

CodonCode Aligner User Manual

Table of Contents

<u>About CodonCode Aligner.....</u>	<u>1</u>
<u>System Requirements.....</u>	<u>1</u>
<u>Note for Mac OS X 10.3 Users.....</u>	<u>2</u>
<u>Licenses.....</u>	<u>2</u>
<u>Licenses for CodonCode Aligner.....</u>	<u>3</u>
<u>Demo Mode.....</u>	<u>3</u>
<u>Time-limited Trials.....</u>	<u>3</u>
<u>Single-user Licenses.....</u>	<u>4</u>
<u>Using Phred and Phrap from CodonCode Aligner.....</u>	<u>4</u>
<u>Replacement License Keys.....</u>	<u>5</u>
<u>License Server Licenses.....</u>	<u>5</u>
<u>Firewall ports for License Server use.....</u>	<u>6</u>
<u>CodonCode Aligner Features.....</u>	<u>7</u>
<u>Aligner Windows and Views.....</u>	<u>7</u>
<u>The Main Window.....</u>	<u>7</u>
<u>Project Window.....</u>	<u>8</u>
<u>Base View Window.....</u>	<u>8</u>
<u>Trace Window.....</u>	<u>9</u>
<u>Quality View Window.....</u>	<u>10</u>
<u>Contig View Window.....</u>	<u>10</u>
<u>Regions of Interest.....</u>	<u>11</u>
<u>Take the Quick Tour!.....</u>	<u>11</u>
<u>Quality Values In CodonCode Aligner.....</u>	<u>12</u>
<u>Quality values explained.....</u>	<u>12</u>
<u>Where quality values come from.....</u>	<u>12</u>
<u>Artificial qualities.....</u>	<u>13</u>
<u>Gap qualities.....</u>	<u>13</u>
<u>Viewing qualities.....</u>	<u>13</u>
<u>Additional Reading.....</u>	<u>14</u>
<u>Aligner Projects.....</u>	<u>15</u>
<u>Working with Aligner Projects.....</u>	<u>16</u>
<u>Creating New Projects.....</u>	<u>16</u>
<u>Opening Existing Projects.....</u>	<u>17</u>
<u>Saving Projects.....</u>	<u>17</u>
<u>Closing Projects.....</u>	<u>17</u>
<u>Adding Sample Files to Aligner Projects.....</u>	<u>18</u>
<u>Opening Single Sample Files.....</u>	<u>18</u>
<u>Adding Several Sample Files.....</u>	<u>18</u>
<u>Adding Entire Folders of Sample Files.....</u>	<u>19</u>
<u>Adding Entire Assemblies.....</u>	<u>20</u>
<u>Importing Sequencher Projects (CAF Files).....</u>	<u>20</u>
<u>Importing Phrap Assemblies.....</u>	<u>21</u>

Table of Contents

<u>Adding Sample Files to Aligner Projects</u>	
Tags in Phrap assemblies	22
Compatible File Formats	22
Sample Names	23
Scripting CodonCode Aligner	23
<u>Organizing Samples And Contigs In Folders</u>	26
Creating Folders	26
Moving Samples and Contigs to Folders	26
Renaming Folders	26
Deleting Folders	26
<u>Removing Samples from an Aligner Project</u>	28
<u>Base Calling with PHRED</u>	29
How to call bases with PHRED from Aligner	29
Prerequisites	29
What Aligner does	30
Editing the PHRED parameter file	30
Base calling problems	32
Cannot find base calling program	32
Missing entries in the Phred parameter file	33
Problems reading the Phred parameter file	34
Wrong command line parameters	34
Problem running the workstation version of Phred	35
More about the Phred parameter file	35
About Phred	36
<u>End Clipping</u>	38
End Clipping Parameters	39
End Clipping Algorithms	40
Method 1: Maximizing regions with error rates below a given threshold	40
Method 2: Using separate criteria at the start and the end of the sequence	40
<u>Trimming Vector Sequences</u>	42
<u>Vector Library Files</u>	44
Using UniVec Library Files	44
Using Custom Vector Files	44
<u>Assembly and Alignment</u>	46
<u>Sequence Assembly</u>	48
Before assembling	48
How to assemble	48
Advanced assembly options	48
Assemble from scratch	49
Compare contigs to each other	50

Table of Contents

<u>Sequence Assembly</u>	
<u>Assemble in groups</u>	53
<u>Pre-process: One-step processing</u>	55
<u>Sequence Assembly With Phrap</u>	57
<u>How Aligner assembles using Phrap</u>	58
<u>Things to Note for Phrap Assemblies</u>	59
<u>About Phrap</u>	59
<u>Alignments to a Reference Sequence</u>	61
<u>Adding Samples to Alignments</u>	62
<u>Advanced alignment options</u>	62
<u>Align to reference from scratch</u>	62
<u>Align in groups</u>	63
<u>Pre-process: One-step processing</u>	64
<u>Contigs</u>	66
<u>Unassembling Contigs</u>	67
<u>Aligner Algorithms for Assembly and Alignments</u>	68
<u>Alignment to a Reference Sequence</u>	68
<u>Assembly</u>	68
<u>Consensus Calculation</u>	69
<u>Quality-based Consensus</u>	69
<u>Majority Consensus</u>	70
<u>Using the Reference Sequence as Consensus</u>	71
<u>Local, large gap, and end-to-end alignments</u>	71
<u>Regions of Interest: Features</u>	73
<u>Defining Features</u>	73
<u>Moving to Features</u>	74
<u>View, Print, Export Features</u>	75
<u>Finding Mutations</u>	76
<u>Prerequisites</u>	76
<u>How To Find SNPs</u>	76
<u>Fixing Errors</u>	79
<u>Tags for Finding Mutations</u>	80
<u>Defining the Coding Region</u>	80
<u>Numbering the Coding Sequence</u>	82
<u>Excluding Regions from Analysis</u>	84
<u>How Aligner Finds SNPs</u>	85
<u>Limitations</u>	85
<u>Heterozygous Insertions and Deletions</u>	87
<u>Finding Heterozygous Indels</u>	88
<u>Hetero Indel Scores</u>	89

Table of Contents

<u>Heterozygous Insertions and Deletions</u>	
False Negatives.....	89
False Positives.....	90
Splitting Heterozygous Indels.....	90
Processing Heterozygous Indels.....	90
Limitations.....	93
Processing Pre-Requisites and Limitations.....	93
Interpreting Results.....	93
Acknowledgements.....	94
<u>Editing Samples</u>	95
<u>Windows for Editing Samples</u>	97
<u>Cursor Positioning and Movement</u>	98
Moving to Features, Ambiguities, Mismatches or Edited Bases.....	98
<u>Selecting Bases</u>	99
Selecting from sequence start to current cursor position.....	99
Selecting from current cursor position to the end of a sequence.....	99
Selecting all bases in a sequence.....	99
<u>Changing Bases</u>	100
Automatic edits.....	100
Match consensus.....	101
Call second peaks higher than.....	101
Change ambiguities to single bases.....	101
Change low quality bases to N.....	101
Undo auto edits.....	102
<u>Deleting Bases</u>	103
Using the Keyboard to Delete Bases.....	103
Menus for Deleting Bases.....	103
<u>Deleting Samples</u>	104
<u>Moving Gaps and Samples</u>	105
Moving gaps in contigs.....	105
Moving samples in contigs.....	105
Moving samples to "Unassembled Samples".....	105
<u>Inserting Gaps and Bases</u>	106
Adding Bases at the End of Reads.....	106
<u>Reverse Complementing</u>	107

Table of Contents

<u>Editing Sample Information</u>	108
<u>Changing Sample Names</u>	108
<u>Viewing Chromatogram Information</u>	108
<u>Viewing Sample Tags</u>	110
<u>Confirming Tags</u>	111
<u>Viewing "Local" Tags</u>	111
<u>Adding Tags</u>	112
<u>Saving Edits</u>	113
<u>Undo and Redo</u>	114
<u>Copy and Paste</u>	115
<u>Copy (selected sequence)</u>	115
<u>Paste</u>	115
<u>Editing Contigs</u>	116
<u>Adding Samples to Contigs</u>	117
<u>Duplicating samples</u>	117
<u>Merging Contigs</u>	118
<u>Removing Samples from Contigs</u>	119
<u>Deleting Parts of Contigs</u>	120
<u>Alignment Locations - Start and End</u>	121
<u>Reverse Complementing</u>	122
<u>Splitting Contigs</u>	123
<u>Unassembling Contigs</u>	125
<u>Rountrip Editing</u>	126
<u>How To Roundtrip Edit with CodonCode Aligner</u>	126
<u>Limitations</u>	127
<u>Editing Contig Information</u>	128
<u>Search for Sequences</u>	129
<u>BLAST Searches</u>	130
<u>MegaBLAST</u>	130
<u>Nucleotide (blastn)</u>	130
<u>Translated (blastx)</u>	130
<u>Translated (tblastx)</u>	130

Table of Contents

<u>Exporting</u>	131
<u>Export Project Summary</u>	132
<u>Exporting Samples</u>	134
<u>Single FASTA file</u>	134
<u>Individual FASTA files</u>	134
<u>SCF files</u>	135
<u>Exporting Consensus Sequences</u>	136
<u>Single FASTA file</u>	136
<u>Individual FASTA files</u>	136
<u>Exporting Assemblies</u>	137
<u>ACE Format Exports</u>	137
<u>NEXUS/PAUP Format Exports</u>	138
<u>Phylip Format Exports</u>	138
<u>Other Formats for Exporting Assemblies</u>	139
<u>Exporting Features</u>	140
<u>The Project View Window</u>	142
<u>Selecting Samples and Contigs</u>	142
<u>Menu shortcuts: Buttons and Popup menus</u>	143
<u>Project View Columns And Sorting</u>	143
<u>Status messages</u>	143
<u>Trace View</u>	145
<u>General Information</u>	145
<u>Opening a Trace View</u>	145
<u>Scrolling and Scaling in Trace View</u>	145
<u>Trace Sharpening</u>	146
<u>Colors and Highlighting</u>	147
<u>Automatic Trace Selection</u>	147
<u>Hiding Some Traces</u>	147
<u>Printing Traces</u>	149
<u>Base View Window</u>	150
<u>Quality View Window</u>	151
<u>Contig View Window</u>	152
<u>Contig Overview Panel</u>	152
<u>Navigating using the overview panel</u>	152
<u>Moving Around</u>	153
<u>Contig quality in the overview panel</u>	153
<u>Aligned Bases and Protein Translation</u>	153
<u>Masking Bases Matching the Consensus</u>	154

Table of Contents

<u>Contig View Window</u>	
<u>Sorting Reads in the Contig View</u>	154
<u>Automatic Trace Selection</u>	155
<u>Printing Contigs</u>	155
<u>Feature Window</u>	157
<u>Restriction Map View</u>	158
<u>Single Line Map</u>	158
<u>Multi Line Map</u>	158
<u>Text Map</u>	159
<u>Selecting Fragments</u>	160
<u>Restriction Enzymes in Aligner</u>	161
<u>Closing Windows</u>	162
<u>Preferences and Settings</u>	163
<u>Alignment Preferences</u>	165
<u>Algorithm</u>	166
<u>Minimum Percent Identity</u>	166
<u>Minimum Overlap Length</u>	166
<u>Minimum Alignment Score</u>	166
<u>Maximum Unaligned End Overlap</u>	167
<u>Bandwidth (Maximum Gap Size)</u>	167
<u>Word Length</u>	168
<u>Match scoring</u>	168
<u>Restoring default parameters</u>	168
<u>Assembly Preferences</u>	169
<u>Algorithm</u>	170
<u>Minimum Percent Identity</u>	170
<u>Minimum Overlap Length</u>	170
<u>Minimum Alignment Score</u>	170
<u>Maximum Unaligned End Overlap</u>	171
<u>Bandwidth (Maximum Gap Size)</u>	171
<u>Word Length</u>	172
<u>Maximum Successive Failures</u>	172
<u>Match scoring</u>	172
<u>Restoring default parameters</u>	173
<u>Base Calling Preferences</u>	174
<u>Base Color Preferences</u>	176
<u>Switching color schemes</u>	177
<u>Quality-based 3 color scheme</u>	177

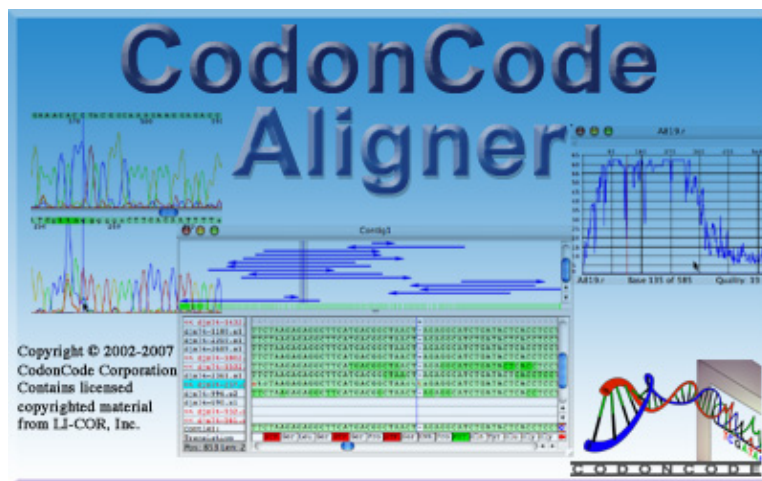
Table of Contents

<u>Continuous, quality-based background colors</u>	179
<u>Base-specific colors</u>	181
<u>Consensus Preferences</u>	184
<u>Double Clicking Preferences</u>	186
<u>End Clipping Preferences</u>	187
<u>Automatically removing short and low-quality sequences after end clipping</u>	188
<u>Feature Preferences</u>	190
<u>Specifying subsets of tags</u>	191
<u>Highlighting Preferences</u>	192
<u>License Server Preferences</u>	194
<u>Memory Preferences</u>	196
<u>Changing memory on OS X</u>	196
<u>Memory on Windows</u>	197
<u>Mutation Detection Preferences</u>	198
<u>Detection sensitivity</u>	198
<u>Marking mutations</u>	199
<u>Finding only homozygous mutations</u>	200
<u>Finding indels</u>	200
<u>Open & Save Preferences</u>	201
<u>Phrap Assembly Preferences</u>	203
<u>Preference Options</u>	205
<u>Printing Preferences</u>	207
<u>Trace View Printing</u>	207
<u>Contig View Printing</u>	208
<u>Protein Translation Preferences</u>	210
<u>Restriction Map Preferences</u>	211
<u>Selecting Enzymes</u>	211
<u>Restriction Map Options</u>	213
<u>Sample Name Preferences</u>	215
<u>Defining sample names</u>	215
<u>Defining delimiters</u>	218

Table of Contents

<u>Startup Preferences</u>	220
<u>The Startup Dialog</u>	220
<u>Toolbar Preferences</u>	222
<u>Vector Trimming Preferences</u>	224
<u>View Preferences</u>	226
<u>Warning Preferences</u>	228
<u>Window Placement Preferences</u>	229
<u>CodonCode Aligner Help by Menu</u>	231
<u>Aligner Menu (on Mac OS X only)</u>	231
<u>File Menu</u>	231
<u>Edit Menu</u>	231
<u>Go Menu</u>	232
<u>Sample Menu</u>	232
<u>Contig Menu</u>	233
<u>View Menu</u>	233
<u>Window Menu</u>	234
<u>Help Menu</u>	234
<u>Memory Requirements</u>	235
<u>How Aligner memory on Windows is set</u>	235
<u>How Aligner memory on Mac OS X is set</u>	236
<u>CodonCode Aligner Release Notes</u>	237
<u>Checking for Aligner Updates</u>	238
<u>Visiting the Aligner Web Site</u>	238

About CodonCode Aligner



Copyright 2002-2007 CodonCode Corporation. All Rights Reserved.
Contains licensed copyrighted material from LI-COR, Inc.

For technical support:

e-mail: support@codoncode.com

WWW: www.codoncode.com

USA: (781) 686-1131

System Requirements

Mac OS X: Requires Mac OS X version 10.3.4 or newer; 256 MB RAM (512 MB or more suggested), and a 400 MHz or faster G3 or better processor. *Please note that OS X 10.2 is not supported anymore; Apple has stopped development of Java for OS X 10.2 several years ago. If you are using OS X 10.2, please upgrade to OS X 10.3 or newer before using CodonCode Aligner.*

Windows: Requires Windows 98, ME, 2000, or XP, 256 MB RAM (512 MB or more suggested), and a 500 MHz or faster Pentium III or better processor.

Other operating systems: Currently, CodonCode Aligner is not available for other operating systems. For Linux and UNIX, many good alternative assembly and contig editing programs are available. For Mac OS 9.x, the required Java version (1.3.1) is not available, and is not expected to ever become available.

Memory requirements: How much memory CodonCode Aligner requires depends on the project size. The default settings should allow you to process projects with several hundred chromatograms. If necessary, you can change increase or reduce Aligner's memory requirements as described in the "[Memory Requirements](#)" section.

Java requirements: CodonCode Aligner is Java program, and requires that the appropriate Java version is installed.

On Windows, the installer will install a Java runtime version (JRE) specifically for Aligner. You should not try to run Aligner with any other Java versions, you'll just run into problems that can easily be avoided.

On Mac OS X, Java 1.4.2 is used (since CodonCode Aligner version 1.2.5).

Note for Mac OS X 10.3 Users

There are two known problems with Java on OS X 10.3 which can cause the installer and/or CodonCode Aligner to crash; however, these problems can easily be fixed.

- **Crashes after upgrading to OS X 10.3.9:** Due to a bug in the OS X 10.3.9 updater, many users saw that Java applications crashed after upgrading to OS X 10.3.9. Apple has released a Java update that fixes this problem. For more information, please visit <http://www.apple.com/support/downloads/javaupdateformacosx1039.html>.
- **Java crashes due to disabled fonts:** On some OS X 10.3 versions, Java will crash randomly if some fonts are disabled. This can easily be fixed by enabling all fonts in the "FontBook" application. For more details, please visit <http://www.codoncode.com/aligner/problems.htm>.

Licenses

After installing, Aligner will start in **Demo** mode. Demo mode allows you to use Aligner for viewing sequence traces, as well as printing and editing unassembled traces. You can also open existing Aligner projects. However, you cannot print, save, and export projects that contain contigs in demo mode.

Demo mode also allows you to try out most of Aligner's functions, including end clipping, assembly, and so on; but again, you will not be able to but not to save, import, export, or print after trying these functions in demo mode.

The first time that you use CodonCode Aligner, you will automatically receive a time-limited trial, which will enable you to use all functions for 30 days. For more information about how to get licenses for **regular** use, please check the "[Licenses for Aligner](#)" section.

Acknowledgements

CodonCode Aligner contains licensed copyrighted material from LI-COR, Inc. The Mac OS X version uses functionality provided by [MRJAdapter](#) (Copyright © Steve Roy) and the "[Quaqua Look and Feel](#)" (Copyright © Werner Randelshofer), and therefore is subject to the terms of the [Quaqua Look and Feel License Agreement](#), a copy of which is provided in the Mac OS X application package directory.

Licenses for CodonCode Aligner

After installing, CodonCode Aligner will start up in [Demo](#) mode. The first time that you use CodonCode Aligner, you will automatically receive a time-limited trial, which will enable you to use all functions for 30 days.

When you are ready to purchase a license, you can choose between [single-user licenses](#) and [license server licenses](#).

All license keys other than the demo key are specific to a given computer, and will not work if transferred to a different computer. Certain changes on your computer, for example installing a new operating system or replacing a system hard disk, may also invalidate your license so that you will need to request a new license key.

Demo Mode

In Demo mode, CodonCode Aligner works as a fully functional **trace viewer and editor**: you can open, view, edit, and print unassembled sequences. You can also save projects that contain only unassembled sequences, unless you have "processed" any of the sequences with Aligner's functions for end clipping, base calling, sequence assembly, and so on (Aligner will warn you before doing anything that will disable saving and printing).

You can also open existing Aligner projects in demo mode, for example the projects in the "Example Files" folder inside the "CodonCode Aligner" folder. However, if an opened project contains any contigs, you will not be able to save, import, export, or print in demo mode.

The demo mode also allows you to test most of Aligner's advanced functions, with the following exceptions:

- Base calling with Phred
- Assembly with Phrap (however, you can assemble using Aligner's own assembly algorithm)
- Finding mutations and heterozygous indels
- Processing of heterozygous insertions or deletions (indels)

If you would like to fully test Aligner with your own data, you can request a license key for a time-limited trial, as explained in the next section.

Time-limited Trials

The first time you use CodonCode Aligner, you will automatically receive a time-limited trial that enables you to use all functions for 30 days. After the initial trial expires, Aligner will revert to Demo mode. You will still be able to open any existing projects you may have, but your ability to save or print will be limited, as described above.

If you need additional time to evaluate CodonCode Aligner, please contact us by email to support (at) codoncode.com. We will need to have your name and email address to send you a trial license (note that the email address cannot be an anonymous email address like xx@yahoo.com).

Single-user Licenses

Single-user license keys that you purchased from CodonCode Corporation allow the use of CodonCode Aligner on a single computer. To enter a single-user license key that you received from CodonCode, start Aligner, and press the "Enter License" button in the startup dialog. If Aligner is already running, select "**License...**" from the "**Help**" menu, and then press the "**Enter New License**" button.

This will display the "Enter New License" dialog:



Enter the license string you received from CodonCode Corporation into the "License String" field, preferably by using copy and paste (*use the "Paste License" button, or the keyboard shortcut for pasting in Aligner - Command-V on OS X or Control-V on Windows*). Then, click the "Apply" button.

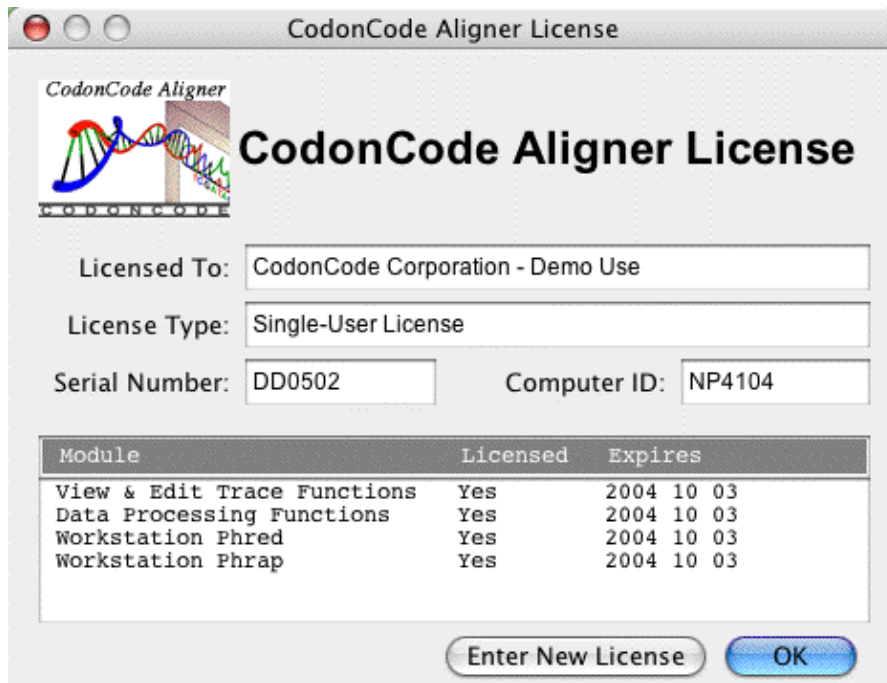
Licenses are bound to a single computer ID. If you try to install the license on different computer, the license will not be valid. A single user license allows you to install Aligner on one computer. If you plan to use your license on one of several computers, you need a [License Server License](#).

Using Phred and Phrap from CodonCode Aligner

CodonCode Aligner allows you to use the base calling program Phred and the assembly program Phrap to base call and assemble your sequence data. Phred and Phrap are separate programs, and distributed by CodonCode Corporation through a distribution agreement with the University of Washington. Users at academic and non-profit institutions may use Phred and Phrap free of charge; users at for-profit institutions have to purchase separate licenses for Phred and Phrap.

When installing CodonCode Aligner, special "workstation" versions of Phred and Phrap are also installed. These workstation versions are identical to the regular Phred-Phrap versions, except that:

- the workstation versions of Phred and Phrap can only be run from CodonCode Aligner (not from the command line or scripts)
- they require a valid license for the "Workstation Phred" and "Workstation Phrap" modules, as shown in the license dialog:



Note that the "Licensed" column in the example above says "Yes" for all modules, indicating that you can use the workstation versions of Phred and Phrap that were installed with Aligner. Time-limited trial license will generally allow the (time-limited) use of workstation Phred and Phrap. Licenses purchased by academic users will also generally include licenses for workstation Phred and Phrap, since Phred and Phrap can be obtained by academic users free of charge. Licenses purchased by users at for-profit institutions, however, will not include permissions to use workstation Phred and Phrap, unless a separate license fee was paid.

If you have your own licensed copies of Phred and Phrap, you can use these from Aligner instead of the workstation versions (except when running in demo mode) by changing the [base calling preferences](#) or the [Phrap assembly preferences](#).

Replacement License Keys

Under certain circumstances, you may need a **replacement license key**. This can happen if you replace your computer, if you install a new operating system, replace the hard drive, and similar circumstances. If you receive a replacement license, you must make sure that the original license is not used anymore. If Aligner detects that a replacement license and the original license are both used, Aligner will invalidate one or both of the licenses, and you may have to request another replacement license.

In general, you should not move the folder where CodonCode Aligner is installed after the installation. If you move the folder to a different location on the same computer, you may have to re-enter the license information.

License Server Licenses

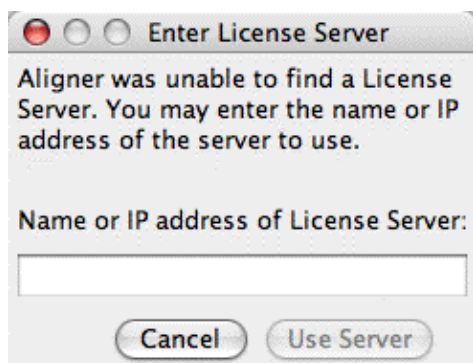
License Server licenses for CodonCode Aligner allow you to install CodonCode Aligner on an unlimited number of computers, but limit concurrent use to the number of licenses you purchased. Using License Server licenses requires installation of a separate program ("Aligner License Server") on a computer that acts as the license server.

If Aligner License Server is available for your laboratory or department, you can choose to use the Aligner License Server in the dialog that CodonCode Aligner displays when starting up:



If you click the "**Use License Server**" button, Aligner will attempt to locate a License Server for CodonCode Aligner on your local network. If a License Server is found, and a license is available, it will be checked out, and you can start using CodonCode Aligner. The license key will be returned so that someone else can use it when you quit CodonCode Aligner.

If CodonCode Aligner cannot find a License Server, the following dialog will be displayed:



If you know the computer name or the IP address for the license server computer, you can enter it here (if you do not know the name or IP address, ask your local system administrator).

Firewall ports for License Server use

If Aligner cannot connect to the License Server, it may be because Aligner License Server is not running, or because a firewall is blocking the network traffic between your computer and the license server computer.

Please make sure that the License Server is running, and to allow communication on the following UDP **and** TCP ports: 16030, 16031, 32156, 32157, 54643, and 54644.

CodonCode Aligner Features

CodonCode Aligner is a versatile, powerful and easy-to-use DNA and RNA sequence assembler, aligner, and editor. Aligner imports a wide variety of sample file types into "projects". Aligner projects are collections of sample files and their resulting alignments or contigs.

Aligner works best with sequences that have chromatograms and quality scores, for example from SCF files generated with PHRED or ABI files that were base called with the KB base caller. However, you can also import text sequences without chromatograms, and trace sequences without quality scores (for example older ABI files, or ABI files analysed with the ABI base caller).

Aligner protects your original sequence data by making copies of the original sample data when files are imported. When you make edits in Aligner, you change only the copied data files.

Aligner Windows and Views

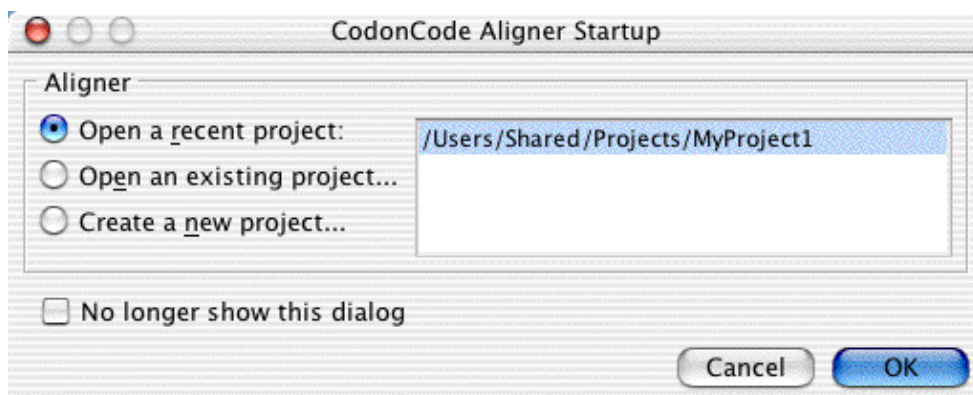
This section describes the most important Aligner windows (or "views") briefly. For more detailed descriptions, follow the links for each window, or read the sub-topics in the "Aligner Windows" help section.

The Main Window

On **Windows**, opening CodonCode Aligner will open the main application window ("root window"), which contains the application menu and all other windows opened by Aligner.

Since CodonCode Aligner follows the common user interface standards for Windows and Mac OS X, there are some user interface differences between the Windows and OS X versions.

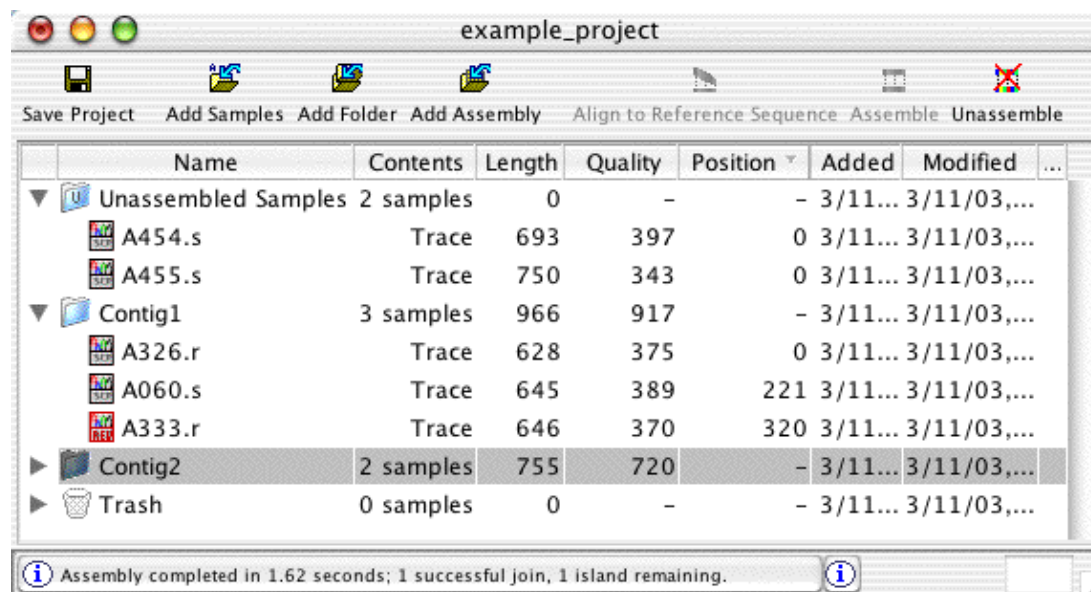
On **Mac OS X**, there is no such "root window". Instead, the main menu bar at the top of the screen is used. When you first start Aligner on Mac OS X, a startup dialog is displayed that allows you to open an existing project or create a new project:



After selecting or creating a project, the project window described in the next section will be shown. *(If you hit cancel, or previously selected the "No longer show this dialog" checkbox, you will not see this window - the application will have only a menu bar without any windows. On Windows, the same dialog is also shown upon startup).*

Project Window

To do anything in CodonCode Aligner, you must create a project, or open a previously created project. This will create a "Project Window", where the contents of the project are shown, similar to the way the Finder or Explorer show contents of disks and folders:

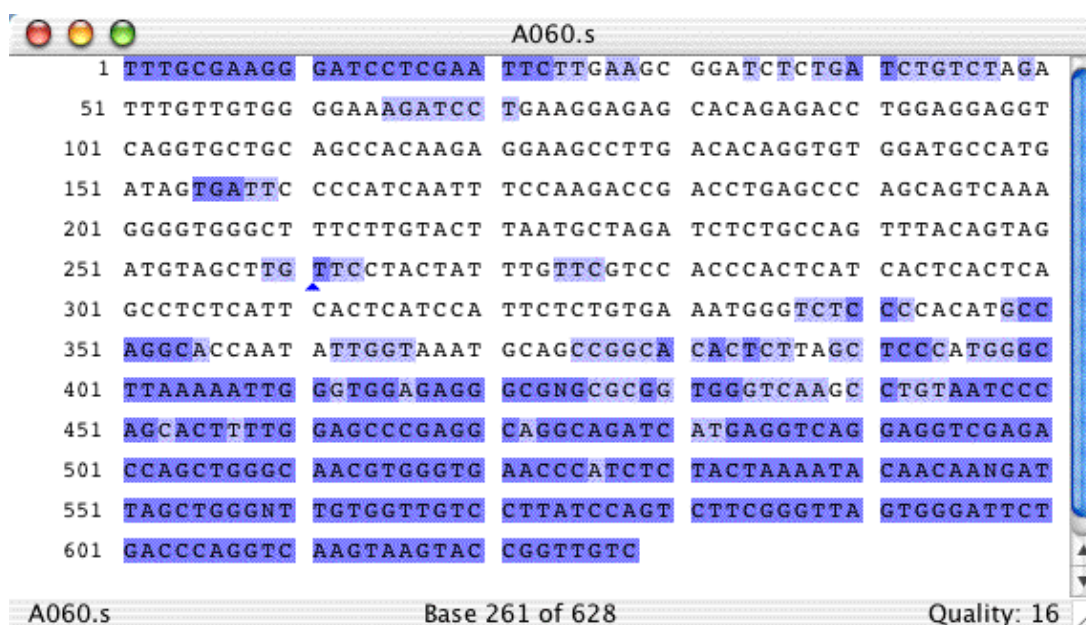


In the project window, you can add samples, entire folder of samples, or Phrap assemblies using the buttons on the top, or the corresponding items in the "File" menu. Newly imported samples will be in the "Unassembled Samples" folder. Any contigs that are created or imported from other assemblies are shown as separate folders.

In the project view, you can also select samples and contigs by clicking or shift-clicking, and then view the bases, traces, or contigs, assemble the samples, and so on.

Base View Window

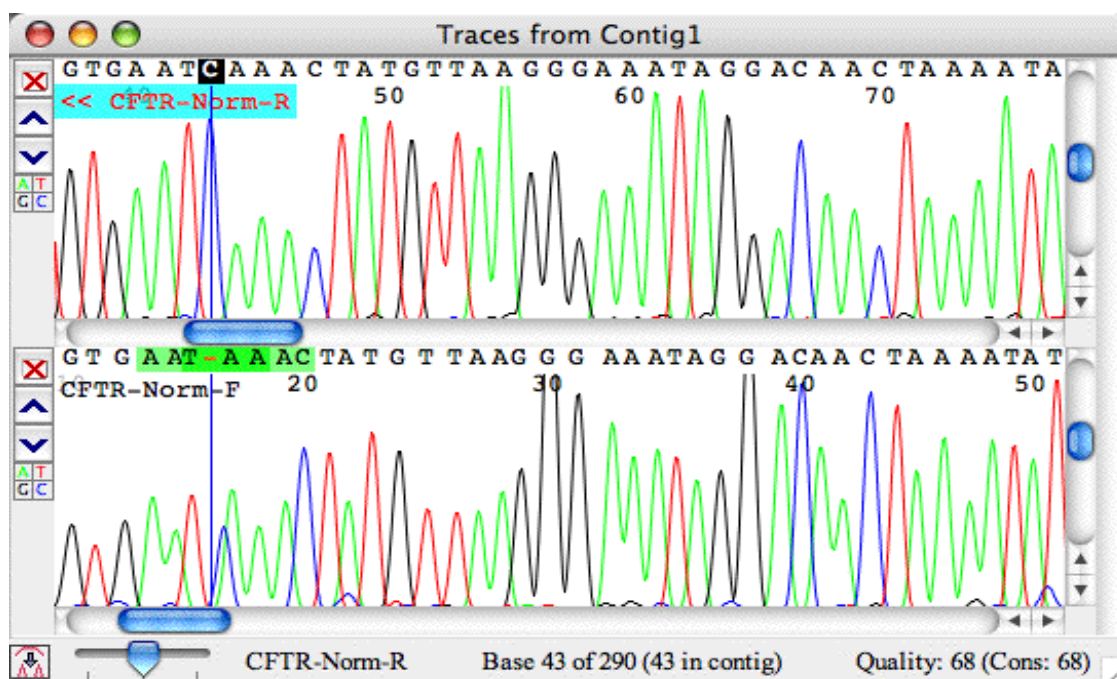
The base view window shows all the bases for a sequence, as shown below:



As in the other windows that show bases, the quality scores for each base can be shown as a colored background. In the example above, base calls with qualities below 20 ("low quality bases", accuracy below 99%) are shown with a dark blue background, base calls with quality scores between 20 and 30 on light blue background, and base calls with qualities above 30 ("very high quality bases", accuracy above 99.9%) on a white background. You can change the colors and thresholds in the "Preferences" dialog.

Trace Window

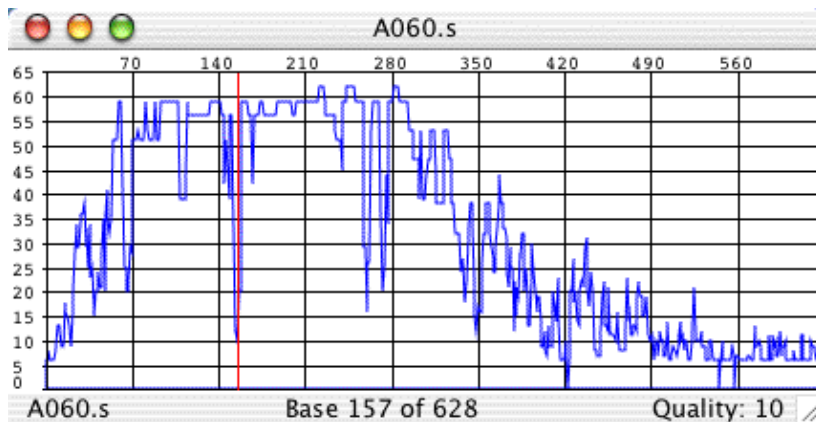
The trace window allows you to view and edit sequences that have chromatogram data, for example ABI and SCF files.



Each contig has its own trace window, which can show the traces for one or more sequences. If the traces for all sequences cannot be shown in the window at the same time, you can use the scroll bar on the right to scroll through the different sequences.

Quality View Window

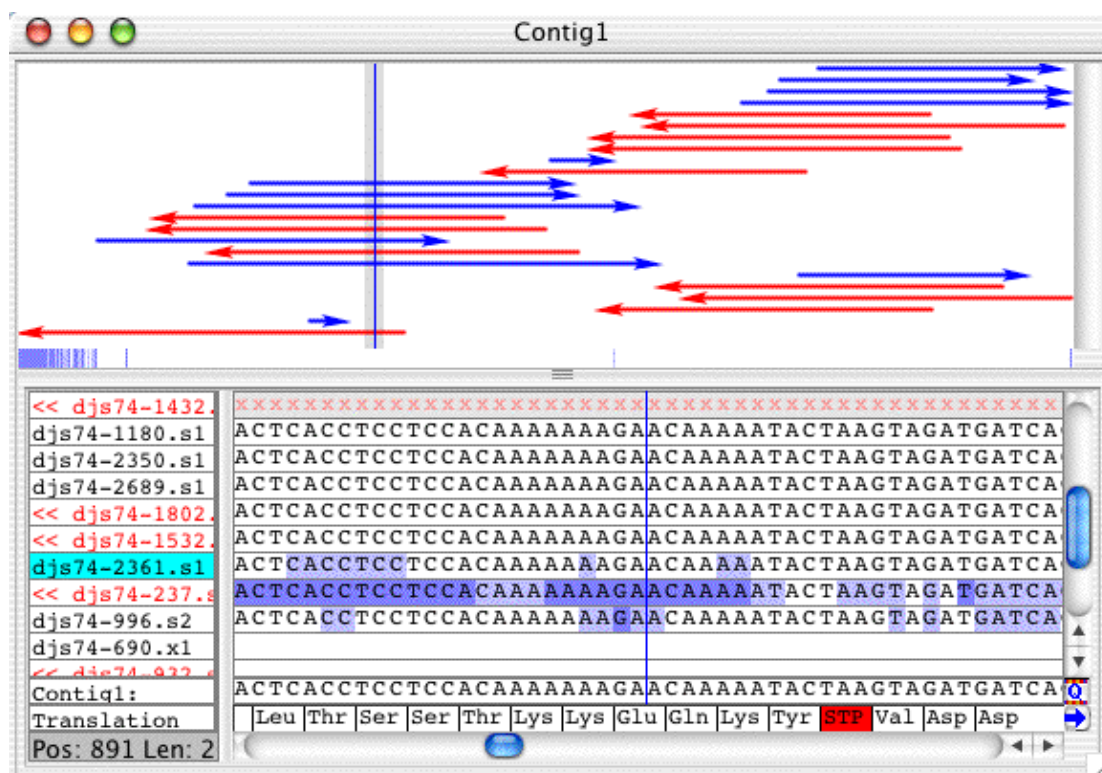
For sequences that have qualities, for example Phred-generated SCF files, you can get a quick overview of the quality by selecting the sample in the project window, and then choosing "Qualities" in the "View" menu. This will open up a window like this one:



The picture shown above is pretty typical: a bit of low-quality sequence at the beginning, followed by rapidly increasing sequence quality and then several hundred bases where the quality is mostly very high, except for some short problem regions. If you open a trace view window for the same sequence, then clicking somewhere in the quality view will reposition the trace to the same region. This allows you to quickly check out problem regions in the sequence - try it out!

Contig View Window

The contig window shows detailed information about contigs - a graphical overview about how the samples are aligned, the aligned bases, and the protein translation.



You can edit sequences directly in the contig window, or select samples and then open trace, base, or quality windows for the sample. You can also set if and how the protein translation is shown using the "Protein translation" sub menu in the "View" menu.

Regions of Interest

One concept in CodonCode Aligner that deserves a closer look is the ability to define "regions of interest", more often called "features". Aligner lets you define what is of interest to you, for example low-quality consensus regions, discrepancies, mutation tags, and so on. You can then use this definition to get an overview of all the features in a contig or the entire project in a [feature view window](#), or to quickly [navigate](#) from feature to feature, or to [export features](#) to text files.

For more information about "region of interest" features, check the [feature help page](#).

Take the Quick Tour!

To quickly learn about the most important features in CodonCode Aligner, we suggest that you take the "Quick Tour", available online at <http://www.codoncode.com/aligner/quicktour/>. It takes about an hour to complete, but will probably save you quite a few hours, for example by showing you some very useful features that you otherwise might miss.

You can access the Quick Tour by selecting the "**Quick Tour**" menu item from the "**Help**" menu in CodonCode Aligner. This will open a new web browser window, and point it to the start of the quick tour. Alternatively, you can look at the Quick Tour in Adobe PDF format, which was installed as "QuickTour.pdf" in the directory where you installed CodonCode Aligner (typically "/Applications/CodonCode Aligner/" on OS X, and "C:\Program Files\CodonCode Aligner\" on Windows; you may not see the ".pdf" extension, depending on your system settings).

Quality Values In CodonCode Aligner

CodonCode Aligner was designed to make optimal use of base-specific quality values, both for pre-processing and for sequence assembly. Accurate, base-specific quality values that are linked to base calling accuracy were first introduced by the program [PHRED](#) (that's why they are often called "Phred qualities" or "Phred scores"). While you can do many things in Aligner even if your sequences do not have quality scores, we strongly suggest that you use sequences with quality scores.

For sequence traces that do not have qualities, Aligner allows you to call bases with PHRED to get PHRED base calls and quality scores, as described in the "[Base Calling with Phred](#)" section .

The following sections briefly [explain](#) the relation between quality scores and error probabilities; which files Aligner can [read](#) quality scores from; and how to [view](#) quality scores in Aligner.

Quality values explained

Phred quality values represent the probability of error for each base call. The quality value q assigned to each base call is defined to be

$$q = -10 \times \log_{10}(p)$$

where p is the estimated error probability for the base call. As indicated below, a base call with a 1 in 100 probability of being incorrect will be assigned a quality value of 20.

<u>Quality Value</u>	<u>Error Probability</u>
10	1 in 10
20	1 in 100
30	1 in 1000
40	1 in 10000
50	1 in 100000

For additional details about how Phred quality scores, you can read the articles listed [below](#).

Where quality values come from

The first widely used program to produce highly accurate quality scores for each base call was the program [PHRED](#). Since then, a number of other base calling programs have been introduced and modified to also produce Phred-like quality scores. The two most common file types with quality scores are:

- SCF ("Standard Chromatogram Format") files produced by **PHRED**.
- ABI chromatogram files (.ab1 files) **processed with the KB base caller** (older ABI chromatogram files, and files produced with the ABI base caller, do not contain quality scores)

Chromatograms SCF format may or may not contain quality values, depending on which program generated the SCF files. SCF files generated by PHRED contain quality scores, but some other programs (like some versions of Sequencher) do not write quality information into SCF files.

When importing Phrap assemblies that were generated with quality scores, Aligner will read the quality information from the corresponding "PHD" files, regardless of whether or not the chromatogram files for the assembly were in ABI- or SCF format.

When reading sequences from FASTA files, Aligner will also read quality scores from ".qual" files - text files in FASTA-like format that contain quality scores. The name of the ".qual" file must be the same as the name of the FASTA file, with ".qual" appended (for example "seq.fasta" and "seq.fasta.qual"). The quality file must be in the same directory as the FASTA file.

Note that not all ABI chromatogram files have quality scores. Only ABI chromatograms processed with the KB base caller have quality scores; ABI chromatograms processed with the older ABI Basecaller do not have quality scores. For questions about the KB base caller, please read the "[KB Base Caller Frequently Asked Questions](#)" brochure.

When importing chromatogram sequences that do not have quality scores, Aligner will assign artificial quality scores to all bases, as described in the next section. You can identify sequences without quality scores quickly in the project view - the "Quality" column shows "0" as the number of high-quality bases for such sequences. To assign sequence quality scores to chromatograms without qualities, you can [base call](#) the samples with PHRED (which may require a separate license for PHRED).

Artificial qualities

There may be times where you need to use sequences that do not have quality values, for example Genbank sequences. When importing such sequences into CodonCode Aligner, Aligner will assign artificial qualities to all bases in these sequences. The default value for artificial qualities is 15, but you can change this in the "[Open & save](#)" preferences.

If you import sequences into Aligner which have quality scores that seem artificial, for example because all scores are 0 or -1, or all scores have the same value, Aligner will also assign artificial quality scores.

Gap qualities

Quality values are associated with base calls, so gaps do not really have quality values. However, since it is often convenient to have some quality value even for gaps, Aligner does assign qualities to gaps; the assigned quality is the average of the two bases on either side (with gaps characters and edited bases being ignored). For bases at the start or end of the sequence that have only one neighboring base, the quality value of this base is used.

Viewing qualities

In CodonCode Aligner, you can "see" quality values in a variety of ways:

- to get an **overview** of the quality in a sample, select the sample in the project view, and then choose "**Qualities**" in the "**View**" menu
- to get information about the quality **at any specific base**, you can click on the base in the base view window or the trace view window; the quality of the selected base is shown at the bottom of the window
- you can also choose to have the **background behind the bases** in the base view, trace view, and contig view windows indicate the quality, using either a continuous scale or a three-color scheme ("the

good, the bad, and the ugly" :-); you can set this in the "Base colors" preference panel.

Additional Reading

R. Durbin and S. Dear, 1998 "Base Qualities Help Sequencing Software" *Genome Research*. 8: 161-162. ([Available](http://www.genome.org/content/vol8/issue3/) online at <http://www.genome.org/content/vol8/issue3/>)

B. Ewing B, L. Hillier L, M.C. Wendl and P. Green. "Base-calling of automated sequencer traces using Phred. I. Accuracy assessment." *Genome Research*. 8: 175-85. ([Available](http://www.genome.org/content/vol8/issue3/) online at <http://www.genome.org/content/vol8/issue3/>).

B. Ewing and P. Green, 1998 "Base-Calling of Automated Sequencer Traces Using Phred. II. Error Probabilities." *Genome Research*. 8: 186-198. ([Available](http://www.genome.org/content/vol8/issue3/) online at <http://www.genome.org/content/vol8/issue3/>)

P. Richterich, 1998. "Estimation of Errors in "Raw" DNA Sequences: A Validation Study" *Genome Research*. 8: 251-259. ([Available](http://www.genome.org/content/vol8/issue3/) online at <http://www.genome.org/content/vol8/issue3/>)

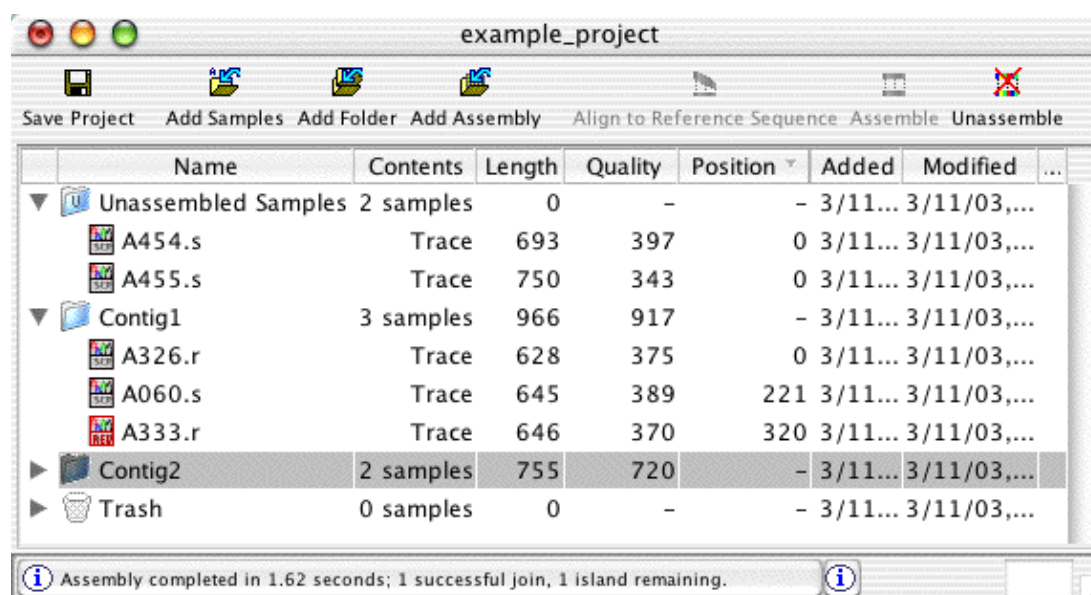
Aligner Projects

An Aligner project contains all the information for a project you are working on - the samples you added, any contigs you may have assembled, and so on. Each project is contained in its own directory, which is created by CodonCode Aligner when you first save a project.

After an Aligner project is created, sample files can be added. Original files are never modified, only copies are added to the project directory.

When alignments or assemblies are performed, the resulting consensus is stored in the project directory. Collectively, all the files in a project directory are referred to as an Aligner project.

Only one project can be open at any point in time. The contents of a project are shown in the **project window**:



In the project window, samples are treated like files, and contigs are treated like folders (directories) in the Finder on MacOS or the Explorer on Windows. Two other folders also exist in every project: the "Unassembled Samples" folder, and the "Trash". Any sample that has been added to a project, but not yet assembled or deleted, gets added to the "Unassembled Samples" folder. If you don't want a sample in your project anymore, you can move it to the Trash by selecting the sample in the project window, and then selecting "Move To Trash" in the "Edit" menu. If you later change your mind, you can move samples from the trash back to the "Unassembled Samples" folder.

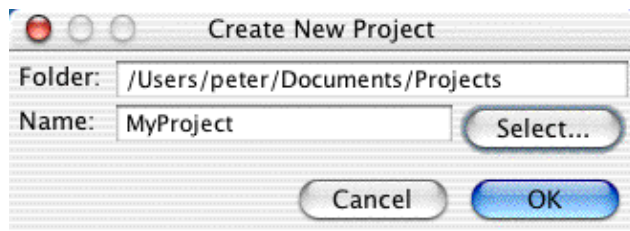
Within the project view, you can also use **drag and drop** to move samples and even contigs around. You can drag samples from the "Unassembled Samples" folder to the "Trash", and the other way round; you can drag samples in contigs or entire contigs to the trash or to the "Unassembled Samples" folder; and you can add samples to existing contigs by dragging them from the "Unassembled Samples" folder onto the contig you want to add them to (of course, the sample aligns with the contig; for more details, check the "[Assembly and Alignment](#)" help section).

Working with Aligner Projects

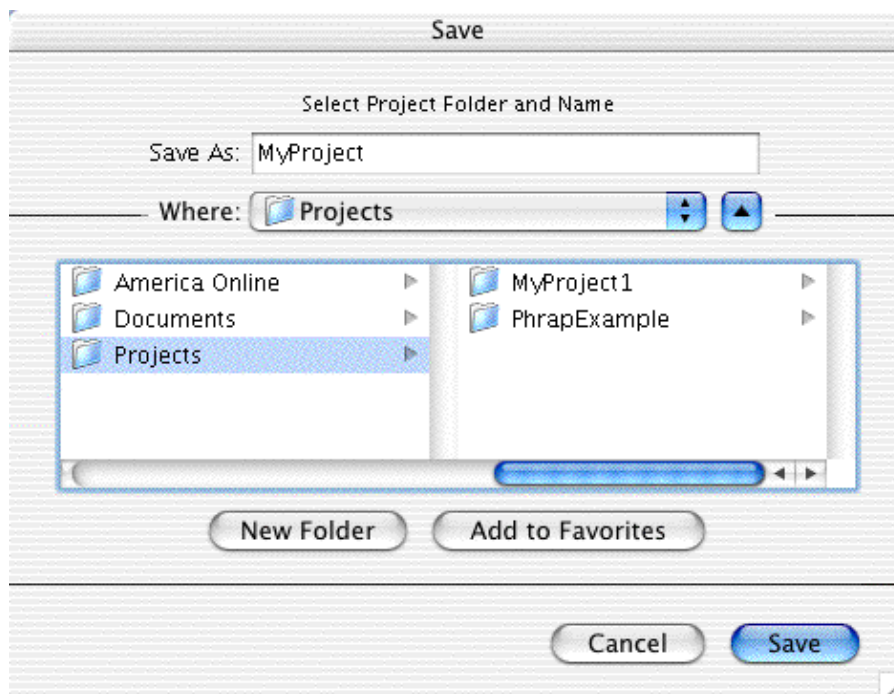
Creating New Projects

To create a new project, choose **"New Project"** from Aligner's **"File"** menu. Aligner will create a new project, and show a project view window for it. The project will be empty, except for the "Trash" and the "Unassembled Samples" folders. You can specify where a project will be saved when you first save the new project.

If you prefer to set the project path when creating a new project, you can change the [Open & Save Preferences](#) accordingly. Now, when creating a new project, Aligner will show the the "Create New Project" dialog:



The simplest way to set the project name and location is to click on the **"Select"** button. This will bring up a standard **"Save As.."** dialog:



After clicking "Ok", an new project window will be opened, which contains no samples, only the "Unassembled Samples" and the "Trash" folders.

Opening Existing Projects

Choose **"Open..."** from the **"File"** menu to open a project you have already saved. Navigate to the directory your project is in, and select the project file. The project file has a ".proj" extension (which you may not see, depending on your system settings). This will read all the contents of the project as it was last saved, and show the contents in a new project window.

Alternatively, you can also use the **"Open Recent"** submenu in the **"File"** menu. This will show the projects you worked with recently. If a project is not longer available, for example because you deleted or renamed it's folder, then it will be grayed out in the **"Open Recent"** submenu.

Saving Projects

Choose **"Save Project"** from the **File** menu to save your project (if the project is a new project for which you have not set a name and location before, you will be prompted to do so now).

To save a copy of an existing project under a different name and/or at a different location, choose **"Save Project As..."** from the **File** menu. The **"Save Project As..."** command will always create a new project directory, and put copies of all files into this new directory (*if you try to create a project with the same name as an existing folder, you will get a warning dialog*).

It's a good idea to save projects on a regular basis, and especially before major processing steps like end clipping, vector trimming, or assembly.

If you are using CodonCode Aligner in **Demo** mode, you can only save projects that (a) do not contain any contigs, and (b) where you have not used any of the advanced Aligner functions like end clipping, sequence assembly, and so on.

Closing Projects

Choose **"Close Project"** from the **File** menu to close the currently open project and all its windows. Aligner will first check if you made any changes to the project since the last time the project was saved. If any changes were made, Aligner will ask you if you want to save the changes before closing the project. If you answer "Don't save", all changes you have made since the last save will be lost.

Adding Sample Files to Aligner Projects

CodonCode Aligner can import sequences from a variety of different [file formats](#), including chromatogram files in ABI and SCF format and text files in FASTA, Genbank, NBRF/PIR, PHD, or plain text format. Files in FASTA, Genbank, or NBRF/PIR format can contain multiple sequences per file.

You can add sequences to Aligner projects by **dragging and dropping** files onto Aligner project views, or by using one of several **"Open"** and **"Import"** options in the **"File"** menu:

1. by [opening single sample files](#)
2. by [adding samples from several files](#)
3. by [adding entire folders of samples](#)
4. by [adding an entire assembly or project](#).

Aligner will always create copies of the samples you import. Keep in mind, though, that these copies will be created only when you save the project, so save on a regular basis! *For files that have pointers to associated files, Aligner will also try to locate and import the related files (for example, PHD files and FASTA files may point to a chromatogram file in ABI or SCF format that contains the trace data.*

Opening Single Sample Files

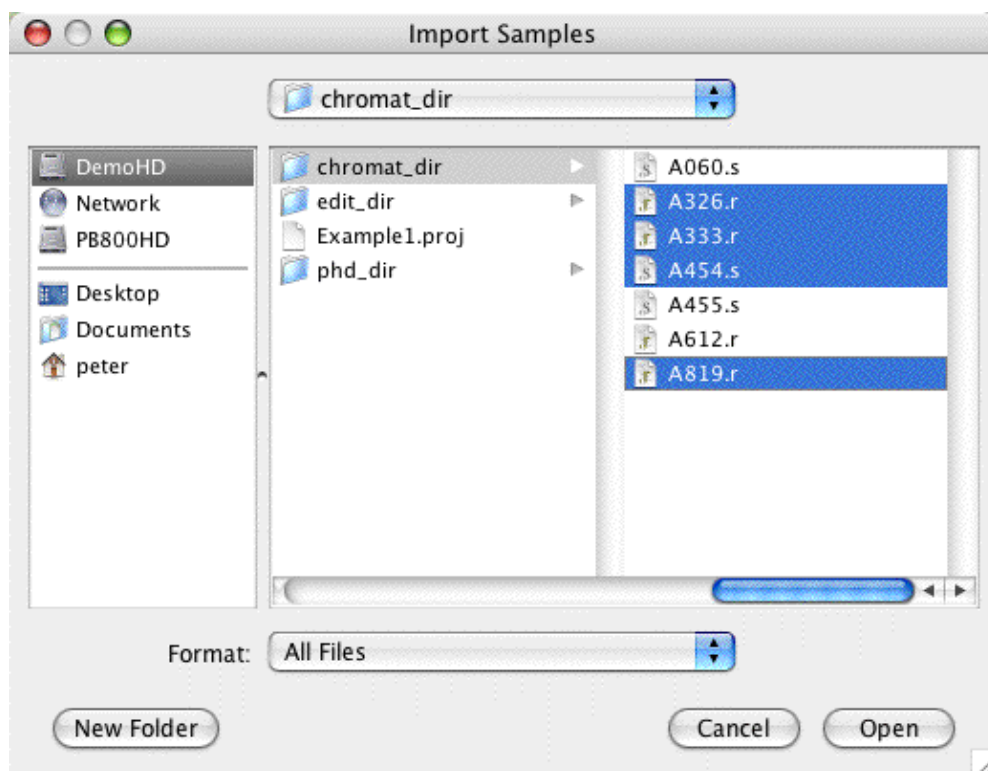
To add a single sample to a project, select **"Open..."** from the **"File"** menu. This will show the standard "Open" file dialog. Select the file that you want to add to your project, and click "OK". The file will be read, and the sample(s) in it will be added to the "Unassembled Samples" folder. Next, Aligner will open one or more views for the sample(s) you just added, based on your settings for which views to open when double-clicking on a sample in the [Double Clicking Preferences](#).

Adding Several Sample Files

To add samples from a few files to a project, select **"Import" -> "Add Samples..."** from the **"File"** menu. This will show an "Open" file dialog. Select the files that you want to add to your project, and click "OK".

- To select several files in a row, click on the first file, and then keep the "shift" key pressed while clicking on the last file.
- To select several files that are not right next to each other, keep the "control" key (OS X: the "command" key) pressed while clicking on files.

The screen shot below shows an example of the "Import Samples" dialog with four selected files:



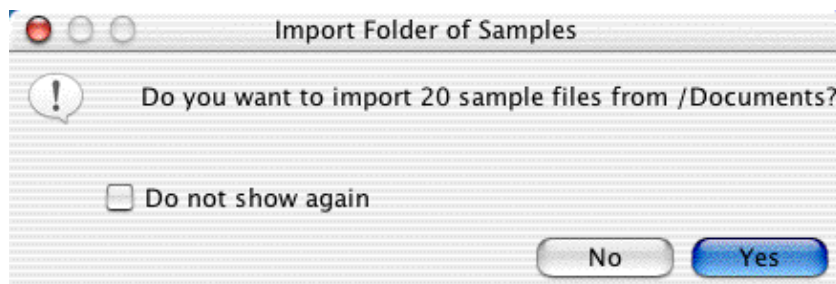
When you are done making your selection, press the "Open" button. CodonCode Aligner will read the files you selected, and add the samples to the "Unassembled Samples" folder.

If any problems were encountered during import, Aligner will tell you so by showing a dialog box. For example, if you already have a sequence with the same name as an imported sequence in the project, you will be given a choice to ignore the duplicated sequences, or to automatically rename them.

Adding Entire Folders of Sample Files

To quickly add many sequence files to a project, it is easiest if you first gather all the files into one folder, and then import the entire folder. To import sequences from all files in a directory, select **"Import" -> "Add Folder..."** from the **"File"** menu. This will show the standard "Open" file dialog. Navigate to the directory you want to import, select any file in this directory, and then click "OK".

Aligner will count the files in the directory, and then ask you if you really want to import all the files:



If you click "Yes", Aligner will proceed to import sequences from all files in the directory. This may take a while if the directory contains many files! All imported sequences will be added to the "Unassembled

Samples" folder.

Note: Aligner will try to be reasonable when importing all files in a folder, and not import some files that you probably did not really want to import. For example, Aligner not import hidden files (like files where the name starts with a period); also, when importing an ABI runfolder, Aligner will import the .abi (or .ab1) files, but ignore all .abi.seq files if the corresponding .abi file was present.

Adding Entire Assemblies

To add an entire assembly to your project, select "**Import**" -> "**Add Assembly...**" from the "**File**" menu. This will show the standard "Open" file dialog. Next, select the assembly file, which can be:

- an Aligner project file (*the file name must end in ".proj"*)
- a "CAF" (Common Assembly Format) file produced by Sequencher (*the file name must end in ".caf"*)
- assemblies (ACE files) generated by Phrap (*the file name must end in ".ace", ".ace.1", ".ace.2", etc.*)

The assembly files **must have the correct extension** - either .proj, .caf, or .ace; ACE files can also have a number appended, for example ".ace.1" (*Note that, depending on your system settings, you may not be able to see the extension*).

After you selected the assembly file, CodonCode Aligner will read the file, and any associated files like chromatogram files. If any of the samples or contigs in the imported assembly have the same names as files that are already in your project, Aligner will give you a choice to either rename the files, or to cancel the import.

Importing Sequencher Projects (CAF Files)

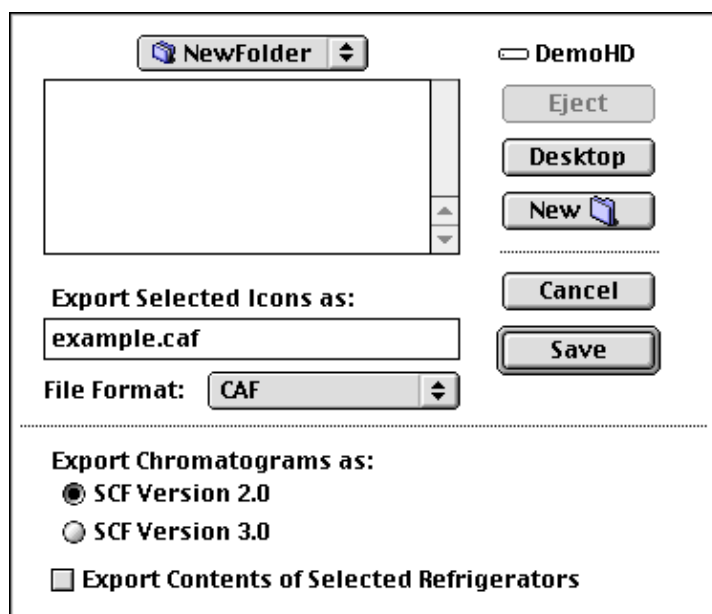
Many current and former users of the assembly program Sequencher have expressed the desire to import existing Sequencher projects into CodonCode Aligner. While CodonCode Aligner cannot read Sequencher project files directly, it is easy to export Sequencher projects in a file format that CodonCode Aligner can read, the "CAF" file format (for "Common Assembly Format").

Exporting Sequencher Projects as CAF Files

To import a Sequencher project into CodonCode Aligner, you must first export the project from Sequencher in CAF format, as follows:

1. Open the project in Sequencher
2. Select the reads and contigs you want to export (use "Select All" from the "Select" menu to export the entire project)
3. Go to the "**File**" menu, and select "**Export => Selection As Subproject**". This will open a "Save" dialog.
4. Click on the "File Format:" pulldown menu, and select "CAF".
5. Use the "New folder" button or icon to create a new folder to export your project to.
6. Choose a name for the exported file. **Make sure that the name ends with ".caf"**.

Here is an example of what the dialog looks like with Sequencher 4.1 on Mac OS 9:



When you click "Save", Sequencer will export your selection to a text file in CAF format. In addition, Sequencer will create chromatogram files in SCF format for all reads that have traces (this is why we suggested to create a new folder in step 5 above).

You can now quit Sequencer, and import the CAF file you just exported into any open CodonCode Aligner project, using "Import => Add Assembly". Please keep a few things in mind when importing CAF files:

- Sequences in old Sequencer projects often do not have base-specific qualities, but many functions in CodonCode Aligner either require quality scores, or work better with quality scores. For unassembled samples that have chromatogram traces, you can use "[Call Bases](#)" to run Phred on the imported samples, and thereby get quality scores.
- Consensus sequences generated in Sequencer are not quality-based, and therefore likely to contain more errors and ambiguities than consensus sequences build in CodonCode Aligner. You can choose to have CodonCode Aligner re-build the consensus sequences when importing in the [consensus method preferences](#).
- CodonCode Aligner's support for CAF files is currently limited to CAF files exported by Sequencer; Aligner will generally not be able to read CAF files produced by other programs.

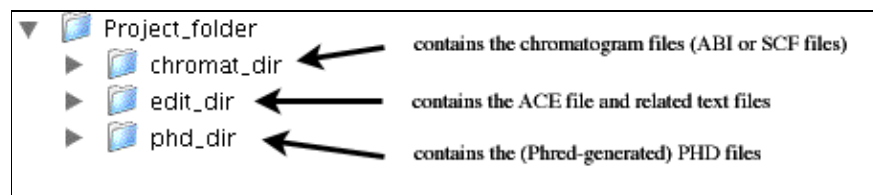
If you encounter any problems when importing CAF files into CodonCode Aligner, please send a description of your problem together with the CAF file to support@codoncode.com.

Importing Phrap Assemblies

To import a Phrap assembly into CodonCode Aligner, select have the Phrap-generated ACE file in the file dialog you see after choosing **Import => Add Assembly** from the **File** menu (you must have a project open for this option to be available). In addition to the ACE file, you should also have the PHD and SCF files for the assemble, located in exactly the same directory structure as Consed would expect, ad described below.

When importing Phrap assemblies, all the contigs that Phrap has created will remain intact, and detail information like unaligned ends and tags will be reserved. Aligner will read the assembly information from the .ace file, and try to get additional information from the other files typically used with Phrap - the .PHD files in the "sister" folder "phd_dir", and the chromatogram files in the "sister" folder "chromat_dir". An

overview of the typical directory structure used with Phrap assemblies is as follows:



Tags in Phrap assemblies

Phrap assemblies can contain tags, which are especially important when you are using PolyPhred. Aligner will import and preserve tags, and display tags based on your highlighting preferences. You can add notes to existing tags, and edit such notes. To do so, open a contig view, select a base that contains a tag, and then right-click (OS X: control-click) on it to display the popup menu. The first item in the popup menu will be "Display Tag..." (but only if you clicked on a base with a tag).

You can also [view](#) all tags for a sample in the ["Sample information" dialog](#), and [add tags](#).

Compatible File Formats

The following sample file types can be read by Aligner:

- **SCF files** - Standard Chromatogram Format, commonly available and generated by Phred, LI-COR software, and many other programs. This is the preferred format, especially if the SCF files were generated by Phred. It is also the format used by Aligner to store chromatogram information. (*Note: Sometimes, SCF files have the file extension .scf; this will be hidden in Windows even if you have your preferences set to display file extensions. However, Aligner will recognize this file extension.*)
- **ABI chromatogram files** - Files generated by systems from Applied Biosystems software. Typically have the extension .abi or .ab1 (but Phred-generated SCF files will typically have the same name and extension!). *Note that older ABI files may not contain base-specific quality scores; after importing ABI files that do not have quality scores, you should first [base call](#) to get quality scores.*
- **FASTA files** - text files with one or more DNA sequences, where each new sequence starts with a line that starts with a '>' sign, followed by the name of the sequence. If the FASTA header line also contains the name of corresponding PHD or SCF files, Aligner will try to read these files, too.
- **GenBank files** - text files exported from Genbank, and many other programs. Genbank files are often used for the reference sequence in re-sequencing and mutation detection projects. When reading Genbank files, Aligner will read the "CDS" (coding sequence) annotation, including the "/codon_start" tag (this tag indicates the number of the base after the CDS start which start the first whole codon, and can either be 1, 2, or 3). *In Aligner, the "CDS" region is shown as a "codingSequence" tags, and the "/codon_start" annotation as a "codonStart" tag.*
- **EMBL files** - text files exported from ENSEMBL and many other programs in EMBL format. Like Genbank files, EMBL files can be used for the reference sequence in re-sequencing and mutation detection projects. Aligner will read the "CDS" (coding sequence) annotation, similar to the way described for Genbank files.
- **NBRF/PIR files** - text files in NBRF/PIR format, a simple format that can contain multiple sequences per file. Sequences in NBRF/PIR format may contain gaps in the sequences as well as at the start and end of sequences. When importing sequences from NBRF/PIR files, CodonCode Aligner will check if the imported sequences contain gaps. If the sequences contain gaps, and are all of the same length,

Aligner will check the sequence names in the project to see if you are re-importing contigs that have been edited with an external sequence editor like MacClade, and offer the option to update existing contigs. For more information, read the ["Roundtrip Editing" help page](#).

- **PHD files** - text files that contain information about base calls and quality scores. PHD files are typically created by PHRED, but a number of other programs (including Aligner) can also write PHD files.
PHD files typically contain the name of the chromatogram file they refer to, which Aligner will then also read to get the trace data.
- **Plain text** format - text files that have only DNA sequence without any annotation. Aligner will read all base letters ("ACGTUacgtu") and IUPAC ambiguity symbols, and ignore any spaces, line endings, and numbers. Plain text files cannot contain any "binary" characters; when creating text files in programs like Microsoft Word, make sure to save the files as "Plain text", not in native formats (like "Microsoft Word Document (.doc)").

In addition to these sample files, CodonCode Aligner can read sequence assembly files in ".ace" format produced by Phrap or Consed, and assemblies in CAF format produced by Sequencher, as described above.

Sample Names

When reading chromatogram files, Aligner will use the name of the file as a sample name. Since version 1.4.0, sample names can contain spaces, accented characters, and so on. However, if your sample have "funny" characters like accents or umlaute, keep this in mind:

- The names may not be drawn correctly in all views (due to minor bugs in Java)
- On Windows, importing a folder of samples with special characters in their names may not work (Aligner may report that it cannot find some of the samples). Similar, adding samples to a project by drag and drop may not work if the sample name contains unusual characters. However, you should always be able to add samples using "Import -> Add Samples".
- When exporting samples, Aligner may replace unusual characters in the sample names with underscores.

Some files contain the sample name in the file (e.g., FASTA files). When this is the case, the name specified in the file is used as the Aligner sample name. When a file contains multiple samples, the sample names **must** be specified in the file.

Within a project, **sample names must be unique**. If a sample name in a file already exists in the project, the new sample is renamed so that the sample name is unique. Samples are renamed by appending an underscore and a unique number (e.g., name_1).

To **rename samples**, you can use the ["Sample Information" dialog](#). Alternatively, you can first select the sample in the project view, and then clicking on the sample name again (similar to the way you would rename files in Finder windows on OS X, or Explorer windows on Windows).

Scripting CodonCode Aligner

Some functions in CodonCode Aligner can be executed from scripts - text files that contain a collection of commands, and have the extension ".alscpt". Scripts can be opened using "Open..." from the "File" menu, or by drag & drop.

CodonCode Aligner User Manual

Some commands allow you to specify files or folders, for example for importing. Please note that any file or folder names that contains spaces must be surrounded with quotes. If the file or folder names contain quotes, change the quotes in the names to double quotes. For example, a file `"/Users/Shared/Name with "quote"` should be written as `"/Users/Shared/Name with ""quote"`.

In scripts, the same logic as when using CodonCode Aligner normally applies. Specifically, to do something like end clipping or assembling, you must first select the samples or contigs to work with. Here is an overview of the available commands:

Command	Parameter	Comments
<code>newProject</code>	<i>none</i>	Creates a new project
<code>openProject</code>	full path to project file (e.g. <code>"/Users/Shared/My Project/My Project.proj"</code>)	Opens an existing project
<code>importFolder</code>	full path to folder	Imports a folder of samples
<code>importProject</code>	full path to project file	Imports a project
<code>importSample</code>	full path to sample file	Imports a sample
<code>closeProject</code>	<i>none</i>	Closes a project
<code>saveProject</code>	<i>none</i>	Saves a project
<code>saveProjectAs</code>	full path to project file (e.g. <code>"/Users/Shared/My Project/My Project.proj"</code>)	Save the project at the given location
<code>select</code>	sample or contig name	Selects the sample with the given name
<code>selectAll</code>	<i>none</i>	Selects everything in the current view
<code>selectAllUnassembledSamples</code>	<i>none</i>	Selects all unassembled samples
<code>selectAllContigs</code>	<i>none</i>	Selects all contigs
<code>unselect</code>	sample or contig name	Unselects the sample or contig with the given name
<code>clipEnds</code>	<i>none</i>	End clips the currently selected samples
<code>makeRefSeq</code>	sample or contig name	Makes the named sample a reference sequence
<code>findHeteroIndels</code>	<i>none</i>	Finds heterozygous indels in the selected samples
<code>assemble</code>	<i>none</i>	Assembles the selected samples and contigs
<code>assembleFromScratch</code>	<i>none</i>	Assembles from scratch
<code>assembleByName</code>	<i>none</i>	Assembles in groups (by name)
<code>compareContigs</code>	Clustal or muscle (optional)	Compares contigs to each other, using Clustal or muscle (if specified), otherwise the current default
<code>assembleUsingPhrap</code>	<i>none</i>	Assembles contigs with PHRAP
<code>alignToReference</code>	<i>none</i>	Aligns selected samples to a reference sequence
<code>alignByName</code>	<i>none</i>	

CodonCode Aligner User Manual

		Aligns to reference sequence by name (in groups)
findMutations	<i>none</i>	Finds mutations in the selected contigs

Please note that scripting is intended only for users who have some experience in scripting applications, and are able to sort through problems that are likely to appear when using scripts.

Organizing Samples And Contigs In Folders

Aligner projects always contain the "Unassembled Samples" folder and the "Trash" folder. However, if your projects contain a lot of unassembled sequences or many contigs, you may want to organize the sequences a bit better. CodonCode Aligner allows you to create folders in projects, to which you can move samples and contigs by drag and drop or by using the "Move To..." menu item.

Creating Folders

To create a new folder for organizing unassembled samples, select "**New Folder**" from the "**File**" menu. This will show a dialog where you can name the new folder.

Folders created this way can hold unassembled samples, contigs, or a mix of samples and contigs.

Moving Samples and Contigs to Folders

You can move samples or contigs to and from folders in several ways:

- By **drag and drop**:
Select the folders or samples you want to move in the project view, and drag & drop them onto the target folder.
- Using "**Move to...**" in the "**Edit**" menu:
Select the samples you want to move, then choose to "**Move to...**" from the "**Edit**" menu. In the dialog that appears, select the target folder that you want to move the samples to, then press "OK".
- Using "**Move to...**" in the popup menu:
Select the sample or samples you want to move in the project view, and then right-click (OS X: control-click) on one of the samples. From the popup menu, select "Move to..."

To move contigs from folders back to the same level as Unassembled Samples and the Trash, select the contig, choose "**Move to...**" from the popup or "**Edit**" menu, and then select "(top level)". You can also drag and drop contigs in folders onto empty space in the project view to see the same menu.

You cannot move unassembled samples to the top level, and the "(top level)" option will not be available if your selection contains unassembled samples.

Renaming Folders

To rename a folder you created, click on the folder in the project view, wait a little bit, and click on it again. You will see when the name becomes editable (it may take a second or so). When you are done editing the name, press enter, or click on any other item in the project view.

If you change your mind about renaming the folder, you can press "escape" while you are still editing, or select "**Undo**" from the "**Edit**" menu right after you changes the name.

Deleting Folders

To delete a folder that you created, first remove all of its contents, then:

CodonCode Aligner User Manual

- Select the folder in the project view, then
- Choose "**Delete Folder**" from the "**File**" menu.

Only empty folders can be deleted. If a folder still contains samples or contigs, the "**Delete Folder**" menu item will be disabled. To delete a folder that contains samples, first move the samples to the trash, and then delete the folder.

You cannot delete the "Unassembled Samples" folder or the "Trash" folder.

Removing Samples from an Aligner Project

To remove samples from an open project:

- Go to the project window
- Select the sample(s) you want to remove (*use shift-click to make continuous selections; use control-click (Windows) or command-click (OS X) to make discontinuous selections*)
- Select **"Move To Trash"** from the **"Edit"** menu (*or display the popup menu by right-clicking (OS X: control-clicking) , and choose **"Move To Trash"** from the pop-up menu*)

You can also select samples *in* contigs and move them to the trash. This will remove the samples from the contig. If removing of a sample would leave a gap in a contig, thereby splitting a contig in two pieces, you will not be able to remove the sample (we plan to add the functionality to automatically split a contig in later versions...)

You can also move entire contigs to the trash this way. This will move the contig and all samples in it to the trash.

Moving samples to the trash does **not** delete them from your project. If you change your mind about a sample in the trash, you can select it, and then choose **"Move To Unassembled Samples"** (or "Move To...") from the **"Edit"** menu.

To permanently delete samples from you project, first move the samples to the trash, and then choose **"Empty Trash..."** from the **"File"** menu.

Base Calling with PHRED

Aligner works best if sequences have quality scores - but what if you have sequence traces that do not have quality scores, for example ABI files? Well, you can simply run Phred to re-do the base calling and assign quality scores from within Aligner (Phred was developed by Phil Green and Brent Ewing at the University of Washington; [more about PHRED below](#)).

How to call bases with PHRED from Aligner

To call bases for sequence traces with PHRED:

1. Select the samples in the project view.
The sequences must be unassembled; you can also select the entire "Unassembled Samples" folder. If you want to call bases for samples that are already in a contig, you must first [unassemble](#) the contig.
2. Choose "Call Bases" from the "Sample" menu.

Depending on the number of samples you have chosen, base calling may take a while (up to a few seconds for each sample), and you will need to wait until it's done before you can do anything else.

Prerequisites

To use base calling in Aligner, your sequences must have traces; any sequences from text files that do not have traces will be ignored.

Furthermore, base calling with PHRED requires that you have **PHRED installed** on the computer that you are running Aligner on. This can either be the workstation version of PHRED that is included with Aligner, or your own copy of PHRED.

Using the workstation version of PHRED

When you install CodonCode Aligner, a workstation version of PHRED is installed in the "Phred-Phrap" folder inside the "CodonCode Aligner" folder. This workstation version can only be run from CodonCode Aligner, but is otherwise identical to PHRED. To use it, you must either have a valid trial license for Aligner, or a full (purchased) license that includes license permissions for the workstation version of PHRED.

CodonCode Aligner customers at academic and non-profit institutions can use the workstation version of PHRED free of charge; customers at for-profit institutions need to purchase a separate license to for the workstation version of PHRED.

Using your own copy of PHRED

If you already have a licensed copy of PHRED installed on the computer that you are using Aligner on, you can use this copy for base calling. First, though, you will need to tell Aligner where your copy of PHRED is installed; you can do this in the [base calling preferences](#).

Academic users who want to use their own copy of PHRED can obtain the source code for PHRED free of charge directly from the authors (with certain usage restrictions); please check www.phrap.org for details.

Users at for-profit organizations need to purchase a license for PHRED. Please check www.phrap.com for information about purchasing PHRED licenses from CodonCode.

What Aligner does

In short, all that happens is that the base calls in the samples will be replaced with PHRED base calls, and PHRED's base-specific quality scores. But if you really want to know the details, here is what happens when you start base calling in Aligner:

1. Aligner will first create two temporary directories. These are created in the system's default temporary directory, for example in /tmp/ on OS X; the names of the directories will start with "bscl".
2. Aligner will write SCF files for each selected sample into the first folder, from which PHRED will read the data.
3. Aligner checks if all necessary entries are present in the Phred parameter file to process the traces you selected. *If any entries need to be added, Aligner will show a warning dialog, and then allow you to add the required entries using the Phred parameter file edit dialog as described [below](#).*
4. Next, Aligner will start PHRED, using the path to the program that you can define in the [base calling preferences](#) (the default is "workstation_phred" in the "Phred-Phrap" folder inside the "CodonCode Aligner" folder). Aligner will also pass the location of the Phred parameter file, as defined in the [base calling preferences](#), to PHRED. Aligner will use the -id and -cd options to tell PHRED where the input files are, and where to write the result files to.
5. Aligner will wait for PHRED to finish. If you are base calling several hundred traces, this may be a good time for a coffee break. If it's just a few traces, it should take only a few seconds.
6. When Phred is done, Aligner will check to see if PHRED produced the expected number of result files; if not, Aligner will try to look at the PHRED output to find out what the problem was (see [below](#)). Aligner will also write the progress and error messages generated by PHRED into two files in the Aligner folder; the file names are "Basecalling_errors.txt" for error messages and warning, and "Basecalling_output.txt" for regular output (typically, the names of the files processed).
7. Aligner will then rename the samples that were base called, move them to the trash, and read the base calls from the new SCF files that PHRED has just created.
8. Finally, Aligner will delete all the temporary files it and PHRED have created. The temporary directories will be deleted when you exit Aligner.

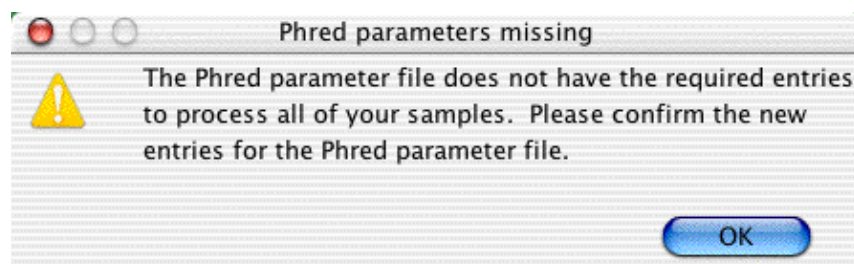
If anything went wrong, Aligner will leave the samples as they were; you will first need to correct the problem, and try to do the base calling again. The next section describes some of the most common problems and their solutions.

Editing the PHRED parameter file

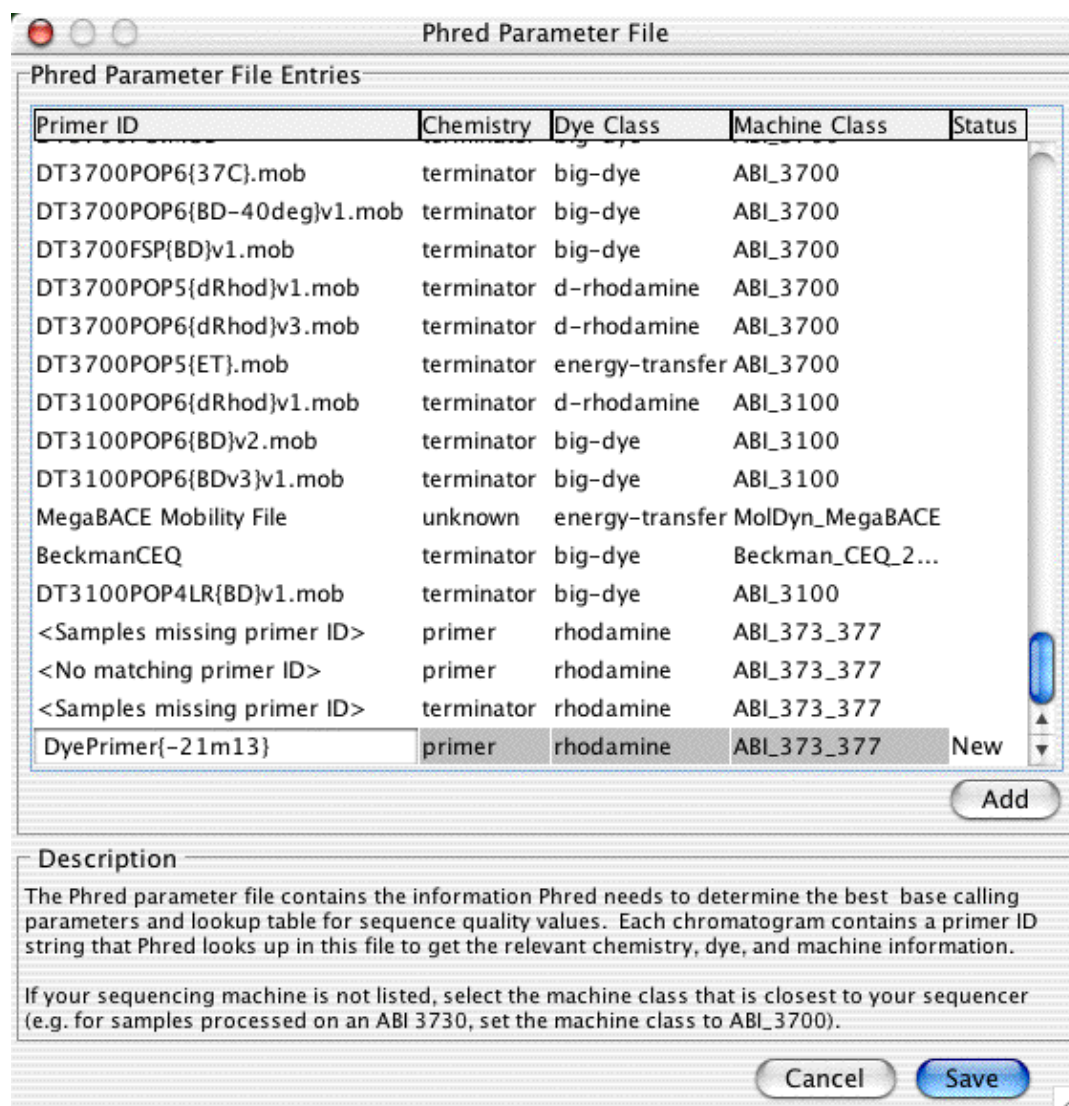
The first time you try to base call your own data with PHRED, you will probably need to add entries to the "Phred parameter file". PHRED uses this file to determine which sequencing chemistry, dye type, and sequencing machine was used to generate each individual sequence. PHRED does that by trying to match a "primer ID" string that is hidden in your chromatogram files with entries in the "Phred parameter file". Since sequencing vendors often come out with slightly changed chemistries that have a new "primer ID" string, chances are that you have to add one or more strings to the Phred parameter file.

Fortunately, CodonCode Aligner makes adding new entries to the Phred parameter file relatively easy by providing a convenient editor for the Phred parameter file, and by making educated guesses for the entries that need to be added. Before running Phred, Aligner checks all samples which you selected for base calling to see if their "primer ID" string can be found in the Phred parameter file. If any entries are missing, you will see the

following dialog:



When you click "OK", Aligner will bring up a new **dialog that allows you to edit the Phred parameter file**. It shows all the current entries in the Phred parameter file, as well as one or more entries that need to be added:



At the bottom, you see a newly added line (in your case, this may be more than one line; you may have to scroll down to see the newly added entries). Here is a brief description:

- The first column contains the **primer ID** string that you need to add. Aligner already filled this out for you, so you should not need to change anything here.
- The second column contains the **chemistry**, which is "primer" for dye-labeled primers, and "terminator" for dye-labeled terminator chemistry. Most sequencing nowadays is done with dye-labeled terminators, but check with the supplier of your sequences or sequencing kits if in doubt.
- The third column contains details about which **class of fluorescent dyes** was used. Most current chemistries from ABI are "big-dye", while chemistries from Amersham Pharmacia are typically "energy-transfer".
- The fourth column lists the **kind of DNA sequencer** used. If you cannot find an exact match, use the machine that is most similar to your machine. For example, if you are using PHRED version 0.020425.c, **use "ABI_3700" for data from ABI 3730 and ABI3730XL sequencers**, and use "ABI_3100" for data from ABI 310 sequencers.

You can also add new entries by pressing the "Add" button, but that should usually not be necessary. After verifying the new entries, press the "Save" button to save your changes. *If you press "Cancel", your changes will not be saved, and you will most likely see the "[Missing entries](#)" error described [below](#).*

You could also use a text editor to edit the Phred parameter file, but unless you really know what you are doing, we strongly suggest that you use Aligner's build-in editor instead, as described above.

You can also check and edit the Phred parameter file by pressing the button labeled "Edit..." in the [Base Calling Preferences](#).

Base calling problems

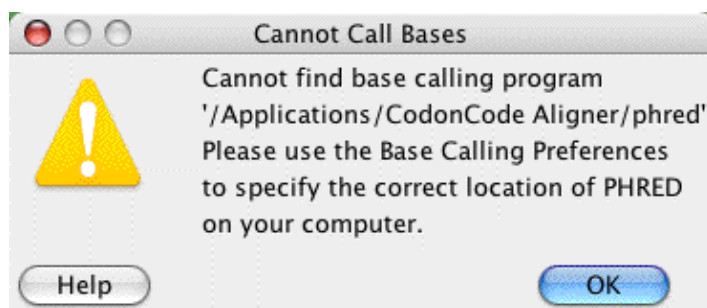
When using Phred for base calling, a number of things can go wrong. This section describes the most common problems, and how to solve them.

- [Cannot find the base calling program](#)
- [Missing entries in the Phred parameter file](#)
- [Problems reading the Phred parameter file](#)
- [Wrong command line parameters](#)
- [Problems running the workstation version of Phred](#)

Many of the problems are related to the Phred parameter file. This is a file used by Phred to determine which dye type, sequencing chemistry, and sequencing machine was used to a given trace. For more information, please read the "[About the Phred parameter file](#)" section below.

Cannot find base calling program

If CodonCode Aligner cannot find the base calling program, Aligner will show the following error message:



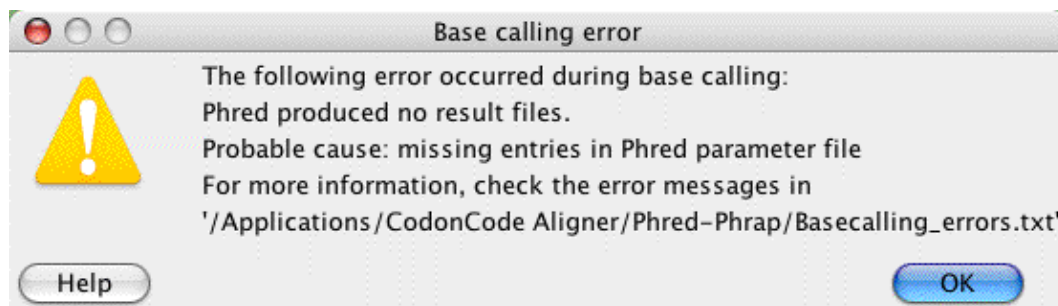
The name and location of the base calling program PHRED is defined in the base calling preferences; if the name specified there is not correct (for example because you moved, renamed, or deleted the program or the CodonCode Aligner folder), then Aligner cannot find Phred.

To solve this problem, do what the dialog says: use the [base calling preferences](#) to specify where exactly on your system Phred is installed. detailed instructions are given [above](#) for the workstation version of Phred that is installed with CodonCode Aligner.

For more information, please read the [Prerequisites](#) section.

Missing entries in the Phred parameter file

Another possible error message you may encounter looks like this:



This happens if Phred cannot find the "dye primer string" in the Phred parameter file (*since version 1.1.1 of Aligner, this should not happen anymore, since Aligner tries to help you in adding the missing entries before running PHRED, as described [above](#); however, you might still see this problem is you accidentally edited or misspelled a primer ID string, pressed "Cancel" in the Phred Parameter File dialog, or (against all our warning!) used a text editor to edit te Phred parameter file*).

Go ahead and open the file "Basecalling_errors.txt" in Phred-Phrap directory inside the CodonCode Aligner directory. In there, you will see one or more entries like this:

```
SampleFileName.abi: unable to match primer ID string:  skipping chromatogram
unknown chemistry (DT3730POP7{BD}.mob) in chromat SampleFileName.abi
add a line of the form
"DT3730POP7{BD}.mob" <chemistry> <dye type> <machine type>
to the file /usr/local/genome/lib/phredpar.dat
```

To fix this problem, you need to add the line as requested to the Phred parameter file, as described [above](#).

At the top of the file, it tells you the known chemistries, dye types, and machines; pick the one that is closest to what was used for your sample. For example, the line you add may read:

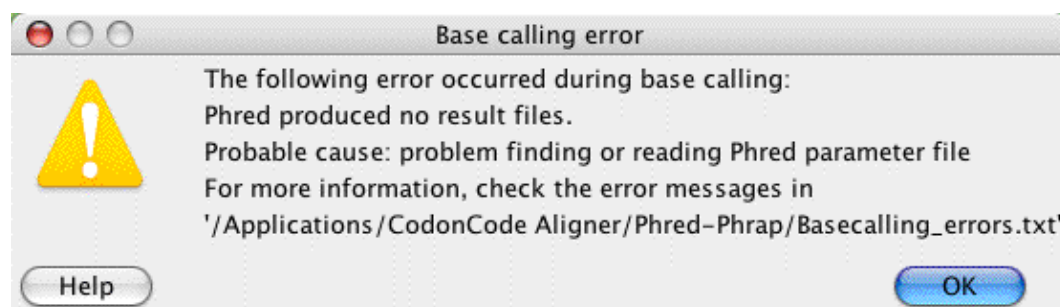
```
"DT3730POP7{BD}.mob" terminator big-dye ABI_3700
```

for the most commonly used sequencing chemistries from ABI 3700 or ABI 3730 sequencers.

Save the changed Phred parameter file (make sure to save it in a "Text only" format if using Word as the editor), and try the base calling again.

Problems reading the Phred parameter file

If PHRED cannot find or read the Phred parameter file, you will see the following error message:

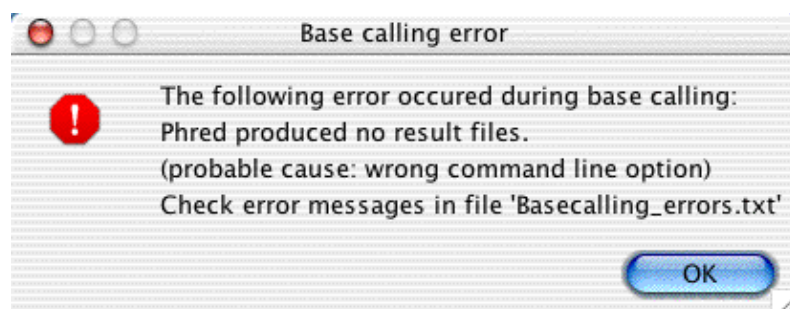


To fix this problem, you can do the following:

1. Make sure that the path to the Phred parameter file that is specified in the [base calling preferences](#) is correct, and that the Phred parameter file is readable.
2. Check the "Basecalling_errors.txt" in the Aligner directory for hints about what is wrong.
3. Verify that the Phred parameter file is a plain text file, and not a binary format like RTF or Microsoft Word's ".doc" format.
4. Make sure that the file has the correct line endings, especially on OS X (due to its mixed origins, OS X files can have either Macintosh-style line endings or UNIX-style line ending; Phred requires UNIX line endings on OS X. You can use tools like [BBEdit Lite](#) or [LineBreak 2.2](#) to convert line endings, if needed.

Wrong command line parameters

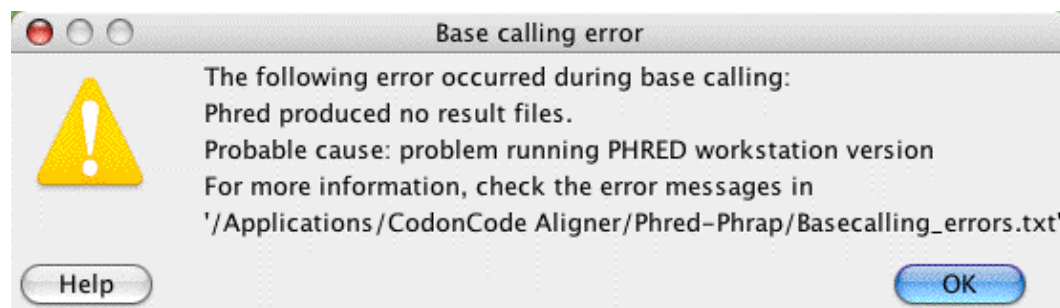
If you see the following error message:



then you did not read the "for experts only" part in the Base Calling Preferences, did you? Or perhaps you just mistyped an option - go back to the [base calling preferences](#) and change the "Additional command line options". If you leave this line blank, as we suggest, you should not see this error message.

Problem running the workstation version of Phred

When Aligner uses the workstation version of Phred, you may see the following error message:



Usually, you should not see this dialog - but if you see it, the first thing to try is to do the base calling again - it might work without problems the second time.

One thing that may cause this problem is renaming the Phred executable. Aligner looks at the program name to determine whether you are using the workstation version of Phred, or the regular version. If you have a regular (non-workstation) version of Phred, and you rename it to "workstation_phred", you will see the dialog above.

If you did not rename the executable, and the problem persists, it's time to contact CodonCode support. Please send an email, and include the following as attachments:

- The "Basecalling_errors.txt" file - the error dialog will tell you where exactly to find it
- The "Aligner_errorlog.txt" file from the "CodonCode Aligner" directory
- The "status.log" file from the project directory
- A screen shot of the error message

Please send all this by email to support@codoncode.com.

More about the Phred parameter file

The newest versions of PHRED (from the 2002 releases on) optimize the base calling for different sequencing chemistries, dye types, and machines. This requires that Phred can somehow determine the chemistry, dye type, and machine. Phred does this by extracting the "primer ID string" from the input files. The primer ID string looks something like this: "DyePrimer{-21m13}" or "ET_MegaTerm" or "DT3700POP5{BD}v3.mob". PHRED looks up the primer ID string in the Phred parameter file, which tells it the dye type, chemistry, and machine through lines like this one:

```
"DT3700POP5{BD}v3.mob"      terminator      big-dye      ABI_3700
```

There are many different dye primer strings, and new ones get added all the time. If your sequence traces contain one is not yet in the Phred parameter file, PHRED will refuse to call the bases in this trace, and generate an error message. In Aligner, this leads to the "[Missing entries in the Phred parameter file](#)" error described above. As described [above](#), you will need to [edit the Phred parameter file](#), and add a new line for

each new primer ID string. The possible entries are:

- for chemistry: primer, terminator, unknown
- for dyes : rhodamine, d-rhodamine, big-dye, energy-transfer, bodipy, unknown
- for machines : ABI_373_377, ABI_3100, ABI_3700, Beckman_CEQ_2000, LI-COR_4000, MolDyn_MegaBACE).

PHRED versions released in 2004 and later also support "ABI_3730" as the machine type (at the time of writing, the new versions are still in beta testing, and have not yet been released). For more information, please check the [discussion groups](#) on CodonCode's support site.

In general, do not use the "unknown" tags. If your dye or machine is not listed, use the one most similar to it (for example ABI_3700 for ABI 3730 XL sequencers if you are using PHRED version 0.020425.). The most common combination for ABI sequencing is terminator, big-dye, ABI_3700.

The Phred parameter file can be in different locations on different systems. On Mac OS X, the default location is "/usr/local/genome/lib". On Windows, the default location is "/C:\Program Files\CodonCode\". To find out where the Phred parameter file is located, Phred checks the environment variable "PHRED_PARAMETER_FILE". This environment variable is temporarily set by Aligner when Aligner starts Phred, using the location of the Phred parameter file defined in the [base calling preferences](#).

Additional tips for common problems with running Phred can also be found at <http://www.codoncode.com/support/faqs/phedpar.html>. If you want to, you can read more about Phred in the original Phred documentation, which is available at <http://www.codoncode.com/support/phred.doc.html>.

About Phred

The base calling program Phred was developed by Phil Green and Brent Ewing at the University of Washington. Phred was widely used for base calling in the Human Genome Project, and still is often regarded as the standard for base calling. CodonCode Corporation distributes Phred executables for a variety of platforms, including Windows and Mac OS X, under license from the University of Washington.

Academic users can obtain the source code for Phred for restricted use directly from the authors at the University of Washington. For more information on this, please visit <http://www.phrap.org/>.

Trial versions of CodonCode Aligner may include time-limited trial versions of Phred. Any use of such trial versions of subject to the license agreements displayed when you installed Aligner or Phred. In particular, you may not use the time-limited trial versions for any commercial purposes.

Please note that Phred is a command line program, not a typical Windows or Mac OS X application! This makes it easy to run Phred through scripts or from programs like CodonCode Aligner, but it means you cannot simply double-click on Phred and expect to see a graphical user interface.

You can read the original documentation for Phred at <http://www.codoncode.com/support/phred.doc.html>, and more about Phred in the following two articles:

- B. Ewing B, L. Hillier L, M.C. Wendl and P. Green. "Base-calling of automated sequencer traces using Phred. I. Accuracy assessment." *Genome Research*. 8: 175-85. ([Available](#) online at <http://www.genome.org/content/vol8/issue3/>).

CodonCode Aligner User Manual

- B. Ewing and P. Green, 1998 "Base-Calling of Automated Sequencer Traces Using Phred. II. Error Probabilities." *Genome Research*. 8: 186-198. ([Available](http://www.genome.org/content/vol8/issue3/) online at <http://www.genome.org/content/vol8/issue3/>)

End Clipping

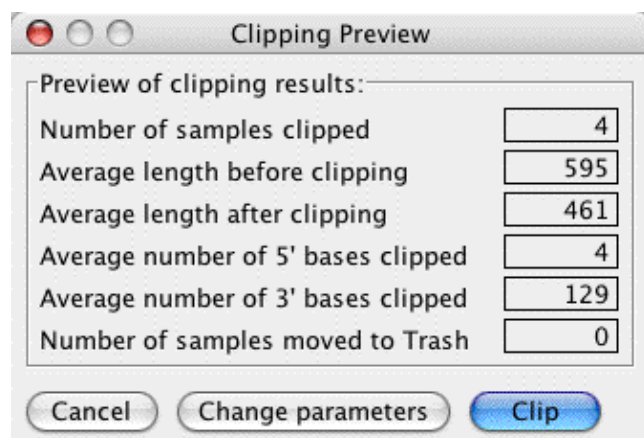
Typically, sequence chromatograms have low-quality sequence at the beginning and at the end of the sequence. If you have sequence traces with quality values, you can use the quality values to automatically remove the low-quality sequence at the ends - a process called "end clipping" (or "end trimming").

*If you plan to use Aligner's features to [detect and process heterozygous insertions and deletions](#), you should search for heterozygous indels **before** you end clip. Otherwise, end clipping will probably remove the parts of samples that have heterozygous insertions or deletions, and some heterozygous indels may not be detected. But if you first search for heterozygous indels, end clipping will leave the parts of sequences that have heterozygous indel tags unclipped.*

To end clip a set of sequences, do the following:

1. Select the sequences to be clipped in the project window. Typically, you can just select the "Unassembled Samples" folder to end clip all samples (only unassembled samples can be end clipped).
 2. Select "Clip Ends..." from the sample menu. This will show the "Clipping preview" window that summarizes the clipping results (if you have selected many samples, it may take a few seconds for the dialog to show up).
 3. If the end clipping results look ok to you, press the "Clip" button to apply the calculate clips. The low-quality bases at the end of each sequences will be removed, and any sequences that do not match the minimum quality criteria will be moved to the trash.
- If you would like to change the settings for the end clipping, press the "Change parameters" button instead of the "Clip" button. This will open a new window where you can change the end clip settings. After making your changes and selecting "OK", the clipping preview window will re-appear, showing the effects of the new parameters.

The clipping preview window is shown in the next picture:



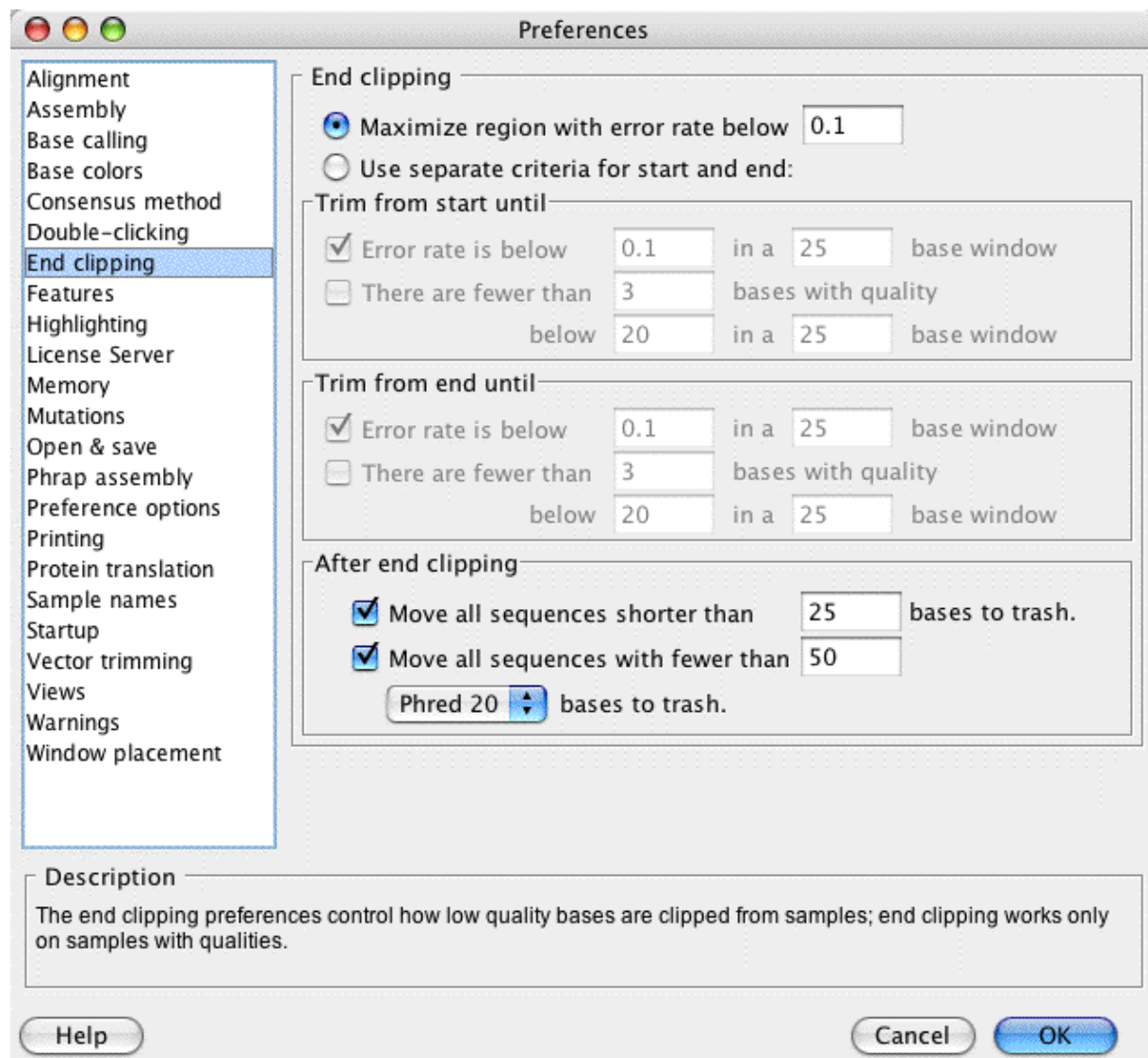
Please note that you do not have to end clip sequences before assembly or alignment - the assembly will typically work fine without end clipping. However, you can get cleaner and sometimes better assemblies by end clipping sequences before vector screening and assembly.

To each sequence that is clipped, Aligner will add a "processing" tag which states how many bases were clipped at the beginning and the end. You can see these tags in the [feature view window](#) if you have the

processing tags included in your [definition of features](#), and in the [tag dialog](#) that is accessible from the [sample information dialog](#).

End Clipping Parameters

You can change the stringency of the end clipping in the end clipping preferences. Either press the "Change parameters" button in the clipping preview window as described above, or select the "End clipping" preference panel in the "Preferences..." window (to see the Preferences window, select "Preferences" in the application menu on OS X, and "Preferences..." in the "Edit" menu on Windows). This will show the following window:



On the top, you have the choice between two different end clipping methods: a method that maximizes the region with an (estimated) error rate below the threshold you define, and a method that uses different criteria at the beginning at at the end of the reads. The first methods is similar to the way Phred trims sequences with the "-trim_alt" option. The second method gives you more options, and is (hopefully) a bit easier to understand. With typical parameters, both methods tend to give similar results, and remove the "junk" sequence at the end of chromatograms. The methods are [explained in more detail below](#).

You have the option to automatically identify "bad" sequences after the end clipping, and move those to the trash. Such bad sequences can be due to failed sequencing reactions, or any number of other problems. We suggest to move all sequences that are too short (for example, less than 25 or 100 bases) after clipping to the trash. A second widely used method to identify low-quality sequences is to count the number of bases with quality scores above 20 ("Phred20 bases"). With the settings shown in the picture above, any samples that, after the end clipping have fewer than 150 Phred20 bases, or are shorter than 200 bases, would be called bad and moved to the trash.

We suggest that you experiment with the different parameters, and find a setting that works for your data. For example, if you sequence short PCR products, you should definitely reduce the number of Phred20 bases required for a sequence to be kept, or perhaps uncheck this option.

Additional details about the end clipping parameters can be found on the ["End Clipping Preferences" help page](#).

End Clipping Algorithms

This section briefly explains the different end clipping methods ("algorithms"). It is intended for the curious - you do not necessarily need to understand the end clipping algorithms to use them :-). All methods use the base-specific quality scores to find the low quality regions. For the end clipping to work correctly, the quality scores have to be reasonably accurate; however, they do not have to be perfect.

Method 1: Maximizing regions with error rates below a given threshold

First, the quality scores at each base call are converted into estimated error rates. The quality scores are on a logarithmic scale - a quality of 10 corresponds to a 10% error rate, a score of 20 to 1%, 30 to 0.1%, and so on. Next, Aligner **finds the longest region** in the sequence where the following conditions are met:

- the average error rate over the entire region is below the (user-defined) threshold
- if the region would be expanded on either side, then the added sections would have an error rate that is **above** the threshold

The region found by this method can contain some lower-quality parts in the middle, where the estimated error rate is above the threshold. However, such regions will only be included if they are followed by a region that has a lower error rate, which drops the combined error rate over both regions below the threshold. Individual bases and short regions outside of the "good" region may also have error rates below the threshold - however, they are not included when they are flanked by higher error regions.

The "maximize region" method is very similar to the method used by the base calling program Phred with the -trim_alt option, and based on the Mott algorithm for sequence trimming. It's a bit hard to understand, but works rather well.

Method 2: Using separate criteria at the start and the end of the sequence

In this method, the clipping is done by coming in from the start and from the end of a sequence, and "chewing into" the sequence until a region that is "good" is encountered. There are two different ways to define what is a "good" region:

CodonCode Aligner User Manual

1. the average estimated error rate in a region (calculated from the quality scores), and
2. the number of "bad" bases (bases with a quality below the defined threshold).

You can also define the length of the region, and use different regions at the beginning and at the end of a sequence. At the beginning, where sequence quality improves rapidly, a short window (e.g. 20 bases) can make sense, while at the end, where the sequences quality drops of slowly, a longer windows (e.g. 50 bases) can make sense. You can use either one of the two definitions, or you can use both together.

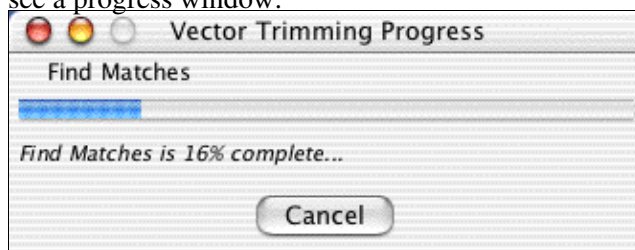
The advantages of this method are that (a) it is a bit easier to understand, and (b) it gives you more options to tune the trimming exactly like you want it.

Trimming Vector Sequences

Vector sequences in your sample sequences can lead to incorrect assemblies, and therefore should be trimmed before assembly or alignment. *If you also plan to use end clipping to remove low-quality sequence, we strongly suggest that you do the end clipping before the vector trimming.*

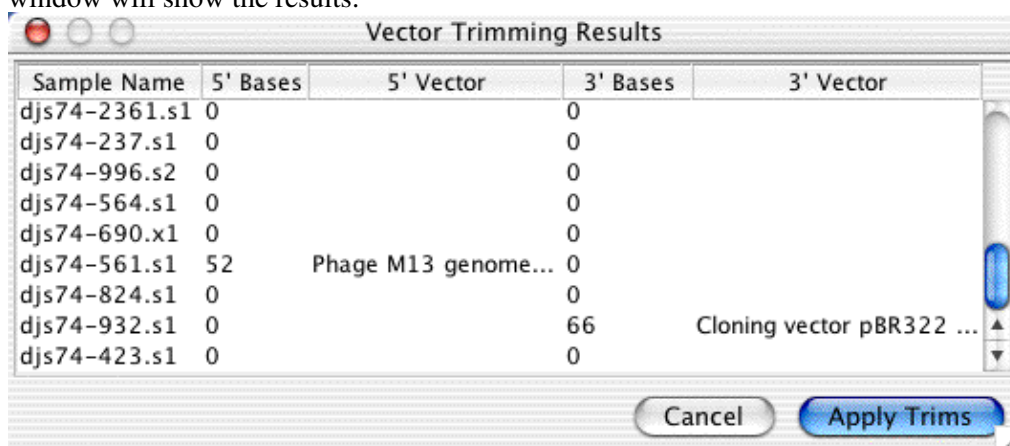
To remove vector contamination from samples, follow these steps:

1. Check your "Vector trimming" Preferences. Make sure that you have selected the cloning vector(s) that you used.
2. Since vector trimming cannot be undone, it's a good idea to save your project now.
3. In the Project window, select the samples that you want to screen. You can screen only unassembled samples, not samples in contigs. Typically, just select the "Unassembled Samples" folder to screen all unassembled samples.
4. Select "Trim Vector" in the "Sample" menu. Aligner will start to look for matches between the samples you selected in the project window, and the vectors you selected in the Preferences. You will see a progress window:



You can cancel the vector trimming at any time before it is complete.

5. After Aligner has found all vector matches that fit the criteria you defined in the Preferences, a new window will show the results:



(You will see a different dialog if no matches between your samples and the vector sequences were found).

You can now choose to apply the trim results and remove the bases at the start or end of the samples that had vector matches; or you can press cancel and not apply the trim results (for example to try different parameters).

You can repeat the vector trimming, for example if you forgot to include a vector sequence the first time you trimmed.

To each sequence that has bases removed due to vector trimming, Aligner will add a "processing" tag which

states how many bases were trimmed at the beginning and/or the end. You can see these tags in the [feature view window](#) if you have the processing tags included in your [definition of features](#), and in the [tag dialog](#) that is accessible from the [sample information dialog](#).

Vector Library Files

Vector sequences for vector trimming can be read from several files that were supplied with CodonCode Aligner, or from your own ("custom") files. A "Vector" folder is created in the "Aligner Data" folder in the folder where CodonCode Aligner was installed (*on OS X, the default folder is "/Applications/CodonCode Aligner/"; on Windows, the default folder is "C:/Program Files/CodonCode Aligner/"*). Three files are installed in this vector folder: two UniVec files from NCBI, and an example custom vector file with some commonly used vector sequences.

To set up vector trimming, you:

1. Open the "Vector trimming" preference panel
2. Select the vector library file to use (typically from your own "Custom Vector" file)
3. Select the sequences you want to screen against

Using UniVec Library Files

UniVec is a database created by the NCBI where redundant sub-sequences from vectors have been removed. UniVec also contains sequences for linkers, primers, and adapters that are commonly used in cloning. Two UniVec library files have been installed in your "Vector" folder inside your settings folder.

- UniVec.txt
- UniVec_Core.txt

For more information about UniVec refer to:

<http://www.ncbi.nlm.nih.gov/VecScreen/UniVec.html>

<http://www.ncbi.nlm.nih.gov/VecScreen/Interpretation.html#Definitions>

The typical use of UniVec for vector screening would be to screen against all sequences in UniVec. In CodonCode Aligner, this is currently not supported - it would take a **really** long time. You can select several sequence fragments from UniVec to screen against, but **typically, it's a better idea to use a custom vector file**.

Using Custom Vector Files

Custom vector files can contain any collection of vector, linker, etc. sequences that you want to screen against in FASTA format. An example custom vector file called "CustomVectors.txt" is installed in your "Vector" folder inside your Preferences folder. It includes some common vectors like BlueScript, M13, pBr, pGEM and pUC. Custom files can be kept in the same vector folder as the UniVec files, or in any folder that you choose. To add sequences you want to screen against, we suggest that you make a copy of the "CustomVectors.txt" file, and edit the copy. If you use linkers or PCR primers in your subcloning, it's a good idea to add the sequences of the linkers and primers to the custom vector file, and select them to be included when screening (in the "Vector trimming" preference panel).

When screening against short sequences like primers, however, keep in mind that short matches may not meet the minimum match criteria, and therefore not be masked. Minimum match criteria are discussed under "Vector screening" preferences.

CodonCode Aligner User Manual

Note: When editing a custom vector file, please make sure that it is saved "text (.txt)" file. Aligner will not be able to read files in other formats, like Microsoft Word's ".doc" format.

Custom vector files must be in FASTA format: each sequence must have a line that has a "greater than" (>) sign, followed by the name. The file can have multiple entries; here is an example:

```
>gnl|uv|L09150.1:3248-3353 pUR291 cloning vector
CGCCGGTCGCTACCATTACCAGTTGGTCTGGTGTCTGGGGATCCGTCGACCTGCAGCCAAGCTTATCGATG
ATAAGCTGTCAAACATGAGAATTCTTGAAGACGAAA
>gnl|uv|L09151.1:3249-3332 pUR292 cloning vector
GCCGGTCGCTACCATTACCAGTTGGTCTGGTGTCTAGGGGATCCGTCGACCTGCAGCCAAGCTTATCGATG
ATAAGCTGTCAAAC
>pUR278 cloning vector bases 3239-3335
CCAGCTGAGCGCCGGTCGCTACCATTACCAGTTGGTCTGGTGTCAAAAAGGGGATCCGTCGACTCTAGAA
AGCTTATCGATGATAAGCTGTCAAACA
```

The names for the entries are used when you select the vector sequences to screen against in the preferences - that's why you cannot use plain text vector sequence files.

Assembly and Alignment

Sequence Assembly

Assembly assembles sequence fragments into a larger sequence by identifying overlaps between sample sequences. Samples that can be joined together are put into "contigs". Joins may fail because samples do not share overlaps that are long enough with other samples or contigs, or because the overlap contains too many discrepancies. Any sequences that cannot be put into contigs remain in the "Unassembled Samples" folder.

The result of the assembly can be one or more contigs, plus samples that remain in the "Unassembled Samples" folder. If none of the samples can be joined, no contigs will be formed.

You can assemble any mix of unassembled samples and previously assembled contigs. Contigs are assembled as they are, they are **not** dissolved first. However, the consensus sequence of contigs may change where new samples are added.

Several [advanced options](#) for sequence assembly are available through the "**Assemble with Options...**" item in the "**Contig**" menu. These include

- [Assembling from scratch](#) - this option dissolves existing contigs before assembly. With this option, you also have a choice between two assembly algorithms - the built-in assembly algorithm and the assembly program [PHRAP](#) (note that users at companies may have to pay a separate license fee to use PHRAP).
- [Comparing contigs](#) - this lets you build "Contigs of contigs", for example for phylogenetic studies. Again, you can choose between two algorithms - the built-in assembly algorithm and the alignment program ClustalW.
- [Assemble in groups](#) - this option lets you define how CodonCode Aligner should group samples based on their names; Aligner will then build separate contigs for each sample group.
- [Pre-processing before assembly](#) - this enables you to pre-process unassembled samples automatically before assembly by base calling, end clipping, and/or vector trimming.

Alignment to a Reference Sequence

Alignment to a reference sequence is used to determine differences between sequences and a known sequence. This requires that you first designate one sequence as the reference sequence. All other samples are then aligned to this sequence. If necessary, the samples are reverse-complemented before alignment.

Alignment results in one new contig that contains the reference sequence as well as any samples that could be aligned to it. If no samples could be aligned to the reference sequence with the current alignment criteria, no new contig will be formed, and all samples will remain in the "Unassembled Samples" folder.

The new contig will be limited to the length of the reference sequence, plus any gaps introduced during alignment.

When building the consensus sequence for contigs that result from alignments to a reference sequence, the [consensus preferences](#) determine how the reference sequence is considered. By default, the reference sequence is excluded from the consensus sequence. In this case, any regions where none of the samples overlap with the consensus sequence will result in gap characters in the consensus sequence. Alternatively, you can choose to use the reference sequence as the consensus sequence.

Limitations: Currently, the implementation of alignments has several limitations. These include:

- The numbering in the aligned contig is not relative to the reference sequence numbering, and includes any gaps introduced in the consensus sequence. However, if you use the "Find mutations" function, the numbering used in the mutation tags will be relative to the reference sequence (and, if present, the coding sequence annotation of the reference sequence).
- With the default settings, aligning a cDNA sequence with several exons to a genomic sequence will typically fail, or give a bad alignment. However, you can change this by selecting the "Large Gap" algorithm in the [alignment preferences](#) (the "Large Gap" algorithm is specifically intended for cDNA to genomic alignments).

Sequence Assembly

Before assembling

Before performing an assembly in CodonCode Aligner, you'll need to create or open a project, and import the sample files you want to assemble. You also may want to pre-process your samples by end clipping and vector trimming (it's a good idea to save your project now, before starting the assembly).

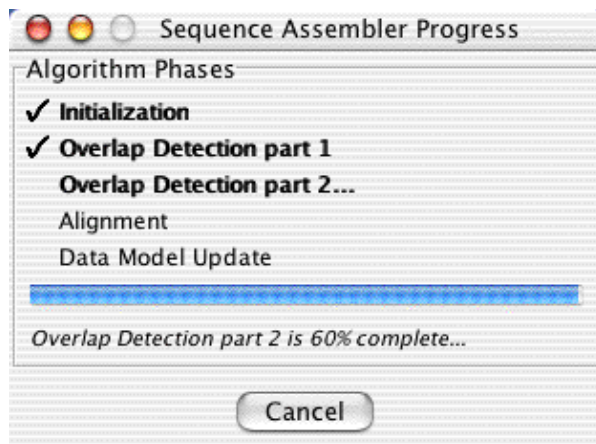
How to assemble

- Go to the project view
- Select the samples and/or contigs that you want to assemble
- Choose "Assemble" from the "Contig" menu

Note:

- To make a **continuous selection**, keep the "shift" key pressed while clicking on samples or contigs
- To make a **discontinuous selection**, press the "control" key (on Windows) or the "command" key (on OS X) while clicking on samples or contigs

The assembly will start and show a progress window:



After the assembly is done, the progress window will disappear, and the project window will now show the newly formed contig(s) (unless no joins were successful, in which case you will see a dialog telling you so).

When CodonCode Aligner assembles pre-formed contigs, it looks for matches between the consensus sequences, and leaves the alignment of reads in the contig (mostly) unchanged. If contigs are merged with other reads or contigs, any necessary gaps will be added.

Advanced assembly options

In addition to the simple assemblies described above, CodonCode Aligner also supports several "advanced" assembly options, including:

- [Assemble from scratch](#) - this will unassemble any existing contigs in your selection before starting the assembly. This option can be useful if you want to undo manual introduction or movement of

gaps, or to try different assembly parameters.

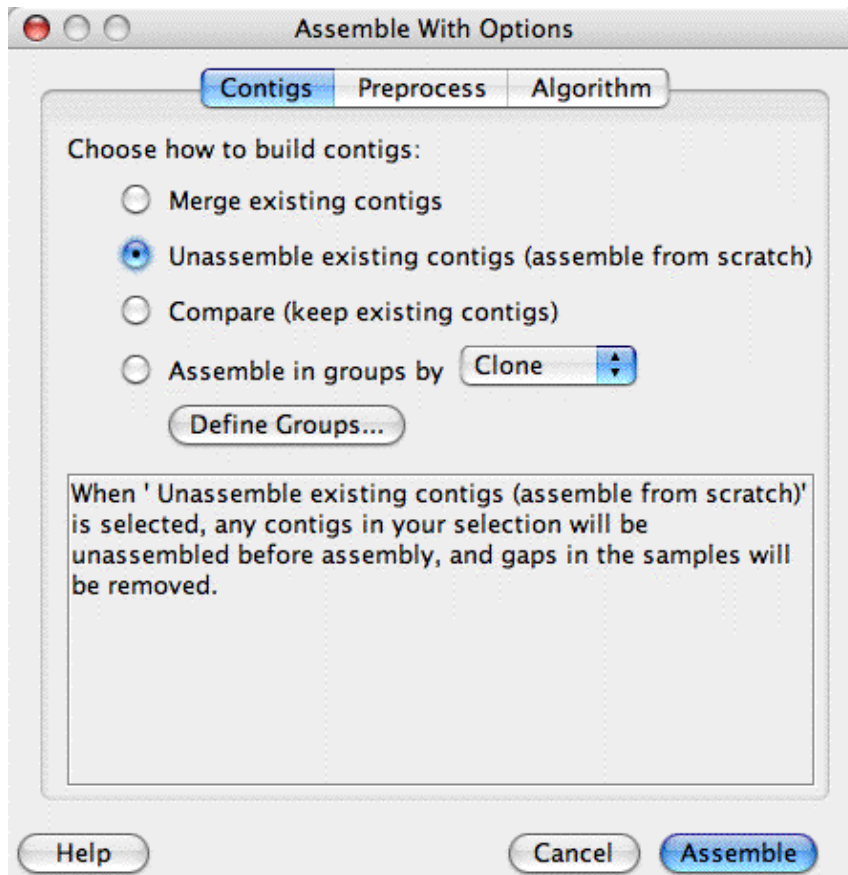
- **[Compare contigs to each other](#)** - this option allows you to compare several contigs, for example when studying genes from different species, isolates, or patients. The consensus sequences will be assembled into a new contig; to check out any differences, double-clicking on a sequence in this "contig of contigs" will bring you straight to the sequence traces.
- **[Assemble in groups](#)** - with this option, CodonCode Aligner can automatically group samples based on their **names**, and form separate contigs for each group. This option can be a real time saver in phylogenetic and medical studies where you look at sequences from many species, isolates, or patients.
- **[Assemble with PHRAP](#)** - this option will use the assembly program PHRAP, rather than CodonCode Aligner's built-in methods, for sequence assembly. It can be useful for larger shotgun assemblies.
- **[Preprocess](#)** - in addition to the choices above, Aligner can also automatically do common pre-processing steps like end clipping and vector trimming before assembly.

Assemble from scratch

Sometimes, you may want to re-assemble a contig, for example after you did some editing and ended up messing up the alignment of reads. Or you may want to merge a contig with another contig and/or additional samples without preserving the existing alignment of reads in the contig.

To re-assemble one or more contigs from scratch:

- Go to the project view
- Select the contig, contigs, or contig(s) and sample(s) you want to assemble from scratch (*keep the shift-, control-, or command-key pressed to make continuous or discontinuous selections, as described [above](#)*)
- Choose "**Assemble with Options...**" from the "**Contig**" menu
- This will open the following dialog:



- Choose "**Unassemble existing contigs (assemble from scratch)**", then click on "**Assemble**"

Aligner will first unassemble the contigs you selected, and then assemble all the samples in the contig(s) and any other samples you had selected. You will see a progress dialog while Aligner is assembling.

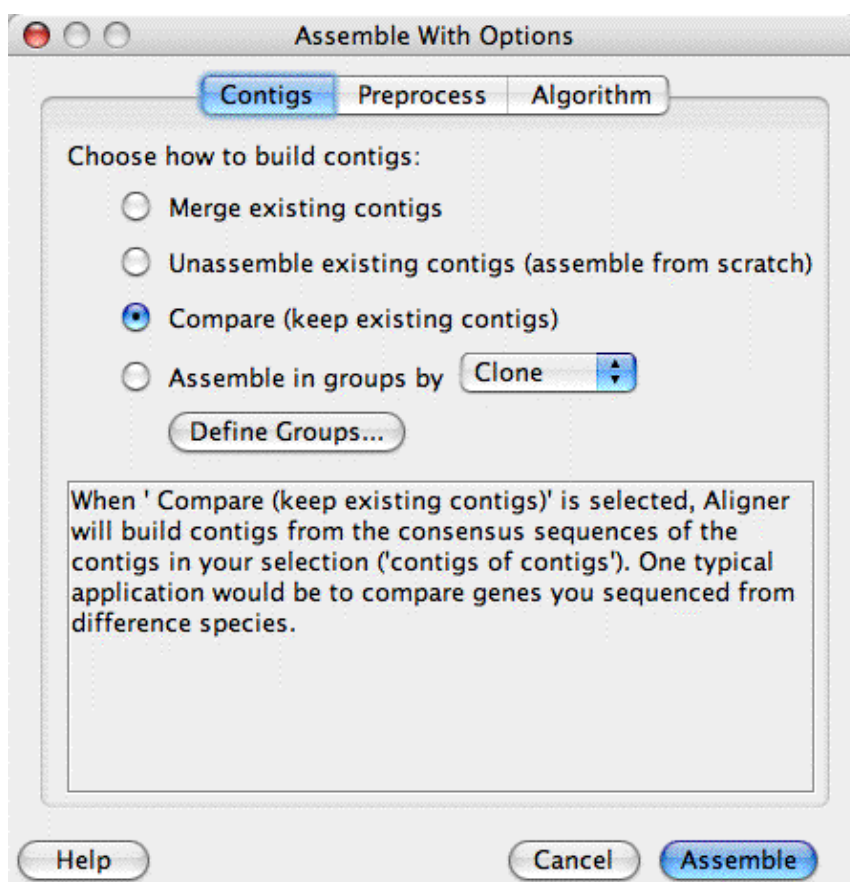
Note that the resulting number of contigs may be different from the initial number of contigs; if you want to just re-assemble contigs without merging the contig with other samples, make sure to just select one contig before choosing "Assemble from Scratch".

If your selection contained only unassembled samples, assemble from scratch will do exactly the same thing as "Assemble" would (unless you also have [preprocessing](#) options selected).

Compare contigs to each other

A common DNA sequencing application is to first sequence the genes from several sources, for example different species or patients, and to then compare (align) the consensus sequences to each other. CodonCode Aligner allows you to do this by creating "contigs of contigs", as follows:

- Assemble the contigs separately, for example generating one contig from several forward and reverse reads for each species.
- Select the contigs you want to compare in the project view. You can also include individual sequences, for example text sequences that you downloaded from sequence databases.
- Choose "**Assemble with Options...**" from the "**Contig**" menu.
- In the dialog that opens, choose "**Compare contigs to each other**". The dialog should now look like this:



- Click on "Assemble".

Aligner will start comparing the contigs to each other, showing you a progress dialog during the assembly. By default, CodonCode Aligner version 1.6 and newer will use the program **ClustalW** to generate contigs of contigs. ClustalW is a program that is widely used to generate alignments for phylogenetic studies; a copy of the program is included with the CodonCode Aligner distribution, and installed in the "Phred-Phrap" folder inside the CodonCode Aligner install folder.

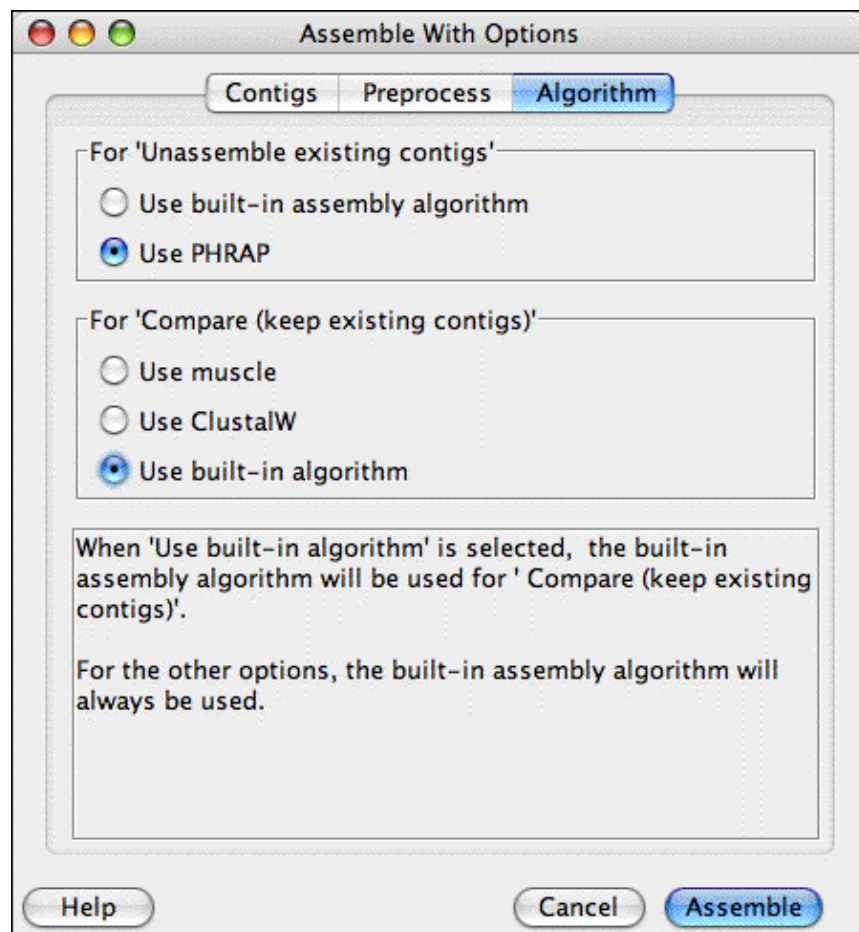
To generate alignments with ClustalW, CodonCode Aligner does the following:

1. Aligner examines your current selection to see if any contigs or samples need to be reverse-complemented before alignment, and reverse-complements these contigs and/or samples.
2. Aligner generates an input file for ClustalW.
3. Aligner starts ClustalW, and waits for the alignment to finish. During this time, a progress dialog is shown. Please note that alignments with ClustalW can be slow, especially if you align many contigs or samples, and if you contigs are large.
4. When ClustalW is finished, CodonCode Aligner analyzes the output generated by ClustalW, and create a new contig from the ClustalW alignment. Aligner also calculates a consensus sequence for the alignment.

Algorithms for comparing contigs: ClustalW, muscle, built-in

If you do not want to use ClustalW to generate contigs of contigs, you have two alternative options: to use either the program **"muscle"**, or CodonCode Aligner's **built-in assembly algorithm** to form contigs of contigs. To do this, click on the "Algorithm" panel in the "Assemble With Options" dialog, and select the "Use muscle" or "Use built-in algorithm" button in the middle of the dialog. Your dialog should now look like

this:



When you click on "Assemble" (and the "Compare" button is selected in the "Contigs" panel), CodonCode Aligner's built-in algorithm will be used to create contigs of contigs. The following table gives an overview of some of the differences between the algorithms:

muscle	ClustalW	CodonCode Aligner's built-in algorithm
Developed and optimized for sequence alignments; newer than ClustalW	Developed and optimized for sequence alignments	Developed for "shotgun" sequence assemblies
Always generates end-to-end alignments	Always generates end-to-end alignments	Uses local alignments that can include unaligned ends by default; can also generate end-to-end alignments and "large gap" alignments
Alignments formed generally include all input sequences	Alignments formed generally include all input sequences	Alignments may or may not include all input sequences, depending on sequence similarities and assembly parameters
Tends to be faster than ClustalW; alignments are often better than ClustalW alignments	Tends to be slow, especially for alignments with many contigs and/or samples	Tends to be faster than ClustalW

Comparisons (alignments) can be generated for samples only	Comparisons (alignments) can be generated for samples only	Comparisons must include at least one contig (for samples only, use "Assemble")
--	--	---

You can work with the new "contig of contigs" the same way you would with normal contigs. Double-clicking on a base for one of the contigs in the contig of contigs will open the trace views for this contig. You can open separate trace views for different contigs to quickly check any discrepancies.

To cite muscle: Edgar, Robert C. (2004), MUSCLE: multiple sequence alignment with high accuracy and high throughput, Nucleic Acids Research 32(5), 1792-97; muscle is available from <http://www.drive5.com/muscle>.

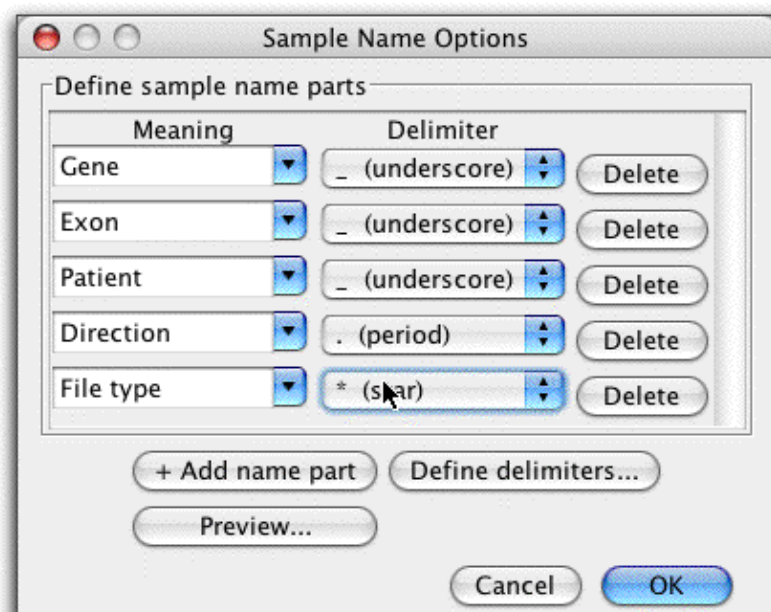
To cite ClustalW: Thompson, J.D., Higgins, D.G. and Gibson, T.J. (1994), CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position specific gap penalties and weight matrix choice, Nucleic Acids Research, 22(22):4673-4680.

Assemble in groups

For sequencing projects where you assemble contigs from a number of different sources (species, patients, isolates...), the "Assemble in groups" option in CodonCode Aligner can simplify your work by automatically grouping and assembling samples based on their names. To use assemble by groups, your samples must be named consistently, with different parts of the name separated by characters like underscores or periods. Let us look at an example sample name, "EGFR_exon19_JJM_F.abi". It consists of:

- The gene name ("EGFR")
- The exon ("exon19")
- A patient identifier ("JJM")
- The direction ("F" for forward)
- The file type extension ("abi")

Most name parts are separated by underscores, except for the last two parts, which are separated by a period. Obviously, sample naming conventions will be different for different projects, so you will need to tell Aligner how to interpret ("parse") sample names. You can do this in the ["Sample names" preferences](#), or by clicking on the "Define Groups..." button in the "Assemble with Options" dialog. For the example above, the definition would look like this:



Note that the last delimiter in this example does not matter, since the last part of the sample name is the file type.

To verify that you name part definition is correct, you can press the "Preview..." button; this will show the parsing of the currently selected samples in a separate window:

Name	Gene	Exon	Patient	Direction	File type
EGFR_exon19_JJM_F.abi	EGFR	exon19	JJM	F	abi
EGFR_exon19_JJM_R.abi	EGFR	exon19	JJM	R	abi
EGFR_exon19_NWS_F.abi	EGFR	exon19	NWS	F	abi
EGFR_exon19_NWS_R.abi	EGFR	exon19	NWS	R	abi
EGFR_exon19_XHS_F.abi	EGFR	exon19	XHS	F	abi
EGFR_exon19_XHS_R.abi	EGFR	exon19	XHS	R	abi
EGFR_exon20_JJM_F.abi	EGFR	exon20	JJM	F	abi
EGFR_exon20_JJM_R.abi	EGFR	exon20	JJM	R	abi
EGFR_exon20_NWS_F.abi	EGFR	exon20	NWS	F	abi
EGFR_exon20_NWS_R.abi	EGFR	exon20	NWS	R	abi
EGFR_exon20_XHS_F.abi	EGFR	exon20	XHS	F	abi
EGFR_exon20_XHS_R.abi	EGFR	exon20	XHS	R	abi

Close

After defining the name scheme, you can choose which name part Aligner should use to group samples by in the "Assemble with Options" dialog. In this example:

- Choosing "Gene" would try to assemble all reads into one contig
- Choosing "Exon" would try to assemble the genes into 2 contigs, one for exon 19 and one for exon

20.

- Choosing "Patient" would assemble the samples for each patient separately. Assuming that there is enough overlap between the exon 19 and exon 20 sequences, this would generate one contig for each of the three patients. Without sufficient overlap between the exon 19 and exon 20 sequences, the assembly would generate 2 separate contigs for each of the patients (for exon 19 and for exon 20), for a total of 6 contigs.
- Choosing "Direction" would assemble all the forward ("F") reads together, and all the reverse ("R") reads separately.

The contigs created will be named according to the name part used to group samples; for example, assembling by patient would create contigs called "JJM", "NWS", and "XHS". It is possible that the assembly will create more than one contig for each group, for example if some of the samples in a group do not overlap, or are too different from each other. If more than one contig per groups is created, the contigs will be named by adding numbers to the end, for example "JJM" and "JJM1".

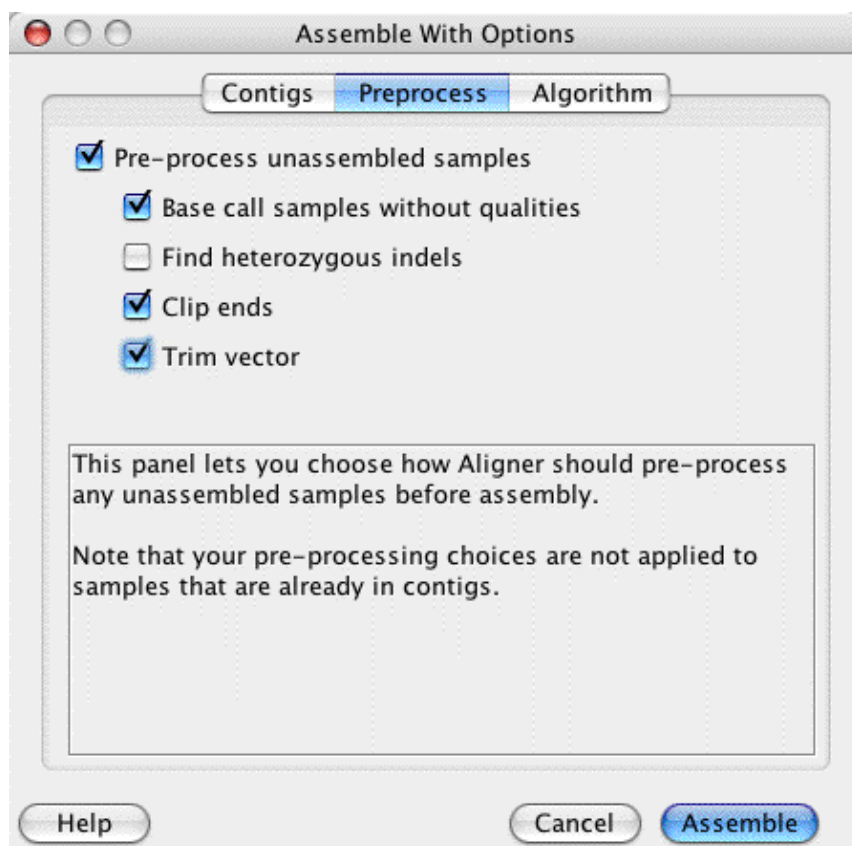
Assembling in groups will try to group and assembly all samples and contigs in your current selection. Any existing contigs in your selection will be unassembled before assembly.

After creating contigs by "Assemble in Groups", you can [compare the contigs to each other](#) by building "contigs of contigs", as described [above](#).

Additional information about defining name parts is available on the ["Sample names" preferences help page](#).

Pre-process: One-step processing

In addition to the different assembly choices described above, "Assemble with Options" also lets you automate common pre-processing steps like end clipping and vector trimming. To set your pre-processing choices, click on the "Preprocess" tab:



The checkbox at the top determines whether or not your samples will be pre-processed before assembly; the four lower checkboxes let you pick the pre-processing steps that will be done. The choices are:

- **Base call samples without qualities:** Any samples that have chromatograms, but no base-specific quality scores, will be base called with PHRED. To use this option, you will need either a trial license or a purchased license; base calling is not enabled in demo mode. Additional information about base calling can be found at the ["Base calling" help page](#). Please note that one important difference to the normal "Call bases" menu choice: in automated pre-processing, samples that already have quality values, for example from the ABI KB basecaller or from calling bases on the sample before, will not be base called again.
- **Find heterozygous indels:** This option will look for potential heterozygous insertion/deletion (indel) mutations in the unassembled samples that have (a) chromatograms and (b) quality scores. If you are sequencing PCR products from genomic DNA that may contain heterozygous indels, you should check this option; if you are sequencing from cloned DNA, this option should not be checked. For more information, please read the ["Heterozygous insertions and deletions" help page](#).
- **Clip ends:** This will remove low-quality sequence from samples that have chromatograms and quality scores. For more information, please read the ["End clipping" help page](#).
- **Trim vector:** When checked, this option will identify and remove vector sequence contamination from the samples in your selection. You may need to set your [vector trimming preferences](#) before using this option. For additional information, please read the ["Vector trimming" help page](#).

You will see progress dialogs for each of the steps performed after you click the "Assemble" button.

Note that any samples that are already in contigs will **not** be pre-processed.

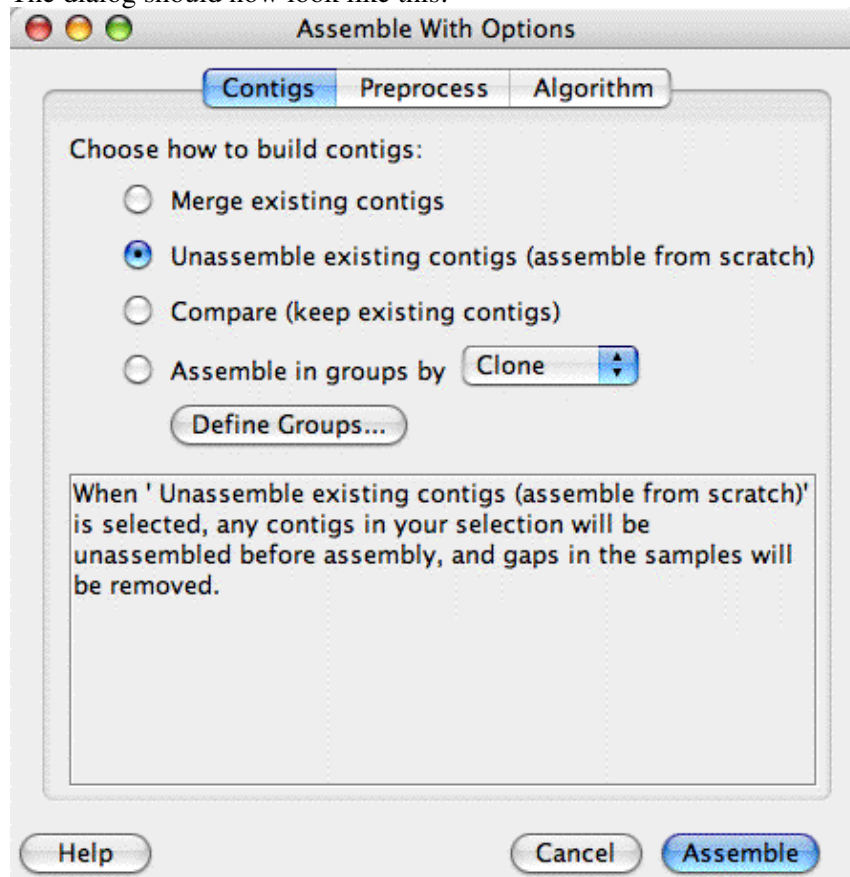
Sequence Assembly With Phrap

While CodonCode Aligner allows you to do assemblies with its own built-in assembly algorithm, Aligner also supports assembly with the assembly program Phrap (see [below for more information about Phrap](#)). Phrap's assembly algorithm is more sophisticated and better tested than Aligner's build-in assembly; Phrap is also better suited for large projects, and will often be faster. You will need to have Phrap installed on your computer to be able to assemble with Phrap, and the "Phrap Assembly" preferences must point to where Phrap is installed on your system.

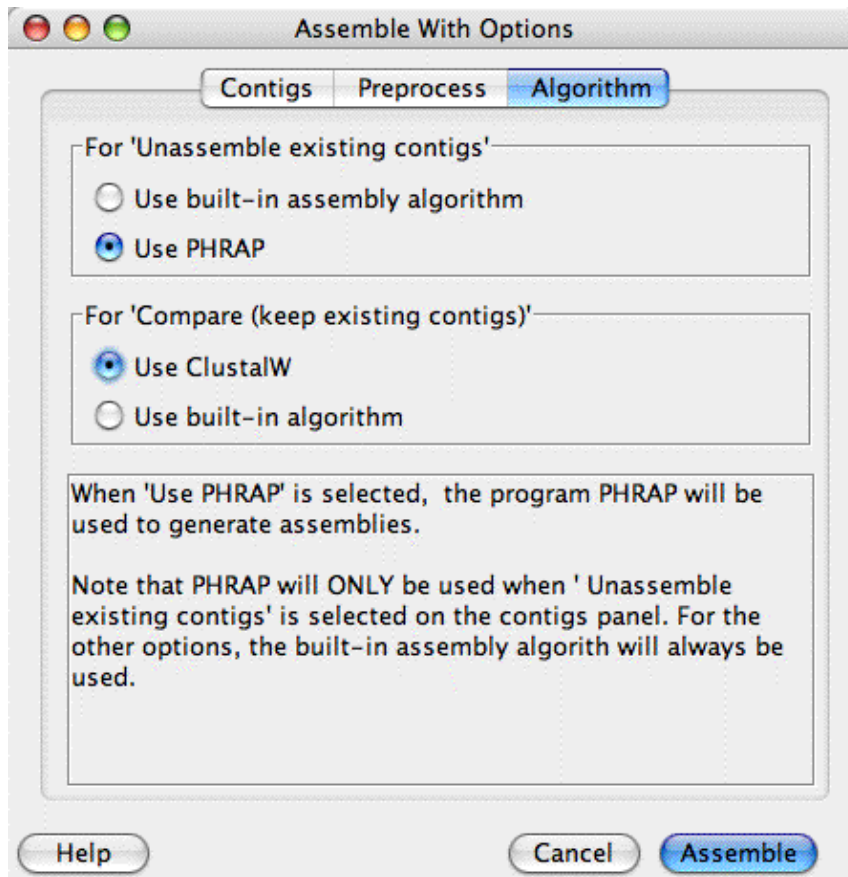
To assemble sequences with Phrap:

- Select the samples and/or contigs that you want to assemble in the project view.
- Choose "**Assemble with Options...**" from the "**Contig**" menu.
- On the dialog that appears, click on the radio button "**Unassemble existing contigs (assemble from scratch)**".

The dialog should now look like this:



- Click on the "**Algorithm**" tab.
- Select the "**Use PHRAP**" radio button. The dialog should now look like this:



- Click on the "Assemble" button.

The assembly will start and show a progress window. Depending on the size of your selection, your depth of coverage, and the number of repeats, the assembly may take a while; assemblies with several hundred reads typically take a minute or a few minutes.

How Aligner assembles using Phrap

Aligner uses Phrap to assemble your selection as follows:

1. Aligner checks if the Phrap program is indeed at the location specified in the "Phrap Assembly" preferences (for example, in the default location /usr/local/genome/bin on Mac OS X). If the Phrap program is missing, Aligner displays a warning and stops here.
2. Aligner exports the selected samples to a single sequence text file in FASTA format, and the corresponding qualities to a similar text file. If you selected any contig, then the samples in the contigs will be exported, not the contig sequence, since Phrap assumes that you always assemble "from scratch". The exported sequences will be without gaps.
3. Aligner now starts Phrap in a separate process, telling Phrap the name and location of the input files that were created in the previous step.
4. Aligner waits until Phrap is finished with the assembly. Aligner will display a progress dialog, but since it cannot be predicted how long Phrap assemblies will take, the progress window does **not** really indicate how far along the assembly is.
5. When Phrap is finished, Aligner will read the Phrap result file (the .ace file), and update the project based on the Phrap results. Any samples in your initial selection that Phrap did not include in contigs will be in the "Unassembled Samples" folder.

Please note that CodonCode Aligner is currently limited to projects with several hundred or at most a few thousand samples, even though PHRAP can handle much larger assemblies.

Things to Note for Phrap Assemblies

While Phrap typically produces very good assemblies, Phrap also sometimes produces results that can be puzzling to the novice Phrap user. Some of these things are:

- Phrap disregards previously formed contigs, and always assembles "from scratch" (this is different from Aligner's assembly method, which leaves pre-existing contigs as they are, and looks for overlaps between the contig sequences).
- Phrap sometimes generates contigs with only one read in it. This happens if the sample has a significant overlap with other samples, but the overlap was not good enough to justify a merging. Aligner will unassemble such single-read contigs, and move the samples into the "Unassembled Samples" folder.
- For very large assemblies (or on computers with low amounts of memory), Phrap may run out of memory. Before running out of memory, Phrap may need to use virtual memory, which can slow Phrap down dramatically. In extreme cases (for example trying to assemble a bacterial genome on an underpowered computer), this can even lead to system crashes.
- Phrap tries to identify different copies of repeats during assembly, based on the quality scores of samples. This means that assemblies will be better if you have accurate quality scores (rather than just "dummy" qualities). If your samples have high-quality discrepancies, for example because you are trying to assemble samples with homozygous mutations, this may lead to a higher number of contigs than you would expect.
- Occasionally, contigs created by Phrap will contain reads that are not aligned to the consensus sequence. You can manually remove these reads in Aligner, using one of the different "Move To.." options.
- Like every assembly program out there, Phrap will occasionally produce incorrect assemblies. For example, multiple copies of highly identical repeats may all be "piled up" on top of each other. You can use Aligner's interactive features to move such misassembled reads to the trash or the "Unassembled Samples" folder.

About Phrap

The assembly program Phrap was developed by Phil Green at the University of Washington. Phrap was widely used for sequence assembly in the Human Genome Project, and still is often regarded as the standard for sequence assembly. CodonCode Corporation distributes Phrap executables for a variety of platforms, including Windows and Mac OS X, under license from the University of Washington.

Academic users can obtain the source code for Phrap for restricted use directly from the authors at the University of Washington. For more information on this, please visit <http://www.phrap.org/>.

You can read the original documentation for Phrap at <http://www.codoncode.com/support/phrap.doc.html>. Please note that Phrap is a command line program, not a typical Windows or Mac OS X application! This makes it easy to run Phrap through scripts or from programs like CodonCode Aligner, but it means you cannot simply double-click on Phrap and expect to see a graphical user interface (there is none).

When CodonCode Aligner is installed, the installer also includes a special "workstation" versions of PHRAP, which can be only from CodonCode Aligner. To run PHRAP from CodonCode Aligner, you need either a trial license, or a purchased license. Licenses purchased for academic use allow the use of PHRAP free of charge.

Non-academic customers who want to use PHRAP must either install their own PHRAP executables, or pay a separate license fee for PHRAP.

Alignments to a Reference Sequence

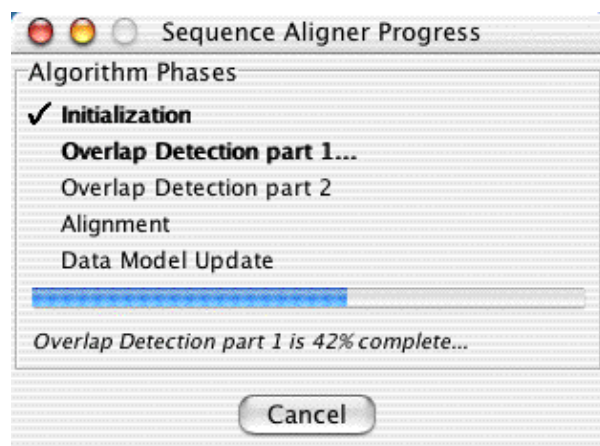
To perform an alignment to a reference sequence:

1. Open an existing project or create a new project (see [Creating Projects](#)).
2. Add the desired sample files (see [Adding Samples to a Project](#)).
3. Designate the **reference sequence**:
 - Select the reference sample in the project window
 - Choose "**Make Reference Sequence**" from the "**Sample**" menu
4. Select the reference sequence and the samples you want to align to it in the project view
5. Choose "**Align to Reference Sequence**" from the "**Contig**" menu

Note:

- To **select all unassembled samples**, simply click on the "Unassembled Samples" folder
- To make a **continuous selection**, keep the "**shift**" key pressed while clicking on the samples you want to select in the project view
- To make a **discontinuous selection**, press the "**control**" key (on Windows) or the "**command**" key (on OS X) while clicking on samples in the project view

Alignment begins immediately, and a progress window shows the status of the alignment.



When the alignment is done, you will see a newly formed contig in the project window. The new contig will contain a copy of the original reference sequence, leaving the original reference sequence unchanged.

Note that the new contig will be limited to the length of the reference sequence (plus any gaps introduced during the alignment). Any parts of sequences that extend beyond the start or end of the reference sequence will be marked as unaligned. You will not be able to see the overhanging unaligned in the contig view, but you can see them in the trace views and base views.

The alignment is performed using the parameters defined by the [alignment preferences](#). Samples that do not meet the minimum criteria will not be aligned, and remain in the "Unassembled samples" folder.

If you wish to return the sample files to their unaligned state, select the alignment contig in the project view, and then choose **Unassemble** from the **Contig** menu. When unassembling alignments, the copy of the reference sequence that was included in the alignment will be move to the trash, unless the original reference sequence was deleted or renamed.

Adding Samples to Alignments

To add new samples to existing alignments:

- Go to the project view
- Select the aligned contig, and the samples you want to add to it
- Go to the "**Contig**" menu, and choose either "**Align to Reference Sequence**"

The end result should be pretty similar in most cases. If you choose "**Align to Reference from Scratch**", the current contig will first be dissolved, so that you lose any manual edits. So if you have done any manual editing that you want to preserve, use "**Align to Reference Sequence**".

Instead of using the menu, you can also use drag and drop in the project view to add unassembled samples to an existing alignment: select the samples you want to add, and then drag and drop them onto the contig that you want to add them to.

You can also choose "**Align to Reference from Scratch**" with just a contig selected, as long as the contig contains a reference sequence.

Note: All tags that had been added to the consensus sequence, including tags added by Aligner's mutation finding, will be lost when you add sequences to an existing alignment. We hope to change this in a future release; if this is important to you, please let us know!

Advanced alignment options

- [Align from scratch](#) - this will unassemble any existing contigs in your selection before starting the alignments. This option can be useful to if you want to undo manual introduction or movement of gaps, or to try different alignment parameters.
- [Align in groups](#) - with this option, CodonCode Aligner can automatically group samples based on their **names**, and form separate contigs for each group. This option can be a real time saver in phylogenetic and medical studies where you look at sequences from many species, isolates, or patients.
- [Preprocess](#) - in addition to the choices above, Aligner can also automatically do common pre-processing steps like end clipping and vector trimming before assembly.

To use the advanced alignment options, select the samples you want to align, and then choose "**Align with Options...**" from the "**Contig**" menu. This will display a dialog where you can choose the different advanced options, as described below.

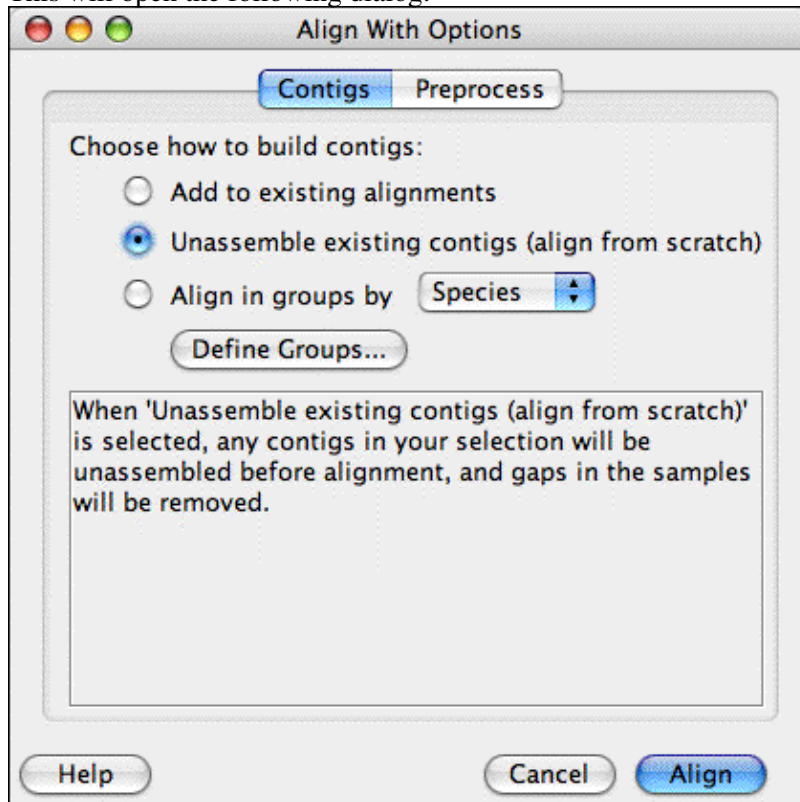
Align to reference from scratch

Sometimes, you may want to re-assemble a contig, for example after you did some editing and ended up messing up the alignment of reads. Or you may want to merge a contig with another contig and/or additional samples without preserving the existing alignment of reads in the contig.

To re-align one or more contigs from scratch:

- Go to the project view

- Select the contig, contigs, or contig(s) and sample(s) you want to assemble from scratch (*keep the shift-, control-, or command-key pressed to make continuous or discontinuous selections, as described [above](#)*)
- Choose "Align with Options..." from the "Contig" menu
- This will open the following dialog:



- Choose "Unassemble existing contigs (align from scratch)", then click on "Align"

Aligner will first unassemble the contigs you selected, and then align all the samples in the contig(s) and any other samples you had selected to your reference sequence(s). You will see a progress dialog while Aligner is aligning.

Note that the resulting number of contigs may be different from the initial number of contigs.

If your selection contained only unassembled samples, align from scratch will do exactly the same thing as "Align to Reference Sequence" would (unless you also have [preprocessing](#) options selected).

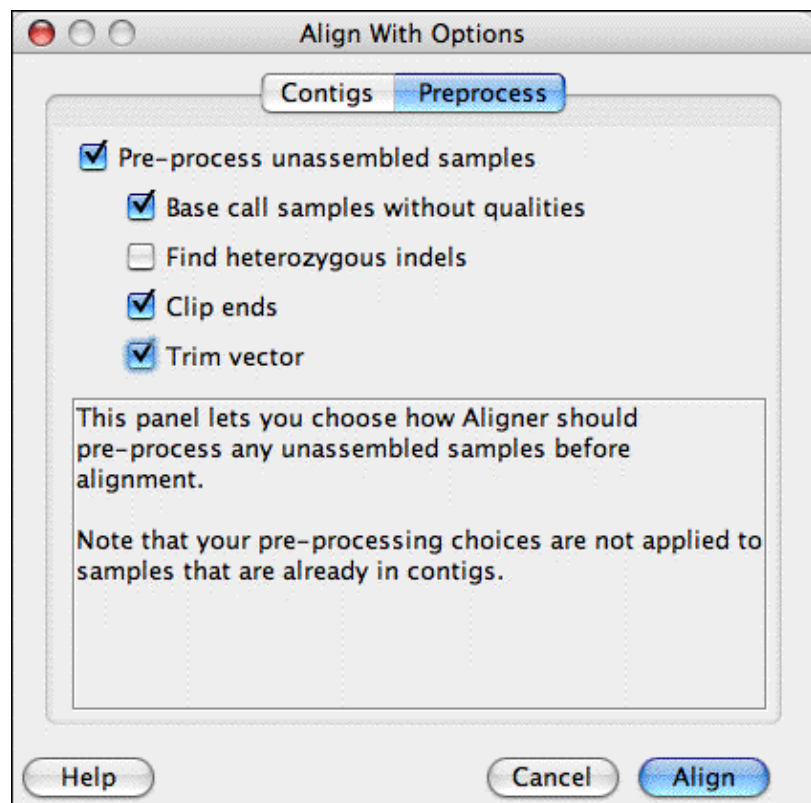
Align in groups

If you want to compare samples from a number of different sources (species, patients, isolates...) to a reference sequence, the "Align in groups" option in CodonCode Aligner can simplify your work. "Align in groups" uses sample names to group samples together, and to form separate contigs for each group. A copy of a reference sequence will be included in each contig.

For a description of how to use sample names to group samples, please see the ["Assemble in groups" help page](#).

Pre-process: One-step processing

In addition to the different assembly choices described above, "Align with Options" also lets you automate common pre-processing steps like end clipping and vector trimming. To set your pre-processing choices, click on the "Preprocess" tab:



The checkbox at the top determines whether or not your samples will be pre-processed before assembly; the four lower check boxes let you pick the pre-processing steps that will be done. The choices are:

- **Base call samples without qualities:** Any samples that have chromatograms, but no base-specific quality scores, will be base called with PHRED. To use this option, you will need either a trial license or a purchased license; base calling is not enabled in demo mode. Additional information about base calling can be found at the ["Base calling" help page](#). Please note that one important difference to the normal "Call bases" menu choice: in automated pre-processing, samples that already have quality values, for example from the ABI KB basecaller or from calling bases on the sample before, will not be base called again.
- **Find heterozygous indels:** This option will look for potential heterozygous insertion/deletion (indel) mutations in the unassembled samples that have (a) chromatograms and (b) quality scores. If you are sequencing PCR products from genomic DNA that may contain heterozygous indels, you should check this option; if you are sequencing from cloned DNA, this option should not be checked. For more information, please read the ["Heterozygous insertions and deletions" help page](#).
- **Clip ends:** This will remove low-quality sequence from samples that have chromatograms and quality scores. For more information, please read the ["End clipping" help page](#).
- **Trim vector:** When checked, this option will identify and remove vector sequence contamination from the samples in your selection. You may need to set your [vector trimming preferences](#) before using this option. For additional information, please read the ["Vector trimming" help page](#).

CodonCode Aligner User Manual

You will see progress dialogs for each of the steps performed after you click the "Align" button.

Note that any samples that are already in contigs will **not** be pre-processed.

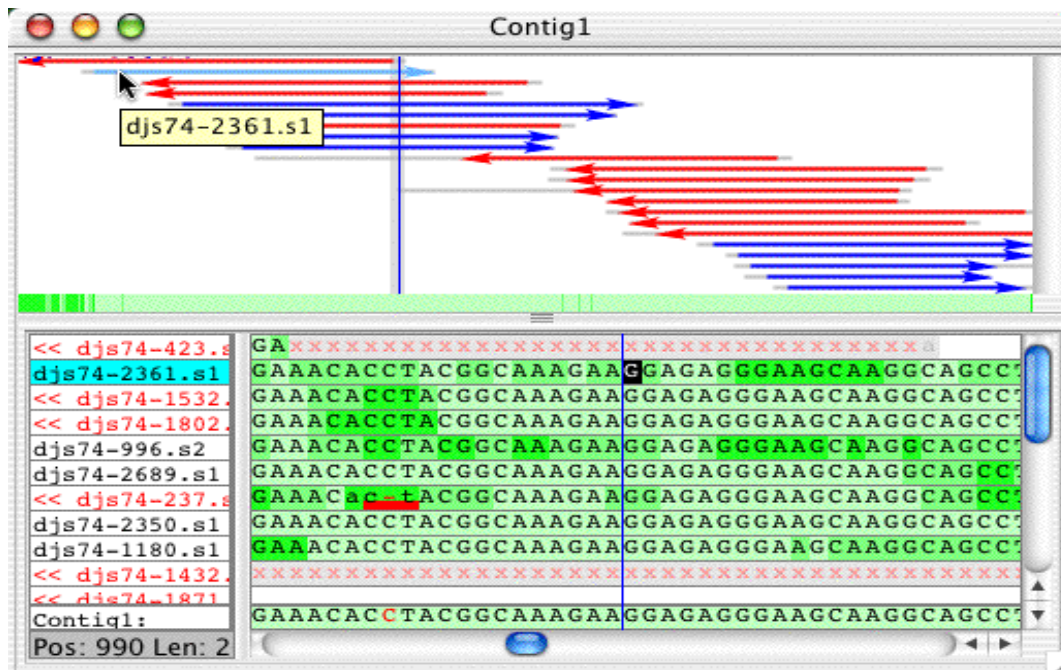
Contigs

Successful assemblies or alignments result in one or more contigs. Contigs are named "Contig1", "Contig2", and so on, and shown as folders in the project view.

If you select a contig in the project view, and then double-click on it (or choose "Contig" from the "View" menu), you will see the [contig view](#), which shows a graphical overview as well as the aligned bases.

You can edit samples in contigs, as described in the "[Editing Samples](#)" section, and you can edit contigs as described in the "[Editing Contigs](#)" section.

Here is an example of a [contig view](#):



Unassembling Contigs

To dissolve an existing contig and return the samples in it to the "Unassembled Samples" folder, select the contig in the project view, and choose "**Unassemble**" from the "**Contig**" menu (or click on the "Unassemble" button in the project window toolbar). Any gaps created by Aligner are removed, and samples that were reverse complemented are returned to their normal state.

You can also unassembled contigs by first selecting the contig in the project view, and then dragging & dropping the contig onto the "Unassembled Samples" folder. Before unassembling contigs this way, Aligner will show a warning dialog, asking you if you really want to unassemble the selected contig(s).

Aligner Algorithms for Assembly and Alignments

CodonCode Aligner uses a fast, banded dynamic programming (Smith-Waterman) algorithm for pairwise alignments in both assembly and alignments. By default, CodonCode Aligner version 2.0.1 and newer will use "[end to end](#)" alignments; however, you can instead choose to have "[large gap](#)" or "[local](#)" alignments.

This section describes how Aligner generates assemblies and alignments to reference sequences.

Alignment to a Reference Sequence

When aligning to a reference sequence, the selected samples are successively aligned to the reference sequence. Alignments that meet the stringency criteria set in the [alignment preferences](#) are accepted, those that do not meet the minimum criteria are rejected. If your selection contains more than one reference sequence, Aligner will align each sample to the reference sequence with which is shared the most "words" (typically 8-mers) first; alignments to other references will be tried only if this alignment is reject. Typically, this will result in alignments to the reference sequence with the highest similarity; however, there can be exceptions to this rule.

When aligning to a reference sequence, any sequence parts that extend beyond the ends of the reference sequence are not included in the alignment.

Assembly

For assembling contigs, the following simple "greedy" algorithm is used:

1. Find potential overlaps between samples by looking for shared 12-nucleotide "words" in the sequence.
2. Find the pair of samples that has the highest number of shared words.
3. Perform a pairwise alignment between the two samples (or contigs in later cycles).
4. If the alignment is good enough, keep it as a new contig, and calculate the consensus sequence; otherwise, reject the merger, and leave the two samples separate.
5. Find the next pair of sequences, looking again for the highest number of matching words. If a sample is in a contig, use the consensus sequence for the contig. If the two samples are already in the same contig, get the next pair.
6. Go back to step 3., and continue the pairwise joins until all possible joins have been tried, or until the maximum number of merge failures in a row has occurred.

The algorithm typically performs well enough for projects containing up to several hundred reads (possibly more, if you have enough memory available and a fast computer). However, it has a few weaknesses, most of which are typical for such greedy algorithms:

- If a project contains multiple copies of a repeat with high identity, there is a good chance that they are mis-assembled (but Alu sequences are generally not a problem)
- Samples have to share at least one perfect word match to be considered for joining. This means very short overlaps, or short overlaps with errors or ambiguities, may not be found. The default word length is 12 bases, but this can be changed in the [assembly preferences](#).
- Information from double-ended sequencing or other ordering information is not used to generate the assembly, which can lead to incorrect assemblies, especially in larger projects containing high-identity repeats.

You can adjust the stringency requirements for successful mergers in the [Assembly preference panel](#). (Note that the assembly preferences apply only to assemblies generated with the built-in algorithm, not for assemblies with PHRAP or contig comparisons generated with ClustalW or muscle).

Consensus Calculation

CodonCode Aligner offers three methods for determining the contig ("consensus") sequence:

1. [Quality-based consensus sequences](#). This is the preferred method for most assembly projects, since it gives the most accurate consensus sequence. It requires that sequences have [base-specific quality scores](#), for example from [base calling](#) with Phred, which can be done directly from Aligner.
2. [Majority consensus](#). The majority consensus simply counts all bases at a given position. If one base (or gaps) are more than 50% of all calls, the base (or gap) is used as the consensus; otherwise, and ambiguity code for the bases present is used (this description is a bit simplified, more details [below](#)).
3. [Inclusive consensus](#). The inclusive consensus considers all aligned bases at a given position, and uses the [IUPAC ambiguity code](#) that represents all bases present at a given location.
4. [Using the reference sequence as the consensus](#). For contigs that were created by aligning to a reference sequence rather than by assembling, you can choose to have the reference sequence be used as the consensus sequence. This can be useful in mutation detection and clone verification projects.

You can set which consensus method is used in the [consensus preferences](#).

Quality-based Consensus

To determine the consensus base and quality score for assemblies where the samples have qualities, CodonCode Aligner tries to emulate what human "finishers" typically do: Aligner identifies the highest quality base at each position, taking into account confirmations by reads in the opposite direction. Slightly simplified, here is the algorithm in detail:

1. For each possible nucleotide (A, G, C, T), find the sample with this base at the given position that has the highest quality. Take this score as the initial consensus score for the nucleotide (use 0 if no sample has this nucleotide).
2. Find the highest quality confirming base from a read in opposite direction, and add the score of the confirming base.
3. Optionally, find the highest quality discrepant base, and subtract the score. *This subtraction can be turned on or off in the "[Consensus](#)" preferences. The default setting is to not subtract the discrepant score.*
4. Calculate the score for a gap in the consensus the same way, using gaps at this position in the individual samples. The quality scores of gaps in samples are taken as the average of the two neighboring bases.
5. Pick the nucleotide (or the gap character) that has the highest quality. Assign the calculated confirmed quality score to the consensus sequence. The maximum quality score assigned to a consensus base is 90.

Several points to remember when trying to understand this algorithm:

- The base quality scores are error probabilities on a logarithmic scale; therefore, the error probabilities can simply be added *as long as the probabilities are independent*.
- Error probabilities for confirming reads in the same direction are **NOT** independent, since most sequencing errors are due to systematic problems like polymerase stops, GC compressions, etc.

Therefore, qualities of confirming reads in the same direction **cannot** simply be added.

- Arguments can be made for and against subtracting the quality scores of discrepant bases. Some scientists feel that having a discrepant base should make a difference in the consensus quality, while others use statistical arguments why qualities of discrepancies should not be subtracted. You can decide for yourself, and change this in the "Preferences" (*if you have a confirmed high-quality base and a low-quality random discrepancy, the difference in the consensus quality will be zero or very small, anyway*).

The algorithm used by Aligner is similar to the algorithm used by the assembly program Phrap, but it is not identical. Phrap's algorithm is more complicated, and uses (for example) "confirmed sequence segments" rather than individual bases. However, the quality scores assigned by Aligner will often (but not always) be identical to the scores Phrap would assign to the same assembly.

Majority Consensus

The majority method will be used to make the consensus sequence if you unchecked "Use quality-based consensus whenever possible" in the [consensus preferences](#) (unless the contig is an alignment, and you selected to use the reference sequence to build the consensus).

The short description: Basically, Aligner will use the most common base at each position as the consensus, unless no base accounts for more than 50% of the base calls there, in which case an ambiguity code will be used. Well, this is actually a bit simplified - here is how the majority consensus is determined in detail:

The long description:

First, Aligner looks at all the bases and gap characters in the aligned parts of all samples at this consensus position. If more than 50% of the samples have a gap here, the consensus will be a gap. Otherwise, the gaps will be ignored for the following analysis.

If none of the base calls are ambiguities here, the rest is simple. If one base (A, G, C, or T) accounts for more than 50% of all non-gap base calls here, it is used as the consensus base. If no base accounts for more than 50%, a IUPAC ambiguity character is used, based on all the base calls here. The IUPAC ambiguity codes are:

Ambiguity Code	Bases
M	A or C
R	A or G
W	A or T
S	C or G
Y	C or T
K	G or T
V	A or C or G
H	A or C or T
D	A or G or T
B	C or G or T
N	G or A or T or C

If at least one of the base calls in the samples at this position already is an ambiguity code, determining the majority consensus is done as follows:

Each regular base call here gets a score of two. For ambiguity codes, each base represented by the code gets a score of one (for example, at a "M" call, both A and C would get a score of one). The scores for all samples

that have aligned bases at this position gets summed up. Now, if one of the four possible bases (A, G, C, or T) got more than 50% of the total score, this base is used as a consensus. Otherwise, an ambiguity code that represents all base calls at this position is used.

For example, if there are just two samples, one "A" and one "W" (meaning A or T), then "A" will be used for the consensus. If one base is "A" and the other "B", then "N" will be used for the consensus.

For sequencing projects where the goal is to determine the correct sequence of a gene, a quality-based consensus sequence will generally be better than a majority-based sequence. For example, if a region is covered by only two samples, the majority consensus will contain ambiguity codes at all discrepancies, while the quality-based consensus will pick the higher quality base, which is most often the correct one. However, a majority-based consensus can make sense for other types of sequencing; one example is genotyping a number of samples, where the majority sequence will show the most common allele.

Using the Reference Sequence as Consensus

When you are comparing samples to a known reference sequence, you may not be interested in building a consensus sequence - you just want to know where your samples differ from the reference. Aligner supports this by letting you use the reference sequence as the consensus sequence. To use the reference sequence as the consensus sequence:

1. In the [consensus preferences](#), select "Use the reference sequence as the consensus sequence", then click "OK".
2. Create your contig by "**Align to Reference Sequence**", not by "Assemble" or "Assemble with Phrap".

Note that assembled contigs may contain the reference sequence - that alone does not mean that the contig is an alignment to a reference sequence. Only contigs that were created by "**Align to Reference Sequence**" can be alignments.

Note that if you merge an aligned contig with other contigs by using "**Assemble**" or "**Assemble with Options...**", the contig will change from an alignment to a regular (assembled rather than aligned) contig; however, you can add unassembled samples to existing alignments without changing contig to a regular (assembled) contig.

Tips for dealing with alignments:

- To find out if a contig is an alignment, look at its icon. Alignments icons have two vertical lines in the folder. Alternatively, you can check the contig information dialog. To see it, go to the project view, select the contig, and then choose "**Contig Information**" from the "**Contig**" menu.
- To convert a contig that contains a reference sequence to an alignment, you can select the contig in the project view, and then select "Align with options" from the "Contig" menu; in the dialog that follows, choose "Align to reference from scratch". However, the resulting contig may be different from the initial contig; for example, not all reads may have been added - some reads may have been moved to the "Unassembled Samples" folder.

Local, large gap, and end-to-end alignments

By default, CodonCode Aligner version 2.0.1 and newer will perform "**end to end**" alignments. Assemblies and alignments generated this way will always end at the end of one sequence, not before. In contrast to the local alignment algorithm, end-to-end alignments will never generate dangling (unaligned) ends. When

using this algorithm, samples should always be end clipped, and if necessary also vector trimmed.

When using the "**Local alignment**" algorithm, Aligner automatically uses the maximum amount of each sequence, without requiring end clipping before assembly. When using local alignments, the alignment only extends as far as the alignment scores improve. This means that low-quality ends of sequences, where high error rates would reduce the alignment score, are left unaligned, or "dangling".

Unaligned parts of sequences are shown on light gray background in the contig view; they alignment end points can manually be changed using "Mark > Start Alignment Location" and "Mark > End Alignment Location" in the "Sample" menu.

The local alignment algorithm was used by default in CodonCode Aligner version 1.6.3 and older. If you started using CodonCode Aligner before version 2.0.1, the local alignment algorithm is probably still being used, unless you changed you assembly and alignment preferences.

When working with samples that have large insertions or deletions, or if you are trying to align **cDNA to genomic DNA**, you should change the alignment algorithm to "**Large gap**" alignment in the [assembly preferences](#) and/or the [alignment preferences](#). This will allow the alignment of multiple parts in one sequence to different sections in a second sequence - for example exons in a cDNA sequence to the corresponding regions in a genomic DNA.

Regions of Interest: Features

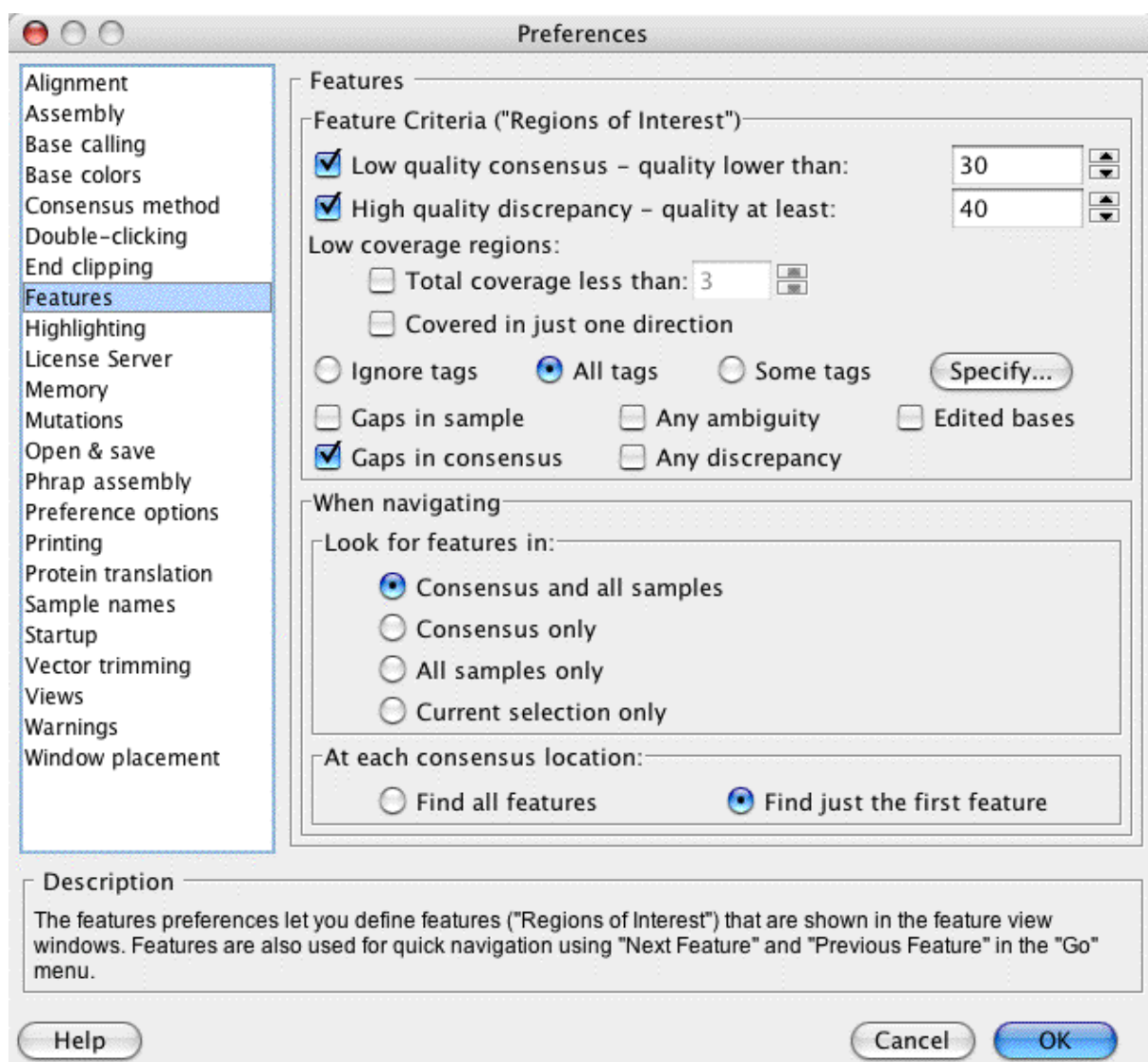
When looking at sequences or contigs, you often may want to not look at the entire sequence, but rather move quickly to specific "regions of interest". Of course, what is interesting to you will depend on the kind of project you are doing. For example, if you are assembling a shotgun sequencing project, you may be interested in regions of low coverage or low consensus quality; but in a mutation analysis project, you may instead be interested only potential homo- or heterozygous mutations.

To enable you to quickly find and navigate to *your* regions of interest, CodonCode Aligner allows you to define "features". You can define a variety of different things as features, for example gaps in sample or contigs, discrepancies, low coverage regions, or tags that have been added by programs like PolyPhred. After defining features, you can:

- view all features in a separate window (the "feature view")
- quickly move to the next or previous feature in a contig
- export features to text files for later analysis

Defining Features

To define feature criteria, select "**Define Features...**" from the "**Go**" menu in CodonCode Aligner. This will open the preferences dialog with the "Features" panel selected:



You can change the definition of features here by using the various buttons and text fields. For a more detailed explanation, check the [feature preference help page](#).

Moving to Features

You can quickly move from feature to feature in contig views by using the "**Next Feature**" and "**Previous Feature**" menu items in the "**Go**" menu.

To try this out, open a contig view, for example for the contig in the "Example1" project (in the "Example" folder in the CodonCode Aligner folder). Then, select the "**Go**" menu. The item we want to use is "**Next Feature**". Notice the keyboard shortcut shown in the menu - it is **Command-Right** arrow on OS X, and **Control-Right** arrow on Windows. Similar, the shortcut for "Previous Feature" is Command-Left arrow respectively Control-Left arrow.

Try the "Next Feature" keyboard shortcut out from the contig view. You will notice that the the cursor and selection in the contig view move to the right, sometimes by a few bases, sometimes by many bases. To see what kind of feature you just navigated to, look at the project view window. The status panel at the bottom

shows a brief description of the feature (in our example, "Feature: Low Quality Consensus" twice in a row, then "Feature: Discrepancy").

Use the keyboard shortcuts to move forward and backward between features. You can also try changing the definition of features in the preferences.

View, Print, Export Features

To get an overview of features in one or more contigs, select the contig(s) in the project view, and then choose **"Feature View"** from the **"View"** menu. This will open a feature view window, showing a table of all features in the contig or contigs. You can double-click on any feature to take a closer look at the feature, or print the feature view; for more information, please read the [feature view help page](#).

You can also export features from a selection of contigs or all contigs in a project to text files, as described on the ["Exporting features"](#) help page.

Finding Mutations

When working with sequence traces from genomic PCR sequencing, CodonCode Aligner can help you to find and analyze homozygous and heterozygous mutations ("SNPs"), both point mutations and insertions and deletions (*of course, you can also look for homozygous mutations by simply defining discrepancies and gaps as features*).

Prerequisites

To find point mutations, the following pre-requisites must be met:

- Any **sequences** to be analyzed **that have chromatograms must base-specific quality scores**.
You can use the [base calling](#) with PHRED in Aligner to get quality scores, if needed.
Sequences without traces (text sequences) do not need to have quality scores; if they have quality scores, the quality scores will be used to ignore low-quality sequence at the ends.
- The **sequences must be in a contig**.
The contig can be generated with Aligner by assembling or aligning to a reference sequence, or it can be from importing an assembly. Only samples that have sequence traces as well as base-specific quality scores can be analyzed.

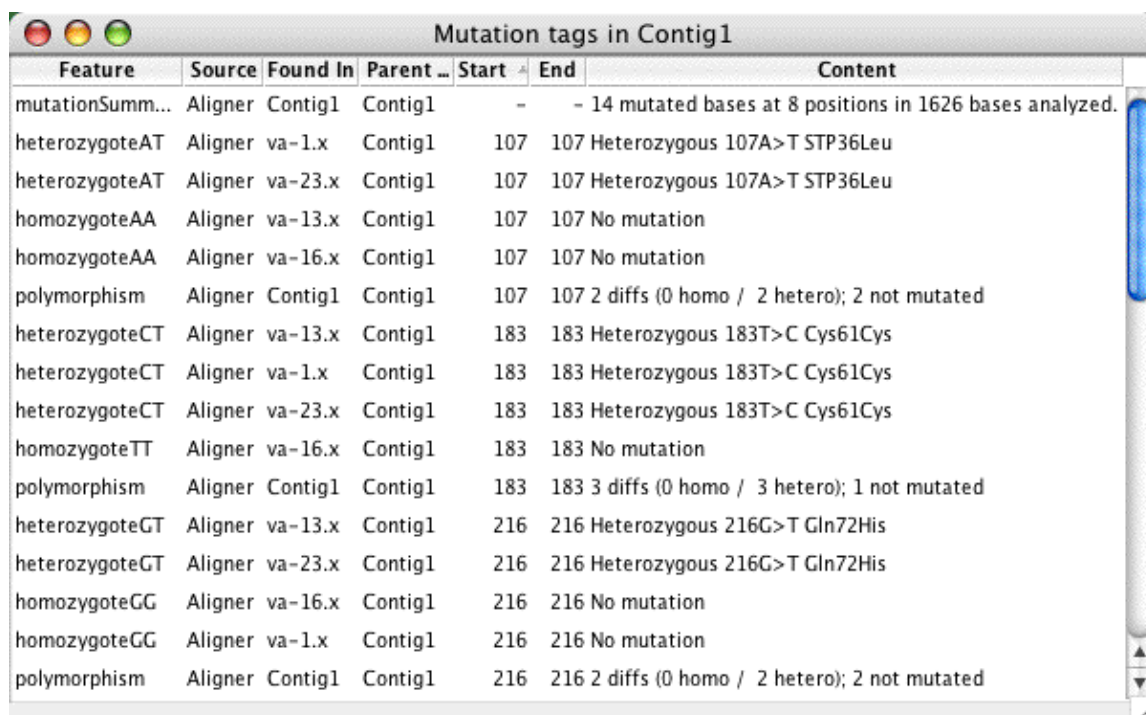
Detection of point mutations in Aligner is intended for re-sequencing projects only. The sequences to be analyzed should have the same sequencing chemistry and primers; mixing different chemistries, for example dye primer and dye terminator sequencing, will likely result in many false identifications.

How To Find SNPs

- Go to the project view
- Select the contig or contigs you want to analyze
- Choose "**Find Mutations**" from the "**Contig**" menu

Finding mutations should take only a few seconds for small projects; Aligner will show a progress dialog, which allows you to cancel if things take too long. When the finding of heterozygous indels is done, Aligner will open a new window that shows the analysis results in a table like the one shown below:

CodonCode Aligner User Manual




Feature	Source	Found In	Parent ...	Start	End	Content
mutationSumm...	Aligner	Contig1	Contig1	-	-	14 mutated bases at 8 positions in 1626 bases analyzed.
heterozygoteAT	Aligner	va-1.x	Contig1	107	107	Heterozygous 107A>T STP36Leu
heterozygoteAT	Aligner	va-23.x	Contig1	107	107	Heterozygous 107A>T STP36Leu
homozygoteAA	Aligner	va-13.x	Contig1	107	107	No mutation
homozygoteAA	Aligner	va-16.x	Contig1	107	107	No mutation
polymorphism	Aligner	Contig1	Contig1	107	107	2 diffs (0 homo / 2 hetero); 2 not mutated
heterozygoteCT	Aligner	va-13.x	Contig1	183	183	Heterozygous 183T>C Cys61Cys
heterozygoteCT	Aligner	va-1.x	Contig1	183	183	Heterozygous 183T>C Cys61Cys
heterozygoteCT	Aligner	va-23.x	Contig1	183	183	Heterozygous 183T>C Cys61Cys
homozygoteTT	Aligner	va-16.x	Contig1	183	183	No mutation
polymorphism	Aligner	Contig1	Contig1	183	183	3 diffs (0 homo / 3 hetero); 1 not mutated
heterozygoteGT	Aligner	va-13.x	Contig1	216	216	Heterozygous 216G>T Gln72His
heterozygoteGT	Aligner	va-23.x	Contig1	216	216	Heterozygous 216G>T Gln72His
homozygoteGG	Aligner	va-16.x	Contig1	216	216	No mutation
homozygoteGG	Aligner	va-1.x	Contig1	216	216	No mutation
polymorphism	Aligner	Contig1	Contig1	216	216	2 diffs (0 homo / 2 hetero); 2 not mutated

The table shows the characterization of samples at each consensus base where Aligner found at least one mutated base; Aligner has added tags at these points to each sample that describe Aligner's classification. For example, at consensus base 183 in the table above, Aligner classified three samples as heterozygous C-T mixes, and one sample as homozygous T-T.

Use the [mutation preferences](#) to adjust the mutation detection sensitivity, whether Aligner should add tags to samples that are not mutated, and so on. You can [use tags](#), for example a "codingSequence" tag assigned to the reference sequence or "dontGenotype" tags, to fine-tune which regions Aligner analyzes, and how the effect of mutations is described in the "Content" field; this is described in detail [below](#).

To take a close look at the results, you can double-click on any entry in the table. This will bring up the views you have chosen in the [double-click preferences](#) - typically the contig view and the trace view. A screen shot of the contig view for example above is shown below:



Contig1	
va-1.x	ACACTCAGAGCTGCAGGAATCTGTCACCTCCTGTTTGCTGCTCACC
va-13.x	ACACTCAGAGCTGCAGGAATCTGTCACCTCCTGTTTGCTGCTCACC
va-16.x	ACACTCAGAGCTGCAGGAATCTGTCACCTCCTGTTTGCTGCTCACC
va-23.x	ACACTCAGAGCTGCAGGAATCTGTCACCTCCTGTTTGCTGCTCACC
Contig1:	ACACTCAGAGCTGCAGGAATCTGTCACCTCCTGTTTGCTGCTCACC
Pos: 183 Len:	

The tags added by Aligner are shown as blue and pink boxes - blue boxes indicating homozygous bases, and pink boxes indicating heterozygous bases (unless you changed the display of tags in the [highlighting](#)

[preferences](#) to something other than "box").

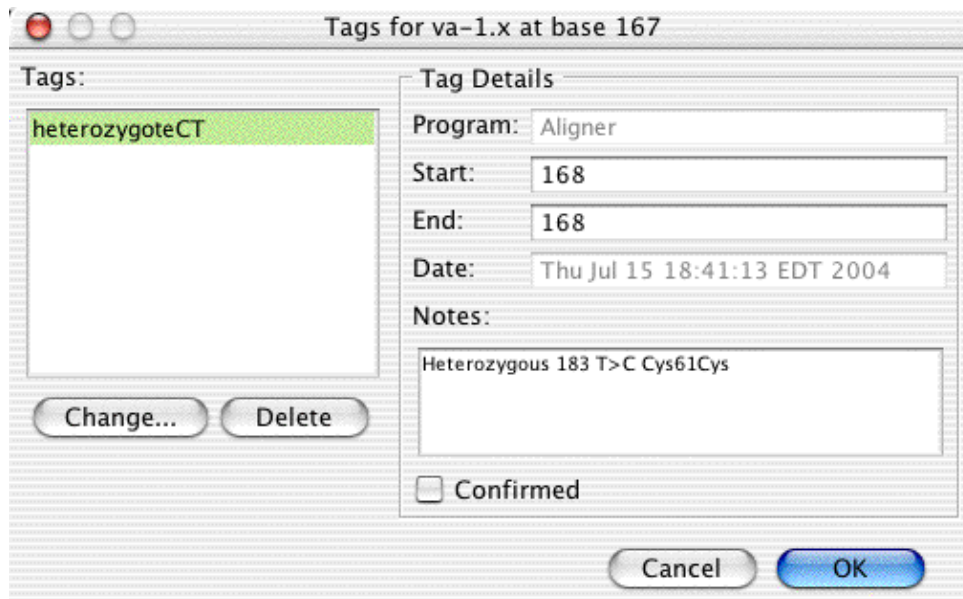
The corresponding trace view for the four samples is shown below:



Note that all three traces that were characterized as heterozygous C-T show a two peaks, a blue C peak and a red T peak (you colors may be different, depending on your [base color preferences](#)). Also note that the T peak in the heterozygous samples is only about half the height of the T peak in the homozygous sample at the top. Aligner uses these two pieces of information, the secondary peak and the reduction in peak intensity, to identify heterozygous bases.

To get more information about a tag, you can right-click (OS X: control-click) on the base with the tag, and select **"Show Tag: ..."** from the popup menu (the menu text will show the type of the tag selected, for example **"Show Tag: heterozygoteCT..."**).

Selecting the **"Show Tag: ..."** popup menu item will display the following tag dialog:



You can quickly confirm a tag by clicking on the "Confirmed" checkbox, delete the tag, or add your comments in the "Notes" text area.

Fixing Errors

There will be cases where you disagree with the classification made by CodonCode Aligner, and want to change it. Depending on the circumstances, this is what you can do:

- **False positives:**

If you think that a base is not mutated, you can change the tag to a "false positive" tag. The fastest way to do this is by using the popup menu from the trace view or contig view. Right-click (OS X: control-click) on the base with the wrong tag, and then choose **"Mark heterozygoteAC as False Positive"** from the popup menu. If you did click on a base with a mutation tag, this will be the second item in the popup menu. The actual text will differ a bit, depending on the original classification. False positive tags get preserved when you re-do the mutation detection.

- **Wrong classifications:**

If a base is mutated, but Aligners classification is incorrect, you can change the tag in a two-step process. First, bring up the tag information dialog, as described just above (right-click resp. control-click on the base, select **"Show tag..."** from the popup menu). In the tag dialog, click on the **"Change..."** button. This will bring up another dialog where you can choose the correct tag from a pulldown menu. This allows you to (for example) change a tag from "heterozygote CT" to "homozygoteCC".

When you change a tag this way, Aligner will update the notation that shows the amino acid effect in the "Notes" section of the tag (unless you have annotated a coding region, and the mutation is in a non-coding section of you sample).

- **Missed mutations:** If Aligner missed a mutation, you can manually add a tag that describes the mutation. In a trace view or contig view, right-click (OS X: control-click) on the mutated base, and then choose **"Add Tag..."** from the popup menu. This will bring up the "Add tag" dialog (an example is shown a bit further below). Select the tag type for your mutation from the pullup menu near the top - for example "heterozygoteCT" or "homozygoteAA". Aligner will show the amino-acid effect in the "Notes" section. Click "OK" when you're done.

If Aligner misses a lot of mutations in your project, or if finding *all* mutated bases is important to you,

you may want to make sure that you have set the mutation detection sensitivity to "High" in the [mutation detection preferences](#).

The three actions described above (Show tag, Add tag, and Mark as False Positive) are also available from the "**Sample => Tag**" menu - but usually, using the popup menu as described will be faster and easier.

Please note that any mutation tags that you change or add, except for "False Positive" tags, are usually deleted when you repeat "Find Mutations" for the same contig.

Tags for Finding Mutations

To fine-tune mutation detection, you can use tags to:

- [Define the coding region](#) in your consensus or reference sequence
- [Exclude regions](#) in samples or the contig from mutation finding

Defining the Coding Region

You probably noticed that the tag content contains a summary of the mutation which also indicates the amino acid change ("183 T>C Cys61Cys" in the example above). Unless to give Aligner more information, this annotation will be relative to the first base in the consensus sequence. However, the coordinates used will be **ungapped** (after removing any gaps from the consensus), while other coordinates shown by Aligner are typically **gapped** (they include the count of gaps in the consensus). In the example above, the gapped and ungapped coordinates are identical, since the consensus does not contain any gaps.

The amino acid change (or lack of change) is shown for the first of the three possible forward reading frames. In your own analysis, that may not be what you want - your coding sequence may start at base 2 or 3, or even further in, for example because you included a part of the sequence before the start codon in your reference sequence.

To define where the coding region is, you can add a "codingSequence" tags to the consensus or to your reference sequence.

If you have a reference sequence, you should:

- add "codingSequence" tags to the reference sequence (not to the consensus sequence).
- create your contigs by [aligning to a reference sequence](#), not by assembling (you can use the "Contig information" menu item in the "Contig" menu to verify this if in doubt).

For contigs formed by alignments to reference sequences, the coding sequence tags will always be taken from the reference sequence, not from the coding sequence. If your original sequences were in Genbank or EMBL format that had coding sequence annotation, this annotation will be used. Otherwise, you can add coding sequence tags manually.

To add a "codingSequence" tag:

- Open a contig view for the contig you are working with (you can also use a base view)
- Select the first base of your coding sequence in the reference sequence (or the consensus sequence)
- Right-click (OS X: control-click) to show the popup menu, and select "**Add Tag...**" (or select "**Add Tag..**" from the "**Tag**" sub-menu in the "**Sample**" menu)

This will bring up the "Add tag" dialog:

Add tag to Contig1

New Tag Type:
codingSequence ▼

Tag Details

Program: User

Start: 62

End: 99999

Date: Thu Apr 29 11:03:05 EDT 2004

Notes:

Tip:
If you want the tag to extend to the end of the contig or sample, just enter a very large number in the "End:" field.

☐ Confirmed

Cancel OK

Select "codingSequence" as the tag type from the pull down menu at the top. Then enter the end coordinate of your coding sequence at the bottom (you could also select the entire coding sequence before choosing "Add Tag..."). When everything looks right, press "OK".

If the first base of your coding sequence is NOT the first base of a codon, you can also add a "codonStart" tag; the codonStart tag must be within two bases of the codingSequence start, though.

If your reference sequence was originally read from a file in Genbank format, and contained a simple "CDS" annotation, Aligner will use this to automatically create "codingSequence" and "codonStart" tags. However, any "CDS" tags in Genbank sequences that point to other sequences or join multiple sequences will be ignored.

You can add as many "codingSequence" tags as you like. You can add the "codonStart" tag only to one of the first three bases in the first codingSequence; if you try to place it anywhere else, it will be ignored.

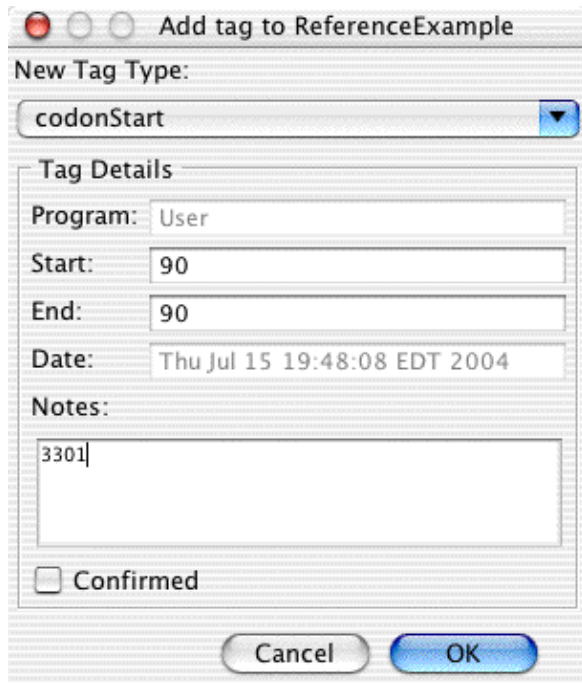
After defining your coding region, Aligner will use this information next time you choose "Find Mutations" for this contig. There are a few things to keep in mind, though:

- The content of mutation tags is not updated when you change your reference sequence or consensus (you can simply find mutations again to get the current results)
- If both the reference sequence and the consensus have a "codingSequence" tag, the tag from the consensus will be used
- The "codingSequence" tag from the reference sequence will only be used if the contig you are analyzing is an alignment, not an assembly

Numbering the Coding Sequence

In mutation analysis projects, you will often want to assign a base number to the first base in your coding sequence - for example because it is not the first exon in a gene. In Aligner, you can do this by adding a "**codonStart**" tag, and entering the corresponding nucleotide number in the "Notes" field. The codonStart tag identifies the first base of the first complete codon, and therefore must be assigned to any of the first three bases in the first coding region (see [Notes](#) below).

Here is an example of what the "Add Tag" dialog looks like in this case:



Add tag to ReferenceExample

New Tag Type:
codonStart ▼

Tag Details

Program: User

Start: 90

End: 90

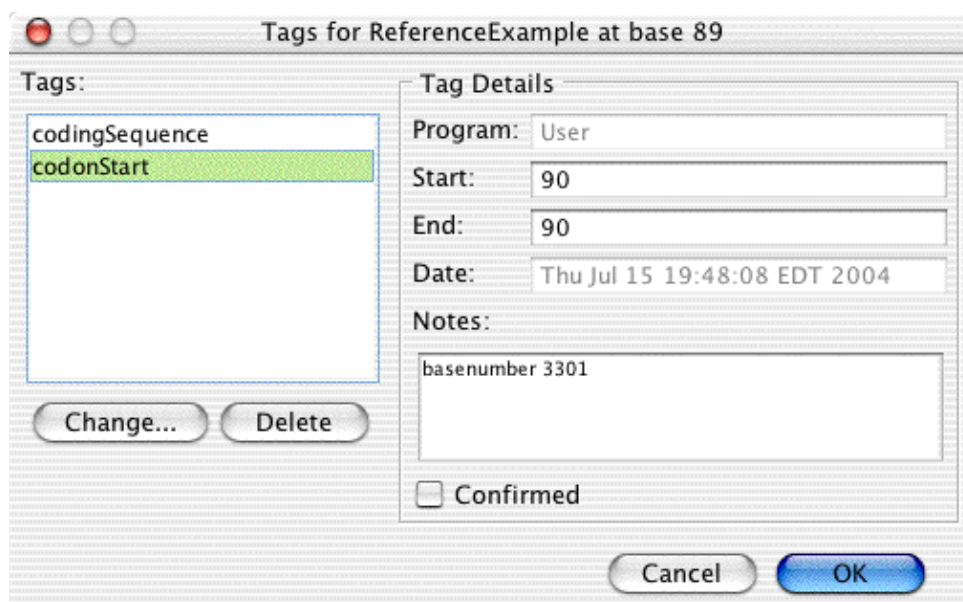
Date: Thu Jul 15 19:48:08 EDT 2004

Notes:
3301

☐ Confirmed

Cancel OK

If you prefer, you can write "basenumber 3301" (or your number, of course!), or "basenumber=3301" in the "Notes:" field. You do not have to enter this information when adding the tag, you can do it later by editing the tag in the "Show tag" dialog:



The number you set this way for the codonStart tag is the nucleotide number of at this position.

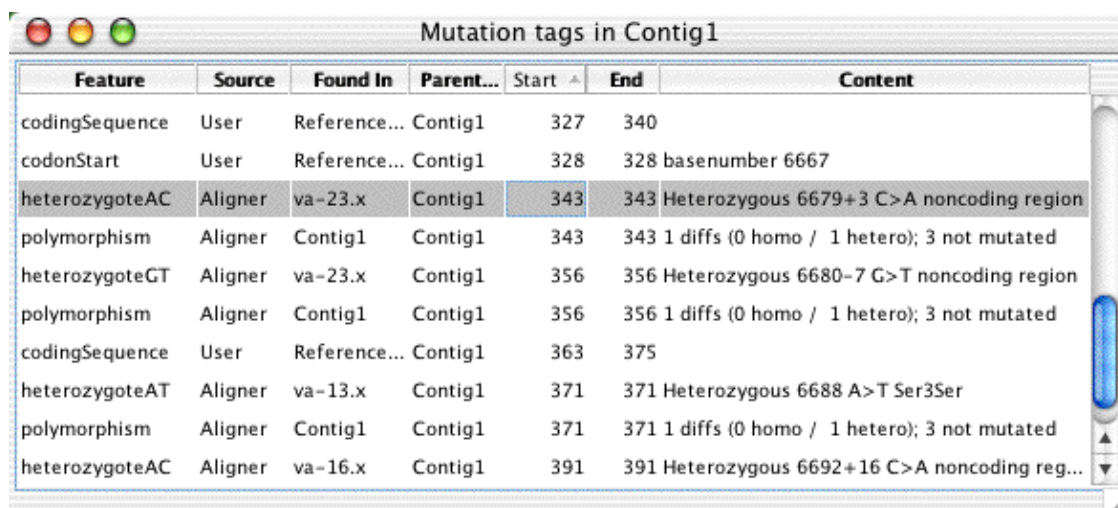
Note: Keep in mind that the codonStart tag identifies the start of the first **complete** codon in this exon. Therefore:

1. You **must** add "codingSequence" tags **before** adding the "codonStart" tag.
2. The codonStart tag must be in the first three bases of a "codingSequence" tag.
3. You can add **only one** codonStart tag per gene in your reference sequence.
4. If your reference sequence has several exons, the codonStart tag must be added to the first exon in the reference sequence. For example, if your reference sequence contains exons 4 and 5, add the codonStart tag to exon 4.
5. When specifying a base number in the codonStart tag, the number refers to the **nucleotide in the cDNA sequence**. Therefore, **the number must give a remainder of 1 when divided by 3** (e.g. 1, 4, 7, 10, ..., 3301, 3304,...).

Aligner will show a warning dialog when any one of these conditions is not met.

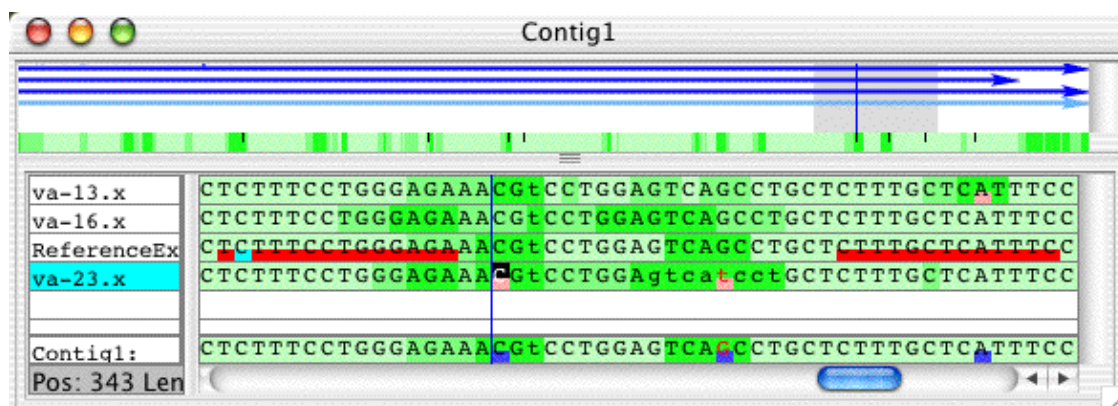
The **easiest way** to get the correct codingSequence and codonStart tags is by **importing the reference sequence from a text file in Genbank format**. CodonCode Aligner will use the "CDS" features in the Genbank file and add corresponding codingSequence and codonStart tags.

If you have more than one coding sequence region defined, Aligner will automatically keep track of the reading frame as the base numbering; mutations in introns will be annotated using the standard "n+x" or "n-y" numbering. Here is an example:



Feature	Source	Found In	Parent...	Start	End	Content
codingSequence	User	Reference...	Contig1	327	340	
codonStart	User	Reference...	Contig1	328	328	basenumber 6667
heterozygoteAC	Aligner	va-23.x	Contig1	343	343	Heterozygous 6679+3 C>A noncoding region
polymorphism	Aligner	Contig1	Contig1	343	343	1 diffs (0 homo / 1 hetero); 3 not mutated
heterozygoteGT	Aligner	va-23.x	Contig1	356	356	Heterozygous 6680-7 G>T noncoding region
polymorphism	Aligner	Contig1	Contig1	356	356	1 diffs (0 homo / 1 hetero); 3 not mutated
codingSequence	User	Reference...	Contig1	363	375	
heterozygoteAT	Aligner	va-13.x	Contig1	371	371	Heterozygous 6688 A>T Ser3Ser
polymorphism	Aligner	Contig1	Contig1	371	371	1 diffs (0 homo / 1 hetero); 3 not mutated
heterozygoteAC	Aligner	va-16.x	Contig1	391	391	Heterozygous 6692+16 C>A noncoding reg...

The corresponding contig view of this region shows the coding sequence regions (red boxes), the codonStar tag (light blue box), and the mutations found (pink boxes in the samples, dark blue boxes in the consensus):



Sample	Sequence
va-13.x	CTCTTTCCTGGGAGAAAACGtCCTGGAGTCAGCCTGCTCTTTGCTCATTTC
va-16.x	CTCTTTCCTGGGAGAAAACGtCCTGGAGTCAGCCTGCTCTTTGCTCATTTC
ReferenceEx	CTCTTTCCTGGGAGAAAACGtCCTGGAGTCAGCCTGCTCTTTGCTCATTTC
va-23.x	CTCTTTCCTGGGAGAAAACGtCCTGGAGtcaCctGCTCTTTGCTCATTTC
Contig1:	CTCTTTCCTGGGAGAAAACGtCCTGGAGTCACCTGCTCTTTGCTCATTTC

Pos: 343 Len

Excluding Regions from Analysis

You can exclude regions from being analyzed when Aligner finds mutations by adding "dontGenotype" tags. You can add "dontGenotype" tags to:

- Individual samples - to exclude tagged regions in the sample only
- The consensus sequence - to exclude the tagged region in all samples
- The reference sequence - to exclude the tagged region in all samples if (a) the analyzed contig is an alignment to the reference sequence, and (b) the reference sequence is used as the consensus sequence (as defined in the [consensus method preferences](#))

Adding tags to a region in a sample can be useful to skip regions with sequence artifacts which can throw off Aligner's analysis. Adding tags to the consensus sequence or the reference sequence in alignments can be useful to limit the analysis to a region of interest, or to avoid regions where all sample traces have artifacts.

How to add tags is described [above](#) (of course, you need to choose "dontGenotype" as the tag type to exclude regions from analysis).

How Aligner Finds SNPs

When searching for heterozygous point mutations, CodonCode Aligner does the following:

1. Each sequence is analyzed for low-quality sequence at the start and at the end. Regions that fall below the threshold set in the [mutation detection preferences](#) are marked with a "dataNeeded" tag, and ignored in the subsequent analysis.
2. Aligner looks for heterozygous insertion/deletion tags ("heterozygoteIndel") in each sequence, and excludes regions identified as heterozygote indels. Note that the tags need to extend to the start or to the end of the sample; tags that do not extend to the start are assumed to go to the end. *(This step is omitted if "Look for heterozygous indels" is deselected in the [mutation detection preferences](#), or when only looking for homoygous mutations).*
3. Aligner loops through all the consensus bases. In all samples that have aligned bases at a given position, Aligner examines the traces in each direction. Aligner looks for both secondary peaks and for drops in intensity that indicate a heterozygous base. For text samples that do not have chromatograms, Aligner just compares the base to the consensus base.
4. When a potential heterozygous base is found, Aligner examines the peaks and secondary peaks in the other samples at this position (and in the same direction) to see if the secondary peak may be due to random "noise". This "noise filter" can be turned off in the [mutation detection preferences](#).
5. Aligner classifies each base as homozygous or heterozygous, based on the height of the secondary peak and the intensity drop. Note that peaks may be classified as heterozygous even if there is no clear intensity drop (for example if all bases at this position are heterozygous).
6. If any of the samples at a given consensus position was classified as having a heterozygous base, tags are added to all analyzed samples at this position (unless "Add tags only to mutated bases" is selected in the [mutation detection preferences](#).)

The sensitivity of the detection can be adjusted in the [mutation detection preferences](#). Keep in mind, however, that Aligner may classify some bases incorrectly, and miss some heterozygous bases, regardless of the setting.

Limitations

CodonCode Aligner's detection of heterozygous point mutations is intended for research use only. It has not been verified for any diagnostic or clinical applications. Aligner will miss certain mutation, and incorrectly classify others. False-positive and false-negative rates depend on the current sensitivity settings, but are never expected to be zero. All results obtained with Aligner should be checked by a qualified scientist.

The following is an incomplete list of some known limitations and suggestion:

- Identification of heterozygous point mutations requires sequencing reactions generated from PCR products from heterozygous template DNA.
- Random peaks in sequence traces may cause mis-classifications (you can add a "dontGenotype" tag in such regions, so that Aligner will ignore it the next time you choose "Find Mutations")
- Aligner relies on peak patterns that are very similar between the different samples in an analysis. Any experimental changes, including, but not limited to, the use of different sequencing primers, kits, enzymes, dyes, sequencing machines, and running conditions may result in peak pattern variations that cause increased error rates.
- The base calls need to be correct in the analyzed regions; at heterozygous bases, one of the two bases must be called correctly, or the correct [IUPAC ambiguity](#) code must be used. Homozygous bases that have ambiguity base calls are likely to be classified incorrectly.
- Vector sequences in the samples may cause analysis errors.

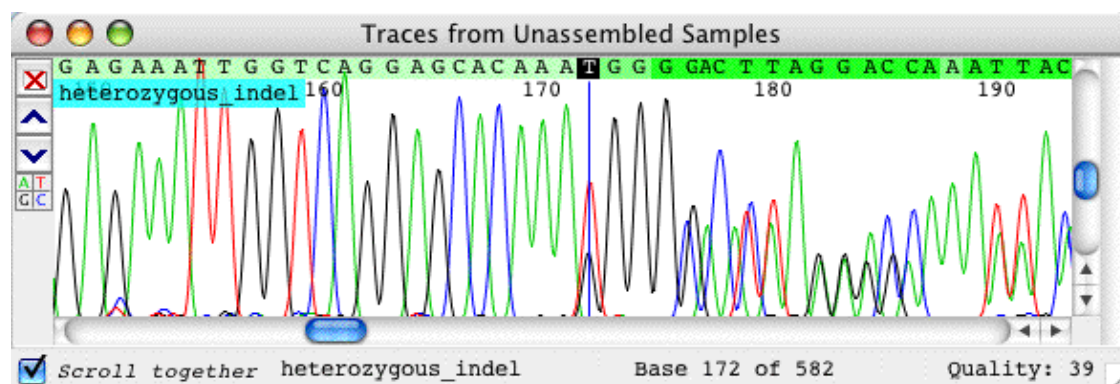
CodonCode Aligner User Manual

- Sequence quality must be high, and base-specific quality scores must be reasonably accurate (but need not be "perfect").
- Mistakes are most likely in low-quality parts of samples, and may affect other samples at this position. Mistakes at the beginning and end of samples can be reduced by increasing the minimum data quality requirement at the start and end in the [mutation detection preferences](#).
- If any samples contain heterozygous insertions or deletions, they should be identified before searching for heterozygous point mutations (for example by checking the "Look for heterozygous indels" check box in [mutation detection preferences](#)). Otherwise, many false classifications may result, especially if the minimum quality at start and end is set to low values.
- You should always adjust the [mutation detection preferences](#) to your specific needs.
- All results obtained with Aligner should be checked by a qualified scientist.

If you find any examples where Aligner's classification seems to be incorrect (and is not due to low-quality data or similar potential causes described above), please let us know by sending an email with your data attached as a Stuffer or Zip archive to support@codoncode.com.

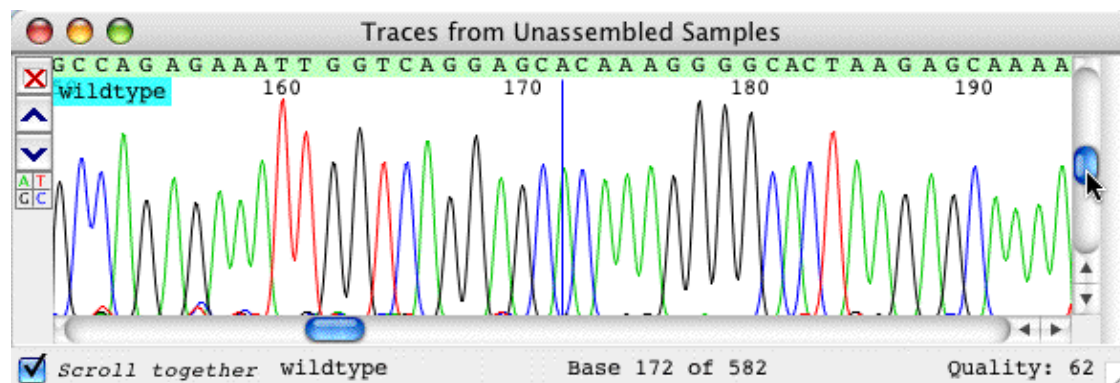
Heterozygous Insertions and Deletions

When sequencing PCR products from genomic DNA for mutation analysis, you will sometimes encounter traces that look like this:



Up to base 174, you have a nice clean sequence; but from base 172 on, you see double peaks at many (but not all) locations. Many of the peaks are also less tall than the peaks before base 172. This is typically caused by an insertion or deletion ("indel") in one of two chromosomes that you sequenced - by a "heterozygous indel".

Let us look at a homozygous "wild type" sequence at this location:



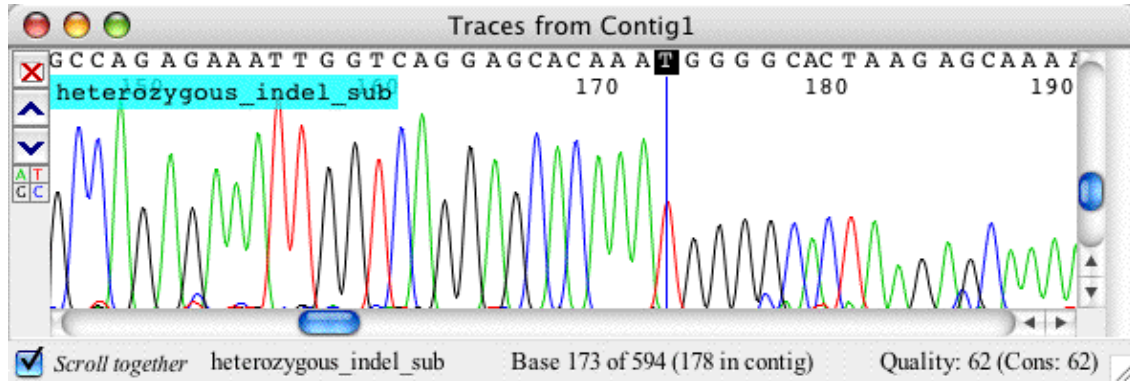
If you compare the wild type sequence to the indel sequence above, you will notice the additional red (T) peak at base 172. You also can see that the first yellow G peak is only about 50% of the intensity in the wild type trace, followed by 3 G peaks of regular height, and then an extra G peak that is half as high as the three preceding G peaks. From this, we can conclude that the indel event here is a one-base 'T' insertion on one of the two chromosomes sequenced.

Analyzing heterozygous indel traces this way is possible, but it can be tedious and error prone, especially if the insertions or deletions are longer than one or a few bases. CodonCode Aligner provides two functions that can help you in analyzing heterozygous insertions and deletions:

- The "[Find Heterozygous Indels](#)" function in the "Sample" menu can automatically identify potential heterozygous insertions and deletions.
- The "[Process Heterozygous Indels](#)" function can help you see the sequence of the mutated allele. CodonCode Aligner uses two algorithms to deduce the sequence of the heterozygous indel. The first algorithm looks at secondary bands and the reference (or consensus) sequence and replaces the

basecalls in the indel region with the likely sequence of the second allele. The second algorithm creates a new "subtracted" artificial sample by subtracting a scaled version of the wild type sequence, base calling the new "subtracted" trace with Phred. After this, both sequences are re-aligned or re-assembled with the rest of the original contig.

Here is an example of a "subtracted" sequence that was created by CodonCode Aligner from the heterozygous indel trace shown above:



In the processed sequence, where the allele that corresponds to the wild type sequence has been subtracted, it is very easy to see the one base "T" insertion.

To someone experienced in analyzing sequence traces with heterozygous indels, this example may appear trivial - after all, we could determine the mutation quickly by looking at the original trace. However, CodonCode Aligner's algorithm will work as well even for large insertions or deletions, which otherwise can be very hard to analyze. There are, of course, some [limitations](#), which are described [below](#).

Finding Heterozygous Indels

CodonCode Aligner's "Find Heterozygous Indel" function analyses sequence chromatograms for characteristics that are typical for heterozygous insertions and deletions. In traces where Aligner finds a putative heterozygous indel, Aligner add a tag that extends from the start of the putative indel to the end of the sequence.

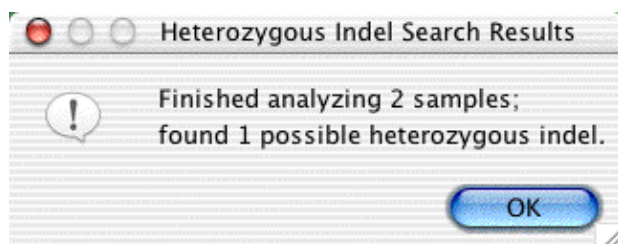
To find heterozygous indels, do the following:

- Select the traces you want to analyze in the project view. Traces can be unassembled or in contigs, but must have base-specific quality scores (run base calling with Phred first if needed).
- Heterozygous indel detection works best if the samples have not been end clipped. End clipping will typically remove the parts of sequences that have heterozygous indels. While Aligner tries to compensate for this, the detection rate for samples that have not been endclipped is slightly higher than for endclipped samples. *If your samples have been end clipped, you can simply re-do the base calling to get end-to-end sequences.*
- Go to the "Sample" menu, and select **"Find Heterozygous Indels"**.

Aligner will show a progress bar while looking at your traces one-by-one. The analysis may take a few seconds per trace.

Aligner will show the results for each sequence analyzed in the status area, and add "heterozygoteIndel" tag to

each sequence where it finds a putative heterozygous insertion/deletion. When all sample have been analyzed, Aligner will show a dialog that summarizes the results:



If any indels were found, Aligner will open a new window that shows information about these indels, similar to a feature view window:

Feature	Source	Found In	Parent Contig	Start	End	Content
heterozygoteIndel	Aligner	heterozygous_indel	Unassembled Samples	172	582	Score: 29

Double-clicking on any line in the table above will open a view for this sample so that you can verify the results (typically, the trace view will be opened, unless you have changed the [double clicking preferences](#)).

To try this out with an example project that is included with CodonCode Aligner, you can open the "Hetero_indel" project in the "Example Files" folder in the directory where you installed CodonCode Aligner.

Hetero Indel Scores

All heterozygous indels found by CodonCode Aligner will receive a score that is shown in the "Content" column. Scores assigned range from 11 to about 35, with higher scores indicating higher confidence. Indels with a score below 15 are more likely to be false positives; the proportion of false positives in indels with scores above 20 is substantially lower. *Indel scores are somewhat similar to Phred quality scores; however, unlike Phred quality scores, indel scores are **not accurately linked to error probabilities**.*

False Negatives

The indel finding algorithm may sometimes miss real mutations, especially if they are very close to the start or end of a sequence, or if the sequence before the indel is of low quality. In this case, you can **add a "heterozygoteIndel" tag by hand**, as follows:

- In a trace view window, select the base where the heterozygous indel starts (for example, base 175 in the "heterozygous indel" sample in the example project)
- Bring up the popup menu by right-clicking on this base on Windows, or Control-clicking on OS X.
- Select "Add Tag..." from the popup menu. This will open a new "Add Tag" dialog.
- In the "Add Tag" dialog, select "heterozygoteIndel" as the tag type from the pull down menu at the top. You can also enter "9999" in the "End" text field if you would like the tag to cover the entire rest of the sample, instead of just one base. Then, click "OK".

You should now see the tag displayed in the trace view for this sample (unless you changed your [highlighting preferences](#) to not show tags).

False Positives

Occasionally, the indel finding may incorrectly identify artifacts in sequencing traces as heterozygous indels. This can happen when the sequence quality suddenly drops dramatically in a sequence, for example due to "polymerase stutter" after poly-A runs. Also, please keep in mind the the heterozygous indel finding is intended only for sequences from genomic PCR.

You can easily **remove incorrect heterozygous indel tags** as follows:

- Open a trace view that shows the indel tag (the easiest way to do this is to double-click on the line describing the indel tag in the report view shown above or in a feature view)
- Right-click (OS X: control-click) on any base with the indel tag to bring up the popup menu
- Select "Show local tags" from the popup menu to open the tag dialog
- In the tag dialog, select the "heterzygousIndel" tag, then press the "Delete" button, followed by the "OK" button

Splitting Heterozygous Indels

You can split heterozygous indels into two new "pseudoallele" samples as follows:

- Select the sample(s) with a heterozygous indel tag.
- Go to the "**Sample**" menu, and select "**Split Heterozygous Indels**".

For each sample with a heterozygous indel, two new samples will be created in the "Unassembled Samples" folder. CodonCode Aligner will try to separate the traces from the indel site on into a longer and a shorter pseudo-allele. This often works reasonable well, but please keep these limitations in mind:

- Splitting will work only for heterozygous indels up to about 25 bases, and only if the indel is in a region where peaks are reasonably well separated. For other indels, the sample traces that result from the splitting will typically have missing peaks or lots of double peaks.
- The separated pseudo-alleles are (currently) only intended for indel size estimates and manual verification. Specifically, any differences in peaks between the shorter and the longer allele after the indel site may or may not be real; even real differences may be attributed to the wrong allele.

Processing Heterozygous Indels

To analyze ("process") heterozygous indels in Aligner, the samples to be analyzed need to have a "heterozygoteIndel" tag. These tags can be added by CodonCode Aligner or manually, as described in the preceding section.

In addition, the sample to be analyzed and the corresponding "wild type" trace must be in the same contig before heterozygous indels can be processed, since CodonCode Aligner uses the alignment information when subtracting the wild type trace.

To illustrate the entire process of analyzing heterozygous indels, let us process the "Hetero_indel" example project that comes with CodonCode Aligner:

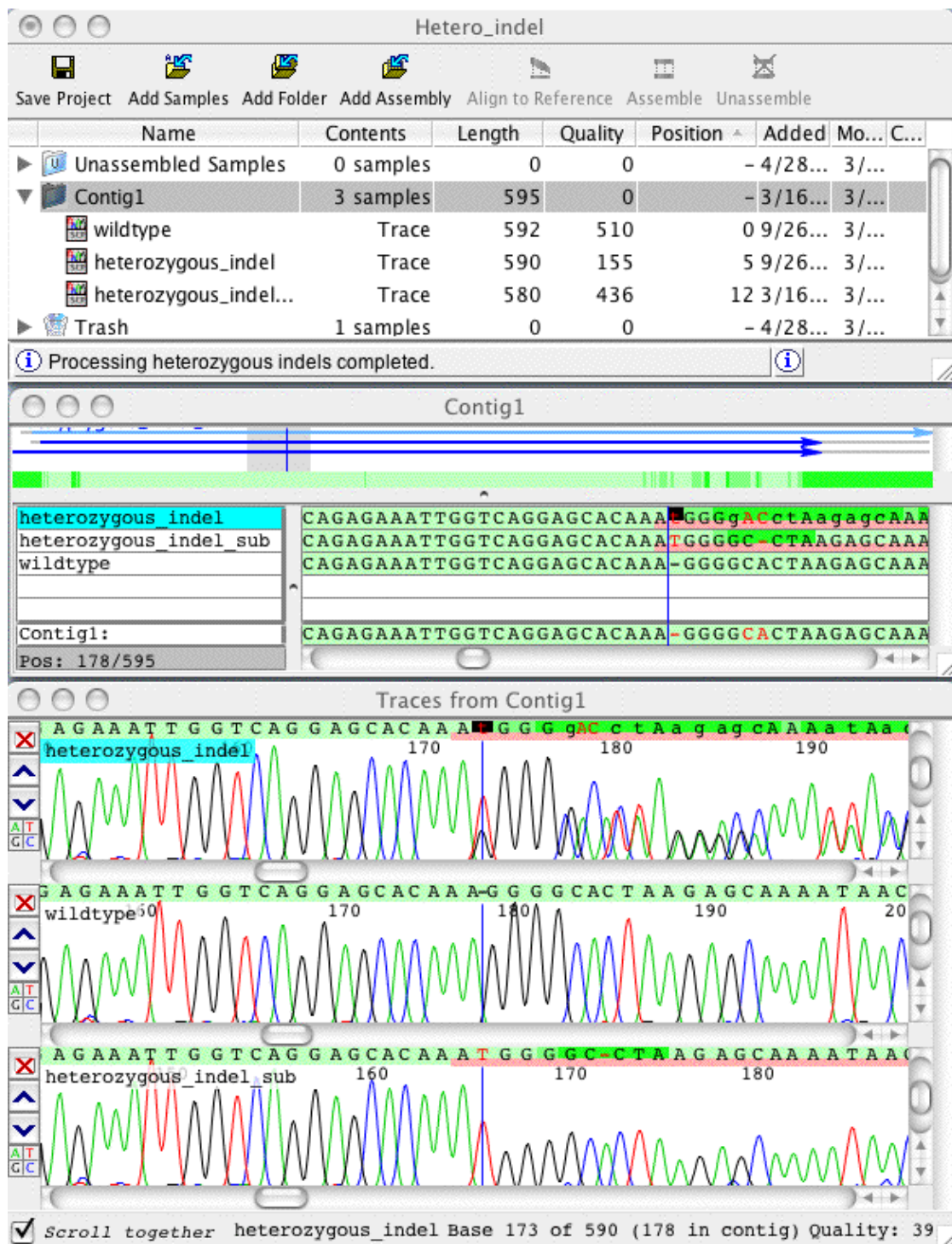
CodonCode Aligner User Manual

- Open the "Hetero_indel" example project (in the "Example Files" directory inside the directory where you installed CodonCode Aligner).
- Select the sample named "heterozygous_indel".
- Go to the "**Sample**" menu, and select "**Find Heterozygous Indels**".
- When Aligner is done with the previous step, select the "Unassembled Samples" folder in the project view.
- Go to the "**Contig**" menu, and select "**Assemble**".
- When the assembly is done, select the contig "Contig1" in the project view. Alternatively, select the two samples in Contig1 in the project view.
- Go to the "**Sample**" menu, and select "**Process Heterozygous Indels**".

You will progress dialog appear which shows that CodonCode Aligner does the following steps:

- Aligner finds all samples that have a "heterozygoteIndel" tag, and then identifies "wild type" traces for each (since we have only 2 traces here, this is trivial).
- Aligner replaces the basecalls in the indel region as follows:
 1. Aligner looks for secondary peaks in the indel region, and replaces the base calls with ambiguity codes for the the two bases.
 2. Aligner compares the ambiguity codes to the consensus sequence, and removes the consensus base call. This leaves the base call that (often) corresponds to the base in the mutated allele.
- Aligner create a new artificial sequence by subtracing the wild type trace from the heterozygous indel trace, starting at the start of the "heterozygoteIndel" tag. The wild type traces are scaled the traces vertically and horizontally as needed, with the goal to subtract the trace that is due to the unmutated allele, and leaving the mutated allele.
- Next, Aligner base calls the newly created trace or traces with Phred (which requires that you have a trial license or a full license).
- When Phred is done with base calling, Aligner moves the initial subtracted trace which still has the original base calls to the trash, and imports the subtracted trace with the new Phred base calls.
- Finally, Aligner re-aligns or re-assembles the original contig, adding the newly created subtracted traces.

Here is what the final result looks like:



The newly created subtracted sample is called "heterozygous_indel_sub". Note that (a) its sequence after the single 'T' insertion is identical to the wild type (unlike the original sample, where the extra peaks led to extra base calls), and that (b) the quality scores of the sequence after the T insertion are higher, as indicated by the lighter background colors. Before getting too excited, however, please read the next section!

Limitations

While we believe that CodonCode Aligner's functions for analyzing heterozygous insertions and deletions can be a valuable tool, it is important to keep a number of limitations in mind - **limitations regarding the performance** of the algorithms, and **limitations regarding the interpretation of the results**.

Processing Pre-Requisites and Limitations

- **Finding heterozygous indels requires base-specific quality scores.** The algorithm will not work on sequences that do not have quality scores. Most testing so far has been done on sequences processed with the base calling program PHRED.
- **The analysis will work only in regions of reasonably high quality, and not close to the start or end of sequence traces.** Both the finding and the subtraction step require that sequence quality before the heterozygous indel is reasonably high; for example, multiple peaks need to be well resolved, and the indel cannot be closer than approximately 50 bases to the start or end of the sequence. In addition, indels in or after long mono- or dinucleotide repeats will generally not be found (since such repeats often result in "polymerase stutter", which looks very similar to heterozygous indels).
- **Trace subtraction requires a sequence without heterozygous indels.** If all your sequences have heterozygous indels, the subtraction does not work. Ideally, the wild type sequence should be not have any heterozygous mutations; however, isolated heterozygous point mutations may be tolerable.
- **The "wild type" sequence must be the same direction and chemistry, and preferably use the same primer.** The subtraction step relies on the peak patterns being very similar in the two sequences. This is generally not the case for sequences produced with different sequencing chemistries or sequenced from different strands.
- **The wild type sequence must be largely identical to one of the two alleles in the mutated sample.** This is typically the case when analyzing sequences from human samples; however, it may not be the case when analyzing regions with a high degree of lengths polymorphisms, for example intronic sequences from different species.
- **The analysis may fail in obvious or non-obvious ways.** While the trace subtraction should always produce a new "subtracted" sequence, the resulting trace may not correctly show the presumed mutated allele. In obvious cases, the subtracted sequence will have peak patterns that are clearly incorrect, and low quality scores. In less obvious cases, the subtracted sequence may look reasonable, but individual peaks and corresponding bases may be missing. Therefore, it is essential to closely look at all three sequence traces together - the indel sequence, the wild type, and the subtracted sequence.

If you happen to have examples of where the algorithm did not work as expected, we would certainly appreciate if you could send us your trace files so that we can continue to optimize CodonCode Aligner's algorithms.

Interpreting Results

When interpreting the results of heterozygous indel processing done with CodonCode Aligner, always keep in mind that CodonCode Aligner is only a tool that helps you to come to conclusions. You will need to use caution whenever interpreting what you see. Some specific things to keep in mind:

- The subtracted sequence you see is **not** a real allele sequence. For example, any mutations you see after the indel site could be on either allele, or on both alleles.
- The base-specific **quality scores** close to and after the indel site in the subtracted sequence should be used **for guidance only**. They do **not** have the same mathematical accuracy as quality scores for "real" sequences. However, we believe that you can use the quality scores to quickly get an idea if the

subtraction worked or not.

- The algorithms for analyzing heterozygous indels have been developed and tested only on a limited number of sequences, and have not been experimentally validated.

This is by no means a complete list, so be cautious! CodonCode Aligner is intended for research use only, and has never been validated for any clinical or medical applications. You should **not** base any medical decisions on any results obtained with CodonCode Aligner.

Acknowledgements

The sequences used for the examples shown herein are from the PolyPhred examples by Dr. Deborah Nickerson's group, available at <http://droog.mbt.washington.edu/example/csft2.tar.gz>. The names of the sequences used from this collection were changed for illustrative purposes. We thank Dr. Nickerson and all others who have made traces with heterozygous indels available to us. The use of trace subtraction to identify heterozygous (point) mutations was originally described by Staden, Rada, and Bonfield and implemented in the program TraceDiff (Bonfield,JK, Rada,C and Staden,R,. 1998, Nucl. Acids Res. 26, 3404-3409).

Early work on the algorithms for analyzing heterozygous insertions and deletions used in CodonCode Aligner was funded by the National Cancer Institute (SBIR grant 1R43CA83384-01).

Editing Samples

CodonCode Aligner allows you to edit your sequences - but before you start editing, remember that

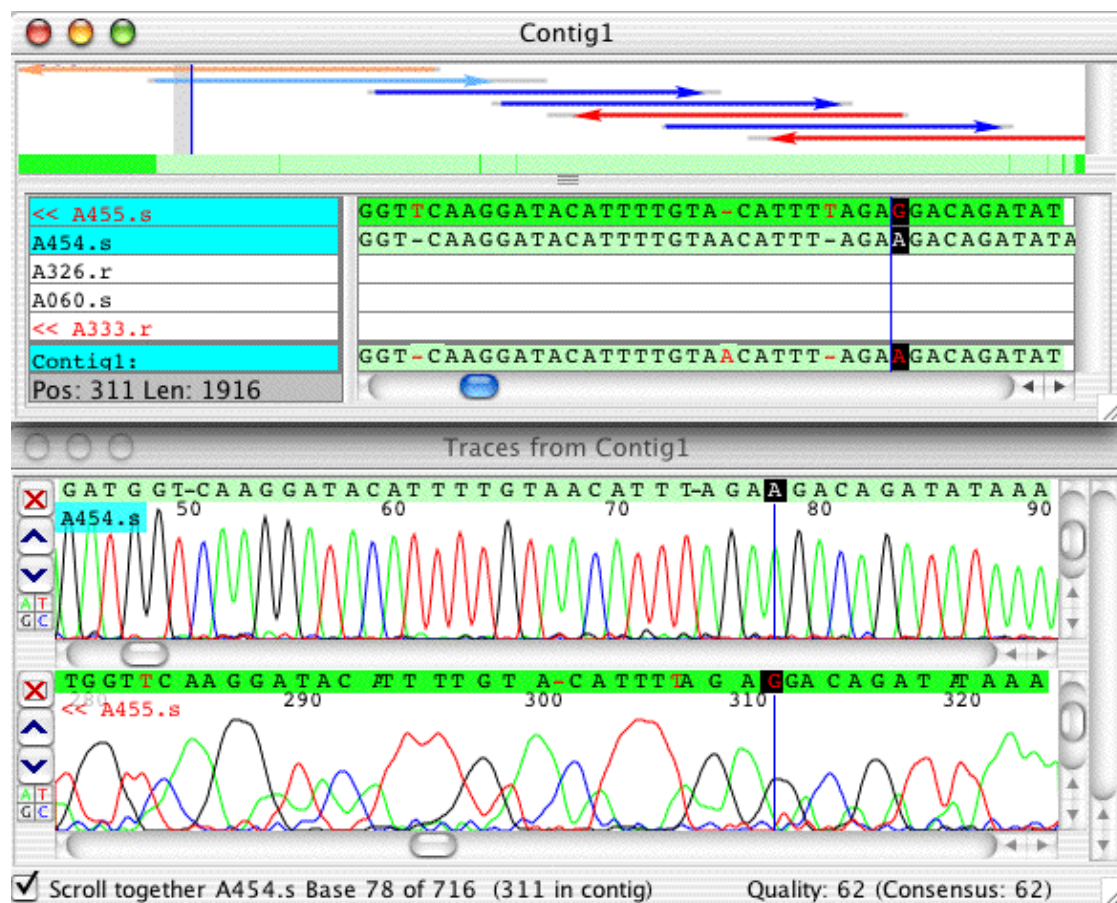
You may not need to edit!

This is because Aligner will examine the quality scores of all aligned bases at each contig position when building the consensus - in other words, Aligner typically builds a **quality-based consensus**, rather than a majority-based consensus.

In the days before Phred quality scores, editing discrepancies was important: consensus sequences were always majority-based, and the only way to make sure that the consensus sequence was indeed correct was to look at the discrepancies. Typically, scientists would simply correct any wrong base calls, so that they would not have to look at the same region again.

With samples that have Phred (or Phred-like) quality scores and sequence assembly algorithms that use the quality scores, a lot of this editing is not necessary anymore. Typically, wrong bases will have lower quality scores, and the correct "consensus" can be determined automatically by just looking for the highest-quality base at any position. The low-quality discrepancies can often safely be ignored.

Here is a typical example - a region covered by two sequences which differ at several places:



CodonCode Aligner User Manual

At the cursor position, Aligner chose the 'A' from sequence 'A454.s' as the consensus, since it is of much higher quality than the 'G' in sequence 'A455.s' (the higher quality is indicated by the lighter background). You can also see three other discrepancies in this region; in each case, the sequence from 'A454.s' is high quality and correct, and therefore chosen as the consensus sequence.

If you would look at the same region in an older assembly program that only supports a majority-based consensus, the consensus base at these four locations here would be an ambiguity, or the single called base at places where one sequence has a gap character. To get a clean consensus, you would have to look at all four regions, and edit three or four times, to come to the same end result.

Also note that Aligner uses the quality scores to estimate how likely it is that the consensus quality is correct (this can be done with rather good accuracy since the quality scores are linked to error probabilities). In the region shown above, the estimated consensus quality is very high, since one of the two sequences is of very high quality through the entire region. When editing contigs with Aligner, you can define quality thresholds for the consensus sequence in the [Feature preferences](#), and use this to [very quickly move](#) to regions that need your attention.

To build a quality-based consensus, two prerequisites need to be met:

1. The sequences must have base-specific quality scores (if you import sequence trace files that do not have quality scores, you should first do the "[Base calling](#)" with Phred to get quality scores).
2. You must have selected to build a quality-based consensus in the [Consensus preferences](#) (this is the default setting when you install Aligner).

In addition to the quality-based consensus, there are two other factors that enable you to get cleaner assemblies with much less need for "contig editing" are:

- The ability to automatically clip low-quality sequence from the end of reads, thereby reducing the total number of discrepancies.
- The use of local alignments (as opposed to end-to-end alignments); regions with high error rates tend to end up in the unaligned ends, and can also be ignored.

Of course, there will be times where contig editing is necessary, and CodonCode Aligner does provide many editing functions that are described on the next pages.

Windows for Editing Samples

You can edit sample sequences in several different views (windows) - the [base view](#), the [trace view](#), and (for assembled or aligned samples) the [contig view](#). The only views that do not allow editing are the views that do not display the bases - the [project view](#), the [quality view](#), and the [feature view](#).

In general, **we suggest to do most editing in the trace view window**, simply because you can see the underlying data while editing. One possible exception of this rule are sequences that are part of contigs - sometimes it may make sense to edit in the contig view (but having a trace view open that shows the sequence is still a good idea!).

All windows are linked. If you move the cursor or make a selection in a contig window, Aligner will scroll to the same position in any open the base view and trace view windows for that sample. Likewise, when editing base calls, changes are immediately made in all other open views for that sample.

Cursor Positioning and Movement

The cursor is positioned by clicking at the desired position in a sample or the consensus. The cursor position is indicated by a vertical cursor position line displayed across all samples and the consensus.

Moving to Features, Ambiguities, Mismatches or Edited Bases

The **Go** menu has selections for quickly moving through the sequence:

- **Next or Previous Feature:** Moves the the next or previous feature (or "region of interest"). Features can be discrepancies, low-quality consensus bases, low coverage regions, or a number of other things that may be of interest to you. You can define *your* regions of interested in the [feature preferences](#).
*Tip: the fastest way to navigate to the next or previous feature is by using the **keyboard shortcuts** : On Windows, use Control-Right Arrow and Control-Left Arrow; on OS X, use Command-Right Arrow and Command-Left Arrow*
- **Next or Previous High Quality Mismatch:** Moves the the next or previous high-quality base that disagrees with the consensus
- **Next or Previous Low Quality Consensus:** Moves the the next or previous place where the consensus quality is low.
- **Next or Previous Ambiguity:** Selects the next or previous ambiguity in the selected sample or consensus.
- **Next or Previous Mismatch:** Moves the cursor to the next or previous mismatch the selected contig.
- **Next or Previous Edited Base:** Selects the next or previous edited base in the selected sample.
- **Base Number:** Displays a [dialog](#) where you specify a specific base number in the selected sample or consensus.
- **First Aligned Base:** Moves the cursor to the first aligned base in the current sample.
- **Last Aligned Base:** Moves the cursor to the last aligned base in the current sample.

In addition to the menu items in the "**Go**" menu and their keyboard shortcuts, you can also use the "home" key to go to the first base of a sequence, and the "end" key to go to the last base of a sequence.

Selecting Bases

To select bases, move the mouse cursor over the first base you wish to select (in the base view, contig view, or trace view). Click and hold down the mouse button, drag the cursor horizontally over the base(s) and release the mouse button after dragging over the last base. *This can be a bit slow for large selections; if you want to select all bases to the start or end of the sequence, it can be a lot faster to use one of the options described below, so please read on!*

Selected bases are displayed with a different background color than surrounding bases. If the selection was made in the consensus, the selection is also shown in all samples that are part of the consensus.

Selecting from sequence start to current cursor position

To select all bases from the start of the sequence to the current cursor position, choose "**Select from Start to Here**" from the "**Edit**" menu. The selection will include any unaligned bases at the start of the sequence (up to the cursor position).

You can accomplish the same thing without having to use the menus by keeping the "shift" key pressed while pressing the "home" key.

Selecting from current cursor position to the end of a sequence

To select all bases from the the current cursor position to the end of the sequence, choose "**Select from Here to End**" from the "**Edit**" menu. The selection will include any unaligned bases at the end of the sequence (up to the cursor position).

You can accomplish the same thing without having to use the menus by keeping the "shift" key pressed while pressing the "end" key.

Selecting all bases in a sequence

To select all bases in a sequence, choose "**Select All**" from the "**Edit**" menu. Alternatively, you can use the keyboard shortcut - Control-A on Windows, and Command-A on OS X (keep the control- respectively command-key pressed while pressing 'A').

The selection will include any unaligned bases at the start and at the end of the sequence.

If the project view is the active view, then "**Select All**" will select all contigs and samples shown in the project view. The "Unassembled Samples" and "Trash" folders will also be included, unless they are empty.

Changing Bases

Selected bases can be changed. To change a single base, select the base and type a new base letter. Letters for ambiguity codes can also be entered.

Be careful if more than one base is selected before you start typing: all the selected bases will be replaced with a single base, which may not be what you want.

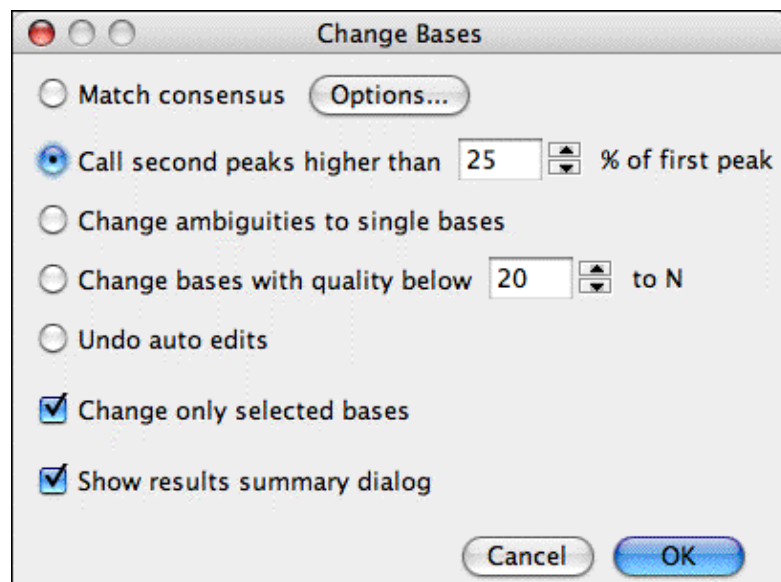
After changing a base, Aligner will automatically move the selection one base to the right. This allows you to quickly edit a number of bases in a row.

The consensus sequence is automatically updated with any changes, if such changes introduce a change in the consensus sequence. Edits made in one View are automatically updated in other Views of the same sample.

If you **edit a consensus base**, then the change will be applied to all samples that have aligned bases at this position - so once again, be careful!

Automatic edits

CodonCode Aligner offers several functions to automatically change bases, for example to call heterozygous bases, or to convert low-quality bases to N. To auto-edit bases, select that bases you want to edit, go to the "Edit" menu, and choose "Change Bases...". This will show the following dialog:



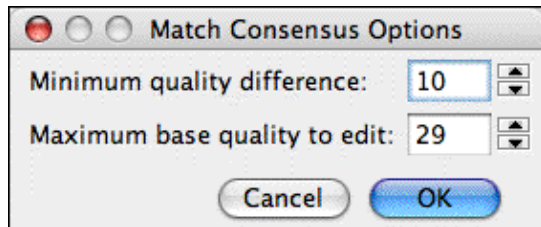
Use the radio buttons to choose how you want to change bases (see below). You can also choose if you want to change all bases in the selected sample(s), or just the selected bases, and whether or not you want to see a dialog summarizing the changes.

CodonCode Aligner will add tags to all auto-edited bases. You can undo all changes right after you made them, or later with the "Undo auto edits" selection in this dialog.

Match consensus

The "Match consensus" option will automatically change bases to match the consensus of the contig they are in (it will have no effect on unassembled samples). One example where this option is useful are "dye blobs" in DNA sequences, where many bases are wrong because of artificial strong peaks.

This option is intended for low-quality sequence, and will not change high-quality bases, unless you change the settings by clicking on the "Options..." button next to "Match consensus". This will show the following dialog:



The number at the top is the minimum difference in sequence quality between the consensus and the mismatch base in question. The default setting of 10 means that bases will only be edited if the consensus sequence quality is at least 10 higher, corresponding to a 90% or higher probability that the edit is correct (assuming that this is a random error).

The number at the bottom determines the highest quality a base can have to be edited; with the default setting, any base with a quality score of 30 or higher will not be edited.

Call second peaks higher than...

The "Call second peaks" option allows you to convert all bases with a significant secondary peak to ambiguities. You can set the threshold of secondary peaks, relative to the intensity of the first peak.

This method can sometimes be useful in mutation detection. Please note, however, that it is much simpler than the methods used in CodonCode Aligners "Find mutations" function, and therefore much more prone to both false-positives and false negatives. For example, the "Find mutations" function compares peaks at a given position to other peaks in other samples at this positions, and can therefore detect heterozygous bases from a drop in peak intensity, even if the secondary peak is very small.

Change ambiguities to single bases

This is the reverse of the previous function - any bases with ambiguity codes will be converted to regular bases (A,C,G, or T), according to the highest peak at the base position. This option is useful if you have sequences that contain ambiguity codes, but you know that there are no heterozygous bases (for example because you sequenced a clone, not a PCR product).

Change low quality bases to N

This option allows you to change all bases with a quality below the given threshold to 'N'.

Undo auto edits

When selected, this option will undo all previous auto-edits (any of the options described above) for your current selection, and convert bases back to the base calls before the first auto edit. To function properly, this requires that the "auto edit" tags that Aligner added originally are still present. As long as your samples stay in the same project, this is usually not a problem. However, if you export samples and then re-import them, the tags will typically be lost, and undoing auto-edits with this options will not be possible.

Deleting Bases

Using the Keyboard to Delete Bases

Backspace Key: Deletes the currently selected base or bases; then, for sequences in contigs, the bases to the left of the cursor are shifted to the right.

Delete Key: Deletes the currently selected base or bases; then, for sequences in contigs, the bases to the right of the cursor are shifted to the left.

Note: If the consensus is selected, Delete or Backspace deletes not only the base(s) in the consensus, but bases at that position in all samples having sequence at that position.

OS X Users: On Macintosh keyboards, the Backspace key is actually labeled "delete", not Backspace! It is part of the main key group. The "Delete" key is labeled "del" (above an "X" in a box), and usually is part of a small group of keys that include the "page up" and "page down" keys.

Menus for Deleting Bases

Instead of using the keyboard shortcuts for deleting bases, you can also go to the "**Sample** " menu, select the "**Delete**" submenu, and then choose one of the following items:

Selection - Fill from Left: Same as Backspace key.

Selection - Fill from Right: Same as Delete key.

From Sample Start: Deletes all bases to the left of the cursor for the selected sample.

To Sample End: Deletes all bases to the right of the cursor for the selected sample.

Deleting Samples

Deleting samples in Aligner is a two-step process: first, you move samples to the "Trash" folder, and then, you empty the trash. To move a sample to the trash, select the sample (for example in the project view), and then choose "**Move to Trash**" from the "**Edit**" menu. For samples in contigs, some special considerations apply, as described in the "[Editing Contigs](#)" section.

Samples in the trash can be re-used by selecting them in the project view, and then choosing " **Move to Unassembled Samples**" from the "**Edit**" menu.

To permanently delete samples from the trash and remove them from your project, choose "**Empty Trash...**" from the "**File**" menu. The corresponding sample files in your project folder will be deleted when you save the project the next time.

Moving Gaps and Samples

Sometimes, you may find that some of the gaps introduced during assembly are not quite at the right positions, and should be moved around a bit; aligning gaps in all reads may even enable you to remove an entire column of gap characters.

Moving gaps in contigs

To move a gap, select the gap in the contig view (or in the base or trace view), and then select "**Move Gap Left**" or "**Move Gap Right**" from the "**Sample**" menu. You can also use the keyboard shortcuts:

- Option-F5 for "**Move Gap Left**"
- Option-F5 for "**Move Gap Right**"

(Due to a Java bug on OS X, the keyboard shortcuts are not always shown in the menus.)

In the contig view, you can also move single gaps around by drag and drop: click on the gap to select it; then click on it again, but keep the mouse pressed and move the cursor to where you want the gap to be; then release the mouse.

Moving samples in contigs

You can also move entire sequences one base to the left or to the right within a contig by selecting the sequence, and then choosing "**Move Sequence Left**" or "**Move Sequence Right**" from the "**Sample**" menu.

Moving samples to "Unassembled Samples"

You can move samples from contigs or from the "Trash" folder to the "Unassembled Samples" folder by selecting the sample(s), and then choosing "**Move to Unassembled Samples**" from the "**Edit**" menu. For samples in contigs, some special considerations apply, as described in the "[Editing Contigs](#)" section.

In the project view, moving samples to the Trash or Unassembled Samples can also be done using drag and drop.

Inserting Gaps and Bases

To **insert gaps**:

- Position the cursor at the base after which you want to insert a gap
- Press the space bar (or select "**Insert gap => Shift Bases Right**" from the "**Sample**" menu).

A new gap will be inserted before the current cursor position. If the sample is part of a contig, the bases *after* the new gap will be shifted to the *right*.

One exception to this rule applies when the last base of a sample is selected: in this case, the gap will be inserted *after* the current base, so that you can [add bases to the end of reads](#).

If you want to insert a gap and **shift the bases before the gap to the left** (instead of the bases *after* it to the *right*), select "**Insert gap => Shift Bases Left**" from the "**Sample**" menu. Alternatively, use the keyboard shortcut Shift-Space (keep the shift key pressed while pressing the space bar).

If you are working in the contig view and have a consensus base selected, gaps will be inserted in the consensus and in all sequences at this position.

To **insert a base**:

- insert a gap character as described above
- then type the letter of the base you wanted to insert.

Since the gap you just inserted was selected after you inserted the gap, you just replaced it with the character you typed.

Adding Bases at the End of Reads

To add bases to the end of a read:

1. Select the last base of the sample
2. Press the space bar to insert a gap
3. Press the letter of the base you want to add to change the gap to the base
4. Repeat steps 2 and 3 for each base you want to add.

The gaps and bases that are added this way will not necessarily line up exactly with the peaks - instead, they will be spaced evenly, with the same spacing as the last 20 or so bases before the last base. This is usually ok if you need to add just a few bases. If you need to add many bases, you may be better off repeating the [Base Calling](#) for this sample (if the sample is in a contig, you first have to move it to Unassembled Samples).

You cannot directly insert a gap or a base right before the last base in a sample, but there is an easy workaround: just insert a gap after the last base, and then choose "**Move => Gap Left**" from the "**Sample**" menu.

***Note:** Aligner is not intended as an editor where you enter sequence by hand; there are plenty of other programs out there that allow you to do that. Aligner is intended to be used with sequence traces, preferably traces with Phred- or Phred-like qualities.*

Reverse Complementing

To reverse complement a sample or contig:

- Go to the project view
- Select the contig (or a sample in the "Unassembled Samples" folder)
- Choose "**Reverse Complement**" from the "**Edit**" menu

In most views, you will be able to identify samples that have been reverse-complemented by the "<<" prefix before the name, and by the fact that the name is drawn in red. In the project view, you can identify reverse-complemented samples by the icon (it has red rather than black borders).

Reverse-complementing of unassembled samples is possible, but usually not necessary, since samples will be reverse-complemented as needed during assembly and alignment.

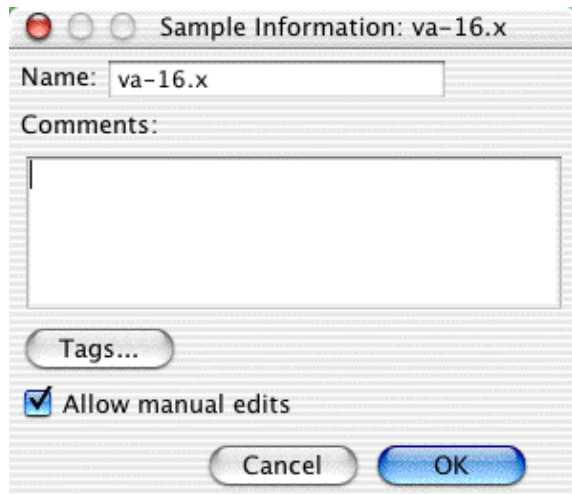
You cannot reverse-complement samples in contigs directly, since that would destroy the alignment to the consensus. If you select "**Reverse Complement**" when a contig view is in the foreground, the entire contig will be reverse-complemented.

Editing Sample Information

To view information about a sample, or to change a sample name:

- Select a single sample in the project view (or in trace view or contig view)
- Go to the choosing "**Sample**" menu, and select "**Sample Information**"
(or use the keyboard shortcut: control-I on Windows, command-I on OS X)

This will show the sample information dialog:



You can add or change remarks about the sample in the comments field. You can also designate a sample file as "read only" by unchecking the "**Allow Manual Edits**" checkbox.

Changing Sample Names

You can **change the sample name** by editing the "Name:" field. Note that Aligner does not allow spaces or other "funny" character in the sample names, since such characters could cause problems when writing sample files. If you use any invalid characters, Aligner will convert them automatically.

Viewing Chromatogram Information

For samples that have chromatograms, additional information about the chromatogram will be shown in the sample information dialog:

Sample Information: Abi_example_file.ab1

Name: Abi_example_file.ab1

Comments:

Chromatogram Information:

- SPAC=10.507583
- LIMS=c78fff66631211d89ed8000874938d7c
- GTYP=Pop-7
- Run_Start=2004-02-19 15:29:17
- Run_Stop=2004-02-19 16:33:52
- BCAL=KB.bcp
- BCSW=KB 1.0

No tags

☒ Allow manual edits

Cancel OK

The example above shows information extracted from an ABI file. Note that the last two lines show which base caller was used for this sample. In this case, the KB base caller, which assigns quality scores to each base, was used (the alternative is the ABI base caller, which does not assign quality scores).

For chromatograms that were base called with PHRED, the last lines in the "Chromatogram Information" will still point to the KB or ABI base caller. But if you **scroll through the chromatogram information**, you can see entries added by PHRED:

Sample Information: PHRED_example_file.ab

Name: PHRED_example_file.ab1

Comments:

Chromatogram Information:

BCSW=phred 0.040406.a
NAME=e
LANE=32
GELN=
PROC=
RTRK=
CONV=phred version=0.040406.a
COMM=

No tags

☒ Allow manual edits

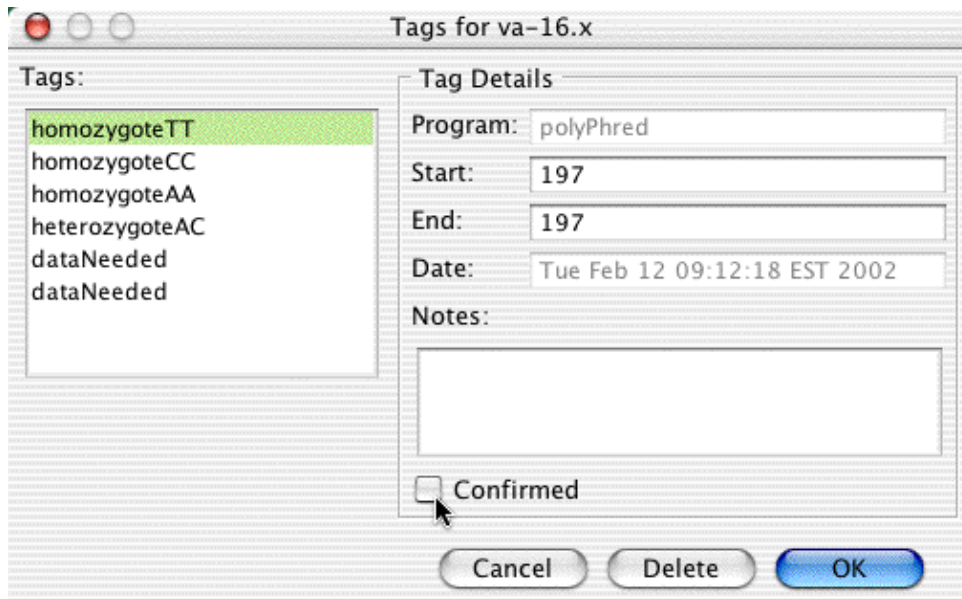
Cancel OK

In the example above, you see an entry labeled "BCSW" (for "base caller software version") that point to PHRED, and list the version of PHRED that was used. A second entry which starts with "CONV" also shows that PHRED was used.

The chromatogram information is read from ABI files, or from the comment field in SCF files. You cannot edit this information.

Viewing Sample Tags

Sample may also contain tags which have been added by users, other programs like PolyPhred, or CodonCode Aligner during steps like end clipping or vector trimming. If a sample contains tags, you can click on the "Tags..." button to display the tag dialog:



(You can also see all tags for a sample by selecting the sample in the project view, going to the **"Sample"** menu, and selecting **"Show All Tags..."** from the **"Tags"** sub-menu).

You can select tag on the left to view details about the tag on the right. You can change the start and end position of tags, and add comments about the tag in the "Notes:" text field. You can also delete tags. Changes will be saved after you press the "OK" button.

Confirming Tags

One common use of the tag "notes" is to mark tags as confirmed after looking at the trace data. You could do this by typing "Confirmed" in the note box, or simply by clicking on the "Confirmed" checkbox. Clicking on the checkbox when it is unchecked will add the word "Confirmed" at the beginning of the text; clicking it when it is checked will remove **all** occurrences of the word "confirmed" from the check box.

Tip: If you find a tag that is wrong, you can either delete the tag, or you could mark it in the "Notes" section. We suggest that you do **not** write "Not confirmed", but rather use different words like "Disagree" or "Incorrect".

The contents of the "Notes" field are shown in the "Content" column in the Feature View, and when [exporting features](#) to text files.

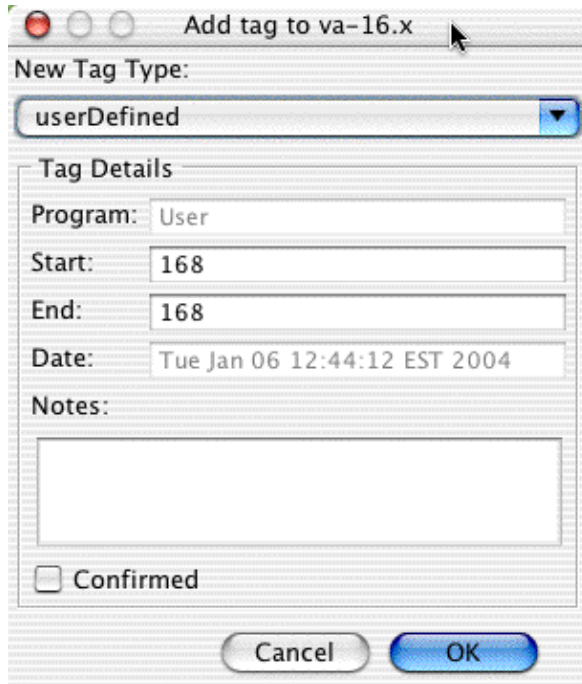
Viewing "Local" Tags

You can also view **tags at specific locations** in a sample from trace view, base view, and contig view windows. Select a base that contains a tag, and then right-click (OS X: control-click) on it to display the popup menu. The first item in the popup menu will be "Display Tag..." (but only if you clicked on a base with a tag). This will bring up the tag dialog as above, except that only tags at this position in the sample will be shown.

(Instead of using the popup menu, you can also go to the **"Sample"** menu, and selecting **"Show Local Tags..."** from the **"Tags"** sub-menu).

Adding Tags

You can also add your own tags to any base or range of bases in a sample. Open a view that shows the bases (a trace view, base view, or contig view), select the bases you want to tag, and then right-click (OS X: control-click) on it to display the popup menu. The first or second menu item in the popup menu will be "Add Tag...". Selecting it will bring up the following dialog:



The default type of the new tag is "userDefined". You can change this to any of the other pre-defined tag types using the pull down menu at the top. You can also edit the start and end position, and add comments about the tag in the "Notes:" text field. If you want a tag to extend to the start of a sequence, just enter "0" in the "Start:" field. If you want a tag to go to the end of the sample, enter a large number like "9999" in the "End" text field. Changes will be saved when you press the "OK" button.

*(Instead of using the popup menu to add tags, you can also go to the "**Sample**" menu, and selecting "**Add Tag...**" from the "**Tags**" sub-menu).*

Saving Edits

Edits to the consensus sequence or sample sequences are not saved until you save the project by selecting "**Save Project**" in the "**File**" menu, or using the corresponding toolbar buttons or keyboard shortcut (Command_S on OS X, Control-S on Windows).

If you are experimenting and think you might want to go back to a previous state that your project was in, you should save the project under a new name by going to the "**File**" menu and selecting "**Save Project As...**". This will save all data for your current project to a new location.

You can change the name of individual samples, which will then save them under a different name the next time you save the project, by selecting the sample and choosing "**Sample Information**" from the "**Sample**" menu, and editing the sample name. For more information, please read the "[Sample Information Dialog](#)" help page.

Undo and Redo

In the current version of Aligner, only limited support for undoing and re-doing is provided. Most actions, for example adding samples, assembling, unassembling, endclipping, and vector screening, cannot be undone. Therefore, please **save your project frequently!**

You can, however, undo simple edits. Just choose "**Undo**" from the "**Edit**" menu to undo an edit, and restore the sequence to the state it was before the edit. If you change your mind, you can choose "**Redo**" from the "**Edit**" menu to re-do the edit that you just undid. If you did anything else that is not undoable, "**Undo**" from the "**Edit**" menu will be not available (grayed out).

We plan to add Undo and Redo support for more actions in future Aligner releases. If undo support for a particular feature is important to you, please let us know about it!

Copy and Paste

Copy (selected sequence)

If your current selection is a single sample or contig, or part of a single sample, you can copy sequences and parts of sequences by selecting "**Copy (selected sequence)**" in the "**Edit**" menu.

If the active window is a **trace view**, **base view**, or **contig view**, then the currently selected bases will be copied to the clip board.

If the currently active window is the **project view**, and a single **sample** is selected, then the entire sequence of this sample will be copied to the clip board. If you have selected a single **contig** in the project view, then the consensus sequence for this contig will be copied. If you have more than one sample or contig selected, copy will not be available.

Paste

The "**Paste**" menu item in the "**Edit**" menu is disabled, since Aligner is not intended to be used as a text editor. You can, however, paste any sequence that you copied (see above) into programs outside of Aligner (for example BLAST web pages).

Please note that **any gaps in sequences will be removed** in the copied sequence, since most programs or web pages you might paste the sequence into will expect ungapped sequences. If you need to copy and paste gapped sequences, please let us know!

There are also some dialogs that support paste through the keyboard shortcuts (control-V on Windows, command-V on OS X). The most notable is the "**Search Sequence...**" dialog, accessible from the "**Go**" menu.

Editing Contigs

CodonCode Aligner lets you manipulate contigs in a number of different ways. You can edit the samples in a contig, as described in the previous section. Several other common tasks with contigs are:

- [Adding new samples to existing contigs](#)
- [Merging contigs](#)
- [Removing samples from contigs](#)
- [Reverse-complementing contigs](#)
- [Splitting contigs](#)
- [Unassembling contigs](#)
- [Editing contig information](#)

Adding Samples to Contigs

To add new samples to an existing contig, you need to first add the samples to the project, and do any pre-processing steps like base calling or end clipping that you want to do. Then, to add the sample to an existing contig:

1. Go to the project view
2. Select both the contig and the samples that you want to add to it
(to make continuous selections, use *shift-click*; to make discontinuous selections, use *control-click* on Windows and *command-click* on OS X)
3. Go to the "**Contig**" menu
4. Choose "**Assemble**" (or, if your contig is an alignment to a reference sequence, choose "**Align To Reference Sequence**")

Aligner will try to merge the samples with the contig. If samples share an overlap that meet the minimum criteria you defined, they will be added to the contig. Samples that do not overlap the contig, or where the overlap is not good enough, will remain in the Unassembled Samples folder.

You can also choose more than one contig and one or more samples - for example two neighboring contigs, and some finishing reads that bridge the gap between the contigs.

Alternatively, you can use **drag and drop** in the project view to add samples to contigs. First, select the samples and/or contigs you want to add in the project view, and then drag and drop them onto the contig to which you want to add them (this "target" contig cannot be in the initial selection). Once you release the mouse, CodonCode Aligner will start the assembly of the samples and contig(s).

When you use the "**Assemble**" or "**Align To Reference Sequence**" menu with existing contigs, the arrangement of the samples in the contig will remain the same; changes are limited to any gaps that need to be introduced, re-building of the consensus sequence, and possibly changing the extend of unaligned regions at the end of samples. In addition, the contig may be reverse-complemented during assembly.

If you do **not** care about keeping the current arrangement of the samples in a contig, you can choose "**Assemble from Scratch**" or "**Align To Reference from Scratch**" instead. This will first unassemble the existing contigs, and then re-assemble or re-align all reads that were in the contig(s), plus any other reads you have selected.

Duplicating samples

You can duplicate samples in a project, and create text sequences from consensus sequences, as follows:

1. Select the sample(s) or contig(s) you want to duplicate.
2. Go to the "**File**" menu.
3. Select "**Duplicate**".

This will create a copy of every selected sample, and a new text sequence for every selected contigs.

Before using this option to align consensus sequences, however, please look at the help for "[Compare contigs](#)" - in CodonCode Aligner, you can compare contigs directly to each other, and keep the relation to the sequences and their traces in a contig intact!

Merging Contigs

To merge two or more contigs:

1. Go to the project view
2. Select the contigs you want to merge (you can also select reads in "Unassembled Samples")
3. Choose "**Assemble**" from the "**Contig**" menu.

Aligner will look for overlaps between the contigs, and merge the contigs if the overlap meets the minimum assembly criteria. If the overlap between the contigs is not good enough, for example because it is very short or has a high discrepancy rate, Aligner will reject the merger, and leave the contigs as they were.

When you use "**Assemble**" with existing contigs, Aligner will compare the consensus sequences of the two contigs to look for an overlap. The relative arrangement of the samples in each contig will remain unchanged. Changes are limited to any gaps that need to be introduced, reverse-complementing (if needed), re-building of the consensus sequence, and possibly changing the extend of unaligned regions at the end of samples.

There are two alternatives to using the "**Assemble**" menu item: using "**Assemble with Phrap**" and "**Assemble from Scratch**". Both options will first unassemble the existing contigs, and then assemble the existing reads, using either Phrap or CodonCode Aligner's build-in assembly algorithm to build the contigs (for-profit users need to have a separate license for Phrap). The potential drawback of both of these options is that any edits made by moving gaps or samples around will be lost (but other edits, like change base calls or removed ends, will of course remain).

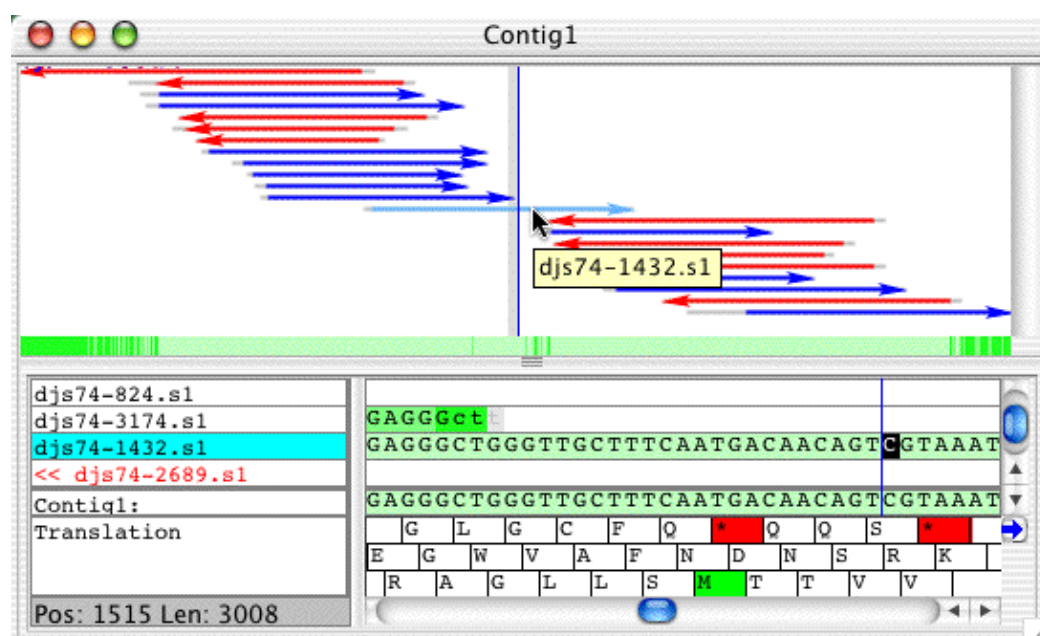
Removing Samples from Contigs

Occasionally, Aligner may put a sample into a contig where it does not belong, or you may want to remove a read from a contig for other reasons.

To remove a sample from a contig, select the sample in the contig view by clicking on its name or a base in it. Then, choose either **"Move to Unassembled Samples"** or **"Move to Trash"** from the **"Edit"** menu.

Before removing the sample from the contig, Aligner first checks if removing the sample would introduce any gaps in the contig. This happens if there are reads to both sides of the sample, and the sample is the only sample that covers one or more bases of the consensus sequence. In this case, removing the sample would in effect split the contig into two pieces, and (at least currently) Aligner will refuse to do so. If splitting the contig is what you wanted to do, you will first have to [split the contig](#) yourself, and then remove the sample from the new contigs formed after splitting.

An example where removing a read would split a contig into two is shown below:



Here, removing the selected sample (djs74-1432.s1) would split the contig into two parts. However, the other samples could be removed without problems, since they only cover regions that are also covered by other reads.

Deleting Parts of Contigs

To delete all bases from the current cursor position to the start of a contig:

1. Select a contig base in the contig view
2. Go to the "**Contig**" menu
3. Go to the "**Delete**" submenu
4. Select "**From Contig Start**" to delete all bases to the start of the contig,
or select "**To Contig End**" to delete all bases to the end of the contig

If any samples are completely within the removed contig region, these samples will be moved to the trash.

If the menu items are disabled, you probably do not have a contig base selected. You may have clicked on a sample accidentally, or perhaps a view other than the contig view is in the foreground.

***Note:** Deleting from the contig start or to the contig end **cannot be undone**, so you may want to save your project first, or to save a copy of your project using "**Save As**" in the "**File**" menu.*

Alignment Locations - Start and End

During contig building by assembly or alignment, Aligner automatically determines the useful parts of reads. For each sample, regions at the start and end with high levels of discrepancies to other reads will remain unaligned. These regions are typically shown dimmed in the contig view. Bases in the unaligned regions are not considered when determining the consensus sequence. Unaligned regions are not aligned, meaning that Aligner will not introduce gaps in unaligned regions.

You can change the start and end of sample alignments through two menu items in "**Sample**" menu: "**Mark Start Alignment Location**" and "**Mark End Alignment Location**". If you use one of these option to extend the aligned region of a sample, you will have to introduce any gaps required for proper alignment by hand.

If you import Phrap assemblies, please note that unaligned regions of samples may extend beyond the ends of a contig on both sides (the start and the end).

Reverse Complementing

To reverse complement a sample or contig:

- Go to the project view
- Select the contig (or a sample in the "Unassembled Samples" folder)
- Choose "**Reverse Complement**" from the "**Edit**" menu

In most views, you will be able to identify samples that have been reverse-complemented by the "<<" prefix before the name, and by the fact that the name is drawn in red. In the project view, you can identify reverse-complemented samples by the icon (it has red rather than black borders).

Reverse-complementing of unassembled samples is possible, but usually not necessary, since samples will be reverse-complemented as needed during assembly and alignment.

You cannot reverse-complement samples in contigs directly, since that would destroy the alignment to the consensus. If you select "**Reverse Complement**" when a contig view is in the foreground, the entire contig will be reverse-complemented.

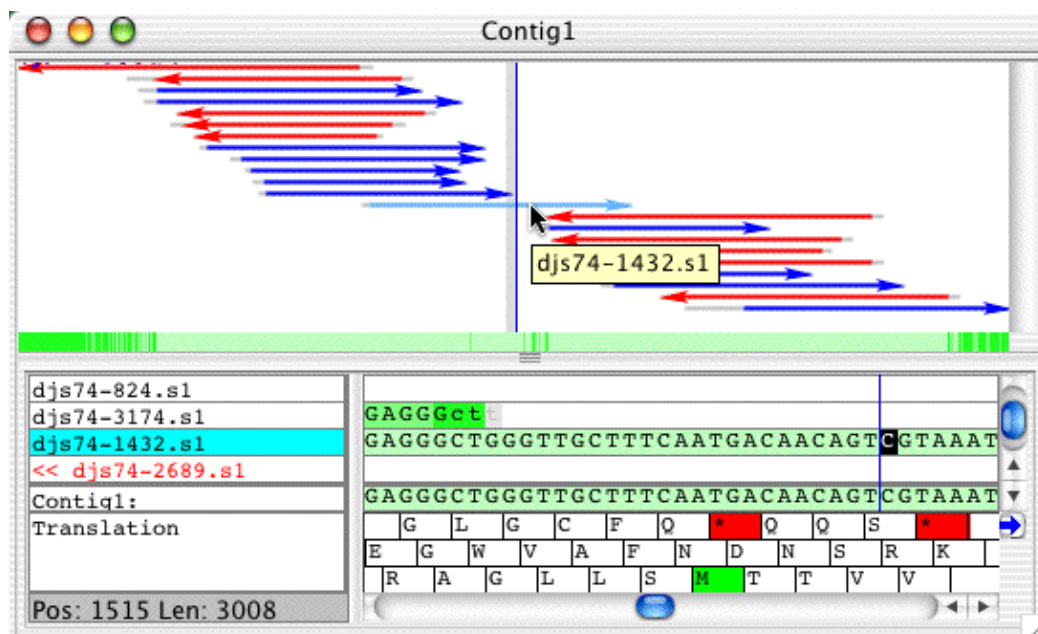
Splitting Contigs

Like other assemblers that use a "greedy" assembly algorithm, CodonCode Aligner will occasional misassemble contigs. You can manually split misassembled contigs into two parts as follows:

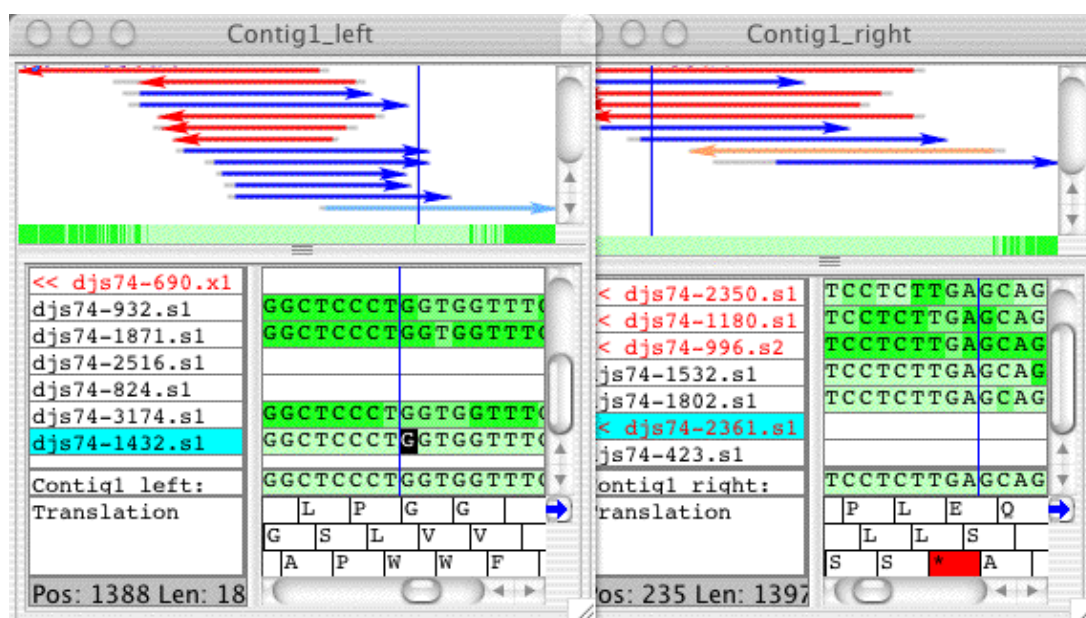
- Open the contig view for the contig in question
- Set the cursor to the position where you want to split up the contig
- Select **"Split contig"** from the **"Contig"** menu

Aligner will create two new contigs - one containing the samples that were to the left of the point where you split the contig, and one containing the samples that were to the right. For any samples that had aligned bases at the position of the split, Aligner will put this either into the new right or the new left contig; the right contig will be chosen for samples that had more aligned bases to the right of the split point, and the left contig for samples that had more aligned bases to the left of it.

Here's an example of a contig before splitting:



After splitting at the position indicated (1448), the two new contigs look like this:



Note that the sample djs74-1432.s1 ended up in the leftt contig.

Unassembling Contigs

To unassemble one or more contigs:

- Go to the project view
- Select the contig(s)
- Choose "**Unassemble**" from the "**Contig**" menu

All samples in the contig will be put into the "Unassembled Samples" folder, and any gaps in the samples will be removed. Any samples that were reverse-complemented in the contigs will be returned to the original (not reverse-complemented) state.

Rountrip Editing

While CodonCode Aligner support manual editing of contigs, Aligner does not provide all the functionality of programs that were specifically developed to edit sequence alignments. However, CodonCode Aligner supports "Roundtrip Editing" - exporting existing contigs for editing in external programs (like MacClade or Mesquite), and re-importing the manually edited contigs. The re-imported contigs maintain the connection to the original chromatograms, so you can quickly check any discrepancies by looking at the original sequence traces.

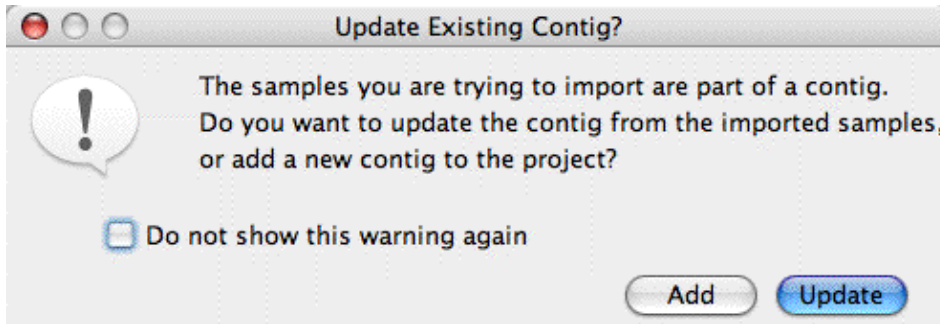
A typical example where roundtrip editing can make sense is a phylogenetic project, where you go through the following steps:

- Create a project in CodonCode Aligner with sequence traces from a number of different species (isolates, ...)
- Create separate contigs for each species (for example using "[Assemble in Groups](#)")
- Edit the initial contigs
- Create an alignment of the contig sequences (a "contig of contigs") with Clustal W from within CodonCode Aligner (using "[Compare](#)" in "[Assemble With Options](#)")
- Export the contig of contigs for editing in MacClade (or similar)
- Use MacClade to move gaps around
- Export the edited assembly from MacClade
- Re-import the edited assembly into CodonCode Aligner, thereby updating the position of gaps in the contig of contigs
- Verify and edit discrepancies between the species in CodonCode Aligner; you can quickly go back to the original sequence traces by double-clicking in the contig view for the contig of contigs
- Export the final assembly in PIR format (or similar) for further analysis

How To Roundtrip Edit with CodonCode Aligner

Here's how to roundtrip edit in CodonCode Aligner. The starting point assumes you have a project where you already have a contig (or contig of contigs) that you would like to edit with an external editor like MacClade.

1. **Check names** - before exporting, check the names of the contigs you want to export. Many export formats and external editors restrict how long names can be, and which characters can be used in names. To be safe, use short names with only numbers and letters. Names of 10 or fewer characters are safest, although up to 30 characters may also work.
2. Select the contig(s) to export in the project view.
3. Go to the **File** menu, and select **Export > Assembly**. From the format pulldown menu, select a format that works for the program you want to use, for example "Interleaved NEXUS/PAUP", and then export the file.
4. Import the file you just created in your favorite sequence editor, for example MacClade.
5. Edit the alignment, but limit your edits to moving gaps.
6. Export the edited alignment in NBRF/PIR format (with gaps).
7. Open the NBRF/PIR file in CodonCode Aligner, using drag and drop or **File > Open Sample....** You should see the following dialog:



Click on "Update". Aligner will import the sequences, and update the existing contig to reflect your edits.

Limitations

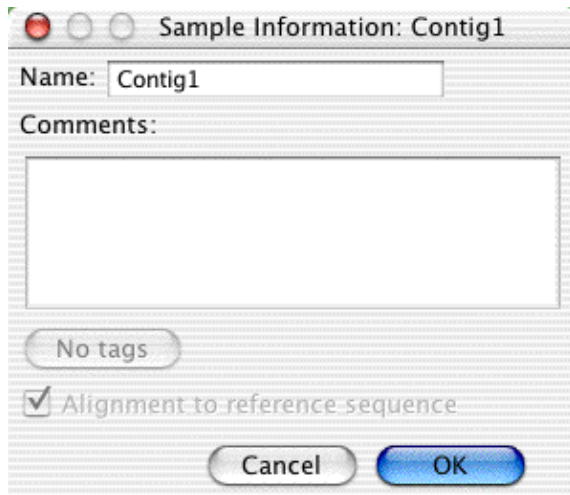
Currently, this "Roundtrip Editing" functionality is limited. You can move gaps around in external editing programs, but you cannot change or delete any bases. Here is a list of additional restrictions:

- Updating existing contigs will work only when importing files in NBRF/PIR format
- All sequences in the file must be exactly the same length, including leading and trailing gaps (*note that some programs (e.g. BioEdit) may not always make sequences of equal length when exporting in NBRF/PIR format*)
- The names of the samples in the imported file must be identical to (or sufficiently similar to) the names already in the project. *Note that some programs do not allow spaces, dashes, and other characters in sequence names, and that many programs and file formats restrict the length of sequence names.*
- The only edits currently supported are gap movements; if you edit, insert, or delete bases, the updating will fail.
- If the exported contig had unaligned (dangling) ends, the updating will likely fail. *One way to create dangling ends is to generate contigs of contigs using the built-in algorithm, and then remove samples or contigs at the beginning or end of the contig. This cannot happen if the contig of contigs is generated with ClustalW (since ClustalW creates end-to-end alignments).*

We plan to remove some of these restrictions in future releases.

Editing Contig Information

You can rename contigs, and add comments, by selecting the contig in the project view, and then choosing "Contig Information" in the "Contig" menu. This will bring up the following dialog box:



You can change the name in the "Name:" textfield at the top. You can also add remarks about the contig in the larger text area below; these will be shown in the "Comments" column in the project view.

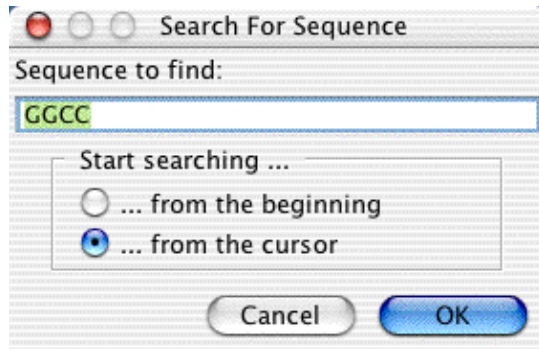
If a contig has tags (for example "polymorphism" tags added from finding mutations), the button that is labeled "No tags" in the image above will be labeled "Tags...", and become active. Clicking on it will bring up a tag dialog that shows all tags for the consensus sequence.

The checkbox at the bottom indicated whether a contigs is an alignment to a references sequence (as in the image above), or a regular assembly.

Please note that contig names and remarks, as well as tags added to a consensus sequence, will change or get lost when you later unassemble the contig or merge it with other contigs or unassembled reads.

Search for Sequences

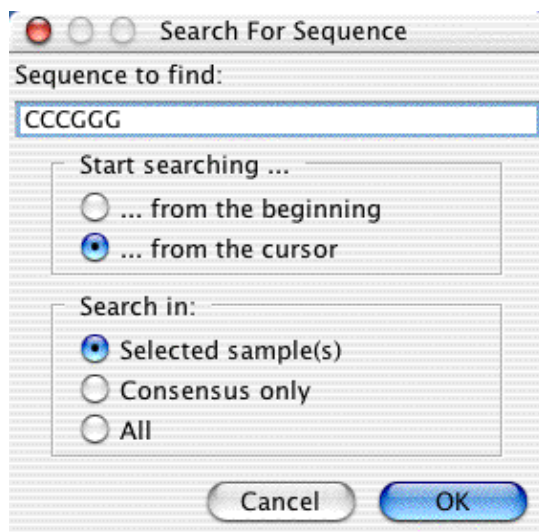
Aligner allows you to look for short sequences within your samples and contigs. First, select the sample or contig that you want to look in, and open a base view, trace view, or contig view for it. Then, selected **"Search Sequence..."** from the **"Go"** menu. This will display the following dialog:



You type the sequence to find in the text box at the top, and then press the "OK" button. The sequence you can look for may contain the four bases A,G,C, and T, as well as ambiguity codes; gaps are ignored. If you search for ambiguity codes, note that you will get a match only if the ambiguity code is found (for example, 'AGTN' will match 'AGTN', but not 'AGTC').

If Aligner finds your search string, it will position the cursor to the first occurrence; if Aligner cannot find the sequence, it will tell you so.

If you started the search from a contig window, the dialog will be slightly different:



When searching in contigs, you can specify where you want to look for your search sequence - only in the currently selected sample(s), only in the consensus, or in all sequences and the consensus.

Note: You can repeat any search by choosing **Search Again** from the **Go** menu.

BLAST Searches

To start a BLAST search from CodonCode Aligner:

- Select the sample or contig that you want to BLAST in the project view, **or**:
select the bases you want to search in a contig view, trace view, or base view
- Go to the "Go" menu, and select one of the options from the "**BLAST Search**" submenu.

Aligner will open web browser page for the NCBI BLAST server, and paste the selected sequence into the "Search" field. You can now change the search options or database to search against on this web page, and start the BLAST search. (If your security settings are very strict so that Aligner is not allowed to open a browser page, nothing may happen - you will need to change the security settings first).

For all BLAST searches except MegaBLAST, the selection must be a single sequence or part of a single sequence. For MegaBLAST, your selection can include more than one sequence.

MegaBLAST

[MegaBLAST](#) is a BLAST version that has been optimized for aligning sequences that differ only slightly, for example because of sequencing errors. It can be up to 10 times faster than other BLAST versions, and allows the submission of multiple sequences in a single search.

Nucleotide (blastn)

The "Nucleotide (blastn)" option can be used to initiate a comparison of nucleotide sequences against nucleotide databases.

Translated (blastx)

The "Translated (blastx)" option can be used to initiate a comparison of a protein translation of your nucleotide sequences against protein databases.

Translated (tblastx)

The "Translated (tblastx)" option can be used to initiate a comparison of a protein translation of your nucleotide sequences against a protein translation of the nucleotide databases.

For more information about BLAST, please visit <http://www.ncbi.nlm.nih.gov/blast/>.

Exporting

Sooner or later, you will probably want to use the data you see with Aligner with other programs. For example, you may want to use BLAST to compare the consensus sequences to nucleotide databases, or you may want to export read names and reads lengths for importing into spreadsheets or data bases.

You can export the key data in Aligner projects using the various sub-menus in the "**Export**" menu in the "**File**" menu. To export samples or consensus sequences, first select what you want to export in the project view. Then, choose one of the following export options:

- [Export Project Summary...](#)
- [Export Samples...](#)
- [Export Consensus Sequences...](#)
- [Export Features...](#)

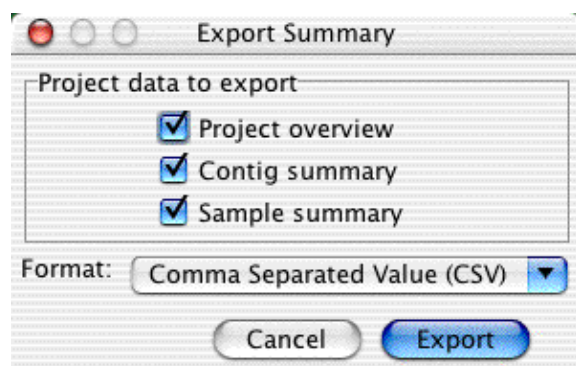
Aligner will then show a dialog box, where you can set options like file formats. Next, Aligner will present a standard "Save As" dialog where you can choose the folder for your exported files and (if you are exporting to single files) the file names. By default, all files will be exported into a folder called "**export_dir**" inside the project directory. You can choose to export to any directory you want, but be careful not to accidentally overwrite any existing files Aligner might need!

If you are exporting multiple items to individual files, Aligner will automatically determine the file names and extensions. If any files with the same name already exist in the folder where you want to save your files, Aligner will warn you that the existing files will be replaced.

Export Project Summary

This allows you to export project data like sample names, read lengths, and so on to text files that can be imported into external spreadsheet programs or databases. The information exported will be the information you see in the project view window, and (optionally) summary information about your project.

To export project summary information, go to the **"File"** menu, and select **"Project Summary..."** from the **"Export"** sub-menu. This will show the following dialog:



When you press the "Export" button, a standard "Save" dialog appears, where you can choose the name and location of the exported file. The file is a text file, and can be imported into word processors or spread sheets like Microsoft Excel (*you may need to select "All Files" in the "Open" dialog to be able to import the file*).

Here is an example of what the exported data look like after importing into Microsoft Excel:

PolyPhred_summary.csv						
	A	B	C	D	E	F
1	Project Name	PolyPhred.proj				
2	Date and Time of export	11/21/03 10:01				
3	Project location	/Users/Shared/Documents/Projects/PolyPhred/				
4	Contig count	2				
5	Sample count	10				
6	Total sample bases	9018				
7	Average sample length	901				
8	Average sample quality	387				
9	Average contig length	597				
10	Average contig quality	0				
11						
12	Contig Name	Number of Samples	Contig Length	Quality	Coverage	Estimated Error Rate
13	Unassembled Samples	0	0	0	0	0.00E+00
14	Contig1	4	473	0	3.9	2.20E-02
15	Contig2	6	722	0	4	3.55E-01
16	Trash	0	0	0	0	0.00E+00
17						
18	Sample Name	Sample Location	Sample Length	Quality	Direction	Contig Offset
19	va-1.x	Contig1	1058	372	Fwd	0
20	va-13.x	Contig1	1020	406	Fwd	0
21	va-23.x	Contig1	1240	354	Fwd	-1
22	va-16.x	Contig1	1025	382	Fwd	1
23	ca-9.r	Contig2	869	415	Rev	-396
24	ca-22.r	Contig2	860	400	Rev	-385
25	ca-22.s	Contig2	765	305	Fwd	18
26	ca-9.s	Contig2	742	394	Fwd	19
27	ca-21.s	Contig2	745	406	Fwd	18
28	ca-23.s	Contig2	694	437	Fwd	21
29						

PolyPhred_summary.csv

Ready Sum=0

Exporting Samples

You can export samples to text files in FASTA format, or to chromatogram files in SCF ("Standard Chromatogram Format") format, as follows:

- Go to the project view
- Select the samples you want to export (*use shift-click to make continuous selections, and control-click (OS X: command-click) to make discontinuous selections*)
- Choose **"Export => Samples"** in the **"File"** menu.

This will display the following dialog:



Using the radio buttons at the top, select to export just the selected samples, or all samples in the project. The Format pull-down menu gives you the following format choices:

Single FASTA file

This will generate a single text file in FASTA format which contains all the exported sequences. You can specify the name and location of the file in a "Save As" dialog box that will be shown when you click the "Export" button.

When exporting FASTA files, you have the option to include gap characters in the output but checking the box labeled "Include gaps in FASTA files". If the box is not checked, the exported sequences will be ungapped.

If the sequences have qualities, then a quality file in FASTA format will also be created in the same file as the FASTA file. The name of the quality file will be the name of the FASTA file, with ".qual" appended at the end of the name.

Individual FASTA files

This option allows you to create a separate file for each sample. Again, this is a text file in FASTA format; each file contains exactly one sequence. You can choose the folder where the files are created in a "Save As" dialog that will be shown once you click on the "Export" button. All files will be created in the same folder; the names of the files will be the sample name, with ".fasta" appended.

When exporting FASTA files, you have the option to include gap characters in the output but checking the box labeled "Include gaps in FASTA files". If the box is not checked, the exported sequences will be ungapped.

If the sequences have qualities, then a separate quality file in FASTA format will also be created for each sample. The name of the quality file will be the name of the corresponding FASTA file, with ".qual" appended at the end of the name.

SCF files

SCF files contain the current base calls as well as the chromatogram data, and allow you to import the sequences into programs that support the standard SCF format. You can choose the folder where the files are created in a "Save As" dialog that will be shown once you click on the "Export" button. All files will be created in the same folder; the names of the files will be the sample name.

Exporting Consensus Sequences

To export the consensus sequences of contigs, select the contigs of interest in the project view, and then choose **"Export Consensus Sequences..."** from the **"File"** menu. This will bring up the following dialog:



Using the radio buttons at the top, select to export the consensus sequences for just the selected samples, or all contigs in the project. The Format pull-down menu gives you the following format choices:

Single FASTA file

This will generate a single text file in FASTA format which contains all the exported consensus sequences. You can specify the name and location of the file in a "Save As" dialog box that will be shown when you click the "Export" button.

Aligner will also create a quality file in FASTA-like format. The name of the quality file will be the name of the FASTA file, with ".qual" appended at the end of the name.

Individual FASTA files

This option allows you to create a separate file for each consensus sequence. Again, this is a text file in FASTA format; each file contains exactly one sequence. You can choose the folder where the files are created in a "Save As" dialog that will be shown once you click on the "Export" button. All files will be created in the same folder; the names of the files will be the sample name, with ".fasta" appended.

Aligner will also create a separate quality file in FASTA format for each sample. The name of the quality file will be the name of the corresponding FASTA file, with ".qual" appended at the end of the name.

When exporting FASTA files, you have the option to include gap characters in the output but checking the box labeled "Include gaps in FASTA files". If the box is not checked, the exported sequences will be ungapped.

Exporting Assemblies

You can export Aligner projects for importing into other programs, for example the contig editor Consed. You can export entire Aligner projects, or parts of Aligner projects, for example single contigs.

First, select the contig(s) you want to export in the project window. Then, choose "**Export Assembly...**" from the "**File**" menu. This will bring up the following dialog:



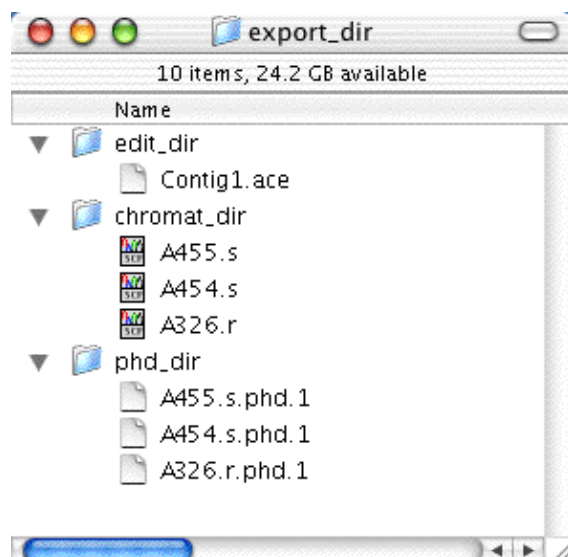
You can choose to export the current selection or the entire project. Currently, only two formats are supported:

- the "[ACE](#)" format that is used by the Phred-Phrap-Consed package, and also supported by other contig editors
- the "[NEXUS/PAUP](#)" format, which is used by many phylogenetic analysis programs
- the "[Phylip](#)" format, another format often used by phylogenetic analysis programs

Please note that many formats may not really export the entire selection or project. For example, the ACE file generated when exporting in the ACE file format will contain only contigs, but not unassembled reads or samples in the trash.

ACE Format Exports

The ACE format for exporting projects is modeled after the format used by the Phred-Phrap-Consed package, which is also supported by other sequence editors. Exporting will create three directories, as illustrated below:



The folder "edit_dir" contains a single file, the ".ace" file. This is a text file that contains the information about the assembly, for example the number of contigs and the consensus sequences.

The second folder, "chromat_dir", contains the chromatogram files format for the exported traces. The trace files are in SCF ("Standard Chromatogram Format") format.

The third folder, called "phd_dir", contains "PHD" files for each sample. The PDH files are text files which contain the base calls, qualities, and additional information like tags.

You should be able to directly open the exported project in Consed, and (hopefully) in other contig editors that support the ACE file format. However, if you transfer the exported files to a different operating system, make sure that the files are transferred correctly. The chromatogram files must be transferred as binary files, and the other files must be transferred as text files. Incorrectly transferred files will likely cause problems when you try to open them.

NEXUS/PAUP Format Exports

The NEXUS/PAUP format allows you to export projects for phylogenetic analysis programs like MacClade or PAUP. Only contigs will be exported, with a single file being created for each contig. Note that the exported files only contain information about the bases and gaps, but not about associated chromatograms or sequence qualities.

You have a choice of exporting in either "interleaved" or "sequential" format. Different phylogenetic programs have different restrictions on which kind of files they can read, so you may need to try both to see which one works for the program you plan to use.

If you have sample names that are very long and/or contain spaces or other "unusual" characters, it may be necessary to truncate or change the name of the samples in the exported files. Again, different programs have different restrictions. If sample names may to be changed in the exported files, Aligner will present you with a list of choices on how to change the sample names.

Phylip Format Exports

The Phylip format allows you to export projects for phylogenetic analysis programs like MacClade or PAUP. Only contigs will be exported, with a single file being created for each contig. Note that the exported files only contain information about the bases and gaps, but not about associated chromatograms or sequence qualities.

You have a choice of exporting in either "interleaved" or "sequential" format. Different phylogenetic programs have different restrictions on which kind of files they can read, so you may need to try both to see which one works for the program you plan to use.

If you have sample names that are very long and/or contain spaces or other "unusual" characters, it may be necessary to truncate or change the name of the samples in the exported files. Again, different programs have different restrictions. If sample names may to be changed in the exported files, Aligner will present you with a list of choices on how to change the sample names.

Other Formats for Exporting Assemblies

We plan to add support for other formats in the future. If you need a specific format, please let us know the format and the program that you need it for.

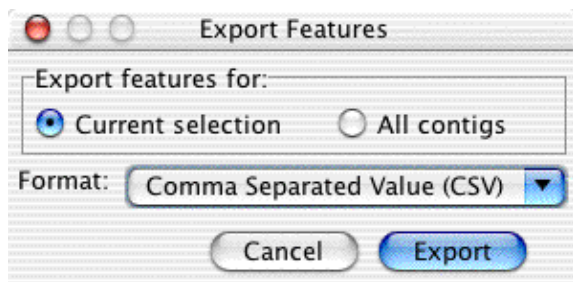
Exporting Features

In Aligner, "Features" are regions which, for one reason or another, deserve special attention. They can include discrepancies, regions of low coverage, bases with tags, or a number or other criteria which you can define in the [feature preferences](#).

To export features for a set of contigs:

- Go to the contig view
- Select the contig(s) for which you want to export features (or, if you want to export features for all contigs, select any contig)
- Choose **"Export Features"** in the **"File"** menu.

A dialog will appear where you can specify details about the output format:

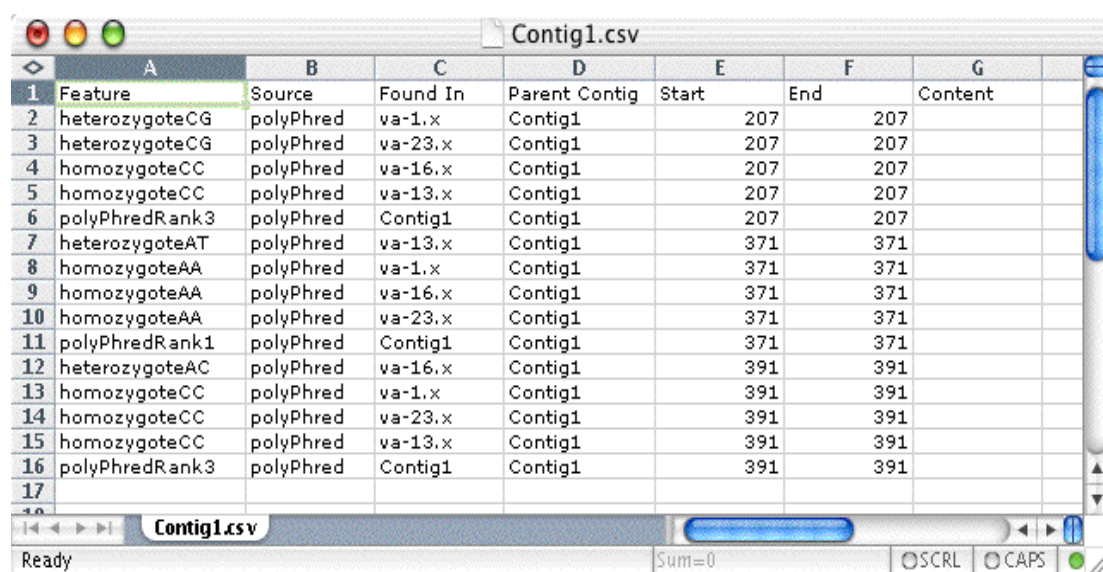


Use the radio button at the top to select whether you want to export only the selected sample or all samples in the contig. Then, use the pull-down menu to select the file format, and click on the "Export" button. Next, Aligner will show you a "Save As" dialog where you can select the location of the exported file or files. The exported files are text files, and can be opened with text editor or spread sheet programs.

One thing to note when exporting features is that CodonCode Aligner will always export all features in the consensus sequence and in all samples in a given contig. This is true even if you used the [Feature preferences](#) to specify to look for features only in the consensus sequences or the samples or the current selection when navigating (this selection does apply only for navigating, not for feature views or exports).

Here is an example of what exported features from the "PolyPhred_example" project can look like after importing an exported feature file into Microsoft Excel:

CodonCode Aligner User Manual



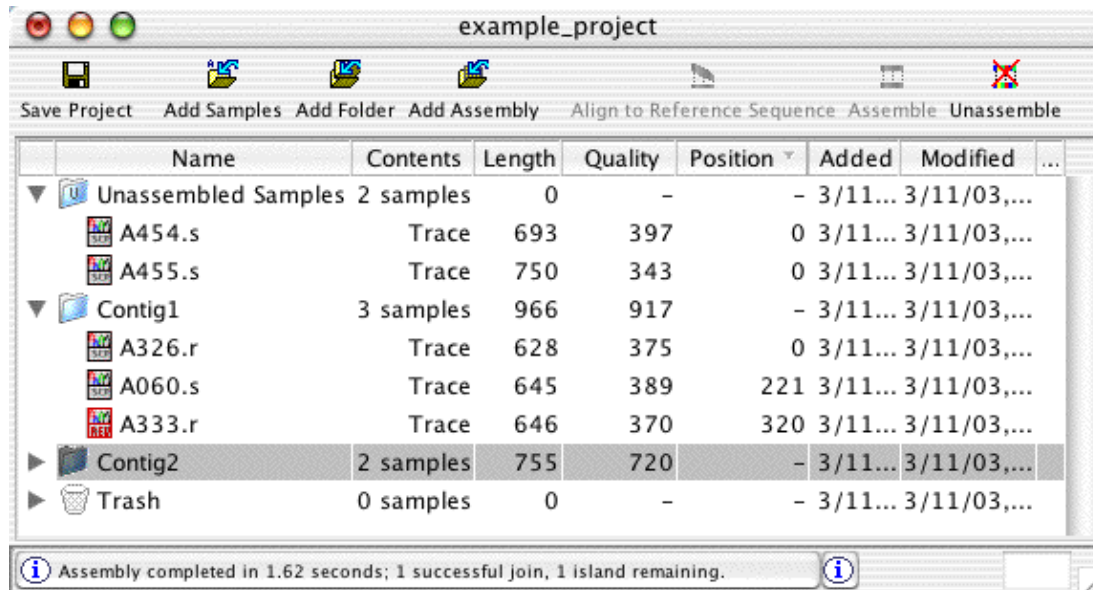
	A	B	C	D	E	F	G
1	Feature	Source	Found In	Parent Contig	Start	End	Content
2	heterozygoteCG	polyPhred	va-1.x	Contig1	207	207	
3	heterozygoteCG	polyPhred	va-23.x	Contig1	207	207	
4	homozygoteCC	polyPhred	va-16.x	Contig1	207	207	
5	homozygoteCC	polyPhred	va-13.x	Contig1	207	207	
6	polyPhredRank3	polyPhred	Contig1	Contig1	207	207	
7	heterozygoteAT	polyPhred	va-13.x	Contig1	371	371	
8	homozygoteAA	polyPhred	va-1.x	Contig1	371	371	
9	homozygoteAA	polyPhred	va-16.x	Contig1	371	371	
10	homozygoteAA	polyPhred	va-23.x	Contig1	371	371	
11	polyPhredRank1	polyPhred	Contig1	Contig1	371	371	
12	heterozygoteAC	polyPhred	va-16.x	Contig1	391	391	
13	homozygoteCC	polyPhred	va-1.x	Contig1	391	391	
14	homozygoteCC	polyPhred	va-23.x	Contig1	391	391	
15	homozygoteCC	polyPhred	va-13.x	Contig1	391	391	
16	polyPhredRank3	polyPhred	Contig1	Contig1	391	391	
17							

Contig1.csv

Ready Sum=0 SCRL CAPS

The Project View Window

The project window is the main window in CodonCode Aligner - most actions are started by selecting samples or contigs in the project window, and then choosing a menu option or one of the buttons at the top of the project window:



The project window is organized similar to the familiar list view in the Macintosh Finder, or the detail view in Windows Explorer. Every project has two folders:

- The "Unassembled Samples" folder contains samples newly added to a project, and not yet assembled or moved to the trash.
- The "Trash" folder, which contains samples marked for deletion, but not yet removed from the project.

In addition, projects that have been assembled or aligned will have one folder for each contig formed. You can **expand or condense folders** by clicking on the little triangle on the left of each folder. In the example above, the "Unassembled Samples" folder and Contig1 are expanded, showing details about the samples in Unassembled Samples and in Contig1.

Selecting Samples and Contigs

For most things you do in CodonCode Aligner, you start by selecting the samples or contigs you want to do something with, for example vector screen or assemble. You can:

- **click** on a sample or contig to select the item
- **shift-click** to make a continuous selection (press on the first item, then press shift while clicking on the last item you want to select)
- **control-click** (OS X: **command-click**) to make a discontinuous selection

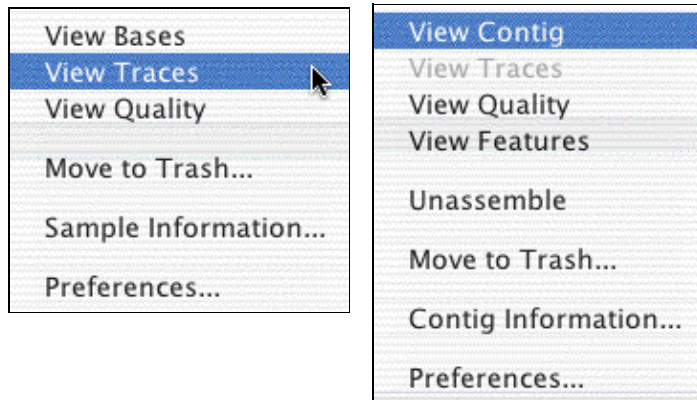
Your current selection determines which menu items and buttons are active. For example, you need to have at least two samples or contigs selected before you can choose "Assemble" from the "Contig" menu, or the "Assemble" button at the top of the window; the "Unassemble" menu item and button require that your

selection contains only contigs; and so on. If a menu item is not available (and the button is "grayed out"), you have not selected the item or items required for this action.

Menu shortcuts: Buttons and Popup menus

The most commonly used actions are available as buttons on the top of the project view, and as popup menus. You can hide the buttons by selecting "**Toolbars > Hide Toolbars**" in the "**View**" menu. If you do not see the toolbar at the top, select "**Toolbars > Show Toolbars**" in the "**View**" menu to make them visible.

If you press the right mouse button on Windows, or control-click on Mac OS X, in the project window, a **popup menu** with the most commonly used actions will be displayed. The content of the popup menu will depend on what you currently have selected - just try it out! Two example popup menus are shown below:



Project View Columns And Sorting

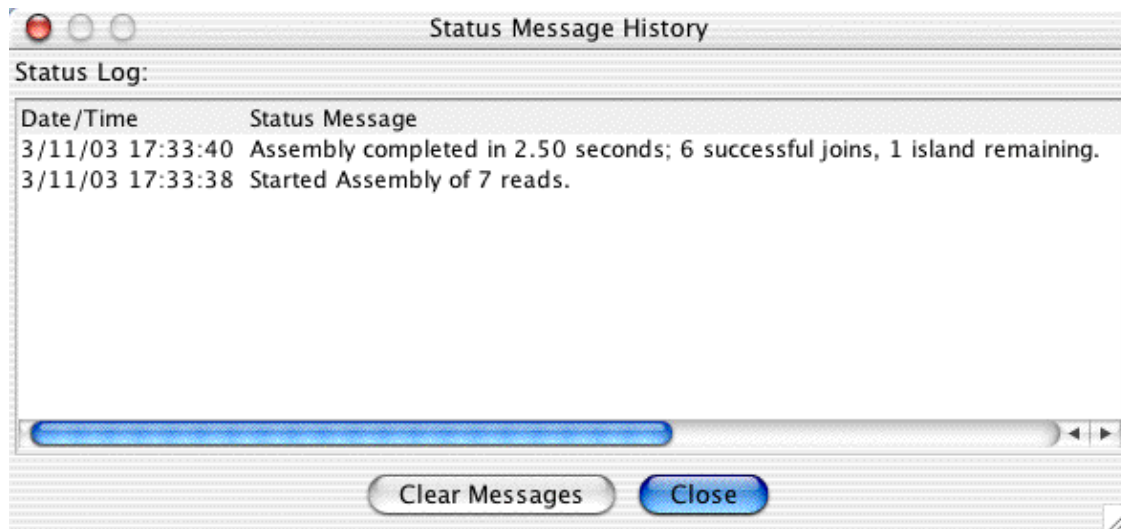
The project view contains a number of columns, most of which are pretty self-explanatory. You can **click on the column headers to sort** the project view; however, the "Unassembled Samples" folder will always be on the top, and the "Trash" folder at the bottom. Here are descriptions of the columns, from left to right:

- the **Triangles** in the left-most column let you expand or condense contigs and other "folders"
- the **Name** column shows the name of the folder or sample; you can change sample and contig names through the "Sample Information" and "Contig Information" dialogs, accessible through the "Sample" and "Contig" menus
- the **Contents** column tells you details about what is in a folder or contig, or about the samples
- the **Length** column shows the length of a contig or sample
- the **Quality** column displays the number of bases with a quality score of 20 or above ("Phred20 bases") for samples and contigs that have qualities
- the **Position** column shows the position of the first base of a sample within a contig (only for samples in contigs, not for unassembled samples or samples in the trash)
- the **Added** column shows the date a sample was added to the project
- the **Modified** column shows the date a sample was last edited or otherwise modified
- the **Comments** column shows and comments you (or a program) may have added to the sample

Status messages

The area at the bottom of the project view window is used by Aligner to display messages - status messages, warning, and error messages. If you click on this area, the "Message history" dialog will be displayed, which

shows all previous messages:

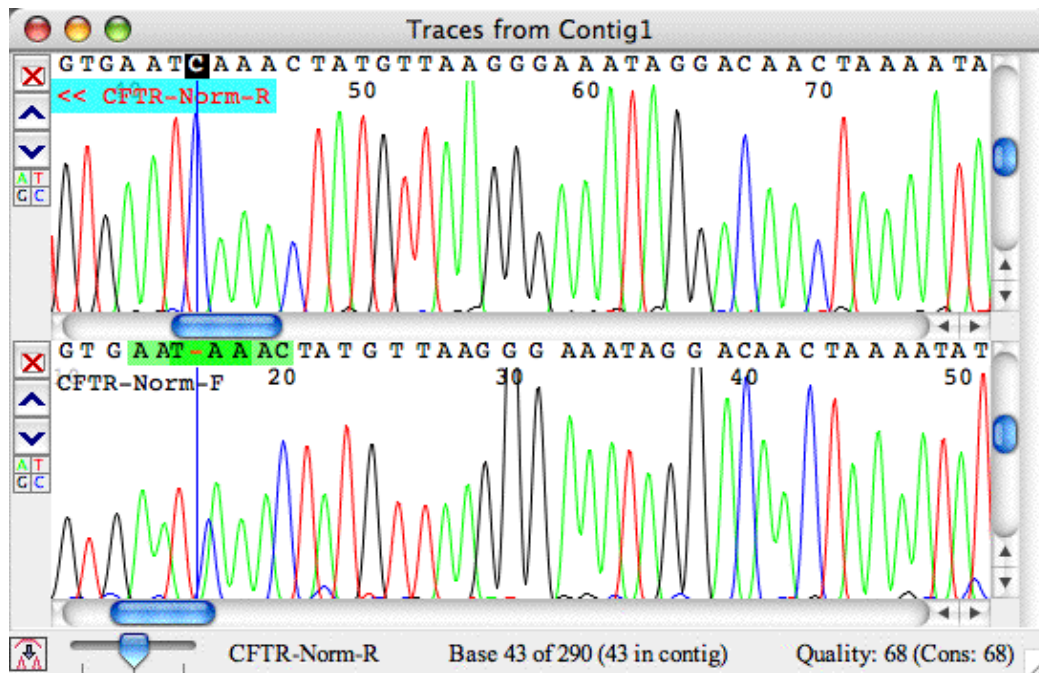


You will have to close this window before you can do anything else in Aligner. You can press the "Clear Messages" button to erase all old messages.

Trace View

General Information

In Trace View, you can view and edit samples that have trace data. Files with trace data that can be opened in a Trace View include standard chromatogram files (*.scf) from LI-COR and other manufacturers, and Applied Biosystems ABI trace files (*.abi, *.ab1).



Each contig has one trace view window, in which all selected traces for this contig will be shown. The same is true for the "Unassembled Samples" folder - it has its own trace view window. The number of traces in a trace view window is limited only by available memory and operating system limitations; however, displaying many traces may be slow.

Opening a Trace View

A trace view for a sample that has trace data associated with it can be opened by:

- Selecting the sample, then choosing **Trace View** from the **View** menu.
- Right clicking on the sample and choosing **Trace View**.
- Double-clicking on the sample (if you selected to open a trace view in the [double clicking preferences](#)).

Scrolling and Scaling in Trace View

The vertical sliders at the right side of each trace let you scale a trace vertically to make the peaks larger or smaller.

You can use the sliders at the bottom of each sample to scroll a trace. If the "Scroll together" checkbox is checked, then all traces will be scrolled together when you scroll one of the traces.

If the trace view window contains more traces than currently fit on the screen, the vertical slider on the right allows you to scroll between the traces. You can remove traces from the trace view window by clicking on the read "X" button on the left. You can move individual traces up and down by clicking on the up and down arrows on the left.

If you have a mouse with a scroll wheel, you can also use the scroll wheel to move around in the trace view window:

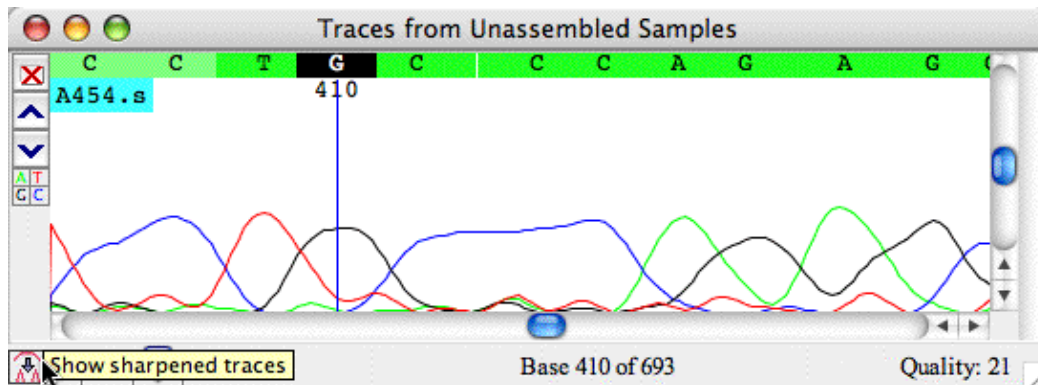
- when the mouse pointer is positioned over a trace, moving the scroll up or down will move horizontally (to the right or to the left)
- to scale a trace view **vertically**, keep the **control key** pressed while scrolling with the mouse wheel: this has the same effect as using the vertical scroll bar, making peaks larger or smaller
- to scroll between traces if the trace view window contains more traces than currently fit on the screen, move the mouse pointer over the vertical slider on the right allows, and then use the scroll wheel

You can change the default height of trace view panels in the "[Views](#)" preferences.

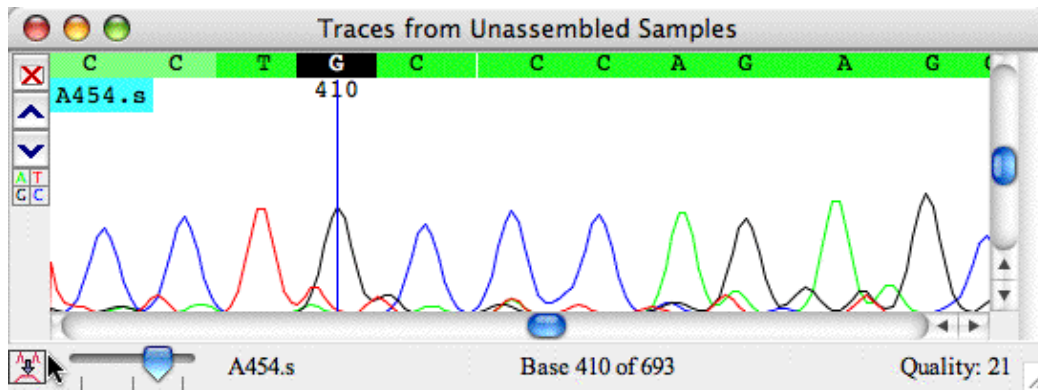
To scale the traces horizontally, use the slider in the bottom left corner of the trace view.\

Trace Sharpening

If you are working with traces where the peaks overlap and are badly resolved, CodonCode Aligner's trace sharpening can help you to get a better look at your data. Here is an example of a badly resolved region in a sequence:



When you click on the little button in the lower right corner, CodonCode Aligner will try to sharpen the peaks in the displayed traces; the result looks like this:



In this example, it is much easier to see that there are indeed 3 Cs after the G, not 2 or 4. Looking at the sharpened traces can be a useful tool when working with less-than-perfect data.

To go back to the original traces, just click the little button in the bottom left corner again. You will notice that showing the sharpened traces may take a few seconds the first time, especially if the trace view shows several traces. This is because the trace sharpening is rather computation intensive.

When looking at the sharpened traces, keep in mind that you are looking at a mathematical artifact, which occasionally may give an incorrect impression - so always look back at the original traces, too, and use your scientific judgement.

Colors and Highlighting

The colors in the Trace view are determined by your [color preferences](#); in the example above, a quality-based 3-color scheme was used. Your [highlighting preferences](#) determine how discrepancies, edits, and tags are displayed. For more information, please check the "[Preferences](#)" help section.

Automatic Trace Selection

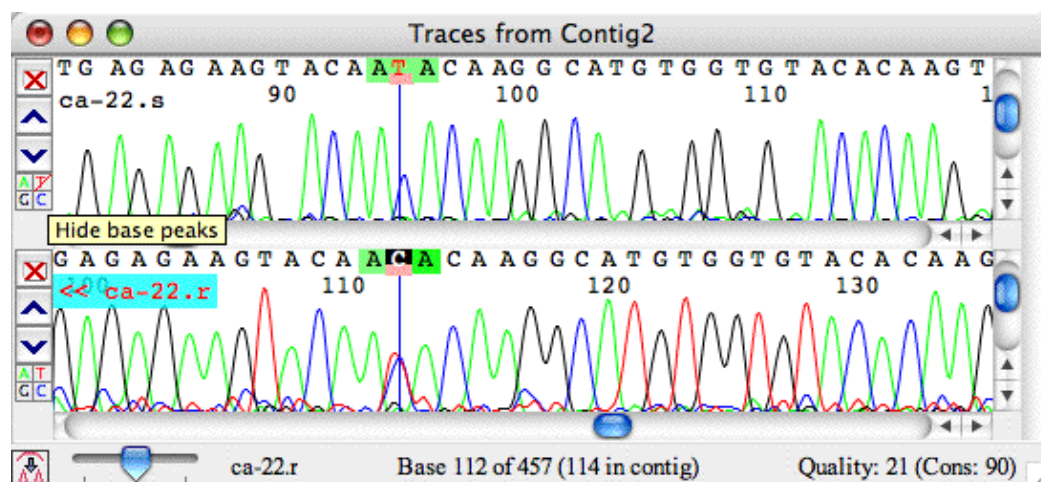
For trace views that show samples in **contigs**, CodonCode Aligner can automatically pick traces to display while you are moving around in a contig. You can turn this option on or off by selecting "**Auto Select Traces**" in the "**View**" menu. You can also set the number of traces to be displayed in the "[Views](#)" preferences.

For automatic trace selection to work, you need to first open both the contig view and the corresponding trace view (for example by double-clicking in the overview panel in the contig view). Once both views are open, Aligner will automatically add and remove traces when you go to a new location in a contig, for example by clicking in the overview panel or by moving the cursor in the bases panel. The trace view window will grow larger and smaller, depending on the number of traces shown and the available space on your screen. If you want to see more than 2 or 3 traces, you may want to change the height of the trace panels to "Small" or "Tiny" in the "[Views](#)" preferences.

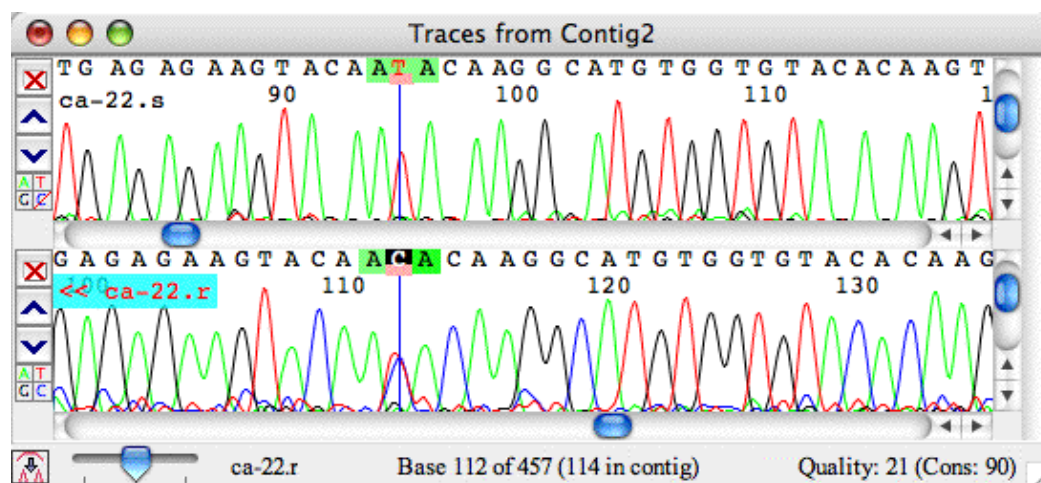
Hiding Some Traces

CodonCode Aligner User Manual

You can hide traces for one or more bases by clicking on little colored boxes with the corresponding letter of the trace on the left side. This can be very useful when analyzing heterozygous mutations, where a peak for one base may obscure a second peak. Below is an example - the same traces as above, but the 'T' lane is hidden:



The red line through the 'T' box at the left indicates that the T lane is hidden. Clicking on it again will show the T lane, and clicking on the 'C' will hide the C lane, as shown below:



Hiding lanes will affect only one trace, and only until it is closed; re-opening the trace will show all traces again.

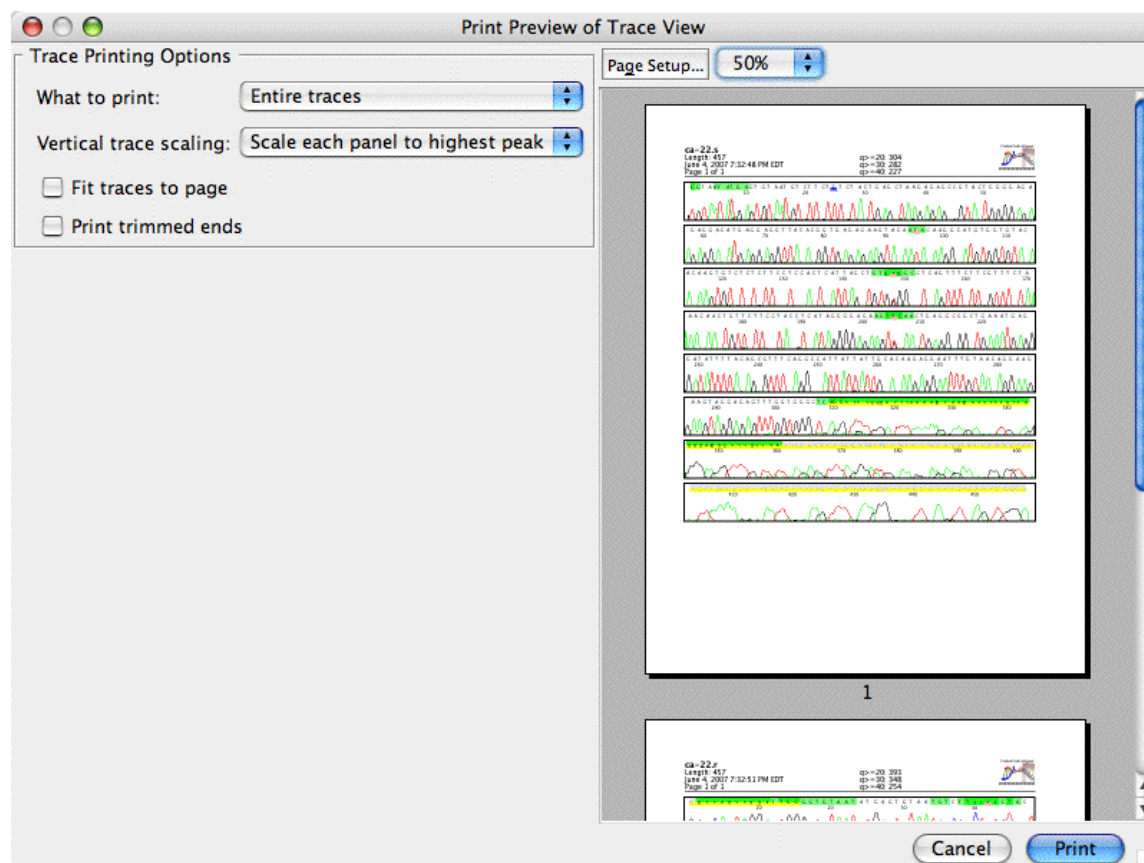
You can also press the shift-key and/or the alt-key while clicking in the base boxes to hide more than just one trace:

- **Shift-click** to hide all traces except the one for the base you are clicking on in the current sample; shift-click again to show all traces again
- **Alt-click** to hide the trace for this base in **all** samples in the trace view window; alt-click again to show the trace in all windows again (OS X users can also use the command key instead of the alt key)
- **Alt-Shift-click** to hide all other traces in all samples. Repeat to show all traces in all samples again. Try it out!

Printing Traces

To print the traces that are currently in your trace view, first select the trace view, and then choose "**Print**" from the "**File**" menu (or use the keyboard shortcut Control-P on Windows, Command-P on OS X).

This will show the a print preview dialog like the one below:



On the left, you can choose whether to print the entire traces or just what you see on the screen, how to scale traces, and a few more options. On the right side, you get a preview of the print results with your current settings. The different options are described in detail on the "[Printing Preferences](#)" page.

Base View Window

Choose **Base View** from the **View** menu to display the bases for a sample or consensus:

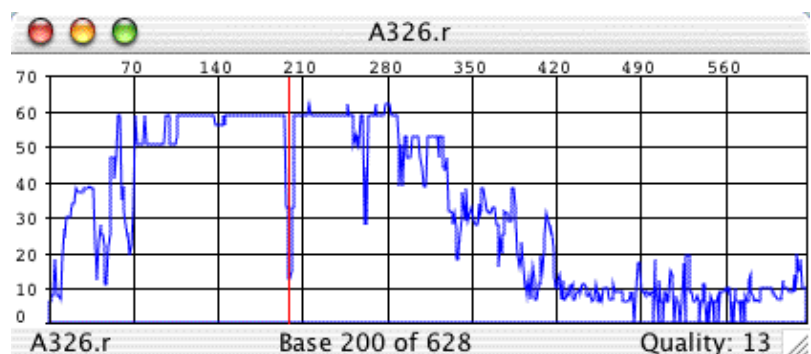


You can edit sequences in the base view if they are editable; but for sequences with traces, it's generally a better idea to edit in the trace view. The bases are displayed according to your [color preferences](#).

At the bottom of the base view window, you see the name of the sample, the current cursor position, and (if the sample has qualities) the quality at the cursor.

Quality View Window

Choose **Quality View** from the **View** menu to display a graph of the quality values for the selected sample or consensus.

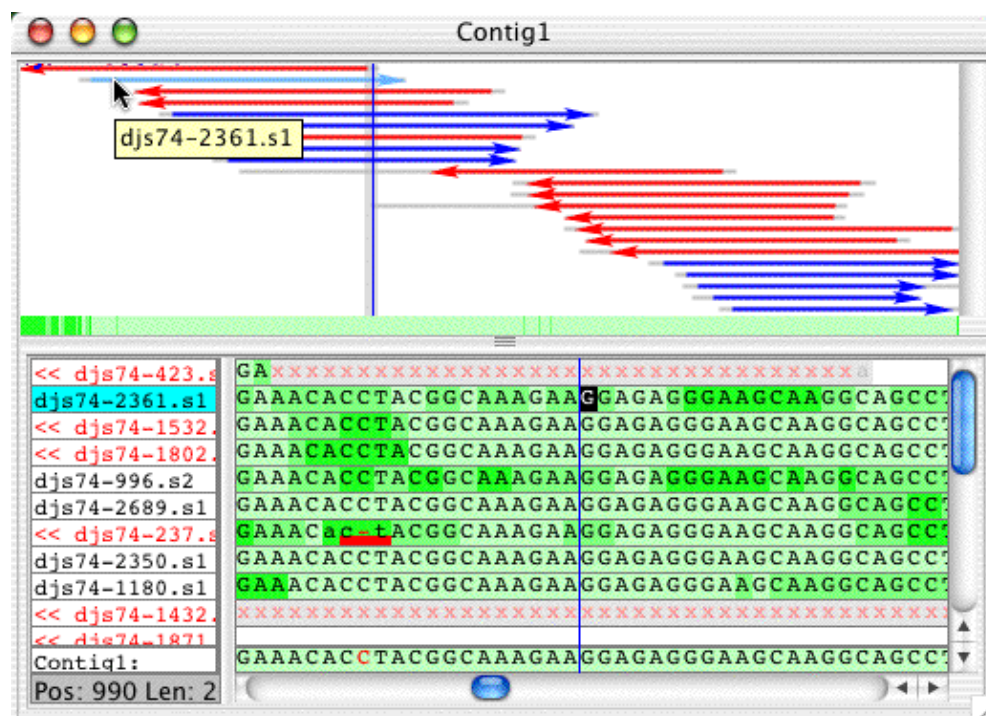


The **Quality View** graphs [quality value](#) (Y-axis) versus base position (X-axis) for all base calls. The quality graph can be enlarged or reduced by resizing the Quality View window (click and drag a corner of the window).

Clicking in the quality view window will move the cursor in all views for this sample. This can be useful to quickly check out problem regions, like the low quality region near base 200 in the example above. Just open a trace view and a quality view for a sample, and click at the low-quality regions in the quality view; then look at this region in the trace view.

Contig View Window

Choose **Contig View** from the **View** menu to display a window showing an overview and the aligned bases for a contig:



Contig Overview Panel

The upper half of the contig view shows a **graphical overview** of the contig (the "overview panel"). Some things worth noting:

- The location and orientation of individual samples in the contig is shown by arrows
- Unaligned ("dangling") ends of reads are shown in a light gray color (in projects where ends were clipped before assembly, samples will typically have no or just short unaligned ends)
- A vertical blue line shows the current cursor position in the contig
- The currently selected sample or samples are drawn in a higher color (in the example, the second read is selected)
- Moving the mouse over an arrow will display the name of a read (after a little while)
- You can click in the overview panel to move around in a contig, as explained in the next section

Navigating using the overview panel

You can **click** in the overview panel to move the cursor. If you click on an arrow, the corresponding read will be selected, and selected read and base will be shown in the lower "aligned bases" panel. If you click somewhere else (in the white spaces), the corresponding consensus bases will be selected (you may have to use the vertical scroll bar to see bases for aligned reads at this location).

Moving Around

You can move around in the aligned bases panel by using the scroll bar at the bottom, or by using the keyboard. When moving around in a contig view, the gray rectangle moves, indicating that a different part of the contig is shown; but the blue line for the cursor position does not change until you also click on a base in the aligned base panel.

Using the keyboard, you can move around as follows:

- Move around with the right and left arrow keys; you move one base at a time.
- Press the "page up" and "page down" keys a few times; this moves you by a full screen forward or backwards.
- Try the "home" and "end" keys; they move you to the beginning respectively end of the selected sequence. If the consensus sequence is selected, you go to the beginning and end of the contig; if you select a read, you move to the beginning or end of the read.

If you have a mouse with a scroll wheel, you can also use the scroll wheel to move around in a contig. Moving the scroll down will move forward in a contig, moving up will move backwards. If you have a contig with many reads in it, then pressing the "control" key while scrolling with the mouse wheel will move the read list up and down.

Contig quality in the overview panel

At the bottom of the overview panel, the consensus quality is displayed as a horizontal bar. The color-coding depends on your color preferences; in the example above, three shades of green were used, with darker colors corresponding to lower quality. You can use this to quickly navigate to low-quality consensus regions. However, keep in mind that the resolution may not be sufficient to show all low-quality regions in this overview, especially for larger contigs.

Aligned Bases and Protein Translation

The lower half of the contig view shows the **aligned bases** for the samples in a contig. It can also show the **protein translation**, depending on how your preferences are set. In the example above, the translation for one frame is shown; you can choose to show no translation, one frame, three frames, or six frames in the [protein translation preferences](#) and in the "View" menu. The cursor position is indicated by a black line, and the currently selected base or bases by a different background color (in the example above by black background). You can use the the cursor keys and the horizontal scroll bar at the bottom to move around in a contig.

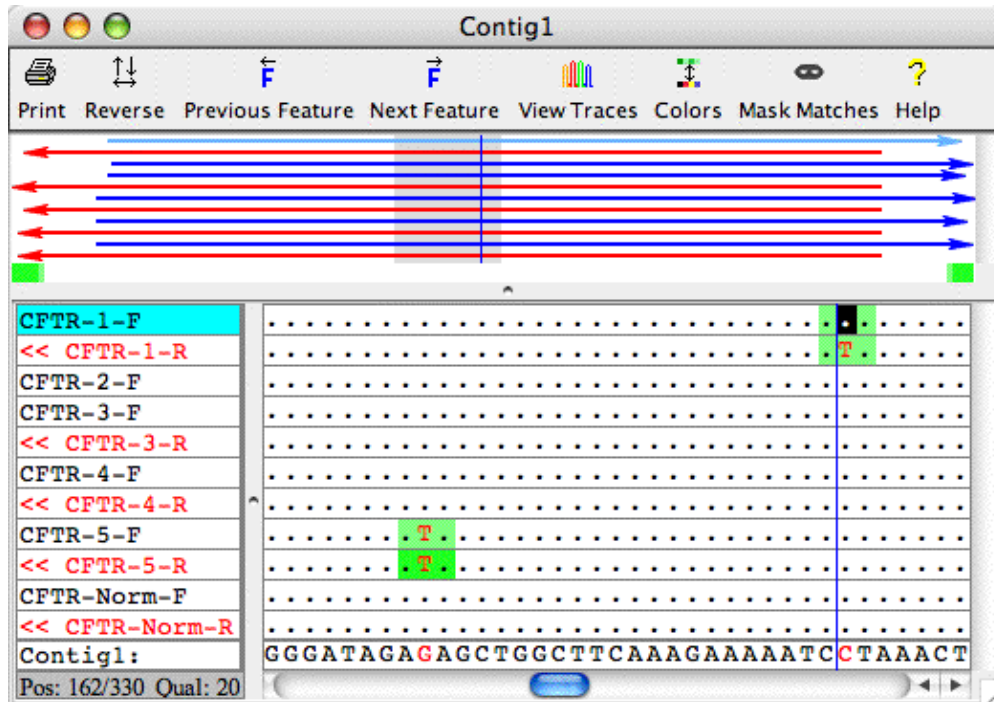
You can **resize** the panels by dragging the bar between the panels.

The bases are displayed according to your [color preferences](#). In the picture above, a quality-based three-color scheme was used.

The names of the samples are shown on the left side of the aligned bases panel. Reverse-complemented samples are indicated by "<<" before the name, and by **red** sample names. You can **select** samples by clicking on the sample.

Masking Bases Matching the Consensus

One way to focus on bases that differ from the consensus sequence is by going to the "**View**" menu, and selecting "**Mask Bases Matching Consensus**". If checked, any bases in aligned regions that are identical to the consensus sequences are replaced by dots; only bases that differ from the consensus sequence, and the consensus sequence itself, are shown as letters. Here is an example:



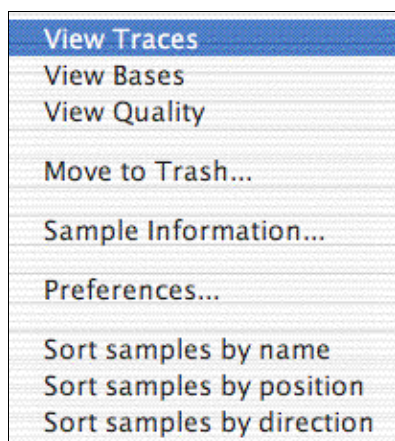
To revert back to the normal display where all bases are drawn, just select "**Mask Bases Matching Consensus**" (or the corresponding toolbar button) again.

Sorting Reads in the Contig View

In the contig view, samples can be sorted in one of several ways:

- by position in the contig (the default)
- by sample name
- by direction (forward or reverse) and then by position in the contig
- by the user

The default sort method for samples in the contig view can be set in the [contig view preferences](#). If you change the sort method in the contig view preferences, the new setting will affect only contig view windows opened after the change. To change the sorting of reads in an open contig view window, right-click (OS X: command-click) anywhere in the "aligned bases" panel to display a popup menu like this:



Select any of the "Sort samples by ..." options at the bottom of the menu, and the reads in this contig view will be re-sorted.

Typically, sorting samples by position will be the best way to sort samples. For re-sequencing and mutation detection projects, sorting by name may work well if you followed a naming scheme that uses similar names for forward- and reverse reads, so that sorting by name will have the forward- and reverse reads right underneath each other.

You can also **manually** sort the samples by selecting and dragging the sample names up or down. After manually sorting samples in a contig, Aligner will keep your manual sort order for this contig until you select one of the sort options in the popup menu; changing the sorting in the contig view preferences will **not** change the sorting of contigs that have been sorted manually.

Automatic Trace Selection

When checking or editing a contig, you will often want to look at a few traces at a given position. CodonCode Aligner can automatically pick traces to display while you are moving around in a contig. You can turn this option on or off by selecting "**Auto Select Traces**" in the "**View**" menu. You can also set the number of traces to be displayed in the "[Views](#)" preferences.

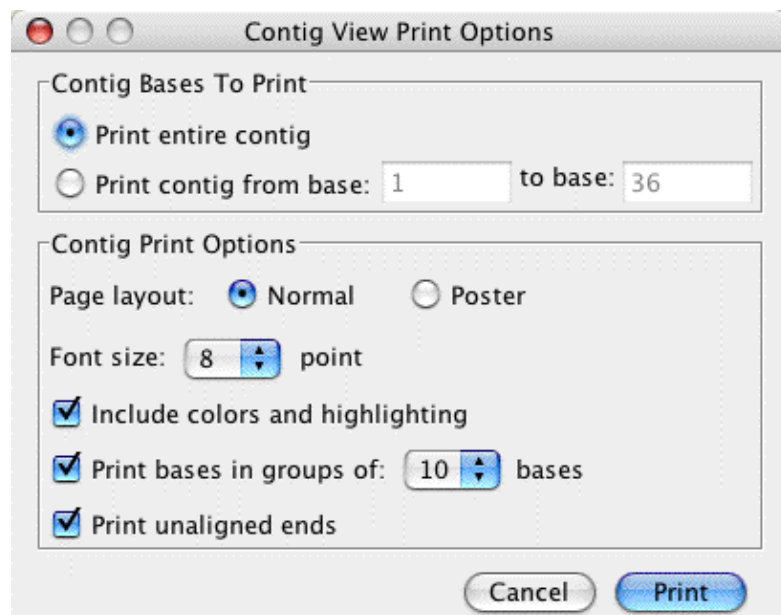
For automatic trace selection to work, you need to first open both the contig view and the corresponding trace view (for example by double-clicking in the overview panel in the contig view). Once both views are open, Aligner will automatically add and remove traces when you go to a new location in a contig, for example by clicking in the overview panel or by moving the cursor in the bases panel. The trace view window will grow larger and smaller, depending on the number of traces shown and the available space on your screen. If you want to see more than 2 or 3 traces, you may want to change the height of the trace panels to "Small" or "Tiny" in the "[Views](#)" preferences.

Printing Contigs

To print a contig:

- open and select the contig view for the contig you want to print
- then, choose "**Print**" from the "**File**" menu (or use the keyboard shortcut Control-P on Windows, Command-P on OS X)

This will show the following option dialog:

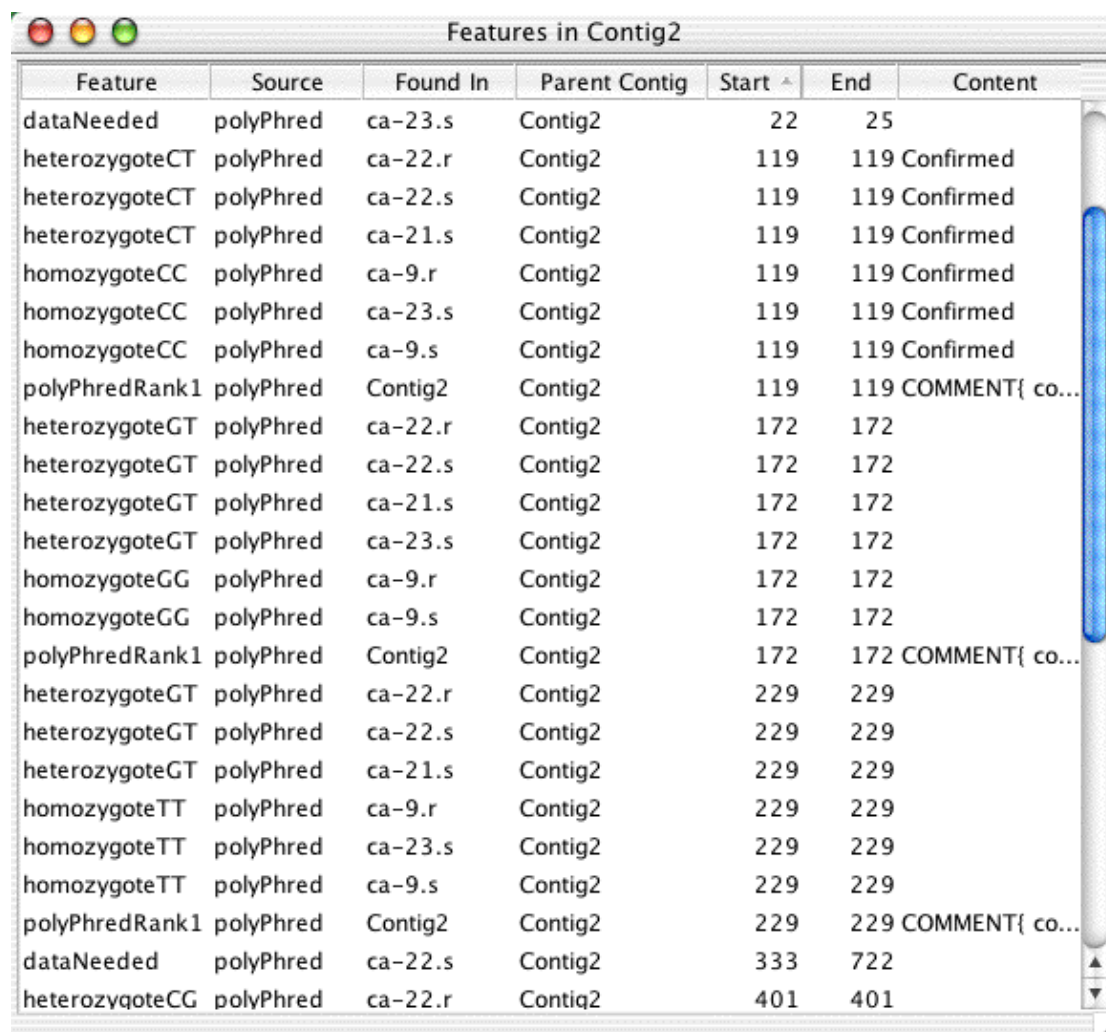


The top section of the dialog lets you choose how much of the contig to print - either the entire contig, or just a section of the contig. Aligner automatically fills in the first and last bases visible in the contig view as the range for printing, but you can edit both numbers to print any section of your contig.

The options in the lower half are described in detail in the [Printing Preferences](#).

Feature Window

Select a contig in the project view, and then choose **Feature View** from the **View** menu to display a window showing features for a contig:



The screenshot shows a window titled "Features in Contig2" with a table of genomic features. The table has seven columns: Feature, Source, Found In, Parent Contig, Start, End, and Content. The features are listed in rows, with some having associated content like "Confirmed" or "COMMENT{ co...".

Feature	Source	Found In	Parent Contig	Start	End	Content
dataNeeded	polyPhred	ca-23.s	Contig2	22	25	
heterozygoteCT	polyPhred	ca-22.r	Contig2	119	119	Confirmed
heterozygoteCT	polyPhred	ca-22.s	Contig2	119	119	Confirmed
heterozygoteCT	polyPhred	ca-21.s	Contig2	119	119	Confirmed
homozygoteCC	polyPhred	ca-9.r	Contig2	119	119	Confirmed
homozygoteCC	polyPhred	ca-23.s	Contig2	119	119	Confirmed
homozygoteCC	polyPhred	ca-9.s	Contig2	119	119	Confirmed
polyPhredRank1	polyPhred	Contig2	Contig2	119	119	COMMENT{ co...
heterozygoteGT	polyPhred	ca-22.r	Contig2	172	172	
heterozygoteGT	polyPhred	ca-22.s	Contig2	172	172	
heterozygoteGT	polyPhred	ca-21.s	Contig2	172	172	
heterozygoteGT	polyPhred	ca-23.s	Contig2	172	172	
homozygoteGG	polyPhred	ca-9.r	Contig2	172	172	
homozygoteGG	polyPhred	ca-9.s	Contig2	172	172	
polyPhredRank1	polyPhred	Contig2	Contig2	172	172	COMMENT{ co...
heterozygoteGT	polyPhred	ca-22.r	Contig2	229	229	
heterozygoteGT	polyPhred	ca-22.s	Contig2	229	229	
heterozygoteGT	polyPhred	ca-21.s	Contig2	229	229	
homozygoteTT	polyPhred	ca-9.r	Contig2	229	229	
homozygoteTT	polyPhred	ca-23.s	Contig2	229	229	
homozygoteTT	polyPhred	ca-9.s	Contig2	229	229	
polyPhredRank1	polyPhred	Contig2	Contig2	229	229	COMMENT{ co...
dataNeeded	polyPhred	ca-22.s	Contig2	333	722	
heterozygoteCG	polyPhred	ca-22.r	Contig2	401	401	

A feature view window shows the features for one or more contigs, depending on your selection when you opened the feature view. You can also look at the features for all samples in the "Unassembled Samples" folder by selecting it in the project view, and then opening a feature view.

You can determine which features are shown in the [feature preferences](#). In the example above, low quality consensus regions and tags are shown.

You can **sort** the table by clicking on the column headers. You can also print and export features through the corresponding file menu items (you may have to switch to the project view first, and select the contig or contigs you want to print or export in the project view).

You can **double-click** on any item in the feature view to take a closer look at the feature. This will open the views as defined in the [double-clicking preferences](#) - typically the trace view for the sample, and the contig view for the corresponding contig (unless you have changed the preferences).

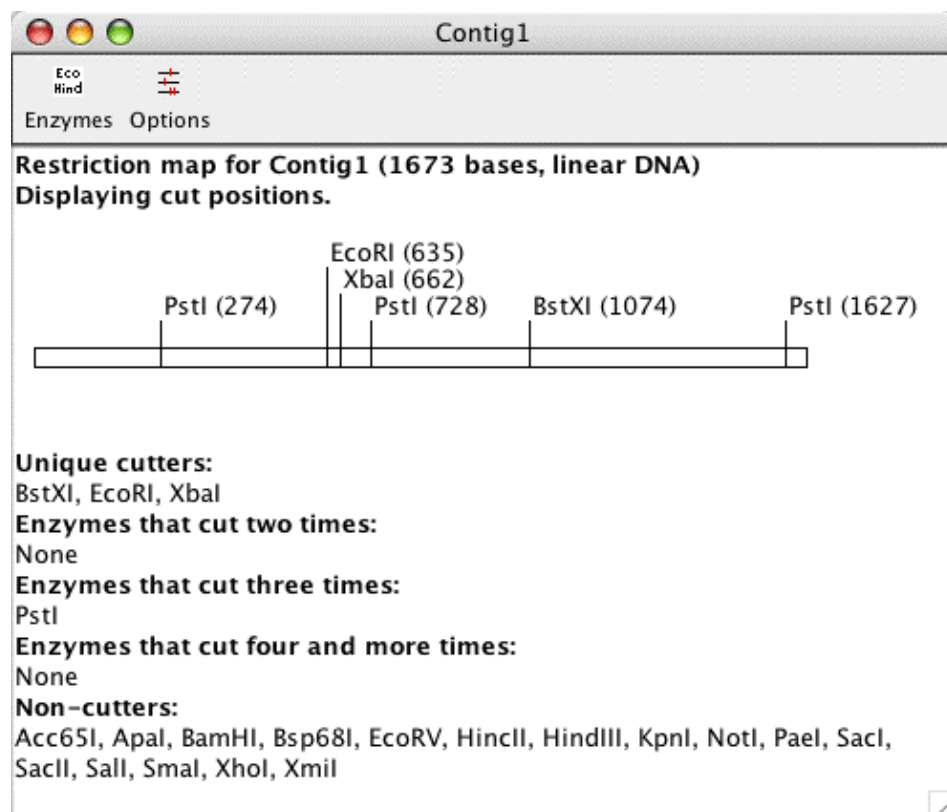
Restriction Map View

Choose **Restriction Map** from the **View** menu to display a window showing the restriction map for the selected sample or contig.

A restriction map can be displayed in three different map styles: **Single Line Map**, **Multiple Line Map** and **Text Map**.

Single Line Map

The **single line map** is shown below:

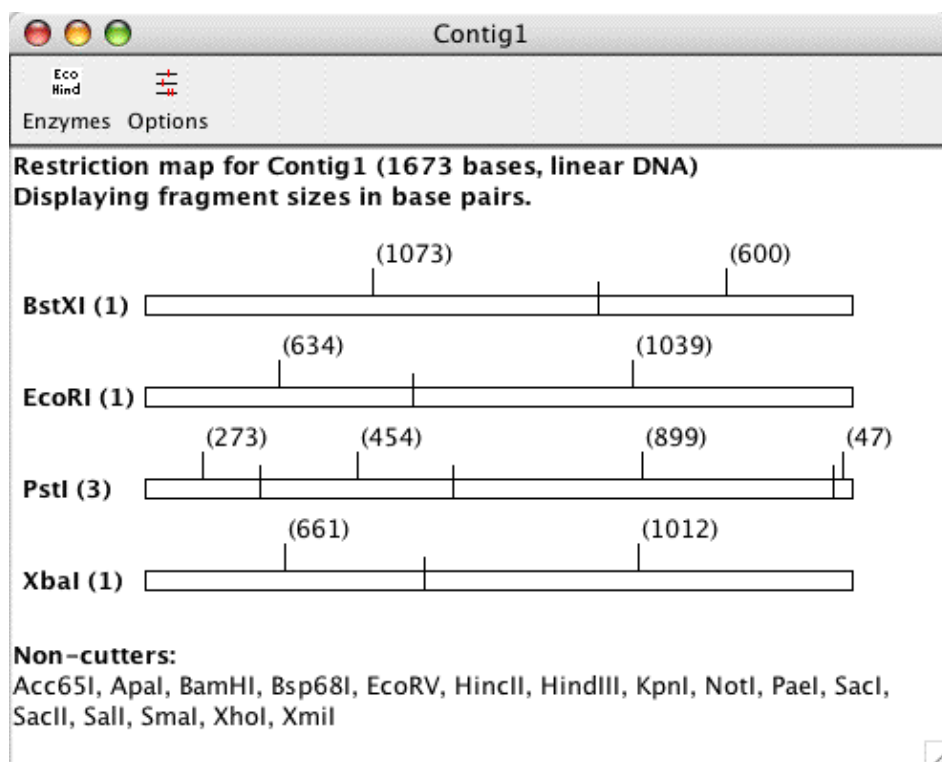


In the example above, you see a single line map for linear DNA that displays **cut positions**. The cut positions are shown behind the enzyme names. A **summary** of the cutters and non-cutters is shown at the bottom of the map.

You can change how the map is displayed and what enzymes to use in the [restriction map preferences](#). You can open the preferences through the icons in the toolbar of the restriction map view.

Multi Line Map

The **multi line map** for the same example is shown below:

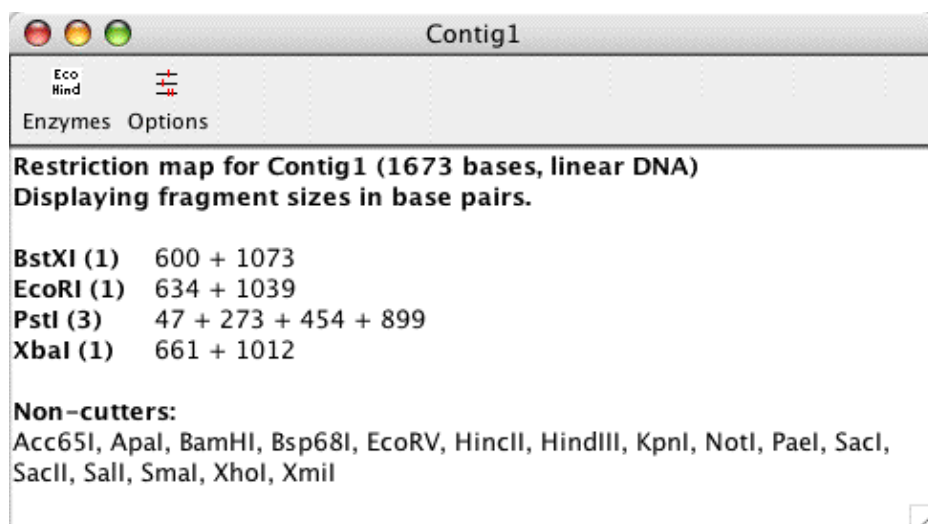


The multiple line map displays a separate graph for each enzyme. The number after the enzyme name tells you how often this enzyme cuts. In the screen shot above, **fragment sizes** (rather than cut positions as in the first screen shot) are displayed. The fragment sizes are displayed in base pairs in the middle of each fragment. For example, BstXI cuts one time, generating two fragments with 1073 bp and 600 bp.

At the bottom of the map a list of all enzymes that do not cut in "Contig1" is given. It is possible to shown only cutting enzymes, only non-cutting enzymes, both, or no summary at all. You can choose what to display in the [restriction map preferences](#).

Text Map

The **text map** for the same example, displaying fragment sizes, is shown below:

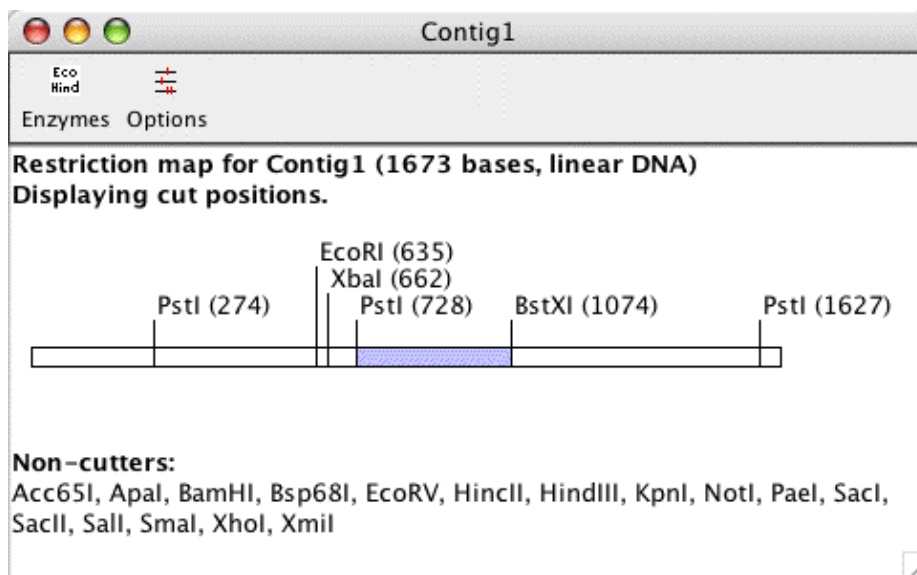


The text map displays the cut results for each enzyme separately like the multi line map. In the screen shot above, you see the fragment sizes generated from the contig by each enzyme.

The restriction maps can be **printed**, and the text map and the summary of each map can be **copied** using keyboard shortcuts. To copy a restriction map, select the part you want to copy and use the keyboard shortcut "Apple + C" on OSX and "Ctrl + C" on Windows to copy, and the keyboard shortcut "Apple + V" on OSX and "Ctrl + V" on Windows to paste.

Selecting Fragments

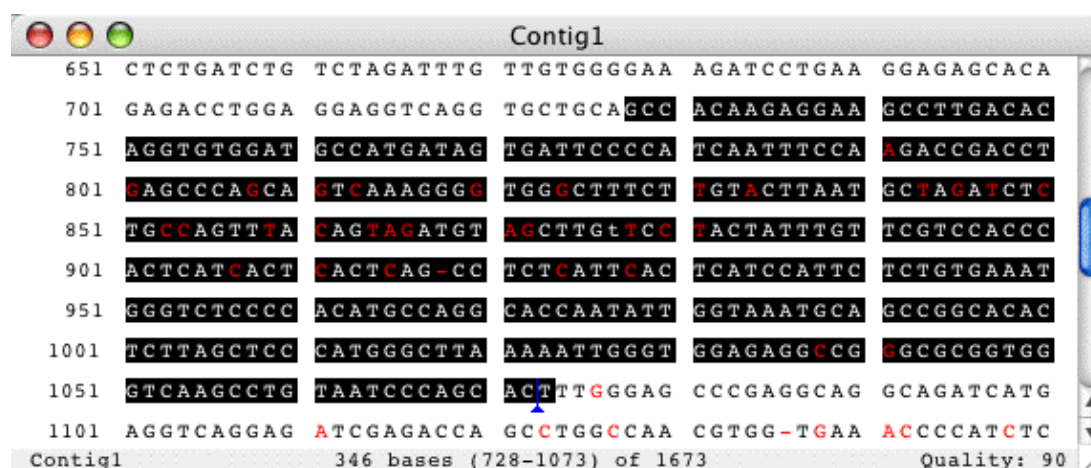
The graphical restriction map views (single and multi line map) allow you to **select** part of the sample shown in the map by clicking on a fragment between two cut positions. The selected part of the sample is highlighted in blue in the restriction view:



In the screen shot above, the fragment between the two enzymes PstI cutting at position 728 and BstXI cutting at 1074, is selected.

You can also make **continuous selections** by holding the Shift key pressed while selecting regions in the view with your mouse.

The **selection updates accordingly in the other views**, like the base view, the contig view and the trace view. The region selected in the screen shot above (from base 728 to base 1073), is now selected in the base view for this contig:



The selection also works the other way round: When you select some bases in the base view, trace view or contig view, they will be highlighted blue in the restriction view.

Restriction Enzymes in Aligner

The restriction enzymes used in Aligner are read in from a file.

This **file** is a text file called "**rebase_bairoch.txt**" and is located in the folder "Aligner Data" in the "CodonCode Aligner" folder.

The enzymes in this file are listed in a certain format compatible with known formats such as the ones from EMBL, PROSITE and SWISS-PROT.

The file was obtained from REBASE (RestrictionEnzyme dataBASE) [<http://rebase.neb.com>]. The rebase file used is the file in rebase format #19, also called "Bairoch format". The current file and a help file describing its structure can be downloaded from <http://rebase.neb.com/rebase/rebase.f19.html>

Aligner uses only the type II restriction enzymes that have known recognition sites from this file.

You can edit or update the file used by Aligner. However, if you decide to modify the file it is a good idea to first make a copy of it. This way you have a working enzyme file in case something unexpected happens when editing the file. You can open and edit the file with a text editor. When the file is open the version of the file is shown in the first line, and the date of this version a few lines below. To add your own enzymes to the file, please make sure to use the right format.

Closing Windows

You can close regular Aligner windows, for example the trace view and project view windows, either by clicking on the close button at the top of the window, or by choosing "**Close**" from the "**Window**" menu. Aligner dialog windows, for example the preference window, can be closed by clicking on one of the buttons at the bottom (often labeled "Cancel" and "OK"). If a dialog window is active, you need to close it before you can do anything else.

If you **close the project view**, Aligner will check if the project has been modified since the last save. If the project has unsaved changes, Aligner will ask you if you want to save the changes first. Closing the project window will also close all open windows that belong to the project.

If you are running Aligner on a Microsoft Windows operating system, you also have a main("root") window which contains all other windows. Closing the main window will exit Aligner (again, with the option to save unsaved changes in open projects).

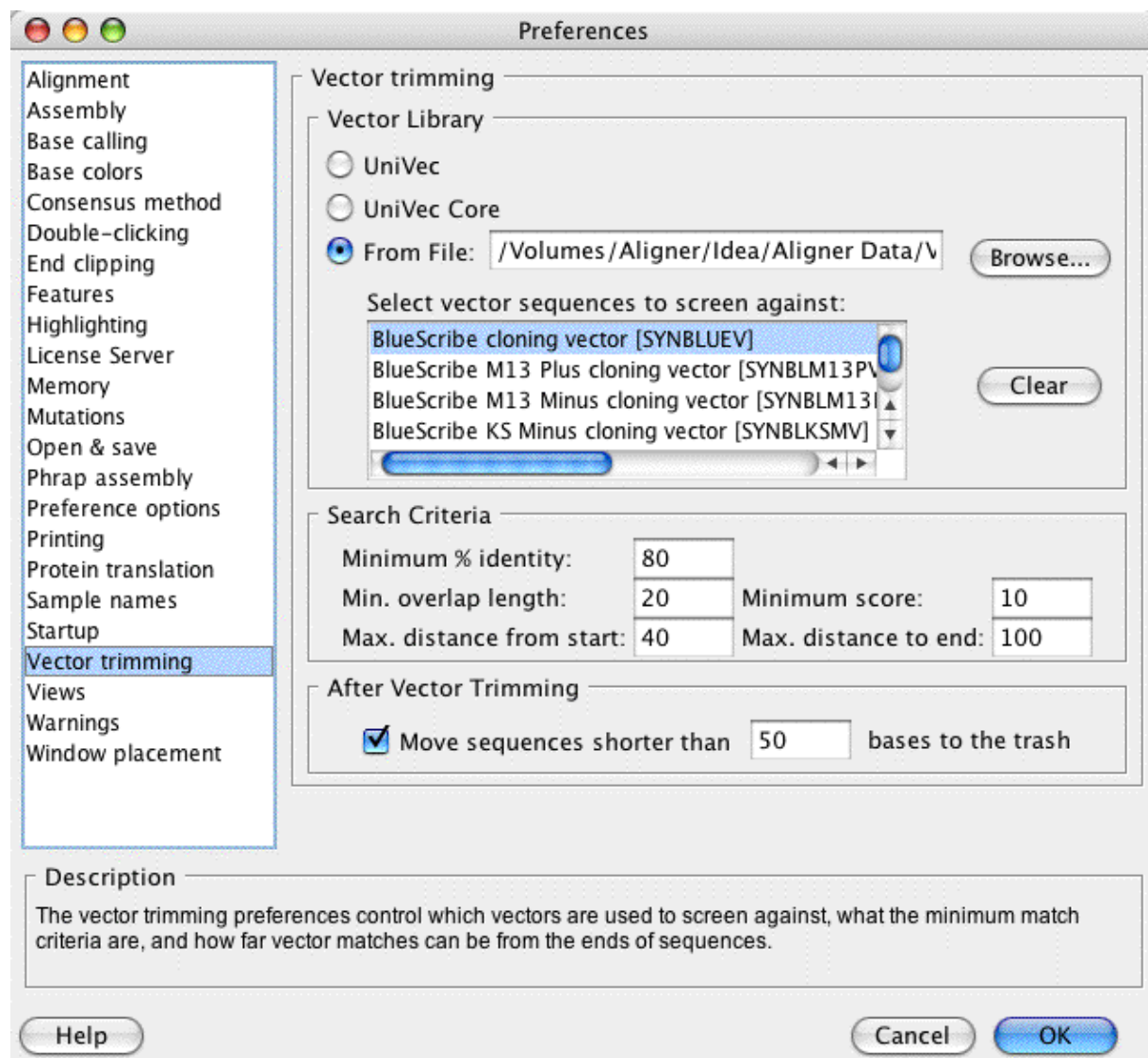
On Mac OS X, there is no main window. You have to choose "Quit" in the "Aligner" menu to exit Aligner.

Preferences and Settings

You can customize many aspects of Aligner in the "Preferences" dialog. To display the Preference dialog:

- On Mac OS X, select "**P**references" in the "**C**odon**C**ode **A**ligner" menu, or press Option-return
- On Windows, select "**P**references" in the "**E**dit" menu, or press Alt-Enter

This will show the preference window, which looks like this:



On the left side, you choose which specific preferences you want to edit. In the image above, "Vector trimming" is selected. The right side changes according to the selection on the left side. You can edit your settings on the left. Any changes you made are only saved when you press "OK" (or hit the return key). If you click "Cancel" (or press the escape key), no changes to the settings will be made.

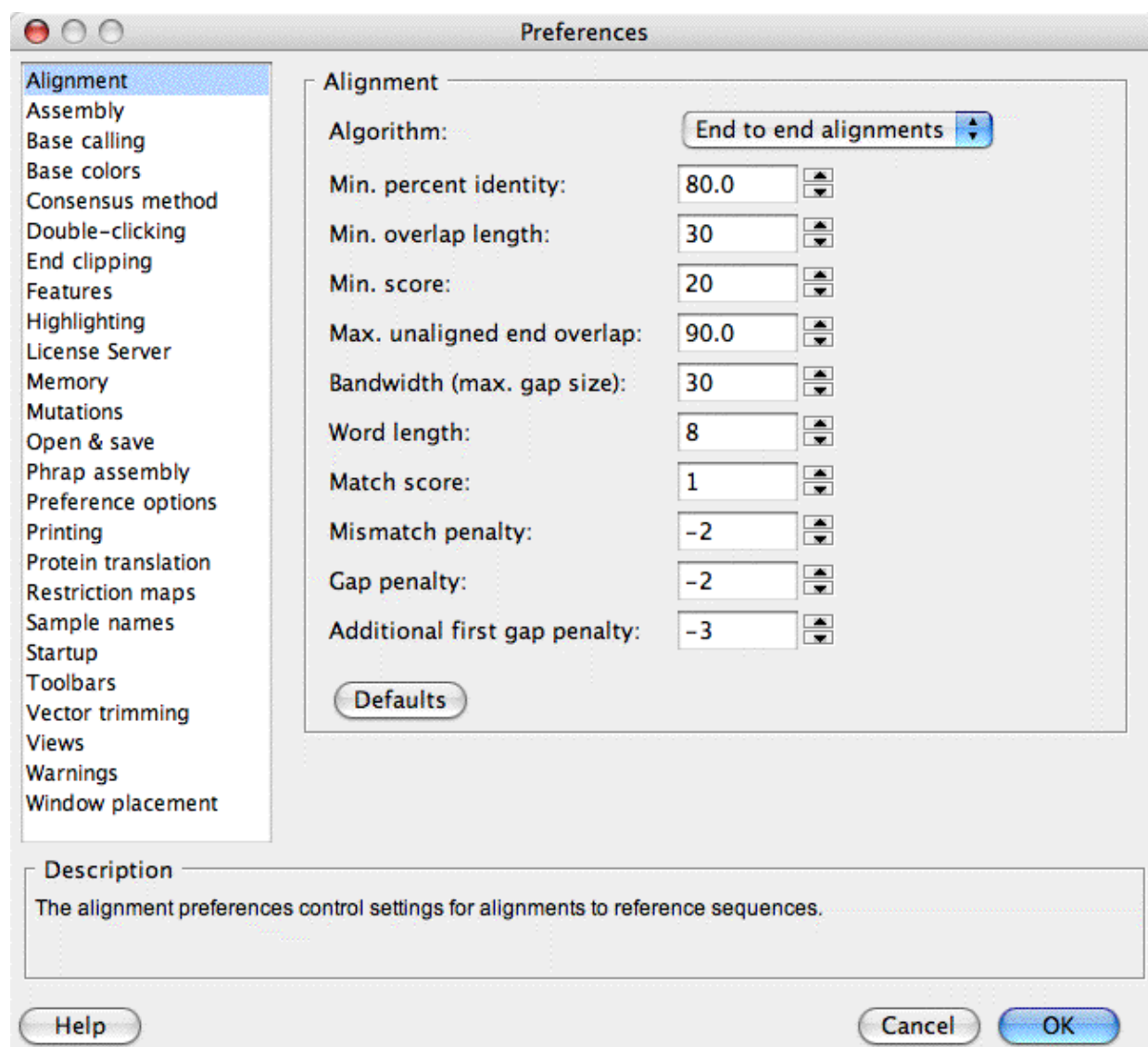
If you make any invalid entries in one of the fields, Aligner will show a warning that indicates the problem, and reset the field to the previous value.

CodonCode Aligner User Manual

By default, the Preferences are specific for each user. However, Aligner offers the option to share preferences between different users; detailed instructions are available on the ["Preference options" help page](#).

Alignment Preferences

You can specify parameters for alignments to reference sequences in the Alignment Preferences:



During the alignment process, the sequences are aligned successively against the reference sequence. Each alignment is evaluated against the match criteria defined in the alignment preferences. Only if the alignment meets all the criteria, the sample will be added. If the alignment does not match any one of the criteria, it will not be added, and remain in the "Unassembled Samples" folder. The parameters are similar to the parameters for sequence assembly, but you can assign different values for assembly and alignment parameters. The meaning of each parameter is discussed in the next sections.

Please note that the alignment parameters will **not** be used when comparing contigs (from the "Assemble with Options..." dialog).

Algorithm

The algorithm pulldown lets you choose how CodonCode Aligner compares sequences during alignment, with the following options:

- **Local alignments:** When this algorithm is used, Aligner uses local alignments. This means the start and the end of sequences is not necessarily included in the alignment - the alignments stop when the alignment score would not improve anymore. This can be due to (for example) too many discrepancies, or unremoved vector sequences. The resulting unaligned ("dangling") ends are shown on gray background in the contig view, base view, and trace view.
This was the default algorithm for CodonCode Aligner version 1.6.3 and older.
- **Large gap alignments:** This algorithm is typically used when aligning cDNA to genomic DNA. It allows for large gaps in between alignments, without penalizing the large gaps. The large gap algorithm can also be useful when analyzing samples with large insertions or deletions.
- **End to end alignment:** When this algorithm is used, alignments always include the entire sequences (*this method is the default algorithm in other programs, for example Sequencher*). When using this algorithm, it is important that samples have been end clipped and vector trimmed.
This is the default algorithm for CodonCode Aligner version 2.0.1 and newer. However, if you used older versions of CodonCode Aligner before, you may need to manually select this algorithm to use it.

Minimum Percent Identity

This is the minimum percentage of identical bases in the aligned region. The default parameter of 80% is relatively relaxed; you may want to use a more stringent setting for your projects, especially if you did use end clipping before the alignment.

Be careful about setting this value to the 100%: only samples that fully agree with the reference sequence will be aligned, samples with even a single discrepancy will not be aligned.

Minimum Overlap Length

This is the minimum length of the aligned region. If the aligned region is shorter than the value you set here (with 30 being the default), alignments will be rejected, and samples will remain in the "Unassembled Samples" folder.

Minimum Alignment Score

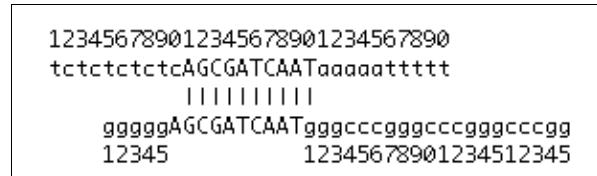
This parameter is similar to the "Minimum Overlap Length", but it takes discrepancies into account. Scores will be scaled so that a match gives a score of 1 - for each matching base in the aligned region, a score of +1 will be added. With the default settings, a score of -2 will be subtracted for each mismatch; for single base insertions or deletions, a score of -5 will be subtracted (-3 gap introduction penalty and -2 gap penalty; additional gaps in the same run lead to a subtraction of -2 per base).

In general, your minimum alignment score should be lower than your minimum overlap length to allow for some level of discrepancies between the sequences.

Maximum Unaligned End Overlap

This parameter is **only** effective when the "Local Alignments" algorithm is used. After doing an alignment, Aligner looks at the unaligned ("dangling") ends of both reads. Since Aligner does local alignments, two aligned sequence may have unaligned bases at the same end. This will happen, for example, when aligning two different copies of a repeat sequence, or if one of the samples is a chimeric clone. It can also happen if the sequence of one or both clones has a very high error rate, as typically seen towards the end of reads that have not been end clipped.

To illustrate how this parameter is calculated, consider the following diagram:



The sequence at the top has 10 unaligned bases at the start and end, and 10 aligned bases in the middle. The bottom sequence has 5 unaligned based at the start and 20 unaligned based at the end. Between the two sequences, there are 15 additional bases that could possibly have been aligned - the 5 unaligned bases at the start of the bottom sequence, and the 10 bases at the start of the top sequence.

Aligner calculated the relative amount of unaligned sequence that could have been aligned by dividing the overlapping bases in the unaligned ends by the length of the shorter sequence. In our example, this is 15 / 30 (the length of the top sequence), or 50%. With the default setting of 70% for the "Maximum Unaligned End Overlap", our example would have passed, at least for this parameter.

You may need to adjust this value, depending on the kind of project you are doing. If you aligned cDNA sequences to genomic DNA, use values of or near 100%, since large stretches of exons may be unaligned. But if you expect your samples to match end-to-end, and pre-processed your sequence with end clipping and vector trimming, you can use lower values to reduce the chance that different copies of repeats will be incorrectly assembled together.

Bandwidth (Maximum Gap Size)

The bandwidth parameter allows you to set the half width of the diagonal used during the banded alignment. This has an effect on the maximum size of gaps (insertions or deletions in one sample) that can still be aligned. A bit simplified, if one sample has an insertion or deletion that is larger than the "bandwidth" number, the alignment will typically stop at the insertion/deletion, and the rest of the sample will be unaligned. If the insertion or deletion is shorter than the bandwidth, the alignment will continue after introducing the necessary number or gaps in one sequence, as long as the aligned parts after the gaps is long enough (the aligned regions before and after the gaps must be at least 1 and 1/2 times as long as the number of gaps, and longer for any mismatches or ambiguities).

The discussion above is a bit simplified - in reality, what counts is the total number of gaps in one sequence, minus the total number of gaps in the other sequence, at any position, since the alignment uses a banded Needleman-Wunsch algorithm.

The bandwidth parameter has an impact on the alignment speed - larger values mean slower alignments. For large projects, you may want to reduce the bandwidth value; for projects where you know that you have larger

insertions and deletions, you may want to increase it. Note, however, that increasing the bandwidth will typically not be enough to extend alignments through very large gaps, like large introns. Support for "Large gap alignments" will be added to later versions of Aligner.

The bandwidth parameter does not apply to large gap alignments; in large gap alignments, gaps between aligned parts can extend for thousands of bases.

Word Length

The "word length" parameter determines the size of "words" that CodonCode Aligner uses when looking for potential overlaps between sequences. Only sequence pairs that have perfect matches of at least this length will be considered for merging.

If you are trying to align sequences with high error or mutation rates, reducing the word length may help to get samples aligned. For very large projects or projects with many repeat sequences, larger numbers may give better results.

The impact of the word length setting on the alignment speed depends on the size of the project; for large projects, larger word length values can lead to faster alignments.

Match scoring

The last four alignment parameters determine how matches are scored. The "Match score" is used when two aligned nucleotides are identical; the "Mismatch penalty" when two base calls are different. The "Gap penalty" and the "Additional first gap penalty" is used when one of the two sequences has a deletion relative to the other sequence. For single base deletions, the penalty score will be the sum of the gap penalty and the additional first gap penalty; for additional deleted bases (multiple gaps in a row), the penalty will be just the gap penalty.

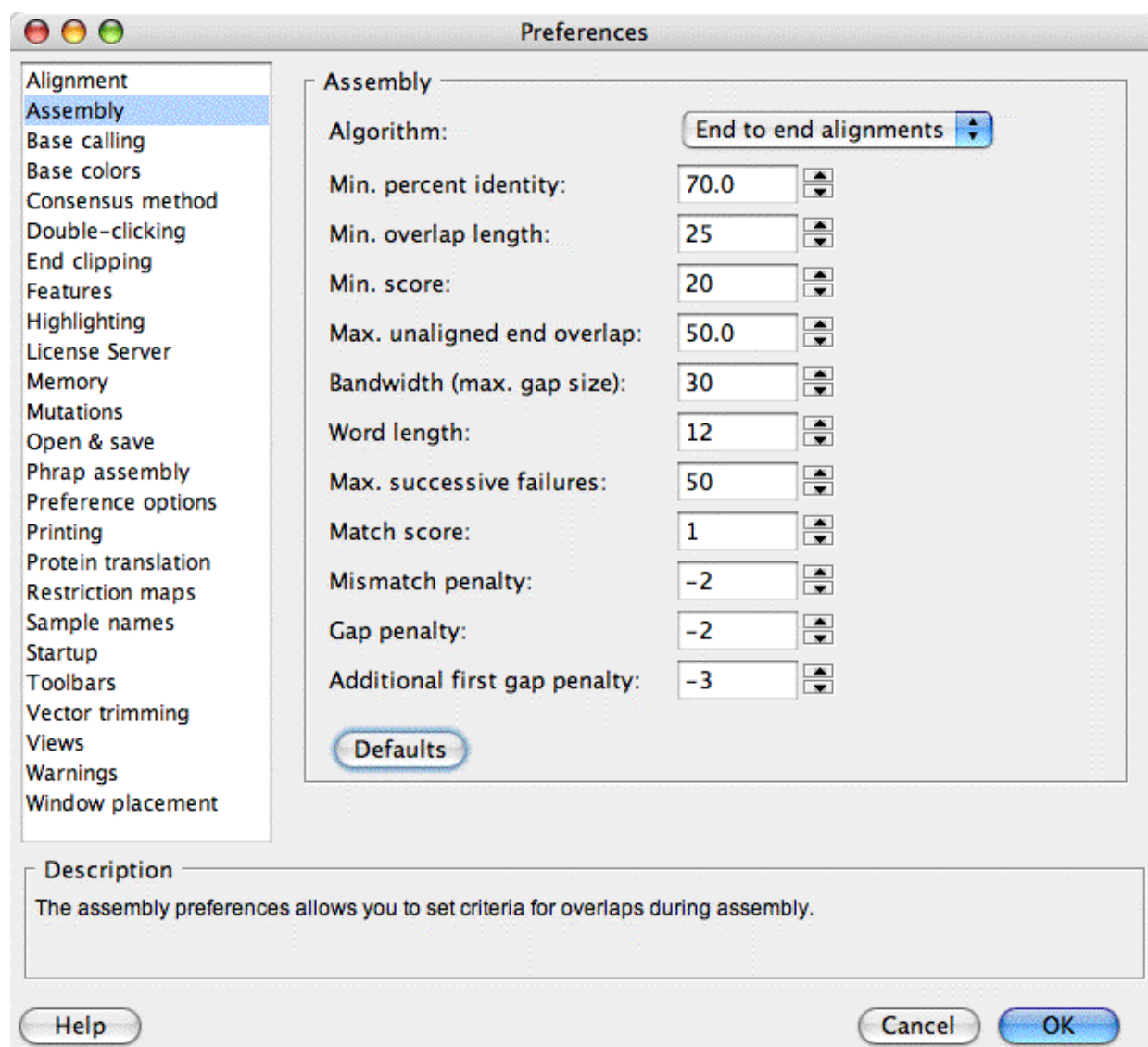
You can change the scoring within limits (scores from 1 to 19 for matches, and penalties of -1 to -19). In general, we suggest that only experts change the match scores and penalties.

Restoring default parameters

To restore the default parameters, click on the "Defaults" button near the bottom. This will reset all parameters to the choices shown in the screen shot above.

Assembly Preferences

The assembly preferences allow you to specify parameters for sequence assembly:



During the assembly process, the sequences or contigs are aligned successively against each other. Each alignment is evaluated against the match criteria defined in the alignment preferences. Only if the alignment meets all the criteria, the samples or contigs will be merged. If the alignment does not match any one of the criteria, the merger will be rejected. The parameters are similar to the parameters for sequence alignment, but you can assign different values for assembly and alignment parameters. The meaning of each parameter is discussed in the next sections.

Please note that the assembly parameters **will not be used when comparing contigs with ClustalW or muscle**, or when assembling with Phrap. When comparing contigs, the assembly preferences will only be used when the built-in algorithm is selected in the "Algorithm" panel of the "Assemble with Options" dialog.

Algorithm

The algorithm pulldown lets you choose how CodonCode Aligner compares sequences during assembly, with the following options:

- **Local alignments:** When this algorithm is used, Aligner uses local alignments (*this method is also used when assembling using Phrap*). This means the start and the end of sequences is not necessarily included in the alignment - the alignments stop when the alignment score would not improve anymore. This can be due to (for example) too many discrepancies, or unremoved vector sequences. The resulting unaligned ("dangling") ends are shown on gray background in the contig view, base view, and trace view.
This was the default algorithm for CodonCode Aligner version 1.6.3 and older.
- **Large gap alignments:** This algorithm is typically used when aligning cDNA to genomic DNA. It allows for large gaps in between alignments, without penalizing the large gaps. The large gap algorithm can also be useful when analyzing samples with large insertions or deletions.
- **End to end alignment:** When this algorithm is used, alignments always include the entire sequences. When using this algorithm, it is important that samples have been end clipped (and possibly also vector trimmed).
This is the default algorithm for CodonCode Aligner version 2.0.1 and newer. However, if you used older versions of CodonCode Aligner before, you may need to manually select this algorithm to use it.

Minimum Percent Identity

This is the minimum percentage of identical bases in the aligned region. The default parameter of 70% is relatively relaxed; you may want to use a more stringent setting for your projects, especially if you did use end clipping before the alignment.

Be careful about setting this value to the 100%: only samples that fully match each other in the overlapping regions will be assembled, samples with even a single discrepancy will not be aligned.

Minimum Overlap Length

This is the minimum length of the aligned region. If the aligned region is shorter than the value you set here (with 25 being the default), alignments will be rejected, and samples will remain in the "Unassembled Samples" folder.

Minimum Alignment Score

This parameter is similar to the "Minimum Overlap Length", but it takes discrepancies into account. Scores will be scaled so that a match gives a score of 1 - for each matching base in the aligned region, a score of +1 will be added. With the default settings, a score of -2 will be subtracted for each mismatch; for single base insertions or deletions, a score of -5 will be subtracted (-3 gap introduction penalty and -2 gap penalty; additional gaps in the same run lead to a subtraction of -2 per base).

In general, your minimum alignment score should be lower than your minimum overlap length to allow for some level of discrepancies between the sequences.

Maximum Unaligned End Overlap

This parameter is perhaps the hardest to understand. After doing an alignment, Aligner looks at the unaligned ("dangling") ends of both reads. Since Aligner does local alignments, two aligned sequence may have unaligned bases at the same end. This will happen, for example, when aligning two different copies of a repeat sequence, or if one of the samples is a chimeric clone. It can also happen if the sequence of one or both clones has a very high error rate, as typically seen towards the end of reads that have not been end clipped.

To illustrate how this parameter is calculated, consider the following diagram:

123456789012345678901234567890
tctctctctcAGCGATCAATaaaaatttt
gggggAGCGATCAATggggccggggccggggccgg
12345 12345678901234512345

The sequence at the top has 10 unaligned bases at the start and end, and 10 aligned bases in the middle. The bottom sequence has 5 unaligned based at the start and 20 unaligned based at the end. Between the two sequences, there are 15 additional bases that could possibly have been aligned - the 5 unaligned bases at the start of the bottom sequence, and the 10 bases at the start of the top sequence.

Aligner calculated the relative amount of unaligned sequence that could have been aligned by dividing the overlapping bases in the unaligned ends by the length of the shorter sequence. In our example, this is $15 / 30$ (the length of the top sequence), or 50%. With the default setting of 70% for the "Maximum Unaligned End Overlap", our example would have passed, at least for this parameter.

You may need to adjust this value, depending on the kind of project you are doing. If you aligned cDNA sequences to genomic DNA, use values of or near 100%, since large stretches of exons may be unaligned. But if you expect your samples to match end-to-end, and pre-processed your sequence with end clipping and vector trimming, you can use lower values to reduce the chance that different copies of repeats will be incorrectly assembled together.

Bandwidth (Maximum Gap Size)

The bandwidth parameter allows you to set the half width of the diagonal used during the banded alignment. This has an effect on the maximum size of gaps (insertions or deletions in one sample) that can still be aligned. A bit simplified, if one sample has an insertion or deletion that is larger than the "bandwidth" number, the alignment will typically stop at the insertion/deletion, and the rest of the sample will be unaligned. If the insertion or deletion is shorter than the bandwidth, the alignment will continue after introducing the necessary number or gaps in one sequence, as long as the aligned parts after the gaps is long enough (the aligned regions before and after the gaps must be at least 1 and 1/2 times as long as the number of gaps, and longer for any mismatches or ambiguities).

The discussion above is a bit simplified - in reality, what counts is the total number of gaps in one sequence, minus the total number of gaps in the other sequence, at any position, since the alignment uses a banded Needleman-Wunsch algorithm.

The bandwidth parameter has an impact on the assembly speed - larger values mean slower assemblies. For large projects, you may want to reduce the bandwidth value; for projects where you know that you have larger insertions and deletions, you may want to increase it. Note, however, that increasing the bandwidth will

typically not be enough to extend alignments through very large gaps, like large introns. Support for "Large gap alignments" will be added to later versions of Aligner.

The bandwidth parameter is ignored when the "large gap alignments" algorithm is used for assembly.

Word Length

The "word length" parameter determines the size of "words" that CodonCode Aligner uses when looking for potential overlaps between sequences. Only sequence pairs that have perfect matches of at least this length will be considered for merging.

If you are trying to assemble sequences with high error or mutation rates, reducing the word length may help to get samples aligned. For very large projects or projects with many repeat sequences, larger numbers may give faster assemblies and better results.

The impact of the word length setting on the assembly speed depends on the size of the project; for large projects, larger word length values can lead to faster alignments.

Maximum Successive Failures

This parameter is only relevant for larger assemblies with tens or hundreds of reads. It can be used to limit how long Aligner will try to merge contigs for very large projects, with larger numbers meaning more tries and longer assembly times, but potentially fewer contigs.

In detail: Aligner will initially look for overlaps between all samples, and then use this map of overlaps to merge samples into larger and larger contigs. If two samples are already in contigs, Aligner will try to merge the contigs. If the merger is rejected, and Aligner later finds two other overlapping samples in the same contigs, Aligner will try to merge the contigs again. If the contigs have changed in the mean time, this can make sense, since the merger may now work. For very large projects with many repeats, however, this may mean that Aligner tries the same mergers many times. The "Maximum Successive Failures" parameter can be used to stop such fruitless efforts - after Aligner has tried this many times (default: 50) in a row to merge contigs without success, the assembly will stop.

Yes, we do realize that the assembly algorithm could be made smarter here, and we may modify it in the future - but we think that most users will not be doing such large assemblies with Aligner, anyway. There are other assemblers out there that can handle large assemblies, for example Phrap - and you can import Phrap assemblies into Aligner for editing.

Match scoring

The last four alignment parameters determine how matches are scored. The "Match score" is used when two aligned nucleotides are identical; the "Mismatch penalty" when two base calls are different. The "Gap penalty" and the "Additional first gap penalty" is used when one of the two sequences has a deletion relative to the other sequence. For single base deletions, the penalty score will be the sum of the gap penalty and the additional first gap penalty; for additional deleted bases (multiple gaps in a row), the penalty will be just the gap penalty.

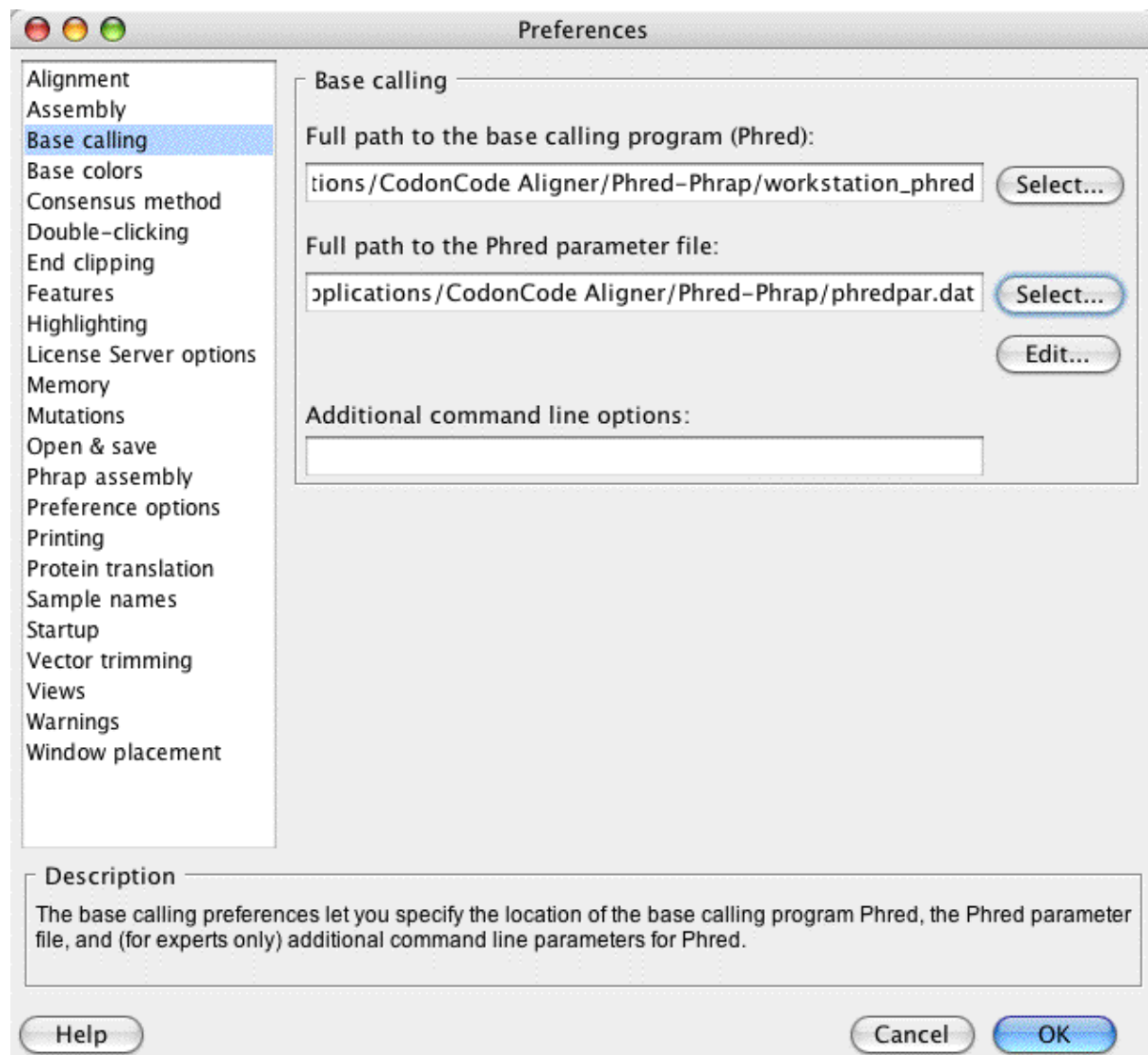
You can change the scoring within limits (scores from 1 to 19 for matches, and penalties of -1 to -19). In general, we suggest that only experts change the match scores and penalties.

Restoring default parameters

To restore the default parameters, click on the "Defaults" button near the bottom. This will reset all parameters to the choices shown in the screen shot above.

Base Calling Preferences

In the base calling preferences, you can specify details needed for running the base calling program Phred (to use base calling in Aligner, **you need to have Phred installed on the system that you are running Aligner on**, of course):



You specify the location and name of the base calling program in the upper text field. Currently, Phred is the only base calling program that is supported by Aligner.

In the second text field, you specify the location of the Phred parameter file, which is typically called "phredpar.dat". For more information about the Phred parameter file, please check the [base calling help page](#). It also gives you some more details about how to [edit](#) the Phred parameter file (which you can do by pressing the "Edit..." button).

If you installed Phred with Aligner, the path to Phred and to the Phred parameter file should be correct, and not need any changes. However, if you use your own installation of Phred, you may need to specify the location of both files on your system. The default location and names are:

CodonCode Aligner User Manual

- On OSX:
/Application/CodonCode Aligner/Phred-Phrap/workstation_phred
/Application/CodonCode Aligner/Phred-Phrap/phredpar.dat
- On Windows:
C:\Program Files\CodonCode Aligner\Phred-Phrap\workstation_phred.exe
C:\Program Files\CodonCode Aligner\Phred-Phrap\phredpar.dat

Note that since Aligner version 1.2.5 beta 5, Aligner will use the workstation versions of Phred (and Phrap) by default. The workstation version of Phred can only be run from CodonCode Aligner, and requires either:

- a valid trial license, or
- a valid full (purchased) license that includes a license for the "Workstation Phred" module. Licenses purchased by academic customers automatically include a license for the "Workstation Phred" module free of charge; users at for-profit organizations need to pay a separate license fee for the workstation version of Phred.

If you started using CodonCode Aligner with a version before 1.2.5 beta 5, and would like to use the workstation version of Phred, please:

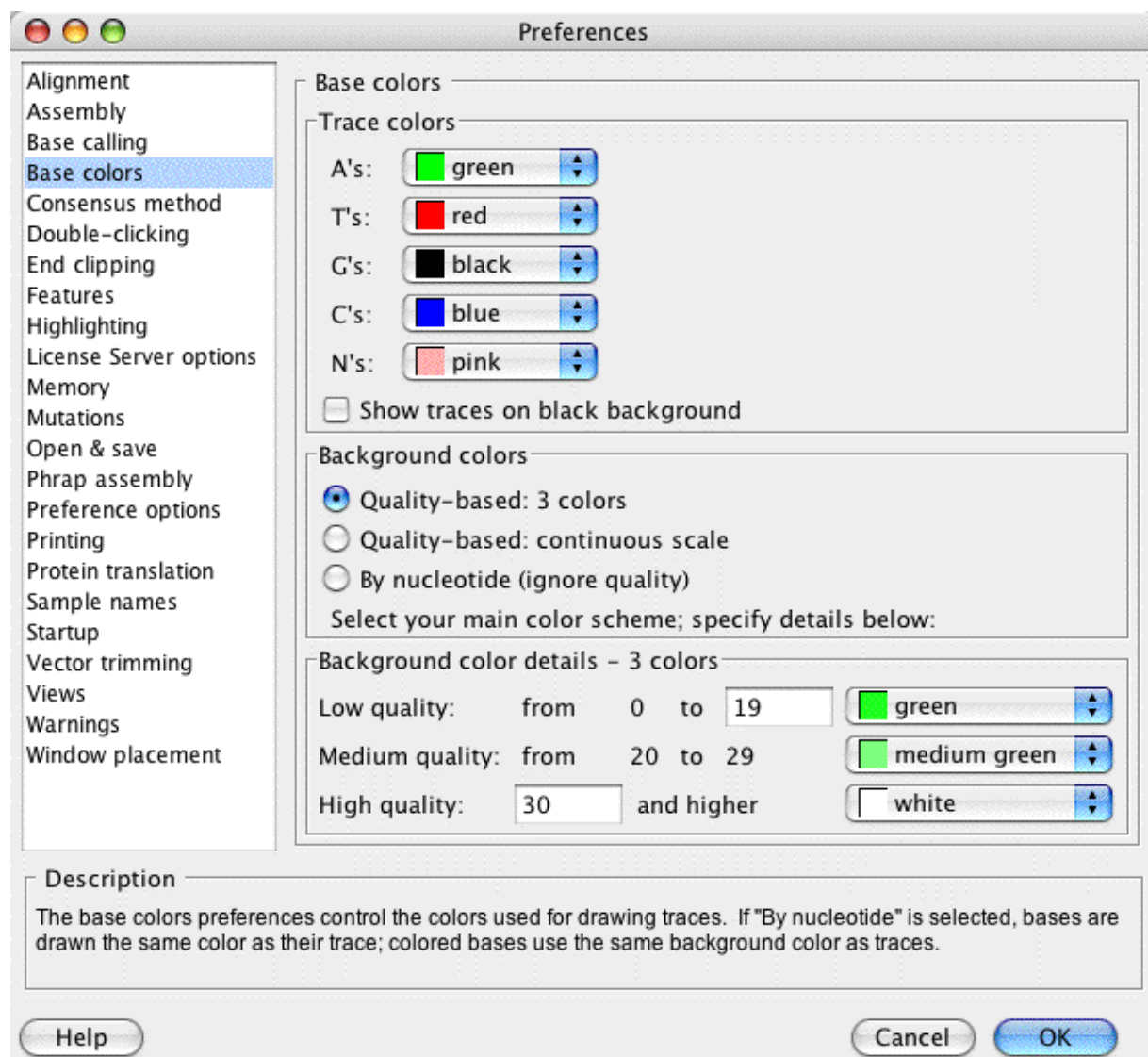
1. Check if your license includes a license for "Workstation Phred" (go to the "Help" menu, select "License...", and check the module permissions in the license dialog).
2. Click on the upper "Select..." (or "Browse...") button in the base calling preferences, and navigate to the workstation_phred executable.
3. Select "workstation_phred", and then click "OK" in the file dialog and in the preference dialog.

One note of caution: CodonCode Aligner uses the location and name of the Phred executable to determine whether you are using the workstation version or the regular version (both versions perform identically, but the workstation version can only be run from Aligner with a valid license). Aligner will assume that you are using the workstation version if it finds "workstation_phred" anywhere in the name, which includes the directories. Therefore, **please do not change the name of the Phred executable** installed by Aligner. If you are using your own copy of Phred rather than the workstation version, please make sure to **not** call it "workstation_phred", or put into a folder hierarchy that contains "workstation_phred"!

In the bottom text field, you can specify additional command line options for Phred. In general, please leave this line blank - **only experts who really know what they are doing should specify command line options** here! Also, note that Aligner uses the -id and the -cd options to specify input- and output-directories for Phred; for more details, please check the [base calling help page](#).

Base Color Preferences

CodonCode Aligner can use several different color schemes to display bases. You can control and fine-tune these schemes in the Base Color Preference dialog:



At the top, you can set the colors used to draw the traces (and, if you are using the "By nucleotide" color scheme, for the bases).

Usually, traces will be shown on a white background. If you prefer to see traces on a black background, make sure the checkbox "Show traces on black background" is selected.

In the middle, the "Background color scheme" panel allows you to choose one of three basic color schemes:

1. three different background colors for low, medium, and high quality bases (similar to Sequencher)
2. continuous background colors that depend on the base quality, with darker backgrounds for low qualities and lighter backgrounds for higher quality (similar to Consed)
3. colors that vary by base, and ignore base quality

Switching color schemes

A quick way to switch between color schemes is to go to the **"View"** menu, and select **"Switch Color Schemes"** . This will switch between the different color schemes in the following order:

1. Quality-based 3 color scheme
2. Quality-based continuous scheme
3. Base-specific background colors
4. Base-specific foreground colors

Then, the circle starts over again - after base-specific foreground colors comes the quality-base 3-color scheme again.

Instead of using the menus, you can also use the corresponding toolbar button; if it is not showing, change the toolbar settings in the [Toolbar Preferences](#).

The View menu and the corresponding toolbar icon provide a fast way to switch between the color schemes. For full control, for example to change colors or threshold, use the Base Color preferences.

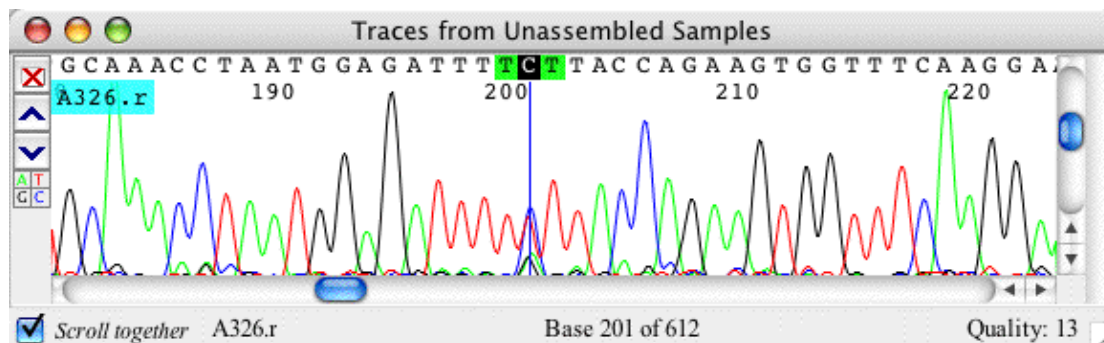
Quality-based 3 color scheme

The picture above shows the settings for the 3 color scheme. In this scheme, the background behind the bases is shown in different colors for "low quality" bases, "medium quality" bases, and "High quality" bases. You can set the ranges for low- and high-quality bases (with medium quality being the range in between) by changing the numbers in the text boxes. The most commonly used numbers are 0 to 19 for low quality, and 20-29 for medium quality, and 30 and higher for high quality. You can also assign background colors for each of these ranges. We suggest to use darker colors for lower qualities, and lighter colors for high quality bases. These colors will be used to draw the bases in the base view , contig view, and the trace view windows. Here is an example of the base view with this setting:



CodonCode Aligner User Manual

Notice that there is a short stretch of low-quality bases at the start of the sequence, and a longer stretch at the end, which is a typical picture. There is also a short stretch of low quality bases near base 200; looking at the trace view, you can see that this is due to multiple peaks at this location:

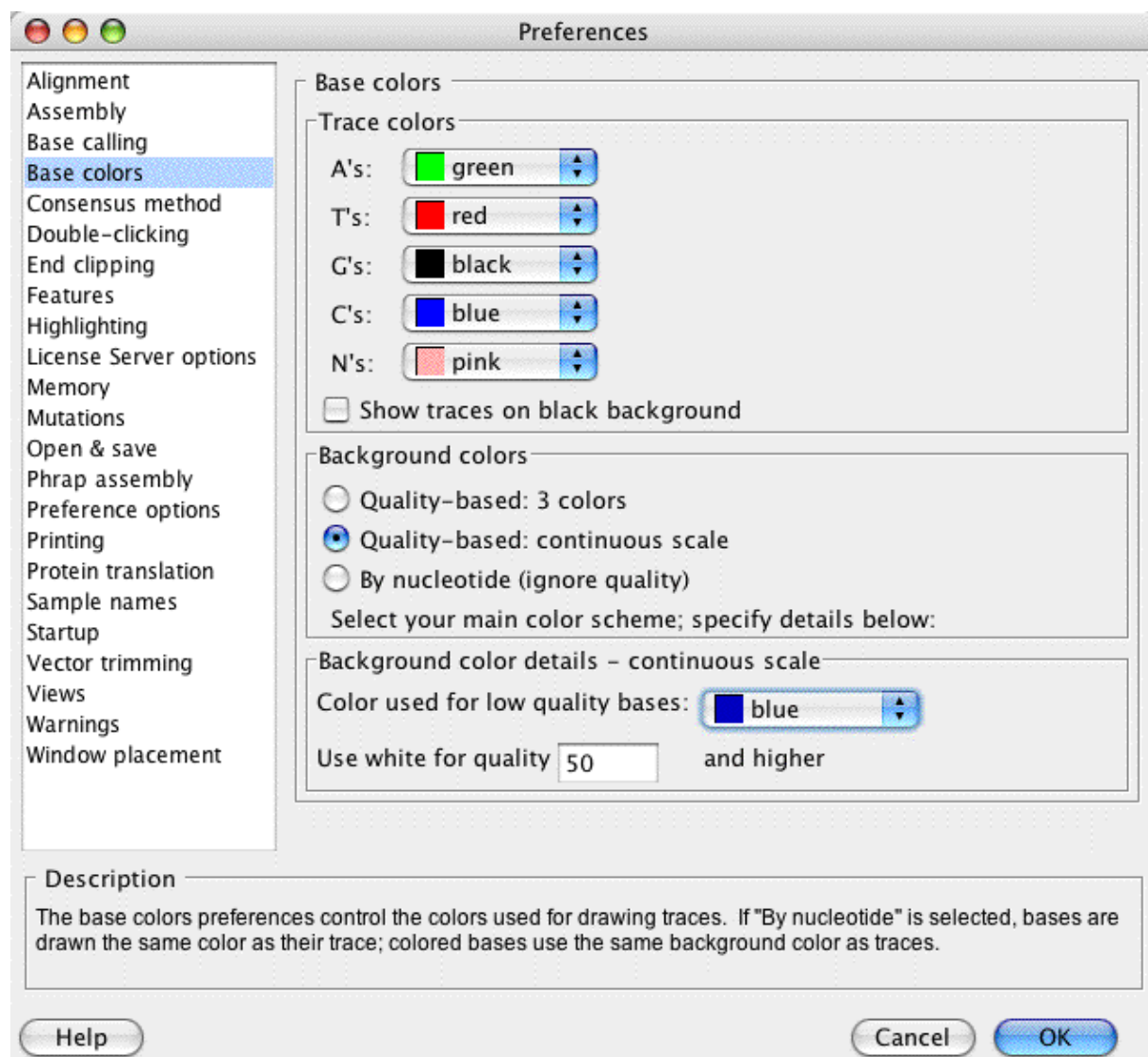


(Note that Phred typically lowers the quality scores of three bases, not just one base, in problem areas, since neighboring bases are also more likely to be wrong).

The 3-color scheme is often useful to get a general idea of the sequence quality. If you want more detailed color shading, select the "Quality-based: continuous scale" color scheme.

Continuous, quality-based background colors

In the continuous color scheme, you can select two settings: the background color for low-quality bases, and the minimum quality for which to use a white background.



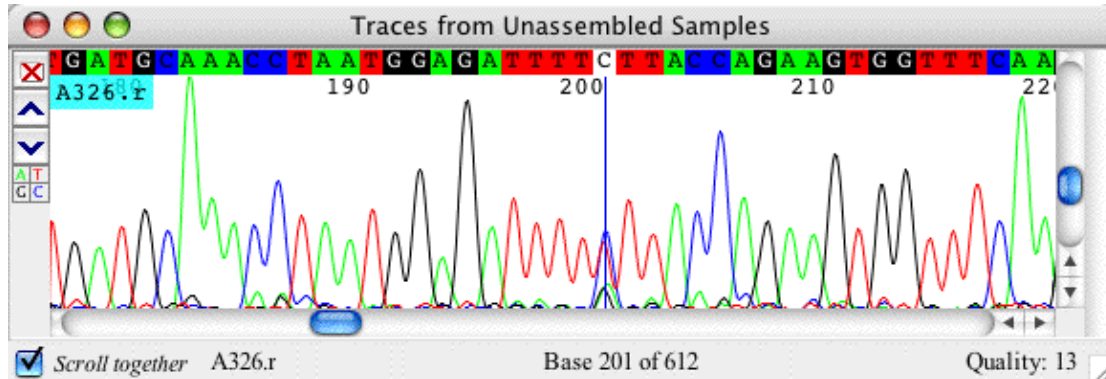
With the settings shown above, bases with a quality of 0 will be shown on blue background, and bases with qualities of 60 and higher on white background. Bases in between will be shown on varying shades of blue, with darker blues for lower qualities:



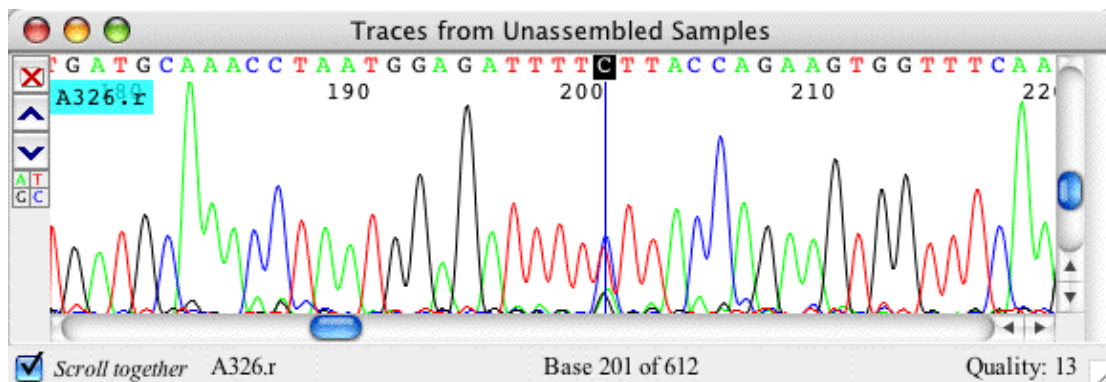
Note that both quality-based color schemes require samples that have Phred qualities (or Phred-like qualities). Aligner will work best with sequences that have such qualities. If you do not have qualities, you can use the base-specific color scheme.

Base-specific colors

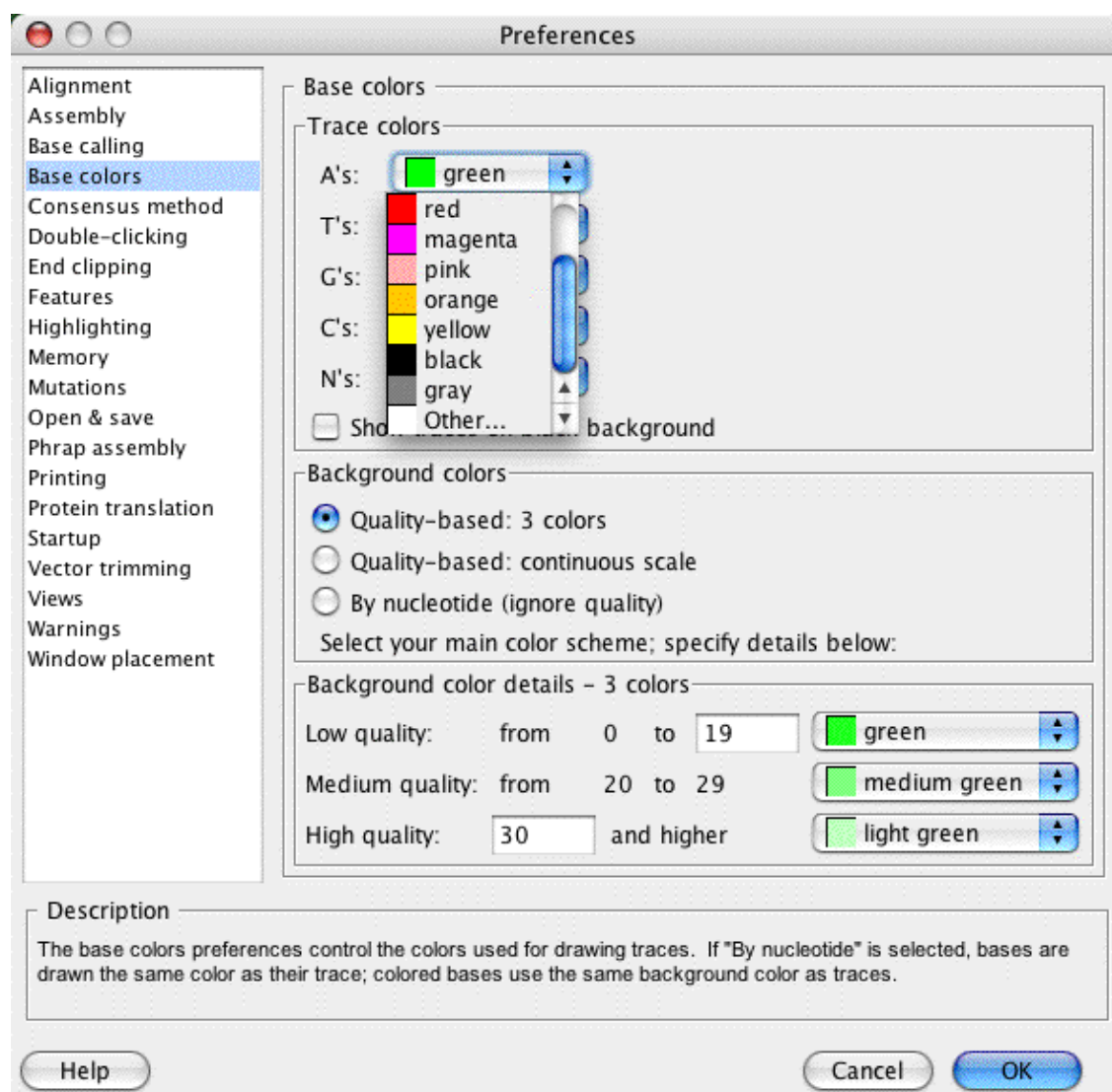
The third color scheme ignores sequence qualities, and chooses the colors from the bases. You can either have bases drawn on colored background, or colored bases on white background. Here's an example of a trace on a colored background:



The same trace with colored bases:

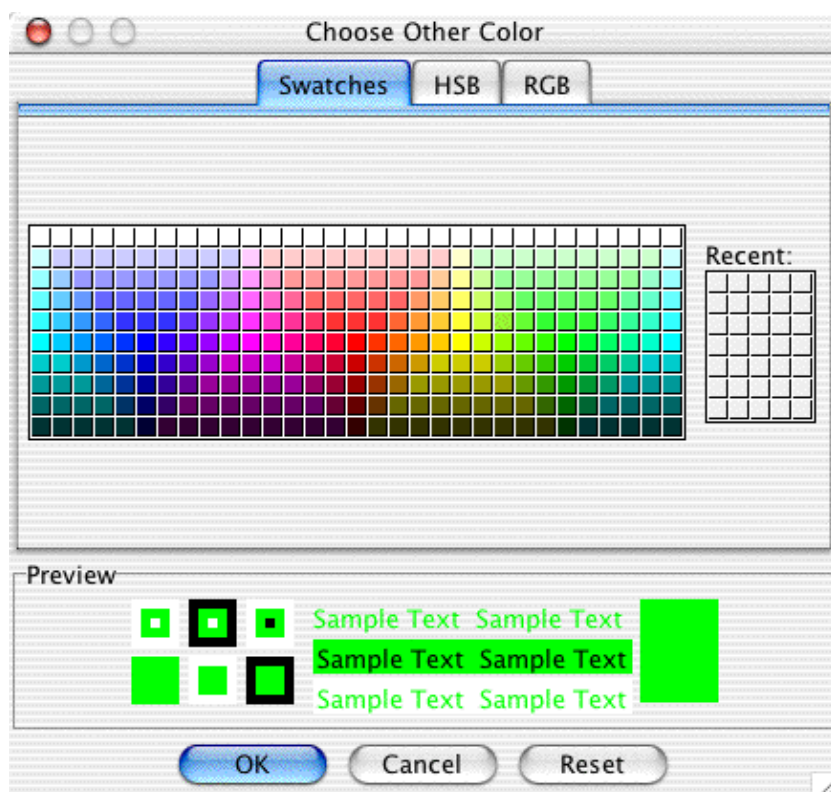


The base-specific color scheme makes most sense when samples do not have qualities, or sometimes when looking for differences in aligned or assembles samples. You can assign the color for each base in top section of the "Base colors" preference panel, as shown below:



To assign a color to a specific base, use the drop down box to the right of the base. As always, any changes will only be saved when you click "OK". *Note that the colors chosen will also be used to draw the sequence traces, regardless of the background color scheme.*

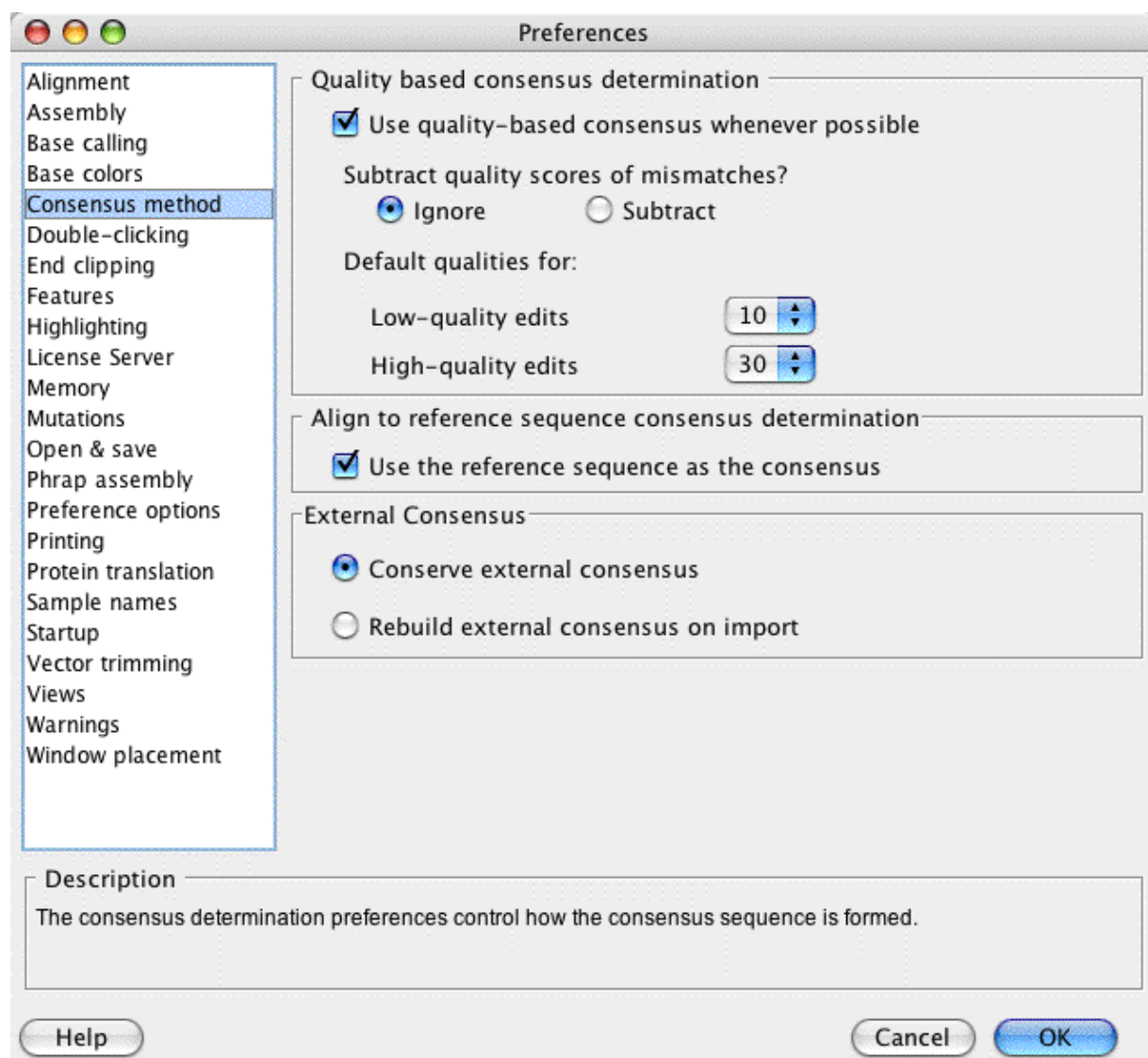
You can pick one of the pre-defined colors, or click on "Other..." to choose a different color. This will bring up the follow "color chooser" dialog:



Select the new color, then press "OK" to use this color, or "Cancel".

Consensus Preferences

The consensus preferences allow you to specify how the contig ("consensus") sequences are calculated:



In the check box at the top, you choose whether Aligner will use a quality-based consensus or a majority-based consensus.

For most projects, **we strongly suggest to use a quality-based consensus whenever possible**. A quality-based consensus mimics what a human "contig editor" would do to determine the correct consensus sequence: Aligner looks for the highest quality sample at each position, taking into consideration confirmation by samples in reverse direction, as well as disagreements.

You can choose whether or not you want Aligner to subtract the quality scores of discrepant bases when calculating the consensus quality score. In general, this is not necessary, since discrepancies are typically errors that do not influence the correctness of the consensus base; however, you can tell Aligner that you think differently, and subtract the quality scores of discrepant bases.

You can also set the qualities values used for edited bases when calculating consensus quality. Internally, Aligner uses qualities of 98 to "low quality" edits, and 99 to "high quality" edits. This follows conventions used by the contig editing programs Gap4 and Consed, and it allows Aligner to take edited bases into account when it matters, for example when counting Phred20 bases. When calculating the consensus quality, Aligner will substitute the quality scores with the scores you specify here; the default values are 10 for low quality edits, and 30 for high quality edits.

One exception where you may not want to use a quality-based consensus are **re-sequencing projects**, where you align your sequences to a known reference sequence. In such projects, you may want to use the **majority consensus** (uncheck the box at the top), so you can see most common base at each position in the consensus sequence (for details about how Aligner builds the majority consensus, check the "[Algorithms](#)" section).

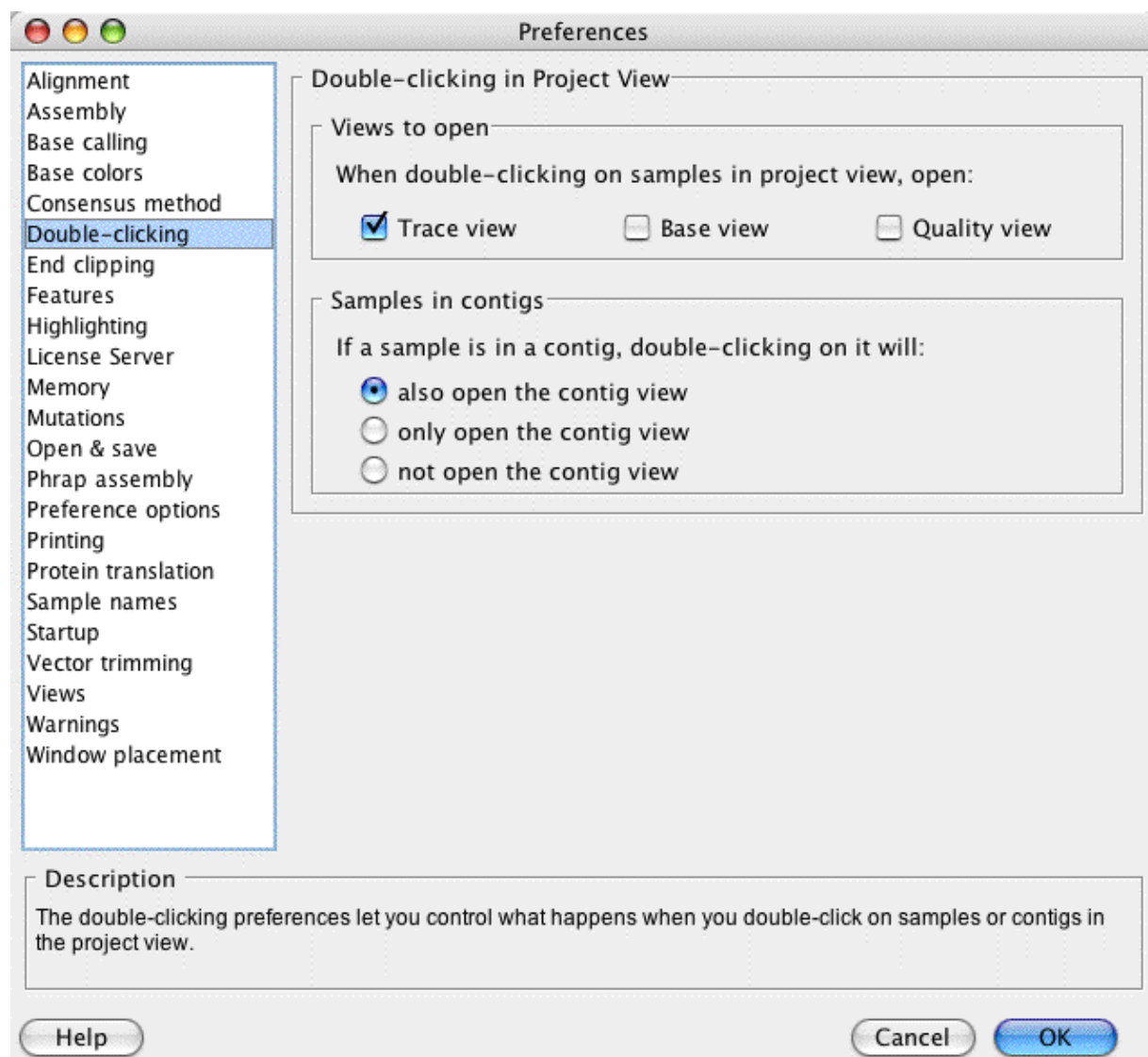
Instead of building a majority consensus, Aligner can also **use the reference sequence as the consensus** sequence for contigs that were generated by alignment to a reference sequence. To do this, check the box labeled "Use the reference sequence as the consensus". Whenever the reference sequence is used to build the consensus, Aligner will also use tags for mutation finding that are added to the reference sequence, in particular "codingSequence" and "dontGenotype" tags.

Tip: If you want to compare a lot of samples against a reference sequence in separate projects, it's a good idea to first create a project with the reference sequence only, and to add the "codingSequence" and (if needed) "codonStart" and "dontGenotype" tags to the reference sequence. Then save the project, and save a copy under a different name. For more information, check the "[Find mutations](#)" section.

The bottom panel only concerns the consensus of imported assemblies. You can choose whether to keep the consensus sequence when you import entire assemblies, or to re-calculate the consensus based on the current settings. Even when you decide to keep the imported consensus sequence, however, Aligner will re-calculate the consensus sequence when you later change a contig, for example by editing bases or removing samples.

Double Clicking Preferences

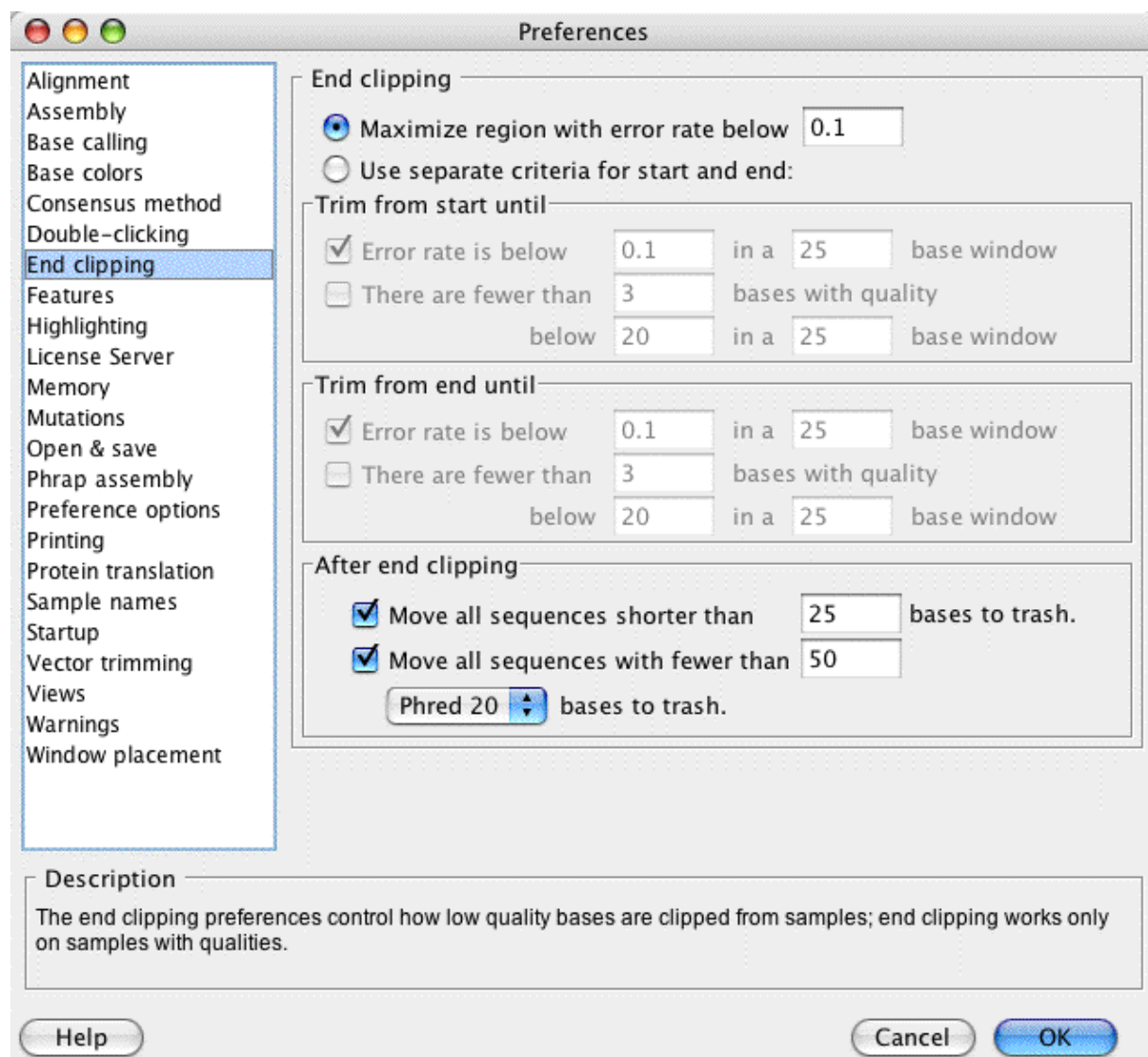
The double-clicking preferences allow you to choose what happens if you double-click on one or more samples in the project view windows:



You can choose which views (windows) Aligner should open in response to double-clicks in Project View. If the samples you selected are in contigs, you can also choose if Aligner should also open the corresponding contig view, or just open the corresponding contig view.

End Clipping Preferences

The end clipping preferences allow you to specify how Aligner will remove low-quality sequence from the ends of samples, and to set up minimum quality criteria after end clipping:



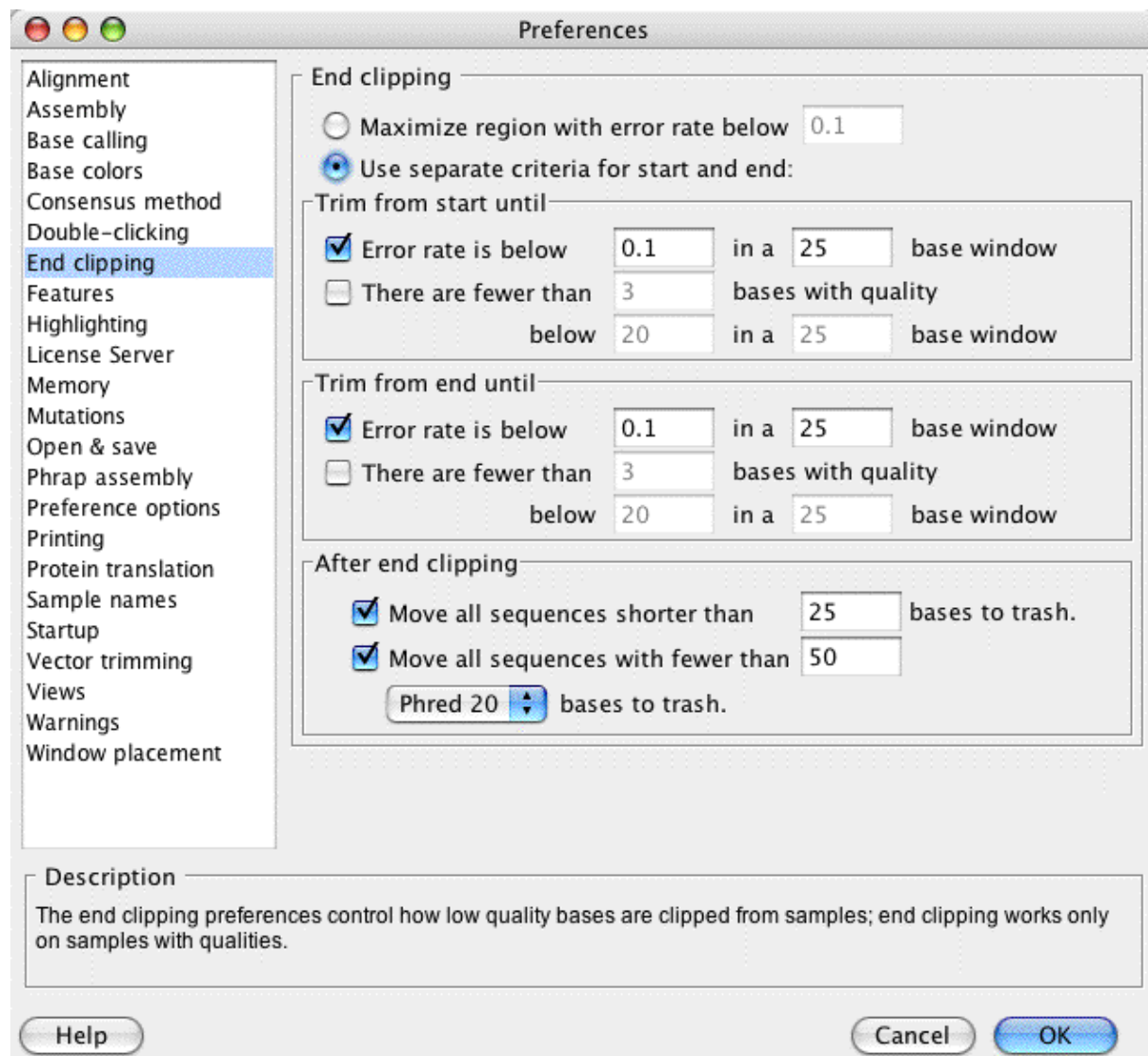
On the top, you have the choice between two different end clipping methods:

1. a method that maximizes the region with an (estimated) error rate below the threshold you define, and
2. a method that uses different criteria at the beginning at the end of the reads.

The first methods is similar to the way Phred trims sequences with the "-trim_alt" option. The second method gives you more options, and is (hopefully) a bit easier to understand. With typical parameters, both methods tend to give similar results, and remove the "junk" sequence at the end of chromatograms. The methods are explained in more detail on the ["End clipping algorithms" help page](#).

If you choose the first method, you only have to specify one parameter, which is the maximum error rate for the clipped region. If you select the second trim method, you can define the end clipping stringency in more

detail, as illustrated below:



You can choose to trim from the start until the estimated error rate drops below the cutoff you choose, and also specify the length of the "window" over which to calculate the expected error rate. Alternatively, you can clip until there are only very few low-quality bases in a window, specifying the window length, the maximum number of low quality bases, and what "low quality" means to you (typically, Phred scores lower than 20). You can also combine these two measures, or choose neither one if you do not want to end clip at the start.

The same applies to clipping from the end; you can choose different numbers at the end of sequences. For example, it often makes sense to use longer windows at the end than at the start, since the quality rapidly improves at the start of sequences, but only slowly deteriorates at the end.

Automatically removing short and low-quality sequences after end clipping

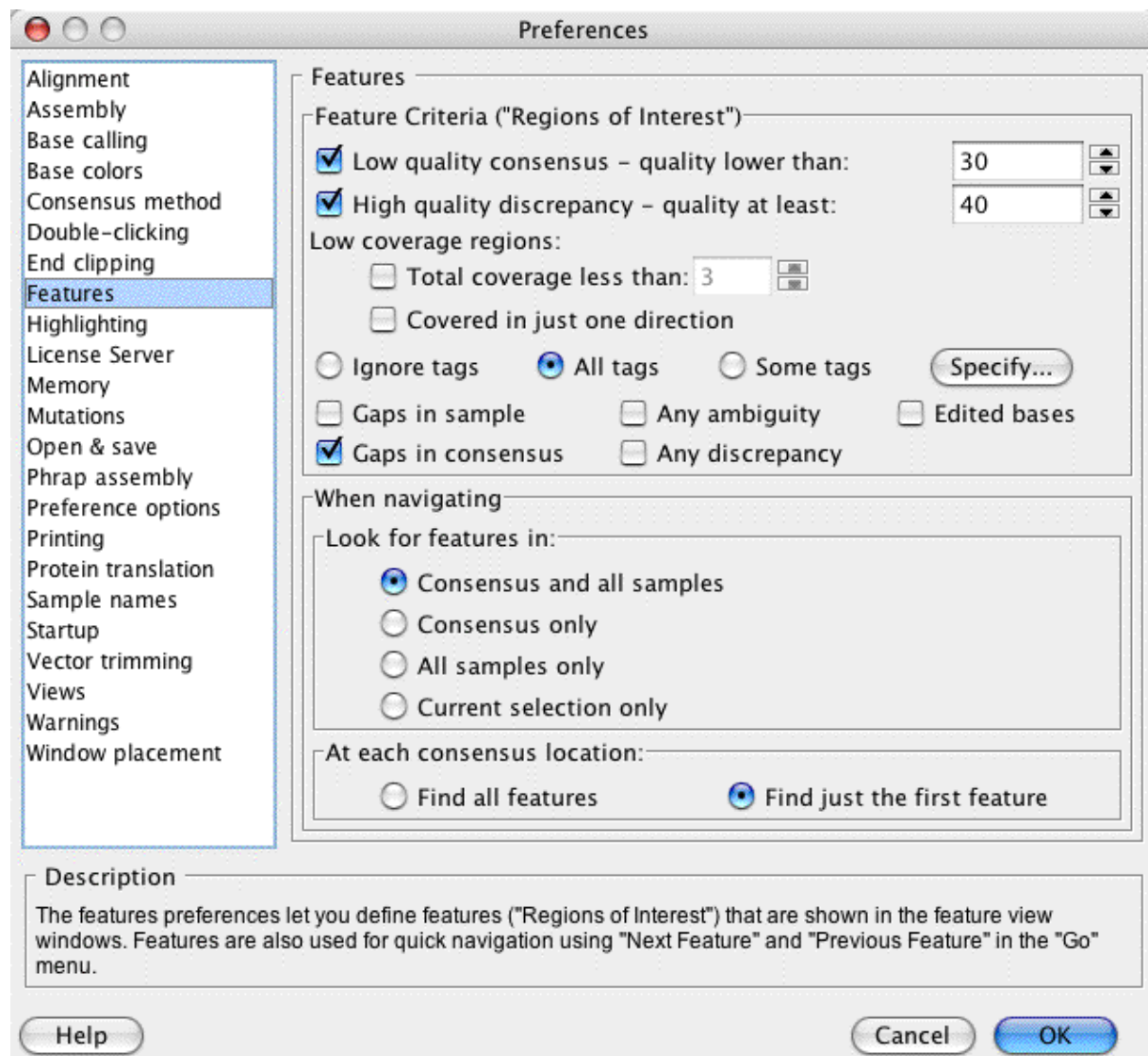
You have the option to automatically identify "bad" sequences after the end clipping, and move those to the trash. Such bad sequences can be due to failed sequencing reactions, or any number of other problems. We

suggest to move all sequences that are too short (for example, less than 25 or 100 bases) after clipping to the trash. A second widely used method to identify low-quality sequences is to count the number of bases with quality scores above 20 ("Phred 20 bases"). With the settings shown in the picture above, any samples that have fewer than 350 Phred 20 bases after the end clipping would be called bad and moved to the trash.

We suggest that you experiment with the different parameters for end clipping, and find a setting that works for your data. For example, if you sequence short PCR products, you should definitely reduce the number of Phred 20 bases required for a sequence to be kept, or perhaps uncheck this option. The results of end clipping are not saved until you save your project - you can use this to try out different parameters. Just close the project without saving, open it again, and end clip with different settings, until you found settings that feel "right" to you.

Feature Preferences

Aligner allows you to define criteria for places that you want to examine more closely, called "Features". Your definition of features can include low-quality consensus bases, low coverage regions, gaps, ambiguities, and tags; you make your choices in the "Features" preferences:

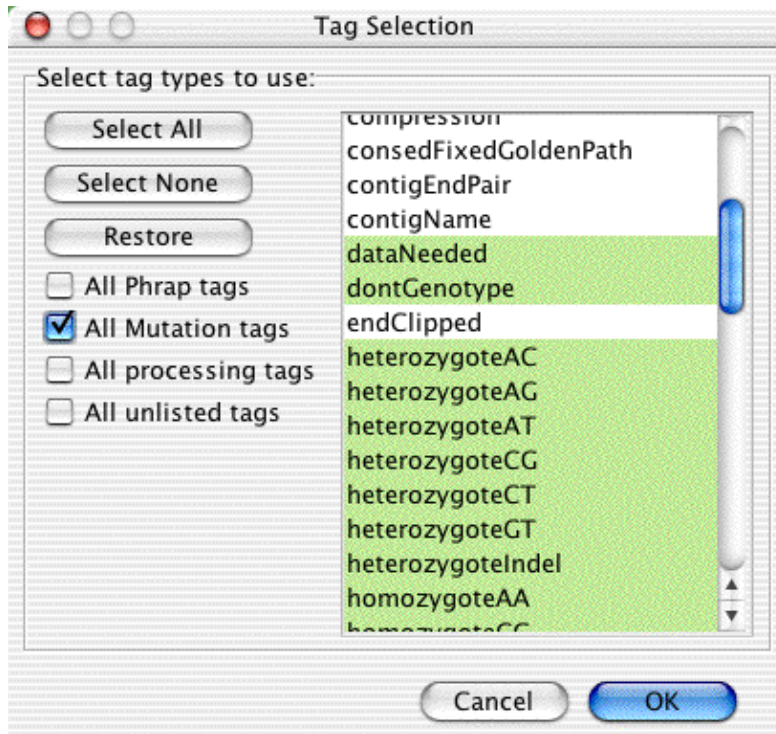


In the top panel, you define the criteria of features (interesting regions). Any place where one or more of the selected criteria are met will become a feature. This definition of features will be used by the "Feature View" window for contigs, and when navigating in a contig view using **"Next Feature"** and **"Previous Feature"** in the **"Go"** menu.

In the lower panel, you can define where Aligner should look for features when navigating - in the consensus sequence and all samples, or just a sub-set of the consensus or samples. This selection is used only for navigating - the feature view windows will always show all features for the entire contig, both in the consensus and in all samples.

Specifying subsets of tags

A number of different programs add tags to regions of sequences. For example, the assembly program Phrap adds tags for compressions and for regions that match regions in other contigs, and PolyPhred uses tags to mark possible mutations it has identified. You may want to use Aligner's feature view or feature navigation to look at only some of these tags - for example, only tags added by PolyPhred, but not tags added by Phrap. To do this, click on the "**Specify**" button in the Feature preference dialog. This will bring up the following dialog:

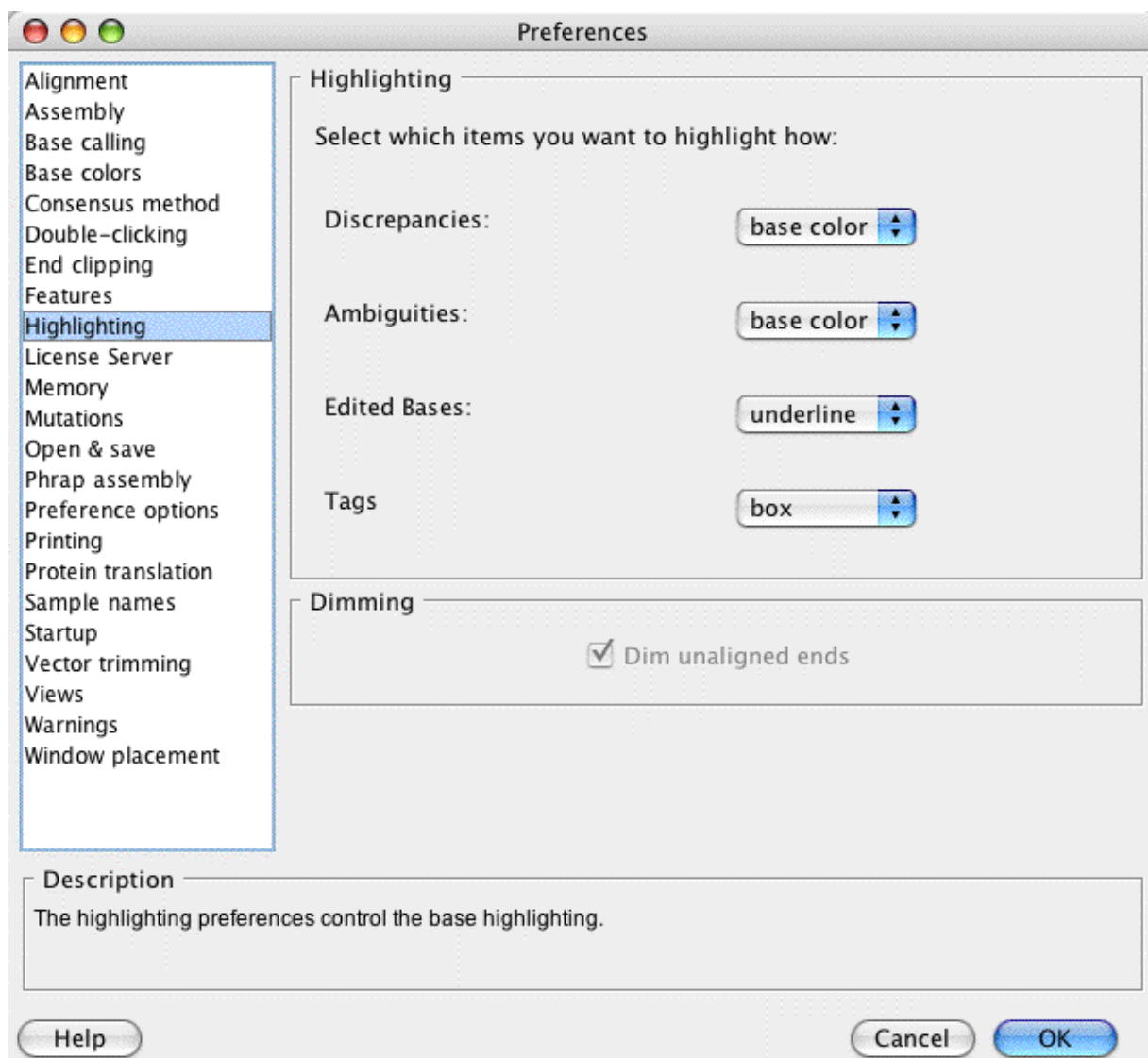


You can use the buttons and checkboxes on the left side to select or unselect groups of tags. You can fine-tune your selection in the list of tags on the right. To make or change **discontinuous selections**, use Control-click on Windows and Command-Click on OS X. Your selections will take effect when you click "OK" in the tag selection dialog, and then in the Preference dialog.

The list of tags in the tag selection dialogs may not show all the tags that are used in your projects. To use any tags that are not listed, make sure the "**All unlisted tags**" checkbox is selected.

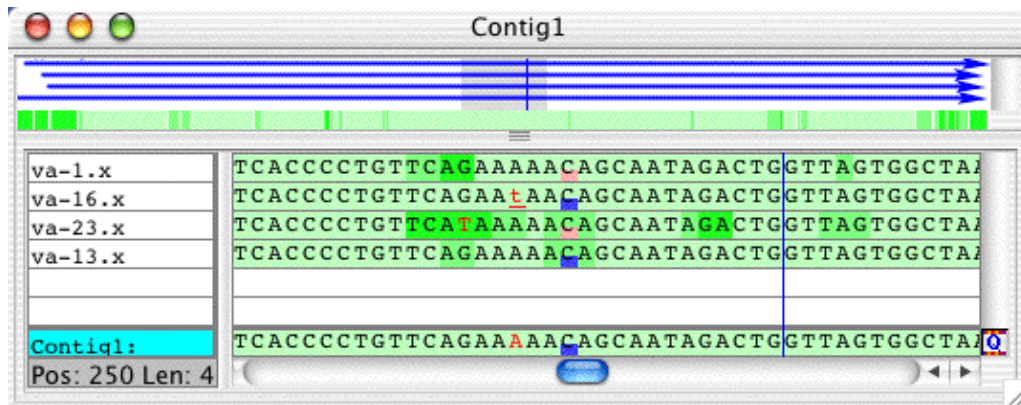
Highlighting Preferences

You can select how Aligner highlights discrepancies, ambiguities, edited bases, and tags in the highlighting preferences:



By default, Aligner will highlight discrepancies and ambiguities by using a different background or foreground color (depending on the color scheme and settings); edited bases by underlining the base call; and tags by a colored box that covers the bottom of the base. An example is shown below:

CodonCode Aligner User Manual

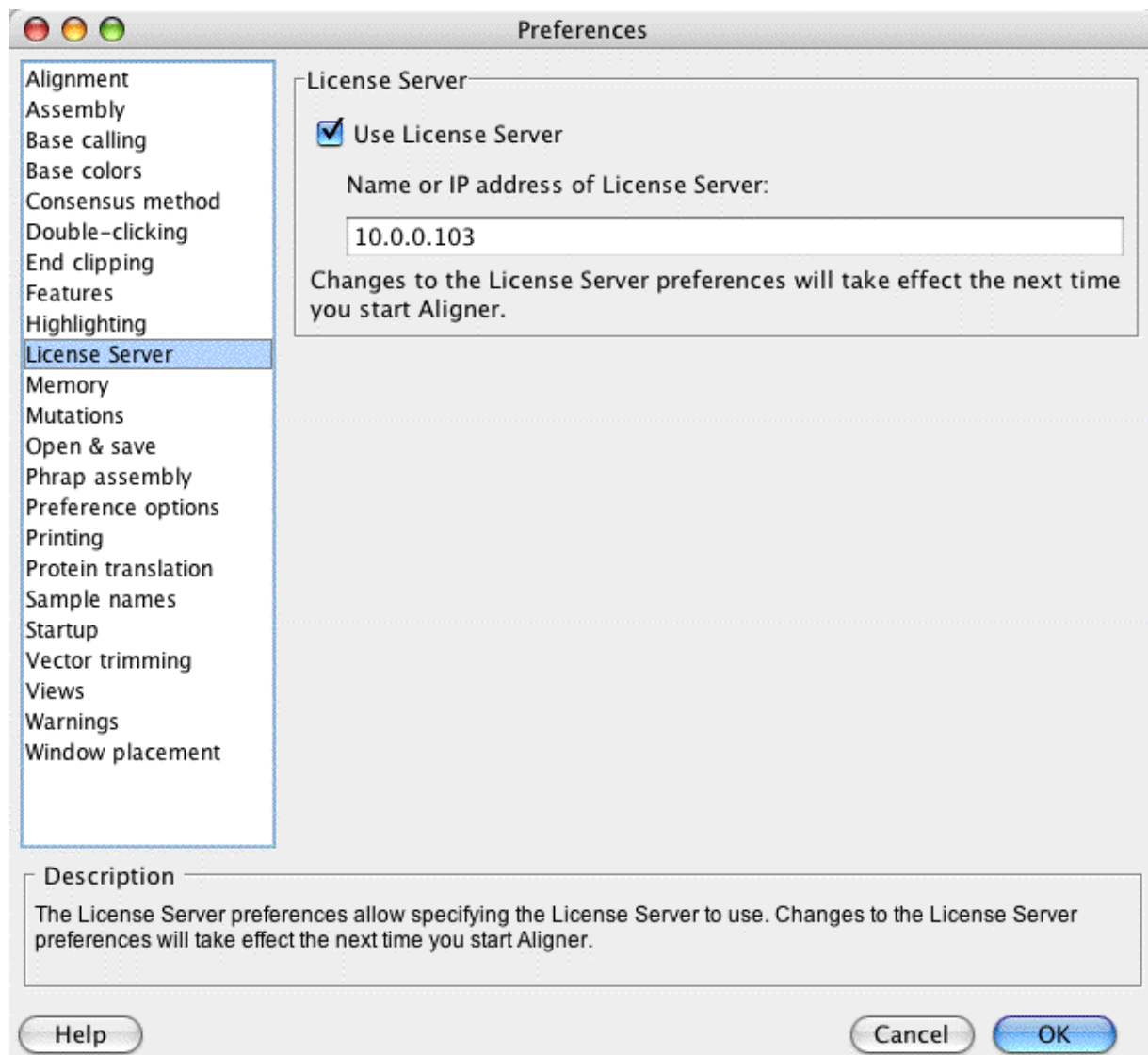


Here, discrepant bases (the two T's near the middle) are shown in red. The lower-case "t" in sample va-16.x was edited, as shown by the underline. The C's in the center have two different kinds of tags, as shown by the pink and blue boxes.

You can change these settings for each of the four categories by choosing an option in the pulldown menu next to it. Changes will take effect when you press "OK".

License Server Preferences

The License Server preferences allow you set whether CodonCode Aligner should use Aligner License Server , and the name or address of the License Server computer used. Most users will not need to use this preference panel, since CodonCode Aligner can automatically find a local License Server to use when starting up.



However, you may need to specify the keyserver address here if Aligner cannot automatically find the correct License Server . This can happen if:

- The License Server is on a different "subnet" (for example in a different department).
- A firewall between (or on) your computer and the License Server computer blocks the automatic License Server detection.
- CodonCode Aligner finds a License Server different from the one you want to use (for example the one from the neighboring lab).
- You are switching from a single-user license to using the License Server .

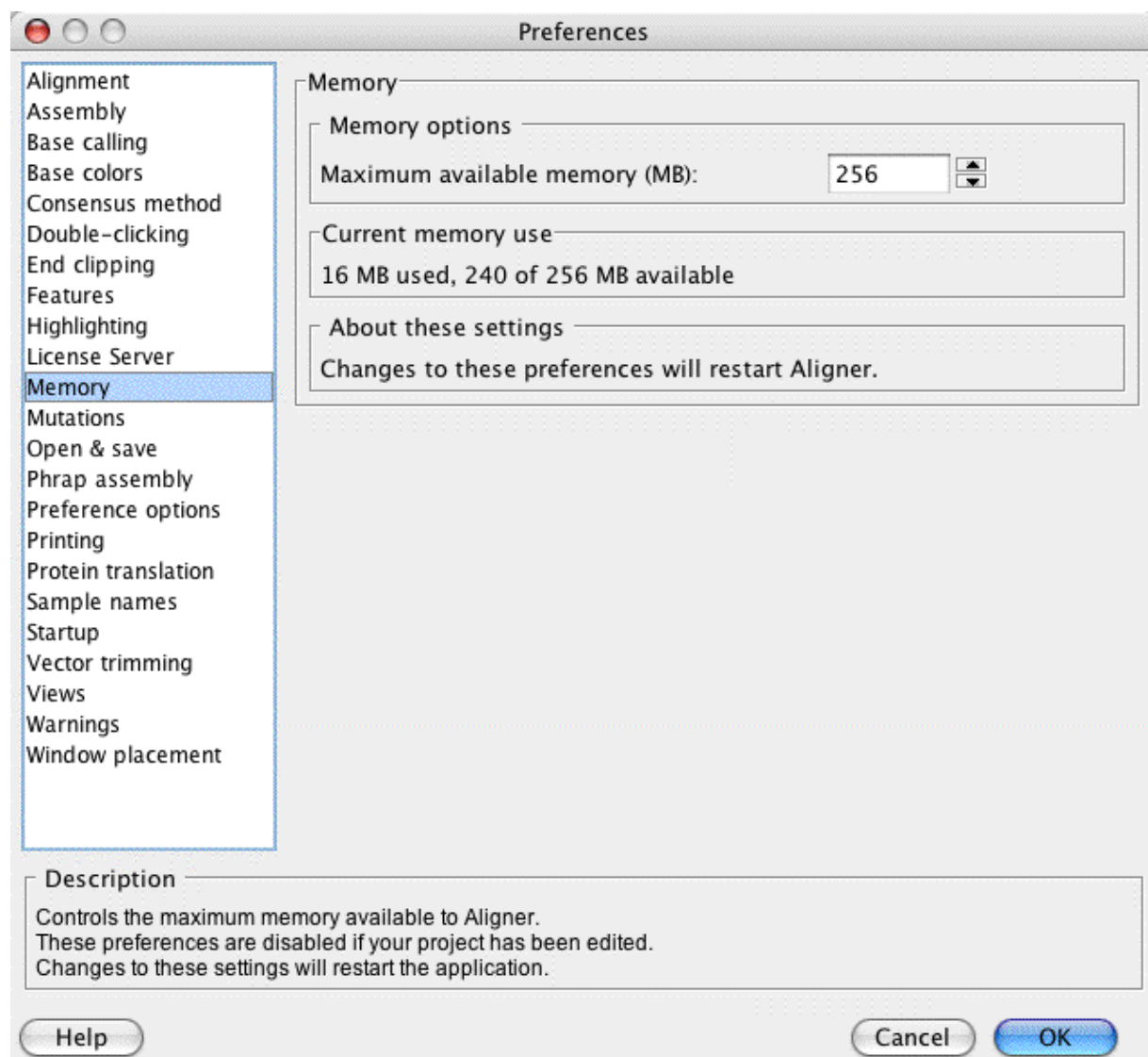
CodonCode Aligner User Manual

You can specify the License Server address either as computer name (for example "mylicenseserver.myuniversity.edu") or as an IP address (for example "10.123.012.20"). If you do not know the name or address to use, please contact your local system administrator.

Any changes made will take effect the next time you start Aligner after quitting your current Aligner session.

Memory Preferences

The memory preferences allow you to see how much memory CodonCode Aligner is currently using. On Mac OS X, it also allows you to set the memory available to Aligner:



The "Memory options" panel at the top shows you the maximum memory available to CodonCode Aligner.

The panel in the middle shows you how much memory CodonCode Aligner is currently using, and how much memory is available. Adding more samples to your project and opening views will increase the amount of memory used by Aligner.

Changing memory on OS X

If you are working with large projects on OS X, and Aligner runs out of memory, you can try to increase the memory Aligner can use using this panel.

Changing the memory requires that CodonCode Aligner restarts. Therefore, you can not change the memory if a project has been edited; you need to save your changes first. If your project has unsaved changes, this preference panel will be disabled.

The maximum amount of memory you can set here is determined by the amount of physical memory (RAM) installed in your computer. On OS X, Aligner will determine this amount, and limit the memory you can set to the installed memory.

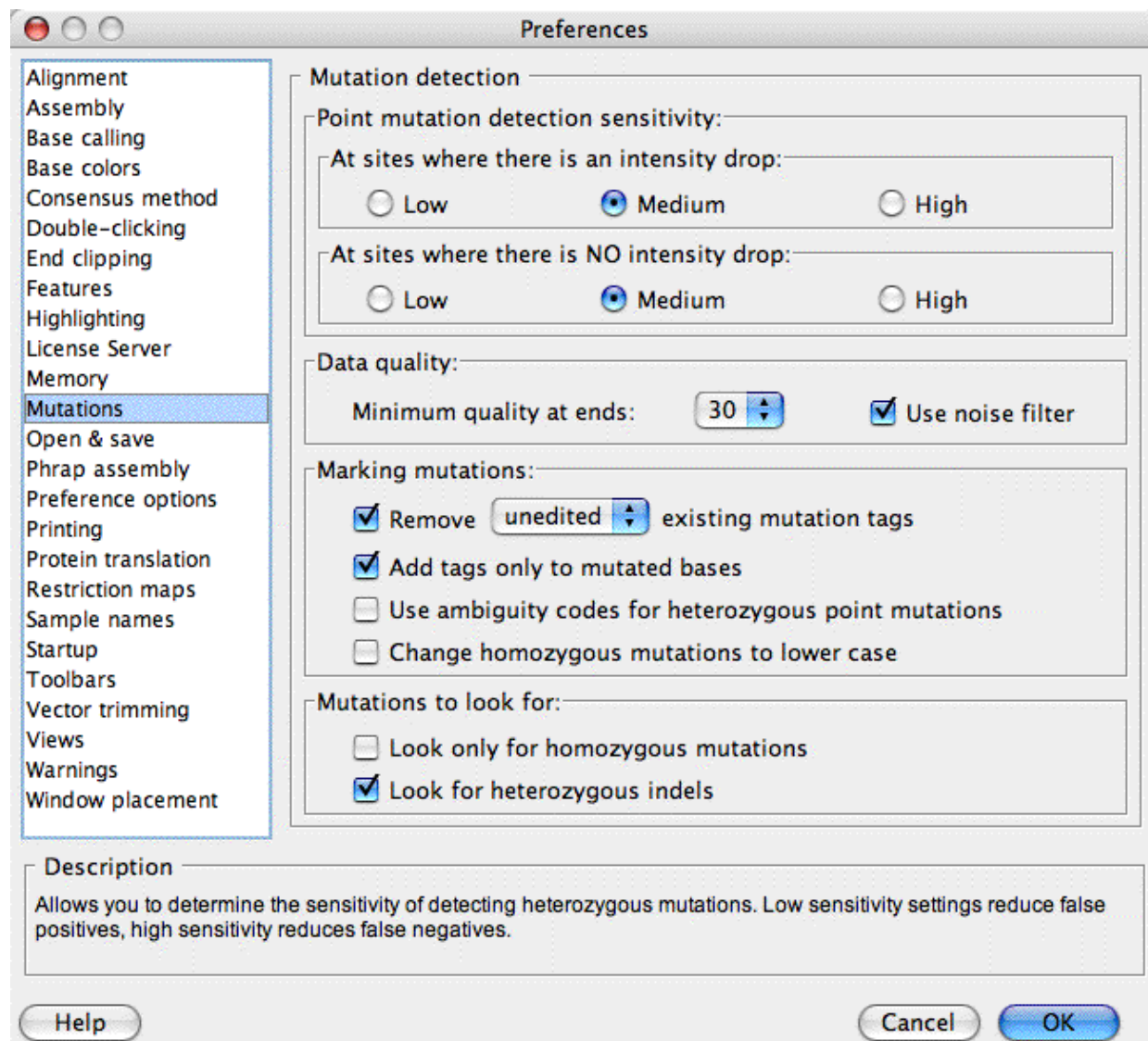
Memory on Windows

On **Windows**, memory used by CodonCode Aligner has to be changed by editing the file "CodonCode Aligner.ini" in the folder where CodonCode Aligner is installed.

1. Navigate to the folder where CodonCode Aligner is installed (usually C:\Program Files\CodonCode Aligner\).
2. Locate the file "CodonCode Aligner.ini" (you may not see the extension, depending on your settings).
3. Create a copy of the file.
If you are unable to create a copy of the file, you likely do not have sufficient permissions to edit this file or folder. Please contact your system administrator for assistance.
Note to Vista users: Even if you are using an "Administrator" account, you may not have permission to edit these particular files/folders. You may still need to adjust permissions to allow these edits.
4. Open the original "CodonCode Aligner.ini" in NotePad.
5. Look for a line that looks similar to:
`Virtual Machine Parameters=-Xms32M -Xmx50:64:384P`
6. Edit the "-Xmx" entry to change the amount of memory. Using the example above, this section reads "-Xmx50:64:384P".
This entry sets the maximum memory for Aligner to 50% of the system memory, assigning at least 64M and no more than 384M.
To increase the memory available to Aligner to 70% of built-in RAM or 512 MB (whichever is smaller), you would need to change this line to:
`Virtual Machine Parameters=-Xms32M -Xmx70:64:512P`
7. Save the changed file. Quit Notepad.
8. Try starting Aligner.
9. **If Aligner fails to start, then the amount of memory requested is too high.** Edit the file again and request less memory. If the problem persists, try quitting other applications and/or installing more memory.

Mutation Detection Preferences

To change the mutation preferences, select "**Preferences**" from the "**Edit**" menu on Windows, or from the "**CodonCode Aligner**" menu on OS X, and then click on "Mutations" on the left panel. The mutation preferences dialog looks like this:



Detection sensitivity

At the top, you can set the **sensitivity** for detection of heterozygous point mutations. The first set of radio buttons determines the sensitivity at places where a secondary peak is accompanied by a drop in intensity of the primary peak, compared to other samples at this position. The second row of radio buttons applies to places where there is no clear drop of intensities, for example because all samples at this position are heterozygous.

There is a trade-off between sensitivity and accuracy. With detection sensitivity set to "**low**", most or all point mutations identified by Aligner will be "real" - in other words, the false-positive rate will be low. However,

Aligner may miss some heterozygous point mutations, for example if the secondary peak is weak. On the other hand, a **"high"** sensitivity setting will miss few, if any, heterozygous point mutations, but may incorrectly identify some secondary peaks as heterozygous point mutations when they are really caused by "noise" in the sequence traces.

Your choices in the **"Data quality"** panel also affect the false-positive to false-negative balance. At the start and end of sequence traces, secondary noise peaks are most common, which can lead to false positives. Aligner tries to reduce this problem by examining sequence quality at the start and end of each trace, and excluding low-quality regions. You can determine the stringency of this step by setting the **"Minimum quality at start and end"**. The value chosen there is the sequence quality ("Phred quality") required before Aligner starts analysing; choosing higher values, e.g. 40, will give fewer errors near the end of sequences. *You can see which regions Aligner excluded from analysis by looking at the "dataNeeded" tags, which are shown as yellow boxes (depending on your [highlighting preferences](#)).*

In general, the **"medium"** sensitivity settings should work reasonably well; however, you should adapt the settings to the particular needs of your project and the quality of your sequence data. If identifying every single potential point mutation is important, select "high" in the sensitivity check boxes, and uncheck the "Use noise filter" check box. We strongly suggest that you try the different settings with your own data to find out which setting works best for you.

Note that most detection sensitivity settings only apply to samples that have chromatograms; for text samples, the bases can be analyzed as they are. The only exception is the "Data quality" section - low-quality sequence at the ends of text sequences will also be ignored if the text sequences have quality scores.

Marking mutations

The section labeled "Marking mutations" lets you fine-tune how CodonCode Aligner marks mutations it finds. The main method Aligner uses is to add tags to mutated bases. To facilitate analysis with other programs, however, Aligner can also convert heterozygous mutations to ambiguities (e.g. using "R" for heterozygous A+G), and change the case of homozygous mutations to lower-case (e.g. changing "A" to "a").

Aligner will add mutation tags at bases where it finds a putative heterozygous or homozygous mutation - except at bases where mutation tags have been edited or confirmed by the user, and kept between successive rounds of mutation detection, as explained in the next paragraph.

If you perform mutation finding more than once, you should check the **"Remove existing mutation tags"** checkbox. If this checkbox is checked, Aligner will remove existing mutation tags in the samples that you have selected before finding mutations. You can either remove only unedited tags or all tags. If "unedited" is selected, any tags edited or added by users (and not the program) will remain; this includes confirmed tags, and tags where the tag type was changed, for example from homozygous to heterozygous. When removing tags from previous mutation detection rounds, Aligner will also undo automatic edits (like changes to ambiguities) done during previous "Find mutation" cycles.

If the **"Add tags only to mutated bases"** checkbox is checked, Aligner will add tags only to bases that differ from the consensus sequence. This is useful if, for example, you are looking for rare mutations (SNPs). In other cases, for example when genotyping, you may want to add a tag to all samples at each consensus position where a mutation is found; to do so, make sure the **"Add tags only to mutated bases"** checkbox is **not** checked. Then, Aligner will add tags that characterize the base at a given consensus position to all samples, even if just one sample out of a hundred differs from the consensus base.

The last two checkboxes in the "Marking mutations" section determine if Aligner will change base calls at homo- or heterozygous mutations. The default behaviour is that Aligner will leave the bases unchanged, and only add tags at mutated bases. If you want to export your data for subsequent analysis with other programs (for example the population genetics program [Arlequin](#)), you may need to have ambiguity codes at heterozygous bases. If you mark the checkbox "**Use ambiguity codes for heterozygous point mutations**", Aligner will change the call for all heterozygous point mutations it identifies to [IUPAC ambiguity codes](#).

You may also need to be able to identify homozygous mutations in exported sequences. If you mark the checkbox "**Change homozygous mutations to lower case**", Aligner will change the base calls of all homozygous mutations it identifies to lower case (e.g. from "A" to "a").

Please note: the changed base calls and the tags are currently not linked - if you would want to manually change a base that Aligner identified incorrectly as a heterozygote, you will need to change **both** the base **and** the tag.

Finding only homozygous mutations

In some projects, you may know that you do now have any heterozygous mutations - for example when you are sequencing clones rather than PCR products. When looking for mutations in such project, make sure that the checkbox "look only for heterozygous mutations" is checked.

Of course, when looking for homozygous mutations, you could just look for any discrepancies rather than using Aligner's mutation finding. However, there are several reasons to use Aligner's "Find Mutations":

- Aligner can automatically ignore the low-quality regions at the end (as defined in the "Data quality" pulldown)
- Aligner adds mutation tags, which can be printed and exported for analysis in other programs like Microsoft Excel

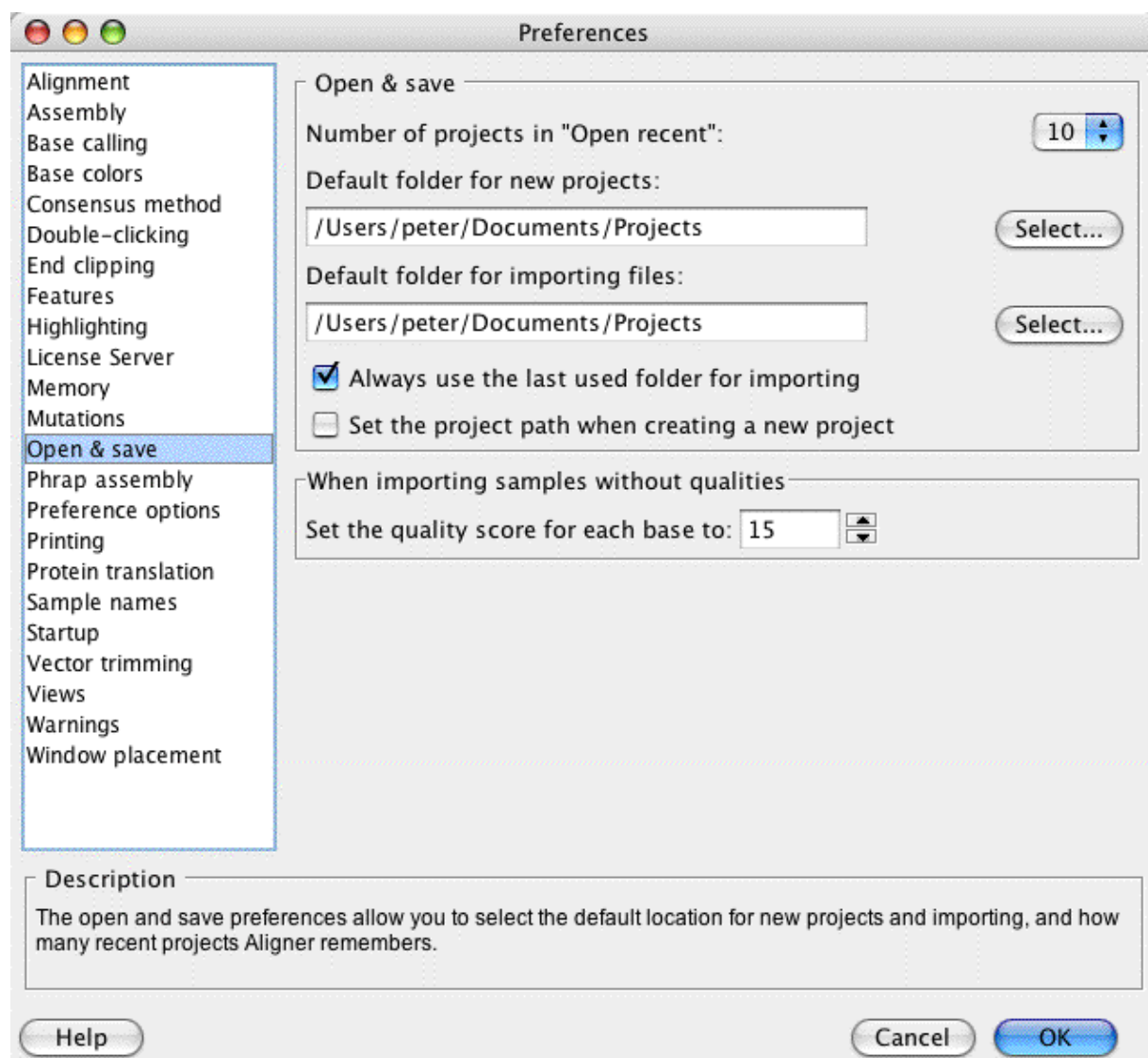
Finding indels

When looking for mutations, you have the option to also look for heterozygous insertions and deletions by checking "**Look for heterozygous indels**". If you have already searched for heterozygous insertions and deletions (for example before end clipping), you can uncheck this checkbox, and the mutation finding will be a bit faster.

Keep in mind that **it is generally recommended to search from heterozygous indels before end clipping**, since end clipping may clip the entire heterozygous part.

Open & Save Preferences

In the "Open & save" preferences, you can set what Aligner does when opening and saving projects:



Specifically, you can:

- determine how many recent projects CodonCode Aligner remembers and displays in the "Open recent" menu
- set default folders for saving your projects
- set default folders for importing
- tell Aligner to remember the last location from where you imported files, so that Aligner will return to this folder the next time you choose an "Import" command ("Add Samples", "Add Folder", or "Add Assembly")
- specify if you want to set the name and path of a project right away when creating new projects (if not, the name and location will be set the first time you save a project)
- set the default quality that will be assigned to sequences without qualities when they are imported

The default quality for sequences without qualities you set here will be applies to any sequences imported later where Aligner determines that the sequence has no qualities, or has artificial qualities. This value is used whenever:

- you import sequences from text files (for example FASTA files) that do not have qualities
- you import sequences from files like PHD or SCF files that have qualities, but Aligner determines that the qualities are artificial because:
 - all qualities are the 0 or -1 (for example, if an SCF file was created from an ABI file without qualities)
 - all qualities have exactly the same value, and the sequence is at least 15 bases long (for example an artificial SCF file created with tools like `mktrace` or `fasta2Phd.perl`)
 - a sequence from a PDF file was previously identifies as having artificial sequences by Aligner, and has a corresponding tag in the PHD file

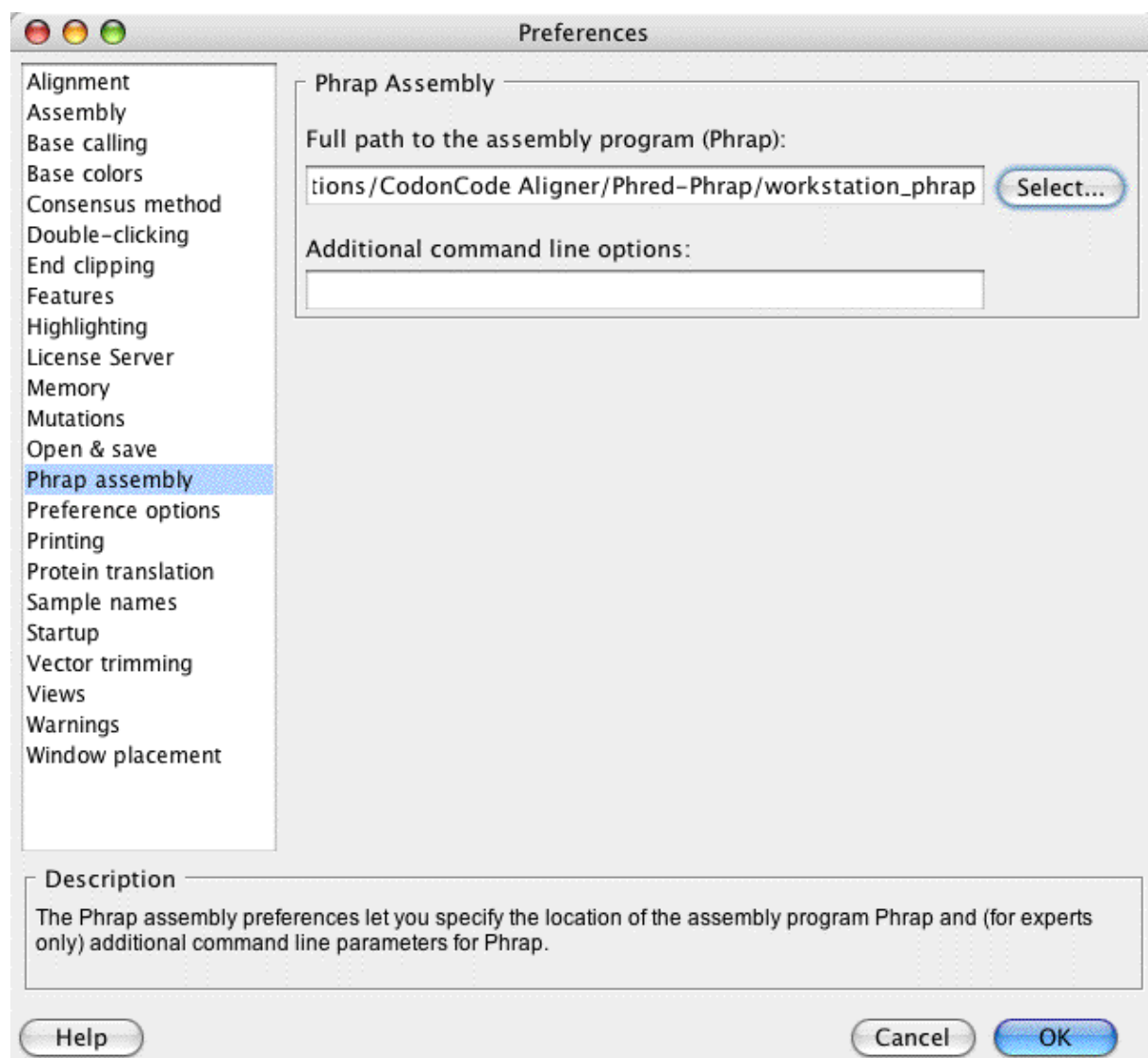
Changing this value will **not** affect any sequences you previously imported. If you want to change the assigned artificial quality values for a sequence in your project, you currently have to remove the sequence, and then import it again.

What are artificial qualities good for?

Good question. They cannot be used for end clipping, since end clipping requires real quality scores assigned by programs like Phred. However, the qualities are used to determine the consensus sequence during assembly and alignment to a reference sequence. In general, you should use low artificial scores (like the default value of 15). However, if you want to make sure that a sequence without real qualities is used as to contribute more to the consensus, you can assign higher values, all the way up to 90.

Phrap Assembly Preferences

In the Phrap assembly preferences, you can specify details needed for assembling samples with [Phrap](#):



You specify the location and name of Phrap assembly program in the upper text field.

If you want to use the workstation version of Phrap that was installed with Aligner, the path to Phrap should be correct, and not need any changes. However, if you use your own installation of Phrap, you may need to specify the location of the file on your system.

Please note that using Phrap from Aligner requires that you have a trial license or a purchased license; Phrap use is not enabled in demo mode. Academic users who purchased a license for CodonCode Aligner can use the workstation version of Phrap free of charge for academic research; users at companies will have to purchase a separate license to use Phrap.

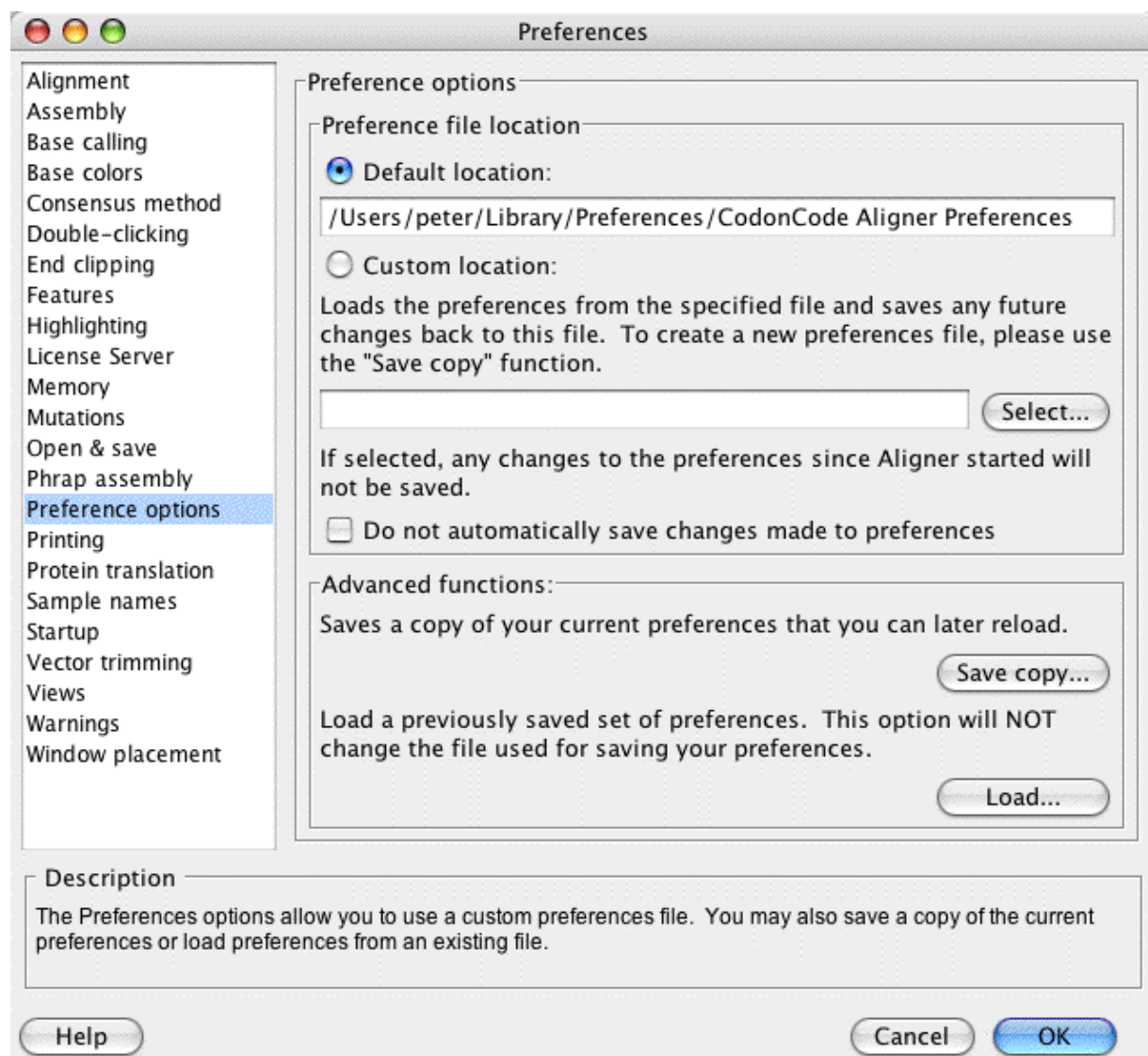
In the bottom text field, you can specify additional command line options for Phrap. In general, please leave this line blank - **only experts who really know what they are doing should specify command line options**

here!

For more information about assembling with Phrap, please read the "[Sequence Assembly With Phrap](#)" section.

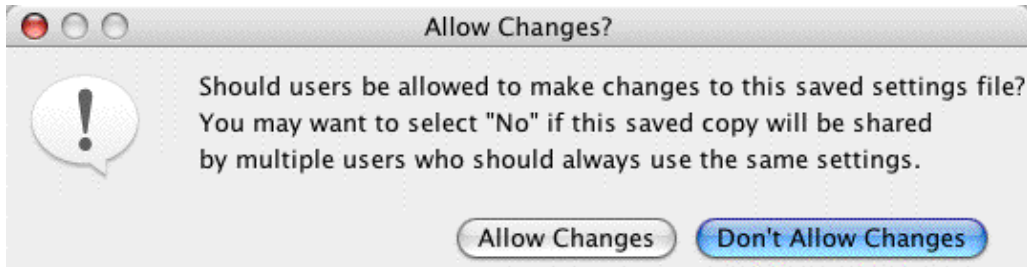
Preference Options

CodonCode Aligner offers the option to save and load copies of Aligner's preferences in the "Preference options" panel of the Preferences dialog:



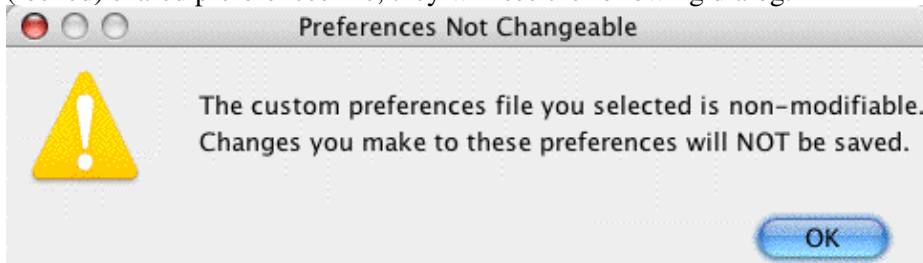
A typical use example is a group of **several scientists who want to share a "locked" set of preferences**, so that all analyses will be performed with the same settings. This can be done as follows:

1. Change all preferences to the desired settings.
2. Open the Preferences dialog, and select "Preference options" on the left.
3. Save a copy of the preferences by using the "Save copy..." button. The following dialog will appear:



If you want to protect the saved file against changes, press the "Don't Allow Changes" button. A "Save As.." dialog will be shown next, where you can choose the name and location the preferences are saved to. Choose a folder that all users can access, for example on a file server or shared disk.

4. Each user that want to use the shared preferences now has to choose this file by starting CodonCode Aligner, opening the Preference options dialog, and clicking the "Select..." button. After selection the (locked) shared preferences file, they will see the following dialog:



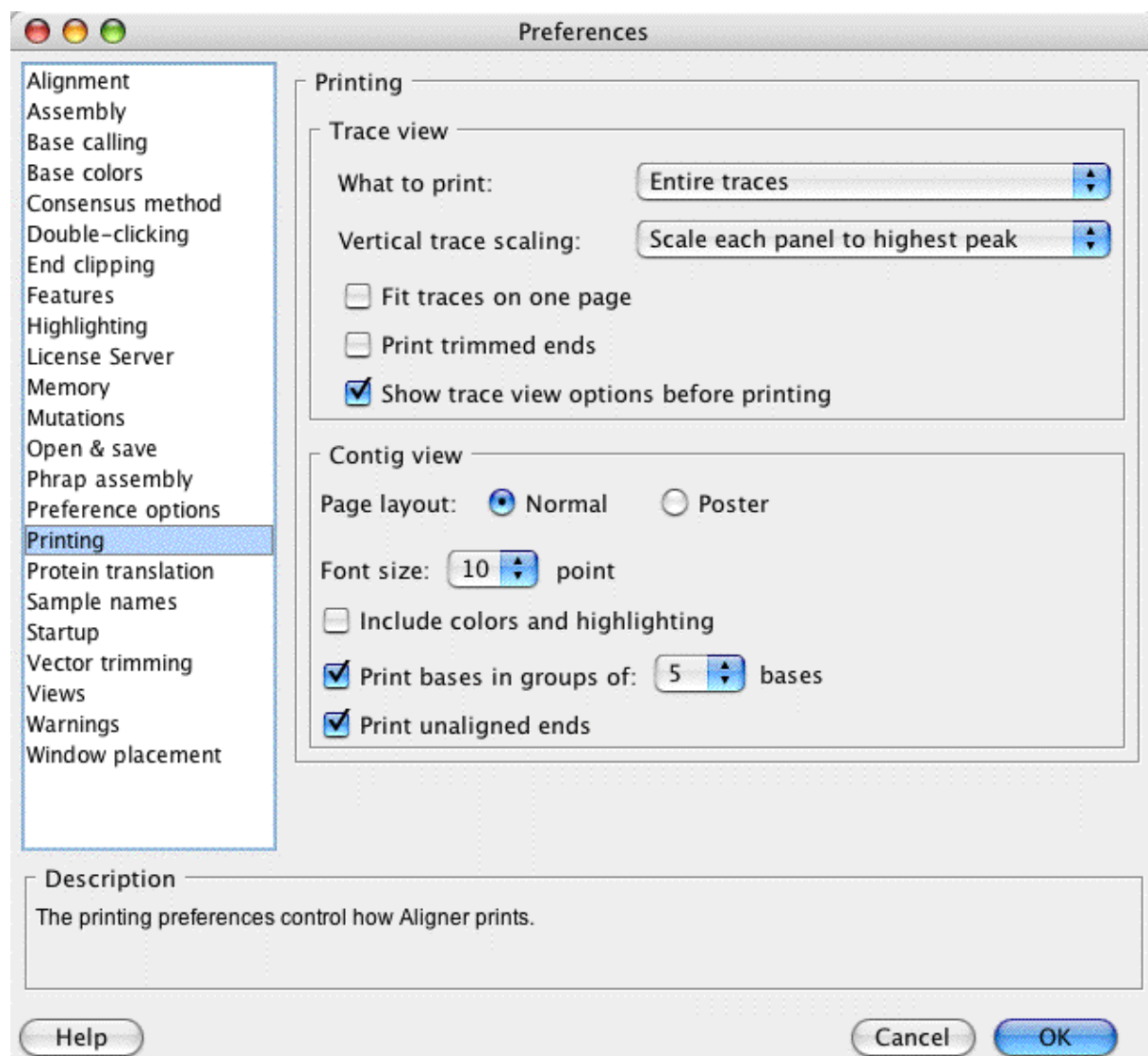
After clicking "OK" in this dialog and then also in the Preferences dialog, the shared preferences will be used. Note that each user can still change settings while Aligner is running; however, such changes will not be saved when Aligner quits.

To change back to separate preferences, a user can select the "Default location" button.

Printing Preferences

The Printing Preferences allows you to determine how different windows will be printed.

To see the printing preferences, select "**P**references" from the "**E**dit" menu on Windows, respectively the "**C**odon**C**ode **A**ligner" menu on OS X. On the left side of the preference dialog, click on "Printing"; your dialog should now look like this:



Currently, you can only set options for [trace view printing](#) and [contig view printing](#). If you would like more options for printing other views, please let us know!

Trace View Printing

What to print when printing traces is determined in the first pull-down menu. Your options are:

- **Entire traces** - this will print the entire sequence trace for each sample currently in the trace view. Each sample will be printed using several panels on one or more pages before the next sample is

printed.

- **Only visible regions** - this will print just the part of the traces that is currently showing on the screen. If your trace view window contains multiple traces, one panel for each trace will be printed, similar to the way the sequences are shown on the screen.

The **vertical trace scaling** lets you select how the traces are scaled when printed:

- **"Use Trace view scaling"** will print traces using a constant scale factor for all panels in a sample; the scale factor used will be the one currently used for this sample in the trace view.
- **"Scale each panel to highest peak"** will calculate a separate scale factor for each panel in each sequence, based on the highest peak in this section of the sequence. This will give you a nicer looking printout (more even peak intensities) when printing entire sequences.
- **"Equalize traces for each panel"** is useful if the intensities for the four bases are very different in your samples. This will calculate separate scaling factors for each of the four traces in each panel, based on the highest peak in this section of the trace.

If you select **"Fit traces on one page"**, Aligner will try to fit everything onto one page, scaling traces as needed. If you chose to print "Entire traces", each sample will be printed onto a separate page.

If the **"Fit traces on one page"** checkbox is not checked, the height of each panel will be determined by the height of the trace view windows defined in your [View Preferences](#) when printing entire traces, or by the height of the panels on the screen when printing only visible regions.

Usually, only the trace between the first called base and the last called base is printed. If you want to also print the trace before the first base and after the last base, make sure **"Include trimmed ends when printing"** is checked.

Contig View Printing

The **"Page Layout"** used when printing is determined by the first set of radio buttons (*note that this is **not** the same thing as selecting the page orientation (i.e. "Portrait" or "Landscape") in the "Page Setup" dialog!*). The page layout options for contig printing are:

- **Normal** – Prints the sample and contig names with as much of the sequences as will fit across the page. If space permits, additional rows with sample and contig names followed by the sequences will be included on the page.
- **Poster** - Prints the sample and contig names with as much of the consensus sequence as will fit across the page. Continues printing the sequences across additional pages, *without* including the sample or contig names on those pages.

The **"Font size"** pull-down menu selects the size of the font used for printing the contig.

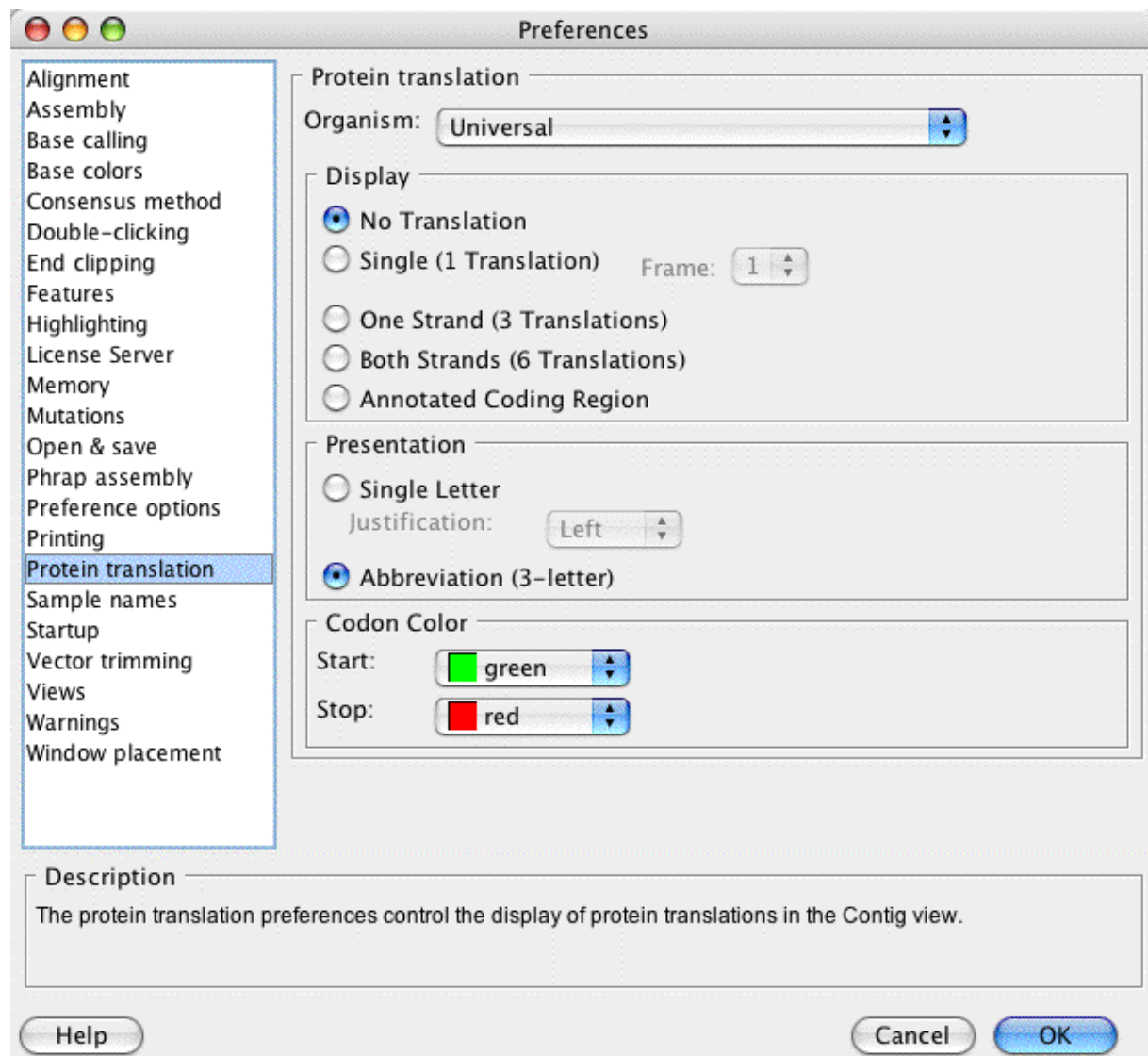
If the **"Include colors and highlighting"** checkbox is checked, the contig will be printed as it appears in the contig view, including colored bases, colored backgrounds, and other highlights. If this box is unchecked, then the bases will be printed as a black letters on a white background, ignoring all highlights other than underlining.

To print sequences grouped every 3, 5, or 10 bases, with spaces in between, check the box labeled **"Print bases in groups of # bases"**. If this box is not checked, bases will be printed continuously across the page.

Usually, unaligned ends of samples are printed with the contig. If do not want to print the unaligned ends, uncheck the “**Print unaligned ends**” checkbox.

Protein Translation Preferences

The organism for which the sequence translation is being performed can be selected from the **Organism** list. In the "Display" section, you can choose if you want to see the translation of one frame, three frames, or all six frames.



Amino acid names are commonly represented by a single letter (A = Alanine) or an abbreviation (Gly = Glycine). The **Single Letter** and **Abbreviation** choices let you set your preference.

The color of start and stop codons can be changed to make them easier to see. Use the **Start** and **Stop** color lists to select the colors you want.

Restriction Map Preferences

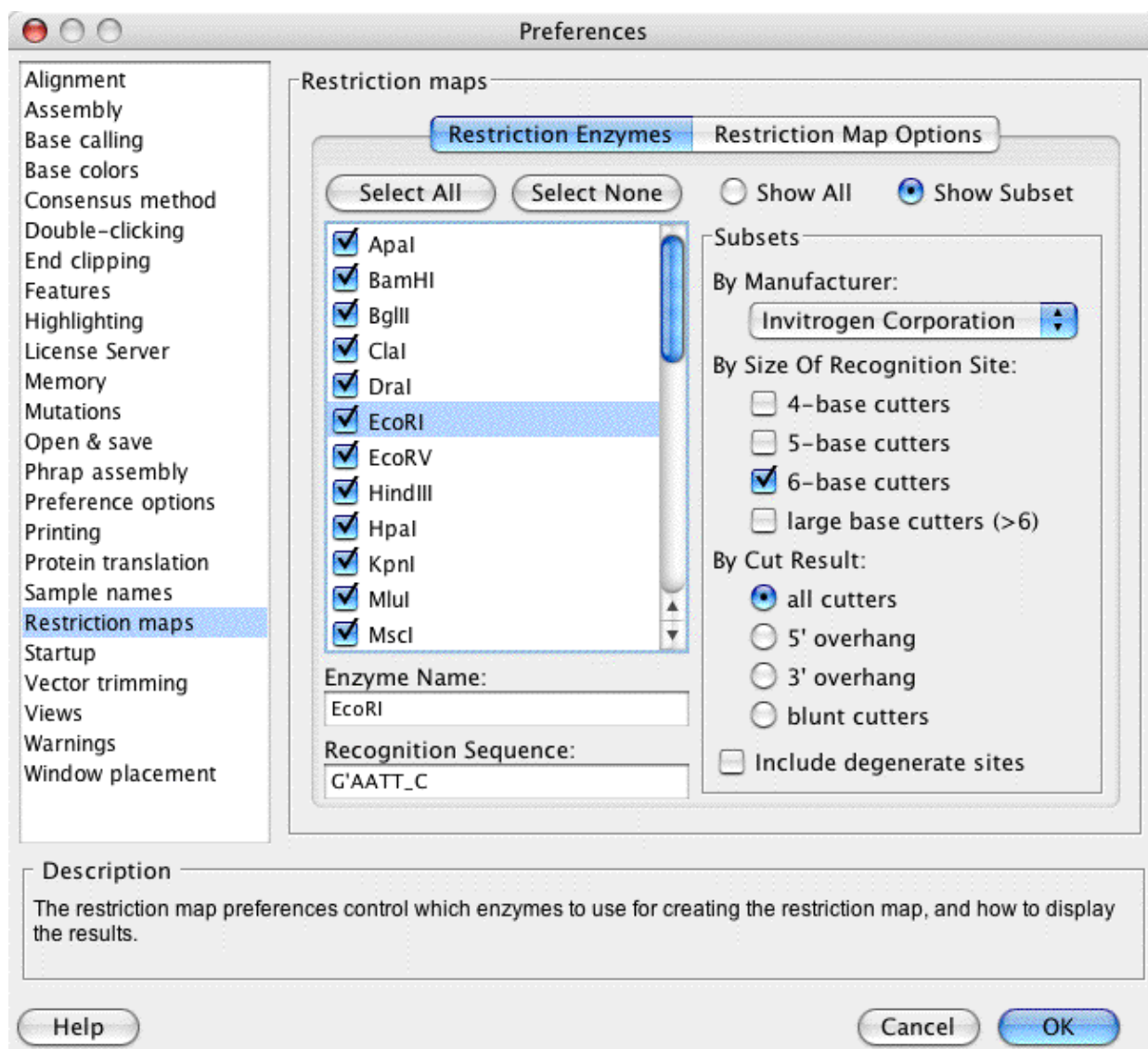
In the restriction map preferences, you specify which enzymes to use for generating restriction maps, and how to display the results.

To change the restriction map preferences select "**Preferences**" from the "**Edit**" menu on Windows, or from the "**CodonCode Aligner**" menu on OS X, and then click on "Restriction maps" on the left panel. You can also access the restriction map preferences from the [restriction map view](#) by clicking on the restriction map icons in the toolbar.

The restriction map preferences are displayed in two different tabs. One tab allows to change the restriction map options, the other tab allows to select enzymes for the digest. The tabs are shown at the top of the restriction map preferences and you can switch from one to the other by clicking on the tab.

Selecting Enzymes

The "Restriction Enzymes" tab in the restriction map preferences allows you to select enzymes for the digest:



The left side of these preferences contains a list of enzymes. This list reflects a set of enzymes that can be chosen on the right side of this view. The two radio buttons on the top right side allow you to show either all enzymes or to show only a subset of enzymes in the list on the left side. If the "Show Subset" radio button is selected, you can choose your subset using the preferences in the "Subsets" section:

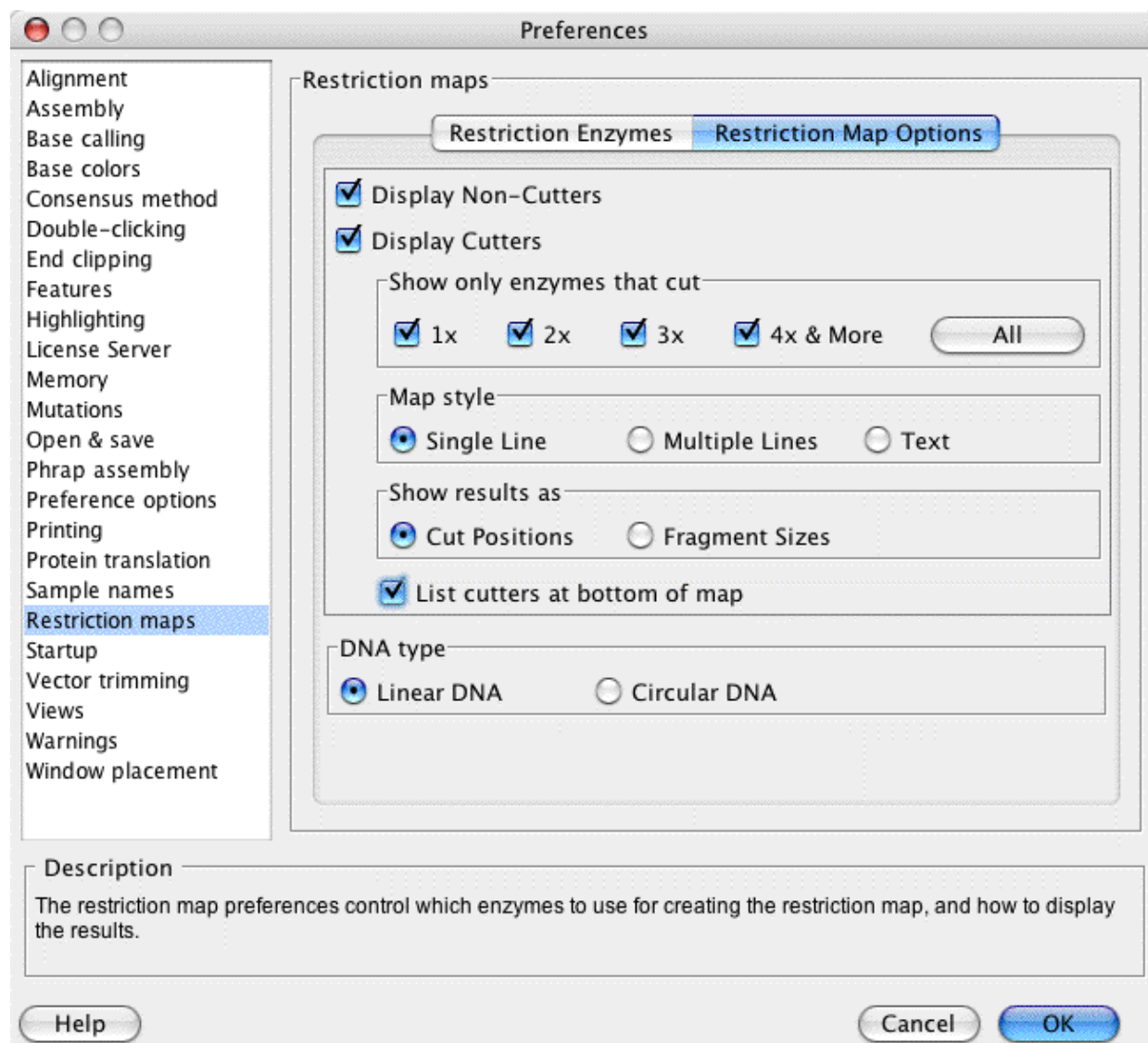
- You can select a subset by **manufacturer**. In the example above only enzymes from Invitrogen Corporation are shown in the list on the left side.
- You have the option to choose a subset of enzymes by the **size of their recognition site**. Above, only 6-base cutters are included in this subset.
- The **cut result** can also be used to specify a specific subset. For example you can include only enzymes that generate a 5' overhang in your subset.
- The "**Include degenerate sites**" checkbox allows you to in- or exclude enzymes that have one or more ambiguities in their recognition site.

The list of enzymes on the left side updates according to the selection for the subsets. You can select or unselect enzymes in the list through the checkboxes in front of the enzymes, and by using the "Select All" and "Select None" buttons above the list.

The enzyme name and recognition sequence for the enzyme currently selected are shown below the list.

Restriction Map Options

The "Restriction Map Options" tab in the restriction map preferences allows you to change how the map is displayed, and which DNA type to use:



In the first section, you can select if **cutting and/or non-cutting enzymes** should be displayed. If you choose to display cutters, you can set how to show them and the cut results in the restriction map:

- You have the option to show only those enzymes that **cut a specific amount of times**. For example, the restriction map will show only unique cutters if you have only the "1x" checkbox selected.
- You can set the **map style**:
 - ◆ A "Single Line" map displays a graphical map where all enzymes that cut are shown together (on a single line).
 - ◆ The "Multiple Lines" map shows a separate graph for each enzyme that cuts.
 - ◆ The "Text" map lists the cut results in text format.

CodonCode Aligner User Manual

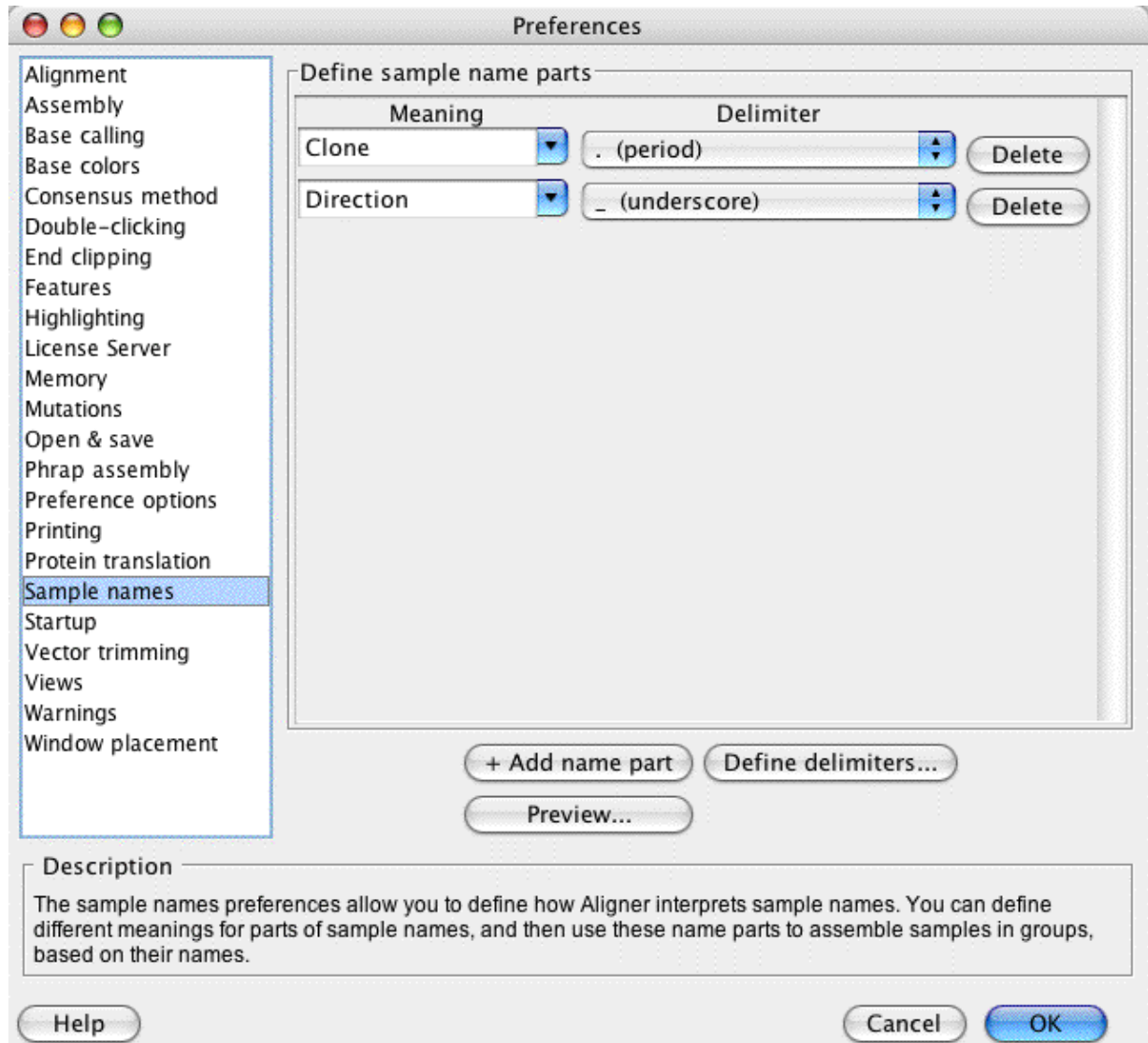
- The **results** can either be displayed as "Cut Positions" or as "Fragment Sizes" in the map. The cut positions and fragment sizes are shown in base pairs.
- You also have the option to **list the cutters** in a summary at the bottom of the map. The summary shows the cutters sorted by the number of cuts. A summary of non-cutters is always listed if the "Display Non-Cutters" checkbox is selected.

In the second section you set the **DNA type** used for the digest to be either linear or circular DNA. This can affect your results for fragment sizes and cut positions shown in the map.

Sample Name Preferences

Defining sample names

CodonCode Aligner can interpret ("parse") sample names to automatically group samples, and assemble samples in groups. You can define how sample names should be parsed in the Sample name preferences:



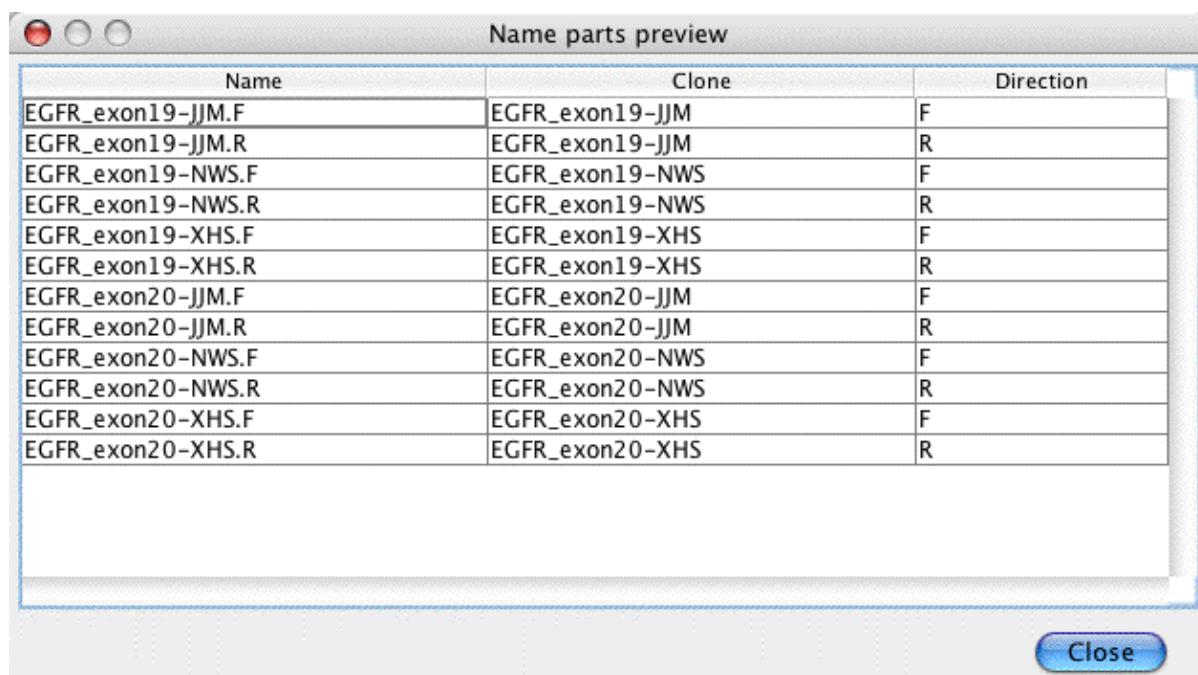
In the screen shot above, Aligner would interpret samples as follows:

- Everything from the beginning of a sample name to the first period ('.') would be interpreted as the clone name.
- The part after the first period up to the next underscore would be interpreted as the direction (typically, 'F' would indicate forward reads and 'R' reverse reads).
- Anything that follows after the underscore will simply be ignored.

You change the meaning of a name part, and the delimiter that indicates the end of a name part, using the respective combo boxes. You can add more name parts using the "+ **Add name part**" button, and delete name

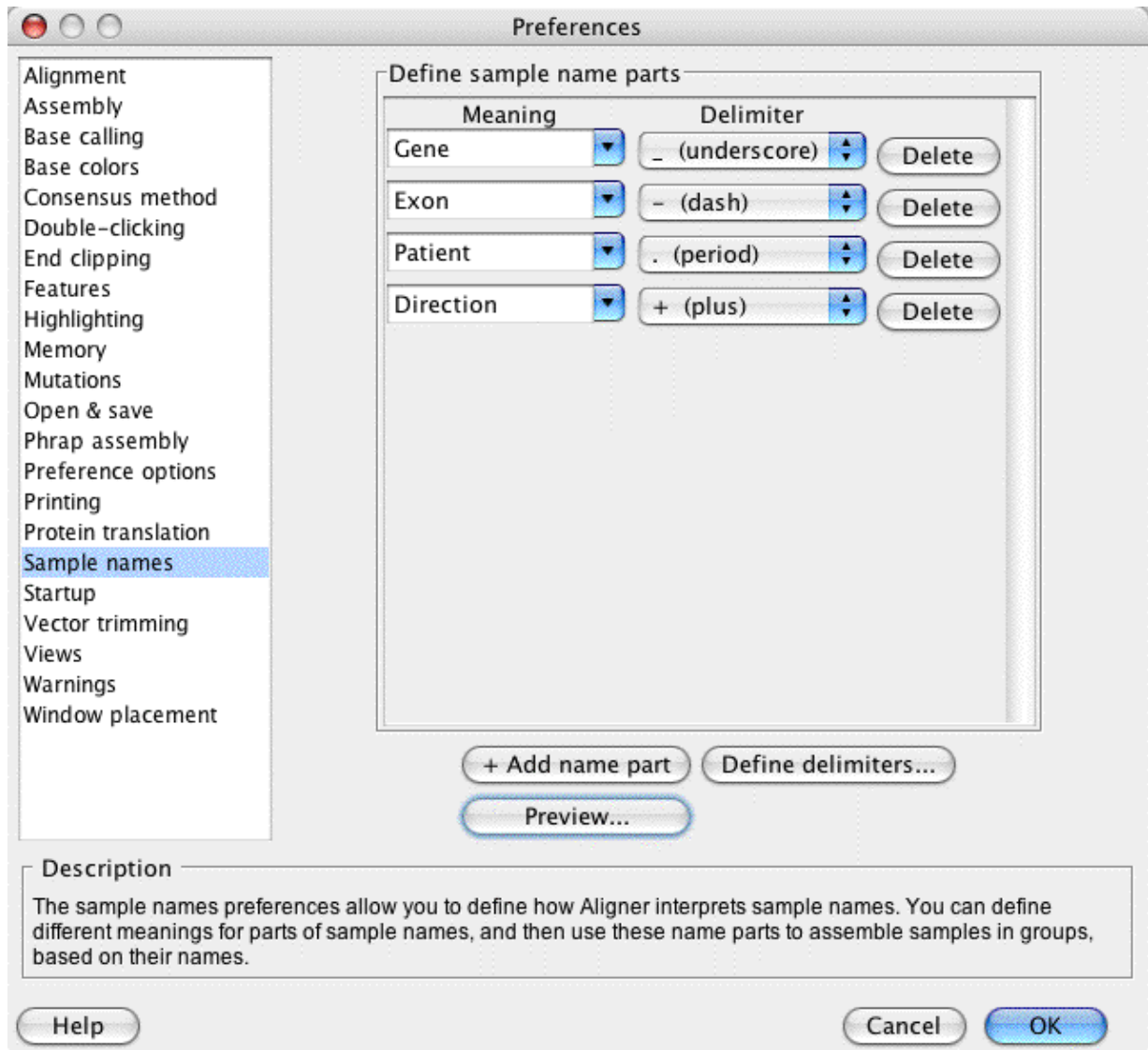
parts using the "**Delete**" button next to the part you want to delete.

You can view how sample names will be interpreted by clicking on the "**Preview...**" button. This will bring up a preview dialog:

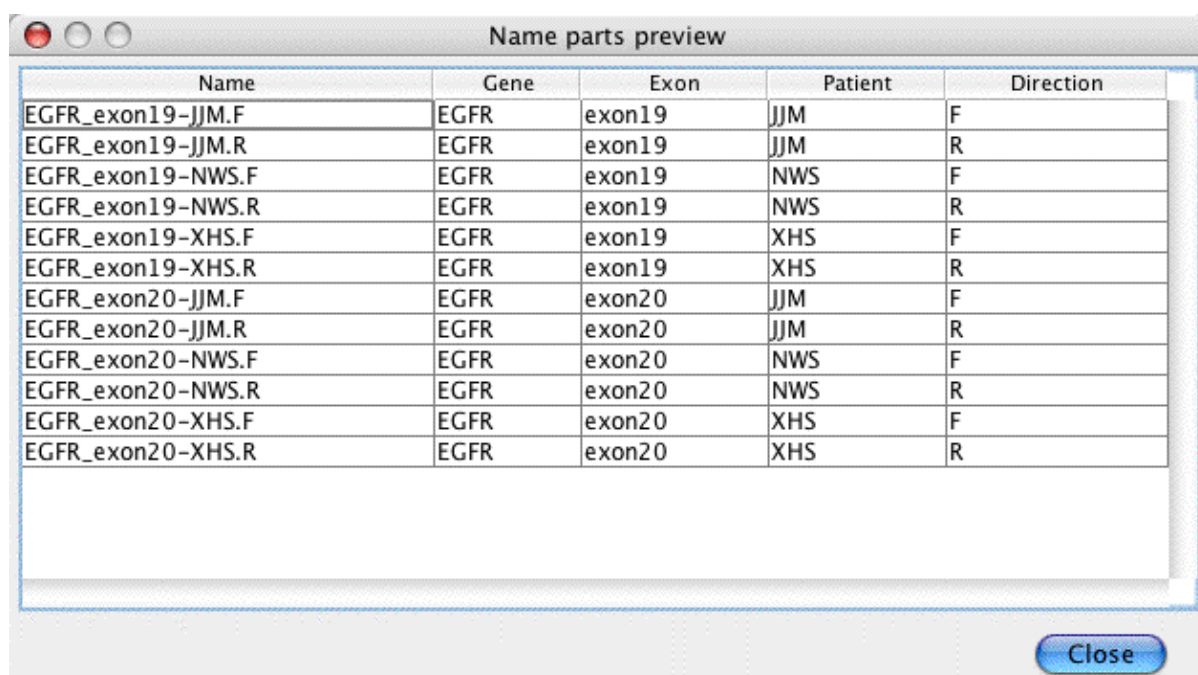


Name	Clone	Direction
EGFR_exon19-JJM.F	EGFR_exon19-JJM	F
EGFR_exon19-JJM.R	EGFR_exon19-JJM	R
EGFR_exon19-NWS.F	EGFR_exon19-NWS	F
EGFR_exon19-NWS.R	EGFR_exon19-NWS	R
EGFR_exon19-XHS.F	EGFR_exon19-XHS	F
EGFR_exon19-XHS.R	EGFR_exon19-XHS	R
EGFR_exon20-JJM.F	EGFR_exon20-JJM	F
EGFR_exon20-JJM.R	EGFR_exon20-JJM	R
EGFR_exon20-NWS.F	EGFR_exon20-NWS	F
EGFR_exon20-NWS.R	EGFR_exon20-NWS	R
EGFR_exon20-XHS.F	EGFR_exon20-XHS	F
EGFR_exon20-XHS.R	EGFR_exon20-XHS	R

The default name scheme is clearly not a good one for these clones. It seems that the name starts with the name of the gene (EGFR), followed by the name of the exon and a sample or patient identifier. If we change the name parts definition to look like this:



and then press "**Preview...**" again, we get:



Name	Gene	Exon	Patient	Direction
EGFR_exon19-JJM.F	EGFR	exon19	JJM	F
EGFR_exon19-JJM.R	EGFR	exon19	JJM	R
EGFR_exon19-NWS.F	EGFR	exon19	NWS	F
EGFR_exon19-NWS.R	EGFR	exon19	NWS	R
EGFR_exon19-XHS.F	EGFR	exon19	XHS	F
EGFR_exon19-XHS.R	EGFR	exon19	XHS	R
EGFR_exon20-JJM.F	EGFR	exon20	JJM	F
EGFR_exon20-JJM.R	EGFR	exon20	JJM	R
EGFR_exon20-NWS.F	EGFR	exon20	NWS	F
EGFR_exon20-NWS.R	EGFR	exon20	NWS	R
EGFR_exon20-XHS.F	EGFR	exon20	XHS	F
EGFR_exon20-XHS.R	EGFR	exon20	XHS	R

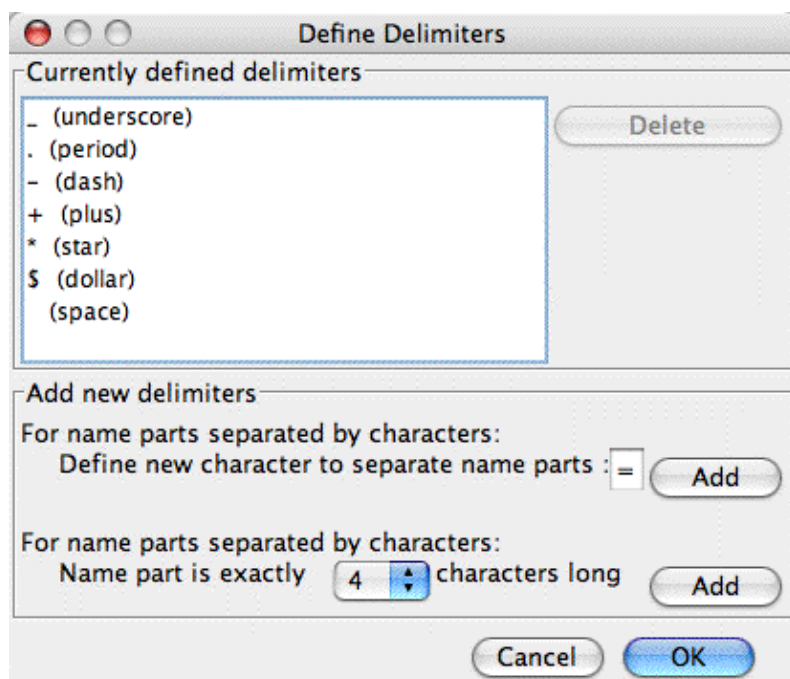
Currently, the only time where CodonCode Aligner uses the name part definition is when [assembling by group](#) (available through "Assemble with Options..." in the "Contig" menu). When assembling by groups, Aligner can use any of the name parts you define to groups sample together for forming contigs - Aligner will try to assemble only samples that belong to the same group. For additional information, please read the [assemble in groups](#) help.

Defining delimiters

CodonCode Aligner offers two different ways to parse sample names:

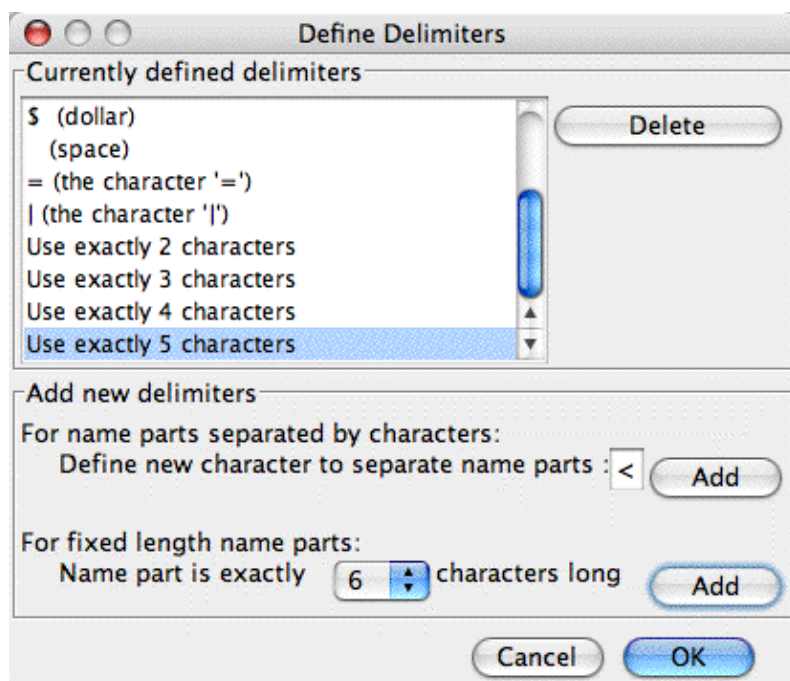
- Using delimiters to separate name parts (as shown in the example above)
- Using a fixed number of characters for a name part

Aligner has several pre-defined delimiter characters; if you want to use a different character as a delimiter, or if you want to use a fixed number of characters, click on the "**Define delimiters...**" button in the sample name preferences (or the "Sample Name Options" dialog from "Assemble With Options"). This will show the following dialog:



To add a new character delimiter, type the character in the text field after "Define new character to separate name parts", and then press the "Add" button next to it.

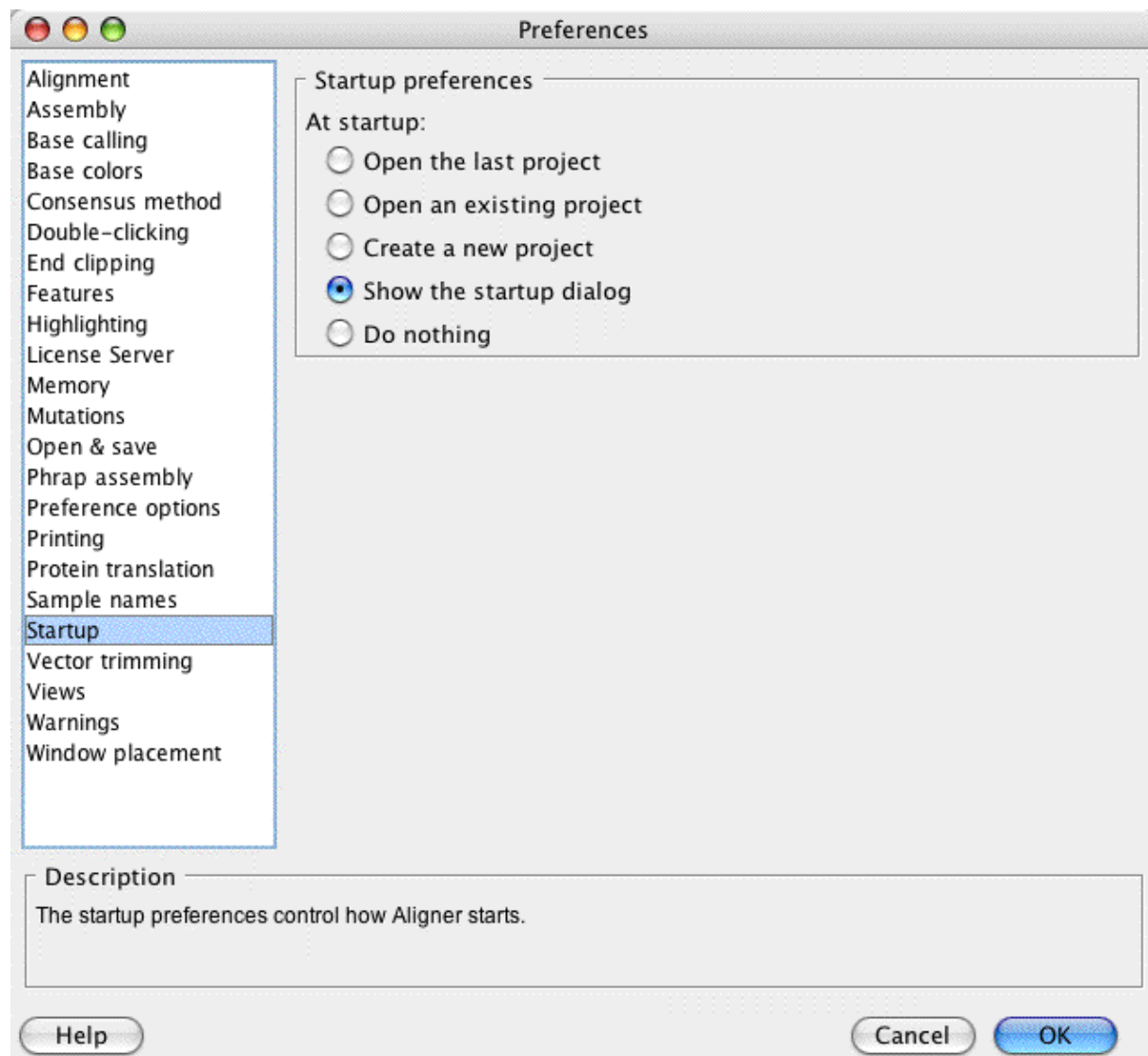
To use fixed length name parts, select the length of your name part in the pulldown menu near the bottom, and then click the "Add" button to the right of it. Repeat this for different lengths, as needed. The screen shot below shows an example of several custom defined delimiters:



After clicking "OK", you will be able to use the newly defined delimiters in the sample name preferences.

Startup Preferences

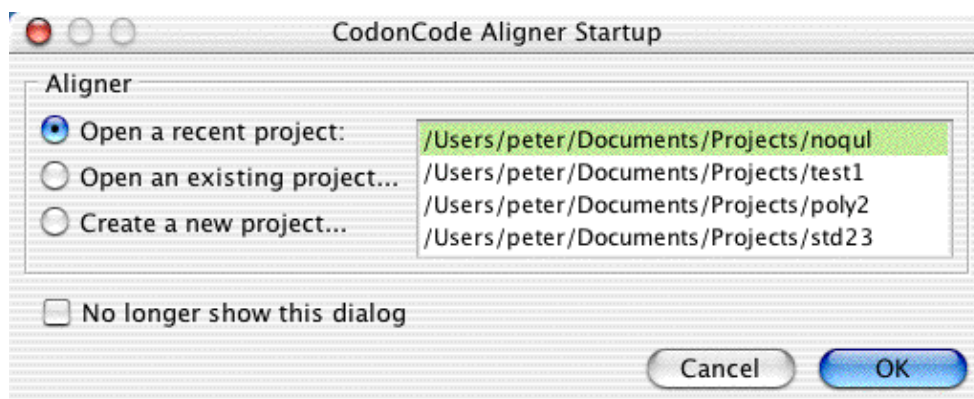
The "Startup Preferences" allow you to control what happens when you start CodonCode Aligner. Aligner can automatically open the last project you worked on, or present dialogs to open existing projects or create a new project, or show the "Startup Wizard" dialog. The following picture shows the startup preference dialog:



If you select "Do nothing", Aligner will start, but not open any project or show any dialogs. On Windows, you will see an empty Aligner window. On Mac OS X, the only thing you see will be a change in the menu bar; Aligner will not open any windows. You will need to select "Open Project...", "New Project...", or "Open Recent..." before you can do anything else (other than looking at the online help).

The Startup Dialog

If you choose "Show the startup dialog", the following window will be shown when Aligner starts:

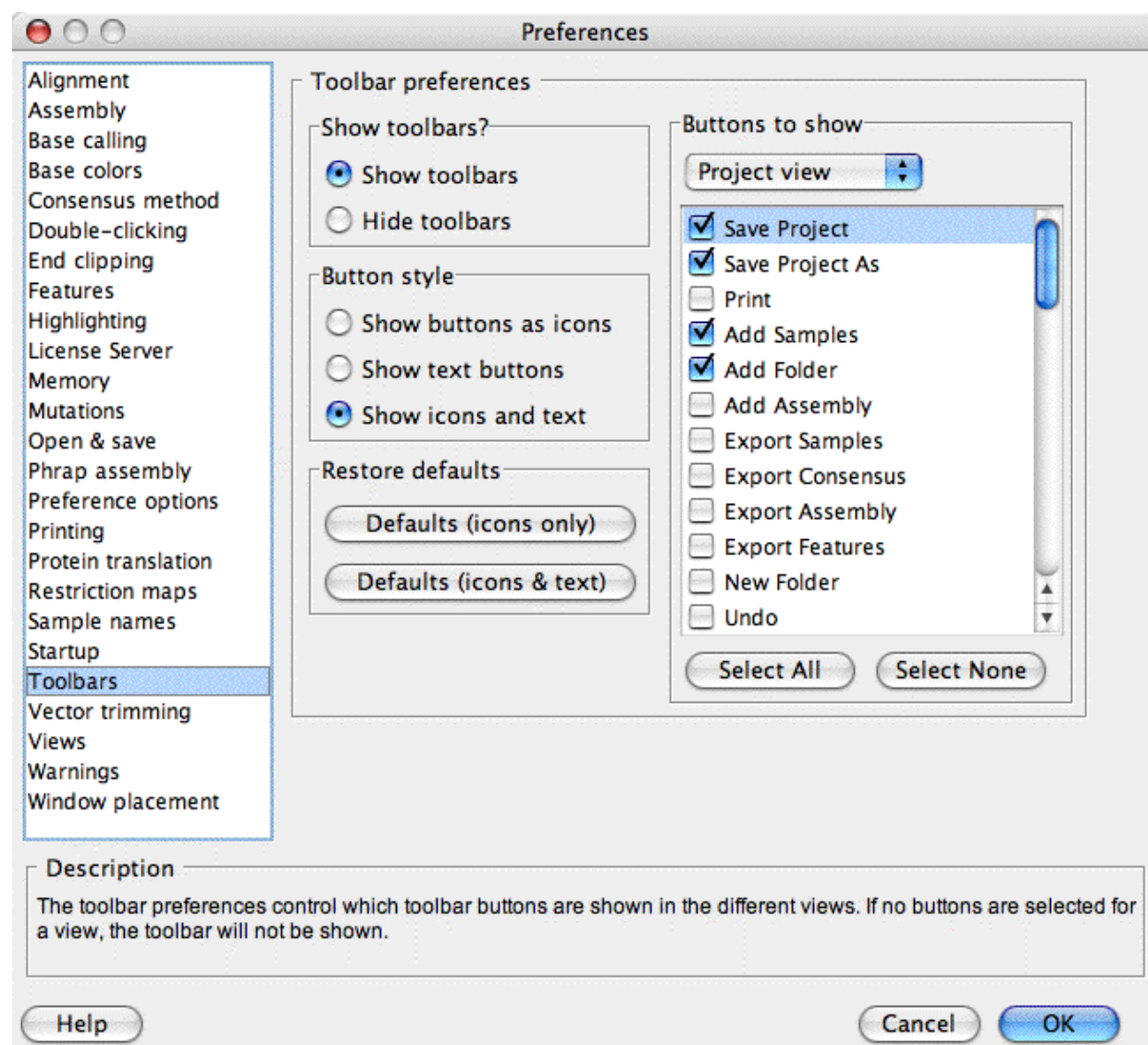


This allows you to quickly open one of the projects you worked on recently, or to open another project, or to create a new project.

***Note:** The first time you start CodonCode Aligner, the list of recent projects will be empty. The list will also be empty when you clear the recent project list by choosing "Clear Menu" in the "Open Recent" submenu in the "File" menu. The currently open project and any projects that are currently unavailable will be unavailable ("grayed out") in the "Open Recent" menu.*

Toolbar Preferences

The toolbar preferences allow you to customize toolbars for the different views in CodonCode Aligner:



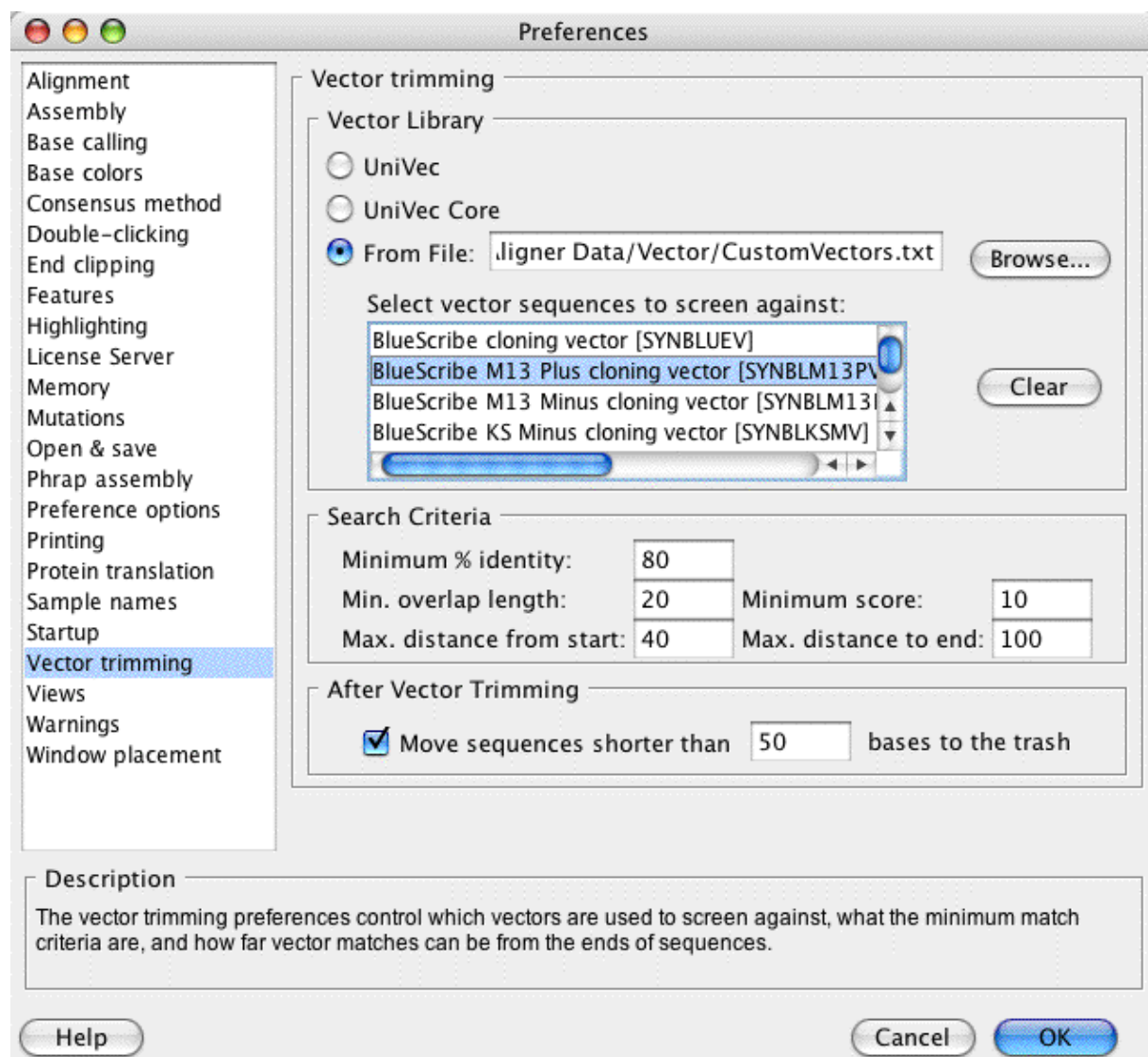
On the left side, you can choose whether or not to show toolbars for all windows, and if toolbars should be shown as buttons, text, or buttons and text. You can also restore the default settings by clicking on one of the "Defaults" buttons. The "Defaults (icons only)" button will restore the initial buttons settings on Windows, where buttons are shown as icons only. The "Defaults (icons & text)" button will restore the initial buttons settings on Mac OS X, where toolbar buttons are by convention shown as icons and text, and therefore need more space. However, you can use either button on both operating systems.

On the right side, you can select which buttons will be shown for the different views. Select the view to customize at the top, then check or uncheck the items you want to see or not see in the toolbar.

If you want to see toolbars in some views, but not in other views, you can simply use the "Select None" button to de-select all buttons for the views where you do not want to see the toolbars.

Vector Trimming Preferences

In the vector trimming preferences, you specify which vector sequences Aligner screens against, and what the minimum criteria for a vector match are:



To setup or change your vector screening parameters, follow these steps:

1. Choose the "vector library" by pressing one of the radio buttons on the top (the vector library is a collection of vector sequences in FASTA format). It's a good idea to **select your own vector sequence file** by pressing the "Browse..." button. Vector sequence files must be in FASTA format, and can contain one or many sequences.
2. Choose the vector sequences to screen against in the scroll pane below. You can select multiple vector sequences by pressing the shift-key while clicking on a vector name, or the command- or control-key for discontinuous selections. In general, you should select only one or a few vector sequences to screen against.
3. Set the search criteria for the stringency and sensitivity you desire. The settings shown in the image above will work fine for most projects.

4. If you want Aligner to automatically move sequences to the trash if they are too short after vector screening, check the check box at the bottom, and enter a minimum sequence length (for example 100).

When Aligner finds a match between a sample sequence and a vector sequence, it first looks at the minimum match criteria you defined. If the match is not good enough (for example because it is too short), then it will be ignored.

Two parameters that need explanation are the "**Max. distance from start**" and "**Max. distance from end**" numbers. To understand the meaning of these numbers, keep in mind that matches between sample sequences and vector sequences are "local" matches - that is, the matches do not necessarily extend to the beginning or the end of the sample. If there is low-quality sequence at the beginning or end of your samples, errors in these parts will often lead to alignments that start a few bases into the sequence, and/or do not extend all the way to the end. Aligner uses the "Max. distance" parameters to determine how far from the start or end of a sample a match can be. For example, if a match starts 35 bases from the start of the sample, and the maximum distance from the start is set to 50, then Aligner will trim the first unaligned 35 bases, plus the bases that actually match. But if the maximum distance from the start was set to less than 35, Aligner would ignore this match, and not trim.

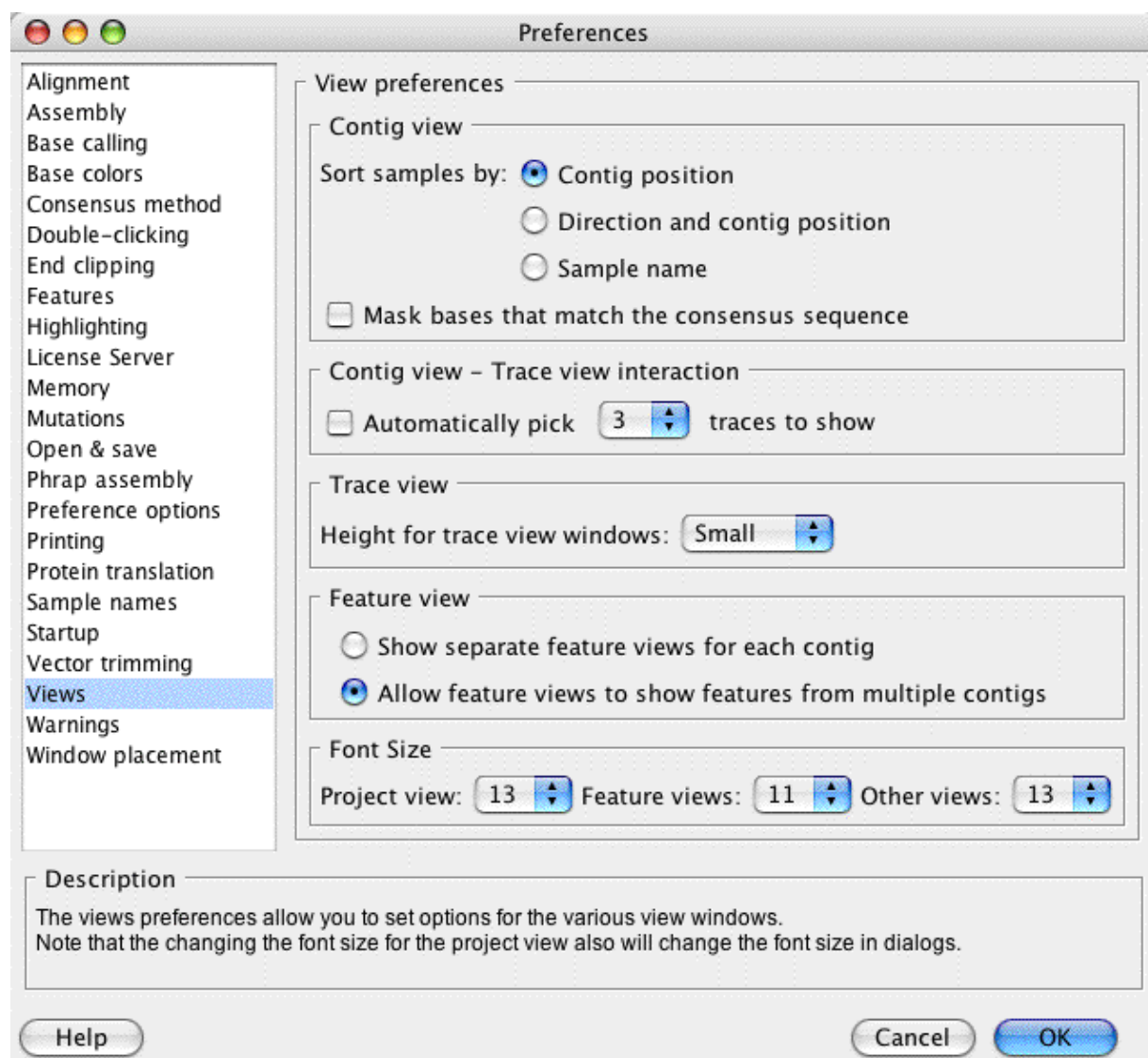
If you use end clipping before vector trimming, you can use low numbers for the "max. distance" parameters, for example 20 and 50. With samples that are not end clipped first, higher numbers (like 50 for the start and 150 - 250 for the end) may make more sense.

The alignment score is calculated based on the number of matches, discrepancies, and insertions/ deletions in a match. Each matching bases gives a score of +1, while mismatches and insertions/deletions reduce the score. Therefore, your minimum score should be lower than the minimum overlap length.

Keep in mind that there is always a trade-off between sensitivity and specificity. If you use very stringent match criteria, you are likely to miss some vector matches; if you use very loose criteria, you may end up trimming sequences from random hits.

View Preferences

The view preferences allow you to set options for the different views:



In the "**Contig view**" panel at the top, you choose how samples should be sorted in contig views. For most projects, the default of sorting samples **by position** in contig makes the most sense. Samples will be sorted in ascending order, based on their first aligned base in a contig.

Sometimes, sorting the samples **by name** is more desirable. One example is a mutation detection project where you forward- and reverse reads have similar names (like "Sample123.f" for forward and "Sample123.r" for reverse), and where you would like to have the forward- and reverse-reads right underneath each other. If you select "By direction and contig position", all reads in forward direction will first be sorted by position, and then, all reads in reverse direction will be sorted.

Changing these preferences here will **not** affect any open contig view windows; it will also **not** affect contigs where you previously sorted samples manually by dragging read names up or down. To change the sorting in open [contig view windows](#) (or to quickly try out the different sort options), open a contig view, and right-click (command-click on OS X) in the "aligned bases" panel to display the contig view [popup](#) menu, then choose

"Sort samples by..."

In the "**Contig view-Trace View interaction**" panel, you can choose what happens when you have the contig view and the trace view for a contig open. If the check box before "Automatically pick 3 traces to show" is checked, Aligner will try to pick 3 traces to show everytime you move around in a contig. Note that Aligner will **not** automatically open the trace view for you, however. You can choose how many traces Aligner should pick - we suggest to use smaller numbers for faster performance.

In the "**Trace view**" panel, you can set the default height of traces in trace view windows - try it out!

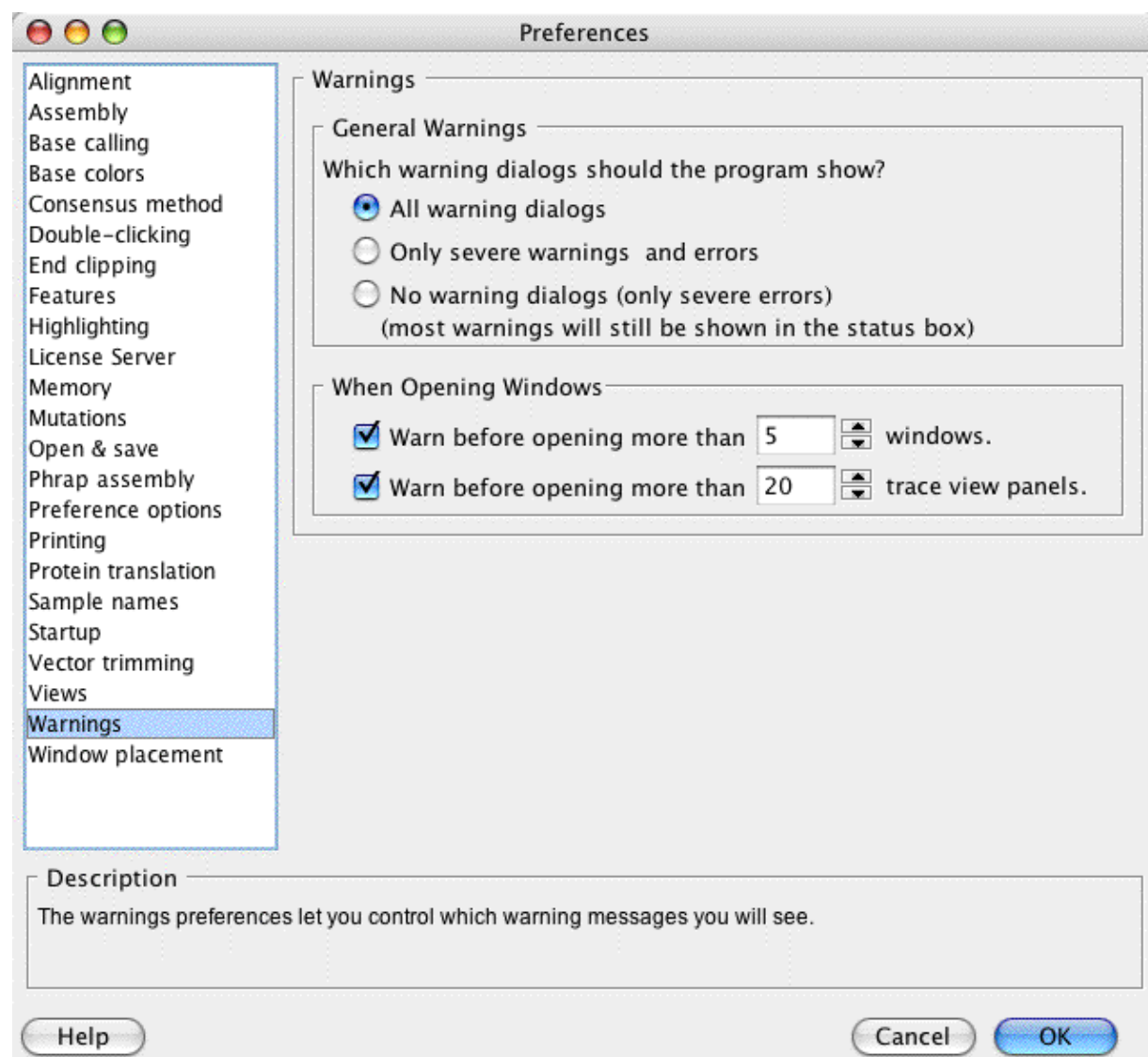
In the "**Feature view**" panel, you can determine how feature views will behave. Since Aligner version 1.1.2, feature views can show the features for more than one contig. For example, you may be interested in viewing all PolyPhred tags in all of your contigs in a single table. You can do this by selecting all contigs, and then selecting "Feature View" in the "View" menu. If you have "Allow feature views to show features from multiple contigs" checked, you will get just one window. But if you have "Show separate feature views for each contig" checked, you will get an individual window for each contig (you guessed that much, didn't you :-)

There's just one thing to notice when you allow feature views to show features for multiple contigs: any feature will be shown in only one window. This means that Aligner may sometimes close older feature views. If, for example, you have a feature view for "Contig1" and "Contig2" open, and then open a feature view for "Contig2" and "Contig3", Aligner will first close the older feature view, and then show you the new feature view for "Contig2" and "Contig3".

In the "**Font size**" panel, you can set the font size used in the different views. The project view settings affect only the project view; the "feature views" settings affect feature views and mutation report windows; and the "other views" setting affects the trace views, base views, contig views, and other views.

Warning Preferences

You can control how many warnings CodonCode Aligner shows in the Warning Preference dialog:



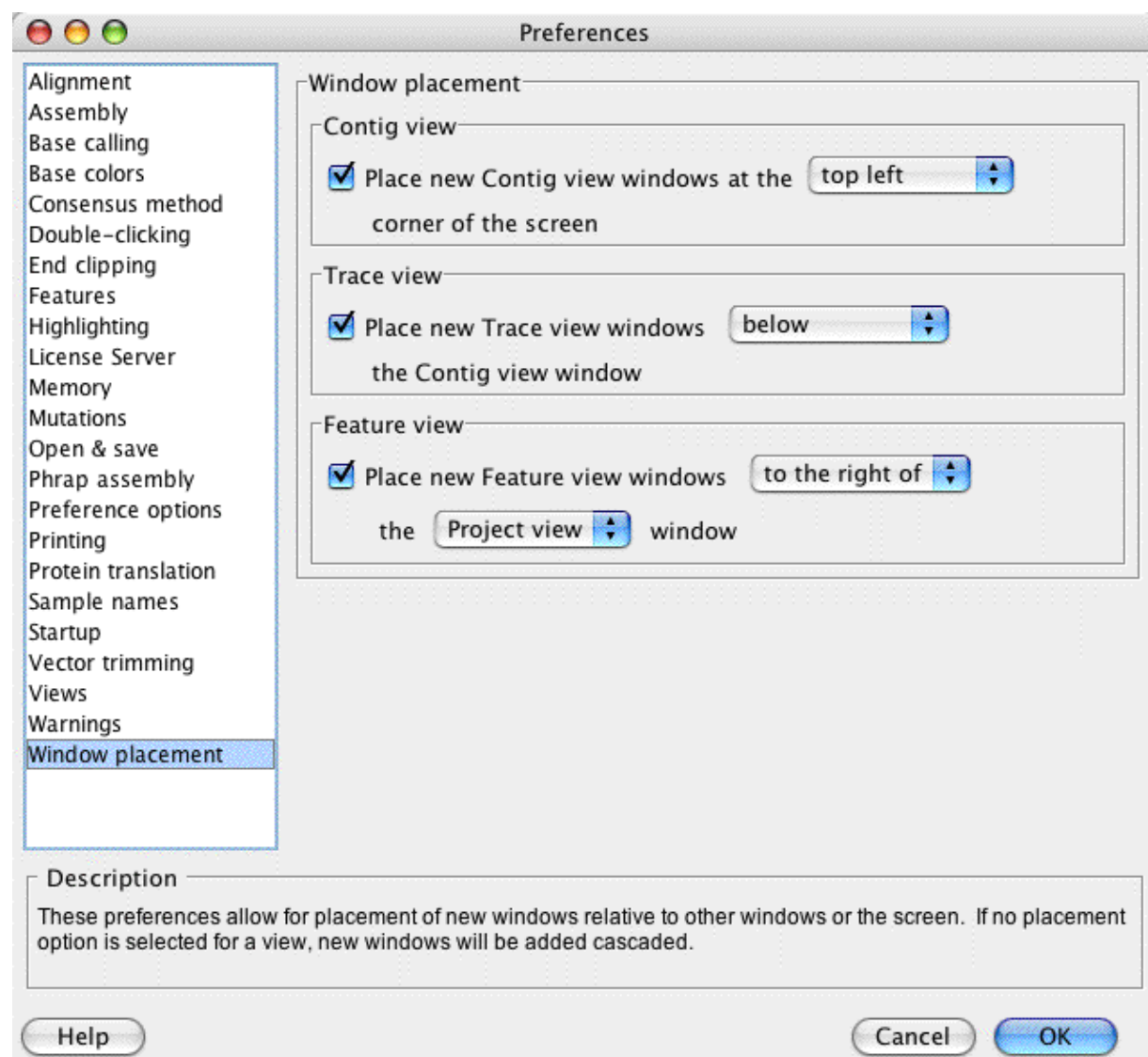
The upper section affects most warnings that Aligner can show. For new users, we suggest that you display all warning dialogs, as shown above. Once you are familiar with Aligner, you can choose to see only the more severe error messages and warning dialogs by selecting the checkbox in the second row. If you really know what you are doing and do not want to see any warning dialogs, select the checkbox in the third row.

In the lower section, you can choose if Aligner should warn you before opening many windows (for example because you accidentally double-clicked on the "Unassembled Reads" folder, which contains many samples). You can also set the threshold for how many windows can be opened before Aligner shows a warning. *Please note that the warning dialog will not be shown if you turned off all warnings in the "Warnings" preferences.*

Some warning dialogs also allow you to turn off the optional warnings, which has the same effect as selecting "Only severe warnings and errors". *Some dialogs can be turned off individually, while others affect all optional warnings.*

Window Placement Preferences

You can control how windows are arranged in CodonCode Aligner in the window placement preferences:



Using the check boxes and drop-down menus, you can select where Aligner places contig views, trace views, and feature views when opening new windows.

For contig views, you can select any of the four corners of the screen (or the "root" window on Windows).

For trace views, you can select where to place trace views relative to the contig view; common choices are to put trace views below or to the right of the corresponding contig view.

For feature views, you can select where to put newly opened feature view windows relative to the project view or relative to the corresponding contig view.

If the check boxes are unchecked, or if a window used for relative placements are not shown, Aligner will add windows cascaded, so that each new window will be a bit lower and to the left of the window that was opened

last.

CodonCode Aligner Help by Menu

[File](#) - [Edit](#) - [Go](#) - [Sample](#) - [Contig](#) - [View](#) - [Window](#) - [Help](#)

Aligner Menu (on Mac OS X only)

[About Aligner..](#)

[Preferences...](#)

[Quit](#)

File Menu

[New Project..](#)

[Open.. \(a sample\)](#)

[Open... \(a project...](#)

[Open Recent](#)

[Close Project](#)

[Save Project](#)

[Save Project As...](#)

[Import](#)

[Add Samples](#)

[Add Folder](#)

[Add Assembly](#)

[Export](#)

[Export Project Summary...](#)

[Export Samples...](#)

[Export Consensus Sequences...](#)

[Export Assembly...](#)

[Export Features...](#)

[New Folder...](#)

[Delete Folder...](#)

[Duplicate Samples](#)

[Exit \(on Windows\)](#)

Edit Menu

[Undo](#)

[Redo](#)

[Copy \(selected sequence\)](#)

[Paste](#)

[Select from Start to Here](#)

[Select from Here to End](#)

[Select All](#)

[Change Bases](#)

[Reverse complement](#)

[Move to Unassembled Samples](#)

[Move to Trash](#)

[Preferences](#) (Windows only)

Go Menu

[Define Features...](#)

[Next Feature](#)

Next

[High Quality Mismatch](#)

[Low Quality Consensus](#)

[Ambiguity](#)

[Mismatch](#)

[Edited Base](#)

[Previous Feature](#)

Previous

[High Quality Mismatch](#)

[Low Quality Consensus](#)

[Ambiguity](#)

[Mismatch](#)

[Edited Base](#)

[Base Number...](#)

[First Aligned Base](#)

[Last Aligned Base](#)

[Search Sequence...](#)

[Search Again](#)

[BLAST Search](#)

[MegaBLAST](#)

[Nucleotide \(blastn\)](#)

[Translated \(blastx\)](#)

[Translated \(tblastx\)](#)

Sample Menu

[Call Bases](#)

[Find heterozygous indels](#)

[Split heterozygous indels](#)

[Clip Ends...](#)

[Trim Vector...](#)

[Insert gap](#)

[Shift Bases Left](#)

[Shift Bases Right](#)

[Delete](#)

[Selection - Fill from Left](#)

[Selection - Fill from Right](#)

[From Sample Start](#)

[To Sample End](#)

[Move](#)

[Gap Left](#)

[Gap Right](#)

[Sequence Left](#)

[Sequence Right](#)

[Mark](#)

[Start Alignment Location](#)

[End Alignment Location](#)
[Tag](#)
[Show Local Tags...](#)
[Show All Tags...](#)
[Add Tags...](#)
[Mark ... as False Positive](#)
[Make Reference Sequence](#)
[Sample Information...](#)

Contig Menu

[Assemble](#)
[Assemble with Options...](#)
[Align to Reference Sequence](#)
[Align with Options...](#)
[Unassemble](#)
[Find Mutations](#)
[Process heterozygous indels](#)
[Delete](#)
[From Contig Start](#)
[To Contig End](#)
[Split Contig](#)
[Contig Information...](#)

View Menu

[Contig](#)
[Traces](#)
[Bases](#)
[Qualities](#)
[Features](#)
[Restriction Map](#)
[Select Enzymes...](#)
[Restriction Map Options...](#)
[Mask Bases Matching Consensus](#)
[Switch Color Scheme](#)
[Auto Select Traces](#)
[Protein Translation](#)
[None](#)
[Frame 1](#)
[Frame 2](#)
[Frame 3](#)
[Frame Forward 3 Frames](#)
[All 6 Frames](#)
[Preferences...](#)
[Toolbars](#)
[Show Toolbars](#)
[Hide Toolbars](#)

Window Menu

[Close](#)

Help Menu

[Aligner Help...](#)

[Quick Tour](#)

[License...](#)

[Aligner Web Site](#)

[Check for updates...](#)

Memory Requirements

CodonCode Aligner is currently intended for projects with up to several hundred reads. On computers with sufficient RAM (512 MB or more), CodonCode Aligner can handle projects with up to several thousand reads (each about 500-1000 bases long).

The size of projects that CodonCode Aligner can handle is determined by the amount of memory available to Aligner. Details depend on the operation system used:

- On **Windows**, CodonCode Aligner will try to use up to 50% of the build-in memory (RAM), or 384 MB, whichever is smaller).
- On **Mac OS X**, CodonCode Aligner the maximum memory available to CodonCode Aligner is limited to a default of 256 MB. This is sufficient for projects with several hundred reads. You can **increase** the memory available to Aligner using the [memory preferences](#), and may need to do so if working with large projects.

Please note that the memory available to Aligner should not exceed the amount of installed memory (RAM) on your computer; if you try to assign more memory, CodonCode Aligner may not start, or behave erratically. In addition, the memory assigned to Aligner should not exceed about 1200 - 1400 MB, even on computers with several GB of RAM.

If you are working on a computer with a limited amount of RAM (for example 256 MB), you may get better performance if you by quitting other open applications, and by reducing the amount of memory available to CodonCode Aligner.

You can always check how much memory CodonCode Aligner is currently using in the [memory preferences](#).

How Aligner memory on Windows is set

On Windows, the maximum memory is set in the file "CodonCode Aligner.ini" in the CodonCode Aligner folder. If you look at the file in a text editor like NotePad, you can find a line that looks like this:

```
Virtual Machine Parameters=-Xms32M -Xmx50:64:384P
```

The numbers after -Xmx sets the maximum memory available to Aligner. The first number (50) limits the memory to 50% of the built-in physical memory (RAM). The next two numbers set further limits: memory available to Aligner can never be less than 64 MB, and never more than 384 MB, regardless of the amount of built-in RAM.

You can change these numbers with a text editor like NotePad. For example, to increase the maximum memory to 75% of the built-in RAM, or 512 MB (whichever is smaller), change the line to read:

```
Virtual Machine Parameters=-Xms32M -Xmx75:64:512P
```

Please note that CodonCode Aligner will not start if you increase these numbers too much. Unfortunately, what is too much depends not only on the built-in RAM, but also on other factors that influence the amount of available virtual memory (for example, which other programs are running, your virtual memory settings, and sometimes even how much space is available on your hard drive). However, the likelihood of startup problems increases if you increase the first (percentage) number to more than 75, and the last (max MB) number to more than 1000 (increasing it to more than 1400 will cause Aligner startup

problems on most computers).

If you experience problems where CodonCode Aligner does not start, you should change the line to read:

```
Virtual Machine Parameters=-Xms32M -Xmx50:64:256P
```

After saving the file, try starting CodonCode Aligner again.

How Aligner memory on Mac OS X is set

On OS X, the maximum memory is set in the file "Info.plist", which is inside the CodonCode Aligner package. When you change the memory settings for CodonCode Aligner in the [memory preferences](#), Aligner will edit this file, and then restart itself so that the new memory settings take effect.

Please always use the [memory preferences](#) to change the memory available to Aligner on OS X. Please do **not** change the memory available to Aligner by editing the "Info.plist" file directly - if you make any mistakes, Aligner may not be able to start, or may behave erratically.

CodonCode Aligner Release Notes

The release notes for the current version of CodonCode Aligner can be found in a separate file, called "ReleaseNotes.txt", in the Aligner directory.

Checking for Aligner Updates

You can check if your version of CodonCode Aligner is current by selection "**Check for Updates**" in the "**Help**" menu. Aligner will then check CodonCode's web site to see if your version is current; therefore, this requires that you have an active internet connection. If you are not currently connected to the internet, you will see an error message "Unable to verify the current Aligner version". The same message can also result from other problems, for example if CodonCode's web site is down (which hopefully will be rare :-). If everything goes

When you start Aligner, Aligner also automatically tries to check for updates. Aligner will let you know if a newer version is available, or if there are any problems during the update check. If your version is current (which should be the case most of the time you start Aligner), you will not be notified.

Updates may be free or charge, or they may require the payment of additional license fees, depending on when you purchased your Aligner license, and the terms of your license.

Visiting the Aligner Web Site

You can visit the Aligner web site by selecting "**Aligner Web Site**" from the "**Help**" menu. This will open a browser window and point it to the CodonCode Aligner web site - the starting point for downloading updates, requesting trial licenses, and the latest Aligner news. The address of the Aligner web site is <http://www.codoncode.com/aligner/>.