KU LEUVEN

FACULTY OF BIOSCIENCE ENGINEERING
FACULTY OF ENGINEERING SCIENCE
FACULTY OF MEDICINE
FACULTY OF SCIENCES

Detecting copy number variations in industrial yeast: a novel algorithm that corrects for bias in read depth

Copynumber detectie in industriële gist bij hoge vertekening in read depth

Promoter:
Prof. Kevin Verstrepen
Department of Microbial and Molecular Systems
Centre of Microbial and Plant Genetics

Dissertation presented in
Fulfillment of the requirements
for the degree of Master of Science in
Bioinformatics

Adriaan Merlevede

June 2015

Abstract

Saccharomyces cerevisiae is a key model organism for genetics and cell biology, a major agent in several culturally and economically important industries, and has more been implicated as human pathogen. In order to understand its biology and ultimately produce superior industrial strains, several recent studies have examined the population structure and evolutionary history of this fascinating microorganism. However, these studies have focused on strains isolated from mainly wine, sake and natural environments. In this project, we have sequenced 197 yeast strains from a variety of industrial sources at high coverage (mostly $100X$-$300X$). The goal is to obtain a broad insight in the population genomics of industrial yeasts with a focus on beer, which has previously been poorly documented in genomics studies.

In this context, the present dissertation attempts to illuminate the diversity in copy number variation (CNV) among industrial *S. cerevisiae* strains. However, conventional means for CNV detection have proven ineffective in a number of our samples. The reason for this is a peculiar bias of read depth in which the number of sequenced reads gradually decreases further away from the terminal ends of each chromosome. In order to still use these samples, I developed a novel, fully non-parametric algorithm, Splint, that is capable of detecting and correcting gradual read depth bias. Visual inspection and comparison with a commercially available CNV detection software, Nexus, shows the viability of Splint's results.

Splint reveals that beer yeasts are by far the most copy number variable *S. cerevisiae* group in this study. Most beer yeasts appear to be tetraploidy and show extensive aneuploidy.

Surprisingly, many CNVs that have been described in literature as industry-specific are more linked to shared evolutionary history than source environment. In addition, no evidence of convergent evolution was observed when comparing copy number variation profiles of strains from the same industry but different lineages. I conclude that most CNV in industrial *S. cerevisiae* strains is not a short-term adaptation to a specific industrial environment. As an alternative, I propose that copy number adaptation in industrial yeast is more related to its effects on long-term evolvability, than to short-term changes in phenotype.

Nederlands

De gist *Saccharomyces cerevisiae* heeft een rijke geschiedenis met menselijke associatie, vooral in industriële fermentatie en als modelorganisme in de biologie. In de laatste jaren is de interesse van de wetenschappelijke gemeenschap in de populatiestructuur en evolutionaire geschiedenis van deze gist sterk gestegen. De meeste studies hebben zich toegespitst op gisten uit de wijn en sake industrie en op natuurlijke isolaten. Het Verstrepen lab heeft in dit project de volledige genomen uitgelezen aan hoge diepte ($100X$-$300X$) van 197 gisten uit verscheidene industriële processen. Het doel is om een breder inzicht te krijgen in deze populatie, met de focus op biergist.

In deze context probeert deze verhandeling een blik te werpen op de copynumber diversiteit van industriële *S. cerevisiae* stammen. Conventionele technieken voor detectie van copynumber variaties bieden evenwel geen grip op een groot deel van onze datastalen, vanwege een unieke afwijking tijdens het sequencing proces. Daarom heb ik zelf een nieuw, volledig niet-parametrisch algoritme ontwikkeld, Splint (*Engels*: spalk) dat in staat is om copynumber variaties te detecteren zelfs wanneer er een continue vertekening aanwezig is in de *read depth*. Visuele inspectie en vergelijking met commerciële software, Nexus, illustreert de bruikbaarheid van Splint.

Data-analyse toont dat biergisten behoren tot de meest variabele groepen van *S. cerevisiae*. De meeste biergisten zijn polyploid en aneuploid voor meerdere chromosomen.

Veel copynumber variaties die eerder zijn beschreven in de literatuur als mogelijke adaptaties aan specifieke industrieën blijken perfect te verklaren door evolutionaire geschiedenis. Er werd geen convergente evolutie geobserveerd tussen stalen uit verschillende groepen. Dit suggereert dat, in tegenstelling tot meerdere andere wetenschappelijke rapporten, de meeste copynumber variaties in industriële gist geen gevolg zijn van adaptatie op korte termijn.

Vulgarizing

Yeast is a tiny unicellular organism. Despite its size, it is responsible for the leavening of bread, the fermentation of wine and beer, countless discoveries in science and even some diseases. Currently, it is unknown how some yeasts evolved to help us make beer, and others evolved to make us sick. One piece of the puzzle may be the cutting, copying and pasting of pieces of DNA.

While modern technology allows scientists to read out a cell's DNA, finding these cut and copied parts can still be difficult. This is because the DNA is not read all at once, but in many small random pieces. Some pieces may be read many times, and others may not be read at all. Using statistics, we can find out how many times a gene probably occurs in the DNA by counting these pieces. However, in some cases, the number of times that a piece of DNA is read is not random. Part of this

thesis describes a computer program that calculates how many times each gene occurs in a yeast's DNA, even if the pieces are not read randomly.

Using this program, which is called Splint, we could compare the number of gene copies between different kinds of yeast. Usually, having multiple copies of a gene is thought of as a way to increase the effect of that gene. For example, wine or beer yeasts should have more copies of a gene that helps to produce alcohol, than bread or chocolate yeasts. Surprisingly, Splint showed that there are many beer and wine yeasts with few alcohol genes, and others with many. The fact that there is no link between the number of gene copies and how much the yeast uses that gene, indicates that the evolution of cutting and pasting genes may be more complicated than previously thought.

Preface

I would like to thank, first of all, my supervisor Brigida Gallone, for giving me the opportunity to start, and the space to explore this interesting project. Thank you for guiding me on this odyssey.

A word of thanks goes out to my promotor, Kevin Verstrepen, for supporting this project and for being a wonderful/merciless critic during group meetings. Thanks to Rob Jelier, Tom Wenseleers and you for taking the time to be on my jury.

Then I would like to do a shout-out to all of the wonderful people of the Verstrepen lab. You were always helpful and approachable. Also thank you for staying awake through a portion of my presentations. Sympathy goes especially to my student colleagues for sharing the bittersweet symphony that is the thesis year.

Ten laatste, aan allen die mijn thesis gelezen hebben, proficiat. Aan allen die mijn thesis niet gelezen hebben, ook proficiat, maar ietsje minder.

Contents

Part 1

Background

This background section consists of three parts. The first part is an introduction to yeast and its use in industry, as well as the current state of the art in some areas of yeast population genomics research. The second part contains an overview of copy number variation detection methods and previously reported observations of CNV in yeast. The third and final part introduces the intuition and theory of smoothing splines.

1.1 Yeast

Yeast is a loose term used to describe a wide and varied group of unicellular fungi (Basiciomycota, Ascomycota). Mutualism between humans and yeasts is probably as old as sedentism, and yeasts continue to be heavily used in modern fermentation industries, including the production of bread, beer, wine, cocoa, bioethanol, sake and various high-alcohol beverages collectively called ‘spirits’. Some yeasts are also known as pathogens for humans or crops. Outside of industry and medicine, yeast functions as one of the most popular model organisms in research, specifically in genetics and cell biology, because it is easy to grow and control while also offering the full complexity of a eukaryotic cell. In the last 50 years, approximately 1% of research articles published in life sciences have featured yeast, according to Pubmed (*Pubmed* n.d.).

By far the most studied species of yeast is *Saccharomyces cerevisiae*. In 1996, this species became the first fully sequenced eukaryote (Goffeau *et al.* 1996). It is also the main agent used in winemaking, sake, baking and brewing, the largest and most culturally relevant fermentation industries. *S. cerevisiae* is so well-known that it is often referred to as simply ‘yeast’.

Ecology

The industrial importance of yeast stems from its ability to convert sugars into alcohol (ethanol, CH_3CH_2OH) and carbon dioxide (CO_2) in a process called ethanol fermentation. Fermentation, in its broadest interpretation, is any anaerobic biochemical process that metabolizes organic molecules and provides energy for the cell, but in the context of yeasts or its related industries, fermentation is often used as a synonym for alcoholic fermentation.

The notable properties of *S. cerevisiae* are ethanol tolerance and the Crabtree effect Fermentation usually occurs only in anaerobic conditions, as they typically produce compounds that can be further metabolized by oxidation in the presence of oxygen. The exception to the rule is yeast, in particular *S. cerevisiae*, which is able to perform fermentation of glucose even in aerobic conditions. This is called the Crabtree effect. Despite the fact that alcoholic fermentation is a much less efficient use of sugar, it does allow faster growth than aerobic respiration and thus may give Crabtree-positive yeasts a growth advantage in sugar-rich media. Alternatively, or simultaneously, the Crabtree effect may have evolved as a mechanism to disrupt the growth of other species competing for the same sugar resource that are less tolerant against the alcohol and heat stress resulting from fermentation (Goddard 2008; Piškur *et al.* 2006). Or the Crabtree effect might not be adaptive at all, as suggested by the nomad model (see below) (Goddard & Greig 2015). Regardless, the Crabtree effect combined with high ethanol-tolerance made *S. cerevisiae* highly successful in conditions set up by prehistoric humans for fermentation of alcoholic beverages from juice or grain, starting a long tradition of association between humans and yeast.

The ecological niche of *S. cerevisiae* is unknown When considering the extensive interest in *S. cerevisiae* as both a key model organism and industrial agent, the lack of knowledge of the natural occurrence and history of this species is striking (Liti 2015). Until relatively recently, it was unknown whether *S. cerevisiae* is a completely domesticated species that had spawned several wild and pathological branches (Mortimer & Polsinelli 1999), or if it is a naturally occurring species merely interacting with humans and retaining an unknown degree of variation in the wild (Martini 1993). It has since become clear that *S. cerevisiae* is in fact a natural species occurring world-wide, albeit with several mostly domesticated lineages (Cromie *et al.* 2013; Fay & Benavides 2005; Liti, Carter, *et al.* 2009; Wang *et al.* 2012). Two or three domesticated lineages have been proposed (Fay & Benavides 2005; Liti, Carter, *et al.* 2009; Schacherer *et al.* 2009), one originally used in ancient sake fermentation, one associated with winery, and possibly a third lineage of lab strains.

The widespread natural occurrence of *S. cerevisiae* is now widely recognized, but its natural ecological niche is still unknown, as are its distribution, interaction with other microorganisms and its life cycle (preferred type of reproduction) in different habitats (Liti 2015). While in the past it has often been considered an opportunistic fermenter based on its abundance in human-made fermentation environments, there is no evidence that it occupies this same ecological niche in nature. According to Goddard & Greig's Nomad model, *S. cerevisiae* should be considered a widely spread generalist, as a scientifically neutral assumption until more data become available (Goddard & Greig 2015). *S. cerevisiae* has been isolated from a wide range of habitats, including fruit (Mortimer & Polsinelli 1999), insects, soil, several plants (particularly oak bark, although this may be due to research bias (Goddard & Greig 2015)) and the human gut. Some studies have also indicated that, aside from humans, insects are an important niche for yeast, and play a major role in their dispersal (Christiaens *et al.* 2014; Stefanini & Dapporto 2012).

Geno- and phenotypic diversity

An important yeast-related topic in recent years has been to unravel the complex ecology and history of *S. cerevisiae.* Insight in its natural habitat, ecological niche(s), evolutionary history and population structure is key to understanding the biology of this model organism, as well as developing superior strains for industrial use.

Five *S. cerevisiae* lineages are currently recognized in scientific canon, plus many mosaic strains, while Asia harbors several highly diverse natural groups Population structure, or population stratification, is the presence of genotypic subgroups within a population, caused by clonal growth or (partial) isolation in mating behavior and subsequent differential drift or selection. The structure of a population and its phylogeny are crucial to elucidating its evolutionary history. In *Saccharomyces paradoxus*, the closest relative of *S. cerevisiae* which has been isolated in similar habitats but is not associated with human activity, the population is composed of three different lineages separated by geography (Liti, Barton, *et al.* 2006). In *S. cerevisiae*, population structure remains more elusive.

In 2009, five world-wide *S. cerevisiae* lineages have been identified, termed Malaysian, West African, North American, Sake and Wine/European (Liti, Carter, *et al.* 2009). These five groups have since become a point of reference for coordinating the complex population landscape of this species (Borneman *et al.* 2011; Ramazzotti *et al.* 2012; Warringer *et al.* 2011). Several studies have added more resolution and/or new lineages to the currently known population structure (Cromie *et al.* 2013; Stefanini & Dapporto 2012). One study in particular showed a large natural diversity among strains collected across China, including some primeval forests that are far away from daily human activity (Wang *et al.* 2012). Some of these samples clustered in lineages that appear to have diverged from the previously studied groups over 20000 years ago, suggesting a geographic origin of *S. cerevisiae* in Asia (Liti 2015; Wang *et al.* 2012). It also shows that, despite earlier reports based mainly on European and industrial strains (Aa *et al.* 2006; Liti, Carter, *et al.* 2009), *S. cerevisiae* is a diverse species adapted to various niches worldwide irrespective of human activity.

An important finding in all of these studies is the presence of many strains that do not entirely belong to any of these lineages. The genome of these ‘mosaic’ strains contains several regions that individually cluster with different groups. The admixture of these mosaics is not uniform, with many mosaics containing genetic material from the European and Sake lineages as well as strains used in baking and clinical isolates, indicating that dispersal and outcrossing of geographically isolated strains through human activity may have played a major role in the evolution of these mosaic groups (Cromie *et al.* 2013; Liti, Carter, *et al.* 2009). The mosaic strains also contain a relatively high (around 24% (Liti, Carter, *et al.* 2009) to 38% (Cromie *et al.* 2013)) amount of unique polymorphisms, which may find their origin in lineages that are as of yet overlooked in population studies.

Beer yeast is mostly comprised of two separate lineages Some groups of *S. cerevisiae* have remained poorly documented. One such group is beer yeasts. So far, only a few ale strains have been studied from a population genomics perspective. It was noted that they appeared to form a separate lineage close to the Wine/European lineage (Schacherer *et al.* 2009). As part of this project, many *S. cerevisiae* ale and lager strains, were sequenced and a phylogenetic tree was inferred (Figure 1.1). This phylogeny shows that beer strains are largely composed of two separate lineages. One group (Beer 2) is closely related to the European/Wine group, the other (Beer 1) is more distant.

It is unclear whether ecology/industry or geography is more related to genotypic and phenotypic variance One of the core questions in current yeast population genomics is the relative importance of ecological/industrial niche and geographic location. This is tightly related to the extent of human activity on the evolution of *S. cerevisiae*, and thus the degree of domestication.

Some authors propose that the industrial lineages may be largely domesticated and may have undergone extensive selective evolution to adapt to fermentative industry, *i.e.* stratification of these groups is most related to industry (Aa *et al.* 2006; Legras *et al.* 2007; Stefanini & Dapporto 2012; Warringer *et al.* 2011). The Wine and Sake lineages appear to be mostly associated with human industry and may reflect a considerable degree of domestication (Fay & Benavides 2005; Schacherer *et al.* 2009). This is evidenced by the dispersal of wine yeast, which, unlike wild yeast, is unconstricted by geographical distance. The same is true to a lesser extent in the Sake lineage. Strikingly, the Sake lineage appears to be more closely related to the North American cluster than to any of the observed Asian groups, although some wild Asian isolates have also been clustered to the Sake group, as well as to the Wine/European group (Wang *et al.* 2012).

As an alternative to this hypothesis of partial domestication, humans may have simply used and transported existing strains which already had the right properties for large-scale fermentation. In this case, the population structure of *S. cerevisiae* would reflect geography (Cromie *et al.* 2013; Knight & Goddard 2015; Liti, Carter, *et al.* 2009).

A related question is whether ecology or geography is more related to phenotypic variation. It is known that phenotypic variation strongly relates to evolutionary history (Warringer *et al.* 2011). Genotypic variation in *S. cerevisiae* is small in com-

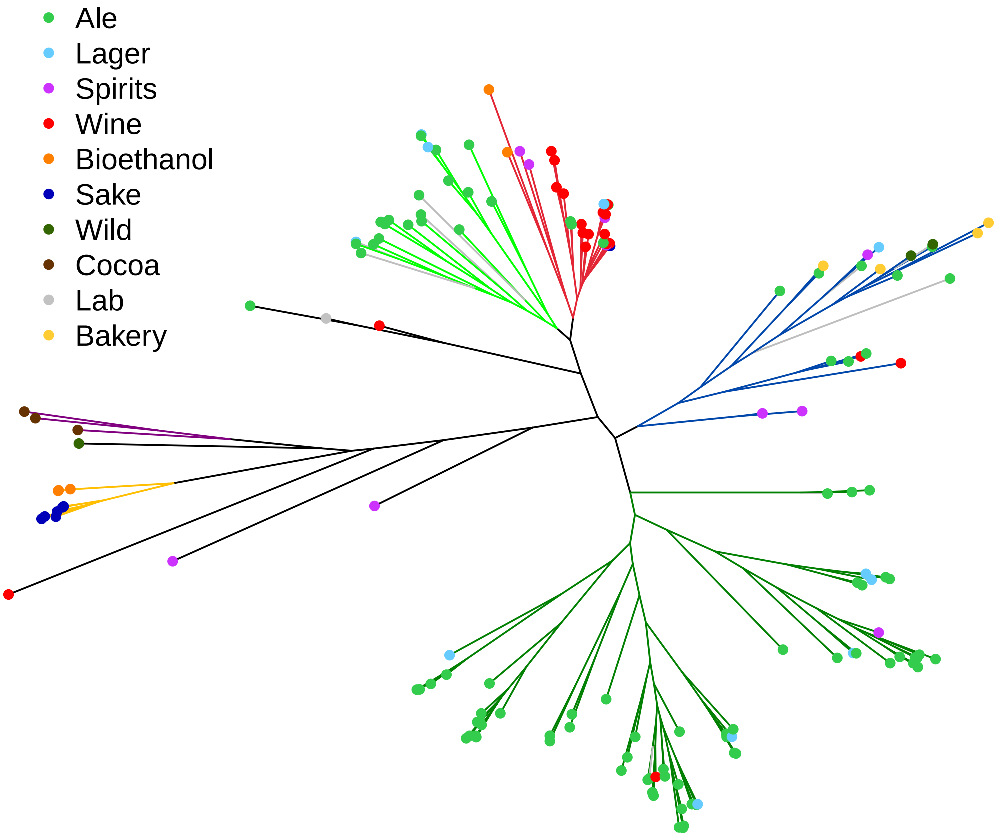


Figure 1.1: Phylogenetic tree constructed as part of this project. Tips are colored by industry, branches are colored according to a division in six main lineages: Sake (yellow), Cocoa (purple), Admixed (blue), Beer 1 (dark green), Wine/European (red) and Beer 2 (light green). Wild strains are scattered and are colored black. Grey branches lead to 'jagged' samples that were not used for this thesis (see Introduction). Most mosaic strains are grouped together in the Admixed lineage. The only lab strain used in this project is S288C (gray).

parison to related species such as *S. paradoxus* (Wang *et al.* 2012), but phenotypic variation is greater in *S. cerevisiae* (Liti, Carter, *et al.* 2009; Warringer *et al.* 2011). This may be interpreted as a consequence of tight population bottlenecks, possibly as a consequence of domestication, selection and dispersal of a small initial population by humans (Warringer *et al.* 2011).

S288C reference genome

In the last decade, there has been considerable interest from the scientific community in *S. cerevisiae* as a species. However, most yeast research in the past and in the present has focused on only a few lab strains. In order to allow direct comparison of results from different labs without confounding strain differences, most of the yeast community uses derivatives of the same reference strain, S288C. While many assemblies of different *S. cerevisiae* genomes are now available, the S288C reference genome is not only the oldest, but also one of the most well annotated and high-quality genome assemblies of any eukaryote (Goffeau *et al.* 1996). Several of the yeast community's most powerful tools, including extensive libraries for deletion and overexpression, are also based on an S288C genomic background.

S288C is composed mostly of EM93, an isolate from a rotting fig S288C was bred by Mortimer as a background for the isolation of biochemical mutants. Its use quickly became widespread throughout the scientific community, because it is easy to manipulate. It does not flocculate and has few nutritional requirements: S288C can grow on minimal media containing only biotin, nitrogen, glucose and trace elements. For the construction of S288C, Mortimer used other lab strains, EM93-1CC and S177A. Both of these strains are descendants of EM93, which was isolated in 1938 by Emil Mrak from a rotten fig in California Central Valley. It was subsequently obtained by Lindegren, and popularized for use in the laboratory. Through a complex lineage, approximately 90% of the S288C genome originates from EM93 (Engel *et al.* 2014; Mortimer & Johnston 1986). Other than that it was isolated from a rotten fig, the ecological background of EM93 is unclear. It may have been a wine strain carried from a nearby winery to the fig by an insect (Aa *et al.* 2006; Mortimer 2000), although this seems less likely as more research reveals the abundance of yeast in the wild. S288C and other wild strains are not clustered in the Wine/European lineage (Liti, Carter, *et al.* 2009; Schacherer *et al.* 2009; Stefanini & Dapporto 2012) (Figure 1.1).

S288C is an established reference, but is also an atypical yeast strain Despite its usefulness, the use of reference strain S288C has also received some criticism. Prolonged use in the lab has isolated this strain from the natural or industrial environment, introducing different evolutionary pressure. S288C is, in fact, one of the most phenotypically deviant yeast strains (Kvitek *et al.* 2008; Warringer *et al.* 2011). This makes extrapolation of results obtained in the lab to natural or industrial environments dubious, especially when considering the stratification of this species. Moreover, using the genome of this single strain as a reference obscures genes and other genomic regions that are not present in S288C (Borneman *et al.* 2011). Some of these regions influence industrially relevenant phenotypes, and they are typically more variant (Dunn, Richter, *et al.* 2012).

1.2 Fermentation

Fermentation is a central process in a variety of industries, particularly those producing alcoholic beverages, but also, among others, bread, cocoa, cheese and yoghurt. While most, but not all, of these are fermented by the same species, *S. cerevisiae*, the strains used are generally different, corresponding to the different fermentative conditions. This section contains a description of beer, wine and other fermentation.

Beer

Beer is made primarily from water, barley (*Hordeum vulgare* L.), hops (female flowers of *Humulus lupulus*) and fermented by yeast. The barley is made to germinate by soaking in water, and later dried to stop the germination, yielding half-germinated seeds called malt. During this process, various enzymes are released that break down complex sugars after the malt is milled and mashed with water. The resulting liquid is called wort. Wort is strained and boiled together with hops and possibly starch and/or flavor additives. Hops can be seen as providing the bitter flavor of the final beer, while carbohydrates and their products after fermentation make up the body. After the boiling, solid chunks are removed from the wort and yeast is added to start fermentation. The liquid can then be called beer. In many cases, the beer is left to age before it is ready for consumption, possibly in a different tank. This is called conditioning or secondary fermentation. During conditioning, more complex, less easily consumed compounds are broken down, which makes the finished product taste and smell more rounded.

The main sugars that are released from malt are glucose (a monosaccharide), maltose (a disaccharide) and maltotriose (a trisaccharide). The other, higher sugars are collectively called maltodextrin. These are the main sugars present during beer fermentation, but other sugars may also be present to varying degrees when adjuncts are used. The main stressors during beer fermentation are oxidative and osmotic at the beginning of the process, shifting towards nutritional and ethanol stress near the end (Gibson, Lawrence, *et al.* 2007).

The two types of beer, ales and lagers, differ in fermentative conditions and yeast While the general brewing process described above is the same for all beers, two types can be distinguished based on their fermentative conditions. The oldest extant type of beer, aleis fermented at higher temperatures (12-24°C), whereas lagers are fermented at lower temperatures (3-10°C). Generally, the higher temperatures in ales allow for the production of more esters which give the beer a heavier taste and fruitier aroma, whereas lagers often have a cleaner taste and lower alcohol percentage. Lager fermentation is also slower (1-3 weeks, as opposed to 2-5 days for ales) and is typically followed by a long secondary fermentation, also at low temperature, in which sulfur-based off-flavors produced during the primary fermentation are broken down. Lager conditioning is also called lagering, from the German word *lagern* (*German*: to store in a cool place).

The different conditions during fermentation are linked to the use of different yeasts. Ales are typically fermented with various strains of *S. cerevisiae*. Lagers are typically fermented with *Saccharomyces pastorianus*. *S. pastorianus* is an allotetraploid hybrid yeast originating from, on the one hand *S. cerevisiae*, possibly a brewing strain, and on the other hand *Saccharomyces eubayanus*, a cryotolerant yeast that is often found as a contaminant in beer fermentation (Libkind *et al.* 2011). Its genetic lineage has been interpreted as the indirect cause of its combined fermentative power (*S. cerevisiae*) and cryotolerance (*S. eubayanus*) (Gibson & Liti 2015).

Typically, ale yeasts float at the top of the wort after fermentation, whereas lager yeasts sink to the bottom, hence they are also called top- and bottom-fermenting yeasts and beers. While *S. cerevisiae* and *S. pastorianus* are often cited as 'the' ale and lager yeasts, *S. cerevisiae* is also used in lager, *S. pastorianus* is used in ale and some beers are fermented by other species and hybrids (Pope *et al.* 2007). In fact, some of these cases have been observed when collecting samples for this project, often unbeknownst to the brewers.

A small third group of beers can be distinguished that is neither ale nor lager. Lambic beers are a small collection of Belgian beers that are fermented in open vats, without adding purified yeast culture. Consequently, Lambics are fermented entirely by wild yeasts, giving them a characteristic flavor and aroma. Such wild fermentation is highly complex and caused by many different yeast strains, but a major contributor is *Brettanomyces bruxellensis*, *a.k.a. Dekkera bruxellensis*. Despite its name, yeasts of this species are found worldwide and are known contaminants in other beer and wine fermentation, where the idiosyncratic 'brett' aroma is unwanted. *B. bruxellensis* grows best in the temperature range of 19-35°C and has very high alcohol tolerance, allowing it to thrive during and after alcoholic fermentation by other yeasts such as *S. cerevisiae* (Curtin & Pretorius 2014; Schifferdecker *et al.* 2014).

Ale yeasts are genotypically and phenotypically complex Despite the cultural and economic relevance of the brewing industry, little attention has been given in recent scientific literature to the genomics of ale yeast. Ale yeasts are known to be complex in terms of genome structure (Johnston 1990)[1]. They are also more phenotypically variant than *S. pastorianus* used in lager brewing (Gibson & Liti 2015).

Wine

Wine production starts by crushing grapes into must, a sully made of grape juice, skins, seeds and sometimes stems. The skins contain tannins, anthocyanins and other flavor compounds that are leached into the juice in a process called maceration, before the juice is separated from the solid part of the must by pressing. The duration of maceration determines the type of wine, with white wines being macerated for several hours and red wines up to several weeks. As a result, yeast fermentation in red wines starts during maceration, thus including contact with the skin, whereas white wines are fermented after pressing. Many wines undergo a secondary fermentation to soften the flavor, called malolactic fermentation, in which bacteria convert lactic acid to malic acid.

In wine, contrary to beer, fermentation is often done with wild yeasts found naturally on the grape skins and in wine cellars. Whether to use wild yeasts or isolated

[1] This property of ale yeasts is known in the sense that authors proclaim to know it (Mortimer 2000; Pérez-Ortín *et al.* 2002), but Johnston 1990 appears to be the only reference.

cultures is decided by the winemaker; wild yeasts may give more complexity but have an increased danger of spoilage when compared to the more reliable purified strains. Natural fermentation of wine is predominantly carried out by *S. cerevisiae*. This species does not naturally dominate the ecology of grapes (Mortimer & Polsinelli 1999), but often takes over during fermentation due to the Crabtree effect (Goddard 2008). Most commercial wine yeasts are descendants of isolates from natural wine fermentation (Johnston *et al.* 2000). Wine yeasts are generally diploid, but sometimes polyploid and show a considerable degree of aneuploidy (Bakalinsky & Snow 1990). Other species, including *Saccharomyces* hybrids, have also been found in the winemaking environment (Masneuf *et al.* 1998).

Other

Sake is produced from rice in a brewing process that includes parallel fermentation with *S. cerevisiae* and *Aspergillus oryzae* Contrary to malt, the rice used in sake production does not contain amylase. Instead, a non-yeast fungus, *A. oryzae*, is added to the rice at the beginning of the process to break down complex carbohydrates. *S. cerevisiae*, which cannot metabolize starch, is largely responsible for the rest of the biotransformation. The combined work of these two yeasts is called parallel fermentation. The different steps of sake fermentation go through temperatures between 4 and 20°C.

Yeast is used in bakery to produce carbon dioxide that leavens the bread, while alcohol evaporates Bread is made from flour (ground wheat, or sometimes rye, barley or corn) and water, which are knead and baked. Yeast (*S. cerevisiae*) is often added to the dough, because it will ferment some of the flour, producing alcohol and carbon dioxide. The production of gas increases the size of the bread, making it lighter and easier to eat (leavening). The alcohol evaporates during baking. Alternatively, some breads are unleavened, or leavened with baking powder (which produces carbon dioxide through an acid-base reaction during baking), steam or fermentation products from naturally occurring microorganisms. Bread fermentation with yeast is characterized by low alcohol concentration, few fermentable sugars (most carbohydrates in the flour are complex sugars), and high temperature (180-220°C).

Cocoa is fermented by wild yeasts and acetic acid bacteria Cocoa, used for the production of chocolate, is left to ferment without adding an inoculum of pure culture simply by putting the beans in a hole in the ground or a wooden structure. Fermentation is done simultaneously by various strains of wild yeasts, including but not limited to *S. cerevisiae*, and also by acetic acid bacteria which convert newly produced alcohol to acetic acid. Alcohol levels do not go higher than 2.5%, while the maximum temperature can be higher than 45°C (Papalexandratou & De Vuyst 2011).

Cocoa beans are not the only case of extensive wild alcoholic fermentation. Some plants are associated with fermenting yeasts and bacteria, resulting in alcohol concentrations that can reach more than 4% (Dudley 2004). In some cases, alcoholic fruits are then consumed by animals (Dudley 2004; Wiens *et al.* 2008).

Bioethanol fermentation is a highly stressful and competitive environment Biofuel is produced by *S. cerevisiae*, primarily in Brazil and the USA. The process uses a fed-batch mode with cane juice or molasses of $150\text{-}200g/L$ of sugar, and is fermented up to ethanol concentrations of $9\text{-}12\%$ in as little as $6\text{-}10h$ with high efficiency (Stambuk *et al.* 2009). Thus, bioethanol yeasts are highly specialized sugar fermenters able to withstand high osmotic pressures and temperatures. More than half of the Brazilian bioethanol production facilities use optimized yeasts isolated from previous fermentation processes. Others use other commercial yeasts, such as those sold for bread baking. These strains then quickly adapt to the highly stressful and competitive biofuel reactors.

1.3 Copy number variation

Copy number variation (CNV) is a type of polymorphism that results from gains and losses of genetic material. The affected region may be of any size, from smaller than a single gene to whole chromosomes or entire genomes.

Genomic loss and duplication has long been recognized as an important source of genomic variation, and a key source of evolutionary innovation (Ohno 1970). Following duplication, redundancy of a gene is hypothesized to reduce evolutionary pressure on one copy, resulting in differential evolution and eventually i) loss of function (nonfunctionalization), ii) specialization into one of its originally multiple functions or situations of expression (subfunctionalization) or iii) mutation to acquire a new function (neofunctionalization). This process has been described on a large scale in yeast (Selmecki *et al.* 2015), as a common ancestor of the *Saccharomyces* species has undergone a whole genome duplication followed by massive gene loss and differential evolution of the retained copies (Kellis *et al.* 2004). For example, the genes ADH1 and ADH2 which encode dual alcohol and acetaldehyde converting enzymes, form one of the pairs retained after this duplication and may be partially responsible for the emergence of the Crabtree effect (Piškur *et al.* 2006).

At the same time, while the long-term effects of CNV are well-known, the short-term consequences of duplication and deletion are not. In the above view, extra gene copies are abstracted as redundant and thus evolutionarily flexible. This raises the question of how redundant genes are retained long enough for evolution to take effect. In fact, the copy number of a gene, genomic segment or whole genome can affect the phenotype in the short term (Pavelka *et al.* 2010). Mostly this effect of CNV on fitness is mediated by increased gene dosage causing higher expression levels. These two seemingly conflicting views have historically been separated and are not adequately unified in literature (Kondrashov 2012).

In yeast

With the increasing availability of non-S288C yeast studies, the importance of CNV to *S. cerevisiae* is becoming increasingly clear. Particularly for *S. cerevisiae*, CNV appears to be an especially important type of polymorphism. While genotypic variation in terms of SNP is lower in *S. cerevisiae* than in its close relative *S. paradoxus* (Liti, Carter, *et al.* 2009; Wang *et al.* 2012), CNV has been found to be three times as extensive in *S. cerevisiae* (Bergström *et al.* 2014). This suggests that evolution in *S. cerevisiae* has operated under a different mode of evolution than regular accumulation of SNP (Bergström *et al.* 2014). It should be noted that *S. cerevisiae* also has wider phenotypic variety, which may be in part due to its extensive CNV.

CNV is more frequent in subtelomeres and near Ty elements Like other types of polymorphism, CNVs are not uniformly distributed across the genome. This may occur because of differences in evolutionary pressure (*e.g.* deletion of an essential gene), or differences in how well different genomic regions are prone to, or biochemically protected against mutations. Subtelomeres are known as regions of extensive polymorphism both in sequence and structure (Horowitz *et al.* 1984; Schacherer *et al.* 2009; Winzeler *et al.* 2003). Subtelomeres are regions near the telomeres that harbor less genes and are more repetitive in sequence than other parts of the genome. Genes with loci close to the telomeres are also known to be less expressed, a phenomenon called the telomeric position effect (Gottschling *et al.* 1990). The genes that do lie within subtelomeres are often concentrated in families and are enriched for gene ontology terms related to metabolism and transport of metal, amino acids and carbohydrates, and also to stress and toxin response (Brown, Murray, *et al.* 2010). Because many of these categories are associated with different environmental niches, the location of subtelomeric genes is thought to be an evolutionary adaptation that increases adaptability.

In addition to the subtelomeres, there are other hotspots of CNV, which are typically located in regions near Ty elements (Dunn, Richter, *et al.* 2012). These are five families (Ty1-5) of approximately 6kb, flanked by long terminal repeat (LTR) sequences of 335bp (Xu & Boeke 1987). Similar to retroviral proviruses, these retrotransposons are part of the genome but can occasionally be transcribed and re-inserted in a different location. Many such elements have been found in yeast (Kim, Vanguri, *et al.* 1998).

Many industry-specific CNVs have been described While we are not close to understanding the full extent of CNV diversity and its effect on the *S. cerevisiae* phenotype, several attempts have been made at uncovering the most important CNVs that allow some strains of *S. cerevisiae* to be so well adapted to their respective industry. With some exceptions, most authors have focused on wine strains. In general, GO enrichment analyses have shown that the most copy number variant genes are related to flocculation, drug response, metabolism and transport of sugar, metal and alcohol (Carreto *et al.* 2008; Dunn, Lavine, *et al.* 2005). Some genes will be discussed in more detail in Results & Discussion.

Detection

Before whole genome sequencing (WGS), information on CNV was generally obtained using cytogenetic methods. In microarray-based comparative genome hybridization (aCGH), cDNA probes from selected regions of a reference genome are attached to a microchip and brought into contact with equal amounts of denatured DNA from the reference and sample of interest. The DNA from both sources must be labelled with different fluorophores. The ratio of fluorescence of both colors to any segment on the chip then serves as a proxy for the ratio of complementary DNA present from both samples *i.e.* the copy number. Microarray-based methods are still used because they are less expensive and can target small regions with more precision than sequencing methods.

There are five classes of NGS-based CNV detection algorithms As whole genome sequencing becomes increasingly popular, more and more studies are using NGS data for CNV detection. The advantage is that NGS naturally has whole-genome coverage rather than a analysing only a few cherrypicked regions (although modern microchips also contain cDNA from many loci across the genome), and that NGS data can be reused for other purposes. Several algorithms have been developed for NGS-based CNV detection. They can be divided into five classes: paired-end mapping, split read, *de novo* assembly, read depth and combinatory (Zhao *et al.* 2013). The first three approaches are briefly discussed below. The read depth approach, which is by far the most popular, will be explained in more depth. In all approaches except assembly, sequenced reads are first mapped to a reference sequence. All detected CNVs are in relation to that reference. Consequently, only assembly-based CNV detection algorithms are capable of discovering variation in regions that are not present in the reference.

Paired-end mapping relies on the assumption that pairs of reads sequenced from the same insert should always be mapped close together, with a gap in between that reflects the insert size. A breakpoint is detected whenever this assumption does not hold. Pairs that are mapped further away than expected indicate that the region in between is deleted in the sample, while pairs that are mapped too close together indicate an insertion. A major disadvantage of the paired-end mapping method is that it cannot detect insertions larger than the insert size, or discriminate between different copy numbers. It is also sensitive to noise coming from multiple mapping in low-complexity and other repeated regions.

The split read method uses reads that are not mapped completely to the reference genome, in the idea that reads may be unmapped, chimerically or partially aligned because they harbor a CNV breakpoint. These reads are split in two, and both ends are aligned separately to the reference. Two ends of a read mapping far away may indicate either a deletion or an insertion. This method is highly accurate in finding exact base pair locations of breakpoints, but does not perform well on non-unique genomic regions, and it gives no indication of the copy number of genomic regions. The split read method is mostly used in conjunction with other strategies.

Finally, the assembly-based approach is the only approach that does not require prior mapping to a reference genome, making it uniquely qualified to detect CNVs and other polymorphisms in locations that do not occur in the reference. Overlapping reads are assembled into large contigs, *i.e.* contiguous genomic subsequences, as in *de novo* assembly. These contigs are then compared to the reference to see which regions are amplified or deleted. The downside to assembly-based CNV detection is that assembly is a difficult process that requires lots of computational resources. *De novo* assemblies from short read data are also known to be of low quality in repeated regions.

The traditional method for CNV detection assumes that copy number is proportional to read depth By far the most popular method for NGS-based CNV detection is the read depth approach. The read depth of a particular nucleotide on a reference genome is defined as the number of reads from a sample that map to a region containing that base. In the read depth approach, this quantity is used as a proxy for copy number, based on the simple idea that regions with higher copy number have a higher probability of being more abundantly sequenced.

Explicitly, copy number inference with this method starts from the assumption of uniform read depth: that reads are uniformly sampled subsequences from the sample genome. If this is true, then the expected value of the read depth in a genomic region is directly proportional to the copy number of the sample genome in that region, compared to the reference. More specifically, the read depth approximately follows a Poisson distribution with parameter λ equal to the copy number times the coverage divided by the ploidy. For example, a diploid genome sequenced at $50X$ has an expected read depth of $25 \times copynumber$ in each genomic position. In practice, the read depth is rarely measured for each base individually because of high variance, and because measurements in neighboring bases are not independent (most reads that map to one base also map to its neighbors). Instead, the genome is partitioned into a set of equally sized regions called frames, generally around 1000bp. The read depth of such a frame is the average of the read depths of the bases that it contains. For all intents and purposes, these frames are the atomic units of the genome, and fulfill the same role in the analysis as individual base pairs except the read depth measurement is more robust, and less precise.

Read depth-based algorithms are useful because they can directly estimate the copy number of a genomic sequence. However, read depth methods are blind to structural variations that do not alter copy number, such as translocations and inversions.

Read depth is not really proportional to copy number While the assumption of uniform read depth is often reasonable in the sense that it leads to acceptable qualitative results, in reality, the assumption is incorrect due to biases introduced by both technical and biological sources. Read depth bias can affect small regions, which increases the variance of the read depth distribution, or larger regions such as chromosomes, which increases or decreases the read depth in a way that may be difficult or impossible to distinguish from the effects of a CNV. Some examples include (•) difficulty of sequencing heterochromatin regions and the general influence of 3D DNA structure on sequencing, (•) the influence of GC-content on primer hybridization, which in turn biases the abundance of GC-rich inserts after PCR amplification and thus the probability of ending up in the sequencing data and (•) parts of the genome where the reference and sample are more dissimilar, or low-complexity regions are more difficult to successfully align so that they will be under-represented in the read depth count. Note that these local biases are especially applicable to subtelomeric regions, as they generally have low complexity, are highly polymorphic and are difficult to amplify and sequence. Subtelomeres are therefore difficult to analyze. For the same reasons, analysis of mitochondrial DNA is even more difficult.

Many modern CNV detection algorithms attempt to estimate and correct read depth bias. The most straightforward method is to use a control sample to normalize read depth. If multiple samples are available, an alternative is to estimate the bias by aggregating read depth measurements from observations, *e.g.* cn.MOPS (Klambauer *et al.* 2012). Other methods use input from only one sample, and estimate bias using a model-based approach. For example, Control-FREEC (Boeva *et al.* 2012) estimates the bias due to GC-content with a LOESS regression.

Inference of copy number breakpoints is done using a variety of methods The main workhorse of any CNV detection algorithm is its statistical method for segmenting the genome into regions of constant copy number, and estimating the copy number in each region. Common approaches include circular binary segmentation (CBS), a method which iteratively breaks up the genome into smaller segments by finding the most likely breakpoint, or hidden Markov models (HMM), which can let the Viterbi algorithm segment the genome (Klambauer *et al.* 2012; Zhao *et al.* 2013). Many of these methods were originally developed for aCGH, and adapted for NGS data. Other algorithms use custom statistical models or heuristics to break up and/or join genomic segments. These methods are sometimes specific to NGS. For example, Control-FREEC uses combined information from read depth and SNP distributions to obtain an absolute copy number estimate. GROM-RD (Smith *et al.* 2015) uses a combination of multiple frame sizes to increase the resolution of breakpoint calling.

1.4 Smoothing splines

Smoothing is a type of curve fitting where a noisy dataset is approximated by a function in an attempt to capture the essence of the underlying pattern. It is related to regression analysis, in which a statistical model is compared to the data, with the goal of obtaining an estimation of the model parameters and/or doing statistical inference. The main difference is that smoothing typically does not set an explicit model for the resulting curve, and that it is designed to ignore both noise and small-scale bias in the data, whereas the statistical model is an integral part of a regression analysis which attempts to reconstruct the generating function of a process. Thus, the typical assumption in regression analysis is that residues are identically and independently distributed (i.i.d.) according to a zero-centered normal distribution. The assumptions of smoothing are less stringent and the goal is less ambitious.

Smoothing

The goal in smoothing is to find a curve that simultaneously captures trends in the data while ignoring noise. In other words, the curve should minimize the distance between the curve and the data (the error), but should also avoid erratic behavior (unsmoothness). These two goals are conflicting: a curve that fits the data completely also matches all the noise, whereas a curve that is maximally smooth – in the extreme, a constant line, but the definition of smoothness varies per smoothing strategy – often does not have the flexibility to capture trends in the data. To balance this trade-off, most smoothing procedures can be tuned with a smoothing parameter.

The balance between error and unsmoothness can be seen as the smoothing equivalent to over- and underfitting in regression. Both trade-offs are determined by the flexibility of the fitting procedure. However, smoothness relates to robustness against local deviations. These 'deviations' might well be present in the true generating function, such as with small CNVs. Overfitting only refers to the decline in performance on new data when a model is allowed to fit to particularities of the training data set.

Moving average, kernel smoothers and local regression are examples of smoothing procedures As an example, probably the simplest smoothing procedure is that of the moving average of a dataset y of equally spaced points. The moving average associates with a set of n data points y_i a 'smoothed' version s, where the ith point s_i equals the average of the $2m + 1$ surrounding points $s_i = \text{mean}\{y_{i-m}, \ldots, y_{i+m}\}$ (Figure 1.2). The smoothness of the result is controlled by the smoothing parameter m. This procedure is well-known because of its simplicity, but the result is rarely very smooth, there is no inter- or extrapolation and it only works for equally spaced one-dimensional data.

These shortcomings can be solved by using kernel smoothing. For non-equally spaced data, associate with each point y_i a pre-image x_i. A kernel smoother transforms this data into a smooth function defined by $s : x \mapsto \sum_{0<i\leq n} K(x - x_i)y_i$, for some function $K : \mathbb{R} \to \mathbb{R}$ called the kernel. Kernel smoothing can be interpreted as a weighted moving average, where the weights are determined by the kernel. For example, the box kernel $K_{box,m}(x)$ equals $1/(2m + 1)$ in the interval $x \in [-m, m]$ and 0 everywhere else. For each datapoint x_i the kernel smoother $s_{box,m}(x_i)$ using the box kernel $K_{box,m}$ equals the moving average s_i with smoothing parameter m for an equally spaced dataset x_i (Figure 1.2). However, the kernel smoother s is also naturally defined for values of x not in the set $\{x_i\}_{0<i\leq n}$, and also deals with data that is not equally spaced. More importantly, the kernel K can be freely chosen. A popular kernel is the Gaussian function, which leads to smoother results than the moving average (Figure 1.2). By choosing a kernel $K : \mathbb{R}^m \to \mathbb{R}^m$, for example a multivariate Gaussian kernel, we can also smooth multidimensional data.

To further generalize the kernel smoother, the mean can be replaced by a linear or polynomial regression. Compare this in interpretation to a Taylor polynomial approximation of a function, which gives the optimal m-degree polynomial, locally near a point x_i. The quality of the approximation, and the interval in which it gives an approximation with an acceptable error increases for each m. The implicit assumption when taking the local (weighted) average is that the function that the smoother approximates is locally constant. By instead using a linear regression, using the kernel values as weights, the function becomes assumed locally linear, which is often acceptable in a much larger region. This idea can be easily generalized towards polynomial regression. The downside to a more flexible model is the possibility of overfitting, and higher degree polynomial regression may become unstable unless the data has a very high density. Most functions – all 'smooth' functions – are well approximated by a linear function provided that the interval of application is small enough (*i.e.* that K goes to 0 fast enough). For this reason, most local regression methods use linear or sometimes quadratic models. These smoothers are called locally weighted scatterplot smoothing or local regression (LOESS or LOWESS) (Figure 1.2).

The smoothing methods described above are all local in nature: the value of the smoothed curve is dependent only on neighboring points. This poses problems near the edges of the domain, where fewer neighboring points are available, so that the curve is either low in quality or undefined. They offer no interpretation of the resulting curve, qualitatively or quantitatively in the form of fitted parameters. A more complex smoothing method is that of the smoothing spline. It has a global approach and a more direct mathematical interpretation, with a clearer set of assumptions and goals in the form of explicit formulas. Smoothing splines also make use of a kernel, *i.e.* a function that helps transform information in or locally around a point to a different space, albeit more rigidly.

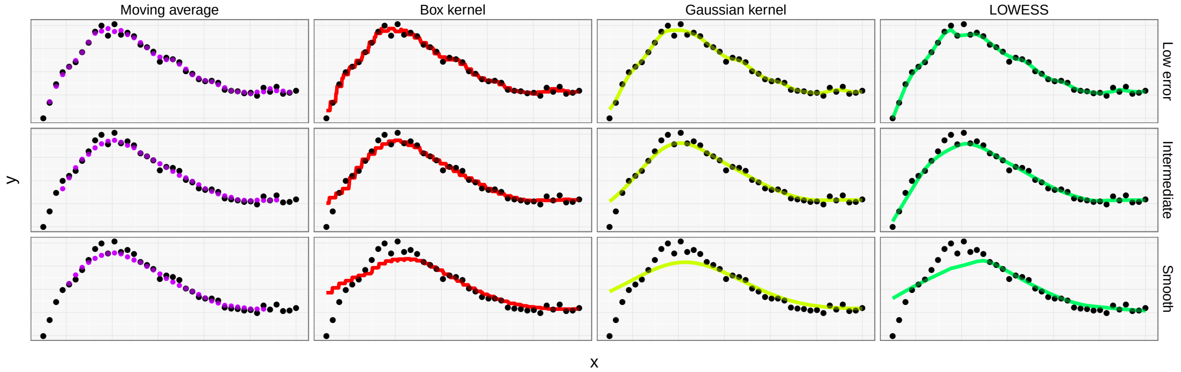


Figure 1.2: Different smoothing methods. Smoothing using moving average, box kernel smoothing, Gaussian kernel smoothing and LOWESS with a local linear model and using the box kernel. Each method is shown with three values of the smoothing parameter, one low (minimizing the error but also capturing some noise), one intermediate and one high (keeping the curve smooth at the cost of fitting precision).

Splines

A spline is a smooth, piecewise polynomial function. The degree of the polynomial is called the degree m of the spline, and a smooth function is defined here as C^{m-1}.[2] The boundaries between the polynomial pieces are called the knots of the spline. More formally, an m-degree spline with domain $[a, b]$ is a function in $C^{m-1}[a, b]$ so that, for a finite set of k points κ_i (the knots) that partition the interval $[a, b]$ into $k + 1$ subintervals I_i, each restriction[3] $f|_{I_i}$ of f to one of these intervals I_i is a polynomial function of degree m.

Below, only splines on $[0, 1]$ will be considered. All results are also applicable to general intervals $[a, b]$ through straightforward linear corrections. While the spline is technically defined on a closed interval, it can be naturally extended to the real numbers.

The power of splines is in their combination of desirable global and local properties: locally, they have a simple and smooth polynomial form, while globally they are flexible.

Splines are used as regression models In this regard, splines are sometimes used as simple but flexible models for ordinary least squares regression. A spline can be expressed as a linear model:

$$f = \sum_{i=1}^{m} a_{i-1} x^{i-1} + \sum_{i=1}^{k} \zeta_i \chi_{[\kappa_i, +\infty)} x^m \tag{1.1}$$

[2] A C^i function is a function that is i times differentiable and has a continuous ith derivative. The set $C^i[a, b]$ is the set of functions $f : [a, b] \to \mathbb{R}$ that are C^i.

[3] A restriction $f|_A$ for $A \in \operatorname{Dom} f$ is the function $f|_A : A \mapsto \operatorname{Range} f : x \mapsto f(x)$.

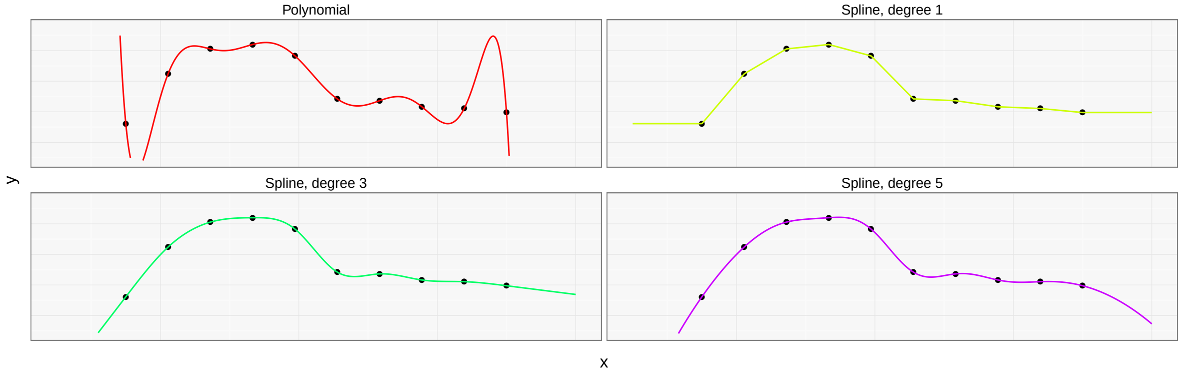


Figure 1.3: Comparison of polynomial and (natural) spline interpolation of different degrees. Data is obtained from the same generating function as in Figure 1.2. Runge's phenomenon is clearly visible for polynomial interpolation. Extrapolation is also shown on both sides of the data. It is constant for degree 1, linear for degree 3 and parabolic for degree 5 natural splines. Cubic splines (degree 3) are the most used, because higher degrees do not add much flexibility.

Where a_i and ζ_i represent the estimated coefficient parameters, and χ_i are indicator functions. The spline has more flexibility near knots, so when possible it is best to choose more knots in regions where the function is least predictable.

Interpolation is a typical spline application One of the main applications of splines is to produce stable interpolation. Polynomial interpolation appears theoretically very attractive, but in fact it often behaves erratically because, to fit n data points, a polynomial of degree $n-1$ is required. This erratic behavior is exemplified by Runge's phenomenon, which states that polynomial interpolations have a tendency to oscillate wildly near the edges of the interpolation interval, especially at higher orders. The principle of Runge's phenomenon extends as well to extrapolation. Splines offer a stable and smooth interpolation that corresponds well to intuition (Figure 1.3), while also remaining easy to describe in mathematical form and being theoretically attractive.

For an interpolating spline, all data points except the outer two are knots. The outer two points define the boundaries of the spline domain $[a, b]$; below we will only consider data in $[0, 1]$. An m-degree spline with $n-2$ knots has $m+n-1$ coefficients as per equation 1.1, leaving $m-1$ degrees of freedom after considering the n restrictions that the interpolating function must go through each data point. Usually, this issue is resolved by setting the last $(m-1)/2$ (for uneven m) derivatives in the boundaries to 0 as an extra condition. This is called a natural spline. Another method, is to fit exact m-degree polynomials for each boundary and reuse the last $(m-1)/2$ obtained conditions for the spline (Forsythe *et al.* 1977).

The smoothness of the interpolating spline is not only intuitive, it is also theoretically attractive, in the sense that the spline interpolator is the 'most smooth interpolator', for some smoothness measure and within some broad class of functions. Specifically, within the class of C^M functions whose Mth derivative is square integrable, and for which conditions 1.2a hold (*i.e.* smooth interpolators), there is a unique function that minimizes the non-smoothness given by 1.2b, and that function is the natural interpolating spline of degree $2M - 1$.

$$\forall 0 < i \leq n : f(x_i) = y_i \tag{1.2a}$$

$$\textit{non-smoothness}(f) = \int_{\mathrm{Dom}\, f} (f^{(m)}(x))^2 dx \tag{1.2b}$$

Measure 1.2b can be interpreted as a measure of deviation from M-degree polynomial behavior. Polynomials of degree $M - 1$ (or lower) are considered maximally smooth, and they are the only functions with a non-smoothness penalty of 0.

Function minimization problems are naturally described in Hilbert spaces

Mathematically, measure 1.2b is attractive because its minimization has a unique solution. This is described using reproducing kernel Hilbert space (RKHS) theory. Below is a short description of the key elements. For a slightly more complete but concise tutorial, see Nosedal-Sanchez *et al.*, 2012 (Nosedal-Sanchez *et al.* 2012).

The set described earlier ($C^M[0,1]$ functions with square integrable Mth derivative) can be interpreted as a vector space and equipped with an inner product (equation 1.3). This structure, H_M, is called a Hilbert space.

$$\langle f, g \rangle = \int f^{(M)}(x) g^{(M)}(x) dx \tag{1.3}$$

This particular inner product has the property that, for each element $x \in [0,1]$, there is a unique function $K_x \in H_M$ so that the product $\langle K_x, f \rangle$ equals $f(x)$, for any $f \in H_M$. In other words, function evaluation can be written in terms of the inner product of functions. The function $K : [0,1] \to H_M : x \mapsto K_x$, or equivalently $K : [0,1]^2 \to \mathbb{R} : (x,y) \mapsto K_x(y) (= K_y(x))$ is called the reproducing kernel (Box 1.1).

The merit of this description is that both 1.2b and 1.2a can be written in terms of the inner product, and the solution to the minimization problem can be written as a linear combination of the functions K_{x_i}(Box 1.2). The coefficients can be extracted by a simple linear equation system. Note that each function K_{x_i} is a spline of degree $2M - 1$ with a single knot at x_i (equations 1.4), so that the result will indeed be a spline with knots at the data points.

In practice, it is easier to find the interpolating spline by simply using the basis 1.1 instead, and solving for the coefficients using the interpolation conditions 1.2a and boundary conditions for a natural spline. However, in the above we have theoretically derived that the interpolation spline is the solution to the minimization problem, and describing the spline using the functions K_{x_i} as a basis will be essential for solving the more complicated smoothing spline problem.

Box 1.1: Reproducing kernels

The reproducing kernels K of the Hilbert spaces H_M are, for some low values of M:

$$M = 1: \quad K(s,t) = \min\{s,t\} \tag{1.4a}$$

$$M = 2: \quad K(s,t) = \frac{1}{2}\max(x,y)\min^2(x,y) - \frac{1}{6}\min^3(x,y) \tag{1.4b}$$

$$\begin{aligned} M = 3: \quad K(s,t) = {} & \frac{1}{30}\min^5(x,y) - \frac{1}{6}\max(x,y)\min^4(x,y) \\ & + \frac{1}{3}\max^2(x,y)\min^3(x,y) \end{aligned} \tag{1.4c}$$

For example, if $M = 1$, then $K(s,t) = \min\{s,t\}$ because for any $x \in [0,1]$:

$$\begin{aligned} \langle K_x, f \rangle &= \int_{\mathbb{R}} K'(x,s) f'(s) ds \\ &= \int_0^x f'(s) 1 ds + \int_s^1 f'(s) 0 ds \\ &= f(x) \end{aligned}$$

Smoothing splines

Smoothing splines are similar to interpolation splines but are bound by less stringent conditions. For smoothing, it is not required that the function passes exactly through all the data points, but the distance between the data and the smoothing function should be as small as possible. Thus, the hard conditions set in 1.2a for interpolation are replaced in smoothing by a simultaneous minimization of the error and non-smoothness. This is a direct formalization of the goals of smoothing described at the beginning of this section. Formally, using the least squares error as the error term, a smoothing spline is the unique solution to the minimization problem:

$$f = \operatorname*{argmin}_{f \in C^{M-1}[0,1]} \sum_i (f(x_i) - y_i)^2 + \lambda \int (f^{(M)}(x))^2 dx \tag{1.5}$$

The first term formalizes the requirement for the resulting function to be close to the data (the error term; the same as in ordinary least squares regression), while the second term enforces smoothing (smoothing term; the same as in spline interpolation). Their relative importance is dictated by the smoothing parameter $\lambda > 0$.

This minimization is indeed a continuation of the above: if $\lambda \to 0$, the error term is infinitely more important than the smoothing term, essentially enforcing the hard conditions of 1.2a. Thus, smoothing is then equivalent to spline interpolation. On the other extreme, if $\lambda \to +\infty$, error is only of infinitesimal importance relative to smoothing. As a result, the smoothing term will be close to zero, making the problem equivalent to ordinary least squares regression of an $M-1$-degree polynomial. For any value λ in between, the solution varies continuously from smooth to fitting.

The framework described above in the context of the interpolation spline can also be used for solving the smoothing spline problem by writing the problem 1.5 as a system of equations (Box 1.3).

Box 1.2: The most smooth interpolator is a linear combination of the functions K_{x_i}

Note first that the conditions 1.2a and minimization 1.2b can be rewritten as:

$$\forall 0 < i \leq n : \langle K(x_i, \cdot), f \rangle = y_i \tag{1.6a}$$

$$\textit{non-smoothness}(f) = \langle f, f \rangle \tag{1.6b}$$

Now assume that there is a function $f^* = \sum_i c_i K_{x_i}$ that is a linear combination of functions K_{x_i}, fulfilling conditions 1.2a (or equivalently 1.6a). Then we have that, for each $0 < k \leq n$:

$$\begin{aligned} y_i &= \langle K_{x_k}, f^* \rangle \\ &= \langle K_{x_k}, \textstyle\sum_i c_i K_{x_i} \rangle \\ &= \textstyle\sum_i c_i \langle K_{x_k}, K_{x_i} \rangle \end{aligned}$$

This is a system of n linear equations in n unknowns and thus has a unique solution (provided that the equations are linearly independent), proving that there is exactly one unique function f^*. Rewriting this in matrix form, if we define a matrix Σ so that $\Sigma_{ij} = \langle K_{x_i}, K_{x_j} \rangle$, then the unique vector of coefficients c is

$$c = \Sigma^{-1} y \tag{1.7}$$

This notation also allows us to describe the smoothing penalty 1.2b (1.6b):

$$\begin{aligned} \textit{penalty}(f^*) &= \langle \textstyle\sum_i c_i K_{x_i}, \sum_j c_j K_{x_j} \rangle \\ &= \textstyle\sum_i \sum_j c_i c_j \langle K_{x_i}, K_{x_j} \rangle \\ &= c^T \Sigma c \end{aligned} \tag{1.8}$$

It remains to be shown that f^* is also the most smooth function f that follows 1.6a. This proof is beyond the scope of this text, as it requires the notion of orthogonality and more background on Hilbert spaces.

Box 1.3: Solution of the spline smoothing problem

Both terms of the smoothing spline minimization 1.5 can be written in a single equation. Without proof, assume that the solution is indeed a natural spline with one knot in each observation, and define $n+M$ functions ϕ_i, so that $\phi_0 = 1, \ldots, \phi_{M-1}$ are polynomials of degree 0 to $m-1$ and $\phi_M, \ldots, \phi_{M+n-1}$ are the functions K_{x_i}. Thus, the solution is $f = \sum_i \beta_i \phi_i$ for some coefficient vector β. In addition, define the data matrix X so that $X_{ij} = \phi_j(x_i)$, and the penalty matrix Σ so that $\Sigma_{ij} = \langle \phi_i, \phi_j \rangle$. Then, equation 1.5 can be reformulated as

$$c = \underset{\beta}{\operatorname{argmin}} \frac{1}{n} \|y - X\beta\|^2 + \lambda \beta^T \Sigma \beta \tag{1.9}$$

For the minimum penalty solution, the derivatives of this equation to all of the variables must be zero. Solving that system gives

$$\beta = (X^T X + \lambda \Sigma)^{-1} X^T y \tag{1.10}$$

This solution bears resemblance to both spline interpolation (*cfr.* equation 1.7) and to the general solution of linear regression (*cfr.* $\beta = (X^T X)^{-1} X^T y$).

Part 2

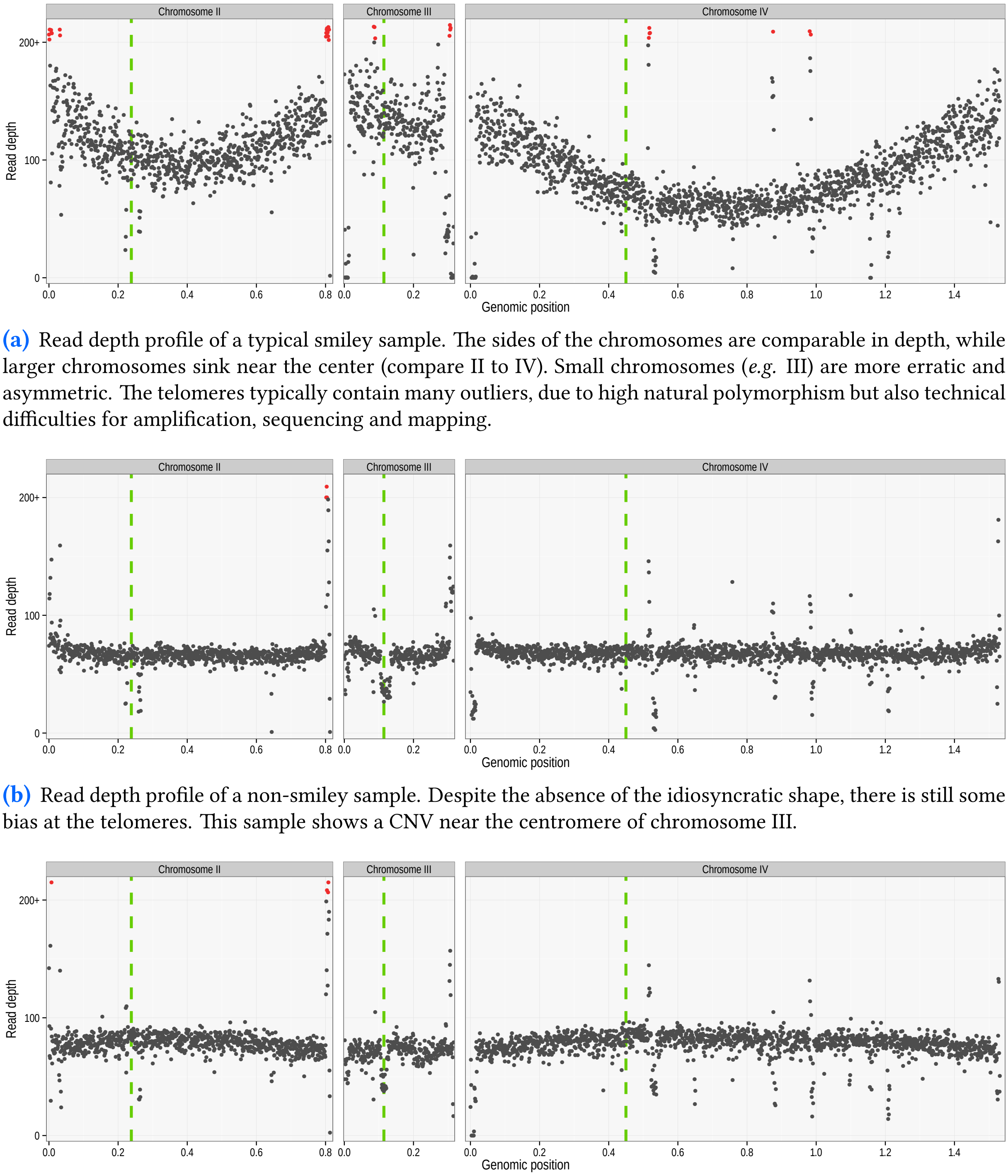
Introduction

This thesis is part of a whole-genome sequencing project, covering a broad range of industrial yeast strains, with a focus on brewing isolates. In order to increase our understanding of the genetic and phenotypic diversity of beer and other industrial yeast, and to trace their evolutionray history and population dynamics, we selected 200 yeast strains for Illumina high coverage sequencing (mostly $100X$-$300X$). Unlike many other population studies (Liti 2015), these strains were sequenced at natural ploidy. Nothing was known about these yeasts beforehand, except their home industry. More than half of the samples were collected from ale fermentation, almost a fifth from lager and the rest from sake, wine, wild, spirits, bakery, bioethanol and cocoa. The reference strain, S288C, was also included.

3 out of 200 samples were unsound and had to be discarded. After preliminary analysis, of the 197 remaining samples, 157 were identified as *Saccharomyces cerevisiae* strains. The other 40 strains consisted of 35 *Saccharomyces sensu stricto* hybrids, and 5 were non-*Saccharomyces* species.

A peculiar bias, called the smiley pattern, prevents CNV analysis in a portion of the samples by conventional means Previous studies have reported that copy number variation (CNV) is surprisingly extensive in *S. cerevisiae* compared to other yeast species (Bergström *et al.* 2014), and is an important factor contributing to phenotypic diversity (Warringer *et al.* 2011). As such, CNV is an essential part of the population analysis. However, when attempting to do a read depth-based CNV analysis, we noted a massive deviation from the uniform read depth assumption in approximately half of the collected *S. cerevisiae* samples.

Instead of fluctuating around a constant line, the read depth profile of these samples has the shape of a convex curve (Figures 2.1, 2.2). The depth of coverage at the terminal regions of the chromosomes is generally roughly comparable within the same sample – without considering the erratic read depth that is found in telomeres. The coverage depth near the center of the chromosomes is much lower than near the terminal regions. The difference varies greatly between different samples (Figure 2.3). The read depth difference is generally more pronounced in larger chromosomes. However, small chromosomes are no less affected: the smallest (I

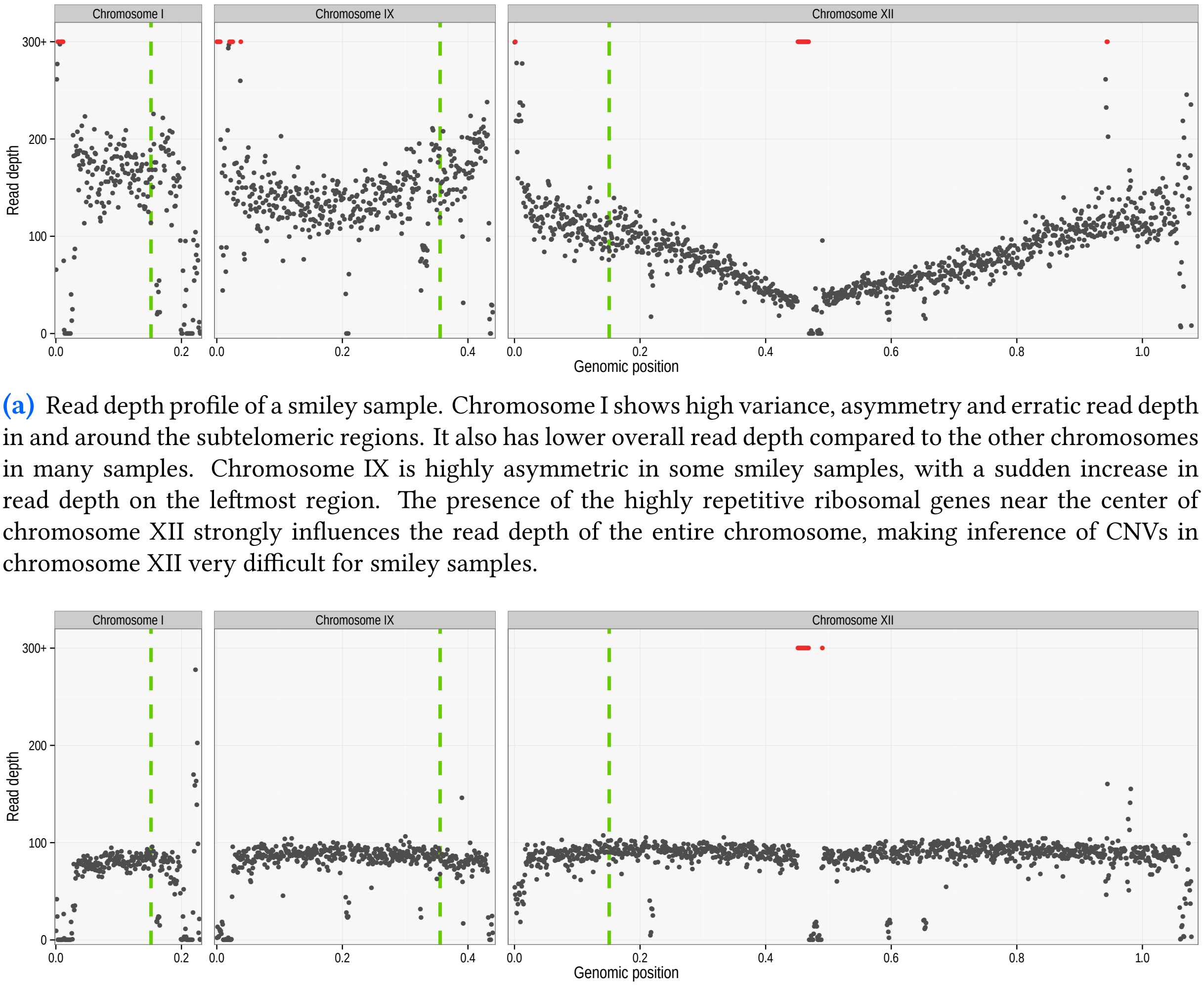


(a) Read depth profile of a typical smiley sample. The sides of the chromosomes are comparable in depth, while larger chromosomes sink near the center (compare II to IV). Small chromosomes (*e.g.* III) are more erratic and asymmetric. The telomeres typically contain many outliers, due to high natural polymorphism but also technical difficulties for amplification, sequencing and mapping.

(b) Read depth profile of a non-smiley sample. Despite the absence of the idiosyncratic shape, there is still some bias at the telomeres. This sample shows a CNV near the centromere of chromosome III.

(c) Read depth profile of a non-smiley sample. While not as distinct as the smiley, the read depth baseline of these samples is not straight but has a slight concave curve.

Figure 2.1: Read depth profiles showing different forms of bias. Chromosomes II, III and IV are shown for three samples of similar coverage showing the different forms of bias. These chromosomes are representative of the general pattern. Frames outside the scale of the vertical axis are colored red. The location of the centromeres is displayed as a green dashed line.



(a) Read depth profile of a smiley sample. Chromosome I shows high variance, asymmetry and erratic read depth in and around the subtelomeric regions. It also has lower overall read depth compared to the other chromosomes in many samples. Chromosome IX is highly asymmetric in some smiley samples, with a sudden increase in read depth on the leftmost region. The presence of the highly repetitive ribosomal genes near the center of chromosome XII strongly influences the read depth of the entire chromosome, making inference of CNVs in chromosome XII very difficult for smiley samples.

(b) Read depth profile of a non-smiley sample. The ribosomal region in chromosome XII has very high read depth, often around 30 times the baseline value. The region to the right of these high regions appears deleted. This is probably an artefact of the mapping procedure, with the true read depth being spread across both the apparent amplification and deletion parts.

Figure 2.2: Read depth profiles of chromosomes I, IX and XII. These chromosomes have region-specific baseline shapes that occur across samples. Plot is in the same style as Figure 2.1.

(230218bp), III (316620bp) and VI (270161bp)) have highly erratic read depths and asymmetric bias. Aside from the global shape, these samples also have high local variance in read depth. Note that the location of the baseline's minimum is, in most cases, located near the center of the chromosome and appears to be unrelated to the location of the centromere.

We have termed this phenomenon 'smiley'. Out of the 157 *S. cerevisiae* genomes analyzed, 58 have a clear smiley pattern. 10 of the other samples have a profile that appears 'jagged' (Figure 2.4). While the read depth of the jagged samples is highly unpredictable and cannot be used for analysis, the smiley pattern can be characterized and the signal recovered.

The cause of the smiley bias is unknown Different groups of samples were processed by different partner labs using different procedures. Only one set of samples displayed the smiley pattern. By repeating parts of the procedure, we were able to ascertain that the smiley only occurs when using the DNeasy® Blood & Tissue Kit from Qiagen (*DNeasy Blood & Tissue Kit* n.d.). However, the biological, chemical or technical reasons for this peculiar pattern could not be determined.

This thesis contains a description of a new CNV detection algorithm, as well as a critical analysis and some preliminary results In order to detect CNVs in these samples, I developed a novel, fully non-parametric CNV detection algorithm, SPLINT, that is capable of measuring and circumventing certain types of read depth bias. SPLINT is written in R (v. 3.2.0) (R Development Core Team n.d.). The main principles used in this algorithm are described below.

Furthermore, I have critically evaluated the strong and weak points of SPLINT and compared its results to those obtained from NEXUS™ (version 7.5) (*Nexus Copy Number Discovery Edition* n.d.), an established commercial software platform for yeast genomics analysis. I also give a high-level overview of the observed CNV profiles, followed by a comparison of my results to CNVs described in literature.

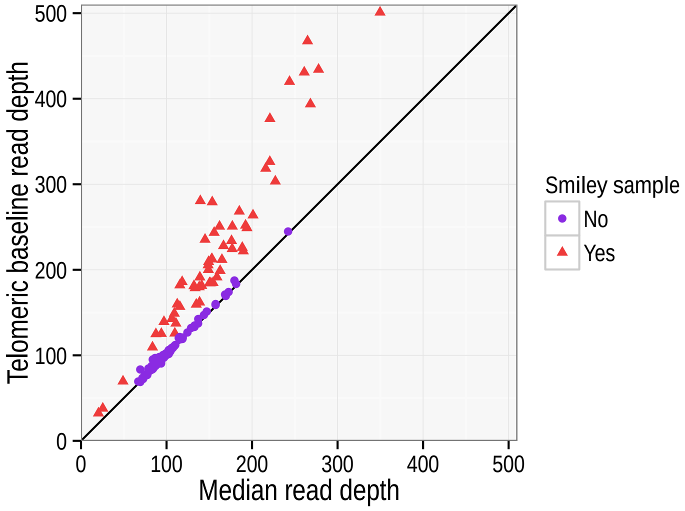


Figure 2.3: Diversity of the smiley pattern. Each dot represents a sample, plotted with its median and terminal read depth on the horizontal and vertical axes. Calculation of the read depth in the terminal regions is done using Splint (see section 3.3). Smiley samples have higher read depth near the telomeres than elsewhere on the chromosome, but the strength of the smiley pattern is variable between samples.

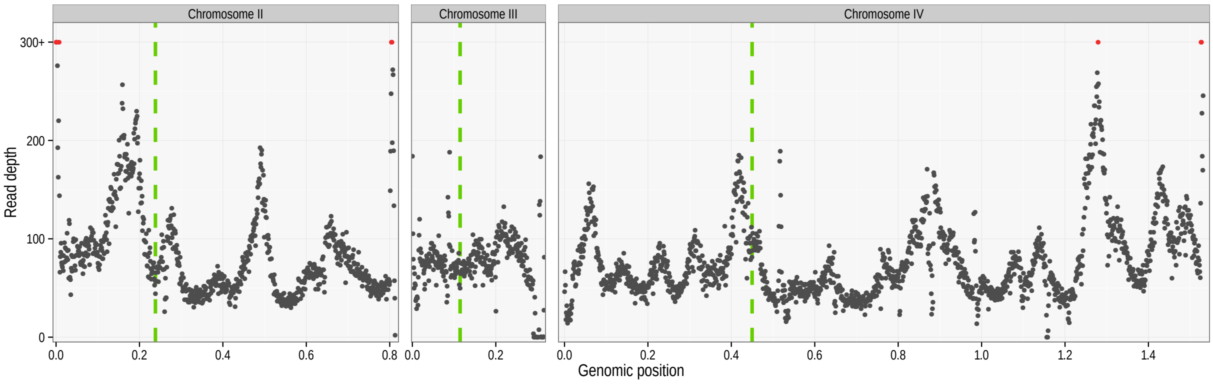


Figure 2.4: Read depth profile of a jagged sample. Figure uses the same style as Figure 2.1. The profile of these samples is too unpredictable to be used for CNV detection based on read depth.

Part 3

Splint

Under the uniform read depth assumption, the expected value of the read depth in each frame equals the coverage times the copy number divided by the sample ploidy. This assumption is broken by various large- and small-scale biases, which are abstracted as part of the error term.

$$\textsf{\textit{read depth}} = \frac{\textsf{\textit{coverage}} \times \textsf{\textit{copy number}}}{\textsf{\textit{ploidy}}} + \textsf{\textit{error}} \tag{3.1}$$

The goal of Splint is to make the same inferences based on less stringent assumptions, enabling it to take into account large-scale bias. To this end, the large-scale bias is explicitly modeled in a new factor called the baseline that replaces the constant coverage. The baseline is assumed to be smooth although the exact shape is not known. This is called the baseline read depth assumption.

$$\textsf{\textit{read depth}} = \frac{\textsf{\textit{baseline}} \times \textsf{\textit{copy number}}}{\textsf{\textit{ploidy}}} + \textsf{\textit{error}} \tag{3.2}$$

Both the baseline and copy number are estimated using the same regression The essence of Splint is a regression method based on the baseline read depth assumption. Both the baseline and copy number are simultaneously estimated using a single regression procedure f. The estimations are written $baseline$ and $copynumber$, respectively. During the fitting, $baseline$ will become a continuous curve, whereas $copynumber$ will become a discontinuous step function that changes value at each copy number breakpoint.

$$f \sim baseline \times copynumber \approx \textsf{\textit{read depth}} \tag{3.3}$$

The use of this regression model has several important consequences. Firstly, it is important to keep in mind that the error term in equation 3.2 contains local bias as well as normal sampling variance. Thus, the assumptions usually made in a regression – i.i.d., zero-centered normal errors – are untrue. In that respect, the procedure is more similar to smoothing than regression.

Secondly, the factor $copynumber$ conflates the factors *ploidy* and *copy number* from 3.2, and also *coverage*. $copynumber$ has real values, as opposed to integer values, in theory equal to $\mathit{copy\ number}/\mathit{ploidy}$. This allows SPLINT to estimate copy numbers without making any assumptions about the sample ploidy. Ideally, the ploidy can be estimated from the occurring values of $copynumber$, because they should be multiples of $1/\mathit{ploidy}$. However, in practice, there may be high variance on the actual values of $copynumber$.

Thirdly, the single regression curve of 3.3 models two quantities of interest at the same time: $baseline$ and $copynumber$. These two factors need to be modeled in such a way that they can be estimated simultaneously but interpreted separately. It is a delicate balance within the regression procedure that decides which of the two factors should adapt to a change occurring in the data. Ideally, the $copynumber$ factor fits only to sudden jumps (copy number breakpoints), and $baseline$ fits to smooth changes (large-scale bias, such as a smiley). Fitting of the $copynumber$ factor to a smooth change introduces a false positive CNV call. Fitting the $baseline$ factor to a sudden change introduces a false negative CNV call. Both cases result in an incorrect estimation of $baseline$ and produce a bias in copy number estimation that accumulates along the rest of the chromosome. In order to avoid fitting the wrong factor of the regression curve to a particular change in read depth, it is crucial to choose the correct regression model, to tell the regression procedure how fast or how sudden $baseline$ is allowed to adapt, and at which locations discontinuity is accepted in the $copynumber$ term. The locations of breakpoints in the $copynumber$ term need to be estimated for each sample prior to fitting.

The algorithm starts with an initial breakpoint detection, followed by fitting of the regression model, more detailed breakpoint detection and some optimization steps.

3.1 Large breakpoints

While failing to model discontinuity in breakpoints delimiting small regions leads to a suboptimal $baseline$ estimate, ignoring read depth jumps in breakpoints delimiting larger regions can severely warp the estimate of $baseline$. In order to get a reasonable initial estimate of $baseline$, these breakpoints need to be identified at the beginning of the procedure, before any regression is done. Small regions are later detected as local variations from the fitted curve.

The extent to which a frame behaves as a copy number breakpoint is quantified in a derivative-like measure In the absence of comparison to a baseline, copy number breakpoints can be identified as sudden jumps in read depth. In other words, locations on a chromosome where the left and right sides have evidently different read depths are likely breakpoints. This idea is quantified in the k-median difference sequence δ_k for some value k.

$$\mathit{left\ depth}_k(i) = \operatorname{median}\{\mathit{depth}_i, \ldots, \mathit{depth}_{i+k-1}\} \tag{3.4}$$

$$\mathit{right\ depth}_k(i) = \operatorname{median}\{\mathit{depth}_i, \ldots, \mathit{depth}_{i+k-1}\} \tag{3.5}$$

$$\delta_k(i) = \textit{right depth}_k(i) - \textit{left depth}_k(i) \tag{3.6}$$

Frames with a high value for δ_k are more likely to be copy number breakpoints. Two adjustments are made to this measure to make it more directly interpretable. Firstly, copy number breakpoints are multiplicative, not additive. In other words, the read depth on the right side of a breakpoint is a continuation of the read depth on the left side *times* the quotient of copy numbers. Thus, the quotient of the depths on the right and left side is a more fundamental measure than the difference. Equivalently, the difference in logarithms Δ_k can be used. In principle, if frame i is a breakpoint, then $\exp(\Delta_k(i))$ should equal the quotient of copy numbers on either side of the breakpoint, but in practice this measure is not precise enough to be useful in that way. Frames with a read depth of 1 or lower are given a log value of 0 in order to avoid extremely low or undefined values.

$$\Delta_k(i) = \log(\textit{right depth}_k(i)) - \log(\textit{left depth}_k(i)) \tag{3.7}$$

Secondly, Δ_k is biased in the shape of a non-constant line, especially in smiley samples. To understand why, first note that the measure δ_k (or Δ_k) is similar to a derivative in interpretation. The analogy between the derivative of a continuous (differentiable) function and the differences of the elements of a discrete sequence is well-known, in formula as well as in interpretation and in usage; for example in the similarity between solving differential equations and difference equations. The described measure δ_k (or Δ_k) is similar to the difference sequence of the read depth, but more robust to noise. From this point of view, the expected shape of δ_k (or Δ_k) – without copy number changes – is a line because it is like the derivative of the read depth smiley curve, which is similar to a parabola. The slope of this line can in practice be quite high and it can obscure peaks that are present around copy number changes. To solve this issue, before identifying the peaks that represent copy number breakpoints, a linear regression is subtracted from the δ_k values in order to remove the linear bias (Figure 3.1).

The actual detection of the breakpoints is done by comparing the corrected Δ_k to a threshold. However, noise often introduces narrow peaks in Δ_k. These noise peaks can be discriminated from a signal around a breakpoint because breakpoints produce a much broader peak. To eliminate thin peaks, the Δ_k values are transformed to a moving average. This heuristic decreases the height of thin peaks much more than those of broad peaks and works acceptably well for removing peaks introduced by bias. Breaks are called in those peak locations that rise above a threshold corresponding to 5 times the median Δ_k value. This value was chosen empirically; it is rather high in order to reduce false positive calls. Telomeres are ignored for this part of the analysis.

The value of k represents a trade-off between accuracy and sensitivity: for low k values, the value is sensitive to noise in the data, and higher k values have increased risk of ignoring smaller copy number variant regions. In addition, a higher k value decreases the precision of the detection method. This is enhanced by the

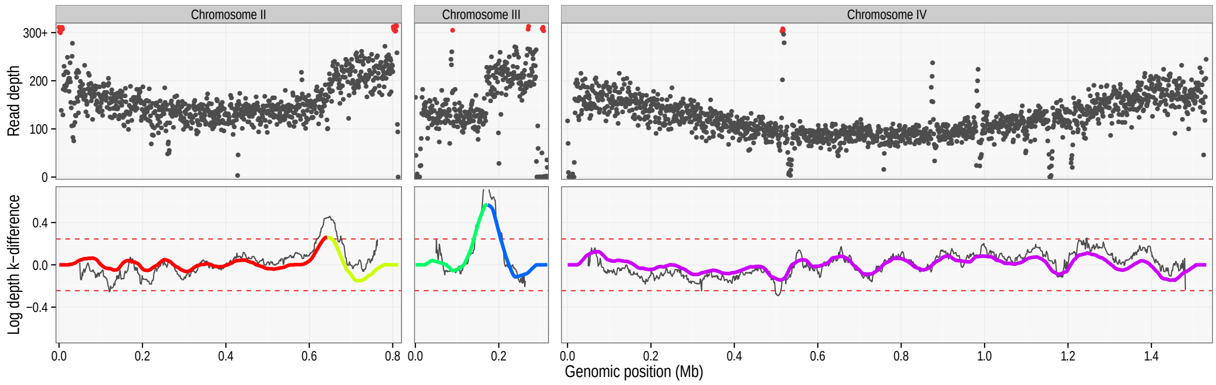


Figure 3.1: Breakpoint detection method. (bottom) The black line shows the uncorrected log depth k-difference Δ_k, the colored line shows the log depth k-difference value after linear correction and smoothing. It has a different color in each inferred copy number region. The read depth profile is shown for reference (top).

fact that larger k values are more sensitive to the linear bias, so that there will be a higher correction which introduces extra imprecision. Finally, the measure is not defined within k frames from the telomeres. This does not pose a problem because the telomeres are too noisy to call breakpoints in this manner, and a breakpoint delimiting large regions cannot be close to the telomeres. Splint uses a rather high value of $k = 50$kb, as only large-scale breakpoints need to be found using this procedure.

3.2 Regression

The conceptual form of the regression is described in equation 3.3, but need to be turned into an explicit regression strategy. The true form of the copy number is known, as it is constant everywhere except for a finite number of known discontinuous locations; in other words, $copynumber$ can be modelled as a piecewise constant step function. Such a function can be expressed as a linear combination of indicator functions[1], one for each region. Hence, the $copynumber$ term follows a linear model. The formula for *baseline*, however, is unknown.

$$copynumber(x) = \sum_{i}^{\#\{\mathit{regions}\}} c_i \chi_{\mathit{region}_i}(x) \tag{3.8}$$

The form of $baseline$ must strike a balance between smoothness and flexibility The model of $baseline$ needs to be flexible enough to capture the unpredictable shape of the smiley bias, but also smooth enough to ignore small copy number variant regions and outliers.

[1] An indicator function χ_A for some set A is a function that returns $\chi_A(x) = 1$ for an argument $x \in A$, and $\chi_A(x) = 0$ for $x \notin A$.

While the smiley bias usually manifests as a parabola-like convex curve, traditional methods of polynomial regression are unsatisfactory for several reasons. First of all, the curve here is not of polynomial shape. Though it looks parabolic on first glance, there is no theoretical basis for this and indeed in practice a parabolic shape proves too stringent for these flexible and irregular profiles. For example, the smiley often flattens out near the telomeres, and some chromosomes (particularly small ones) have irregular non-convex shapes. Higher order polynomials are more flexible, but this also opens up the Pandora's box of overfitting, and they still suffer from the same problem: fitting a polynomial imposes a predefined shape on the baseline curve that does not correspond to the reality. Runge's phenomenon exemplifies this; it is normally described for interpolation but is also applicable to a lesser extent to regression. In other words, the fact that ordinary least squares fitting requires a pre-set parametrized model makes it unfeasible. A non-parametric smoothing method is required.

A popular method used for capturing non-parametric trendlines is LOESS fitting. In LOESS fitting, each prediction is made using a linear model based on the k nearest neighbors. The smiley is however global in nature, and does have some basic shape despite all the complex variations. More importantly, since the $copynumber$ term has a global (eager) model, simultaneous fitting with a local (lazy) model would be impossible because the calculations are inherently done at different moments in time.

Splines allow custom tuning to balance smoothness and flexibility An apt solution can be found starting from smoothing splines. Spline smoothing can be seen as a polynomial regression that is allowed to deviate from its regular shape if it fits the data sufficiently better (see 1.4). The extent to which the smoothing spline deviates from the polynomial shape is defined by the smoothing parameter λ, with $\lambda \to +\infty$ giving a polynomial and $\lambda \to 0$ giving an interpolation spline. Thus, the value of λ defines the balance between smoothness and flexibility.

$$baseline(x) = \sum_{i=0}^{m-1} b_i x^i + \sum_{i=1}^{n} \beta_i \kappa_i(x) \tag{3.9}$$

Smoothing splines offer several other advantages. Unlike most smoothing procedures, the smoothing spline has a closed form, allowing $baseline$ and $copynumber$ to be simultaneously fitted but also cleanly separated. Also unlike most smoothing procedures, smoothing splines are not less performant near the borders of the interval domain (*i.e.* the telomeres), and they offer stable extrapolation. Relatedly, whereas most smoothing methods are purely local, smoothing splines have a global component. This precludes the procedure from overly adapting to sudden local changes such as CNV breakpoints. The main disadvantage of smoothing spline is time complexity, as estimation of a smoothing spline is $O(n^3)$ complex, with n the number of data points, or frames. Note that, because smoothing splines are a regularization procedure, *i.e.* instead of ordinary least squares an extra penalty

Box 3.1: The non-linear regression problem

Using the same terminology as in Box 1.3, define in addition the functions $\chi_0, \ldots, \chi_k$ (with k the number of breakpoints), so that χ_i is the indicator function of the ith interval of constant copy number. Then the model for f obtained by filling in $baseline$ (3.9) and $copynumber$ (3.8) in the regression model 3.3 is of the form $f = (\sum_{i=0}^{\#regions} c_i \chi_i)(\sum_{i=0}^{i<n+M} \beta_i \phi_i)$. Using X^ϕ, X^χ to denote the matrices where $X^\phi_{i,j} = \phi_j(x_i)$ and $X^\chi_{i,j} = \chi_j(x_i)$, the spline minimization can be written as:

$$c, \beta = \operatorname*{argmin}_{c,\beta} \frac{1}{n} \|y - (X^\phi \beta) \cdot (X^\chi c)\|^2 + \lambda \beta^T \Sigma \beta \tag{3.10}$$

Where $\cdot$ denotes the element-wise product of vectors. This non-linear system does not have a closed form solution in c and β.

term is added to the minimization problem, the whole regression 3.3 needs to be fitted as a regularization.

For large chromosomes, splines are fit with $m = 3$, as their basic shape is parabola-like. Smaller chromosomes are fitted with $m = 1$. Because they contain so many outliers this leads to a much more robust fit.

The non-linear regression problem can be approximated by a linear one through logarithmic transformation Equations 3.8 and 3.9 both describe a linear model, but their product is non-linear. This makes the regression very difficult to solve, as non-linear models tend to not have a solution for writing the minimizing function in closed form and indeed might have no unique solution at all (Box 3.1).

On first glance, it might seem that fitting $f \approx baseline + copynumber$ should lead to a similar result and thus can be used as a suitable approximation: both estimates are essentially a spline curve with discontinuities at the breakpoints. The difference in terms of the resulting estimates is that for this additive model the slope of the function remains the same through each breakpoint, rather than being multiplied by the same value as the height of the curve. There is also a more serious problem with this model: the estimated coefficients of the $copynumber$ term are *local* properties, and as such cannot be used as baseline bias-independent estimates of read depth height in a region. Consider that the absolute difference in expected read depth at a breakpoint will be proportional to the baseline value, and is thus lower near the center of the chromosome, than near the terminal regions for a smiley sample. The size of the jump is thereore dependent on its location.

An alternative is to fit the additive model described above on the logarithms of the read depth data. This linear model is of the form $\log f = baseline + copynumber$, so that the resulting estimate for f can be transformed back to:

$$f(x) = \exp\left(\sum_{i=1}^{\#\{\mathit{regions}\}} c_i \chi_{\mathit{region}_i}(x)\right) \times \exp\left(\sum_{i=1}^{m} b_i x^{i-1} + \sum_{i=1}^{n} \beta_i \kappa_i(x)\right) \quad (3.11)$$

This corresponds to the initial model represented in equation 3.3. The estimation of *baseline* is not exactly a smoothing spline, but the exponential of one. As the smoothing spline can adapt its shape to the data, this is merely a conceptual difference with little consequence for the result. More importantly, the logarithmic transformation changes the relative locations of the data points, modifying the effect that an individual point can have on the fit. In the extreme case, a frame with a depth close to 0 has a value close to $-\infty$ after log transformation, so that its effect on the fitting will be so big as to marginalize all other points. By mapping read depths of 1 or less to 0 instead of a negative or undefined value, this extreme is removed, but the core problem remains: small values have more influence in logarithmic fitting than large values. For example, for two close points with values of 10 and 1000, the regression value that minimizes the mean square error of the logarithms is 100, which is much closer to 10 than to 1000 from a non-logarithmic perspective.

Conceptually it seems that this amounts to the estimate being close to the geometric mean, as opposed the the arithmetic mean for a non-logarithmic model, but a major consequence is that small values can easily influence the shape of the whole curve. Frames with very low read depth should influence the depth of the fit (*i.e. copynumber*) appropriately, but not the shape (*i.e. baseline*). In order to achieve this, the points are weighted according to their read depth. The intuition behind this heuristic is that the influence of a small change in the logarithm compared to the original non-logarithmic value is proportional to the derivative of the log function (the reciprocal function). The inverse of this (the identity function) can be used to transform the influence of that change back to its non-logarithmic value. This is however no more than an informal argument.

This heuristic can improve the quality of the fit, but low-depth frames now have no influence on either the shape (*baseline*) or depth (*copynumber*) of the fit, leading to poor fits in low-depth regions that have a few outliers. To remedy this, *copynumber* is fitted again, without weights. The *copynumber* part of the fit is linear when modeled separately from an already known *baseline* curve, and it does not influence the smoothing penalty term of the minimization, so that it can be easily computed. The principle of the logarithmic fit and these corrections are illustrated in Figure 3.2.

3.3 Rescaling

The factor *baseline* in 3.2 serves to replace the factor *coverage*, conceptually generalizing the expectation that *read depth* is proportional to a constant number

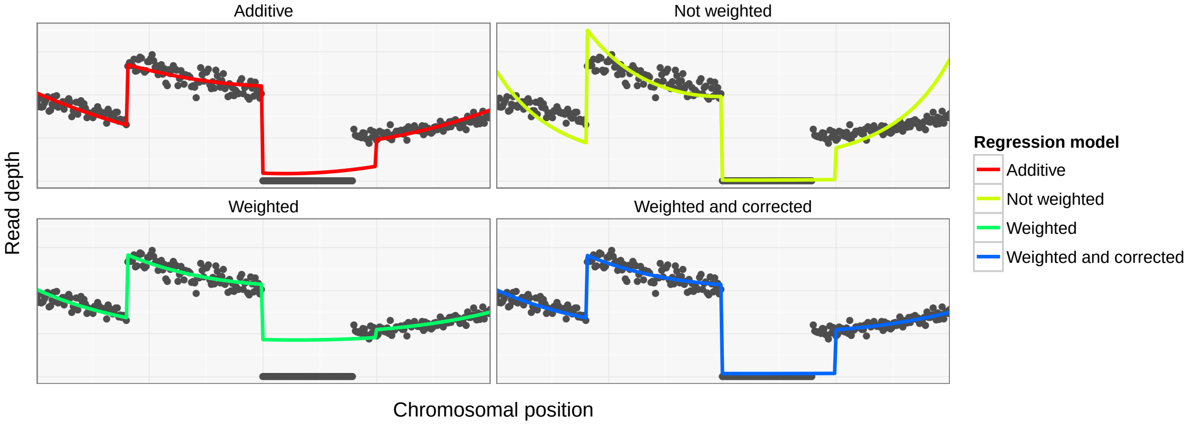


Figure 3.2: Weighted logarithmic fitting. A parabolic model is fit on a set of sample data consisting of a cosine with error multiplied by a piecewise constant step function. (red) The additive model consists of a parabola plus a step function. Unlike the underlying data, only the height and not the curvature of the parabola changes at the discontinuities, giving a mediocre result, especially in extreme regions such as the simulated deletion. (yellow) The non-weighted model is the logarithm of the additive model. It can adapt, producing a near-constant curve in the deleted region, but it is highly sensitive to incorrect breakpoint allocations near such regions. Because the third breakpoint is in the wrong location, the difference between the logarithms of the small and large values demands a considerable slope that skews the entire fit. (green) The weighted model solves this by giving each point a weight proportional to its read depth, so that the small values have such small weight that they do not strongly influence the curve. This also means that the height of the curve in the region of small read depth is mostly dictated by the larger values, which are actually outliers. (blue) The depth is fitted separately without weighting to correct for this in the weighted and corrected model, which gives a satisfactory result. Note that the regression is not perfect because the model is parabolic and the data is not. Using a spline could allow the curve to slightly change shape and fit the data better.

with the expectation of a smooth curve. However, in the regression model 3.3, $copynumber$ is the factor that models the scale of the read depth, whereas $baseline$ models only the shape of the read depth profile background curve. The form 1.1 makes $baseline$ necessarily equal to 1 for $x = 0$, *i.e.* at the left terminal end ($5'$ end) of the chromosome. Thus, the factor *coverage* in practice gets absorbed into $copynumber$, not $baseline$ where it conceptually belongs. As a result, $copynumber$ only equals *copy number*/*ploidy* up to a constant factor. To obtain real estimations of *copy number*/*ploidy*, and to compare copy numbers between different chromosomes and different samples, this factor, which will be called C, must be divided from $copynumber$.

Uder the uniform read depth assumption 3.1, the coverage can be estimated as the genome-wide median read depth value. This calculation is based on the idea that most of the genome is probably not copy number variant, so that the genomic median read depth should give an estimation of the read depth at natural ploidy and copy number. Here, a similar approach is adopted, by estimating C as the genomic median value of $copynumber$.

$$C = \operatorname*{median}_{i}(copynumber_i) \tag{3.12}$$

A better estimation of baseline can be obtained after realizing that $copynumber$ contains most but not all of the scale factor. A small part of the scale is hidden inside $baseline$. For asymmetric bias, because $baseline$ equals 1 at the left ($5'$) end of the chromosome, it will not equal 1 on the right ($3'$) end of the chromosome. A different but equivalent estimation of $baseline$, noted $baseline_{\textit{right}}$ would equal 1 on the right side of the chromosome, and would vary on the left side. The scale factor C_{right} calculated from the corresponding values of $copynumber$ is then relative to a different but equally good baseline. We can take advantage of this redundancy to balance the estimation of C, obtaining a new, better estimate C'. This approach amounts to using both the left and right ends of the chromosomes to estimate the globally central scale of the read depth, as opposed to using only the left ends. (In 3.14, *right end(i)* refers to the right-most frame on the chromosome of frame i)

$$C_{\textit{left}} = \underset{i}{\text{median}}(copynumber_i) \tag{3.13}$$

$$C_{\textit{right}} = \underset{i}{\text{median}}(copynumber_i/baseline_{\textit{right end}(i)}) \tag{3.14}$$

$$C' = \sqrt{C_{\textit{left}}C_{\textit{right}}} \tag{3.15}$$

Using this genome-wide scale factor, the estimates $baseline$ and $copynumber$ can be rescaled to $C'baseline$ and $\frac{1}{C'}copynumber$ and interpreted as estimates of *baseline* and *copy number*/*ploidy* as in 3.2.

3.4 Small breakpoints

Splint identifies breakpoints that delimit large regions at the beginning of every run. The same method cannot be used for finding small regions due to the nature of the trade-off that comes with choosing a value k. Instead, smaller regions of variant copy number are detected using a hidden Markov model (HMM) approach.

Splint's HMM is fully non-parametric The most straightforward Markov chain for modeling CNVs explicitly models the copy number at each genomic position. The observed signals in the corresponding HMM are the read depths in each frame. This requires a probability distribution function of read depth to be defined for each copy number. While the results are not highly sensitive to the precise values of such a distribution, no model adequately captures the distribution of read depth, especially in the smiley samples, due to local read depth biases. Moreover, the ploidy of these samples is not known, and a HMM that can model both tetraploid and triploid samples is far too specific to give correct results. Instead, the actual copy number estimation is left to the regression, and the real purpose of the HMM is simply to find variant regions.

Since there is no need for absolute copy numbers, the model used in this algorithm is fully non-parametric. Rather than directly using the read depth values, the signals of the HMM are comparisons of the read depth to the estimated regression curve. There are only three such signals, representing i) read depth higher ($> 30\%$) than the estimated value (*signal+*), ii) read depth lower ($< 30\%$) than the estimated value (*signal-*) and iii) read depth similar to the estimated value (*signal1*). There are three states, *state+*, *state-*, *state1*, each of which is favored by its corresponding signal (Table 3.1). A fourth state, *state0*, is added that corresponds to regions with almost exclusively *signal-* signals, to increase performance near total deletions. Transition probabilities $e_{a \to b}$ are $1 - 3p$ for $a = b$ and p if $a \neq b$, with $p = \frac{0.05}{1000}$*frame size*.

	signal+	*signal1*	*signal-*
state+	0.5	0.45	0.05
state1	0.15	0.7	0.15
state-	0.05	0.45	0.50
state0	0.9	0.05	0.05

Table 3.1: Emission probabilities of Splint's hidden Markov model..

For each chromosome, the state sequence of optimal probability is computed using the Viterbi algorithm (from R's HMM package, v. 1.0), and each frame in which the state changes is annotated as a copy number breakpoint. After this, the baseline is refitted and the HMM is run again until no more new variant regions are found.

3.5 Optimization

As mentioned previously, the log k-median difference method for detecting breakpoints delimiting large regions is necessary at the start of the procedure, but it is not very precise. Breakpoint locations are often a few frames off, and in some cases they can be as far as 0.05Mb away from the correct location. It is important to relocate these breakpoints in order to get accurate results. This problem does not occur generally with breakpoints detected by the HMM method, as the Viterbi procedure already optimizes the breakpoint location.

Frames that might have been set in the wrong region are identified using the hidden Markov model Because it is so precise, the HMM and the Viterbi algorithm can be recycled to identify inferred breakpoints that might be in the wrong location. For each region, a new regression is fitted on the whole chromosome, using only the frames in the region as a training set. Because it represents only one region of constant copy number, there is no $copynumber$ factor. The fitted curve is a spline with $m = 1$ and a loose (low) value for λ, representing a rough estimate of the central values. This smoothing procedure was chosen because it has constant extrapolation. The HMM is run, and any extension of the region that is called in the *state1* state is annotated as possibly belonging to that region. Considering that this procedure is performed for each region, all frames that might

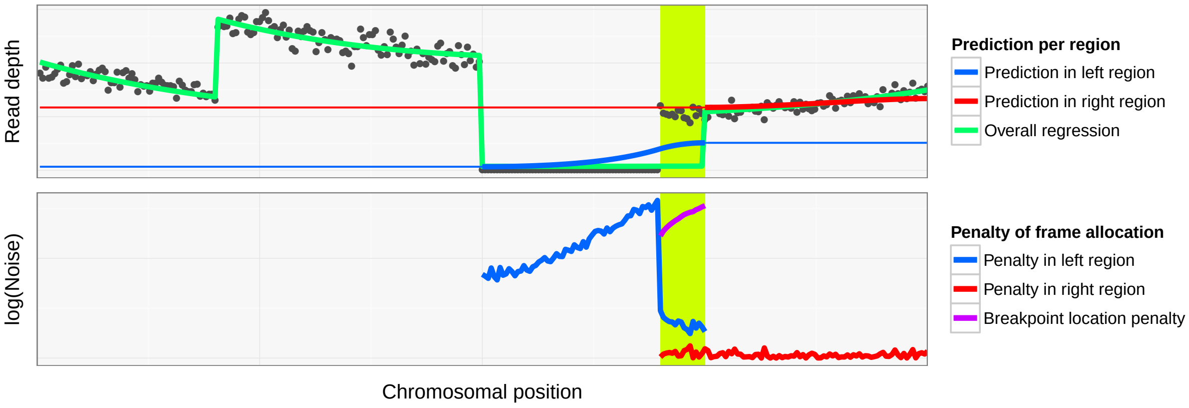


Figure 3.3: Breakpoint optimization procedure. The same sample data are the same as in 3.2, illustrating a chromosome with a breakpoint called in an incorrect location. (top) The region-specific baseline predictions (regressions) are shown in blue and red. Predictions span the whole chromosome, but are only fitted on one region. Extrapolation (thin line) is constant. Because the frames just to the left of the rightmost region correspond to the predicted curve (this would be measured by the HMM), the colored region is contested between the two regions. (bottom) The penalty for assigning each frame to either region is shown in red and blue. For each position i in the contested region, the penalty P_i is shown (purple). The minimal value of the purple line, or the new position of the breakpoint, is all the way to the left, so that the entire contested region will be assigned to the rightmost region.

belong to multiple regions are marked as ‘contested’ and will serve as possible new locations for their associated breakpoint(s). A region in which all frames are marked as possibly belonging to another region is immediately absorbed into the other region.

Contested breakpoints are relocated to the position that minimizes the deviation between observed read depth and the region-specific baselines

For each contested region, the breakpoint is relocated in a small optimization procedure. The new location of the breakpoint is the one that minimizes the deviance (measured as the quotient) between the observed read depth values of the frames in the surrounding region, and the expected read depth values of the regions that they are allocated to by the location of the breakpoint. In other words, the breakpoint is relocated to the position i that minimizes the penalty P_i (Figure 3.3).

$$\begin{aligned} P_i = &\sum_{j<i} \max\left\{\frac{\mathit{read\ depth}_i}{\mathit{prediction}_{\mathit{left},j}}, \frac{\mathit{prediction}_{\mathit{left},j}}{\mathit{read\ depth}_i}\right\} \\ &+ \sum_{j>i} \max\left\{\frac{\mathit{read\ depth}_i}{\mathit{prediction}_{\mathit{right},j}}, \frac{\mathit{prediction}_{\mathit{right},j}}{\mathit{read\ depth}_i}\right\} \end{aligned} \tag{3.16}$$

After moving the breakpoints, the procedure is repeated until no further changes are made.

Adjacent regions that differ by less than 15% or regions on the same chromosome that are close to the genomic nominal value are merged. The last step in the procedure is the merging of similar regions on a chromosome, eliminating false positive breakpoint calls. Whenever two adjacent regions differ in depth by less than 15%, they are not considered different enough to keep them separate. The breakpoint between such two regions is removed, the regression is repeated and the process starts again until no more adjacent regions are similar enough to merge.

Additionally, any regions on a chromosome with a read depth close to the genomic nominal value (within a 15% margin) are merged, forcing their read depth to be the same.

Part 4

Results & discussion

To detect CNVs in the collection of 147 industrial *Saccharomyces cerevisiae* samples, SPLINT was run using frames of 1000bp and 500bp. In theory, using smaller frames can give more accuracy and allows detection of smaller CNV regions, but at the cost of higher noise and increased dependency of neighboring datapoints. An overlay of the results of the two frame sizes revealed several differences, especially in large CNVs (≈ 0.1Mb and more). Possible reasons are listed below. These dubious CNVs are likely artefacts of the smiley correction method, and most further analyses were done considering only the variations that were called using both 1000bp and 500bp frames (Figure 4.1). Data from the mitochondrial DNA was discarded because its read depth is highly erratic.

In total, 15288 variant regions were detected across all samples, covering on average approximately 1.57Mb per sample. There are considerably more deletions (11259 regions, 1.07Mb per sample) than amplifications (4029 regions, 0.50Mb per sample). On average, there are 112349 CNVs in coding regions (39189 amplifications, 73160 deletions) throughout all 147 genomes. Note that all of these values are relative to the reference sequence, thus parts of the genome that do not occur in the reference are not counted, and multiple changes in the same region are counted only once.

Below is an evaluation of SPLINT by visual inspection of its results, as well as a comparison with CNVs observed using a more established platform NEXUS™ (version 7.5) (*Nexus Copy Number Discovery Edition* n.d.). Using SPLINT's results, several patterns are revealed when comparing CNV between different locations on the genome or across different samples within the population. Finally, our data are compared to previous observations regarding specific genes with CNV described in literature.

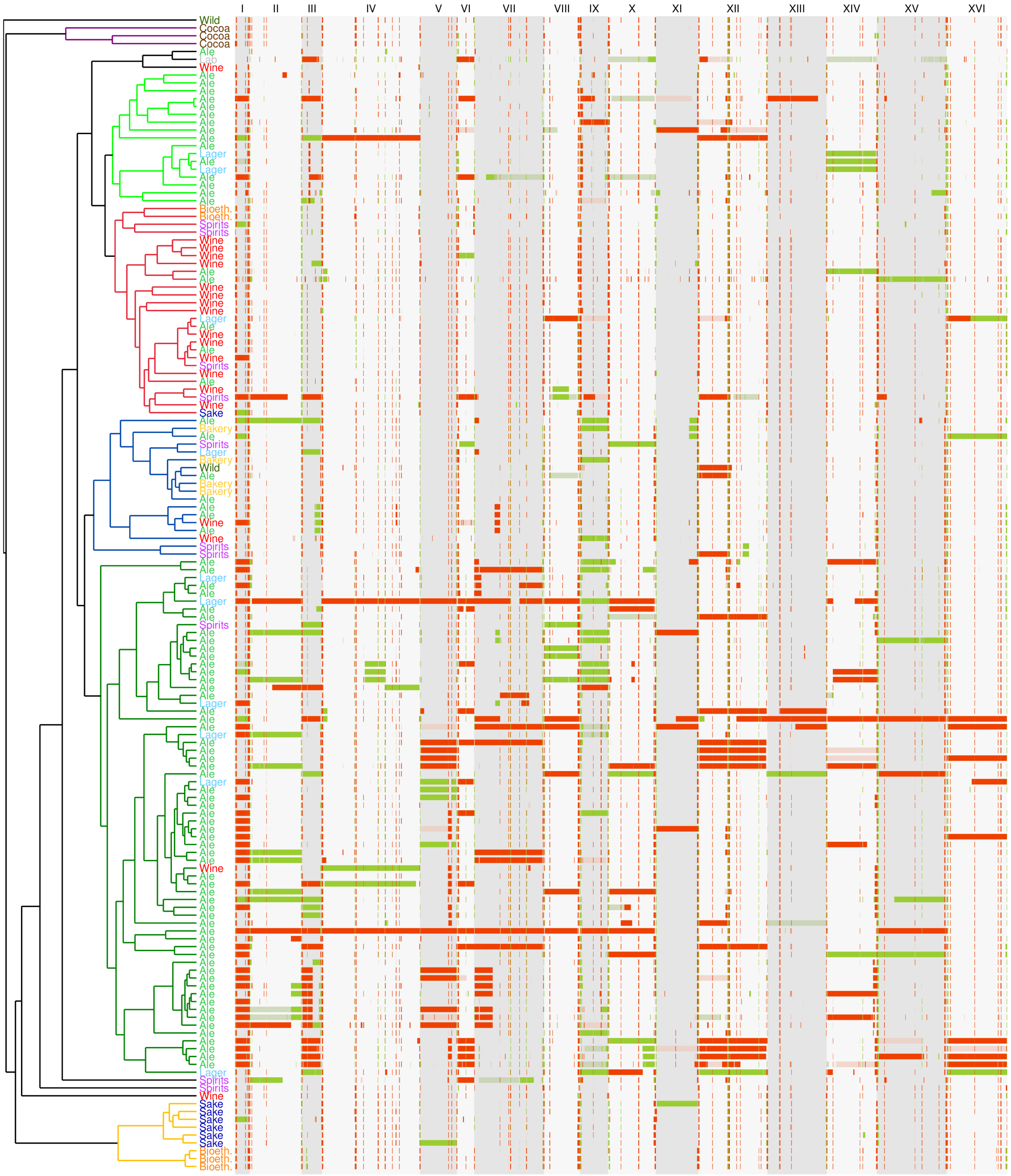


Figure 4.1: CNV profile of each of the 147 samples. Samples are sorted by phylogeny (see Figure 1.1). Red indicates loss, green indicates gain of copy number. Gains or losses that were observed when running the algorithm on 1000bp frames, but not on 500bp frames, are faded. The single lab strain is S288C.

4.1 Evaluation

Visual inspection

Overall, inferences made by SPLINT appear to be confirmed by visual examination of its results, both for smiley (Figure 4.2a) and non-smiley (Figure 4.2b) samples. It is sensitive to small regions and small variations. However, some caveats should be added before interpreting SPLINT's results (Figures 4.2c, 4.2d).

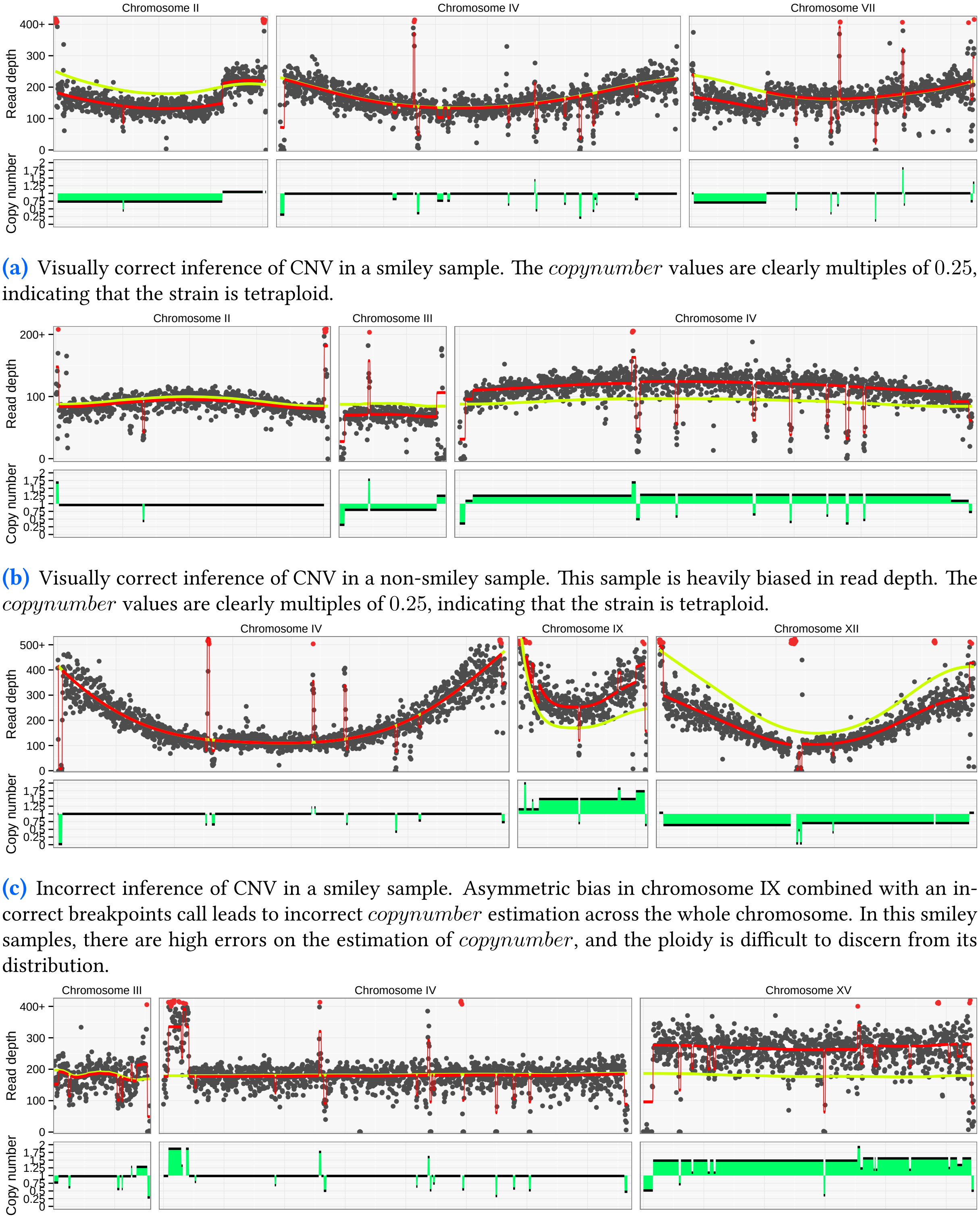
The assumption of equal bias at terminal regions is volatile One of the vulnerabilities of SPLINT's correction process is the assumption that different chromosomes in the same sample have similar read depth bias in the terminal regions. This assumption is necessary in order to obtain copy number estimates (see 3.3), but there is considerable variance on the read depth in terminal regions and some chromosomes can be asymmetric. Many non-smiley samples also show considerable large-scale bias in read depth. Their profile has a different character, and the subtelomeric regions may not be the best reference points in these samples.

Numeric results are of low quality Qualitatively, SPLINT is successful at detecting many CNVs in the data, even if the region is small in size or variation. Quantitatively, the numeric values found for the copy number changes must be interpreted with caution. Contrary to what would be expected in an ideal case, the (rescaled) $copynumber$ values are not always close to multiples of $1/\mathit{ploidy}$. There is high noise on this value, which obfuscates differentiation between a variation of a single copy number and two copy numbers, especially in smiley samples.

Errors in $copynumber$ are composed of local errors (introduced by outlier frames and sampling variance), and errors in chromosome-wide depth estimation (introduced by volatility of the assumptions that bias in terminal regions is similar). The latter type of error can be eliminated by rescaling $copynumber$ according to the chromosomal instead of the genome-wide median. This disables chromosome-wide errors, but also detection of whole-chromosome CNV. It also makes the read depth values more difficult to interpret, as they are no longer expected to be multiples of $1/\mathit{ploidy}$, but of $1/\mathit{chromosomal\ copy\ number}$, which is not constant in aneuploid samples.

Small-scale bias can have large-scale consequences As a consequence of the baseline read depth assumption, incorrect fitting of the baseline causes incorrect copy number estimation across a whole chromosome. This can be caused by local bias (Figure 4.2c). The fact that local bias can affect read depth estimation across a whole chromosome is probably SPLINT's weakest feature.

Chromosomes I, III, VI, IX and XII are of low quality In the small chromosomes (I, III, VI) of smiley samples, variance in read depth is high and the smiley



(a) Visually correct inference of CNV in a smiley sample. The *copynumber* values are clearly multiples of 0.25, indicating that the strain is tetraploid.

(b) Visually correct inference of CNV in a non-smiley sample. This sample is heavily biased in read depth. The *copynumber* values are clearly multiples of 0.25, indicating that the strain is tetraploid.

(c) Incorrect inference of CNV in a smiley sample. Asymmetric bias in chromosome IX combined with an incorrect breakpoints call leads to incorrect *copynumber* estimation across the whole chromosome. In this smiley samples, there are high errors on the estimation of *copynumber*, and the ploidy is difficult to discern from its distribution.

(d) Incorrect inference of CNV in a non-smiley sample. CNV detection in a sample with very high local bias. SPLINT calls many small CNV regions that are probably due to bias. As copy number values appear to be mostly centered around multiples of 0.5, this strain is probably a diploid, but it is unclear.

Figure 4.2: Examples of results produced by SPLINT. Parts of the read depth profiles of several samples are shown (black) together with the fitted regression (red). The yellow line shows the *baseline* factor (after rescaling, see 3.3). The (rescaled) *copynumber* term (green) approximates the *copy number*/*ploidy* quotient.

bias can have unpredictable shapes. As a result, sudden but continuous changes are virtually indistinguishable from copy number breakpoints. In chromosome IX and XII, the smiley bias has different characteristics than in the other chromosomes (Figure 2.2a), which may lead to false positive inferences (Figure 4.2c).

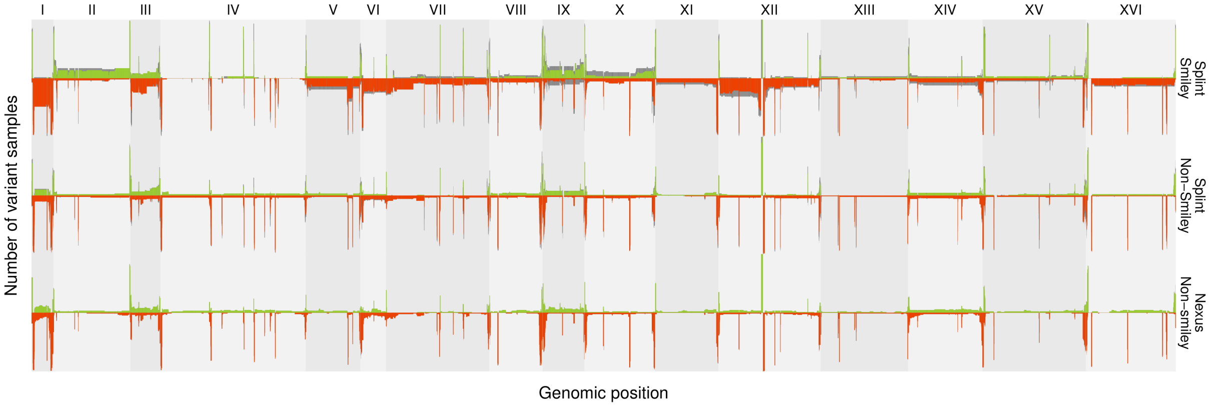
Splint is too sensitive for samples with high local noise Some samples have higher local variance in read depth than others. This can be caused simply by low coverage, but also by unknown local bias factors. In the data presented here, some of the samples sequenced at the highest coverage are also the worst samples in terms of local variance. This does not exclusively affect smiley samples (Figure 4.2d), although in general local variance is higher for smiley than for non-smiley samples. High local variance means that Splint is much more likely to call false positive CNVs simply due to random sampling.

In particular, the reference sample, lab strain S288C, was sequenced at the highest coverage, but also displayed one of the most extreme smileys (median read depth $261X$ vs. telomeric baseline read depth $485X$, *cfr.* Figure 2.3), and local variance among the highest of all samples. Consequently, many false positive CNVs were called in this strain, which was originally intended as a negative control (Figure 4.1). The sequenced isolate of S288C is not exactly the same as the reference genome, but it is unlikely that more than a few detected CNVs are correctly inferred.

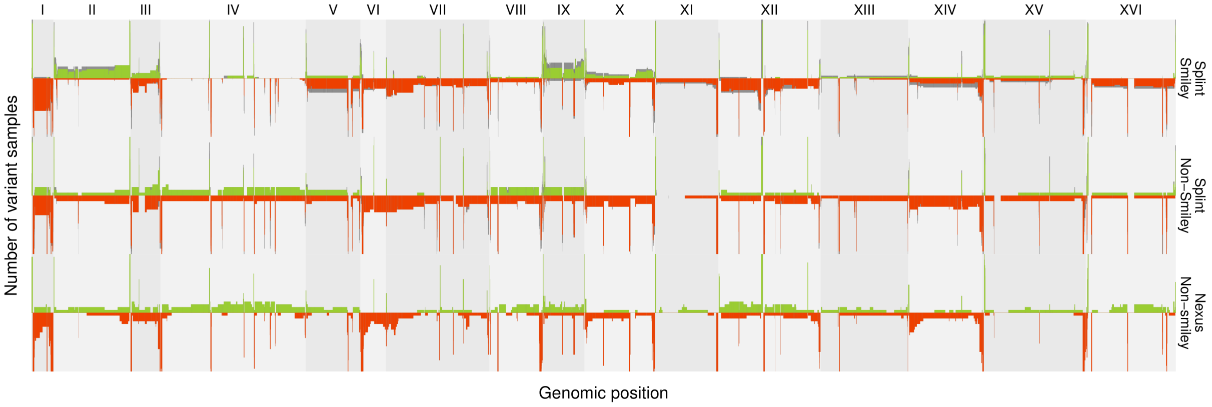
Nexus comparison

Splint's inference on non-smiley samples can be compared directly with other software such as Nexus. For smiley samples, Nexus is not able to correctly infer copy number changes. In order to assess the quality of Splint's results on all samples, patterns are compared between smiley and non-smiley samples. This leaves three groups of results: results from Splint on smiley and non-smiley samples, and results from Nexus only on smiley samples.

On non-smiley samples, Splint is similar to Nexus, but calls more variants Generally, Splint is more sensitive, measuring a higher rate of CNV than Nexus (Figure 4.4). In total, Nexus infers 5655 CNVs (1414 amplifications, 4243 deletions), covering an average of 1.20Mb per sample (0.51Mb amplification, 0.69Mb deletion). By comparison, on the same subset of 88 non-smiley samples, Splint infers 8349 CNVs (1893 amplifications, 6456 deletions) for an average of 1.31Mb per sample (0.45Mb amplification, 0.87Mb deletion). Both algorithms find more deletions than amplifications, but this difference is higher in Splint's results than in those of Nexus. This is likely due to a small number of false positive whole chromosome deletions. The CNV profiles produced by Splint and Nexus are qualitatively similar (Figure 4.3).



(a) CNV profiles obtained using all samples. The results from the non-smiley samples are similar when analyzed with Nexus or Splint. The smiley group is different. This can be partially explained by the fact that smiley samples are unequally distributed among different phylogenetic groups, with most smileys occurring in samples that are expected to have high copy number variance. But the smiley samples also have a higher false positive rate, as evidenced by the higher gray bars, specifically for chromosome-wide CNV.



(b) CNV profiles made using only the Beer 1 group. Some of the features that appear specific to the smiley samples in 4.3a are in fact common in the Beer 1 group, which contains most of the smiley samples. The profiles of the non-smiley samples according to Splint and Nexus are more different than in 4.3a due to lower aggregation.

Figure 4.3: Comparison of the copy number profiles inferred by Splint and Nexus. Three groups of results are compared: CNV profiles of smiley samples inferred by Splint, and non-smiley samples inferred by Splint and Nexus. The green and red bars show the number of amplifications and deletions, respectively, in each genomic location. Gray bars represent results obtained by Splint with 1000bp but not 500bp frames.

Smiley samples contain more false positives than non-smiley samples Comparing smiley and non-smiley samples (using results from either Splint or Nexus), smiley samples appear to be more variant (Figure 4.3a). This is in part because smiley samples are unequally distributed among different lineages, and some lineages are more variant than others (see below). Considering only the samples from the Beer 1 group, which contains most of the smiley samples, eliminates this bias (Figure 4.3b), but differences remain. On average, 1.98Mb of the genome is affected by CNV in smiley samples (0.58Mb amplification, 1.40Mb deletion).

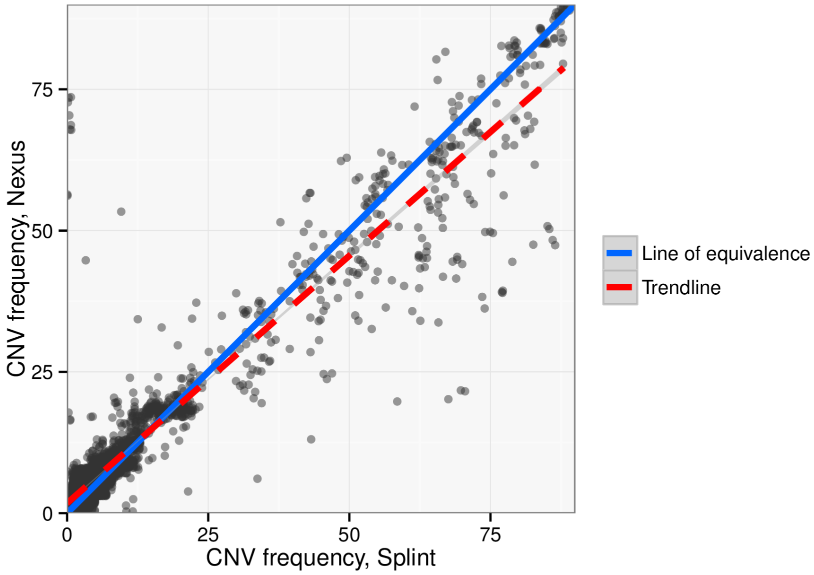


Figure 4.4: Comparison of Splint and Nexus in terms of sensitivity and conservativity. Each dot represents a gene. The number of samples carrying copy number variance according to Splint and Nexus is plotted on the horizontal and vertical axis, respectively. Genes located on the blue line are equally variant according to both algorithms. The red line shows a linear regression. For most genes, Splint is either equally or more sensitive than Nexus.

4.2 Genomics

Splint was run on each of the 147 *S. cerevisiae* samples (Figure 4.1). Below, general emergent patterns are described, comparing either different locations in the genome, or different samples in the population. In the final section, CNV of specific genes of interest is compared with literature.

Genome level

CNV is not uniformly spread across the genome (Figure 4.5). In consistence with literature, CNV is enriched in telomeric regions and appears to target non-random groups of genes. Notably, CNV is not convincingly more present in coding re-

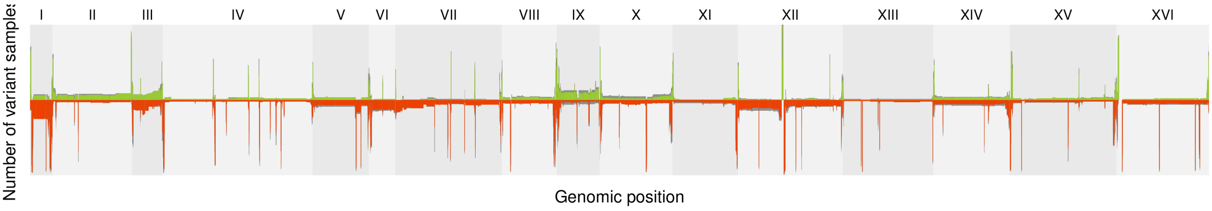


Figure 4.5: Overall CNV profile of all samples..

Amplified			Deleted		
Term	P-value	#	Term	P-value	#
Processes					
maltose metabolism	0.000	9	maltose metabolism	0.000	8
cellular amide catabolism	0.000	6	carbohydrate transport	0.000	12
allantoin metabolism	0.001	5	transmembrane transport	0.003	35
allantoin catabolism	0.001	5	flocculation	0.027	5
disaccharide metabolism	0.003	10	asparagine catabolism	0.034	4
oligosaccharide metabolism	0.022	10			
Functions					
sucrose α-glucosidase	0.000	6	carbohydrate transport	0.000	13
helicase	0.000	19	transmembrane transport	0.000	34
glucosidase	0.000	10	fructose transport	0.000	7
alpha-glucosidase	0.000	7	mannose transport	0.000	7
beta-fructofuranosidase	0.000	6	glucose transport	0.001	7
DNA helicase	0.001	9	hexose transport	0.001	9
carbon-nitrogen lyase	0.001	6	mannose binding	0.002	4
oligo-1,6-glucosidase	0.002	4	asparaginase	0.011	4
O-glycosyl hydrolase	0.002	10			
amine-lyase	0.004	4			
glutamine hydrolysing[1]	0.004	4			
glycosyl hydrolase	0.006	11			
hydrolase	0.006	58			

Table 4.1: GO enrichment of 5% most amplified, resp. deleted genes. The column # shows the number of genes that were variant. Enrichment analysis done using YeastMine (Balakrishnan *et al.* 2012)

gions than in non-coding regions, affecting on average approximately 7.8% of the genome, and 8.6% of the genes in each sample.

CNV is frequent in telomeres Overall, CNV affects samples with a per-chromosome base frequency, representing whole-chromosome and large-scale CNVs, and several thin peaks of local variations occurring with high frequency. The highest peaks are located near the center of chromosome XII, at the locus of the ribosomal genes. These amplifications and deletions occurring in 100% of the samples are artefacts of mappability. Many of the other peaks are located near the sides of chromosomes, *i.e.* in the subtelomeric regions. Defining subtelomeres as the most extreme 33kb on each chromosome (Brown, Murray, *et al.* 2010), 7115 CNVs (2652 amplifications, 4463 deletions) were located entirely within a telomeric region, affecting 0.35Mb (0.13Mb amplification, 0.22Mb deletion) on average per sample (*cfr.* 8173 CNVs were not subtelomeric, affecting 1.22Mb per sample).

Not all subtelomeres are equally prone to CNV. Previous reports of differences in subtelomeric copy number variance have been inconsistent in appointing the most and least variant regions. For example, one paper attributed the most subtelomeric copy number variation to chromosomes V, VII, X, XIV, XV and XVI (Ames *et al.* 2010), while another found the least variation in the subtelomeres of chromosomes II, XIII, XIV and XVI (Dunn, Richter, *et al.* 2012). Here, the most variant telomeres are those of chromosome I, VI, VIII, X, XI, XII, XV and XVI.

Some chromosomes are more variant than others For whole-chromosome CNV, by far the most variant chromosome is chromosome I. Aneuploidy of chromosome I is particularly frequent in the Beer 1 group (Figure 4.1). This deletion appears to be scattered throughout the phylogeny of the Beer 1 lineage, suggesting multiple deletion and possible re-amplification events. Alternatively, this signal may be a product of sequencing bias, although it is not unique to smiley samples (Figure 4.3b). Similarly, chromosome IX is often amplified, possibly as a result of asymmetric smiley bias (Figure 2.2a), but it is equally frequent in smiley and non-smiley samples, although only when considering both 1000bp and 500bp frames. Amplification of chromosome II is mostly concentrated in smiley samples, but there is no conspicuous aspect of the smiley bias that is specific to chromosome II. Finally, deletion of chromosome XII is almost certainly related to the smiley pattern, which produces severely deformed depth profiles in this chromosome.

Amplification of chromosome XIII has been implicated as a positive adaptation for S288C to growth on low-glucose medium in *in vitro* conditions (Selmecki *et al.* 2015). In the present study, this chromosome is among the least variant in terms of copy number.

A small number of genes is highly variant in copy number As expected, a small number of genes is much more prone to CNV than the rest of the genome (Figure 4.6). It is known that highly copy number variant genes are enriched for GO

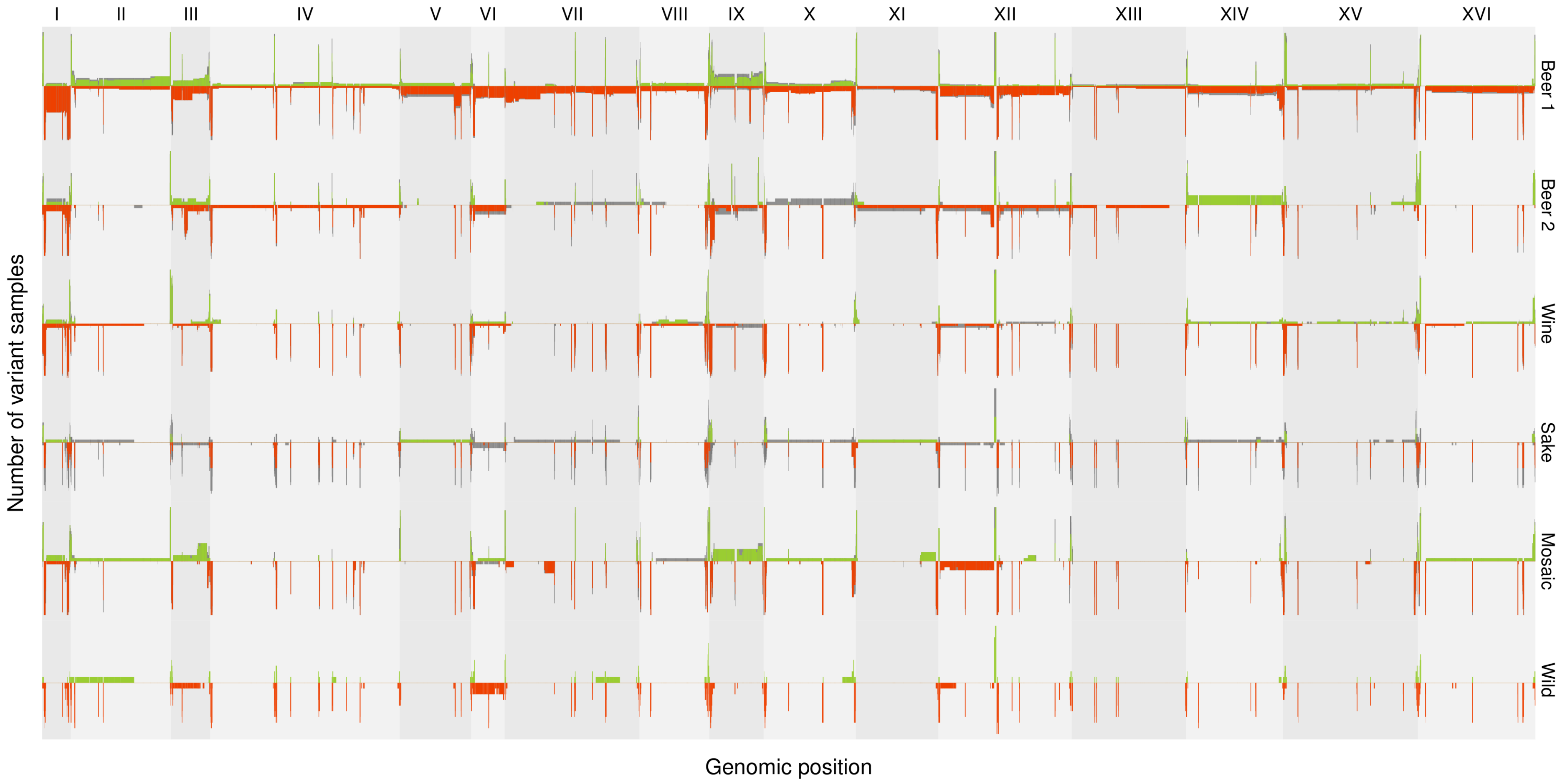


Figure 4.7: **Copy number profiles of the six lineages**. See: (Figure 1.1)

terms relating to flocculation, stress response, metal transport and carbohydrate usage (Carreto *et al.* 2008; Dunn, Lavine, *et al.* 2005). Here, only flocculation and carbohydrate usage are enriched (Table 4.1). It should be noted that, using a simple, custom hypergeometric test, more terms were significantly enriched, many relating to metal ion transport, stress response (particularly nitrogen-related), other sugars and alcohol metabolism.

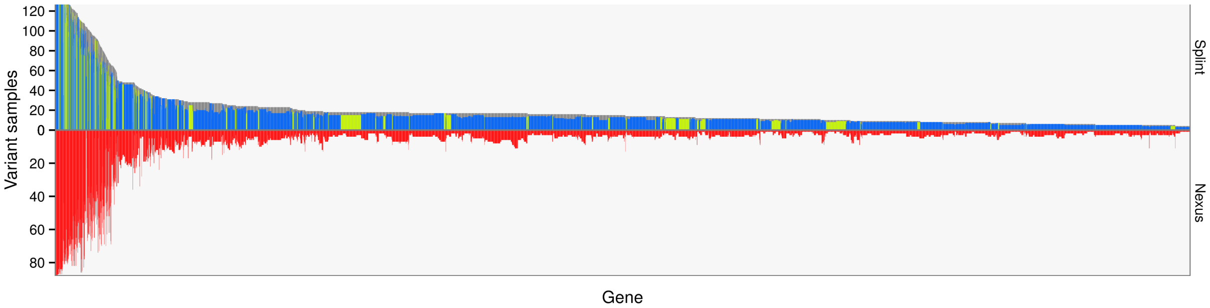


Figure 4.6: Histogram showing the copy number variability of each gene. Each bar represents a gene, sorten by frequency of copy number variation in all samples. Gray (up-facing) bars represent results from SPLINT using 1000 basepair frames, blue (up-facing) bars represent results from SPLINT that were found both in the 1000 base pair and 500 base pair frame approaches, red (down-facing) bars represent results from NEXUS. Telomeric genes (up to 33kb from the chromosome ends) are colored in yellow. Note that the results for NEXUS were obtained using only the non-smiley samples.

Population level

The CNV profiles of different lineages have both shared and unique properties (Figure 4.7).

Beer 1 strains have complex genomes One of the most striking patterns of divergence between lineages is the distinctive character of the Beer 1 group. Many strains in this group appear to be tetraploid (Figure 4.8). Perhaps relatedly, many Beer 1 strains are highly variant in copy number, with some strains having almost 70% of their genome affected (Figure 4.9). Curiously, this pattern does not arise when looking purely at the number of CNVs per strain. On average, 2.21Mb (0.67Mb amplification, 1.54Mb deletion) of the genome is affected by CNV in non-smiley samples, and 2.00Mb (0.55Mb amplification, 1.45Mb deletion) in smiley samples. The unexpected difference is likely due to a few non-smiley outliers that have large portions of their genome deleted.

CNV is more related to evolutionary history than niche The high variability and polyploidy of Beer 1 strains is not present in Beer 2. Other signs of convergent evolution between the two groups are similarly absent. Defining 'highly variable' genes as genes that are within the 5% (328) most copy number variable

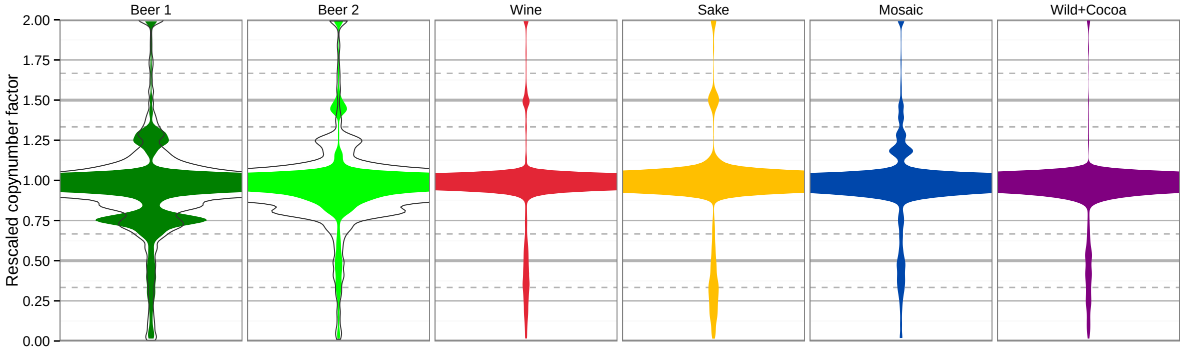


Figure 4.8: Violin plot showing the distribution of (rescaled) *copynumber* **values between strains of different lineages**. Beer 1 is the only group showing a noticeable fraction of regions with a depth of 0.75 or 1.25, a clear sign of tetraploidy. These distributions are based on non-smiley samples only. For the lineages containing a non-negligible number (> 3) of smiley samples, the same signal is added based on smiley samples (black line). The *copynumber* estimates of smiley samples have higher error. Note that these distributions are aggregated, and signals around 0.75 and 0.25 from individual tetraploid samples in groups other than Beer 1 may be obscured.

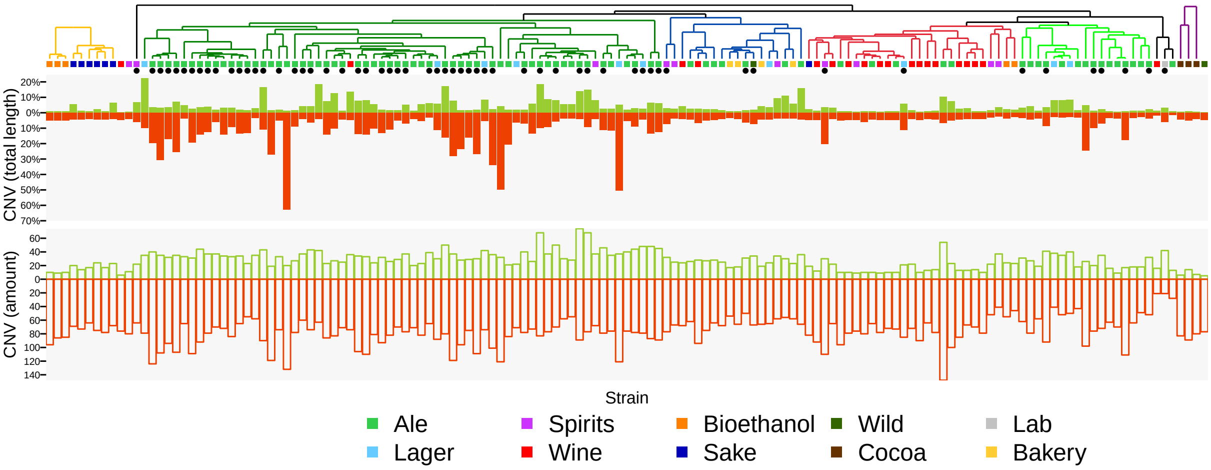


Figure 4.9: Copy number variability across the phylogenetic tree. The color of the squares mark the industry of each strain. Smiley samples are annotated with a black dot. The extent of amplification (green), and the extent of deletion (red) are shown, expressed in affected percentage of the genome (top, full) and in number of CNVs counted (bottom, empty). Beer 1 strains are more variable than other lineages, but only in terms of covered percentage, not in amount of variations. Only the reference and its closest relatives have noticeably fewer copy number variations.

genes of a set of samples, respectively 87 and 75 genes are highly variable in Beer 1 and Beer 2 but not in the general population. However, these two sets do not overlap: every gene that is highly variable in Beer 1 and in Beer 2 is also highly variable in the general population. In other words, all genes that are highly variable in one group are either also highly variable in general, or not highly variable in the other group, and thus not convergent in copy number variance.

Several more systematic analyses were done to find evidence of CNV shared between strains of the same industry but not the same lineage, not only between the two brewing lineages. These include methods based on Shannon entropy, principal component analysis and neural networks. None of these methods could recapitulate the source environment of the samples based on their CNV pattern. Because they also do not have enough power to convincingly show a negative result, these methods are not included in this thesis.

By contrast, our results consistently show shared CNV between strains that are related at all levels of the phylogenetic tree (Figure 4.1).

Gene level

Many authors have observed that some industries promote CNV of specific genes, and have concluded that these CNVs are adaptive. This conclusion is usually framed within the idea that higher copy number of a gene (gene dosage) leads to increased expression and thus influences the phenotype. However, almost all studies on CNV in yeast were based on a small number of strains (typically 4 to 10) and, especially until 2009 (Liti, Carter, *et al.* 2009), represented only a small fraction of the *S. cerevisiae* population. Most research has focused on strains isolated from wineries or vineyards whereas insight in the CNV profile of brewing yeast has been lacking. For these reasons, it is interesting to investigate how previously described CNVs are characterized within the context of a broad phylogeny. Below is a short list of genes and gene families that have received special attention in many other studies, together with our results.

CUP1 The CUP1 gene codes for a metallothionein that bestows resistance to high concentrations of copper (Cu) and cadmium (Cd). The reference genome contains two paralogs for this gene, called CUP1-1 and CUP1-2, which lie in the same locus. It is one of the most well-known copy number variant loci in *S. cerevisiae*. Amplification of CUP1 has long been recognized as a beneficial adaptation that increases resistance to high copper concentrations (Fogel & Welch 1982; Warringer *et al.* 2011). It is also deleted in many wine strains (Carreto *et al.* 2008; Dunn, Lavine, *et al.* 2005), although a more recent population-wide phenotype study found amplification of CUP1 in the Wine/European and Sake lineages (Warringer *et al.* 2011). Copper concentrations are often high in vineyards due to the usage of copper sulphate as an antibacterial stabilizing agent.

Here we report a scattered distribution of CUP1 CNVs across the phylogeny, including both amplifications and deletions. Compared to many other loci, CUP1

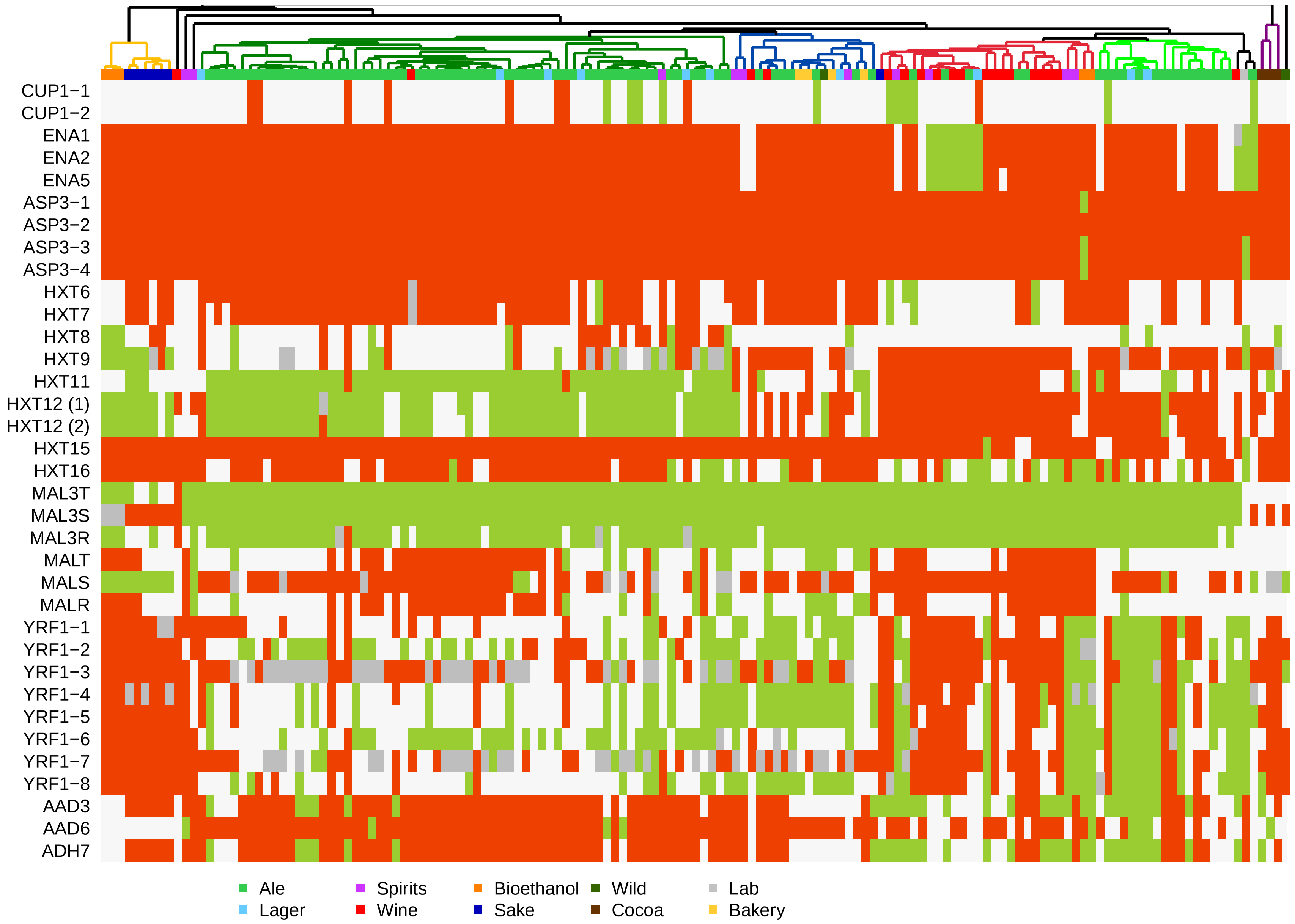


Figure 4.10: CNV map of some interesting genes. For eech of the discussed genes, a tile represents gain (green), loss (red), nominal copy number (white) or, disagreement between 1000bp and 500bp methods (gray). The different strains are ordered by phylogeny (top). The colored box represents the industry of origin, a black dot marks smiley samples.

is relatively stable in terms of copy number. A small subset of Wine/European strains has acquired an amplification in this locus.

ENA ENA1 and ENA2 code for sodium transporters and give resistance to high concentrations of sodium (Na^+) and lithium (Li^+). These genes are widely amplified in the reference genome, as they are almost universally found as deleted in other strains by comparison (Carreto *et al.* 2008; Dunn, Lavine, *et al.* 2005; Dunn, Richter, *et al.* 2012; Stambuk *et al.* 2009). The third paralog that is present in the reference genome, ENA5, is only sometimes reported as deleted (Stambuk *et al.* 2009), and sometimes not (Carreto *et al.* 2008; Dunn, Richter, *et al.* 2012). This is largely corroborated by our results, which include even an amplification in our S288C sample and the strains closest related to it. Interestingly, a sublineage of the Wine/European group also has an amplification in the ENA genes relative to the reference.

ASP3 Genes of the ASP3 family, which encode asparaginase, are similarly considered amplified in the reference (Carreto *et al.* 2008; Dunn, Lavine, *et al.* 2005; Stambuk *et al.* 2009). Correspondingly, a deletion of the ASP3 genes is present in nearly all of our samples. ASP3,4 also appear to be amplified in our S288C sample. Difference in copy number of ASP3 has also been proposed as a possible way to distinguish different types of beer yeast (Pope *et al.* 2007). Variation of ASP3 in beer yeast is not observed here, possibly because we did not include *Saccharomyces pastorianus* in this study.

HXT Several genes of the HXT family have been implicated as beneficial CNV adaptations. These genes encode transmembrane hexose transporters. Evolution experiments have shown that HXT6/7 amplification can occur in glucose-limited environments, resulting in increased expression of the transporter genes and increased access to the limiting resource (Brown, Todd, *et al.* 1998; Selmecki *et al.* 2015). In high-glucose environments, HXT6/7 can be maladaptive (Wenger *et al.* 2011). HXT9, HXT11 and HXT12 have been found deleted in wine strains (Carreto *et al.* 2008). HXT6, HXT9, HXT15 and HXT16 are thought to be deleted in bioethanol and wild strains (Stambuk *et al.* 2009).

Widespread duplication of HXT6/7 duplication is not present in our data. They are frequently deleted but not universally. HXT9, HXT11 and HXT12 are deleted in most of the Wine/European cluster, but not specifically in wine strains from other lineages. Deletion of HXT15 and HXT16 is frequent but not universal. Notably, these genes are amplified in the bioethanol strains of the Wine/European lineage, but deleted in the bioethanol strains of the Sake lineage.

MAL MAL is a family of genes coding for proteins related to hexose transport and metabolism, including a high-affinity transporter (MALT), a maltase (MALS) and transcription factor activating other MAL genes (MALR). They are located in the complex locus MAL1 on chromosome VII, and a second copy (MAL3T, MAL3S, MAL3R) is located in the MAL3 complex locus on chromosome II. Both loci are subtelomeric. MAL1 genes have been found deleted in some wine strains (Carreto *et al.* 2008). These genes are highly copy number variant (Brown, Murray, *et al.* 2010).

Interestingly, the MAL3 locus is amplified in most strains, except those isolated from the wild and cocoa fermentation, as well as the sake strains of the Sake lineage. This is one of the few apparently adaptive copy number changes in our data. MAL1, on the other hand, appears frequently deleted in some sublineages which are known heavy maltose users.

YRF1 Deletion of the YRF1 genes has been described as a universal CNV in non-laboratory strains, from research in wine and clinical strains (Carreto *et al.* 2008). This deletion is here shown to be far from universal, even in wine strains, and YRF1 appears to have a highly variant copy number across strains.

ADH and ADD ADH and AAD encode for alcohol dehydrogenases and would be expected to have increased copy numbers in many samples under a model where CNV is adaptive by change in gene dosage and consequently gene expression. AAD3 and ADH7 have been found amplified in some wine strains (Carreto *et al.* 2008; Dunn, Lavine, *et al.* 2005), and variable in beer yeast (Pope *et al.* 2007). Comparatively high copy number of ADH6 has been described as one of the most 'wine-like' CNV properties (Dunn, Richter, *et al.* 2012).

AAD3 and ADH7 are amplified in a specific sublineage of wine strains, but not in general, and is deleted in many sake and beer strains. ADH6 is only slightly less deleted in the Wine/European lineage compared to the large ale group, where the deletion is almost universal.

Part 5

Conclusion

Several papers in the last decade have investigated CNV in yeast strains. Many have found a significant relation between CNV of specific genes and source environment. Some have interpreted this as a likely adaptation that increases fitness in a particular niche. Adaptation of a CNV would be mediated by increase or decrease in gene dosage, and a consequent change in gene expression. Some of these papers have confirmed phenotypic change in transformed yeast strains with variant copy numbers. Thus, the commonly held view is that most CNV in *Saccharomyces cerevisiae* is directly adaptive in the short term by changing gene dosage and gene expression. At the same time, the adaptive potential of CNV as a method for changing genomic plasticity in the long term is widely recognized (Ohno 1970). The results presented above suggest that most CNV in industrial yeast is not a short-term adaptation.

Evidence for the gene dosage theory is circumstantial Most of the evidence for gene dosage based CNV adaptation is based on small numbers of genes, and a small number of strains. Using a small sample size can be dangerous because source environment is so strongly confounded with evolutionary history. This is especially true when samples are not consciously selected to reflect different parts of the *S. cerevisiae* population, such as in studies that predate Liti's monumental paper in 2009 (Liti, Carter, *et al.* 2009). Many studies have concluded adaptation because certain phenotypic of genetic traits correlate with source environment, without considering that source environment may be confounded with evolutionary history.

At the same time, extrapolating small-scale results to the whole genome is dangerous because the amount of unpublished negative results is unknown. For example, the prevalence of CUP1 deletion in industrial strains is contrary to the gene dosage theory, and may have only been described in literature because CUP1 duplication was investigated before it could be studied in natural strains. Comprehensive studies on the relationship between CNV and adaptation in *S. cerevisiae* are needed but lacking (Kondrashov 2012).

One paper has reported on a genomic scale that many CNVs are related to ecological environment, rather than phylogenetic relation, and are therefore likely to be adaptive (Ames *et al.* 2010). Using hierarchical clustering based on GO enrichment of copy number variant genes in each strain, the authors were able to obtain a phenetic tree which, according to them, clusters the strains by source environment. However, only laboratory strains are located close together in the tree, whereas the other two groups present in their analysis, wild and (industrial) fermentation, are scattered. Moreover, their analysis contains only four laboratory strains, which are phylogenetically clustered in groups of two, and only three out of the four strains are close together in the phenetic tree.

Another high-level result is the enrichment of highly copy number variant genes for GO terms relating to sugar utilization and metal transport (Carreto *et al.* 2008; Dunn, Lavine, *et al.* 2005). These are typical processes that need to be adaptive on short term, as different media require different phenotypes. This observation apparently adds to the evidence for the gene dosage theory because CNV would change expression of these genes and therefore add to the evolvability of these traits. However, many of these GO terms are also found in subtelomeric regions (Brown, Murray, *et al.* 2010), which are more variant in copy number. While it is possible that these genes are located in the subtelomeric regions because it allows faster adaptability through CNV, it can also be interpreted as confounding.

There is also evidence that contradicts the gene dosage theory. Strikingly, only approximately 25% of *S. cerevisiae* genes show a direct relation between gene dosage and expression (Kvitek *et al.* 2008). This suggests that, for the majority of genes, copy number variations cannot be interpreted straightforwardly as modifications of gene expression. Furthermore, if CNV is primarily selected by its effect on gene dosage, an enrichment of CNV would be expected in coding regions. Such a pattern was not observed in the data presented in this study.

CNV may be an important evolutionary mechanism for other reasons than changing gene expression through dosage In the present study, we have observed that CNV in industrial *S. cerevisiae* is more closely related to evolutionary history than ecological niche. Many of the CNVs that were previously described as putative adaptations to a specific industry, were found here to be more lineage-related, with little evidence of convergent evolution. We were not able to regenerate meaningful industry-related CNV patterns that were not confounded by evolutionary dependency. These findings contradict the theory that most CNV in *S. cerevisiae* is adaptive by changing gene expression, at least on short term.

As an alternative hypothesis, we propose that CNV may be adaptive by perturbing the evolutionary landscape and plasticity of the genome. A higher copy number can remove evolutionary pressure on one copy of a genomic sequence, result in accelerated evolution and a better phenotype in the mid- to long run. At the same time, deletions may be adaptive in other cases because multiple copies of a gene can buffer the phenotypic effect of mutations, lowering evolvability. This way of thinking about CNV is well established in literature, but is often ignored in current population genomics studies.

Future work

With only a year of time to develop Splint and analyze its results, the analysis of this data has only just begun. The first concern is validation of some of Splint's results using the quantitative polymerase chain reaction (qPCR). Further, a phenotypic analysis could give more insight in the relationship between CNV and phenotype, which is crucial for understanding how this interesting type of polymorphism is situated within the process of evolution in industrial environments. Phenotypic data is available for the strains that are used here, but adequately exploring it is enough for a separate thesis project on its own. Correlating CNV with other mutations such as SNP and InDels in the affected regions would help investigate the relationship between CNV and genomic adaptability. Finally, adding transcriptome sequencing to this project would greatly increase the potential to link CNV to gene expression, and possibly provide stronger evidence for the main conclusion of this work.

Publishing Splint

These results clearly show that Splint is a useful tool for yeast genomics research, especially because of its unique ability to analyze biased samples. However, in its current version, Splint is not ready for general use.

The most obvious constraint is time. Currently, running Splint on a single sample divided in 1000bp frames takes approximately 6-10 minutes on our server. Using 500bp frames, it takes 2-3 hours. The cubic time complexity of the spline fitting procedure clearly makes it unpractical to use on larger genomes such as those sequenced from human cells. In order to remedy this, the spline would have to be approximated by a less time-complex method. Such approximations exist (Kim & Gu 2004), but are typically described in dense mathematics and are difficult to implement.

Other than time, some internal code would have to be adapted to incorporate different formats, some options would have to be made adjustable and defensive error catching would need to be added, which is not native to the programming language R. It may be more efficient to rewrite the program in a more universal and robust platform such as Java.

Bibliography

Aa, E., Townsend, J. P., Adams, R. I., Nielsen, K. M. & Taylor, J. W. Population structure and gene evolution in *Saccharomyces cerevisiae. FEMS Yeast Research* **6,** 702–715 (2006).

Ames, R. M. *et al.* Gene duplication and environmental adaptation within yeast populations. *Genome Biology and Evolution* **2,** 591–601 (2010).

Bakalinsky, A. T. & Snow, R. The chromosomal constitution of wine strains of *Saccharomyces cerevisiae. Yeast* **6,** 367–382 (1990).

Balakrishnan, R. *et al.* YeastMine—an integrated data warehouse for *Saccharomyces cerevisiae* data as a multipurpose tool-kit. *Database* **2012,** 1–8 (2012).

Bergström, A. *et al.* A high-definition view of functional genetic variation from natural yeast genomes. *Molecular Biology and Evolution* **31,** 872–888 (2014).

Boeva, V. *et al.* Control-FREEC: a tool for assessing copy number and allelic content using next-generation sequencing data. *Bioinformatics* **28,** 423–425 (2012).

Borneman, A. R. *et al.* Whole-genome comparison reveals novel genetic elements that characterize the genome of industrial strains of *Saccharomyces cerevisiae. PLoS Genetics* **7,** e1001287 (2 2011).

Brown, C. J., Todd, K. M. & Rosenzweig, R. F. Multiple duplications of yeast hexose transport genes in response to selection in a glucose-limited environment. *Molecular Biology and Evolution* **15,** 931–942 (1998).

Brown, C. A., Murray, A. W. & Verstrepen, K. J. Rapid expansion and functional divergence of subtelomeric gene families in yeasts. *Current Biology* **20,** 895–903 (2010).

Carreto, L. *et al.* Comparative genomics of wild type yeast strains unveils important genome diversity. *BMC Genomics* **9,** 524 (2008).

Christiaens, J. F. *et al.* The fungal aroma gene *ATF1* promotes dispersal of yeast cells through insect vectors. *Cell Reports* **9,** 425–432 (2014).

Cromie, G. A. *et al.* Genomic sequence diversity and population structure of *Saccharomyces cerevisiae* assessed by RAD-seq. *G3 (Bethesda, Md.)* **3,** 2163–2171 (2013).

Curtin, C. D. & Pretorius, I. S. Genomic insights into the evolution of industrial yeast species *Brettanomyces bruxellensis. FEMS Yeast Research* **14,** 997–1005 (2014).

DNeasy Blood & Tissue Kit Qiagen. <https://www.qiagen.com>.

Dudley, R. Ethanol, fruit fipening, and the historical origins of human alcoholism in primate frugivory. *Integrative and Comparative Biology* **44,** 315–323 (2004).

Dunn, B., Lavine, R. P. & Sherlock, G. Macroarray karyotyping of commercial wine strains reveals shared, as well as unique, genomic signatures. *BMC Genomics* **6** (53 2005).

Dunn, B., Richter, C., Kvitek, D. J., Pugh, T. & Sherlock, G. Analysis of the *Saccharomyces cerevisiae* pan-genome reveals a pool of copy number variants distributed in diverse yeast strains from differing industrial environments. *Genome Research* **22,** 908–924 (2012).

Engel, S. R. *et al.* The reference genome sequence of *Saccharomyces cerevisiae*: then and now. *G3 (Bethesda, Md.)* **4,** 389–398 (2014).

Fay, J. C. & Benavides, J. A. Evidence for domesticated and wild populations of *Saccharomyces cerevisiae. PLoS Genetics* **1,** e5 (1 2005).

Fogel, S. & Welch, J. W. Tandem gene amplification mediates copper resistance in yeast. *Proceedings of the National Academy of Sciences of the United States of America* **79,** 5342–5346 (1982).

Forsythe, G. E., Malcolm, M. A. & Moler, C. B. Computer methods for mathematical computations. *Journal of Applied Mathematics and Mechanics* **59,** 141–142 (1977).

Gibson, B. R., Lawrence, S. J., Leclaire, J. P. R., Powell, C. D. & Smart, K. A. Yeast responses to stresses associated with industrial brewery handling. *FEMS Microbiology Reviews* **31,** 535–569 (2007).

Gibson, B. & Liti, G. *Saccharomyces pastorianus*: genomic insights inspiring innovation for industry. *Yeast* **32,** 17–27 (2015).

Goddard, M. R. & Greig, D. *Saccharomyces cerevisiae*: a nomadic yeast with no niche? *FEMS Yeast Research* **15,** fov009 (2015).

Goddard, M. R. Quantifying the complexities of *Saccharomyces cerevisiae*'s ecosystem engineering via fermentation. *Ecology* **89,** 2077–2082 (2008).

Goffeau, A. *et al.* Life with 6000 genes. *Science* **274,** 546–567 (1996).

Gottschling, D. E., Aparicio, O. M., Billington, B. L. & Zakian, V. A. Position effect at *S. cerevisiae* telomeres: reversible repression of Pol II transcription. *Cell* **63,** 751–762 (1990).

Horowitz, H., Thorburn, P. & Haber, J. E. Rearrangements of highly polymorphic regions near telomeres of *Saccharomyces cerevisiae. Molecular and Cellular Biology* **4,** 2509–2517 (1984).

Johnston, J. R. Brewing and distilling yeasts. *Yeast Technology* (eds Spencer, J. F. T. & Spencer, D. M.) 55–14 (Springer-Verlag, Berlin, Germany, 1990).

Johnston, J. R., Baccari, C. & Mortimer, R. K. Genotypic characterization of strains of commercial wine yeasts by tetrad analysis. *Research in Microbiology* **151,** 583–590 (2000).

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

Kim, J. M., Vanguri, S., Boeke, J. D., Gabriel, A. & Voytas, D. F. Transposable elements and genome organization: a comprehensive survey of retrotransposons revealed by the complete *Saccharomyces cerevisiae* genome sequence. *Genome Research* **8,** 464–478 (1998).

Kim, Y. J. & Gu, C. Smoothing spline Gaussian regression: More scalable computation via efficient approximation. *Journal of the Royal Statistical Society. Series B: Statistical Methodology* **66,** 337–356 (2004).

Klambauer, G. *et al.* cn.MOPS: mixture of Poissons for discovering copy number variations in next-generation sequencing data with a low false discovery rate. *Nucleic Acids Research* **40,** e69 (2012).

Knight, S. & Goddard, M. R. Quantifying separation and similarity in a *Saccharomyces cerevisiae* metapopulation. *The ISME Journal* **9,** 361–370 (2015).

Kondrashov, F. A. Gene duplication as a mechanism of genomic adaptation to a changing environment. *Proceedings of the Royal Society B: Biological Sciences* **282,** 5048–5057 (2012).

Kvitek, D. J., Will, J. L. & Gasch, A. P. Variations in stress sensitivity and genomic expression in diverse *S. cerevisiae* isolates. *PLoS Genetics* **4,** e1000223 (10 2008).

Legras, J. L., Merdinoglu, D., Cornuet, J. M. & Karst, F. Bread, beer and wine: *Saccharomyces cerevisiae* diversity reflects human history. *Molecular Ecology* **16,** 2091–2102 (2007).

Libkind, D. *et al.* Microbe domestication and the identification of the wild genetic stock of lager-brewing yeast. *Proceedings of the National Academy of Sciences of the United States of America* **108,** 14539–14544 (2011).

Liti, G. The fascinating and secret wild life of the budding yeast *S. cerevisiae. eLife* **4,** 1–9 (2015).

Liti, G., Barton, D. B. H. & Louis, E. J. Sequence diversity, reproductive isolation and species concepts in *Saccharomyces. Genetics* **174,** 839–850 (2006).

Liti, G., Carter, D. M., *et al.* Population genomics of domestic and wild yeasts. *Nature* **458,** 337–341 (2009).

Martini, A. Origin and domestication of the wine yeast *Saccharomyces cerevisiae. Journal of Wine Research* **4,** 165–176 (1993).

Masneuf, I., Hansen, J., Groth, C., Piskur, J. & Dubourdieu, D. New hybrids between *Saccharomyces* sensu stricto yeast species found among wine and cider production strains. *Applied and Environmental Microbiology* **64,** 3887–3892 (1998).

Mortimer, R. K. & Johnston, J. R. Genealogy of principal strains of the yeast genetic stock center. *Genetics* **113,** 35–43 (1986).

Mortimer, R. & Polsinelli, M. On the origins of wine yeast. *Research in Microbiology* **150,** 199–204 (1999).

Mortimer, R. K. Evolution and variation of the yeast (*Saccharomyces*) genome. *Genome Research* **10,** 403–409 (2000).

Nexus Copy Number Discovery Edition BioDiscovery, Inc. <http://www.biodiscovery.com/software/nexus-copy-number/>.

Nosedal-Sanchez, A., Storlie, C. B., Lee, T. C. & Christensen, R. Reproducing kernel Hilbert spaces for penalized regression: a tutorial. *The American Statistician* **66,** 50–60 (2012).

Ohno, S. *Evolution by gene duplication* (Springer-Verlag, New York, 1970).

Papalexandratou, Z. & De Vuyst, L. Assessment of the yeast species composition of cocoa bean fermentations in different cocoa-producing regions using denaturing gradient gel electrophoresis. *FEMS Yeast Research* **11,** 564–574 (2011).

Pavelka, N. *et al.* Aneuploidy confers quantitative proteome changes and phenotypic variation in budding yeast. *Nature* **468,** 321–325 (2010).

Pérez-Ortín, J. E., García-Martínez, J. & Alberola, T. M. DNA chips for yeast biotechnology: the case of wine yeasts. *Journal of Biotechnology* **98,** 227–241 (2002).

Piškur, J., Rozpędowska, E., Polakova, S., Merico, A. & Compagno, C. How did *Saccharomyces* evolve to become a good brewer? *Trends in Genetics* **22,** 183–186 (4 2006).

Pope, G. A. *et al.* Metabolic footprinting as a tool for discriminating between brewing yeasts. *Yeast* **24,** 667–679 (2007).

Pubmed Accessed 2015-05-10. National Center for Biotechnology Information. <http://www.ncbi.nlm.nih.gov/pubmed>.

R Development Core Team. *R: A Language and Environment for Statistical Computing* R Foundation for Statistical Computing. <http://www.R-project.org>.

Ramazzotti, M., Berná, L., Stefanini, I. & Cavalieri, D. A computational pipeline to discover highly phylogenetically informative genes in sequenced genomes: application to *Saccharomyces cerevisiae* natural strains. *Nucleic Acids Research* **40,** 3834–3848 (2012).

Schacherer, J., Shapiro, J. A., Ruderfer, D. M. & Kruglyak, L. Comprehensive polymorphism survey elucidates population structure of *Saccharomyces cerevisiae. Nature* **334,** 997–1003 (2009).

Schifferdecker, A. J., Dashko, S., Ishchuk, O. P. & Piškur, J. The wine and beer yeast *Dekkera bruxellensis. Yeast* **31,** 323–332 (2014).

Selmecki, A. M. *et al.* Polyploidy can drive rapid adaptation in yeast. *Nature* **519,** 349–352 (2015).

Smith, S. D., Kawash, J. K. & Grigoriev, A. GROM-RD: resolving genomic biases to improve read depth detection of copy number variants. *PeerJ* **3,** e836 (2015).

Stambuk, B. U. *et al.* Industrial fuel ethanol yeasts contain adaptive copy number changes in genes involved in vitamin B1 and B6 biosynthesis. *Genome Research* **19,** 2271–2278 (2009).

Stefanini, I. & Dapporto, L. Role of social wasps in *Saccharomyces cerevisiae* ecology and evolution. *Proceedings of the National Academy of Sciences of the United States of America* **109,** 13398–13403 (2012).

Wang, Q. M., Liu, W. Q., Liti, G., Wang, S. A. & Bai, F. Y. Surprisingly diverged populations of *Saccharomyces cerevisiae* in natural environments remote from human activity. *Molecular Ecology* **22,** 5404–5417 (2012).

Warringer, J. *et al.* Trait variation in yeast is defined by population history. *PLoS Genetics* **7,** e1002111 (6 2011).

Wenger, J. W. *et al.* Hunger artists: Yeast adapted to carbon limitation show trade-offs under carbon sufficiency. *PLoS Genetics* **7,** e1002202 (8 2011).

Wiens, F. *et al.* Chronic intake of fermented floral nectar by wild treeshrews. *Proceedings of the National Academy of Sciences of the United States of America* **105,** 10426–10431 (2008).

Winzeler, E. A. *et al.* Genetic diversity in yeast assessed with whole-genome oligonucleotide arrays. *Genetics* **163,** 79–89 (2003).

Xu, H. & Boeke, J. D. High-frequency deletion between homologous sequences during retrotransposition of Ty elements in *Saccharomyces cerevisiae. Proceedings of the National Academy of Sciences of the United States of America* **84,** 8553–8557 (1987).

Zhao, M., Wang, Q., Wang, Q., Jia, P. & Zhao, Z. Computational tools for copy number variation (CNV) detection using next-generation sequencing data: features and perspectives. *BMC Bioinformatics* **14,** Suppl 11:S1 (2013).