Saccharomyces Genome Resequencing Project: User Manual

David Carter
Wellcome Trust Sanger Institute

September 20, 2008

1 Introduction

This manual describes the downloadable data for the Saccharomyces Genome Resequencing Project (SGRP), available at

`ftp://ftp.sanger.ac.uk/pub/dmc/yeast/latest`

There is also a web interface to some aspects of the data at

`http://www.sanger.ac.uk/Teams/Team71/durbin/sgrp/browser.shtml`

Unfortunately the author of this document is no longer working in bioinformatics and is not available to answer queries on how to work with the data. However, past experience shows that the answers to a very large majority of the queries received are already contained in this documentation, and that most answers to questions of the form "How do I..." involve the use of the `alicat.pl` *perl script described below. So our best advice is: read this documentation carefully, and make sure you understand all that* `alicat.pl` *can do!*

Version History

- 2008-02-29: Corrected and extended `recombinationFrequencies.txt`, and added ARG-inference (Margarita) files to the released data set; see Section 7.6.
- 2008-02-20: First version.

1.1 Overview of project and data

The project was a collaboration between Richard Durbin's group at the Wellcome Trust Sanger Institute and Prof. Ed Louis' group at the Institute of Genetics, University of Nottingham. Our goal was to advance understanding of genomic variation and evolution by analysing sequences from multiple strains of the two Saccharomyces species, *S. cerevisiae* (henceforth Sc) and *S. paradoxus* (henceforth Sp).

We carried out ABI sequencing of haploids of 36 Sc strains and 27 Sp strains to a depth of between 1x and 3x, yielding a total of 1.42 million reads (1.172 Gb, clipped). We also performed Illumina GA (Solexa) sequencing of four of the ABI-sequenced Sc strains and of ten additional Sp strains. Details are given in Section 3.

We created a sequence for each of the sequenced strains by means of an iterative process called PALAS (Parallel ALignment and ASsembly). This process was run separately on the data for each of the two species. Each strain sequence was initialized to the reference sequence, and then alignments between reads and their strain sequences were created using SsahaSNP. On each iteration, a subset of the alignments was accepted and a new sequence for each strain was created. Because of the low coverage per strain, information was shared between strains to impute both SNPs and short and long indels, by means of a method based on ancestral recombination graphs. This allowed decisions to be made at positions in the genome where some strains had no or only poor-quality evidence while other, closely-related ones were better represented. The final output of PALAS, once the process had converged, was a multiple alignment between the reference sequence and each of the strain sequences.

The data available for download include the following:

- Sc reference genome, downloaded from SGD on October 10th 2007; this is the same as the version used in the December 2007 release of Ensembl.
- An Sp reference genome constructed from our reads and those created by [1]. with a first attempt at GFF file describing its genes and other features. See Section 2 for more details.
- Contigs for *Saccharomyces bayanus* (Sb), also created by [1].
- A multiple alignment between the above three reference genomes.

- All the quality-filtered and clipped ABI reads.
- PALAS alignments for each species, Sc and Sp.
- A sequence for each strain, extracted from the PALAS alignments, and its alignment to the species reference.
- SNP tables, also derived from the PALAS alignments.
- Translations of all coding genes in the Sc alignments.
- Details of how individual reads were aligned to contribute to the PALAS per-strain alignments.
- Tables of recombination points inferred during the imputation process.
- Contigs created from the reads for each strain separately by Casey Bergman of the University of Manchester with the PCAP assembler. For details, see

  `ftp://ftp.sanger.ac.uk/pub/dmc/yeast/latest/pcap/README.SGRP.pcap`

1.2 Data Release Policy

The release of pre-publication data from large resource-generating scientific projects was the subject of a meeting held in January 2003, the "Fort Lauderdale meeting", sponsored by the Wellcome Trust, one of the Project funders. The report from that meeting is available at

`http://www.wellcome.ac.uk/assets/wtd003207.pdf`

The recommendations of the Fort Lauderdale meeting address the roles and responsibilities of data producers, data users, and funders of "community resource projects", with the aim of establishing and maintaining an appropriate balance between the interests of data users in rapid access to data and the needs of data producers to receive recognition for their work. The conclusion of the attendees at the meeting was that responsible use of the data is necessary to ensure that first-rate data producers will continue to participate in such projects and produce and quickly release valuable large-scale data sets. "Responsible use" was defined as allowing the data producers to

have the opportunity to publish the initial global analyses of the data, as articulated at the outset of the project. Doing so also will ensure that the data generated are fully described.

2 Reference genomes and inter-species alignments

2.1 S. cerevisiae reference

We used a version the Sc reference genome, downloaded from SGD on October 10th 2007; this is the same as the version used in the December 2007 release of Ensembl.

2.2 S. paradoxus reference

We built an Sp reference sequence using the Phusion assembler [2] from reads from the following sources.

- The ten strains listed in Table 1, which were collected in Silwood Park, London and were each sequenced to about 1x coverage. We tried including further European strains, but found that doing so produced less good results because the greater divergence confused the assembly process more than the extra data helped it.
- All reads from the original Sp sequencing project [1] (denoted by "Broad" in the table). [1]
- Artificial reads, denoted by "Shreds" in the table, constructed by shredding the 832 contigs created by [1] into paired 1,000 bp sequences, 5kbp insert size, with reads overlapping by 500bp, i.e. to depth 2.

The numbers of reads from each source, with the number and proportion placed in the assembly, are given in Table 1.

[1] Note that the strain identified as having been used in this project was the Russian strain `CBS432`, which we also sequenced. However, our analysis indicates that the strain used in [1] was extremely similar to the London strains we used and markedly different from `CBS432`: the alignments of our London reads to the contigs produced in [1] involve far fewer polymorphisms than our alignments of `CBS432` to these contigs.

Source	Reads	Placed	Proportion
Q32_3	19439	16207	0.8337
Q59_1	20630	15852	0.7684
Q62_5	21066	17118	0.8126
Q89_8	13107	10731	0.8187
Q95_3	22699	19174	0.8447
S36_7	9304	5590	0.6008
T21_4	21946	17433	0.7944
Y6_5	15459	12964	0.8386
Y7	19609	15039	0.7669
Z1_1	14606	11643	0.7971
Shreds	24805	22806	0.9194
Broad	177886	131075	0.7368

Table 1: Strains used to create the Sp reference assembly, with numbers of reads used, number aligned, and proportion aligned.

Running Phusion on these reads created 608 contigs organized into 471 supercontigs, for a total length of 12,265,206 bp. The N50 contig size was 172,493, compared to 11,872,617 bp total and N50 49,124 for the Broad assembly.

Next, we aligned the contigs against the Sc reference using Ssaha2 with default settings. The best alignments for the contigs were in good agreement with the supercontigs that Phusion had independently created: we were able to place 52 supercontigs (roughly the largest 52, covering 11,639,177 bp, or 94.9% of our assembled sequence). All but two of these placements were completely syntenic, once allowance had been made for subtelomeric rearrangements. We broke the other two supercontigs: one at a point between two contigs, and the other in the middle of a contig which appeared to be chimeric owing to sequence similarity between two regions of approximately 2,500 bp located at roughly 1,243,250 to 1,245,753 on ChrIV and 575,099 to 577,600 on ChrXV.

We then formed most of the Sp assembly by juxtaposing all the placed contigs, separating contigs within the same supercontig with 50 "N" symbols and separating adjacent supercontigs by 100 "N" symbols. However, coverage of the Sc sequence by the Sp contigs was poor in the mitochondrion (only

about 20%, all short contigs) and the rDNA region on ChrXII because of its multiple repeats. We therefore did not attempt to create a mitochondrial sequence. We carried out a separate Phusion run on the reads aligning to the Sc rDNA region; these assembled into a sequence of 8,393 bp for the each of the rDNA sequences proper (cf 8,375 for Sc) separated by spacers of 721 bp (cf 762 for Sc), and ending with a partial copy of the region containing only the first 1,200 bp.

2.3 Multiple alignment with S. bayanus

We downloaded contigs for the next closest sequenced species to Sc and Sp, *S. bayanus* (henceforth Sb) described in [1], and then created a multiple alignment between Sc, Sp and Sb.

Because the Sc and (see above) Sp genomes were in chromosome-sized fragments but Sb consisted of 1,098 contigs with N50 = 25,082, it was problematic to use existing multiple alignment software. For this reason, and given the relative small sizes of the genomes, we created WU-Blastn alignments (default parameters except V=10000) of Sp to Sc and Sb to Sc; Sp and Sb were not directly aligned. We then filtered alignments using an algorithm that allowed (non-nested) inversions but otherwise required colinearity. This gave clean-looking alignments, using 93% of the Sp sequence and 62.5% of the Sb, which given the divergences between these species seemed reasonable.

2.4 Lifting over the feature file

We then used the Sp-to-Sc alignment to lift over to Sp the Sc GFF feature file provided by SGD. We discarded annotations only when there was an obvious lack of homology (one or both ends did not align, or there was a gross change in the length or an inversion of one end). This means that many of the annotations of coding genes are inaccurate or nonsensical; there are many examples of non-final stop codons, final sense codons and coding regions whose lengths are not a multiple of three. Improving on this situation is unfortunately beyond our current resources. Because the situation would be even worse for Sb, given the greater distance from Sc and the more fragmented state of the sequence, we did not attempt a corresponding liftover for Sb.

3 Strains and Sequencing Coverage

Tables 2 and 3 list the strain names and places of origin, and the ABI and Illumina coverage calculated on the basis of nucleotides aligned and (for ABI only) nucleotides sequenced.

In the tables, coverage is calculated on the basis of a genome size of 13,057,722 for both species. This size are calculated on the assumption that there are 100 copies of the rDNA repeat on chromosome 12. This repeat is only represented by two copies and the intervening spacer in the reference sequence, which is of length 12,162,296 in Sc. Adding 98 extra copies of the repeat itself (8,375 bp in Sc) and of the spacer (762 bp in Sc) gives a total size of 13,057,722 bp. The rDNA component sizes are slightly different in Sp (8,393 bp and 721 bp respectively) but the reference genome is incomplete, so estimating the true size to be the same as that of Sc seems safest.

From the alignments, we estimated that the ABI clones had a mean length of abour 4,500 bp and a standard deviation of 1,000 bp. We also included in the Sc alignments read pairs derived by shredding the published assemblies of strains `YJM789` [3] and `RM11-1A` ([4]) into reads of 1,000 bp, with a constant clone size of 4,500 bp, starting at the first position in each chromosome and then every 1,000 positions thereafter. Thus the first pair came from positions 1 to 1,000 and 3,501 to 4,500, the second from 1,001 to 2,000 and 4,501 to 5,501, and so on. We used both the complete chromosomes and the leftover contigs from the `RM11-1A` data; the `YJM789` data contained only complete chromosomes.

Earlier versions of the SGRP data included the Sc strain `YGPM`. This was excluded from the final release because we could not discover its origin, and because the reads had some unusual characteristics (lengths, quality scores and apparent reliability) that made it difficult to fit them in with the others.

4 Alignment and assembly with PALAS

The alignments giving rise to the per-strain assemblies were performed by PALAS (Parallel ALignment and ASsembly). This is an iterative process that works roughly as follows.

1. Generalize the reference sequence to a graph, in two ways. First, identify repeats in it, and hypothesize that any copy of a repeat may be absent in other strains. Second, align all reads against the reference

Strain	Location	Notes	Seqd	Aligd	IGA
273614N	RVI, Newcastle UK		0.93	0.87	
322134S	RVI, Newcastle UK		0.98	0.91	
378604X	RVI, Newcastle UK		1.05	0.99	
BC187	Napa Valley, USA	spore from UCD2120	0.67	0.63	
DBVPG1106	Australia		0.69	0.64	
DBVPG1373	Netherlands		1.28	1.22	
DBVPG1788	Finland		1.18	1.08	
DBVPG1853	Ethiopia		0.97	0.91	
DBVPG6040	Netherlands	ex S. fructum	0.86	0.81	
DBVPG6044	West Africa	ex S. manginii	1.40	1.33	
DBVPG6765	Unknown	ex S. boulardii	3.28	3.00	
K11	Japan	Awamori-1	0.88	0.83	
L-1374	Chile		1.01	0.92	
L-1528	Chile		1.17	1.01	
NCYC110	West Africa	ex S. chevalieri	0.85	0.81	
NCYC361	Ireland	ex S. diastaticus	0.65	0.59	
S288c	California, USA	Lab strain	1.20	1.14	9.78
SK1	USA	Lab strain	3.58	3.27	15.61
UWOPS03-461.4	Malaysia	Race uvarum	0.99	0.93	
UWOPS05-217.3	Malaysia	Mel-	0.99	0.93	
UWOPS05-227.2	Malaysia	Mel+	1.02	0.98	
UWOPS83-787.3	Bahamas		0.94	0.87	
UWOPS87-2421	Hawaii		0.95	0.89	
W303	Unknown	Lab strain, Eurofan 64/A	2.44	2.33	3.01
Y12	Africa	NRRL-Y12663	0.88	0.82	
Y55	France	Lab strain	3.83	3.42	8.94
Y9	Japan	NRRL-Y5997	0.81	0.76	
YIIc17-E5	Sauternes, France	No growth in minimal media	1.01	0.95	
YJM975	Bergamo, Italy		1.03	0.98	
YJM978	Bergamo, Italy		1.06	0.98	
YJM981	Bergamo, Italy		0.84	0.78	
YPS128	Pennsylvania, USA		1.28	1.23	
YPS606	Pennsylvania, USA		1.59	1.47	
YS2	Australia		0.66	0.61	
YS4	Netherlands		1.08	1.01	
YS9	Singapore	Le Saffre	1.03	0.98	

Table 2: "Seqd" is the number of ABI nucleotides sequenced and surviving clipping and quality filtering. "Aligd" is the number incorporated into the final alignment. "IGA" is the number of of Illumina GA (Solexa) nucleotides successfully aligned. All three are divided by the genome size estimate of 13,057,722 to give mean coverage depth.

Strain	Location	Notes	Seqd	Aligd	IGA
A12	Canada		1.27	1.07	
A4	Canada		1.25	1.11	
CBS432	Russia		4.20	3.92	
CBS5829	Denmark	1 non-reciprocal translocation	2.72	2.44	
DBVPG4650	Italy		1.46	1.34	
DBVPG6304	California, USA		1.57	1.35	
IFO1804	Japan		0.82	0.78	
KPN3828	Siberia		0.83	0.77	
KPN3829	Siberia		0.79	0.76	
N-17	Russia		2.87	2.62	
N-43	Far Eastern Russia	1 reciprocal translocation	1.57	1.44	
N-44	Far Eastern Russia	No Y'	1.30	1.20	
N-45	Far Eastern Russia		3.40	2.97	
Q31.4	London, UK				28.82
Q32.3	London, UK		1.23	1.13	
Q59.1	London, UK		1.32	1.10	
Q62.5	London, UK		1.26	1.16	
Q69.8	London, UK				27.83
Q74.4	London, UK				1.77
Q89.8	London, UK		0.86	0.76	
Q95.3	London, UK		1.42	1.31	
S36.7	London, UK		0.65	0.42	
T21.4	London, UK	chrI amplification	1.40	1.23	
UFRJ50791	Catalao Point, Brazil		0.74	0.69	
UFRJ50816	Tijuca Forest, Brazil	ex S. cariocanus	1.28	1.05	
UWOPS91-917.1	Hawaii		2.01	1.58	
W7	London, UK				2.99
Y6.5	London, UK		0.97	0.89	
Y7	London, UK		1.15	1.04	
Y8.1	London, UK				3.55
Y8.5	London, UK				1.70
Y9.6	London, UK				1.59
YPS138	Pennsylvania, USA		1.28	1.17	
Z1	London, UK				3.52
Z1.1	London, UK		0.92	0.83	

Table 3: Columns are as in Table 2

sequence to detect evidence (from broken alignments) of transposons being present in some strains in locations where they are absent in the reference, and add edges for these transposons to the graph.

2. Initialize the assembly for each strain to be identical to the (generalized) reference.

3. Align all (so far unplaced) reads for each strain to that strain's current assembly, using ssahaSNP (essentially ssaha2). The scores returned by ssahaSNP are discarded and replaced by a score $L - (P - 90)/10$, where L is the number of nucleotides aligned from the read and P is the percentage identity. Alignments that score less than 20 on this basis, including all alignments with percentage identity 90 or less, are discarded. This score change is made partly because ssahaSNP's scoring scheme is sensitive to the length of the sequence (e.g. chromosome) matched against, which is undesirable in our context, but mostly so as to ensure inexact matches are heavily penalized. Inexact matches are quite likely to be from (perhaps distant) homologues that are not direct orthologues of anything in the sequence being aligned against.

4. Look for the best "cluster" (consistent set) of alignments for each read pair. In the ideal case, there is one alignment of each of the paired reads, correctly oriented on the same chromosome and separated by a distance consistent with the known insert size distribution. However, it is necessary to allow for this pattern being disrupted by rearrangements (causing multiple or broken alignments for one read), chimeric read pairs and bad data (contaminants).

5. Whenever the best cluster for a read pair is clearly better than the next best, put the alignments in it forward to be placed onto the assembly.

6. Accept a subset of the alignments proposed in step 5. If alignments from clusters A and B overlap in the alignment of the assemblies of all strains, and A scores more than B, we reject B if either A and B are from the same strain, and/or one (or both) of the overlapping alignments from A and B is does not account for the whole of its read, i.e. accepting it implies hypothesizing an indel.

7. Place the accepted alignments from step 6 onto their respective assemblies. Where alignments of two or more reads for the same strain over-

lap (necessarily arising from different iterations), at locations where nucleotide values disagree, take the highest-quality nucleotide. This generates an incomplete consensus sequence for each strain (incomplete because there will not be alignments covering every part of the genome).

8. Fill in the incomplete sequences by imputing nucleotide values where necessary (see Section 5). The imputation process also identifies likely sequencing errors, i.e. where nucleotide values differ between strains, it attempts to distinguish SNPs from sequencing errors. Then if this iteration is the final one (see step 10 below), go to step 11, otherwise go to step 9.

9. Set the assembly for each strain to its complete sequence created in step 8. When there is no evidence for either the presence or absence of an indel (of more than a few base pairs) that exists in some other strains, allow both possibilities; i.e. we preserve the generalizations created in step 1, and create new ones when required, only resolving them when there the evidence is available to do so.

10. If the number of sequenced nucleotide positions generated in step 7 increased by a fraction less than 0.0001 compared to the previous iteration, set the next iteration to be the final one, else set it to be non-final. Go to step 3.

11. In a parallel way to step 9, set the assembly for each strain to its complete sequence created in step 8. But this time, when there is no evidence for either the presence or absence of an indel, we do not preserve generalizations; instead, we force a decision, using a variant of the imputation process as in 8, averaging over all positions in each region in question. However, when there is a long region (several thousand bp) with no reads placed, this is at least as likely to be due to divergence as to deletion; such regions are common in telomeres. In this case, fill the region in with "N" symbols. These symbols make up about 5 to 6 per cent of most final strain sequences.

12. Bring in Solexa data for strains for which it is available by aligning it to the sequence created in step 11 (or in the case of Sp, where the Solexa-sequenced strains were disjoint from the ABI-sequenced ones,

align to the original reference). Merge the aligned Solexa data with the ABI alignments from step 7, and carry out a final round of imputation (as in steps 8 and 11).

13. Run through steps 3 and 4 once more. Save the information on the possible placements (of ambiguous reads, by definition) for display in the browser, but do not attempt to add them to the strain sequences.

The final iteration differs from the others in certain parameter settings: in order to place as many reads as possible, we only allow "simple" clusters (one alignment of each read) in step 4, but we then permit overlapping alignments from the same strain. We also increase imputation accuracy in step 8 by sampling 25 ARGs (described in Section 5) rather than 3 which are sufficient (and not too slow) in the other iterations.

One issue that PALAS does not directly address, except for transposons in Sc, is duplication, i.e. when a strain has more than one copy of a region that is only present in one copy in the reference sequence. Another is inversion, which is believed to be rare. However, the `alignmentClashRates.txt` files (see Section 7.3) indicate places where overlapping reads from the same strain disagree more often than would be expected by chance, and one possible cause of this is that they come from different copies of a duplicated region (the other cause is mapping error).

5 Imputing Nucleotide Values

We impute nucleotide values and associated error probabilities for all reference base positions for all strains by applying Felsenstein's algorithm to the trees created by Margarita [5], an ancestral recombination graph (ARG) constructor. Margarita is run on polymorphisms in the data created in step 7 of the PALAS algorithm (this is the `sequenced.gz` files described in Section 7.2). The imputation process is described in full in [6]. Its output differs from its input in the following ways.

- Imputation can change a value at a site if closely related strains in that part of the genome disagree, thus correcting many sequencing errors. For example if strain S_1 has an `A` with quality 10 (error probability 0.1), and a closely related strain S_2 has a `C` with quality 40, it is much more likely that the `A` is a sequencing error than that a mutation has

occurred on the lineage between S_1 and S_2 or that the `C` in S_2 is an error.

- Imputation will also infer values for a strain when there is no data at all in that position for that strain (generally with lower confidence).

The imputation process is known to have some limitations. The main ones are:

1. The model of mutation in the phylogenetic trees is very simplistic. Gap symbols are treated as if they were a "fifth nucleotide", and mutations from one such nucleotide to any of the other four are assumed to be equally likely (transitions are not favoured over transversions). Each node in the ARG has an age assigned to it, so that mutation probabilities are calculated on branches using the molecular clock assumption. However, no allowance is made for selection, i.e. for (effectively) different mutation rates in, for example, silent and non-silent positions in coding regions.

2. Mutations on different branches and sequencing errors in different strains are all treated as independent. In reality, selection may partly invalidate the first assumption, and mapping errors and correlated (context-dependent) sequencing errors may partly invalidate the second. The main consequence of this is probably that many imputed quality scores are too high in cases where nucleotide values in neighbouring strains agree.

3. We run Margarita to create either 3 or 25 ARGs per chromosome (depending on the iteration of PALAS – see above), and then assume these ARGs sample the space of ARGs uniformly, i.e. we assign a prior probability of 1/3 or 1/25 to each one.

4. We assume that the recombination, if any, always occurs exactly halfway between each pair of polymorphisms used by Margarita, whereas in fact it can occur anywhere in this range (if it occurs at all). This may invalidate the resolutions of some apparent polymorphisms not used by Margarita (singleton and low-quality apparent polymorphisms are not used).

We found that earlier versions of the imputation process tended to preserve too many short indels, resulting in dubious-looking frame shifts in coding regions. We therefore reduced the qualities associated with the gap characters on short indels (those created within rather than between SsahaSNP alignments) as follows.

We reasoned that overall, genuine indels must be less likely in verified coding regions than in intergenic regions. If the true probability of an insertion or deletion of length L (in one or more strains) occurring between two nucleotides (in the other strains) is $P_v(L)$ in verified coding regions and $P_I(L)$ in intergenic regions, then the rate actually observed should be roughly $P_v(L)+P_f(L,Q)$ in verified coding regions and $P_I(L)+P_f(L,Q)$ in intergenic regions, where $P_f(L,Q)$ is the false positive rate, which depends on L and also on Q, which is some function (such as the mean) of the quality scores on the apparent indel. We also assume that $P_f(L,Q) \to 0$ as Q increases, while $P_V(L)$ and $P_I(L)$ are independent of Q. This means that for a given value of L and Q (or a given range of those values, to avoid data being too sparse) we can estimate the proportion of apparent indels, in both verified coding and intergenic regions, that are true positives; and this proportion can be converted into a quality score.

We take S_G as the sum of quality scores over all the N_G strains with a sequenced gap character at a site, and S_N as the sum over the N_N strains with a nucleotide. If $S_G \leq S_N$, set Q (as in the previous paragraph) to $S_G/(N_G+N_N)$, otherwise to $S_N/(N_G+N_N)$. We found that it made sense to treat the cases $L=1$ and $L=2$ separately, but that $L \geq 3$ behaved similarly so we treated them together. For intergenic regions, the following produced a good fit to our observed Sc data:

L	q'
1	2.13 Q - 3.19
2	2.69 Q - 0.43
≥ 3	2.68 Q

When $q' < \sqrt{10}$, the corresponding error probability is greater than 1/2, so we alter gaps to nucleotides (if $S_G \leq S_N$) or vice versa (otherwise), thus deleting many unlikely indels.

We applied this formula to all short indels, not just intergenic ones, on the basis that it would not be advisable to treat indels differently on the basis of external features (which are not even reliably available for Sp). This

means that while the indels in intergenic regions do represent our best guess, there are almost certainly more false positives than false negatives in coding regions, especially verified ones, and it is up to the user to judge which ones are genuine on the basis of specific knowledge about the gene in question. Even this conservative approach removes the majority of apparent indels, however, and results in a much lower density of them in verified coding than intergenic regions.

6 Getting and using the data

6.1 To download or to browse?

As we stated at the beginning of this document, you can either download the SGRP data by ftp, or browse some of it on the project web site. The browser is useful for getting overviews of regions: it will show you how reads aligned for each strain, what features are annotated, what indels have been detected for what strains, and how the SNP density varies. However, it has three major limitations:

1. It does not show you information on individual SNPs.
2. It is developed and maintained less enthusiastically than the downloadable data, because we hope to switch from a Gbrowse implementation to an Ensembl one at some point.
3. It does not allow you to aggregrate information. For example, if you wanted to know how many reads had aligned to each chromosome for a particular strain, the browser would not really help you.

So if you want to analyse SNPs, do whole-genome or whole-chromosome analyses, or request additional functionality, you are best advised to use the download data.

To help you extract useful information from this, we include a perl script, `alicat.pl`, so named because it started life as a generalization of the Unix `cat` command for alignment files. Detailed documentation on its usage is at the top of the script, and can be conveniently viewed using the command `perldoc alicat.pl`. But to give you a flavour of it, its features include:

- Printing alignments of either reads (`layout.gz`, described in Section 7.2) or consensus (`sequenced.gz` and `imputed.gz`).

- Restricting attention to specified strains and/or to regions specified by chromosomal offsets or GFF feature names,
- Printing in either forward-strand or reverse-strand orientation.
- Printing only polymorphic loci, or all loci.
- Printing sequences either vertically (one per column) or horizontally (one per row in each block).
- Printing quality values in several different formats.

An example command is:

```
alicat.pl -dy -v -t 100 -P -r GAL1:/foo/ref/cere/genome.gff \
          -l -500 -h 200 */imputed.gz
```

This tells `alicat.pl` to look for (gene) `GAL1` in the specified GFF file, then find the first of the specified `imputed.gz` files whose directory name is the appropriate chromosome name. Convert and print that range of lines from that file, starting 500 base pairs upstream of the start of the gene (`-l -500`) and finishing 200 base pairs after the end (`-h 200`). Lines are "turned" (`-t 100`) to be horizontal, with 100 alignment positions per line. Within each batch of turned lines, only print details of strains that are different from the reference (`-P`) somewhere in the line. Replace (`-d`) sequenced values identical to the reference by ":", and imputed by ".". Replace (`-y`) each Phred quality character by its tens digit; thus `J`, for quality 41 (see 8) becomes 4.

6.2 Downloading and unpacking the tar files

The directory

```
ftp://ftp.sanger.ac.uk/pub/dmc/yeast/latest
```

contains a number of compressed tar files:

```
cere_assemblies.tgz   para_assemblies.tgz
cere_matches.tgz      para_matches.tgz
cere_reads.tgz        para_reads.tgz
cere_recombs.tgz      para_recombs.tgz
misc.tgz
```

The directory

```
ftp://ftp.sanger.ac.uk/pub/dmc/yeast/latest/pcap
```

contains PCAP contigs for reads of each strain separately, in files

```
README.SGRP.pcap      cere_pcap.tgz      para_pcap.tgz
```

The "misc" file is fairly small, but the "assemblies" and "reads" file are each about half a gigabyte, the "recombs" files are slightly larger, and the "matches" files are just over a gigabyte. The PCAP files are each about 1.5 gigabytes.

Download whichever of the files you want (see Section 6.3 to find what data is in which tar file), then in a Unix-like system[2] `cd` to an empty directory and unpack them with commands such as

```
tar zxf cere_main.tgz
```

This will create files in the directory structure shown in the remainder of the current section.

In all files in the download data, chromosome names for both species are are `chr01` ... `chr16`, with (for Sc only) `chrm` for the mitochondrion and `scplasm1` for the 2-micron circle.

Strain names are as in the tables, except that hyphens and periods are both replaced by underscore; thus `UWOPS05-217.3` becomes `UWOPS05_217_3`.

Some information is given for strain name `MRCA` in both the Sc and Sp data sets. This represents the sequence inferred during the imputation process for the Most Recent Common Ancestor of the strains of each species (thus perhaps confusingly, `MRCA` represents a different hypothesized strain for the two species).

In what follows, `$SP` represents a species code (`cere` for Sc, `para` for Sp, `baya` for Sb); `$STR` represents a strain name as specified above; and `$CHR` is a chromosome name.

[2]That is, Unix, Linux and Mac. If you are using Windows, you will need Cygwin: `http://www.cygwin.com`. If you want to use `alicat.pl`, which is very likely, then you need to select perl to be installed alongside the default Cygwin packages. If you need help with this, please ask your system administrator, not me.

6.3 Overview of the data files

Here, we list the types of files in the data. The `tgz` file containing each one can be found by this algorithm:

- `$SP/match/$CHR/recombs` files are in `$SP_recombs.tgz`.
- All other `$SP/match` files are in `$SP_matches.tgz`.
- `$SP/strains/$STR/assembly` files are in `$SP_assemblies.tgz`.
- All other `$SP/strains` files are in `$SP_reads.tgz`.
- All other files are in `misc.tgz`.

`code/alicat.pl` . Perl script for extract alignment data. See Section 6.1.

`$SP/trees/$SP.matrix` . Distance matrix showing divergences between strains, expressed in polymorphic positions per thousand (calculated from aligned positions where both strains have a nucleotide rather than a gap character or no aligned reads; so be sceptical of fine details and the absolute magnitudes of the values).

`$SP/trees/$SP.tree` . Newick-format neighbour-joining tree derived from the matrix. Note that this tree does not represent evolutionary history because the strains have recombined; it is best viewed as a compromise between the different trees applying to different parts of the genome (see also Section 7.6).

`$SP/trees/$SP.ps` . Postscript file derived from the tree, with some manual editing. See caveats in above paragraphs.

`$SP/ref/genome.fa` . FASTA format reference genome.

`$SP/ref/genome.sz` . File showing number of nucleotides in each chromosome.

`$SP/ref/genome.gff` . GFF annotation file (see Section 2 for a caveat on the Sp version of this).

`$SP/strains/$STR/renamed.fastq` FASTQ format file of all (capillary) reads for `$STR` that survived initial, fairly mild filtering. You may want to check on the reasons why some reads were renamed after sequencing (and being placed in the trace archive); the main objective was to ensure that reads named *prefix*`.p1k` and *prefix*`.q1k` actually are mates as consistently as possible.

`$SP/strains/$STR/renamed.wm` "Watermark" file, showing the extent of the good-quality sequence of each read for the strain. The first field is the read name; the second is its length; and subsequent pairs of numbers denote (inclusive) regions of high-quality sequence.

`$SP/strains/$STR/name_corrections.txt` Specific name corrections carried out to create renamed.fastq. A line containing a single name means that read was discarded and does not appear in renamed.fastq; a line consisting of two names, "name1 name2", means the read named "name1" in the trace archive is "name2" in renamed.fastq.

`$SP/strains/$STR/fates.txt` For every read, shows its eventual fate in the alignment and assembly process. The format is described in Section 7.1.

`$SP/match/$CHR/sequenced.gz` Alignment of all sequenced nucleotide values and quality scores for the given species and chromosome. The format is described in Section 7.2.

`$SP/match/$CHR/imputed.gz` Alignment of all nucleotide values and quality scores for the given species and chromosome, after nucleotide imputation has been carried out to fill in missing values and correct errors. The format is described in Section 7.2.

`$SP/match/$CHR/layout.gz` Individual read alignments. The format is described in Section 7.2.

`$SP/match/$CHR/alignmentClashRates.txt` The extent to which individual reads disagreed with other reads for the same strain aligned over the same span, possibly suggesting copy-number variation or mis-mapping. The format is described in Section 7.3.

`$SP/match/$CHR/coverageABI.gz` These files show the ABI alignment depth for every strain at every point in the alignments. The format is described in Section 7.4.

`$SP/match/$CHR/coverageIllumina.gz` Ditto, for Illumina sequence data.

`$SP/match/$CHR/sequencedSNPs.txt` List of all SNPs (with respect to the reference sequence) on this chromosome for every strain. The format is described in Section 7.7. The equivalent for imputed data is in `$SP/match/$CHR/imputedSNPs.txt`

`$SP/match/$CHR/translations_fwd` Directory containing Translations of every coding gene on the forward strand of `$CHR`, suitable for printing with `alicat.pl`. Reverse strand genes are in `$SP/match/$CHR/translations_rev`.

`$SP/match/$CHR/recombs/mgr_in` and `$SP/match/$CHR/recombs/mNNN/mgr_in`. Margarita input files. Formats of this and the other files in `recombs` are described in Section 7.6.

`$SP/match/$CHR/recombs/mNNN/mgr_out` Margarita output files.

`$SP/match/$CHR/recombs/args_bl` Trees derived from ARGS, with branch lengths, for all 25 ARGs constructed in the final PALAS iteration, between each pair of markers.

`$SP/match/$CHR/recombs/recombinationFrequencies.txt` Mean numbers of recombinations between each pair of markers in the ARG.

`$SP/strains/$STR/assembly` Files in these directories are extracted from `$SP/match/$CHR/imputed.gz` files. They are:

`genome.fa` FASTA file of the PALAS-derived assembly for `$STR`.

`genome.gff` GFF file for the assembly, lifted over from the reference (cere only)

`genome.map` File showing how genome.fa maps onto `$SP/match/$CHR/sequenced.gz` and `$SP/match/$CHR/imputed.gz` Details to follow; mail me if you want this.

`genome.dangles` File showing offsets (in `genome.fa`) at which further material may exist. The format is chromosome name, offset (from 1) and either L or R, depending on whether the extra material is dangling to the left or the right of the position. The latter happens when, for example, the first part of a read is aligned in a forward direction and (the good-quality part of) the rest is taken

to be an insertion; the insertion probably continues past the end of the read.

`genome.refalign.gz` Cut-down versions of `$SP/match/$CHR/sequenced.gz` and `$SP/match/$CHR/imputed.gz` showing only the reference sequence and the strain sequence. The format is described in Section 7.2.

`genome.seqr` Regions in `genome.fa` that are based on sequence data for the current strain rather than imputation from other strains.

`$SP/strains/$STR/solexa.fastq.gz` FASTQ format files showing consensus MAQ alignments for Solexa data. The results of these alignments are fed into

`$SP/match/$CHR/sequenced.gz` , and then into the final round of imputation, producing

`$SP/match/$CHR/imputed.gz` .

`$SP/match/$CHR/sbw.txt` . "Similarities By Windows". Shows how similar each pair of strains is in all 10kbp windows in all chromosomes. The format is described in Section 7.5.

7 File formats

This section gives details of the formats of some of the files listed in Section 6.3.

7.1 Format of `fates.txt` files

In each line:

- Field 1 is the read name.
- Fields 2 to 4 show the PALAS iteration at which the read was first aligned, approved and placed, respectively. The first iteration is zero; the final one, F, was 30 for Sc and 36 for Sp. A value of $F + 1$ (31 or 37, respectively) means that the event in question never took place. "Aligned" means the read received some alignment from SsahaSNP

against the assembly for its strain that was current at the iteration in question; this assembly is the reference genome itself at iteration zero. "Approved" means that PALAS was able to select a set of alignments for this read and its mate that appeared to be consistent with each other and with placed alignments for the same strain at earlier iterations (the latter condition is broken when, for example, a read bridges a location where an insertion was recognized at an earlier iteration). "Placed" means that PALAS decided to irrevocably accept the approved alignment, allowing it to contribute to further versions of the assembly. The main constraint against early placement of reads is that, before the final iteration, PALAS does not accept overlapping alignments for different reads of the same strain in the same iteration. Thus if coverage depth is N at a given point, PALAS requires at least N iterations to place all the reads.

- Fields 5 to 7 condense the information in fields 2 to 4: 0 maps to 0 (first iteration), 1 to $F - 1$ inclusive map to 1 (non-initial, non-final iteration), F maps to 2 (final iteration), and $F + 1$ maps to 3 (event never occurred).

- Field 8 is the score of the best-scoring alignment for the read. The score we use is 3S-2L where S is the ssahaSNP score and L is the length of the alignment. This has the effect of penalizing mismatches about three times as heavily as ssahaSNP does, which seems to be appropriate for the levels of relatedness of our strains. Alignments scoring less than 20 (after this adjustment) are discarded.

- Field 9 is the percentage identity of the best scoring alignment.

- Fields 10 and 11 are the low and high positions (in the read) of the alignment.

- Field 12 is the length of the read, copied from renamed.wm.

- Field 13 is the number of bases inside the "good" regions of the read, from renamed.wm

- Field 14 is the chromosome to which the read was aligned, or `NULL` if it was not aligned.

- Field 15 describes the fate of the read, as follows:

`placed` : read was successfully placed.

`ambiguous` : not placed because alignments to two or more locations scored about equally well. These reads are from within repeats, often transposons.

`clashed` : not placed because of a structural clash with a better scoring alignment of another read for the same strain: one diverges (requires an insertion or deletion) where the other does not, and both cannot be right.

`discarded` : alignment was initially accepted but later rejected (this is a workaround for some problems and should disappear in later releases, if there are any).

`no_good_cluster` : all clusters (sets of consistent alignments of this read and its mate if available) scored too poorly to be safely accepted.

`no_good_mate` : mate of this read appeared good, but could not be consistently aligned.

`unsafe_insertion` : best cluster required either more than one insertion, or only one insertion but alignments did not score well enough to make it safe.

`bad_read` : read never aligned at all, and had fewer than 100 bases of good sequence.

`mediocre_read` : read never aligned at all, and had at least 100 bases but fewer than 400 bases of good sequence.

`never_aligned` : read never aligned at all, in spite of having at least 400 bases of good sequence (and therefore being good enough that we would normally expect an alignment).

7.2 Format of alignment files

The alignment files `sequenced.gz`, `imputed.gz`, `genome.refalign.gz` and `layout.gz` have closely related formats. They may be best inspected using the `alicat.pl` script.

The first line is of the form

```
>strain1 strain2 ...
```

showing the (alphabetical) order applied to the strains whose values occur in the file. (However, `genome.refalign.gz` has a chromosome name instead of a strain list here, and has one line of this form for each chromosome).

Each following line consists of interleaved nucleotide and quality characters, one such pair for each strain, beginning with the reference genome and continuing with strain1, strain2 etc. Thus the third column is the nucleotide value for strain1, and the fourth is its quality value. The following conventions apply:

- If the nucleotide is blank, no read for that strain was matched (and survived filtering) at this point.
- For `sequenced.gz`, if a non-blank nucleotide is upper case, it passes NQS (neighbourhood quality standard); if lower-case, it fails. For `imputed.gz` a lower-case nucleotide instead denotes a value inferred where there was no value, or occasionally a different nucleotide, in `sequenced.gz`. Upper case means that the same nucleotide occurs here in `sequenced.gz`, irrespective of NQS.
- In `sequenced.gz`, individual gap locations are indicated by digits 1 to 9. The different digits are equivalent except that they represent overlapping runs of different extent; they are distinguished only so that Margarita treats them differently. In the other two files, a hyphen or underscore is used; the underscore functions as the "lower case" counterpart of the hyphen in the sense of the paragraph above. In all three file types, an equals symbol = represents a large gap: one resulting from the inference of an insertion or deletion on the basis of partial SsahaSNP alignments, as opposed to the small gaps that can occur within single alignments.
- If the nucleotide is non-blank and non-gap but the quality is blank, the error probability is very low: either this is the reference genome, i.e. column 2 of the line, or the evidence is so strong that Q would have an ascii code greater than 126.

`layout.gz` files, which are currently found in `cere_layouts.gz` and `para_layouts.gz`, show how individual reads have aligned. The values in `sequenced.gz` are essentially consensi derived from the per-read values in `layout.gz` (although a certain amount of local realignment has been done,

so the correspondence is not always completely exact). Annotation lines in `layout.gz` are of the form

```
>readName:from:to[more] posn iter
```

where readName is the name of a read (the strain name followed by a hyphen, then a plate location, and finally `.p1k` or `.q1k`). from and show which offsets in the read have been aligned here. `[more]`, if present, will be `|L` or `|R`, showing that the alignment is in fact an insertion to the left or the right. `posn` is a non-negative integer; if `posn=P` then the nucleotides for this read will be in column `2P+3` and the quality values in column `2P+4` (thus the first two columns are for the reference sequence, the next two for the read with `posn=0`, and so on). `iter` is the PALAS iteration on which the read was placed. There is no corresponding annotation line for the end of a read; that is simply shown by another read being annotated for the same position, or by the columns for the read being filled by blanks, whichever happens first.

7.3 Format of alignment-clash files

`alignmentClashRates.txt` files have one line for every read aligned to a chromosome where there is a non-negligible degree of disagreement with one or more other reads from the same strain. Each line is of the form

```
Z  Sum  Exp  NDis  Read  Chr  Extra  HC  From  To  Quals
```

If a read is present at all in this file, and especially if its `Z` value is 2 or more, then the read does have significant clashes with at least one other, and the chances are either that one of the reads was misaligned (we cannot in general tell which one), or that they come from different copies of a region that is only present in one copy in the aligned sequence; PALAS does not try to spot copy number variation, so this situation will sometimes arise.

In more detail: the algorithm that constructs these files considers every position where alignments of two reads of the same strain coincides, and considers the number of agreements and disagreements and the quality scores involved. Specifically, when reads r_1 and r_2 have nucleotide values v_1 and v_2 and qualities q_1 and q_2 respectively, we calculate p_e, the probability that two nucleotides of these qualities will disagree by chance (assuming independence).

The value $I = 1/p_e$ then has expectation 1 and variance I. `Exp` is the number of I values in which a read participates, `Sum` is the sum of I values

over positions where $v_1 \neq v_2$, and `NDis` is the number of such positions. To prevent runs of disagreements inflating `Sum` too much, we only look at disagreements more than 10 bp apart. The expected value of `Sum` is `Exp`, and `Z` is an estimate of the number of standard deviations by which `Sum` exceeds `Exp`, applying a suitable Bonferroni correction. Thus any `Z` value greater than about 2 is likely to indicate a significant degree of disagreement.

The other values on the line are as follows. `Read` and `Chr` identify the read and chromosome involved, and `From` and `To` are the alignment coordinates spanned by the read. `Quals` is the list of qualities (derived by converting p_e back to a quality value) of the disagreements, in decreasing order, with at most ten shown. `Extra` is 1 if the read was only tentatively placed by PALAS (see step 13 of Section 4), and 0 otherwise; we increment `Sum`, `Exp` or `NDis` only for clashes with non-tentative alignments. `HC` stands for "high coverage", and indicates that more than twenty reads for the strain were aligned together. In this situation, not all overlapping pairs of reads are compared, so the statistics are incomplete; but there are significant numbers of such reads only in the rDNA region on chromosome XII.

7.4 Format of coverage files

`coverage.gz` files have the following format. The first line is a list of the N strains for which (ABI) reads have been aligned for the current chromosome. Each subsequent line consists of N+2 numbers. The first two numbers are the low and high ends (starting from 1) of a range of alignment positions, i.e. of (non-annotation) lines in `sequenced.gz`, `imputed.gz` or `layout.gz`. Each of the next N numbers is the number of reads aligning in that region for the corresponding strain. A new line in `coverage.gz` is created every time that quantity changes for one or more strains. Things to be aware of:

- The counts are for ABI reads only, not Solexa ones, so if `sequenced.gz` shows a nucleotide but the corresponding value in `coverage.gz` is zero, the presence of Solexa reads is probably the explanation.
- Minor realignments exist between `layout.gz`, from which `coverage.gz` is calculated, and `sequenced.gz` and `imputed.gz`.
- The coordinates of the ranges in `coverage.gz` are alignment positions not reference genome positions, i.e. they increase by 1 for every line not starting with "¿" in the alignment files.

7.5 Format of SNPs-By-Windows files

Each line in an `sbw.txt` file is of the form `str1 str2 chr block n0 p0 d0 n`
where:

- `str1` and `str2` are strain names (including `REF`).
- `chr` is the chromosome name.
- `block` is the block number: block N consists of the alignments to reference-sequence nucleotides $N*10000+1$ to $(N+1)*10000$ inclusive.
- `n0` is the number of places in block N where both strains had a sequenced nucleotide and the two nucleotides were different.
- `p0` is the number of places in block N where both strains had a sequenced nucleotide and *any* two strains had different nucleotide values.
- `d0` is the number of places in block N where both strains had a sequenced nucleotide.
- `n40`, `p40` and `d40` are as for `n0`, `p0` and `d0` respectively, but only nucleotides with a quality of 40 or more are considered.

The perl scripts `plotStrainSimilarities.pl` and `tabulateSNPsByWindows.pl`, available in `misc.tgz`, use this data to create similarity plots between sets of strains and various tables, respectively. The former requires the perl module `GD::Simple`, which can be downloaded from CPAN.

7.6 Format of recombination-modelling files

Note: a fuller description of Margarita input and output formats can be found in the documentation at

`http://www.sanger.ac.uk/Software/analysis/margarita/Margarita.pdf`

Within a `recombs` directory, `mgr_in` is the Margarita input file for a whole chromosome. Its first line is of the form `nS nM`, where `nS` is the number of strains aligned (including the reference) and `nM` is the number of markers: polymorphic positions that may contain information suggesting recombinations. After the first line, there are `nM` lines each containing an integer; the i'th such line gives the offset (in alignment coordinates) of the i'th marker.

There then follow `nM` lines each containing the line from `sequenced.gz` for the relevant marker; see Section 7.2.

Because the resource requirements of Margarita grow faster than linearly with the size of the data set, it is usually necessary to split it into (slightly overlapping) sets of markers. The files `mNNN/mgr_in` (where `NNN = 000, 001, ...`) result from doing this.

`mNNN/mgr_out` consists of the concatenated output of 25 Margarita runs on `mNNN/mgr_in`. The first four lines of each run output are a header starting with `%ARGINFERENCE`. There then follow lines denoting events in the ARG. The first field on each line is the time at which the event is deemed to have occurred; these times have no absolute reference, but their relative ordering is significant. The event types are recombination, coalescence, mutation and (sequencing) error, each denoted by its first two letters. The event-type code is followed by two (for `mu` and `er`) or three (for `co` and `re`) ARG node identifiers indicating the daughter and mother node(s) of the event. For `re` nodes only, the final field is the marker at which the recombination is hypothesized to have occurred. Markers are numbered from zero; ARG nodes are also numbered from 0, and the first `Ns` nodes represent the sequenced strains. The 0'th strain is the reference sequence, and subsequent strains are in the order in which they occur in the first line of `sequenced.gz`.

`args_bl` has a line for each marginal tree for each of the 25 ARGs. The format of each line is

```
low argID high totLen len0 mthr0 len1 mthr1 ... lenN mthrN
```

This specifies the marginal tree for ARG `argID` between markers `low` and verb!high! inclusive. The total length (in mutation units) of all the branches in the tree is `totLen`. The pair `lenI mthrI` then gives the relative length (as a fraction of `totLen`) of the tree branch leading up from node `I`, and specifies that the mother of node `I` in the tree is `mthrI`. (Leaf node identifiers here are as in `mgr_out`, but internal node identifiers are not related to those in `mgr_out`). Thus the `lenI` values should sum to one, modulo rounding errors, and there should be 2(`Nm`-1) of them, since a binary tree with `Nm` leaves has `Nm`-1 internal nodes, and there is no `lenI mthrI` pair for the root node of the tree.

Recombination frequency files (`recombinationFrequencies.txt`) can be used to look for possible recombination hotspots. They are derived from Margarita output, and may or may not closely reflect real recombination events. Each line in one of these files is of the form

```
startInAli endInAli startInRef endInRef meanFreq
```

`startInAli` and `endInAli` are the start and end offsets of a region in alignment coordinates. (Because `sequenced.gz` and `imputed.gz` start with a single header line, offset O is at line $O + 1$ of those files. Note that these values were incorrect in versions of the data available before 2008-02-29). `startInRef` and `endInRef` are the corresponding reference coordinates. When the region is entirely within an insertion into the reference, `endInRef` will be one less than `startRef`.

The meaning of the line that on average, over the 25 ARGs constructed by Margarita to support sequence imputation during the final PALAS iteration, `meanFreq` recombinations (per ARG) were hypothesized between (or at) the two offsets; Margarita does not make any more fine-grained than this. `meanFreq` can be more than one because there can be more than one recombination in the same interval (in different branches of the ARG).

7.7 Format of SNP list files

A typical entry in a SNP list file is:

```
AAGCGTTCCATGGCAAATATT chr05 152481
AAGGGTTTCACGGTACATATT chr05 173045 A4 T>C 44
ONMNPQQPPNMLKOOQQQQPQ
```

This is interpreted as follows. There is a polymorphism between the reference and (Sp) strain A4 at offset 152481 in the reference sequence of `chr05`, which corresponds to offset 173045 in the alignment (i.e. there are 173045-152841 inserted positions upstream of this point in the alignment). The polymorphism is at the middle base of the 21-base sequences shown: in this case, `T` in the reference becomes `C` in A4. The quality score on the A4 nucleotide is 44. The third line is the sequence of quality characters (see Section 8) for the A4 sequence; the `M` below the middle position corresponds to 44.

8 Nucleotide qualities

Nucleotide qualities are represented by single characters in various places, following the PHRED convention. Each character represents an upper bound

on the probability that the corresponding nucleotide value in the sequence line is wrong. The formulae relating a character c, with ASCII code between 33 and 126 inclusive, to a quality score q and then to an error probability p are:

$q = ord(c) - 33$

$p = 10^{-q/10} = e^{-(qlog_e 10)/10}$

$q = -10log_{10}p = -(10/log_e(10))log_e p$

$c = chr(q + 33)$ where the `ord` function returns the ASCII code of a character, and the `chr` function is its inverse. Thus for example, $c =' A'$ gives $q = 65$, giving a difference of 32, so $p = 10^{-3.2} \sim 0.00063$. The following perl script prints out this value for every character you give it:

```
perl -ne '{print exp(log(0.1)*0.1*(ord($_)-33)) . "\n"}'
```

Note that $q = 0$ and $q = 1$ give $p > 0.75$, which makes little sense because the specified nucleotide must then be less likely than some other nucleotide. These quality values are best interpreted as "don't know" markers.

As a special case in the SGRP data (not in the PHRED convention itself), the space character, whose ASCII code is 32, is given the interpretation $q \geq 127 - 33 = 94$, i.e. virtual certainty. This character is used for the reference sequence and whenever an error probability is calculated (during imputation) to be $10^{-9.4}$ or less. However, it is not clear that qualities of more than about 55 ($p \sim 3 \times 10^{-6}$) can really be distinguished from each other in the SGRP data, because some sources of error (principally alignment and mapping error) are not accounted for in them.

When quality scores for different ABI reads for the same strain are combined to give a (pre-imputation) consensus value for that strain, the following formulae, derived by assuming that forward-strand and reverse-strand alignments are independent and measuring the agreement between them, are used. Let v_i be the nucleotide values, q_i be the associated qualities for $i = 1...n$, and $Q_u = \sum_{v_i=u} q_i$.

When all the v_i's are equal to v, we use the consensus quality

$qc_v = round(55(1 - e^{-0.029Q_v}))$

where "round" rounds a value to the nearest integer. This flattens off consensus qualities at 55 no matter how much evidence there is for them (see the above remarks). It also slightly boosts low qualities; thus for example if there is a single read and it has quality 16, $q_c = 55(1 - e^{-0.029*16}) \sim 20$.

When there is a disagreement, let x be the nucleotide with the highest Q value and y have the next highest, ignoring any third or subsequent values

and breaking ties arbitrarily. Then we take x as the consensus value, and give it the quality

$qc_x = round(38.5(1 - e^{-0.029(Q_x - Q_y)}))$

Thus these formulae assume that reads are independent (it does not matter whether qc_v comes from one read with quality 20 or two with quality 10, say) but that low qualities should be boosted and high qualities should be reduced.

Illumina read qualities are combined within Maq (`http://maq.sourceforge.net`). ABI qualities (derived as above) and Maq-derived Illumina qualities are calculated fairly cautiously, by taking the maximum when they agree and the difference when they disagree.

Gap characters as well as nucleotides are given qualities in the alignments, since qualities are required for all symbols participating in the imputation process. We adopt a cautious and simple approach of giving all gaps in a run the minimum of the qualities of the two nucleotides between which they fall.

References

[1] Kellis M, Patterson N, Endrizzi M, Birren B, Lander ES: **Sequencing and comparison of yeast species to identify genes and regulatory elements**. *Nature* 2003, **423**(6937):241–254.

[2] Mullikin JC, Ning Z: **The phusion assembler**. *Genome research* 2003, **13**:81–90.

[3] Wei W, McCusker JH, Hyman RW, Jones T, Ning Y, Cao Z, Gu Z, Bruno D, Miranda M, Nguyen M, Wilhelmy J, Komp C, Tamse R, Wang X, Jia P, Luedi P, Oefner PJ, David L, Dietrich FS, Li Y, Davis RW, Steinmetz LM: **Genome sequencing and comparative analysis of Saccharomyces cerevisiae strain YJM789**. *Proceedings of the National Academy of Sciences of the United States of America* 2007, **104**(31):12825–12830.

[4] **Saccharomyces cerevisiae RM11-1a Sequencing Project, Broad Institute of Harvard and MIT**. *[http://www.broad.mit.edu]* 2008.

[5] Minichiello MJ, Durbin R: **Mapping trait loci by use of inferred ancestral recombination graphs**. *American Journal of Human Genetics* 2006, **79**(5):910–922.

[6] Carter D, Minichiello M, Durbin R: **Imputing missing DNA values and discovering SNPs from low-coverage sequencing** In preparation.