input reads and the total length of reference sequences is less than 40Mbp. For larger projects, **-info** or **-infoall** or the corresponding GUI **Output** selection for **Alignment Info** must be used to generate this file. From the command line, the **-infoall** option is used to tell the mapper to report all lines of the 454AlignmentInfo.tsv file, even if there is no coverage. (This option guarantees that every reference position is reported in the 454AlignmentInfo.tsv file, with the exception that if regions are specified in the mapper, only the regions are reported (but every position in the regions will be reported).

The columns of each line contain the following information:

1. **Position** – the position in the reference
2. **Consensus** – the consensus nucleotide for that position in the reference
3. **Quality Score** – the quality score of the consensus base
4. **Unique Depth** – the number of non-duplicate, uniquely mapping reads that align at that location
5. **Align Depth** – the number of uniquely mapping reads aligned at that location
6. **Total Depth** – an estimated unique plus repeat mapping depth at that location, where the repeat depth is estimated. The estimate is made by randomly assigning each repeat read to one of its assigned locations and incrementing the existing count for that location.
7. **Signal** – the average signal of the read flowgrams, for the flows that correspond to that position in the alignment
8. **StdDeviation** – the standard deviation of the read flowgram signals at the corresponding flows

Prior to lines for each contig, a header line beginning with a '>' displays the Reference sequence accno, as shown in Figure 61.

Position	Reference	Consensus	Quality Score	Unique Depth	Align Depth	Total Depth	Signal	StdDeviation
>gi 85666109 ref NC_001133.6		1						
1	C	C	16	1	1	2	2.24	2.24
2	C	C	14	1	1	2	2.24	2.24
3	A	A	19	1	1	2	1.17	1.17
4	C	C	19	1	1	2	0.97	0.97
5	A	A	23	1	1	2	1.16	1.16
6	C	C	23	1	1	2	2.19	2.19
7	C	C	23	1	1	2	2.19	2.19
8	A	A	30	1	1	2	0.98	0.98
9	C	C	25	1	1	2	0.85	0.85
10	A	A	19	1	1	2	1.14	1.14
11	C	C	14	1	1	2	3.27	3.27
12	C	C	16	1	1	2	3.27	3.27
13	C	C	16	1	1	2	3.27	3.27
14	A	A	26	1	1	2	1.20	1.20
15	C	C	22	1	1	2	1.04	1.04
16	A	A	33	1	1	2	1.16	1.16
17	C	C	33	1	1	2	0.99	0.99
18	A	A	31	1	1	2	1.01	1.01
19	C	C	16	1	1	2	3.28	3.28
20	C	C	16	1	1	2	3.28	3.28

Figure 61: 454AlignmentInfo.tsv file portion example

2.17.1.2 *fna* and *qual* files: 454AllContigs, 454LargeContigs

These files contain the nucleotide sequences of all the contigs (Figure 62) and associated nucleotide Quality Scores (Phred-equivalent; Figure 63) produced by the GS Reference Mapper application. The AllContig and LargeContig output lengths are specified in the GUI or by CLI options described in Table 3, above.

```
>contig00008 gi|85666109|ref|NC_001133.6|, 156677..190890 length=34215 numreads=2646
ggtGgAAAGTACATAGGCCGACaTTTgATAAGGTGTATACGGAAATCaTAGATGGGTgTcG
TAAATGACCAaCcAGATGGATTGGCTtGGTTTTGGGTcATCATGCACTGCTgTGGGTAC
GGCCATTtCtGTgTGAATGTGACTGAGCAGTTTGAGGAGAGGCATGATGGGGTTCTCT
GGAACAGCTGATGAAGCAGGTGTTGTTGTCTGTTGAGAGTTAGCCTTAGTGAAGCCTTC
TCACATTCTTCTGTTTTGGAAGCTGAAACGCTCTAACGGATCTTGATTTGTGTGGACTTCC
TTAGAAGTAACCGAAGCACAGGCGCTACCATGAGAAATGGGTGAATGTTGAGATAATTGT
```

Figure 62: 454AllContigs.fna file portion example

```
>contig00008 gi|85666109|ref|NC_001133.6|, 156677
..190890 length=34215 numreads=2646
15 15 21 64 30 64 64 64 53 64 64 64 64 64 64 64
64 64 48 64 37 64 64 59 18 64 64 64 64 64 51 64 6
4 64 64 64 64 64 64 64 64 64 55 17 64 64 64 64
64 64 64 56 64 35 64 64 31 64
64 64 64 64 64 64 64 64 64 64 7 64 20 64 64 64
64 64 64 64 64 41 64 64 64 64 11 64 64 64 64 64
```

Figure 63: 454AllContigs.qual file portion example

2.17.1.3 454ReadStatus.txt

The 454ReadStatus.txt file (Figure 64) contains the status identifiers for all the reads used in the mapping computation, plus the position for each mapped read's alignment within the reference (unless the mapping status is "chimeric"). The reads are listed one per line, in tab-delimited format. Furthermore, if the "-reg" option is given (to specify a set of regions of the reference, such as in a NimbleGen sequence capture experiment), then the per-read "InRegion" or "OutOfRegion" status is given, to describe which reads aligned in the regions and which ones aligned elsewhere in the genome. Each line contains the following information:

The following columns are reported:

1. **Read Accno** – Accession number of the input read.
2. **Mapping Status** – Status of the read in the mapping, which can be one of the following:
 - a. **Full** – the read is fully aligned to the reference (every base)
 - b. **Partial** – only part of the read aligned to the reference
 - c. **Chimeric** – part of the read aligned to one location on the reference and a different part of the read aligned to a different reference or to a distant location on the same reference
 - d. **Repeat** – the read aligned equally well to multiple locations in the reference
 - e. **Unmapped** – the read did not align to the reference
 - f. **TooShort** – the trimmed read was too short to be used in the computation (shorter than 50 bases and longer than minlen bases, unless 454 Paired End Reads are included in the dataset, in which case, all reads at least "minlen" bases are used and 454NewblerMetrics.txt will report the value of numberTooShort as 0 since any shotgun reads at least as long as the minimum read length will be used in the mapping).
3. **Mapped Accuracy** – The percentage identity of the alignment, rounded to the nearest whole number (reads with 'Full' and 'Partial' status only)
4. **% of Read Mapped** – The percentage of the read that occurs in the alignment (reads with 'Full' or 'Partial' status only)

5. **Ref Accno** – The accno of the reference sequence to which the read is aligned
6. **Ref Start** – The position in the reference sequence where the read's alignment begins
7. **Ref Stop** – The position in the reference sequence where the read's alignment ends
8. **Strand** – The orientation of the read's alignment relative to the reference sequence. A '+' indicates the alignment orientation of the read is the same as the orientation of the reference. A '-' indicates the alignment orientation of the read is opposite to the orientation of the reference.
9. **Region Status** (with **-reg** option) – Indicates whether or not the read intersects a targeted region as defined by the parameter given with the **-reg** option: 'InRegion' means that the read intersects a target region, 'OutOfRegion' means that the read does not intersect any target regions. This column is not present without the **-reg** option.

Read	Mapping Status	Mapped Accuracy(%)	% of Read Mapped	Ref	Ref Accno	Ref Start	Ref Stop	Strand
FWV1IC001AS2NQ	Partial	98	62	chr20	60145823	60145880		-
FWV1IC001AVVUO	Full	90	100	chr10	8194944	8195005	+	
FWV1IC001ATU7G	Repeat							
FWV1IC001AOB1M	Repeat							
FWV1IC001AN8ZR	Repeat							
FWV1IC001APKBN	Chimeric							
FWV1IC001AQCUCO	Full	100	100	chr6	45636937	45637018		+

Figure 64: 454ReadStatus.txt file portion example for a Mapping Project

2.17.1.4 454TrimStatus.txt

This file contains a per-read report of the original and revised trimpoints used in the mapping project, where the revised trimpoints include the effects of primer, vector and/or quality trimming of the original input reads (Figure 65). Each line contains the following information (these are the columns in the tab-delimited format):

1. **Accno** – accession number of the input read
2. **Trimpoints Used** – the final trimpoints used in the mapping, in #-# format
3. **Trimmed Length** – the final trimmed length of the read
4. **Orig. Trimpoints** – the original trimpoints of the read, found in the SFF or FASTA file
5. **Orig. Trimmed Length** – the original trimmed length of the read
6. **Raw Length** – the length of the raw read (without any trimming)

Accno	Trimpoints Used	Used	Trimmed Length	Orig Trimpoints	Orig Trimmed Length	Raw Length
FWV1IC001AS2NQ	5-94	90	5-94	90	142	
FWV1IC001AVVUO	5-67	63	5-67	63	250	
FWV1IC001ATU7G	5-93	89	5-93	89	230	
FWV1IC001AOB1M	5-147	143	5-147	143	357	
FWV1IC001AN8ZR	5-222	218	5-222	218	275	
FWV1IC001APKBN	5-120	116	5-120	116	327	
FWV1IC001AQCUCO	5-86	82	5-86	82	244	
FWV1IC001ALYPF	13-205	193	5-205	201	406	

Figure 65: 454TrimStatus.txt file portion example

2.17.1.5 *454Contigs.ace or ace/ContigName.ace or consed/...*

This viewer-ready genome file shows all the reference sequences to which reads mapped. The file allows the display of how the individual reads aligned to those reference sequences, in an ACE format file suitable for use in various third-party sequence finishing programs (Figure 66). (The freeware “clview” application can be downloaded from: <http://compbio.dfci.harvard.edu/tgi/software/>; a full description of the .ace file format can be found at: <http://bozeman.mbt.washington.edu/consed/consed.html>.) It should be noted, however, that such third-party viewing software will not be able to make full use of the flow-space mapping information available with Genome Sequencer reads, and that conversely some of the third-party program’s functions (e.g. involving sequence chromatogram input) are not usable with Genome Sequencer datasets. Nonetheless, these programs may be useful to view and assess read characteristics and coverage depth in regions of interest.

The data software analysis applications can also output the mapping results in a complete directory structure suitable for use as input to the consed software (see the above link for a description of this structure and its files). When the appropriate option is selected, a “consed” sub-directory is produced in the output (or project) directory for the computation. This directory contains the sub-directories, the ACE file and the PHD file, so that the functions of consed (viewing traces, editing reads, autofinishing) can be performed on the mapping. In order to integrate the 454 Sequencing reads (and their SFF files) into the consed data, an extra “sff_dir” directory is created in the consed sub-directory, and a number of consed options are automatically specified in this directory (see the “consed/edit_dir/.consedrc” file for the options specified by the generated structure). One option tells consed to use the “sff2scf” command to access any trace information requested by a user. See Section 3.3 for a description of the sff2scf command, and how it can be used to generate synthetic traces from SFF data, as well as provide a pass-through access to SCF data for Sanger reads.

A

```
AS      15 690514

C0 gi|85666109|ref|NC_001133.6| 226558 14877 1 U
CCACACCACACCCACACCCACACACCACACACCACACACCACACACC
CACACACACACATCCTAACACTACCCTAACACAGCCCTAATCTAACCC*T
GGCCAACCTGTCTCTCAAC*TTACCCCTCCATTACCC*TGCCTCCACTCG*
TTACCCCTGTCCCAITCAACCATAACCACTCCGAACCACCATCCATCCCTCT
ACTTACTACCACTCACCCACCGTTACCCCTCCAATTA*CCC*ATA*TCCAA
CCCCT*GCCAC*TTACCCCTACCATTACCC*TACCATCCACCATGACCTA
CTCACCATACTGTTCTTCTACCCACCATATTGAAACGCTAACAAATGATC
GTAAATAACACACACCGT*GCTT*ACCCTACCACCTTT*ATACC*ACCACCA
```

B

```

AF FKHFISHO2RVIHK_right.366-293.to131 C 346
AF FKHFISHO2RCO1X_left.29-1.fm32.pr32 C -199
AF FKHFISHO2QZABD_left.1-3.to131.pr32 U 461
AF FKHFISHO2TF65J_left.246-301.fm2.pr3 U -244
AF FKHFISHO2P54QU_right.14-114.fm46.pr45 U -12
AF FKHFISHO2TBL3K_right.1-115.to131.pr124 U 280
AF FKHFISHO2PK309.211-478.fm32 U -209
AF FKHFISHO2QS2XI_right.96-1.fm63.pr62 C -14
AF FKHFISHO2RVVA3_right.pr2 C 246
AF FKHFISHO2SO6XK_left.348-356.fm66.pr67 U -346
AF FKHFISHO2PN9GI_right.243-279.fm66.pr67 U -241
AF FKHFISHO2SSH8E_left.1-280.to131 U 24
AF FKHFISHO2SVYVD_left.1-192.to131.pr32 U 169
AF FKHFISHO2PVOGQ_left.248-287.fm32.pr33 U -246

BS 1 196 FKHFISHO2PYT8Z.356-61.fm60.to12
BS 197 198 FKHFISHO2QYZGC_right.28-326.fm32.to131.pr33
BS 199 402 FKHFISHO2PYT8Z.356-61.fm60.to12
BS 403 404 FKHFISHO2QYZGC_right.28-326.fm32.to131.pr33
BS 405 462 FKHFISHO2PYT8Z.356-61.fm60.to12
BS 463 463 FKHFISHO2REOJT_right.1-296.to131.pr32
BS 464 464 FKHFISHO2PYT8Z.356-61.fm60.to12
    
```

C

```

RD FKHFISHO2Q1TDD.1-280.to12 622 0 0
AC*TGG*ATTT*AG*TGTA*TG*AT*GG**T*GTTTT**G**A*GG*
*T**GC*T*CC*A*G*T*GGC*TT*C*GTTTT*CT*A*T*C**AGC*T*G
TCCC*TCC**T*GTT*CAG*CTAC*TG***A*C*GGG*T*GG*T*GCG
*TAA*CGG*CAAAA*G*CAC*TG*CCGG*ACAT*CA*G*C*GC*T*A*TC
TC*TG*CTC*TCA*CT*G*CCG*T*AAAA**C*A*T*GGC*AA*CTG*C
*AGTT*C*ACTT*A*CA*CC*G*C*TT*C*T*C***AA*CCC*GG*T*AC
GCA*CC*A***GAAAA*T*C*ATTG*ATATGGCC*AT*G*AATGGCGTT
**GG*A*T*G***CC**GGG**CAACC*G**CCC*GCA*TT*A**T*G*G
G*C*GTT**GG*CC**T*C*AACA**G**A*T*TT*CGCCATTAAA
AAAACTCAGGCGCAGTGGTAACCTCGCGCATAACAGCGGGCAGTGAAGT
CATCGTCTGCGGGAAATGGACGAACAGTGGGGATACGTGGGGCTAAAT
CGCGCCAGCGCTGGCTGTTTTACGCGTATGACAGGCTCGGAAGACGGTT
GTTGCGCACGTATTGGTGAAC

QA 1 438 1 438
DS CHROMAT_FILE: FKHFISHO2Q1TDD.1-280.to12 PHD_FILE: FKHFISHO2Q1TDD.1-280.to12.phd.1 TIME: Thu Jul 27 12:33:48 2000 CHEM: 454
    
```

Figure 66: 454Contigs.ace file portion portions example for a genomic DNA project with Paired End reads. (A) Contig sequence and quality information; (B) Information mapping reads to contigs; (C) Read sequence and quality information



The structure of the ACE file produced by the GS Reference Mapper application differs from the traditional assembly ACE files because it is displaying the sequence information within the context of the given reference sequence(s). Each individual sequence in the reference to which at least one read maps is output as a “contig” in the ACE file (so that the alignments of the 454 reads to each reference sequence can be viewed relative to the reference sequence bases in the “contig alignment viewers” that most third-party ACE file viewers contain). The 454 reads and FASTA reads, as well as the contigs generated by the GS Reference Mapper application, are output in the ACE file as “reads”, again to organize all the data relative to the reference.

To ensure compatibility with ACE file viewers, as well as to assist in the post-analysis of the mapping, the identifiers for reads whose alignments are split and Paired End reads (if present), are output with an additional suffix. Several suffixes may be added to the original read identifier:

- For 454 Paired End reads, the two halves of the 454 read that constitute the sequences at the two ends of the original clone (from which the Paired End read was generated) are marked by “_left” and “_right” suffixes.
- If only part of the trimmed read aligns in this contig (either because the read is only Partially Mapped, or the read is aligned with different sections in different contigs due to a structural variation, for example), the base position range of the region aligned in this contig is added, as in “.1-60” if the read has bases 1-60 aligned in the contig.

2.17.1.6 *NewblerMetrics.txt*

The 454NewblerMetrics.txt file is a 454 parser (see the General Overview section of this manual for a description of the file formats) file that reports the key input, algorithmic and output metrics for the data analysis software applications. Each application that uses the Newbler algorithm creates a 454NewblerMetrics.txt file with relevant output. Below is a description of all the sections present in this file when produced by the GS Reference Mapper, with their named groups and keywords.

- **referenceSequenceData group** – contains information about the reference sequence file(s) (Figure 67)
- **runData group** – contains information about the read data used in the analysis (both Sanger and non-Paired-End 454 Sequencing read files are reported on in this section; not shown on Figure 67 since only Paired End data files were used in this example)
- **pairedReadData group** – contains information about the Paired End input data [Paired End only; 454 Sequencing reads only (not Sanger reads)] (Figure 67)

```
/*  
** Input information.  
*/  
referenceSequenceData  
{  
    file  
    {  
        path = "/remote/rigdata/nas17/watson/data2/rwiner/rwTest2/YeastSV/yeast2WithSVs.fna";  
        numberOfReads = 15;  
        numberOfBases = 12117509;  
    }  
}  
runData  
{  
}  
pairedReadData  
{  
    file  
    {  
        path = "/remote/rigdata/nas17/watson/data2/pairedDataSets/yeast3kb/RandD1108/FKHERYT01.sff";  
        numberOfReads = 471503, 806319;  
        numberOfBases = 182478083, 163918236;  
        numWithPairedRead = 336755;  
    }  
}  
}
```

Figure 67: 454NewblerMetrics file, referenceSequenceData and runData groups portion example

- **runMetrics group** – contains information about the mapping computation (Figure 68)
- **readMappingResults group** – contains information about the mapping process for each input file [SFF, FASTA (including Sanger Paired End), or Run regions from wells file; not shown on Figure 68 since only Paired End data files were used in this example]
- **pairedReadResults group** – contains information about the Paired End input data (Paired End only; Genome Sequencer FLX System) (Figure 68)

```

/**
** Operation metrics.
**/
runMetrics
{
    numberOfReferenceSequences = 15;
    totalReferenceNumberOfBases = 12117509;

    totalNumberOfReads = 806319;
    totalNumberOfBases = 163918236;

    numberSearches = 786416;
    seedHitsFound = 28877290, 36.72;
    overlapsFound = 1842905, 2.34, 6.38%;
    overlapsReported = 725605, 0.92, 39.37%;
    overlapsUsed = 725074, 0.92, 99.93%;
}

readMappingResults
{
}

pairedReadResults
{
    file
    {
        path = "/remote/rigdata/nas17/watson/data2/pairedDataSets/yeast3kb/RandD1108/FKHERYT01.sff";

        numMappedReads = 796069, 98.73%;
        numMappedBases = 162075384, 98.88%;
        inferredReadError = 0.85%, 1219283;

        numberWithBothMapped = 274754;
        numWithOneUnmapped = 3596;
        numWithMultiplyMapped = 57141;
        numWithBothUnmapped = 1264;
    }
}

```

Figure 68: 454NewblerMetrics file, runMetrics and readMappingResults groups portion example

- **consensusDistribution group** – contains information about the consensus signals and basecalling thresholds
- **consensusResultsgroup** – contains summary information about the read status and, if Paired End reads are used in the project, Paired End library statistics. This is followed by contig statistics and variation metrics for large contigs (longer than 'largeContigThreshold'; default is 500 bp) and all contigs (longer than 'allContigsThreshold'; default is 100 bp) (Figure 69).

```
/*  
** Consensus results.  
*/  
consensusResults  
{  
    readStatus  
    {  
        numMappedReads    = 796492, 98.73%;  
        numMappedBases    = 162075384, 98.88%;  
        inferredReadError = 0.85%, 1219283;  
  
        numberFullyMapped    = 685031, 84.96%;  
        numberPartiallyMapped = 4452, 0.55%;  
        numberUnmapped       = 9827, 1.22%;  
        numberRepeat         = 106586, 13.22%;  
        numberChimeric       = 423, 0.05%;  
        numberTooShort       = 0, 0.00%;  
    }  
  
    pairedReadStatus  
    {  
        numberWithBothMapped    = 274754;  
        numberWithOneUnmapped   = 3596;  
        numberMultiplyMapped    = 57141;  
        numberWithBothUnmapped  = 1264;  
  
        library  
        {  
            libraryName    = "FKHERYT01.sff";  
            pairDistanceAvg = 2772.8;  
            pairDistanceDev = 693.2;  
        }  
    }  
  
    largeContigMetrics  
    {  
        numberOfContigs    = 284;  
        numberOfBases      = 11564622;  
  
        avgContigSize      = 40720;  
        N50ContigSize      = 125282;  
        largestContigSize  = 473736;  
  
        Q40PlusBases      = 11420767, 98.76%;  
        Q39MinusBases     = 143855, 1.24%;  
  
        numUndercalls     = 8296;  
        numOvercalls      = 2651;  
        numHCUndercalls   = 1353;  
        numHCOvercalls    = 708;  
        consensusAccuracy  = 99.9053%;  
        HCconsensusAccuracy = 99.9820%;  
    }  
  
    allContigMetrics  
    {  
        numberOfContigs = 352;  
        numberOfBases   = 11585711;  
    }  
}
```

Figure 69: 454NewblerMetrics file, consensusResults group example for a Paired End genomic project



In the ConsensusResults section contains an entry line for chimeric reads. These are reads split into two or more segments, where the segments map to non-consecutive positions of the reference. A typical mapping of genomic DNA reads to a genomic DNA reference will have less than 1% chimeric reads. However, in some cases, such as when mapping cDNA reads to a genomic reference, the percentage of reads demonstrating this trait can be quite large.

2.17.1.7 *454NewblerProgress.txt*

This file represents the text log of the messages sent to standard output by the runProject or runMapping command (showing the progress of the execution of the mapping computation). If Runs are added incrementally and multiple executions of runProject occur, the output messages are appended to this file. If the “Incremental reference mapping analysis” checkbox option is not selected in the GUI application, or the “-r” option is given on the runProject command line, the GS Reference Mapper deletes intermediate data and “restarts” the mapping computation, and this file is deleted and restarted as well.

2.17.1.8 *454PairAlign.txt*

This file contains the pairwise alignments of the overlaps that were found during the mapping computation (Figure 70). By default, this file is not generated, but if the “-p” or “-pt” options are given on the runProject command line, it will be generated either in a human-readable text format (“-p”) or in tab-delimited format (“-pt”).

Each of the displayed alignments contains the following information (these are the columns in the tab-delimited format):

1. **QueryAccno** – accession number of the read used in the overlap detection search (the “query sequence”)
2. **QueryStart** – starting position of the alignment in query sequence
3. **QueryEnd** – ending position of the alignment in query sequence
4. **QueryLength** – length of the query sequence
5. **SubjAccno** – accession number of the other read (the “subject sequence”)
6. **SubjStart** – starting position of the alignment in subject sequence
7. **SubjEnd** – ending position of the alignment in subject sequence
8. **SubjLength** – length of the subject sequence
9. **NumIdent** – number of identities in the pairwise alignment, *i.e.* where query and subject characters match
10. **AlignLength** – the length of the pairwise alignment
11. **QueryAlign** – query alignment sequence
12. **SubjAlign** – subject alignment sequence

```
>FWV1IC001ADC3I, 1..53 of 53 and chr10, 80942605..80942658 of 135374737 (53/54 ident)
  1 CTGGGAAGAAATGAAGATATCTGCCCGTCAACTCATAATATGGATGTCCC-A 53
  80942605 CTGGGAAGAAATGAAGATATCTGCCCGTCAACTCATAATATGGATGTCCCCA 80942658
>FWV1IC001ADC3I, 1..53 of 53 and chr10, 81262507..81262560 of 135374737 (53/54 ident)
  1 CTGGGAAGAAATGAAGATATCTGCCCGTCAACTCATAATATGGATGTCCC-A 53
  81262507 CTGGGAAGAAATGAAGATATCTGCCCGTCAACTCATAATATGGATGTCCCCA 81262560
```

Figure 70: 454PairAlign.txt file portion example

2.17.1.9 454PairStatus.txt

This file contains the per-pair report of the location and status of how each Paired End pair of reads used in the mapping. Each line contains the following information (these are the columns in the tab-delimited format):

1. **Template** – template string for the pair (this will be the original 454 accession for 454 Paired End reads, and the “template” string for Sanger reads)
2. **Status** – the status of the pair in the mapping, with the following possible values:
 - a. **BothUnmapped** – both halves of the pair were unmapped
 - b. **OneUnmapped** – one of the reads in the pair was unmapped
 - c. **MultiplyMapped** – one or both of the reads in the pair were marked as Repeat
 - d. **TruePair** – both halves of the pair were mapped into the same reference sequence, with the correct relative orientation, and are within the expected distance of each other
 - e. **FalsePair** – the halves were mapped to the same reference sequence, but the orientation of their alignment is inconsistent with a Paired End pair or the distance between the halves is outside the expected distance
3. **Distance** – for “TruePair” or “FalsePair” pairs, the distance between the halves
4. **Left Contig** – the contig where the left half was mapped, or “-” if the read was Unmapped or Repeat
5. **Left Pos** – the position in the contig where the 5’ end of the left half was mapped
6. **Left Dir** – the direction (+’ for the forward strand of the reference sequence and ‘-’ for reverse strand) in which the left half was mapped
7. **Right Contig** – the contig where the right half was mapped, or “-” if the read was Unmapped or Repeat
8. **Right Pos** – the position in the contig where the 3’ end of the right half was mapped
9. **Right Dir** – the direction (+’ for the forward strand of the reference sequence and ‘-’ for reverse strand) in which the right half was mapped
10. **Left Distance** – the distance from the Left Pos to the respective end of the reference sequence (for forward matches, this is the distance to the 3’ end of the sequence; for reverse matches, to the 5’ end)
11. **Right Distance** – the distance from the Right Pos to the respective end of the reference sequence (for forward matches, this is the distance to the 3’ end of the sequence; for reverse matches, to the 5’ end).

Template	Status	Distance	Left Accno	Left Pos	Left Dir	Right Accno	Right Pos	Right Dir	Left Distance	Right Distance
FKHERYTO1AQG4Y	TruePair	1803	gi 50593503 ref NC_001148.3	446317	+	gi 50593503 ref NC_001148.3	448120	-		
FKHERYTO1BDW6L	TruePair	1849	gi 85666116 ref NC_001142.6	722421	-	gi 85666116 ref NC_001142.6	720572	+		
FKHERYTO1C28NP	FalsePair	1134	gi 85666116 ref NC_001142.6	124962	+	gi 85666116 ref NC_001142.6	126096	-		
FKHERYTO1AMVUQ	OneUnmapped	-	Unmapped			gi 42742172 ref NC_001138.4	156622	+		
FKHERYTO1BHX74	FalsePair	1317	gi 83578099 ref NC_001139.7	931296	-	gi 83578099 ref NC_001139.7	929979	+		
FKHERYTO1A3M3	TruePair	2290	gi 50593503 ref NC_001148.3	539154	+	gi 50593503 ref NC_001148.3	541444	-		
FKHERYTO1AGERV	TruePair	2230	gi 83578099 ref NC_001139.7	136830	-	gi 83578099 ref NC_001139.7	134600	+		
FKHERYTO1BPQ22	TruePair	2887	gi 50593503 ref NC_001148.3	723645	-	gi 50593503 ref NC_001148.3	720758	+		

Figure 71: 454PairStatus.txt file portion example

2.17.1.10 454TagPairAlign.txt

This file contains the pairwise alignments of “short” reads that were found during the mapping computation. When 454 Paired End reads are included in the dataset, reads that are between 15 and 50 bases long are not included in the main overlap computation, but are mapped to the references in a later step of the computation (using different alignment detection settings). This file reports the alignments of the uniquely mapping 15-50 bp reads against the reference (this version of software does not report the alignments for multiply mapped reads). By default, this file is not generated, but if the “-p” or “-pt” options are given on the runProject command line, it will be generated either in a human-readable text format (“-p”) or in tab-delimited format (“-pt”). The format of this file is identical to the 454PairAlign.txt file described in section 2.17.1.8.

2.17.1.11 454MappingQC.xls

This file contains a number of detailed metrics regarding the mapping results; its format and structure are intended for reading by MS Excel or a similar spreadsheet program, so that the metrics can be easily visualized using Excel’s charting/graphing tools. The file is in tab-delimited format, and contains six main sections (Note: The section titles given here are not displayed in the file.)

1. Summary Statistics (Figure 72)

- a. **Num. Reads** – the number of input reads used in the mapping computation
- b. **Num. Bases** – the number of bases in the input reads
- c. **Mapped Reads** – the number and percentage of reads that uniquely mapped to the reference, followed by the number and percentage of reads that uniquely or multiply mapped
- d. **Mapped Bases** – the number and percentage of bases that uniquely mapped to the reference, followed by the number and percentage of reads that uniquely or multiply mapped
- e. **Inf. Read Error** – the “inferred read error” percentage and quality score (calculated as the number of read alignment differences over the number of mapped bases), along with the counts of the number of read alignment differences and mapped bases
- f. **Exp. Read Error** – the expected read error computed from the input read quality scores, given as a percentage, quality score and expected number of alignment differences. This is computed by summing the expected number of errors for each quality score value (*i.e.* number of bases with a quality score times the accuracy rate of that quality score).
- g. **Last 100 Base IRE** – the “inferred read error” numbers, using only the last (3’) 100 bases of each read
- h. **Last 50 Base IRE** – the “inferred read error” numbers, using only the last (3’) 50 bases of each read
- i. **Last 20 Base IRE** – the “inferred read error” numbers, using only the last (3’) 20 bases of each read
- j. **Genome Size** – the number of bases in the reference
- k. **Num. Large Contigs** – the number of large contigs reported in the 454LargeContigs.fna file
- l. **Num. Large Contig Bases** – the number of bases in the large contigs

- m. **Avg. Depth** – the average alignment depth (*i.e.* how many reads aligned to each position of the reference)
- n. **Avg. Map Length** – the average length of the alignment of a read (the read’s “map length”)

Num. Reads:	983341			
Num. Bases:	375038046			
Mapped Reads:	555132	56.45%	972810	98.93%
Mapped Bases:	200592935		53.49%	219954145 58.65%
Inf. Read Error:	1.81%	17.4	3637358	200592935
Exp. Read Error:	0.58%	22.4	1160576	
Last 100 Base IRE:	2.59%	15.9	1387202	53540362
Last 50 Base IRE:	3.05%	15.2	825766	27048769
Last 20 Base IRE:	3.42%	14.7	364166	10636427
Genom Size:	3080436051			
Num. Large Contigs:	37802			
Num. Large Contigs Bases:			30062887	0.98%
Avg. Depth:	0.1			
Avg. Map Length:	361			

Figure 72: 454MappingQC, summary statistics portion

2. Read Error Histogram (Figure 73)

- a. This section breaks down the number of reads based on their read status and/or number of alignment differences, and displays percentages and counts per category.
- b. Reads that did not map to the reference (“Unmapped”), reads where only part of the sequence mapped to the reference (“Partial”) and reads that mapped to multiple locations in the reference (“Multiple”) are displayed as one category each.
- c. Reads that mapped fully to the reference (meaning every base of the read occurred in the alignment) are then divided by the number of alignment differences found, and percentages and number of reads having 0, 1, 2, ..., 9 or 10+ (10 or more) errors are shown.

3. Overall/Undercall Tables (Figure 73)

- a. These two tables display the percentages and numbers of homopolymer accuracy, using the read alignments to count when the read contains the same or different homopolymer length as compared to the reference.
- b. The rows are the read homopolymer length and the columns are the reference homopolymer length.
- c. When scanning the read alignments, the reference sequence homopolymer is first determined, then the alignment of the read to that homopolymer is evaluated and counted
 - i. Some reads may have multiple homopolymers aligned to a single reference homopolymer (like “ACA” align to “AAA”). These are counted and displayed separately on a “Mult” row.
 - ii. The percentages shown in the first overall/undercall table are given as a percentage of the column (*e.g.* what percent of the time at a reference 5-mers did the read have a 4-mer). Also, the percent table does not show the percentage of the correct alignments (*e.g.* 5-mer to 5-mer), nor does it show percentages less than 0.1% (in order to highlight the overall/undercall trend).

- iii. Below the counts table, a “%Ident” row displays the percentages for each reference n-mer where the read was called correctly (*i.e.* its homopolymer length matched the reference).

Unmapped:	0.35%	3413															
Partial:	4.90%	48166															
Chimeric:	36.16%	355552															
Multiple:	6.32%	62126	0.00%	0													
10+	9.88%	97156															
9	1.09%	10683															
8	1.38%	13584															
7	1.81%	17812															
6	2.44%	24015															
5	3.21%	31608															
4	4.28%	42085															
3	5.46%	53674															
2	6.79%	66752															
1	7.80%	76655															
0	7.42%	72942															
Overcall/Undercall Percents (All Bases)																	
Reference N-Mer Lengths																	
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14		
Mult	15	16	17	18	19	20											
0	35.5%	31.4%	34.4%	45.9%	54.5%	69.8%	4.2%	3.7%	4.6%	6.7%	9.4%	13.3%	15.4%	18.9%	27.2%		
1		95.7%		0.4%													
2		3.8%	0.2%		1.0%												
3		0.4%		0.3%		3.1%	0.2%										

Figure 73: 454MappingQC, Read Error Histogram and Overcall/Undercall Tables portion

4. GC Content Information (Figure 74)

- a. **GC Observed/Expected** – the two lines below this display the GC content percentages (from 0 to 100) and the observed over expected mapping depth. This is calculated by first counting the number of reads with particular GC content and counting the GC content of all windows of the reference (where the window length matches the average read flowspace or nucleotide length). Then the two counts (read and reference) for a specific GC content value are divided by the read/reference totals to compute the percentage of the reads/references with that GC content. The observed/expected value is the ratio of those two percentages.
- b. **GC Std. Dev.** – this is the standard deviation of the GC Observed/Expected (based on the sampling at that GC content value). The values on this line are useful for setting the “Y Error Bars” information in Excel, if an “XY (Scatter)” chart is made using the GC Observed/Expected two lines as the source data. This line can then be used as the “+” and “-” data of the “Custom” Error amount, found inside the “Y Error Bars” tab of the “Format Data Series” dialog box).

GC Observed/Expected:																				
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	
26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	
52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	
78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	
0.7948	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.1982	0.1175	0.0577	0.2860	0.2521	0.4176	0.4615	0.6209	0.8161	0.9145	
5555	0.4783	0.4677	0.4475	0.4406	0.4427	0.4533	0.4572	0.4790	0.4810	0.4746	0.4780	0.4811	0.5026	0.5296	0.6396	0.6564	0.7038	0.7854	0.9814	1.17
82	1.8567	2.3777	2.7890	3.2404	3.5603	3.9121	4.6001	4.3954	4.4683	4.7574	4.8582	5.1635	5.1194	5.0061	5.2853	5.8894	5.7092	5.6513	5.4573	5.2500
3.8497	3.2172	2.8760	2.9451	3.7963	4.6187	8.5737	8.4135	11.1209	40.5385	20.7136	6.2141	77.5668	119.6574	14.8279	0.0000	165.9763	223.7072			
GC Std. Dev.:																				
0.7948	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.1144	0.0831	0.0577	0.1279	0.1127	0.1392	0.1392	0.1506	0.1601	0.1568	0.1
0.187	0.0130	0.0100	0.0079	0.0067	0.0058	0.0053	0.0049	0.0048	0.0045	0.0043	0.0042	0.0041	0.0041	0.0042	0.0040	0.0052	0.0056	0.0065	0.0065	0.007
0.0147	0.0181	0.0215	0.0253	0.0290	0.0331	0.0386	0.0406	0.0442	0.0493	0.0540	0.0594	0.0648	0.0708	0.0804	0.0956	0.1057	0.1182	0.1274	0.1368	0.1368
0.2212	0.2281	0.2397	0.2808	0.4413	0.6468	1.2248	1.7938	3.2121	11.2433	9.2634	6.2141	44.7832	119.6574	14.8279	0.0000	165.9763	223.7072			

Figure 74: 454MappingQC, GC content information portion

5. Quality Score Information (Figure 75)

- a. **Predicted Score** – The quality score values, from 0 to 60
- b. **Observed Quality** – The observed quality score obtained from the read alignments (computed as “Observed Num. Errors” over “Num. Bases With Score” values, see below)
- c. **Observed Accuracy** – The observed quality score expressed as an accuracy percentage
- d. **Num. Bases With Score** – the number of mapped bases having the Predicted Score (only mapped bases are used, because they can be evaluated for accuracy)
- e. **Expected Num. Errors** – the expected number of errors for a quality score, given the number of mapped bases with that quality score
- f. **Observed Num. Errors** – the number of bases which did not match in the read alignment (*i.e.* the alignment column containing that base was not an identity)

Quality Scores:																				
Predicted Score:	0.0	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0	9.0	10.0	11.0	12.0	13.0	14.0	15.0	16			
0	23.0	24.0	25.0	26.0	27.0	28.0	29.0	30.0	31.0	32.0	33.0	34.0	35.0	36.0	37.0	38.0	39.0	40.0	41.0	42.0
49.0	50.0	51.0	52.0	53.0	54.0	55.0	56.0	57.0	58.0	59.0	60.0									
Observed Quality:	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	6.3	7.2	8.1	8.0	10.6	10.9	12.1	12.5	13.7	14		
7	17.9	18.2	18.5	18.8	18.7	19.2	19.1	19.3	19.1	19.5	19.6	19.1	19.7	18.7	20.2	19.2	19.4	18.5	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Observed Accuracy:	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	76.46%	81.04%	84.58%	84.29%	91.22%	91.82%	93.83%	94.34%	95.77%	96		
30%	98.39%	98.49%	98.58%	98.67%	98.65%	98.81%	98.76%	98.82%	98.77%	98.87%	98.90%	98.76%	98.93%	98.66%	99.05%	98.79%	98.86%	98.60%	0.0%	0.0%
0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
Num. Bases With Score:	48929	0	0	0	0	0	0	27787	72050	120120	167520	3340624	2193987	2073383	1998529	2186244	23			
62667	4243122	3314456	4318038	4172001	4753691	5143940	3459810	6447321	3861971	6335745	5890134	4911663	7131367	4417313	17166412	4864666	13713462	6165		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Expected Num. Errors:	48929	0	0	0	0	0	0	5544	11419	15122	16752	265355	138431	103915	79563	69135	60			
693	21266	13195	13655	10480	9485	8153	4356	6447	3068	3998	2952	1955	2255	1110	3425	771	1726	6165	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Observed Num. Errors:	48929	0	0	0	0	0	0	6540	13659	18524	26320	293269	179509	127990	113161	92426	95			
482	68389	50195	61333	55471	64052	61351	42899	75783	47631	71652	65073	60826	76516	59213	162475	58676	156267	863529	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 75: 454MappingQC, Quality score information portion

6. Histogram/Window-based Information

This final section contains the data for larger histograms and datasets, and is organized column-wise, instead of the row-wise layout of the previous sections. It contains three sub-sections

- a. **Read Length and Map Length Histograms** – histograms showing the number of reads of each read length (the “Read Length Histogram” column) and number of reads at each length of the aligned regions per the reference sequence, *i.e.* counting

- only the read bases in the alignment, not the alignment length (the “Map Length Histogram” column). Histogram values are displayed up to 400 bases.
- b. **Errors by Base Position** – plot values showing position-by-position errors in the reads, *i.e.*, how many errors occurred at the Nth base across all the reads. The four columns show the accuracy percentage and equivalent quality score of the accuracy at a specific position (the “Errors by Base Position” columns) and the cumulative accuracy up to that position (the “Cumul. Errors by Base Position” columns)
 - i. Note: if an alignment column contains a gap in the read, that is counted as an error at the previous base position (*i.e.*, any alignment gaps between base 5 and 6 in a read are counted as errors at position 5)
 - c. Cross-Reference Depth and GC Information

The last six columns of this section contain region-by-region statistics of the alignments across the reference, where the reference is evenly divided into 1000 regions

Important Note: This division of the reference into regions has no understanding of repeat regions, and simply reports on the alignments of the uniquely mapping reads. Since repeat reads are not aligned to the reference, the values in this column will count repeat regions as unaligned regions.

- i. The first column displays the **position** in the reference sequence at the center of the region
- ii. **Avg. Depth** – the average alignment depth in the region
- iii. **Min. Depth** – the minimum alignment depth in the region
- iv. **Max. Depth** – the maximum alignment depth in the region
- v. **Depth Score** – a score that is indicative of the shallowness of the alignment in the region. Each alignment column in the region is given a score of “max(0, 4-depth)” where “depth” is the alignment depth of the column. The Depth Score for a region is the sum of the column scores.
 - 01. This score is a very sensitive metric for use in resequencing projects, in order to gauge when enough sequencing has been performed (and the addition of more reads will not fill in any more unaligned or shallowly aligned regions of the reference)
- vi. **GC** – the average GC content of the region

Quality Depth	Read Length Histogram		Errors by Base Position				% Error Rate Avg. Align Depth	Cumul. Errors by Base Position			Cum. Min. Align	
	Max. Align	Depth	Accno	Position	Mapped Length Histogram	Avg. Unique Depth		Obs. Quality	Cum. % Error Rate	Avg. Total Depth		
1	0	0	0	0.39%	24.0	0.39%	24.0	gi 85666109 ref NC_001133.6	2020	12.7	15.5	
15.6	1	27	181	38.5	0.22%	26.7	0.30%	25.2	gi 85666109 ref NC_001133.6	6060	12.8	17.5
2	0	0	0	0.30%	25.3	0.30%	25.2	gi 85666109 ref NC_001133.6	10100	12.7	16.1	
17.5	4	40	0	33.2	0.39%	24.1	0.33%	24.9	gi 85666109 ref NC_001133.6	14140	11.2	14.0
3	0	0	0	35.3	0.51%	22.9	0.36%	24.4	gi 85666109 ref NC_001133.6	18180	11.0	13.5
16.3	7	28	0	34.4	0.59%	22.3	0.40%	24.0	gi 85666109 ref NC_001133.6	22220	10.5	13.8
4	0	0	0	35.9	0.63%	22.0	0.43%	23.6	gi 85666109 ref NC_001133.6	26260	10.9	13.3
14.8	0	26	458	43.2	0.69%	21.6	0.47%	23.3	gi 85666109 ref NC_001133.6	30300	9.8	12.2
5	0	0	0									
13.7	3	25	5									
6	0	0	0									
14.5	3	29	9									
7	0	0	0									
15.1	0	68	3232									
8	0	0	0									

Figure 76: 454MappingQC, Histogram/Window-based Information portion

2.17.1.12 454RefStatus.txt

This file gives statistical information regarding the number of reads that mapped to each reference sequence. It has six fields:

1. **Reference Accession** – accession number of a reference sequence
2. **Num Unique Matching Reads** – how many reads mapped uniquely to the reference. To be considered unique, a given portion of a read may only map to a single reference location. If a portion of a read maps to multiple reference locations (or multiple transcript variants of the same gene in the case of cDNA mapping projects), the read is considered to be a repeat.
3. **Pct of All Unique Matches** – the number of reads mapping uniquely to an individual reference sequence divided by the total number of reads that mapped uniquely to any reference sequence
4. **Pct of All Reads** - number of reads that mapped uniquely to this reference divided by the total number of reads in this mapping project
5. **Pct Coverage of Reference** – number of reference bases covered by at least one uniquely mapping read divided by the total number of bases in this reference
6. **Description** – reference description obtained from the renaming file or annotation files

Reference Accession	Num Unique Matching Reads	Pct of All Unique Matches	Pct of All Reads	Pct Coverage of Reference	Description
chr1 93489	9.8%	9.5%	23.33%		
chr2 64857	6.8%	6.6%	21.62%		
chr17 54998	5.7%	5.6%	25.26%		
chr11 54316	5.7%	5.5%	23.97%		
chr3 53950	5.6%	5.5%	21.93%		
chr12 53786	5.6%	5.5%	23.36%		
chr19 49539	5.2%	5.0%	25.12%		

Figure 77: 454RefStatus file example

2.17.1.13 454AllDiffs.txt

This file contains the list of variations where at least 2 reads differ either from the reference sequence or from other reads aligned at a specific location. SNPs, insertion-deletion pairs, multi-homopolymer insertion or deletion regions, and single-base overcalls and undercalls are reported. Also, in order for a difference to be identified and reported, there must be at least two non-duplicate reads that (1) show the difference, (2) have at least 5 bases on both sides of the difference, and (3) have few other isolated sequence differences in the read. In addition, if the -e option is used to set an expected depth, then there must be at least 5% of that depth in differing reads. Finally, for single-base overcalls or undercalls to be reported, they must have a flow signal distribution that differs from the signal distribution of the reads matching the reference (*i.e.*, not all overcalls and undercalls are reported as variations). Once the difference is identified, all reads that fully span the difference location and have at least 5 additional flanking nucleotides on both sides are used in reporting the difference. For a sample 454AllDiffs.txt file, see Figure 78.

The file consists of one tab-delimited summary line for each difference, plus a multiple alignment of the reads spanning the difference location. Each of the difference summary lines

begin with a '>' character, so the summary list of differences can be extracted from the file using the command "fgrep '>' 454AllDiffs.txt". The full summary lines contain fifteen columns of information, of which the first seven columns are always output, columns eight through eleven are output if gene annotations or known SNP information is given to the GS Reference Mapper (or the -fd option is given), and the rest are output only if the -fd option is given to the GS Reference Mapper. The summary lines contain the following columns:

1. **Reference Accno** - The accession number of the reference sequence in which the difference was detected
2. **Start Pos** - The start position within the reference sequence, where the difference occurs
3. **End Pos** - The end position within the reference sequence, where the difference occurs
4. **Ref Nuc** - The reference nucleotide sequence at the difference location
5. **Var Nuc** - The differing nucleotide sequence at the difference location
6. **Total Depth** - The total number of reads that fully span the difference location
7. **Var Freq** - The percentage of different reads versus total reads that fully span the difference location
8. **Ref AA** - The reference amino acid sequence at the difference location, if it occurs within the coding region of an annotated gene
9. **Var AA** - The differing amino acid sequence at the difference location, if it occurs within the coding region of an annotated gene
10. **Coding Frame** - The gene name at the difference location, if it occurs with the region of an annotated gene
11. **Region name** - The list of known SNP IDs that occur at the difference location
12. **Known SNP's** - The number of forward reads that include the difference
13. **# Fwd w/ Var** - The number of forward reads that include the difference (with -fd only)
14. **# Rev w/ Var** - The number of reverse reads that include the difference (with -fd only)
15. **# Fwd Total** - The total number of forward reads that fully span the difference location (with -fd only)
16. **# Rev Total** -The total number of reverse reads that fully span the difference location (with -fd only)

>Reference >Accno	Start Pos	End Pos	Ref Nuc	Var Nuc	Total Depth	Var Freq	Ref AA	Var AA	Coding Frame	Region Name	Known SNP's
>chr1	4770	4770	A	G	2	100%				rs71252252,rs6682375	
Reads with Difference:											
chr1			4739+	TCCCCCAGCCCCCGGAGACTTAAATACAGGAAG-AAAAAGGCAGGACAGAATTAC-AAGG-TGC		4799					

FWV1IC007I40RV			141-	TCCCCCAGCCCCCGGAGACTTAAATACAGGA-GGAAAAAGGCAGGACAGAATTACGA-G-ATGC		81					
FWV1IC005GACR9			115+	TCCCCCAGCCCCCGGAGACTTAAATACAGGA-GGAAAAAGGCAGGACAGAATTACGA-G-ATGC		175					

Other Reads:											

Figure 78: 454AllDiffs.txt portion example

The multiple alignment for a difference shows the difference location plus approximately 30 bases on either side. The alignment is divided into two sections: the first section displays the reads found to contain the difference and the second, the reads not found to have the difference. Each line of the alignment displays the following information for a read:

1. The identifier for the read
2. The number of duplicate reads whose alignment matches this read (when there are duplicates of the read)
3. The position of the read's first base displayed in the alignment region
4. The orientation of the read in the displayed alignment (“+” is forward orientation, “-” is reverse-complement orientation, relative to the reference)
5. The aligned bases of the read
6. The position of the read's last base displayed in the alignment region

If the difference is a homopolymer undercall or overcall, the alignment rows are followed by a histogram depicting the signal distribution for the homopolymer.

2.17.1.14 *454HCDiffs.txt*

This file contains the same type of information as the *454AllDiffs.txt* file (section 2.17.1.13, above), but restricted to the “High-Confidence” differences. The GS Reference Mapper application uses a combination of flow signal information, quality score information and difference type information to determine if a difference is High-Confidence. The general rules are:

- There must be at least 3 non-duplicate reads with the difference, unless the `-e` option is specified, in which case at least 10% of the expected depth must have the difference
- There must be both forward and reverse reads showing the difference, unless there are at least 5 reads with quality scores over 20 (or 30 if the difference involves a 5-mer or higher)
- If the difference is a single-base overcall or undercall, then the reads with the difference must form the consensus of the sequenced reads (*i.e.*, at that location, the overall consensus must differ from the reference) and the signal distribution of the differing reads must vary from the matching reads (and the number of bases in that homopolymer of the reference).

2.17.1.15 *454HCStructVars.txt* and *454AllStructVars.txt*

These two files show rearrangement points and rearrangement regions observed in the reads of the data set analyzed, relative to the reference. The structure of the two files is the same, but *454AllStructVars.txt* uses a lower threshold for the frequency or variant reads relative to non-variant reads than *HCStructVars.txt* for reporting purposes.

The columns found on the summary line for each variation are described below. Some columns (the Region Name columns) require that annotations be supplied to the project. Some other columns are output only if the “`-fd`” option is specified.

1. **Ref Accno1** – the accession number of the reference sequence on one side of the variation
2. **Ref Pos1** – the reference position on one side of the variation
3. **Var Side1** – a direction arrow “`→`” or “`←`” describing the direction of the variation on the reference (*i.e.*, the 3' end of the reads or paired-end clones that diverge from the reference occur in this direction)

4. **Region Name1** – the gene name, or annotated region name, covering the location in the reference denoted by Ref Accno1 and Ref Pos1 (Gene or Region annotation must be included in the project for this field to contain a value)
5. **Ref Accno2** – the accession number of the reference sequence on the other side of the variation, if known. (If only one side of the variation is known, a question mark is given here)
6. **Ref Pos2**– the reference position on the other side of the variation, or a question mark if only one side is known
7. **Var Side2** – a direction arrow “→” or “←” for the direction on the other side of the variation, or a question mark if only one side is known
8. **Region Name2** – the gene name, or annotated region name, covering the location in the reference denoted by Ref Accno2 and Ref Pos2 (Gene or Region annotation must be included in the project for this field to contain a value)
9. **Total Depth** – the number of reads (for rearrangement points) or pairs (for rearrangement regions) covering the variation location(s)
10. **Var Freq** – the percentage of the reads/pairs that support the variation
11. **Deviation Length** – if both sides of the variation occur on the same reference, this is the distance between the two variation locations
12. **Type** – the string “Point” or “Region” to denote whether the rearrangement is a rearrangement point identified by split-read alignments or a rearrangement region identified by paired-end reads
13. **# Fwd w/ var** – number of reads on the forward orientation that contain the variation (requires -fd option)
14. **# Rev w/ var** – number of reads on the reverse orientation that contain the variation (requires -fd option).
15. **# Fwd Total** – total number of reads in the forward orientation that map to this area of the reference (requires -fd option).
16. **# Rev Total** – total number of reads in the reverse orientation that map to this area of the reference (requires -fd option).

Ref Accno1	Ref Pos1	Var Side1	Region Name1	Ref Accno2	Ref Pos2	Var Side2	Region Name2	Total Depth	Var Freq	Deviation Length	Type	# Fwd w/ var	# Rev w/ var	# Fwd Total	# Rev Total	Region	O	O	O
gi 85666109 ref NC_001133.6	25291	<--	Hox99	gi 85666109 ref NC_001133.6	191035	<--	CDR5	19	73.68	165743	Region	0	0	0	0				
			FKHERYTO1B7NQR				gi 85666109 ref NC_001133.6	25291	<--	gi 85666109 ref NC_001133.6		193599	<--						
			FKHERYTO1ELGSV				gi 85666109 ref NC_001133.6	25459	<--	gi 85666109 ref NC_001133.6		191939	<--						
			FKHERYTO1AXFZX				gi 85666109 ref NC_001133.6	26564	<--	gi 85666109 ref NC_001133.6		192702	<--						
			FKHERYTO1CXLRR				gi 85666109 ref NC_001133.6	26595	<--	gi 85666109 ref NC_001133.6		193584	<--						
			FKHERYTO1BXZVI				gi 85666109 ref NC_001133.6	26640	<--	gi 85666109 ref NC_001133.6		192140	<--						
			FKHERYTO1AU6VF				gi 85666109 ref NC_001133.6	26663	<--	gi 85666109 ref NC_001133.6		192473	<--						
			FKHERYTO1A5RGP				gi 85666109 ref NC_001133.6	26721	<--	gi 85666109 ref NC_001133.6		191646	<--						
			FKHERYTO1A41FP				gi 85666109 ref NC_001133.6	26771	<--	gi 85666109 ref NC_001133.6		192198	<--						
			FKHERYTO1D58EM				gi 85666109 ref NC_001133.6	26944	<--	gi 85666109 ref NC_001133.6		191035	<--						
			FKHERYTO1EAZOP				gi 85666109 ref NC_001133.6	26978	<--	gi 85666109 ref NC_001133.6		192315	<--						
			FKHERYTO1DZQNF				gi 85666109 ref NC_001133.6	27230	<--	gi 85666109 ref NC_001133.6		192234	<--						
			FKHERYTO1CWJX3				gi 85666109 ref NC_001133.6	28170	<--	gi 85666109 ref NC_001133.6		192114	<--						
			FKHERYTO1DXWYO				gi 85666109 ref NC_001133.6	29472	<--	gi 85666109 ref NC_001133.6		191299	<--						
			FKHERYTO1BOBAP				gi 85666109 ref NC_001133.6	29775	<--	gi 85666109 ref NC_001133.6		191369	<--						

Figure 79: 454HCStructVars.txt portion example of rearrangement region

2.17.1.15.1 Rearrangement Points

A coupled rearrangement point is found to have both a beginning and ending. A single rearrangement point has only one or the other.

In the 454HCStructVars.txt file, coupled rearrangement points are displayed twice. The beginning is inverted, such that reads that do not contain the variation are displayed first,

followed by the reads that do. The ends are in typical order, showing the reads with the variation before those without it. An example is below:

```

>gi|50593115|ref|NC_001134.7| 46830 --> TNF1 gi|50593115|ref|NC_001134.7| 47040 <-- TNF1 11 100.00 209 Point 6 5 6
Other Reads:
gi|50593115|ref 46792+ CCTCCAOGTTTT-GGT-AAGCTGCOGGGATAGGAAGTGTAACTGACCCATCAACAACCTGGGTTTCTCGCOGCGAAA 46869
Reads with Difference:
FKHERYTO1C2SCP_ (2) 259+ CCTCCAOGTTTT-GGT-AAGCTGCOGGGATAGGAAGTGTAA * 297
FKHERYTO1COGFH_ 258+ CCTCCAOGTTTT-GGT-AAGCTGCOGGGATAGGAAGTGTAA 296
FKHERYTO1AIBJ9_ 172+ CCTCCAOGTTTT-GGT-AAGCTGCOGGGATAGGAAGTGTAA 210
FKHERYTO1AJDG4_ 142+ CCTCCAOGTTTT-GGT-AAGCTGCOGGGATAGGAAGTGTAA 180
FKHERYTO1BE3P5_ 95- CCTCCAOGTTTT-GGT-AAGCTGCOGGGATAGGAAGTGTAA 57
FKHERYTO1CH2A2_ 102+ CCTCCAOGTTTTGGT-AAGCTGCOGGGATAGGAAGTGTAA 141
FKHERYTO1DW0J9_ 85+ CCTCCAOGTTTT-GGT-AAGCTGCOGGGATAGGAAGTGTAA 123
FKHERYTO1CYJN4_ (3) 82- CCTCCAOGTTTT-GGT-AAGCTGCOGGGATAGGAAGTGTAA 44
FKHERYTO1AXKEX 272- CCTCCAOGTTTT-GGT-AAGCTGCOGGGATAGGAAGTGTAA 234
FKHERYTO1AHMST_ 206- CCTCCAOGTTTT-GGTAAAGCTGCOGGGATAGGAAGTGTAA 167
FKHERYTO1BG7YF_ 228- CCTCCAOGTTTT-GGT-AAGCTGCOGGGATAGGAAGTGTAA 190
Reads with Difference:
gi|50593115|ref 47000+ AACGATAGTGTGAGTGTGCGAGTAGCCATGCATAATGCAGCOGATCAAGAATTTGTC-TGTTTTTTT-GGTCCATTTTCATC 47077
FKHERYTO1CYJN4_ (3) 43- C-GATCAAGAATTTGTC-TGTTTTTTT-GGTCCATTTTCATC 7
FKHERYTO1BE3P5_ 56- C-GATCAAGAATTTGTC-TGTTTTTTT-GGTCCATTTTCATC 20
FKHERYTO1AJDG4_ 181+ C-GATCAAGAATTTGTC-TGTTTTTTT-GGTCCATTTTCATC 216
FKHERYTO1C2SCP_ (3) 298+ C-GATCAAGAATTTGTC-TGTTTTTTT-GGTCCATTTTCATC 334
FKHERYTO1DW0J9_ 124+ C-GATCAAGAATTTGTC-TGTTTTTTTGGTCCATTTTCATC 161
FKHERYTO1CH2A2_ 142+ C-GATCAAGAATTTGTC-TGTTTTTTT-GGTCCATTTTCATC 178
FKHERYTO1AHMST_ 166- C-GATCAAGAATTTGTC-TGTTTTTTT-GGTCCATTTTCATC 130
FKHERYTO1BG7YF_ 189- C-GATCAAGAATTTGTC-TGTTTTTTT-GGTCCATTTTCATC 153
FKHERYTO1AIBJ9_ 211+ C-GATCAAGAATTTGTC-TGTTTTTTT-GGTCCATTTTCATC 247
FKHERYTO1AXKEX 233- C-GATCAAGAATTTGTCM-OGTTTTTTT-GGTCCATTTTCATC 197
    
```

Figure 80: 454HCStructVars.txt example of coupled rearrangement point

For single rearrangement points, the unknown end is represented with a series of questions marks (“???”) for the reference, min pos, and max pos, as seen below:

```

>gi|85666109|ref|NC_001133.6| 33255 --> ? ? ? 11 63.64 - Point 4 3 6 5
Other Reads:
gi|85666109|ref 33219+ ACGGACCTGCATGAGTACACAG-A-TGGT-C-CTATTGCTTGC-AAAAAGGGCTACTATAA-CATTGTTC-ATATTGG 33291
FKHERYTO1BZ6IL_ 249+ ACGGACCTGCATGAGTACACAG-A-TGGT-C-CTAT--CTT 283
FKHERYTO1DI52V_ (2) 119- ACGGACCTGCATGAGTACACAG-A-TGGT-C-CTAT--CTT 85
FKHERYTO1D5JQ2_ 154+ ACGGACCTGCATGAGTACACAG-A-TGGT-C-CTAT--CTT 188
FKHERYTO1AH3D7 (2) 214- ACGGACCTGCATGAGTACACAG-A-TGGT-C-CTAT--CTT 180
FKHERYTO1CF1YE_ 281- ACGGACCTGCATGAGTACACAG-A-TGGT-C-CTAT--CTT 247
FKHERYTO1DLDED_ 52+ ACGGACCTGCATGAGTACACAG-A-TGGT-C-CTAT--CTT 86
FKHERYTO1D9CLE_ 17+ ACGGACCTGCATGAGTACACAG-A-TGGT-C-CTAT--CTT 51
Other Reads:
FKHERYTO1B6FZL_ 183- CA-AGGAATGGTGGCT---GCTTGC-AAAAAGGGCTACTATAA-CATTGTTC-ATATTGG 128
FKHERYTO1DC6HX_ 135+ A-TGGTGGCT---GCTTGC-AAAAAGGGCTACTATAA-CATTGTTC-ATATTGG 184
FKHERYTO1CRX8F_ 327+ T-GCTTGC-AAAAAGGGCTACTATAA-CATTGTTC-ATATTGG 367
FKHERYTO1CDXIT_ 146- T-GCTTGC-AAAAAGGGCTACTATAA-CATTGTTC-ATATTGG 106
    
```

Figure 81: 454HCStructVars.txt example of a rearrangement point

2.17.1.15.2 Rearrangement Regions

Rearrangement Regions are represented by clusters of paired-end reads that are considered False Pairs (see section 4.7.4 for a brief discussion of True Pair vs. False Pair reads). The detection of Rearrangement Regions currently depends on finding clusters of False Pairs of

Paired End reads near each other. When a cluster shows a consistent deviation from the reference (varying either in size or expected orientation), a rearrangement region is reported. An example rearrangement region is shown in Figure 79.

In the above examples, the +/- refers to whether the lowest location was the left-half of the pair (signaling that the read was in the same strand as the reference) or the right-half of the pair (signaling that the read was in the opposite strand of the reference). The arrows give the read direction for each half of the pair.

2.17.2 GS Reference Mapper cDNA / Transcriptome Output

Output for cDNA / transcriptome mapping is found in the 454RefStatus.txt file (see Section 2.17.1.12) and the 454GeneStatus.txt file (below). There is also relevant information in the 454MappingQC.xls file (Section 2.17.1.11), but with certain differences described in Section 2.17.2.2.

2.17.2.1 454GeneStatus.txt

This gives statistical information regarding the number of reads that mapped exclusively to each gene. It has five fields:

1. **Gene Name** – the name is the gene's identifier in the related annotation database
2. **Num Unique Matching Reads** – the number of reads mapping exclusively to one or more of the transcript variants for each gene. Note that reads mapping equally well to more than one transcript variant for a gene are still considered Repeats in terms of their final ReadStatus, but are included in this statistic if they only map to variants of a single gene.
3. **Pct of all unique matches** – the number of reads exclusively mapped to the gene (as described above) divided by the total number of reads that mapped exclusively to any of the reference sequences
4. **Pct of All Reads** – the number of reads exclusively mapped to the gene (as described above) divided by the total number of reads in this mapping project
5. **Description** – the description obtained from the annotation data or renaming file

Gene Name	Num Unique Matching Reads	Pct of All Unique Matches	Pct of Gene All Reads	Description
chr1	82677	14.8932	8.40755	
chr2	56855	10.2417	5.78167	
chr17	50584	9.11207	5.14396	
chr11	50520	9.10054	5.13746	
chr12	48729	8.77791	4.95533	
chr3	48630	8.76008	4.94526	
chr19	45368	8.17247	4.61354	

Figure 82: 454GeneStatus.txt file example

2.17.2.2 454MappingQC.xls

The following six columns present in the cumulative errors by base position table for Genomic mapping projects (see Section 2.17.1.11) do not appear when mapping cDNA reads to a cDNA

reference: **accno**, **position**, **Avg Unique Depth**, **Min Unique Depth**, **Max Unique Depth**, and **Depth Score**. In their place, the following five columns are reported:

1. **Type** – cDNA references are grouped into three categories: Short, Medium or Long. References less than 1500 base pairs long are categorized as “Short”. References greater than or equal to 1500 base pairs long but less than 3500 base pairs are categorized as “Medium”. References equal to or greater than 3500 base pairs are categorized as “Long”.
2. **Percent** – this column is a histogram of averaged length percentiles for all cDNA references of a given Type
3. **Obs./Exp. Depth** – number of reference bases within the percentile that were covered by reads, divided by the total number of reference bases in the percentile
4. **Avg. Depth** – average of the number of reads mapping to each reference in the given length percentile
5. **MaxDepth** – maximum number of reads mapping to a single reference in the given length percentile

Quality	Read Length	Histogram Type	Percent	Errors by Base Position				% Error Rate	Cumul. Obs.	Errors by Base Position			Cum.
				Mapped	Length	Histogram	Depth			Obs. Quality	Cum. %	Error Rate	
1	0	0	3.51%	14.5	3.51%	14.5	Short	5'	0.23	10.29	73		
2	0	0	1.97%	17.1	2.74%	15.6	Short	1	0.27	12.05	130		
3	0	0	1.60%	18.0	2.36%	16.3	Short	2	0.32	14.45	272		
4	0	0	1.19%	19.2	2.07%	16.8	Short	3	0.39	17.39	323		
5	0	0	1.10%	19.6	1.87%	17.3	Short	4	0.46	20.57	430		
6	0	0	0.96%	20.2	1.72%	17.6	Short	5	0.53	23.35	519		
7	0	0	0.80%	20.9	1.59%	18.0	Short	6	0.61	27.00	644		

Figure 83: 454MappingQC.xls file example for mapping to a cDNA reference

3. SFF TOOLS COMMANDS

The Genome Sequencer FLX System data analysis software package contains five programs related to the handling of Standard Flowgram Format (SFF) files or other read data. These programs allow you to:

- combine files into the Standard Flowgram Format (SFF) and filter reads present in SFF files [sfffile]
- access SFF file information [sffinfo]
- dynamically generate an SCF trace file suitable for display by the consed software [sff2scf]
- prepare FASTA files with the necessary annotations for use by the GS *De Novo* Assembler and GS Reference Mapper [fnafile]
- rescore older SFF files (generated with software versions prior to v. 1.1.03) using the new phred-based quality scores [sffrescore]

These tools are all evoked at the UNIX command line level. The descriptions below assume that the reader is familiar with the Genome Sequencer FLX System data and formats, including the SFF file format. For more details on the SFF and other file formats used in the Genome Sequencer FLX System software, see the General Overview section of this manual.

3.1 sfffile

The sfffile command constructs a single SFF file containing the reads from a list of SFF files and/or datasets from sequencing Runs. The reads written to the new SFF file can be filtered using inclusion and exclusion lists of accession numbers, and their sequence trimming points or flowgram lengths can be adjusted. This command is used to pool the results of multiple sequencing Runs (or specified regions of one or multiple Runs) into a single SFF file, to simplify further handling of the combined data.

The sfffile command uses the following syntax:

```
sfffile [options...] [MIDList@](sfffile | datadir)
```

where "MIDList" is a multiplexing information string used to filter the set of file reads output. (See the Appendix 4.6 for details on the use of MIDs in the data analysis.)

Command	Description
sfffile	Constructs a single SFF file containing the reads from a list of SFF files and/or datasets from older sequencing Runs.

Option / Argument	Description
-o output	The default output of the sfffile program is a single SFF file containing all the reads of the SFF files / sequencing Runs given in argument on the command line, output to the file "454Reads.sff". The "-o" option allows the user to specify a different filename.
-r	This option re-generates the phred-based quality scores for each of the input reads using the current quality scoring table, and overwrites the existing quality scores with these new quality scores in the output file.
-i accnofile	By default, all reads in the inputs are written to the output SFF file. If the "-i" (Include only these reads in output) or "-e" (Exclude these reads from output) options are used, the accession numbers given in the files specified will change the reads that are output. (If both -i and -e are used, the output will include the reads that are in the -i file and not in the -e file.) Those files should list the accessions one per line, where the accession should be the first word on the line after an optional '>' (<i>i.e.</i> , if the line begins with '>', that character is skipped, and the following characters up to the first whitespace character is read as the accession). These options are cumulative, <i>i.e.</i> if multiple -i options are given, the reads included will be the union of the -i accession lists.
-e accnofile	These two options adjust the trim points for some or all reads in the output SFF file. The specified "trimfile" should contain one or more lines consisting of (1) a read accession number, (2) a starting trimpoint, and (3) an ending trimpoint, separated by whitespace characters or where the trimpoints are separated by a dash (<i>e.g.</i> , "accno 12 543" or "accno 12-543"). The trimpoint values are 1-based positions that denote the first and last base of the trimmed region (<i>e.g.</i> for a read 800 bases in length, the lines above specify that bases 1-11 and 544-800 should be ignored, and bases 12-543 form the trimmed region of the read). A value of 0 specifies that the beginning or end of the read should be used (<i>e.g.</i> for a read 800 bases in length, the line "accno 12 0" sets the trimmed region to 12-800). The -t option will merge the given trimpoints with any existing trim points for the input read, writing the largest starting trimpoint and smallest ending trimpoint into the output SFF file. By contrast, the -tr option will "reset" the trimpoints, using only the trimpoint information occurring in this file. Note: The use of this option does not limit the reads that will be written into the output SFF file. To only output the reads in the trim file, the -i option must also be used, with this file as its argument.
-t trimfile or -tr trimfile	

<p>-c # or -gs20 or -20 or -gsflx or -flx</p>	<p>These options change the length of the flowgrams written to the output SFF file, shortening or lengthening the flowgrams (and associated basecalled sequence) to the given number of cycles in the argument. The -c option allows an arbitrary number of cycles to be specified, while the -gs20 and -20 options are a shortcut for “-c 42” and the -gsflx and -flx options are a shortcut for “-c 100” (the number of cycles for sequencing Runs carried out using the GS 20 chemistry and the GS FLX standard chemistry, respectively). If a flowgram is lengthened, 0.0 flowgram values will be added to the end of the flowgram, and no bases will be called for those flowgram values. If a flowgram is shortened, the ending flows, and all basecalls made from those flows, will be removed from the read’s entry. Reads whose flowgram length matches the specified number of cycles will not be altered. Note: If a flowgram and its read are shortened, the removed data does not exist in, and is not recoverable from, the output SFF file.</p>
<p>-s setname</p>	<p>This option “splits” the input reads into separate output files, based on the multiplexing sequences occurring at the beginning of the reads. The argument to the option is an MID set name (occurring in the MID configuration file). Each input read is matched against each MID in the set, and output to a separate file. The output files will be named by adding the MID name as a suffix to the output file (but before the “.sff” suffix). So, if the default GS Multiplex Identifiers (MID) Kit set is given as the set name and the output filename is “454Reads.sff”, the output files will be “454Reads.MID1.sff”, “454Reads.MID2.sff”, and so on. Note: If no reads match a given MID sequence, the corresponding output file is not written (<i>i.e.</i>, only files with one or more reads will be output).</p>
<p>[-mcf <i>filename</i>]</p>	<p>This option specifies a different MID configuration file to be used by the command for decoding the multiplex information appearing on the command line.</p>
<p>[-pick #] or [-pickb #]</p>	<p>This option tells sffile to generate an output file containing a certain number of bases, by randomly “picking” reads from the input. The argument can be a number, optionally followed by either a ‘k’ or ‘m’ to specify thousands or millions of bases, respectively. For example, “-pick 3000000” and “-pick 3m” both tell sffile to pick random reads from the input and generate an output file containing 3 million bases. Note: Reads are not broken to achieve exactly the number of bases specified, so the actual number of bases in the output file may differ slightly from the desired number.</p>

[-pickr #]	This option tells sffile to generate an output file containing a certain number of reads, by randomly “picking” reads from the input. The argument number can be followed by a ‘k’ or ‘m’ to specify thousands or millions of reads, respectively. Reads in any input SFF files are merged directly into the single output SFF file generated by this command. If SFF files are not present in a Data Processing folder (e.g. for a Run whose data has been processed with a version of the Genome Sequencer System software anterior to 1.0.52), the signal processing step of the GS Run Processor application must be rerun on the Data Processing folder, and can then merged into the output SFF file generated by the sffile command. Input “D_...” directories may be prepended with a list of regions separated by a colon. For example, “1,3-5,7:R_dir/D_dir” tells sffile to use regions 1, 3, 4, 5 and 7 of R_dir/D_dir. An optional multiplexing information string can be prepended to each file/data-directory argument.
[-nmft]	By default, a “manifest” listing the set of sequencing Runs that were the source of the reads in an SFF file is written into the index section of the SFF file (to provide an explicit traceback to the origin of the reads). This includes the Run name (the R_... name), the Data Processing name (the D_... name) and the full path to the Data Processing directory used in the conversion from Run data to SFF file(s). The “-nmft” option prevents the propagation of the manifest from the input SFF files into the output SFF file.
[--help]	Displays a usage line and short description of the command.
[--version]	Displays the current software version of the command.
[MIDList@] (sffile datadir)	Path to one or more SFF files and/or Data Processing directories (“D_...”). Reads in any input SFF files or the SFF files in the Data Processing directories are merged directly into the single output SFF file generated by this command. Input “D_...” directories may be prepended with a list of regions separated by a colon. For example, “1,3-5,7:R_dir/D_dir” tells sffile to use regions 1, 3, 4, 5 and 7 of R_dir/D_dir. An optional multiplexing information string can be prepended to each file/data-directory argument (see Appendix 4.6).

3.1.1 Using sffile for Evaluation of Individual Reads

One of the most common uses of the sffile program is to collect and organize your sequences in terms of a user’s projects, instead of the standard timestamp- and Run-based organization generated by the Genome Sequencer FLX Instrument. For example, multiple sequencing Runs for a large sequencing project can easily be combined into a single “mygenome.sff” file using

```
sffile -o mygenome.sff path_to_D1 path_to_D2 path_to_D3...
```

This file could then be given to the GS *De Novo* Assembler and GS Reference Mapper software, or carved and divided into separate pieces (see below), while still preserving the traceback information to the source data, since:

- The universal accessions for each read encode the Run timestamp, the region and the X,Y location of the read, and all but ensure uniqueness for the accessions.

- The manifest of the SFF file provides the mapping from the Run prefix of the universal accessions to the actual Run and Data Processing directories used to generate the read. When universal accessions are given to the reads, the manifest is kept updated through multiple invocations of `sffile` (as best it can).

Thus, if a particular read needs investigation, the user can invoke the `sffinfo` command (see section 3.2, below) for that read to see the traceback information, including the source Run and Data Processing directories. The GS Run Browser (see Part B of this Reference Manual) could be used to evaluate the read within the context of the Run [where the Find location feature on the Wells tab allows entry of a universal accession and will then center the image and place a marker at the read's X,Y location].

3.1.2 Using sffile to Extract Read Subsets

Another common use of `sffile` is to extract subsets of the reads from a sequencing Run or a set of sequencing Runs, to perform further processing. Using the `-i` and `-e` options, one can construct an SFF file containing only the reads of interest, and then use that for further processing, e.g. input it into the GS *De Novo* Assembler or GS Reference Mapper applications. For example, a user can extract the singletons from an assembly (when performing, say, survey sequencing) by executing the following commands (which assume that the current working directory is the output directory of the assembly):

```
grep Singleton 454ReadStatus.txt > singles.txt
sffile -o singles.sff -i singles.txt sff/*
sffinfo -s singles.sff > singles.fna
```

The Unix `grep` command extracts the lines in the `454ReadStatus.txt` file that list the accession numbers of the singleton reads, then the `-i` option in the `sffile` command is used to signify that the command should “include” the reads in the new SFF file. These reads are then output to FASTA format using `sffinfo`.

Or, after running the GS Reference Mapper application, you can retrieve the reads that did not map to your reference sequence (so that, for example, you can give them as input to the GS *De Novo* Assembler software to see if there are any contigs that are part of what you sequenced but are not part of the reference) by using the following commands (which assume that the current working directory is the output directory of the assembly):

```
grep Unmapped 454ReadStatus.txt > unmapped.txt
sffile -o unmapped.sff -i unmapped.txt sff/*
runAssembly unmapped.sff
```

The Unix `grep` command extracts the lines in the `454ReadStatus.txt` file that list the accession numbers of the unmapped reads, then the `-i` option in the `sffile` command is used to signify that the command should “include” the reads in the new SFF file. (Similarly, there is a `-e` option to “exclude” reads from the output file.)



The `-i` and `-e` options cannot take a list of files, so to specify the inclusion or exclusion of multiple files on the command, you must precede each with an `-i` or `-e` option.

3.2 sffinfo

The `sffinfo` command extracts read information from an SFF file, and reports it in text form. It can be used for generating the FASTA and quality score files of the read sequences, which can be given to standard bioinformatics tools. The `sffinfo` command uses the following syntax:

```
sffinfo [options...] [- | sfffile] [accno...]
```

Command	Description
<code>sffinfo</code>	The <code>sffinfo</code> command extracts read information from an SFF file, and reports it in text form. By default, a text summary of all the read information is output, but the output can be limited to only the sequences, quality scores, flowgrams or manifest. All output is written to standard output.

Option / Argument	Description
<code>-a</code> or <code>-accno</code>	This option limits the output to only the accession numbers.
<code>-s</code> or <code>-seq</code>	This option limits the output to only the sequences.
<code>-q</code> or <code>-qual</code>	This option limits the output to only the quality scores.
<code>-f</code> or <code>-flow</code>	This option limits the output to only the flowgrams.
<code>-t</code> or <code>-tab</code>	The default format for the sequences, quality scores and flowgram is FASTA. This option changes this to tab-delimited lines.
<code>-n</code> or <code>-notrim</code>	By default, the trimmed data is output. The <code>-notrim</code> option changes this to output the untrimmed sequences or quality scores.
<code>-m</code> or <code>-mft</code>	The command does not output the manifest by default. This option outputs the manifest text, if it exists in the SFF file. In this case, the <code>-tab</code> and <code>-notrim</code> options and the <code>accnos</code> on the command line are ignored.
<code> [--help]</code>	Displays a usage line and short description of the command.
<code> [--version]</code>	Displays the current software version of the command.
<code>- sfffile</code>	Path to the SFF file whose data will be output. If this is replaced by a “-” on the command line, then standard input is read for the SFF contents.
<code>accno...</code>	List of the accession numbers of the reads whose data will be output. If no <code>accnos</code> are given on the command line, the information from all reads in the file are output. If an SFF file is specified and it contains an <code>sfffile</code> -created index, then an index-based lookup is used and the reads are output in the order they appear on the command line. If not, the complete SFF file is scanned and the reads are output in the order they appear in the file.



Only one of `-accno`, `-seq`, `-qual`, `-flow` or `-mft` may be specified: the program uses the last one specified on the command line.

3.3 sff2scf

The sff2scf command converts the flowgram information from a read's SFF entry into an SCF file, creating a synthetic "trace" for the read. In this software version, it has been written mainly for program-triggered execution by the consed software, and minimal support is provided for direct command line use. (The main reasons for this are that the command converts a 1000 byte SFF entry into a 45,000 byte SCF file, and that the synthetic trace generated cannot be used by other software, like polyphred, that expect a trace from a Sanger read.)

The sff2scf command uses the following syntax:

```
sff2scf locationstring [outputfile]
```

Command	Description
sff2scf	The sff2scf command extracts one read's information from an SFF file and converts it into an SCF file (or performs "call throughs" to access SCF data for Sanger reads).

Option / Argument	Description
[--help]	Displays a usage line and short description of the command.
[--version]	Displays the current software version of the command.
locationstring	A "locationstring" that tells sff2scf what path or what command to use to access the read's trace information. See below for a description of the format of this string.
outputfile	The path to where the SCF file should be written. If no outputfile is given, the output is written to <code>/tmp/locationstring</code> (the location that consed expects to read the data).

The locationstring argument can take one of 4 forms, which allow the sff2scf command to fully support accessing SFF and SCF data for the reads that may appear in an assembly or mapping output from the Genome Sequencer System. The forms are the following:

1. If the locationstring begins with "sff:", then it specifies an SFF file and read accession to get the SFF data, and will cause the generation of a synthetic SCF trace file. The format of the locationstring should be either "sff:path:accno" or "sff:-f:path:accno" and is processed using the following rules:
 - a. Any instance of the string "._:" in the path will be translated into the character "/", so that the locationstring can be used as a simple filename when executed by consed even though it encodes a path through the directory structure.
 - b. If the path begins with the character "/" (after rule a), it is treated as the absolute path to the SFF file.
 - c. If the path does not begin with "/", both "path" and "../sff_dir/path" are checked for the existence of a file (the "../sff_dir/path" format is the form used by the GS *De Novo* Assembler and the GS Reference Mapper applications when generating a full consed directory structure).
 - d. The command "sffinfo truepath accno" is executed to access the read's SFF data (where "truepath" is the path determined above).
 - e. If the "-f" string appears immediately following "sff:", the read's flowgram is reverse-complemented (or "flipped") prior to the generation of the SCF file. (This occurs when the read is the left half of a 454 Paired End read. The consed software (and other

- software) assume a directionality of Paired End reads, where the clone that was used to generate the Paired End read is assumed to extend off the 3' end of both reads in the pair. In the processing of 454 Paired End reads, this is not the case for the left half of the Paired End read (the clone extends from the 5' end of this half). To support *consed* and the standard view of Paired End reads, the GS *De Novo* Assembler reverse-complements all left halves of paired end reads, and so this command must support the reverse-complementing in the generation of the SCF trace).
2. If the locationstring begins with "file:", it is treated as specifying the path to an SCF file, of the form "file:path", and is processed using the following rules:
 - a. Any instance of the string "._:" in the path will be translated into the character "/", so that the locationstring can be used as a simple filename when executed by *consed* even though it encodes a path through the directory structure.
 - b. If the path begins with the character "/" (after rule a), it is treated as the absolute path to the SCF file.
 - c. If the path does not begin with "/", both "path" and "../chromat_dir/path" are checked for the existence of a file.
 - d. The bytes of the file (determined from above) are copied into the output file without change.
 3. If the locationstring begins with "cmd:" it is treated as specifying a command string to be executed to access the SCF contents, is assumed to have the form "cmd:commandline", and is processed using the following rules:
 - a. Any instance of the string "._:" in the command line will be translated into the character "/", so that the locationstring can be used as a simple filename when executed by *consed* even though it encodes a path through the directory structure.
 - b. After the above, any colon ":" will be translated into a space " ", so that the locationstring can be used as a CHROMAT string, which cannot contain whitespace, in the ACE and PHD files used by *consed* (and will be the strings passed to *sff2scf* file by *consed*).
 - c. The resulting command line string is executed using the proper command, and the standard output of the command is copied into the output file without change.
 4. Otherwise, the locationstring is treated as a path to an SCF file, and processed using the following rules:
 - a. If the path begins with "/", that is treated as the absolute path to the file
 - b. If the path does not begin with "/", both "path" and "../chromat_dir/path" are checked for the existence of a file, where the "../chromat_dir/path" form exists to support the addition of new reads when running *consed* itself (the default operation of *consed*, looking in the ../chromat_dir directory to find the SCF file for a read).
 - c. The bytes of the file (as determined above) are copied into the output file.



The complex syntax of the location strings is needed because when *sff2scf* is used within *consed*, (1) it is responsible for accessing the traces for all the reads in the assembly (454 Sequencing reads with SFF data, Sanger reads with SCF files, or commands that generate SCF data); (2) it is only given a single argument (the "CHROMAT" string in the ACE file or PHD file, which cannot contain whitespace) and must produce a file named by that same argument, and (3) that argument string must be a valid simple filename (not a path or command), even if the source of the data is a full path or command string.

3.4 fnafile

The fnafile command provides similar functionality for FASTA files as the sfffile command provides for SFF files, and in addition provides a mechanism to easily generate and add annotation strings to the description lines of reads in the file. The command constructs a single FASTA file containing the reads from a list of FASTA, PHD or SCF files, or directories containing FASTA, PHD or SCF files. The reads written to the new FASTA file can be filtered using inclusion and exclusion lists of accession numbers, and their sequence trimming points or annotation strings can be adjusted. This command is used to pool the results of many Sanger reads into a single FASTA file, to simplify further handling of the combined data.

The fnafile command uses the following syntax:

```
fnafile [options] (fastafile or PHDfile or SCFfile or directory)...
```

Command	Description
fnafile	Constructs a single FASTA file (plus associated quality score file) containing the reads from a list of FASTA files (possibly with associated quality score files), PHD files, SCF files or directories containing FASTA, PHD or SCF files.

Option / Argument	Description
-o output	The “-o” option allows the user to specify a different filename for the output fna and qual files.
-i accnofile	By default, all reads in the inputs are written to the output file. If the “-i” (Include only these reads in output) or “-e” (Exclude these reads from output) options are used, the accession numbers given in the files specified will change the reads that are output. The associated accnofile files should list the accession numbers one per line. These options are cumulative, <i>i.e.</i> if multiple -i options are given, the reads included will be the union of the -i accession lists.
-e accnofile	

<p>-t trimfile or -tr trimfile</p>	<p>These two options adjust the trim points for some or all reads in the output file. The specified “trimfile” should contain one or more lines consisting of (1) a read accession number, (2) a starting trimpoint and (3) an ending trimpoint, separated by whitespace characters or where the trimpoints are separated by a dash (e.g., “accno 12 543” or “accno 12-543”).</p> <p>The trimpoint values are 1-based position values that denote the first and last base of the trimmed region (e.g. for a read 800 bases in length, the example above specify that bases 1-11 and 544-800 should be ignored, and bases 12-543 form the trimmed region of the read). A value of 0 specifies that the beginning or end of the read should be used (e.g. for a read 800 bases in length, the line “accno 12 0” sets the trimmed region to 12-800).</p> <p>The –t option will merge the given trimpoints with any existing trim points for the input read, writing the largest starting trimpoint and smallest ending trimpoint into the output.</p> <p>The –tr option will “reset” the trimpoints, using only the trimpoint information occurring in this file.</p> <p>Note: The use of this option does not limit the reads that will be written into the output file. To only output the reads whose accessions appear in the trim file, the –i option must also be used, with the same file as its argument.</p>
<p>-af filename</p>	<p>This option specifies a file that contains adjustments to the description line for some or all reads. Each line should contain the adjustments for a single read, and the line must begin with the accession number of the read. The text that follows the accession number (and subsequent whitespace characters) can take one of two forms.</p> <p>(1) If the text to the right of the accession number does not begin with a '>', it is appended to the description line for the read. For example, the line</p> <pre>DJS045A03F template=DJS045A03 dir=F library=DJS045</pre> <p>will append the text “template=DJS045A03 dir=F library=DJS045” to the end of the current description line for read “DJS045A03F”.</p> <p>(2) If the text to the right of the accession number does begins with a '>', it will completely replace the description line for the input read, <i>including changing the accession number for the read</i>. For example, the line <pre>gi tl 00103402131 >DJS045A03F template=DJS045A03...</pre> <p>will change the description line for input read “gi tl 00103402131” to be “>DJS045A03F template=DJS045A03...” in the output FASTA file that is written (and effectively change the accession number of the read for any program which uses the output FASTA file).</p> <p>This feature is useful for translating less descriptive read accession numbers (such as the NCBI Trace Archive accessions) into more descriptive read accession numbers (such as the genome project accession numbers, which encode library, plate-well and direction in the accession numbers).</p> </p>

-ac command	<p>This option specifies a command to be executed for each input read, to determine adjustments to make to the read's description line in the output FASTA file.</p> <p>The <code>fnfile</code> command will execute the command string "<i>command accno 'description'</i>" for each read, where "<i>command</i>" is the value of the <code>-ac</code> option, "<i>accno</i>" is the first non-whitespace string on the input read's description line, and "<i>description</i>" is the input read's full description line (including the <code>accno</code>).</p> <p>The command processes each line of the file individually and writes the result to standard output. The output should be formatted as described above for the "text after the accession" for the <code>-af</code> option and will be processed by the <code>fnfile</code> command the same way. If no line of output is written, the input read's description line will remain unchanged.</p>
[--help]	Displays a usage line and short description of the command.
[--version]	Displays the current software version of the command.
fastafilename or PHD file or SCF file or directory	<p>Path to a FASTA file, PHD file, SCF file or a directory containing FASTA, PHD or SCF files. The reads in any input files are merged directly into the single output FASTA file generated by this command. If a directory is given, all the files it contains are read, and any files in the directory that are FASTA, PHD or SCF are processed and their reads are included in the output.</p> <p>Note: This command is <i>not</i> aware of <code>consed</code>'s <code>phd_dir</code> versioning of reads. If a <code>phd_dir</code> directory is given that contains multiple versions of the PHD file for a single read, all versions will be read and used by the command, and multiple versions of those reads will appear in the output FASTA file. The same holds true if a <code>phd.ball</code> file and individual PHD files are given on the command line.</p>

3.5 sffrescore

The `sffrescore` command can be used to rewrite existing SFF files with the new Phred-like quality scores. It recursively searches through a list of files or directories, identifies all the SFF files, determines which SFF files contain the older quality scores, runs the `sfffile` command (with the `-r` option) to generate a new SFF file with the new quality scores, then overwrites the existing SFF file with the new file. The effect is that the reads, flowgrams and basecalls in the files are left unchanged, but the files now contain new quality scores.

The `sffrescore` command uses the following syntax:

```
sffrescore [-f] (file | directory)...
```

Command	Description
sffrescore	The <code>sffrescore</code> command recursively searches through a list of files or directories, and rescores any SFF file that contains the older quality scores. Any file rescored is rewritten in place with a new version of the SFF file.

Option / Argument	Description
-f	This option forces the rescoring of all SFF files (by default, any SFF file found to already have the new scores are not rewritten).
file or directory...	List of the files and directories to be processed. The command will recursively search through the directories to find all of the SFF files in the directory trees.

4. GS DE NOVO ASSEMBLER AND GS REFERENCE MAPPER APPENDICES

4.1 Options Common to Mapping and Assembly

The options for the CLI assembly and mapping commands and the associated assembler and mapper GUI parameter locations are given in the table below.

Option	Description	runAssembly	runMapping	runProject (Assembly)	runProject (Mapping)	gsAssembler	gsMapper
--help	Flag to display command usage	X	X	X	X	Help button	Help button
--version	Flag to display command version	X	X	X	X	About button	About button
-cdna	Flag for transcriptome projects	X (also newAssembly)	X(also newMapping)			Sequence Type dropdown menu	Sequence Type dropdown menu
-cpu #	Flag to specify number of CPUs to use for the computations. Default of '0' means use all.	X	X	X	X	Parameters: Number of CPUs to use	Parameters: Number of CPUs to use
-minlen #	Flag to set the minimum length of reads used in the computations.	X	X	X	X	Parameters: Minimum read length	Parameters: Minimum read length
-qo	Flag to generate quick output for mapping and assembly. Disables signal distribution computation for calling consensus sequences and can decrease accuracy.	X	X	X	X	Quick output toggle	Quick output toggle
-it #	Flag to specify the maximum number of isotigs in an isogroup. Default is 100 isotigs. Maximum is 10,000.	X		X		Parameters: Isotig Threshold	
-ig #	Flag to specify the maximum number of contigs in an isogroup. Default is 500 contigs.	X		X		Parameters: Isogroup Threshold	
-icc #	Flag to specify the maximum number of contigs in an isotig. Default is 100 contigs. Maximum is 200 and corresponds to the recursion depth during graph traversal.	X		X		Parameters: Isotig count threshold	
-icl #	Flag to specify the minimum length a contig must be to be part of an isotig. Default is 3 bp. Minimum is 3 bp.	X		X		Parameters: Isotig length threshold	

-consed16	Flag to generate output compatible with versions of consed older than V17.0. (-consed is compatible with version 17.0)	X	X	X	X	Ace/Consed: consed16 <u>selection</u>	Ace/Consed: consed16 <u>selection</u>
-fi filename	Flag for include filter file to be specified.	X	X	X	X	Parameters: Include filter file	Parameters: Include filter file
-fe filename	Flag for exclude filter file to be specified.	X	X	X	X	Parameters: Exclude filter file	Parameters: Exclude filter file
-info	Flag to output to 454AlignmentInfo.tsv file.	X	X	X	X	Alignment info: Output small <u>Selection</u>	Alignment info: Output small <u>Selection</u>
-infoall	Flag to output to 454AlignmentInfo.tsv file including 0-coverage positions.	X	X	X	X	Alignment info: Output <u>Selection</u>	Alignment info: Output <u>selection</u>
-noinfo	Flag to suppress output of the 454AlignmentInfo.tsv file.	X	X	X	X	Alignment info: No output <u>Selection</u>	Alignment info: No output <u>selection</u>
-e #	Flag to specify expected depth of data, needed if depth >50, 0 resets to default	X	X	X	X	Parameters: Expected Depth <u>field</u>	Parameters: Expected Depth <u>field</u>
-m	Flag to keep sequence data in memory to speed up cpu time (requires large CPU memory)	X	X	X	X		
-r	Flag to restart the computation			X	X	Parameters: Incremental <i>De Novo</i> assembler analysis <u>toggle</u>	Parameters: Incremental reference mapper analysis <u>toggle</u>
-o dir	Flag to open specified directory	X	X				
-p filedesc	Flag to specify SFF data in filedesc comes from Paired End sequencing	X (also addRun)	X (also addRun)			Treat reads as specified read type: <u>dropdown</u> <u>menu</u>	Treat reads as specified read type: <u>dropdown</u> <u>menu</u>
-mcf filedesc	Flag to specify non-default MID config file	X (also addRun)	X (also addRun)	X	X		

-vs fastafile	Flag to specify a vector screening database FASTA file of read sequences that indicate contaminants. Read is screened if it matches completely, partial matches are not screened.	X	X	X	X	Parameters: Config Files; <u>Screening</u> <u>deatabase</u>	Parameters: Config Files; <u>Screening</u> <u>deatabase</u>
-novs	Flag to turn off a vector screening database used in an earlier part of the computation			X	X	Parameters: Config Files; <u>Screening</u> <u>deatabase</u> (leave blank)	Parameters: Config Files; <u>Screening</u> <u>deatabase</u> (leave blank)
-v fastafile -vt fastafile	Flag to specify a vector trimming database FASTA file for cloning vectors, primers, adaptors and other end sequences	X	X	X	X	Parameters: Config Files; <u>Trimming</u> <u>database</u>	Parameters: Config Files; <u>Trimming</u> <u>database</u>
-nov, -novt	Flag to turn off a vector trimming database FASTA file for cloning vectors, primers, adaptors and other end sequences specified in an earlier part of the computation			X	X	Parameters: Config Files; <u>Trimming</u> <u>database</u> (leave blank)	Parameters: Config Files; <u>Trimming</u> <u>database</u> (leave blank)
-notrim	Flag to turn off default quality and primer trimming of input reads	X	X	X	X		Parameters: Automatic Trimming <u>toggle</u>
-trim	Flag to turn on default quality and primer trimming of input reads			X	X		Parameters: Automatic Trimming <u>toggle</u>
-nor	Flag to turn off the automatic rescore function for read quality scores	X	X	X	X		
-ud	Flag to treat each read separately, no grouping into duplicates	X	X	X	X		
-ss	Flag to set the seed step parameter	X	X	X	X	Parameters: <u>Seed step</u>	Parameters: <u>Seed step</u>
-sl	Flag to set the seed length parameter	X	X	X	X	Parameters: <u>Seed Length</u>	Parameters: <u>Seed Length</u>
-sc	Flag to set the seed count parameter	X	X	X	X	Parameters: <u>Seed Count</u>	Parameters: <u>Seed Count</u>
-ml	Flag to set the minimum overlap length parameter (From the CLI, the value can be either a minimum length in bases or a percentage of read length. In the case of a percentage, simply include "%" immediately following the numeric value)	X	X	X	X	Parameters: <u>Minimum</u> <u>Overlap</u> <u>Length</u>	Parameters: <u>Minimum</u> <u>Overlap</u> <u>Length</u>

-mi	Flag to set the minimum overlap identity parameter	X	X	X	X	Parameters: <u>Minimum Overlap Identity</u>	Parameters: <u>Minimum Overlap Identity</u>
-ais	Flag to set the alignment identity score parameter	X	X	X	X	Parameters: <u>Alignment Identity Score</u>	Parameters: <u>Alignment Identity Score</u>
-nrm	Flag to specify 'no removal' of meta data files which are needed for GUI viewing of data	X	X				
-no	Flag to specify no output	X	X	X	X		
-nobig	Flag to skip output of large files; ACE/consed files, 454PairAlign.txt, 454AlignmentInfo.tsv	X	X	X	X	Pairwise Alignment: None, Ace/Consed: No files <u>Selections</u>	Pairwise Alignment: None, Ace/Consed: No files <u>Selections</u>
-noace	Flag to specify no ace file generation	X	X	X	X	Ace/Consed: No files <u>Selection</u>	Ace/Consed: No files <u>Selection</u>
-ace	Flag to specify single ace file generation	X	X	X	X	Ace/Consed: Single ACE file <u>Selection</u>	Ace/Consed: Single Ace file <u>Selection</u>
-acedir	Flag to generate ace file subdirectory with an ace file for each contig output	X	X	X	X	Ace/Consed: Ace file per contig <u>selection</u>	Ace/Consed: Ace file per reference <u>Selection</u>
-consed	Flag to generate a consed file subdirectory and place all consed necessary file and folders within (consed V17.0)	X	X	X	X	Ace/Consed: Complete Consed Folder <u>Selection</u>	Ace/Consed: Complete Consed Folder <u>Selection</u>
-ar	Flag to output full raw read sequences in ace or consed folder	X	X	X	X	Ace read mode: raw <u>selection</u>	Ace read mode: raw <u>selection</u>
-at	Flag to output only trimmed sequence reads in ace or consed folder	X	X	X	X	Ace read mode: trimmed <u>selection</u>	Ace read mode: trimmed <u>selection</u>
-ad	Flag to use default to output reads in ACE or consed folder			X (trimmed)	X (raw)	Ace read mode: default (trimmed) <u>selection</u>	Ace read mode: default (raw) <u>selection</u>

-pair	Flag to output pairwise overlaps in txt file	X	X	[also -p]	[also -p]	Pairwise Alignment: Simple Selection	Pairwise Alignment: Simple Selection
-pairt	Flag to output pairwise overlaps in tab delim file	X	X	[also -pt]	[also -pt]	Pairwise Alignment: Tabbed selection	Pairwise Alignment: Tabbed selection
-a #	Flag to specify minimum length for contig reported in 454AllContig.fna. Default =100	X	X	X	X	All contig threshold	All contig threshold
-l #	Flag to set the minimum length for contig to appear in 454LargeContigs.fna, default=500	X	X	X	X	Large contig threshold	Large contig threshold

Table 4: Mapping and Assembly options

4.2 Options Specific to Assembly

Option	Description	runAssembly	runProject (Assembly)	gsAssembler
-rip	Flag to output each read in only one contig.	X	X	Reads limited to one contig <u>toggle</u>
-large	Flag to specify large dataset	X		Large or complex genome <u>toggle</u>

Table 5: Options Specific to Assembly

4.3 Options Specific to Mapping

Option	Description	runMapping	runProject (Mapping)	gsMapper
-gref	Flag for genomic reference sequence	X (also for setRef)	X	Reference type <u>selection</u>
-cref	Flag for cDNA reference sequence	X (also for setRef)	X	Reference type <u>selection</u>
-accno filename	Flag to allow reference accession numbers and descriptions to be changed in the generation of the output.	X	X	Parameter: <u>Accno renaming file</u>
-random	Flag to allow GOLDENPATH _random and _hap references to be automatically used.	X		
-ref/-read	Flags to separate reference & read files	X		
-n, -nimblegen	Flag to specify reads should be primer-trimmed using the early NimbleGen Sequence Capture protocol's primer sequence	X	X	Parameters: <u>Nimblegen Sequence Capture toggle</u>
-non, nonimblegen	Flag to turn off NimbleGen mapping mode	X	X	Parameters: <u>Nimblegen Sequence Capture toggle</u>
-annot filedesc	Flag for optional file with gene-coding/region annotation for reference sequence(s) for gene name and protein translation info in Variant GUI tab	X	X	Parameters: <u>Genome Annotation</u>
-noannot	Flag to disable the automatic reading of annotation file	X	X	Parameters: <u>Genome Annotation</u> (leave blank)
-snp filedesc	Flag for optional file with known SNP information for the reference sequence(s) for linking to identified variations on the HCDiffs GUI tab	X	X	Parameters: <u>Known SNP</u>

-nosnp	Flag to disable the automatic reading of known SNP files	X	X	Parameters: <u>Known SNP</u> (leave blank)
-rst	Flag to set the repeat score threshold parameter	X	X	Parameters: <u>Repeat score threshold</u>
-hll	Flag to set the Hit location limit parameter (no longer used in v2.3 algorithm, default set to 1,000,000)	X	X	Parameters: <u>Hit location limit</u>
-hsl	Flag to set the Hit-per-seed limit parameter (new default value of 70, optimized for newer algorithms)	X	X	Parameters: <u>Hit per seed limit</u>
-srv	Flag to specify single read variant output	X	X	Parameters: Single read variant <u>Toggle</u>
-reg accno# or regstringfile or Nimblegen DARFile	Flag to only output specified sequence region of the reference and aligned reads	X	X	Parameters: Config files; Targeted Regions
-noreg	Flag to turn off sequence region based generation of output		X	Parameters: Config files; Targeted Regions (leave blank)

Table 6: Options Specific to Mapping

4.4 Options for the addRun Command

A description of the options for the addRun command is given in the Table below

Option	Description
[-p]	Flag to specify SFF data in filedesc comes from Paired End sequencing. If the “-p” option is not given, auto-detect will be used to determine the type of each Read Data set. Auto-detect will determine the presence or absence of the Paired End linker sequence: if at least 25% of the first 500 reads in the file (or 25% of all the reads if there are fewer than 500 reads in the file) contain the linker, the file is recognized as a Paired End file.
[-lib libname]	This option specifies that the default Paired End library for each of these files should be the string “libname”. It can be used to group together the Paired End reads from different files, so that they are all used together in calculating a mean distance between the halves of each Paired End pair of reads. (By default, each SFF file is treated as a separate library, and a separate mean Paired End distance is calculated for each.)
[-mcf filename]	This option specifies an MID configuration file (other than the default file) to be used by the assembly for decoding the multiplex information appearing on the command line.



External files:

- GS Read Data files (SFF files) are not copied to the project directory. A symbolic link to the file is created and placed in the project's sff directory. Thus, if the original file is moved or erased from the file system, the project will not operate correctly. This applies whether the files are added via the GUI or the command line.
- FASTA files are not copied to the project directory. The file path is recorded in the project setup. Thus, if the original file is moved or erased from the file system, the project will not operate correctly. This applies to all instances of FASTA files that can be added to an Assembly or Mapping project, whether done via the GUI or the command line:
 - FASTA reads files, including Sanger reads
 - Trimming database files
 - Screening database filesUnlike with the GS Read Data files, no symbolic link to the file is created for FASTA files.

4.5 Mutually Exclusive Options for Assembly and Mapping Commands

Mutually exclusive options of the assembly and mapping CLI commands are:

- -no or -nobig **with** -ace, -acedir, -consed, -pair, -pairt
- -ace, -acedir, -noace, -consed
- -ar, -at
- -a #, -g
- -annot, -noannot
- -nimblegen (-n), -nonimblegen (-non)
- -pair (-p), -pairt (-pt)
- -reg #, -noreg
- -snp, -nosnp
- -trim, -notrim
- -v, -nov
- -vs, -novs
- -vt, -novt

4.6 Multiplex Identifiers

4.6.1 The Use of Multiplex Identifiers in the Data Analysis (MIDs)

The path for any filename or data directory name can be pre-pended with a multiplexing information string, separated from the filename/directory-string by an "@" sign. Multiplexing information strings should be used when multiple samples have been prepared using different initial "tag" sequences (such as those provided by the GS Multiplex Identifiers (MID) Kits), then mixed together for sequencing. The software uses these tag sequences to segregate the reads from each sample, by matching the initial bases of the reads to the known tag sequences used

in the preparation of the libraries. If a multiplex string is given, sffile will automatically match the 5' bases of each read against the multiplex sequences, and will only use the reads that match the specified sequences (and reset the trim point of those reads so that the multiplex sequence is trimmed off the output read sequence).

Three examples of multiplex information strings are the following:

- mid2@myreads.sff
- GSMIDs:mid1,mid4,mid8@/home/xxx/morereads.sff
- aattctc/1@1-3,4,6-8:D_2005_01_01_01_01_testuser_runImagePipe

The first example gives “myreads.sff” as the input SFF file and tells sffile to only use reads whose initial 5' sequence matches the “mid2” MID (as listed in the MID configuration file). The second example tells sffile to use the file “/home/xxx/morereads.sff” and filter that file, using only the reads starting with the mid1, mid4 and mid8 sequences, as defined in the “GSMIDs” MID set. The third example tells sffile to read regions 1, 2, 3, 4, 6, 7 and 8 of the “D_2005_...” sequencing Run, but only use the reads whose 5' end matches the DNA sequence “AATTCTC” with 1 error or less.

The format of the multiplexing information string is the following:

```
[setname:]mid[/#](,mid[/#]... )@
```

where

- the “setname” is an optional name of an MID set found in the MID configuration file and can be given in uppercase or lowercase letters (the matching is case-insensitive)
- each “mid” is either an MID name found in the configuration file or a DNA sequence
- each “#” is an optional allowed number of errors

No spaces are allowed in the multiplexing information string, and no colons, slashes or “@” signs are allowed in the MID set names or MID names.

A default MID configuration file can be found in /usr/local/rig/config/MIDConfig.parse. This file is read by sffile and used to match MID set names and MID names with their multiplexing information. Users can edit this file to add their own MID sets (following the format and syntax described in the file), or can copy this file locally and create their own separate MID configuration file and then use the ‘-mcf filedesc’ option to specify the newly created MID configuration file.

The list of MID schemes provided in the Scheme drop down menu is read from the MIDConfig.parse file. For pre-existing MID schemes, the names and sequences of the MIDs, and the number of errors allowed, cannot be changed in the Select GS Read Data window. To modify these values, or to add new user-specified MID schemes, edit the MIDConfig.parse file using any text editor. The standard location of this file is /opt/454/config/MIDConfig.parse; more generally, if the software is installed to the /install/directory folder, the configuration file will be found at /install/directory/config/MIDConfig.parse. See section 4.6.2 to view the default MIDConfig.parse file and for a description of how to edit it.

You can also create new MIDs directly in the Select GS Read Data window, by selecting “Custom Multiplexing” from the Scheme drop-down menu. In this case, a table will appear and

new MID names, sequences, and error limits can be typed into the table (see Figure 84). However, these names, sequences and error limits will only apply to the current project.

Name	Sequence	Number of Errors Allowed
cust1	TTGTTGAA	0
cust2	CCACTTGC	1
cust3	AA	0
		0
		0
		0
		0
		0
		0

Figure 84: GS Read Import Dialog Rapid Library MID Support

There is no formal limit to the number or length of custom MIDs that one can define in an Assembly or Mapping project. However, efficient demultiplexing requires that the MID sequences be divergent enough that the software will be able to differentiate them from one another (within the “number of errors allowed” specified). The Genome Sequencer FLX System software comes with 2 MID schemes, GSMIDs and RLMIDs. 14 MIDs (of which MIDs 1 to 12 match the “MID Adaptors” available in kit form) are listed for GSMIDs. 12 MIDs are available as RLMIDs.

4.6.2 The MIDConfig.parse File

```

/*
**
** MIDConfig.parse
**
** This file contains the multiplex sequences used by the Genome Sequence
** MID library kits, and may contain user-defined sets of multiplex
** identifiers. This file is used by the post-run applications to access
** the defined MID sets.
**
** To use your own MID set, you can either copy this file to a local
** directory, add or edit your own sets (see below), then use the
** "-mcf" option of the mapper and assembler to specify the MID
** configuration file. Or, you can edit and save this file, to have
** your MID sets accessed by default by the post-run applications.
**
** To create a new MID set, copy the examples at the end of the file into
** the top section, then edit the text as follows:
**
** * The name of the MID set should begin the group (appear above the

```

```

**      open brace '{'
**
**      * Each line in the MID set should contain three or four values after
**      the equals sign:
**          - A name for the specific MID sequence
**          - The DNA sequence of the MID
**          - The number of errors allowed in matching to the sequence
**          - An optional 3' trimming sequence that may appear at the
**            end of reads beginning with the MID
**              * This sequence is used only for trimming, not
**                for additional multiplexing
**
**      * The syntax of the line must be preserved (the "mid = " beginning,
**        the semi-colon at the end of the line, the open and close braces
**        for the set.
**
**      Note:  The names below use a combination of uppercase and lowercase
**             characters, but all matching to the names is insensitive to
**             case (so, for example "gsmids" will match the MID set below).
**
*****
/*
** User-defined MID sets.
*/
/*
** IMPORTANT:  DO NOT EDIT BELOW THIS LINE.
**
** Genome Sequencer MID sets.
*/

GSMIDs
{
    mid = "MID1", "ACGAGTGCCT", 2;
    mid = "MID2", "ACGCTCGACA", 2;
    mid = "MID3", "AGACGCACTC", 2;
    mid = "MID4", "AGCACTGTAG", 2;
    mid = "MID5", "ATCAGACACG", 2;
    mid = "MID6", "ATATCGCGAG", 2;
    mid = "MID7", "CGTGTCTCTA", 2;
    mid = "MID8", "CTCGCGTGTC", 2;
    mid = "MID9", "TAGTATCAGC", 2;
    mid = "MID10", "TCTCTATGCG", 2;
    mid = "MID11", "TGATACGTCT", 2;
    mid = "MID12", "TACTGAGCTA", 2;
    mid = "MID13", "CATAGTAGTG", 2;
    mid = "MID14", "CGAGAGATAC", 2;
}

RLMIDs
{
    mid = "RL1", "ACACGACGACT", 2, "AGTCGTGGTGT";
    mid = "RL2", "ACACGTAGTAT", 2, "ATACTAGGTGT";
    mid = "RL3", "ACACTACTCGT", 2, "ACGAGTGGTGT";
    mid = "RL4", "ACGACACGTAT", 2, "ATACGTGGCGT";
    mid = "RL5", "ACGAGTAGACT", 2, "AGTCTACGCGT";
    mid = "RL6", "ACGCGTCTAGT", 2, "ACTAGAGGCGT";
    mid = "RL7", "ACGTACACACT", 2, "AGTGTGTGCGT";
    mid = "RL8", "ACGTACTGTGT", 2, "ACACAGTGCGT";
    mid = "RL9", "ACGTAGATCGT", 2, "ACGATCTGCGT";
    mid = "RL10", "ACTACGTCTCT", 2, "AGAGACGGAGT";
    mid = "RL11", "ACTATACGAGT", 2, "ACTCGTAGAGT";
    mid = "RL12", "ACTCGGTCGT", 2, "ACGACGGGAGT";
}

```

}

4.7 Paired End Libraries in the Genome Sequencer FLX System

4.7.1 Paired End Read Naming Convention

In traditional Sanger Paired End sequencing, the ends of a fragment are sequenced using Forward and Reverse Primers (see Figure 85). In the Genome Sequencer FLX System, by contrast, Paired End sequencing is performed by first circularizing the fragment using a linker to join the 2 ends of the fragment of sample DNA. The circularized fragment is then nebulized, and the nebulized fragments containing the linker are then sequenced (see Figure 86).

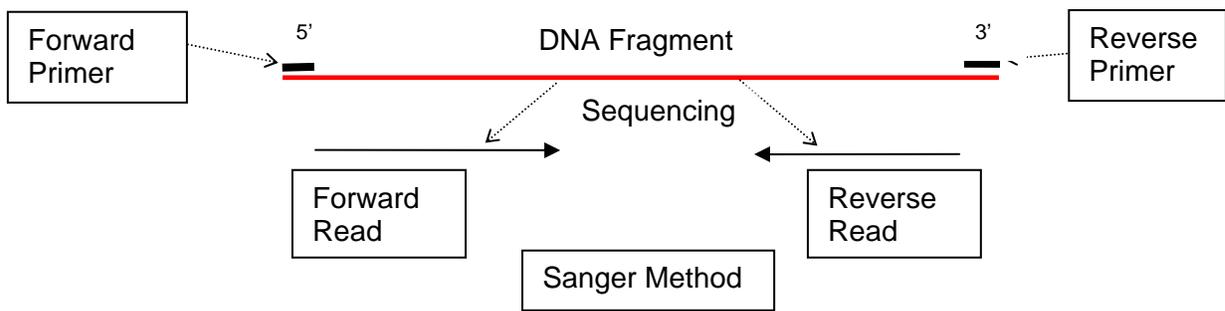


Figure 85: Sanger Paired End reads

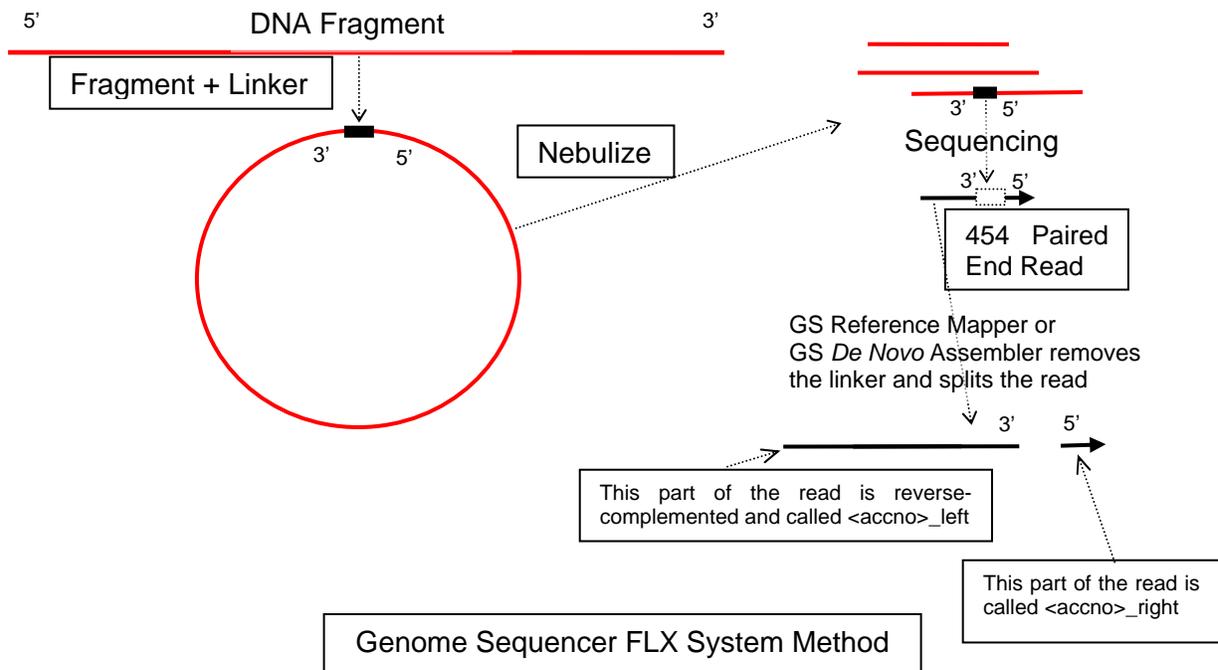


Figure 86: Genome Sequencer FLX System Paired End reads

The GS *De Novo* Assembler or GS Reference Mapper application takes Paired End reads that have been basecalled, removes the linker sequence, and creates two “half” reads for computation and reporting purposes. The half of the read corresponding to the reverse read in the Sanger method (coming from the 3'-end of the original linear fragment) is reverse-complemented to match the orientation one would obtain by extending the Reverse primer in the Sanger method; this half read is labeled with the original read's accno and given the extension ‘_left’. The half of the read corresponding to the forward read in the Sanger method is left in its original orientation and labeled with the original read's accno and given the extension ‘_right’.

4.7.2 Ends of Paired End Reads Shorter Than 50 bp (Tags)

Halves of Paired End reads shorter than 50 bp are treated separately from longer reads in the mapping and assembly computations. These are referred to by the software as ‘Tags’. The minimum length of tags to be used in the computation can be adjusted using “-minlen” command line option or the **Minimum read length** parameter in the GUI (the default is 20 bp, the allowed value range is 15-45).

4.7.3 Paired End Library Span Estimation

Estimates of the distance spanned by Paired End reads in a library are made when at least 8 consistent mate pairs are found that align to the same contig or scaffold. Both halves of a Paired End read must align to the same contig with the expected directionality (the read halves 3' ends point toward each other, after reverse-complementation of the left half). Statistics for the distance between mated pairs are kept for each library. As additional scaffolds are formed, statistics for additional Paired End reads become available and the library span is re-estimated. Paired End reads whose halves are too far away from the mean of the distribution and those whose halves don't have the expected relative orientation are excluded from the span distance calculation. The estimate is less robust when either little Paired End information for a library is available or when very few contigs are significantly longer than the actual library span (in the latter case, the estimated span may be significantly lower than the actual span).

4.7.4 True Pairs and False Pairs

“True Pairs” follow the normal rules for Paired End reads, i.e. they are within the expected library distance from each other, map to the same reference sequence or align within the same assembled contig or scaffold, and map or align in opposite orientations.

False Pairs are Paired End reads whose ends either:

- map to different reference sequences or align in contigs that are in different scaffolds
- map to locations on a single reference sequence or assembled contig with a distance outside the expected library span
- have an unexpected relative orientation

4.8 Trimming and Screening Files

Trimming and screening files use the same format (FASTA) and can even contain the same information. Each entry is simply a unique description line followed by one or more lines of sequence text. Only upper- and lower-case A, C, G, T, and N characters should be used. The N's are expected to correspond to low quality sequence and will NOT be considered a match to any read character except an N. Do not use X's or N's as masking characters. An example follows:

```
>VectorSeq1
TCGATCGTACGATGATCGATCGATCGGATCGATCGAT
GGCTACTTAGGGCTATAAAACCCATG
>VectorSeq2
GGCTAGATTATTAGCCCCTAGATAAACCTTTTTAGCTAG
TCGATCGGATCGATCGATCATG
```

4.9 Accno Renaming File

Scenario 1: Mapping a transcript to a gene name and adding an optional description.

Example of when this would be used: When there is no annotation file but the reference file contains gene identifiers (gene=geneName) in the header line.

```
YAL001C    TFC3  "Largest of six subunits of the RNA polymerase III transcription initiation
factor complex (TFIIIC); part of the TauB domain of TFIIIC that binds DNA at the BoxB promoter
sites of tRNA and similar genes; cooperates with Tfc6p in DNA binding"
```

Explanation: YAL001C is the transcript name. TFC3 is the gene name. The quoted text is the description (in this case used for both the transcript and the gene).

Scenario 2: Adding a transcript level description.

Example of when this would be used: When there is no annotation file or the annotation file does not contain a description for one or more transcripts.

```
YAL011W    YAL011W    "Protein of unknown function, component of the SWR1 complex,
which exchanges histone variant H2AZ (Htz1p) for chromatin-bound histone H2A; required for
formation of nuclear-associated array of smooth endoplasmic reticulum known as karmellae"
```

Explanation: YAL011W is the transcript name. The quoted text is the transcript description. In this case we do not want or need to map the transcript name to a gene name.

Scenario 3: Adding a gene level description.

Example of when this would be used: When there is no annotation file or the annotation file does not contain a description for one or more genes.

```
CYS3    CYS3  "Cystathionine gamma-lyase, catalyzes one of the two reactions involved in the
transsulfuration pathway that yields cysteine from homocysteine with the intermediary formation
of cystathionine"
```

Explanation: CYS3 is the gene name. The quoted text is the gene description. In this case we do not want or need to rename the gene.

Scenario 4: Renaming a gene.

Example of when this would be used: A cDNA Assembly is the reference and there is external data available to map isogroup names to genes.

```
isogroup00001      geneName      geneDescription (optional)
```

Explanation: isogroup00001 is the reference transcript name. geneName is the name of a gene that has been determined to correspond to the reference transcript. The geneDescription is optional.

4.10 Assembling with Reads Obtained Using the Sanger Sequencing Method

Introducing reads obtained using the Sanger sequencing method (Sanger reads) into an assembly or mapping can be more complicated than including 454 Sequencing reads, because Paired End information, trimming information and vector (and other non-sample) information is often not directly associated with the read sequences. To utilize the full capabilities of the GS *De Novo* Assembler or the GS Reference Mapper, the FASTA files of input Sanger reads must be prepared so that they provide the software with the Paired End, trimming and non-sample information. This section describes how the data analysis software applications may use Sanger read data FASTA files.



External files: FASTA files are not actually copied to the project directory. Rather, the file path is simply recorded in the project setup. Consequently, if the original file is moved or erased from the file system, the project will not operate correctly. This applies to all instances of FASTA files that can be added to an Assembly or a Mapping project, whether done via the GUI or the command line:

- FASTA reads files, including Sanger reads
- Trimming database files
- Screening database files
- Reference sequence files (Mapper only)

Unlike with the GS Read Data files, no symbolic link to the file is created for FASTA files.

4.10.1 Simple Sanger Reads

In the simplest case, the data analysis software applications can take FASTA files and assemble or map the sequences they contain directly, without any preparation. In this case, it treats all the reads present in the FASTA files as single ended, shotgun reads, and assumes that the sequence has been trimmed for quality and for vector, primer, adapter, linker, etc. sequences (all non-sample sequence). The GS *De Novo* Assembler or the GS Reference Mapper will use all the sequence data present in the file, and produce or map contigs based on those sequences.

4.10.2 Sanger Reads with available Quality Scores

If there are quality score files associated with the FASTA files, the data analysis software applications will automatically read the quality scores for the reads, and use them in the assembly or mapping and consensus calling. The GS *De Novo* Assembler or the GS Reference Mapper will look for a file that begins with the same file name or file prefix as the FASTA file but ending with “.qual”.

4.10.3 Sanger Read Annotations

Additional information about the Sanger reads can be specified using “*name=value*” annotation strings on the description line of each sequence in the FASTA file. The data analysis software applications looks for and uses the following annotation strings:

1. **template** – the Paired End template string for this read (Paired End reads are matched by having the same template string)
2. **dir** – values “F”, “R”, “fwd” or “rev” giving the direction of the Paired End read
3. **library** – the name of the library that generated this Paired End read (all Paired End reads are grouped by library name for the determination of expected pair distance)
4. **trim** – the trimmed region of the sequence, given as “#-#”
5. **scf** – the path or “command string” to use to access the SCF file for the read
6. **phd** – the path or “command string” to use to access the PHD file for the read

For example, the description line:

```
>DJS045A03F template=DJS054A03 dir=F library=DJS045 trim=12-543
```

tells the data analysis software that this read (accession number “DJS045A03F”) is a Paired End read whose template is “DJS045A03”, is a forward read from library “DJS045”, and bases 12-543 should be used in the assembly.

The data analysis software looks for the six “*name=*” strings on the description line, and then takes the text from the “=” to the next whitespace character as the “*value*” of the annotation string. So, other text besides annotation strings can appear on the description line, and no whitespace may appear in the value of an annotation string.

4.10.3.1 The “template”, “dir”, and “library” Annotations

The first three annotation strings, template, dir and library, are used to specify the Paired End information for a read; only reads with template, dir and library annotations will be treated as Paired End reads. The data analysis software performs exact string matching of the template and library annotations to determine which reads come from the same template or library. Multiple reads may have the same template, dir and library information: they all will be assembled or mapped but only the best aligned sequence (by aligned length or number of differences from the consensus) will be used further. There is no requirement that the accession number for the read encode the Paired End information, and the data analysis software will not try to “parse” the accession number for any information. However, if you use a naming convention for encoding the Paired End information in the accession numbers for reads, the `fnfile` command can be very useful for translating that encoded information into the annotation

strings that the data analysis software can then read. (See Section 3.4 for a description of this use.)

4.10.3.2 The “trim” Annotation

The trim annotation describes the region of the sequence that the data analysis software should use for assembly or mapping (with the assumption that the rest of the sequence is non-sample or low quality). For a read 800 bases in length and a trim annotation string “trim=12-543”, for example, the assembler will ignore bases 1-11 and 544-800, and only use bases 12-543 in the assembly or mapping. If either number in the trim annotation string is 0, the beginning or end of the read will be treated as the trimming point (*i.e.*, for the same example read as above, “trim=12-0” would tell the assembler to use bases 12-800 of the read in the assembly).

This trim annotation should combine the results of all of the sources of trimming that may occur (low quality, vector, primer, adapter, linker, *etc.*), so that the data analysis software is given just the sequence region that represents the bases to be included in the assembly or mapping. The “-v” option can be used with the runAssembly, runMapping and runProject commands to specify a FASTA file containing sequences to be trimmed: each read will then be screened against this database, and the ends of reads that match the sequences included will be trimmed off.

4.10.3.3 The “scf” and “phd” Annotations

The scf and phd annotations provide a link back to the “trace” information for the read, and are only used when generating the full consed directory structure (when the -consed option is used). The value string for scf or phd annotations can take one of two forms, either as a simple path to the SCF or PHD file, or as a “command string” to be executed to access the SCF or PHD file contents. If the value string does not begin with “cmd:”, it is treated as a path. If the value string begins with “cmd:”, it should have the format “cmd:*commandline*”, where any whitespace in the *commandline* portion of the command should be replaced with colons.

Any command specified in the scf or phd annotation strings must, when executed, write the PHD or SCF contents to standard output. The command’s standard output is read, and those bytes are written or processed as needed, by the assembler/mapper and/or the sff2scf command (the programs in the current software that require access to the PHD or SCF contents).

4.10.4 Recommended Method for Including Sanger Reads

The best method for including Sanger reads into an assembly is to first run phred with vector screening (or an equivalent basecaller that generates a FASTA file with vector sequence marked as ‘X’ or ‘N’, plus a quality score file). The “.fasta.screen” and “fasta.screen.qual” files produced by phred contain the screening and trimming information needed by the GS *De Novo* Assembler or the GS Reference Mapper (and it is setup to properly process those reads and the marking and quality information).

There are two ways to include the pairing information for the reads, each of which requires some Perl scripting, because the Paired End information is typically encoded in the accession numbers of the reads. The simplest way is to write a Perl script like the one below. This script is designed to:

- handle a paired read naming convention where there is only one Paired End library;
- where the forward and reverse reads have suffixes ".f1" or ".r1"; and
- where the Paired End reads can be distinguished from follow-up finishing reads by having only alphanumeric characters before the ".f1" or ".r1"):

```
#!/usr/bin/perl
if ($ARGV[0] =~ /^(\\w*)\\.([rf])1$/) {
    print "template=$1 dir=$2 library=pairlib\\n";
}
```

where "\$ARGV[0]" will be the accession number for the read, when used in the procedure below.

The regular expression in this script states "`^(\\w*)\\.([rf])1$`" matches a word of any length, "`\\w*`", followed by a period, "`\\.`", followed by either an '`r`' or an '`f`', followed by a '`1`'. The parentheses are there to then extract the characters that matched the word and matched the '`r`' or '`f`' and place them into the "`$1`" and "`$2`" parameters (so that they can be used in the print statement on the next line). For a local naming convention, customize the Perl regular expression to extract out the template, library and forward/reverse direction strings from the accession, and output them appropriately (see Section 4.10.3) for a description of this output.

If this script is named "accnoPair", then the following procedure can be used to incorporate Sanger reads:

1. Run phred with vector screening to produce the "fasta.screen" file (plus fasta.screen.qual).
2. Run the command "fnafile -o sanger.fna -ac accnoPair fasta.screen" to attach the Paired End information to each read, where fnafile calls the accnoPair program for each read, then adds the output of that command to the read's description line.
3. Give sanger.fna to the assembler or mapper, where it will read and use the pairing information.

The one drawback to this method is that the execution of the Perl script for each read causes the fnafile program to take a couple of minutes to convert 15,000-20,000 reads. An advanced script and slightly different procedure that runs the attachments faster can be used:

If the following Perl script is generated (called "detPairs"):

```
#!/usr/bin/perl
die "Usage: detPairs screenfile\\n" unless defined($ARGV[0]);
open(FILE, $ARGV[0]) or die "Error: Unable to open file:
$ARGV[0]\\n";
while (<FILE>) {
    if (/^\\>(\\S*)/) {
        my $accno = $1;
        if ($accno =~ /^(\\w*)\\.([rf])1$/) {
            print "$accno template=$1 dir=$2 library=pairlib\\n";
        }
    }
}
```

then the following procedure can be used to incorporate Sanger reads:

1. Run phred with vector screening to get the fasta.screen file.
2. Run the command "detPairs fasta.screen > sanger.acc" to get a file with the pairing information.
3. Run the command "fnafile -o sanger.fna -af sanger.acc fasta.screen".
4. Give the sanger.fna file to the assembler or mapper, where it will read and use the pairing information.

For both of these scripts, see Section 3.4 for a description of the command line structure and arguments of fnafile.

4.11 Using the Flowgrams Tab

The term "flowgram" is defined in the Glossary and a full description of flowgram views is given in the section on the GS Run Browser, in Part B of this reference Manual.

The flowgram view as displayed on the **Flowgram tab** of the GS *De Novo* Assembler and the GS Reference Mapper (Figure 87) is most similar to the tri-flowgram view described in detail in the GS Run Browser section in that it shows 3 plots:

- The top plot is an idealized flowgram for either:
 - the consensus sequence of the contig containing the selected read (when using the assembler) or,
 - the selected reference sequence (when using the mapper).The top flowgram may include insertions of flow cycle shifts as needed to maintain the alignment to the selected read's actual flowgram.
- The center plot is the aligned flowgram for the selected read, including the insertions of any flow cycle shifts that may be needed to maintain the alignment to the idealized consensus flowgram.
- The lower plot is the flow-by-flow difference between the two upper plots.

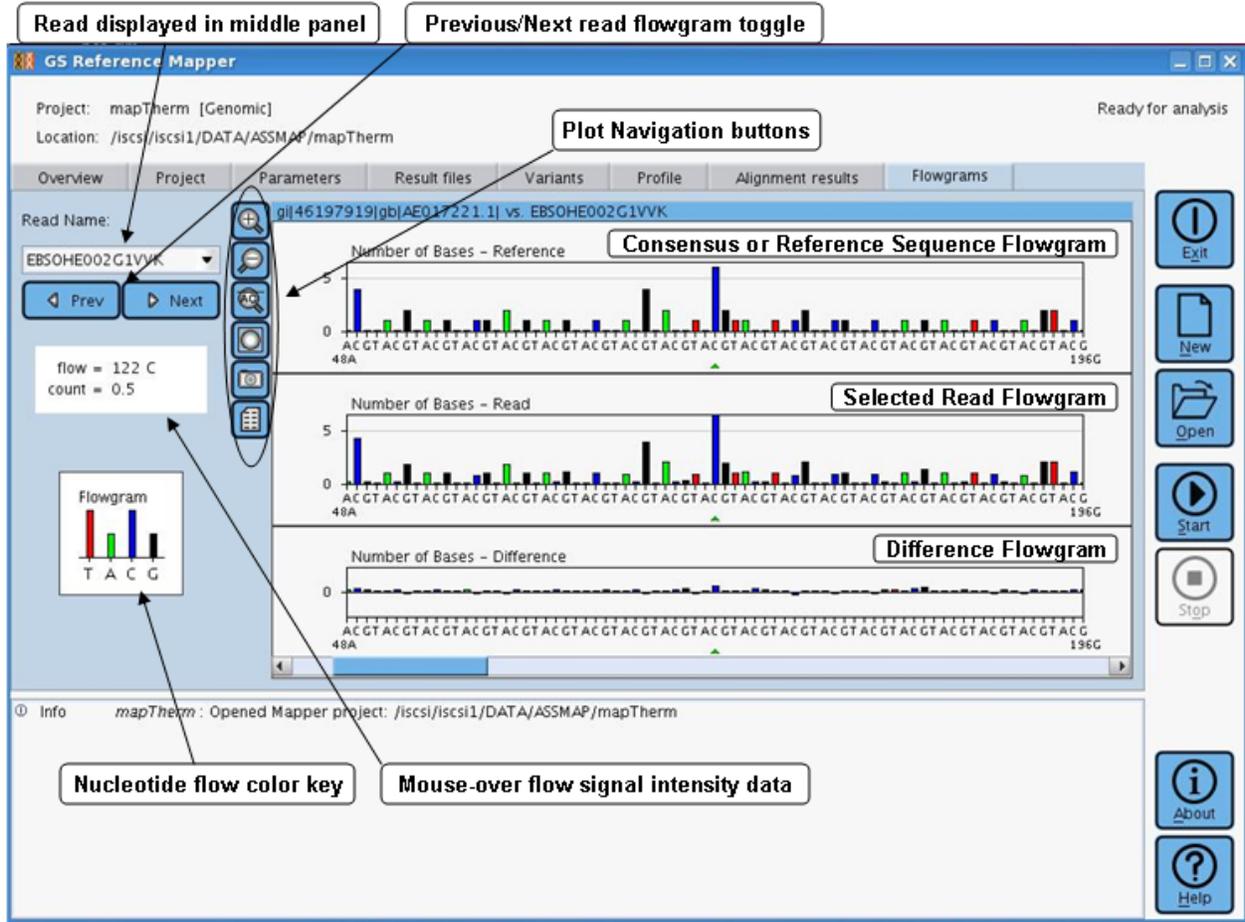


Figure 87: gsMapper Flowgrams Tab

A small green triangle indicates the flow corresponding to the base of the read on which you right-clicked when launching the Flowgrams tab view.

The Flowgram tab provides various navigation options and other features:

- The top-left corner of the tab provides two controls that allow the user to change the display to a different read:
 - A pull-down menu allows you to select from a list of all the reads in the contig to which the read currently displayed belongs (Figure 87), and which span the base position of the alignment that was used when initially launching the Flowgrams tab. That base position may be quickly updated by navigating back to the Alignment results tab and selecting a new alignment column (nucleotide in the sequence) by simply left-clicking in that column. Then switch back to the Flowgrams tab (by clicking on the tab). The drop down menu will be updated to include only those reads that have flowgrams and also which intersect with the new column of interest. The read previously displayed in the flowgram view will remain displayed (unless the new column of interest is not spanned by that read), but the green triangle will be updated to point to the flow corresponding to the nucleotide in the new column of interest. If the previously displayed read does

not span the new column of interest, then the display will automatically go to the first read, topmost in the displayed alignment, which does span the column.

- The pair of buttons, “Prev” and “Next”; allows you to change the display to the previous or the next read of the contig’s multiple alignment (in the order shown in the Alignment Results tab).
- You can also return to the Alignment Results tab and select a different read, from the alignment to the same or to any other contig.
- The plot navigation buttons appearing to the left of the flowgram have the following functions:



Zoom in – Zoom in by a factor of 1.5. For plots, this button zooms only the y-axis scale (use the **Zoom to labels** and **Freehand zooming** functions described below to zoom the x-axis).



Zoom out – Zoom out by a factor of 1.5. For plots, this will zoom only the y-axis scale and, unlike most zoom operations, this will zoom out past the data limits (to allow the user to get a better perspective of the data).



Zoom to labels – This button zooms the x-axis of the flowgram so that the nucleotide/flow characters can fit below the axis.



Zoom to fit data – Fit means ‘zoom all the way out.’ On plots, scale out to the limits of the data.



Snapshot – Save a snapshot image of the current view to disk. This will open a dialog asking for a location and filename, and then will save a PNG image file at the location specified. The saved image contains only the visible region of the plot.



Spreadsheet – Save an Excel-compatible, tab-delimited text file of the spreadsheet data for this graphic. This will open a dialog asking for the location and filename to save the file. It then saves the data, along with summary information describing where the data came from. The data for all three plot subsections is saved to one file, with white space between subsection plot data.

- A mouse tracker area that shows the values for the nucleotide flow under the mouse pointer, when you pause it over the graph area:
 - position of the nucleotide on the consensus or reference sequence,
 - name of the nucleotide, and
 - the “y” value on the plot
 - number of bases added during that flow (idealized per the consensus [assembly] or reference sequence [mapping] or observed in the read), or
 - the value of the “difference”, where bases in the read that are absent in the consensus or reference are positive differences.
- There is also a free hand zoom function that allows you to zoom in to any area of a plot by drawing a box with the left mouse button. This function has the following unique features (not available in the corresponding GS Run Browser freehand zoom in functions) to facilitate the comparison between the three plots:
 - zooming on the consensus/reference or on the read plots will apply to the Y axis of both those plots, and the X-axis of all three plots (including the difference plot).
 - zooming on the difference plot zooms its Y axis alone, but zooms the X axis of all three plots (not shown).

4.12 Project Error Indicators

Descriptive warning and error messages are available on the Parameters and Project tabs. Errors are indicated on the tabs and sub-tabs as red circles with an 'x' both on the tab headings and on the field on the tab producing the error. Placing the mouse over the field indicated reveals a descriptive tooltip of the error including, when available, the default value for the field. A list of the errors to be addressed can also be viewed via a tooltip when mousing over the status message in the upper right hand corner of the main application window. When the errors have been addressed, the status message will read 'Ready for Analysis'. Some examples are shown in the figures below.

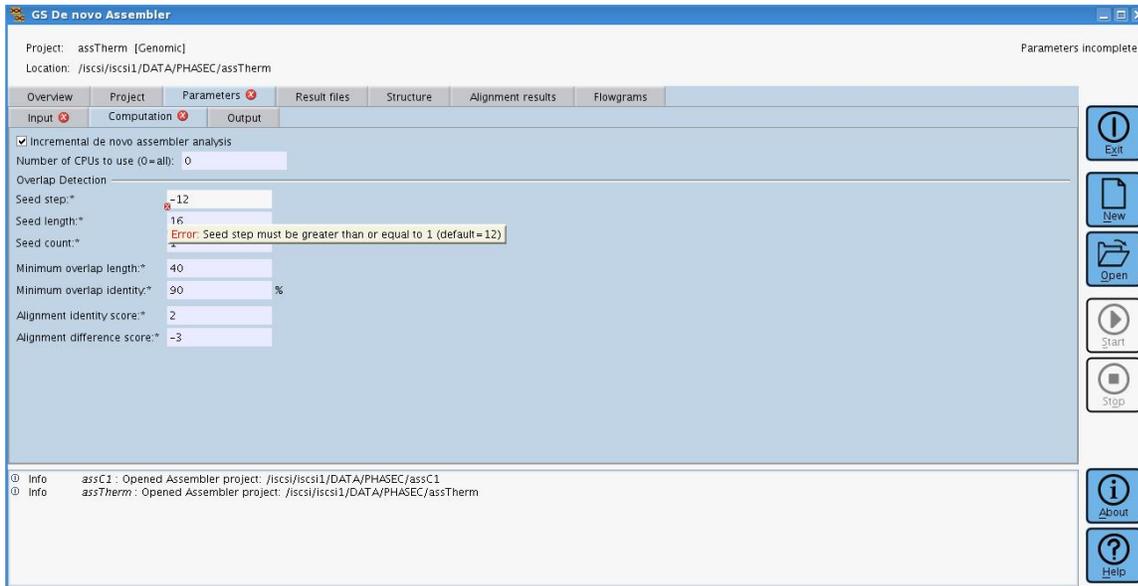


Figure 88: Error messages on the Tab headings and associated tab fields

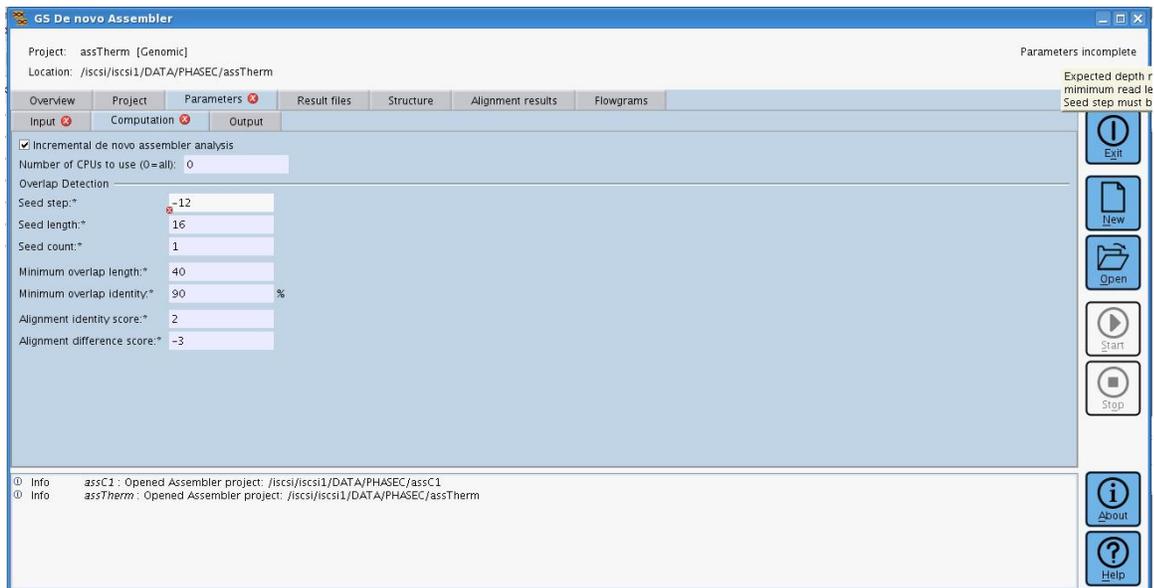


Figure 89: Error messages listed as tooltip on mouse-over of the status message

 When a project is first created and no read files (or read or reference files, for a mapping project) have yet been added, a warning icon on the Project Tab header will be displayed along with the status message 'Not ready for Analysis'.

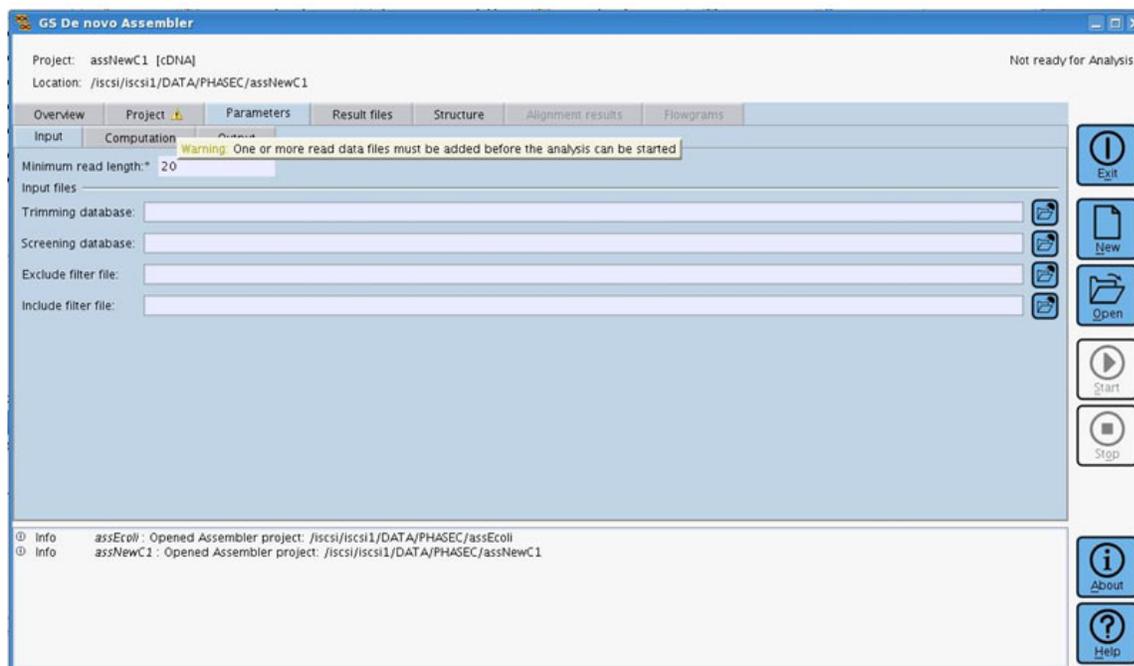


Figure 90: Warning message displayed before read data is added to a project

5. CDNA / TRANSCRIPTOME SEQUENCING APPENDIX

5.1 Introduction to cDNA Sequencing Analysis

cDNA is produced from the RNA (often, mRNA) in a sample. The cDNA is then sequenced resulting in reads that represent the original sample's mRNA. These reads can then be mapped against one or more references to gain information about the representation of individual mRNAs and genes from which the mRNAs are transcribed and their level of variation in the original sample. Alternatively, the reads can be assembled together to discover novel transcripts and splice variants for those mRNAs represented by multiple overlapping reads.

5.2 Transcriptome Assembly Concepts

5.2.1 Definitions

5.2.1.1 *Isogroup*

An isogroup is a collection of contigs containing reads that imply connections between them. A discussion of the assembly process (see Section 1.1) explains how breaks can be introduced into the multiple alignments of overlapping reads, leading to branching structures between them. After attempting to resolve the branching structures, the Transcriptome Assembler groups all contigs whose branches could not be resolved into collections called isogroups. Using rules described in the following section, the assembler traverses the various paths through the contigs in an isogroup to produce the set of isotigs that gets reported. All possible paths through the contigs in an isogroup are traversed unless one or more thresholds is reached (see Section 5.2.2).

5.2.1.2 *Isotig*

An isotig is meant to be analogous to an individual transcript. Different isotigs from a given isogroup can be inferred splice-variants. The reported isotigs are the putative transcripts that can be constructed using overlapping reads provided as input to the assembler. Connections between contigs in an isogroup are represented by sequences (reads) that have alignments diverging consistently towards two or more different contigs (see Figure 91) or by a depth spike (5.2.2.1.1). Traversal from the start contig to the end contig or from the end contig to the start contig should yield the same but reverse-complemented isotig sequence.

While many reads may contain poly-A tails, these tails are trimmed off prior to assembling the reads. Presently, the assembler ignores the fact that poly-A tails existed, so the orientation of reads in the assembly cannot be determined. Because of this lack of directionality, an isotig may be output as the reverse-complement of the biological transcript it represents. Contigs forming an isotig may be thought of as exons. This is not strictly correct, however, since untranslated regions (UTRs) and introns (in the case of primary transcripts) may exist in the reads generated from the sample.

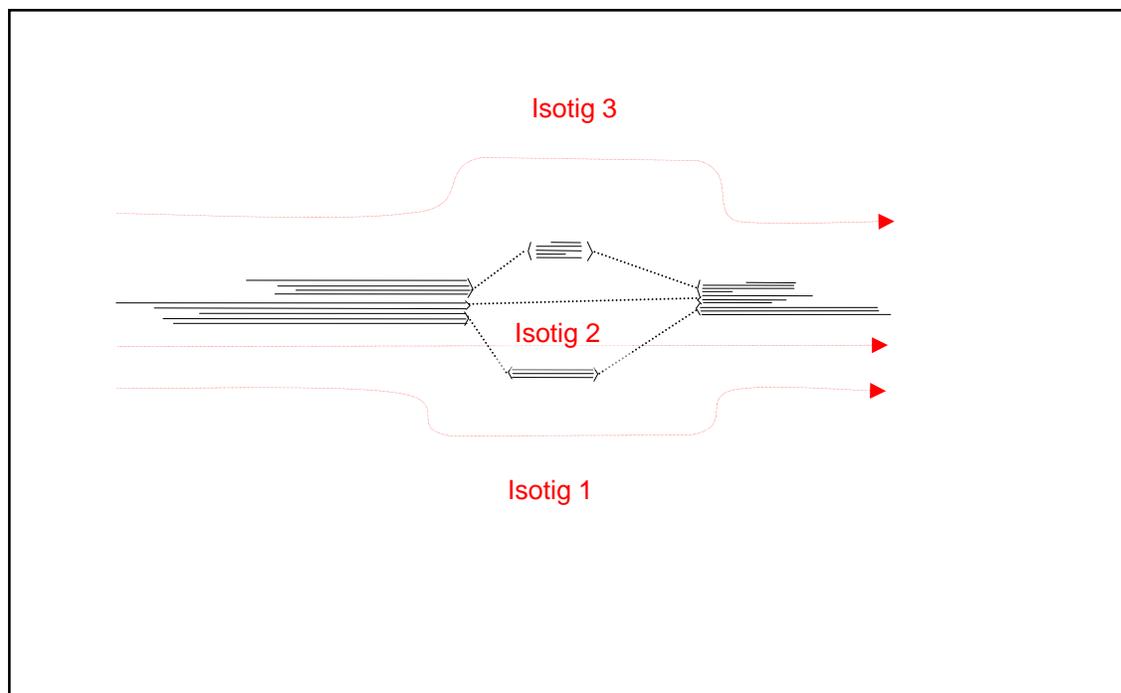


Figure 91: Traversal of contigs in an isogroup

5.2.2 Rules for Path Traversal of Contigs in an Isogroup

5.2.2.1 Path Initiation

Contigs used to initiate the traversal of an individual isotig are:

- contigs lacking reads connecting them to other contigs on one or both ends (if a contig has no reads connecting it to any other contigs, it may become an isotig composed of a single contig). (See Figure 92)
- contigs that show a “spike” in depth of at least 50% between one alignment column and the next

When a “spike” in depth is found while traversing an isotig path through the alignment for a contig (see Section 5.2.2.1.1), the alignment column in which the relative depth change occurs is marked for future use as an initiation point for further isotig traversal. The isotig path up to that point is reported as an isotig, and the original isotig's path continues until it terminates. The marked initiation point is then used as a starting contig and all paths starting from it are reported until they are exhausted.

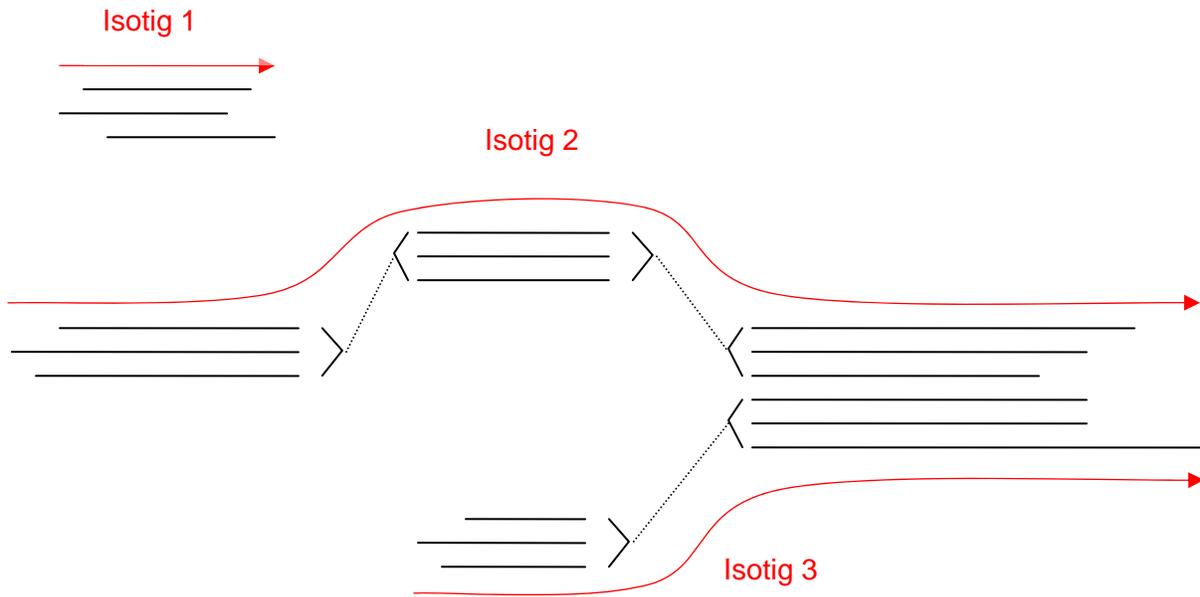


Figure 92: Isotig initiation

5.2.2.1.1 Spike Detection

Identification of a spike along a path occurs when all of the following conditions are met at the end of a contig where a spike is detected:

- The alignment depth is at least 10 reads
- A minimum of 20% of the aligned reads must be in the opposite orientation relative to the more abundant orientation of the aligned reads
- A spike may not occur within 10 bases of an already detected spike
- A change in alignment depth between one alignment column and the next of at least 50% signals the location of a “spike”.

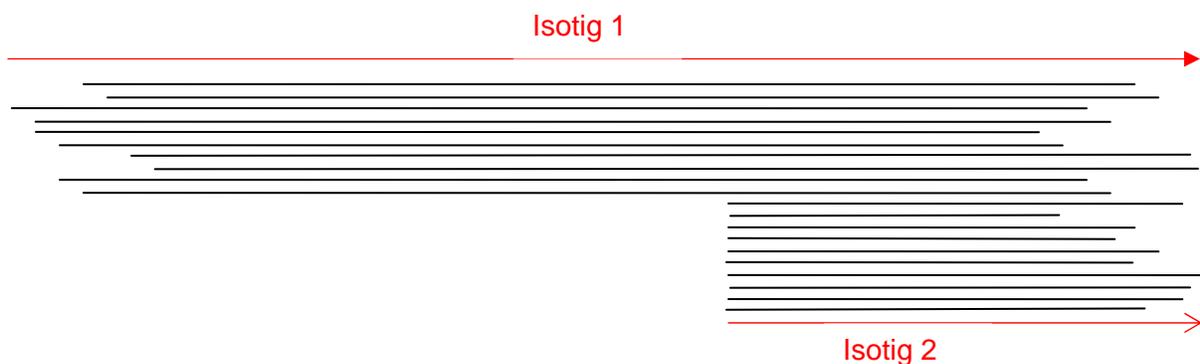


Figure 93: Spike identification

5.2.2.2 *Path Extension*

An isotig path that starts at an initiating contig (call this Contig I) will be extended by following reads that consistently lead to another contig. Once a terminating condition is found on a particular path, the isotig is reported, and then all other paths starting from Contig I are reported until they are exhausted. The process used to explore all possible paths is called “recursion.” Extension along a particular path will continue until a terminating condition is met (see Section 5.2.2.3).

5.2.2.3 *Path Termination*

The end of an isotig is found when any of the following conditions occur:

- No reads are found that extend from the contig currently at the end of the isotig path
- The number of reads connecting two contigs is less than 5% of the alignment depth of either
- The **Isotig Contig Count Threshold** is reached. In this case, the further traversal of a particular isotig in an isogroup will be stopped.
- A contig is reached whose length is below the **Isotig Contig Length Threshold**. If a contig is reached with a length shorter than this threshold, the further traversal of a particular isotig in an isogroup will be stopped. The contig shorter than the icl threshold will be marked as such and reported in the output files.
- A cyclic path is encountered. Recursive path traversal will stop if cyclic structures are detected, i.e. revisiting one contig which has already been included in an earlier part of the isotig being traversed. Such cyclic structures will be marked in the output files by assigning cyclic status for the first contig detected.

5.2.2.4 *Other Rules that Prevent Path Traversal*

- If the number of contigs in an isogroup exceeds the **Isogroup Threshold**, the paths in the isogroup will not be traversed to report isotigs.
- When the number of isotigs in an isogroup is found to exceed the **Isotig Threshold**, further traversal of isotig paths for the group stops.

5.3 Transcriptome Mapping Concepts

5.3.1 Mapping cDNA Reads to a Transcriptomic Reference

Reads derived from cDNA samples can be mapped to a collection of reference sequences representing the set of transcripts that might be found in the sample. Often this collection of reference sequences is a list of known or putative transcripts (such as refMrna.fa from UCSC).

Some of the applications that can be facilitated by such a mapping include the detection of novel SNPs within known transcripts and the profiling of expression levels for known transcripts and genes.

Many of the reads from cDNA sample may map uniquely to only one transcript. Uniquely mapped reads can be used for SNP detection and for profiling transcript levels. However, because some exons may be shared by many transcripts (e.g. splice-variants from a single gene), reads covering such exons may map equally well to many transcripts. In such cases, it may not be possible to assign the reads to any single transcript reference. Nonetheless, when mapping to transcriptomic references, it is often possible to determine the number of reads that map to a single gene even when reads cannot be mapped uniquely to a single transcript produced by the gene.

5.3.2 Mapping cDNA Reads to a Genomic Reference

When mapping reads from cDNA samples to genomic reference sequences, it is often possible to detect splice variants that might not be apparent when mapping to a transcriptomic reference. Breaks in the alignments of individual reads against a region of the genome often indicate exon boundaries. Also, while one may be able to map only parts of reads derived from novel splice variants to a set of known transcripts, mapping against the genomic reference allows the identification of genomic regions not previously known to exist as part of the transcriptome. Mapping to a genomic reference only allows per-gene coverage statistics to be reported while mapping to a transcriptomic reference can facilitate the profiling of individual transcripts.

Published by

454 Life Sciences Corp.
A Roche Company
Brandford, CT 06405
USA

© 2009 454 Life Sciences Corp.
All rights reserved.

**For Life Science Research Only.
Not for Use in Diagnostic Procedures**

454, 454 LIFE SCIENCES, 454 SEQUENCING, GS FLX, GS FLX TITANIUM, emPCR,
PICOTITERPLATE, and PTP are trademarks of Roche.

Other brands or product names are trademarks of their respective holders.

①1009