Supplemental Methods

Genomes

Seven yeast genomes feature in version 1.0 of YGOB (Fig. 1). The sources of these data and annotations are as follows:

The *Saccharomyces cerevisiae* genome was completely sequenced by an international consortium (Goffeau et al. 1997). The set of 5516 *S. cerevisiae* protein-coding genes used in YGOB was curated in our laboratory by S. Wong, D.R. Scannell and K.P. Byrne and is based on the *Saccharomyces* Genome Database (SGD) (Christie et al. 2004) release dated June 2004, except that (i) we excluded genes annotated as "hypothetical" in SGD, and (ii) we rejected a small number of other *S. cerevisiae* genes that failed the RFC test of Kellis et al. (2003) and for which we could find no experimental evidence, or that were rejected by Brachat et al. (2003). "Mouse-over" information that appears for *S. cerevisiae* genes in the YGOB display comes from SGD title lines.

S. castellii was sequenced to 4x coverage by Cliften et al. (2003) so the contigs from this species are not complete chromosomes. We loaded *S. castellii* data into YGOB in a two-step process. We first used the annotations and homology data provided by Cliften et al. (distributed through SGD), but we found that a significant number of genes were either not annotated or had incorrect coordinates in this distribution. We then completely re-annotated the *S. castellii* genome using an in-house annotation pipeline, but we retained the gene names used by Cliften et al. if the gene sequence was unchanged, and added suffixes to indicate modified or newly-discovered genes. We also merged some of Cliften et al.'s contigs into scaffolds in cases where two ends of the same gene were on different contigs. This reannotation of the *S. castellii* genome will be described elsewhere (D.R. Scannell and K.H.W., in preparation).

C. glabrata and *K. lactis* were sequenced completely by Dujon et al. (2004), and *A. gossypii* was sequenced completely by Dietrich et al. (2004). We used the original annotations for these genomes without modification. Mouse-over information that appears for these species in YGOB comes from the GenBank annotation tags for each gene.

K. waltii was sequenced to 8x coverage by Kellis et al. (2004), and the sequence consists of hundreds of contigs linked together into ten scaffolds corresponding to its eight chromosomes. The *K. waltii* tracks in YGOB represent the scaffolds rather than the contigs, which means that some genes that are missing from the *K. waltii* track in YGOB may not in fact be missing from the genome, if they lie in a gap between two sequence contigs within a scaffold. YGOB uses the original annotation of Kellis et al. We have noticed that intron-containing genes tend to be unannotated in this genome but have not attempted to correct the problem.

S. kluyveri was sequenced by Cliften et al. (2003) to 4x coverage. The sequence consists of relatively short contigs that have not been joined into scaffolds. YGOB uses the annotation provided by Cliften et al. and distributed through SGD. We did not try to improve the *S. kluyveri* annotation in the same way that we did for *S. castellii*.

Genome Editing

The first draft of the YGOB dataset was made in 2003 and included only the genomes of *S. cerevisiae, S. castellii* and *S. kluyveri*, with homology assignments taken directly from Cliften et al.'s (2003) annotation. *S. kluyveri* was included because at that time it was the only pre-WGD genome sequence available publicly, though its genome sequence is fragmented into a large number of relatively small contigs. We subsequently reannotated the *S. castellii* genome (see above), and revised the pillar structures accordingly. Following this the entire browser dataset was curated manually, with any singleton loci or regions of dubious syntenic alignment being examined by appraisal of BLASTP scores, synteny and phylogenetic trees, to decide if they should be placed in another (usually neighboring) pillar. The two pre-WGD species *K. waltii* and *A. gossypii* were integrated in early 2004 using automated bi-directional best hit (BDBH) BLASTP assignments of homology to *S. cerevisiae* (BLASTP cutoff E < 1e-5), followed by genome-wide editing. All singleton pillars in the two new genomes were checked at this stage and merged with other pillars where appropriate. Later in 2004 *C. glabrata* and *K. lactis* were integrated, again using BDBH BLASTP (E < 1e-5). The *C. glabrata* BLASTPs were done versus *S. cerevisiae*, and *K. lactis* BLASTPs versus *K. waltii*, to maximize the number of one-to-one mappings in each case. This was followed by a round of automated searching for instances of singleton pillars that could be merged with adjacent pillars on the basis of a BLASTP hit (E < 1e-5) to at least one gene in the adjacent pillar, provided there was no better hit and the assignment was also supported by the syntenic context.

Finally, our laboratory inspected the whole dataset by systematically browsing along each *K. lactis* chromosome, examining and editing pillars. We looked in particular for instances where a species (or a track from a post-WGD species) had a gene in a singleton pillar that was beside another pillar where the same species (or track) was absent. This manual step revealed many genes with very weak BLASTP hits (E > 1e-5) that are in fact orthologs or ohnologs on the basis of their chromosomal gene context, transcriptional orientation, and (often) protein length. Rather than being ignored these are perhaps some of the most interesting loci since they have most sequence, and likely functional, divergence.

There remain a few groups of neighboring pillars in YGOB where each species has a singleton gene that seem likely to be orthologs on the basis of their context and orientation, but where none of the genes has any BLASTP hit (E < 1) to any of the others. We have left these pillars unmerged until there is stronger evidence that these are orthologous but very rapidly evolving loci. Similarly, ultra-fast evolving genes such as *S. cerevisiae* YBR184W (Kellis et al. 2003), for which there is no evidence to assign homology, also appear as singleton pillars in YGOB.

Software & Algorithms

YGOB is a Perl (www.perl.org) driven application with a web interface. At its core is a cross-genome data structure representing gene homology as pillars of orthologous or paralogous relationships. A pillar corresponds to a set of homologous genes that can appear as one column in the browser. Each pillar can maximally contain ten homologous genes: two genes from each of the post-WGD species, and one from each of the pre-WGD species, but slots in these pillars can also be left empty. Importantly, the pillar does not contain any information about whether a gene in a post-WGD species "belongs" to Track A or Track B in the browser; that is decided later by a series of algorithms and will vary depending on the size of the window being considered and on which species are selected. The pillars are implemented using Perl's Storable module (www.cpan.org). Graphics are created using the GD package (www.boutell.com/gd).

The principal software design challenges faced in YGOB are encapsulated in four key algorithms that are essential to the browser accurately displaying and assessing gene order relationships within and among yeast genomes. Details of these four algorithms are outlined below. Computer code is available on request (kevin.byrne@tcd.ie).

Before the four algorithms described below are used, YGOB requires an initial in-focus gene around which information is processed. It gets the homology pillar for that gene and, ignoring synteny at this early stage, places the homologs into tracks for each genome (two tracks for each post-WGD species, and one track for each pre-WGD species). For each of the neighboring genes upstream and downstream, until the selected window size is filled, it calls each gene's homology pillar and places any homologous genes into the genome tracks. The algorithms thus begin with a set of unaligned non-syntenic tracks, with gaps between nearby genes in genomes other than the one under focus. It is worth noting that due to the unambiguous orthology and chromosomal alignment in the pre-WGD species, their tracks are almost finalized even at this point, though they will contain gaps that need opening.

Algorithm 1: Alignment of Chromosomal Segments

Alignment of chromosomal segments is not a challenge for the single tracked pre-WGD species (because homology unambiguously decides placement in most cases). The algorithm relates only to the placement of chromosomal segments from post-WGD species, where a segment could appear on either track but in terms of genome evolution should be on one in particular. The problem encompasses the accurate alignment of chromosomal fragments with respect to the other post-WGD species, in tandem with placement within their own genome. In practical terms this problem involves the evolutionarily accurate placement of data from the pillars onto tracks.

The algorithm (a simplified representation of which is shown in Figure S1) places each chromosome arm or contig alignment unit (AU), that features in a given browser generated genome space and has at least two genes present, onto Track A or Track B, and it does this for each post-WGD species. The algorithm is slightly different for *S. cerevisiae* than for the other post duplication species. The principal difference is the order in which AUs are placed onto Track A or Track B. The *S. cerevisiae* tracks are placed in descending order of length (in the captured genome space) while the other

post duplication species are placed after *S. cerevisiae* in order of the strength of an AU's clear agreement with the *S. cerevisiae* tracks. For each AU we calculate the degree to which it clashes directly at any locus with AUs already placed on Tracks A and B. A "clash" locus is any pillar containing two genes from the post-WGD genome we are aligning, where one gene belongs to an AU already placed on a track and the other belongs to the AU we are trying to place. These clear disagreements with already placed AUs are the strongest indicator of which track an AU should be placed on. If an AU clashes directly with neither track the algorithm examines whether placing it on a given track will force a yet-to-be-placed AU to clash with a track in the light of current alignments. If this too gives no signal then the tests whether placement of the AU on either track would cause interlacing between AU fragments and tries to avoid this if possible. In the rare case that there is no signal at all an AU will go onto Track A arbitrarily – for example the first AU being placed. One modification with the non-*S. cerevisiae* species is if there is no in-species clash signal, then the degree of alignment with *S. cerevisiae* Track A or B is used to decide placement. If a user decides not to use the *S. cerevisiae* genome then the first placed other post-WGD species takes its role.

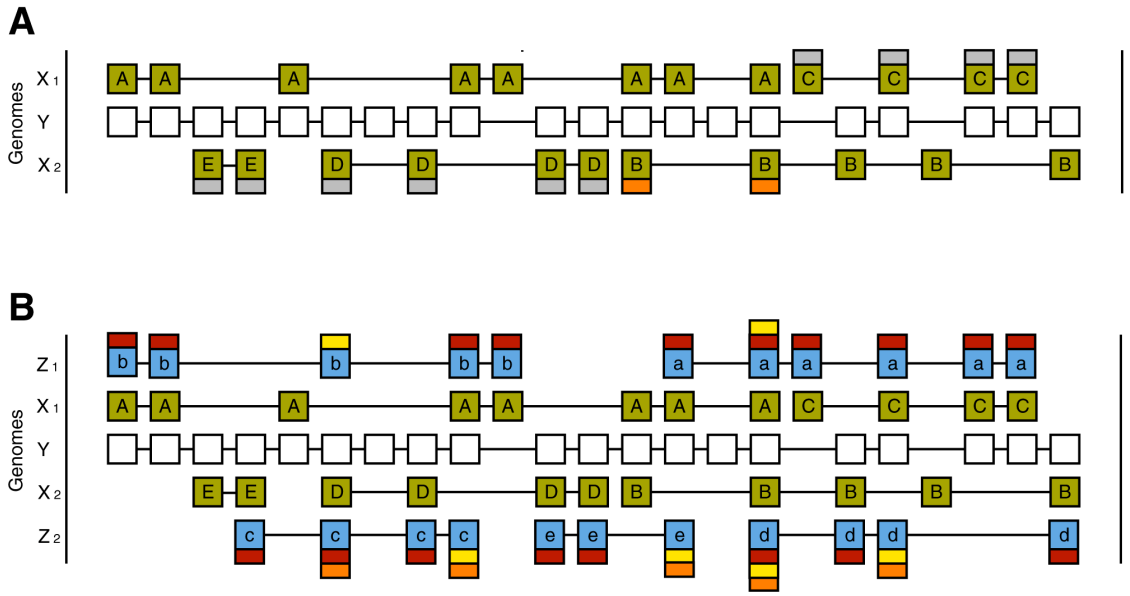

Figure S1 Simplified representation of the algorithm for track aligning. The central track Y is a pre-WGD genome. (*A*) For the first post-WGD genome X to be placed, alignment units (AUs, i.e. chromosome arms or contig fragments), are placed on the two tracks X_1 and X_2 (representing the 1:2 post-WGD pattern) in descending order of size. The largest AU, marked **A**, is arbitrarily placed on the top track. The next largest AU, **B**, is placed on the lower track because it clashes with the already placed **A** at two loci (marked by orange caps). The third AU, **C**, is placed on the top track because placing it on the bottom track would interlace (gray caps) it with the already placed **B** at every locus. The final two AUs, **D** and **E**, go on the bottom track because they interlace with **A** on the top track, again at every locus. (*B*) Once the first post-WGD genome has been placed, subsequent ones are aligned against it, with AUs placed in descending order of clear agreement with a track. This can be thought of as the difference between the number of red caps and yellow caps on each AU in tracks Z_1 and Z_2. AU **a** is placed first because it agrees at every locus with track X_1 and only at one locus with X_2. AU **b** has one locus that exclusively agrees with X_2 but clearly should be aligned with X_1. AU **c** goes on X_2 as it agrees with it at three loci and with X_1 only once, while AUs **d** and **e** are placed because of the clashes (orange caps) they have with already placed AUs from genome Z.

One other point to note is that some chromosomes have intra-genomic sister regions on the same arm (e.g. the left arm of *S. cerevisiae* chromosome IV, which contains both copies of “block 12” in Wolfe and Shields 1997) so it would be a mistake to assign such a whole chromosome arm to Track A or B. We identify AUs that are prone to this problem, either when they contain a “self-clash” pillar, i.e. if both slots in a pillar contain genes from the same AU, or when more than 30% of genes from that AU are relatively distant in their genome (>10 genes apart) but near in the browser (<5 pillars). These AUs then skip the AU alignment process and genes from them are placed on the fly as the tracks are filled.

Tracks are filled primarily with reference to the alignments that the above algorithm has assigned. However in the case of single genes not on an AU, they are placed on the fly so as not to clash with an aligned AU. In the case of "self-clash" AUs, genes are placed on the track that nearby genes from the "self-clash" AU need to go on, so as to be on the same track with each other, and not clash with any aligned AUs. Alternatively genes from such AUs may be placed on the track opposite that which the most distant genes from the "self-clash" AU need to go on so as not to clash with aligned AUs. These signals are sufficient to accurately place these problematic chromosomal fragments.

This algorithm is heuristic and was elaborated by exploring test cases and problematic regions of the genome. It embodies a set of rules whose end product is a visual presentation of syntenic regions from multiple post-WGD species assigned to tracks in a way that is identical to how we would assign them by eye.

Algorithm 2: Gap Opening

If on a genome track a gene is next to (though not necessarily in the next column as) another gene from the same chromosome and it is within five genes of the first gene, then we “open the gap” between these genes (Fig. S4). This is done by introducing into the browser track any intervening genes from that genome and simultaneously introducing into the other browser tracks any genes in those genomes that homology (as stored in the data pillars) brings into view as the gap opens. That is, we insert the intervening gene’s pillar into the browser at the gap point. The details as to how genes from post-WGD species are assigned to tracks is as in Algorithm 1.

Gaps in pre-WGD species’ tracks are opened first, followed by those in post-WGD species. The pillar being introduced can be either placed to the left or right of the gap, which will not matter in the track being opened, but the number of pillars separating the gap edges may be large and if opened at the wrong edge could incorrectly disrupt a chromosomal fragment on a different track. Therefore each genome’s track is examined with respect to any homologs in that genome that the new pillar is introducing. How these relate, in terms of genome position, to the genes already in the track guides the positioning of the pillar on the left or the right. In practical terms we look for the insertion side that is beside a pillar that contains genes adjacent to one or more genes in the pillar being inserted. The choice of edge is particularly unambiguous if it is flanked on both sides by such “adjacent” pillars. There is also a check to ensure that no pillar that is already displayed in the browser can be introduced a second time. Lastly gaps that are in fact inversions are not opened and the algorithm used to identify inversions is set out below.

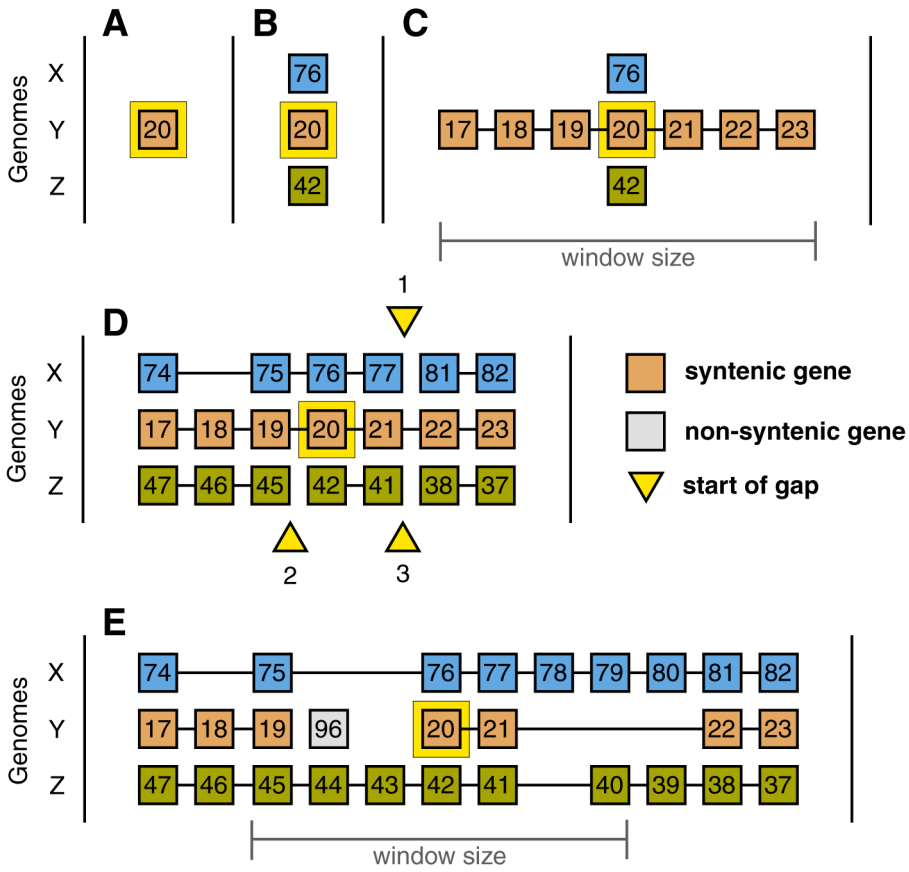

Figure S2 Schematic representation of the algorithm for gap opening. (*A*) The process begins with the gene being focused on, *Y20*. (*B*) *Y20*'s homology pillar is expanded. (*C*) Neighboring genes in genome Y are added, up to the selected window size of ±3. (*D*) When the homology pillars are expanded, gaps (denoted by yellow triangles) are visible in the genomes X and Z. (*E*) Gaps are opened creating a genome space with continuity in all genomes. YGOB will only display the section of this genome space corresponding to the selected window size of ±3 around the focused gene, *Y20*. The gene *Y96* is a homolog of the gene *Z44* that gap opening reveals, but it has no synteny with the rest of the genes visible in genome Y and so is colored gray. A gap in any genome at any locus means it has no homolog there.

Algorithm 3: Identifying Inversions

It was necessary to identify when a gap between genes that are not neighbors in a genome, but have no intervening genes between them in the browser display (and so would usually signify a gap opening), should not be opened due to this gap being an inversion. We open up the gap to the inversion itself, but not through it.

For each gene X, its order in its genome is represented by a number N_X. This simple counting number is used to help identify inversions. On a browser track imagine that we have a gene X with the gene W next on its left in the browser and the genes Y and then Z next on its right. So these genes appear on the browser track in the order W, X, Y, Z. Then, the gap between X and Y is the left end of an inversion if Z is situated between X and Y in their genome. If this is true, the product $(N_Y – N_X)(N_Y – N_Z)$ is greater than zero. Similarly the gap between X and Y is the right end of an inversion if W is between X and Y in their genome, and this is the case if the product $(N_X – N_Y)(N_X – N_W)$ is greater than zero.

For illustration, Figure 2B shows a three gene inversion in *S. cerevisiae* track A. The start of the inversion is between the genes *TMS1* and *ARX1* and is marked by orange connectors. In this example *TMS1* (YDR105C) corresponds to gene X, *ARX1* (YDR101C) to gene Y and *STE5* (YDR103W) to gene Z. As required for the left edge of an inversion YDR103W (Z) is located between YDR105C (X) and YDR101C (Y) in the *S. cerevisiae* genome and if we use the genes' systematic numbers as a proxy for order number in the genome, it is clear that the product (101-105)(101-103) is greater than zero.

It is worth noting that an inversion can only be identified in the syntenic context of the other genomes and that is a relative object. An inversion seen in one species P when the gene-in-focus is in species Q will be an equivalent inversion in Q if we focus on P.

Algorithm 4: Syntenic Scoring

To carry out genome-wide analyses to study the fate of genes in different lineages, we developed a system for scoring the syntenic status of a locus in a track. By syntenic status we mean whether the gene is present or absent in the track, and whether the arrangement of nearby genes is conserved between the track under study and other tracks. Pseudocode outlining the implementation of the algorithm can be seen in Figure S3.

A locus is considered to have a *syntenic presence* in a given track, if a gene is present in that track at that locus, and it has a gene on that track on at least one side, that is from the same chromosome and is within 20 genes in their genome and within *n* pillars in the browser (where *n* is a parameter as described below). A syntenic presence is scored as a "1". A non-syntenic presence, i.e. a gene that is present but out of syntenic context, is scored as a "!".

A locus is considered to have a *syntenic absence* in a given track, if no gene is present and there are bracketing genes on the chromosome either (i) within 20 genes in their genome, and both within *n* pillars of the locus to be scored in the browser, or (ii) within 5 genes in their genome, and within 2*n* pillars of each other in the browser, but with no intervening genes from another chromosome, or (iii) within 2 genes in their genome, and within 2(*n*+1) pillars of each other in the browser, but again with no intervening genes from another chromosome. A syntenic absence is scored as a "0". A non-syntenic absence (i.e., a gene that is absent but where the syntenic context does not meet any of these criteria) is scored as a "?".

We found *n*=6 to be the optimal value of *n* that maximized the number of scorable loci without allowing misassignment of synteny, and used this for the analyses in Table 4. This was found by manually examining (in the YGOB visual browser) the loci assigned to the 1:1 paralog and 2:1 classes between *S. cerevisiae* and *S. castellii* as *n* was increased. *n*=6 was the highest cutoff that returned no false positives when these two loss classes were scored against the two scaffolding pre-WGD genomes (*K. waltii* and *A. gossypii*) available at the time the optimization was carried out.

```
$score; # variable that will hold syntenic score
$n = 6; # optimised as explained above

ForEach pre-WGD genome = $pre {
 ForEach gene = $g {
  run YGOB focused on $g with window of 20 # generate syntenic YGOB track arrays
  ForEach post-WGD genome = $post {
   For A and B tracks = @track {

# the post-WGD track has a gene at its track position in $g's pillar

    If element of @track at the $g pillar is Defined = $f {
     $test=0;
     ForEach other element of @track = $e {
      If $e and $f are on the same chromosome
      And $e and $f are within 20 genes in their genome
      And $e and $f are within $n pillars in YGOB {
       $test=1; last;
      }
     }
     If $test==1 { $score = "1"; } # syntenic presence
     Else        { $score = "!"; } # non-syntenic presense
    }

# the post-WGD track has no gene at its track position in $g's pillar

    ElseIf element of @track at the $g pillar is UnDefined {
     $test=0;
     ForEach element of @track from $g pillar to its right edge = $e {
      If $e is Defined
       ForEach element of @track from $g pillar to its left edge = $d
        If $e and $d are on the same chromosome
        And (1) within 20 genes in their genome And both within $n pillars of $g in YGOB
        Or  (2) have no genes in between them from another chromosome
            And (2.1) are within 5 genes in their genome And 2*$n pillars in YGOB
            Or  (2.2) are within 2 genes in their genome And 2*($n+1) pillars in YGOB {
                $test=1; last; last;
        }
       }
      }
     }
     If $test==1 { $score = "0"; } # syntenic absense
     Else        { $score = "?"; } # non-syntenic absence
    }
   }
  }
 }
}
```

Figure S3 Pseudocode for Algorithm 4: Syntenic Scoring.

References for Supplemental Methods

Brachat, S., F.S. Dietrich, S. Voegeli, Z. Zhang, L. Stuart, A. Lerch, K. Gates, T. Gaffney, and P. Philippsen. 2003. Reinvestigation of the *Saccharomyces cerevisiae* genome annotation by comparison to the genome of a related fungus: Ashbya gossypii. *Genome Biol* **4:** R45.

Christie, K.R., S. Weng, R. Balakrishnan, M.C. Costanzo, K. Dolinski, S.S. Dwight, S.R. Engel, B. Feierbach, D.G. Fisk, J.E. Hirschman et al. 2004. *Saccharomyces* Genome Database (SGD) provides tools to identify and analyze sequences from *Saccharomyces cerevisiae* and related sequences from other organisms. *Nucleic Acids Res* **32:** D311-314.

Cliften, P., P. Sudarsanam, A. Desikan, L. Fulton, B. Fulton, J. Majors, R. Waterston, B.A. Cohen, and M. Johnston. 2003. Finding functional features in *Saccharomyces* genomes by phylogenetic footprinting. *Science* **301:** 71-76.

Dietrich, F.S., S. Voegeli, S. Brachat, A. Lerch, K. Gates, S. Steiner, C. Mohr, R. Pohlmann, P. Luedi, S. Choi et al. 2004. The *Ashbya gossypii* genome as a tool for mapping the ancient *Saccharomyces cerevisiae* genome. *Science* **304:** 304-307.

Dujon, B., D. Sherman, G. Fischer, P. Durrens, S. Casaregola, I. Lafontaine, J. De Montigny, C. Marck, C. Neuveglise, E. Talla et al. 2004. Genome evolution in yeasts. *Nature* **430:** 35-44.

Goffeau, A., R. Aert, M.L. Agostini-Carbone, A. Ahmed, M. Aigle, L. Alberghina, E. Allen, J. Alt-Mörbe, B. André, S. Andrews et al. 1997. The Yeast Genome Directory. *Nature* **387 (Suppl.):** 5-105.

Kellis, M., B.W. Birren, and E.S. Lander. 2004. Proof and evolutionary analysis of ancient genome duplication in the yeast *Saccharomyces cerevisiae*. *Nature* **428:** 617-624.

Kellis, M., N. Patterson, M. Endrizzi, B. Birren, and E.S. Lander. 2003. Sequencing and comparison of yeast species to identify genes and regulatory elements. *Nature* **423:** 241-254.