

Subset of complete manual

CLC Genomics Workbench

User manual

Manual for

CLC Genomics Workbench 5.0

Windows, Mac OS X and Linux

February 23, 2012

This software is for research purposes only.

CLC bio
Finlandsgade 10-12
DK-8200 Aarhus N
Denmark



Contents

1 Introduction to *CLC Genomics Workbench* **5**

2 High-throughput sequencing **6**

2.1 Import high-throughput sequencing data 8
2.2 Multiplexing 26
2.3 Trim sequences 36
2.4 *De novo* assembly 46
2.5 Map reads to reference 60
2.6 Mapping reports 69
2.7 Mapping table 75
2.8 Color space 76
2.9 Interpreting genome-scale mappings 81
2.10 Merge mapping results 92
2.11 SNP detection 92
2.12 DIP detection 101
2.13 ChIP sequencing 107
2.14 RNA-Seq analysis 114
2.15 Expression profiling by tags 128
2.16 Small RNA analysis 139

3 Expression analysis **156**

3.1 Experimental design 157
3.2 Transformation and normalization 169
3.3 Quality control 174
3.4 Statistical analysis - identifying differential expression 186

3.5 Feature clustering 194
3.6 Annotation tests 200
3.7 General plots 207

Bibliography **213**

I Index **216**

Chapter 1

Introduction to *CLC Genomics Workbench*

This manual is a subset of the complete user manual for *CLC Genomics Workbench*. It only contains the sections that are special for Next Generation Sequencing and expression analysis. For the complete user manual, see http://www.clcbio.com/usermanuals.

You will see some missing references, indicated by two question marks **??**. They refer to chapters in the complete user manual.

Chapter 2

High-throughput sequencing

Contents

2.1 Import high-throughput sequencing data 8
2.1.1 454 from Roche Applied Science 9
2.1.2 Illumina Genome Analyzer from Illumina 10
2.1.3 SOLiD from Life Technologies 15
2.1.4 Fasta format 18
2.1.5 Sanger sequencing data 19
2.1.6 Ion Torrent PGM from Life Technologies 20
2.1.7 Complete Genomics 21
2.1.8 General notes on handling paired data 22
2.1.9 SAM and BAM mapping files 22
2.1.10 Tabular mapping files 24
2.2 Multiplexing 26
2.2.1 Sort sequences by name 27
2.2.2 Process tagged sequences 29
2.3 Trim sequences 36
2.3.1 Quality trimming 36
2.3.2 Adapter trimming 38
2.3.3 Length trimming 44
2.3.4 Trim output 44
2.4 *De novo* assembly 46
2.4.1 How it works 47
2.4.2 Resolve repeats using reads 50
2.4.3 Optimization of the graph using paired reads 52
2.4.4 Bubble resolution 54
2.4.5 Converting the graph to contig sequences 55
2.4.6 Summary 56
2.4.7 Randomness in the results 56
2.4.8 SOLiD data support in de novo assembly 56
2.4.9 De novo assembly parameters 57
2.4.10 De novo assembly report 58

2.5 Map reads to reference 60
2.5.1 Starting the read mapping 60
2.5.2 Including or excluding regions (masking) 60
2.5.3 Mapping parameters 61
2.5.4 General mapping options 66
2.5.5 Assembly reporting options 67
2.6 Mapping reports 69
2.6.1 Detailed mapping report 69
2.6.2 Summary mapping report 74
2.7 Mapping table 75
2.8 Color space 76
2.8.1 Sequencing 76
2.8.2 Error modes 77
2.8.3 Mapping in color space 78
2.8.4 Viewing color space information 80
2.9 Interpreting genome-scale mappings 81
2.9.1 Getting an overview - zooming and navigating 81
2.9.2 Single reads - coverage and conflicts 82
2.9.3 Interpreting genomic re-arrangements 83
2.9.4 Output from the mapping 89
2.9.5 Extract parts of a mapping 90
2.9.6 Working with multiple contigs from read mappings 91
2.10 Merge mapping results 92
2.11 SNP detection 92
2.11.1 Assessing the quality of the neighborhood bases 92
2.11.2 Significance of variation: is it a SNP? 95
2.11.3 Reporting the SNPs 96
2.11.4 Adjacent SNPs affecting the same codon 100
2.12 DIP detection 101
2.12.1 Experimental support of a DIP 103
2.12.2 Reporting the DIPs 104
2.13 ChIP sequencing 107
2.13.1 Peak finding and false discovery rates 107
2.13.2 Peak refinement 109
2.13.3 Reporting the results 110
2.14 RNA-Seq analysis 114
2.14.1 Defining reference genome and mapping settings 115
2.14.2 Exon identification and discovery 119
2.14.3 RNA-Seq output options 121
2.14.4 Interpreting the RNA-Seq analysis result 124
2.15 Expression profiling by tags 128
2.15.1 Extract and count tags 129
2.15.2 Create virtual tag list 132
2.15.3 Annotate tag experiment 136

2.16 Small RNA analysis 139
2.16.1 Extract and count 140
2.16.2 Downloading miRBase 144
2.16.3 Annotating and merging small RNA samples 145
2.16.4 Working with the small RNA sample 152
2.16.5 Exploring novel miRNAs 155

The so-called Next Generation Sequencing (NGS) technologies encompass a range of technologies generating huge amounts of sequencing data at a very high speed compared to traditional Sanger sequencing. The *CLC Genomics Workbench* lets you import, trim, map, assemble and analyze DNA sequence reads from these high-throughput sequencing machines:

- The 454 FLX System from Roche
- Illumina's Genome Analyzer
- SOLiD system from Applied Biosystems (read mapping is performed in color space, see section 2.8)
- Ion Torrent from Life Technologies

The *CLC Genomics Workbench* supports paired data from all platforms. Knowing the approximate distance between two reads can enable better determination over repeat regions, where assembly of short reads can be difficult, and enhances the possibility of correctly assembling data. It also enables a wide array of new approaches to interpreting the sequencing data.

The first section in this chapter focuses on importing NGS data. These data are different from general data formats accepted by the *CLC Genomics Workbench*, and require more explanation. After the import section, the trimming capability of the *CLC Genomics Workbench*is described. This includes the ability to trim on quality and length, as well as trim on adapters and de-multiplex datasets.

After these sections, we go on to describe the various analysis possibilities available once you have imported your data into the *CLC Genomics Workbench*.

2.1 Import high-throughput sequencing data

This section describes how to import data generated by high-throughput sequencing machines.

Clicking on the button in the top toolbar labelled **NGS Import** will bring up a list of the supported data types as shown in figure 2.1.

Going to:

File | Import High-Throughput Sequencing Data ( )

will bring up the same list of formats.

Select the appropriate format and then fill in the information as explained in the following sections.

Please note that alignments of *Complete Genomics* data can be imported using the SAM/BAM importer, see section 2.1.7 below.

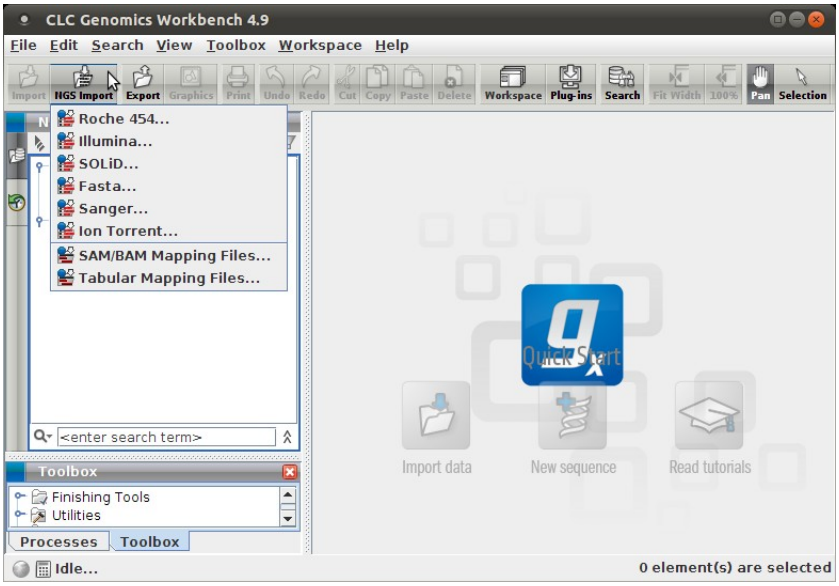


Figure 2.1: *Choosing what kind of data you wish to import.*

2.1.1 454 from Roche Applied Science

Choosing the Roche 454 import will open the dialog shown in figure 2.2.

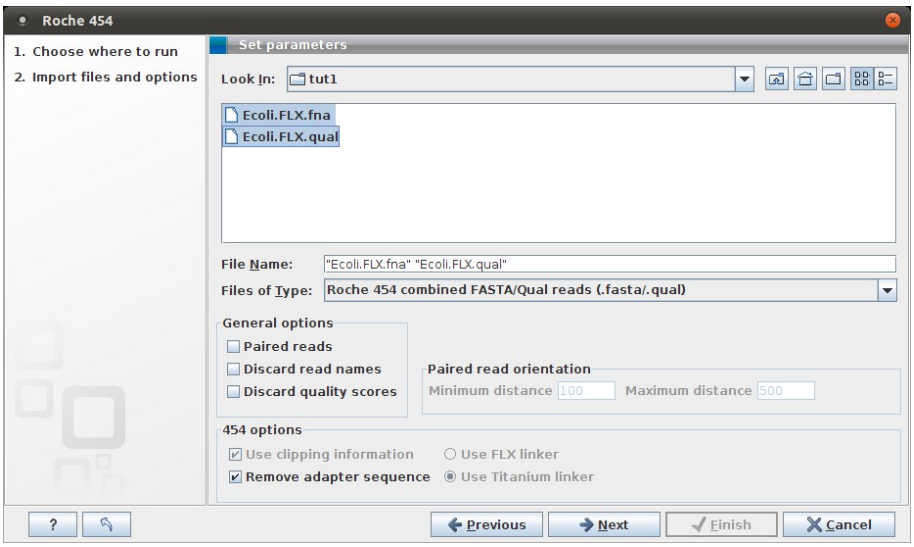


Figure 2.2: *Importing data from Roche 454.*

We support import of two kinds of data from 454 GS FLX systems:

- Flowgram files (`.sff`) which contain both sequence data and quality scores amongst others. However, the flowgram information is currently not used by *CLC Genomics Workbench*. There is an extra option to make use of clipping information (this will remove parts of the sequence as specified in the .sff file).
- Fasta/qual files:
  - 454 FASTA files (`.fna`) which contain the sequence data.
  - Quality files (`.qual`) which contain the quality scores.

For all formats, compressed data in gzip format is also supported (.gz).

The **General options** to the left are:

- **Paired reads**. The paired protocol for 454 entails that the forward and reverse reads are separated by a linker sequence. During import of paired data, the linker sequence is removed and the forward and reverse reads are separated and put into the same sequence list (their status as forward and reverse reads is preserved). You can change the linker sequence in the **Preferences** (in the **Edit** menu) under **Data**. Since the linker for the FLX and Titanium versions are different, you can choose the appropriate protocol during import, and in the preferences you can supply a linker for both platforms (see figure 2.3. Note that since the FLX linker is palindromic, it will only be searched on the plus strand, whereas the Titanium linker will be found on both strands. Some of the sequences may not have the linker in the middle of the sequence, and in that case the partial linker sequence is still removed, and the single read is put into a separate sequence list. Thus when you import 454 paired data, you may end up with two sequence lists: one for paired reads and one for single reads. Note that for de novo assembly projects, only the paired list should be used since the single reads list may contain reads where there is still a linker sequence present but only partially due to sequencing errors. Read more about handling paired data in section 2.1.8.
- **Discard read names**. For high-throughput sequencing data, the naming of the individual reads is often irrelevant given the huge amount of reads. This option allows you to discard this option to save disk space.
- **Discard quality scores**. Quality scores are visualized in the mapping view and they are used for SNP detection. If this is not relevant for your work, you can choose to **Discard quality scores**. One of the benefits from discarding quality scores is that you will gain a lot in terms of reduced disk space usage and memory consumption. If you have selected the fna/qual option and choose to discard quality scores, you do not need to select a .qual file.

Note! During import, partial adapter sequences are removed (`TCAG` and `ATGC`), and if the full sequencing adapters `GCCTTGCCAGCCCGCTCAG`, `GCCTCCCTCGCGCCATCAG` or their reverse complements are found, they are also removed (including tailing Ns). If you do not wish to remove the adapter sequences (e.g. if they have already been removed by other software), please uncheck the **Remove adapter sequence** option.

Click **Next** to adjust how to handle the results (see section **??**). We recommend choosing **Save** in order to save the results directly to a folder, since you probably want to save anyway before proceeding with your analysis. There is an option to put the import data into a separate folder. This can be handy for better organizing subsequent analysis results and for batching (see section **??**).

2.1.2 Illumina Genome Analyzer from Illumina

Choosing the Illumina import will open the dialog shown in figure 2.4.

The file formats accepted are:

- Fastq

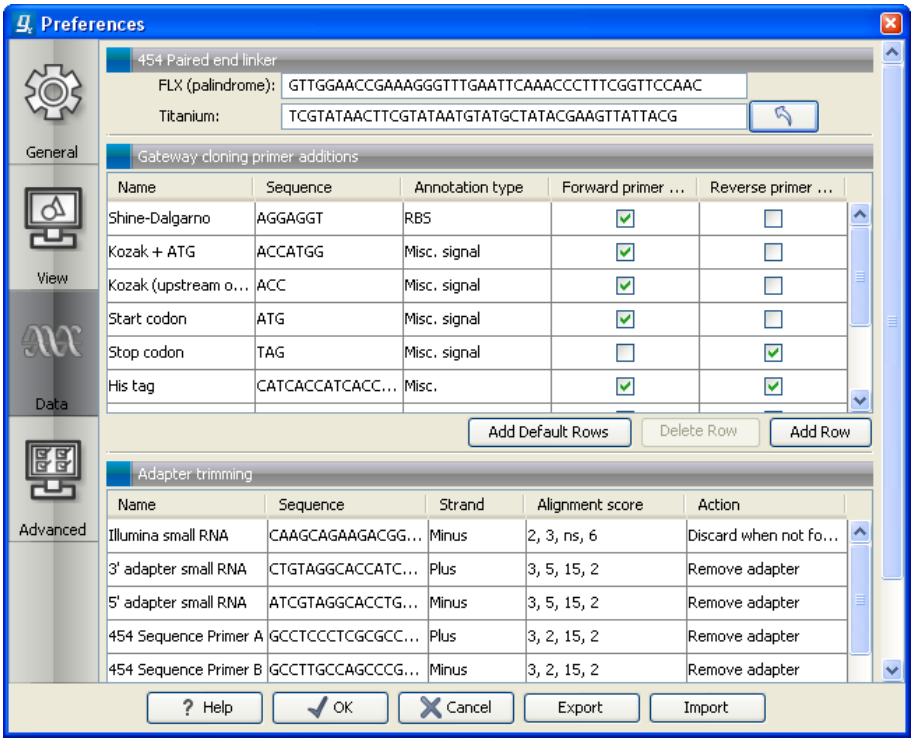


Figure 2.3: *Specifying linkers for 454 import.*

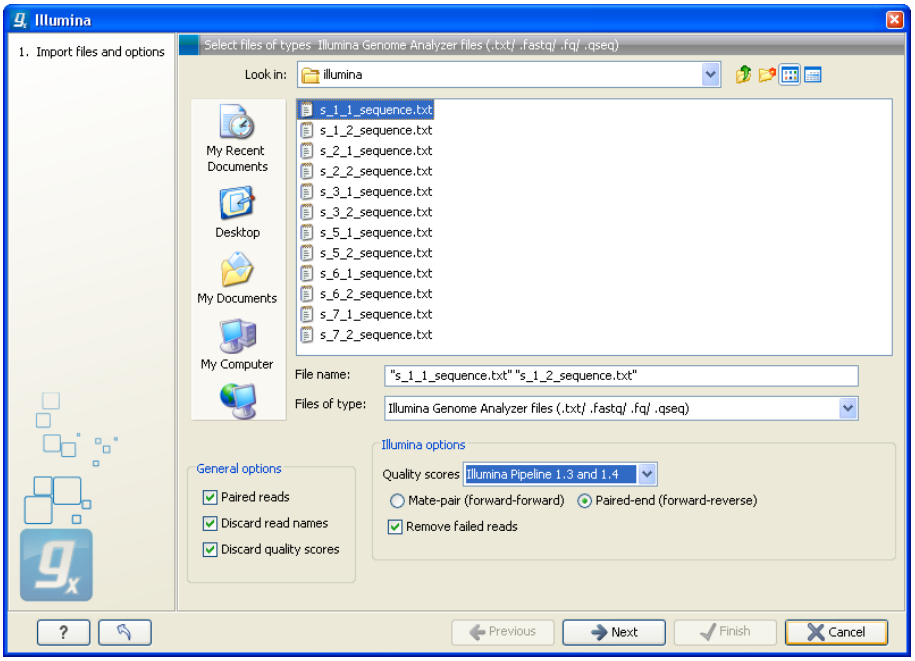


Figure 2.4: *Importing data from Illumina's Genome Analyzer.*

- Scarf
- Qseq

Paired data in any of these formats can be imported.

Note that there is information inside qseq and fastq files specifying whether a read has passed a quality filter or not. If you check **Remove failed reads** these reads will be ignored during import.

For qseq files there is a flag at the end of each read with values 0 (failed) or 1 (passed). In this example, the read is marked as failed and if Remove failed reads is checked, the read is removed.

```
M10  68  1  1  28680  29475  0  1  CATGGCCGTACAGGAAACACACATCATAGCATCACACGA  BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB  0
```

For fastq files, part of the header information for the quality score has a flag where Y means failed and N means passed. In this example, the read has not passed the quality filter:

```
@EAS139:136:FC706VJ:2:2104:15343:197393 1:Y:18:ATCACG
```

Note! In the **Illumina pipeline 1.5-1.7**, the letter B in the quality score has a special meaning. 'B' is used as a trim clipping. This means that when selecting Illumina pipeline 1.5-1.7, the *reads are automatically trimmed* when a B is encountered in the input file. This will happen also if you choose to discard quality scores during import.

If you import paired data and one read in a pair is removed during import, the remaining mate will be saved in a separate sequence list with single reads.

For all formats, compressed data in gzip format is also supported (.gz).

The **General options** to the left are:

- **Paired reads**. For paired import, you can select whether the data is **Paired-end** or **Mate-pair**. For paired data, the Workbench expects the first reads of the pairs to be in one file and the second reads of the pairs to be in another. When importing one pair of files, the first file in a pair will is assumed to contain the first reads of the pair, and the second file is assumed to contain the second read in a pair. So, for example, if you had specified that the pairs were in forward-reverse orientation, then the first file would be assumed to contain the forward reads. The second file would be assumed to contain the reverse reads.

  When loading files containing paired data, the *CLC Genomics Workbench* sorts the files selected according to rules based on the file naming scheme:

  - For files coming off the CASAVA1.8 pipeline, we organise pairs according to their identifier and chunk number. Files named with `_R1_` are assumed to contain the first sequences of the pairs, and those with `_R2_` in the name are assumed to contain the second sequence of the pairs.
  - For other files, we sort them all alphanumerically, and then group them two by two. This means that files 1 and 2 in the list are loaded as pairs, files 3 and 4 in the list are seen as pairs, and so on.

  In the simplest case, the files are typically named as shown in figure 2.4. In this case, the data is paired end, and the file containing the forward reads is called `s_1_1_sequence.txt` and the file containing reverse reads is called `s_1_2_sequence.txt`. Other common filenames for paired data, like `_1_sequence.txt`, `_1_qseq.txt`, `_2_sequence.txt` or `_2_qseq.txt` will be sorted alphanumerically. In such cases, files containing the final `_1` should contain the first reads of a pair, and those containing the final `_2` should contain the second reads of a pair.

  For files from CASAVA1.8, files with basenames like these: ID_R1_001, ID_R1_002, ID_R2_001, ID_R2_002 would be sorted in this order:

1. ID_R1_001
2. ID_R2_001
3. ID_R1_002
4. ID_R2_002

The data in files ID_R1_001 and ID_R2_001 would be loaded as a pair, and ID_R1_002, ID_R2_002 would be loaded as a pair.

Within each file, the first read of a pair will have a `1` somewhere in the information line. In most cases, this will be a `/1` at the end of the read name. In some cases though (e.g. CASAVA1.8), there will be a `1` elsewhere in the information line for each sequence. Similarly, the second read of a pair will have a `2` somewhere in the information line - either a `/2` at the end of the read name, or a `2` elsewhere in the information line.

If you do not choose to discard your read names on import (see next parameter setting), you can quickly check that your paired data has imported in the pairs you expect by looking at the first few sequence names in your imported paired data object. The first two sequences should have the same name, except for a `1` or a `2` somewhere in the read name line.

Paired-end and mate-pair data are handled the same way with regards to sorting on filenames. Their data structure is the same the same once imported into the Workbench. The only difference is that the expected orientation of the reads: reverse-forward in the case of mate pairs, and forward-reverse in the case of paired end data. Read more about handling paired data in section 2.1.8.

- **Discard read names**. For high-throughput sequencing data, the naming of the individual reads is often irrelevant given the huge amount of reads. This option allows you to discard quality scores to save disk space.
- **Discard quality scores**. Quality scores are visualized in the mapping view and they are used for SNP detection. If this is not relevant for your work, you can choose to **Discard quality scores**. One of the benefits from discarding quality scores is that you will gain a lot in terms of reduced disk space usage and memory consumption. Read more about the quality scores of Illumina below.

Click **Next** to adjust how to handle the results (see section **??**). We recommend choosing **Save** in order to save the results directly to a folder, since you probably want to save anyway before proceeding with your analysis. There is an option to put the import data into a separate folder. This can be handy for better organizing subsequent analysis results and for batching (see section **??**).

Quality scores in the Illumina platform

The quality scores in the FASTQ format come in different versions. You can read more about the FASTQ format at `http://en.wikipedia.org/wiki/FASTQ_format`. When you select to import Illumina data and click **Next** there is an option to use different quality score schemes at the bottom of the dialog (see figure 2.5).

There are three options:

- **Automatic**. Choosing this option, the Workbench attempts to automatically detect the quality score format. Sometimes this is not possible, and you have to specify the format

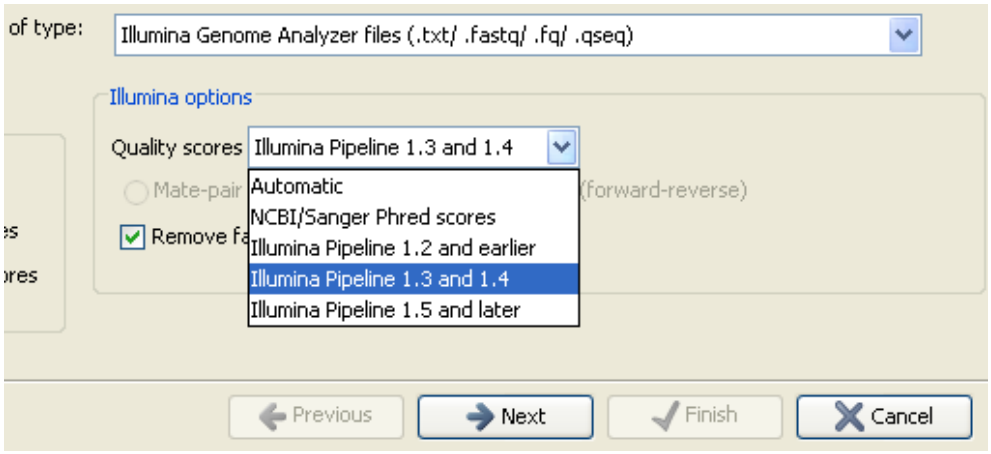


Figure 2.5: *Selecting the quality score scheme.*

yourself. In the cases where the Workbench is unable to determine the format, it is usually one of the Illumina Pipeline format files. If there are characters ; < = > or ? in the quality score information, it is the old Illumina pipeline format (ASCII values 59 to 63).

- **NCBI/Sanger or Illumina 1.8 and later**. Using a Phred scale encoded using ASCII 33 to 93. This is the standard for fastq formats except for the early Illumina data formats (this changed with version 1.8 of the Illumina Pipeline).
- **Illumina Pipeline 1.2 and earlier**. Using a Solexa/Illumina scale (-5 to 40) using ASCII 59 to 104. The Workbench automatically converts these quality scores to the Phred scale on import in order to ensure a common scale for analyses across data sets from different platforms (see details on the conversion next to the sample below).
- **Illumina Pipeline 1.3 and 1.4**. Using a Phred scale using ASCII 64 to 104.
- **Illumina Pipeline 1.5 to 1.7**. Using a Phred scale using ASCII 64 to 104. Values 0 (@) and 1 (A) are not used anymore. Value 2 (B) has special meaning and is used as a trim clipping. This means that when selecting Illumina Pipeline 1.5 and later, the *reads are automatically trimmed* when a B is encountered in the input file.

Small sample of all three kinds of files are shown below. The names of the reads have no influence on the quality score format:

NCBI/Sanger Phred scores:

```
@SRR001926.1 FC00002:7:1:111:750 length=36
TTTTTGTAAGGAGGGGGGTCATCAAAATTTGCAAAA
+SRR001926.1 FC00002:7:1:111:750 length=36
IIIIIIIIIIIIIIIIIIIIIIIIFIIII'IB<IH
@SRR001926.7 FC00002:7:1:110:453 length=36
TTATATGGAGGCTTTAAGAGTCATAGGTTGTTCCCC
+SRR001926.7 FC00002:7:1:110:453 length=36
IIIIIIIIII:'III?=IIIIII+&III/3I8F/&
```

Illumina Pipeline 1.2 and earlier (note the question mark at the end of line 4 - this is one of the values that are unique to the old Illumina pipeline format):

```
@SLXA-EAS1_89:1:1:672:654/1
GCTACGGAATAAAACCAGGAACAACAGACCCAGCA
+SLXA-EAS1_89:1:1:672:654/1
ccccccccccccccccccc]c``cVcZccbSYb?
@SLXA-EAS1_89:1:1:657:649/1
GCAGAAAATGGGAGTGAAAATCTCCGATGAGCAGC
+SLXA-EAS1_89:1:1:657:649/1
ccccccccccbccbccb``cccbcccZcc`^bR^`
```

The formulas used for converting the special Solexa-scale quality scores to Phred-scale:

$Q_{phred} = -10 \log_{10} p$

$Q_{solexa} = -10 \log_{10} \frac{p}{1-p}$

A sample of the quality scores of the Illumina Pipeline 1.3 and 1.4:

```
@HWI-E4_9_30WAF:1:1:8:178
GCCAGCGGCGCAAAATGNCGGCGGCGATGACCTTC
+HWI-E4_9_30WAF:1:1:8:178
babaaaa\ababaaaaREXabaaaaaaaaaaaaaa
@HWI-E4_9_30WAF:1:1:8:1689
GATGGAGATCTCGACCTNATAGGTGCCCTCATCGG
+HWI-E4_9_30WAF:1:1:8:1689
aab`]_aaaaaaaaa[ER`abaaa\aaaaaaaa[
```

Note that it is not possible to see from that data itself that it is actually not Illumina Pipeline 1.2 and earlier, since they use the same range of ASCII values.

To learn more about ASCII values, please see http://en.wikipedia.org/wiki/Ascii#ASCII_printable_characters.

2.1.3 SOLiD from Life Technologies

Choosing the SOLiD import will open the dialog shown in figure 2.6.

The file format accepted is the csfasta format which is the color space version of fasta format. If you want to import quality scores, a qual files should also be provided. The reads in a csfasta file look like this:

```
>2_14_26_F3
T0112131222002211230321112210212101313322221O1
>2_14_192_F3
T1100212211003100301200220322211132102211222 3
>2_14_233_F3
T0110013323111212123120223102033122011321112 3
>2_14_294_F3
T2130121323000000213232 12232.03300033102330332
```

All reads start with a T which specifies the right phasing of the color sequence.

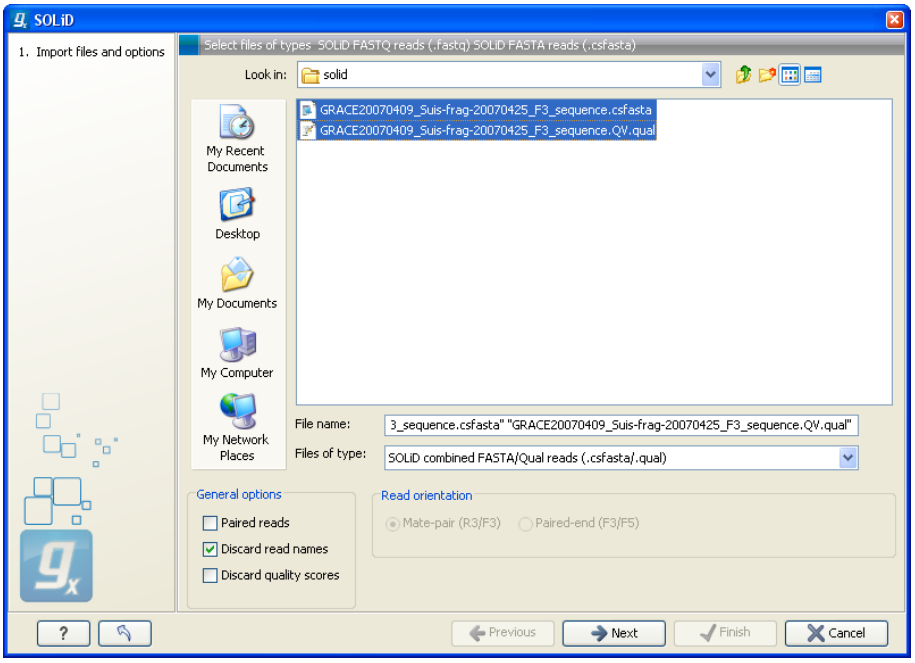


Figure 2.6: *Importing data from SOLiD from Applied Biosystems.*

If a reads has a `.` as you can see in the last read in the example above, it means that the color calling was ambiguous (this would have been an `N` if we were in base space). In this case, the Workbench simply cuts off the rest of the read, since there is no way to know the right phase of the rest of the colors in the read. If the read starts with a dot, it is not imported. If all reads start with a dot, a warning dialog will be displayed.

When the example above is imported into the Workbench, it looks as shown in figure 2.7.

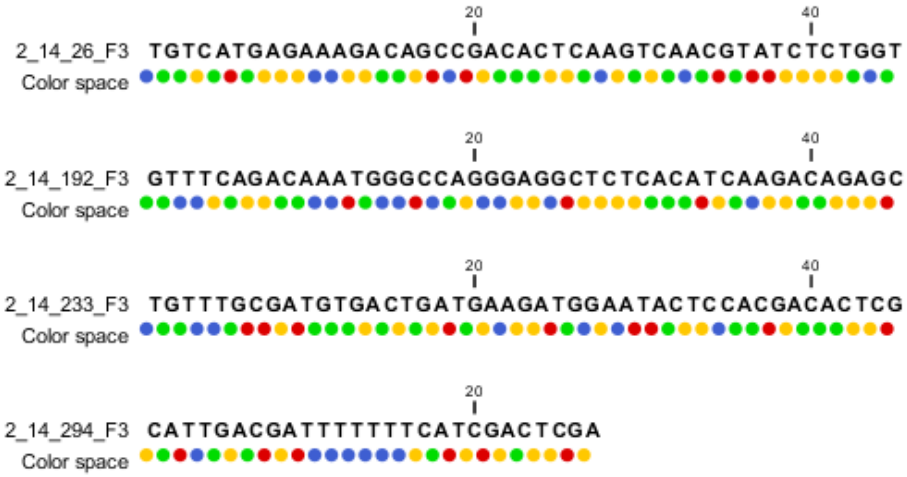


Figure 2.7: *Importing data from SOLiD from Applied Biosystems. Note that the fourth read is cut off so that the color following the dot are not included*

For more information about color space, please see section 2.8.

In addition to the native csfasta format used by SOLiD, you can also input data in fastq format. This is particularly useful for data downloaded from the Sequence Read Archive at NCBI (`http://www.ncbi.nlm.nih.gov/Traces/sra/`). An example of a SOLiD fastq file is

shown here with both quality scores and the color space encoding:

```
@SRR016056.1.1 AMELIA_20071210_2_YorubanCGB_Frag_16bit_2_51_130.1 length=50
T3100031312131021102231222331121211302212120133 2213
+SRR016056.1.1 AMELIA_20071210_2_YorubanCGB_Frag_16bit_2_51_130.1 length=50
!*%;2'%%050%'0'3%%5*.%%%),%%%%&%%%%%%'%%%%%'%%3+%%%
@SRR016056.2.1 AMELIA_20071210_2_YorubanCGB_Frag_16bit_2_51_223.1 length=50
T2000220112002121101201033221112213321233122130 2222
+SRR016056.2.1 AMELIA_20071210_2_YorubanCGB_Frag_16bit_2_51_223.1 length=50
!%%)%'))'&'%(((&%/&)%+(%%%&%%%%%%%%%%%%%%%+%%%%%%+'
```

For all formats, compressed data in gzip format is also supported (.gz).

The **General options** to the left are:

- **Paired reads**. When you import paired data, two different protocols are supported:
  - **Mate-pair**. For mate-pair data, the reads should be in two files with `_F3` and `_R3` in front of the the file extension. The orientation of the reads is expected to be forward-forward.
  - **Paired-end**. For paired-end data, the reads should be in two files with `_F3` and `_F5-P2` or `_F5-BC`. The orientation is expected to be forward-reverse.

  Read more about handling paired data in section 2.1.8.

  An example of a complete list of the four files needed for a SOLiD mate-paired data set including quality scores:

  ```
  dataset_F3.csfasta   dataset_F3.qual
  dataset_R3.csfasta   dataset_R3.qual
  ```

  or

  ```
  dataset_F3.csfasta   dataset_F3_.QV.qual
  dataset_R3.csfasta   dataset_R3_.QV.qual
  ```

- **Discard read names**. For high-throughput sequencing data, the naming of the individual reads is often irrelevant given the huge amount of reads. This option allows you to discard this option to save disk space.
- **Discard quality scores**. Quality scores are visualized in the mapping view and they are used for SNP detection. If this is not relevant for your work, you can choose to **Discard quality scores**. One of the benefits from discarding quality scores is that you will gain a lot in terms of reduced disk space usage and memory consumption. If you choose to discard quality scores, you do not need to select a .qual file.

Click **Next** to adjust how to handle the results (see section **??**). We recommend choosing **Save** in order to save the results directly to a folder, since you probably want to save anyway before proceeding with your analysis. There is an option to put the import data into a separate folder. This can be handy for better organizing subsequent analysis results and for batching (see section **??**).

2.1.4 Fasta format

Data coming in a standard fasta format can also be imported using the standard **Import** (), see section **??**. However, using the special high-throughput sequencing data import is recommended since the data is imported in a "leaner" format than using the standard import. This also means that all descriptions from the fasta files are ignored (usually there are none anyway for this kind of data).

The dialog for importing data in fasta format is shown in figure 2.8.

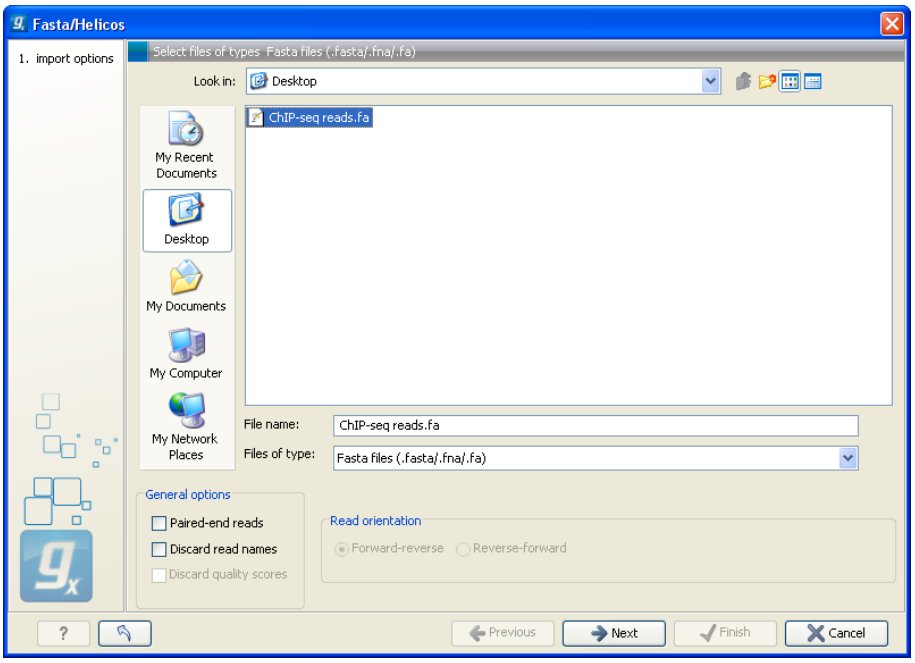


Figure 2.8: *Importing data in fasta format.*

Compressed data in gzip format is also supported (.gz).

The **General options** to the left are:

- **Paired reads**. For paired import, the Workbench expects the forward reads to be in one file and the reverse reads in another. The Workbench will sort the files before import and then assume that the first and second file belong together, and that the third and fourth file belong together etc. At the bottom of the dialog, you can choose whether the ordering of the files is **Forward-reverse** or **Reverse-forward**. As an example, you could have a data set with two files: `sample1_fwd` containing all the forward reads and `sample1_rev` containing all the reverse reads. In each file, the reads have to match each other, so that the first read in the `fwd` list should be paired with the first read in the `rev` list. Note that you can specify the insert sizes when running mapping and assembly. If you have data sets with different insert sizes, you should import each data set individually in order to be able to specify different insert sizes. Read more about handling paired data in section 2.1.8.
- **Discard read names**. For high-throughput sequencing data, the naming of the individual reads is often irrelevant given the huge amount of reads. This option allows you to discard this option to save disk space.
- **Discard quality scores**. This option is not relevant for fasta import, since quality scores are

not supported.

Click **Next** to adjust how to handle the results (see section **??**). We recommend choosing **Save** in order to save the results directly to a folder, since you probably want to save anyway before proceeding with your analysis. There is an option to put the import data into a separate folder. This can be handy for better organizing subsequent analysis results and for batching (see section **??**).

2.1.5 Sanger sequencing data

Although traditional sequencing data (with chromatogram traces like abi files) is usually imported using the standard **Import** (), see section **??**, this option has also been included in the High-Throughput Sequencing Data import. It is designed to handle import of large amounts of sequences, and there are three differences from the standard import:

- All the sequences will be put in one sequence list (instead of single sequences).
- The chromatogram traces will be removed (quality scores remain). This is done to improve performance, since the trace data takes up a lot of disk space and significantly impacts speed and memory consumption for further analysis.
- Paired data is supported.

With the standard import, it is practically impossible to import up to thousands of trace files and use them in an assembly. With this special High-Throughput Sequencing import, there is no limit. The import formats supported are the same: ab, abi, ab1, scf and phd.

For all formats, compressed data in gzip format is also supported (.gz).

The dialog for importing data Sanger sequencing data is shown in figure 2.9.

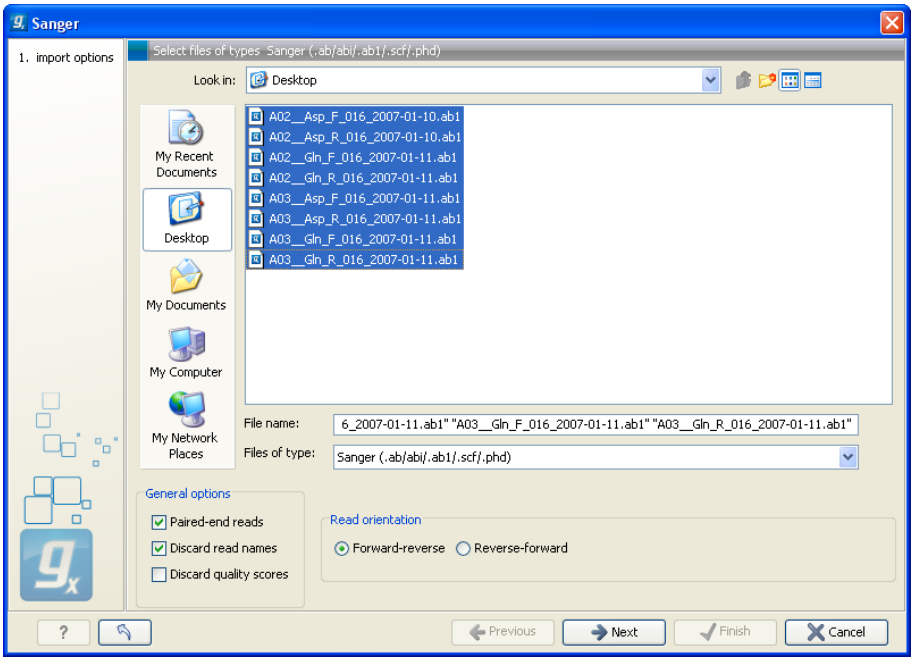


Figure 2.9: *Importing data from Sanger sequencing.*

The **General options** to the left are:

- **Paired reads**. The Workbench will sort the files before import and then assume that the first and second file belong together, and that the third and fourth file belong together etc. At the bottom of the dialog, you can choose whether the ordering of the files is **Forward-reverse** or **Reverse-forward**. As an example, you could have a data set with two files: `sample1_fwd` for the the forward read and `sample1_rev` for the reverse reads. Note that you can specify the insert sizes when running the mapping and the assembly. If you have data sets with different insert sizes, you should import each data set individually in order to be able to specify different insert sizes. Read more about handling paired data in section 2.1.8.
- **Discard read names**. For high-throughput sequencing data, the naming of the individual reads is often irrelevant given the huge amount of reads. This option allows you to discard this option to save disk space.
- **Discard quality scores**. Quality scores are visualized in the mapping view and they are used for SNP detection. If this is not relevant for your work, you can choose to **Discard quality scores**. One of the benefits from discarding quality scores is that you will gain a lot in terms of reduced disk space usage and memory consumption.

Click **Next** to adjust how to handle the results (see section **??**). We recommend choosing **Save** in order to save the results directly to a folder, since you probably want to save anyway before proceeding with your analysis. There is an option to put the import data into a separate folder. This can be handy for better organizing subsequent analysis results and for batching (see section **??**).

2.1.6 Ion Torrent PGM from Life Technologies

Choosing the Ion Torrent import will open the dialog shown in figure 2.10.

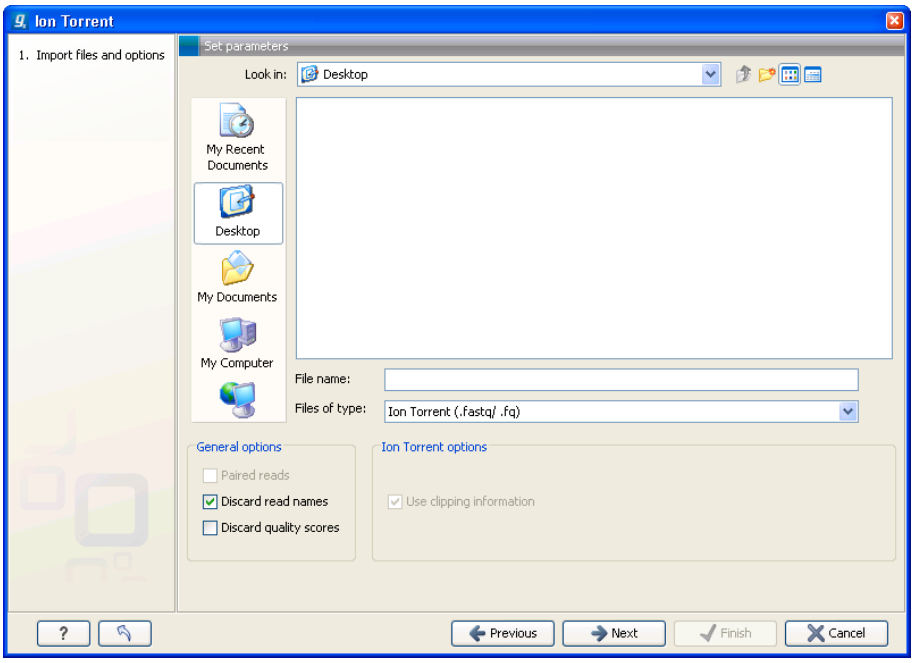


Figure 2.10: *Importing data from Ion Torrent.*

We support import of two kinds of data from the Ion Torrent system:

- SFF files (`.sff`)
- Fastq files (`.fastq`). Quality scores are expected to be in the NCBI/Sanger format (see section 2.1.2)

For all formats, compressed data in gzip format is also supported (.gz).

The **General options** to the left are:

- **Discard read names**. For high-throughput sequencing data, the naming of the individual reads is often irrelevant given the huge amount of reads. This option allows you to discard this option to save disk space.
- **Discard quality scores**. Quality scores are visualized in the mapping view and they are used for SNP detection. If this is not relevant for your work, you can choose to **Discard quality scores**. One of the benefits from discarding quality scores is that you will gain a lot in terms of reduced disk space usage and memory consumption. If you have selected the fna/qual option and choose to discard quality scores, you do not need to select a .qual file.

For sff files, you can also decide whether to use the clipping information in the file or not.

2.1.7 Complete Genomics

With *CLC Genomics Workbench 5.0* you can import *evidence* files from Complete Genomics. Support for other data types from Complete Genomics will be added later. The evidence files can be imported using the SAM/BAM importer, see section 2.1.9.

In order to import the data it need to be converted first. This is achieved using the CGA tools that can be downloaded from `http://www.completegenomics.com/sequence-data/cgatools/`.

The procedure for converting the data is the following.

1. Download the human genome in fasta format and make sure the chromosomes are named `chr<number>.fa`, e.g. `chr9.fa`.
2. Run the **fasta2crr** tool with a command like this:
   `cgatools fasta2crr --input chr9.fa --output chr9.crr`
3. Run the **evidence2sam** tool with a command like this:
   `cgatools evidence2sam --beta -e evidenceDnbs-chr9-.tsv -o chr9.sam -s chr9.crr`
   where the .tsv file is the evidence file provided by Complete Genomics (you can find sample data sets on their ftp server: `ftp://ftp2.completegenomics.com/`.
4. **Import** () the fasta file from 1. into the Workbench.
5. Use the SAM/BAM importer (section 2.1.9) to import the file created by the evidence2sam tool.

Please refer to the CGA documentation for a description about these tools. Note that this is not software supported by CLC bio.

2.1.8 General notes on handling paired data

During import, information about the orientation of paired data is stored by the *CLC Genomics Workbench*. This means that all subsequent analyses will automatically take differences in orientation into account. Once imported, both reads of a pair will be stored in the same sequence list. The forward and reverse reads (e.g. for paired-end data) simply alternate so that the first read is forward, the second read is the mate reverse read; the third is again forward and the fourth read is the mate reverse read. When deleting or manipulating sequence lists with paired data, be careful not break this order.

You can view and edit the orientation of the reads after they have been imported by opening the read list in the Element information view ( ), see section **??** as shown in figure 2.11.

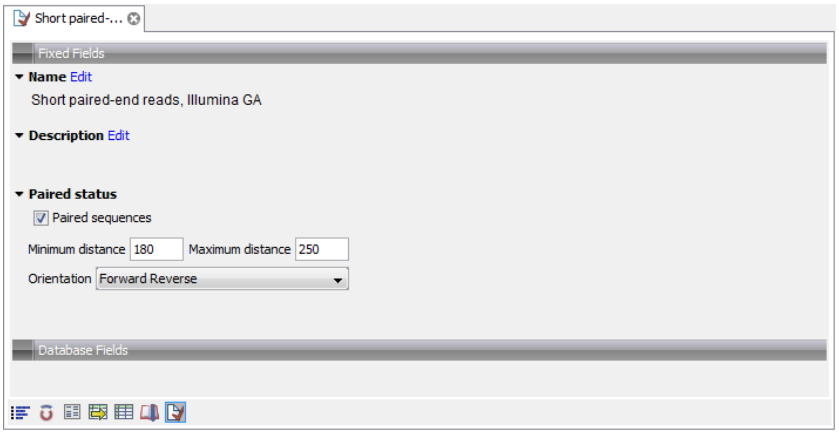


Figure 2.11: *The paired orientation and distance.*

In the **Paired status** part, you can specify whether the *CLC Genomics Workbench* should treat the data as paired data, what the orientation is and what the preferred distance is. The orientation and preferred distance is specified during import and can be changed in this view.

Note that the **paired distance** measure that is used throughout the *CLC Genomics Workbench* is always *including the full read sequence*. For paired-end libraries it means from the beginning of the forward read to the beginning of the reverse read.

2.1.9 SAM and BAM mapping files

The *CLC Genomics Workbench* supports import and export of files in SAM (Sequence Alignment/Map) and BAM format which are generic formats for storing large nucleotide sequence alignments. Read more and see the format specification at `http://samtools.sourceforge.net/`.

Please note that the *CLC Genomics Workbench* also supports SAM and BAM files from **Complete Genomics**.

For a detailed explanation of the SAM and BAM files exported from *CLC Genomics Workbench*, please see section **??**.

The idea behind the importer is that you import the sam/bam file which includes all the reads and then you specify one or more reference sequences which have already been imported into the Workbench. The Workbench will then combine the two to create a mapping result ( ) or mapping tables ( ). To import a SAM or BAM file:

File | Import High-Throughput Sequencing Data ( ) | SAM/BAM Mapping Files ( )

This will open a dialog where you choose the reference sequences to be used as shown in figure 2.12.

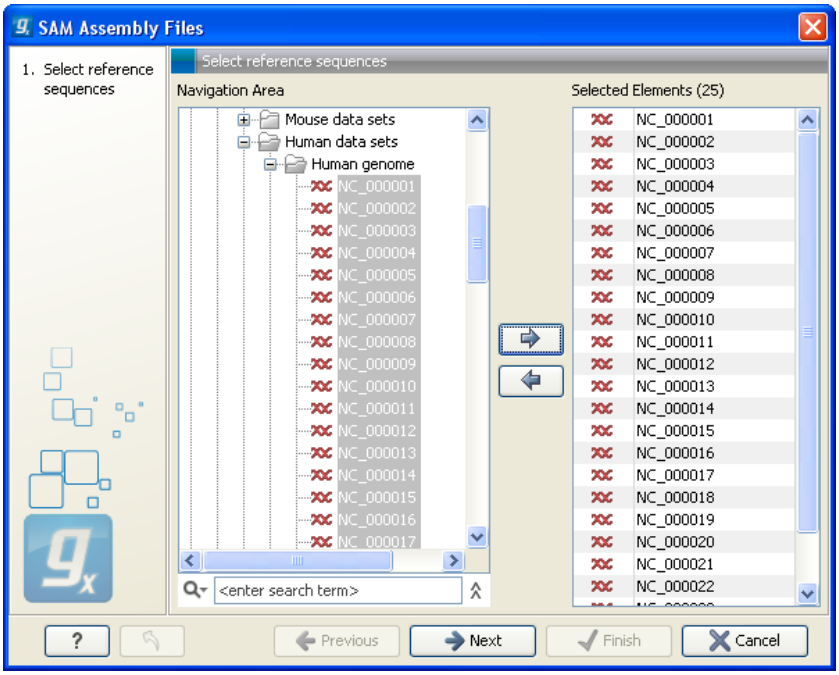


Figure 2.12: *Defining reference sequences.*

Select one or more reference sequence. Note that the name of your reference sequence has to match the reference name specified in the SAM/BAM file. Click **Next**.

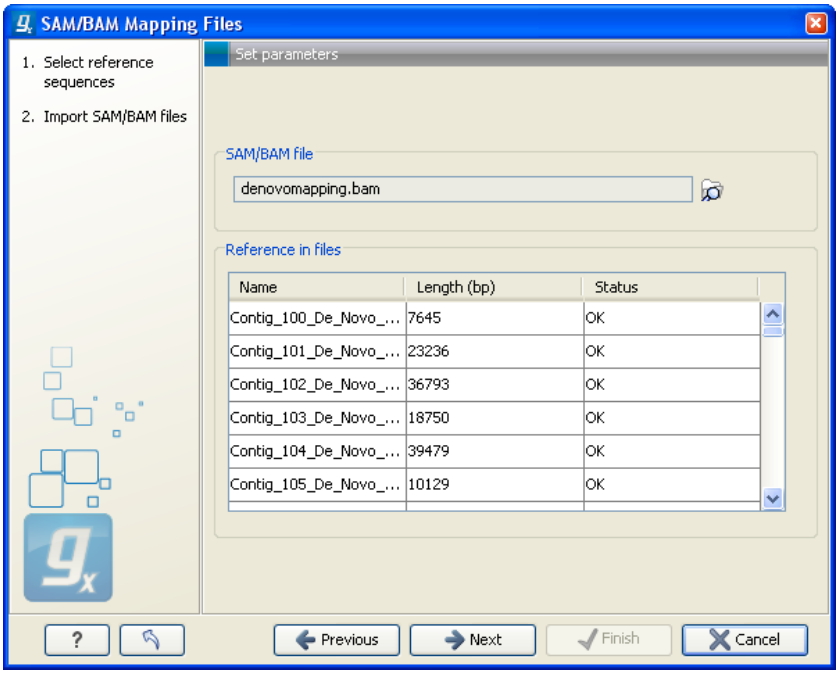


Figure 2.13: *Selecting the SAM/BAM file containing all the read information.*

In this dialog, select ( ) one or more SAM/BAM files as shown in figure 2.13.

In the panel below, all the reference sequences found in the SAM/BAM file will be listed included their lengths. In addition, it is indicated in the **Status** column whether they match the reference sequences selected from the Workbench. This can be used to double-check that the naming of the references are the same. (Note that reference sequences in a SAM/BAM file cannot contain spaces. If a reference sequence in the Workbench contains spaces, the space will be replaced with _ when comparing with the SAM/BAM file.). Figure2.14 shows an example where a reference sequence has not been provided (**input missing**) and one where the lengths of the reference sequences do not match (**Length differs**).

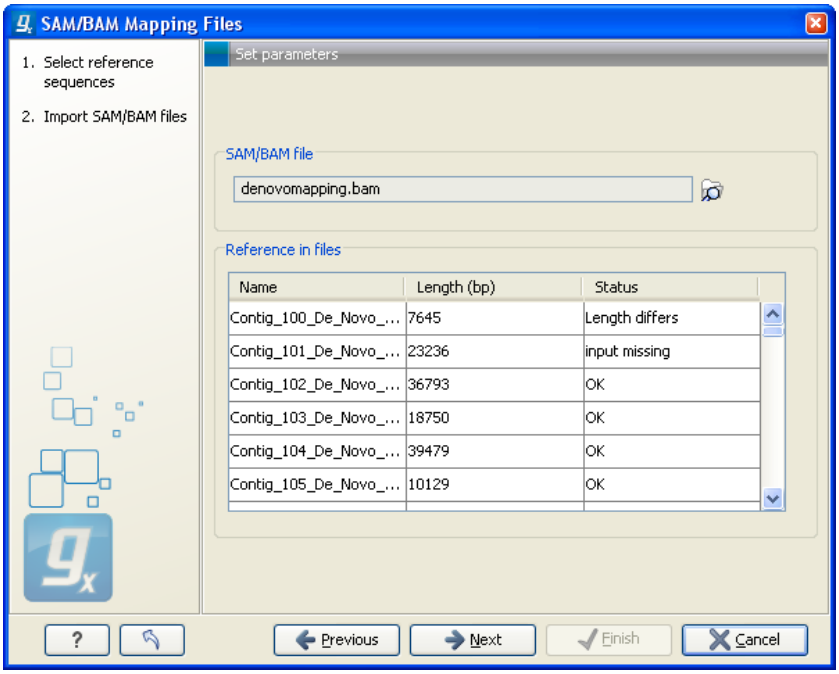


Figure 2.14: *When there is inconsistency in the naming and sizes of reference sequences, this is shown in the dialog prior to import.*

Click **Next** to adjust how to handle the results (see section **??**). We recommend choosing **Save** in order to save the results directly to a folder, since you probably want to save anyway before proceeding with your analysis.

Note that this import operation is very memory-consuming for large data sets.

2.1.10 Tabular mapping files

The *CLC Genomics Workbench* supports import and export of files in tabular format such as Eland files coming from the Illumina Pipeline. The importer is quite flexible which means that it can be used to import any kind of mapping file in a tab-delimited format where each line in the file represents one read.

The idea behind the importer is that you import the mapping file which includes all the reads and then you specify one or more reference sequences which have already been imported into the Workbench. The Workbench will then combine the two to create mapping results ( ) or mapping tables ( ). To import a tabular mapping file:

File | Import High-Throughput Sequencing Data ( ) | Tabular Mapping Files ( )

This will open a dialog where you choose the reference sequences to be used as shown in figure 2.15.

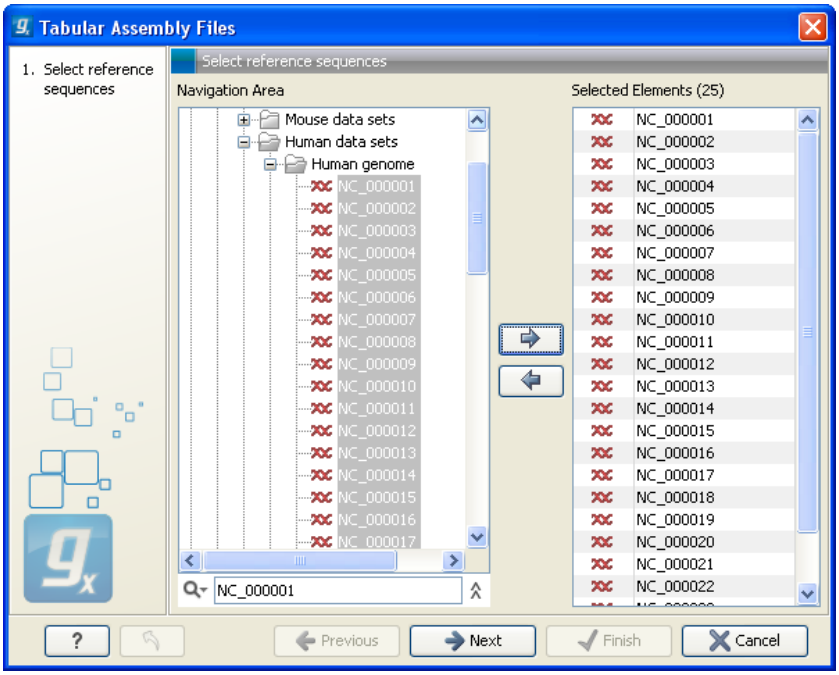


Figure 2.15: *Defining reference sequences.*

Select one or more reference sequence. Note that the name of your reference sequence has to match the reference name specified in the file. Click **Next**.

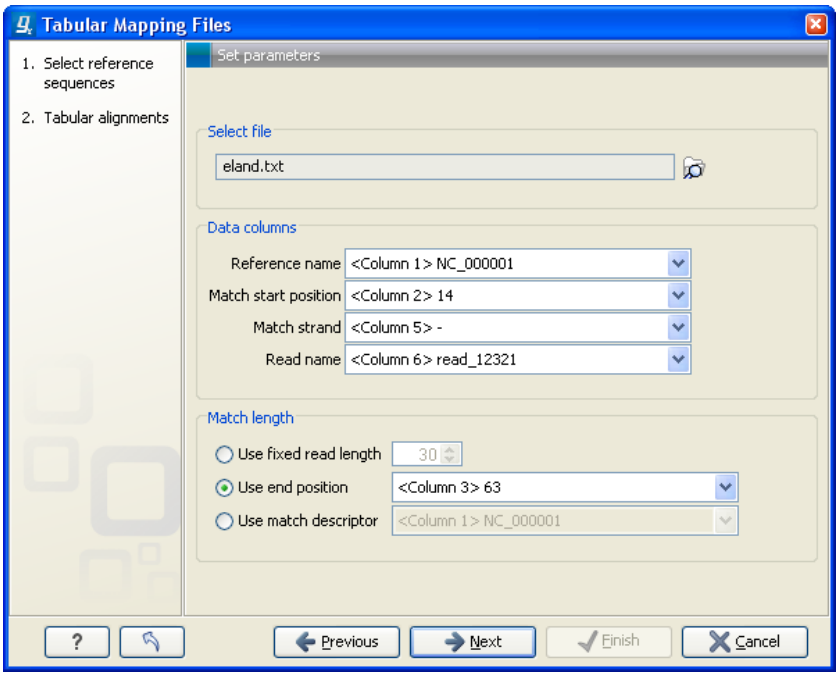


Figure 2.16: *Defining reference sequences.*

In this dialog, select ( ) one or more tab delimited files as shown in figure 2.16.

Once the tab delimited file has been selected, you have to specify the following information:

- **Data columns**. The Workbench needs to know how the file is organized in order to create a result where the reads have been mapped correctly.
  - **Reference name**. Select the column where the name reference sequence is specified. In the example above, this is in column 1.
  - **Match start position**. The position on the reference sequence where the read is mapped. The numbering starts from position 1.
  - **Match strand**. Whether the read is mapped the positive or negative strand. This should be specified using `F / R` (denoting forward and reverse reads) or `+ / −`.
  - **Read name**. Whether the read is mapped the positive or negative strand. This should be specified using `F / R` (denoting forward and reverse reads) or `+ / −`.
- **Match length**. The start position of the read is set above. In this section you specify the length of the match which can be done in any of the following ways:
  - **Use fixed read length**. If all reads have the same length, and if the read length or match end position is not provided in the file, you can specify a fixed length for all the reads.
  - **Use end position**. If you have a match end position just as a match start position, this can be used to determine match length.
  - **Use match descriptor**. This can be used to denote mismatches in the alignment. For a 35 base read, `35` denotes an exact match and `32C2` denotes substitution of a `C` at the 33rd position.

Note that the Workbench looks in the first line of the file to provide a preview when filling in this information.

Click **Next** to adjust how to handle the results (see section **??**). We recommend choosing **Save** in order to save the results directly to a folder, since you probably want to save anyway before proceeding with your analysis.

Note that this import operation is very memory-consuming for large data sets.

2.2 Multiplexing

When you do batch sequencing of different samples, you can use multiplexing techniques to run different samples in the same run. There is often a data analysis challenge to separate the sequencing reads, so that the reads from one sample are mapped together. The *CLC Genomics Workbench* supports automatic grouping of samples for two multiplexing techniques:

- **By name**. This supports grouping of reads based on their name.
- **By sequence tag**. This supports grouping of reads based on information within the sequence (tagged sequences).

The details of these two functionalities are described below.

2.2.1 Sort sequences by name

With this functionality you will be able to group sequencing reads based on their file name. A typical example would be that you have a list of files named like this:

```
...
A02__Asp_F_016_2007-01-10
A02__Asp_R_016_2007-01-10
A02__Gln_F_016_2007-01-11
A02__Gln_R_016_2007-01-11
A03__Asp_F_031_2007-01-10
A03__Asp_R_031_2007-01-10
A03__Gln_F_031_2007-01-11
A03__Gln_R_031_2007-01-11
...
```

In this example, the names have five distinct parts (we take the first name as an example):

- **A02** which is the position on the 96-well plate
- **Asp** which is the name of the gene being sequenced
- **F** which describes the orientation of the read (forward/reverse)
- **016** which is an ID identifying the sample
- **2007-01-10** which is the date of the sequencing run

To start mapping these data, you probably want to have them divided into groups instead of having all reads in one folder. If, for example, you wish to map each sample separately, or if you wish to map each gene separately, you cannot simply run the mapping on all the sequences in one step.

That is where **Sort Sequences by Name** comes into play. It will allow you to specify which part of the name should be used to divide the sequences into groups. We will use the example described above to show how it works:

> **Toolbox** | **High-throughput Sequencing ()** | **Multiplexing ()** | **Sort Sequences by Name ()**

This opens a dialog where you can add the sequences you wish to sort. You can also add sequence lists or the contents of an entire folder by right-clicking the folder and choose: **Add folder contents**.

When you click **Next**, you will be able to specify the details of how the grouping should be performed. First, you have to choose how each part of the name should be identified. There are three options:

- **Simple**. This will simply use a designated character to split up the name. You can choose a character from the list:
  - Underscore _

  - Dash -
  - Hash (number sign / pound sign) #
  - Pipe |
  - Tilde ~
  - Dot .

- **Positions**. You can define a part of the name by entering the start and end positions, e.g. from character number 6 to 14. For this to work, the names have to be of equal lengths.
- **Java regular expression**. This is an option for advanced users where you can use a special syntax to have total control over the splitting. See more below.

In the example above, it would be sufficient to use a simple split with the underscore _ character, since this is how the different parts of the name are divided.

When you have chosen a way to divide the name, the parts of the name will be listed in the table at the bottom of the dialog. There is a checkbox next to each part of the name. This checkbox is used to specify which of the name parts should be used for grouping. In the example above, if we want to group the reads according to sample ID and gene name, these two parts should be checked as shown in figure 2.17.

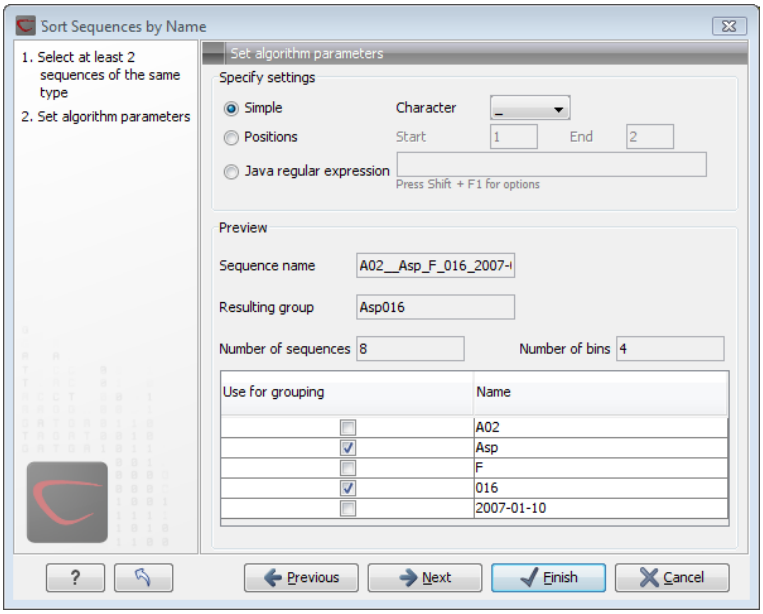


Figure 2.17: *Splitting up the name at every underscore (_) and using the sample ID and gene name for grouping.*

At the middle of the dialog there is a preview panel listing:

- **Sequence name**. This is the name of the first sequence that has been chosen. It is shown here in the dialog in order to give you a sample of what the names in the list look like.
- **Resulting group**. The name of the group that this sequence would belong to if you proceed with the current settings.
- **Number of sequences**. The number of sequences chosen in the first step.

- **Number of groups**. The number of groups that would be produced when you proceed with the current settings.

This preview cannot be changed. It is shown to guide you when finding the appropriate settings.

Click **Next** if you wish to adjust how to handle the results (see section **??**). If not, click **Finish**. A new sequence list will be generated for each group. It will be named according to the group, e.g. *Asp016* will be the name of one of the groups in the example shown in figure 2.17.

Advanced splitting using regular expressions

You can see a more detail explanation of the regular expressions syntax in section **??**. In this section you will see a practical example showing how to create a regular expression. Consider a list of files as shown below:

```
...
adk-29_adk1n-F
adk-29_adk2n-R
adk-3_adk1n-F
adk-3_adk2n-R
adk-66_adk1n-F
adk-66_adk2n-R
atp-29_atpA1n-F
atp-29_atpA2n-R
atp-3_atpA1n-F
atp-3_atpA2n-R
atp-66_atpA1n-F
atp-66_atpA2n-R
...
```

In this example, we wish to group the sequences into three groups based on the number after the "-" and before the "_" (i.e. 29, 3 and 66). The simple splitting as shown in figure 2.17 requires the same character before and after the text used for grouping, and since we now have both a "-" and a "_", we need to use the regular expressions instead (note that dividing by position would not work because we have both single and double digit numbers (3, 29 and 66)).

The regular expression for doing this would be `(.*)-(.*)_(.*)` as shown in figure 2.18. The round brackets () denote the part of the name that will be listed in the groups table at the bottom of the dialog. In this example we actually did not need the first and last set of brackets, so the expression could also have been `.*-(.*)_.*` in which case only one group would be listed in the table at the bottom of the dialog.

2.2.2 Process tagged sequences

Multiplexing as described in section 2.2.1 is of course only possible if proper sequence names could be assigned from the sequencing process. With many of the new high-throughput technologies, this is not possible.

However, there is a need for being able to input several different samples to the same sequencing run, so multiplexing is still relevant - it just has to be based on another way of identifying the

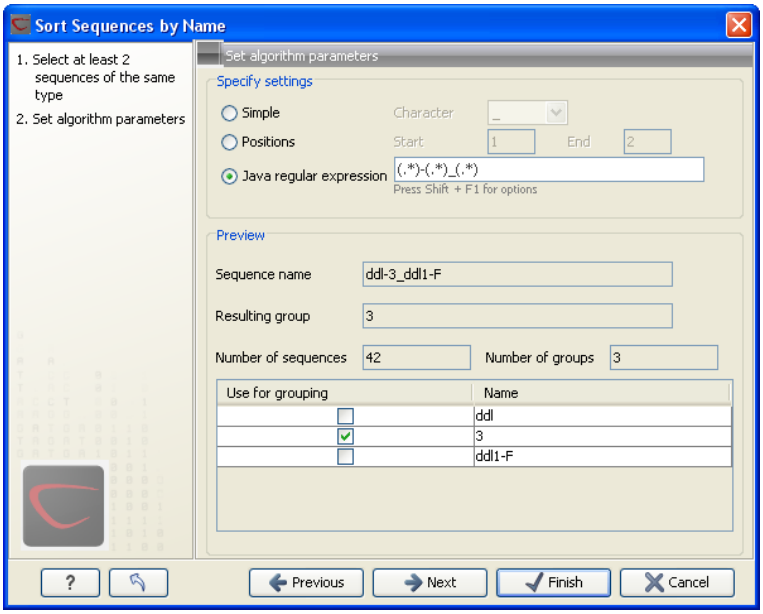


Figure 2.18: *Dividing the sequence into three groups based on the number in the middle of the name.*

sequences. A method has been proposed to *tag* the sequences with a unique identifier during the preparation of the sample for sequencing [Meyer et al., 2007].

With this technique, each sequence will have a sample-specific tag - a special sequence of nucleotides before and after the sequence of interest. This principle is shown in figure 2.19 (please refer to [Meyer et al., 2007] for more detailed information).

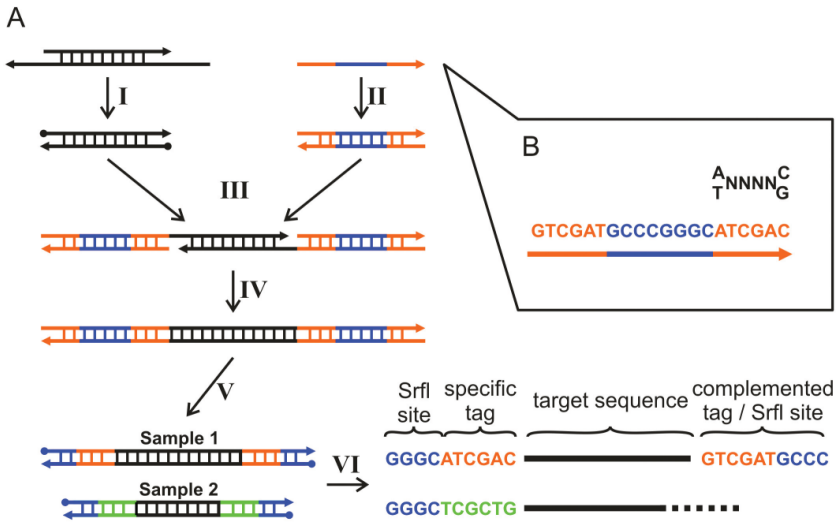


Figure 2.19: *Tagging the target sequence. Figure from [Meyer et al., 2007].*

The sample-specific tag - also called the barcode - can then be used to distinguish between the different samples when analyzing the sequence data. This post-processing of the sequencing data has been made easy by the multiplexing functionality of the *CLC Genomics Workbench* which simply divides the data into separate groups prior to analysis. Note that there is also an example using Illumina data at the end of this section.

Before processing the data, you need to import it as described in section 2.1.

The first step is to separate the imported sequence list into sublists based on the barcode of the sequences:

> **Toolbox** | **High-throughput Sequencing** ( ) | **Multiplexing** ( ) | **Process Tagged Sequences** ( )

This opens a dialog where you can add the sequences you wish to sort. You can also add sequence lists.

When you click **Next**, you will be able to specify the details of how the de-multiplexing should be performed. At the bottom of the dialog, there are three buttons which are used to **Add**, **Edit** and **Delete** the elements that describe how the barcode is embedded in the sequences.

First, click **Add** to define the first element. This will bring up the dialog shown in 2.20.

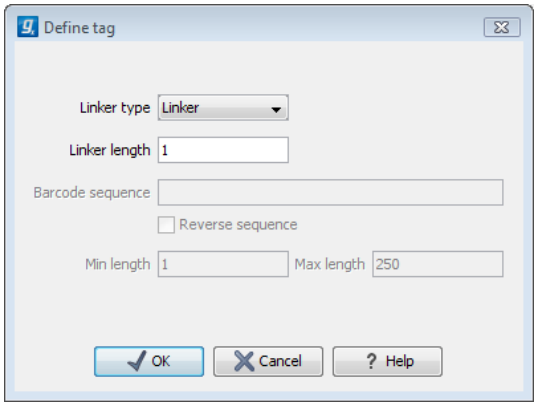


Figure 2.20: *Defining an element of the barcode system.*

At the top of the dialog, you can choose which kind of element you wish to define:

- **Linker**. This is a sequence which should just be ignored - it is neither the barcode nor the sequence of interest. Following the example in figure 2.19, it would be the four nucleotides of the *SrfI* site. For this element, you simply define its length - nothing else.
- **Barcode**. The barcode is the stretch of nucleotides used to group the sequences. For that, you need to define what the valid bases are. This is done when you click **Next**. In this dialog, you simply need to specify the length of the barcode.
- **Sequence**. This element defines the sequence of interest. You can define a length interval for how long you expect this sequence to be. The sequence part is the only part of the read that is retained in the output. Both barcodes and linkers are removed.

The concept when adding elements is that you add e.g. a linker, a barcode and a sequence in the desired sequential order to describe the structure of each sequencing read. You can of course edit and delete elements by selecting them and clicking the buttons below. For the example from figure 2.19, the dialog should include a linker for the *SrfI* site, a barcode, a sequence, a barcode (now reversed) and finally a linker again as shown in figure 2.21.

If you have paired data, the dialog shown in figure 2.21 will be displayed twice - one for each part of the pair.

Clicking **Next** will display a dialog as shown in figure 2.22.

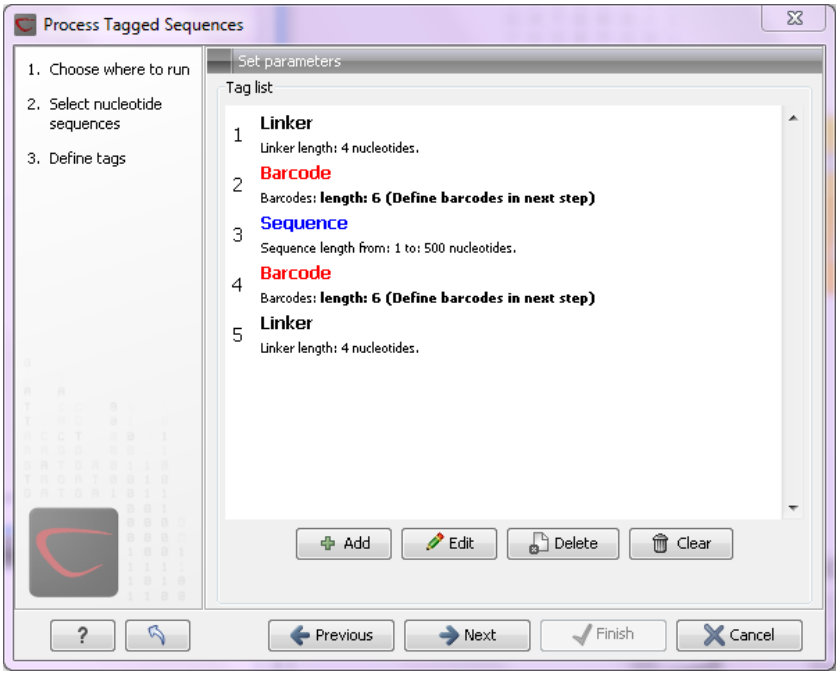


Figure 2.21: *Processing the tags as shown in the example of figure 2.19.*

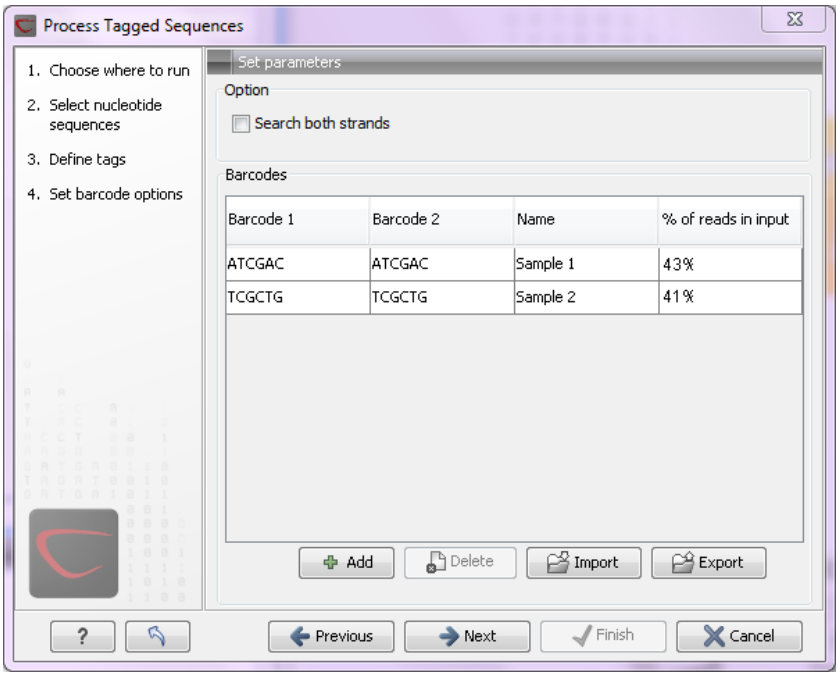


Figure 2.22: *Specifying the barcodes as shown in the example of figure 2.19.*

The barcodes can be entered manually by clicking the **Add** ( ) button. You can edit the barcodes and the names by clicking the cells in the table. The name is used for naming the results.

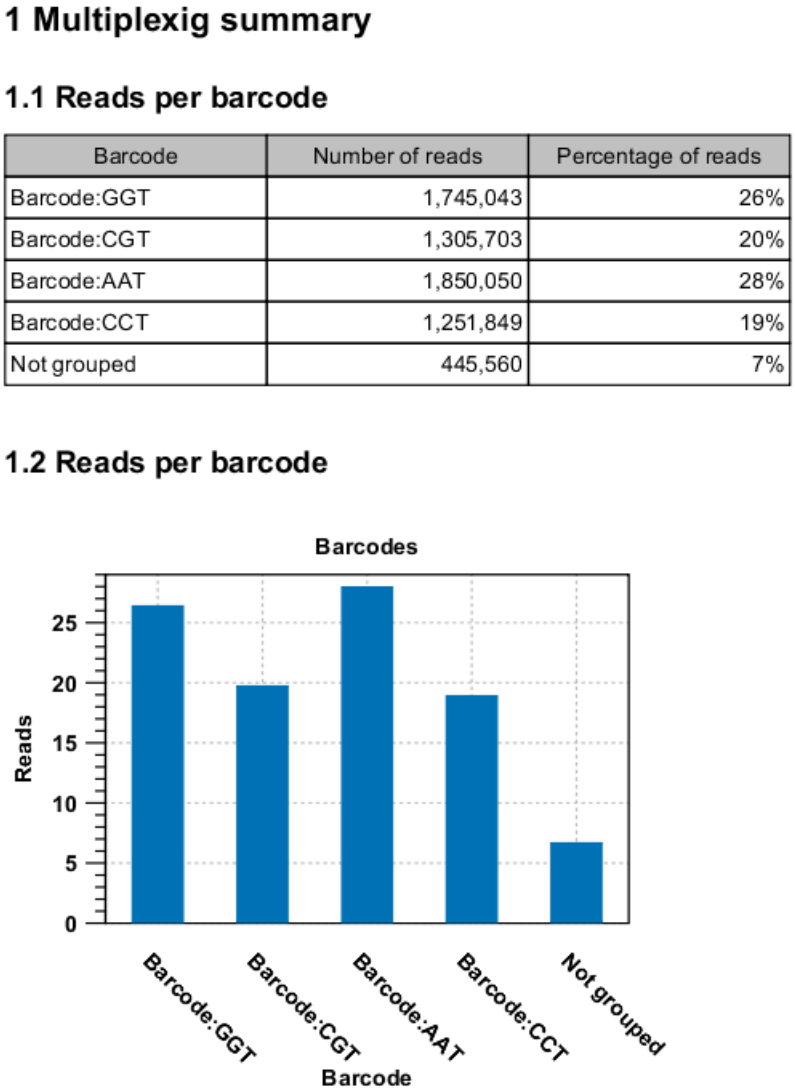
In addition to adding barcodes manually, you can also **Import** ( ) barcode definitions from an Excel or CSV file. The input format consists of two columns: the first contains the barcode sequence, the second contains the name of the barcode. An acceptable csv format file would contain columns of information that looks like:

```
 "AAAAAA","Sample1"
"GGGGGG","Sample2"
 "CCCCCC","Sample3"
```

The **Preview** column will show a preview of the results by running through the first 10,000 reads.

At the top, you can choose to search on both strands for the barcodes (this is needed for some 454 protocols where the MID is located at either end of the read).

Click **Next** to specify the output options. First, you can choose to create a list of the reads that could not be grouped. Second, you can create a summary report showing how many reads were found for each barcode (see figure 2.23).



1 Multiplexig summary

1.1 Reads per barcode

Barcode	Number of reads	Percentage of reads
Barcode:GGT	1,745,043	26%
Barcode:CGT	1,305,703	20%
Barcode:AAT	1,850,050	28%
Barcode:CCT	1,251,849	19%
Not grouped	445,560	7%

1.2 Reads per barcode

Figure 2.23: *An example of a report showing the number of reads in each group.*

There is also an option to create subfolders for each sequence list. This can be handy when the results need to be processed in batch mode (see section **??**).

A new sequence list will be generated for each barcode containing all the sequences where this barcode is identified. Both the linker and barcode sequences are removed from each of

the sequences in the list, so that only the target sequence remains. This means that you can continue the analysis by doing trimming or mapping. Note that you have to perform separate mappings for each sequence list.

An example using Illumina barcoded sequences

The data set in this example can be found at the Short Read Archive at NCBI: http://www.ncbi.nlm.nih.gov/sra/SRX014012. It can be downloaded directly in fastq format via the URL http://trace.ncbi.nlm.nih.gov/Traces/sra/sra.cgi?cmd=dload&run_list=SRR030730&format=fastq. The file you download can be imported directly into the Workbench.

The barcoding was done using the following tags at the beginning of each read: CCT, AAT, GGT, CGT (see supplementary material of [Cronn et al., 2008] at http://nar.oxfordjournals.org/cgi/data/gkn502/DC1/1).

The settings in the dialog should thus be as shown in figure 2.24.

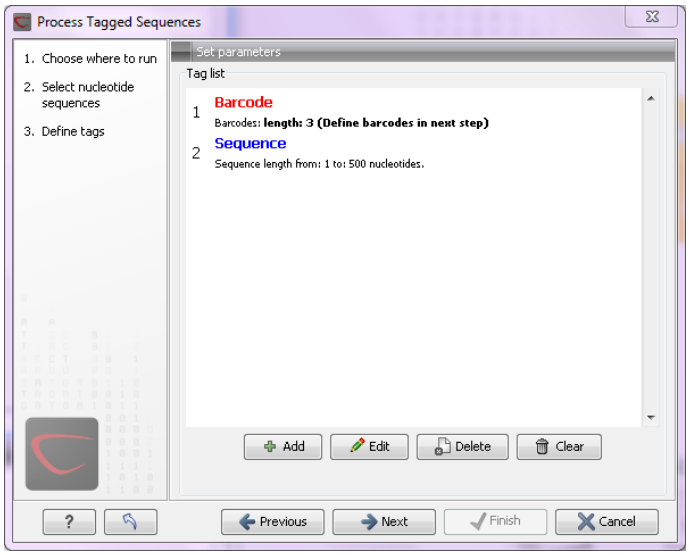


Figure 2.24: *Setting the barcode length at three*

Click **Next** to specify the bar codes as shown in figure 2.25 (use the **Add** button).

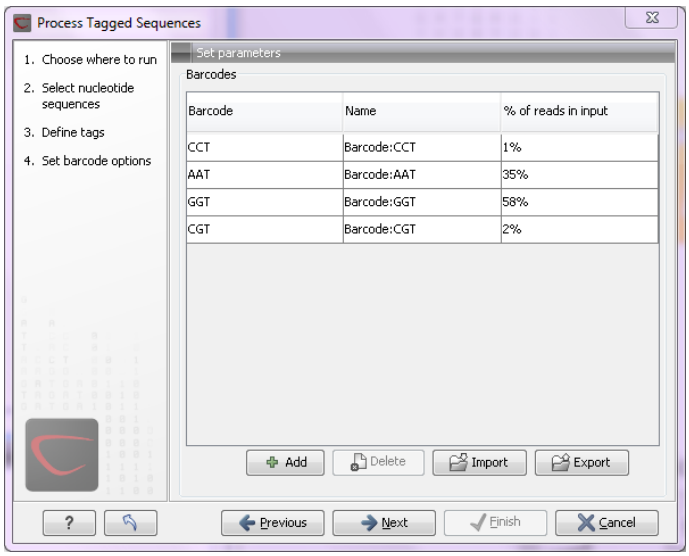


Figure 2.25: *A preview of the result*

With this data set we got the four groups as expected (shown in figure 2.26). The **Not grouped** list contains 445,560 reads that will have to be discarded since they do not have any of the barcodes.

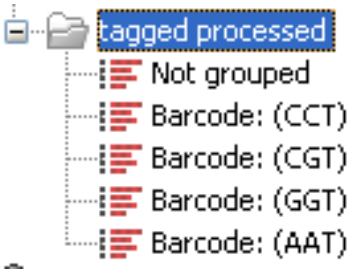


Figure 2.26: *The result is one sequence list per barcode and a list with the remainders*

2.3 Trim sequences

CLC Genomics Workbench offers a number of ways to trim your sequence reads prior to assembly and mapping, including adapter trimming, quality trimming and length trimming. Note that different types of trimming are performed sequentially in the same order as they appear in the trim dialogs:

1. Quality trimming based on quality scores
2. Ambiguity trimming to trim off e.g. stretches of Ns
3. Adapter trimming
4. Base trim to remove a specified number of bases at either 3' or 5' end of the reads
5. Length trimming to remove reads shorter or longer than a specified threshold

The result of the trim is a list of sequences that have passed the trim (referred to as the trimmed list below) and optionally a list of the sequences that have been discarded and a summary report (list of discarded sequences). The original data will be not be changed.

To start trimming:

Toolbox | **High-throughput Sequencing** ( ) | **Trim Sequences** ( )

This opens a dialog where you can add sequences or sequence lists. If you add several sequence lists, each list will be processed separately and you will get a a list of trimmed sequences for each input sequence list.

When the sequences are selected, click **Next**.

2.3.1 Quality trimming

This opens the dialog displayed in figure 2.27 where you can specify parameters for quality trimming.

The following parameters can be adjusted in the dialog:

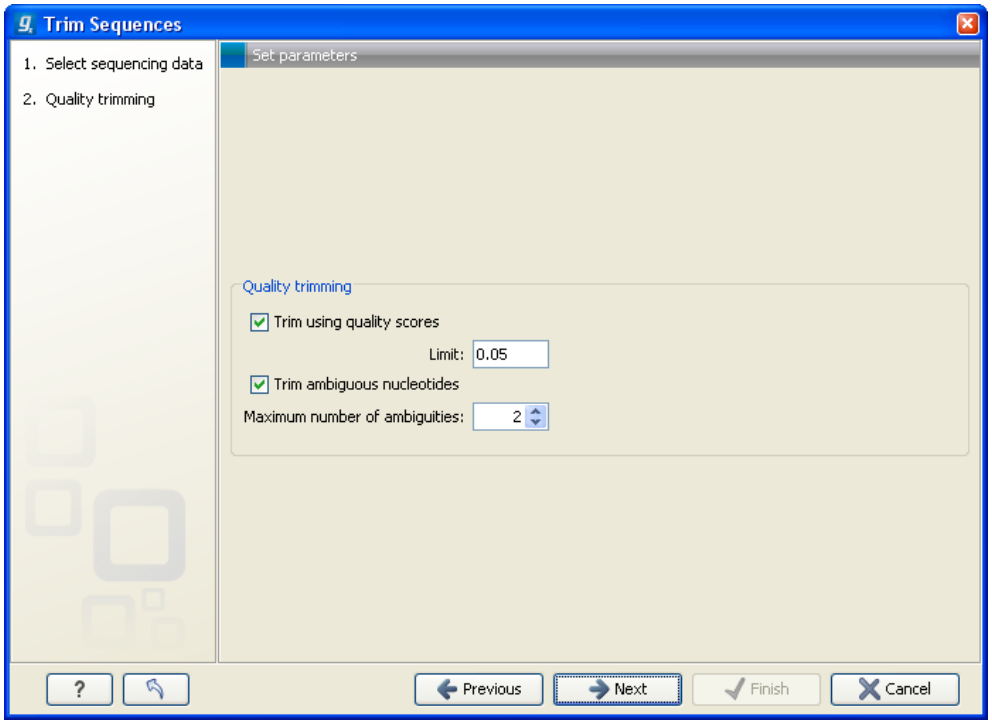


Figure 2.27: *Specifying quality trimming.*

- **Trim using quality scores.** If the sequence files contain quality scores from a base-caller algorithm this information can be used for trimming sequence ends. The program uses the modified-Mott trimming algorithm for this purpose (Richard Mott, personal communication):

  Quality scores in the Workbench are on a Phred scale in the Workbench (formats using other scales are converted during import). First step in the trim process is to convert the quality score (*Q*) to error probability: $p_{error} = 10^{\frac{Q}{-10}}$. (This now means that low values are high quality bases.)

  Next, for every base a new value is calculated: $Limit - p_{error}$. This value will be negative for low quality bases, where the error probability is high.

  For every base, the Workbench calculates the running sum of this value. If the sum drops below zero, it is set to zero. The part of the sequence to be retained after trimming is the region between the first positive value of the running sum and the highest value of the running sum. Everything before and after this region will be trimmed off.

  A read will be completely removed if the score never makes it above zero.

  At `http://www.clcbio.com/files/usermanuals/trim.zip` you find an example sequence and an Excel sheet showing the calculations done for this particular sequence to illustrate the procedure described above.

- **Trim ambiguous nucleotides.** This option trims the sequence ends based on the presence of ambiguous nucleotides (typically N). Note that the automated sequencer generating the data must be set to output ambiguous nucleotides in order for this option to apply. The algorithm takes as input the *maximal number of ambiguous nucleotides allowed in the sequence after trimming*. If this maximum is set to e.g. 3, the algorithm finds the maximum length region containing 3 or fewer ambiguities and then trims away the ends not included in this region.

2.3.2 Adapter trimming

Clicking **Next** will allow you to specify adapter trimming.

The *CLC Genomics Workbench* comes with a set of predefined adapter sequences from the most common kits provided by the high-throughput sequencing vendors. You can easily add or modify the adapters on this list in the preferences:

Edit | **Preferences (**) | **Data**

This will display the adapter trim panel as shown in figure 2.28 where each row represents an adapter sequence including the settings used for trimming.

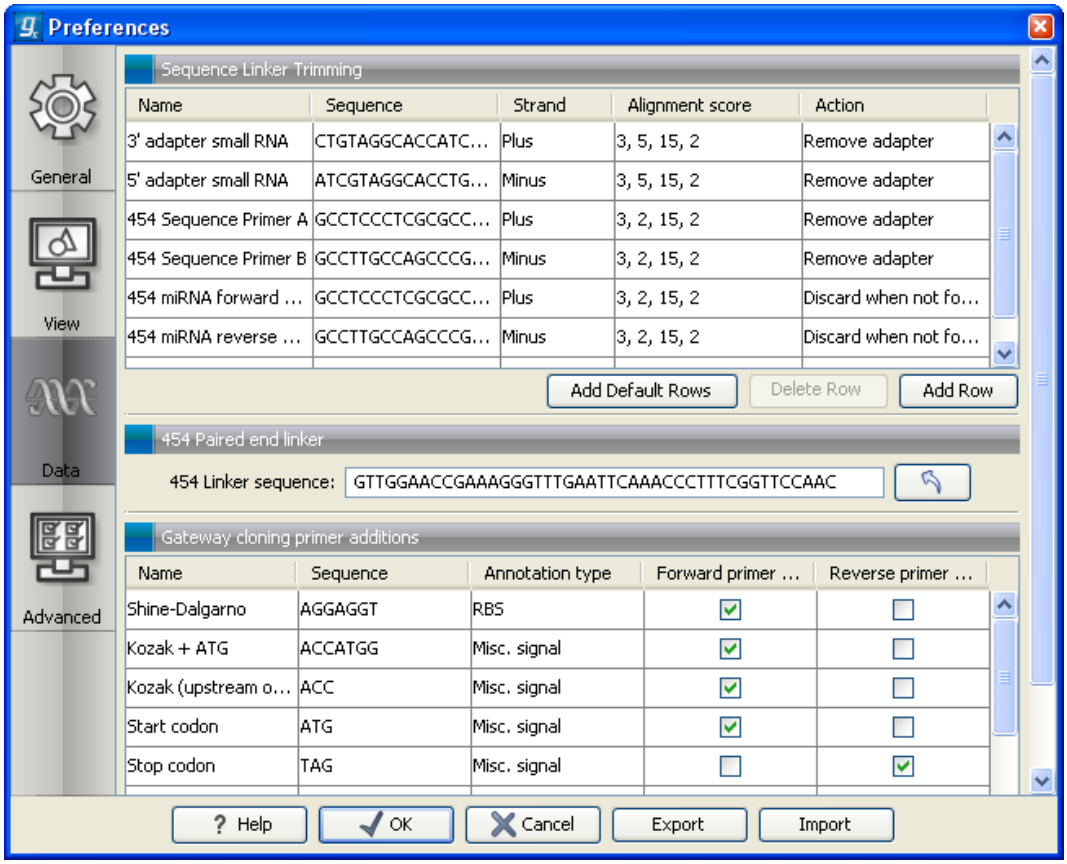


Figure 2.28: *Editing the set of adapters for adapter trimming.*

At the bottom of the panel, you have the following options:

- **Add Default Rows.** If you have deleted or changed the pre-defined set of adapters, you can add them to the list using this button (note that they will not replace existing adapters).
- **Delete Row**. Delete the selected adapter.
- **Add Row**. Add a new empty row where you can specify your own adapter settings.

All the information in the panel can be edited by clicking or double-clicking. The **Strand**, **Alignment score** and **Action** settings can also be modified when running the trim (see figure 2.34).

Action to perform when a match is found

For each read sequence in the input to trim, the Workbench performs a Smith-Waterman alignment [Smith and Waterman, 1981] with the adapter sequence to see if there is a match (details described below). When a match is found, the user can specify three kinds of actions:

- **Remove adapter.** This will remove the adapter *and all the nucleotides 5' of the match*. All the nucleotides 3' of the adapter match will be preserved in the read that will be retained in the trimmed reads list. If there are no nucleotides 3' of the adapter match, the read is added to the **List of discarded sequences** (see section 2.3.4).
- **Discard when not found**. If a match is found, the adapter sequence is removed (including all nucleotides 5' of the match as described above) and the rest of the sequence is retained in the list of trimmed reads. If no match is found, the whole sequence is discarded and put in the list of discarded sequences. This kind of adapter trimming is useful for small RNA sequencing where the remnants of the adapter is an indication that this is indeed a small RNA.
- **Discard when found**. If a match is found, the read is discarded. If no match is found, the read is retained in the list of trimmed reads. This can be used for quality checking the data for linker contaminations etc.

When is there a match?

To determine whether there is a match there is a set of scoring thresholds that can be adjusted and inspected by double-clicking the **Alignment score** column. This will bring up a dialog as shown in figure 2.29.

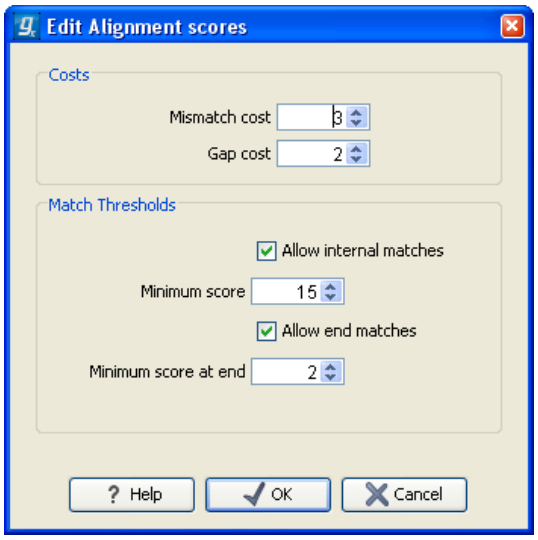


Figure 2.29: *Setting the scoring thresholds for adapter trimming.*

At the top, you can choose the costs for mismatch and gaps. A match is rewarded one point (this cannot be changed), and per default a mismatch costs 2 and a gap (insertion or deletion) costs 3. A few examples of adapter matches and corresponding scores are shown below.

In the panel below, you can set the **Minimum score** for a match to be accepted. Note that there is a difference between an **internal match** and an **end match**. The examples above are

```
    CGTATCAATCGATTACGCTATGAATG
a)      ||||||| ||||            11 matches - 2 mismatches = 7
       TTCAATCGGTTAC


    CGTATCAATCGATTACGCTATGAATG
       |||||||||| ||||          14 matches - 1 gap = 11
b)     ATCAATCGAT-CGCT


    CGTATCAATCGATTACGCTATGAATG
c)      |||||||                 7 matches - 3 mismatches = 1
       TTCAATCGGG
```

Figure 2.30: Three examples showing a sequencing read (top) and an adapter (bottom). The examples are artificial, using default setting with mismatch costs = 2 and gap cost = 3.

all internal matches where the alignment of the adapter falls within the read. Below are a few examples showing an adapter match at the end:

```
        CGTATCAATCGATTACGCTATGAATG
d)      |||||                         5 matches = 5 (as end match)
    GATTCGTAT


        CGTATCAATCGATTACGCTATGAATG
e)      || ||||                       6 matches - 1 mismatch = 4 (as end match)
    GATTCGCATCA


    CGTATCAATCGATTACGCTATGAATG
f)  |||| |||||                       9 matches - 1 gap = 6 (as end match)
    CGTA-CAATC


    CGTATCAATCGATTACGCTATGAATG
g)                  ||||||||||      10 matches = 10 (as internal match)
                   GCTATGAATG
```

Figure 2.31: Four examples showing a sequencing read (top) and an adapter (bottom). The examples are artificial.

In the first two examples, the adapter sequence extends beyond the end of the read. This is what typically happens when sequencing e.g. small RNAs where you sequence part of the adapter. The third example shows an example which could be interpreted both as an end match and an internal match. However, the Workbench will interpret this as an end match, because it starts at beginning (5' end) of the read. Thus, the definition of an end match is that the alignment of the adapter starts at the read's 5' end. The last example could also be interpreted as an end match, but because it is a the 3' end of the read, it counts as an internal match (this is because you would not typically expect partial adapters at the 3' end of a read). Also note, that if **Remove adapter** is chosen for the last example, the full read will be discarded because everything 5' of the adapter is removed.

Below, the same examples are re-iterated showing the results when applying different scoring schemes. In the first round, the settings are:

- Allowing internal matches with a minimum score of 6
- Not allowing end matches
- Action: Remove adapter

The result would be the following (the retained parts are green):

```
    CGTATCAATCGATTACGCTATGAATG
a)      ||||||| ||||                11 matches - 2 mismatches = 7
       TTCAATCGGTTAC

    CGTATCAATCGATTACGCTATGAATG
       |||||||||| ||||             14 matches - 1 gap = 11
b)     ATCAATCGAT-CGCT

    CGTATCAATCGATTACGCTATGAATG
c)      |||||||                     7 matches - 3 mismatches = 1
       TTCAATCGGG

        CGTATCAATCGATTACGCTATGAATG
d)      |||||                       5 matches = 5 (as end match)
    GATTCGTAT

        CGTATCAATCGATTACGCTATGAATG
e)      || ||||                     6 matches - 1 mismatch = 4 (as end match)
    GATTCGCATCA

    CGTATCAATCGATTACGCTATGAATG
f)  |||| |||||                      9 matches - 1 gap = 6 (as end match)
    CGTA-CAATC

    CGTATCAATCGATTACGCTATGAATG
g)                  ||||||||||      10 matches = 10 (as internal match)
                    GCTATGAATG
```

Figure 2.32: The results of trimming with internal matches only. Red is the part that is removed and green is the retained part. Note that the read at the bottom is completely discarded.

A different set of adapter settings could be:

- Allowing internal matches with a minimum score of 11
- Allowing end match with a minimum score of 4
- Action: Remove adapter

The result would be:

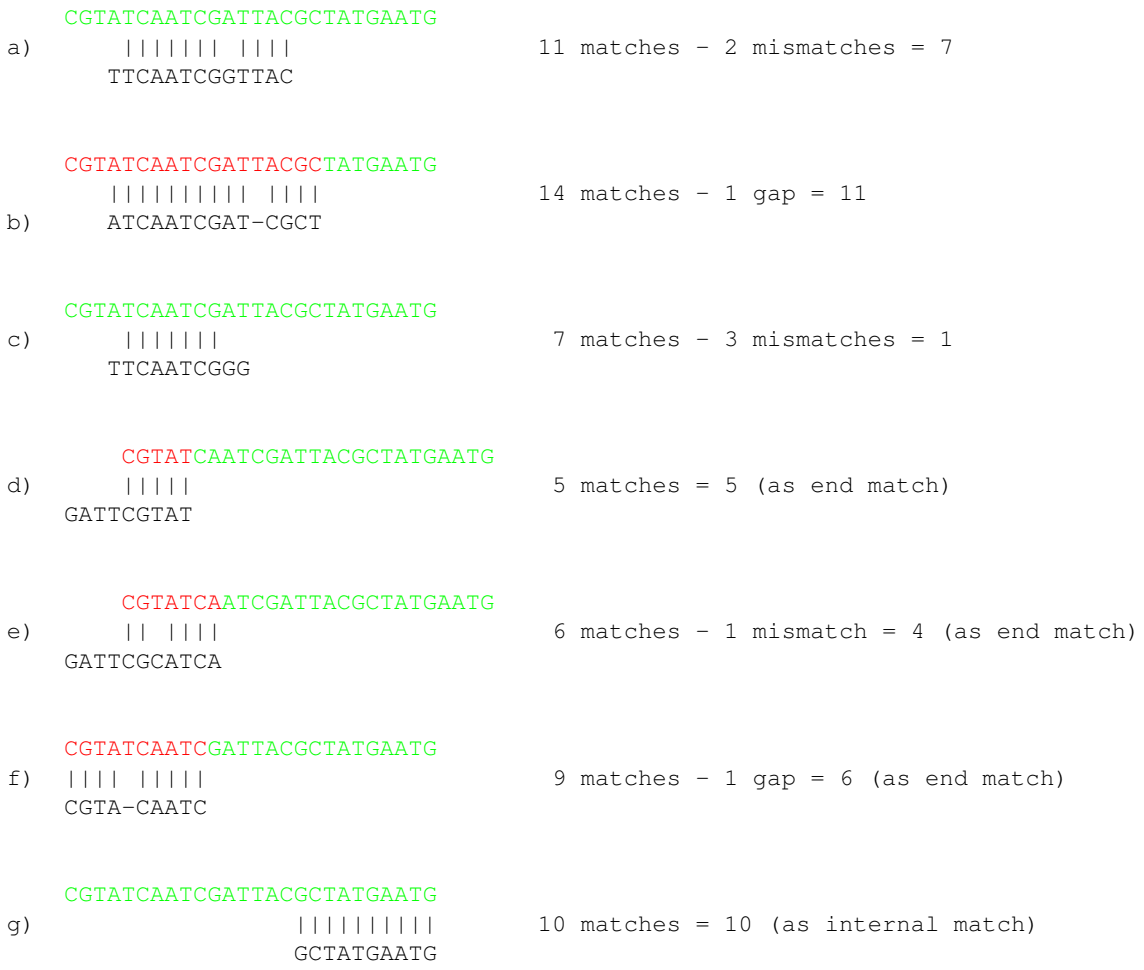


Figure 2.33: The results of trimming with both internal and end matches. Red is the part that is removed and green is the retained part.

Other adapter trimming options

When you run the trim, you specify the adapter settings as shown in figure 2.34.

You select an adapter to be used for trimming by checking the checkbox next to the adapter name. You can overwrite the settings defined in the preferences regarding **Strand**, **Alignment score** and **Action** by simply clicking or double-clicking in the table.

At the top, you can specify if the adapter trimming should be performed in **Color space**. Note that this option is only available for sequencing data imported using the SOLiD import (see section 2.1.3). When doing the trimming in color space, the Smith-Waterman alignment is simply done using colors rather than bases. The adapter sequence is still input in base space, and the Workbench then infers the color codes. Note that the scoring thresholds apply to the color space alignment (this means that a perfect match of 10 bases would get a score of 9 because 10 bases are represented by 9 color residues). Learn more about color space in section 2.8.

Besides defining the **Action** and **Alignment scores**, you can also define on which strand the adapter should be found. This can be done in two ways:

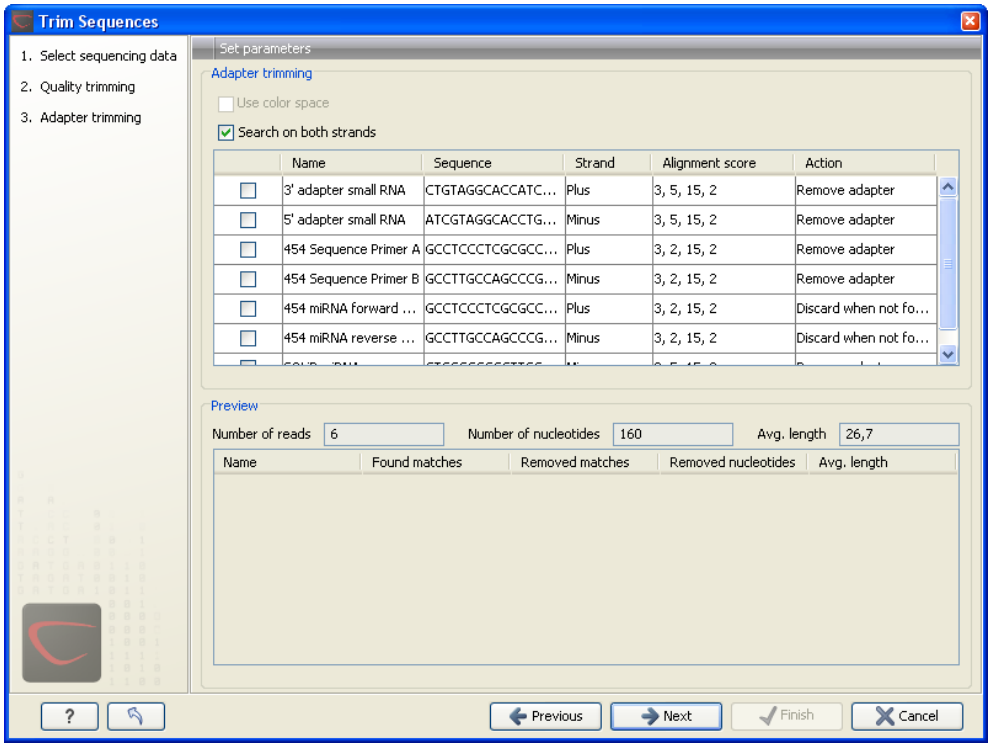


Figure 2.34: *Trimming your sequencing data for adapter sequences.*

- Defining either **Plus** or **Minus** for the individual adapter sequence (this can be done either in the **Preferences** or in the dialog shown in figure 2.34). Note that all the definitions above regarding 3' end and 5' end also apply to the minus strand (i.e. selecting the Minus strand is equivalent to reverse complementing all the reads). The adapter in this case should be defined as you would see it on the plus strand of the reverse complemented read. Figure 2.35 below shows a few examples of an adapter defined on the minus strand.

- Checking the **Search on both strands** checkbox will search both the minus and plus strand for the adapter sequence (the result would be equivalent to defining two adapters and searching one on the plus strand and one on the minus strand).

Below is an example showing hits for an adapter sequence defined as `CTGCTGTACGGCCAAGGCG`, searching on the minus strand. You can see that if you reverse complemented the adapter you

```
    ACCGAGAAACGCCTTGGCCGTACAGCAG
a)          ||||||||||||||||||    19 matches = 19
            GCGGTTCCAAGGCCGTACAGCTG

    ACCGATAAACGCCTTGGCCGTACAGCAGATGCC
b)          ||||||||| |||||||||    18 matches - 2 mismatches = 16
            GCGGTTCCAAGGCCGTACAGCTG
```

Figure 2.35: An adapter defined as CTGCTGTACGGCCAAGGCG searching on the minus strand. Red is the part that is removed and green is the retained part. The retained part is 3' of the match on the minus strand, just like matches on the plus strand.

would find the hit on the plus strand, but then you would have trimmed the wrong end of the read. So it is important to define the adapter as it is, without reverse complementing.

Below the adapter table you find a preview listing the results of trimming with the current settings on 1000 reads in the input file (reads 1001-2000 when the read file is long enough). This is useful for a quick feedback on how changes in the parameters affect the trimming (rather than having to run the full analysis several times to identify a good parameter set). The following information is shown:

- **Name**. The name of the adapter.
- **Matches found**. Number of matches found based on the strand and alignment score settings.
- **Reads discarded**. This is the number of reads that will be completely discarded. This can either be because they are completely trimmed (when the **Action** is set to Remove adapter and the match is found at the 3' end of the read), or when the **Action** is set to Discard when found or Discard when not found.
- **Nucleotides removed**. The number of nucleotides that are trimmed include both the ones coming from the reads that are discarded and the ones coming from the parts of the reads that are trimmed off.
- **Avg. length** This is the average length of the reads that are retained (excluding the ones that are discarded).

Note that the preview panel is only showing how the adapter trim affects the results. If other kinds of trimming (quality or length trimming) is applied, this will not be reflected in the preview but still influence the results.

Next time you run the trimming, your previous settings will automatically be remembered. Note that if you change settings in the **Preferences**, they may not be updated when running trim because the last settings are always used. Any conflicts are illustrated with text in *italics*. To make the updated preference take effect, press the **Reset to CLC Standard Settings** ( ) button.

2.3.3 Length trimming

Clicking **Next** will allow you to specify length trimming as shown in figure 2.36.

At the top you can choose to **Trim bases** by specifying a number of bases to be removed from either the 3' or the 5' end of the reads. Below you can choose to **Discard reads below length**. This can be used if you wish to simply discard reads because they are too short. Similarly, you can discard reads above a certain length. This will typically be useful when investigating e.g. small RNAs (note that this is an integral part of the small RNA analysis together with adapter trimming).

2.3.4 Trim output

Clicking **Next** will allow you to specify the output of the trimming as shown in figure 2.37.

No matter what is chosen here, the list of trimmed reads will always be produced. In addition the following can be output as well:

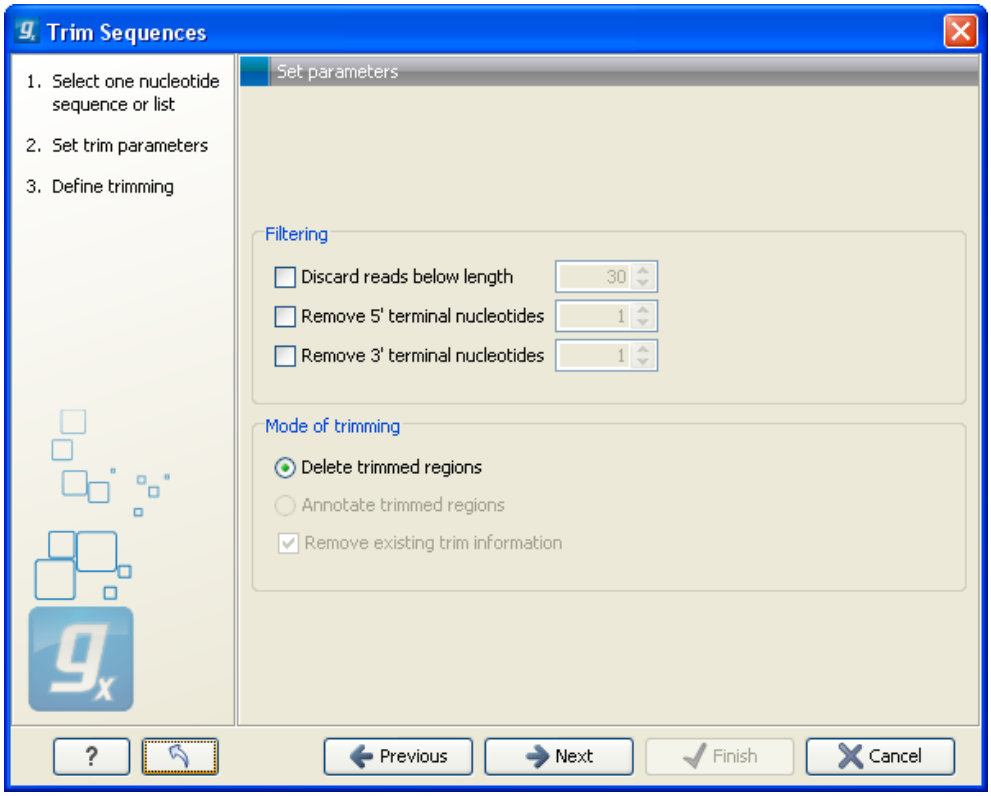


Figure 2.36: *Trimming on length.*

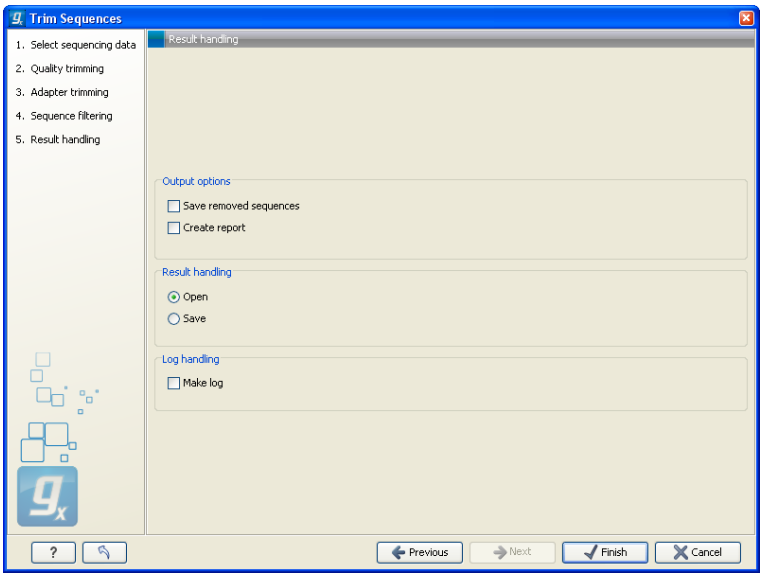


Figure 2.37: *Specifying the trim output. No matter what is chosen here, the list of trimmed reads will always be produced.*

- **Create list of discarded sequences**. This will produce a list of reads that have been discarded during trimming. When only parts of the read has been discarded, it will now show up in this list.
- **Create report**. An example of a trim report is shown in figure 2.38. The report includes the following:

- **Trim summary.**
  - **Name.** The name of the sequence list used as input.
  - **Number of reads.** Number of reads in the input file.
  - **Avg. length.** Average length of the reads in the input file.
  - **Number of reads after trim.** The number of reads retained after trimming.
  - **Percentage trimmed.** The percentage of the input reads that are retained.
  - **Avg. length after trim.** The average length of the retained sequences.
- **Read length before / after trimming**. This is a graph showing the number of reads of various lengths. The numbers before and after are overlayed so that you can easily see how the trimming has affected the read lengths (right-click the graph to open it in a new view).
- **Trim settings** A summary of the settings used for trimming.
- **Detailed trim results**. A table with one row for each type of trimming:
  - **Input reads.** The number of reads used as input. Since the trimming is done sequentially, the number of retained reads from the first type of trim is also the number of input reads for the next type of trimming.
  - **No trim.** The number of reads that have been retained, unaffected by the trimming.
  - **Trimmed.** The number of reads that have been partly trimmed. This number plus the number from **No trim** is the total number of retained reads.
  - **Nothing left or discarded.** The number of reads that have been discarded either because the full read was trimmed off or because they did not pass the length trim (e.g. too short) or adapter trim (e.g. if **Discard when not found** was chosen for the adapter trimming).

Click **Next** if you wish to adjust how to handle the results (see section **??**). If not, click **Finish**. This will start the trimming process.

If you trim paired data, the result will be a bit special. In the case where one part of a paired read has been trimmed off completely, you no longer have a valid paired read in your sequence list. In order to use paired information when doing assembly and mapping, the Workbench therefore creates two separate sequence lists: one for the pairs that are intact, and one for the single reads where one part of the pair has been deleted. When running assembly and mapping, simply select both of these sequence lists as input, and the Workbench will automatically recognize that one has paired reads and the other has single reads.

2.4 *De novo* assembly

The de novo assembly algorithm of *CLC Genomics Workbench* offers comprehensive support for a variety of data formats, including both short and long reads, and mixing of paired reads (both insert size and orientation).

The *de novo* assembly process has two stages:

1. First, simple contig sequences are created by using all the information that are in the read sequences. This is the actual *de novo* part of the process. These simple contig sequences do not contain any information about which reads the contigs are built from. This part is elaborated in section 2.4.1.

1 Trim summary

Name	Number of reads	Avg.length	Number of reads after trim	Percentage trimmed	Avg.length after trim
reads	57.213	228,0	55.754	~100%	232,8

2 Read length before / after trimming

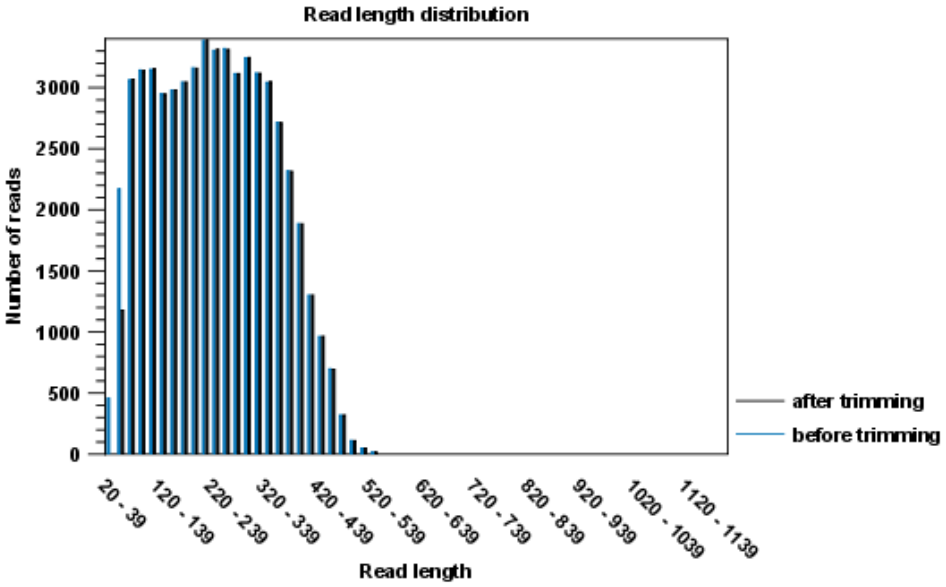


Figure 2.38: *A report with statistics on the trim results.*

2. Second, all the reads are mapped using the simple contig sequence as reference. This is done in order to show e.g. coverage levels along the contigs and enabling more downstream analysis like SNP detection and creating mapping reports. Note that although a read aligns to a certain position on the contig, it does not mean that the information from this read was used for building the contig, because the mapping of the reads is a completely separate part of the algorithm.

If you wish to only have the simple contig sequences as output, this can be chosen when starting the de novo assembly (see section 2.4.9).

2.4.1 How it works

CLC bio's de novo assembly algorithm works by using de Bruijn graphs. This is similar to how most new de novo assembly algorithms work [Zerbino and Birney, 2008, Zerbino et al., 2009, Li et al., 2010, Gnerre et al., 2011]. The basic idea is to make a table of all sub-sequences of a certain length (called words) found in the reads. The words are relatively short, e.g. about 20 for small data sets and 27 for a large data set (the word size is determined automatically, see explanation below).

Given a word in the table, we can look up all the potential neighboring words (in all the examples here, word of length 16 are used) as shown in figure 2.39.

Typically, only one of the backward neighbors and one of the forward neighbors will be present in

Backward neighbors	Starting word	Forward neighbors
AACGTAGCTAGCGCAT	ACGTAGCTAGCGCATG	CGTAGCTAGCGCATGA
CACGTAGCTAGCGCAT		CGTAGCTAGCGCATGC
GACGTAGCTAGCGCAT		CGTAGCTAGCGCATGG
TACGTAGCTAGCGCAT		CGTAGCTAGCGCATGT

Figure 2.39: *The word in the middle is 16 bases long, and it shares the 15 first bases with the backward neighboring word and the last 15 bases with the forward neighboring word.*

the table. A graph can then be made where each node is a word that is present in the table and edges connect nodes that are neighbors. This is called a de Bruijn graph.

For genomic regions without repeats or sequencing errors, we get long linear stretches of connected nodes. We may choose to reduce such stretches of nodes with only one backward and one forward neighbor into nodes representing sub-sequences longer than the initial words.

Figure 2.40 shows an example where one node has two forward neighbors:



Figure 2.40: *Three nodes connected, each sharing 15 bases with its neighboring node and ending with two forward neighbors.*

After reduction, the three first nodes are merged, and the two sets of forward neighboring nodes are also merged as shown in figure 2.41.



Figure 2.41: *The five nodes are compacted into three. Note that the first node is now 18 bases and the second nodes are each 17 bases.*

So bifurcations in the graph leads to separate nodes. In this case we get a total of three nodes after the reduction. Note that neighboring nodes still have an overlap (in this case 15 nucleotides since the word length is 16).

Given this way of representing the de Bruijn graph for the reads, we can consider some different situations:

When we have a *SNP or a sequencing error*, we get a so-called bubble (this is explained in detail in section 2.4.4) as shown in figure 2.42.



Figure 2.42: *A bubble caused by a heterozygous SNP or a sequencing error.*

Here, the central position may be either a C or a G. If this was a sequencing error occurring only once, we would see that one path through the bubble will only be words seen a single time. On the other hand if this was a heterozygote SNP we would see both paths represented more or less equally. Thus, having information about how many times this particular word is seen in all the reads is very useful and this information is stored in the initial word table together with the words.

The most difficult problem for de novo assembly is repeats. Repeat regions in large genomes often get very complex: a repeat may be found thousands of times and part of one repeat may also be part of another repeat. Sometimes a repeat is longer than the read length (or the paired distance when pairs are available) and then it becomes impossible to resolve the repeat. This is simply because there is no information available about how to connect the nodes before the repeat to the nodes after the repeat.

In the simple example, if we have a *repeat sequence* that is present twice in the genome, we would get a graph as shown in figure 2.43.



Figure 2.43: *The central node represents the repeat region that is represented twice in the genome. The neighboring nodes represent the flanking regions of this repeat in the genome.*

Note that this repeat is 57 nucleotides long (the length of the sub-sequence in the central node above plus regions into the neighboring nodes where the sequences are identical). If the repeat had been shorter than 15 nucleotides, it would not have shown up as a repeat at all since the word length is 16. This is an argument for using long words in the word table. On the other hand, the longer the word, the more words from a read are affected by a sequencing error. Also, for each extra nucleotide in the words, we get one less word from each read. This is in particular an issue for very short reads. For example, if the read length is 35, we get 16 words out of each read of the word length is 20. If the word length is 25, we get only 11 words from each read.

To strike a balance, CLC bio's de novo assembler chooses a word length based on the amount of input data: the more data, the longer the word length. It is based on the following:

```
word size 12: 0 bp - 30000 bp
word size 13: 30001 bp - 90002 bp
word size 14: 90003 bp - 270008 bp
word size 15: 270009 bp - 810026 bp
word size 16: 810027 bp - 2430080 bp
word size 17: 2430081 bp - 7290242 bp
word size 18: 7290243 bp - 21870728 bp
word size 19: 21870729 bp - 65612186 bp
word size 20: 65612187 bp - 196836560 bp
word size 21: 196836561 bp - 590509682 bp
word size 22: 590509683 bp - 1771529048 bp
word size 23: 1771529049 bp - 5314587146 bp
word size 24: 5314587147 bp - 15943761440 bp
word size 25: 15943761441 bp - 47831284322 bp
word size 26: 47831284323 bp - 143493852968 bp
word size 27: 143493852969 bp - 430481558906 bp
word size 28: 430481558907 bp - 1291444676720 bp
word size 29: 1291444676721 bp - 3874334030162 bp
word size 30: 3874334030163 bp - 11623002090488 bp
etc.
```

This pattern (multiplying by 3) continues until word size of 64 which is the max. Please note that the range of word sizes is 12-24 on 32-bit computers and 12-64 on 64-bit computers. See how

to adjust the word size in section 2.4.9

2.4.2 Resolve repeats using reads

Having build the de Bruijn graph using words CLC bio's de novo assembler removes repeats and errors using reads. This is done in the following order:

- Remove weak edges
- Remove dead ends
- Resolve repeats using reads without conflicts
- Resolve repeats with conflicts
- Remove weak edges
- Remove dead ends

Each phase will be explained in the following subsections.

Remove weak edges

The de Bruijn graph is expected to contain artifacts from errors in the data. The number of reads agreeing upon an error is likely to be low especially compared to the number of reads without errors for the same region. When this relative difference is large enough, it's possible to conclude something is an error.

In the remove weak edges phase we consider each node and calculate the number c_1 of edges connected to the node and the number of times k_1 a read is passing through these edges. An average of reads going through an edge is calculated $avg_1 = k_1/c_1$ and then the process is repeated using only those edges which have more than or equal avg_1 reads going though it. Let c_2 be the number of edges which meet this requirement and k_2 the number of reads passing through these edges. A second average $avg_2 = k_2/c_2$ is used to calculate a limit,

$$limit = \frac{\log(avg_2)}{2} + \frac{avg_2}{40}$$

and each edge connected to the node which has less than or equal $limit$ number of reads passing through it will be removed in this phase.

Remove dead ends

Some read errors might occur more often than expected, either by chance or because they are systematic sequencing errors. These are not removed by the "Remove weak edges" phase and will cause "dead ends" to occur in the graph which are short paths in the graph that terminate after a few nodes. Furthermore, the "Remove weak edges" sometimes only removes a part of the graph which will also leave dead ends behind. Dead ends are identified by searching for paths in the graph where there exits an alternative path containing four times more nucleotides. All nodes in such paths are then removed in this step.

Resolve repeats without conflicts

Repeats and other shared regions between the reads lead to ambiguities in the graph. These must be resolved otherwise the region will be output as multiple contigs, one for each node in the region.

The algorithm for resolving repeats without conflicts considers a number of nodes called the *window*. To start with, a window only contains one node, say *R*. We also define the *border nodes* as the nodes outside the window connected to a node in the window. The idea is to divide the border nodes into sets such that border nodes *A* and *C* are in the same set if there is a read going through *A*, through nodes in the window and then through *C*. If there are strictly more than one of these sets we can resolve the repeat area, otherwise we expand the window.

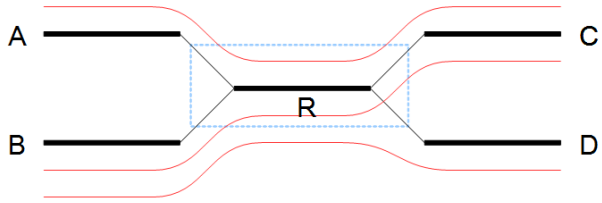


Figure 2.44: *A set of nodes.*

In the example in figure 2.44 all border nodes *A*, *B*, *C* and *D* are in the same set since one can reach every border nodes using reads (shown as red lines). Therefore we expand the window and in this case add node *C* to the window as shown in figure 2.45.

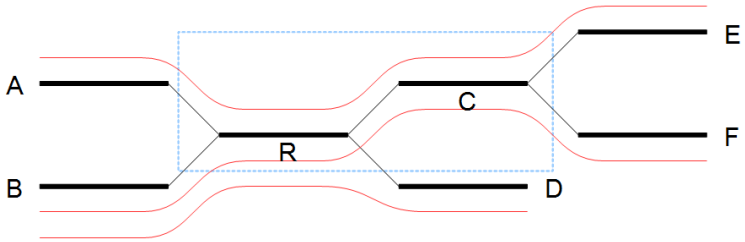


Figure 2.45: *Expanding the window to include more nodes.*

After the expansion of the window, the border nodes will be grouped into two groups being set *A*, *E* and set *B*, *D*, *F*. Since we have strictly more than one set, the repeat is resolved by copying the nodes and edges used by the reads which created the set. In the example the resolved repeat is shown in figure 2.46.

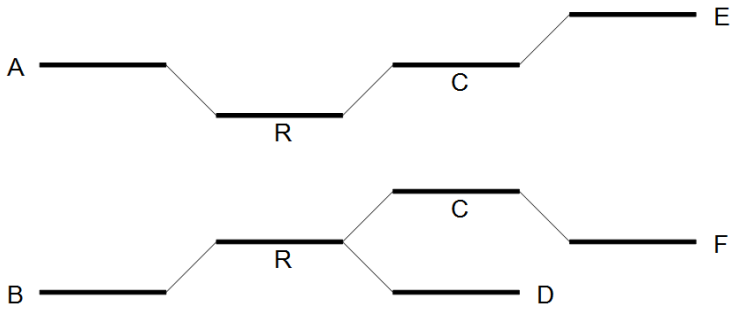


Figure 2.46: *Resolving the repeat.*

The algorithm for resolving repeats without conflict can be described the following way:

1. A node is selected as the window
2. The border is divided into sets using reads going through the window. If we have multiple sets, the repeat is resolved.
3. If the repeat cannot be resolved, we expand the window with nodes if possible and go to step 2.

The above steps are performed for every node.

Resolve repeats with conflicts

In the previous section repeats were resolved without excluding any reads that goes through the window. While this lead to a simpler graph, the graph will still contain artifacts which have to be removed. The next phase removes most of these errors and is similar to the previous phase:

1. A node is selected as the initial window
2. The border is divided into sets using reads going through the window. If we have multiple sets, the repeat is resolved.
3. If the repeat cannot be resolved, the border nodes are divided into sets using reads going through the window where reads containing errors are excluded. If we have multiple sets, the repeat is resolved.
4. The window is expanded with nodes if possible and step 2 is repeated.

The algorithm described above is similar to the algorithm using in previous section except step 3, where the reads with errors are excluded. This is done by calculating an average $avg_1 = m_1/c_1$ where m_1 is the number of reads going through the window and c_1 is the number of distinct pairs of border nodes having one (or more) of these reads connecting them. A second average $avg_2 = m_2/c_2$ is calculated where m_2 is the number of reads going through the window having at least avg_1 or more reads connecting their border nodes and c_2 the number of distinct pairs of border nodes having avg_1 or more reads connecting them. Then, a read between two border nodes *B* and *C* is excluded if the number of reads going through *B* and *C* is less than or equal to $limit$ given by

$$limit = \frac{\log(avg_2)}{2} + \frac{avg_2}{16}$$

An example were we resolve a repeat with conflicts is given in 2.47 where we have a total of 21 reads going through the window with $avg_1 = 21/3 = 7$, $avg_2 = 20/2 = 10$ and $limit = 1/2 + 10/16 = 1.125$. Therefore all reads between border nodes *B* and *C* are excluded resulting in two sets of border nodes *A*, *C* and *B*, *D*. The resolved repeat is shown in figure 2.48.

2.4.3 Optimization of the graph using paired reads

When paired reads are available, we can use the paired information to resolve large repeat regions that are not spanned by individual reads, but are spanned by read pairs. Given a set of paired reads that align to two nodes connected by a repeat region, the repeat region may be

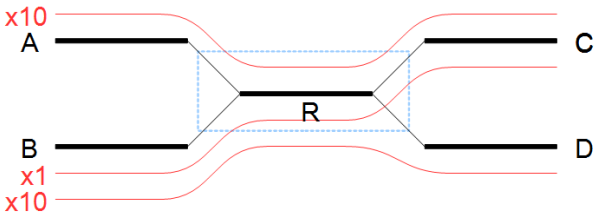


Figure 2.47: *A repeat with conflicts.*

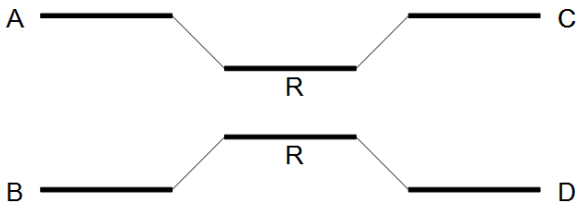


Figure 2.48: *Resolving a repeat with conflicts.*

resolved for those nodes if we can find a path connecting the nodes with a length corresponding to the paired read distance. However, such a path must be supported by a minimum of four sets of paired reads before the repeat is resolved.

If it's not possible to resolve the repeat, scaffolding is performed where paired read information is used to determine the distances between contigs and the orientation of these. Scaffolding is only considered between contigs with a minimum length of 120 to ensure that enough paired read information is available. An iterative greedy approach is used when performing scaffolding where short gaps are closed first, thus increasing the paired read information available for closing gaps (see figure 2.49).

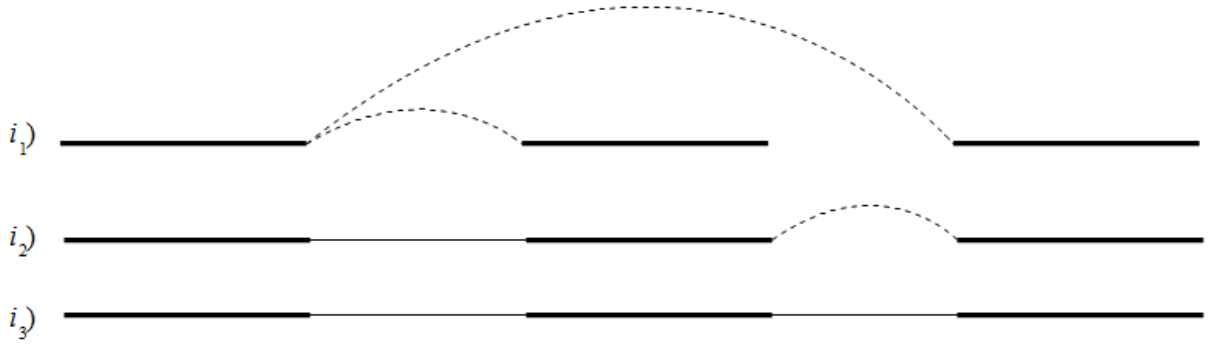


Figure 2.49: *Performing iterative scaffolding of the shortest gaps allows long pairs to be optimally used.* i_1 *shows three contigs with dashed arches indicating potential scaffolding.* i_2 *is after first iteration when the shortest gap has been closed and long potential scaffolding has been updated.* i_3 *is the final results with three contigs in one scaffold.*

Contigs in the same scaffold are output as one large contig with Ns inserted in between. The number of Ns inserted correspond to the estimated distance between contigs which is calculated based on the paired read information. More precisely, for each set of paired reads spanning two contigs a distance estimate is calculated based on the supplied distance between the reads. The average of these distances is then used as the final distance estimate. It is possible to get a negative distance estimate which happens when the paired information indicate that contigs overlap but for some reason could not be joined in the graph. Additional information about repeats being resolved using paired reads and scaffolded contigs is available as annotations on

the contig sequences and as summary in the report (see section 2.6.2).There are three types of annotations:

- **Scaffold** refers to the estimated gap region between two contigs where Ns are inserted. The region may have a negative size and therefore not contain any Ns.
- **Contigs joined** is when a repeat or another ambiguous structure in the graph was solved using paired reads, thus enabling the join of two contigs.
- **Alternatives excluded** refers to the exclusion of an unknown graph structure using paired reads which resulted in a join of two contigs.

2.4.4 Bubble resolution

Before the graph structure is converted to contig sequences, bubbles are resolved. As mentioned previously, a bubble is defined as a bifurcation in the graph where a path furcates into two nodes and then merge back into one. An example is shown in figure 2.50.



Figure 2.50: *A bubble caused by a heteroygous SNP or a sequencing error.*

In this simple case the assembler will collapse the bubble and use the variant that has the highest count of words. For a diploid genome with a heterozygous variant, there will a fifty-fifty distribution of reads on the two variants, and this means that the choice of one allele over the other will be arbitrary. If heterozygous variants are important, they can be identified after the assembly by mapping the reads back to the contig sequences and performing standard variant calling. For random sequencing errors, it is more straightforward; given a reasonable level of coverage, the erroneous variant will be suppressed.

Figure 2.51 shows an example of a data set where the reads have systematic errors. Some reads include five As and others have six. This is a typical example of the homopolymer errors seen with the 454 and Ion Torrent platforms.



Figure 2.51: *Reads with systematic errors.*

When these reads are assembled, this site will give rise to a bubble in the graph. This is not a problem in itself, but if there are several of these sites close together, the two paths in the graph will not be able to merge between each site. This happens when the distance between the sites is smaller than the word size used (see figure 2.52).

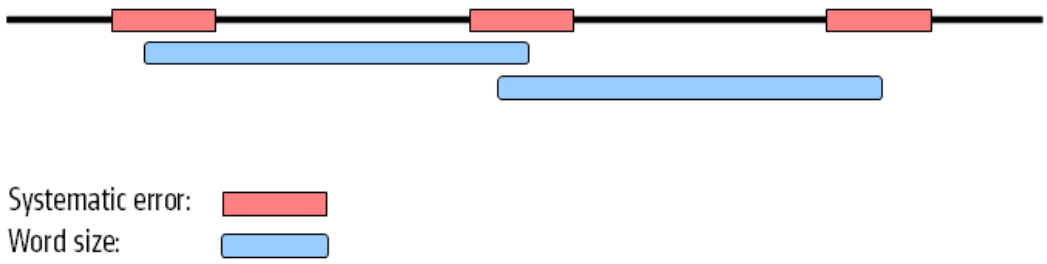


Figure 2.52: *Several sites of errors that are close together compared to the word size.*

In this case, the bubble will be very large because there are no complete words in the regions between the homopolymer sites, and the graph will look like figure 2.53.

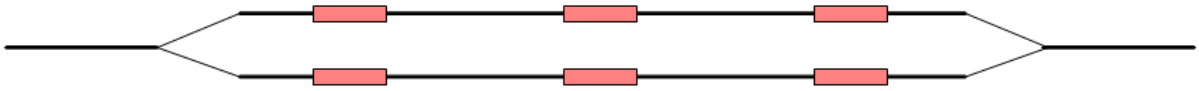


Figure 2.53: *The bubble in the grap gets very large.*

If the bubble is too large, the assembler will have to break it into several separate contigs instead of producing one single contig.

The maximum size of bubbles that the assembler should try to resolve can be set by the user. In the case from figure 2.53, a bubble size spanning the three error sites will mean that the bubble will be resolved (see figure 2.54).

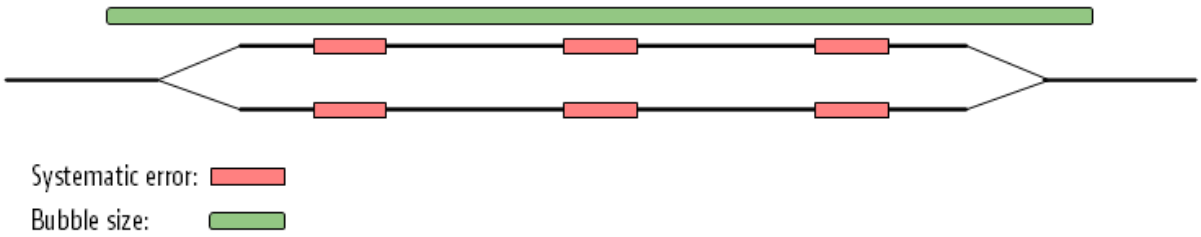


Figure 2.54: *The bubble size needs to be set high enough to encompass the three sites.*

The bubble size is especially important for reads generated by sequencing platforms yielding long reads with either systematic errors or a high error rate. In this case, a higher bubble size is recommended. Our benchmarks indicate that setting the bubble size at approximately twice the read length will produce a good result. But please use this as an advice for a starting point for testing different settings rather than a solid rule to apply at all times.

2.4.5 Converting the graph to contig sequences

The output of the assembly is not a graph but a list of contig sequences. When all the previous optimization and scaffolding steps have been performed, a contig sequence will produced for every non-ambiguous path in the graph. If the path cannot be fully resolved, Ns are inserted as an estimation of the distance between two nodes as explained in section 2.4.3.

2.4.6 Summary

So in summary, the de novo assembly algorithm goes through these stages:

- Make a table of the words seen in the reads.
- Build a *de Bruijn* graph from the word table.
- Use the reads to resolve the repeats in the graph.
- Use the information from paired reads to resolve larger repeats and perform scaffolding if necessary.
- Output resulting contigs based on the paths, optionally including annotations from the scaffolding step.

These stages are all performed by the assembler program.

2.4.7 Randomness in the results

A side-effect of the very compact data structures needed in order to keep the memory consumption low, is that the results will vary slightly from run to run, using the same data set. When counting the number of occurrences of a word, the assembler does not keep track of the exact number (which would consume a lot of memory) but uses an approximation which relies on some probability calculations. When using a multi-threaded CPU, the data structure is build in different ways for each run, and this means that the probability calculations for certain parts of the algorithm will be a bit different from run to run. This leads to differences in the results.

It should be noted that the differences are minor and will not affect the overall results. Keep in mind that whether you use CLC bio's assembler or other assemblers, there will never be one correct answer to the problem of de novo assembly. In this perspective, the small differences should not be considered a problem.

2.4.8 SOLiD data support in de novo assembly

SOLiD sequencing is done in color space. When viewed in nucleotide space this means that a single sequencing error changes the remainder of the read. An example read is shown in figure 2.55.



Figure 2.55: *How an error in color space leads to a phase shift and subsequent problems for the rest of the read sequence*

Basically, this color error means that C's become A's and A's become C's. Likewise for G's and T's. For the three different types of errors, we get three different ends of the read. Along with the correct reads, we may get four different versions of the original genome due to errors. So if SOLiD reads are just regarded in nucleotide space, we get four different contig sequences with jumps from one to another every time there is a sequencing error.

Thus, to fully accommodate SOLiD sequencing data, the special nature of the technology has to be considered in every step of the assembly algorithm. Furthermore, SOLiD reads are fairly short and often quite error prone. Due to these issues, we have chosen not to include SOLiD support in the first algorithm steps, but only use the SOLiD data where they have a large positive effect on the assembly process: when applying paired information.

2.4.9 De novo assembly parameters

To start the assembly:

Toolbox | **High-throughput Sequencing** ( ) | **De Novo Assembly** ( )

In this dialog, you can select one or more sequence lists or single sequences.

Click **Next** to set the parameters for the assembly. This will show a dialog similar to the one in figure 2.56.

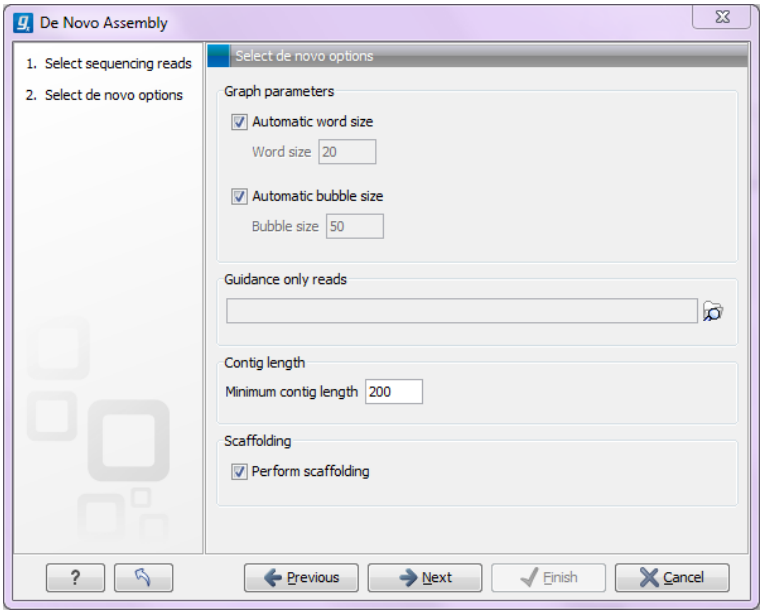


Figure 2.56: *Setting parameters for the assembly.*

At the top, you select the **Word size** and the **Bubble size** to be used. The principles of setting the word size are described in section 2.4.1. When using automatic calculation, you can see the word size in the **History** ( ) of the result files. Please note that the range of word sizes is 12-24 on 32-bit computers and 12-64 on 64-bit computers. The meaning of the bubble size parameter is explained in section 2.4.4. The bubble size used when the setting is automatic is 50 for reads shorter than 110 bp, and for longer reads it is the average read length. The value used is also recorded in the **History** ( ) of the result files.

The next option is to specify **Guidance only reads**. Only the pair information on these reads will be used, and the reads will only contribute in the scaffolding step. The construction of the word table and the graph will not be based on these reads. An example of a use case for this is SOLiD data which has a high error rate when used in base space. By using SOLiD for guidance only, it is possible to make use of the pair information without having the errors complicating the graph.

You can also specify the **Minimum contig length** when doing de novo assembly. Contigs below

this length will not be reported. The default value is 200 bp.

Finally, there is an option to **Perform scaffolding**. The scaffolding step is explained in greater detail in section 2.4.3. This will also cause scaffolding annotations to be added to the contig sequences (except when you also choose to Update contigs, see below).

When you click **Next**, you will see the dialog shown in figure 2.57

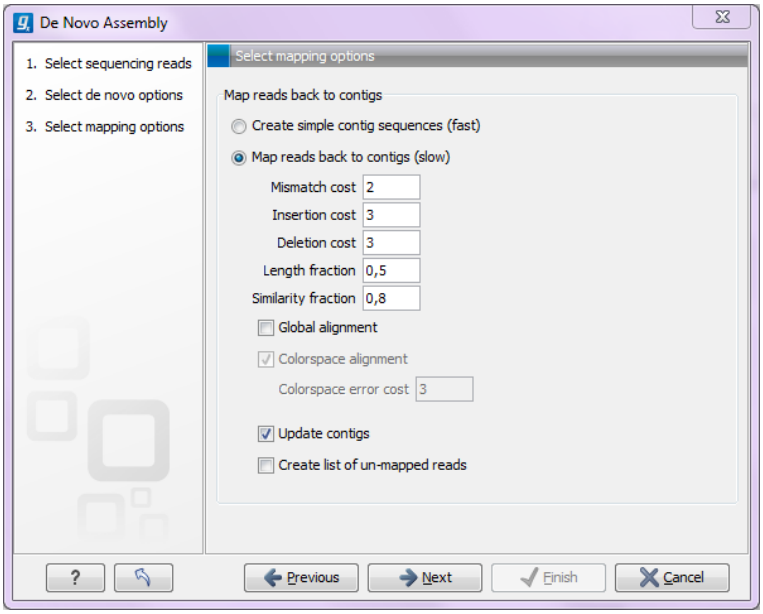


Figure 2.57: *Parameters for mapping reads back to the contigs.*

At the top, you choose whether a read mapping should be performed after the initial contig creation. If you choose to do that, you can specify the parameters for the read mapping. These are all explained in section 2.5.3.

At the bottom, you can choose to **Update contigs based on mapped reads**. This means that the original contig sequences produced from the de novo assembly will be updated to reflect the mapping of the reads (in most cases it will mean no change, but in some cases, the subsequent mapping step leads to new information). In effect, this means that all contig sequences in the output will be supported by at least one read mapped back. Note that if this option is selected, the contig lengths may get below the threshold specified in figure 2.56 because this threshold is applied to the original contig sequences. If the **Update contigs based on mapped reads** option is not selected, the original contig sequences from the assembler will be preserved completely also in situations where the reads that are mapped back do not support the contig sequences.

If you update the contigs, it means that scaffolding annotations will not be added to the contig sequences (since the contig sequences may change in this process it is not possible to place these annotations correctly). This in turn affects the de novo assembly report which will not have statistics about scaffolding when the update contigs option is selected.

2.4.10 De novo assembly report

In the last dialog of the de novo assembly, you can choose to create a report of the results (see figure 2.58).

1 Summary de novo report

1.1 Nucleotide distribution

Nucleotide	Count	Frequency
Adenine (A)	1.113.919	24,5%
Cytosine (C)	1.142.129	25,2%
Guanine (G)	1.157.663	25,5%
Thymine (T)	1.118.847	24,6%
Any nucleotide (N)	6.409	0,1%

1.2 Contig measurements

	Length
N75	41.694
N50	80.414
N25	132.325
Minimum	202
Maximum	191.299
Average	32.421
Count	140
Total	4.538.967

Figure 2.58: *Creating a de novo assembly report.*

The report contains the following information when both scaffolding and read mapping is performed:

Nucleotide distribution . This includes Ns when scaffolding has been performed.

Contig measurements . This section includes statistics about the number and lengths of contigs. When scaffolding is performed and the update contigs option is not selected, there will be two separate sections with these numbers: one including the scaffold regions with Ns and one without these regions.

N25, N50 and N75 . The N25 contig set is calculated by summarizing the lengths of the biggest contigs until you reach 25 % of the total contig length. The minimum contig length in this set is the number that is usually used to report the N25 value of a de novo assembly. The same goes with N50 and N75 which are the 50 % and 75 % of the total contig length, respectively.

Minimum, maximum and average . This refers to the contig lengths.

Count The total number of contigs.

Total The number of bases in the result. This can be used for comparison with the estimated genome size to evaluate how much of the genome sequence is included in the assembly.

Contig length distribution . A graph showing the number of contigs of different lengths.

Accumulated contig lengths . This shows the summarized contig length on the y axis and the number of contigs on the x axis, with the biggest contigs ranked first. This answers the question: how many contigs are needed to cover e.g. half of the genome.

Mapping information . The rest of the sections provide statistics from the read mapping (if performed). These are explained in section 2.6.2.

2.5 Map reads to reference

This section describes how to map a number of sequence reads to one or more reference sequences. When the reads come from a set of known sequences with relatively few variations, read mapping is often the right approach to assembling the data. The result of mapping reads to a reference is a "mapping" or a "mapping table" which is the term we use for an alignment of reads against a reference sequence.

2.5.1 Starting the read mapping

To start the read mapping:

Toolbox | **High-throughput Sequencing ( )** | **Map Reads to Reference ( )**

In this dialog, select the sequences or sequence lists containing the sequencing data. Note that the reference sequences should be selected in the next step.

When the sequences are selected, click **Next**, and you will see the dialog shown in figure 2.59.

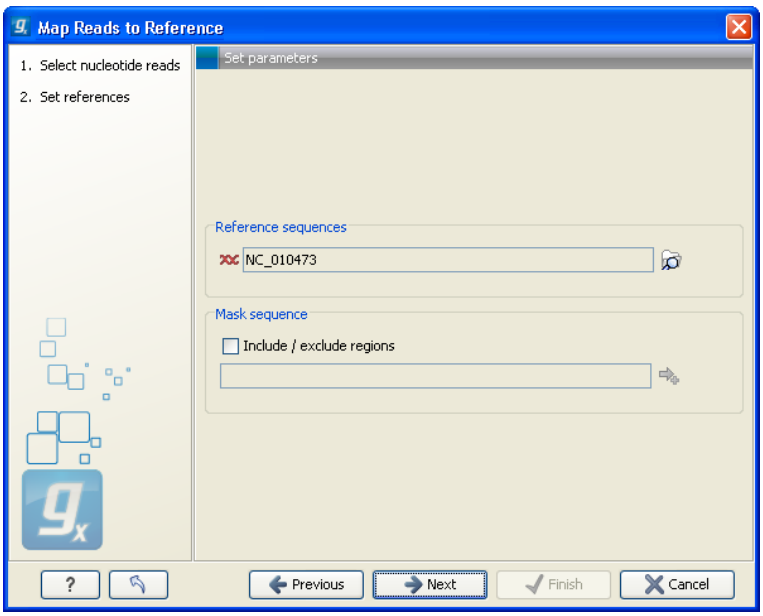


Figure 2.59: *Specifying the reference sequences and masking.*

At the top you select one or more reference sequences by clicking the **Browse and select element** ( ) button. You can select either single sequences or a list of sequences as reference sequences. When multiple reference sequence are used, the result of the mapping will be a mapping table with one entry per reference sequence.

2.5.2 Including or excluding regions (masking)

The next part of the dialog lets you *mask* the reference sequences. Masking refers to a mechanism where parts of the reference sequence are not considered in the mapping. This

can be extremely useful for example when mapping human data, where more than 50 % of the sequence consists of repeats. Note that you should be careful masking all the repeat regions if your sequenced data contains the repeats. If you do that, some of the reads that would have matched a masked repeat region perfectly may be placed wrongly at another position even with a less-perfect match and lead to wrong results.

In order to mask e.g. repeat regions when doing read mapping, the repeat regions have to be annotated on the reference sequences.

Because the masking is based on annotations, any kind of annotations can be selected for masking. This means that you can choose to e.g. only map against the genes in the genome, or only the exons. As long as the reference sequences contain the relevant information in the form of annotations, it can be masked.

To mask a reference sequence, first click the **Include / exclude regions** checkbox, and second click the **Select annotation type** ( ) button.

This will bring up a dialog with all the annotation types of the reference sequences listed to the left. Select one or more annotation types and click **Add** ( ) button. Then select at the bottom whether you wish to **Include** or **Exclude** these annotation types. If you include, it means that only the regions covered by the selected type of annotations will be used in the read mapping. If you exclude, it means that all of the reference sequences except the regions covered by the selected type of annotations will be used in the read mapping.

You can see an example in figure 2.60.

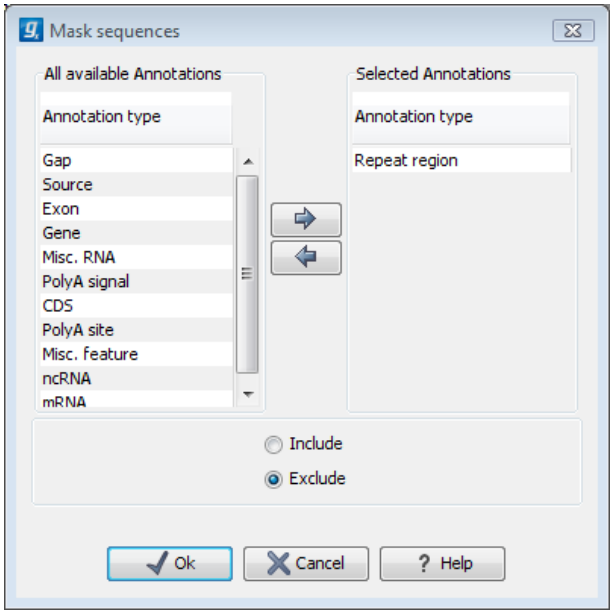


Figure 2.60: *Masking for repeats. The repeat region annotation type is selected and excluded in the mapping.*

2.5.3 Mapping parameters

Click **Next** to set the parameters for the mapping. This will show a dialog similar to the one in figure 2.61.

In order to understand what is going on here, a little explanation is needed: The *CLC Genomics*

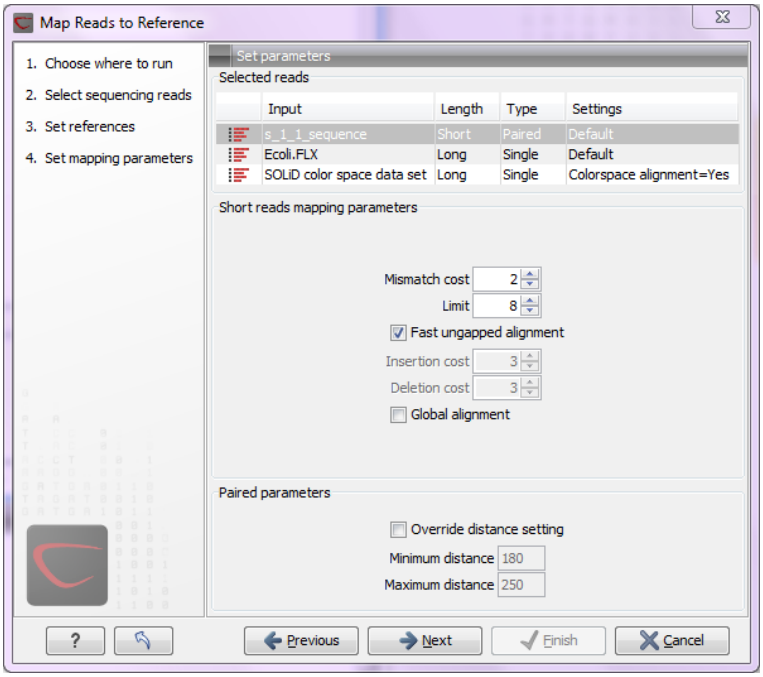


Figure 2.61: *Setting parameters for mapping.*

Workbench supports assembly of mixed data sets. This means that you can assemble and map both short reads, long reads, single reads and paired reads in one go. This makes it easy to combine the information from different sources, but it also makes the parameters a little more complex, because each data set may need its own parameters.

At the top of the dialog shown in figure 2.61 is a table of all the sequence lists that were chosen in the first step. Clicking one of the lists shows the parameters that will be used this particular data set. Note that the Workbench automatically categorizes each of the lists into short/long reads and single/paired. Reads are considered short when they are less than 56 nucleotides, unless the data is in color space where the long reads algorithm is always applied regardless of the read length.

In the example in figure 2.61, you first adjust the parameters for the data set called `s_1_1_sequence` and then click the next data set called `Ecoli.FLX` and adjust parameters for this data set (as shown in figure 2.62). Because these data sets are different in terms of length and single/paired content, you have to set the parameters for each one. If you had two similar data sets, you could select both of them in the table and then change the settings for both.

Each of the parameters are described below:

Common parameters for short and long reads

Three parameters are identical for both short and long reads:

Mismatch cost The cost of a mismatch between the read and the reference sequence.

Insertion cost The cost of an insertion in the read (causing a gap in the reference sequence)

Deletion cost The cost of having a gap in the read.

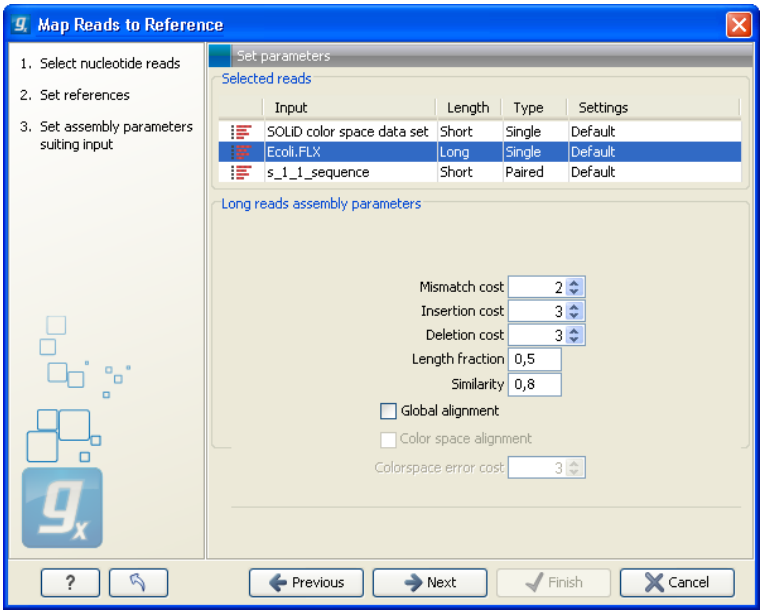


Figure 2.62: *Setting parameters for the mapping.*

Global alignment Per default, the reads are aligned locally, allowing a number of "unaligned" nucleotides at the ends of the read. By selecting the global alignment option, you force the whole read to be aligned to the reference. Mismatches at the ends will then count as any other mismatch.

The score for a match is always 1.

Short reads parameters

For short reads, there is a threshold that determines whether the read should be included in the mapping:

Limit The relationship between the length of the read and the score. A limit of 8 (which is default) means that the total score for the alignment has to be more than the length of the read minus 8. This is explained in detail with examples below. This means that with the default costs, two mismatches, two deletions or two insertions will be allowed. If no mismatches or gaps are involved, it means that up to 8 unaligned nucleotides in the ends would be allowed. For very short reads, a limit of 5 could typically be used instead, allowing up to one mismatch and two unaligned nucleotides in the ends (or no mismatches and five unaligned nucleotides).

Given a certain quality threshold, it is possible to guarantee that all optimal ungapped alignments are found for each read. Alignments of short reads to reference sequences usually contain no gaps, so the short read assembly operates with a strict scoring threshold to allow the user to specify the amount of errors to accept.

With other short read mapping programs like Maq and Soap, the threshold is specified as the number of allowed mismatches. This works because those programs do global alignment. For local alignments it is a little more complicated.

The default alignment scoring scheme for short reads is +1 for matches and -2 for mismatches. The limit for accepting an alignment is given as the alignment score relative to the read length. For example, if the score limit is 8 below the length, up to two mismatches are allowed as well as two ending nucleotides not assembled (remember that a mismatch costs 2 points, but when there is a mismatch, a potential match is also lost). Alternatively, with one mismatch, up to 5 unaligned positions are allowed. Or finally, with no mismatches, up to 8 unaligned positions are allowed. See figure 2.63 for examples. The default setting is exactly this limit of 8 below the length.

```
CGTATCAATCGATTACGCTATGAATG          CGTATCAATCGATTACGCTATGAATG
   ||||||||||||||||||||      20          ||||||||||||||||||      19
   ATCAATCGATTACGCTATGA                TTCAATCGATTACGCTATGA

CGTATCAATCGATTACGCTATGAATG          CGTATCAATCGATTACGCTATGAATG
   ||||||||  ||||||||||||      17          |||||||  ||||||||||||      16
   ATCAATCGGTTACGCTATGA                TTCAATCGGTTACGCTATGA

CGTATCAATCGATTACGCTATGAATG          CGTATCAATCGATTACGCTATGAATG
    |||||||  ||||||||||       15          |||||  ||  ||||||||||       14
   CTCAATCGGTTACGCTATGA                ATCAACCGGTTACGCTATGA

CGTATCAATCGATTACGCTATGAATG          CGTATCAATCGATTACGCTATGAATG
    |||||||  ||||  ||||||       13          ||||||||||  ||||          12
   TTCAATCGGTTACCCTATGA                ATCAATCGATTGCGCTCTTT

CGTATCAATCGATTACGCTATGAATG          CGTATCAATCGATTACGCTATGAATG
    |||||||  ||||  |||||       12                ||||||||||||          12
   TTCAATCGGTTACCCTATGC                AGCTATCGATTACGCTCTTT
```

Figure 2.63: Examples of ungapped alignments allowed for a 20 bp read with a scoring limit of 8 below the length using the default scoring scheme. The scores are noted to the right of each alignment. For reads this short, a limit of 5 would typically be used instead, allowing up to one mismatch and two unaligned nucleotides in the ends (or no mismatches and five unaligned nucleotides).

Note that if you choose to do global alignment, the default setting means that up to two mismatches are allowed (because "unaligned positions" at the ends are counted as mismatches as well).

The match score is always +1. If the mismatch cost is changed, the default score limit will also change to:

$$score\ limit = 3 \times (1 + mismatch\ cost) - 1$$

The default mismatch *score* of -2 equals a mismatch *cost* of 2 and a score limit of 8 below the read length, as stated above. For any mismatch cost, the default score limit allows any alignment scoring strictly better than 3 mismatches.

The maximum score limit also depends on the mismatch cost:

$$max\ score\ limit = 4 \times (1 + mismatch\ cost) - 1$$

Gapped alignment is also allowed for short reads. Contrary to ungapped alignments, it is very difficult to guarantee that all gapped alignments of a certain quality are found. The scoring limit discussed above applies to both gapped and ungapped alignments and there is a guarantee that there are no ungapped exceeding the limit, but there is is no such guarantee for gapped

alignments. This being said, the program does a good effort to find the best gapped alignments and usually succeeds. Besides the limit, there are also two options related to mapping of color space data (from SOLiD systems). If you do not have color space data, these will be disabled and are not relevant.

Color space alignment This will determine if mapping is to be performed in color space. This is strongly recommended for SOLiD data.

Color error cost The cost of a color error.

An example of a color space data set is shown in figure 2.64.

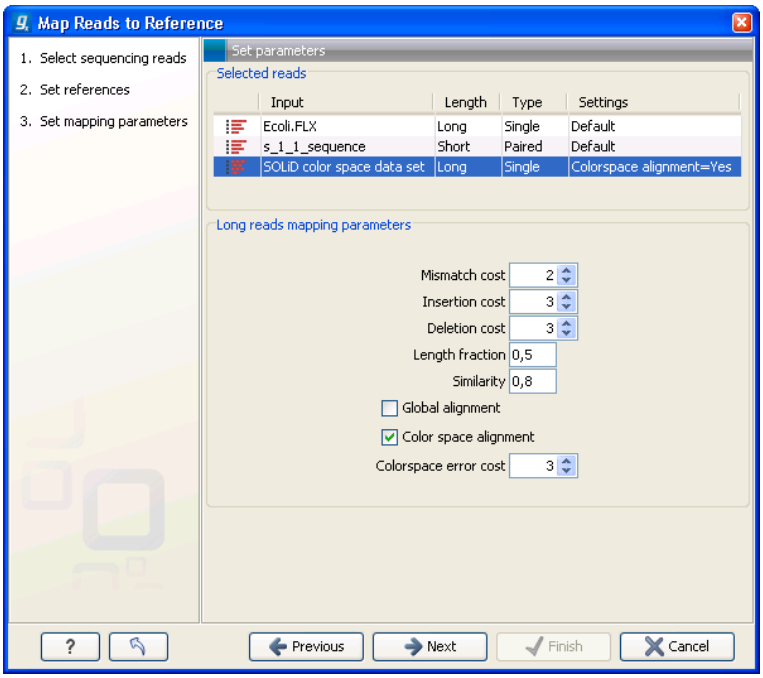


Figure 2.64: *Setting parameters for the mapping.*

For more details about this, please see section 2.8 which explains how color space mapping is performed in greater detail.

Long reads parameters

For long reads, the read mapping as two stages: First, the optimal alignment of the read is found, based on the costs specified above (e.g. to favor mismatches over indels). Second, a filtering process determines whether this match is good enough for the read to be included in the alignment. The filtering threshold is determined by two fractions:

Length fraction Set minimum length fraction of a read that must match the reference sequence. Setting a value at 0.5 means that at least half the read needs to match the reference sequence for the read to be included in the final mapping.

Similarity Set minimum fraction of identity between the read and the reference sequence. If you want the reads to have e.g. at least 90% identity with the reference sequence in order to be included in the final mapping, set this value to 0.9. Note that the similarity fraction does

not apply to the whole read; it relates to the Length fraction. With the default values, it means that *at least 50 % of the read must have at least 90 % identity*.

Paired reads

At the bottom you can specify how **Paired reads** should be handled. You can read more about how paired data is imported and handled in section 2.1.8. If the sequence list used as input contains paired reads, this option will automatically be shown - if it contains single reads, this option will not be shown.

For the paired reads, you can specify a distance interval between the two sequences in a pair. This will be used to determine how far it can expect the two reads to be from each other. This value includes the length of the read sequence as well (not just the distance in between). If you set this value very precisely, you will miss some of the opportunities to detect genomic rearrangements as described in section 2.9.3. On the other hand, a precise distance interval will give a more accurate assembly in the places where there are not significant variation between the sequencing data and the reference sequence.

We recommend running the detailed mapping report (see section 2.6.1) and check that the paired distances reported show a nice distribution and that not too many pairs are broken.

The approach taken for determining the placement of read pairs is the following:

- First, all the optimal placements for the two individual reads are found.
- Then, the allowed placements according to the paired distance interval are found.
- If both reads can be placed independently but no pairs satisfies the paired criteria, the reads are treated as independent and not marked as a pair.
- If only one pair of placements satisfy the criteria, the reads are placed accordingly and marked as uniquely placed even if either read may have multiple optimal placements.
- If several placements satisfy the paired criteria, the read is treated as a "non-specific match" (see section 2.5.4 for more information.)

By default, mapping is done with local alignment of reads to a set of reference sequences. The advantage of performing local alignment instead of global alignment is that the ends are automatically removed if there are sufficiently many sequencing errors there. If the ends of the reads contain vector contamination or adapter sequences, local alignment is also desirable. Note that the aligned region has to be greater than the length threshold set.

2.5.4 General mapping options

When you click **Next**, you will see the dialog shown in figure 2.65

At the top, you can choose to **Add conflict annotations** to the consensus sequence. Note that there may be a huge number of annotations and that it may give a visually cluttered overview of the mapping of the reads. A subtle detail about the annotations - if you add annotations, you will be able to see *resolved* conflicts in the table view () of the mapping (e.g. if you edit the bases after mapping). If there are no annotations, only the non-resolved conflicts are shown.

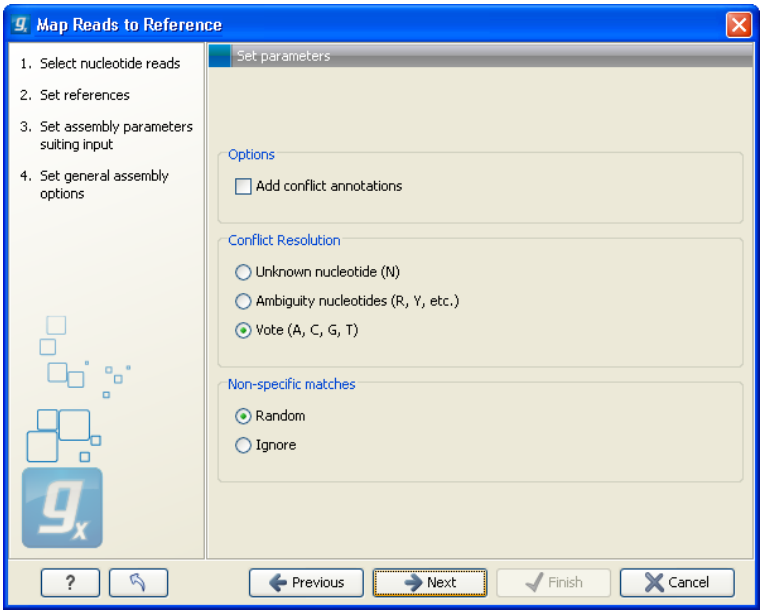


Figure 2.65: *Conflict resolution and annotation.*

If there is a conflict between reads, i.e. a position where there is disagreement about which base is correct, you can specify how the consensus sequence should reflect the conflict:

- **Vote (A, C, G, T).** The conflict will be solved by counting instances of each nucleotide and then letting the majority decide the nucleotide in the consensus. In case of equality, ACGT are given priority over one another in the stated order.
- **Unknown nucleotide (N).** The consensus will be assigned an 'N' character in all positions with conflicts.
- **Ambiguity nucleotides (R, Y, etc.).** The consensus will display an ambiguity nucleotide reflecting the different nucleotides found in the reads. For an overview of ambiguity codes, see Appendix **??**.

At the bottom of the dialog, you can specify how **Non-specific matches** should be treated. The concept of Non-specific matches refers to a situation where a read aligns at more than one position. In this case you have two options:

- **Random**. This will place the read in one of the positions randomly.
- **Ignore**. This will not include the read in the final mapping.

Note that a read is only considered non-specific when the read matches equally well at several alignment positions. If there are e.g. two possible alignment positions and one of them is a perfect match and the other involves a mismatch, the read is placed at the position with the perfect match and it is not marked as a non-specific match.

2.5.5 Assembly reporting options

Click **Next** lets you choose how the output of the assembly should be reported (see figure 2.66).

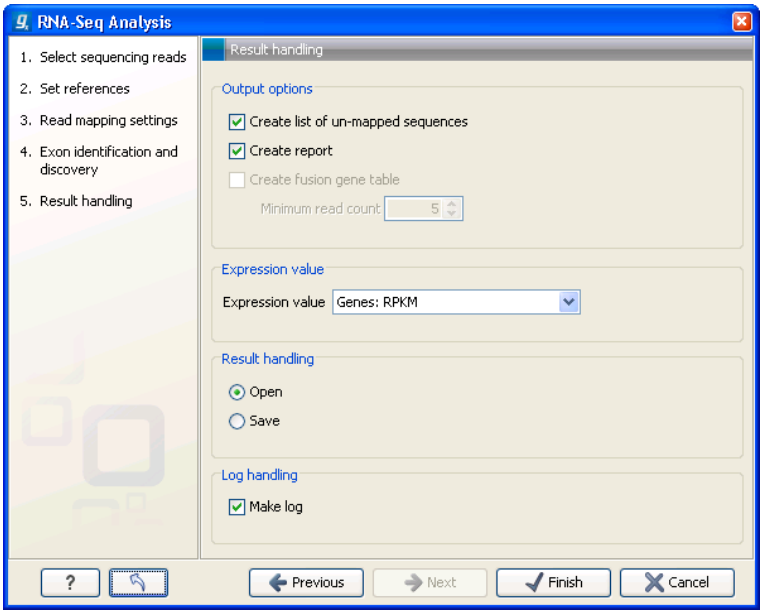


Figure 2.66: *Assembly reporting options.*

First, you can choose to save or open the results, and if you wish to see a log of the process (see section **??**).

No matter what you choose, you will always see the visual read mapping, but in addition you have two extra output options:

- **Create Report**. This will generate a summary report as described in section 2.6.2.
- **Create list of non-mapped sequences**. This will put all the reads that could not be mapped to the reference into a sequence list.

If you have used more than one reference sequence, the Workbench creates a table which makes it easier to get an overview. The table includes this information:

1. Length of reference sequence
2. Length of consensus sequence
3. Number of reads
4. Average coverage
5. Total number of conflicts

Double-clicking one of the rows in the table will open the corresponding mapping. Furthermore, you can select a number of rows and click the **Open Consensus** at the bottom of the table. That will open a sequence list of all the consensus sequences of the selected rows.

Clicking **Finish** will start the mapping. See section **??** for general information about viewing and editing the resulting mappings. For special information about genome-size mapping, see section 2.9.

2.6 Mapping reports

You can create two kinds of reports regarding read mappings and de novo assemblies: *First*, you can choose to generate a summary report about the mapping process itself (see sections **??** and 2.5.5). This report is described in section 2.6.2 below. *Second*, you can generate a detailed statistics report after the mapping or assembly has finished. This report is useful if you want to generate statistics across results made in different processes, and it generates more detailed statistics than the summary mapping report. This report is described below.

2.6.1 Detailed mapping report

To create a detailed mapping report:

Toolbox | **High-throughput Sequencing ( )** | **Create Detailed Mapping Report ( )**

This opens a dialog where you can select mapping results ( )/ ( ) or RNA-Seq analysis results ( ) (see sections 2.4 and 2.5 for information on how to create a contig and section 2.14 for information on how to create RNA-Seq analysis results).

Clicking **Next** will display the dialog shown in figure 2.67

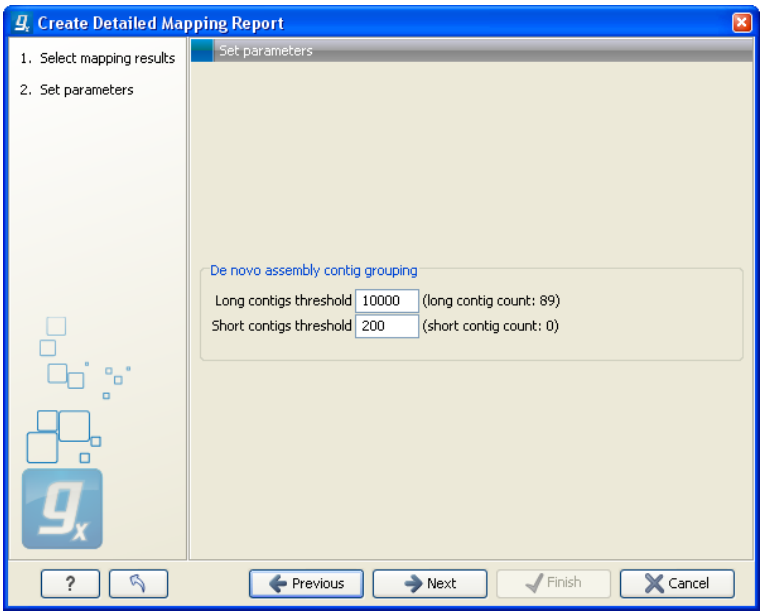


Figure 2.67: *Parameters for mapping reports.*

The first option is to set thresholds for grouping long and short contigs. The grouping is used to show statistics like number of contigs, mean length etc for the contigs in each group. This is only relevant for de novo assemblies. Note that the de novo assembly in the *CLC Genomics Workbench* per default only reports contigs longer than 200 bp (this can be changed when running the assembly).

Click **Next** to select output options as shown in figure 2.68

Per default, an overall report will be created as described below. In addition, by checking **Create table with statistics for each reference** you can create a table showing detailed statistics for each reference sequence (for de novo results the contigs act as reference sequences, so it will be one row per contig). The following sections describe the information produced.

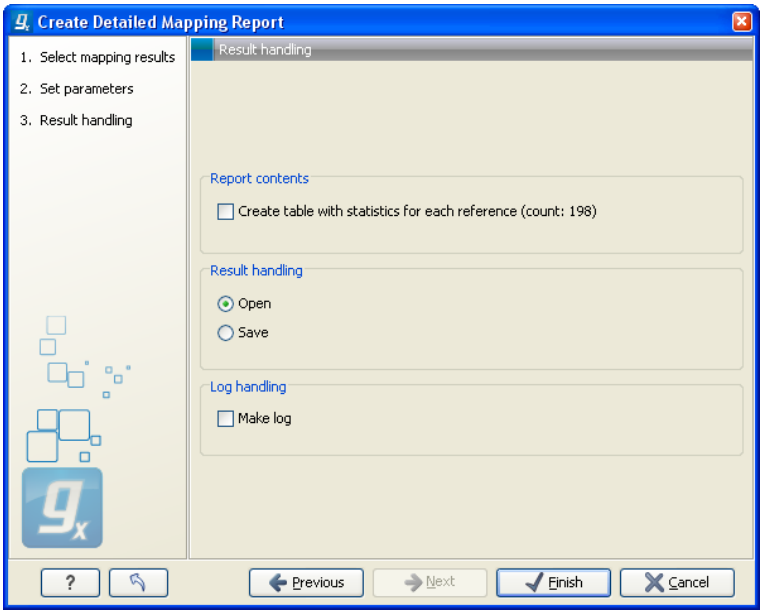


Figure 2.68: *Optionally create a table with detailed statistics per reference.*

Reference sequence statistics

For reports on results of read mapping, section two concerns the reference sequences. The reference identity part includes the following information:

Reference name The name of the reference sequence.

Reference Latin name The reference sequence's Latin name.

Reference description Description of the reference.

If you want to inspect and edit this information, right-click the reference sequence in the contig and choose **Open This Sequence** and switch to the **Element info** ( ) tab (learn more in section **??**). Note that you need to create a new report if you want the information in the report to be updated. If you update the information for the reference sequence within the contig, you should know that it doesn't affect the original reference sequence saved in the **Navigation Area**.

The next part of the report reports coverage statistics including GC content of the reference sequence. Note that coverage is reported on two levels: including and excluding zero coverage regions. In some cases, you do not expect the whole reference to be covered, and only the coverage levels of the covered parts of the reference sequence are interesting. On the other hand, if you have sequenced the full genome that you use as reference, the overall coverage is probably the most relevant number (i.e. including zero coverage regions).

A position on the reference is counted as "covered" when at least one read is aligned to it. Note that unaligned ends (faded nucleotides at the ends) that are produced when mapping using local alignment do not contribute to the coverage. In the example shown in figure 2.69, there is a region of zero coverage in the middle and one time coverage on each side. Note that the gaps to the very right are within the same read which means that these two positions on the reference sequence are still counted as "covered".

The identity section is followed by some statistics on the zero-coverage regions; the number, minimum and maximum length, mean length, standard deviation, total length and a list of the

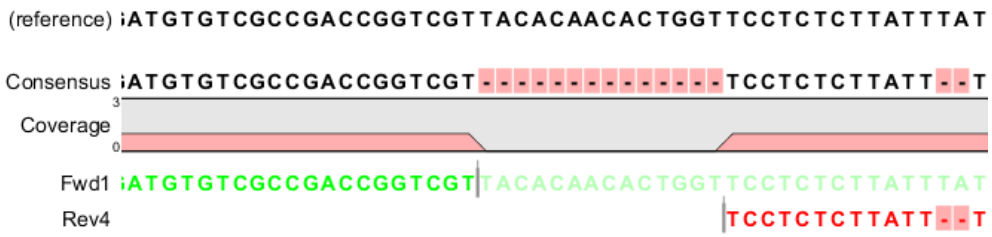


Figure 2.69: *A region of zero coverage in the middle and one time coverage on each side. Note that the gaps to the very right are within the same read which means that these two positions on the reference sequence are still counted as "covered".*

regions. If there are too many regions, they will not all be listed in the report (if there are more than 20, only the first 10 are reported).

Next follow two bar plots showing the distribution of coverage with coverage level on the x-axis and number of contig positions with that coverage on the y-axis. An example is shown in figure 2.70.

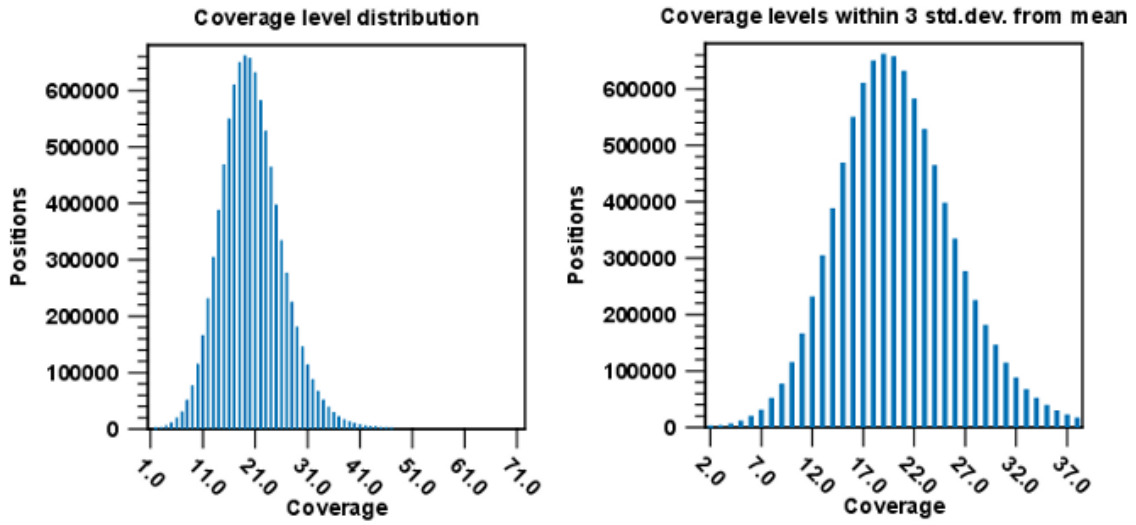


Figure 2.70: *Distribution of coverage - to the left for all the coverage levels, and to the right for coverage levels within 3 standard deviations from the mean.*

The graph to the left shows all the coverage levels, whereas the graph to the right shows coverage levels within 3 standard deviations from the mean. The reason for this is that for complex genomes, you will often have a few regions with extremely high coverage which will affect the resolution of the graph, making it impossible to see the coverage distribution for the majority of the contigs. These coverage outliers are excluded when only showing coverage within 3 standard deviations from the mean. Note that zero-coverage regions are not shown in the graph but reported in text below (this information is also in the zero-coverage section). Below the second coverage graph there are some statistics on the data that is outside the 3 standard deviations.

One of the biases seen in sequencing data concerns GC content. Often there is a correlation between GC content and coverage. In order to investigate this correlation, the report includes a graph plotting coverage against GC content (see figure 2.71). Note that you can see the GC content for each reference sequence in the table above.

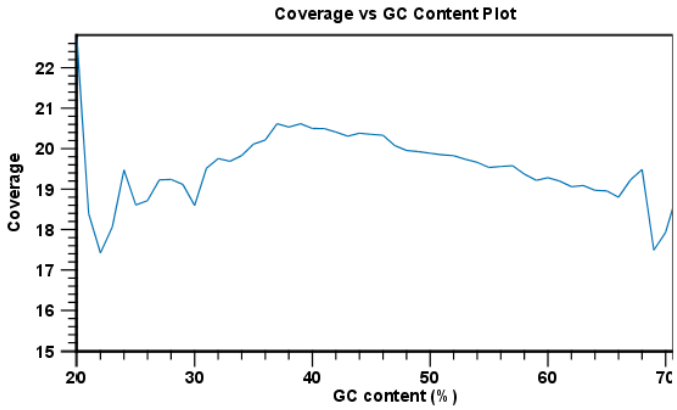


Figure 2.71: *The plot displays, for each GC content level (0-100 %), the mean read coverage of 100bp reference segments with that GC content.*

The plot displays, for each GC content level (0-100 %), the mean read coverage of 100bp reference segments with that GC content.

At the end follows statistics about the reads which are the same for both reference and de novo assembly (see section 2.6.1 below).

Contig statistics for de novo assembly

After the summary there is a section about the contig lengths. For each set of contigs, you can see the number of contigs, minimum, maximum and mean lengths, standard deviation and total contig length (sum of the lengths of all contigs in the set). The contig sets are:

N25 contigs The N25 contig set is calculated by summarizing the lengths of the biggest contigs until you reach 25 % of the total contig length. The minimum contig length in this set is the number that is usually used to report the N25 value of a de novo assembly.

N50 This measure is similar to N25 - just with 50 % instead of 25 %. This is probably the most well-known measure of de novo assembly quality - it is a more informative way of measuring the lengths of contigs.

N75 Similar to the ones above, just with 75 %.

All contigs All contigs that were selected.

Long contigs This contig set is based on the threshold set in the dialog in figure 2.67.

Short contigs This contig set is based on the threshold set in the dialog in figure 2.67. Note that the de novo assembly in the *CLC Genomics Workbench* per default only reports contigs longer than 200 bp.

Next follow two bar plots showing the distribution of coverage with coverage level on the x-axis and number of contig positions with that coverage on the y-axis. An example is shown in figure 2.72.

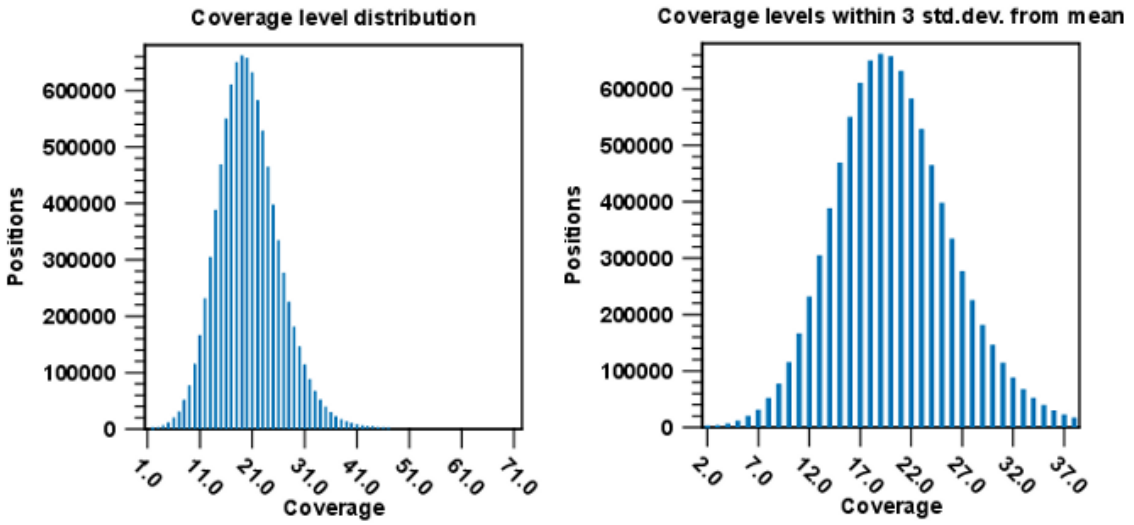


Figure 2.72: *Distribution of coverage - to the left for all the coverage levels, and to the right for coverage levels within 3 standard deviations from the mean.*

The graph to the left shows all the coverage levels, whereas the graph to the right shows coverage levels within 3 standard deviations from the mean. The reason for this is that for complex genomes, you will often have a few regions with extremely high coverage which will affect the resolution of the graph, making it impossible to see the coverage distribution for the majority of the contigs. These coverage outliers are excluded when only showing coverage within 3 standard deviations from the mean. Below the second coverage graph there are some statistics on the data that is outside the 3 standard deviations. At the end follows statistics about the reads which are the same for both reference and de novo assembly (see section 2.6.1 below).

Read statistics

This section contains simple statistics for all mapped reads, non-specific matches (reads that match more than place during the assembly), non-perfect matches and paired reads. **Note!** Paired reads are counted as two, even though they form one pair. The section on paired reads also includes information about paired distance and counts the number of pairs that were broken due to:

Wrong distance When starting the mapping, a distance interval is specified. If the reads during the mapping are placed outside this interval, they will be counted here.

Mate inverted If one of the reads has been matched as reverse complement, the pair will be broken (note that the pairwise orientation of the reads is determined during import).

Mate on other contig If the reads are placed on different contigs, the pair will also be broken.

Mate not matched If only one of the reads match, the pair will be broken as well.

Below these tables follow two graphs showing distribution of paired distances (see figure 2.73) and distribution of read lengths. Note that the distance includes both the read sequence and the insert between them as explained in section 2.1.8.

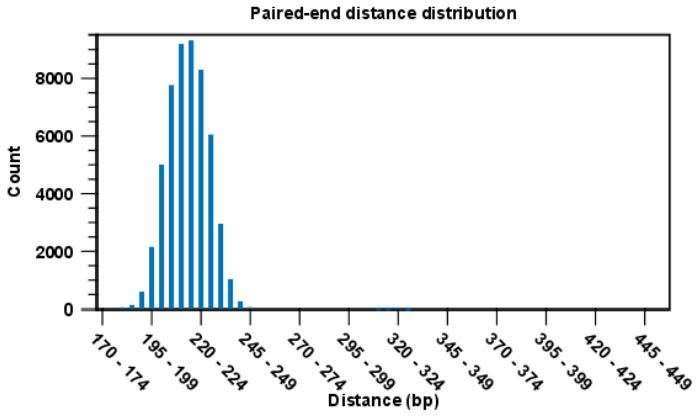


Figure 2.73: *A bar plot showing the distribution of distances between intact pairs.*

2.6.2 Summary mapping report

If you choose to create a report as part of the read mapping (see section 2.5.5), this report will summarize the results of the mapping process. An example of a report is shown in figure 2.74

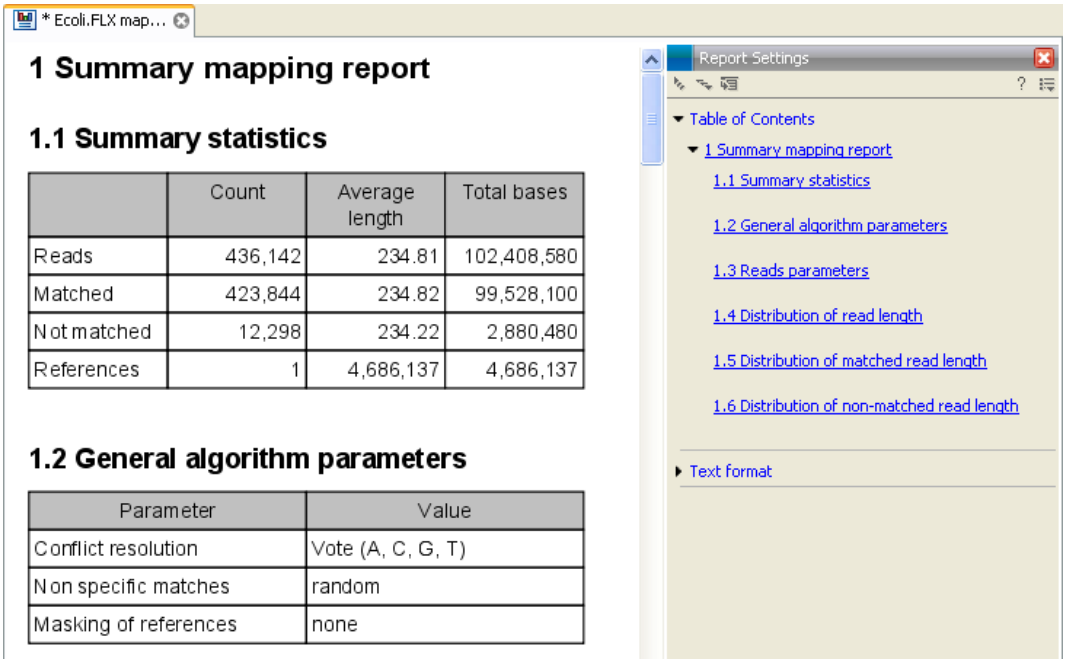


Figure 2.74: *The summary mapping report.*

The information included in the report is:

- **Summary statistics**. A summary of the mapping statistics:
  - **Reads**. The number of reads and the average length.

  - **Mapped**. The number of reads that are mapped and their average length.
  - **Not mapped**. The number of reads that do not map and their average length.
  - **References**. Number of reference sequences.

- **Parameters**. The settings used are reported for the process as a whole and for each sequence list used as input.
- **Distribution of read length**. For each sequence length, you can see the number of reads and the distribution in percent. This is mainly useful if you don't have too much variance in the lengths as you have in e.g. Sanger sequencing data.
- **Distribution of matched reads lengths**. Equivalent to the above, except that this includes only the reads that have been matched to a contig.
- **Distribution of non-matched reads lengths**. Show the distribution of lengths of the rest of the sequences.

You can copy the information from the report by selecting in the report and click **Copy** (). You can also export the report in Excel format.

2.7 Mapping table

When several reference sequence are used or you are performing de novo assembly with the reads mapped back to the contig sequences, (see sections **??** and 2.5.5), all your mapping data will be accessible from a table (). It means that all the individual mappings are treated as one single file to be saved in the **Navigation Area** as a table.

An example of a mapping table for a *de novo* assembly is shown in figure 2.75.

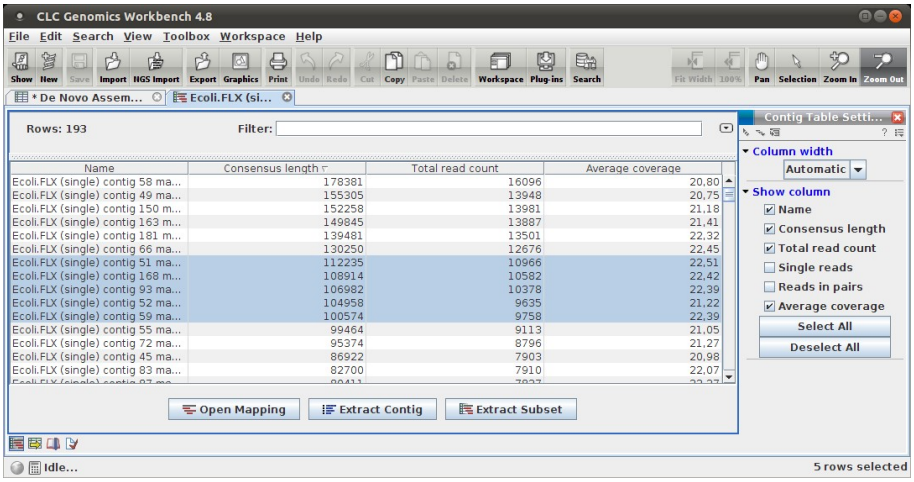


Figure 2.75: *The mapping table.*

The information included in the table is:

- **Name**. When mapping reads to a reference, this will be the name of the reference sequence.
- **Length of consensus sequence**. The length of the consensus sequence. Subtracting this from the length of the reference will indicate how much of the reference that has not been covered by reads.

- **Number of reads**. The number of reads. Reads hitting multiple places on different reference sequences are placed according to your input for **Non-specific matches**
- **Average coverage**. This is simply summing up the bases of the aligned part of all the reads divided by the length of the reference sequence.

For read mapping, there is more information taken from the reference sequence used as input. An example of a contig table produced by mapping reads to a reference is shown in figure 2.76.

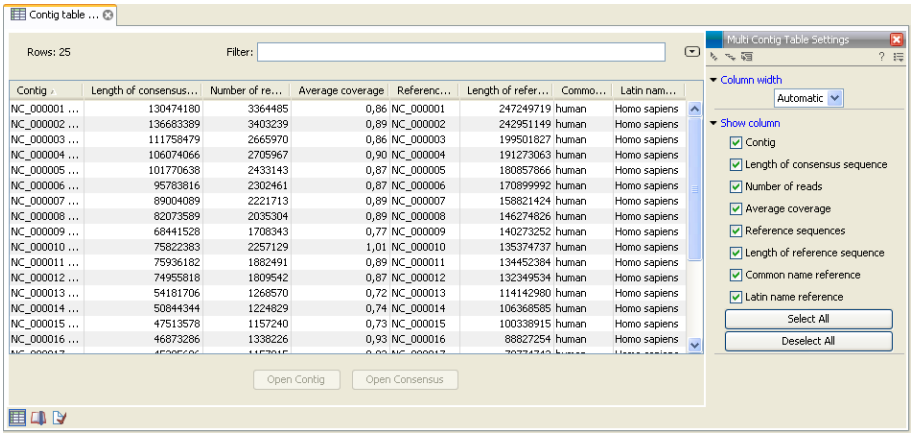


Figure 2.76: *The contig table.*

Besides the information that is also in the *de novo* table, there is information about name, common name and Latin name of each reference sequence.

At the bottom of the table, there are two buttons which apply to the rows that you select (press Ctrl + A - ⌘ +A on Mac - to select all):

- **Open Mapping**. Simply opens the read mapping for visual inspection. You can also open one mapping simply by double-clicking in the table.
- **Open Consensus/Open Contig**. Creates a sequence list of all the consensus sequences. This can be used for further analysis or exported ( ) in e.g. fasta format. For de novo assembly results, it is the contig sequences that are opened.
- **Extract Subset**. Creates a new mapping table with the mappings that you have selected.

You can copy the textual information from the table by selecting in the table and click **Copy** ( ). This can then be pasted into e.g. Excel. You can also export the table in Excel format.

2.8 Color space

2.8.1 Sequencing

The SOLiD sequencing technology from Applied Biosystems is different from other sequencing technologies since it does not sequence one base at a time. Instead, two bases are sequenced at a time in an overlapping pattern. There are 16 different dinucleotides, but in the SOLiD technology, the dinucleotides are grouped in four carefully chosen sets, each containing four dinucleotides. The colors are as follows:

Base 1	Base 2			
	A	C	G	T
A	●	●	●	●
C	●	●	●	●
G	●	●	●	●
T	●	●	●	●

Notice how a base and a color uniquely defines the following base. This approach can be used to deduce a whole sequence from the initial nucleotide and a series of colors. Here is a sequence and the corresponding colors.

Sequence T A C T C C A T G C A
Colors ● ● ● ● ● ● ● ● ● ●

The colors do not uniquely define the sequence. Here is another sequence with the same list of colors:

Sequence A T G A G G T A C G T
Colors ● ● ● ● ● ● ● ● ● ●

But if the first nucleotide is known, the colors do uniquely define the remaining sequence. This is exactly the strategy used in SOLiD sequencing: The first nucleotide is known from the primer used, and the remaining nucleotides are deduced from the colors.

2.8.2 Error modes

As with other sequencing technologies, errors do occur with the SOLiD technology. If a single nucleotide is changed, two colors are affected since a single nucleotide is contained in two overlapping dinucleotides:

Sequence T A C T C C A T G C A
Colors ● ● ● ● ● ● ● ● ● ●

Sequence T A C T C C A A G C A
Colors ● ● ● ● ● ● ● ● ● ●

Sometimes, a wrong color is determined at a given position. Due to the dependence between dinucleotides and colors, this affects the remaining sequence from the point of the error:

Sequence T A C T C C A T G C A
Colors ● ● ● ● ● ● ● ● ● ●

Sequence T A C T C C A A C G T
Colors ● ● ● ● ● ● ● ● ● ●

Thus, when the instrument makes an error while determining a color, the error mode is very different from when a single nucleotide is changed. This ability to differentiate different types of errors and differences is a very powerful aspect of SOLiD sequencing. With other technologies sequencing errors always appear as nucleotide differences.

2.8.3 Mapping in color space

Reads from a SOLiD sequencing run may exhibit all the same differences to a reference sequence as reads from other technologies: mismatches, insertions and deletions. On top if this, SOLiD reads may exhibit color errors, where a color is read wrongly and the rest of the read is affected. If such an error is detected, it can be corrected and the rest of the read can be converted to what it would have been without the error.

Consider this SOLiD read:

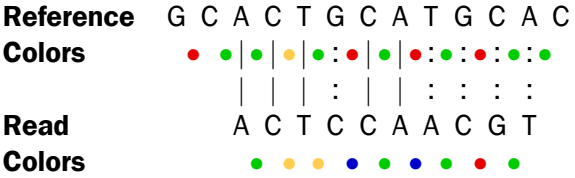


The first nucleotide (T) is from the primer, so is ignored in the following analysis. Now, assume that a reference sequence is this:



Here, the colors are just inferred since they are not the result of a sequencing experiment.

Looking at the colors, a possible alignment presents itself:



In the beginning of the read, the nucleotides match (ACT), then there is a mismatch (G in reference and C in read), then two more matches (CA), and finally the rest of the read does not match. But, the colors match at the end of the read. So a possible interpretation of the alignment is that there is a nucleotide change in position four of the read and a color space error between positions six and seven in the read. Such an interpretation can be represented as:



Here, the * represents a color error. The remaining part of the displayed read sequence has been adjusted according to the inferred error. So this alignment scores nine times the match score minus the mismatch cost and a color error cost. This color error cost is a new parameter that is introduced when performing read mapping in color space.

Note that a color error may be inferred before the first nucleotide of a read. This is the very first color after the known primer nucleotide that is wrong, changing the whole read.

Here is an example from a set of real SOLiD data that was reference assembled by taking color space into account using ungapped global alignments.

```
444_1840_767_F3 has 1 match with a score of 35:
```

```
    1046535 GATACTCAATGCCGCCAAAGATGGAAGCCGGGCCA 1046569    reference
            ||||||||||||||||||||||||||||||||||
            GATACTCAATGCCGCCAAAGATGGAAGCCGGGCCA            reverse read

444_1840_803_F3 has 0 matches

444_1840_980_F3 has 1 match with a score of 29:

    2620828 GCACGAAAACGCCGCGTGGCTGGATGGT*CAAC*GTC 2620862    reference
            |||||||||||||||||||||||||||*||||*|||
            GCACGAAAACGCCGCGTGGCTGGATGGT*CAAC*GTC            read

444_1840_1046_F3 has 1 match with a score of 32:

    3673206 TT*GGTCAGGGTCTGGGCTTAGGCGGTGAATGGGGC 3673240    reference
            ||*|||||||||||||||||||||||||||||||
            TT*GGTCAGGGTCTGGGCTTAGGCGGTGAATGGGGC            reverse read

444_1841_22_F3 has 0 matches

444_1841_213_F3 has 1 match with a score of 29:

    1593797 CTTTG*AGCGCATTGGTCAGCGTGTAATCTCCTGCA 1593831    reference
            |||||*|||||||| ||||||||||||||||||||
            CTTTG*AGCGCATTAGTCAGCGTGTAATCTCCTGCA            reverse read
```

The first alignment is a perfect match and scores 35 since the reads are all of length 35. The next alignment has two inferred color errors that each count is -3 (marked by * between residues), so the score is *35 - 2 x 3 = 29*. Notice that the read is reported as the inferred sequence taking the color errors into account. The last alignment has one color error and one mismatch giving a score of *34 - 3 - 2 = 29*, since the mismatch cost is 2.

Running the same reference assembly without allowing for color errors, the result is:

```
444_1840_767_F3 has 1 match with a score of 35:

    1046535 GATACTCAATGCCGCCAAAGATGGAAGCCGGGCCA 1046569    reference
            ||||||||||||||||||||||||||||||||||
            GATACTCAATGCCGCCAAAGATGGAAGCCGGGCCA            reverse read

444_1840_803_F3 has 0 matches

444_1840_980_F3 has 0 matches

444_1840_1046_F3 has 1 match with a score of 29:

    3673206 TTGGTCAGGGTCTGGGCTTAGGCGGTGAATGGGGC 3673240    reference
              |||||||||||||||||||||||||||||||||
            AAGGTCAGGGTCTGGGCTTAGGCGGTGAATGGGGC            reverse read

444_1841_22_F3 has 0 matches

444_1841_213_F3 has 0 matches
```

The first alignment is still a perfect match, whereas two of the other alignment now do not match since they have more than two errors. The last alignment now only scores 29 instead of 32, because two mismatches replaced the one color error above. This shows the power of including the possibility of color errors when aligning: many more matches are found.

The reference assembly program in the *CLC Genomics Workbench* does not directly support alignment in color space only, but if such an alignment was carried out, sequence 444_1841_213_F3 would have three errors, since a nucleotide mismatch leads to two color space differences. The alignment would look like this:

```
444_1841_213_F3 has 1 match with a score of 26:

    1593797 CTTTG*AGCGCATT*G*GTCAGCGTGTAATCTCCTGCA 1593831    reference
            |||||*|||||||*|*|||||||||||||||||||||
            CTTTG*AGCGCATT*G*GTCAGCGTGTAATCTCCTGCA           reverse read
```

So, the optimal solution is to both allow nucleotide mismatches and color errors in the same program when dealing with color space data. This is the approach taken by the assembly program in the *CLC Genomics Workbench*.

Note! If you set the color error cost as low as 1 while keeping the mismatch cost at 2 or above, a mismatch will instead be represented as two adjacent color errors.

2.8.4 Viewing color space information

Importing data from SOLiD systems (see section 2.1.3) will from *CLC Genomics Workbench* version 3.1 be imported as color space. This means that if you open the imported data, it will look like figure 2.77

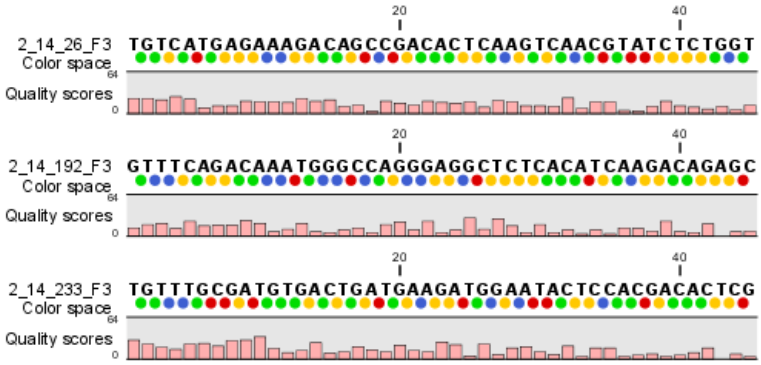


Figure 2.77: *Color space sequence list.*

In the **Side Panel** under **Nucleotide info**, you find the **Color space encoding** group which lets you define a few settings for how the colors should appear. These settings are also found in the side panel of mapping results and single sequences.

Infer encoding This is used if you want to display the colors for non-color space sequence (e.g. a reference sequence). The colors are then simply inferred from the sequence.

Show corrections This is only relevant for mapping results - it will show where the mapping process has detected color errors. An example of a color error is shown in figure 2.78.

Hide unaligned ends This option determines whether color for the unaligned ends of reads should be displayed. It also controls whether colors should be shown for gaps. The idea behind this is that these color dots will interfere with the color alignment, so it is possible to turn them off.

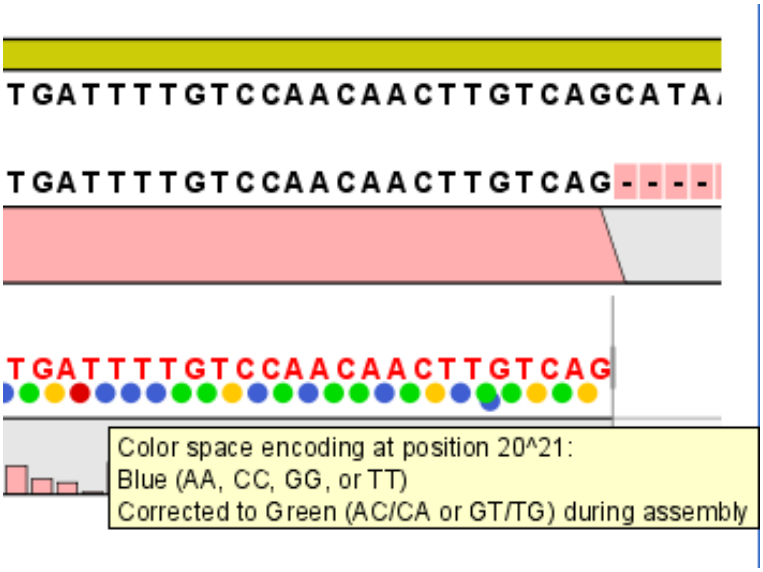


Figure 2.78: *One of the dots have both a blue and a green color. This is because this color has been corrected during mapping. Putting the mouse on the dot displays the small explanatory message.*

2.9 Interpreting genome-scale mappings

A big challenge when working with high-throughput sequencing projects is interpretation of the data. Section 2.11 describes how to automatically detect SNPs, whereas this section describes the manual inspection and interpretation techniques which are guided by visual information about the mapping. (We will not cover all the functionalities of the mapping view here, instead we refer to section **??** for general information about viewing and editing the resulting mappings).

Of particular interest for high-throughput sequencing data is probably the opportunity to extract part of mapping result, see section 2.9.5.

2.9.1 Getting an overview - zooming and navigating

Results from mapping high-throughput sequencing data may be extremely large, requiring an extra effort when you navigate and zoom the view. Besides the normal zoom tools and scrolling via the arrow keys, there are some of the settings in the **Side Panel** which can help you navigate a large mapping:

- **Gather sequences at top**. Under **Read layout** at the top of the Side Panel. you find this option. When you zoom in, only the reads aligning to the visible part of the view will be shown. This will save a lot of vertical scrolling.
- **Compactness**. Under **Read layout**, you can use different modes of compactness. This affects the way reads are shown. For example, you can display reads as **Packed** - very

thin stacked lines as shown in figure 2.79. The compactness also affects what information should be displayed below the reads (i.e. quality scores or chromatogram traces).

- **Text size**. Under **Text format** at the bottom of the **Side Panel**, you decrease the size of the text. This can improve the overview of the results (at the expense of legibility of sequence names etc.).

2.9.2 Single reads - coverage and conflicts

When you only have single reads data, **coverage** is one of the main resources for interpretation. You can display a coverage graph by clicking the checkbox in the Side Panel as shown in figure 2.79.

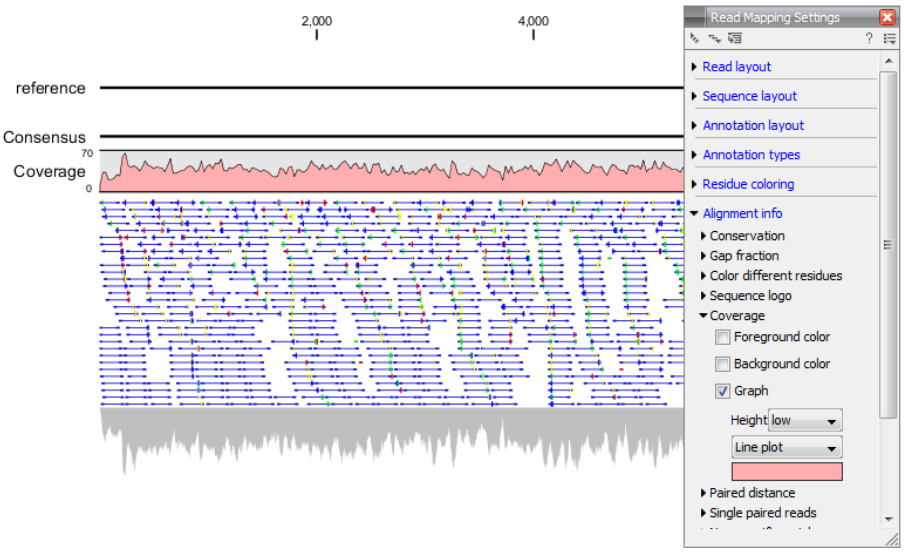


Figure 2.79: *The coverage graph can be displayed in the Side Panel under Alignment info.*

If you wish to see the exact coverage at a certain position, place the mouse cursor on the graph and see the exact value in the status bar at the very lower right corner of the Workbench window. Learn how to export the data behind the graph in section **??**.

When you zoom out on a large reference sequence, it may be difficult to discern smaller regions of low coverage. In this case, click the **Find Low Coverage** button at the top of the **Side Panel**. Clicking once will select the first part of the mapping with coverage at or below the number specified above the button (**Low coverage threshold**). Click again to find the next part with low coverage.

When mapping reads to a reference, a region of no coverage indicates genome-scale mutations. If the sequencing data contains e.g. a deletion, this will appear as a region of no coverage.

Problems during the sequencing process will also result in low coverage regions. In this case, you may wish to re-sequence these parts, e.g. using traditional "Sanger"-sequencing techniques. Due to the integrated nature of the *CLC Genomics Workbench* you can easily go to the primer designer and design PCR and sequencing primers to cover the low-coverage region. First select the low coverage region (and some extra nucleotides in order to get a good quality of the sequencing in the area of interest), and then:

right-click the selection | Open Selection in New View () | Show Primer Design () at the bottom of the view

Read more about designing primers in section **??**.

Besides looking at coverage, you can of course also inspect the conflicts by clicking the **Find Conflict** button at the top of the **Side Panel**. However, this will be practically impossible for large mappings, and it will not provide the same kind of overview as other approaches.

2.9.3 Interpreting genomic re-arrangements

Most of the analyses in this section are based on paired data which allows for much more powerful approaches to detecting genome rearrangements. Figure 2.80 shows a part of a mapping with paired reads.

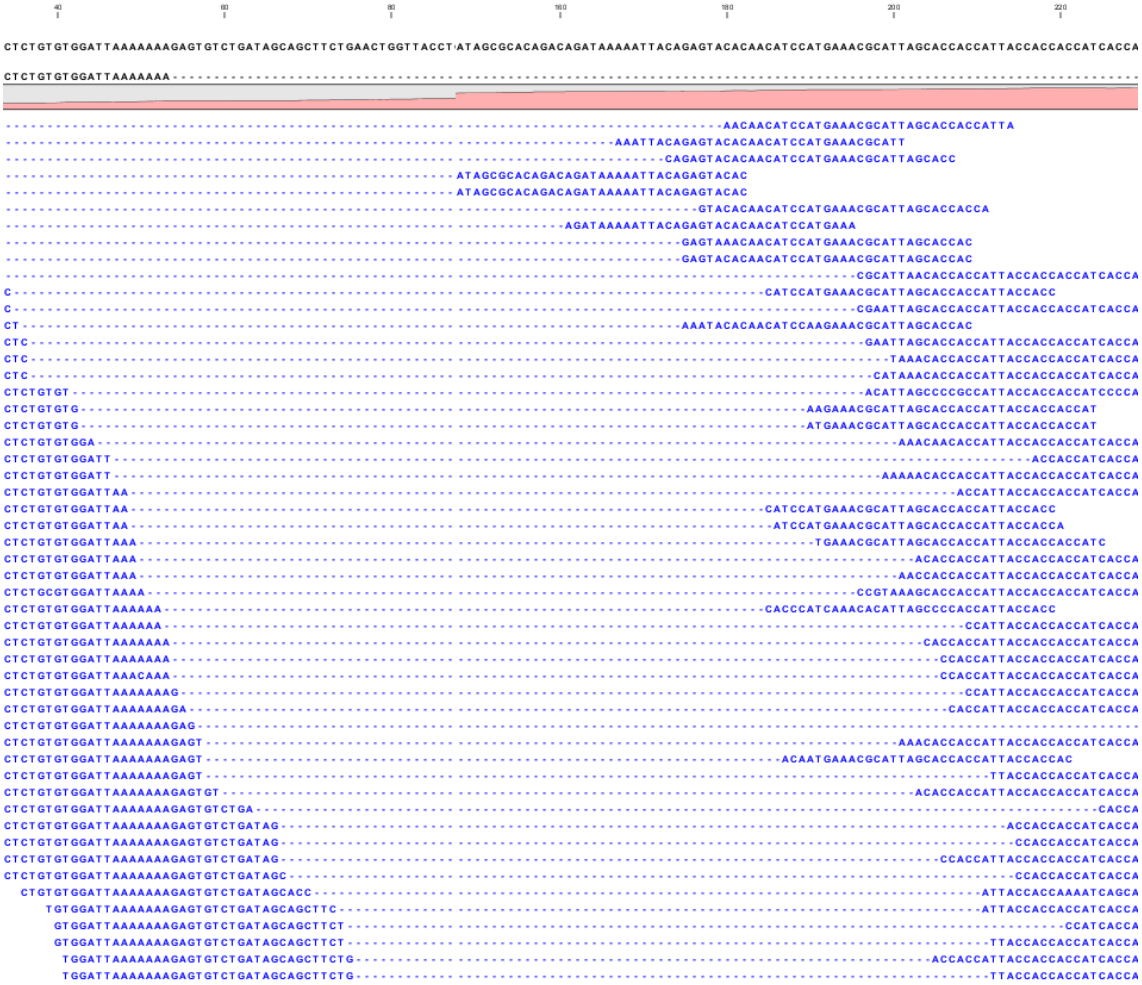


Figure 2.80: *Paired reads are shown with both sequences in the pair on the same line. The letters are probably too small to read, but it gives you the impression of how it looks.*

You can see that the sequences are colored blue and this leads us to the color settings in the **Side Panel**: under **Residue coloring** you find the group **Sequence colors** where you can specify the following colors:

- **Mapping**. The color of the consensus and reference sequence. Black per default.

- **Forward**. The color of forward reads (single reads). Green per default.
- **Reverse**. The color of reverse reads (single reads). Red per default.
- **Paired**. The color of paired reads. Blue per default.
- **Non-specific matches**. When a read would have matched another place in the mapping, it is considered a double match. This color will "overrule" the other colors. Note that if your mapping with several reference sequences, either using *de novo* assembly or read mapping with multiple reference sequences, a read is considered a double match when it matches more than once *across all the contigs/references*. A double match is yellow per default.

The settings are shown in figure 2.81.

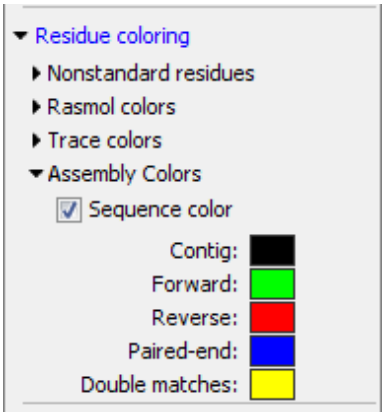


Figure 2.81: *Coloring of the reads.*

In addition to these colors, there are three graphs that will prove helpful when inspecting the paired reads, both found under **Alignment info** in the **Side Panel** (see figure 2.82):

- **Paired distance**. Displays the average distance between the forward and the reverse read in a pair.
- **Single paired reads**. Displays the percentage of the reads where only one of the reads in a pair matches.
- **Non-perfect matches**. Displays the percentage of the reads covering the current position which have at least one mismatch or a gap (the mismatch or gap does not need to be on this position - if there is just one anywhere on the read, it will count).
- **Non-specific matches**. Displays the percentage of the reads which match more than once. Note that if you are mapping against several sequences, either using *de novo* assembly or read mapping with multiple reference sequences, a read is considered a non-sepcific match when it matches more than once *across all the contigs/references*. A non-specific match is yellow per default.

These three graphs in combination with the read colors provide a great deal of information, guiding interpretations of the mapping result. A few examples will give directions on how to take advantage of these powerful tools:

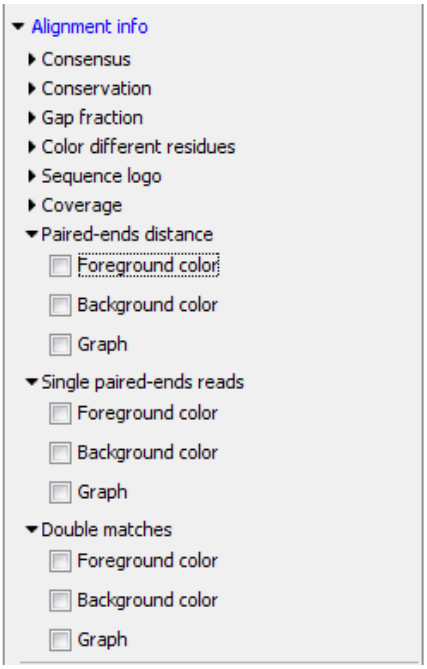


Figure 2.82: *More information about paired reads can be displayed in the Side Panel.*

Insertions

Looking at the **Single paired reads** graph in figure 2.83, you can see a sudden rise and fall. This means that at this position, only one part of the pair matches the reference sequence.

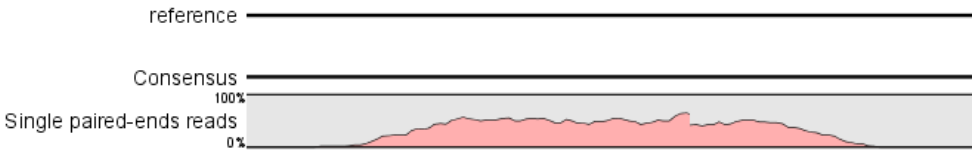


Figure 2.83: *More information about paired reads can be displayed in the Side Panel.*

Zooming in on the reads, you see how the color of the reads changes (see figure 2.84. They go from blue (paired) to green, meaning that at this point, the reverse part of the paired reads no longer match the reference sequence.

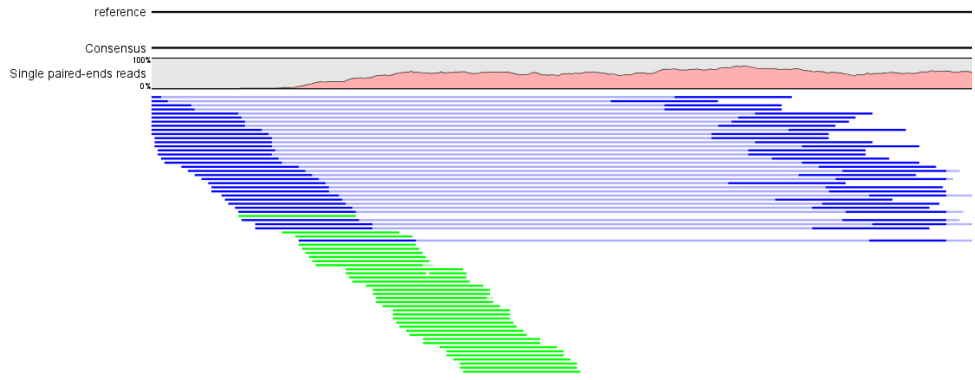


Figure 2.84: *Zooming where the single reads kick in.*

Since their reverse partners do not match the reference, there must be an insertion in the sequenced data. Looking further down the view, the color changes from green to a combination

of red (only reverse reads match) and blue (see figure 2.85).

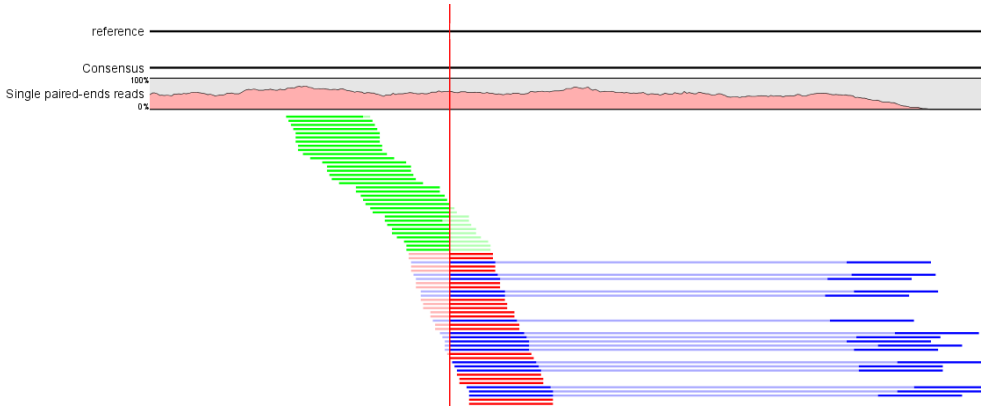


Figure 2.85: *Zooming where the paired reads kick in again.*

The reverse reads colored in red have a forward counterpart which do not match the reference sequence, for the same reason as we see the lonely forward reads before the insertion. Among the reverse reads, the "ordinary" paired reads start again, marking the end of the insertion.

As we now have established the presence of an insertion, it would be nice to know the exact location of insertion. You can see its exact position in figure 2.85 where the green reads stop matching the reference and the reverse reads take over (marked by the vertical red selection line).

Deletions

Deletions are much easier to detect - they are simply areas of no coverage (see figure 2.86).

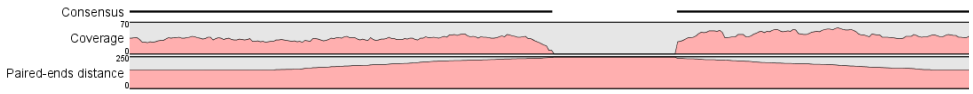


Figure 2.86: *A deletion in the sequenced data results in coverage of 0.*

Depending on the size of the deletion, you will see a rise in other graphs as well:

- A small deletion will result in an increase of the **Paired distance**, because the gap between the forward and the reverse read will just extend the deletion. This is the case in figure 2.86.
- A larger deletion will result in an increase of **Single paired reads** when the deletion is larger than the maximum distance allowed between paired reads (because the "other" part of the read has a match which is too far away). This maximum value can be changed when mapping the reads, see section 2.5. This is not illustrated.

When you zoom in on the deletion, you can see how the distance between the reads increase (see figure 2.87).

Duplications

In figure 2.88, the **Non-specific matches** graph is now shown.

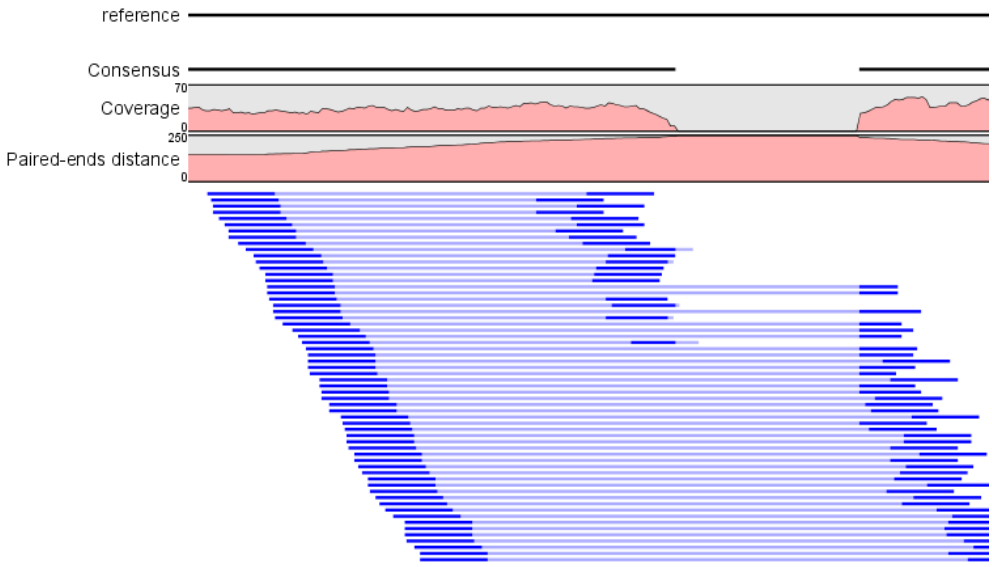


Figure 2.87: *Each part of the pair still match because the deletion is smaller than the maximum distance between the reads.*

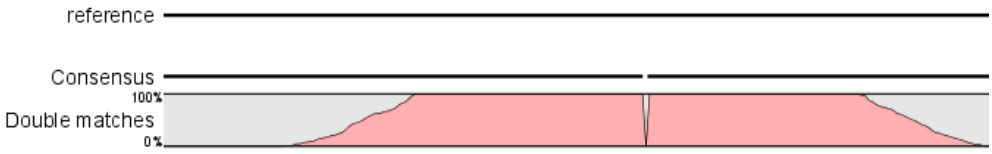


Figure 2.88: *A rise in the Non-specific matches.*

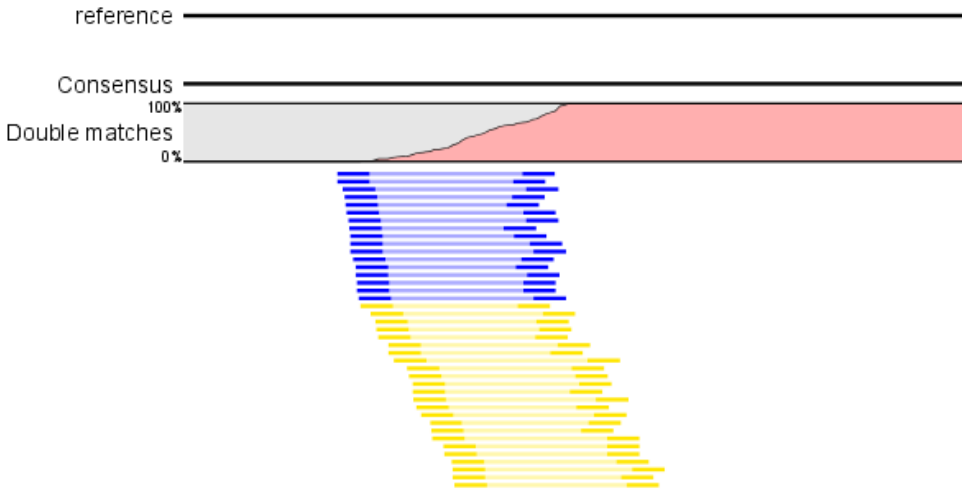


Figure 2.89: *Non-specific matches are shown in yellow.*

The Non-specific matches are reads that match more than once on the reference sequence. Zooming in on the reads puts a new color into play as shown in figure 2.89.

The yellow color means that the reads also match other positions on the reference, and this indicates that there is a duplication.

For a smaller duplication, you will see an increase in the **Paired distance**, because some of the reads are then matched to the other part of the duplication (this is shown in figure 2.90.

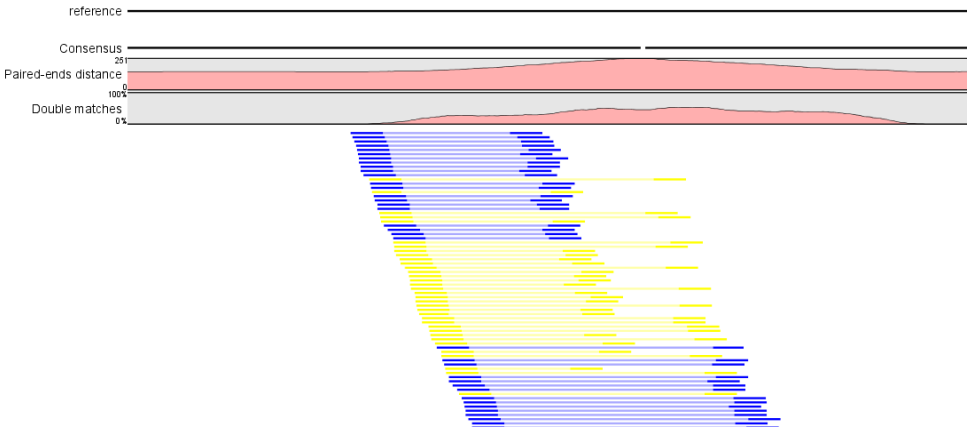


Figure 2.90: *Paired distance increases.*

Inversions

The interesting part in figure 2.91 is once again the **Single paired reads** graph which display a distinct pattern.

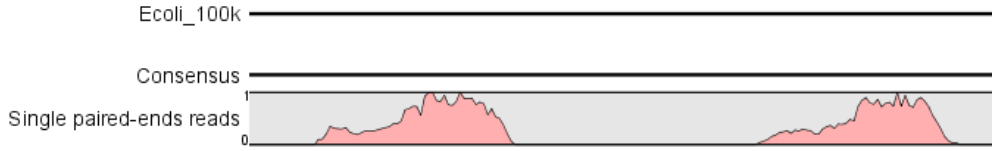


Figure 2.91: *Two peaks in the Single paired reads graph.*

The explanation of this is as follows. When the first peak starts, it is because the reverse part of the pairs no longer matches the reference sequence. This is shown in detail in figure 2.92.

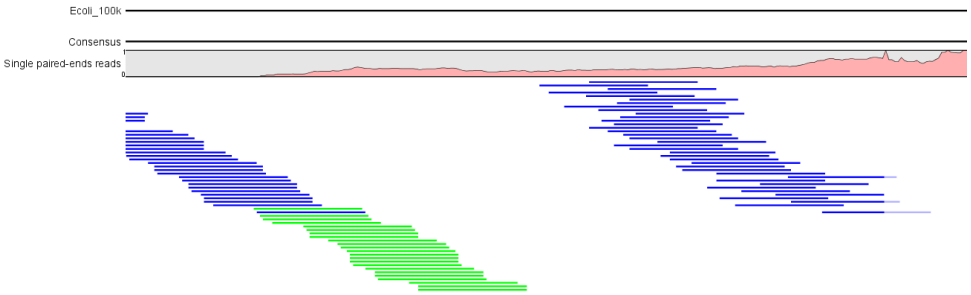


Figure 2.92: *Just before the inversion, only the forward reads match.*

Scrolling further along the view we can see the starting point of the inverted region. This is were the forward reads ends. At the same point, you will see a new pattern: a combination of reverse and paired reads as shown in figure 2.93.

The forward counterpart of the reverse reads has no match because of the inversion, whereas the paired reads have been reversed compared to the other paired reads in the mapping (this is not visible in the user interface, but a conclusion you can draw from the pattern of the other reads). Scrolling to the end of the inversion, you will see a similar pattern as in the beginning - it is just mirrored: Forward reads kick in at the end of the inversion, and reverse reads take over at when we get back to a "normal" sequence (see figure 2.94).

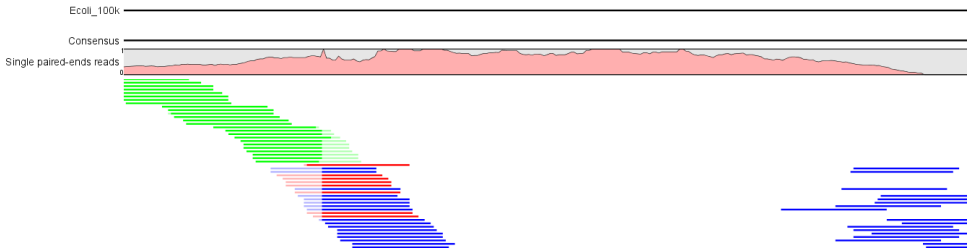


Figure 2.93: *The inversion starts where the reads shift from green (forward) to a combination of red and blue (reverse and paired) reads.*

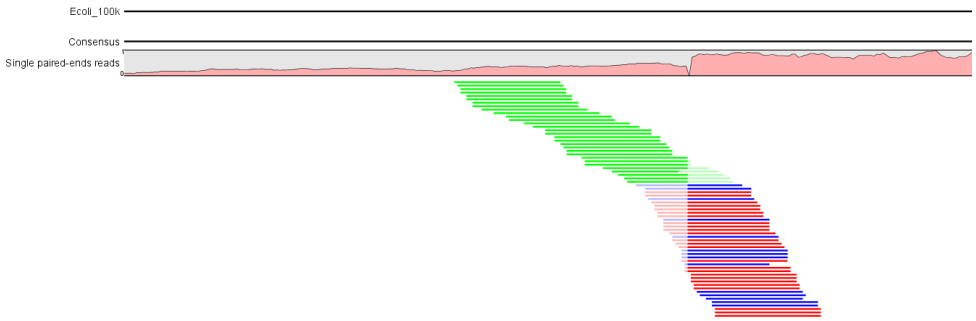


Figure 2.94: *The inversion ends where the reads shift from green (forward) to a combination of red and blue (reverse and paired) reads.*

2.9.4 Output from the mapping

Due to the integrated nature of *CLC Genomics Workbench* it is easy to use the consensus sequences as input for additional analyses. There are three options when you are viewing a mapping:

right-click the name of the consensus sequence (to the left) | Open Copy of Sequence | Save ( ) the new sequence

right-click the name of the consensus sequence (to the left) | Open Copy of Sequence Including Gaps | Save ( ) the new sequence

right-click the name of the consensus sequence (to the left) | Open This Sequence

Open Copy of Sequence creates a copy of the sequence, omitting all gap regions, which can be saved and used independently.

Open Copy of Sequence Including Gaps replaces all gaps with Ns. Any regions that appear to be deletions will be removed if this option is chosen. For example:

```
reference CCCGGAAAGGTTT
consensus CCC--AAA--TTT
match1    CCC--AAA
match2              TTT
```

Here, if you chose to open a copy of the consensus with gaps, you would get this output

```
CCCAAANNTTT
```

Open This Sequence will not create a new sequence but simply let you see the sequence in a

sequence view. This means that the sequence still "belong" to the mapping and will be saved together with the mapping. It also means that if you add annotations to the sequence, they will be shown in the mapping view as well. This can be very convenient e.g. for Primer design ( ).

If you wish to BLAST the consensus sequence, simply select the whole contig for your BLAST search. It will automatically extract the consensus sequence and perform the BLAST search.

In order to preserve the history of the changes you have made to the contig, the contig itself should be saved from the contig view, using either the save button ( ) or by dragging it to the **Navigation Area**.

2.9.5 Extract parts of a mapping

Sometimes it is useful to extract part of a mapping for in-depth analysis. This could be the case if you have performed an assembly of several genes and you want to look at a particular gene or region in isolation.

This is possible through the right-click menu of the reference or consensus sequence:

> **Select on the reference or consensus sequence the part of the contig to extract** | **Right-click** | **Extract from Selection**

This will present the dialog shown in figure 2.95.

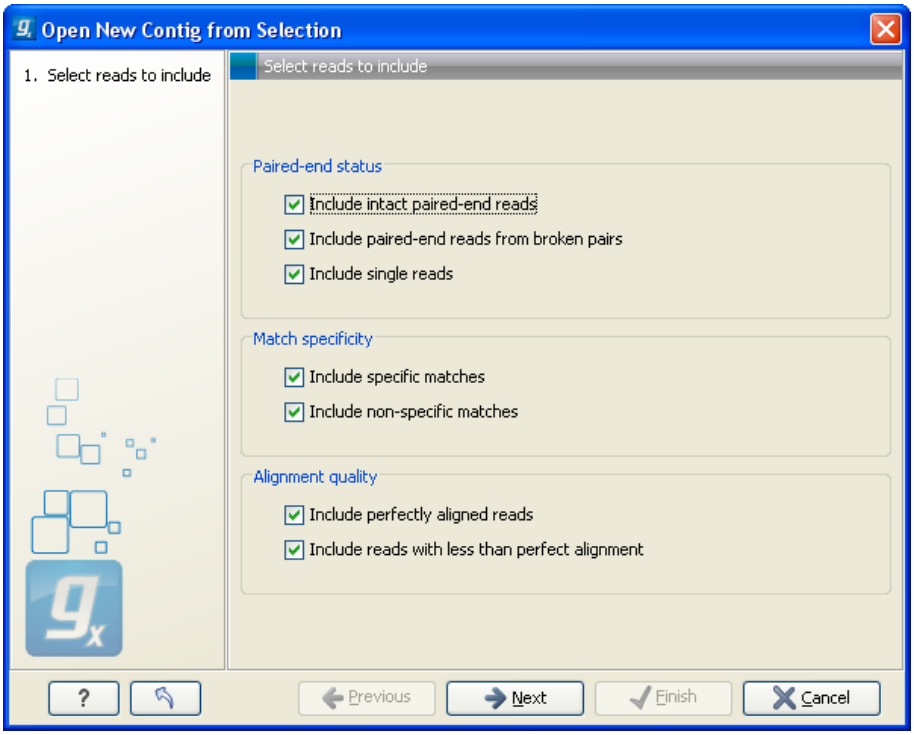


Figure 2.95: *Selecting the reads to include.*

The purpose of this dialog is to let you specify what kind of reads you want to include. Per default all reads are included. The options are:

Paired status **Include intact paired reads** When paired reads are placed within the paired distance specified, they will fall into this category. Per default, these reads are colored in blue.

Include paired reads from broken pairs When a pair is broken, either because only one read in the pair matches, or because the distance or relative orientation is wrong, the reads are placed and colored as single reads, but you can still extract them by checking this box.

Include single reads This will include reads that are marked as single reads (as opposed to paired reads). Note that paired reads that have been broken during assembly are not included in this category. Single reads that come from trimming paired sequence lists are included in this category.

Match specificity **Include specific matches** Reads that only are mapped to one position.

Include non-specific matches Reads that have multiple equally good alignments to the reference. These reads are colored yellow per default.

Alignment quality **Include perfectly aligned reads** Reads where *the full read* is perfectly aligned to the reference sequence (or consensus sequence for de novo assemblies). Note that at the end of the contig, reads may extend beyond the contig (this is not visible unless you make a selection on the read and observe the position numbering in the status bar). Such reads are not considered perfectly aligned reads because they don't align in their entire length.

Include reads with less than perfect alignment Reads with mismatches, insertions or deletions, or with unaligned nucleotides at the ends (the faded part of a read).

Note that only reads that are completely covered by the selection will be part of the new contig.

One of the benefits of this is that you can actually use this tool to extract subset of reads from a contig. An example work flow could look like this:

1. Select the whole reference sequence
2. Right-click and **Extract from Selection**
3. Choose to include only paired matches
4. Extract the reads from the new file (see section **??**)

You will now have all paired reads from the original mapping in a list.

2.9.6 Working with multiple contigs from read mappings

Alternatively, if you have several mappings in a table (as described in sections **??** and 2.5.5), you can extract the consensus sequences by selecting the relevant rows and clicking on the button labeled **Extract Contig** at the bottom of the view.

The sequence(s) you extract are copies of the consensus sequences. Tey are not attached to the original mapping.

The button marked **Extract Subset** allows you to extract a subset of your mappings to a new mapping object.

If you have annotated open reading frames on your sequences and wish to analyze each of these regions separately, e.g. translating and BLASTing or using other protein analysis tools, you

can extract all the ORF annotations by using our **Extract Annotations** plug-in, available from the **Plug-in Manager** ( ). This will give you a sequence list containing all the ORFs, making it easy to do batch analyses with other tools from *CLC Genomics Workbench*.

2.10 Merge mapping results

If you have performed two mappings with the same reference sequences, you can merge the results using the **Merge Mapping Results** ( ). This can be useful in situations where you have already performed a mapping with one data set, and you receive a second data set that you want to have mapped together with the first one. In this case, you can run a new mapping of the second data set and merge the results:

Toolbox | **High-throughput Sequencing ( )** | **Merge Mapping Results ( )**

This opens a dialog where you can select two or more mapping results. Note that they have to be based on the same reference sequences (it doesn't have to be the same file, but the sequence (the residues) should be identical).

Click **Next** if you wish to adjust how to handle the results (see section **??**). If not, click **Finish**. . For all the mappings that could be merged, a new mapping will be created. If you have used a mapping table as input, the result will be a mapping table. Note that the consensus sequence is updated to reflect the merge. The consensus voting scheme for the first mapping is used to determine the consensus sequence. This also means that for large mappings, the data processing can be quite demanding for your computer.

2.11 SNP detection

Instead of manually checking all the conflicts of a mapping to discover significant single-nucleotide variations, *CLC Genomics Workbench* offers automated SNP detection (see our Bioinformatics explained article on SNPs at `http://www.clcbio.com/BE`). The SNP detection in *CLC Genomics Workbench* is based on the *Neighborhood Quality Standard (NQS)* algorithm of [Altshuler et al., 2000] (also see [Brockman et al., 2008] for more information).

Based on your specifications on what you consider a valid SNP, the SNP detection will scan through the entire data and report all the SNPs that meet the requirements:

Toolbox | **High-throughput Sequencing ( )** | **SNP detection ( )**

This opens a dialog where you can select read mappings ( )/ ( ) to scan for SNPs (see sections 2.4 and 2.5 for information on how to map reads). You can also select RNA-Seq results ( ) as input.

Clicking **Next** will display the dialog shown in figure 2.96

2.11.1 Assessing the quality of the neighborhood bases

The SNP detection will look at each position in the mapping to determine if there is a SNP at this position. In order to make a qualified assessment, it also considers the general quality of the neighboring bases. The **Window size** is used to determine how far away from the current position this quality assessment should extend, and it can be specified in the upper part of the dialog. Note that at the ends of the read, an asymmetric window of the specified length is used.

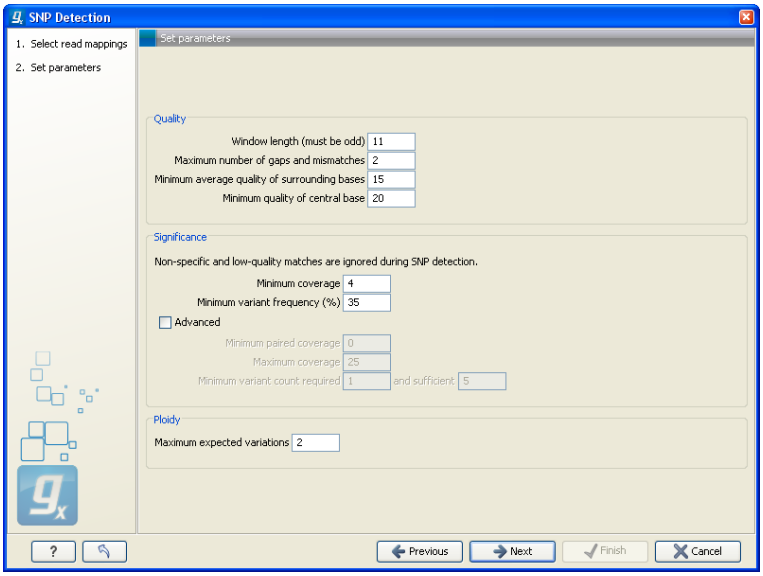


Figure 2.96: *SNP detection parameters.*

If the mapping is based on local alignment of the reads, there will be some reads with un-aligned ends (these ends are faded when you look at the mapping). These unaligned ends are not included in the scanning for SNPs but they are included in the quality filtering (elaborated below).

In figure 2.97, you can see an example with a window size of 11. The current position is high-lighted, and the horizontal high-lighting marks the nucleotides considered for a read when using a window size of 11.

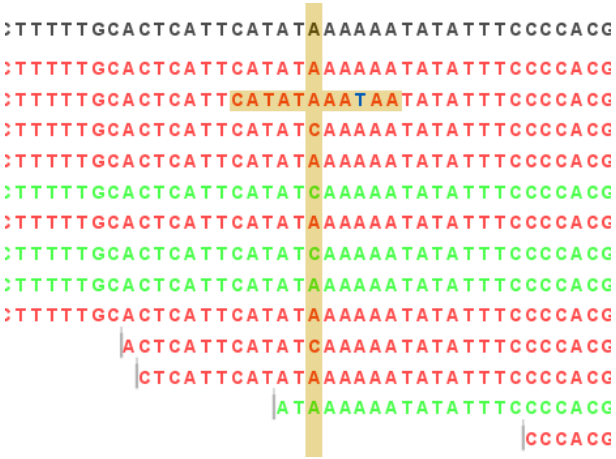


Figure 2.97: *An example of a window size of 11 nucleotides.*

For each read and within the given window size,[1] the following two parameters are used to assess the quality:

- **Minimum average quality of surrounding bases**. The average quality score of the nucleotides in a read within the specified window length has to exceed this threshold for the base to be included in the SNP calculation for this position (learn more about importing quality scores from different sequencing platforms in section 2.1).

[1]The window size is defined as the number of positions in the local alignment between that particular read and the reference sequence (for de novo assembly it would be the consensus sequence).)

- **Max. number of gaps and mismatches**. The number of gaps and mismatches allowed within the window length of the read. Note that this is excluding the "mismatch" that is considered a potential SNP. If there are more gaps or mismatches, this read will not be included in the SNP calculation at this position. Unaligned regions (the faded parts of a read) also count as mismatches, *even if some of the bases match*.

Note that for sequences without quality scores, the quality score settings will have no effect. In this case only the gap/mismatch threshold will be used for filtering low quality reads.

Figure 2.97 shows an example of a read with a mismatch, marked in dark blue. The mismatch is inside the window of 11 nucleotides.

When looking at a position near the end of a read (like the read at the bottom in figure 2.97), the window will be asymmetric as shown in figure 2.98. The window size will thus still be 11 in this case.

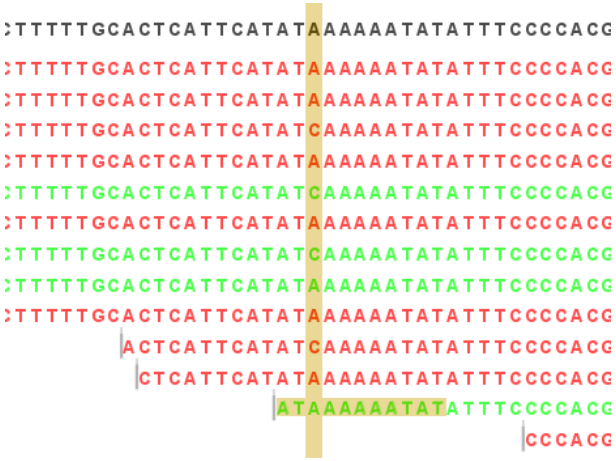


Figure 2.98: *A window near the end of a read.*

Besides looking horizontally within a window for each read, the quality of the central base is also examined: **Minimum quality of central base**. This is the quality score for the central base, i.e. the bases in the column high-lighted in figure 2.99. Bases with a quality score below this value are not considered in the SNP calculation at this position.

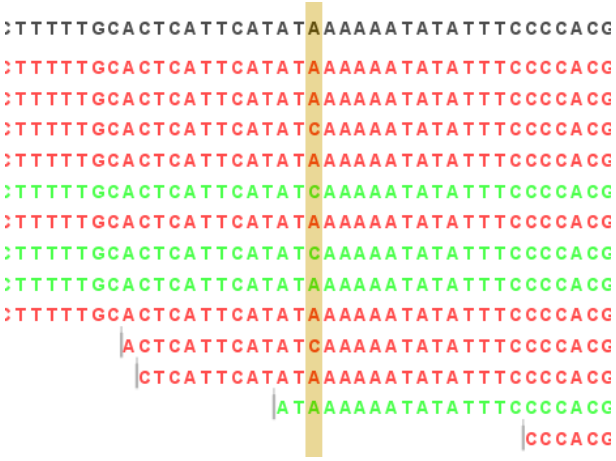


Figure 2.99: *A column of central bases in the neighborhood.*

In addition to low-quality reads, reads that match more than once on the reference sequence(s)

are also ignored. These reads are also called **Non-specific matches** and are colored in yellow in the view.

2.11.2 Significance of variation: is it a SNP?

At a given position, when the reads with low quality and multiple matches have been removed, the reads which pass the quality assessment will be compared to the reference sequence to see if they are different at this position (for *de novo* assembly the consensus sequence is used for comparison). For a variation to count as a SNP, it has to comply with the significance threshold specified in the dialog shown in figure 2.96.

- **Minimum coverage**. If SNPs were called in areas of low coverage, you would get a higher amount of false positives. Therefore you can set the minimum coverage for a SNP to be called. Note that the coverage is counted as the number of valid reads at the current position (i.e. the reads remaining when the quality assessment has filtered out the bad ones).
- **Minimum variant frequency**. This option is the threshold for the number of reads that display a variant at a given position. The threshold can be set as a frequency percentage or as a count. Setting the percentage at 35 % means that at least 35 % of the validated reads at this position should have a different base.

Below, there is an **Advanced** option letting you specify additional requirements. These will only take effect if the **Advanced** checkbox is checked.

- **Minimum paired coverage**. In samples based on paired data, more confidence is often attributed to valid paired reads than to single reads. You can therefore set the minimum coverage of valid paired reads in addition to the minimum coverage of all reads. Again, the paired coverage is counted as the number of valid reads completely covering the SNP (the space between mating pairs does not cover anything.) Note that when a value is provided for minimum paired coverage, reads from broken pairs will not be considered for SNP detection.
- **Maximum coverage**. Although it sounds counter-intuitive at first, there is also a good reason to be suspicious about high-coverage regions. Read coverage often displays peaks in repetitive regions where the alignment is not very trustworthy. Setting the maximum coverage threshold a little higher than the expected average coverage (allowing for some variation in coverage) can be helpful in ruling out false positives from such regions. You can see the distribution of coverage by creating a detailed mapping report (see section 2.6.1). The result table created by the SNP detection includes information about coverage, so you can specify a high threshold in this dialog, check the coverage in the result afterwards, and then run the SNP detection again with an adjusted threshold.
- **Minimum variant count**. This option is the threshold for the number of reads that display a variant at a given position. In addition to the percentage setting in the simple panel above, these settings are based on absolute counts. If the count required is set to 3, and the sufficient count is set to 5, it means that **even though** less than the required percentage of the reads have a variant base, it will still be reported as a SNP if at least 5 reads have it. However, if the count is 2, the SNP will not be called, regardless the percentage

setting. This distinction is especially useful with deep sequencing data where you have very high coverage and many different alleles. In this case, the percentage threshold is not suitable for finding valid SNPs in a small subset of the data. If you are not interested in reporting SNPs based on counts but only rely on the relative frequency, you can simply set the sufficient count number very high.

Positions where the reference sequences (consensus sequences for de novo assembly) have gaps and unaligned ends of the reads (faded part of the read) will not be considered in the SNP detection.

The last setting in this dialog (figure 2.96) concerns ploidy: **Maximum expected variations**. This is not a filtering option but a reporting option that is related to the minimum variant frequency setting. If the frequency or count threshold is set low enough the algorithm can call more allelic variants than the ploidy number of the organism sequenced. Such a result may occur as a real result but is inconsistent with the common assumption of an infinite sites mutation model where mutations are assumed to be so rare that they never affect the same position twice. For this reason, you can use the maximum expected variations setting to mark reported SNPs as "complex" when they involve more allelic variations then expected from the ploidy number under an infinite sites model. Note, that with this interpretation the "complex" flag holds true regardless of whether the sequencing data are generated from a population sample or from an individual sample (however, see below for an exception). For example, using a minimum variant frequency of 30% with a diploid organism, you are allowing SNPs with up to 3 variations within the sequencing reads, and by then setting the maximum expected variations count to 2 (the default), any SNPs with 3 variations will be marked as "complex" (see below). A ploidy level of 1 with two allelic variants represents a special case. Two allelic variants can occur if all reads are found to agree on one base that differs from the reference. Here, the number of allelic variants is higher than the ploidy level. but this is not inconsistent with an infinite sites mutation model and will not be termed complex. Two allelic variants can also occur if two variants are found within the sequencing reads where one of the variants is the same as the reference. Again, the data are not inconsistent with an infinite sites model if the sequencing data are generated from a population sample, but they are inconsistent with a clonal mutation-free origin of a sample from a single individual. For this reason we have chosen to also designate this latter case as "complex".

When there are **ambiguity** bases in the reads, they will be treated as separate variations. This means that e.g. a `Y` will not be collapsed with `C` or `T` in other reads. Rather, the Ys will be counted separately.

2.11.3 Reporting the SNPs

When you click **Next**, you will be able to specify how the SNPs should be reported (see figure 2.100).

- **Add SNP annotations to reference**. This will add an annotation for each SNP to the reference sequence.
- **Add SNP annotations to consensus**. This will add an annotation for each SNP to the consensus sequence.
- **Create table**. This will create a table showing all the SNPs found in the data set. The table will provide a valuable overview, whereas the annotations are useful for detailed

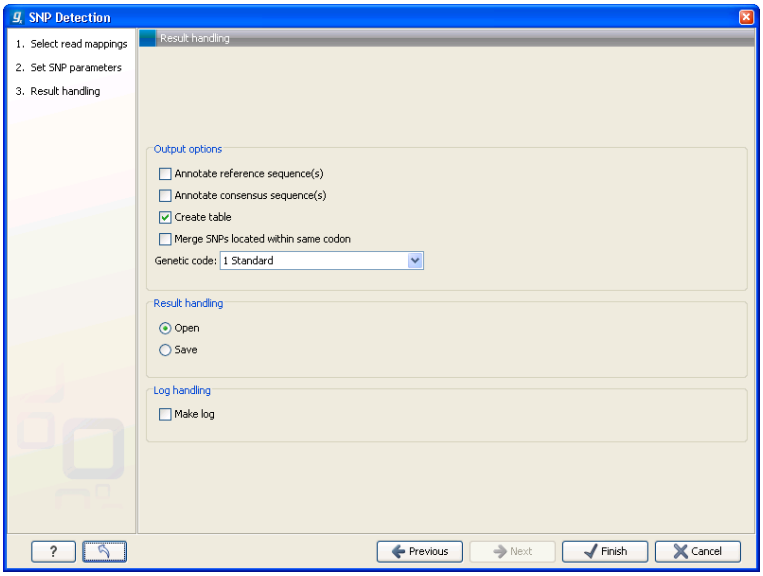


Figure 2.100: *Reporting options for SNP detection.*

inspection of a SNP, and also if the consensus sequence is used in further analysis in the *CLC Genomics Workbench*. The table displays the same information as the annotation for each SNP.

- **Genetic code**. When reporting the effect of a SNP on the amino acid, this translation table specified here is used.
- **Merge SNPs located within same codon**. This will merge SNPs that fall within the same codon (see section 2.11.4).

Figure 2.101 shows a SNP annotation.

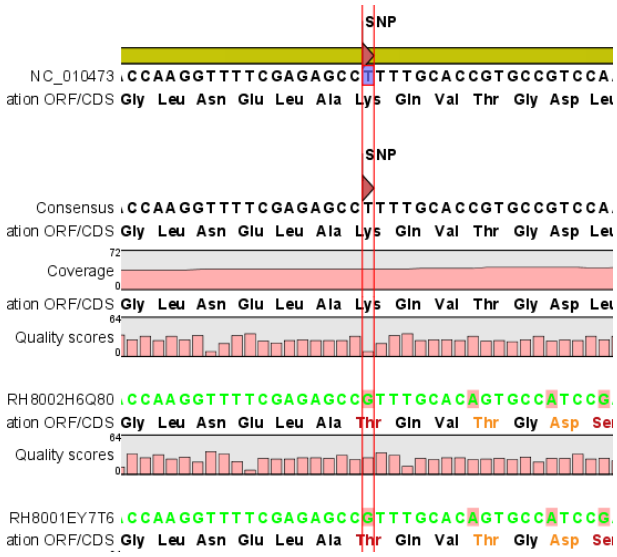


Figure 2.101: *A SNP annotation within a coding region.*

The SNP in figure 2.101 is within a coding region and you can see that one of the variations actually changes the protein product (from Lys to Thr). Placing your mouse on the annotation will reveal additional information about the SNP as shown in figure 2.102.

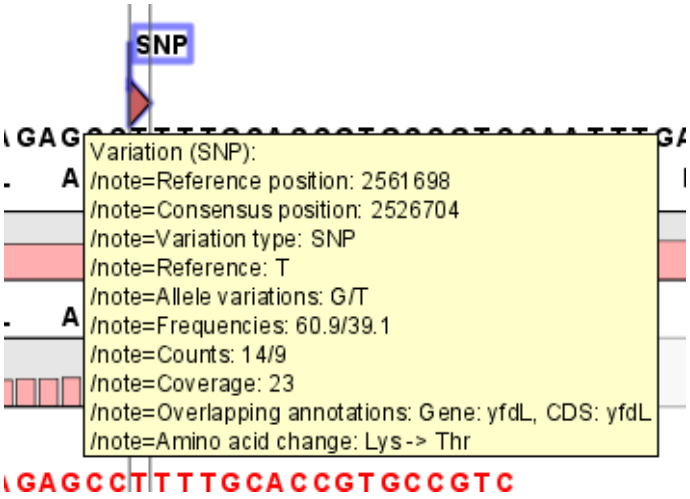


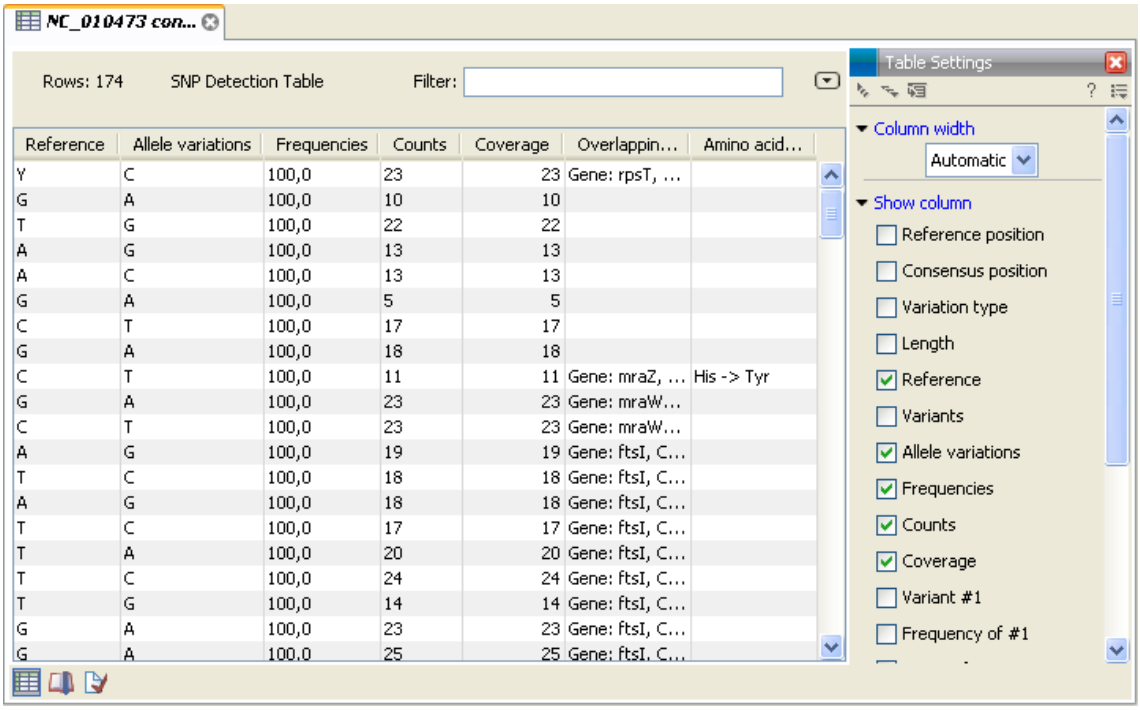
Figure 2.102: *A SNP annotation with associated information.*

The SNP annotation includes the following additional information:

- **Reference position**. The SNP's position on the reference sequence.
- **Consensus position**. The SNP's position on the consensus sequence.
- **Variation type**. The SNP is described as complex, if it has more variations than specified in the ploidy setting in figure 2.96.
- **Length**. The length of the SNP will always be one, as the name implies, unless two SNPs are found within the same codon (see section 2.11.4).
- **Reference**. The base found in the reference sequence. For results from de novo assembly, it will be the base found in the consensus sequence.
- **Variants**. The number of variants among the reads.
- **Allele variations**. Displays which bases are found at this position. In the example shown in figure 2.101, the reference sequence has a **T** whereas some of the reads have a **G**.
- **Frequencies**. The frequency of a given variant. In the example shown in figures 2.101 and 2.102, 61 % of the reads have a **G** and 39 % have a **T**.
- **Counts**. This is similar to the frequency just reported in absolte numbers. In the example shown in figures 2.101 and 2.102, 14 reads have a **G** and 9 have a **T**.
- **Coverage**. The coverage at the SNP position. Note that only the reads that pass the quality filter will be reported here.
- **Variant numbers and frequencies**. The information from the Allele variations, frequencies and counts are also split apart and reported for each variant individually (variant #1, #2 etc., depending on the ploidy setting.
- **Overlapping annotations**. This line shows if the SNP is covered by an annotation. The annotation's type and name will displayed. For annotated reference sequences, this information can be used to tell if the SNP is found in e.g. a coding or non-coding region of the genome. **Note** that annotations of type `Variation` and `Source` are not reported.

- **Amino acid change**. If the reference sequence of the is annotated with ORF or CDS annotations, the SNP detection will also report whether the SNP is synonymous or non-synonymous. If the SNP variant changes the amino acid in the protein translation, the new amino acid will be reported (see figure 2.103). Note that adjacent SNPs within the same codon are reported as one SNP in order to determine the impact on the protein level (see section 2.11.4)..

The same information is also recorded in the table. An example of a table is shown in figure 2.103.



Reference	Allele variations	Frequencies	Counts	Coverage	Overlappin...	Amino acid...
Y	C	100,0	23	23	Gene: rpsT, ...	
G	A	100,0	10	10		
T	G	100,0	22	22		
A	G	100,0	13	13		
A	C	100,0	13	13		
G	A	100,0	5	5		
C	T	100,0	17	17		
G	A	100,0	18	18		
C	T	100,0	11	11	Gene: mraZ, ...	His -> Tyr
G	A	100,0	23	23	Gene: mraW...	
C	T	100,0	23	23	Gene: mraW...	
A	G	100,0	19	19	Gene: ftsI, C...	
T	C	100,0	18	18	Gene: ftsI, C...	
A	G	100,0	18	18	Gene: ftsI, C...	
T	C	100,0	17	17	Gene: ftsI, C...	
T	A	100,0	20	20	Gene: ftsI, C...	
T	C	100,0	24	24	Gene: ftsI, C...	
T	G	100,0	14	14	Gene: ftsI, C...	
G	A	100,0	23	23	Gene: ftsI, C...	
G	A	100,0	25	25	Gene: ftsI, C...	

Figure 2.103: *A table of SNPs.*

In addition to the information shown as annotation, the table also includes the name of the mapping (since the table can include SNPs for many mappings, you need to know which one it belongs to). The table can be **Exported** ( ) as a csv file (comma-separated values) and imported into e.g. Excel. Note that the CSV export includes all the information in the table, regardless of filtering and what has been chosen in the **Side Panel**. If you only want to use a subset of the information, simply select and **Copy** ( ) the information. The columns in the SNP and DIP tables have been synchronized to enable merging in a spreadsheet.

Note that if you make a split view of the table and the mapping (see section **??**), you will be able to browse through the SNPs by clicking in the table. This will cause the view to jump to the position of the SNP.

If you wish to investigate the SNPs further, you can use the filter option (see section **??**). Figure 2.104 show how to make a filter that only shows homozygote SNPs.

You can also use the filter to show e.g. nonsynonymous SNPs (filter the **Amino acid change** column to not being empty as shown in figure 2.105).

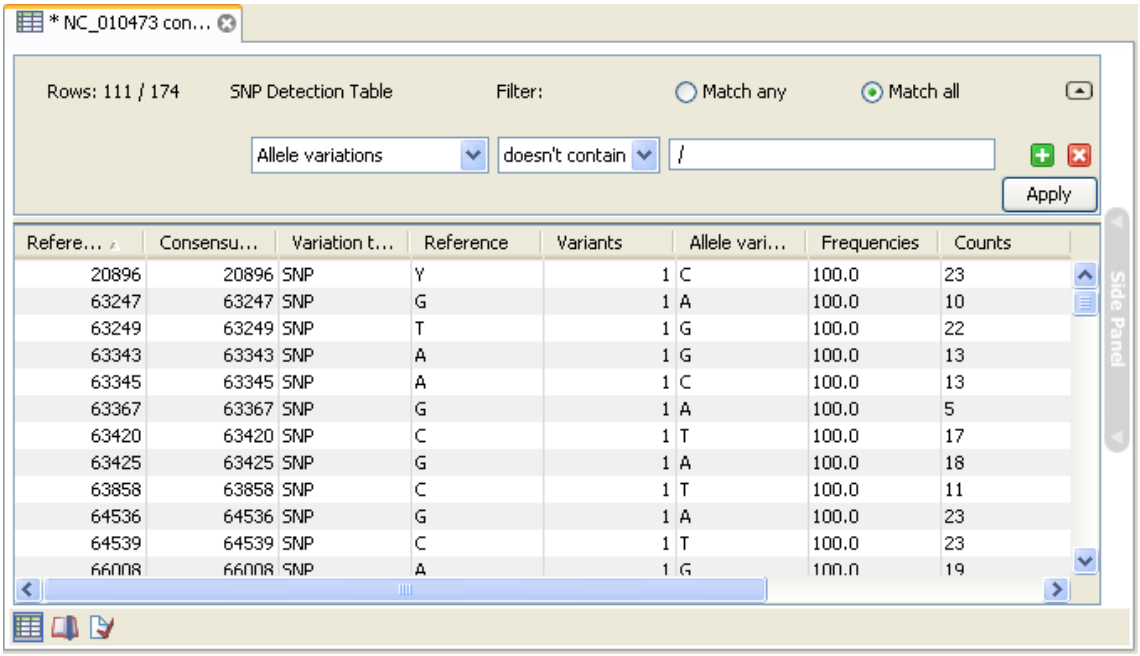


Figure 2.104: *Filtering away the SNPs that have more than one allele variant.*

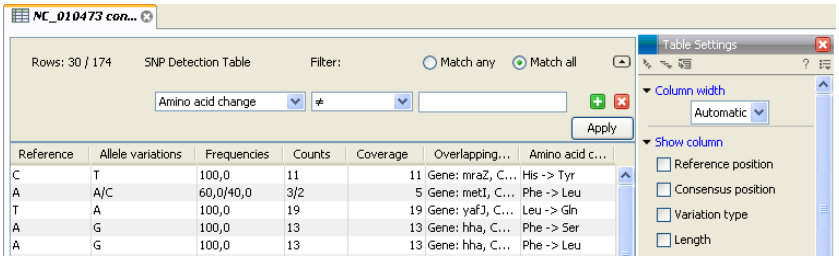


Figure 2.105: *Filtering the SNP table to only display nonsynonymous SNPs.*

2.11.4 Adjacent SNPs affecting the same codon

Figure 2.106 shows an example where two adjacent SNPs are found within the same codon.

The *CLC Genomics Workbench* can report these SNPs as one SNP in order to evaluate the combined effect on the translation to protein. If these SNPs were considered individually, the predicted amino acid change for each individual SNP would not have been reflecting the sequencing data.

The *CLC Genomics Workbench* will first find the individual SNPs and in the cases where two SNPs are found within the same codon, they are considered for a merge. Note that the merged SNP needs to be supported by the same reads that gave rise to the individual SNP calls. Consider the case shown in figure 2.107, where there are still two adjacent SNPs within the same codon, but there are no reads supporting the merged SNP.

In this case, no SNP will be reported since there are no reads supporting the merged SNP.

Note that both the individual SNP and the merged SNP need to fulfill the quality filtering and significance criteria to be reported. When reporting merged SNPs please be aware that these will find it harder to pass through the quality filtering since there are requirements for both the individual SNPs and the merged SNP to be fulfilled.

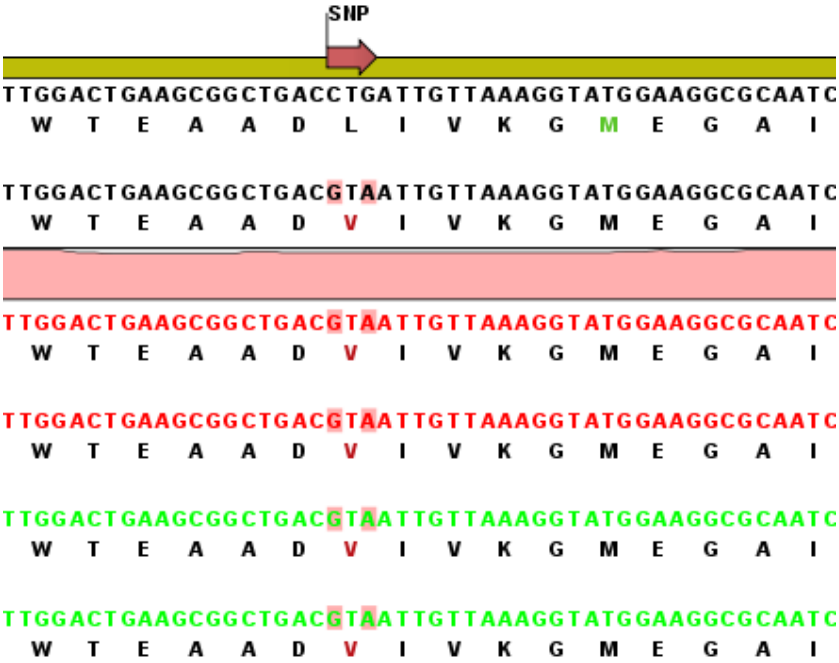


Figure 2.106: *Two adjacent SNPs in the same codon.*

2.12 DIP detection

CLC Genomics Workbench offers automated detection of small deletion/insertion polymorphisms (also known as DIPs) when reads are mapped to a reference.

If you have high coverage in your mapping, you will often find a lot of gaps in the consensus sequence. This is because just a single insertion in one of the reads will cause a gap in all other sequences at this position. The majority of all these gaps should simply be ignored as they were introduced due to sequencing errors in a single read or a very few reads. Automated DIP detection can be used to find the gaps that are significant. If you want to use the consensus sequence for other purposes, you can simply ignore all the gaps (they will disappear once the consensus sequence is out of the mapping view), and the significant ones can then be annotated as DIPs (see section 2.12.2).

In *CLC Genomics Workbench*, a DIP is a deletion or an insertion of consecutive nucleotides present in experimental sequencing data when compared to a reference sequence. Automated DIP detection is therefore possible only for results from read mapping.

The terms "deletion" and "insertion" are understood as events that have happened to the sequencing sample relative to the reference sequence: when the local alignment between a read and the reference exhibits gaps in the read, nucleotides have been *deleted* (in the read, relative to the reference), and when the local alignment exhibits gaps in the reference sequence, nucleotides have been *inserted* (in the read, relative to the reference). Figure 2.108 shows an insertion (of TC, to the left) and a deletion (of CC, to the right).

The automated DIP detection in *CLC Genomics Workbench* bases all reported DIPs on DIPs found in *individual* reads. The length of reported deletions and insertions is therefore bounded by the number of insertions and deletions allowed per read by the read mapping algorithm.

In most situations, a DIP in a single read is not sufficient experimental evidence. The *CLC*

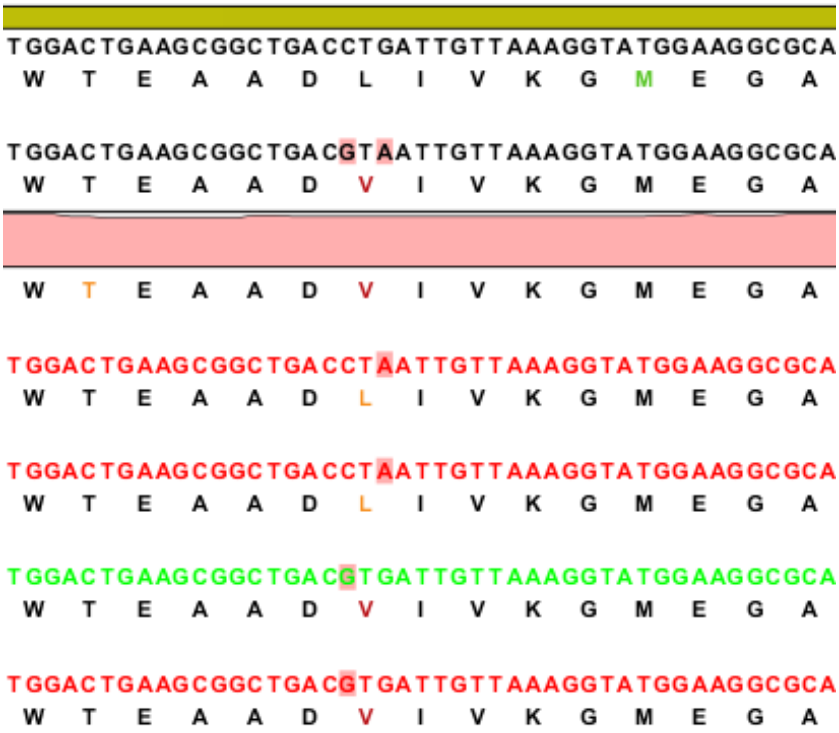


Figure 2.107: *Two adjacent SNPs in the same codon but with different reads.*



Figure 2.108: *Two DIPs, an insertion and a deletion.*

Genomics Workbench allows you to specify how many reads must cover and agree on a DIP in order for it to be reported by the automated DIP detection. Two reads agree on a deletion if their local alignments to the reference sequence both contain the same number of consecutive gaps aligned to the same reference positions. Likewise, two reads agree on an insertion if their local alignments specify the same number of consecutive gaps at the same position in the reference sequence *and* the nucleotides inserted in the two reads are the same. Figure 2.109 shows some reads disagreeing on an insertion (of TC or TA?, on the left) and agreeing on a deletion (of CC, on the right).

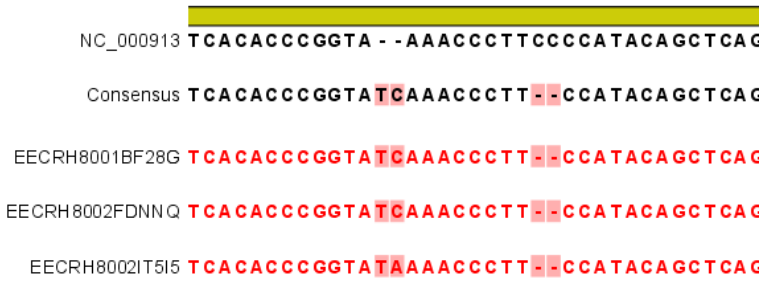


Figure 2.109: *Disagreement and agreement of DIPs.*

Based on your specifications on what you consider a valid DIP, the DIP detection will scan through

the entire mapping and report all the DIPs that meet the requirements:

Toolbox | **High-throughput Sequencing ( )** | **DIP detection ( )**

This opens a dialog where you can select read mapping results ( )/ ( ) to scan for DIPs (see section 2.5 for information on how to map reads to a reference).

2.12.1 Experimental support of a DIP

Clicking **Next** will display the dialog shown in figure 2.110.

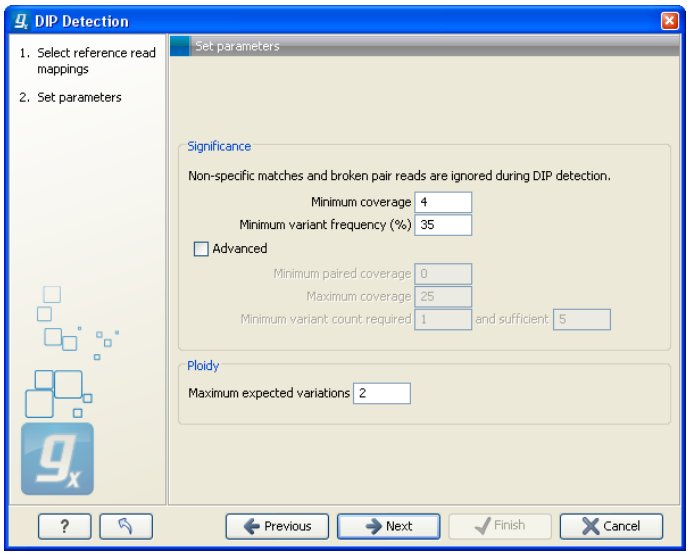


Figure 2.110: *DIP detection parameters.*

To avoid false positives, the automated DIP detection of *CLC Genomics Workbench* ignores reads that have multiple hit positions on the reference (marked by yellow color) or reads that come from broken pairs.

In addition, *CLC Genomics Workbench* allows you to specify thresholds for the experimental support of reported DIPs, thus filtering the DIPs found in single, valid reads, see figure 2.110.

- **Minimum coverage**. DIPs called in areas of low coverage will likely result in a higher amount of false positives. Therefore you can set the minimum coverage for a DIP to be called. Note that the coverage is counted as the number of valid reads completely covering the DIP.
- **Minimum variant frequency**. Often reads do not completely agree on a DIP, and you may want to report only the most frequent variants at each DIP site. This threshold can be specified as the percentage of the reads or the absolute number of reads. By default, the frequency in percent is set to 35%, which means that at least 35% of the valid reads covering the DIP site must agree on the DIP for it to be reported. In effect, this means that at most two different variants will be reported at each site, which is reasonable for diploid organisms. If a DIP is frequent enough to be reported, the DIP annotation or table entry will contain information about all other variants which are also frequent enough—even if they are not DIPs.

Below, there is an **Advanced** option letting you specify additional requirements. These will only take effect if the **Advanced** checkbox is checked.

- **Minimum paired coverage**. In based on paired data, more confidence is often attributed to valid paired reads than to single reads. You can therefore set the minimum coverage of valid paired reads in addition to the minimum coverage of all reads. Again, the paired coverage is counted as the number of valid reads completely covering the DIP (the space between mating pairs does not cover anything.) Note that regardless of this setting, reads from broken pairs are never considered for DIP detection.

- **Maximum coverage**. Read coverage often displays peaks in repetitive regions where the alignment is not very trustworthy. Setting the maximum coverage threshold a little higher than the expected average coverage (allowing for some variation) can be helpful in ruling out false positives from such regions.

- **Minimum variant counts**. This option is the threshold for the number of reads that display a DIP at a given position. In addition to the percentage setting in the simple panel above, these settings are based on absolute counts. If the count required is set to 3, and the sufficient count is set to 5, it means that **even though** less than the required percentage of the reads have a DIP, it will still be reported as a DIP if at least 5 reads have it. However, if the count is 2, the DIP will not be called, regardless the percentage setting. This distinction is especially useful with deep sequencing data where you have very high coverage and many different alleles. In this case, the percentage threshold is not suitable for finding valid DIPs in a small subset of the data. If you are not interested in reporting DIPs based on counts but only rely on the relative frequency, you can simply set the sufficient count number very high.

- **Maximum expected variations**. This is not a filtering option, but is related to the minimum variant frequency setting. By setting the frequency threshold low enough to allow more variants than the ploidy of the organism sequenced, you can use the maximum expected variations setting to mark reported DIPs as "complex", if they involve more variations then expected from the ploidy. For example, using a minimum variant frequency of 30% with a diploid organism, you are allowing DIPs with up to 3 variations, and then by setting the maximum expected variations count to 2 (the default), any DIPs with 3 variations will be marked as complex (see below).

2.12.2 Reporting the DIPs

When you click **Next**, you will be able to specify how the DIPs should be reported:

- **Annotate reference sequence(s)**. This will add an annotation for each DIP to the reference sequences in the input.

- **Annotate consensus sequence(s)**. This will add an annotation for each DIP to the consensus sequences in the input. In either way, DIP annotations contain the following information:

  - **Reference position**. The first position of the DIP in the reference sequence.
  - **Consensus position**. The first position of the DIP in the consensus sequence.
  - **Variation type**. Will be "DIP" or "Complex DIP", depending on the value of the maximum expected variations setting and the actual number of variations found at the DIP site.

- **Length**. The length of the DIP. Note that only small deletions and insertions are found. This is because the DIP detection is based on the alignment of the reads generated by the mapping process, and the mapping only allows a few insertions/deletions (see section 2.5 for information on how to map reads to a reference).
- **Reference**. The residues found in the reference sequence (either gaps for insertions or bases for deletions).
- **Variants**. The number of variants among the reads.
- **Allele variation**. The variations found in the reads at the DIP site. Contains only those variations whose frequency is at least that specified by the minimum variant frequency setting.
- **Frequencies**. The frequencies of the variations, both absolute (counts) and relative (percentage of coverage).
- **Coverage**. The number of valid reads completely covering the DIP site.
- **Variant numbers and frequencies**. The information from the Allele variations, frequencies and counts are also split apart and reported for each variant individually (variant #1, #2 etc., depending on the ploidy setting.
- **Overlapping annotations**. Says if the DIP is covered, in part or in whole, by an annotation. The annotation's type and name will displayed. For annotated reference sequences, this information can be used to tell if the DIP is found in e.g. a coding or non-coding region of the genome. **Note** that annotations of type `Variation` and `Source` are not reported.
- **Amino acid change**. If the reference sequence of is annotated with ORF or CDS annotations, the DIP detection will also report whether the DIP changes the amino acid sequence resulting from translation, and, if so, whether the change involves frame-shifting.

- **Create table**. This will create a table showing all the DIPs found. The table will provide a valuable overview, whereas the annotations are useful for detailed inspection of a DIP, and also if the annotated sequences are used for further analysis in the *CLC Genomics Workbench*.

Figure 2.111 shows the result of a DIP detection output as annotations on the reference sequence. The DIP detection found the DIPs of figure 2.109.

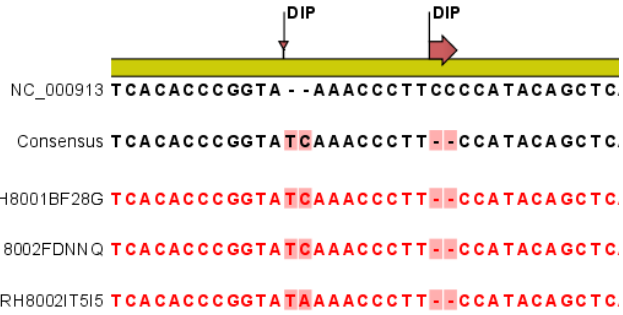


Figure 2.111: *DIPs detected witin a coding region.*

The DIPs occur within a coding region (identified by the long yellow annotation) and you can see that they both shift the frame of the translation, since their sizes are not divisible by 3.

Placing your mouse on the annotations will reveal detailed information about the DIPs as shown in figure 2.112.

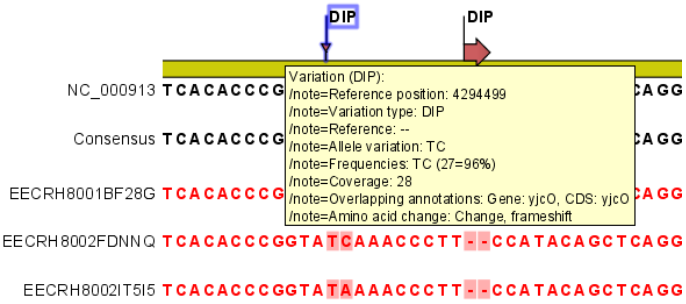
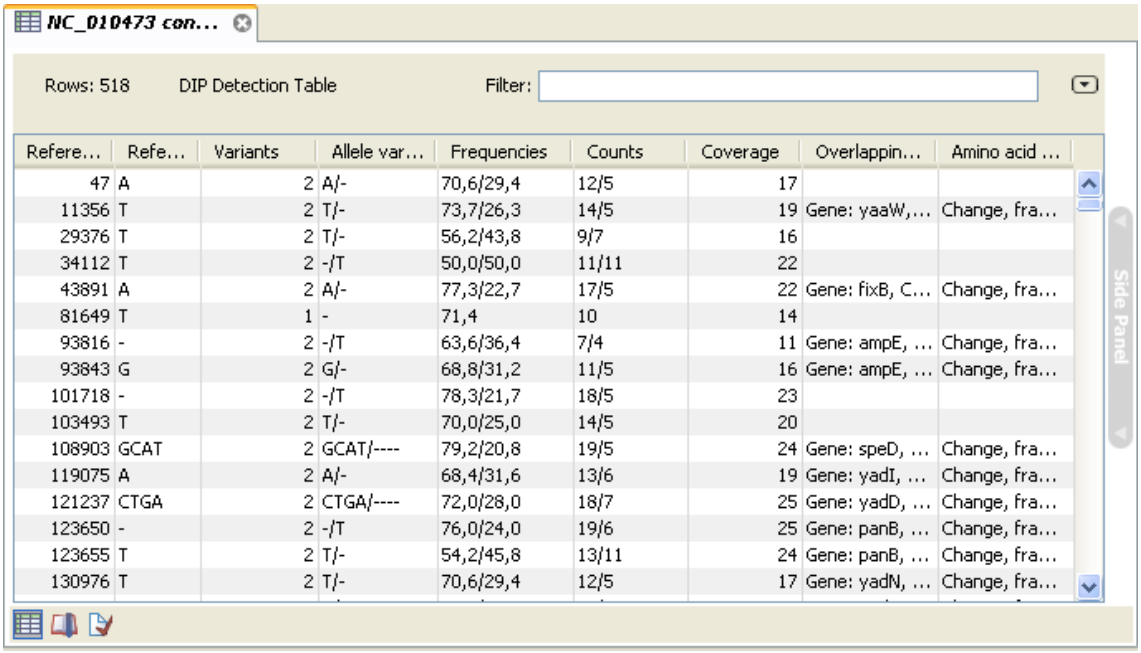


Figure 2.112: *A DIP annotation with detailed information.*

The same information is also recorded in the table output. An example of a table is shown in figure 2.113.



Refere...	Refe...	Variants	Allele var...	Frequencies	Counts	Coverage	Overlappin...	Amino acid ...
47	A	2	A/-	70,6/29,4	12/5	17		
11356	T	2	T/-	73,7/26,3	14/5	19	Gene: yaaW,...	Change, fra...
29376	T	2	T/-	56,2/43,8	9/7	16		
34112	T	2	-/T	50,0/50,0	11/11	22		
43891	A	2	A/-	77,3/22,7	17/5	22	Gene: fixB, C...	Change, fra...
81649	T	1	-	71,4	10	14		
93816	-	2	-/T	63,6/36,4	7/4	11	Gene: ampE, ...	Change, fra...
93843	G	2	G/-	68,8/31,2	11/5	16	Gene: ampE, ...	Change, fra...
101718	-	2	-/T	78,3/21,7	18/5	23		
103493	T	2	T/-	70,0/25,0	14/5	20		
108903	GCAT	2	GCAT/----	79,2/20,8	19/5	24	Gene: speD, ...	Change, fra...
119075	A	2	A/-	68,4/31,6	13/6	19	Gene: yadI, ...	Change, fra...
121237	CTGA	2	CTGA/----	72,0/28,0	18/7	25	Gene: yadD, ...	Change, fra...
123650	-	2	-/T	76,0/24,0	19/6	25	Gene: panB, ...	Change, fra...
123655	T	2	T/-	54,2/45,8	13/11	24	Gene: panB, ...	Change, fra...
130976	T	2	T/-	70,6/29,4	12/5	17	Gene: yadN, ...	Change, fra...

Figure 2.113: *A table of DIPs.*

In addition to the information shown as annotation, the table also includes the name of the mapping (since the table can include DIPs for many references, you need to know which one it belongs to). The table can be **Exported** ( ) as a csv file (comma-separated values) and imported into e.g. Excel. Note that the CSV export includes all the information in the table, regardless of filtering and what has been chosen in the **Side Panel**. If you only want to use a subset of the information, simply select and **Copy** ( ) the information. The columns in the SNP and DIP tables have been synchronized to enable merging in a spreadsheet.

Note that if you make a split view of the table and the mapping (see section **??**), you will be able to browse through the DIPs by clicking in the table. This will cause the view to jump to the position of the DIP.

2.13 ChIP sequencing

*CLC Genomics Workbench*can perform analysis of Chromatin immunoprecipitation sequencing (ChIP-Seq) data based on the information contained in a single sample subjected to immunoprecipitation (ChIP-sample) or by comparing a ChIP-sample to a control sample where the immunoprecipitation step is omitted. The first step in a ChIP-Seq analysis is to map the reads to a reference (see section 2.5) which maps your reads against one or more specified reference sequences. If both a ChIP- and a control sample is used, these must be mapped separately to produce separate ChIP- and control samples. These samples are then used as input to the ChIP-Seq tool which surveys the pattern in coverage to detect significant peaks:

Toolbox | **High-throughput Sequencing ( )** | **ChIP-Seq Analysis ( )**

This opens a dialog where you can select one or more mapping results ( )/ ( ) to use as ChIP-samples. Control samples are selected in the next step.

2.13.1 Peak finding and false discovery rates

Clicking **Next** will display the dialog shown in figure 2.114.

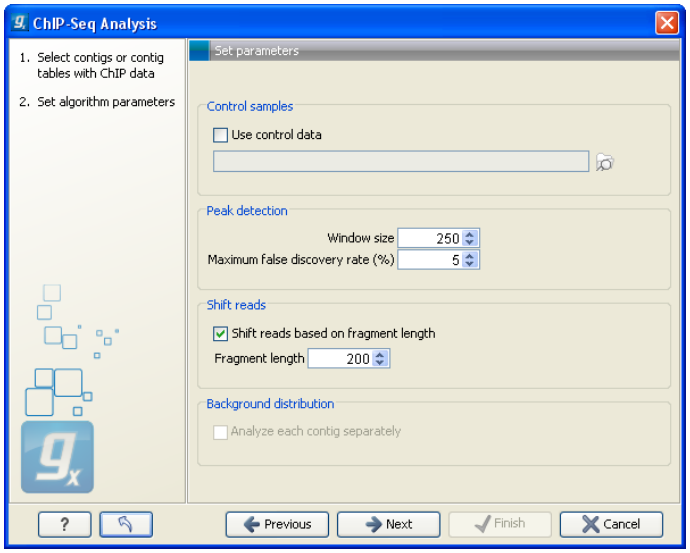


Figure 2.114: *Peak finding and false discovery rates.*

If the option to include control samples is included, the user must select the appropriate sample to use as control data. If the mapping is based on several reference sequences, the Workbench will automatically match the ChIP-samples and controls based on the length of the reference sequences.

The peak finding algorithm includes the following steps:

- Calculate the null distribution of background sequencing signal
- Scan the mappings to identify candidate peaks with a higher read count than expected from the null distribution
- Merge overlapping candidate peaks

- Refine the set of candidate peaks based on the count and the spatial distribution of reads of forward and reverse orientation within the peaks

The estimation of the null distribution of coverage and the calculation of the false discovery rates are based on the **Window size** and **Maximum false discovery rate (%)** parameters. The **Window size** specifies the width of the window that is used to count reads both when the null distribution is estimated and for the subsequent scanning for candidate peaks.

The **Maximum false discovery rate** specifies the maximum proportion of false positive peaks that you are willing to accept among your called peaks. A value of 10 % means that you are willing to accept that 10 % of the peaks called are expected to be false discoveries.

To estimate the false discovery rate (FDR) we use the method of [Ji et al., 2008] (see also Supplementary materials of the paper).

In the case where only a ChIP-sample is used, a negative binomial distribution is fitted to the counts from low coverage regions. This distribution is used as a null distribution to obtain the numbers of windows with a particular count of reads that you would expect in the absence of significant binding. By comparing the number of windows with a specific count you expect to see under the null distribution and the number you actually see in your data, you can calculate a false discovery rate for a given read count for a given window size as: 'fraction of windows with read count expected under the null distribution'/'fraction of windows with read count observed'.

In the case where both a ChIP- and a control sample are used, a sampling ratio between the samples is first estimated, using only windows in which the total numbers of reads (that is, the sum of those in the sample and those in the control) is small. The sampling ratio is estimated as the ratio of the cumulated sample read counts ($c^{sample} = \sum_i k_i^{sample}$) to cumulated control read counts ($c^{control} = \sum_i k_i^{control}$) in these windows. The sampling ratio is used to estimate the proportion of the reads that are expected to be ChIP-sample reads under the null distribution, as $p_0 = c^{sample}/(c^{sample} + c^{control})$. For a given total read count, *n*, of a window, the numbers of reads expected in the ChIP-sample under the null distribution can then be estimated from the binomial distribution with parameters *n* and p_0. By comparing the expected and observed numbers, a false discovery rate can then be calculated. Note, that when a control sample is used different null-distributions are estimated for different total read counts, *n*.

In both cases, the user can specify whether the null distribution should be estimated separately for each reference sequence by checking the option **Analyze each reference separately**.

Because the ChIP-seq experimental protocol selects for sequencing input fragments that are centered around a DNA-protein binding site it is expected that true peaks will exhibit a signature distribution where forward reads are found upstream of the binding site and reverse reads are found downstream of the binding site leading to reduced coverage at the exact binding site. For this reason, the algorithm allows you to shift forward reads towards the 3' end and reverse reads towards the 5' end in order to generate a more marked peak prior to the peak detection step. This is done by checking the **Shift reads based on fragment length** box. To shift the reads you also need to input the expected length of the sequencing input fragments by setting the **Fragment length** parameter, this is the size of the fragment isolated from gel (L in the illustration below).

The illustration below shows a peak where the forward reads are in one window and the reverse reads fall in another window (window 1 and 3).

```
---------------------------------------------------------- reference
        ----------------------|------------------          (actual sequenced fragment length = L bp)
```

```
        ---->                                          reads
         ---->                                         reads
                                      <---             reads
                                    <---               reads.

|--------------------|--------------------|-------------  window size W
         1                    2                 3
```

If the reads are not shifted, the algorithm will count 2 reads in window 1 and 3. But if the forward reads are shifted 0.5XL to the right and reverse reads are shifted 0.5xL to left, the algorithm will find 4 reads in window 2 as shown below:

```
------------------------------------------------------------ reference
        ----------------------|------------------          (actual sequenced fragment length = L basepairs)
                            ---->                            reads
                             ---->                           reads
                         <---                                reads
                        <---                                 reads

|--------------------|--------------------|-------------  window size W
         1                    2                 3
```

So shifting reads will increase the signal to noise ratio.

The following peak refinement step, the reporting of the peak and the visualization will use the original position of the reads, so the shifting is only a virtual shift performed as part of the peak detection.

2.13.2 Peak refinement

Clicking **Next** will display the dialog shown in figure 2.115.

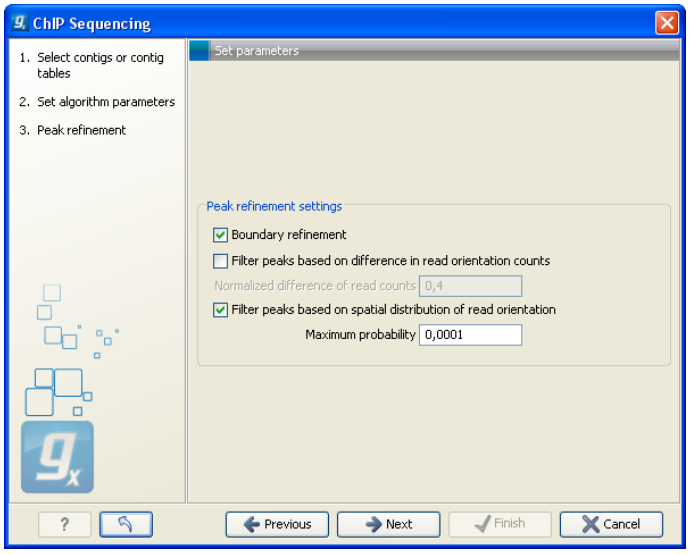


Figure 2.115: *Peak refinement settings.*

This dialog presents the parameters and options that can be used to refine the set of candidate peaks discovered when scanning the read mapping. All three refinement options again utilize the fact that coverage around a true DNA-protein binding site is expected to exhibit a signature distribution where forward reads are found upstream of the binding site and reverse reads are found downstream of the binding site. Peak refinement can be performed both with- and

without a control sample but the algorithm only uses information contained in the reads from the ChIP-samples, not the control samples.

If the **Boundary refinement** option is checked, the algorithm will estimate the position of the DNA-protein binding interaction and place the resulting annotations on this region, rather than on the region where a peak in coverage is found. A center of sequencing intensity is defined for all forward reads as the median value of the center points of all forward reads and likewise for all reverse reads. The "refined peak" is thus defined as the region between these two points.

One of the advantages of including this boundary refinement is that shorter regions can be given as input to subsequent pattern discovery analysis.

By checking the **Filter peaks based on difference in read orientation counts** the algorithm will calculate the normalized difference in the number of forward and reverse reads within a peak as

$$\frac{|\text{count forward reads - count reverse reads}|}{\text{count forward reads + count reverse reads}}$$

The desired maximum value of this parameter can be set in the **Normalized difference of read counts** field and any candidate peak with a value above this will then be dismissed. Setting a low value will ensure that peaks are only called if there is a well balanced number of forward and reverse reads.

As an example if you have 15 forward reads and 5 reverse reads, you will end up with a value of 0.5. With the default limit set to 0.4, a peak like that would be excluded.

By checking the **Filter peaks based on spatial distribution of read orientation** the algorithm will evaluate how clearly separated the location of forward and reverse reads are within a peak. This is done via the Wilcoxon rank-sum test (see http://en.wikipedia.org/wiki/Mann-Whitney-Wilcoxon_test). The null hypothesis here is that the positions of forward and reverse reads within a peak are drawn from the same distribution i.e. that their locations are not significantly different and the alternative hypothesis is that the forward reads have a sum of ranked positions that is shifted to lower positions than the reverse reads. Peaks will be dismissed if the probability of the null hypothesis exceeds the value set in the **Maximum probability** field.

Setting a low **Maximum probability** will ensure that peaks are only called if there is a clear signature distribution where forward reads are found upstream of reverse reads within the peak.

A general comment about peak filtering is that the relevant statistics are all reported in the peak table that the algorithm outputs. If it is desirable to explore a large set of candidate peaks it is recommended to use no or relatively loose filtering criteria and then use the advanced table filtering options to explore the effect of the different parameters (see section **??**). It may be desirable to omit the addition of annotations in this exploratory analysis and rely on the information in the table instead. Once a desired set of parameters is found, the algorithm can be rerun using these as filtering criteria to add annotations to the reference sequence and to produce a final list of peaks.

2.13.3 Reporting the results

When you click **Next**, you will be able to specify how the results should be reported (see figure 2.116).

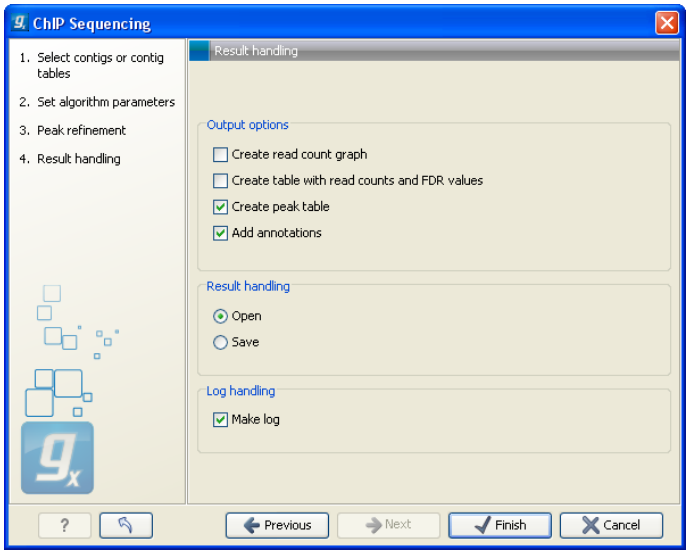


Figure 2.116: *Output options.*

The different output options are described in detail below. Note, that it is not possible to output a graph and table of read counts in the case where a control sample is used. These options are therefore disabled in this case.

Graph and table of background distribution and false discovery statistics

An example of a FDR graph based on a single ChIP-sample is shown in figure 2.117.

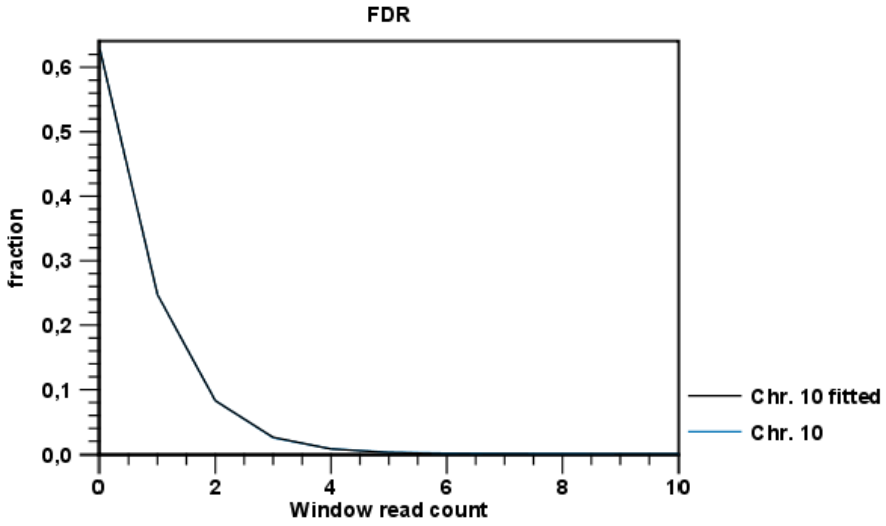


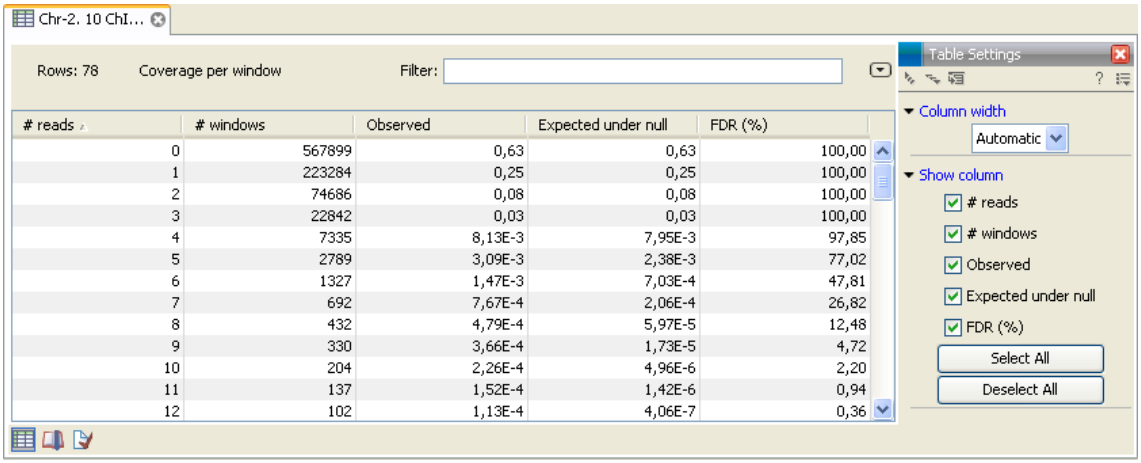
Figure 2.117: *FDR graph.*

The graph shows the estimated background distribution of read counts in discrete windows and the observed counts and can thus be used to inspect how well the estimated distribution fits the observed pattern of coverage.

The FDR table displays the observed and expected fraction of windows with a given read count and also shows the rate of false discovery related to a given level of coverage within a window:

- **# reads** - the number of reads within a window.
- **# windows** - the number of windows with the given read count. A window of a fixed width is slid across the sequence. For every window position the number of reads in that window is recorded and stored as the read count. After this, the windows are counted based on their recorded read counts. # windows of read count x is thus the number of windows that were found to contain x reads during this process. This is done to establish the background distribution of coverage and to evaluate the fit of the estimated distribution.
- **Observed** - the observed faction of windows with the given read count.
- **Expected under null** - the expected fraction of windows with a given read count, under the null distribution.
- **FDR %** - the false discovery rate which is the fraction of the peaks with the given read count that can be expected to be false positives.

An example is shown in figure 2.118.



# reads	# windows	Observed	Expected under null	FDR (%)
0	567899	0,63	0,63	100,00
1	223284	0,25	0,25	100,00
2	74686	0,08	0,08	100,00
3	22842	0,03	0,03	100,00
4	7335	8,13E-3	7,95E-3	97,85
5	2789	3,09E-3	2,38E-3	77,02
6	1327	1,47E-3	7,03E-4	47,81
7	692	7,67E-4	2,06E-4	26,82
8	432	4,79E-4	5,97E-5	12,48
9	330	3,66E-4	1,73E-5	4,72
10	204	2,26E-4	4,96E-6	2,20
11	137	1,52E-4	1,42E-6	0,94
12	102	1,13E-4	4,06E-7	0,36

Figure 2.118: *FDR table.*

From this table you can see that less than 5% of the called peaks with 9 reads can be expected to be false discoveries and for peaks with 11 reads the FDR is less than 1%.

Peak table and annotations

The main result is the table showing the peaks and the annotations added to the reference sequence.

An example of a peak table is shown in figure 2.119.

The table includes information about each peak that has been found:

- **Name**. If the mapping was based on more than one reference sequence, the name of the reference sequence in question will be shown here.
- **Region**. The position of the peak. To find that position in the ChIP-sample mapping, you can make a split view of the table and the mapping (see section **??**). You will then be able to browse through the peaks by clicking in the table. This will cause the view to jump to the position of the peak region.

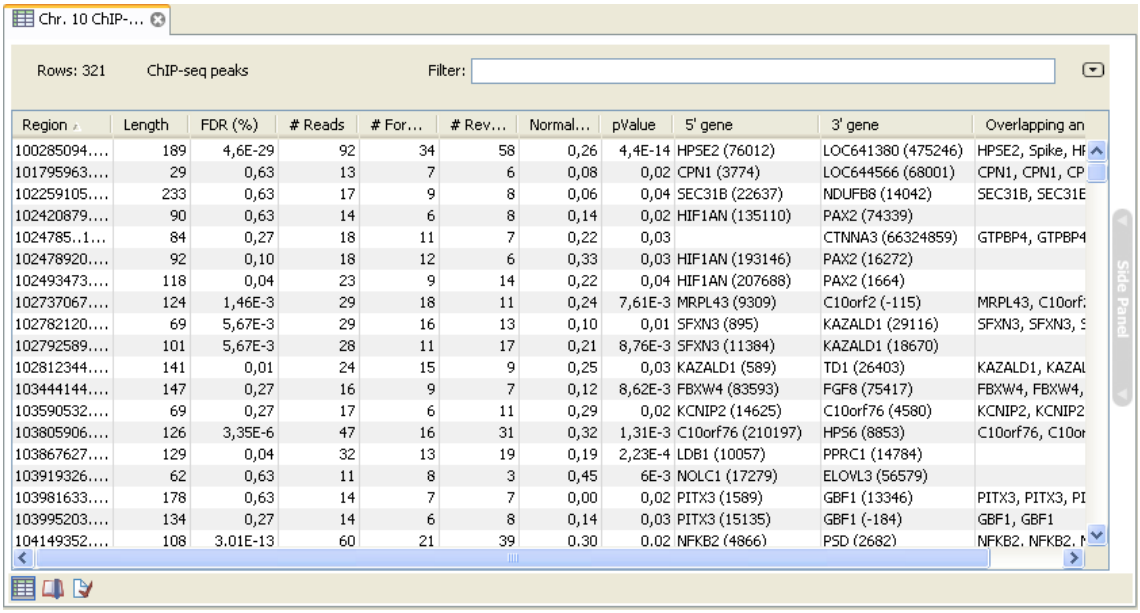


Figure 2.119: *ChIP sequencing peak table.*

- **Length**. The length of the peak.
- **FDR (%)**. The false discovery rate for the peak (learn more in section 2.13.1).
- **# Reads**. The total number of reads covering the peak region.
- **# Forward reads**. The number of forward reads covering the peak region.
- **# Reverse reads**. The number of reverse reads covering the peak region. The normalized difference in the count of forward-reverse reads is calculated based on these numbers (see figure 2.115).
- **Normalized difference**. See section 2.13.2.
- **P-value**. The p-value is for the Wilcoxon rank sum test for the equality of location of forward and reverse reads in a peak. See section 2.13.2.
- **Max forward coverage**. The refined region described in section 2.13.2 is calculated based on the maximum coverage of forward and reverse reads.
- **Max reverse coverage**. See previous.
- **Refined region**. The refined region.
- **Refined region length**. The length of the refined region.
- **5' gene**. The nearest gene upstream, based on the start position of the gene. The number in brackets is the distance from the peak to the gene start position.
- **3' gene**. The nearest gene downstream, based on the start position of the gene. The number in brackets is the distance from the peak to the gene start position.
- **Overlapping annotations**. Displays any annotations present on the reference sequence that overlap the peak.

Note that if you make a split view of the table and the mapping (see section **??**), you will be able to browse through the peaks by clicking in the table. This will cause the view to jump to the position of the peak.

An example of a peak is shown in figure 2.120.

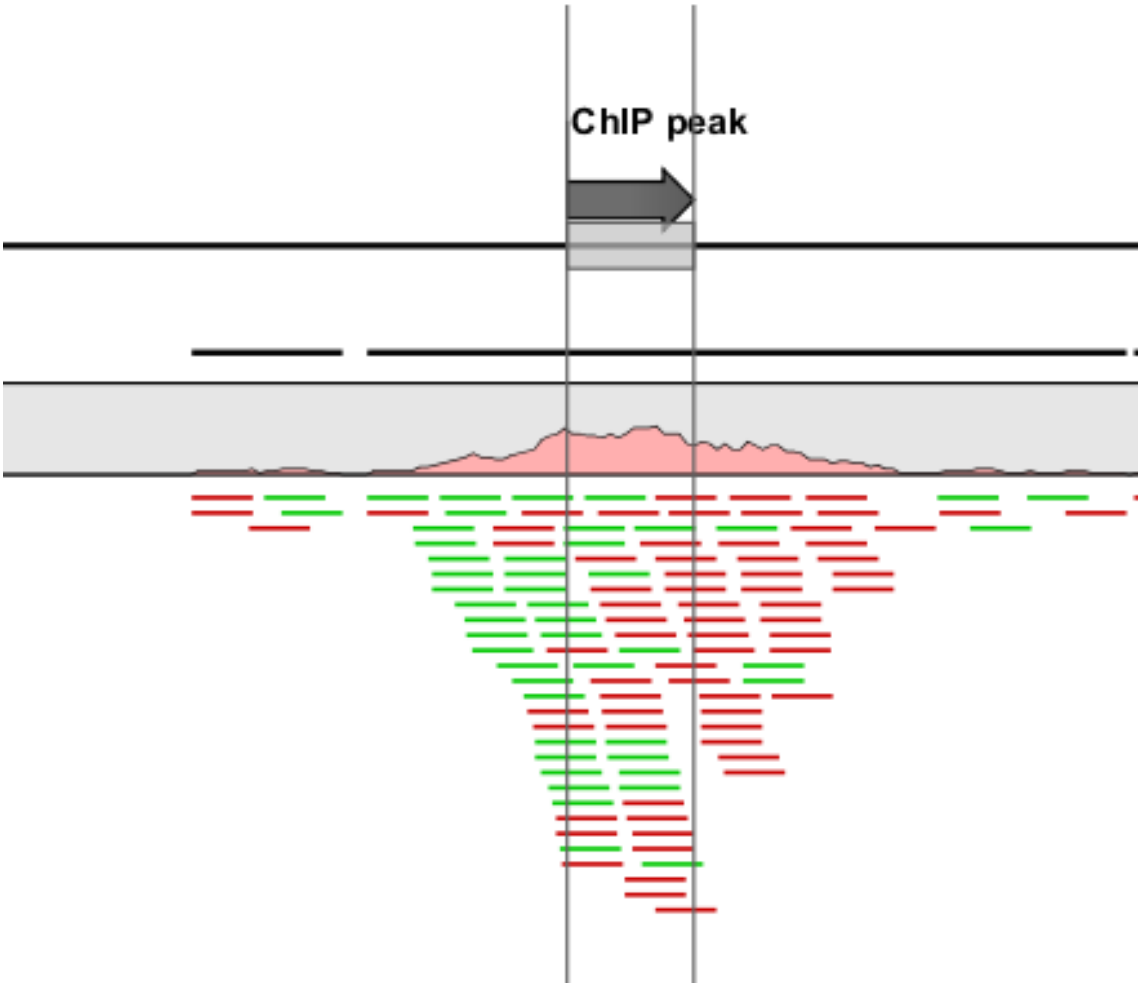


Figure 2.120: *Inspecting an annotated peak. The green lines represent forward reads and the red lines represent reverse reads.*

If you want to extract the sequence of all the peak regions to a list, you can use the **Extract Annotations** plug-in (see `http://www.clcbio.com/index.php?id=938`) to extract all annotations of the type "Binding site".

2.14 RNA-Seq analysis

Based on an annotated reference genome and mRNA sequencing reads, the *CLC Genomics Workbench* is able to calculate gene expression levels as well as discover novel exons. The key annotation types for RNA-Seq analysis of eukaryotes are of type *gene* and type *mRNA*. For prokaryotes, annotations of type *gene* are considered.

The approach taken by the *CLC Genomics Workbench* is based on [Mortazavi et al., 2008].

The RNA-Seq analysis is done in several steps: First, all genes are extracted from the reference genome (using annotations of type *gene*). Other annotations on the gene sequences are preserved (e.g. CDS information about coding sequences etc). Next, all annotated transcripts (using annotations of type *mRNA*) are extracted. If there are several annotated splice variants, they are all extracted. Note that the *mRNA* annotation type is used for extracting the exon-exon boundaries.

An example is shown in figure 2.121.

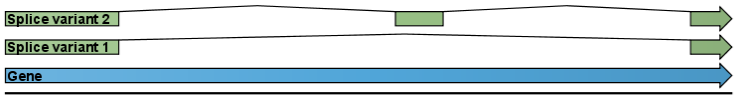


Figure 2.121: *A simple gene with three exons and two splice variants.*

This is a simple gene with three exons and two splice variants. The transcripts are extracted as shown in figure 2.122.

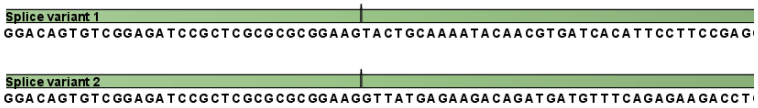


Figure 2.122: *All the exon-exon junctions are joined in the extracted transcript.*

Next, the reads are mapped against all the transcripts plus the entire gene (see figure 2.123).

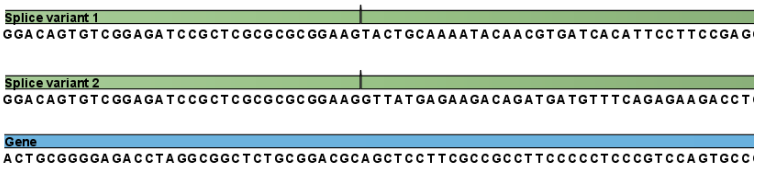


Figure 2.123: *The reference for mapping: all the exon-exon junctions and the gene.*

From this mapping, the reads are categorized and assigned to the genes (elaborated later in this section), and expression values for each gene and each transcript are calculated. After that, putative exons are identified.

Details on the process are elaborated below when describing the user interface. To start the RNA-Seq analysis analysis:

Toolbox | **High-throughput Sequencing (**) | **RNA-Seq Analysis (**)

This opens a dialog where you select the sequencing reads (not the reference genome or transcriptome). The sequencing data should be imported as described in section 2.1.

If you have several different samples that you wish to measure independently and compare afterwards, you should run the analysis in batch mode (see section **??**).

Click **Next** when the sequencing data is listed in the right-hand side of the dialog.

2.14.1 Defining reference genome and mapping settings

You are now presented with the dialog shown in figure 2.124.

At the top, there are two options concerning how the reference sequences are annotated:

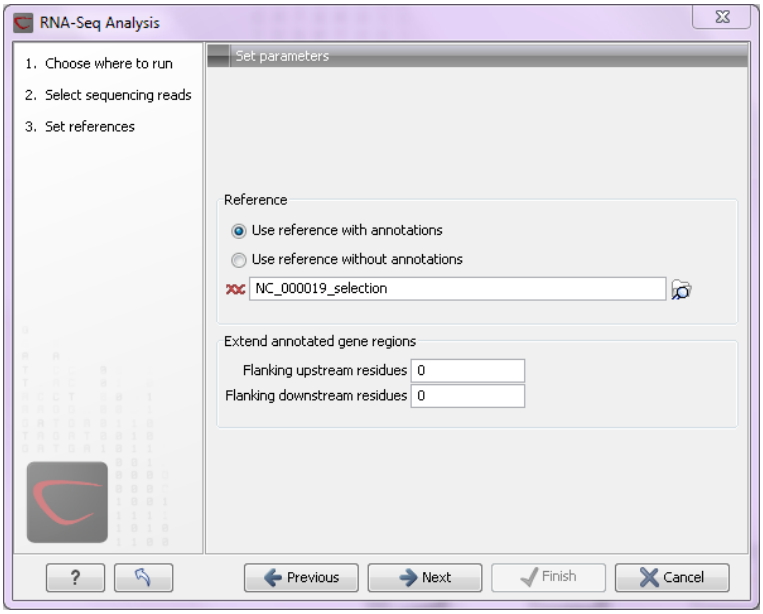


Figure 2.124: *Defining a reference genome for RNA-Seq.*

- **Use reference with annotations**. Typically, this option is chosen when you have an annotated genome sequence. Choosing this option means that gene and mRNA annotations on the sequence will be used if you choose the option **Eukarotes** in the next window. If you choose the option **Prokaryotes** in the next window, the annotations of type gene only are used. See section 2.14.1 for more information.

- **Use reference without annotations**. This option is suitable for situations like mapping back reads to un-annotated EST consensus sequences. The reference in this case is a list of sequences. A common situation is for a multi-fasta file to be imported into the Workbench to be used for this purpose. Each sequence in the list will be treated as a "gene" (or "transcript"). Note that the Workbench uses prokaryote settings here. This means that it does not look for new exons (see section 2.14.2) and it assumes that the sequences have no introns).

Just below these two options, you click to select the reference sequences.

Next, you can choose to extend the region around the gene to include more of the genomic sequence by changing the value in **Flanking upstream/downstream residues**. This also means that you are able to look for new exons before or after the known exons (see section 2.14.2).

When the reference has been defined, click **Next** and you are presented with the dialog shown in figure 2.125.

The mapping parameters are:

- **Maximum number of mismatches**. This parameter is available if you use short reads (shorter than 56 nucleotides, except for color space data which are always treated as long reads). This is the maximum number of mismatches to be allowed. Maximum value is 3, except for color space where it is 2.

- **Minimum length fraction**. For long reads, you can specify how much of the sequence should be able to map in order to include it. The default is 0.9 which means that at least

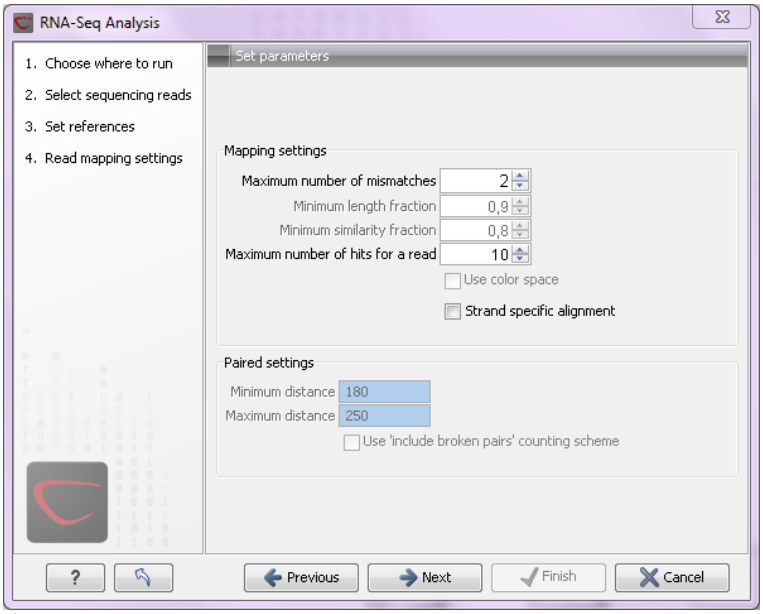


Figure 2.125: *Defining mapping parameters for RNA-Seq.*

90 % of the bases need to align to the reference.

- **Minimum similarity fraction**. This also applies to long reads and it is used to specify how exact the matching part of the read should be. When using the default setting at 0.8 and the default setting for the length fraction, it means that 90 % of the read should align with 80 % similarity in order to include the read.

- **Maximum number of hits for a read**. A read that matches to more *distinct* places in the references than the 'Maximum number of hits for a read' specified will not be mapped (the notion of *distinct* places is elaborated below). If a read matches to multiple distinct places, but below the specified maximum number, it will be randomly assigned to one of these places. The random distribution is done proportionally to the number of unique matches that the genes to which it matches have, normalized by the exon length (to ensure that genes with no unique matches have a chance of having multi-matches assigned to them, 1 will be used instead of 0, for their count of unique matches). This means that if there are 10 reads that match two different genes with equal exon length, the two reads will be distributed according to the number of unique matches for these two genes. The gene that has the highest number of unique matches will thus get a greater proportion of the 10 reads.

  Places are *distinct* in the references if they are not identical once they have been transferred back to the gene sequences. To exemplify, consider a gene with 10 transcripts and 11 exons, where all transcripts have exon 1, and each of the 10 transcripts have only one of the exons 2 to 11. Exon 1 will be represented 11 times in the references (once for the gene region and once for each of the 10 transcripts). Reads that match to exon 1 will thus match to 11 of the extracted references. However, when transferring the mappings back to the gene it becomes evident that the 11 match places are not distinct but in fact identical. In this case the read will *not* be discarded for exceeding the maximum number of hits limit, but will be mapped. In the RNA-seq action this is algorithmically done by allowing the assembler to return matches that hit in the 'maximum number of hits for a read' *plus* 'the maximum number of transcripts' that the genes have in the specified references. The

algorithm post-processes the returned matches to identify the number of distinct matches and only discards a read if this number is above the specified limit. Similarly, when a multi-match read is randomly assigned to one of it's match places, each distinct place is considered only once.

- **Strand-specific alignment**. When this option is checked, the reads will only be mapped in their forward orientation (genes on the minus strand are reverse complemented before mapping). This is useful in places where genes overlap but are on different strands because it is possible to assign the reads to the right gene. Without the strand-specific protocol, this would not be possible (see [Parkhomchuk et al., 2009]).

There is also a checkbox to **Use color space** which is enabled if you have imported a data set from a SOLiD platform containing color space information. Note that color space data is always treated as long reads, regardless of the read length.

Paired data in RNA-Seq

The *CLC Genomics Workbench* supports the use of paired data for RNA-Seq. A combination of single reads and paired reads can also be used. There are three major advantages of using paired data:

- Since the mapped reads span a larger portion of the reference, there will be less non-specifically mapped reads. This means that there is in general a greater accuracy in the expression values.
- This in turn means that there is a greater chance of accurately measuring the expression of transcript splice variants. Since single reads (especially from the short reads platforms) will usually only span one or two exons, there are many cases where the expression splice variants sharing the same exons cannot be determined accurately. With paired reads, more combinations of exons will be identified as unique for a particular splice variant.[2]
- It is possible to detect **Gene fusions** where one read in a pair maps in one gene and the other part maps in another gene. If several reads exhibit the same pattern, there is evidence of a fusion gene.

At the bottom you can specify how **Paired reads** should be handled. You can read more about how paired data is imported and handled in section 2.1.8. If the sequence list used as input for the mapping contains paired reads, this option will automatically be shown - if it contains single reads, this option will not be shown. Learn more about mapping paired data in section 2.5.3.

When counting the mapped reads to generate expression values, the *CLC Genomics Workbench* needs to decide how to handle paired reads. The standard behavior is this: if two reads map as a pair, the pair is counted as one. If the pair is broken, none of the reads are counted. The reasoning is that something is not right in this case, it could be that the transcripts are not represented correctly on the reference, or there are errors in the data. In general, more confidence is placed with an intact pair. If a combination of paired and single reads are used, "true" single reads will also count as one (the single reads that come from broken pairs will not count).

[2]Note that the *CLC Genomics Workbench* only calculates the expression of the transcripts already annotated on the reference.

In some situations it may be too strict to disregard broken pairs. This could be in cases where there is a high degree of variation compared to the reference or where the reference lacks comprehensive transcript annotations. By checking the **Use 'include broken pairs' counting scheme**, both intact and broken pairs are now counted as two. For the broken pairs, this means that each read is counted as one. Reads that are single reads as input are still counted as one.

When looking at the mappings, reads from broken pairs have a darker color than reads that are intact pairs or originally single reads.

Finding the right reference sequence for RNA-Seq

For prokaryotes, the reference sequence needed for RNA-Seq is quite simple. Either you input a genome annotated with `gene` annotations, or you input a list of genes and select the **Use reference without annotations**.

For eukaryotes, it is more complex because the Workbench needs to know the intron-exon structure as explained in in the beginning of this section. This means that you need to have a reference genome with annotations of type `mRNA` and `gene` (you can see the annotations of a sequence by opening the annotation table, see section **??**). You can obtain an annotated reference sequence in different ways:

- Download the sequences from NCBI from within the Workbench (see section **??**). Figure 2.126 shows an example of a search for the human refseq chromosomes.
- Retrieve the annotated sequences in supported format, e.g. GenBank format, and **Import** () them into the Workbench.
- Download the unannotated sequences, (e.g. in fasta format) and annotate them using a GFF/GTF file containing gene and mRNA annotations (learn more at `http://www.clcbio.com/annotate-with-gff`). Please do not over-annotate a sequence that is already marked up with gene and mRNA annotations unless you are sure that the annotation sets are exclusive. Overlapping gene and mRNA annotations will lead to useless RNA-Seq results.

  You need to make sure the annotations are the right type. GTF files from Ensembl are fully compatible with the RNA-Seq functionality of the *CLC Genomics Workbench*: `ftp://ftp.ensembl.org/pub/current_gtf/`. Note that GTF files from UCSC cannot be used for RNA-Seq since they do not have information to relate different transcript variants of the same gene.

  If you annotate your own files, please ensure that you use annotation types `gene` and, if it is a eurkarote, `mRNA`. To annotate with these types, they must be spelled correctly, and the RNA part of mRNA must be in capitals. Please see see section **??**annotation table.

2.14.2 Exon identification and discovery

Clicking **Next** will show the dialog in figure 2.127.

The choice between **Prokaryote** and **Eukaryote** is basically a matter of telling the Workbench whether you have introns in your reference. In order to select **Eukaryote**, you need to have reference sequences with annotations of the type mRNA (this is the way the Workbench expects exons to be defined - see section 2.14).

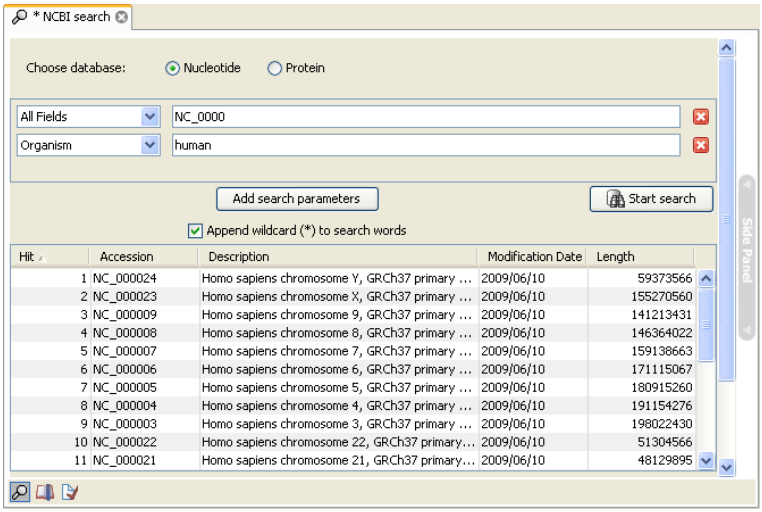


Figure 2.126: *Downloading the human genome from refseq.*

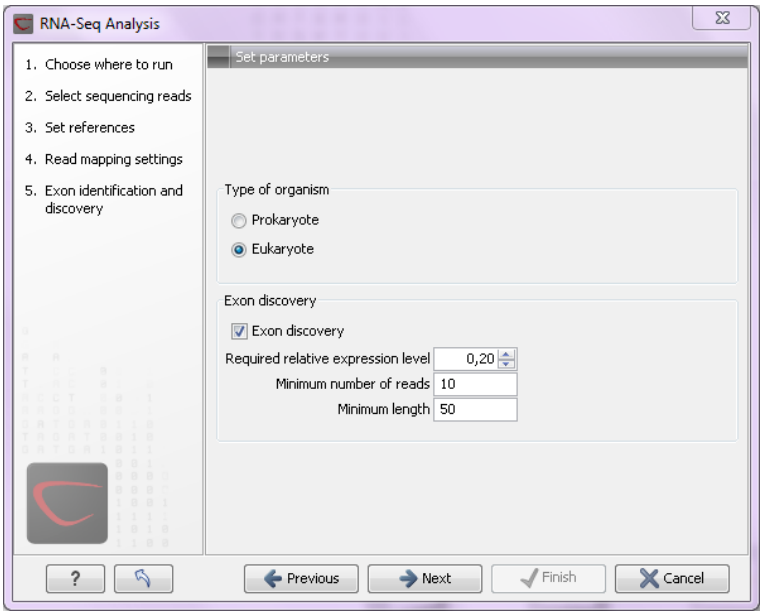


Figure 2.127: *Exon identification and discovery.*

Here you can specify the settings for discovering novel exons. The mapping will be performed against the entire gene, and by analyzing the reads located between known exons, the *CLC Genomics Workbench* is able to report new exons. A new exon has to fulfill the parameters you set:

- **Required relative expression level**. This is the expression level relative to the rest of the gene. A value of 20% means that the expression level of the new exon has to be at least 20% of that of the known exons of this gene.
- **Minimum number of reads**. While the previous option asks for the percentage relative to the general expression level of the gene, this option requires an absolute value. Just a few matching reads will already be considered to be a new exon for genes with low expression levels. This is avoided by setting a minimum number of reads here.
- **Minimum length**. This is the minimum length of an exon. There has to be overlapping reads

for the whole minimum length.

Figure 2.128 shows an example of a putative exon.

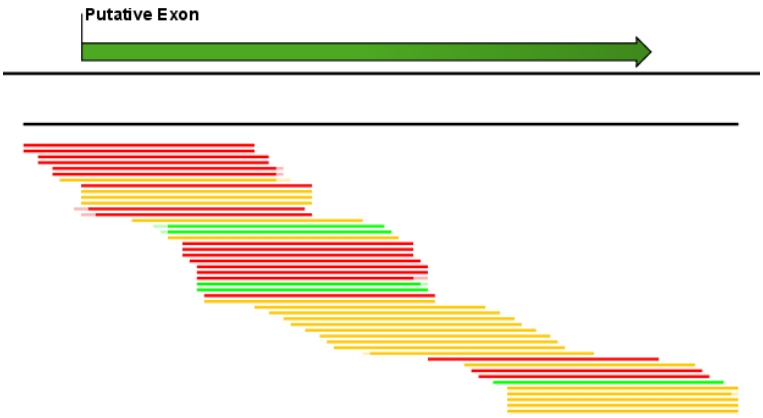


Figure 2.128: *A putative exon has been identified.*

2.14.3 RNA-Seq output options

Clicking **Next** will allow you to specify the output options as shown in figure 2.129.

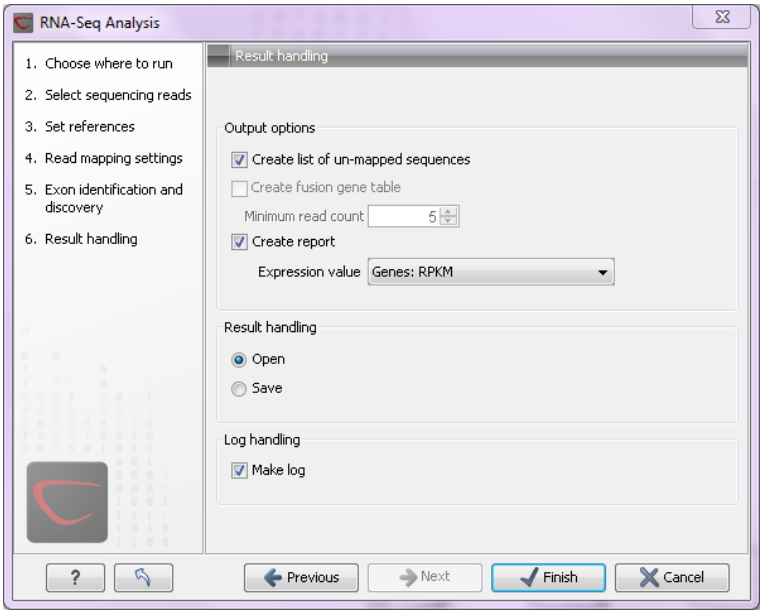


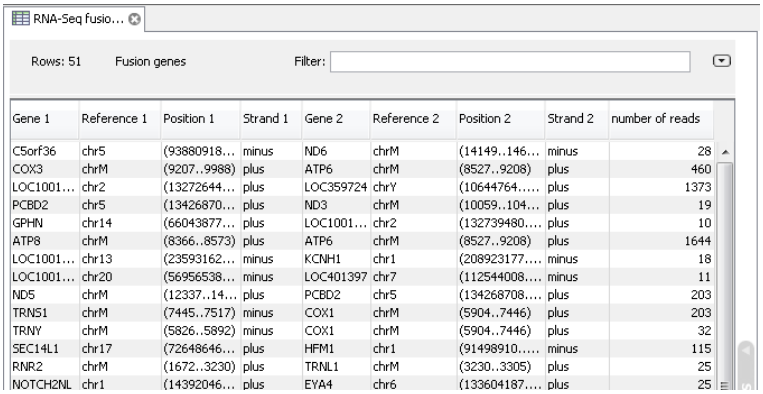
Figure 2.129: *Selecting the output of the RNA-Seq analysis.*

The standard output is a table showing statistics on each gene and the option to open the mapping (see more below). Furthermore, the expression of individual transcripts is reported (for eukaryotes). The expression measure used for further analysis can be specified as well. Per default it is set to Genes RPKM. This can also be changed at a later point (see below).

Furthermore, you can choose to create a sequence list of the **non-mapped sequences**. This could be used to do *de novo* assembly and perform BLAST searches to see if you can identify new genes or at least further investigate the results.

Gene fusion reporting

When using paired data, there is also an option to create a table summarizing the evidence for gene fusions. And example is shown in figure 2.130.



Gene 1	Reference 1	Position 1	Strand 1	Gene 2	Reference 2	Position 2	Strand 2	number of reads
C5orf36	chr5	(93880918...	minus	ND6	chrM	(14149..146...	minus	28
COX3	chrM	(9207..9988)	plus	ATP6	chrM	(8527..9208)	plus	460
LOC1001...	chr2	(13272644...	plus	LOC359724	chrY	(10644764.....	plus	1373
PCBD2	chr5	(13426870...	plus	ND3	chrM	(10059..104...	plus	19
GPHN	chr14	(66043877...	plus	LOC1001...	chr2	(132739480....	plus	10
ATP8	chrM	(8366..8573)	plus	ATP6	chrM	(8527..9208)	plus	1644
LOC1001...	chr13	(23593162...	minus	KCNH1	chr1	(208923177....	minus	18
LOC1001...	chr20	(56956538...	minus	LOC401397	chr7	(112544008....	minus	11
ND5	chrM	(12337..14...	plus	PCBD2	chr5	(134268708....	plus	203
TRNS1	chrM	(7445..7517)	minus	COX1	chrM	(5904..7446)	plus	203
TRNY	chrM	(5826..5892)	minus	COX1	chrM	(5904..7446)	plus	32
SEC14L1	chr17	(72648646...	plus	HFM1	chr1	(91498910.....	minus	115
RNR2	chrM	(1672..3230)	plus	TRNL1	chrM	(3230..3305)	plus	25
NOTCH2NL	chr1	(14392046...	plus	EYA4	chr6	(133604187....	plus	25

Figure 2.130: *An example of a gene fusion table.*

The table includes the following columns for each part of the pair:

- **Gene**. The name of the gene.
- **Reference**. The name of the reference sequence (typically the chromosome name).
- **Position**. The position of the gene.
- **Strand**. The strand of the gene.

Most importantly, the table lists the number of read pairs that are supporting the combination of genes listed. The threshold for when a combination of genes should be reported in the table can be set in the RNA-Seq dialog in figure 2.129. The default value is 5.

Note that the reporting of gene fusions is very simple and should be analyzed in much greater detail before any evidence of gene fusions can be verified. The table should be considered more of a pointer to genes to explore rather than evidence of gene fusions.

RNA-Seq report

In addition, there is an option to **Create report**. This will create a report as shown in figure 2.131.

The report contains the following information:

- **Sequence reads**. Information about the number of reads.
- **Reference sequences**. Information about the reference sequences used and their lengths and the total number of genes found in the reference.
- **Transcripts per gene**. A graph showing the number of transcripts per gene. For eukaryotes, this will be equivalent to the number of mRNA annotations per gene annotation.
- **Exons per gene**. A graph showing the number of exons per gene.

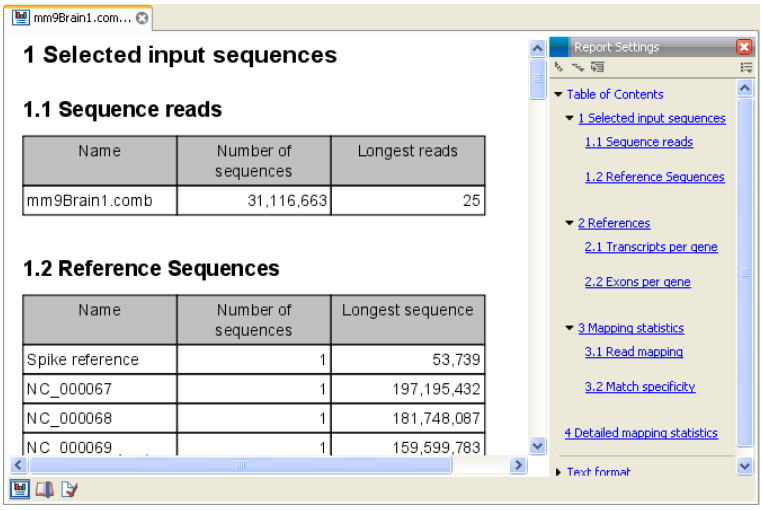


Figure 2.131: *Report of an RNA-Seq run.*

- **Exons per transcript**. A graph showing the number of exons per transcript.
- **Read mapping**. Shows statistics on:
  - **Mapped reads**. This number is divided into uniquely and non-specifically mapped reads (see the point below on match specificity for details).
  - **Unmapped reads**.
  - **Total reads**. This is the number of reads used as input.
- **Paired reads**. (Only included if paired reads are used). Shows the number of reads mapped in pairs, the number of reads in broken pairs and the number of unmapped reads.
- **Match specificity**. Shows a graph of the number of match positions for the reads. Most reads will be mapped 0 or 1 time, but there will also be reads matching more than once in the reference. This depends on the **Maximum number of hits for a read** setting in figure2.124. Note that the number of reads that are mapped 0 times includes both the number of reads that cannot be mapped at all and the number of reads that matches to more than the 'Maximum number of hits for a read' parameter that you set in the second wizard step. If paired reads are used, a separate graph is produced for that part of the data.
- **Paired distance**. (Only included if paired reads are used). Shows a graph of the distance between mapped reads in pairs.
- **Detailed mapping statistics**. This table divides the reads into the following categories.
  - **Exon-exon reads**. Reads that overlap two exons as specified in figure 2.127.
  - **Exon-intron reads**. Reads that span both an exon and an intron. If you have many of these reads, it could indicate a low splicing-efficiency or that a number of splice variants are not annotated on your reference.
  - **Total exon reads**. Number of reads that fall entirely within an exon or in an exon-exon junction.
  - **Total intron reads**. Reads that fall entirely within an intron or in the gene's flanking regions.

– **Total gene reads**. All reads that map to the gene and it's flanking regions. This is the *mapped reads* number used for calculating RPKM, see definition below.

For each category, the number of uniquely and non-specifically mapped reads are listed as well as the relative fractions. Note that all this detailed information is also available on the individual gene level in the RNA-Seq table ( )(see below). When the input data is a combination of paired and single reads, the mapping statistics will be divided into two parts.

Note that the report can be exported in pdf or Excel format.

2.14.4 Interpreting the RNA-Seq analysis result

The main result of the RNA-Seq is the reporting of expression values which is done on both the gene and the transcript level (only eukaryotes).

Gene-level expression

When you open the result of an RNA-Seq analysis, it starts in the gene-level view as shown in figure 2.132.

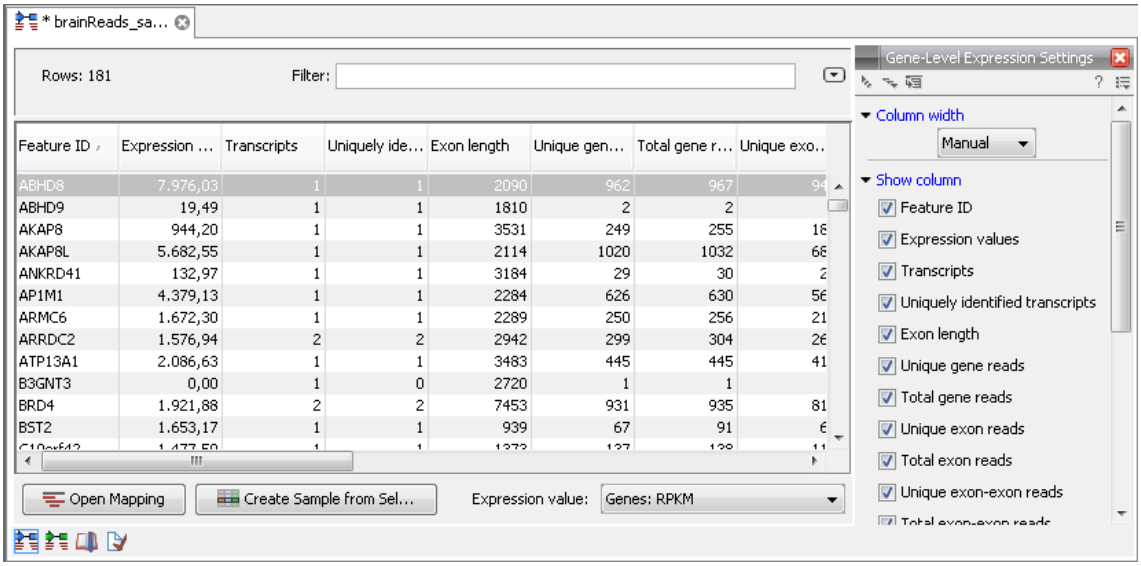


Figure 2.132: *A subset of a result of an RNA-Seq analysis on the gene level. Not all columns are shown in this figure*

The table summarizes the read mappings that were obtained for each gene (or reference). The following information is available in this table:

- **Feature ID**. This is the name of the gene.
- **Expression values**. This is based on the expression measure chosen in figure 2.129.
- **Transcripts**. The number of transcripts based on the mRNA annotations on the reference. Note that this is not based on the sequencing data - only on the annotations already on the reference sequence(s).

- **Detected transcripts**. The number of transcripts which have reads assigned (see the description of transcript-level expression below).
- **Exon length**. The total length of all exons (not all transcripts).
- **Unique gene reads**. This is the number of reads that match uniquely to the gene.
- **Total gene reads**. This is all the reads that are mapped to this gene — both reads that map uniquely to the gene and reads that matched to more positions in the reference (but fewer than the 'Maximum number of hits for a read' parameter) which were assigned to this gene.
- **Unique exon reads**. The number of reads that match uniquely to the exons (including the exon-exon and exon-intron junctions).
- **Total exon reads**. Number of reads mapped to this gene that fall entirely within an exon or in exon-exon or exon-intron junctions. As for the 'Total gene reads' this includes both uniquely mapped reads and reads with multiple matches that were assigned to an exon of this gene.
- **Unique exon-exon reads**. Reads that uniquely match across an exon-exon junction of the gene (as specified in figure 2.127). The read is only counted once even though it covers several exons.
- **Total exon-exon reads**. Reads that match across an exon-exon junction of the gene (as specified in figure 2.127). As for the 'Total gene reads' this includes both uniquely mapped reads and reads with multiple matches that were assigned to an exon-exon junction of this gene.
- **Unique intron-exon reads**. Reads that uniquely map across an exon-intron boundary. If you have many of these reads, it could indicate that a number of splice variants are not annotated on your reference.
- **Total intron-exon reads**. Reads that map across an exon-intron boundary. As for the 'Total gene reads' this includes both uniquely mapped reads and reads with multiple matches that were assigned to an exon-intron junction of this gene. If you have many of these reads, it could indicate that a number of splice variants are not annotated on your reference.
- **Exons**. The number of exons based on the mRNA annotations on the reference. Note that this is not based on the sequencing data - only on the annotations already on the reference sequence(s).
- **Putative exons**. The number of new exons discovered during the analysis (see more in section 2.14.2).
- **RPKM**. This is the expression value measured in RPKM [Mortazavi et al., 2008]: $RPKM = \frac{\textit{total exon reads}}{\textit{mapped reads(millions)} \times \textit{exon length (KB)}}$. See exact definition below. Even if you have chosen the RPKM values to be used in the **Expression values** column, they will also be stored in a separate column. This is useful to store the RPKM if you switch the expression measure. See more in section 2.14.4.
- **Median coverage**. This is the median coverage for all exons (for all reads - not only the unique ones). Reads spanning exon-exon boundaries are not included.
- **Chromosome region start**. Start position of the annotated gene.

- **Chromosome region end**. End position of the annotated gene.

Double-clicking any of the genes will open the mapping of the reads to the reference (see figure 2.133).

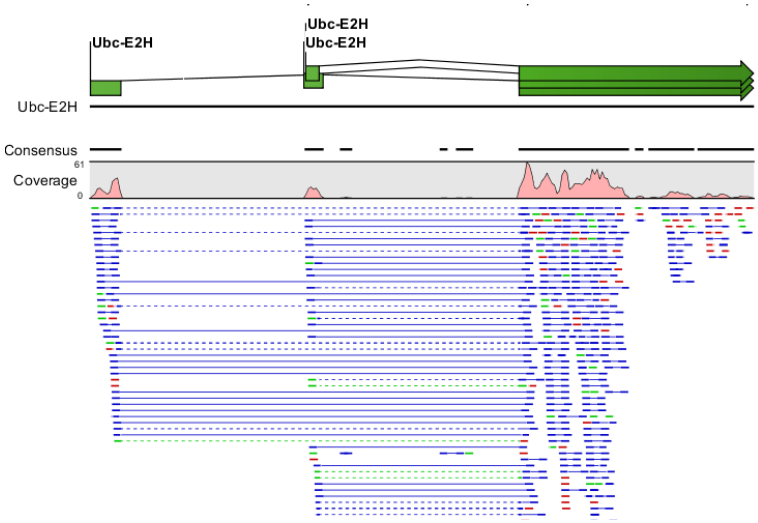


Figure 2.133: *Opening the mapping of the reads. Zoomed out to provide a better overview.*

Reads spanning two exons are shown with a dashed line between each end as shown in figure 2.133.

At the bottom of the table you can change the expression measure. Simply select another value in the drop-down list. The expression measure chosen here is the one used for further analysis. When setting up an experiment, you can specify an expression value to apply to all samples in the experiment.

The RNA-Seq analysis result now represents the expression values for the sample, and it can be further analyzed using the various tools described in chapter 3.

Transcript-level expression

In order to switch to the transcript-level expression, click the **Transcript-level expression** ( ) button at the bottom of the view. You will now see a view as shown in figure 2.134.

The following information is available in this table:

- **Feature ID**. This is the gene name with a number appended to differentiate between transcripts.
- **Expression values**. This is based on the expression measure chosen in figure 2.129.
- **Transcripts**. The number of transcripts based on the mRNA annotations on the reference. Note that this is not based on the sequencing data - only on the annotations already on the reference sequence(s).
- **Transcript length**. The total length of all exons of that particular transcript.
- **Transcript ID**. This information is retrieved from transcript_ID key on the mRNA annotation.

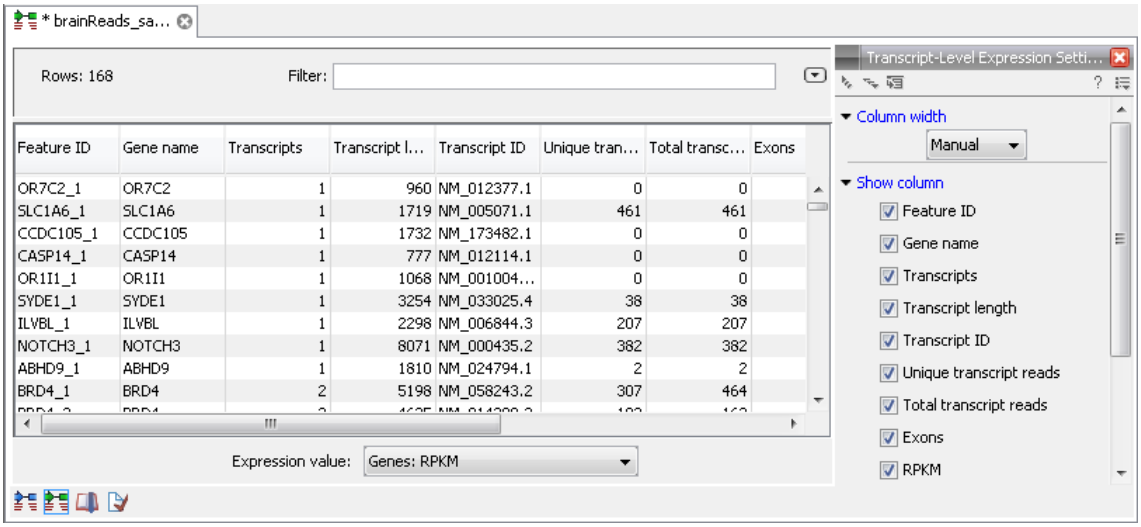


Figure 2.134: *A subset of a result of an RNA-Seq analysis on the transcript level. Not all columns are shown in this figure*

- **Unique transcript reads**. This is the number of reads in the mapping for the gene that are uniquely assignable to the transcript. This number is calculated after the reads have been mapped and both single and multi-hit reads from the read mapping may be unique transcript reads.
- **Total transcript reads**. Once the 'Unique transcript read's have been identified and their counts calculated for each transcript, the remaining (non-unique) transcript reads are assigned randomly to one of the transcripts to which they match. The 'Total transcript reads' counts are the total number of reads that are assigned to the transcript once this random assignment has been done. As for the random assignment of reads among genes, the random assignment of reads within a gene but among transcripts, is done proportionally to the 'unique transcript counts' normalized by transcript length, that is, using the RPKM (see the description of the 'Maximum number of hits for a read' option', 2.14.1). Unique transcript counts of 0 are not replaced by 1 for this proportional assignment of non-unique reads among transcripts.
- **Ratio of unique to total (exon reads**. This will show the ratio of the two columns described above. This can be convenient for filtering the results to exclude the ones where you have low confidence because of a relatively high number of non-unique transcript reads.
- **Exons**. The number of exons for this transcript. Note that this is not based on the sequencing data - only on the annotations already on the reference sequence(s).
- **RPKM**. The RPKM value for the transcript, that is, the number of reads assigned to the transcript divided by the transcript length and normalized by 'Mapped reads' (see below).
- **Relative RPKM**. The RPKM value for the transcript divided by the maximum of the RPKM values for transcripts for this gene.
- **Chromosome region start**. Start position of the annotated gene.
- **Chromosome region end**. End position of the annotated gene.

Definition of RPKM

RPKM, Reads Per Kilobase of exon model per Million mapped reads, is defined in this way [Mortazavi et al., 2008]: $RPKM = \frac{\text{total exon reads}}{\text{mapped reads(millions)} \times \text{exon length (KB)}}$.

Total exon reads This is the number in the column with header **Total exon reads** in the row for the gene. This is the number of reads that have been mapped to a region in which an exon is annotated for the gene or across the boundaries of two exons or an intron and an exon for an annotated transcript of the gene. For eukaryotes, exons and their internal relationships are defined by annotations of type mRNA.

Exon length This is the number in the column with the header **Exon length** in the row for the gene, divided by 1000. This is calculated as the sum of the lengths of all exons annotated for the gene. Each exon is included only once in this sum, even if it is present in more annotated transcripts for the gene. Partly overlapping exons will count with their full length, even though they share the same region.

Mapped reads The sum of all the numbers in the column with header **Total gene reads**. The **Total gene reads** for a gene is the total number of reads that after mapping have been mapped to the region of the gene. Thus this includes all the reads uniquely mapped to the region of the gene as well as those of the reads which match in more places (below the limit set in the dialog in figure 2.124) that have been allocated to this gene's region. A gene's region is that comprised of the flanking regions (if it was specified in figure 2.124), the exons, the introns and across exon-exon boundaries of all transcripts annotated for the gene. Thus, the sum of the total gene reads numbers is the number of mapped reads for the sample. This number can be found in the RNA-seq report's table 3.1, in the 'Total' entry of the row 'Counted fragments'. (The term 'fragment' is used in place of the term 'read', because if you analyze paired reads and have chosen the 'Default counting scheme' it is 'fragments' that is counted, rather than reads (two reads in a pair will be counted as one fragment).

2.15 Expression profiling by tags

Expression profiling by tags, also known as *tag profiling* or *tag-based transcriptomics*, is an extension of Serial analysis of gene expression (SAGE) using next-generation sequencing technologies. With respect to sequencing technology it is similar to RNA-seq (see section 2.14), but with tag profiling, you do not sequence the mRNA in full length. Instead, small tags are extracted from each transcript, and these tags are then sequenced and counted as a measure of the abundance of each transcript. In order to tell which gene's expression a given tag is measuring, the tags are often compared to a virtual tag library. This consists of the 'virtual' tags that would have been extracted from an annotated genome or a set of ESTs, had the same protocol been applied to these. For a good introduction to tag profiling including comparisons with different micro array platforms, we refer to ['t Hoen et al., 2008]. For more in-depth information, we refer to [Nielsen, 2007].

Figure 2.135 shows an example of the basic principle behind tag profiling. There are variations of this concept and additional details, but this figure captures the essence of tag profiling, namely the extraction of a tag from the mRNA based on restriction cut sites.

The *CLC Genomics Workbench* supports the entire tag profiling data analysis work flow following the sequencing:

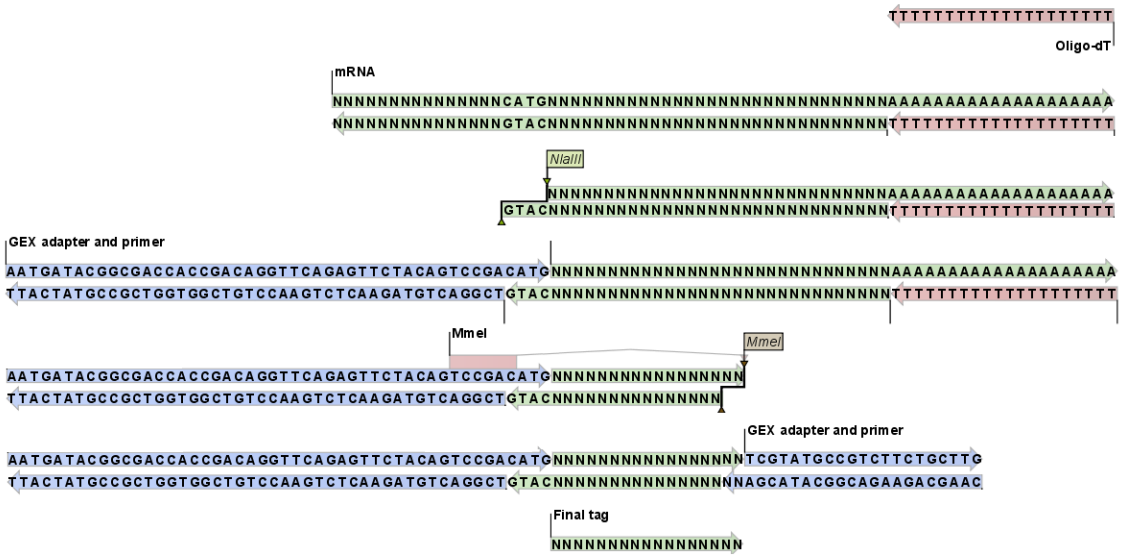


Figure 2.135: *An example of the tag extraction process. 1+2. Oligo-dT attached to a magnetic bead is used to trap mRNA. 3. The enzyme* NlaIII *cuts at CATG sites and the fragments not attached to the magnetic bead are removed. 4. An adapter is ligated to the GTAC overang. 5. The adapter includes a recognition site for* MmeI *which cuts 17 bases downstream. 6. Another adapter is added and the sequence is now ready for amplification and sequencing. 7. The final tag is 17 bp. The example is inspired by ['t Hoen et al., 2008].*

- Extraction of tags from the raw sequencing reads (tags from different samples are often barcoded and sequenced in one pool).
- Counting tags including a sequencing-error correction algorithm.
- Creating a virtual tag list based on an annotated reference genome or an EST-library.
- Annotating the tag counts with gene names from the virtual tag list.

Each of the steps in the work flow are described in details below.

2.15.1 Extract and count tags

First step in the analysis is to import the data (see section 2.1).

The next step is to extract the tags and count them:

> **Toolbox** | **High-throughput Sequencing** ( ) | **Expression Profiling by Tags** ( ) | **Extract and Count Tags** ( )

This will open a dialog where you select the reads that you have imported. Click **Next** when the sequencing data is listed in the right-hand side of the dialog.

This dialog is where you define the elements in your reads. An example is shown in figure 2.136.

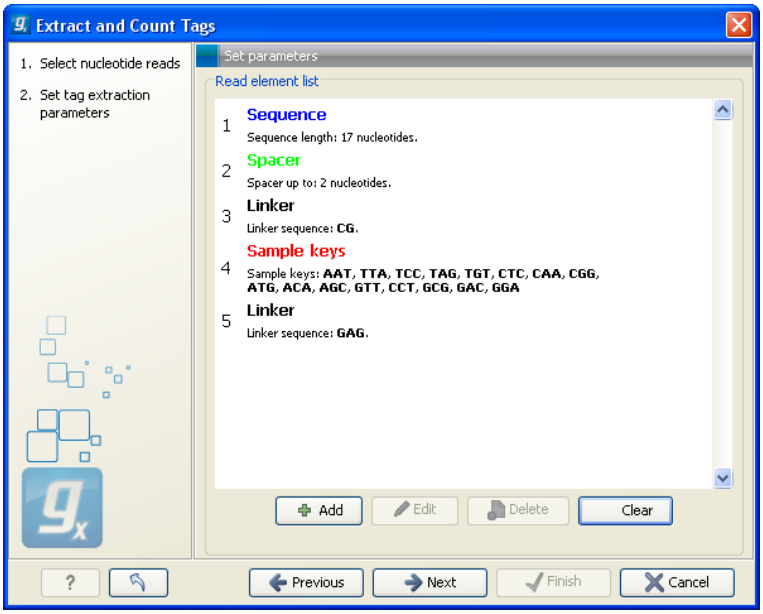


Figure 2.136: *Defining the elements that make up your reads.*

By defining the order and size of each element, the Workbench is now able to both separate samples based on bar codes and extract the tag sequence (i.e. removing linkers, bar codes etc). The elements available are:

Sequence This is the part of the read that you want to use as your final tag for counting and annotating. If you have tags of varying lengths, add a spacer afterwards (see below).

Sample keys Here you input a comma-separated list of the sample keys used for identifying the samples (also referred to as "bar codes"). If you have not pooled and bar coded your data, simply omit this element.

Linker This is a known sequence that you know should be present and do not want to be included in your final tag.

Spacer This is also a sequence that you do not want to include in your final tag, but whereas the linker is defined by its sequence, the spacer is defined by its length. Note that the length defines the maximum length of the spacer. Often not all tags will be exactly the same length, and you can use this spacer as a buffer for those tags that are longer than what you have defined as your sequence. In the example in figure 2.136, the tag length is 17 bp, but a spacer is added to allow tags up to 19 bp. Note that the part of the read that is extracted and used as the final tag does not include the spacer sequence. In this way you homogenize the tag lengths which is usually desirable because you want to count short and long tags together.

When you have set up the right order of your elements, click **Next** to set parameters for counting tags as shown in figure 2.137.

At the top, you can specify how to tabulate (i.e. count) the tags:

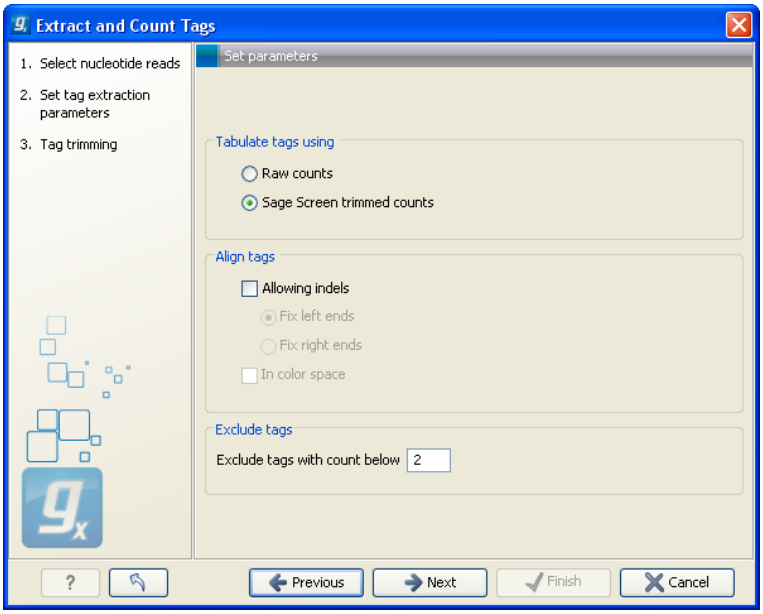


Figure 2.137: *Setting parameters for counting tags.*

Raw counts This will produce the count for each tag in the data.

Sage Screen trimmed counts This will produce trimmed tag counts. The trimmed tag counts are obtained by applying an implementation of the SAGEscreen method ([Akmaev and Wang, 2004]) to the raw tag counts. In this procedure, raw counts are trimmed using probabilistic reasoning. In this procedure, if a tag with low count has a neighboring tag with high count, and it is likely, based on the estimated mutation rate, that the low count tags have arisen through sequencing errors of the tags with higher count, the count of the less abundant tag will be attributed to the higher abundant neighboring tag. The implementation of the SAGEscreen method is highly efficient and provides considerable speed and memory improvements.

Next, you can specify additional parameters for the alignment that takes place when the tags are tabulated:

Allowing indels Ticking this box means that, when SAGEscreen is applied, neighboring tags will, in addition to tags which differ by nucleotide substitutions, also include tags with insertion or deletion differences.

Color space This option is only available if you use data generated on the SOLiD platform. Checking this option will perform the alignment in color space which is desirable because sequencing errors can be corrected. Learn more about color space in section 2.8.

At the bottom you can set a minimum threshold for tags to be reported. Although the SAGEscreen trimming procedure will reduce the number of erroneous tags reported, the procedure only handles tags that are neighbors of more abundant tags. Because of sequencing errors, there will be some tags that show extensive variation. There will by chance only be a few copies of these tags, and you can use the minimum threshold option to simply discard tags. The default value is two which means that tags only occurring once are discarded. This setting is a trade-off between removing bad-quality tags and still keeping tags with very low expression (the ability to

measure low levels of mRNA is one of the advantages of tag profiling over for example micro arrays ['t Hoen et al., 2008]).

Note! If more samples are created, SAGEscreen and the minimum threshold cut-offs will be applied to the cumulated counts (i.e. all tags for all samples).

Clicking **Next** allows you to specify the output of the analysis as shown in figure 2.138.

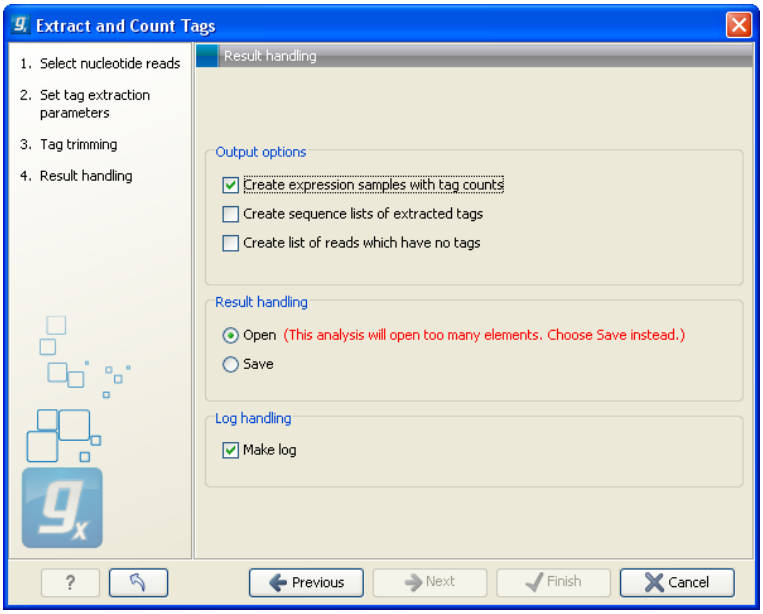


Figure 2.138: *Output options.*

The options are:

Create expression samples with tag counts This is the primary result showing all the tags and respective counts (an example is shown in figure 2.139). For each sample defined via the bar codes, there will be an expression sample like this. Note that all samples have the same list of tags, even if the tag is not present in the given sample (i.e. there will be tags with count 0 as shown in figure 2.139). The expression samples can be used in further analysis by the expression analysis tools (see chapter 3).

Create sequence lists of extracted tags This is a simple sequence list of all the tags that were extracted. The list is simple with no counts or additional information.

Create list of reads which have no tags This list contains the reads from which a tag could not be extracted. This is most likely bad quality reads with sequencing errors that make them impossible to group by their bar codes. It can be useful for troubleshooting if the amount of real tags is smaller than expected.

Finally, a log can be shown of the extraction and count process. The log gives useful information such as the number of tags in each sample and the number of reads without tags.

2.15.2 Create virtual tag list

Before annotating the tag sample ( ) created above, you need to create a so-called *virtual tag list*. The list is created based on a DNA sequence or sequence list holding, an annotated genome

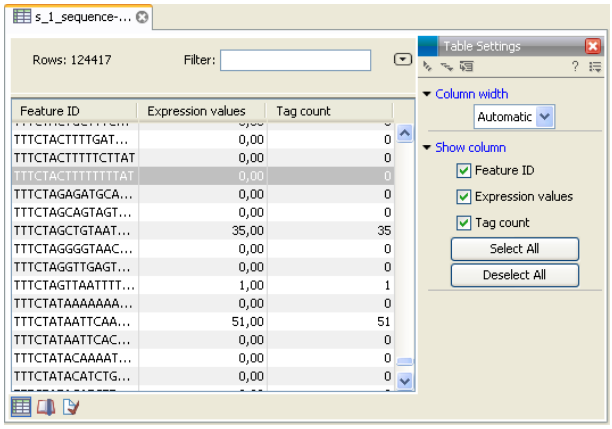


Figure 2.139: *The tags have been extracted and counted.*

or a list of ESTs. It represents the tags that you would expect to find in your experimental data (given the reference genome or EST list reflects your sample). To create the list, you specify the restriction enzyme and tag length to be used for creating the virtual list.

The virtual tag list can be saved and used to annotate experiments made from tag-based expression samples as shown in section 2.15.3.

To create the list:

> **Toolbox** | **High-throughput Sequencing** () | **Expression Profiling by Tags** () | **Create Virtual Tag List** ()

This will open a dialog where you select one or more annotated genomic sequences or a list of ESTs. Click **Next** when the sequences are listed in the right-hand side of the dialog.

This dialog is where you specify the basis for extracting the virtual tags (see figure 2.140).

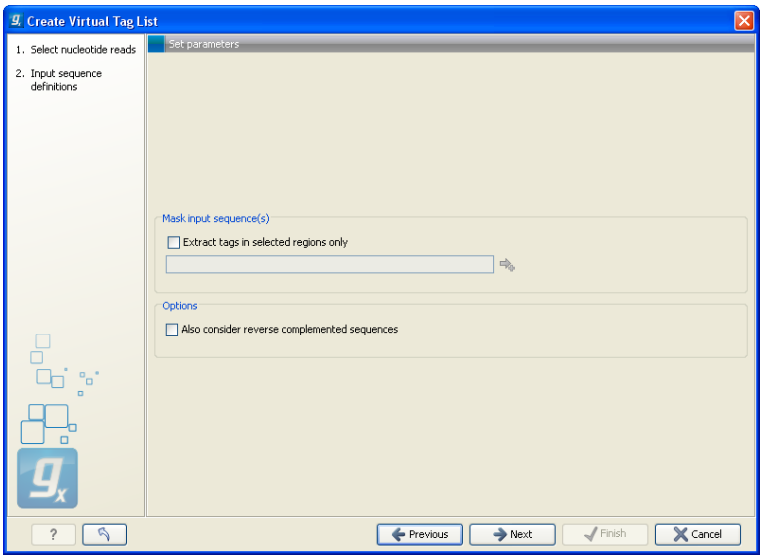


Figure 2.140: *The basis for the extraction of reads.*

At the top you can choose to extract tags based on annotations on your sequences by checking the **Extract tags in selected areas only** option. This option is applicable if you are using annotated genomes (e.g. Refseq genomes). Click the small button () to the right to display a dialog

showing all the annotation types in your sequences. Select the annotation type representing your transcripts (usually *mRNA* or *Gene*). The sequence fragments covered by the selected annotations will then be extracted from the genomic sequence and used as basis for creating the virtual tag list.

If you use a sequence list where each sequence represents your transcript (e.g. an EST library), you should not check the **Extract tags in selected areas only** option.

Below, you can choose to include the reverse complement for creating virtual tags. This is mainly used if there is uncertainty about the orientation of sequences in an EST library.

Clicking **Next** allows you to specify enzymes and tag length as shown in figure 2.141.

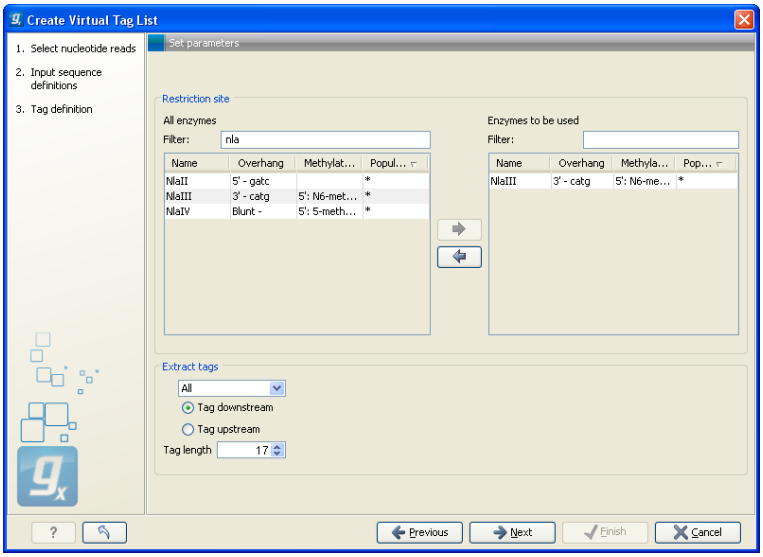


Figure 2.141: *Defining restriction enzyme and tag length.*

At the top, find the enzyme used to define your tag and double-click to add it to the panel on the right (as it has been done with *NlaIII* in figure 2.141). You can use the filter text box so search for the enzyme name.

Below, there are further options for the tag extraction:

Extract tags When extracting the virtual tags, you have to decide how to handle the situation where one transcript has several cut sites. In that case there would be several potential tags. Most tag profiling protocols extract the 3'-most tag (as shown in the introduction in figure 2.135), so that would be one way of defining the tags in the virtual tag list. However, due to non-specific cleavage, new alternative splicing or alternative polyadenylation ['t Hoen et al., 2008], tags produced from internal cut sites of the transcript are also quite frequent. This means that it is often not enough to consider the 3'-most restriction site only. The list lets you select either **All**, **External 3'** which is the 3'-most tag or **External 5'** which is the 5' most tag (used by some protocols, for example CAGE - cap analysis of gene expression - see [Maeda et al., 2008]). The result of the analysis displays whether the tag is found at the 3' end or if it is an internal tag (see more below).

Tag downstream/upstream When the cut site is found, you can specify whether the tag is then found downstream or upstream of the site. In figure 2.135, the tag is found downstream.

Tag length The length of the tag to be extracted. This should correspond to the sequence length defined in figure 2.136.

Clicking **Next** allows you to specify the output of the analysis as shown in figure 2.142.

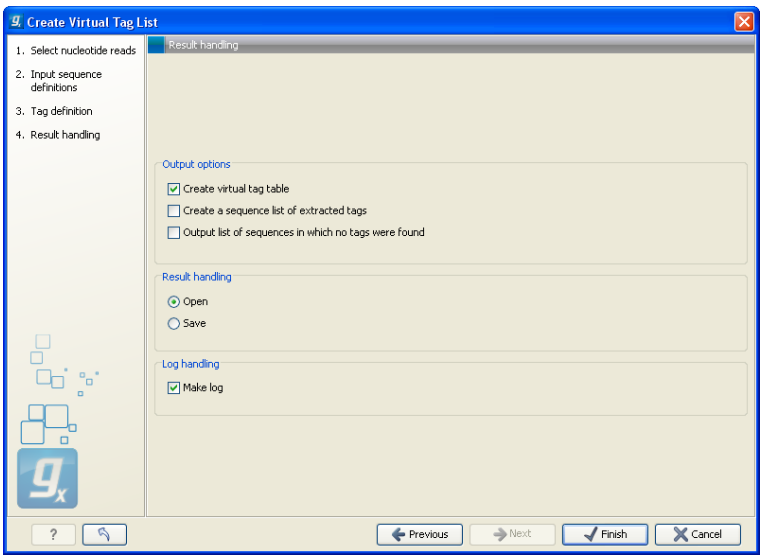


Figure 2.142: *Output options.*

The output options are:

Create virtual tag table This is the primary result listing all the virtual tags. The table is explained in detail below.

Create a sequence list of extracted tags All the extracted tags can be represented in a raw sequence list with no additional information except the name of the transcript. You can e.g. **Export** ( ) this list to a fasta file.

Output list of sequences in which no tags were found The transcripts that do not have a cut site or where the cut site is so close to the end that no tag could be extracted are presented in this list. The list can be used to inspect which transcripts you could potentially fail to measure using this protocol. If there are tags for all transcripts, this list will not be produced.

In figure 2.143 you see an example of a table of virtual tags that have been produced using the **3' external** option described above.

The first column lists the tag itself. This is the column used when you annotate your tag count samples or experiments (see section 2.15.3). Next follows the name of the tag's origin transcript. Sometimes the same tag is seen in more than one transcript. In that case, the different origins are separated by /// as it is the case for the tag of *LOC100129681 /// BST2* in figure 2.143. The row just below, *UBA52*, has the same name listed twice. This is because the analysis was based on mRNA annotations from a Refseq genome where each splice variant has its own mRNA annotation, and in this case the *UBA52* gene has two mRNA annotations including the same tag.

The last column is the description of the transcript (which is either the sequence description if you use a list of un-annotated sequences or all the information in the annotation if you use annotated sequences).

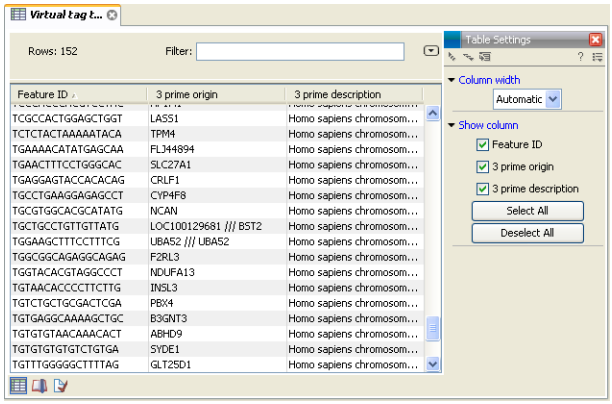


Figure 2.143: *A virtual tag table of 3' external tags.*

The example shown in figure 2.143 is the simplest case where only the 3' external tags are listed. If you choose to list **All** tags, the table will look like figure 2.144.

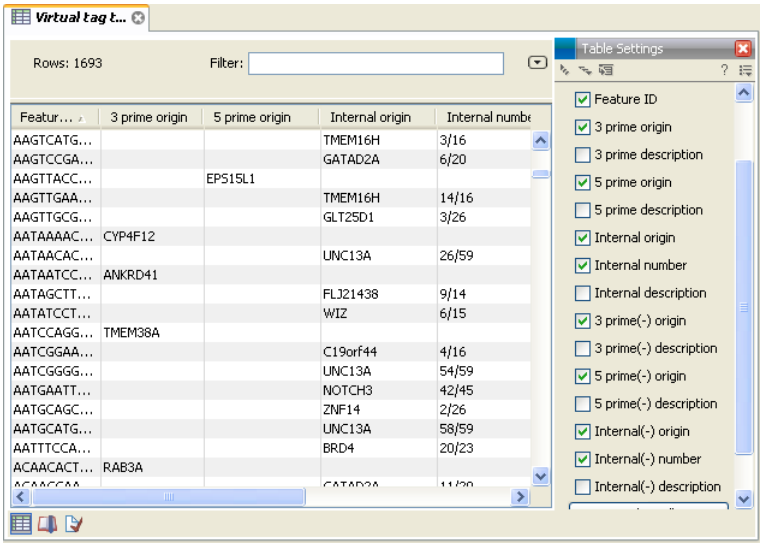


Figure 2.144: *A virtual tag table where all tags have been extracted. Note that some of the columns have been ticked off in the Side Panel.*

In addition to the information about the 3' tags, there are additional columns for 5' and internal tags. For the internal tags there is also a numbering, see for example the top row in figure 2.144 where the *TMEM16H* tag is tag number 3 out of 16. This information can be used to judge how close to the 3' end of the transcript the tag is. As mentioned above, you would often expect to sequence more tags from cut sites near the 3' end of the transcript.

If you have chosen to include reverse complemented sequences in the analysis, there will be an additional set of columns for the tags of the other strand, denoted with a (-).

You can use the advanced table filtering (see section **??**) to interrogate the number of tags with specific origins (e.g. define a filter where 3' origin != and then leave the text field blank).

2.15.3 Annotate tag experiment

Combining the tag counts ( ) from the experimental data (see section 2.15.1) with the virtual tag list ( ) (see above) makes it possible to put gene or transcript names on the tag counts.

The Workbench simply compares the tags in the experimental data with the virtual tags and transfers the annotations from the virtual tag list to the experimental data.

This is done on an experiment level (experiments are collections of samples with defined groupings, see section 3.1):

> **Toolbox** | **High-throughput Sequencing** ( ) | **Expression Profiling by Tags** ( ) | **Annotate Tag Experiment** ( )

You can also access this functionality at the bottom of the **Experiment table** ( ) as shown in figure 2.145.

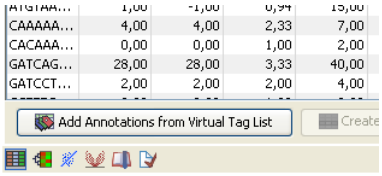


Figure 2.145: *You can annotate an experiment directly from the experiment table.*

This will open a dialog where you select a virtual tag list ( ) and an experiment ( ) of tag-based samples. Click **Next** when the elements are listed in the right-hand side of the dialog.

This dialog lets you choose how you want to annotate your experiment (see figure 2.146).

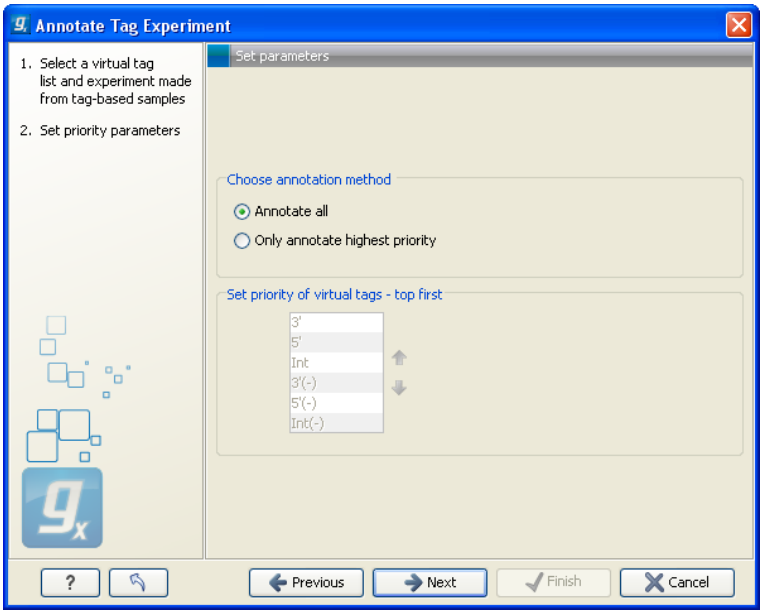


Figure 2.146: *Defining the annotation method.*

If a tag in the virtual tag list has more than one origin (as shown in the example in figure 2.144) you can decide how you want your experimental data to be annotated. There are basically two options:

Annotate all This will transfer all annotations from the virtual tag. The type of origin is still preserved so that you can see if it is a 3' external, 5' external or internal tag.

Only annotate highest priority This will look for the highest priority annotation and only add this to the experiment. This means that if you have a virtual tag with a 3' external and an

internal tag, only the 3' external tag will be annotated (using the default prioritization). You can define the prioritization yourself in the table below: simply select a type and press the **up** ( ) and **down** ( ) arrows to move it up and down in the list. Note that the priority table is only active when you have selected **Only annotate highest priority**.

Click **Next** to choose how you want to tags to be aligned (see figure 2.147). When the tags

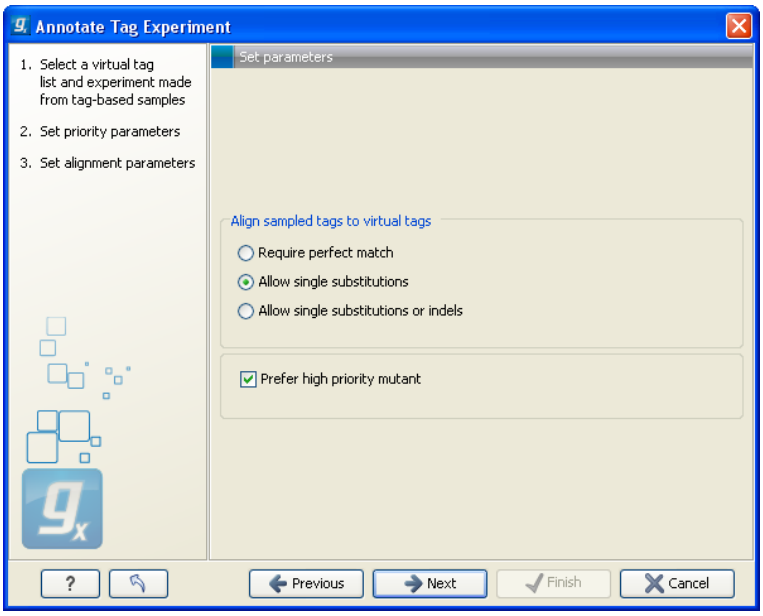


Figure 2.147: *Settings for aligning the tags.*

from the virtual tag list are compared to your experiment, the tags are matched using one of the following options:

Require perfect match The tags need to be identical to be matched.

Allow single substitutions If there is up to one mismatch in the alignment, the tags will still be matched. If there is a perfect match, single substitutions will not be considered.

Allow single substitutions or indels Similar to the previous option, but now single-base insertions and deletions are also allowed. Perfect matches are preferred to single-base substitutions which are preferred to insertions, which are again preferred to deletions. [3]

If you select any of the two options allowing mismatches or mismatches and indels, you can also choose to **Prefer high priority mutant**. This option is only available if you have chosen to annotate highest priority only in the previous step (see figure 2.146). The option is best explained through an example: In this case, you have a tag that matches perfectly to an internal tag from the virtual tag list. Imagine that in this example, you have prioritized the annotation so that 3' external tags are of higher priority than internal tags. The question is now if you want to accept the perfect match (of a low priority virtual tag) or the high-priority virtual tag with one mismatch? If you check the **Prefer high priority mutant**, the 3' external tag in the example above will be used rather than the perfect match.

Click **Next** if you wish to adjust how to handle the results (see section **??**). If not, click **Finish**. . This will add extra annotation columns to the experiment. The extra columns corresponds to the

[3] Note that if you use color space data, only color errors are allowed when choosing anything but perfect match.

```
Tag from experiment:                          CGTATCAATCGATTAC
                                              ||||||||||||||||
Tag1 from virtual tag list (internal):        CGTATCAATCGATTAC
                                              | ||||||||||||||
Tag1 from virtual tag list (3' external):     CCTATCAATCGATTAC
```

columns found in your virtual tag list. If you have chosen to annotate highest priority-only, there will only be information from one origin-column for each tag as shown in figure 2.148.

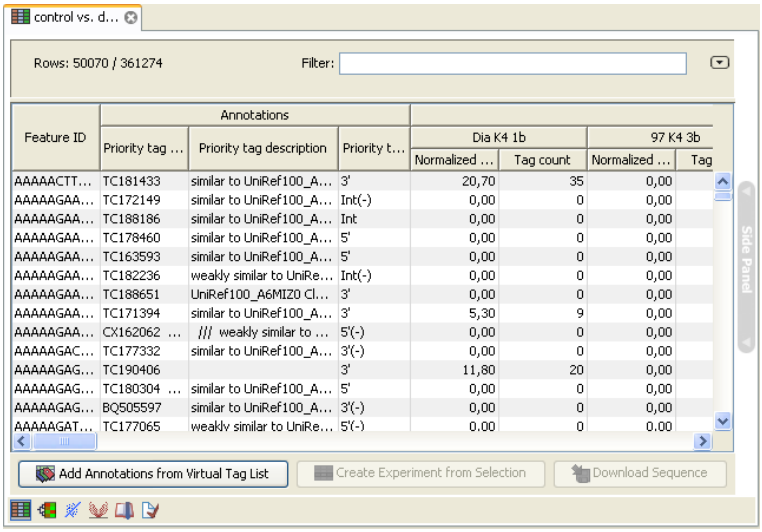


Figure 2.148: *An experiment annotated with prioritized tags.*

2.16 Small RNA analysis

The small RNA analysis tools in *CLC Genomics Workbench* are designed to facilitate trimming of sequencing reads, counting and annotating of the resulting tags using miRBase or other annotation sources and performing expression analysis of the results. The tools are general and flexible enough to accommodate a variety of data sets and applications within small RNA profiling, including the counting and annotation of both microRNAs and other non-coding RNAs from any organism. Both Illumina, 454 and SOLiD sequencing platforms are supported. For SOLiD, adapter trimming and annotation is done in color space.

The annotation part is designed to make special use of the information in miRBase but more general references can be used as well.

There are generally two approaches to the analysis of microRNAs or other smallRNAs: (1) count the different types of small RNAs in the data and compare them to databases of microRNAs or other smallRNAs, or (2) map the small RNAs to an annotated reference genome and count the numbers of reads mapped to regions which have smallRNAs annotated. The approach taken by *CLC Genomics Workbench* is (1). This approach has the advantage that it does not require an annotated genome for mapping — you can use the sequences in miRBase or any other sequence list of smallRNAs of interest to annotate the small RNAs. In addition, small RNAs that would not have mapped to the genome (e.g. when lacking a high-quality reference genome or if the RNAs have not been transcribed from the host genome) can still be measured and their expression be compared. The methods and tools developed for *CLC Genomics Workbench* are inspired by the

findings and methods described in [Creighton et al., 2009], [Wyman et al., 2009], [Morin et al., 2008] and [Stark et al., 2010].

In the following, the tools for working with small RNAs are described in detail. Look at the tutorials on http://www.clcbio.com/tutorials to see examples of analyzing specific data sets.

2.16.1 Extract and count

First step in the analysis is to import the data (see section 2.1).

The next step is to extract and count the small RNAs to create a *small RNA sample* that can be used for further analysis (either annotating or analyzing using the expression analysis tools):

Toolbox | **High-throughput Sequencing** ( ) | **Small RNA Analysis** ( ) | **Extract and Count** ( )

This will open a dialog where you select the sequencing reads that you have imported. Click **Next** when the sequencing data is listed in the right-hand side of the dialog. Note that if you have several samples, they should be processed separately.

This dialog (see figure 2.149) is where you specify whether the reads should be trimmed for adapter sequences prior to counting. It is often necessary to trim off remainders of adapter sequences from the reads before counting.

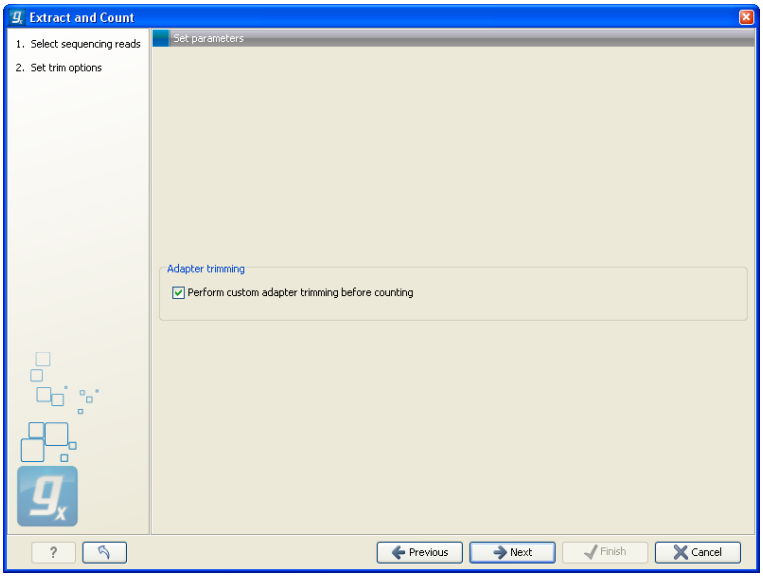


Figure 2.149: *Specifying whether adapter trimming is needed.*

When you click **Next**, you will be able to specify how the trim should be performed as shown in figure 2.150.

If you have chosen not to trim the reads for adapter sequence, you will see figure 2.151 instead.

The trim options shown in figure 2.150 are the same as described under adapter trim in section 2.3.2. Please refer to this section for more information.

It should be noted that if you expect to see part of adapters in your reads, you would typically choose **Discard when not found** as the action. By doing this, only reads containing the adapter sequence will be counted as small RNAs in the further analysis. If you have a data set where the

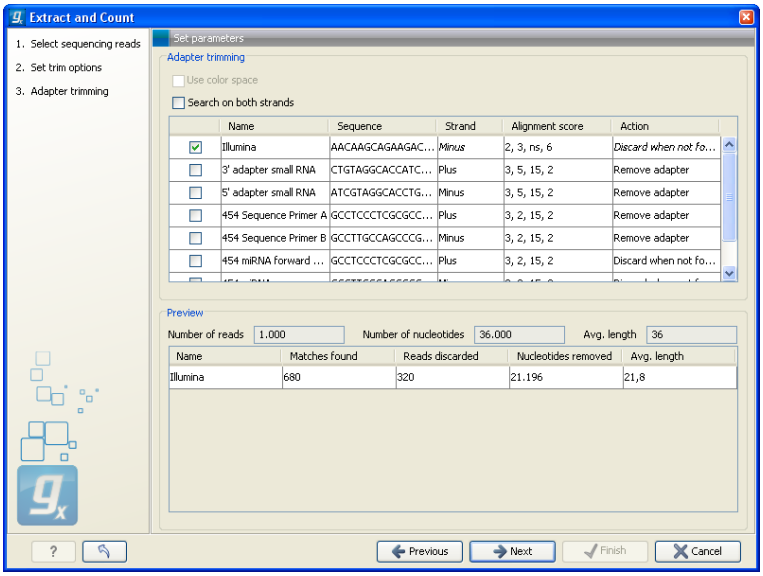


Figure 2.150: *Setting parameters for adapter trim.*

adapter may be there or not you would choose **Remove adapter**.

Note that all reads will be trimmed for ambiguity symbols such as N before the adapter trim.

Clicking **Next** allows you to specify additional options regarding trimming and counting as shown in figure 2.151.

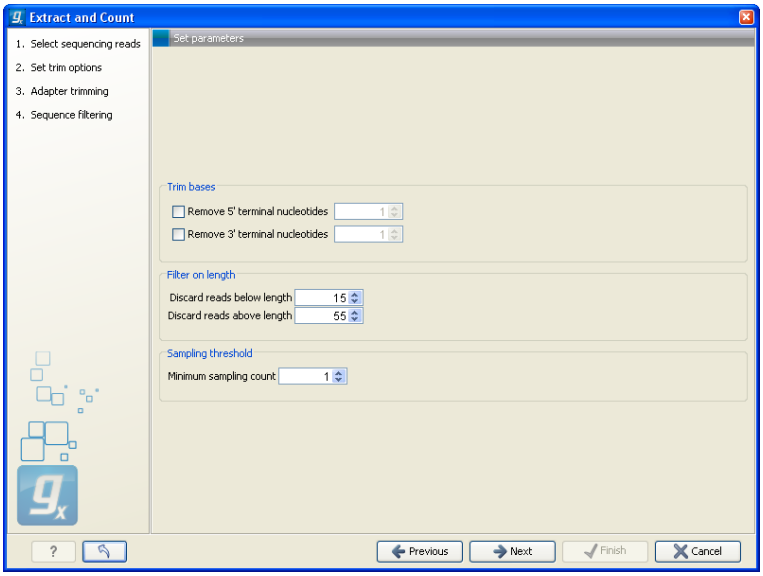


Figure 2.151: *Defining length interval and sampling threshold.*

At the top you can choose to **Trim bases** by specifying a number of bases to be removed from either the 3' or the 5' end of the reads. Below, you can specify the minimum and maximum lengths of the small RNAs to be counted (this is the length after trimming). The minimum length that can be set is 15 and the maximum is 55.

At the bottom, you can specify the **Minimum sampling count**. This is the number of copies of the small RNAs (tags) that are needed in order to include it in the resulting count table (the small RNA sample). The actual counting is very simple and relies on **perfect match** between the reads

to be counted together[4]. This also means that a count threshold of 1 will include a lot of unique tags as a result of sequencing errors. In order to set the threshold right, the following should be considered:

- If the sample is going to *be annotated*, annotations may be found for the tags resulting from sequencing errors. This means that there is no negative effect of including tags with a low count in the output.
- When using *un-annotated sequences* for discovery of novel small RNAs, it may be useful to apply a higher threshold to eliminate the noise from sequencing errors. However, this can be done at a later stage by filtering the sample and creating a sub-set.
- When multiple samples are compared, it is interesting to know if one tag which is abundant in one sample is also found in another, even at a very low number. In this case, it is useful to include the tags with very low counts, since they may become more trustworthy in combination with information from other samples.
- Setting the count threshold higher will reduce the size of the sample produced which will reduce the memory and disk usage when working with the results.

Clicking **Next** allows you to specify the output of the analysis as shown in 2.152.

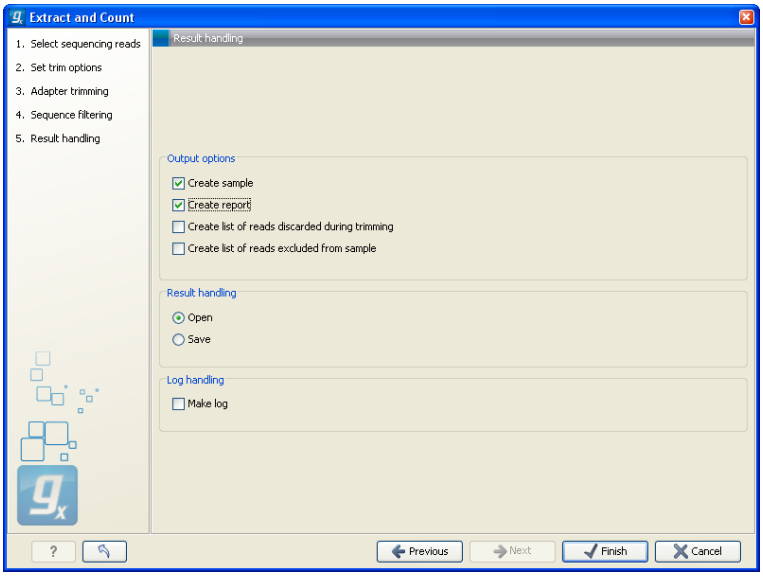


Figure 2.152: *Output options.*

The options are:

Create sample This is the primary result showing all the tags and respective counts (an example is shown in figure 2.153). Each row represents a tag with the actual sequence as the feature ID and a column with **Length** and **Count**. The actual count is based on 100 % similarity[5]. The sample can be used in further analysis by the expression analysis tools (see chapter 3) in the "raw" form, or you can annotate it (see below). The tools for working with the data in the sample are described in section 2.16.4.

[4]Note that you can identify variants of the same miRNA when annotating the sample (see below).
[5]Note that you can identify variants of the same miRNA when annotating the sample (see below).

Create report This will create a summary report as described below.

Create list of reads discarded during trimming This list contains the reads where no adapter was found (when choosing **Discard when not found** as the action).

Create list of reads excluded from sample This list contains the reads that passed the trimming but failed to meet the sampling thresholds regarding minimum/maximum length and number of copies.

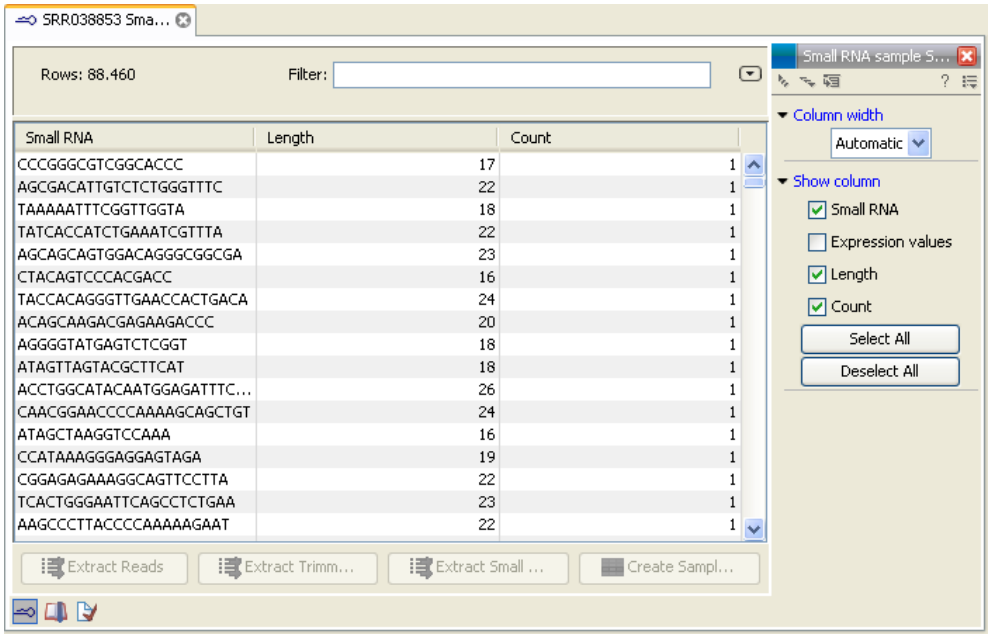


Figure 2.153: *The tags have been extracted and counted.*

The summary report includes the following information (an example is shown in figure 2.154):

Trim summary Shows the following information for each input file:

- Number of reads in the input.
- Average length of the reads in the input.
- Number of reads after trim. The difference between the number of reads in the input and this number will be the number of reads that are discarded by the trim.
- Percentage of the reads that pass the trim.
- Average length after trim. When analyzing miRNAs, you would expect this number to be around 22. If the number is significantly lower or higher, it could indicate that the trim settings are not right. In this case, check that the trim sequence is correct, that the strand is right, and adjust the alignment scores. Sometimes it is preferable to increase the minimum scores to get rid of low-quality reads. The average length after trim could also be somewhat larger than 22 if your sequenced data contains a mixture of miRNA and other (longer) small RNAs.

Read length before/after trimming Shows the distribution of read lengths before and after trim. The graph shown in figure 2.154 is typical for miRNA sequencing where the read lengths after trim peaks at 22 bp.

Trim settings The trim settings summarized. Note that ambiguity characters will automatically be trimmed.

Detailed trim results This is described under adapter trim in section 2.3.2.

Tag counts The number of tags and two plots showing on the x-axis the counts of tags and on the y-axis the number of tags for which this particular count is observed. The plot is in a zoomed version where only the lower part of the y-axis is shown to make it possible to see the numbers of tags higher counts.

1 Trim summary

Name	Number of reads	Avg.length	Number of reads after trim	Percentage trimmed	Avg.length after trim
SRR038853	2,070,061	36.0	1,720,241	83.1%	21.9

2 Read length before / after trimming

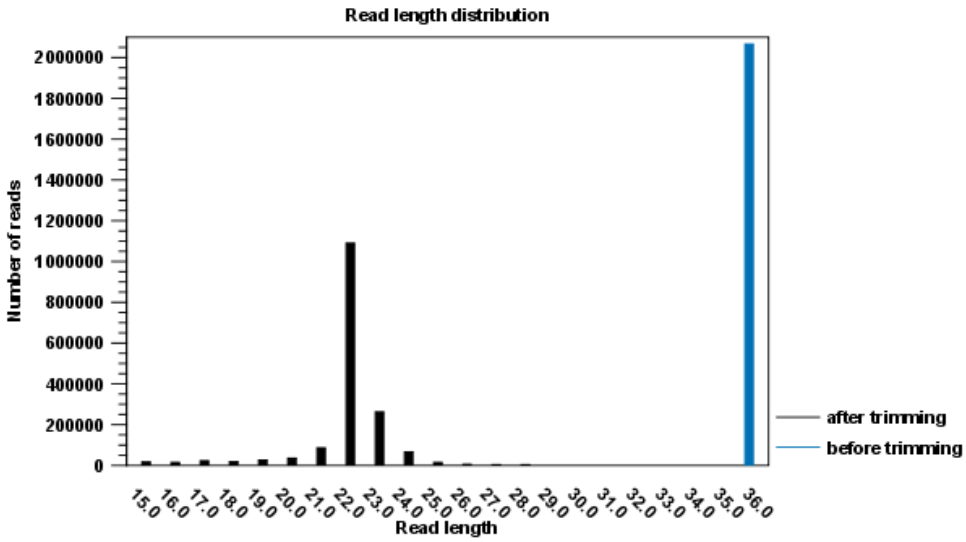


Figure 2.154: *A summary report of the counting.*

2.16.2 Downloading miRBase

In order to make use of the additional information about mature regions on the precursor miRNAs in miRBase, you need to use the integrated tool to download miRBase rather than downloading it from `http://www.mirbase.org/`:

> **Toolbox** | **High-throughput Sequencing** ( ) | **Small RNA Analysis** ( ) | **Download miRBase** ( )

This will download a sequence list with all the precursor miRNAs including annotations for mature regions. The list can then be selected when annotating the samples with miRBase (see section 2.16.3).

The downloaded version will always be the latest version (it is downloaded from `ftp://mirbase.org/pub/mirbase/CURRENT/miRNA.dat.gz`). Information on the version number of miRBase is also available in the **History** ( ) of the downloaded sequence list, and when using this for annotation, the annotated samples will also include this information in their **History** ( ).

2.16.3 Annotating and merging small RNA samples

The small RNA sample produced when counting the tags (see section 2.16.1) can be enriched by *CLC Genomics Workbench* by comparing the tag sequences with annotation resources such as miRBase and other small RNA annotation sources. Note that the annotation can also be performed on an experiment, set up from small RNA samples (see section 3.1.2).

Besides adding annotations to known small RNAs in the sample, it is also possible to merge variants of the same small RNA to get a cumulated count. When initially counting the tags, the Workbench requires that the trimmed reads are identical for them to be counted as the same tag. However, you will often see different variants of the same miRNA in a sample, and it is useful to be able to count these together. This is also possible using the tool to annotate and merge samples.

Toolbox | **High-throughput Sequencing ( )** | **Small RNA Analysis ( )** | **Annotate and Merge Counts ( )**

This will open a dialog where you select the small RNA samples ( ) to be annotated. Note that if you have included several samples, they will be processed separately but summarized in one report providing a good overview of all samples. You can also input **Experiments** ( ) (see section 3.1.2) created from small RNA samples. Click **Next** when the data is listed in the right-hand side of the dialog.

This dialog (figure 2.155) is where you define the annotation resources to be used.

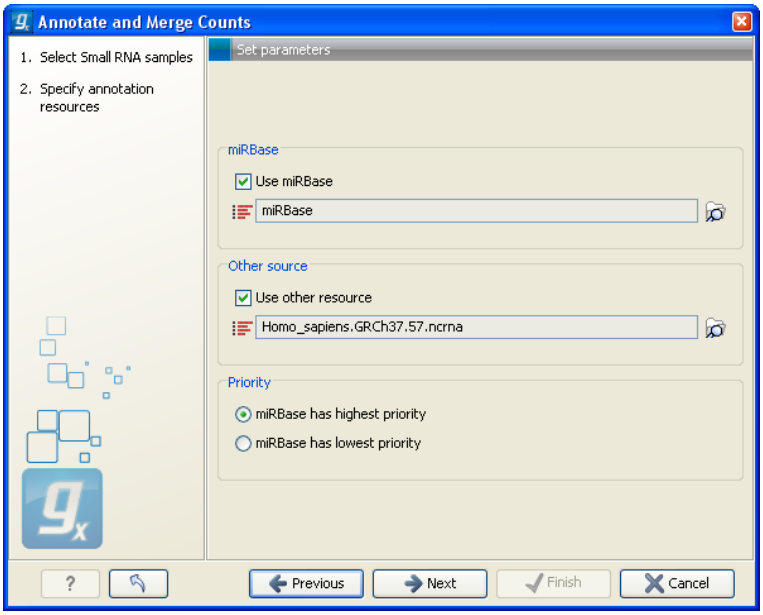


Figure 2.155: *Defining annotation resources.*

There are two ways of providing annotation sources:

- Downloading miRBase using the integrated download tool (explained in section 2.16.2).
- Importing a list of sequences, e.g. from a fasta file. This could be from Ensembl, e.g. `ftp://ftp.ensembl.org/pub/release-57/fasta/homo_sapiens/ncrna/Homo_sapiens.GRCh37.57.ncrna.fa.gz` or from ncRNA.org: `http://www.ncrna.org/frnadb/files/ncrna.zip`.

Note: We recommend using the integrated download tool to import miRBase. Although it is possible to import it as a fasta file, the same options with regards to species will not be available if you import from a file.

The downloaded miRBase file contains all precursor sequences from the latest version of miRBase `http://www.mirbase.org/` including annotations defining the mature regions (see an example in figure 2.156).

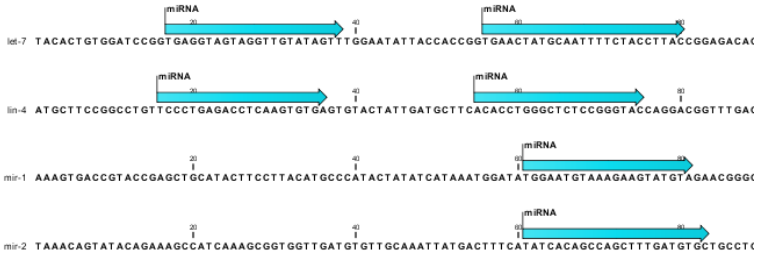


Figure 2.156: *Some of the precursor miRNAs from miRBase have both 3' and 5' mature regions (previously referred to as mature and mature*) annotated (as the two first in this list).*

This means that it is possible to have a more fine-grained classification of the tags using miRBase compared to a simple fasta file resource containing the full precursor sequence. This is the reason why the miRBase annotation source is specified separately in figure 2.155.

At the bottom of the dialog, you can specify whether miRBase should be prioritized over the additional annotation resource. The prioritization is explained in detail later in this section. To prioritize one over the other can be useful when there is redundant information (e.g. if you have an additional source that also contains all the miRNAs from miRBase and you prefer the miRBase annotations when possible).

When you click **Next**, you will be able to choose which species from miRBase should be used and in which order (see figure 2.157). Note that if you have not selected a miRBase annotation source, you will go directly to the next step shown in figure 2.158.

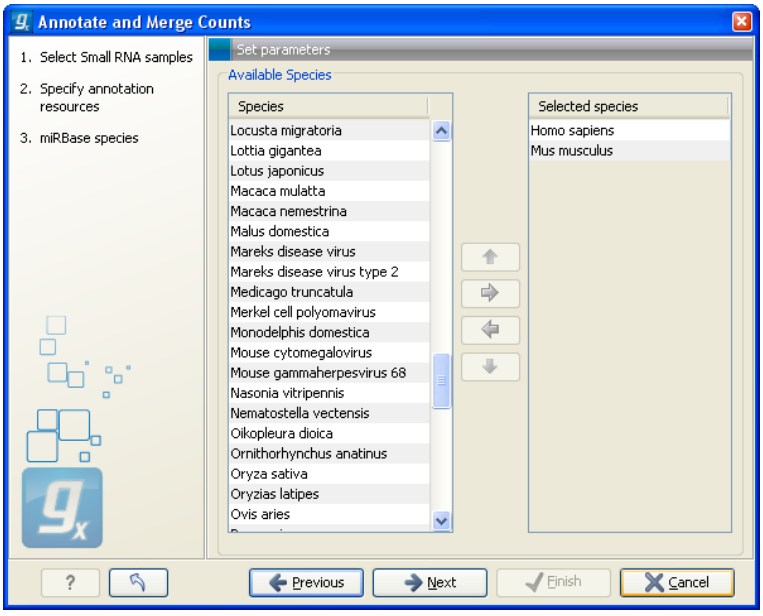


Figure 2.157: *Defining and prioritizing species in miRBase.*

To the left, you see the list of species in miRBase. This list is dynamically created based on the information in the miRBase file. Using the arrow button ( ) you can add species to the right-hand panel. The order of the species is important since the tags are annotated iteratively based on the order specified here. This means that in the example in figure 2.157, a human miRNA will preferred over mouse, even if they are identical in sequence (the prioritization is elaborated below). The up and down arrows ( )/ ( ) can be used to change the order of species.

When you click **Next**, you will be able to specify how the alignment of the tags against the annotation sources should be performed (see figure 2.158).

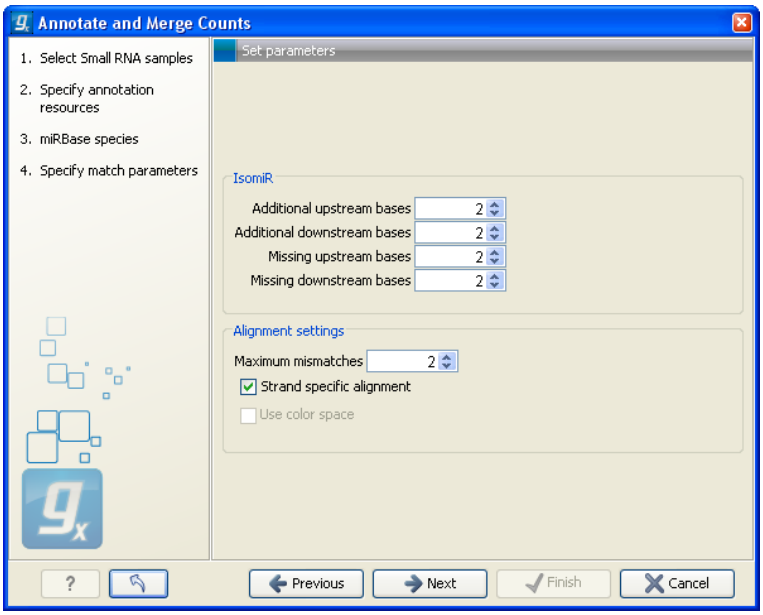


Figure 2.158: *Setting parameters for aligning.*

The panel at the top is active only if you have chosen to annotate with miRBase. It is used to define the requirements to the alignment of a read for it to be counted as a mature or mature* tag:

Additional upstream bases This defines how many bases the tag is allowed to extend the annotated mature region at the 5' end and still be categorized as mature.

Additional downstream bases This defines how many bases the tag is allowed to extend the annotated mature region at the 3' end and still be categorized as mature.

Missing upstream bases This defines how many bases the tag is allowed to miss at the 5' end compared to the annotated mature region and still be categorized as mature.

Missing downstream bases This defines how many bases the tag is allowed to miss at the 3' end compared to the annotated mature region and still be categorized as mature.

At the bottom of the dialog you can specify the **Maximum mismatches** (default value is 2). Furthermore, you can specify if the alignment and annotation should be performed in **color space** which is available when your small RNA sample is based on SOLiD data. [6] Finally, you can choose

[6] Note that this option is only going to make a difference for tags with low counts. Since the actual tag counting in the first place is done based on perfect matches, the highly abundant tags are not likely to have sequencing errors, and aligning in color space does not add extra benefit for these.

whether the tags should be aligned against both strands of the reference or only the positive strand. Usually it is only necessary to align against the positive strand.

At this point, a more elaborate explanation of the annotation algorithm is needed. The short read mapping algorithm in the *CLC Genomics Workbench* is used to map all the tags to the reference sequences which comprise the full precursor sequences from miRBase and the sequence lists chosen as additional resources. The mapping is done in several rounds: the first round is done requiring a perfect match, the second allowing one mismatch, the third allowing two mismatches etc. No gaps are allowed. The number of rounds depend on the number of mismatches allowed[7] (default is two which means three rounds of read mapping, see figure 2.158).

After each round of mapping, the tags that are mapped will be removed from the list of tags that continue to the next round. This means that a tag mapping with perfect match in the first round will not be considered for the subsequent one-mismatch round of mapping.

Following the mapping, the tags are classified into the following categories according to where they match.

- Mature 5' exact
- Mature 5' super
- Mature 5' sub
- Mature 5' sub/super
- Mature 3' exact
- Mature 3' super
- Mature 3' sub
- Mature 3' sub/super
- Precursor
- Other

All these categories except *Other* refer to hits in miRBase. For hits on mirBase sequences we distinguish between where on the sequences the tags match. The mirBase sequences may have up to two mature micro RNAs annotated. We refer to a mature miRNA that is located closer (or equally close) to the 5' end than to the 3' end as 'Mature 5''. A mature miRNA that is located closer to the 3' end is referred to as 'Mature 3''. *Exact* means that the tag matches exactly to the annotated mature 5' or 3' region; *Sub* means that the observed tag is shorter than the annotated mature 5' or mature 3'; *super* means that the observed tag is longer than the annotated mature 5' or mature 3'. The combination *sub/super* means that the observed tag extends the annotation in one end and is shorter at the other end. *Precursor* means that the tag matches on a mirBase sequence, but outside of the annotated mature region(s). The *Other* category is for hits in the other resources (the information about resource is also shown in the output).

An example of an alignment is shown in figure 2.159 using the same alignment settings as in figure 2.158.

[7] For color space, the maximum number of mismatches is 2.

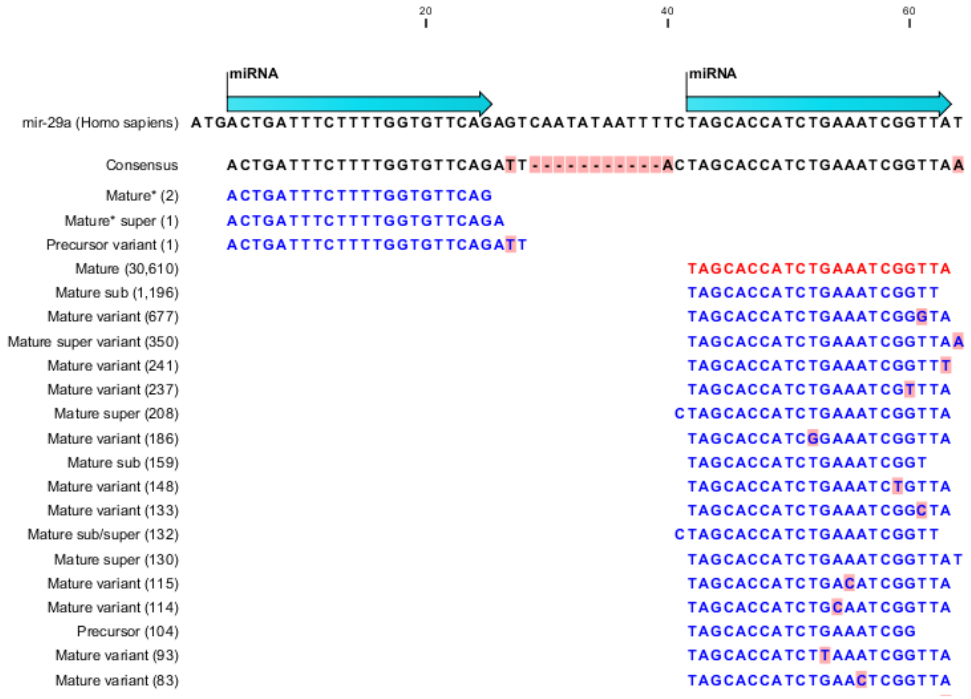


Figure 2.159: *Alignment of length variants of mir-30a.*

The two tags at the top are both classified as *mature 5' super* because they cover and extend beyond the annotated mature 5' RNA. The third tag is identical to the annotated mature 5'. The fourth tag is classified as *precursor* because it does not meet the requirements on length for it to be counted as a mature hit — it lacks 6 bp compared to the annotated mature 5' RNA. The fifth tag is classified as mature 5' sub because it also lacks one base but stays within the threshold defined in figure 2.158.

If a tag has several hits, the list above is used for prioritization. This means that e.g. a *Mature 5' sub* is preferred over a *Mature 3' exact*. Note that if miRBase was chosen as lowest priority (figure 2.155), the *Other* category will be at the top of the list. All tags mapping to a miRBase reference without qualifying to any of the mature 5' and mature 3' types will be typed as *Precursor*.

In case you have selected more than one species for miRBase annotation (e.g. Homo Sapiens and Mus Musculus) the following rules for adding annotations apply:

1. If a tag has hits with the same priority for both species, the annotation for the top-prioritized species will be added.

2. Read category priority is stronger than species category priority: If a read is a higher priority match for a mouse miRBase sequence than it is for a human miRBase sequence the annotation for the mouse will be used

Clicking **Next** allows you to specify the output of the analysis as shown in 2.160.

The options are:

Create unannotated sample All the tags where no hit was found in the annotation source are included in the unannotated sample. This sample can be used for investigating novel

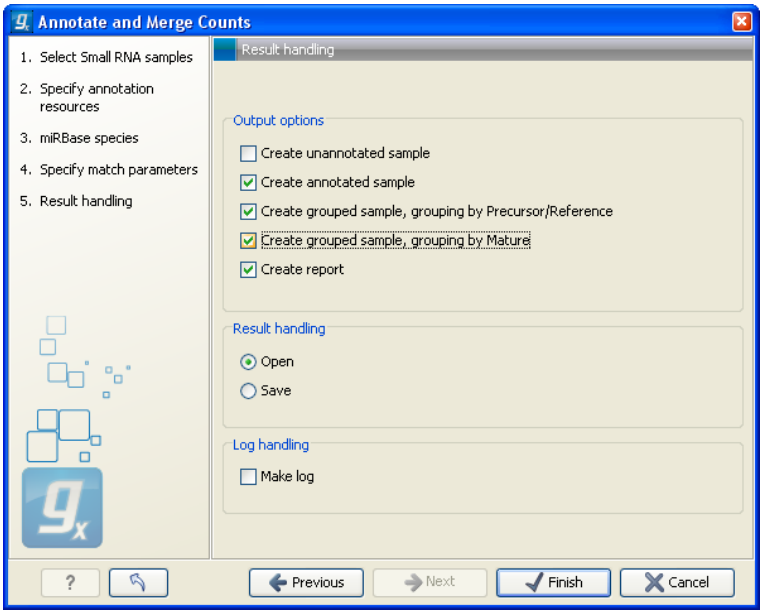


Figure 2.160: *Output options.*

miRNAs, see section 2.16.5. No extra information is added, so this is just a subset of the input sample.

Create annotated sample This will create a sample as described in section 2.16.4. In this sample, the following columns have been added to the counts.

Name This is the name of the annotation sequence in the annotation source. For miRBase, it will be the names of the miRNAs (e.g. *let-7g* or *mir-147*), and for other source, it will be the name of the sequence.

Resource This is the source of the annotation, either miRBase (in which case the species name will be shown) or other sources (e.g. Homo_sapiens.GRCh37.57.ncrna).

Match type The match type can be exact or variant (with mismatches) of the following types:

- Mature 5'
- Mature 5' super
- Mature 5' sub
- Mature 5' sub/super
- Mature 3'
- Mature 3' super
- Mature 3' sub
- Mature 3' sub/super
- Precursor
- Other

Mismatches The number of mismatches.

Note that if a tag has two equally prioritized hits, they will be shown with // between the names. This could be e.g. two precursor sequences sharing the same mature sequence (also see the sample grouped on mature below).

Create grouped sample, grouping by Precursor/Reference This will create a sample as described in section 2.16.4. All variants of the same reference sequence will be merged to create one expression value for all.

Expression values. The expression value can be changed at the bottom of the table. The default is to use the counts in the mature 5' column.

Name. The name of the reference. For miRBase this will then be the name of the precursor.

Resource. The name of the resource that the reference comes from.

Exact mature 5'. The number of exact mature 5' reads.

Mature 5'. The number of all mature 5' reads including sub, super and variants.

Unique exact mature 5'. In cases where one tag has several hits (as denoted by the // in the ungrouped annotated sample as described above), the counts are distributed evenly across the references. The difference between *Exact mature 5'* and *Unique exact mature 5'* is that the latter only includes reads that are unique to this reference.

Unique mature 5'. Same as above but for all mature 5's, including sub, super and variants.

Exact mature 3'. Same as above, but for mature 3'.

Mature 3'. Same as above, but for mature 3'.

Unique exact mature '3. Same as above, but for mature 3'.

Unique mature '3. Same as above, but for mature 3'.

Exact other. Exact match in the resources chosen besides miRBase.

Other. All matches in the resources chosen besides miRBase including variants. The last two numbers are the only ones used when the reference is not from miRBase.

Total. The total number of tags mapped and classified to the precursor/reference sequence.

Create grouped sample, grouping by Mature This will create a sample as described in section 2.16.4. This is also a grouped sample, but in addition to grouping based on the same reference sequence, the tags in this sample are grouped on the same mature 5'. This means that two precursor variants of the same mature 5' miRNA are merged. Note that it is only possible to create this sample when using miRBase as annotation resource (because the Workbench has a special interpretation of the miRBase annotations for mature as described previously). To find identical mature 5' miRNAs, the Workbench compares all the mature 5' sequences and when they are identical, they are merged. The names of the precursor sequences merged are all shown in the table.

Expression values. The expression value can be changed at the bottom of the table. The default is to use the counts in the mature 5' column.

Name. The name of the reference. When several precursor sequences have been merged, all the names will be shown separated by //.

Resource. The species of the reference.

Exact mature 5'. The number of exact mature 5' reads.

Mature 5'. The number of all mature 5' reads including sub, super and variants.

Unique exact mature 5'. In cases where one tag has several hits (as denoted by the // in the ungrouped annotated sample as described above), the counts are distributed evenly across the references. The difference between *Exact mature 5'* and *Unique exact mature 5'* is that the latter only includes reads that are unique to one of the precursor sequences that are represented under this mature 5' sequence.

Unique mature 5'. Same as above but for all mature 5's, including sub, super and variants.

Create report. A summary report described below.

The summary report includes the following information (an example is shown in figure 2.161):

Summary Shows the following information for each input sample:

- Number of small RNAs(tags) in the input.
- Number of annotated tags (number and percentage).
- Number of reads in the sample (one tag can represent several reads)
- Number of annotated reads (number and percentage).

Resources Shows how many matches were found in each resource:

- Number of sequences in the resource.
- Number of sequences where a match was found (i.e. this sequence has been observed at least once in the sequencing data).

Reads Shows the number of reads that fall into different categories (there is one table per input sample). On the left hand side are the annotation resources. For each resource, the count and percentage of reads in that category are shown. Note that the percentage are relative to the overall categories (e.g. the miRBase reads are a percentage of all the *annotated* reads, not all reads). This is information is shown for each mismatch level.

Small RNAs Similar numbers as for the reads but this time for each small RNA tag and without mismatch differentiation.

Read count proportions A histogram showing, for each interval of read counts, the proportion of annotated (respectively, unannotated) small RNAs with a read count in that interval. Annotated small RNAs may be expected to be associated with higher counts, since the most abundant small RNAs are likely to be known already.

Annotations (miRBase) Shows an overview table for classifications of the number of reads that fall in the miRBase categories for each species selected.

Annotations (Other) Shows an overview table with read numbers for total, exact match and mutant variants for each of the other annotation resources.

2.16.4 Working with the small RNA sample

Generally speaking, the small RNA sample comes in two variants:

- The *un-grouped* sample, either as it comes directly from the **Extract and Count** () or when it has been annotated. In this sample, there is one row per tag, and the feature ID is the tag sequence.
- The *grouped* sample created using the **Annotate and Merge Counts** () tool. In this sample, each row represents several tags grouped by a common Mature or Precursor miRNA or other reference.

Below, these two kinds of samples are described in further detail. Note that for both samples, filtering and sorting can be applied, see section **??**.

1 Summary

Name	Small RNAs	Annotated	Percentage	Reads	Annotated	Percentage
SRR038853 Small RNA sample	88,460	31,841	36.0%	1,720,241	1,511,704	87.9%

2 Resources

Resource	Sequences in resource	Sequences found	Percentage found
miRBase (Homo sapiens)	940	453	48.2%
miRBase (Mus musculus)	590	77	13.1%
Homo_sapiens.GRCh37.57.ncrna	12,887	3,586	27.8%

3 Reads

Annotation	Count	Percentage	Perfect matches	%	1 mismatch	%	2 mismatches	%
Annotated	1,511,704	87.9%	1,213,635	80.3%	247,319	16.4%	50,750	3.4%
- with miRBase	1,470,812	97.3%	1,190,140	80.9%	234,618	16.0%	46,054	3.1%
- Homo sapiens	1,436,510	97.7%	1,165,868	81.2%	226,769	15.8%	43,873	3.1%
- Mus musculus	34,302	2.3%	24,272	70.8%	7,849	22.9%	2,181	6.4%
- with Homo_sapiens.GRCh37.57.ncrna	40,892	2.7%	23,495	57.5%	12,701	31.1%	4,696	11.5%
Unannotated	208,537	12.1%						
Total	1,720,241	100.0%						

Figure 2.161: *A summary report of the annotation.*

The un-grouped sample

An example of an un-grouped annotated sample is shown in figure 2.162.

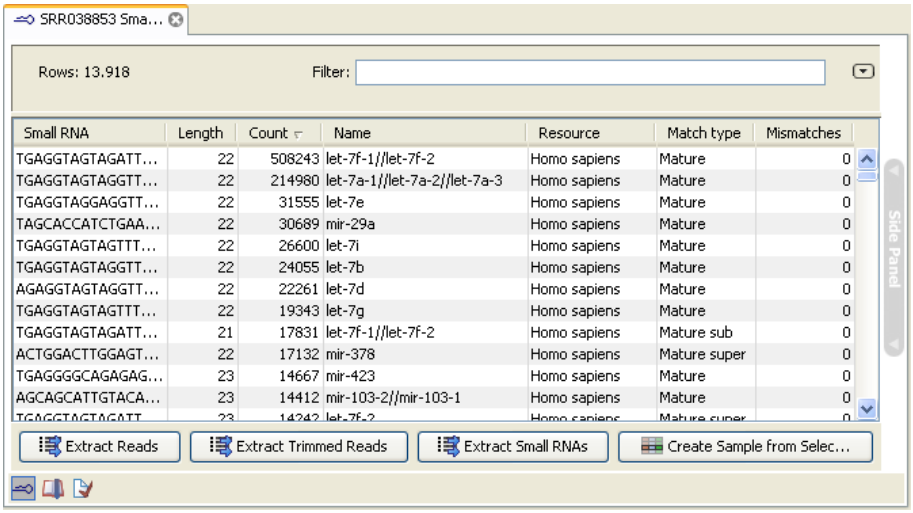


Figure 2.162: *An ungrouped annotated sample.*

By selecting one or more rows in the table, the buttons at the bottom of the view can be used to extract sequences from the table:

Extract Reads ( ) This will extract the original sequencing reads that contributed to this tag. Figure 2.163 shows an example of such a read. The reads include trim annotations (for use when inspecting and double-checking the results of trimming). Note that if these reads are used for read mapping, the trimmed part of the read will automatically be removed. If

all rows in the sample are selected and extracted, the sequence list would be the same as the input except for the reads that did not meet the adapter trim settings and the sampling thresholds (tag length and number of copies).

Extract Trimmed Reads ( ) The same as above, except that the trimmed part has been removed.

Extract Small RNAs ( ) This will extract only one copy of each tag.

Note that for all these, you will be able to determine whether a list of DNA or RNA sequences should be produced (when working within the *CLC Genomics Workbench* environment, this only effects the RNA folding tools).

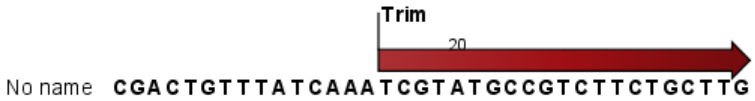


Figure 2.163: *Extracting reads from a sample.*

The button **Create Sample from Selection** ( ) can be used to create a new sample based on the tags that are selected. This can be useful in combination with filtering and sorting.

The grouped sample

An example of a grouped annotated sample is shown in figure 2.164.

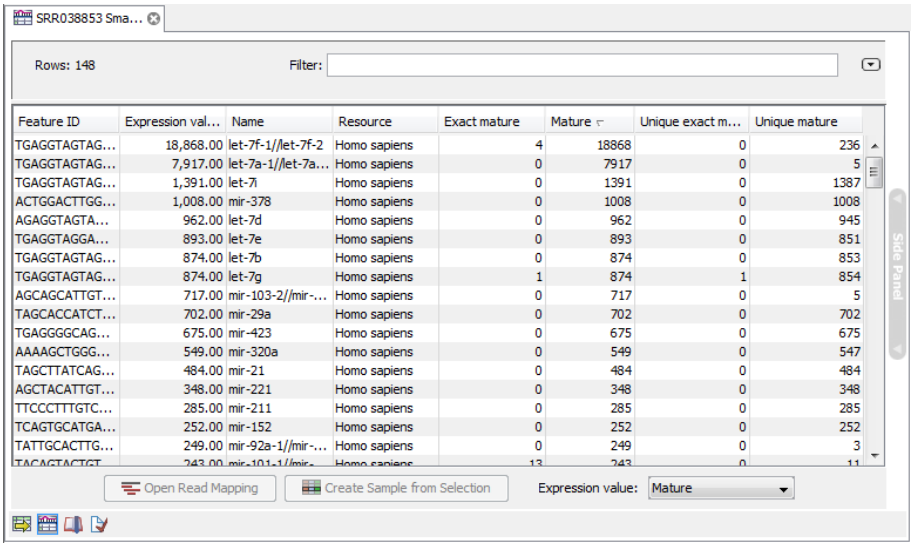


Figure 2.164: *A sample grouped on mature 5' miRNAs.*

The contents of the table are explained in section 2.16.3. In this section, we focus on the tools available for working with the sample.

By selecting one or more rows in the table, the buttons at the bottom of the view become active:

Open Read Mapping ( ) This will open a view showing the annotation reference sequence at the top and the tags aligned to it as shown in figure 2.165. The names of the tags indicate their status compared with the reference (e.g. Mature 5', Mature super 5', Precursor).

This categorization is based on the choices you make when annotating. You can also see the annotations when using miRBase as the annotation source. In this example both the mature 5' and the mature 3' are annotated, and you can see that both are found in the sample. In the **Side Panel** to the right you can see the **Match weight** group under **Residue coloring** which is used to color the tags according to their relative abundance. The weight is also shown next to the name of the tag. The left side color is used for tags with low counts and the right side color is used for tags with high counts, relative to the total counts of this annotation reference. The sliders just above the gradient color box can be dragged to highlight relevant levels of abundance. The colors can be changed by clicking the box. This will show a list of gradients to choose from.

Create Sample from Selection ( ) This is used to create a new sample based on the tags that are selected. This can be useful in combination with filtering and sorting.

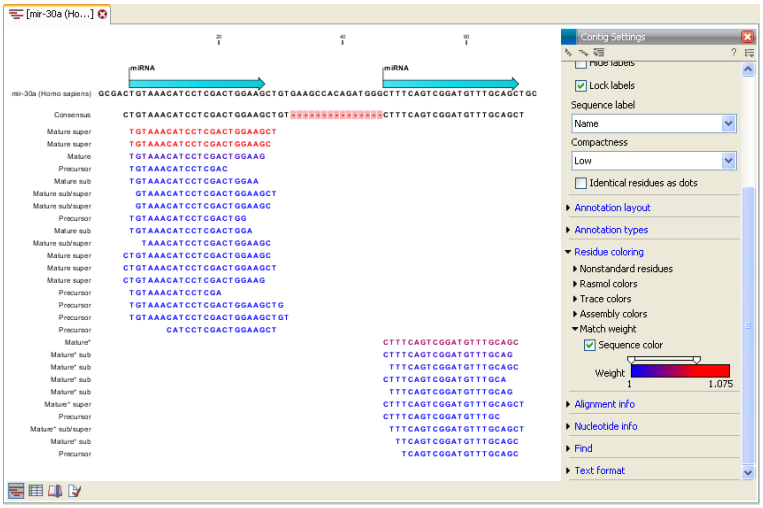


Figure 2.165: *Aligning all the variants of this miRNA from miRBase, providing a visual overview of the distribution of tags along the precursor sequence.*

2.16.5 Exploring novel miRNAs

One way of doing this would be to identify interesting tags based on their counts (typically you would be interested in pursuing tags with not too low counts in order to avoid wasting efforts on tags based on reads with sequencing errors), **Extract Small RNAs** ( ) and use this list of tags as input to **Map Reads to Reference** ( ) using the genome as reference. You could then examine where the reads match, and for reads that map in otherwise unannotated regions you could select a region around the match and create a subsequence from this. The subsequence could be folded and examined to see whether the secondary structure was in agreement with the expected hairpin-type structure for miRNAs.

Chapter 3

Expression analysis

Contents

3.1 Experimental design 157
3.1.1 Supported array platforms 157
3.1.2 Setting up an experiment 158
3.1.3 Organization of the experiment table 160
3.1.4 Adding annotations to an experiment 165
3.1.5 Scatter plot view of an experiment 166
3.1.6 Cross-view selections 169
3.2 Transformation and normalization 169
3.2.1 Selecting transformed and normalized values for analysis 170
3.2.2 Transformation 171
3.2.3 Normalization 171
3.3 Quality control 174
3.3.1 Creating box plots - analyzing distributions 174
3.3.2 Hierarchical clustering of samples 177
3.3.3 Principal component analysis 182
3.4 Statistical analysis - identifying differential expression 186
3.4.1 Gaussian-based tests 187
3.4.2 Tests on proportions 189
3.4.3 Corrected p-values 190
3.4.4 Volcano plots - inspecting the result of the statistical analysis 191
3.5 Feature clustering 194
3.5.1 Hierarchical clustering of features 194
3.5.2 K-means/medoids clustering 198
3.6 Annotation tests 200
3.6.1 Hypergeometric tests on annotations 200
3.6.2 Gene set enrichment analysis 203
3.7 General plots 207
3.7.1 Histogram 207
3.7.2 MA plot 209
3.7.3 Scatter plot 212

The *CLC Genomics Workbench* is able to analyze expression data produced on microarray platforms and high-throughput sequencing platforms (also known as Next-Generation Sequencing platforms).

Note that the calculation of expression levels based on the raw sequence data is described in section 2.14.

The *CLC Genomics Workbench* provides tools for performing quality control of the data, transformation and normalization, statistical analysis to measure differential expression and annotation-based tests. A number of visualization tools such as volcano plots, MA plots, scatter plots, box plots and heat maps are used to aid the interpretation of the results.

The various tools available are described in the sections listed below.

3.1 Experimental design

In order to make full use of the various tools for interpreting expression data, you need to know the central concepts behind the way the data is organized in the *CLC Genomics Workbench*.

The first piece of data you are faced with is the **sample**. In the Workbench, a sample contains the expression values from either one array or from sequencing data of one sample. Note that the calculation of expression levels based on the raw sequence data is described in sections 2.14 and 2.15 .

See more below on how to get your expression data into the Workbench as samples (under Supported array platforms).

In a sample, there is a number of **features**, usually genes, and their associated expression levels.

To analyze differential expression, you need to tell the workbench how the samples are related. This is done by setting up an **experiment**. An experiment is essentially a set of samples which are grouped. By creating an experiment defining the relationship between the samples, it becomes possible to do statistical analysis to investigate differential expression between the groups. The **Experiment** is also used to accumulate calculations like t-tests and clustering because this information is closely related to the grouping of the samples.

3.1.1 Supported array platforms

The workbench supports analysis of one-color expression arrays. These may be imported from GEO soft sample- or series- file formats, or for Affymetrix arrays, tab-delimited pivot or metrics files, or from Illumina expression files. Expression array data from other platforms may be imported from tab, semi-colon or comma separated files containing the expression feature IDs and levels in a tabular format (see see section **??**).

The workbench assumes that expression values are given at the gene level, thus probe-level analysis of e.g. Affymetrix GeneChips and import of Affymetrix CEL and CDF files is currently not supported. However, the workbench allows import of txt files exported from R containing processed Affymetrix CEL-file data (see see section **??**).

Affymetrix NetAffx annotation files for expression GeneChips in csv format and Illumina annotation

files can also be imported. Also, you may import your own annotation data in tabular format see see section **??**).

See section **??** in the Appendix for detailed information about supported file formats.

3.1.2 Setting up an experiment

To set up an experiment:

Toolbox | **Expression Analysis (**) | **Set Up Experiment (**)

Select the samples that you wish to use by double-clicking or selecting and pressing the **Add** ( ) button (see figure 3.1).

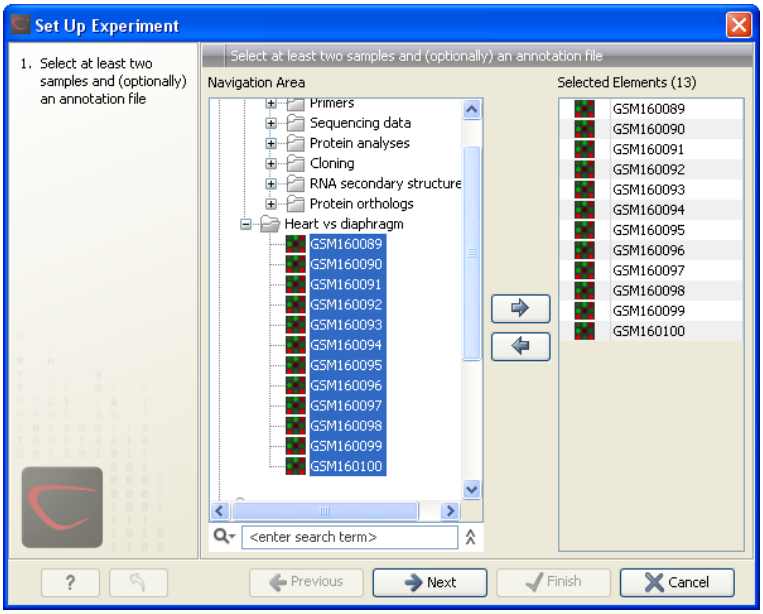


Figure 3.1: *Select the samples to use for setting up the experiment.*

Note that we use "samples" as the general term for both microarray-based sets of expression values and sequencing-based sets of expression values.

Clicking **Next** shows the dialog in figure 3.2.

Here you define the number of groups in the experiment. At the top you can select a two-group experiment, and below you can select a multi-group experiment and define the number of groups.

Note that you can also specify if the samples are paired. Pairing is relevant if you have samples from the same individual under different conditions, e.g. before and after treatment, or at times 0, 2 and 4 hours after treatment. In this case statistical analysis becomes more efficient if effects of the individuals are taken into account, and comparisons are carried out not simply by considering *raw* group means but by considering these *corrected for* effects of the individual. If the **Paired** is selected, a paired rather than a standard t-test will be carried out for two group comparisons. For multiple group comparisons a repeated measures rather than a standard ANOVA will be used.

For RNA-Seq experiments, you can also choose which expression value to be used when setting

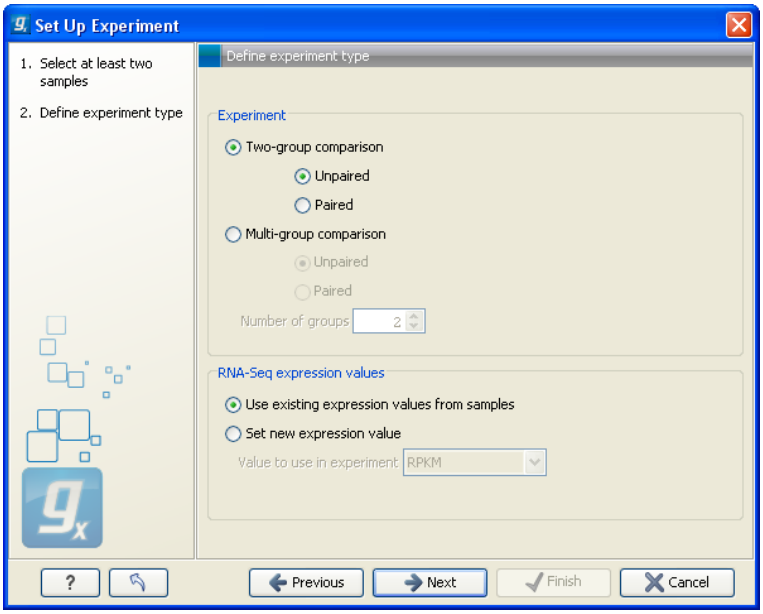


Figure 3.2: *Defining the number of groups.*

up the experiment. This value will then be used for all subsequence analyses.

Clicking **Next** shows the dialog in figure 3.3.

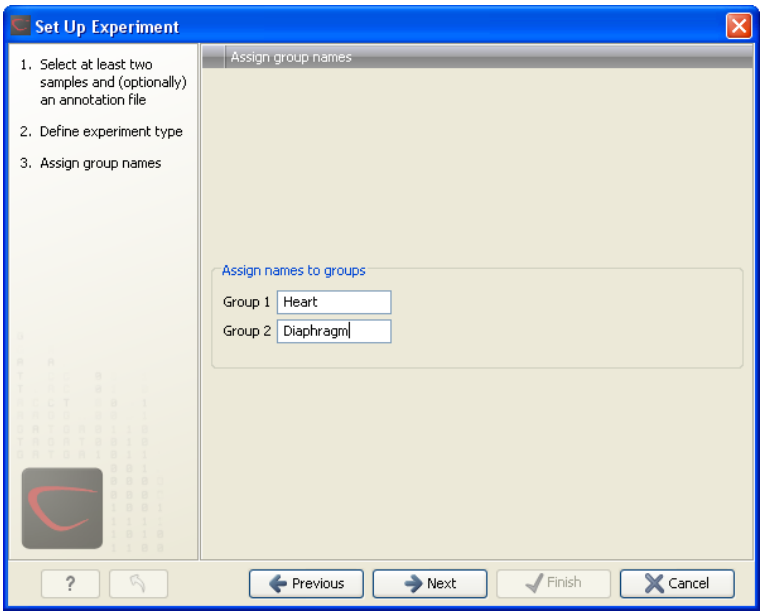


Figure 3.3: *Naming the groups.*

Depending on the number of groups selected in figure 3.2, you will see a list of groups with text fields where you can enter an appropriate name for that group.

For multi-group experiments, if you find out that you have too many groups, click the **Delete** ( ) button. If you need more groups, simply click **Add New Group**.

Click **Next** when you have named the groups, and you will see figure 3.4.

This is where you define which group the individual sample belongs to. Simply select one or

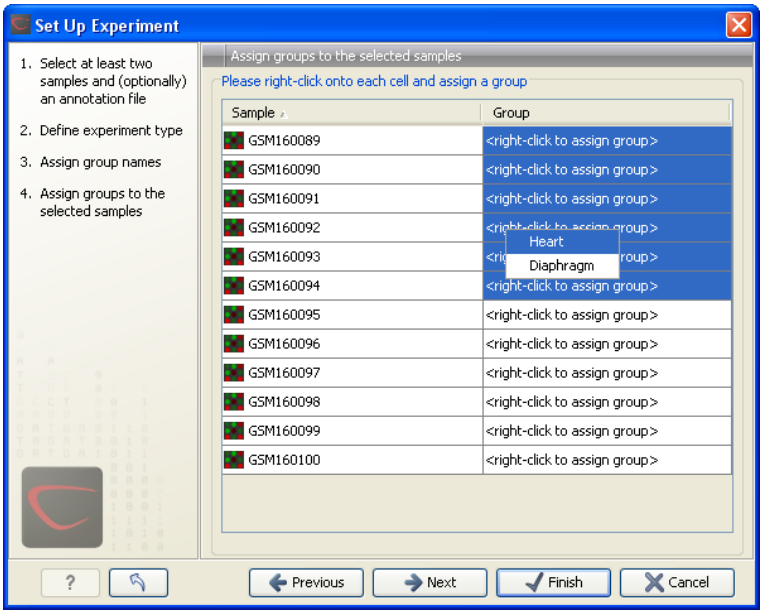


Figure 3.4: *Putting the samples into groups.*

more samples (by clicking and dragging the mouse), right-click (Ctrl-click on Mac) and select the appropriate group.

Note that the samples are sorted alphabetically based on their names.

If you have chosen **Paired** in figure 3.2, there will be an extra column where you define which samples belong together. Just as when defining the group membership, you select one or more samples, right-click in the pairing column and select a pair.

Click **Next** if you wish to adjust how to handle the results (see section **??**). If not, click **Finish**.

3.1.3 Organization of the experiment table

The resulting experiment includes all the expression values and other information from the samples (the values are copied - the original samples are not affected and can thus be deleted with no effect on the experiment). In addition it includes a number of summaries of the values across all, or a subset of, the samples for each feature. Which values are in included is described in the sections below.

When you open it, it is shown in the experiment table (see figure 3.5).

For a general introduction to table features like sorting and filtering, see section **??**.

Unlike other tables in *CLC Genomics Workbench*, the experiment table has a hierarchical grouping of the columns. This is done to reflect the structure of the data in the experiment. The **Side Panel** is divided into a number of groups corresponding to the structure of the table. These are described below. Note that you can customize and save the settings of the **Side Panel** (see section **??**).

Whenever you perform analyses like normalization, transformation, statistical analysis etc, new columns will be added to the experiment. You can at any time **Export** ( ) all the data in the experiment in csv or Excel format or **Copy** ( ) the full table or parts of it.

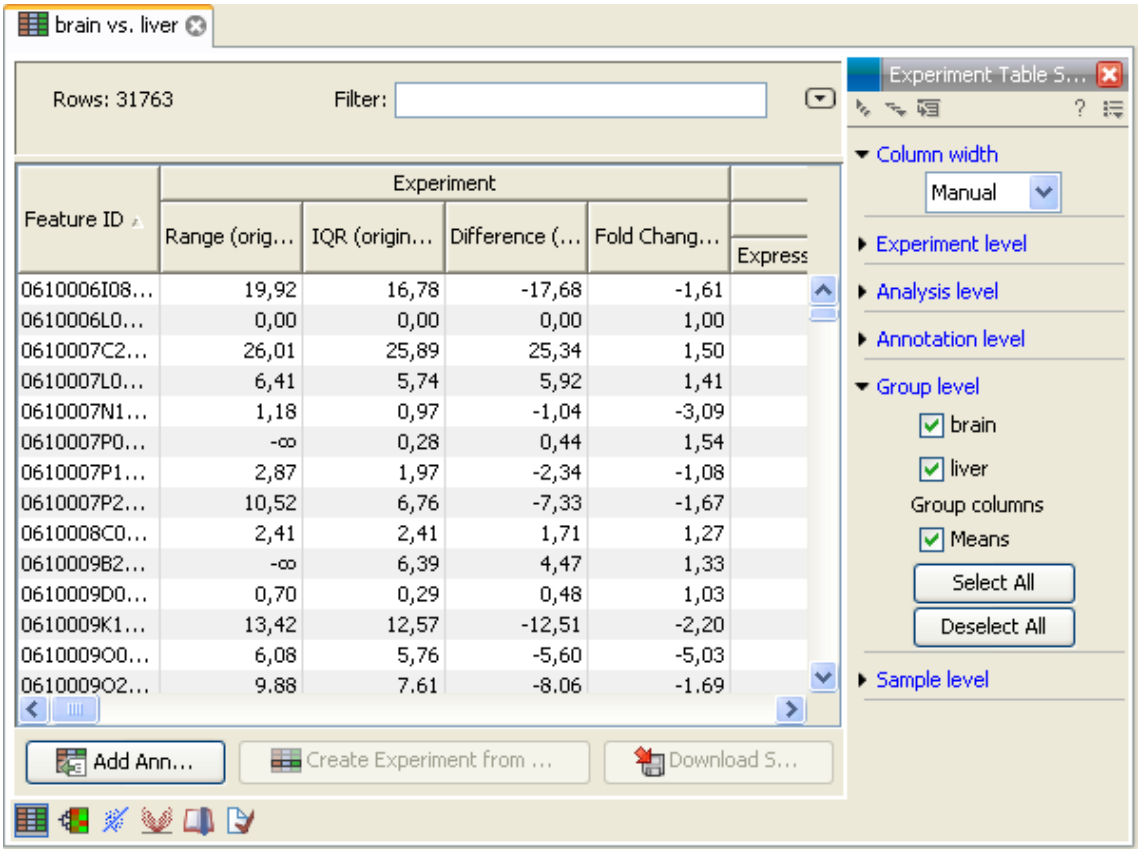


Figure 3.5: *Opening the experiment.*

Column width

There are two options to specify the width of the columns and also the entire table:

- **Automatic**. This will fit the entire table into the width of the view. This is useful if you only have a few columns.
- **Manual**. This will adjust the width of all columns evenly, and it will make the table as wide as it needs to be to display all the columns. This is useful if you have many columns. In this case there will be a scroll bar at the bottom, and you can manually adjust the width by dragging the column separators.

Experiment level

The rest of the **Side Panel** is devoted to different levels of information on the values in the experiment. The experiment part contains a number of columns that, for each feature ID, provide summaries of the values across all the samples in the experiment (see figure 3.6).

Initially, it has one header for the whole **Experiment**:

- **Range (original values)**. The 'Range' column contains the difference between the highest and the lowest expression value for the feature over all the samples. If a feature has the value NaN in one or more of the samples the range value is NaN.

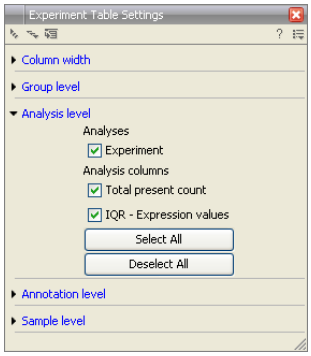


Figure 3.6: *The initial view of the experiment level for a two-group experiment.*

- **IQR (original values)**. The 'IQR' column contains the inter-quantile range of the values for a feature across the samples, that is, the difference between the 75 %-ile value and the 25 %-ile value. For the IQR values, only the numeric values are considered when percentiles are calculated (that is, NaN and +Inf or -Inf values are ignored), and if there are fewer than four samples with numeric values for a feature, the IQR is set to be the difference between the highest and lowest of these.

- **Difference (original values)**. For a two-group experiment the 'Difference' column contains the difference between the mean of the expression values across the samples assigned to group 2 and the mean of the expression values across the samples assigned to group 1. Thus, if the mean expression level in group 2 is higher than that of group 1 the 'Difference' is positive, and if it is lower the 'Difference' is negative. For experiments with more than two groups the 'Difference' contains the difference between the maximum and minimum of the mean expression values of the groups, multiplied by -1 if the group with the maximum mean expression value occurs before the group with the minimum mean expression value (with the ordering: group 1, group 2, ...).

- **Fold Change (original values)**. For a two-group experiment the 'Fold Change' tells you how many times bigger the mean expression value in group 2 is relative to that of group 1. If the mean expression value in group 2 is bigger than that in group 1 this value is the mean expression value in group 2 divided by that in group 1. If the mean expression value in group 2 is smaller than that in group 1 the fold change is the mean expression value in group 1 divided by that in group 2 with a negative sign. Thus, if the mean expression levels in group 1 and group 2 are 10 and 50 respectively, the fold change is 5, and if the and if the mean expression levels in group 1 and group 2 are 50 and 10 respectively, the fold change is -5. For experiments with more than two groups, the 'Fold Change' column contains the ratio of the maximum of the mean expression values of the groups to the minimum of the mean expression values of the groups, multiplied by -1 if the group with the maximum mean expression value occurs before the group with the minimum mean expression value (with the ordering: group 1, group 2, ...).

Thus, the sign of the values in the 'Difference' and 'Fold change' columns give the direction of the trend across the groups, going from group 1 to group 2, etc.

If the samples used are Affymetrix GeneChips samples and have 'Present calls' there will also be a 'Total present count' column containing the number of present calls for all samples.

The columns under the 'Experiment' header are useful for filtering purposes, e.g. you may wish to ignore features that differ too little in expression levels to be confirmed e.g. by qPCR by

filtering on the values in the 'Difference', 'IQR' or 'Fold Change' columns or you may wish to ignore features that do not differ at all by filtering on the 'Range' column.

If you have performed normalization or transformation (see sections 3.2.3 and 3.2.2, respectively), the IQR of the normalized and transformed values will also appear. Also, if you later choose to transform or normalize your experiment, columns will be added for the transformed or normalized values.

Note! It is very common to filter features on fold change values in expression analysis and fold change values are also used in volcano plots, see section 3.4.4. There are different definitions of 'Fold Change' in the literature. The definition that is used typically depends on the original scale of the data that is analyzed. For data whose original scale is *not* the log scale the standard definition is the ratio of the group means [Tusher et al., 2001]. This is the value you find in the 'Fold Change' column of the experiment. However, for data whose original *is* the log scale, the difference of the mean expression levels is sometimes referred to as the fold change [Guo et al., 2006], and if you want to filter on fold change for these data you should filter on the values in the 'Difference' column. Your data's original scale will e.g. be the log scale if you have imported Affymetrix expression values which have been created by running the RMA algorithm on the probe-intensities.

Analysis level

If you perform statistical analysis (see section 3.4), there will be a heading for each statistical analysis performed. Under each of these headings you find columns holding relevant values for the analysis (P-value, corrected P-value, test-statistic etc. - see more in section 3.4).

An example of a more elaborate analysis level is shown in figure 3.7.

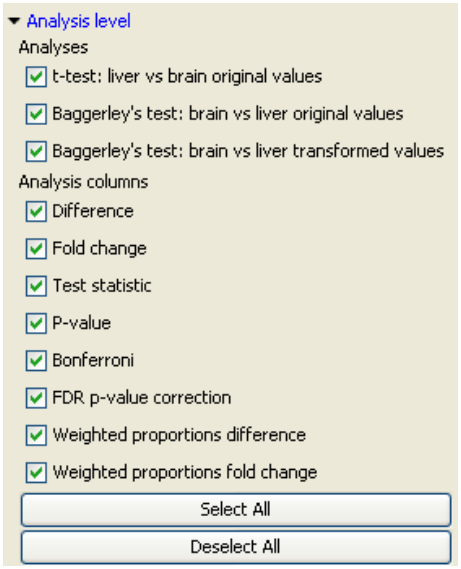


Figure 3.7: *Transformation, normalization and statistical analysis has been performed.*

Annotation level

If your experiment is annotated (see section 3.1.4), the annotations will be listed in the **Annotation level** group as shown in figure 3.8.

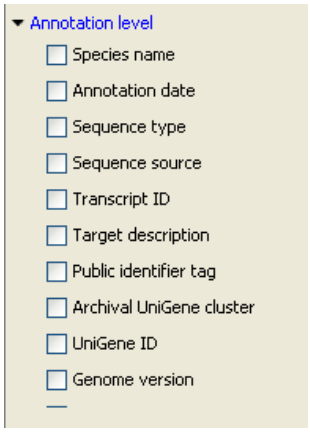


Figure 3.8: *An annotated experiment.*

In order to avoid too much detail and cluttering the table, only a few of the columns are shown per default.

Note that if you wish a different set of annotations to be displayed each time you open an experiment, you need to save the settings of the **Side Panel** (see section **??**).

Group level

At the group level, you can show/hide entire groups (*Heart* and *Diaphragm* in figure 3.5). This will show/hide everything under the group's header. Furthermore, you can show/hide group-level information like the group means and present count within a group. If you have performed normalization or transformation (see sections 3.2.3 and 3.2.2, respectively), the means of the normalized and transformed values will also appear.

Sample level

In this part of the side panel, you can control which columns to be displayed for each sample. Initially this is the all the columns in the samples.

If you have performed normalization or transformation (see sections 3.2.3 and 3.2.2, respectively), the normalized and transformed values will also appear.

An example is shown in figure 3.9.

Creating a sub-experiment from a selection

If you have identified a list of genes that you believe are differentially expressed, you can create a subset of the experiment. (Note that the filtering and sorting may come in handy in this situation, see section **??**).

To create a sub-experiment, first select the relevant features (rows). If you have applied a filter and wish to select all the visible features, press Ctrl + A (⌘ + A on Mac). Next, press the **Create Experiment from Selection** ( ) button at the bottom of the table (see figure 3.10).

This will create a new experiment that has the same information as the existing one but with less features.

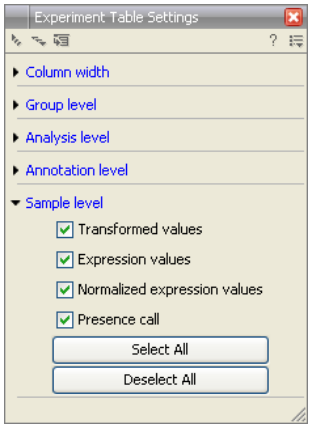


Figure 3.9: *Sample level when transformation and normalization has been performed.*

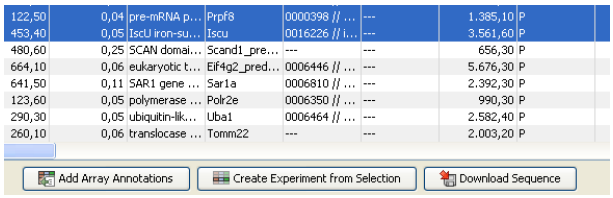


Figure 3.10: *Create a subset of the experiment by clicking the button at the bottom of the experiment table.*

Downloading sequences from the experiment table

If your experiment is annotated, you will be able to download the GenBank sequence for features which have a GenBank accession number in the 'Public identifier tag' annotation column. To do this, select a number of features (rows) in the experiment and then click **Download Sequence** ( ) (see figure 3.11).

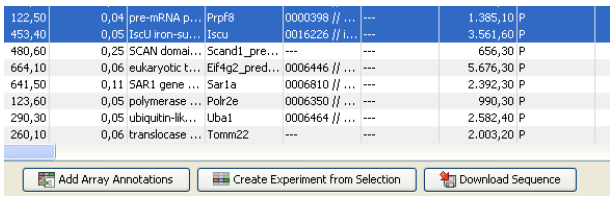


Figure 3.11: *Select sequences and press the download button.*

This will open a dialog where you specify where the sequences should be saved. You can learn more about opening and viewing sequences in chapter **??**. You can now use the downloaded sequences for further analysis in the Workbench, e.g. performing BLAST searches and designing primers for QPCR experiments.

3.1.4 Adding annotations to an experiment

Annotation files provide additional information about each feature. This information could be which GO categories the protein belongs to, which pathways, various transcript and protein identifiers etc. See section **??** for information about the different annotation file formats that are supported *CLC Genomics Workbench*.

The annotation file can be imported into the Workbench and will get a special icon ( ). See an overview of annotation formats supported by *CLC Genomics Workbench*in section **??**. In order to

associate an annotation file with an experiment, either select the annotation file when you set up the experiment (see section 3.1.2), or click:

Toolbox | **Expression Analysis (**) | **Annotation Test** | **Add Annotations (**)

Select the experiment ( ) and the annotation file ( ) and click **Finish**. You will now be able to see the annotations in the experiment as described in section 3.1.3. You can also add annotations by pressing the **Add Annotations** ( ) button at the bottom of the table (see figure 3.12).

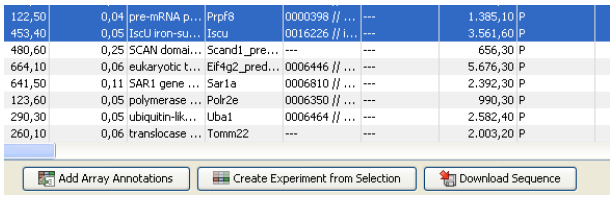


Figure 3.12: *Adding annotations by clicking the button at the bottom of the experiment table.*

This will bring up a dialog where you can select the annotation file that you have imported together with the experiment you wish to annotate. Click **Next** to specify settings as shown in figure 3.13).

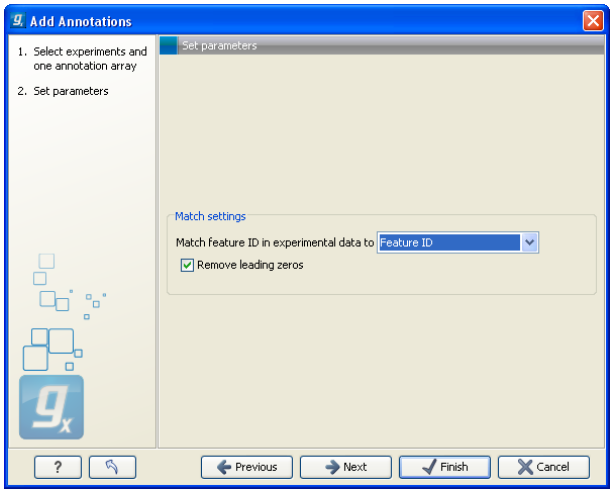


Figure 3.13: *Choosing how to match annotations with samples.*

In this dialog, you can specify how to match the annotations to the features in the sample. The Workbench looks at the columns in the annotation file and lets you choose which column that should be used for matching to the feature IDs in the experimental data (samples or experiment). Usually the default is right, but for some annotation files, you need to use another column.

Some annotation files have leading zeros in the identifier which you can remove by checking the **Remove leading zeros** box.

Note! Existing annotations on the experiment will be overwritten.

3.1.5 Scatter plot view of an experiment

At the bottom of the experiment table, you can switch between different views of the experiment (see figure 3.14).



Figure 3.14: *An experiment can be viewed in several ways.*

One of the views is the **Scatter Plot** ( ). The scatter plot can be adjusted to show e.g. the group means for two groups (see more about how to adjust this below).

An example of a scatter plot is shown in figure 3.15.

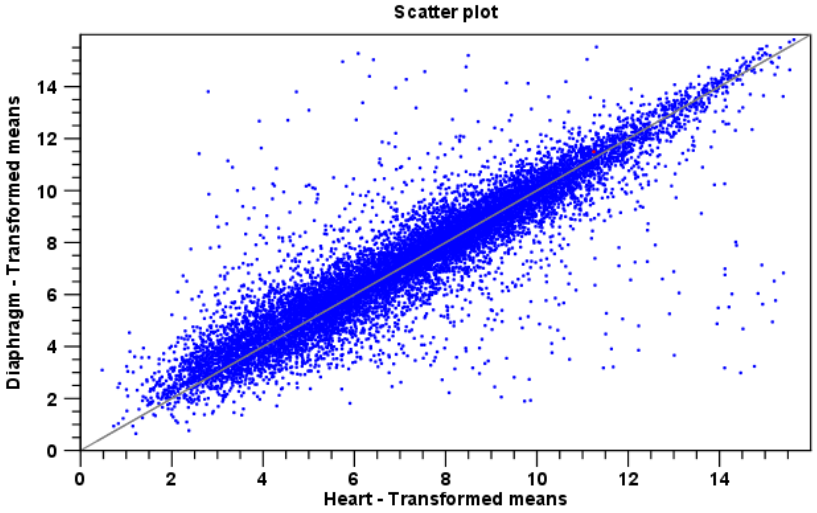


Figure 3.15: *A scatter plot of group means for two groups (transformed expression values).*

In the **Side Panel** to the left, there are a number of options to adjust this view. Under **Graph preferences**, you can adjust the general properties of the scatter plot:

- **Lock axes.** This will always show the axes even though the plot is zoomed to a detailed level.
- **Frame.** Shows a frame around the graph.
- **Show legends.** Shows the data legends.
- **Tick type**. Determine whether tick lines should be shown outside or inside the frame.
  - Outside
  - Inside
- **Tick lines at.** Choosing Major ticks will show a grid behind the graph.
  - None
  - Major ticks
- **Horizontal axis range**. Sets the range of the horizontal axis (x axis). Enter a value in **Min** and **Max**, and press Enter. This will update the view. If you wait a few seconds without pressing Enter, the view will also be updated.

- **Vertical axis range**. Sets the range of the vertical axis (y axis). Enter a value in **Min** and **Max**, and press Enter. This will update the view. If you wait a few seconds without pressing Enter, the view will also be updated.
- **Draw x = y axis**. This will draw a diagonal line across the plot. This line is shown per default.
- **Line width**
  - Thin
  - Medium
  - Wide
- **Line type**
  - None
  - Line
  - Long dash
  - Short dash
- **Line color.** Allows you to choose between many different colors. Click the color box to select a color.

Below the general preferences, you find the **Dot properties** preferences, where you can adjust coloring and appearance of the dots:

- **Dot type**
  - None
  - Cross
  - Plus
  - Square
  - Diamond
  - Circle
  - Triangle
  - Reverse triangle
  - Dot
- **Dot color.** Allows you to choose between many different colors. Click the color box to select a color.

Finally, the group at the bottom - **Columns to compare** - is where you choose the values to be plotted. Per default for a two-group experiment, the group means are used.

Note that if you wish to use the same settings next time you open a scatter plot, you need to save the settings of the **Side Panel** (see section **??**).



Figure 3.16: *An experiment can be viewed in several ways.*

3.1.6 Cross-view selections

There are a number of different ways of looking at an experiment as shown in figure 3.16).

Beside the **Experiment table** ( ) which is the default view, the views are: **Scatter plot** ( ), **Volcano plot** ( ) and the **Heat map** ( ). By pressing and holding the Ctrl (⌘ on Mac) button while you click one of the view buttons in figure 3.16, you can make a split view. This will make it possible to see e.g. the experiment table in one view and the volcano plot in another view.

An example of such a split view is shown in figure 3.17.

Selections are shared between all these different views of an experiment. This means that if you select a number of rows in the table, the corresponding dots in the scatter plot, volcano plot or heatmap will also be selected. The selection can be made in any view, also the heat map, and all other open views will reflect the selection.

A common use of the split views is where you have an experiment and have performed a statistical analysis. You filter the experiment to identify all genes that have an FDR corrected p-value below 0.05 and a fold change for the test above say, 2. You can select all the rows in the experiment table satisfying these filters by holding down the Cntrl button and clicking 'a'. If you have a split view of the experiment and the volcano plot all points in the volcano plot corresponding to the selected features will be red. Note that the volcano plot allows two sets of values in the columns under the test you are considering to be displayed on the x-axis: the 'Fold change's and the 'Difference's. You control which to plot in the side panel. If you have filtered on 'Fold change' you will typically want to choose 'Fold change' in the side panel. If you have filtered on 'Difference' (e.g. because your original data is on the log scale, see the note on fold change in 3.1.3) you typically want to choose 'Difference'.

3.2 Transformation and normalization

The original expression values often need to be transformed and/or normalized in order to ensure that samples are comparable and assumptions on the data for analysis are met [Allison et al., 2006]. These are essential requirements for carrying out a meaningful analysis. The raw expression values often exhibit a strong dependency of the variance on the mean, and it may be preferable to remove this by log-transforming the data. Furthermore, the sets of expression values in the different samples in an experiment may exhibit systematic differences that are likely due to differences in sample preparation and array processing, rather being the result of the underlying biology. These noise effects should be removed before statistical analysis is carried out.

When you perform transformation and normalization, the original expression values will be kept, and the new values will be added. If you select an experiment ( ), the new values will be added to the experiment (not the original samples). And likewise if you select a sample (( ) or ( )) - in this case the new values will be added to the sample (the original values are still kept on the sample).

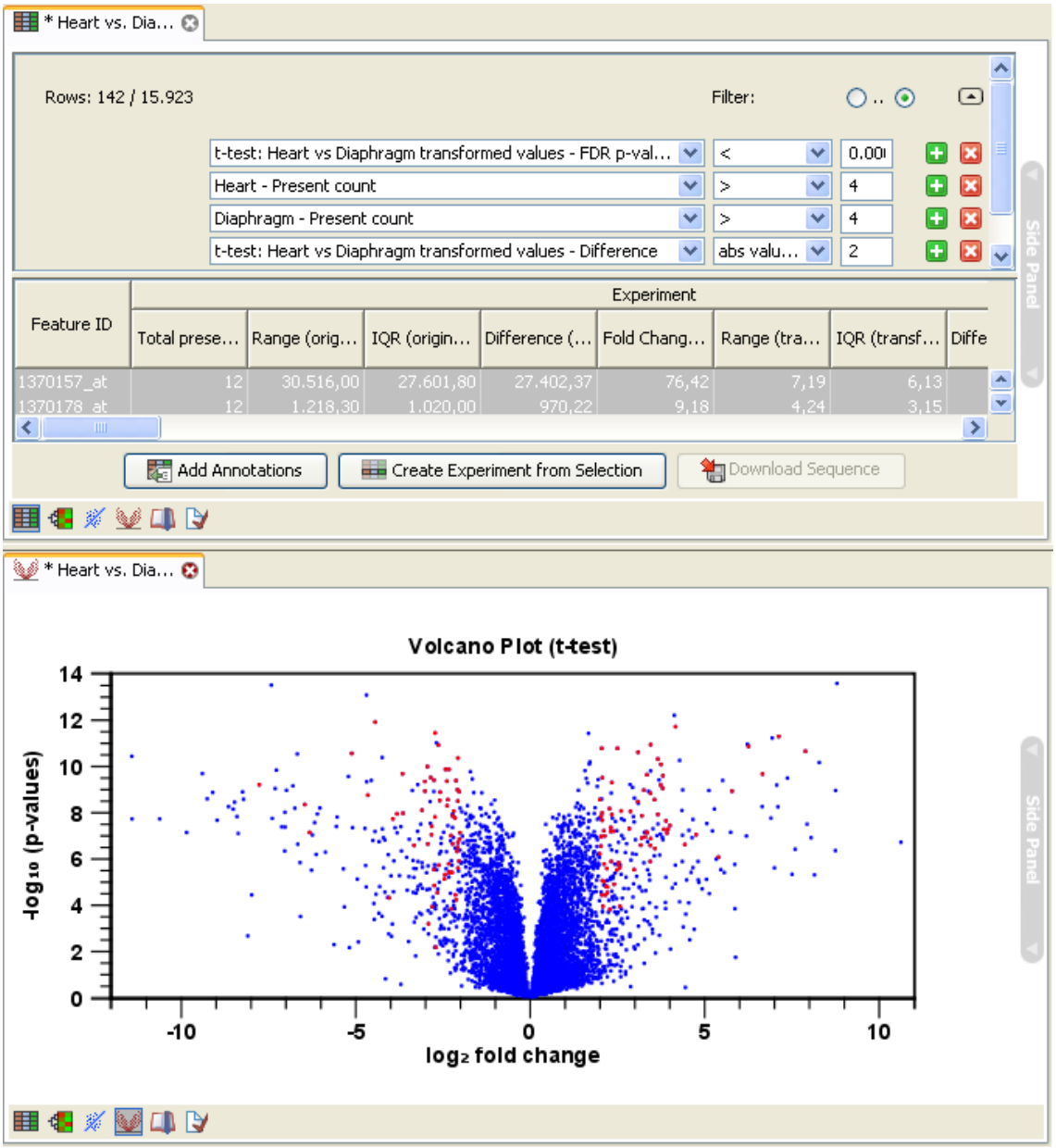


Figure 3.17: *A split view showing an experiment table at the top and a volcano plot at the bottom (note that you need to perform statistical analysis to show a volcano plot, see section 3.4).*

3.2.1 Selecting transformed and normalized values for analysis

A number of the tools in the **Expression Analysis** ( ) folder use expression levels. All of these tools let you choose between *Original*, *Transformed* and *Normalized* expression values as shown in figure 3.18.

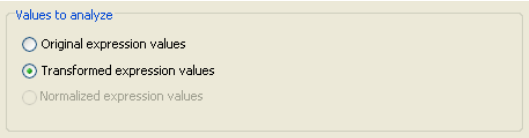


Figure 3.18: *Selecting which version of the expression values to analyze. In this case, the values have not been normalized, so it is not possible to select normalized values.*

In this case, the values have not been normalized, so it is not possible to select normalized values.

3.2.2 Transformation

The *CLC Genomics Workbench* lets you transform expression values based on logarithm and adding a constant:

Toolbox | **Expression Analysis (**) | **Transformation and Normalization** | **Transform (** **)**

Select a number of samples (( ) or ( )) or an experiment ( ) and click **Next**.

This will display a dialog as shown in figure 3.19.

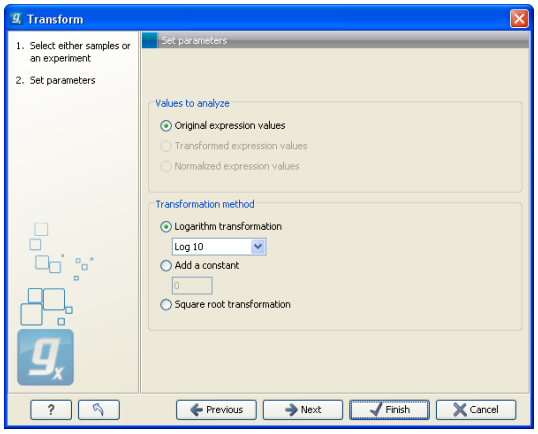


Figure 3.19: *Transforming expression values.*

At the top, you can select which values to transform (see section 3.2.1).

Next, you can choose three kinds of transformation:

- **Logarithm transformation**. Transformed expression values will be calculated by taking the logarithm (of the specified type) of the values you have chosen to transform.
  - **10**.
  - **2**.
  - **Natural logarithm**.
- **Adding a constant**. Transformed expression values will be calculated by adding the specified constant to the values you have chosen to transform.
- **Square root transformation**.

Click **Next** if you wish to adjust how to handle the results (see section **??**). If not, click **Finish**.

3.2.3 Normalization

The *CLC Genomics Workbench* lets you normalize expression values.

To start the normalization:

Toolbox | **Expression Analysis** ( ) | **Transformation and Normalization** | **Normalize** ( )

Select a number of samples (( ) or ( )) or an experiment ( ) and click **Next**.

This will display a dialog as shown in figure 3.20.

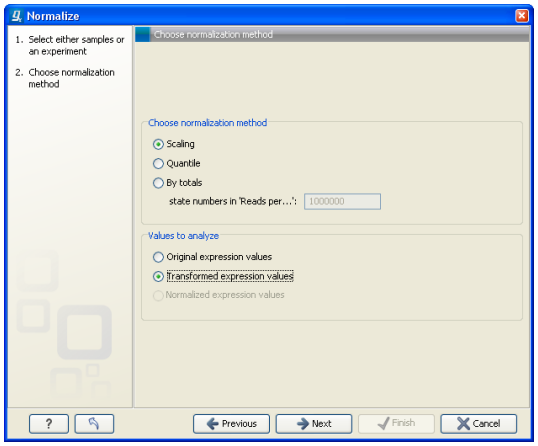


Figure 3.20: *Choosing normalization method.*

At the top, you can choose three kinds of normalization (for mathematical descriptions see [Bolstad et al., 2003]):

- **Scaling**. The sets of the expression values for the samples will be multiplied by a constant so that the sets of normalized values for the samples have the same 'target' value (see description of the **Normalization value** below).
- **Quantile**. The empirical distributions of the sets of expression values for the samples are used to calculate a common target distribution, which is used to calculate normalized sets of expression values for the samples.
- **By totals**. This option is intended to be used with count-based data, i.e. data from RNA-seq, small RNA or expression profiling by tags. A sum is calculated for the expression values in a sample. The transformed value are generated by dividing the input values by the sample sum and multiplying by the factor (e.g. per '1,000,000').

Figures 3.21 and 3.22 show the effect on the distribution of expression values when using scaling or quantile normalization, respectively.

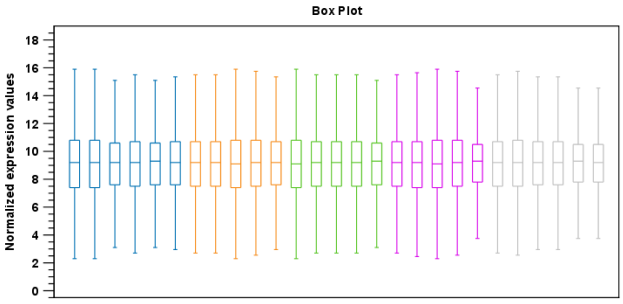


Figure 3.21: *Box plot after scaling normalization.*

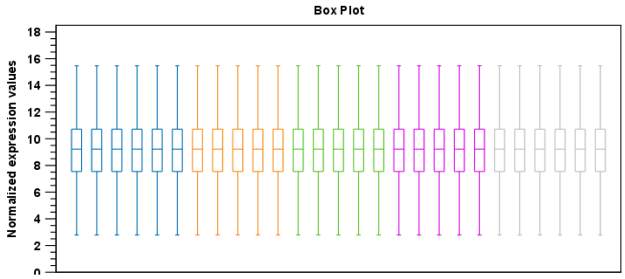


Figure 3.22: *Box plot after quantile normalization.*

At the bottom of the dialog in figure 3.20, you can select which values to normalize (see section 3.2.1).

Clicking **Next** will display a dialog as shown in figure 3.23.

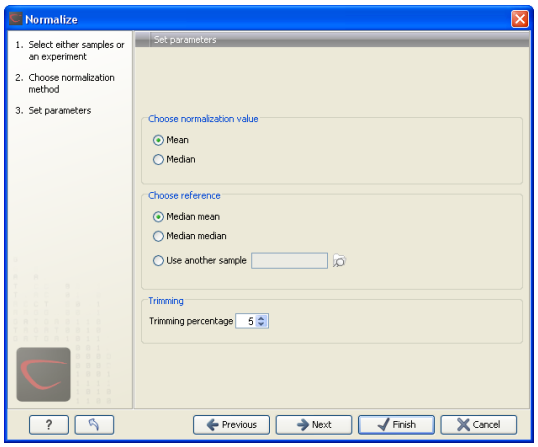


Figure 3.23: *Normalization settings.*

The following parameters can be set:

- **Normalization value**. The type of value of the samples which you want to ensure are equal for the normalized expression values
  - **Mean**.
  - **Median**.
- **Reference**. The specific value that you want the normalized value to be after normalization.
  - **Median mean**.
  - **Median median**.
  - **Use another sample**.
- **Trimming percentage**. Expression values that lie below the value of this percentile, or above 100 minus the value of this percentile, in the empirical distribution of the expression values in a sample will be excluded when calculating the normalization and reference values.

Click **Next** if you wish to adjust how to handle the results (see section **??**). If not, click **Finish**.

3.3 Quality control

The *CLC Genomics Workbench* includes a number of tools for quality control. These allow visual inspection of the overall distributions, variability and similarity of the sets of expression values in samples, and may be used to spot unwanted systematic differences between samples, outlying samples and samples of poor quality, that you may want to exclude.

3.3.1 Creating box plots - analyzing distributions

In most cases you expect the majority of genes to behave similarly under the conditions considered, and only a smaller proportion to behave differently. Thus, at an overall level you would expect the distributions of the sets of expression values in samples in a study to be similar. A boxplot provides a visual presentation of the distributions of expression values in samples. For each sample the distribution of it's values is presented by a line representing a center, a box representing the middle part, and whiskers representing the tails of the distribution. Differences in the overall distributions of the samples in a study may indicate that normalization is required before the samples are comparable. An atypical distribution for a single sample (or a few samples), relative to the remaining samples in a study, could be due to imperfections in the preparation and processing of the sample, and may lead you to reconsider using the sample(s).

To create a box plot:

Toolbox | **Expression Analysis ( )** | **Quality Control** | **Create Box Plot ( )**

Select a number of samples (( ) or ( )) or an experiment ( ) and click **Next**.

This will display a dialog as shown in figure 3.24.

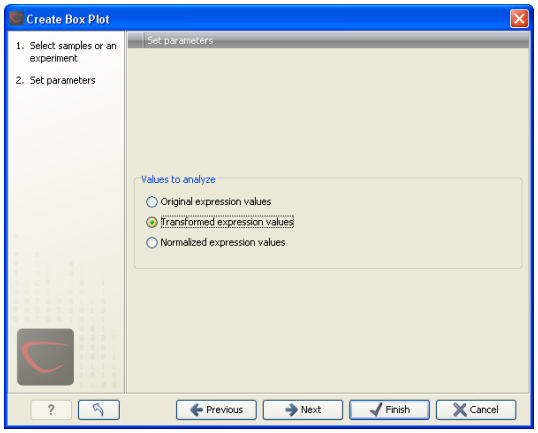


Figure 3.24: *Choosing values to analyze for the box plot.*

Here you select which values to use in the box plot (see section 3.2.1).

Click **Next** if you wish to adjust how to handle the results (see section **??**). If not, click **Finish**.

Viewing box plots

An example of a box plot of a two-group experiment with 12 samples is shown in figure 3.25.

Note that the boxes per default are colored according to their group relationship. At the bottom you find the names of the samples, and the y-axis shows the expression values (note that sample

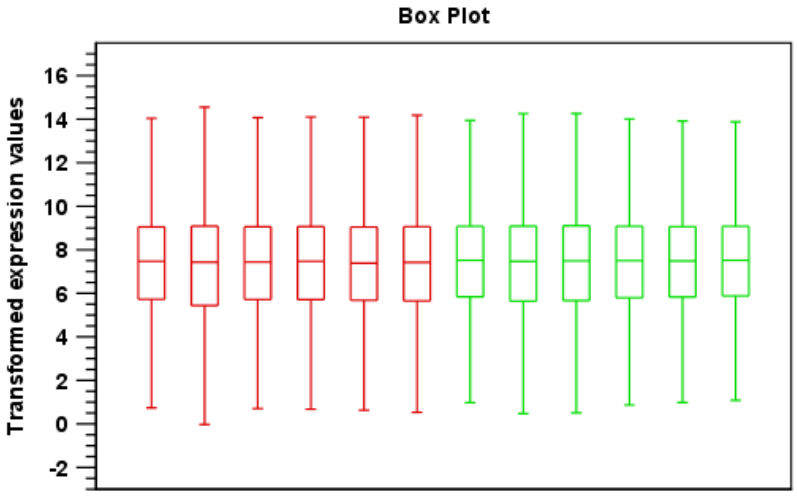


Figure 3.25: *A box plot of 12 samples in a two-group experiment, colored by group.*

names are not shown in figure 3.25).

Per default the box includes the IQR values (from the lower to the upper quartile), the median is displayed as a line in the box, and the whiskers extend 1.5 times the height of the box.

In the **Side Panel** to the left, there is a number of options to adjust this view. Under **Graph preferences**, you can adjust the general properties of the box plot (see figure 3.26).

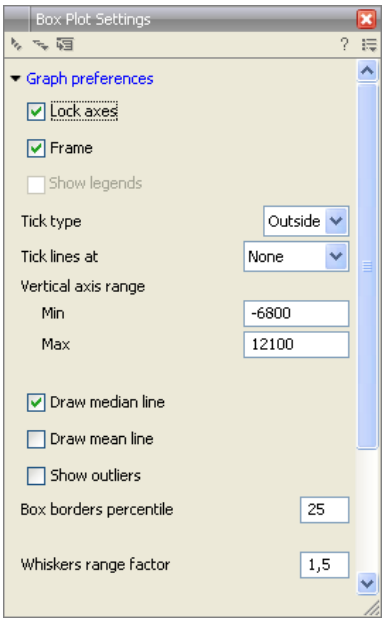


Figure 3.26: *Graph preferences for a box plot.*

- **Lock axes.** This will always show the axes even though the plot is zoomed to a detailed level.
- **Frame.** Shows a frame around the graph.
- **Show legends.** Shows the data legends.

- **Tick type**. Determine whether tick lines should be shown outside or inside the frame.
  - Outside
  - Inside
- **Tick lines at.** Choosing Major ticks will show a grid behind the graph.
  - None
  - Major ticks
- **Vertical axis range**. Sets the range of the vertical axis (y axis). Enter a value in **Min** and **Max**, and press Enter. This will update the view. If you wait a few seconds without pressing Enter, the view will also be updated.
- **Draw median line**. This is the default - the median is drawn as a line in the box.
- **Draw mean line**. Alternatively, you can also display the mean value as a line.
- **Show outliers**. The values outside the whiskers range are called outliers. Per default they are not shown. Note that the dot type that can be set below only takes effect when outliers are shown. When you select and deselect the **Show outliers**, the vertical axis range is automatically re-calculated to accommodate the new values.

Below the general preferences, you find the **Lines and dots** preferences, where you can adjust coloring and appearance (see figure 3.27).

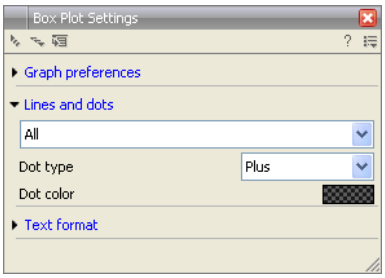


Figure 3.27: *Lines and dot preferences for a box plot.*

- **Select sample or group**. When you wish to adjust the properties below, first select an item in this drop-down menu. That will apply the changes below to this item. If your plot is based on an experiment, the drop-down menu includes both group names and sample names, as well as an entry for selecting "All". If your plot is based on single elements, only sample names will be visible. Note that there are sometimes "mixed states" when you select a group where two of the samples e.g. have different colors. Selecting a new color in this case will erase the differences.
- **Dot type**
  - None
  - Cross
  - Plus
  - Square

  - Diamond
  - Circle
  - Triangle
  - Reverse triangle
  - Dot

- **Dot color.** Allows you to choose between many different colors. Click the color box to select a color.

Note that if you wish to use the same settings next time you open a box plot, you need to save the settings of the **Side Panel** (see section **??**).

Interpreting the box plot

This section will show how to interpret a box plot through a few examples.

First, if you look at figure 3.28, you can see a box plot for an experiment with 5 groups and 27 samples.

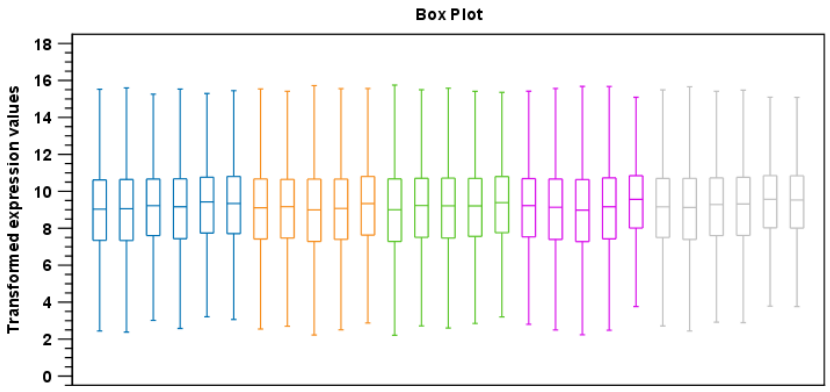


Figure 3.28: *Box plot for an experiment with 5 groups and 27 samples.*

None of the samples stand out as having distributions that are atypical: the boxes and whiskers ranges are about equally sized. The locations of the distributions however, differ some, and indicate that normalization may be required. Figure 3.29 shows a box plot for the same experiment after quantile normalization: the distributions have been brought into par.

In figure 3.30 a box plot for a two group experiment with 5 samples in each group is shown.

The distribution of values in the second sample from the left is quite different from those of other samples, and could indicate that the sample should not be used.

3.3.2 Hierarchical clustering of samples

A hierarchical clustering of samples is a tree representation of their relative similarity. The tree structure is generated by

1. letting each feature be a cluster

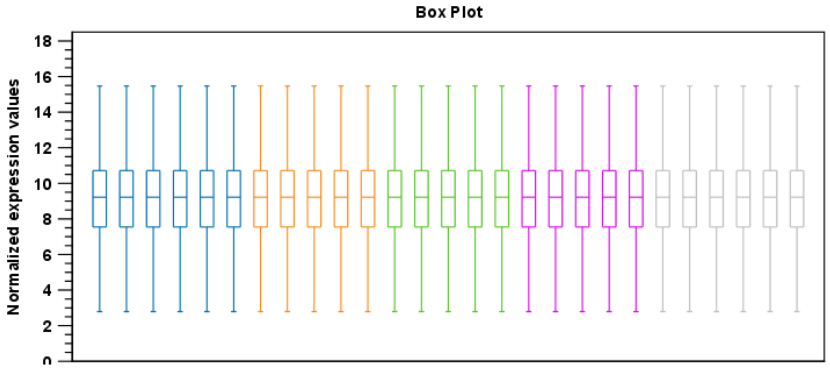


Figure 3.29: *Box plot after quantile normalization.*

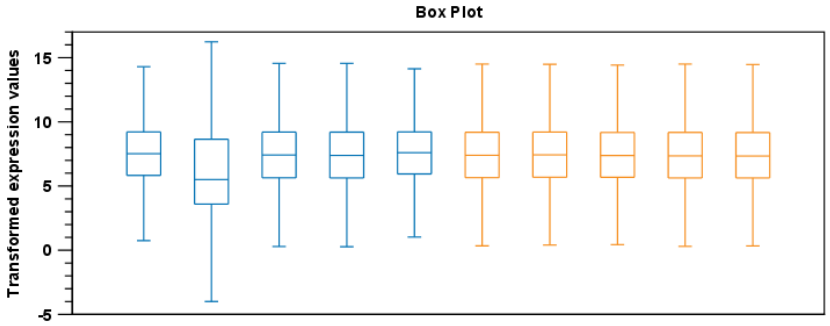


Figure 3.30: *Box plot for a two-group experiment with 5 samples.*

2. calculating pairwise distances between all clusters
3. joining the two closest clusters into one new cluster
4. iterating 2-3 until there is only one cluster left (which will contain all samples).

The tree is drawn so that the distances between clusters are reflected by the lengths of the branches in the tree. Thus, features with expression profiles that closely resemble each other have short distances between them, those that are more different, are placed further apart. (See [Eisen et al., 1998] for a classical example of application of a hierarchical clustering algorithm in microarray analysis. The example is on features rather than samples).

To start the clustering:

> **Toolbox** | **Expression Analysis** ( ) | **Quality Control** | **Hierarchical Clustering of Samples** ( )

Select a number of samples (( ) or ( )) or an experiment ( ) and click **Next**.

This will display a dialog as shown in figure 3.31. The hierarchical clustering algorithm requires that you specify a distance measure and a cluster linkage. The similarity measure is used to specify how distances between two samples should be calculated. The cluster distance metric specifies how you want the distance between two clusters, each consisting of a number of samples, to be calculated.

At the top, you can choose three kinds of **Distance measures**:

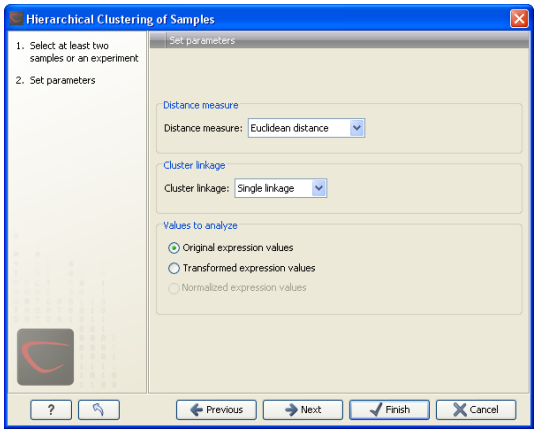


Figure 3.31: *Parameters for hierarchical clustering of samples.*

- **Euclidean distance**. The ordinary distance between two points - the length of the segment connecting them. If $u = (u_1, u_2, \ldots, u_n)$ and $v = (v_1, v_2, \ldots, v_n)$, then the Euclidean distance between u and v is

$$|u - v| = \sqrt{\sum_{i=1}^{n}(u_i - v_i)^2}.$$

- **1 - Pearson correlation**. The Pearson correlation coefficient between two elements $x = (x_1, x_2, ..., x_n)$ and $y = (y_1, y_2, ..., y_n)$ is defined as

$$r = \frac{1}{n-1}\sum_{i=1}^{n}(\frac{x_i - \overline{x}}{s_x}) * (\frac{y_i - \overline{y}}{s_y})$$

  where $\overline{x}/\overline{y}$ is the average of values in x/y and s_x/s_y is the sample standard deviation of these values. It takes a value $\in [-1, 1]$. Highly correlated elements have a high absolute value of the Pearson correlation, and elements whose values are un-informative about each other have Pearson correlation 0. Using $1 - |Pearsoncorrelation|$ as distance measure means that elements that are highly correlated will have a short distance between them, and elements that have low correlation will be more distant from each other.

- **Manhattan distance**. The Manhattan distance between two points is the distance measured along axes at right angles. If $u = (u_1, u_2, \ldots, u_n)$ and $v = (v_1, v_2, \ldots, v_n)$, then the Manhattan distance between u and v is

$$|u - v| = \sum_{i=1}^{n}|u_i - v_i|.$$

Next, you can select the cluster linkage to be used:

- **Single linkage**. The distance between two clusters is computed as the distance between the two closest elements in the two clusters.

- **Average linkage**. The distance between two clusters is computed as the average distance between objects from the first cluster and objects from the second cluster. The averaging is performed over all pairs (x, y), where x is an object from the first cluster and y is an object from the second cluster.

- **Complete linkage**. The distance between two clusters is computed as the maximal object-to-object distance $d(x_i, y_j)$, where x_i comes from the first cluster, and y_j comes from the second cluster. In other words, the distance between two clusters is computed as the distance between the two farthest objects in the two clusters.

At the bottom, you can select which values to cluster (see section 3.2.1).

Click **Next** if you wish to adjust how to handle the results (see section **??**). If not, click **Finish**.

Result of hierarchical clustering of samples

The result of a sample clustering is shown in figure 3.32.

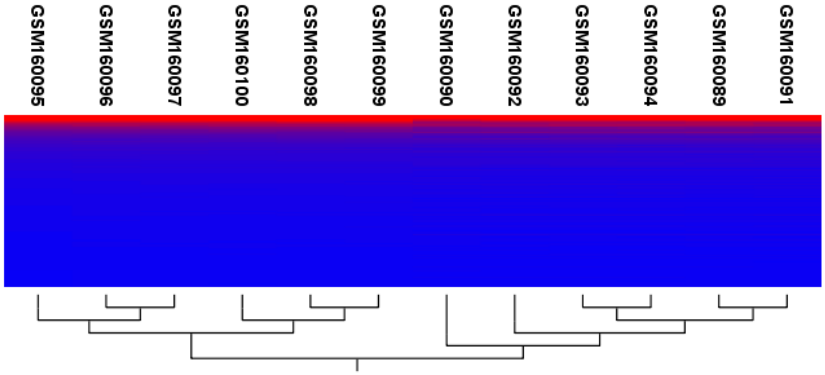


Figure 3.32: *Sample clustering.*

If you have used an **experiment** ( ) as input, the clustering is added to the experiment and will be saved when you save the experiment. It can be viewed by clicking the **Show Heat Map** ( ) button at the bottom of the view (see figure 3.33).



Figure 3.33: *Showing the hierarchical clustering of an experiment.*

If you have selected a number of **samples** (( ) or ( )) as input, a new element will be created that has to be saved separately.

Regardless of the input, the view of the clustering is the same. As you can see in figure 3.32, there is a tree at the bottom of the view to visualize the clustering. The names of the samples are listed at the top. The features are represented as horizontal lines, colored according to the expression level. If you place the mouse on one of the lines, you will see the names of the feature to the left. The features are sorted by their expression level in the first sample (in order to cluster the features, see section 3.5.1).

Researchers often have a priori knowledge of which samples in a study should be similar (e.g. samples from the same experimental condition) and which should be different (samples from biological distinct conditions). Thus, researches have expectations about how they should cluster. Samples that are placed unexpectedly in the hierarchical clustering tree may be samples that have been wrongly allocated to a group, samples of unintended or unclean tissue composition

or samples for which the processing has gone wrong. Unexpectedly placed samples, of course, could also be highly interesting samples.

There are a number of options to change the appearance of the heat map. At the top of the **Side Panel**, you find the **Heat map** preference group (see figure 3.34).

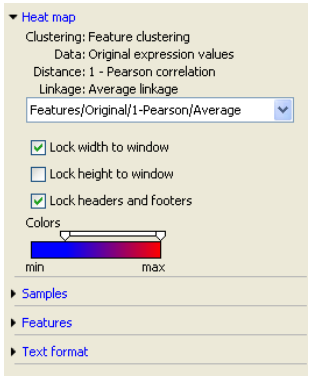


Figure 3.34: *Side Panel of heat map.*

At the top, there is information about the heat map currently displayed. The information regards type of clustering, expression value used together with distance and linkage information. If you have performed more than one clustering, you can choose between the resulting heat maps in a drop-down box (see figure 3.46).

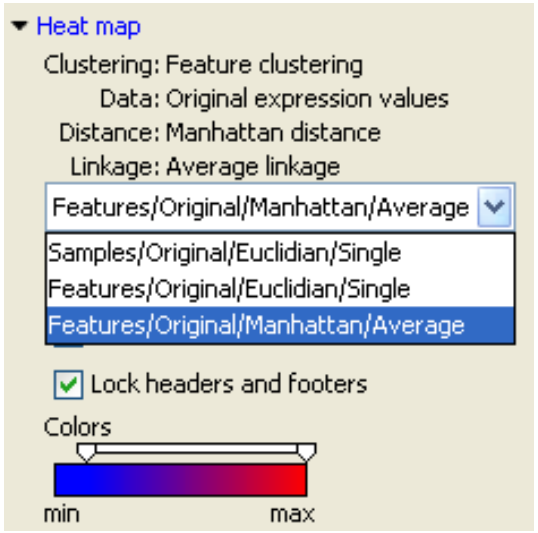


Figure 3.35: *When more than one clustering has been performed, there will be a list of heat maps to choose from.*

Note that if you perform an identical clustering, the existing heat map will simply be replaced. Below this box, there is a number of settings for displaying the heat map.

- **Lock width to window**. When you zoom in the heat map, you will per default only zoom in on the vertical level. This is because the width of the heat map is locked to the window. If you uncheck this option, you will zoom both vertically and horizontally. Since you always have more features than samples, it is useful to lock the width since you then have all the samples in view all the time.

- **Lock height to window**. This is the corresponding option for the height. Note that if you check both options, you will not be able to zoom at all, since both the width and the height is fixed.
- **Lock headers and footers**. This will ensure that you are always able to see the sample and feature names and the trees when you zoom in.
- **Colors**. The expression levels are visualized using a gradient color scheme, where the right side color is used for high expression levels and the left side color is used for low expression levels. You can change the coloring by clicking the box, and you can change the relative coloring of the values by dragging the two knobs on the white slider above.

Below you find the **Samples** and **Features** groups. They contain options to show names above/below and left/right, respectively. Furthermore, they contain options to show the tree above/below or left/right, respectively. Note that for clustering of samples, you find the tree options in the **Samples** group, and for clustering of features, you find the tree options in the **Features** group. With the tree options, you can also control the **Tree size**, from tiny to very large, and the option of showing the full tree, no matter how much space it will use.

Note that if you wish to use the same settings next time you open a heat map, you need to save the settings of the **Side Panel** (see section **??**).

3.3.3 Principal component analysis

A principal component analysis is a mathematical analysis that identifies and quantifies the directions of variability in the data. For a set of samples, e.g. an experiment, this can be done by finding the eigenvectors and eigenvalues of the covariance matrix of the samples. The eigenvectors are orthogonal. The first principal component is the eigenvector with the largest eigenvalue, and specifies the direction with the largest variability. The second principal component is the eigenvector with the second largest eigenvalue, and specifies the direction with the second largest variability. Similarly for the third, etc. The data can be projected onto the space spanned by the eigenvectors. A plot of the data in the space spanned by the first and second principal component will show a simplified version of the data with variability in other directions than the two major directions of variability ignored.

To start the analysis:

> **Toolbox** | **Expression Analysis (**) | **Quality Control** | **Principal Component Analysis (** **)**

Select a number of samples (( ) or ( )) or an experiment ( ) and click **Next**.

This will display a dialog as shown in figure 3.36.

In this dialog, you select the values to be used for the principal component analysis (see section 3.2.1).

Click **Next** if you wish to adjust how to handle the results (see section **??**). If not, click **Finish**.

Principal component analysis plot

This will create a principal component plot as shown in figure 3.37.

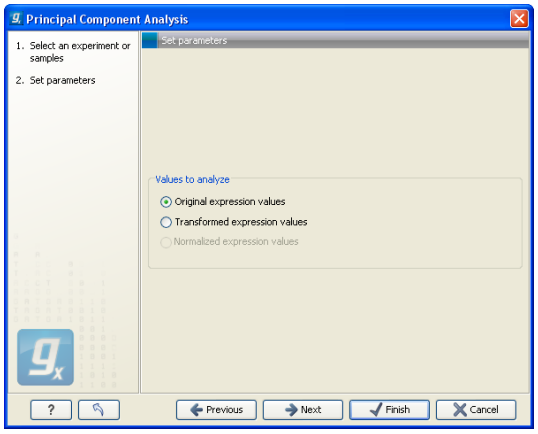


Figure 3.36: *Selcting which values the principal component analysis should be based on.*

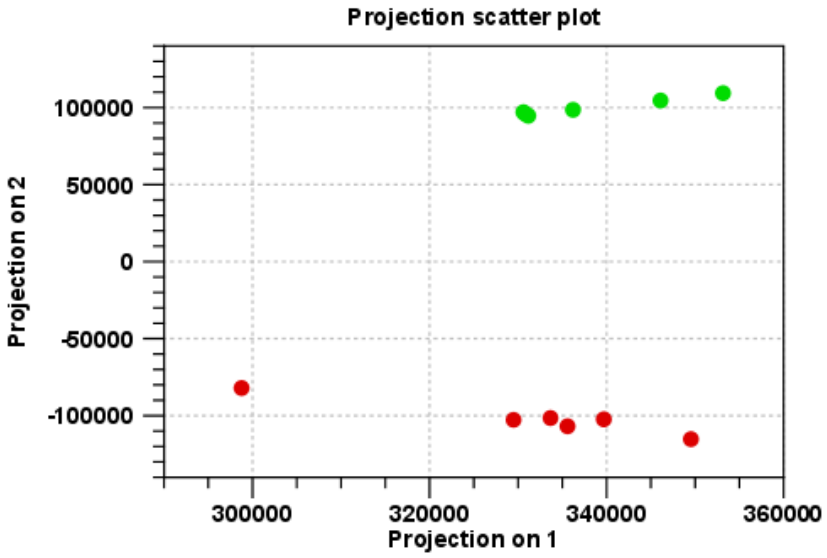


Figure 3.37: *A principal component analysis colored by group.*

The plot shows the projection of the samples onto the two-dimensional space spanned by the first and second principal component. (These are the orthogonal directions in which the data exhibits the largest and second-largest variability).

The plot in figure 3.37 is based on a two-group experiment. The group relationships are indicated by color. We expect the samples from within a group to exhibit less variability when compared, than samples from different groups. Thus samples should cluster according to groups and this is what we see. The PCA plot is thus helpful in identifying outlying samples and samples that have been wrongly assigned to a group.

In the **Side Panel** to the left, there is a number of options to adjust the view. Under **Graph preferences**, you can adjust the general properties of the plot.

- **Lock axes.** This will always show the axes even though the plot is zoomed to a detailed level.
- **Frame.** Shows a frame around the graph.

- **Show legends.** Shows the data legends.
- **Tick type**. Determine whether tick lines should be shown outside or inside the frame.
  - Outside
  - Inside
- **Tick lines at.** Choosing Major ticks will show a grid behind the graph.
  - None
  - Major ticks
- **Horizontal axis range**. Sets the range of the horizontal axis (x axis). Enter a value in **Min** and **Max**, and press Enter. This will update the view. If you wait a few seconds without pressing Enter, the view will also be updated.
- **Vertical axis range**. Sets the range of the vertical axis (y axis). Enter a value in **Min** and **Max**, and press Enter. This will update the view. If you wait a few seconds without pressing Enter, the view will also be updated.
- **y = 0 axis**. Draws a line where y = 0. Below there are some options to control the appearance of the line:
  - **Line width**
    - Thin
    - Medium
    - Wide
  - **Line type**
    - None
    - Line
    - Long dash
    - Short dash
  - **Line color.** Allows you to choose between many different colors. Click the color box to select a color.

Below the general preferences, you find the **Dot properties**:

- **Select sample or group**. When you wish to adjust the properties below, first select an item in this drop-down menu. That will apply the changes below to this item. If your plot is based on an experiment, the drop-down menu includes both group names and sample names, as well as an entry for selecting "All". If your plot is based on single elements, only sample names will be visible. Note that there are sometimes "mixed states" when you select a group where two of the samples e.g. have different colors. Selecting a new color in this case will erase the differences.
- **Dot type**
  - None
  - Cross

  - Plus
  - Square
  - Diamond
  - Circle
  - Triangle
  - Reverse triangle
  - Dot

- **Dot color.** Allows you to choose between many different colors. Click the color box to select a color.
- **Show name**. This will show a label with the name of the sample next to the dot. Note that the labels quickly get crowded, so that is why the names are not put on per default.

Note that if you wish to use the same settings next time you open a principal component plot, you need to save the settings of the **Side Panel** (see section **??**).

Scree plot

Besides the view shown in figure 3.37, the result of the principal component can also be viewed as a scree plot by clicking the **Show Scree Plot** () button at the bottom of the view. The scree plot shows the proportion of variation in the data explained by the each of the principal components. The first principal component explains about 99 percent of the variability.

In the **Side Panel** to the left, there is a number of options to adjust the view. Under **Graph preferences**, you can adjust the general properties of the plot.

- **Lock axes.** This will always show the axes even though the plot is zoomed to a detailed level.
- **Frame.** Shows a frame around the graph.
- **Show legends.** Shows the data legends.
- **Tick type**. Determine whether tick lines should be shown outside or inside the frame.
  - Outside
  - Inside
- **Tick lines at.** Choosing Major ticks will show a grid behind the graph.
  - None
  - Major ticks
- **Horizontal axis range**. Sets the range of the horizontal axis (x axis). Enter a value in **Min** and **Max**, and press Enter. This will update the view. If you wait a few seconds without pressing Enter, the view will also be updated.
- **Vertical axis range**. Sets the range of the vertical axis (y axis). Enter a value in **Min** and **Max**, and press Enter. This will update the view. If you wait a few seconds without pressing Enter, the view will also be updated.

The **Lines and plots** below contains the following parameters:

- **Dot type**
  - None
  - Cross
  - Plus
  - Square
  - Diamond
  - Circle
  - Triangle
  - Reverse triangle
  - Dot
- **Dot color.** Allows you to choose between many different colors. Click the color box to select a color.
- **Line width**
  - Thin
  - Medium
  - Wide
- **Line type**
  - None
  - Line
  - Long dash
  - Short dash
- **Line color.** Allows you to choose between many different colors. Click the color box to select a color.

Note that the graph title and the axes titles can be edited simply by clicking them with the mouse. These changes will be saved when you **Save** ( ) the graph - whereas the changes in the **Side Panel** need to be saved explicitly (see section **??**).

3.4 Statistical analysis - identifying differential expression

The *CLC Genomics Workbench* is designed to help you identify differential expression. You have a choice of a number of standard statistical tests, that are suitable for different data types and different types of experimental settings. There are two main categories of tests: tests that assume that the data has Gaussian distributions and compare means (described in section 3.4.1) and tests that compare proportions and assume that data consists of counts and (described in section 3.4.2). To run the statistical analysis:

Toolbox | **Expression Analysis ( )** | **Statistical Analysis** | **On Gaussian Data ( )**

or **Toolbox** | **Expression Analysis** ( ) | **Statistical Analysis** | **On Proportions** ( )

For both kinds of statistics you first select the experiment ( ) that you wish to use and click **Next** (learn more about setting up experiments in section 3.1.2).

The first part of the explanation of how to proceed and perform the statistical analysis is divided into two, depending on whether you are doing Gaussian-based tests or tests on proportions. The last part has an explanation of the options regarding corrected p-values which applies to all tests.

3.4.1 Gaussian-based tests

The tests based on the Gaussian distribution essentially compare the mean expression level in the experimental groups in the study, and evaluates the significance of the difference relative to the variance (or 'spread') of the data within the groups. The details of the formula used for calculating the test statistics vary according to the experimental setup and the assumptions you make about the data (read more about this in the sections on t-test and ANOVA below). The explanation of how to proceed is divided into two, depending on how many groups there are in your experiment. First comes the explanation for t-tests which is the only analysis available for two-group experimental setups (t-tests can also be used for pairwise comparison of groups in multi-group experiments). Next comes an explanation of the ANOVA test which can be used for multi-group experiments.

Note that the test statistics for the t-test and ANOVA analysis use the estimated group variances in their denominators. If all expression values in a group are identical the estimated variance for that group will be zero. If the estimated variances for both (or all) groups are zero the denominator of the test statistic will be zero. The numerator's value depends on the difference of the group means. If this is zero, the numerator is zero and the test statistic will be 0/0 which is NaN. If the numerator is different from zero the test statistic will be + or - infinity, depending on which group mean is bigger. If all values in all groups are identical the test statistic is set to zero.

T-tests

For experiments with two groups you can, among the Gaussian tests, only choose a **T-test** as shown in figure 3.38.

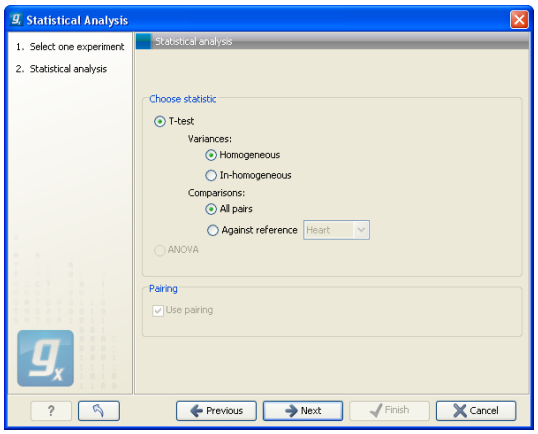


Figure 3.38: *Selecting a t-test.*

There are different types of t-tests, depending on the assumption you make about the variances

in the groups. By selecting 'Homogeneous' (the default) calculations are done assuming that the groups have equal variances. When 'In-homogeneous' is selected, this assumption is not made.

The t-test can also be chosen if you have a multi-group experiment. In this case you may choose either to have t-tests produced for all pairs of groups (by clicking the 'All pairs' button) or to have a t-test produced for each group compared to a specified reference group (by clicking the 'Against reference' button). In the last case you must specify which of the groups you want to use as reference (the default is to use the group you specified as Group 1 when you set up the experiment).

If a experiment with pairing was set up (see section 3.1.2) the **Use pairing** tick box is active. If ticked, paired t-tests will be calculated, if not, the formula for the standard t-test will be used.

When a t-test is run on an experiment four columns will be added to the experiment table for each pair of groups that are analyzed. The 'Difference' column contains the difference between the mean of the expression values across the samples assigned to group 2 and the mean of the expression values across the samples assigned to group 1. The 'Fold Change' column tells you how many times bigger the mean expression value in group 2 is relative to that of group 1. If the mean expression value in group 2 is bigger than that in group 1 this value is the mean expression value in group 2 divided by that in group 1. If the mean expression value in group 2 is smaller than that in group 1 the fold change is the mean expression value in group 1 divided by that in group 2 with a negative sign. The 'Test statistic' column holds that value of the test statistic, and the 'P-value' holds the two-sided p-value for the test. Up to two more columns may be added if the options to calculate Bonferroni and FDR corrected p-values were chosen (see 3.4.3).

ANOVA

For experiments with more than two groups you can choose **T-test** as described above, or **ANOVA** as shown in figure 3.39.

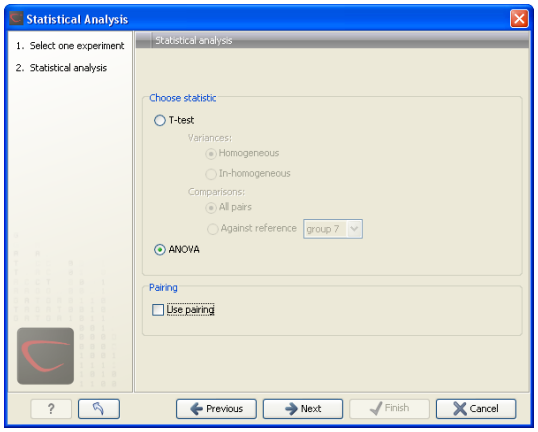


Figure 3.39: *Selecting ANOVA.*

The ANOVA method allows analysis of an experiment with one factor and a number of groups, e.g. different types of tissues, or time points. In the analysis, the variance within groups is compared to the variance between groups. You get a significant result (that is, a small ANOVA p-value) if the difference you see between groups relative to that within groups, is larger than what you would expect, if the data were really drawn from groups with equal means.

If an experiment with pairing was set up (see section 3.1.2) the **Use pairing** tick box is active. If

ticked, a repeated measures one-way ANOVA test will be calculated, if not, the formula for the standard one-way ANOVA will be used.

When an ANOVA analysis is run on an experiment four columns will be added to the experiment table for each pair of groups that are analyzed. The 'Max difference' column contains the difference between the maximum and minimum of the mean expression values of the groups, multiplied by -1 if the group with the maximum mean expression value occurs before the group with the minimum mean expression value (with the ordering: group 1, group 2, ...). The 'Max fold change' column contains the ratio of the maximum of the mean expression values of the groups to the minimum of the mean expression values of the groups, multiplied by -1 if the group with the maximum mean expression value occurs before the group with the minimum mean expression value (with the ordering: group 1, group 2, ...). The 'Test statistic' column holds the value of the test statistic, and the 'P-value' holds the two-sided p-value for the test. Up to two more columns may be added if the options to calculate Bonferroni and FDR corrected p-values were chosen (see 3.4.3).

3.4.2 Tests on proportions

The proportions-based tests are applicable in situations where your data samples consists of counts of a number of 'types' of data. This could e.g. be in a study where gene expression levels are measured by RNA-Seq or tag profiling. Here the different 'types' could correspond to the different 'genes' in a reference genome, and the counts could be the numbers of reads matching each of these genes. The tests compare counts by considering the proportions that they make up the total sum of counts in each sample. By comparing the expression levels at the level of proportions rather than raw counts, the data is corrected for sample size.

There are two tests available for comparing proportions: the test of [Kal et al., 1999] and the test of [Baggerly et al., 2003]. Both tests compare pairs of groups. If you have a multi-group experiment (see section 3.1.2), you may choose either to have tests produced for all pairs of groups (by clicking the 'All pairs' button) or to have a test produced for each group compared to a specified reference group (by clicking the 'Against reference' button). In the last case you must specify which of the groups you want to use as reference (the default is to use the group you specified as Group 1 when you set up the experiment).

Note that the proportion-based tests use the total sample counts (that is, the sum over all expression values). If one (or more) of the counts are NaN, the sum will be NaN and all the test statistics will be NaN. As a consequence all p-values will also be NaN. You can avoid this by filtering your experiment and creating a new experiment so that no NaN values are present, before you apply the tests.

Kal et al.'s test (Z-test)

Kal et al.'s test [Kal et al., 1999] compares a single sample against another single sample, and thus requires that each group in you experiment has only one sample. The test relies on an approximation of the binomial distribution by the normal distribution [Kal et al., 1999]. Considering proportions rather than raw counts the test is also suitable in situations where the sum of counts is different between the samples.

When Kal's test is run on an experiment four columns will be added to the experiment table for each pair of groups that are analyzed. The 'Proportions difference' column contains the difference

between the proportion in group 2 and the proportion in group 1. The 'Fold Change' column tells you how many times bigger the proportion in group 2 is relative to that of group 1. If the proportion in group 2 is bigger than that in group 1 this value is the proportion in group 2 divided by that in group 1. If the proportion in group 2 is smaller than that in group 1 the fold change is the proportion in group 1 divided by that in group 2 with a negative sign. The 'Test statistic' column holds that value of the test statistic, and the 'P-value' holds the two-sided p-value for the test. Up to two more columns may be added if the options to calculate Bonferroni and FDR corrected p-values were chosen (see 3.4.3).

Baggerley et al.'s test (Beta-binomial)

Baggerley et al.'s test [Baggerly et al., 2003] compares the proportions of counts in a group of samples against those of another group of samples, and is suited to cases where replicates are available in the groups. The samples are given different weights depending on their sizes (total counts). The weights are obtained by assuming a Beta distribution on the proportions in a group, and estimating these, along with the proportion of a binomial distribution, by the method of moments. The result is a weighted t-type test statistic.

When Baggerley's test is run on an experiment four columns will be added to the experiment table for each pair of groups that are analyzed. The 'Weighted proportions difference' column contains the difference between the mean of the weighted proportions across the samples assigned to group 2 and the mean of the weighted proportions across the samples assigned to group 1. The 'Fold Change' column tells you how many times bigger the mean of the weighted proportions in group 2 is relative to that of group 1. If the mean of the weighted proportions in group 2 is bigger than that in group 1 this value is the mean of the weighted proportions in group 2 divided by that in group 1. If the mean of the weighted proportions in group 2 is smaller than that in group 1 the fold change is the mean of the weighted proportions in group 1 divided by that in group 2 with a negative sign. The 'Test statistic' column holds that value of the test statistic, and the 'P-value' holds the two-sided p-value for the test. Up to two more columns may be added if the options to calculate Bonferroni and FDR corrected p-values were chosen (see 3.4.3).

3.4.3 Corrected p-values

Clicking **Next** will display a dialog as shown in figure 3.40.

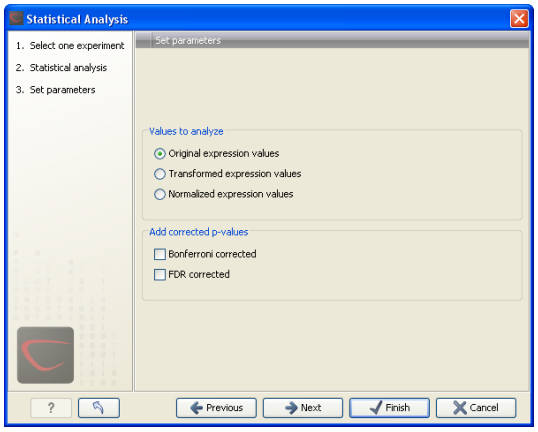


Figure 3.40: *Additional settings for the statistical analysis.*

At the top, you can select which values to analyze (see section 3.2.1).

Below you can select to add two kinds of corrected p-values to the analysis (in addition to the standard p-value produced for the test statistic):

- **Bonferroni corrected**.
- **FDR corrected**.

Both are calculated from the original p-values, and aim in different ways to take into account the issue of multiple testing [Dudoit et al., 2003]. The problem of multiple testing arises because the original p-values are related to a single test: the p-value is the probability of observing a more extreme value than that observed in the test carried out. If the p-value is 0.04, we would expect an as extreme value as that observed in 4 out of 100 tests carried out among groups with no difference in means. Popularly speaking, if we carry out 10000 tests and select the features with original p-values below 0.05, we will expect about 0.05 times 10000 = 500 to be false positives.

The Bonferroni corrected p-values handle the multiple testing problem by controlling the 'family-wise error rate': the probability of making at least one false positive call. They are calculated by multiplying the original p-values by the number of tests performed. The probability of having at least one false positive among the set of features with Bonferroni corrected p-values below 0.05, is less than 5%. The Bonferroni correction is conservative: there may be many genes that are differentially expressed among the genes with Bonferroni corrected p-values above 0.05, that will be missed if this correction is applied.

Instead of controlling the family-wise error rate we can control the false discovery rate: FDR. The false discovery rate is the proportion of false positives among all those declared positive. We expect 5 % of the features with FDR corrected p-values below 0.05 to be false positive. There are many methods for controlling the FDR - the method used in *CLC Genomics Workbench* is that of [Benjamini and Hochberg, 1995].

Click **Next** if you wish to adjust how to handle the results (see section **??**). If not, click **Finish**.

Note that if you have already performed statistical analysis on the same values, the existing one will be overwritten.

3.4.4 Volcano plots - inspecting the result of the statistical analysis

The results of the statistical analysis are added to the experiment and can be shown in the experiment table (see section 3.1.3). Typically columns containing the differences (or weighted differences) of the mean group values and the fold changes (or weighted fold changes) of the mean group values will be added along with a column of p-values. Also, columns with FDR or Bonferroni corrected p-values will be added if these were calculated. This added information allows features to be sorted and filtered to exclude the ones without sufficient proof of differential expression (learn more in section **??**).

If you want a more visual approach to the results of the statistical analysis, you can click the **Show Volcano Plot** () button at the bottom of the experiment table view. In the same way as the scatter plot presented in section 3.1.5, the volcano plot is yet another view on the experiment. Because it uses the p-values and mean differences produced by the statistical analysis, the plot is only available once a statistical analysis has been performed on the experiment.

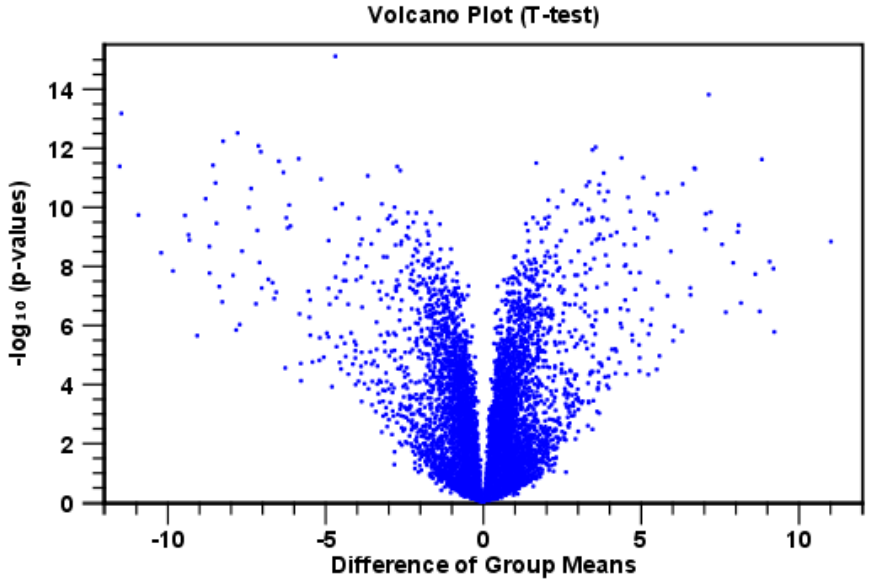


Figure 3.41: *Volcano plot.*

An example of a volcano plot is shown in figure 3.41.

The volcano plot shows the relationship between the p-values of a statistical test and the magnitude of the difference in expression values of the samples in the groups. On the y-axis the $-\log_{10}$ p-values are plotted. For the x-axis you may choose between two sets of values by choosing either 'Fold change' or 'Difference' in the volcano plot side panel's 'Values' part. If you choose 'Fold change' the log of the values in the 'fold change' (or 'Weighted fold change') column for the test will be displayed. If you choose 'Difference' the values in the 'Difference' (or 'Weighted difference') column will be used. Which values you wish to display will depend upon the scale of you data (Read the note on fold change in section 3.1.3).

The larger the difference in expression of a feature, the more extreme it's point will lie on the X-axis. The more significant the difference, the smaller the p-value and thus the higher the $-\log_{10}(p)$ value. Thus, points for features with highly significant differences will lie high in the plot. Features of interest are typically those which change significantly and by a certain magnitude. These are the points in the upper left and upper right hand parts of the volcano plot.

If you have performed different tests or you have an experiment with multiple groups you need to specify for which test and which group comparison you want the volcano plot to be shown. You do this in the 'Test' and 'Values' parts of the volcano plot side panel.

Options for the volcano plot are described in further detail when describing the **Side Panel** below.

If you place your mouse on one of the dots, a small text box will tell the name of the feature. Note that you can zoom in and out on the plot (see section **??**).

In the **Side Panel** to the right, there is a number of options to adjust the view of the volcano plot. Under **Graph preferences**, you can adjust the general properties of the volcano plot

- **Lock axes.** This will always show the axes even though the plot is zoomed to a detailed level.

- **Frame.** Shows a frame around the graph.
- **Show legends.** Shows the data legends.
- **Tick type**. Determine whether tick lines should be shown outside or inside the frame.
  - Outside
  - Inside
- **Tick lines at.** Choosing Major ticks will show a grid behind the graph.
  - None
  - Major ticks
- **Horizontal axis range**. Sets the range of the horizontal axis (x axis). Enter a value in **Min** and **Max**, and press Enter. This will update the view. If you wait a few seconds without pressing Enter, the view will also be updated.
- **Vertical axis range**. Sets the range of the vertical axis (y axis). Enter a value in **Min** and **Max**, and press Enter. This will update the view. If you wait a few seconds without pressing Enter, the view will also be updated.

Below the general preferences, you find the **Dot properties**, where you can adjust coloring and appearance of the dots.

- **Dot type**
  - None
  - Cross
  - Plus
  - Square
  - Diamond
  - Circle
  - Triangle
  - Reverse triangle
  - Dot
- **Dot color.** Allows you to choose between many different colors. Click the color box to select a color.

At the very bottom, you find two groups for choosing which values to display:

- **Test**. In this group, you can select which kind of test you want the volcano plot to be shown for.
- **Values**. Under **Values**, you can select which values to plot. If you have multi-group experiments, you can select which groups to compare. You can also select whether to plot **Difference** or **Fold change** on the x-axis. Read the note on fold change in section 3.1.3.

Note that if you wish to use the same settings next time you open a box plot, you need to save the settings of the **Side Panel** (see section **??**).

3.5 Feature clustering

Feature clustering is used to identify and cluster together features with similar expression patterns over samples (or experimental groups). Features that cluster together may be involved in the same biological process or be co-regulated. Also, by examining annotations of genes within a cluster, one may learn about the underlying biological processes involved in the experiment studied.

3.5.1 Hierarchical clustering of features

A hierarchical clustering of features is a tree presentation of the similarity in expression profiles of the features over a set of samples (or groups). The tree structure is generated by

1. letting each feature be a cluster
2. calculating pairwise distances between all clusters
3. joining the two closest clusters into one new cluster
4. iterating 2-3 until there is only one cluster left (which will contain all samples).

The tree is drawn so that the distances between clusters are reflected by the lengths of the branches in the tree. Thus, features with expression profiles that closely resemble each other have short distances between them, those that are more different, are placed further apart.

To start the clustering of features:

> **Toolbox** | **Expression Analysis (** **)** | **Feature Clustering** | **Hierarchical Clustering of Features (** **)**

Select at least two samples (( ) or ( )) or an experiment ( ).

Note! If your data contains many features, the clustering will take very long time and could make your computer unresponsive. It is recommended to perform this analysis on a subset of the data (which also makes it easier to make sense of the clustering. Typically, you will want to filter away the features that are thought to represent only noise, e.g. those with mostly low values, or with little difference between the samples). See how to create a sub-experiment in section 3.1.3.

Clicking **Next** will display a dialog as shown in figure 3.42. The hierarchical clustering algorithm requires that you specify a distance measure and a cluster linkage. The distance measure is used specify how distances between two features should be calculated. The cluster linkage specifies how you want the distance between two clusters, each consisting of a number of features, to be calculated.

At the top, you can choose three kinds of **Distance measures**:

- **Euclidean distance**. The ordinary distance between two points - the length of the segment connecting them. If $u = (u_1, u_2, \ldots, u_n)$ and $v = (v_1, v_2, \ldots, v_n)$, then the Euclidean distance between u and v is

$$|u - v| = \sqrt{\sum_{i=1}^{n} (u_i - v_i)^2}.$$

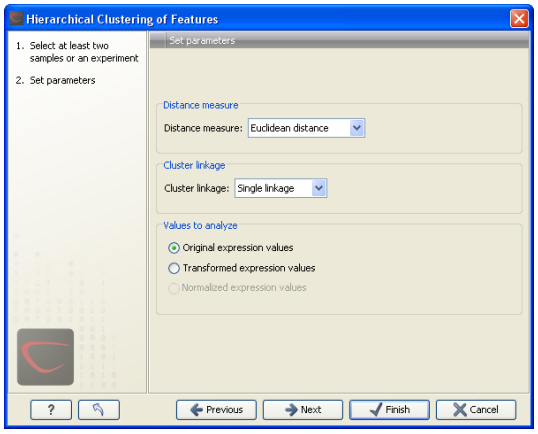


Figure 3.42: *Parameters for hierarchical clustering of features.*

- **1 - Pearson correlation**. The Pearson correlation coefficient between two elements $x = (x_1, x_2, ..., x_n)$ and $y = (y_1, y_2, ..., y_n)$ is defined as

$$r = \frac{1}{n-1}\sum_{i=1}^{n}\left(\frac{x_i - \overline{x}}{s_x}\right) * \left(\frac{y_i - \overline{y}}{s_y}\right)$$

  where $\overline{x}/\overline{y}$ is the average of values in x/y and s_x/s_y is the sample standard deviation of these values. It takes a value $\in [-1, 1]$. Highly correlated elements have a high absolute value of the Pearson correlation, and elements whose values are un-informative about each other have Pearson correlation 0. Using $1 - |Pearsoncorrelation|$ as distance measure means that elements that are highly correlated will have a short distance between them, and elements that have low correlation will be more distant from each other.

- **Manhattan distance**. The Manhattan distance between two points is the distance measured along axes at right angles. If $u = (u_1, u_2, \ldots, u_n)$ and $v = (v_1, v_2, \ldots, v_n)$, then the Manhattan distance between u and v is

$$|u - v| = \sum_{i=1}^{n} |u_i - v_i|.$$

Next, you can select different ways to calculate distances between clusters. The possible cluster linkage to use are:

- **Single linkage**. The distance between two clusters is computed as the distance between the two closest elements in the two clusters.
- **Average linkage**. The distance between two clusters is computed as the average distance between objects from the first cluster and objects from the second cluster. The averaging is performed over all pairs (x, y), where x is an object from the first cluster and y is an object from the second cluster.
- **Complete linkage**. The distance between two clusters is computed as the maximal object-to-object distance $d(x_i, y_j)$, where x_i comes from the first cluster, and y_j comes from the second cluster. In other words, the distance between two clusters is computed as the distance between the two farthest objects in the two clusters.

At the bottom, you can select which values to cluster (see section 3.2.1). Click **Next** if you wish to adjust how to handle the results (see section **??**). If not, click **Finish**.

Result of hierarchical clustering of features

The result of a feature clustering is shown in figure 3.43.

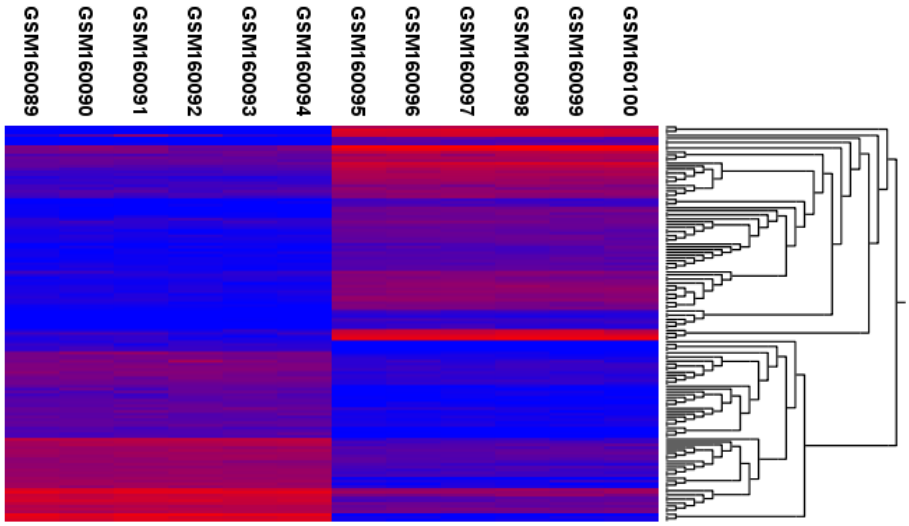


Figure 3.43: *Hierarchical clustering of features.*

If you have used an **experiment** ( ) as input, the clustering is added to the experiment and will be saved when you save the experiment. It can be viewed by clicking the **Show Heat Map** ( ) button at the bottom of the view (see figure 3.44).



Figure 3.44: *Showing the hierarchical clustering of an experiment.*

If you have selected a number of **samples** (( ) or ( )) as input, a new element will be created that has to be saved separately.

Regardless of the input, a hierarchical tree view with associated heatmap is produced (figure 3.43). In the heatmap each row corresponds to a feature and each column to a sample. The color in the i'th row and j'th column reflects the expression level of feature i in sample j (the color scale can be set in the side panel). The order of the rows in the heatmap are determined by the hierarchical clustering. If you place the mouse on one of the rows, you will see the name of the corresponding feature to the left. The order of the columns (that is, samples) is determined by their input order or (if defined) experimental grouping. The names of the samples are listed at the top of the heatmap and the samples are organized into groups.

There are a number of options to change the appearance of the heat map. At the top of the **Side Panel**, you find the **Heat map** preference group (see figure 3.45).

At the top, there is information about the heat map currently displayed. The information regards type of clustering, expression value used together with distance and linkage information. If you have performed more than one clustering, you can choose between the resulting heat maps in a drop-down box (see figure 3.46).

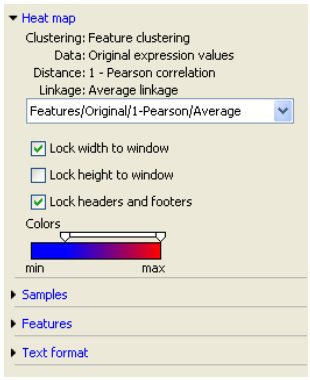


Figure 3.45: *Side Panel of heat map.*

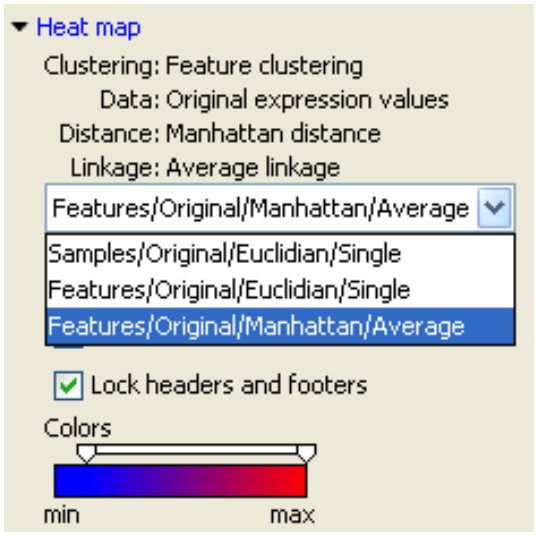


Figure 3.46: *When more than one clustering has been performed, there will be a list of heat maps to choose from.*

Note that if you perform an identical clustering, the existing heat map will simply be replaced. Below this box, there is a number of settings for displaying the heat map.

- **Lock width to window**. When you zoom in the heat map, you will per default only zoom in on the vertical level. This is because the width of the heat map is locked to the window. If you uncheck this option, you will zoom both vertically and horizontally. Since you always have more features than samples, it is useful to lock the width since you then have all the samples in view all the time.
- **Lock height to window**. This is the corresponding option for the height. Note that if you check both options, you will not be able to zoom at all, since both the width and the height is fixed.
- **Lock headers and footers**. This will ensure that you are always able to see the sample and feature names and the trees when you zoom in.
- **Colors**. The expression levels are visualized using a gradient color scheme, where the right side color is used for high expression levels and the left side color is used for low expression levels. You can change the coloring by clicking the box, and you can change the relative coloring of the values by dragging the two knobs on the white slider above.

Below you find the **Samples** and **Features** groups. They contain options to show names above/below and left/right, respectively. Furthermore, they contain options to show the tree above/below or left/right, respectively. Note that for clustering of samples, you find the tree options in the **Samples** group, and for clustering of features, you find the tree options in the **Features** group. With the tree options, you can also control the **Tree size**, from tiny to very large, and the option of showing the full tree, no matter how much space it will use.

Note that if you wish to use the same settings next time you open a heat map, you need to save the settings of the **Side Panel** (see section **??**).

3.5.2 K-means/medoids clustering

In a k-means or medoids clustering, features are clustered into k separate clusters. The procedures seek to find an assignment of features to clusters, for which the distances between features within the cluster is small, while distances between clusters are large.

> **Toolbox** | **Expression Analysis** ( ) | **Feature Clustering** | **K-means/medoids Clustering** ( )

Select at least two samples (( ) or ( )) or an experiment ( ).

Note! If your data contains many features, the clustering will take very long time and could make your computer unresponsive. It is recommended to perform this analysis on a subset of the data (which also makes it easier to make sense of the clustering). See how to create a sub-experiment in section 3.1.3.

Clicking **Next** will display a dialog as shown in figure 3.47.

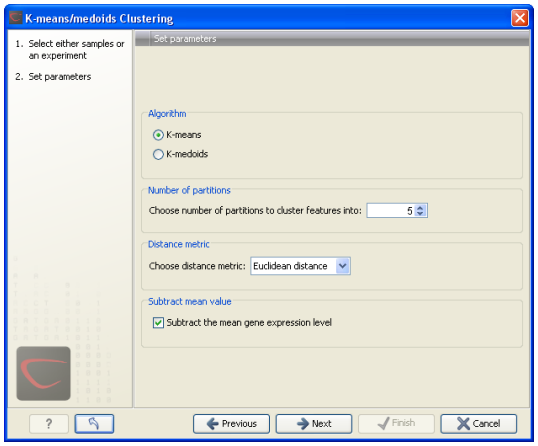


Figure 3.47: *Parameters for k-means/medoids clustering.*

The parameters are:

- **Algorithm**. You can choose between two clustering methods:
  - **K-means**. K-means clustering assigns each point to the cluster whose center is nearest. The center/centroid of a cluster is defined as the average of all points in the cluster. If a data set has three dimensions and the cluster has two points $X = (x_1, x_2, x_3)$ and $Y = (y_1, y_2, y_3)$, then the centroid Z becomes $Z = (z_1, z_2, z_3)$, where $z_i = (x_i + y_i)/2$ for $i = 1, 2, 3$. The algorithm attempts to minimize the

intra-cluster variance defined by:

$$V = \sum_{i=1}^{k} \sum_{x_j \in S_i} (x_j - \mu_i)^2$$

where there are k clusters $S_i, i = 1, 2, \ldots, k$ and μ_i is the centroid of all points $x_j \in S_i$. The detailed algorithm can be found in [Lloyd, 1982].

– **K-medoids**. K-medoids clustering is computed using the PAM-algorithm (PAM is short for Partitioning Around Medoids). It chooses datapoints as centers in contrast to the K-means algorithm. The PAM-algorithm is based on the search for k representatives (called medoids) among all elements of the dataset. When having found k representatives k clusters are now generated by assigning each element to its nearest medoid. The algorithm first looks for a good initial set of medoids (the BUILD phase). Then it finds a local minimum for the objective function:

$$V = \sum_{i=1}^{k} \sum_{x_j \in S_i} (x_j - c_i)^2$$

where there are k clusters $S_i, i = 1, 2, \ldots, k$ and c_i is the medoid of S_i. This solution implies that there is no single switch of an object with a medoid that will decrease the objective (this is called the SWAP phase). The PAM-agorithm is described in [Kaufman and Rousseeuw, 1990].

- **Number of partitions**. The number of partitions to cluster features into.
- **Distance metric**. The metric to compute distance between data points.
  – **Euclidean distance**. The ordinary distance between two elements - the length of the segment connecting them. If $u = (u_1, u_2, \ldots, u_n)$ and $v = (v_1, v_2, \ldots, v_n)$, then the Euclidean distance between u and v is

$$|u - v| = \sqrt{\sum_{i=1}^{n} (u_i - v_i)^2}.$$

  – **Manhattan distance**. The Manhattan distance between two elements is the distance measured along axes at right angles. If $u = (u_1, u_2, \ldots, u_n)$ and $v = (v_1, v_2, \ldots, v_n)$, then the Manhattan distance between u and v is

$$|u - v| = \sum_{i=1}^{n} |u_i - v_i|.$$

- **Subtract mean value**. For each gene, subtract the mean gene expression value over all input samples.

Clicking **Next** will display a dialog as shown in figure 3.48.

At the top, you can choose the **Level** to use. Choosing 'sample values' means that distances will be calculated using all the individual values of the samples. When 'group means' are chosen, distances are calculated using the group means.

At the bottom, you can select which values to cluster (see section 3.2.1).

Click **Next** if you wish to adjust how to handle the results (see section **??**). If not, click **Finish**.

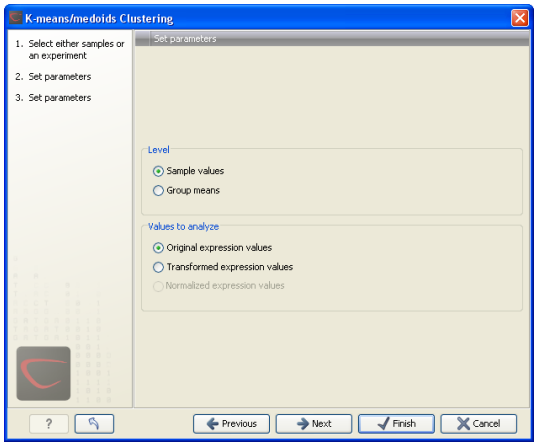


Figure 3.48: *Parameters for k-means/medoids clustering.*

Viewing the result of k-means/medoids clustering

The result of the clustering is a number of graphs. The number depends on the number of partitions chosen (figure 3.47) - there is one graph per cluster. Using drag and drop as explained in section **??**, you can arrange the views to see more than one graph at the time.

Figure 3.49 shows an example where four clusters have been arranged side-by-side.

The samples used are from a time-series experiment, and you can see that the expression levels for each cluster have a distinct pattern. The two clusters at the bottom have falling and rising expression levels, respectively, and the two clusters at the top both fall at the beginning but then rise again (the one to the right starts to rise earlier that the other one).

Having inspected the graphs, you may wish to take a closer look at the features represented in each cluster. In the experiment table, the clustering has added an extra column with the name of the cluster that the feature belongs to. In this way you can filter the table to see only features from a specific cluster. This also means that you can select the feature of this cluster in a volcano or scatter plot as described in section 3.1.6.

3.6 Annotation tests

The annotation tests are tools for detecting significant patterns among features (e.g. genes) of experiments, based on their annotations. This may help in interpreting the analysis of the large numbers of features in an experiment in a biological context. Which biological context, depends on which annotation you choose to examine, and could e.g. be biological process, molecular function or pathway as specified by the Gene Ontology or KEGG. The annotation testing tools of course require that the features in the experiment you want to analyze are annotated. Learn how to annotate an experiment in section 3.1.4.

3.6.1 Hypergeometric tests on annotations

The first approach to using annotations to extract biological information is the hypergeometric annotation test. This test measures the extend to which the annotation categories of features in a smaller gene list, 'A', are over or under-represented relative to those of the features in larger gene list 'B', of which 'A' is a sub-list. Gene list B is often the features of the full experiment, possibly with features which are thought to represent only noise, filtered away. Gene list A is

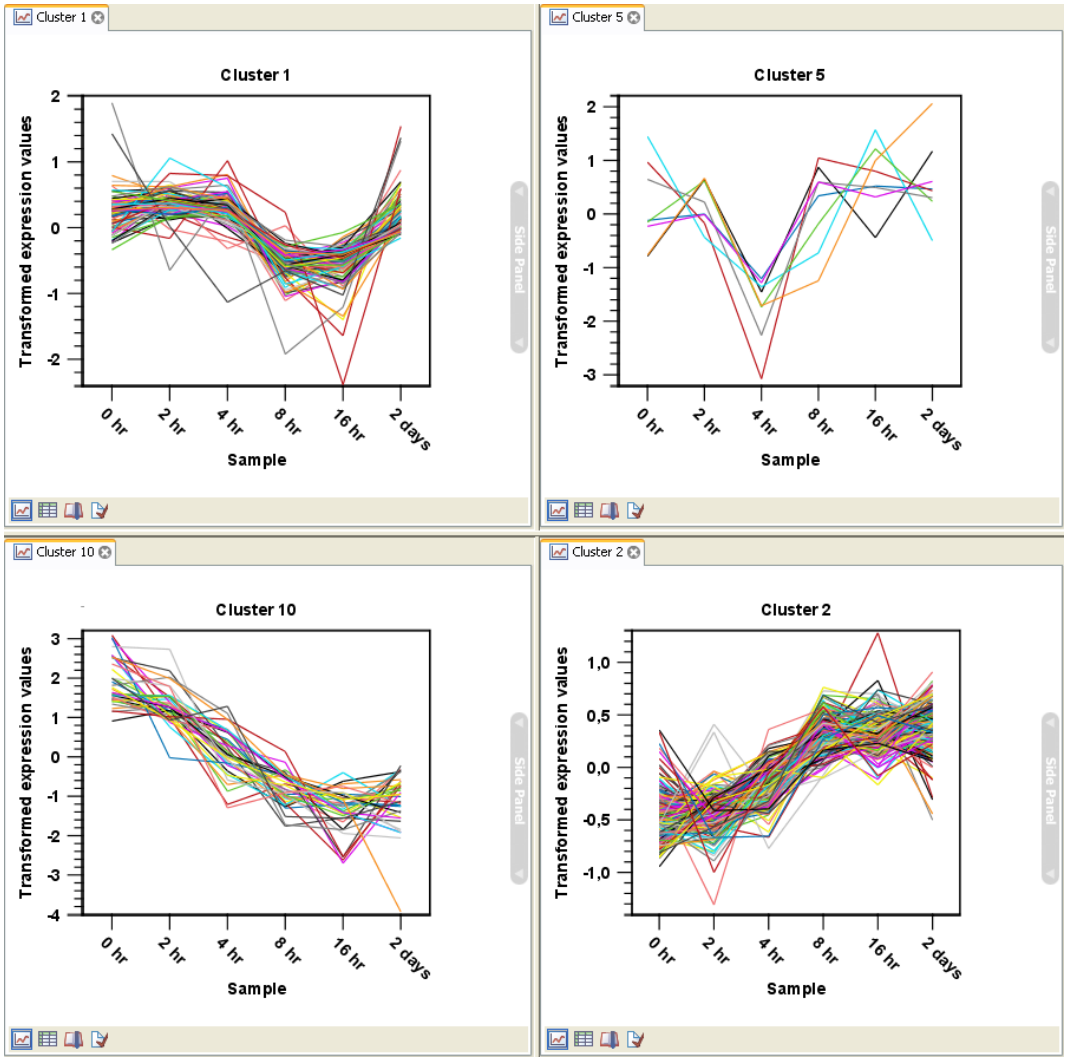


Figure 3.49: *Four clusters created by k-means/medoids clustering.*

a sub-experiment of the full experiment where most features have been filtered away and only those that seem of interest are kept. Typically gene list A will consist of a list of candidate differentially expressed genes. This could be the gene list obtained after carrying out a statistical analysis on the experiment, and keeping only features with FDR corrected p-values <0.05 and a fold change which is larger than 2 in absolute value. The hyper geometric test procedure implemented is similar to the unconditional GOstats test of [Falcon and Gentleman, 2007].

Toolbox | Expression Analysis ( ) | Annotation Test | Hypergeometric Tests on Annotations ( )

This will show a dialog where you can select the two experiments - the larger experiment, e.g. the original experiment including the full list of features - and a sub-experiment (see how to create a sub-experiment in section 3.1.3).

Click **Next**. This will display the dialog shown in figure 3.50.

At the top, you select which annotation to use for testing. You can select from all the annotations available on the experiment, but it is of course only a few that are biologically relevant. Once you have selected an annotation, you will see the number of features carrying this annotation below.

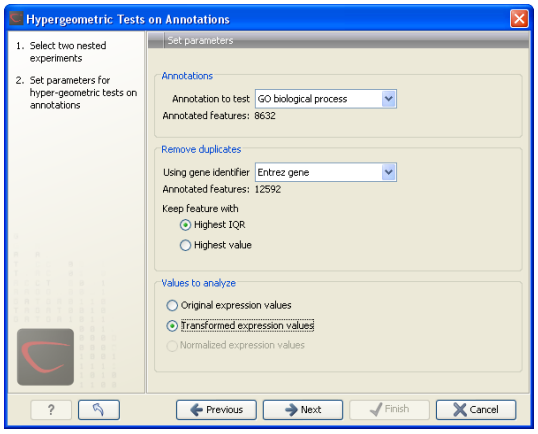


Figure 3.50: *Parameters for performing a hypergeometric test on annotations*

Annotations are typically given at the gene level. Often a gene is represented by more than one feature in an experiment. If this is not taken into account it may lead to a biased result. The standard way to deal with this is to reduce the set of features considered, so that each gene is represented only once. In the next step, **Remove duplicates**, you can choose how you want this to be done:

- **Using gene identifier**.
- **Keep feature with**:
  - **Highest IQR**. The feature with the highest interquartile range (IQR) is kept.
  - **Highest value**. The feature with the highest expression value is kept.

First you specify which annotation you want to use as gene identifier. Once you have selected this, you will see the number of features carrying this annotation below. Next you specify which feature you want to keep for each gene. This may be either the feature with the highest inter-quartile range or the highest value.

At the bottom, you can select which values to analyze (see section 3.2.1).

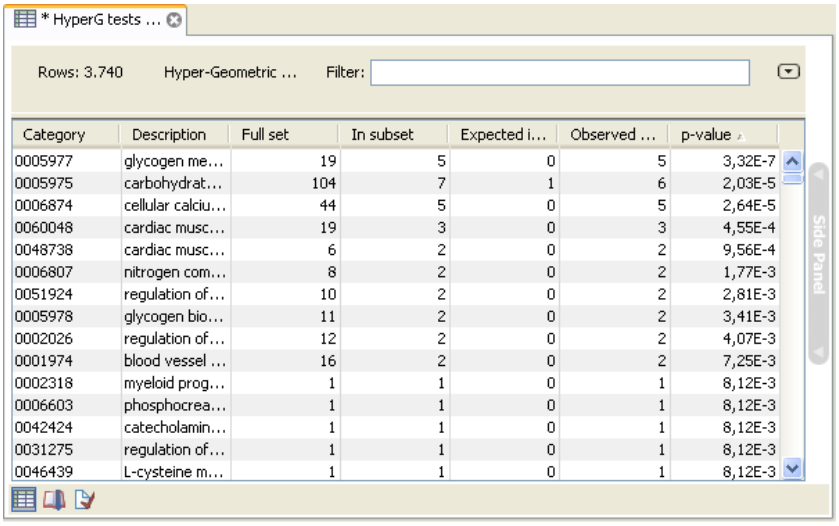
Click **Next** if you wish to adjust how to handle the results (see section **??**). If not, click **Finish**.

Result of hypergeometric tests on annotations

The result of performing hypergeometric tests on annotations using GO biological process is shown in figure 3.51.

The table shows the following information:

- **Category**. This is the identifier for the category.
- **Description**. This is the description belonging to the category. Both of these are simply extracted from the annotations.
- **Full set**. The number of features in the original experiment (not the subset) with this category. (Note that this is after removal of duplicates).



Category	Description	Full set	In subset	Expected i...	Observed ...	p-value
0005977	glycogen me...	19	5	0	5	3,32E-7
0005975	carbohydrat...	104	7	1	6	2,03E-5
0006874	cellular calciu...	44	5	0	5	2,64E-5
0060048	cardiac musc...	19	3	0	3	4,55E-4
0048738	cardiac musc...	6	2	0	2	9,56E-4
0006807	nitrogen com...	8	2	0	2	1,77E-3
0051924	regulation of...	10	2	0	2	2,81E-3
0005978	glycogen bio...	11	2	0	2	3,41E-3
0002026	regulation of...	12	2	0	2	4,07E-3
0001974	blood vessel ...	16	2	0	2	7,25E-3
0002318	myeloid prog...	1	1	0	1	8,12E-3
0006603	phosphocrea...	1	1	0	1	8,12E-3
0042424	catecholamin...	1	1	0	1	8,12E-3
0031275	regulation of...	1	1	0	1	8,12E-3
0046439	L-cysteine m...	1	1	0	1	8,12E-3

Figure 3.51: *The result of testing on GO biological process.*

- **In subset**. The number of features in the the subset with this category. (Note that this is after removal of duplicates).
- **Expected in subset**. The number of features we would have expected to find with this annotation category in the subset, if the subset was a random draw from the full set.
- **Observed - expected**. 'In subset' - 'Expected in subset'
- **p-value**. The tail probability of the hyper geometric distribution This is the value used for sorting the table.

Categories with small p-values are categories that are over or under-represented on the features in the subset relative to the full set.

3.6.2 Gene set enrichment analysis

When carrying out a hypergeometric test on annotations you typically compare the annotations of the genes in a subset containing 'the significantly differentially expressed genes' to those of the total set of genes in the experiment. Which, and how many, genes are included in the subset is somewhat arbitrary - using a larger or smaller p-value cut-off will result in including more or less. Also, the magnitudes of differential expression of the genes is not considered.

The Gene Set Enrichment Analysis (GSEA) does NOT take a sublist of differentially expressed genes and compare it to the full list - it takes a single gene list (a single experiment). The idea behind GSEA is to consider a measure of association between the genes and phenotype of interest (e.g. test statistic for differential expression) and rank the genes according to this measure of association. A test is then carried out for each annotation category, for whether the ranks of the genes in the category are evenly spread throughout the ranked list, or tend to occur at the top or bottom of the list.

The GSEA test implemented here is that of [Tian et al., 2005]. The test implicitly calculates and uses a standard t-test statistic for two-group experiments, and ANOVA statistic for multiple group experiments for each feature, as measures of association. For each category, the test statistics for the features in than category are summed and a category based test statistic is calculated

as this sum divided by the square root of the number of features in the category. Note that if a feature has the value NaN in one of the samples, the t-test statistic for the feature will be NaN. Consequently, the combined statistic for each of the categories in which the feature is included will be NaN. Thus, it is advisable to filter out any feature that has a NaN value before applying GSEA.

The p-values for the GSEA test statistics are calculated by permutation: The original test statistics for the features are permuted and new test statistics are calculated for each category, based on the permuted feature test statistics. This is done the number of times specified by the user in the wizard. For each category, the lower and upper tail probabilities are calculated by comparing the original category test statistics to the distribution of the permutation-based test statistics for that category. The lower and higher tail probabilities are the number of these that are lower and higher, respectively, than the observed value, divided by the number of permutations.

As the p-values are based on permutations you may some times see results where category x's test statistic is lower than that of category y and the categories are of equal size, but where the lower tail probability of category x is higher than that of category y. This is due to imprecision in the estimations of the tail probabilities from the permutations. The higher the number of permutations, the more stable the estimation.

You may run a GSEA on a full experiment, or on a sub-experiment where you have filtered away features that you think are un-informative and represent only noise. Typically you will remove features that are constant across samples (those for which the value in the 'Range' column is zero' — these will have a t-test statistic of zero) and/or those for which the inter-quantile range is small. As the GSEA algorithm calculates and ranks genes on p-values from a test of differential expression, it will generally not make sense to filter the experiment on p-values produced in an analysis if differential expression, prior to running GSEA on it.

> **Toolbox** | **Expression Analysis (**  **)** | **Annotation Test** | **Gene Set Enrichment Analysis (GSEA) (** **)**

Select an experiment and click **Next**.

Click **Next**. This will display the dialog shown in figure 3.52.

At the top, you select which annotation to use for testing. You can select from all the annotations available on the experiment, but it is of course only a few that are biologically relevant. Once you have selected an annotation, you will see the number of features carrying this annotation below.

In addition, you can set a filter: **Minimum size required**. Only categories with more genes (i.e. features) than the specified number will be considered. Excluding categories with small numbers of genes may lead to more robust results.

Annotations are typically given at the gene level. Often a gene is represented by more than one feature in an experiment. If this is not taken into account it may lead to a biased result. The standard way to deal with this is to reduce the set of features considered, so that each gene is represented only once. Check the **Remove duplicates** check box to reduce the feature set, and you can choose how you want this to be done:

- **Using gene identifier**.
- **Keep feature with**:
  - **Highest IQR**. The feature with the highest interquartile range (IQR) is kept.

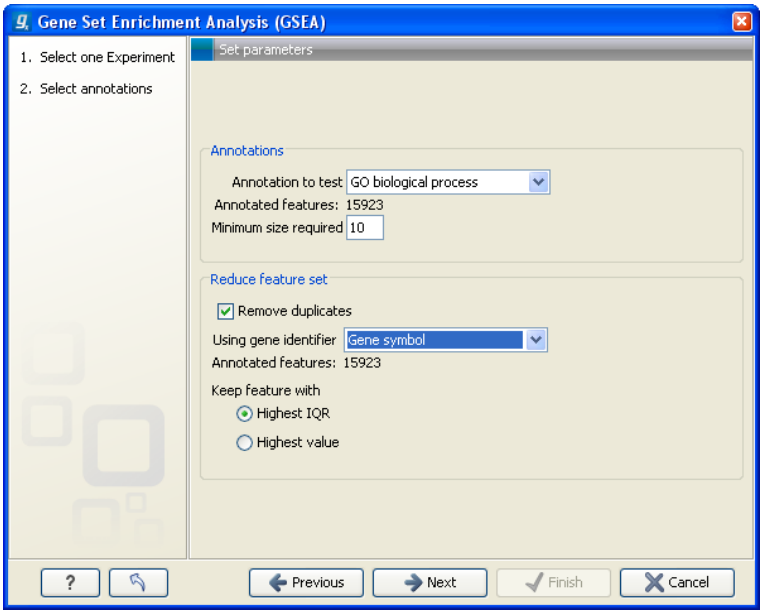


Figure 3.52: *Gene set enrichment analysis on GO biological process*

– **Highest value**. The feature with the highest expression value is kept.

First you specify which annotation you want to use as gene identifier. Once you have selected this, you will see the number of features carrying this annotation below. Next you specify which feature you want to keep for each gene. This may be either the feature with the highest inter-quartile range or the highest value.

Clicking **Next** will display the dialog shown in figure 3.53.

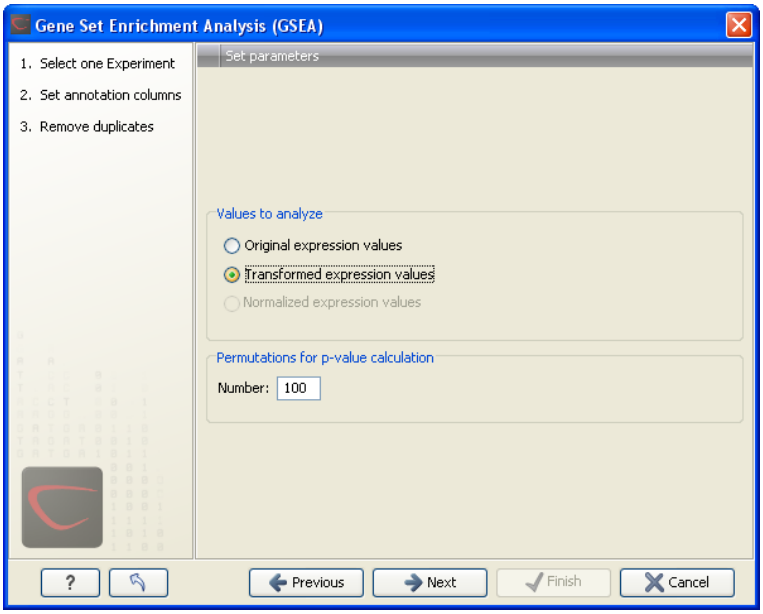


Figure 3.53: *Gene set enrichment analsysis parameters.*

At the top, you can select which values to analyze (see section 3.2.1).

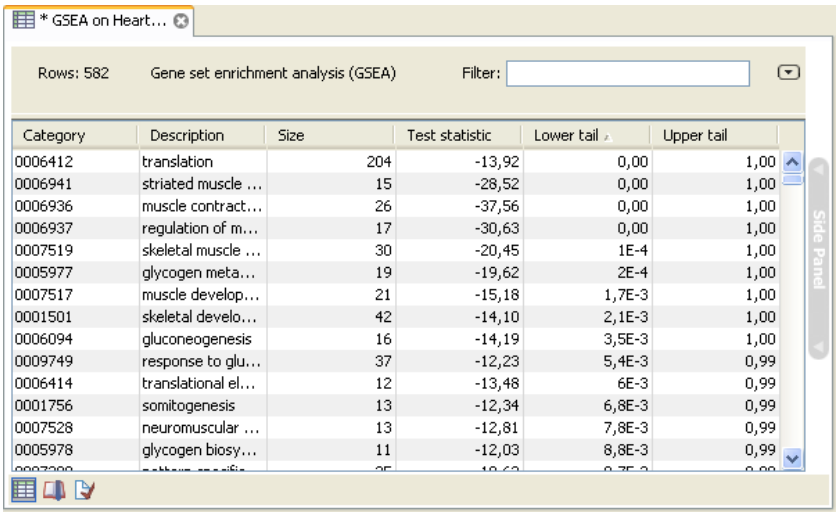
Below, you can set the **Permutations for p-value calculation**. For the GSEA test a p-value is

calculated by permutation: p permuted data sets are generated, each consisting of the original features, but with the test statistics permuted. The GSEA test is run on each of the permuted data sets. The test statistic is calculated on the original data, and the resulting value is compared to the distribution of the values obtained for the permuted data sets. The permutation based p-value is the number of permutation based test statistics above (or below) the value of the test statistic for the original data, divided by the number of permuted data sets. For reliable permutation-based p-value calculation a large number of permutations is required (100 is the default).

Click **Next** if you wish to adjust how to handle the results (see section **??**). If not, click **Finish**.

Result of gene set enrichment analysis

The result of performing gene set enrichment analysis using GO biological process is shown in figure 3.54.



Category	Description	Size	Test statistic	Lower tail	Upper tail
0006412	translation	204	-13,92	0,00	1,00
0006941	striated muscle ...	15	-28,52	0,00	1,00
0006936	muscle contract...	26	-37,56	0,00	1,00
0006937	regulation of m...	17	-30,63	0,00	1,00
0007519	skeletal muscle ...	30	-20,45	1E-4	1,00
0005977	glycogen meta...	19	-19,62	2E-4	1,00
0007517	muscle develop...	21	-15,18	1,7E-3	1,00
0001501	skeletal develo...	42	-14,10	2,1E-3	1,00
0006094	gluconeogenesis	16	-14,19	3,5E-3	1,00
0009749	response to glu...	37	-12,23	5,4E-3	0,99
0006414	translational el...	12	-13,48	6E-3	0,99
0001756	somitogenesis	13	-12,34	6,8E-3	0,99
0007528	neuromuscular ...	13	-12,81	7,8E-3	0,99
0005978	glycogen biosy...	11	-12,03	8,8E-3	0,99

Figure 3.54: *The result of gene set enrichment analysis on GO biological process.*

The table shows the following information:

- **Category**. This is the identifier for the category.
- **Description**. This is the description belonging to the category. Both of these are simply extracted from the annotations.
- **Size**. The number of features with this category. (Note that this is after removal of duplicates).
- **Test statistic**. This is the GSEA test statistic.
- **Lower tail**. This is the mass in the permutation based p-value distribution below the value of the test statistic.
- **Upper tail**. This is the mass in the permutation based p-value distribution above the value of the test statistic.

A small lower (or upper) tail p-value for an annotation category is an indication that features in this category viewed as a whole are perturbed among the groups in the experiment considered.

3.7 General plots

The last folder in the **Expression Analysis** ( ) folder in the **Toolbox** is **General Plots**. Here you find three general plots that may be useful at various point of your analysis work flow. The plots are explained in detail below.

3.7.1 Histogram

A histogram shows a distribution of a set of values. Histograms are often used for examining and comparing distributions, e.g. of expression values of different samples, in the quality control step of an analysis. You can create a histogram showing the distribution of expression value for a sample:

Toolbox | **Expression Analysis ( )** | **General Plots** | **Create Histogram ( )**

Select a number of samples (( ) or ( )). When you have selected more than one sample, a histogram will be created for each one. Clicking **Next** will display a dialog as shown in figure 3.55.

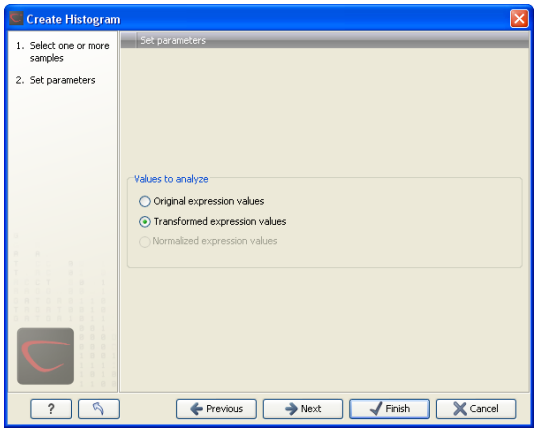


Figure 3.55: *Selcting which values the histogram should be based on.*

In this dialog, you select the values to be used for creating the histogram (see section 3.2.1).

Click **Next** if you wish to adjust how to handle the results (see section **??**). If not, click **Finish**.

Viewing histograms

The resulting histogram is shown in a figure 3.56

The histogram shows the expression value on the x axis (in the case of figure 3.56 the transformed expression values) and the counts of these values on the y axis.

In the **Side Panel** to the left, there is a number of options to adjust the view. Under **Graph preferences**, you can adjust the general properties of the plot.

- **Lock axes.** This will always show the axes even though the plot is zoomed to a detailed level.
- **Frame.** Shows a frame around the graph.
- **Show legends.** Shows the data legends.

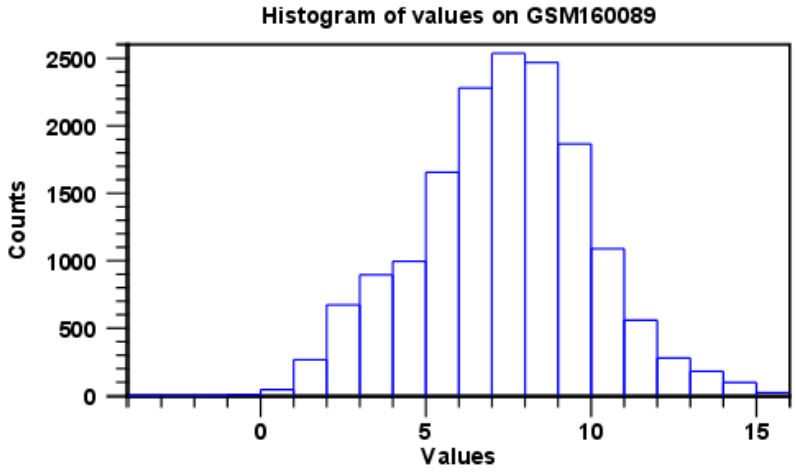


Figure 3.56: *Histogram showing the distribution of transformed expression values.*

- **Tick type**. Determine whether tick lines should be shown outside or inside the frame.
  - Outside
  - Inside
- **Tick lines at.** Choosing Major ticks will show a grid behind the graph.
  - None
  - Major ticks
- **Horizontal axis range**. Sets the range of the horizontal axis (x axis). Enter a value in **Min** and **Max**, and press Enter. This will update the view. If you wait a few seconds without pressing Enter, the view will also be updated.
- **Vertical axis range**. Sets the range of the vertical axis (y axis). Enter a value in **Min** and **Max**, and press Enter. This will update the view. If you wait a few seconds without pressing Enter, the view will also be updated.
- **Break points**. Determines where the bars in the histogram should be:
  - **Sturges method**. This is the default. The number of bars is calculated from the range of values by Sturges formula [Sturges, 1926].
  - **Equi-distanced bars**. This will show bars from **Start** to **End** and with a width of **Sep**.
  - **Number of bars**. This will simply create a number of bars starting at the lowest value and ending at the highest value.

Below the graph preferences, you find **Line color.** Allows you to choose between many different colors. Click the color box to select a color.

Note that if you wish to use the same settings next time you open a principal component plot, you need to save the settings of the **Side Panel** (see section **??**).

Besides the histogram view itself, the histogram can also be shown in a table, summarizing key properties of the expression values. An example is shown in figure 3.57.

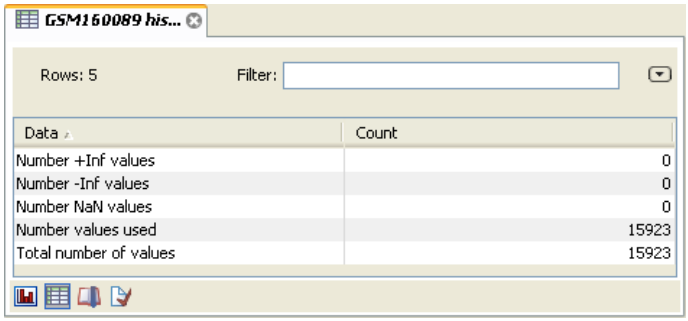


Figure 3.57: *Table view of a histogram.*

The table lists the following properties:

- **Number +Inf values**
- **Number -Inf values**
- **Number NaN values**
- **Number values used**
- **Total number of values**

3.7.2 MA plot

The MA plot is a scatter rotated by $45°$. For two samples of expression values it plots for each gene the difference in expression against the mean expression level. MA plots are often used for quality control, in particular, to assess whether normalization and/or transformation is required.

You can create an MA plot comparing two samples:

Toolbox | **Expression Analysis (**) | **General Plots** | **Create MA Plot (**)

Select two samples (( ) or ( )). Clicking **Next** will display a dialog as shown in figure 3.58.

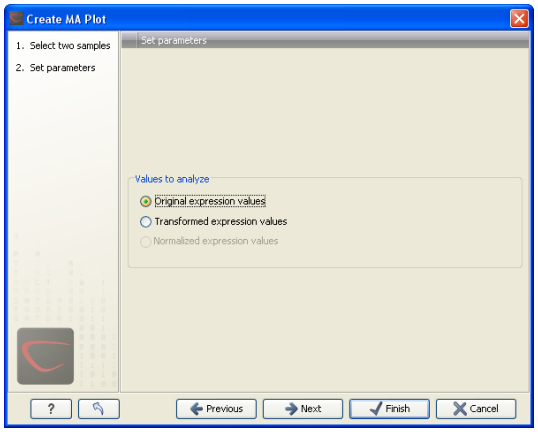


Figure 3.58: *Selcting which values the MA plot should be based on.*

In this dialog, you select the values to be used for creating the MA plot (see section 3.2.1).

Click **Next** if you wish to adjust how to handle the results (see section **??**). If not, click **Finish**.

Viewing MA plots

The resulting plot is shown in a figure 3.59.

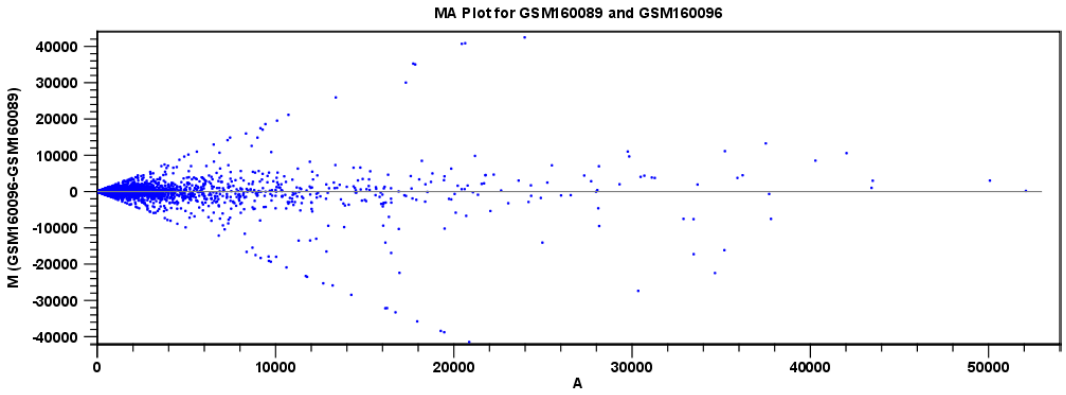


Figure 3.59: *MA plot based on original expression values.*

The X axis shows the mean expression level of a feature on the two samples and the Y axis shows the difference in expression levels for a feature on the two samples. From the plot shown in figure 3.59 it is clear that the variance increases with the mean. With an MA plot like this, you will often choose to transform the expression values (see section 3.2.2).

Figure 3.60 shows the same two samples where the MA plot has been created using log2 transformed values.

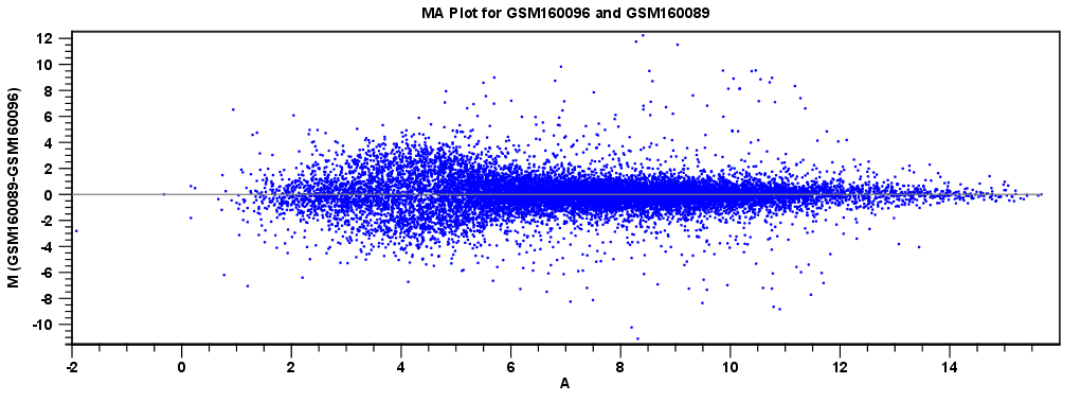


Figure 3.60: *MA plot based on transformed expression values.*

The much more symmetric and even spread indicates that the dependance of the variance on the mean is not as strong as it was before transformation.

In the **Side Panel** to the left, there is a number of options to adjust the view. Under **Graph preferences**, you can adjust the general properties of the plot.

- **Lock axes.** This will always show the axes even though the plot is zoomed to a detailed level.
- **Frame.** Shows a frame around the graph.

- **Show legends.** Shows the data legends.
- **Tick type**. Determine whether tick lines should be shown outside or inside the frame.
  - Outside
  - Inside
- **Tick lines at.** Choosing Major ticks will show a grid behind the graph.
  - None
  - Major ticks
- **Horizontal axis range**. Sets the range of the horizontal axis (x axis). Enter a value in **Min** and **Max**, and press Enter. This will update the view. If you wait a few seconds without pressing Enter, the view will also be updated.
- **Vertical axis range**. Sets the range of the vertical axis (y axis). Enter a value in **Min** and **Max**, and press Enter. This will update the view. If you wait a few seconds without pressing Enter, the view will also be updated.
- **y = 0 axis**. Draws a line where y = 0. Below there are some options to control the appearance of the line:
  - **Line width**
    - Thin
    - Medium
    - Wide
  - **Line type**
    - None
    - Line
    - Long dash
    - Short dash
  - **Line color.** Allows you to choose between many different colors. Click the color box to select a color.
- **Line width**
  - Thin
  - Medium
  - Wide
- **Line type**
  - None
  - Line
  - Long dash
  - Short dash
- **Line color.** Allows you to choose between many different colors. Click the color box to select a color.

Below the general preferences, you find the **Dot properties** preferences, where you can adjust coloring and appearance of the dots:

- **Dot type**
  - None
  - Cross
  - Plus
  - Square
  - Diamond
  - Circle
  - Triangle
  - Reverse triangle
  - Dot
- **Dot color.** Allows you to choose between many different colors. Click the color box to select a color.

Note that if you wish to use the same settings next time you open a scatter plot, you need to save the settings of the **Side Panel** (see section **??**).

3.7.3 Scatter plot

As described in section 3.1.5, an experiment can be viewed as a scatter plot. However, you can also create a "stand-alone" scatter plot of two samples:

Toolbox | **Expression Analysis (** **)** | **General Plots** | **Create Scatter Plot (** **)**

Select two samples (( ) or ( )). Clicking **Next** will display a dialog as shown in figure 3.61.

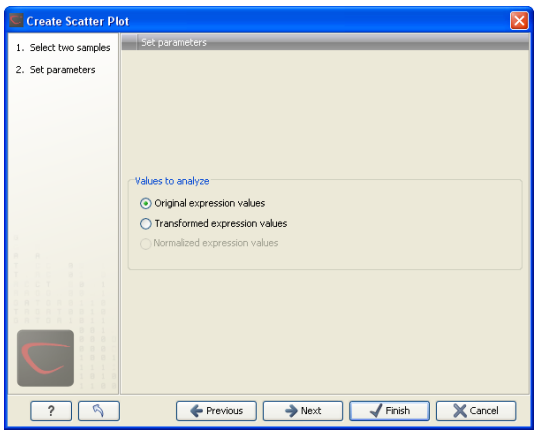


Figure 3.61: *Selcting which values the scatter plot should be based on.*

In this dialog, you select the values to be used for creating the scatter plot (see section 3.2.1).

Click **Next** if you wish to adjust how to handle the results (see section **??**). If not, click **Finish**.

For more information about the scatter plot view and how to interpret it, please see section 3.1.5.

Bibliography

[Akmaev and Wang, 2004] Akmaev, V. R. and Wang, C. J. (2004). Correction of sequence-based artifacts in serial analysis of gene expression. *Bioinformatics*, 20(8):1254–1263.

[Allison et al., 2006] Allison, D., Cui, X., Page, G., and Sabripour, M. (2006). Microarray data analysis: from disarray to consolidation and consensus. *NATURE REVIEWS GENETICS*, 7(1):55.

[Altshuler et al., 2000] Altshuler, D., Pollara, V. J., Cowles, C. R., Etten, W. J. V., Baldwin, J., Linton, L., and Lander, E. S. (2000). An snp map of the human genome generated by reduced representation shotgun sequencing. *Nature*, 407(6803):513–516.

[Baggerly et al., 2003] Baggerly, K., Deng, L., Morris, J., and Aldaz, C. (2003). Differential expression in SAGE: accounting for normal between-library variation. *Bioinformatics*, 19(12):1477–1483.

[Benjamini and Hochberg, 1995] Benjamini, Y. and Hochberg, Y. (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. *JOURNAL-ROYAL STATISTICAL SOCIETY SERIES B*, 57:289–289.

[Bolstad et al., 2003] Bolstad, B., Irizarry, R., Astrand, M., and Speed, T. (2003). A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics*, 19(2):185–193.

[Brockman et al., 2008] Brockman, W., Alvarez, P., Young, S., Garber, M., Giannoukos, G., Lee, W. L., Russ, C., Lander, E. S., Nusbaum, C., and Jaffe, D. B. (2008). Quality scores and snp detection in sequencing-by-synthesis systems. *Genome Res*, 18(5):763–770.

[Creighton et al., 2009] Creighton, C. J., Reid, J. G., and Gunaratne, P. H. (2009). Expression profiling of microrna by deep sequencing. *Brief Bioinform*, 10(5):490–497.

[Cronn et al., 2008] Cronn, R., Liston, A., Parks, M., Gernandt, D. S., Shen, R., and Mockler, T. (2008). Multiplex sequencing of plant chloroplast genomes using solexa sequencing-by-synthesis technology. *Nucleic Acids Res*, 36(19):e122.

[Dudoit et al., 2003] Dudoit, S., Shaffer, J., and Boldrick, J. (2003). Multiple Hypothesis Testing in Microarray Experiments. *STATISTICAL SCIENCE*, 18(1):71–103.

[Eisen et al., 1998] Eisen, M., Spellman, P., Brown, P., and Botstein, D. (1998). Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences*, 95(25):14863–14868.

[Falcon and Gentleman, 2007] Falcon, S. and Gentleman, R. (2007). Using GOstats to test gene lists for GO term association. *Bioinformatics*, 23(2):257.

[Gnerre et al., 2011] Gnerre, S., Maccallum, I., Przybylski, D., Ribeiro, F. J., Burton, J. N., Walker, B. J., Sharpe, T., Hall, G., Shea, T. P., Sykes, S., Berlin, A. M., Aird, D., Costello, M., Daza, R., Williams, L., Nicol, R., Gnirke, A., Nusbaum, C., Lander, E. S., and Jaffe, D. B. (2011). High-quality draft assemblies of mammalian genomes from massively parallel sequence data. *Proceedings of the National Academy of Sciences of the United States of America*, 108(4):1513–8.

[Guo et al., 2006] Guo, L., Lobenhofer, E. K., Wang, C., Shippy, R., Harris, S. C., Zhang, L., Mei, N., Chen, T., Herman, D., Goodsaid, F. M., Hurban, P., Phillips, K. L., Xu, J., Deng, X., Sun, Y. A., Tong, W., Dragan, Y. P., and Shi, L. (2006). Rat toxicogenomic study reveals analytical consistency across microarray platforms. *Nat Biotechnol*, 24(9):1162–1169.

[Ji et al., 2008] Ji, H., Jiang, H., Ma, W., Johnson, D., Myers, R., and Wong, W. (2008). An integrated software system for analyzing ChIP-chip and ChIP-seq data. *Nature Biotechnology*, 26(11):1293–1300.

[Kal et al., 1999] Kal, A. J., van Zonneveld, A. J., Benes, V., van den Berg, M., Koerkamp, M. G., Albermann, K., Strack, N., Ruijter, J. M., Richter, A., Dujon, B., Ansorge, W., and Tabak, H. F. (1999). Dynamics of gene expression revealed by comparison of serial analysis of gene expression transcript profiles from yeast grown on two different carbon sources. *Mol Biol Cell*, 10(6):1859–1872.

[Kaufman and Rousseeuw, 1990] Kaufman, L. and Rousseeuw, P. (1990). Finding groups in data. an introduction to cluster analysis. *Wiley Series in Probability and Mathematical Statistics. Applied Probability and Statistics, New York: Wiley, 1990*.

[Li et al., 2010] Li, R., Zhu, H., Ruan, J., Qian, W., Fang, X., Shi, Z., Li, Y., Li, S., Shan, G., Kristiansen, K., Li, S., Yang, H., Wang, J., and Wang, J. (2010). De novo assembly of human genomes with massively parallel short read sequencing. *Genome research*, 20(2):265–72.

[Lloyd, 1982] Lloyd, S. (1982). Least squares quantization in PCM. *Information Theory, IEEE Transactions on*, 28(2):129–137.

[Maeda et al., 2008] Maeda, N., Nishiyori, H., Nakamura, M., Kawazu, C., Murata, M., Sano, H., Hayashida, K., Fukuda, S., Tagami, M., Hasegawa, A., Murakami, K., Schroder, K., Irvine, K., Hume, D., Hayashizaki, Y., Carninci, P., and Suzuki, H. (2008). Development of a dna barcode tagging method for monitoring dynamic changes in gene expression by using an ultra high-throughput sequencer. *Biotechniques*, 45(1):95–97.

[Meyer et al., 2007] Meyer, M., Stenzel, U., Myles, S., Prüfer, K., and Hofreiter, M. (2007). Targeted high-throughput sequencing of tagged nucleic acid samples. *Nucleic Acids Res*, 35(15):e97.

[Morin et al., 2008] Morin, R. D., O'Connor, M. D., Griffith, M., Kuchenbauer, F., Delaney, A., Prabhu, A.-L., Zhao, Y., McDonald, H., Zeng, T., Hirst, M., Eaves, C. J., and Marra, M. A. (2008). Application of massively parallel sequencing to microrna profiling and discovery in human embryonic stem cells. *Genome Res*, 18(4):610–621.

[Mortazavi et al., 2008] Mortazavi, A., Williams, B. A., McCue, K., Schaeffer, L., and Wold, B. (2008). Mapping and quantifying mammalian transcriptomes by rna-seq. *Nat Methods*, 5(7):621–628.

[Nielsen, 2007] Nielsen, K. L., editor (2007). *Serial Analysis of Gene Expression (SAGE): Methods and Protocols*, volume 387 of *Methods in Molecular Biology*. Humana Press.

[Parkhomchuk et al., 2009] Parkhomchuk, D., Borodina, T., Amstislavskiy, V., Banaru, M., Hallen, L., Krobitsch, S., Lehrach, H., and Soldatov, A. (2009). Transcriptome analysis by strand-specific sequencing of complementary dna. *Nucleic Acids Res*, 37(18):e123.

[Smith and Waterman, 1981] Smith, T. F. and Waterman, M. S. (1981). Identification of common molecular subsequences. *J Mol Biol*, 147(1):195–197.

[Stark et al., 2010] Stark, M. S., Tyagi, S., Nancarrow, D. J., Boyle, G. M., Cook, A. L., Whiteman, D. C., Parsons, P. G., Schmidt, C., Sturm, R. A., and Hayward, N. K. (2010). Characterization of the melanoma mirnaome by deep sequencing. *PLoS One*, 5(3):e9685.

[Sturges, 1926] Sturges, H. A. (1926). The choice of a class interval. *Journal of the American Statistical Association*, 21:65–66.

['t Hoen et al., 2008] 't Hoen, P. A. C., Ariyurek, Y., Thygesen, H. H., Vreugdenhil, E., Vossen, R. H. A. M., de Menezes, R. X., Boer, J. M., van Ommen, G.-J. B., and den Dunnen, J. T. (2008). Deep sequencing-based expression analysis shows major advances in robustness, resolution and inter-lab portability over five microarray platforms. *Nucleic Acids Res*, 36(21):e141.

[Tian et al., 2005] Tian, L., Greenberg, S., Kong, S., Altschuler, J., Kohane, I., and Park, P. (2005). Discovering statistically significant pathways in expression profiling studies. *Proceedings of the National Academy of Sciences*, 102(38):13544–13549.

[Tusher et al., 2001] Tusher, V. G., Tibshirani, R., and Chu, G. (2001). Significance analysis of microarrays applied to the ionizing radiation response. *Proc Natl Acad Sci U S A*, 98(9):5116–5121.

[Wyman et al., 2009] Wyman, S. K., Parkin, R. K., Mitchell, P. S., Fritz, B. R., O'Briant, K., Godwin, A. K., Urban, N., Drescher, C. W., Knudsen, B. S., and Tewari, M. (2009). Repertoire of microrna s in epithelial ovarian cancer as determined by next generation sequencing of small rna cdna libraries. *PLoS One*, 4(4):e5311.

[Zerbino and Birney, 2008] Zerbino, D. R. and Birney, E. (2008). Velvet: algorithms for de novo short read assembly using de Bruijn graphs. *Genome Res*, 18(5):821–829.

[Zerbino et al., 2009] Zerbino, D. R., McEwen, G. K., Margulies, E. H., and Birney, E. (2009). Pebble and rock band: heuristic resolution of repeats and scaffolding in the velvet short-read de novo assembler. *PloS one*, 4(12):e8407.

Part I

Index

Index

mapping
    extract from selection, 90

Adapter trimmming, 38
Affymetrix arrays, 157
Annotate tag experiment, 136
Annotation level, 163
Annotation tests, 200
    Gene set enrichment analysis (GSEA), 203
    GSEA, 203
    Hypergeometric test, 200
Annotations
    add to experiment, 165
    expression analysis, 165
Array platforms, 157
Assemble
    de novo, 46
    report, 74
    to reference sequence, 60

BAM format, 22
BED, import of, 24
Bibliography, 215
BLAST
    contig, 90
    sequencing data, assembled, 90
Box plot, 174

CASAVA1.8, paired data, 12
ChIP sequencing, 107
Chromatin immunoprecipitation, *see* ChIP sequencing
Cluster linkage
    Average linkage, 179
    Complete linkage, 179
    Single linkage, 179
Color space
    Digital gene expression, 118
    RNA sequencing, 118
    tag profiling, 129
Complete Genomics data, 21
Consensus sequence
    extract, 75
    open, 75
Consensus sequence, extract, 89
Consensus sequence, open, 68
Contig
    BLAST, 90
Count
    small RNAs, 140
    tag profiling, 129
Coverage, definition of, 70
Create virtual tag list, 132
csfasta, file format, 15

De novo, assembly, 46
de-multiplexing, 26
Digital gene expression(DGE), 114
    tag-based, 128
DIP
    detect, 101
DIP detection, 101
Directional RNA-Seq, 118
Distance measure, 178

ELAND, import of, 24
Epigenomics, ChIP sequencing, 107
Experiment
    set up, 158
Experiment, 157
Expression analysis, 157
Extract
    part of a mapping, 90
Extract and count small RNAs, 140
Extract and count tags, 129
Extract consensus sequences, from mapping table, 75

FASTQ, file format, 11
Feature clustering, 194
    K-means clustering, 198
    K-medoids clustering, 198
Feature, for expression analysis, 157
File name, sort sequences based on, 27

Gapped/ungapped alignment, 63
Gene expression, 157
Gene expression, sequencing-based, 114
Gene expression, sequencing-by tag, 128
GOstats, *see* Hypergeometric tests on annotations
Groups, define, 158

Heat map
  clustering of features, 196
  clustering of samples, 180
Hierarchical clustering
  of features, 194
  of samples, 177
Histogram, 207
  Distributions, 207
Hypergeometric tests on annotations, 200

Import
  High-throughput sequencing data, 8
  Next-Generation Sequencing data, 8
  NGS data, 8

K-means clustering, 198
K-medoids clsutering, 198

Linker trimming, 38

MA plot, 209
Map
  to coding regions, 60
Map reads to reference
  masking, 60
  select reference sequences, 60
Mapping
  report, 69
  short reads, 63
Mapping reads to a reference sequence, 60
Mapping table, 75
Mappings
  merge, 92
Mask, reference sequence, 60
Match weight, 154
Merge mapping results, 92
Microarray analysis, 157
Microarray platforms, 157
microRNA analysis, 139
Mixed data, 92
mRNA sequencing
  by tag, 128
Multi-group experiment, 158
Multiple testing
  Benjamini-Hochberg corrected p-values, 190
  Benjamini-Hochberg FDR, 190
  Bonferroni, 190
  Correction of p-values, 190
  FDR, 190
Multiplexing, 26
  by name, 27

N50, 69
Non-coding RNA analysis, 139
Non-perfect matches, 84
Non-specific matches, 67, 84
Normalization, 171
  Quantile normlization, 171
  Scaling, 171

Open
  consensus sequence, 68

Paired data, 8, 17, 20, 22
Paired distance graph, 84
Paired reads
  combined with single reads, 92
Paired samples, expression analysis, 158
Paired status, 22
Partitioning around medoids (PAM), *see* K-medoids clustering
PCA, 182
Peak finding, ChIP sequencing, 107
Principal component analysis, 182
  Scree plot, 185

QC, 174
QSEQ,file format, 11
Quality control
  MA plot, 209
Quality of trace, 36
Quality score of trace, 36

Read mapping, 60
Reference assembly, 60
References, 215
Repeat masking, 60
Report
  of assembly, 74
RNA-Seq analysis, 114
RPKM,definition, 128

SAGE

tag-based mRNA sequencing, 128
SageScreen, tag profiling by, 129
SAM format, 22
Sample, for expression analysis, 157
SCARF, file format, 11
Scatter plot, 212
Scree plot, 185
Short reads, mapping, 63
Single paired reads, 84
Small RNA analysis, 139
Small RNAs
extract and count, 140
trim, 140
SNP
detect, 92
SNP detection, 92
Sort sequences by name, 27
Statistical analysis, 186
ANOVA, 186
Corrected of p-values, 190
Paired t-test, 186
Repeated measures ANOVA, 186
t-test, 186
Volcano plot, 191
Subcontig, extract part of a mapping, 90

Tag profiling, 128
annotate tag experiment, 136
create virtual tag list, 132
Tags
extract and count, 129
Trace data
quality, 36
Transcriptome analysis, 157
Transcriptome sequencing, 114
tag-based, 128
Transcriptomics, 114
tag-based, 128
Transformation, 171
Trim, 36
small RNAs, 140
Two-color arrays, 157
Two-group experiment, 158

UniVec, trimming, 36

Vector contamination, find automatically, 36
Virtual tag list
create, 132
how to annotate, 136
Volcano plot, 191