

SANGER DATA ON LOBLOLLY

Table of Contents

Table of Contents	1
Introduction	2
PART 1: PHRED-PHRAP PROCESSING PIPELINE	2
Overview	2
Step 1: Project Directory Initialization	2
Step 2: phred	4
Step 3: phd2fasta	5
Step 4: cross_match	6
Step 5: AmpMatch (checkAmpPrimer_MaskDownstream.pl)	6
Step 6: Phrap	7
PART 2: ALIGNMENT PIPELINE	9
Overview	9
Step 1: Contig Set Selection	9
Step 2: Contig Alignment	9
PART 3: MOST COMMON HAPLOTYPE (MCH) PIPELINE	10
Overview	10
ADEPT2 MCH Method	10
Overview	10
Step 1: MCH Contig Selection	10
John Liechty's Notes	11
Poplar MCH Method	11
Overview	11
Step 1: MCH Contig Selection	11
MCH Consensus Sequence Generation	12
Notes	12
PART 4: SITE ANNOTATION PIPELINE	12
Overview	12
Step 1: Site Annotations File Format	13
PART 5: SNP ANNOTATION PIPELINE	13
Overview	13
Step 1: SNP Annotations File Format	13
PART 6: GENOTYPING PIPELINE	Error! Bookmark not defined.
Overview	Error! Bookmark not defined.
APPENDIX	Error! Bookmark not defined.
Section Title	Error! Bookmark not defined.

Introduction

The Neale Lab hosts its Sanger sequence data on the loblolly.ucdavis.edu server. There are three types of sequence stored on Loblolly: Read data (a.k.a. chromatogram data), Assembly data, and SNP data. Each of these types of data located in project directories found under the /Data/ directory tree.

The purpose of this document is to (a) help explain how the Neale Lab processes Sanger data, and (b) explain where each type of data is located. This document is intended to serve as a guide and reference for developers seeking to mine information from the Sanger data set housed on Loblolly.

This document is divided into six parts. The first five parts describes how the Neale Lab processes Sanger data into assemblies, alignments, SNPs, genotypes, and other forms of output data. The sixth part discusses how key pieces information can be extracted from the various output files generated in parts one through five.

PART 1: PHRED-PHRAP PROCESSING PIPELINE

Overview

The Neale Lab uses a custom version of phredPhrap for processing Sanger sequence data into assemblies. Both the original phredPhrap script and the Neale Lab's custom phredPhrap script are located in the paths shown below.

Original script: /usr/local/genome/bin/phredPhrap
Custom script: /usr/local/genome/bin/phredPhrap.NealeLab.20100510

Our custom version of the phredPhrap script was developed to allow phredPhrap to perform additional sequence masking, and produce more accurate results.

The phredPhrap execution process can roughly be broken down into six steps. Each step will be detailed in the sections that follow.

Step 1: Project Directory Initialization

All Sanger sequence data stored on loblolly is stored in the form of AB1 Files (a.k.a. Read Files or Chromatogram Files). Complementary pairs of Read Files (Forward and Reverse Read Files) are stored in directories referred to as "PhredPhrap Directories". The general structure of a PhredPhrap Directory is shown below.

General Format:

- PhredPhrapDirectory/
- PhredPhrapDirectory/chromat_dir/
- PhredPhrapDirectory/edit_dir/
- PhredPhrapDirectory/phd_dir/
- PhredPhrapDirectory/poly_dir/

Typical Example:

```
/Data/populus/4CL1_10/Pong/read_pairs/4CL1_10-12373_Pong/  
/Data/populus/4CL1_10/Pong/read_pairs/4CL1_10-12373_Pong/chromat_dir/  
/Data/populus/4CL1_10/Pong/read_pairs/4CL1_10-12373_Pong/edit_dir/  
/Data/populus/4CL1_10/Pong/read_pairs/4CL1_10-12373_Pong/phd_dir/  
/Data/populus/4CL1_10/Pong/read_pairs/4CL1_10-12373_Pong/poly_dir/
```

Initially, the only files that exist under a PhredPhrap Directory are the AB1 Files found in the chromat_dir subdirectory. These files may be actual AB1 Files, or they may be links to actual AB1 Files. In any given chromats_dir directory, there may be zero, one, or two Read Files (never more).

Read Files (as well as many other files generated by the phred/phrap pipeline) are named using a special prefix called the Read Pair Name. The Read Pair Name is composed of an Amplicon Name and an ID (separated by a dash character "-"). The ID can represent a number of different things, however it is most often representative of the Sample Name, the Extraction ID, or the Well ID.

Generally, PhredPhrap Directories associated with a populous project will have a Sample ID, ACESAP project will have a Well ID (a.k.a. Plate Sample ID), all other projects will use an Extraction ID. However, this is NOT always the case! The ID field may represent some other value that may not seem like it relates to anything.

Populus Read Pair Name: [Amplicon]-[SampleID]
ACESAP Read Pair Name: [Amplicon]-[WellID]
General Read Pair Name: [Amplicon]-[ExtractionID]

Example Populus Read Pair Name: /chromat_dir/4CL1_10-12214_Pong-F.ab1
Example ACESAP Read Pair Name: /chromat_dir/0_14853_01-9016555_Lade-F.ab1
Example General Read Pair Name: /chromat_dir/0_1143_01-4667_Pt-F.ab1

All PhredPhrap Directories are stored under one of the two Project Directories shown below.

Pinus Project Directory: /Data/amplicons
Poplar Project Directory: /Data/populous

The Pinus Project Directory encompasses data generated from the ADEPT2, WHISP, CRSP/CRSP2, ACESAP, and CRIEC projects. The Poplar Project Directory encompasses data generated from the PBGP project.

Project Directories are originally created by one or more scripts know as Project Directory Initialization Scripts. These scripts create the PhredPhrap Directories, place AB1 Files (or links to AB1 Files) into the chromat_dir subdirectory, and generate an Info File that is placed at the root of the PhredPhrap Directory. Info Files are named using the format shown below.

Info File Format: [ReadPairName]_[SpeciesCode]_info.txt
Example: Cesa1A_12-12638_Pong_info.txt

Info Files are generated in order to pass necessary data to the Neale Lab's custom phredPhrap script. The Info File contains data formatted as name/value pairs. Below are two listings of Names that may be found in an Info File of a Project Directory.

Names found in the /Data/populous Project Directory:

READ_PAIR - The Read Pair Name
ORIGINAL_F_PATH - The full path to the Forward Read File
ORIGINAL_R_PATH - The full path to the Reverse Read File

Names found in the /Data/amplicons Project Directory:

ORIGINAL_F_PATH - The full path to the Forward Read File
ORIGINAL_R_PATH - The full path to the Reverse Read File
ORIGINAL_SAMPLE_ID - [ID]_[SpeciesCode]
PINESAP_F_PATH - IGNORE THIS DATA (obsolete)
PINESAP_R_PATH - IGNORE THIS DATA (obsolete)
PLATE_NAME - Database field: sample_seqplates.plate_name
PROJECT - Project (possible values: ACESAP, ADEPT2, CRSP, CRSP2, WHISP)
READ_PAIR - The Read Pair Name
SAMPLE_ID - Format: [ID]_[SpeciesCode]
TREE_EXTRACTION_ID - Database field: sample_dnaextractions.tree_extraction_id
TREE_ID - Database field: sample_treesamples.tree_identifier
WELL_NUMBER - Database field: sample_seqplates.well_number

Step 2: phred

The second step of our custom phredPhrap pipeline is the phred run. Phred is responsible for reading the AB1 Files and calling nucleotide bases. Phred also produces quality scores for each base call in order to give a measure of confidence for each call.

During this step, phred is run with additional flags to generate the Poly files found in the /poly_dir directory. These files are normally used by polyphred, however we use them to determine secondary noise.

When the phred run begins, it reads each AB1 File found in the chomats_dir directory one by one, and produces a PHD File. These files are named using the format shown below.

PHD File: /phd_dir/[ReadPairName]_[SpeciesCode]-[ReadDir].ab1.phd.[#]
Example: /phd_dir/4CL1_01-G_KR5_Pofr-F.ab1.phd.1

These files contain output similar to the output shown below:

```
BEGIN_SEQUENCE 4CL1_01-P9_Poal-R.ab1  
  
BEGIN_COMMENT  
  
CHROMAT_FILE: 4CL1_01-P9_Poal-R.ab1  
ABI_THUMBPRINT: 0
```

```
TRACE_PROCESSOR_VERSION: KB 1.3.0fc2
BASECALLER_VERSION: phred 0.071220.b
PHRED_VERSION: 0.071220.b
CALL_METHOD: phred
QUALITY_LEVELS: 99
TIME: Wed Dec 15 18:31:06 2010
TRACE_ARRAY_MIN_INDEX: 0
TRACE_ARRAY_MAX_INDEX: 9879
TRIM: 0 52 -1.00
TRACE_PEAK_AREA_RATIO: 0.1069
CHEM: term
DYE: big
```

```
END_COMMENT
```

```
BEGIN_DNA
a 26 279
c 29 291
...
c 21 916
END_DNA
```

```
END_SEQUENCE
```

The values found between the BEGIN_DNA and END_DNA delimiters give the base call, phred quality score, and position in the gel or column respectively. If the PHD File does not contain any values between the BEGIN_DNA and END_DNA delimiters, then Phred was unable to process that file.

In some cases there may be extra PHD Files found in the phd_dir directory. These files appear when the assembly is manually edited and saved in consed. These files will be named similarly to another PHD File in the directory, except that it will have a higher number at the end of the file extension. When these files are encountered, the highest numbered PHD File should always be used over a lower numbered file. See the example below.

Example:

```
ls -l /Data/populus/4CL3_21/Pofr/read_pairs/4CL3_21-B_3_Pofr/phd_dir/
total 24
-r--r--r-- 1 jdl  staff 3400 Jan 15  2011 4CL3_21-B_3_Pofr-F.ab1.phd.1
-r--r--r-- 1 jdl  staff 3623 Jan 15  2011 4CL3_21-B_3_Pofr-R.ab1.phd.1
-r--r--r-- 1 jdl  staff 3704 Jan 15  2011 4CL3_21-B_3_Pofr-R.ab1.phd.2
```

IMPORTANT NOTE: Read Pairs that do not have both a Forward and Reverse Read File are not processed beyond this step!

Step 3: phd2fasta

The third step of our custom phredPhrap pipeline is the phd2fasta run. The phd2fasta program takes both the Forward Read PHD File and the Reverse Read PHD File (or one if only one exists), and generates a single FASTA File and a single Quality Scores File, and places it in the edit_dir directory. These files contain the raw

FASTA base call data. These files will be referred to as the Raw FASTA File and the Raw Quality Scores File.

It is important to note that you should NEVER rely on the Raw FASTA Files for sequence information! The masking process that occurs in the next two steps does NOT update these files. Therefore, this data should NOT be used for extracting information about the read sequences. (For that information, refer to the ACE Files discussed in the upcoming sections.)

The filenames for Raw FASTA Files and Raw Quality Score Files are formatted as shown below.

Raw FASTA File: /edit_dir/[ReadPairName]_[SpeciesCode].fasta
Example: /edit_dir/4CL1_01-B_26_Pofr.fasta

Raw Quality Scores File: /edit_dir/[ReadPairName]_[SpeciesCode].fasta.qual
Example: /edit_dir/4CL1_01-B_26_Pofr.fasta.qual

Even if all of the Read Files failed the phred run, the phd2fasta program will still generate an empty FASTA and Quality Scores File in the edit_dir directory.

Step 4: cross_match

The fourth step of our custom phredPhrap pipeline is the cross_match run. The cross_match program reads the FASTA and Quality Score files produced during the phd2fasta run, searches for vector sequence in the read sequences, and masks out any vector sequence that it detects (by substituting the base value with an 'x' character).

The cross_match program places its output in a Screen File. The filename format of this file is shown below.

Screen File: /edit_dir/[ReadPairName]_[SpeciesCode].fasta.screen
Example: /edit_dir/4CL1_10-12373_Pong.fasta.screen

The cross_match program also moves the Quality Scores File generated in the previous step from *.fasta.qual to *.fasta.screen.qual. No changes are made to this file, it is simply moved.

Step 5: AmpMatch (checkAmpPrimer_MaskDownstream.pl)

The fifth step of our custom phredPhrap pipeline is known as the AmpMatch run. The script that runs during this step is checkAmpPrimer_MaskDownstream.pl, and is located in the /usr/local/genome/bin/ directory. This program reads the Screen File created by cross_match, and performs further masking of the sequences by detecting the amplification primer sequence and masking downstream.

The AmpMatch program makes a backup of the Screen File generated by cross_match by copying the file and appending the filename with a '.original' file

extension. It then overwrites the old Screen File created by `cross_match`, and replaces it with its own Screen File (with the same filename as the previous Screen File).

Backup Screen File: `/edit_dir/[ReadPairName]_[SpeciesCode].fasta.screen.original`
Example: `/edit_dir/4CL1_10-12373_Pong.fasta.screen.original`

The `AmpMatch` program also concatenates data to the end of the PHD Files in the `phd_dir` directory. It adds information about the match for amplification primer so that data can be viewed in `consed`. It also creates an `AmpMatch Output File` that can be parsed when running a script that is collecting data from the data set. The `AmpMatch Output File` is formatted as shown below.

`AmpMatch Output File: /edit_dir/[ReadPairName]_[SpeciesCode].ampmatch`
Example: `/edit_dir/4CL1_10-12373_Pong.ampmatch`

The `AmpMatch` run also produces the `AmpMatch Log File` with the format shown below.

`AmpMatch Log File:`
`/edit_dir/[ReadPairName]_[SpeciesCode].checkAmpPrimer_MaskDownstream.log`
Example: `/edit_dir/4CL1_10-12373_Pong.checkAmpPrimer_MaskDownstream.log`

Step 6: Phrap

The sixth step of our custom `phredPhrap` pipeline is the `phrap` run. The `phrap` program reads in the Screen File and the PHD Files, and generates an ACE File.

The filename for the ACE File is shown below.

ACE File: `/edit_dir/[Amplicon]-[ID]_[SpeciesCode].fasta.screen.ace.[#]`
Example: `/edit_dir/4CL1_10-12373_Pong.fasta.screen.ace.1`

ACE Files may be updated several times, in which case the last file extension number is incremented. Thus, the highest numbered ACE File should always be used over lower numbered ACE Files.

ACE Files uses two letter codes to indicate the beginning of a segment of data. The following ACE File codes are of particular interest to a data mining effort. (Note: see John Liechty's `consed` manual, pages 86 – 95.)

Code: AS
Description: Indicates the number of Contig and Read sequences that exist in the ACE file.
NOTE: For the Lobolly Sanger data, there should only be 1 contig and a maximum of 2 reads. Any more or less indicates that something has gone wrong!
Parameter 1: <# of contigs>
Parameter 2: <# of read files>
Example: AS 1 2

Code: CO
NOTE: This sequence contains asterisk characters (*), which represent gaps. You may want to parse these out!
Ignore the complement parameter!
Description: Indicates the beginning of a Contig sequence block.
Parameter 1: <contig name>
Parameter 2: <# of bases including gaps>
Parameter 3: <# of reads in contig>
Parameter 4: <U or C for uncomplemented or complemented>
Example: CO Contig1 1475 8 156 U
agccccgggcccgtggggttccttgagcactcca
gagccgtgggaccagcactccaaggggtcctt
aagttccaaccag

Code: BQ
Description: Indicates the base quality scores for each contig base.
NOTE: The first line of the base quality data begins with a space character. Ignore it!
Example: CO Contig1 1475 8 156 U
agccccgggcccgtggggttccttgagcactcca
gagccgtgggaccagcactccaaggggtcctt
aagttccaaccag

Code: RD
Description: Indicates the beginning of a Read sequence block.
Parameter 1: <read name>
Parameter 2: <# of padded bases>
Parameter 3: <# of whole read info items>
Parameter 4: <# of read tags>
Parameter 5: <complemented or uncomplemented>
Example: CO Contig1 1475 8 156 U
agccccgggcccgtggggttccttgagcactcca
gagccgtgggaccagcactccaaggggtcctt
aagttccaaccag

This code indicates a contig. The contig sequence contains gaps (asterisk characters) which may need to be filtered out.

RD – This code indicates a read sequence. May be complemented or uncomplemented. Contains gaps (asterisk characters) that need to be removed!

NOTE: All forward reads files should be marked as uncomplemented (U), all reverse reads should be marked as complemented (C), and the complement of the contig should be ignored (as it is only used by consed to define the default display direction).

PART 2: ALIGNMENT PIPELINE

Overview

The purpose of the Alignment Pipeline is to take the contig sequences generated by the Phred-Phrap pipeline and modify them so that each consensus sequences aligns to each other. This is performed by a two step process: (1) contig set selection, and (2) contig alignment.

Step 1: Contig Set Selection

The first step in the Alignment Pipeline process is to select the set of contigs that will be used for generating the alignment. There is no absolutely consistent way of performing this step. In general, this step is performed partially by manual selection, and partially by utilizing the program *muscle*.

The criteria used for deciding which contigs to include and which reject depends on the nature of the project. Below is a list of possible decision factors:

- Presence of both forward and reverse reads.
- Phrap quality scores (BQ block in the ACE File).
- Manual contig adjustments and manipulations performed by biologists.
- GenBank limitations on the lengths of read sequences it will accept.
- Other project objectives and requirements.

Masking based on Phrap Scores is commonly performed to set a minimum threshold for contig quality. *Muscle* (an open source alignment program) can be configured to define a masking threshold. When a threshold is set, *muscle* reads the Phrap Scores associated with each candidate contig, and rejects (masks) contigs that do not meet that threshold limit.

Phrap Scores reflect the probability that the base call is correct. A Phrap Score of 10 means the probability of an incorrect call is 1/10 (or 10 percent), while a Phrap Score of 20 means that the probability of an incorrect call is 1/100 (or 1%). A summary is shown below.

Phrap Score = 10	→	10% probability of incorrect call
Phrap Score = 20	→	1% probability of incorrect call
Phrap Score = 30	→	0.1% probability of incorrect call
Phrap Score = 40	→	0.01% probability of incorrect call
...		
Phrap Score = 98	→	Manually edited (assume it is 100% correct)
Phrap Score = 99	→	Manually edited (assume it is 100% correct)

Step 2: Contig Alignment

The second step of the Alignment Pipeline process is to align the Contig Set selected in the previous step. This has largely been performed by *muscle*, but in some cases it has been performed by manually. The alignment process aligns the contig sequences together by adding padding and gaps to each sequence so that they align with one another. It then produces a FASTA file containing the aligned sequences, a quality

scores file, a table file, and an info file. These output files are called the Alignment File, Alignment Quality Scores File, Alignment Table File, and the Alignment Info File respectively. The file names of these files are formatted as shown below.

Alignment File: [ReadPairName].[SpeciesCode].v[VersionNumber].fsa
Example: 0_10453_01.v1.fsa

Alignment Quality Scores File: [ReadPairName].[SpeciesCode].v[VersionNumber].qvl
Example: 0_10453_01.v1.qvl

Alignment Table File: [ReadPairName].[SpeciesCode].v[VersionNumber].tbl
Example: 0_10453_01.v1.tbl

Alignment Info File: [ReadPairName].[SpeciesCode].v[VersionNumber].info.txt
Example: 0_10453_01.v1.info.txt

All alignment files are stored in the Alignments Directory (NOT the Phred-Phrap Directory).

Example Alignment File Path: /Data/populus/4CL3_09/Potr/alignments/

PART 3: MOST COMMON HAPLOTYPE (MCH) PIPELINE

Overview

There are two methods for determining the most common haplotype, the ADEPT2 MCH Method and the Poplar MCH Method. The ADEPT2 MCH Method was applied to the data found in the /Data/amplicons directory, and the Poplar MCH Method was applied to the data found under the /Data/populous directory. Each method is described separately below.

ADEPT2 MCH Method

Overview

The ADEPT2 Most Common Haplotype (MCH) Method selects the most common haplotype of the sequences present in the Alignment File based on a hash algorithm. (This method does not produce a consensus sequence.)

Step 1: MCH Contig Selection

The first step of the MCH Contig Selection pipeline is to select the most common haplotype based on John Liechty's hash based algorithm. This takes in each alignment sequence, removes the gaps, and compares the ungapped sequences to one another. The sequence that occurs most often is then chosen as the most common haplotype. In the case of a tie, the most common haplotype is selected randomly among the top scoring sequences.

Example:

Read Name	Sequence	Ungapped Hash	Total Matches
1001:	GT-GATG	GTGATG	1
1002:	GTCGACG	GTCGACG	2
1003:	GCTG-C-	GCTGC	1
1004:	GTCGACG	(same as 1002)	-

Final MCH Selection: GTCGACG

The MCH Contig Sequence is then written to a file called the MCH Contig Sequence File. The file name format for these files is shown below.

MCH Contig Sequence File: [AmpliconName].[SpeciesCode].v[VersionNumber].mch

Example: 0_10004_02.Pita.v1.mch

John Liechty's Notes

* Gaps are maintained so that the sequence reflects the sequence in the related aligned .fsa file. However, gaps were removed when determining the most common haplotype.

* There is an additional piece of info in the fasta header, '>0_10004_02-4032_Pita [mch frequency = 11/18]', that will give a quick note as to how common the haplotype was in the alignment. Note that for 4545 out of 5772, most common haplotype was more than half the samples, and for 5441 out of 5772, most common haplotype was more than a third of the samples in the alignment.

* These all reflect an actual sample (i.e. not a consensus sequence).

* Method for generating the most common haplotype was pretty simplistic, and I think it is fine for haploid sequence but we may want to rethink how we do this if we are working with diploid tissue with IUPAC ambiguity codes. If we want to do this with future data in other species, the original script can be found at /Users/jliechty/find_mch.py

Poplar MCH Method

Overview

The Poplar Most Common Haplotype (MCH) Method performs two tasks, (1) it creates a Consensus Sequence based on the alignments found in the Alignment File (generated in the Alignment Pipeline), and (2) it selects the most common haplotype of the sequences present in the Alignment File.

Step 1: MCH Contig Selection

The second step of the MCH Pipeline process is to select the contig sequence that is most similar to the MCH Consensus Sequence. This is performed by comparing the MCH Consensus sequence to each contig sequences found in the Alignment File. The

contig sequence with the fewest dissimilarities to the MCH Consensus Sequence is selected.

The MCH Contig Sequence is then written to a file called the MCH Contig Sequence File. The file name format for these files is shown below.

MCH Contig Sequence File: [AmpliconName]_[SpeciesCode]_mch_v2.fasta
Example: 4CL1_04_Pong_mch_v2.fasta

MCH Consensus Sequence Generation

The first step in the MCH Pipeline Process is to generate the MCH Consensus Sequence. This is performed by the MCH program, which loops through each position of each alignment sequence one by one, and determines the most common base amongst the sequences. An example is shown below.

Example Read A: G|T|T|G|A|T|G
Example Read B: G|C|T|G|T|C|G
Example Read C: G|T|C|G|A|C|G
MCH Consensus: G T T G A C G

Note that the MCH Consensus Sequence does NOT necessarily match any of the read pair in the Alignment File.

Once an MCH Consensus Sequence is generated, it is placed into a file referred to as the MCH Consensus Sequence File. The file name format for this file is shown below.

MCH Consensus Sequence File: [AmpliconName]_[SpeciesCode]_mch.fasta
Example: 4CL1_04_Pong_mch.fasta

NOTE: MCH Consensus sequences are only generated for PhredPhrap Directories that have successfully passed the Phred-Phrap Pipeline.

Notes

For Adept2 and ACESAP projects, the program used for alignment was *muscle*, with a little hand editing when needed. When introns were present, sibs4 was used to align genomic to mRNA data. For the Populus project *muscle* was used as well, but more hand editing was performed. For WHISP, Andrew Eckert performed the alignments (ask him for more information).

PART 4: SITE ANNOTATION PIPELINE

Overview

The Site Annotation Pipeline generates annotations for each Amplicon (a.k.a. Site or Locus). There are slight differences between the information produced for the Populus project and the ADEPT2 project. This pipeline generates a single data file called the Site Annotations File.

Step 1: Site Annotations File Format

The Site Annotations File is a tab-delimited file that consists of twenty columns. The column definitions are described in the table below.

Column	Name	Description
1	Amplicon Name	This field contains two pieces of data separated by a hyphen character: the Amplicon Name, and the Species Code. Example: 4CL3_01-Pofr
2	Site Length	The total length of the Amplicon.
3	UTR_start1	The starting position of a UTR region (if it exists).
4	UTR_stop1	The ending position of a UTR region (if it exists).
5	UTR_start2	The starting position of a UTR region (if it exists).
6	UTR_stop2	The ending position of a UTR region (if it exists).
7	exon_start1	The starting position of an exon region (if it exists).
8	exon_stop1	The ending position of an exon region (if it exists).
9	exon_start2	The starting position of an exon region (if it exists).
10	exon_stop2	The ending position of an exon region (if it exists).
11	exon_start3	The starting position of an exon region (if it exists).
12	exon_stop3	The ending position of an exon region (if it exists).
13	exon_start4	The starting position of an exon region (if it exists).
14	exon_stop4	The ending position of an exon region (if it exists).
15	intron_start1	The starting position of an intron region (if it exists).
16	intron_stop1	The ending position of an intron region (if it exists).
17	intron_start2	The starting position of an intron region (if it exists).
18	intron_stop2	The ending position of an intron region (if it exists).
19	intron_start3	The starting position of an intron region (if it exists).
20	intron_stop3	The ending position of an intron region (if it exists).
21	intron_start4	The starting position of an intron region (if it exists).
22	intron_stop4	The ending position of an intron region (if it exists).
23	frame	Defines the reading frame position of the first base in the Amplicon. This value may be 1, 2, 3, or 0. A value of 0 should only occur if no SNPs were found in the Amplicon.

Please note that fields two through twenty may be empty, depending on the type of data encountered.

PART 5: SNP ANNOTATION PIPELINE

Overview

The SNP Annotation Pipeline identifies SNPs that exist in each Amplicon (a.k.a. Site or Locus). This pipeline produces a data file known as the SNP Annotations File. The structure of this file is discussed below.

Step 1: SNP Annotations File Format

The SNP Annotations File is formatted as tab-delimited file. The first 9 fields contain Amplicon data. Fields that are found after the 9th appear in sets of four, and contain SNP data. For example, if zero SNPs were found for a particular Amplicon, then no data will be found after field 9. However, if three SNPs were found in a given

Amplicon, then 12 additional fields (four for each SNP) will appear after the 9th field. The composition of these fields are as follows.

Amplicon Data (first ten fields)

Column	Name	Description
1	Amplicon Name	Name of the Amplicon.
2	Species Code	Four letter species code.
3	Amplicon Length	Length of the Amplicon.
4	Number of read pairs	Total number of read pairs associated with this Amplicon (regardless of whether they were successful or not).
5	Number of alignment sequences	Number of contig sequences used to produce the alignment. This field only counts successful contigs, but may not include all successful contigs. Thus, this number should be less than or equal to the number of successful Phred-Phrap contigs.
6	Number of SNPs	Number of SNPs called on this Amplicon.
7	Scaffold Name	The name of the scaffold that the Amplicon resides on. (Scaffolds are somewhat analogous to chromosomes, but are not necessarily equivalent.)
8	Physical Start Location	Physical starting position of the Amplicon (relative to the genome).
9	Physical End Location	Physical ending position of the Amplicon (relative to the genome).

SNP Data (fields that appear after the tenth field)

Column	Name	Description
10,14, ...	SNP Name	This field contains the SNP name, which is comprised of three pieces of information: the SNP Name, the Species Code, and the Relative Position. , A hyphen character separates each piece of information. Example: 4CL3_01-Pong-122
11,15, ...	Physical Position	The physical position of the SNP relative to the genome. This number should fall within the range indicated in columns 8 and 9.
12,16, ...	SNP Annotation (i.e. SNP region description)	Describes the region that the SNP resides in (such as UTR, Intron, etc). The list of possible values and their definitions are summarized below: UTR - UTR SNP Intron - Intron SNP S - Synonymous NS - Nonsynonymous S/S - Three state synonymous/synonymous SNP S/NS - Three state synonymous/nonsynonymous SNP NS/S - Three state nonsynonymous/synonymous SNP NS/NS - Three state nonsynonymous/nonsynonymous SNP
13,17, ...	Genotype	The genotype of the SNP. This consists of two genotype values separated by a slash character. Example: G/C