Copyright is owned by the Author of the thesis. Permission is given for a copy to be downloaded by an individual for the purpose of research and private study only. The thesis may not be reproduced elsewhere without the permission of the Author.

The evolution of selfish genetic elements within bacterial genomes

A thesis submitted in partial fulfilment of the requirements for the degree

of

Ph.D.

in

Molecular Evolution

at Massey University, Auckland, New Zealand.

Frederic Bertels

2012

Abstract

Genes that increase their copy number relative to that of the host genome are termed selfish. Selfish genes are found ubiquitously in bacterial genomes. Within genomes they can often be identified due to their repetitive nature. Short repetitive sequences such as repetitive extragenic palindromic (REP) sequences have been proposed to be selfish genetic elements. However, evidence for the selfishness of REPs is scarce due to the lack of knowledge about their origin, evolution and mechanisms of dispersal. Here, REPs are studied in the model bacterium *Pseudomonas fluorescens* SBW25. The evidence provided suggests that REPs are part of a greater mobile genetic element, which is termed REP doublet forming hairpins (REPINs).

Subsequently, I investigate the cause of REPIN dispersal: a putative transposase. The transposase, named REP-associated tyrosine transposase (RAYT) shares essential motifs with the IS*200* family of insertion sequences. However, unlike insertion sequences, RAYTs are found only as single copy genes. This indicates that RAYTs may not be entirely selfish; instead they may have been co-opted by the host to perform a beneficial function.

Finally, two more repetitive sequence classes are studied in the SBW25 genome. Interestingly, both sequence classes consist of a protein coding sequence and a sequence that forms a stable secondary structure in single stranded DNA or RNA. This arrangement is reminiscent of bacterial toxin-antitoxin (TA) systems. Evidence from sequence analyses suggests that the repetitive nature of these elements in SBW25 may be the result of cooperation between REPINs or other replicative elements and the TA systems.

The presented analyses show that despite the streamlined nature of bacterial genomes selfish genetic elements frequently arise, replicate and probably increase their

persistence and spread through cooperation with addictive and duplicative elements respectively.

Acknowledgements

Foremost, I would like to thank my supervisor Professor Paul Rainey for all the advice, guidance and inspiration he has given me over the past three years. Without him the research presented in this thesis would not have been possible, and I hope we keep in touch for years to come. Further, I would like to thank my co-supervisors Dr Justin O'Sullivan and Professor Allen Rodrigo for not only supporting me during the course of my PhD, but also for their help during my first year in New Zealand. Their fascination for research and science was one of the main reasons why I decided to stay in New Zealand for my PhD.

Another essential factor for the completion of a PhD is funding. Hence, I am very grateful for a doctoral scholarship from the Allan Wilson Centre. It was great being part of the Allan Wilson Centre. I especially enjoyed the scientific exchange and social activities at the annual meetings.

The scientific and social interactions with past and present members of the Rainey Lab also played an important role in my research. I had a great time and I hope to meet all of you again! I would like to especially thank Dr Jenna Gallie for endless advice and discussion. Jenna also proofread countless manuscripts and thesis attempts and taught me how to improve my English on many occasions. Dr Xue-Xian Zhang and Yunhao Liu were of great help during my attempts to conduct lab work.

Furthermore I would like to thank Ben Kerr and the rest of the Kerr lab for stimulating discussions and advice during my stay in Seattle, WA. The time in Seattle and the Kerr lab greatly supported my professional development by exposing me to new ideas and different types of thinking.

Finally, I would like to thank my parents Elke and Ralf Bertels, my sister Helen Bertels, my brothers Felix and Julian Bertels and again my fiancée Jenna Gallie for providing

the support I needed to finish this thesis. I also would like to thank Elaine Riley for helping me move around the world, and numerous other things that made my life much easier.

Table of Contents

Table of Abbreviations

Abbreviation	Meaning
BIMEs	Bacterial interspersed mosaic elements
BLAST	Basic local alignment search tool
BLASTP	BLAST Protein (protein query against protein database)
BLASTN	BLASTN Nucleotide (nucleotide query against nucleotide database)
TBLASTN	Translated BLASTN (protein query against nucleotide database)
bp	base pairs
CAS genes	CRISPR associated genes
CRISPRs	Clustered regularly interspaces short palindromic repeats
ERICs	Enterobacterial repetitive intergenic consensus sequences
IR	Inverted repeat
IS	Insertion sequence
LARDs	Large retrotransposon derivatives
LINEs	Long interspersed elements
LTR	Long terminal repeat
MITE	Miniature inverted repeat transposable elements
NEMISs	*Neisseria* miniature insertion sequences
NGS	Next-Generation Sequencing
PSK	Post segregational killing
PU	Palindromic units
ORF	Open reading frame
RAYTs	REP-associated tyrosine transposase
REPs	Repetitive extragenic palindromic sequences
REPINs	REP doublets forming hairpins
RMS	Restriction modification system
RNAi	RNA interference
RUP	Repeat unit of pneumococcus
SDR	Small dispersed repeats
SINEs	Short interspersed element
ssDNA	Single stranded DNA
TA	Toxin-antitoxin system
TPRT	Target primed reverse transcription
TRIMs	Terminal-repeat retrotransposons in miniature

Chapter 1:
Introduction

1.1 The role of DNA sequence amplification in life

DNA sequence amplification encompasses a wide variety of processes, among which semiconservative DNA replication is the most important and the basis of life, since all known organisms require it for copying their genetic information (genome) in order to leave offspring (arguably RNA viruses are an exception depending on one's definition of life) [1]. In multicellular organisms DNA replication is not only required to reproduce, but also to generate a variety of differentiated somatic cells. Somatic cells, unlike germ cells, contain almost exact copies of the genome, but cannot pass on their genetic information to the next generation. However, DNA contained within somatic cells can be modified and sometimes amplified to, for example, increase cell size and gene expression levels, as in polytene chromosomes from *Drosophila* [2, 3]. In contrast, DNA amplification events such as chromosome and genome duplications in the germline (cells that pass on their DNA to the next generation) are rarely observed and retained within populations, due to deleterious dosage effects [4]. Nonetheless, these events have probably occurred multiple times over the course of eukaryotic genome evolution and may be important drivers for the evolution of complexity and diversity [5-7].

On a smaller scale, DNA amplification within genomes leads to gene duplications; a process that can provide the raw material for the evolution of novel genes [8]. Gene duplications can be caused by several different mechanisms. Homologous recombination, for example, requires the presence of repeated DNA sequences within the genome. Recombination between the two sequences can result in duplication, deletion or inversion of the intervening DNA sequence [9, 10]. Another mechanism is the insertion and amplification of mobile genetic elements within and between genomes. This usually requires limited sequence homology and is catalyzed by a protein, usually a

transposase, encoded by the element itself. This mechanism is utilized by most viruses and transposons for their propagation between and within genomes [11].

In short, evolution and life are based on, and depend on, the amplification of DNA sequences. Amplification of DNA sequences in somatic cells is favoured by natural selection under conditions where higher levels of gene expression are required to maximize organismal performance. Organisms also reproduce and replicate their DNA when environmental conditions allow it, thereby increasing their own fitness. However, some genetic elements, like viruses do not necessarily play by strict Mendelian rules, and can amplify within a given genome even when their activity does not benefit the host organism [12]. Generally, genetic elements that increase their own copy number relative to the copy number of the host are considered selfish [13]. This includes genes that encode a transposase to actively copy themselves within and between genomes (replicative selfish genetic elements) as well as genes that ensure their persistence within the genome by killing cells that lose or simply do not possess a copy of the element (addictive selfish genetic elements).

1.2 Selfish genetic elements

1.2.1 Defining selfish genetic elements

According to Hurst et al. (1996), selfish genetic elements are DNA sequences that "*are vertically transmitted genetic entities that manipulate their "host" so as to promote their own growth*" [13]. This means any DNA sequence or gene that duplicates within the genome or eliminates an organism that does not possess a copy, is considered selfish. However, categorization of genes into 'selfish' and 'non-selfish' is challenging as most genes reside in between the two extremes. Thus, it is perhaps more accurate to rank genes on a scale between entirely selfish (genes that are maintained despite not contributing to the fitness of an organism) and entirely non-selfish (genes that are only ever transmitted as a single copy per genome and thus never actively increase their proportion within the gene pool). The two extremes are unlikely to exist in reality as almost all genes are thought to have arisen as a consequence of duplication events and hence have a selfish evolutionary history [14, 15]. Equally, most selfish genes provide

some benefit to the host, if only as raw material for recombination [16, 17]. Selfish genetic elements are difficult to identify within genomes although characteristics such as repetitiveness and frequent horizontal transfer are very good indications for selfishness.

Selfish genetic elements can be crudely divided into two classes. The first encompasses those that increase their frequency within the host population through duplication (duplicative selfish genetic elements). The second class includes those elements that increase their frequency within the gene pool by killing cells that have lost a gene copy, or kill off competing cells without a copy (addictive selfish genetic elements).

1.2.2 Duplicative selfish genetic elements

Duplicative selfish genetic elements comprise autonomously and non-autonomously replicating sequences (for examples see section 1.3). Autonomous elements in bacteria include all elements that encode their own replicative ability. For example, transposons and insertion sequences (that encode an active transposase) [18] and plasmids that carry the genes necessary for their own replication [19] are autonomous elements. Non-autonomous elements are DNA sequences that do not encode their own replicative ability (*e.g.* miniature inverted repeat transposable elements (MITEs) [20]). All duplicative selfish genetic elements ensure their persistence within the gene pool through frequent amplification and horizontal transfer to avoid deactivation by selection and genetic drift [21].

1.2.3 Addictive selfish genetic elements

Addictive selfish genetic elements (for examples see section 1.4) are immobile and hence drift and selection cannot be actively avoided through transposition processes. Instead, as soon as the gene copy is lost from the DNA, the element ensures its persistence by killing the host [22, 23]. Host-killing is realized in a number of different ways but follows a common theme. Addictive selfish genetic elements usually consist of at least two components. The first component kills the host and the second prevents the first from killing the host. The protein product of the component that prevents the killing is usually less stable than that of the killer component. Hence, when the element

is deleted from the DNA, the element that prevents the killing is depleted first, which allows the more stable killing component to kill the host and eliminate the cell that lost the addictive genetic element from the population.

1.3 Duplicative selfish genetic elements

1.3.1 Autonomous and non-autonomous transposons

Autonomous transposons encode proteins (transposases) that recognize and move transposons within and between replicons (e.g. chromosomes and plasmids). A non-autonomous transposon is a transposon that has lost the ability to move/transpose independently and requires the transposase function encoded by the corresponding autonomous element for movement. Autonomous elements are frequently associated with a corresponding class of non-autonomous elements that parasitize autonomous elements to a degree that may eventually lead to their extinction [24]. There are at least two lines of evidence that support this hypothesis. Firstly, non-autonomous transposons accumulate within the genome through a number of mechanisms. For example, DNA transposons can become non-autonomous by losing the ability to produce a transposase by accumulating deleterious mutations [24, 25], whereas non-autonomous non-long terminal repeat (LTR) retrotransposons (see section 1.3.2) can evolve by high jacking the replication machinery of autonomous transposons [26]. As soon as there are more non-autonomous elements than autonomous elements within the genome, it is more likely that a non-autonomous element is transposed than an autonomous one, due to the greater supply of transposition templates (in the absence of *cis* preference). This is also referred to as the titration effect and steadily increases the non-autonomous to autonomous element ratio [24]. Secondly, as the chance of an autonomous transposition event decreases the time between transposition events increases, as well as the number of mutations that occur within the gene as a result of genetic drift. These mutations are likely to cause inactivation of the transposase and eventually the extinction of the whole transposon family within the genome [27]. However, the coupling of transcription and translation in prokaryotes was argued to enhance the transposition of autonomous elements (transposition of the transposase encoding gene *i.e. cis* preference) and prevent the accumulation of non-autonomous elements [27]. In eukaryotes it was proposed that

the extinction of autonomous elements is evaded through vertical diversification of the transposase, hence reducing the tendency to transpose non-autonomous elements [24, 28, 29].

1.3.2 Retrotransposons

Retrotransposons are probably the best known and most widely spread selfish genetic elements in eukaryotes. Their abundance is particularly apparent in the human genome, almost half of which consists of transposable elements. The most common transposon families in the human genome are autonomous long interspersed elements (LINEs) and non-autonomous short interspersed elements (SINEs). SINEs are considered non-autonomous elements since they do not encode proteins required for transposition and hence cannot move autonomously. LINEs are non-long terminal repeat (non-LTR) retrotransposons and encode an endonuclease and a reverse transcriptase. Together, these cleave the target DNA and introduce the element into the genome through a process called target-primed reverse transcription (TPRT). SINEs only encode RNA sequences and use proteins encoded by their LINE counterpart for transposition. The most prevalent LINEs are of the LINE-1 (L1) family. Repeated over 500,000 times, L1 elements make up more than 16% of the human genome, and thus are the largest transposon family by total sequence length. They are topped in copy number only by *Alu* repeats, their non-autonomous parasite and part of the SINE family, which occur over 1,000,000 times and make up about 10.6% of the human genome [16].

Although these elements are considered to be mainly selfish, it should be noted that their activity and presence has had a large impact on the evolution of the human genome. The most apparent impact is the likely destruction of open reading frames following L1 or *Alu* element insertion. Additionally, their mere presence can lead to deletions, inversion or duplication of intervening DNA through homologous recombination. All such events can lead to severe diseases such as cystic fibrosis and cancer [30, 31]. Another interesting consequence results from a host defence mechanism to L1/*Alu* retrotransposition. Regions containing L1 or *Alu* elements are methylated in order to reduce transcriptional activity. This can lead to the spread of

heterochromatin into neighbouring regions, changing the expression pattern of adjacent genes [16, 32].

In contrast to non-LTR retrotransposons, LTR retrotransposons start and end with a long terminal repeat. While LTR retrotransposons make up about 8% of the human genome, their activity is presumed to be very limited. In other organisms, LTR-retrotransposons are more active; they comprise a great proportion of *Saccharomyces* (yeast), *Drosophila* (fruit fly) and maize genomes [33]. LTR retrotransposons usually consist of a *gag*, *pol* and sometimes an *env*-like gene. The Gag protein forms a virus-like particle, in which reverse transcription takes place. The Pol protein has a variety of enzymatic functions, among which are reverse transcriptase and integrase function. It has been suggested that LTR retrotransposons evolved as the consequence of a fusion between a non-LTR retrotransposon and a DNA transposon [34]. Hence, the terminal repeats were probably acquired from an ancient DNA transposon and the reverse transcriptase from an ancient non-LTR transposon. Retroviruses possess similar (related) genes also called *gag, pol* and *env* and probably evolved multiple times from LTR retrotransposons through the acquisition of *env*-like proteins from other viruses [35]. In contrast to non-LTR retrotransposons, LTR retrotransposons do not seem to have highly abundant non-autonomous counterparts. Only a few have been identified, as for example large retrotransposon derivatives (LARDs) or terminal-repeat retrotransposons in miniature (TRIMs) [36].

In bacteria, retrotransposons and other mobile genetic elements are much less prevalent. This may be due to the importance of sexual reproduction and diploid genomes for the spread of mobile genetic elements [37]. However, some variants of retrotransposons can be found in bacterial genomes. A well-known representative is the bacterial Group II intron, a retrotransposon similar to non-LTR retrotransposons [38]. The protein encoded by bacterial Group II introns usually has reverse transcriptase, DNA endonuclease and maturase (splicing) activity. They can amplify within genomes by either site-specific retro-homing or ectopic retrotransposition [39, 40]. Retro-homing involves the recognition of and insertion into a suitable target site, whereas retrotransposition is about 100 times less efficient than retrohoming and describes the transposition into a random location within the genome.

LTR-retrotransposons, retroviruses or similar entities have not been described in prokaryotes. Only the decoupling of transcription and translation in eukaryotes made the evolution of more complex transposons necessary in order to ensure *cis* preference (transposition of the same gene that encoded the transposase) [34, 35]. DNA transposons in eukaryotes are translated outside the nucleus. Therefore, after re-entering the nucleus DNA transposons are likely to transpose any sequence that is flanked by the required terminal repeats. Non-LTR retrotransposons on the other hand have to either bind to the transcript and channel it back through the nucleus' membrane or reverse transcribe mRNA in the nucleus and lose *cis* specificity.

1.3.3 DNA transposons

DNA transposons found in the human genome can be divided into four superfamilies: hAT, piggyBac, MuDR and Tc1/*mariner*. Together, DNA transposons constitute approximately 3% of the human genome, but have not been active over the last 40 million years (My) [41]. It is noteworthy that in other mammalian lineages DNA transposons appear to have become extinct or inactive at around the same time that this occurred in the human genome [24]. The only reported instances of active DNA transposons in mammals are hAT transposons in the bat genus *Myotis* [42]. In other eukaryotes DNA transposons such as the P-element in *Drosophila* or the *Ac/Ds* elements in maize, which were the first identified transposable elements [43], remain highly active. Interestingly, some DNA transposons have been shown to be 'domesticated' by their hosts; that is, they are now fulfilling new beneficial roles within the cell (*e.g.* [44-47]). One of the most prominent examples is the origin of the V(D)J component of the vertebrate immune system [45].

Similar to retrotransposons, an excess of non-autonomous elements accompanies almost every known class of DNA transposons. Non-autonomous DNA transposons are commonly called miniature inverted-repeat transposable elements (MITEs) and were first described in plants [20].

DNA transposons are the most common type of transposable elements in bacteria and are usually called insertion sequences (IS) [18]. They consist of one or two open reading frames flanked by two (inverted) repeat sequences and range in size from

approximately 0.4 kb to 2.5 kb. While the flanking repeats are usually inverted, they can be direct (for example, the flanking repeats of IS*200* elements) [48]. IS elements are also found in more complex genetic structures (called 'composite transposons') where two IS elements flank a cassette of genes. These genes usually confer a benefit to the host bacterium in a particular environment, thus increasing the probability of the transposon's persistence in a new host. For example, antibiotic resistance genes are commonly spread by composite transposons [49].

MITEs are also associated with DNA transposons in bacteria, although the relationship is harder to determine due to the low abundance of correlated transposases. Many short repetitive sequences like enterobacterial repetitive intergenic consensus sequences (ERICs) [50] or *Neisseria* miniature insertion sequences (NEMISs) [51] show MITE-like structures [52]. However, only for repeat units of *Pneumococcus* (RUP) a potential transposase encoded in *trans* has been implicated in their mobilization [53].

1.3.4 Short repetitive sequences in bacteria

Short repetitive sequences in bacteria fall into two classes: sequences longer than 10 bp and sequences shorter than 10 bp. Over-represented sequences shorter than 10 bp are mainly due to replication slippage or selection on genome architecture and therefore it is unlikely that they are selfish genetic elements [54-57]. Prominent examples for repetitive sequences shorter than 10 bp are architecture imparting sequences (AIMS) [57]. AIMS are about eight nucleotides long and have been shown to be conserved by selection. They are preferentially found on leading strands and their abundance decreases with increasing distance to the replication terminus. Since it has been shown that their location is independent of the position of genes, it was proposed that their function is involved in DNA replication and segregation.

Although ubiquitously found within bacterial genomes, not much is known about the evolution of repetitive sequences longer than 10 bp. The first family of short sequence repeats reported in bacteria are repetitive extragenic palindromic (REP) sequences [58, 59] (recently reviewed in [60]). REP sequences are now widely used for genotyping purposes and reported to be present in a wide range of bacteria [61, 62]. However, due to the limited understanding of their evolution it is likely that REP sequences represent a

collection of repeat families evolved from multiple independent origins, all showing similar structural properties (palindrome, extragenicity and repetitiveness). Although little is known about the evolution and origin of REP sequences, it has been proposed that REPs are a family of selfish genetic elements, however this hypothesis has never been tested [63, 64].

1.3.5 Plasmids

Plasmids are ubiquitously found in bacteria [65] and to a lesser degree in eukaryotes [66] (with the yeast 2µm plasmid as the most prominent example). Plasmids are between 0.8 and 2600 kb long (see ncbi website, http://www.ncbi.nlm.nih.gov/genomes/genlist.cgi?taxid=2&type=2&name=Bacteria%20Plasmids) and can be found in linear or circular form. They frequently carry genes that promote their horizontal (*e.g.* genes that enhance conjugation [65]) and vertical transfer (*e.g.* genes that kill bacteria without plasmid copy, see section 1.4). Especially in eukaryotes, plasmids have been shown to be parasitic selfish genetic elements. This is based on the observation that the fitness of plasmid free cells is higher than the fitness of cells that contain a plasmid as well as the fact that plasmids are propagated to plasmid free cells during sexual recombination (increase their number in the gene pool disproportionally to the host genome) [67]. In prokaryotes plasmids frequently carry genes that allow the adaptation of the bacterial host to new environments (*e.g.* antibiotic resistance genes [65, 68]) and hence can provide a competitive advantage. However the fact remains that their copy number is disproportionately increased compared to the host genome by actively promoting spread within the population through horizontal transfer [68, 69].

1.4 Addictive selfish genetic elements

1.4.1 Toxin-antitoxin (TA) systems

TA systems in bacterial genomes represent a class of selfish genetic elements that enhances their copy number in the bacterial population by killing the competition *i.e.* bacteria that lose the element. They are found in almost all prokaryotes and typically

consist of two (in some cases three) genes: a toxin, usually encoding a protein, and a neighbouring antitoxin, which is not necessarily translated. TA systems are found on plasmids and chromosomes. Plasmid encoded TA systems are maintained through post-segregational killing (PSK). The PSK mechanism ensures that any bacterial cell that loses the plasmid (or TA system) is killed due to the faster degradation of the antitoxin relative to the toxin. TA systems located on bacterial chromosomes are also thought to confer a benefit to the host to ensure the TA system's persistence vertically. Based on the function of the antitoxin, TA systems are divided into three separate families, each of which is described in more detail below [70].

1.4.1.1 Type I TA systems – antisense RNA antitoxins

Type I TA systems encode RNA antitoxins that bind to the mRNA of the corresponding toxin and thereby usually prevent translation, which leads to the subsequent degradation of toxin mRNA. Prominent examples of this group of TA systems are the Hok/Sok system of plasmid R1 and the SOS induced TisB/IstR-1 system described in *Eschericha coli* [71].

1.4.1.1.1 The hok/sok *TA system*

The Hok/Sok TA system is found on the plasmid R1 and ensures its maintenance through PSK. The *hok* (host killing) gene encodes a toxic transmembrane protein, while the *sok* (supression of killing) gene encodes an antisense RNA that binds to the mRNA of the *mok* (modulation of killing) gene and prevents its translation. Since without *mok* translation *hok* cannot be translated, *sok* indirectly prevents the translation of *hok* mRNA. The binding of *sok* RNA to *mok* mRNA leads to the formation of an RNA duplex which inhibits translation and is cleaved by RNAse III. *Hok* mRNA is very stable (half life ~20min) but under the control of a weak promoter. *Sok* mRNA is unstable (half life ~30secs) but under the control of a strong promoter. Thus, the system does not have an effect on the host while the plasmid is present within the cell. However, if the plasmid is lost, the *sok* RNA quickly degrades and leaves the more stable *hok* mRNA available for translation. This leads to the production of the toxic transmembrane protein and death of the cell. Hence, the mechanism ensures the maintenance of the host plasmid within a bacterial population [71].

1.4.1.1.2 The TisB/IstR-1 TA system

The TisB/IstR-1 system is a chromosomally-encoded TA system found in *E. coli*. It is regulated at the transcriptional level by LexA, a protein that represses the SOS response in bacteria [71]. The toxin, TisB (toxicity induced by SOS), is predicted to be a transmembrane protein that operates by halting cell growth through disruption of the cell membrane, ultimately leading to decreased rates of transcription, translation and replication [72]. The *istR-1* gene encodes a short antisense RNA that contains a complementary region of 21 nucleotides to *tisB* mRNA in the 5′ region of the gene. Binding of IstR-1 to *tisB* mRNA inhibits translation and leads to cleavage of the duplex by RNase III. Following initiation of the SOS response, the transcription of *tisB* is induced, leading to an excess of mRNA and the production of toxic TisB [71, 73]. The activity of the TisB/IstR-1 TA system has been shown to be important to tolerate DNA damaging agents such as ciprofloxacin [74]. Hence, this is a good example of how a selfish element has been adopted by the host to perform a beneficial function thereby presumably losing some of its selfish characteristics.

1.4.1.2 Type II TA systems

Type II TA systems consist of a toxin, an antitoxin and sometimes an additional gene, which is involved in the regulation of the TA operon. In contrast to type I TA systems the antitoxin gene encodes a protein. The toxin and the antitoxin form a stable, non-toxic complex that inhibits the transcription of the TA operon. In a number of cases, it has been shown that in the presence of excess toxin, a different complex is formed that induces the transcription of the operon [75, 76]. This typically happens under stressful conditions when the antitoxin is degraded by specific proteases, thereby releasing the toxin from the non-toxic complex [77]. As with type I TA systems, type II TA systems were first observed as plasmid maintenance genes encoding PSK mechanisms [78].

Type II toxins have been shown to arrest cell growth in at least two different ways. The first involves binding to DNA gyrase, which leads to an excess of supercoiled DNA, preventing RNA/DNA polymerases from binding to the DNA [79]. Alternatively, the toxin may bind to the A site of the ribosome during translation, and cleave bound mRNA at specific sites [80, 81]. For chromosome-borne type II TA systems it has been

shown that induction of toxicity does not necessarily lead to cell death, but growth arrest. Similar to type I TA systems this can result in the persistence of the bacterium under unfavourable conditions such as antibiotic stress [78, 82].

1.4.1.3 Type III TA systems

Recently, a third TA type was described, named ToxIN, this system consists of a toxin (encoded by *toxN*) and an RNA antitoxin (encoded by *toxI* a 36 bp repeat region upstream of *toxN*) [83]. Unlike type I TA systems, the *toxI* antitoxin does not encode an antisense RNA that inhibits the translation of *toxN* mRNA. Instead, experiments suggest that the antitoxin forms a non-toxic complex with the toxin, in a similar fashion to type II TA systems (however, formation of the complex has not yet been directly observed). Interestingly, the toxic protein shares similarities to a well-described group of toxins called Abi (abortive infection) toxins. ToxN and Abi toxins have been shown to confer resistance to certain phages by killing the host before the phage can replicate [83, 84]. According to Fineran et al. [83], it is possible that other Abi toxins are actually part of a type III TA system where the antitoxin has gone unnoticed.

1.4.2 Bacteriocins

Bacteriocins are mainly found on plasmids and are similar to type II TA systems. They typically comprise two or three genes, which encode a toxin, an immunity protein and, in some cases, a lysis or release protein. As in type II TA systems, the toxin and the immunity protein form a neutral non-toxic complex. However, unlike type II TA systems the toxin is secreted by host cell lysis (if a lysis gene is part of the bacteriocin operon) or exported through the membrane [85, 86]. The toxin then attaches to other related bacterial cells, which do not contain the bacteriocin and causes cell death through DNA breakdown or disrupting essential cellular processes (e.g. protein synthesis). Hence, bacteriocins ensure plasmid persistence not only within the host line, but within the population. The best-studied bacteriocins are colicins, named after their host species *E. coli*. Colicins are divided into two groups based on whether the outer membrane of the target cell is passed through the Tol or Ton transport system [86].

1.4.3 Restriction-modification systems (RMS)

Restriction-modification systems (RMS) are selfish genetic elements that protect the host from the invasion of unmodified (foreign) DNA, or to put it in a selfish gene context make sure that any bacterium that loses the system is killed (similar to TA systems) [87]. They are named due to the property of RMS containing bacterial strains to restrict the growth of certain viruses, through sequence specific DNA cleavage; modification of the same DNA prevents cleavage [88].

R-M systems also share similarities with toxin antitoxin systems. They consist of one or multiple genes encoding proteins that are harmful to the cell if it were not for the product of the second set of genes that neutralize the effect of the first. The two components are: an endonuclease (acting as toxin) and a DNA methyltransferase (acting as antitoxin). Unlike type II TA systems, toxin and antitoxin do not form a neutral non-toxic complex. Rather, the DNA methyltransferase prevents endonuclease mediated DNA cleavage by attaching methyl groups to nucleotides found in a specific sequence context also recognized by the endonuclease. On plasmids RMSs can cause post-segregational killing of plasmid free cells leading to increased plasmid persistence. Chromosome borne RMSs (also applies for RMSs on plasmids) have been shown to protect the bacterium against invading DNA like phages or plasmids [88].

1.4.3.1 Type I R-M systems

Type I R-M systems recognize two short, asymmetric DNA sequences that are separated by a short non-specific spacer sequence. DNA cleavage occurs at variable distances to the recognition site. Hence, digestions by type I R-M systems cannot be visualized on polyacrylamide gels. Type I systems typically encode three proteins: one responsible for DNA cleavage, one for DNA methylation and one that determines DNA specificity. The three proteins form a complex that acts as endonuclease as well as methyltransferase. Once the complex recognizes the specific DNA motif, it cleaves the DNA if unmethylated, fully methylates the DNA if hemimethylated, or dissociates from the DNA if fully methylated. This system ensures that after DNA replication, the DNA is present in a hemimethylated form, prevents the R-M complex from cleaving its own DNA and restores the fully methylated state before the next replication cycle. Therefore

both forward and reverse DNA strands must be modified in order to achieve three different methylation states. This is accomplished by methylating an adenine on the top strand of the first part of the recognition sequence and an adenine on the bottom strand of the second part of the recognition sequence [88].

1.4.3.2 Type II R-M systems

Type II R-M systems are the simplest and most numerous among R-M systems. Endonuclease and methyltransferase act as independent proteins and recognize symmetric (palindromic) recognition sequences. Hence, the same protein domain can recognize both the forward and reverse DNA strand. Cleavage occurs either in the centre of the recognition sequence, producing blunt ends, or is shifted to the side producing staggered ends. Precise cleavage of DNA is essential for almost all cloning reactions, hence, type II R-M systems are very important tools in molecular biology [88].

1.4.3.3 Type III R-M systems

Type III R-M systems consist of an R and an M subunit. The M subunit alone acts as methyltransferase and contains the specificity domain. Together with the R subunit the complex can act as both methyltransferase and endonuclease. Type III R-M systems recognize asymmetric uninterrupted DNA motifs. Unlike Type I systems cleavage and methylation occurs only on one DNA strand. Therefore complete cleavage is only possible if the recognition sequence occurs on both the forward and reverse strands in close proximity. Even in situations where the recognition motif occurs on both strands, digestion of DNA is not usually complete due to competition between the methyltransferase and endonuclease for activity on unmethylated target sites. Interestingly, phage T7 contains the target sequence (CAGCAG) of EcoP15. However, the sequence is present on only one DNA strand, which may be indicative of selection acting to evade cleavage by EcoP15 [88].

1.5 Other selfish genetic elements

1.5.1 Clustered regularly interspaced short palindromic repeats (CRISPRs)

Even though their genetic structure was described over 20 years ago [89], clustered regular interspaced short palindromic repeats (CRISPRs) became objects of interest only recently [90]. They consist of a set of CRISPR associated genes (CAS genes) followed by an AT-rich sequence of low complexity and an array of tandemly repeated short palindromic sequences interrupted by short variable spacer regions. In 2005 CRISPRs were proposed to provide acquired immunity against phages and mobile genetic elements, based on homology between the variable spacer regions and phage sequences [91-93]. Further analyses of CAS genes showed similarities between the CRISPR system and the eukaryotic RNA interference machinery, suggesting the degradation of foreign DNA guided by RNA [94]. A plethora of detailed studies support this hypothesis and demonstrate that CRISPRs play an important role in defence against invading genetic elements for about 40% of all bacteria and most archaea (reviewed in [95]).

A system conferring acquired immunity to invading foreign DNA is not immediately obvious as a selfish trait. However, CRISPRs and their associated CAS genes show considerable variation even among bacterial and archaeal strains. Furthermore, they are frequently found on mobile genetic elements such as plasmids and viruses, suggesting a high rate of horizontal transfer [96]. A high rate of horizontal transfer in conjunction with the presence of multiple CRISPR systems in certain genomes indicates selfish behaviour of CRISPR systems, *i.e.* an increase in copy number relative to their host's, despite conferring a potential benefit (similar to antibiotic resistance cassettes or other genomic islands).

1.6 Characteristics of the model organism Pseudomonas fluorescens *SBW25*

Pseudomonas fluorescens SBW25 is a plant-associated bacterium originally isolated from the surface of a sugar beet leaf at Wytham farm in Oxford, UK [97]. It has been extensively studied and used as a model organism in experimental evolution (*e.g.* [98, 99]). *P. fluorescens* SBW25 is a particularly useful model organism for the study of short repetitive sequences and other dispersed selfish genetic elements. Not only is the genome sequence of SBW25 known, but also the genome sequences of the relatively

closely related strains Pf0-1 and Pf-5, which allows comparative studies to be conducted [100].

The genome of *P. fluorescens* SBW25 is 6,722,539 bp long, of which about 88.3 % encode for genes. According to Silby et al., approximately 11.91% of the genome consists of repetitive sequences, a great proportion of which are repeated gene families [100]. Aside from intragenic (within gene) repeats 1,199 extragenic (outside gene) repeats were identified. Interestingly, the most abundant repeat families (called R0, R1 and R2) are highly strain specific, which may indicate horizontal transfer and rapid evolution, hallmarks of selfish genetic elements.

1.7 Summary and objectives of this study

The four nucleotides that make up DNA are the building blocks of the hereditary material of almost all known life forms (exceptions include RNA viruses). Shifts in GC content lead to an over-representation of either G/C or A/T nucleotides and therefore represent the shortest repetitive sequences. With increasing sequence length the reasons for their over-representation change. For example, one reason for the over-representation of sequences of less than 10 nucleotides in length is a bias in DNA replication (*e. g.* [57]), while for greater sequence lengths over-representation is presumably due to the activity of selfish genetic elements [60].

The largest gap between the description and characterization of putative selfish genetic elements and the study of their evolution and origin seems to exist for bacterial short repetitive sequences. This applies in particular to REP sequences [58-60] but also to other repetitive sequences such as the different classes described in the *P. fluorescens* SBW25 genome [100]. Thus, SBW25 is an ideal candidate for the study of repetitive and presumably selfish genetic elements. Furthermore, the presence of a range of other fully sequenced *Pseudomonas* strains (and other related bacteria) allows for testing the general applicability of hypotheses formed on the basis of observations in the SBW25 genome.

Objective 1. Short repetitive sequences in the *P. fluorescens* SBW25 genome (Chapter 3)

As mentioned above, the evolution and origin of most short repetitive sequences is obscure. Those contained within the genome of *P. fluorescens* SBW25 are no exception. Hence, the first objective of this study is to characterize short repetitive sequences, with particular focus on their patterns of diversity and distribution within the SBW25 genome.

Objective 2. Cause for REPIN dissemination and replication (Chapter 4)

Having shown that short repetitive sequences are part of a greater replicative unit called REP doublet forming hairpins (REPINs), the cause for REPIN amplification and distribution in bacterial genomes is investigated.

Objective 3. Characterization of RAYTs: a new class of REP and REPIN associated genes (Chapter 5)

REP associated tyrosine transposases (RAYTs) [101] have been proposed to be the cause for REPIN dispersal. To elucidate the functional relationship between RAYTs and REPs/REPINs the third objective is the detailed characterization of this class of genes within bacteria.

Objective 4. Analysis of other repetitive elements in the SBW25 genome (Chapter 6)

There are two major repetitive sequence classes in the SBW25 genome that were identified but not analysed in detail by Silby et al. [100]. The fourth objective is to characterize these repetitive sequences and study their evolution in the SBW25 genome.

Chapter 2:
Methods

2.1 General Methods

2.1.1 Bioinformatics

BLAST searches were performed using NCBI BLAST [102]. The genome was browsed using Artemis [103]. DNA secondary structures were predicted using the mfold web server [104].

2.1.2 Specific genomes used for analyses

Pseudomonas fluorescens SBW25 (NC_012660.1) [100]
Pseudomonas fluorescens Pf0-1 (NC_007492.2) [100]
Pseudomonas fluorescens Pf-5 (NC_004129.6) [105]
Pseudomonas syringae phaseolicola 1448A (NC_005773.3) [106]
Pseudomonas syringae syringae B728a (NC_007005.1) [107]
Pseudomonas syringae tomato DC3000 (NC_004578.1) [108]
Pseudomonas entomophila L48 (NC_008027.1) [109]
Pseudomonas putida W619 (NC_010501.1)
Pseudomonas putida KT2440 (NC_002947.3) [110]
Pseudomonas putida F1 (NC_009512.1)
Pseudomonas putida GB-1 (NC_010322.1)
Pseudomonas aeruginosa PAO1 (NC_002516.2) [111]
Pseudomonas aeruginosa PA7 (NC_009656.1) [112]
Pseudomonas aeruginosa LESB58 (NC_011770.1) [113]
Pseudomonas mendocina ymp (NC_009439.1)
Pseudomonas stutzeri A1501 (NC_009434.1) [114]
Salmonella enterica serovar Paratyphi *A* AKU_12601 (NC_011147.1) [115]

Escherichia coli K-12 DH10B (NC_010473.1) [116]

Thioalkalivibrio sp HL-EbGR7 (NC_011901.1)

Nostoc punctiforme PCC 73102 (NC_010628.1)

Xanthomonas campestris B100 (NC_010688) [117]

Planctomyces limnophilus DSM 3776 (NC_014148)

Geobacter sp. FRC-32 (NC_011979)

Prosthecochloris aestuarii DSM 271 (NC_011059)

2.2 Methods Chapter 3

2.2.1 Bioinformatics and phylogenies

Inverted repeats were identified using Repeat Finder [118]. The multiple alignments in Figure 3.8 were displayed with Geneious [119] (due to the perfectly conserved distances between the 16-mers, the sequences were aligned after extraction from the genome, no alignment method was needed).

2.2.2 Generation of randomized genomes

100 genomes with the same dinucleotide content of the leading/lagging strand and length as the genome of *P. fluorescens* SBW25 were generated by randomly choosing nucleotides according to their occurrence probability based on the preceding nucleotide. To account for dinucleotide skew in the leading or lagging strand of the SBW25 genome, the dinucleotide content of the top strand was determined for the first half of the genome and of the bottom strand for the second half of the genome [100] (source code A4.1).

2.2.3 Frequency determination of most abundant oligonucleotides

Sequence frequencies for all oligonucleotides of length 10 to 20 were determined using a sliding window with a step size of one for leading and lagging strand separately. The most abundant oligonucleotide for each sequence length was determined. This analysis

was conducted for randomly generated genomes as well as for *P. fluorescens* SBW25 and Pf0-1 (source code A4.2).

2.2.4 Grouping of highly abundant oligonucleotides in SBW25

All oligonucleotides of the chosen sequence length that occur more often in SBW25 than in Pf0-1 were ordered into groups using the following algorithm: 1, Select the most abundant 16-mer from the list of 16-mers that occur more frequently than the most abundant 16-mer in Pf0-1; 2, interrogate the SBW25 genome; 3, extract all occurrences including 20 bp of flanking DNA; 4, concatenate, separating each sequence by a vertical bar (a symbol that is not part of the genomic alphabet); 5, search all remaining 16-mers from the list against the generated string; 6, remove from the list of 16-mers all those sequences found within the generated string and place into the same group as the query; 7, repeat until the list of 16-mers is empty (Figure 2.1, source code A4.3).

2.2.5 Extending REP sequence groups and identifying the frequency of false positives

The genome was searched for related elements by introducing base pair substitutions into the most abundant sequence of each group to a maximum of four. The newly generated sequences, as well as the most abundant sequence of each group, were then used to interrogate the genome and the number of occurrences was counted. In order to determine the false positive rate, a simulation program was written to determine the number of sequences found in randomly generated extragenic space (with the same dinucleotide content, source code A4.4).

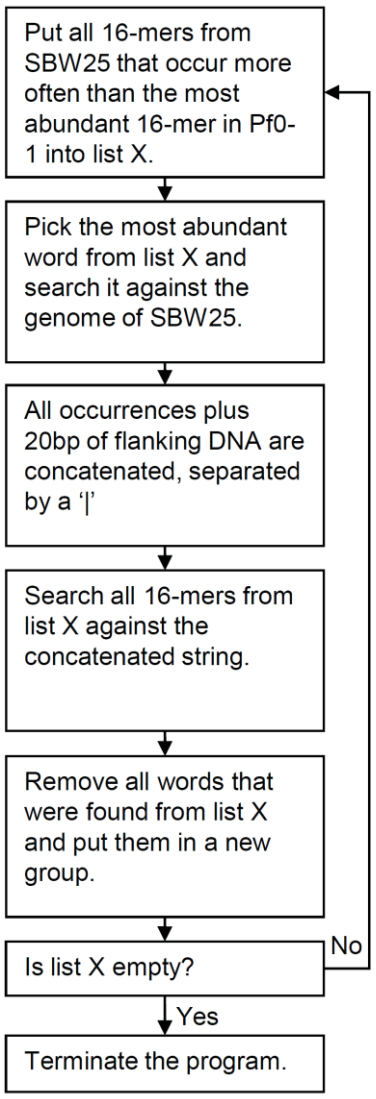


Figure 2.1. Flowchart for grouping over-represented 16-mers. The algorithm sorts all 16-mers that occur more frequently in SBW25 than the most abundant 16-mer in Pf0-1 into groups.

2.2.6 Distribution simulation

In order to produce a null model against which the observed next-neighbor distances could be compared, 1,053 segments of length 16 were randomly assigned to the extragenic space of SBW25. The simulation was repeated 10,000 times and for each simulation the distances to neighboring segments were determined. Additionally, the formation of clusters by GI, GII and GIII sequences with up to two mismatches (1,422 sequences) was measured. A cluster of REP sequences was defined as a group of REP sequences where each REP sequence has two neighboring REP sequences within the group that are separated by less than 400 bp (the next-neighbor distances showed no significant deviations from randomly expected distances above 400 bp) and a maximum of two REP elements that have only one neighbor within the group which is separated by less than 400 bp.

The same method was applied when distributing doublets randomly over the genome. Instead of 1,422 16 bp long segments, 560 x 71 bp and 560 x 110 bp long segments respectively, were randomly assigned. The number of REP doublets was determined by only counting doublets and clusters of doublets. For clusters that contain an odd number of REP sequences, only the even proportion was counted, thus excluding singlets (Figure 2.2, source code A4.5).

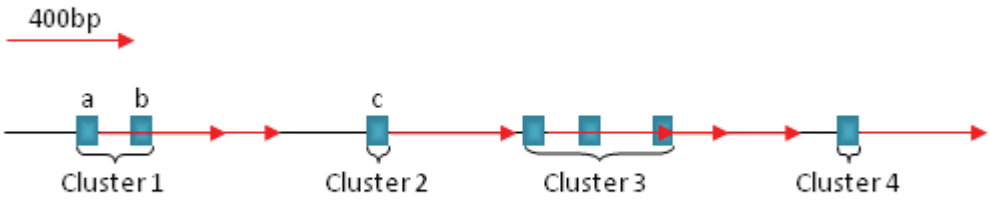


Figure 2.2. Process of REP sequence cluster determination. REP sequences are blue boxes. Red arrows indicate a sequence length of 400 bp. The algorithm starts with the position of the first REP sequence (a) and adds it to cluster 1. It then checks the distance to the next REP sequence. The distance to REP sequence (b) is less than 400 bp, hence, the size of cluster 1 increases by one. The distance from (b) to the next REP sequence (c) is greater than 400 bp, therefore, the final size of cluster 1 is two and a new cluster of size one is created called cluster 2. The distance from REP sequence (c) to the next REP sequence is greater than 400 bp; hence, cluster 2 is closed.

2.2.7 Singlet decay

To compare the rate of decay between REP singlets and REP sequences that are part of clusters, REP sequences were divided into their respective groups and then subdivided depending on whether they are found in clusters, or as singlets. In order to include related sequences, the 16-mers were allowed to vary at up to two positions. Since GI 16-mers differ from GII and GIII 16-mers by only two nucleotides, GII and GIII sequences also had to have two group-specific bases (GII: 2T, 6C; GIII: 6A, 13T).

The significance of the singlet decay data was tested using a permutation test. Nine different REP sequence pools were created. Three sequence pools for each sequence group, one of which contained REP singlets, one REP doublets and one greater REP cluster sequences. Two sequences were randomly drawn without replacement from a specific sequence pool and their pairwise identity (the number of sites that are identical between the two sequences divided by the total number of sites) was calculated. This procedure was repeated until the sequence pool was empty. The whole process was repeated 100,000 times for each sequence pool, resulting in the calculation of 100,000 average pairwise identities (mean). For GI sequences the maximum mean calculated for REP singlets never exceeded the minimum mean for REP sequences arranged as doublets. For GII and GIII sequences the maximum mean of REP singlets did exceed the minimum mean of REP sequences from doublets when more than 1,000 means were produced, hence the lower significance of 1e-8. Additionally, for GI and GIII sequences the maximum mean for singlets also never exceeds the minimum mean for clusters (*P*-value 1e-10). The average of the calculated means and the standard deviation are displayed in Figure 3.5 (source code A4.6).

2.2.8 Population sequencing

Pure genomic DNA was isolated from a single SBW25 colony using a combination of chloroform, CTAB and column (Qiagen DNeasy Blood & Tissue Kit) purification techniques. The genomic DNA was sheared to ~400 bp and 76 bp paired-end were sequenced on two channels of an Illumina GA-II flowcell using standard protocols. Raw data were filtered to generate a set of sequences no less than 36 bp in length. After

mapping short reads to the SBW25 genome using the Mosaik software suite (http://bioinformatics.bc.edu/marthlab/Mosaik), reads that could not be mapped were screened for REPIN excisions. The screening was accomplished in two steps: 1, for each REPIN present in the SBW25 genome 12 bp of the 5′ and 3′ flanking sequences were extracted; 2, since all reads are shorter than 76 bp, none of the extracted flanking sequences should occur within one read, hence reads containing both 5′ and 3′ REPIN flanking sequences contain an excision. Details of the sequences from which REPINs were excised are given in Figure A1.2.

2.2.9 Testing for excision of REP singlets

In order to identify excisions of short palindromic sequences it was necessary to define a seed sequence. The GI and GII sequences described in section 3.2.1.1 do not overlap the palindromic region and hence are not suitable for this purpose (Table 3.1). Therefore an 18-mer containing the palindrome of the GI REP as the seed sequence (GGGGGCTTGCCCCCTCCC) was used. From this seed sequence a set of 18-mers with up to five mismatches was generated. These sequences matched a total of 1376 positions in the SBW25. This set of 1376 sequences encompassed all three GI, GII and GIII REP sequence groups and their relatives. In addition, to allow for the possibility of inexact excisions of palindromes, the excision was allowed to include three additional base pairs on each side of the seed sequence. Armed with this set of sequences the ~56 million Illumina-generated sequence reads were interogated for evidence of excision events (source code A4.7).

2.3 Methods Chapter 4

2.3.1 Bioinformatics and phylogenies

The alignment in Figure 4.2 was created using ClustalW2 [120]. The phylogenetic tree in Figure 4.3 was based on a translation alignment (ClustalW2 [120]) as implemented within Geneious [119]. The tree was constructed using a neighbour-joining [121] bootstrap analysis (1000 replicates) also embedded in Geneious.

2.3.2 REP sequence selection in other genomes

Since REP sequences have been shown to be associated with RAYT genes [101], the non-coding DNA flanking RAYT genes was searched for 16-mers that were repetitive, extragenic and palindromic. The most frequent 16-mers found within the flanking DNA were also part of or contained a palindrome and were found predominantly in extragenic space, thereby fulfilling all REP sequence prerequisites (Table A2.2). These 16-mers were then used for a subsequent cluster analysis (flanking clade I RAYTs) or a sample DNA secondary structure prediction (flanking clade II RAYTs, source code A4.8).

2.4 *Methods Chapter 5*

2.4.1 Genomes

Bacterial genomes were downloaded from the NCBI ftp site on the 09th of March 2011 (ftp://ftp.ncbi.nih.gov/genomes/Bacteria/). On that day 1398 bacterial chromosomes and 1015 plasmids were fully sequenced.

2.4.2 BLAST search

For each gene family two proteins were used as queries for a BLAST search. The query sequences for the RAYT gene family were YafM from *E. coli* K-12 and *P. fluorescens* SBW25, for the family of peptide deformylases, *def* from SBW25 and *E. coli* K-12, for the IS*200* family IS*609* from *E. coli* O157:H7 and ISHp*608* from *Helicobacter pylori* and for the IS*110* family ISPfl*1* from *P. fluorescens* Pf0-1 and ISEc*32* from *E. coli* S88 plasmid pECOS88. The protein pairs were then searched against each of the 1398 chromosomes and 1015 plasmids individually using TBLASTN.

The search results were analyzed in the following steps: 1, search results were sorted into different groups, where each group contains all hits below a certain e-Value threshold; 2, hits were checked for overlaps with genes from the corresponding genbank annotation; for multiple overlaps the longest overlapping gene was extracted; hits without overlaps were ignored; in the case of multiple overlaps with the same gene only the first hit is recorded; 3, genes with overlaps were extracted and saved in the

corresponding group as well as the translated amino acid sequence and the flanking 5′ and 3′ non-coding DNA (source code A4.9).

2.4.3 Identifying duplications

For all homologues that occur in the same genome the nucleotide sequences were aligned using the Needleman-Wunsch algorithm [122] and the pairwise identities (the number of sites that are identical between the two sequences divided by the total number of sites) were calculated. All pairs with a pairwise identity greater than 95% were reported as duplicates (source code A4.10 and A4.12).

2.4.4 Taxonomy information

Taxonomy information was downloaded from the ncbi ftp site (ftp://ftp.ncbi.nih.gov/pub/taxonomy/taxdump.tar.gz). Taxonomic classes were determined by climbing up the taxonomic tree until the class level was reached. If no class was specified the next higher classification is used (*e.g.* phylum, source code A4.11).

2.4.5 Frequency determination of flanking 16-mers

The frequency of all 16-mers from all replicons (chromosomes and plasmids) was determined, according to the analysis described in section 2.2.3. This way the frequency of the most abundant 16-mer from each flanking non-coding DNA sequence could easily be determined. The mean and standard error were displayed for each sequence family in Figure 5.7 (source code A4.12).

2.4.6 Calculating the pairwise identity for amino acid sequences and its significance

Pairwise alignments between protein sequences were computed by applying the Needleman-Wunsch algorithm [122]. The pairwise identity is the number of identical sites within the alignment divided by the total number of sites.

Whether the pairwise identity is significantly higher than expected by chance was tested by shuffling (draw amino acids without replacement) the two protein sequences 10,000 times. For each of the 10,000 trials the shuffled sequence pair is aligned and its pairwise identity is determined. The *P*-value is the proportion of the 10,000 pairwise identities that were greater than or equal to the pairwise identity of the two original sequences (source code A4.13).

2.4.7 Calculating phylogenetic clusters

Pairwise identities between large numbers of proteins can be visualized as phylogenetic clusters. In those clusters, proteins are represented by nodes and pairwise identities between proteins are represented as connections between the nodes (edges), if the pairwise identity between a protein pair exceeds a certain threshold. The phylogenetic clusters were displayed by cytoscape (www.cytoscape.org) [123]. Cytoscape provides several options to display networks. The 'organic layout' was selected, since it shows the formation of clusters within the data by reducing the distance between highly connected groups of nodes. Those groups are reffered to as phylogenetic clusters and are representative of groups of closely related proteins (source code A4.14).

2.5 Methods Chapter 6

2.5.1 Bioinformatics

Phylogenetic trees were constructed based on ClustalW2 [120] alignments and applying the neighbour-joining [121] method in Geneious [119]. The predition of transmembrane helices was performed with TMpredict (http://www.ch.embnet.org/software/TMPRED_form.html [124]).

2.5.2 Pairwise identities for R200 sequences

The average pairwise identities for different R200 sequence groups were calculated similar to the singlet decay method under section 2.2.7. The only difference between the above method and the comparison of R200 sequence groups was that before the

pairwise identity was calculated the sequences were aligned by applying the Needleman-Wunsch algorithm [122] (source code A4.15).

Chapter 3:
Within-genome evolution of REPINs: a new class of bacterial mobile DNA

Based on:

Bertels F, Rainey PB (2011) Within-Genome Evolution of REPINs: a New Family of Miniature Mobile DNA in Bacteria. PLoS Genet 7: e1002132. doi: 10.1371/journal.pgen.1002132. (attached to the end of the thesis)

Contributions: Conceived and designed the experiments: FB PBR. Performed the experiments: FB. Analyzed the data: FB. Wrote the paper: FB PBR.

3.1 Introduction

Short repetitive sequences are a feature of most genomes and have consequences for genome function and evolution [16, 125]. Often attributable to the proliferation of selfish elements [21, 37], short repeats also arise from amplification processes, such as replication slippage [54] and *via* selection on genome architecture [55-57].

Repetitive DNA in bacterial genomes is less prominent than in eukaryotes, nonetheless, an over abundance of short oligomers is a hallmark of almost every microbial genome [60]. Known generically as interspersed repetitive sequences, these elements have a history of exploitation as signatures of genetic diversity (e.g., [61, 62, 126]), but their evolution, maintenance and mechanism of within- and between-genome dissemination are poorly understood [60, 127-130].

3.1.1 Interspersed repetitive sequences

Interspersed repetitive sequences fall into several broad groups each sharing short length (individual units range from ~20 to ~130 bp), extragenic placement, and

palindromic structure [60, 131]. REPs (repetitive extragenic palindromic sequences) – also known as PUs (palindromic units) – range from ~20 to ~60 bp in length, possess an imperfect palindromic core, are widespread among bacteria, and occur hundreds of times per genome [58, 59, 100, 101, 127, 132, 133]. While often existing as singlets, REPs also form a range of complex higher order structures termed BIMEs (bacterial interspersed mosaic elements) [128]. CRISPRs (clustered regularly interspaced short palindromic repeats) are a further, higher order composite of REP-like sequences that are formed from direct repeats of short (~30 bp) palindromic sequences interspersed by similar size unique non-repeated DNA ([89]; reviewed in [95]). Recent work shows that the unique sequences are often phage derived and that CRISPRs, along with associated proteins, confer resistance to phage by targeting viral DNA [95, 134].

3.1.2 Non-autonomous DNA transposons (MITEs)

Non-autonomous DNA transposons form a more distinct family of repetitive sequences defined by their size (~100 to ~400 bp) and presence of terminal inverted repeats. Also known generically as MITEs (miniature inverted repeat transposable elements), non-autonomous transposons depend on transposase activity encoded by co-existing autonomous transposons for dissemination [21]. Identified initially in plants [20], where evidence of active transposition has been obtained [135], recent bioinformatic analyses suggest that they also occur in bacteria [52, 53]. For example, ERICs (enterobacterial repetitive intergenic consensus) – found in a range of enteric bacteria including *Escherichia coli*, *Salmonella* and *Yersinia* [50] – and NEMISs (*Neisseria* miniature insertion sequences) in pathogenic neisseriae [51] are thought to be non-autonomous transposons (MITEs).

3.1.3 Evolution and origin of repetitive sequences in bacteria

Scenarios for the origins and functional significance of non-autonomous elements, and to a lesser extent CRISPRs, can be envisaged, but this is not so for the majority of short interspersed repetitive sequences. Nonetheless, studies of specific elements in particular genetic contexts have uncovered evidence of functional roles ranging from transcription termination and control of mRNA stability, to binding sites for DNA polymerase I

(reviewed in [60]). However, the fact that the distribution and abundance of elements show substantial among-strain diversity [100, 130] suggests that the range of functional roles is incidental, arising from, for example, co-option or genetic accommodation [50].

Differences in the distribution and abundance of repetitive elements among closely related strains carries additional significance in that it suggests that the evolution of these elements is independent of the core genome. This is particularly apparent from comparisons of closely related strains. For example, *Pseudomonas fluorescens* isolates SBW25 and Pf0-1 are closely related and yet highly dissimilar in terms of the nature, abundance and distribution of interspersed repetitive elements [100], even, as shown here, at the level of REPs. While this may reflect unequal rates of element loss, an alternative possibility is independent acquisition. Implicit in this suggestion is the notion that repetitive elements are genetic parasites [50, 63, 127].

3.1.4 Overview

The work presented here defines the minimal replicative unit for a class of interspersed repetitive sequences. Beginning with focus on the *P. fluorescens* SBW25 genome a simple, transparent and assumption-free approach to characterize common short sequences is employed. Suitable null models are used to show that over abundant short sequences – which cannot be accounted for by mutation pressure – fall into three separate groups, each with characteristics typical of REPs. By characterizing REP distribution and conservation REP doublets as opposed to REP singlets are shown to be the replicative unit, which will be referred to as REPINs (REP doublets forming hairpins). Excision events identified in population sequencing data suggest that REPINs are mobile and possibly represent transposition intermediates. Together the evidence presented here suggests that REP sequences organized as REPINs, define a class of hitherto unrecognized miniature non-autonomous mobile DNA in *P. fluorescens* SBW25.

3.1.5 Aims

The overall aim of this chapter is to investigate the within-genome evolution of REPINs. Specifically the aims are:

(1) Identify the most abundant class of short repetitive sequences in SBW25 through comparisons to suitable null models.

(2) Elucidate the distribution of the most abundant class of short sequences (REPs) in SBW25.

(3) Determine the replicative unit for REP sequences.

(4) Characterize higher order arrangements of REP sequences in the SBW25 genome.

3.2 Results

3.2.1 Short sequence frequencies in *P. fluorescens* SBW25 and *P. fluorescens* Pf0-1

Defining repetitive DNA on the basis of short sequences ranging from 10 – 20 nucleotides is simple and can be done logically without invoking heuristics and approximations (for longer sequences exact repetitions are rare). Figure 3.1 shows that the *P. fluorescens* SBW25 genome harbours numerous repetitive sequences: the most common 10-mer occurs 832 times; the most common 20-mer occurs 427 times. While these numbers appear significant, it is possible that they are no more than expected by random chance. To test this hypothesis, 100 random genomes were generated, with the same dinucleotide content, replication bias and length, as the SBW25 genome. The frequency of the most abundant oligonucleotides was determined from both leading and lagging strands. Figure 3.1 shows that the most abundant 10-mer from the randomly generated genomes occurs 304 times. For longer sequence lengths this number rapidly decreases (four instances in the case of 20-mers): the number of repeats expected by chance alone is thus much lower than observed. In total, there are 108 different 10-mers and 14,351 different 20-mers that occur significantly more often in the *P. fluorescens* genome than the most abundant oligonucleotides from randomly generated genomes ($P < 0.01$, Figure 3.2). While compelling evidence for the existence

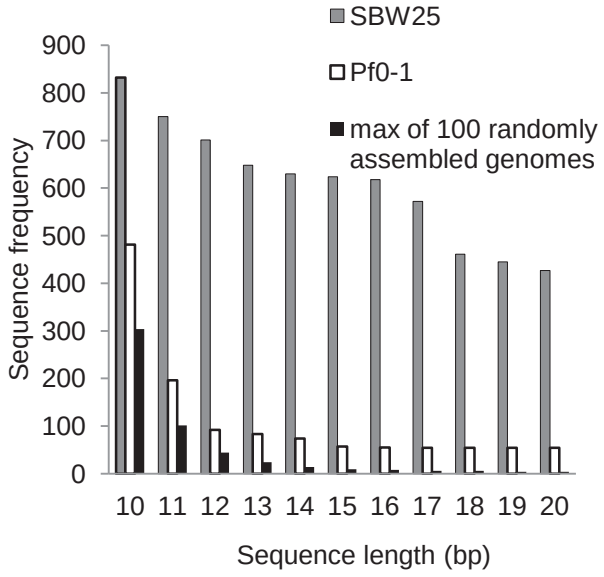


Figure 3.1. Frequency of the most common oligonucleotides in the genome of *P. fluorescens* SBW25 compared to a random model, and to the closely related *P. fluorescens* Pf0-1 genome. The random model is based on 100 genomes generated with the same dinucleotide content, replication bias and length as the SBW25 genome. *P. fluorescens* Pf0-1 shares the same GC-content as SBW25 and highly similar dinucleotide content; coding density differs by 1.7% and the genome length differs by 4%.

of over-representation of short sequences, gene duplications could in part account for these findings [136]. Hence, an alternative null model was sought.

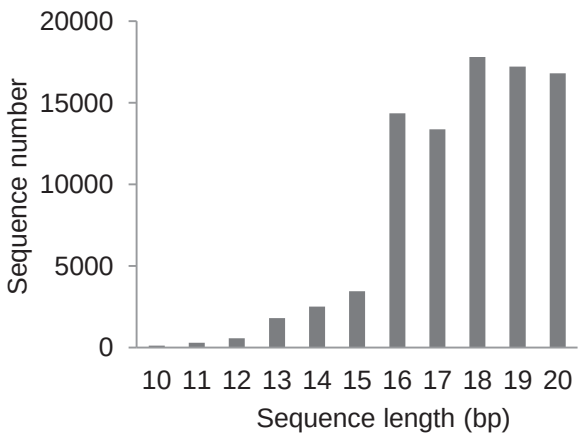


Figure 3.2. Number of different oligonucleotides in the genome of *P. fluorescens* SBW25 that occur more often than the most frequent oligonucleotides from randomly assembled genomes.

P. fluorescens Pf0-1, one of the closest relatives of SBW25, shares the same GC-content and has a highly similar dinucleotide content (Table A); coding density differs by 1.7% and the genome length differs by 4% (6,722,539 bp for SBW25 and 6,438,405 bp for Pf0-1, [100]). The close similarity means that any bias in the representation of short sequences due to duplicative evolutionary processes, or other selective mechanisms, should be similar in both genomes.

As in SBW25, over-represented short sequences in Pf0-1 are more frequent than expected by chance (Figure 3.1), however, a considerable difference in short sequence frequency is apparent. The difference between SBW25 and Pf0-1 is greatest at a sequence length of 16, where the most abundant sequence in SBW25 occurs 618 times, over 11 times more frequently than the most abundant 16-mer in Pf0-1 (Figure 3.3). On the basis of comparisons to both the random null model and the Pf0-1 genome all 16-mers occurring more than 55 times (the frequency of the most abundant 16-mer in Pf0-1) in the SBW25 genome were deemed over-represented. This led us to reject the null hypothesis that chance alone explains the occurrence of short repetitive sequences in the SBW25 genome. Accordingly, over-representation of oligonucleotides is attributed to selective processes.

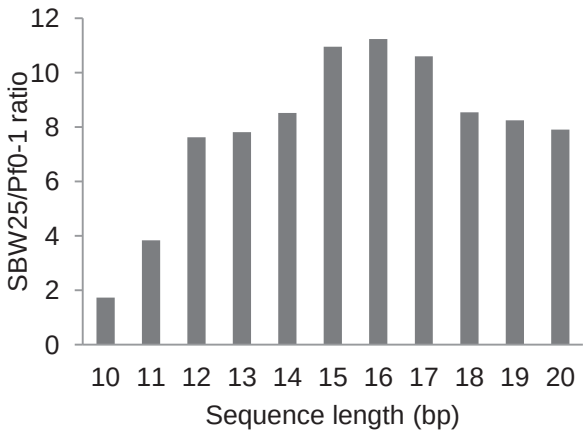


Figure 3.3. Ratio between the most abundant oligonucleotides from SBW25 and Pf0-1.

3.2.1.1 Short repetitive sequences in *P. fluorescens* SBW25 are synonymous with REPs

The collection of over-represented 16-mers together encompasses 96 different sequences; however, a cursory glance suggested that many share similarity. Using a grouping method designed to detect overlapping subsets of sequences (Methods Figure 2.1), the 96 sequences were found to be members of just three separate sequence groups (GI, GII and GIII (Figure A1.1)), each containing an imperfect palindrome (the palindrome overlaps the most abundant 16-mer in GI and GII, but is part of the most abundant 16-mer in GIII (Table 3.1)). The most abundant 16-mers of each group together occur 1,067 times. The majority of these sequences are extragenic; only 14 16-mers overlap with genes. Together these data show that the three groups of 16-mers are over-represented in the SBW25 genome, contain an imperfect palindromic core and are primarily extragenic. Possessing the hallmarks of repetitive extragenic palindromic (REP) sequences, the three groups of 16-mers are, for all intents and purposes, REPs.

Table 3.1. **Short repetitive sequence groups in the SBW25 genome.**

Group[a]	Sequence[b]	Occurrences	Palindromic core[c]
I	GTGGGAGGGGGCTTGC	618	GGGGGCTTGCCCCC
II	GTGAGCGGGCTTGCCC	241	GCGGGCTTGCCCCGC
III	GAGGGAGCTTGCTCCC	208	GGGAGCTTGCTCCC

[a]16-mers were sorted into three groups (GI, GII and GIII) using a grouping algorithm (Figure 2.1 & Figure A1.1). [b]Sequence of the most common 16-mer from each group. [c]Each GI, GII and GIII sequence either contains, or overlaps, an imperfect palindrome (the palindromic core).

3.2.1.2 Determining REP sequence family size

In order to accommodate the possibility of related family members, a pool of sequences that differed to GI, GII and GIII sequences by up to four bases was generated. This resulted in 488,373 different 16-mers of which 1,861 were located in extragenic space. To define the proportion of false positives the search was repeated by interrogating randomly generated extragenic space (with the same dinucleotide content and length of each individual extragenic space) for matches to the 488,373 different 16-mers. This showed that 12 % of all sequences with up to four substitutions are false positives (sequences unrelated to GI, GII or GIII). Repeating the analysis with the subset of sequences, which differ firstly by three and subsequently, two substitutions showed that

2 % and 0.2 % of matches are false positive, respectively. For two substitutions the false positive rate is low enough to conclude that the described repetitive sequence families consist of at least 1,422 members (Table 3.2). The precise number of members belonging to each of the GI, GII and GIII groups cannot be determined because with a degeneracy of two, some sequences fall into more than one group.

Table 3.2. Frequency of GI, GII and GIII 16-mers in the extragenic space of the SBW25 genome

	Number of occurrences	
Number of 16-mers[a]	**Extragenic space**	**Randomly assembled extrangenic space**[b]
0 substitutions (3 sequences)	1053	< 0.01
1 substitution (147 sequences)	1249	0.13 ± 0.33
2 substitutions (3,387 sequences)	1422	2.24 ± 1.41
3 substitutions (48,707 sequences)	1560	31.18 ± 5.18
4 substitutions (488,373 sequences)	1861	264.74 ± 15.87

[a]In order to identify closely related members of each GI, GII and GIII sequence family extragenic space was searched for all possible sequences that differed by up to four substitutions. The number in brackets is the number of variant sequences: e.g., with no substitutions allowed there are just the three sequences (Table 3.1); allowing for one substitution there are 147 different sequences, and so forth. The number found in extragenic space was compared to a null (random) model based on randomly assembled extragenic space (see text). [b]Data are means and standard deviation from 100 independent extragenic space randomizations.

3.2.2 The distribution of REP sequences in the genome of SBW25

The selective causes for the prevalence of GI, GII and GIII sequences in the SBW25 genome are of considerable interest. Although implicit in many studies is the notion that REP-like sequences have evolved because of their selective benefit to the cell (as transcription binding sites, termination signals and the like [132, 137, 138]), it is also possible that selection has favoured their evolution as a consequence of benefits

delivered to a genetic (parasitic) element, of which the repeat sequence is a component. The highly significant differences in the frequency, nature and genomic location of short repetitive sequences in SBW25, compared to Pf0-1 make a compelling case for the latter.

If the prevalence of GI, GII and GIII sequences is a consequence of gene-level selection, then this implies the existence of a replicative entity – a genetic element that has the capacity to reproduce within the genome. The distribution of REP sequences is likely to provide some information. One way to quantify the distribution is to measure distances between neighbouring REP sequences and compare these to distances between REPs generated by a null (random) model. If individual REPs are randomly distributed then this would suggest the individual REP as replicative unit. If the distance between adjacent REPs is non-random, then this may suggest the evolving entity is some higher order arrangement of REPs.

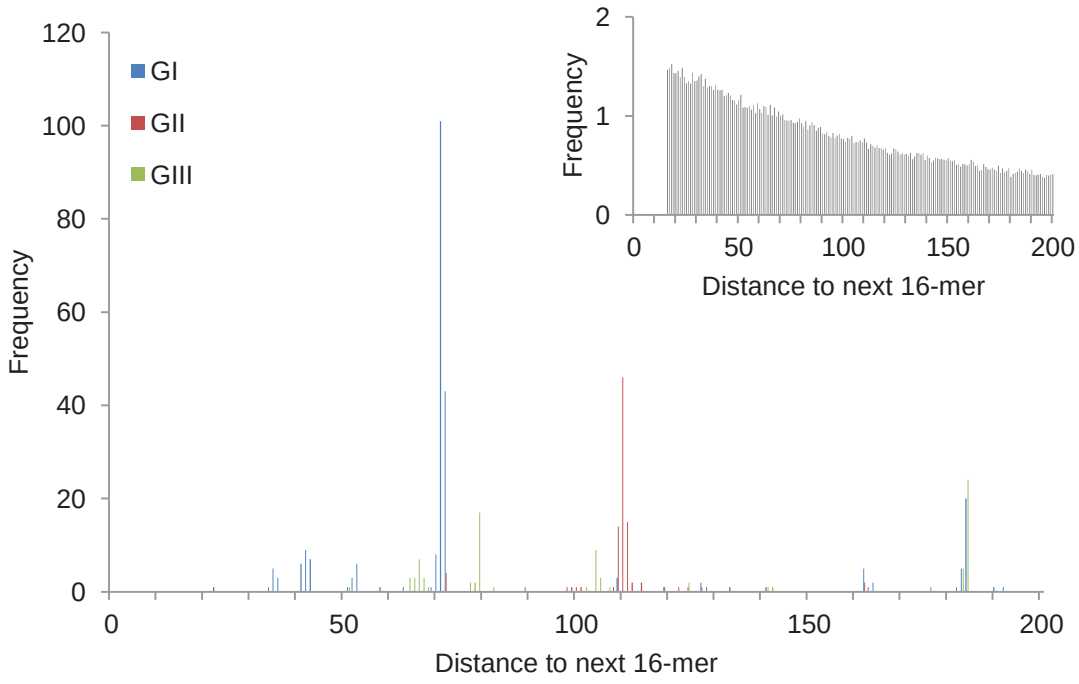


Figure 3.4. Frequency of next neighbour distance for 1,053 GI, GII and GIII sequences from *P. fluorescens* SBW25 compared to a random model (inset). Data are next neighbour distances for GI, GII and GIII sequences in extragenic space. The peaks at 71 and 110 bp correspond to doublets of GI and GII sequences, respectively. The peak at 184 bp corresponds to GI-GIII tandem repeat clusters (see text). No significant deviation from the random model was noted for next neighbour distances above 200 bp. The next neighbour distances of 16-mers randomly assigned to extragenic space is the average of 10,000 simulations (inset).

To construct the null model, 1,053 (the number of invariant GI, GII and GIII sequences in extragenic space) non-overlapping 16 bp segments were positioned at random within the extragenic space of the SBW25 genome. This process was repeated 10,000 times and the average occurrence of the distance between neighbouring elements calculated. Equivalent data for the 1,053 over-represented REPs is shown in Figure 3.4. A comparison between the two histograms reveals marked differences in the distributions of distances between next-neighbours. Most striking is the strong bias toward specific inter-element distances. This marked skew shows that REPs are not independently distributed and is suggestive of an underlying copying mechanism involving at least two REP sequences. Of note is the fact that doublets typically comprise pairs of identical GI, GII or GIII sequences and are rarely mixed (although some exceptions are discussed below) (Figure 3.4)

3.2.3 The replicative unit

To explore the possibility that the replicative unit is an entity comprised of two REP elements (a REP doublet) the number of singlets, doublets, triplets and higher order arrangements of REPs was determined (REP clusters) by examining the 400 bp flanking either side of each REP for the presence of REP sequences (Methods Figure 2.2). Once again, the results of this analysis were compared to the null (random) model used above.

3.2.3.1 The frequency of higher order arrangements (clusters) of REP sequences

According to the random model, 58 % of all REP sequences are expected to occur as singlets, whereas data from SBW25 shows that just 18 % are singlets. In contrast, 61 % of all REPs are organized as doublets, which is significantly greater than the 17 % expected by chance (Table 3.3). Interestingly, REP triplets are rarer than expected, whereas several higher order arrangements of REPs, including two sets of twelve (see below), are more frequent than expected (Table 3.3).

Table 3.3. Frequency of REP clusters within the SBW25 genome

Cluster Size	**Number of occurrences**		***P*-Value**	
	Observed[a]	**Expected (random model)[b]**	**≤ [c]**	**≥ [d]**
1	267	832 ± 22.24	1	0
2	431	181.4 ± 11.12	0	1
3	26	44.3 ± 6.1	0.9998	0.0009
4	12	13.1 ± 3.42	0.6658	0.4537
5	1	4.38 ± 1.96	0.9893	0.0615
6	6	1.67 ± 1.03	0.0070	0.9989
7	5	0.66 ± 0.65	0.0007	0.9999
8	5	0.31 ± 0.46	0	1
9	3	0.14 ± 0.35	0.0006	1
10	0	0.07 ± 0.25	1	0.9364
11	0	0.04 ± 0.18	1	0.9658
12	2	0.02 ± 0.14	0	1
Sum	1422	1421.76		

Data are the number of REPs occurring as clusters (from singlets to clusters of 12) in extragenic space compared to expectations from a null model based on the random assignment of 1,422 16-mers (to extragenic space) (see text). [a]Observed occurrences from the SBW25 genome. [b]Expected values (means and standard deviation) based on 10,000 simulations. [c]The proportion of times the observed frequency was less than or equal to the expected value. [d]The proportion of times the observed frequency was greater than or equal to the expected value.

The highly significant over-representation of REP doublets suggests that the doublet defines an appropriate replicative unit. If true, then the distribution of doublets across extragenic space should be unaffected by neighbouring REP elements and should thus conform approximately to a null (random) model.

Table 3.4. Frequency of REP doublets within the SBW25 genome

		Number of occurrences	
Segment length	**Cluster size**	**Extragenic space**	**Randomly assigned 16-mers**[a]
71 bp	2	457	434.76 ± 12.9
	4	13	46.3 ± 5.75
	6	11	7.69 ± 2.6
	8	8	1.63 ± 1
	10	0	0.4 ± 0.5
	12	2	0.12 ± 0.3
	14	0	0.03 ± 0.18
	16	0	0.01 ± 0.1
	18	0	0.002 ± 0.06
110 bp	2	457	419.2 ± 13
	4	13	49.1 ± 5.9
	6	11	9.4 ± 2.8
	8	8	2.2 ± 1.2
	10	0	0.7 ± 0.6
	12	2	0.2 ± 0.4
	14	0	0.09 ± 0.25
	16	0	0.02 ± 0.16
	18	0	0.02 ± 0.1

Data are the frequency of REP clusters (from doublets to cluster of 18 REPs) found in extragenic space compared to a null model based on the random assignment of 560 x 71 bp and 560 x 110 bp segments (to extragenic space). REP clusters containing an uneven number of REP sequences are included in the next lower cluster size (REP singlets are omitted). [a] Data are means and standard deviation of 10,000 simulations.

To test this hypothesis, random distributions of REP doublets over extragenic space were compared to actual REP clusters found in SBW25 (Table 3.4). However, because

the distance between REPs (in the doublet conformation) varies (Figure 3.4), two random models were generated based on the two most common inter-REP spacings: 71 bp (a doublet of GI REPs) and 110 bp (a doublet of GII REPs). Simulations were based on the random assignment of 560 REP doublets (corresponding to the sum of REP clusters (of two or more) in Table 3.3) to extragenic space and were repeated 10,000 times. Although the two segments differ significantly in size, simulations for each family gave remarkably similar results (Table 3.4). Together these data show that the observed number resembles that predicted if the doublets are randomly distributed.

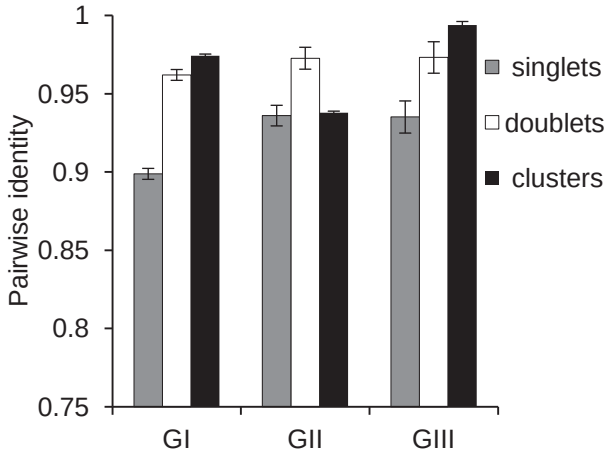


Figure 3.5. Average pairwise identity of REP sequences found in singlets, doublets and clusters. Data are average pairwise identity of REPs found as singlets, doublets and clusters (clusters contain more than three REPs). Error bars show standard deviation. Statistical testing (jackknife) shows the average pairwise identity of 16-mers from REP doublets (and clusters for GI and GIII, *P*-value < 1e-10) to be significantly greater than the average pairwise identity of 16-mers obtained from REP singlets: this is true for comparisons within each of the REP groups ($P < 1e\text{-}10$ for GI; $P < 1e\text{-}8$ for GII and GIII).

3.2.3.2 Comparison of the conservation of REPs in singlets, doublets and clusters

A further prediction concerns evolutionary processes affecting doublets vs. singlets. If REP doublets are the replicative unit, then singlets are likely to derive from doublets, either by decay (divergence) of the neighbouring element, or by destruction of the doublet through insertion or deletion. In either case the REP singlet is expected to be non-functional (immobile) and thus subject to random genetic drift. REP doublets on the other hand – being (according to the hypothesis) functional and potentially mobile – are expected to be shaped by selection: genetic diversity of REP singlets should thus be greater than doublets. To test this hypothesis GI, GII and GIII sequences were extracted from the SBW25 genome plus all related sequences that varied by up to two positions. Since only two nucleotide differences distinguish GII and GIII sequences from a GI sequence, GII and GIII sequences were defined by two fixed (invariant) positions (GII: 2T, 6C; GIII: 6A, 13T). After extraction, sequences from each group were divided into a set of 16-mers obtained from singlets, a set of 16-mers from doublets and a set of 16-mers obtained from

clusters (where a cluster contains three or more REPs). For all nine sequence groups (three from each GI, GII and GIII group) the pairwise identity was calculated (Figure 3.5, see Methods for details). The average pairwise identity of 16-mers obtained from REP doublets is significantly greater than the average pairwise identity of 16-mers obtained from REP singlets: this is true for comparisons within each of the REP groups (P < 1e-10 for GI; P < 1e-8 for GII and GIII).

Table 3.5. Characteristics of REP doublets found in the SBW25 genome.

REP doublet group	Distance between REPs	Number of occurrences within SBW25	REP orientation[a]
GI	35	5	AA-TT
	36	3	
	41	6	
	42	9	
	43	7	
	51	2	
	52	3	
	53	6	
	70	8	TT-AA
	71	102	
	72	43	
GII	72	4	TT-AA
	109	14	
	110	50	
	111	17	
	112	2	
GIII	64	3	TT-AA
	65	3	
	66	7	
	67	3	
	68	3	
	77	2	

78	2
79	17

[a]Shows the two bases that are observed in the centre of each palindrome. In a REP doublet the central two bases of each REP on the top strand determines the doublet's orientation.

3.2.3.3 REP doublet diversity

REPINs show a considerable level of within group diversity. Table 3.5 shows a variety of observed distances between the two REP sequences within a REP doublet. For GI REP doublets alone 11 different distances are observed. However, all 11 distances can

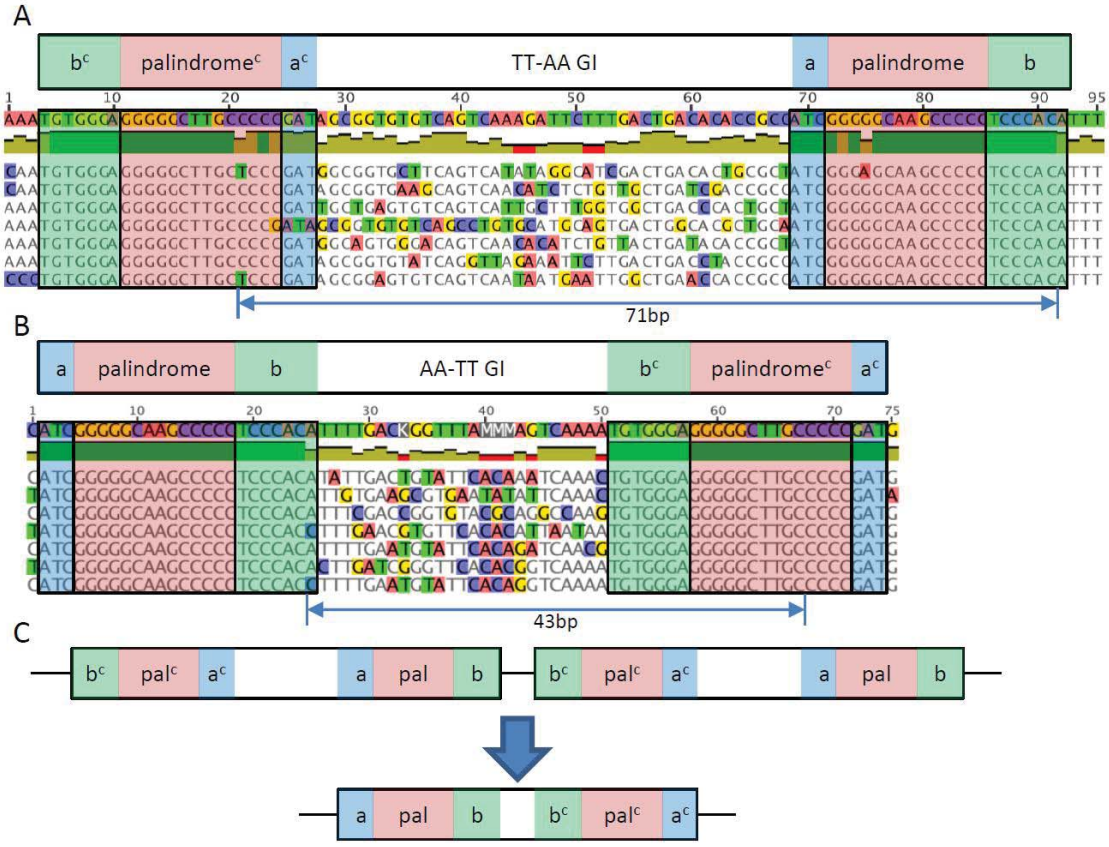


Figure 3.6. REP sequence orientation within GI doublets. (*A*) Alignment of 101 GI REP doublets from SBW25 (seven are shown) that are found at a distance of 71 bp to each other. REP sequences within the doublet are found in opposite orientations and are divided by a less conserved spacer sequence. Each REP sequence consists of a palindrome, a 5′ and a 3′ flanking sequence. The bases in the centre of each palindrome indicate the orientation within the doublet. TT is found in the centre of the first palindrome and AA in the centre of the second, hence, the shown doublet is of type AA-TT. Three conserved As and Ts are found at the 5′ and 3′ end respectively, indicating the co-option of this REP doublet class as transcription terminator. (*B*) Alignment of the less commonly found AA-TT GI doublet conformation separated by 43 bp. Note that the conserved As and Ts at the 5′ and 3′ end of the alignment do not exist. However, As are found at the 5′ end of the b^c sequence and at the 3′ end of the b sequence similar to GI doublets in TT-AA orientation. (*C*) A potential scenario for the evolution of AA-TT GI doublets from TT-AA GI doublets. An accidental transposition of the 3′ and 5′ end of two co-localized TT-AA GI doublet could have been sufficient to create the new AA-TT REP doublet type.

be sorted into only four groups, where a group only consists of consecutive distances.

GI REP doublets not only show a diverse set of REP sequence distances but are also found in two different orientations. Since REP sequences are imperfect palindromes they have an orientation, which is determined by the imperfections in the palindrome, namely the central two bases (AA or TT for all REP sequence groups in SBW25). Furthermore, the orientation of a REP sequence can also be determined by the sequences flanking the palindrome. Since the vast majority of REP doublets consist of two inverted REP sequences, there are two possible doublet configurations, either the predominant TT-AA configuration (most GI, all GII and GIII doublets) or the much less common AA-TT configuration (minority of GI doublets, Figure 3.6A and B). Interestingly, GI doublets in TT-AA configuration are flanked by multiple conserved 'A's and 'T's on the 5′ and 3′ end respectively, which is likely to be a result of co-option of the REP doublet for transcription attenuation [137]. GI doublets in AA-TT orientation are not flanked by runs of 'A's or 'T's, however there are 'A's and 'T's directly flanking the REP sequences inside the doublet (Figure 3.6A and B). This suggests that the AA-TT configuration evolved from the 3′ and 5′ REP sequences of two co-localized TT-AA GI doublets (Figure 3.6C).

3.2.3.4 Evolution of long palindromic singlets

While analysing REP singlets, usually consisting of a 5′ flanking sequence (*a*), a central palindrome and a 3′ flanking sequence (*b*), long palindromic sequences with the structure *b-palindrome-*b^c (b^c is the complement of *b*) or *a-palindrome-*a^c were observed. Interestingly, the observed long palindromic REP sequences could be created when the central sequence of a REP doublet is excised (Figure 3.7A). If this hypothesis is true then one would predict to find both types of long palindromes only for GI sequences, since only GI doublets occur in AA-TT and TT-AA orientation. Figure 3.7B shows that in line with our prediction only for GI sequences both types of long palindromic REPs are found. Furthermore the abundance of the observed long palindromes correlates with the abundance of the respective doublet configuration, which further supports the hypothesis that the observed long palindromes arose from REP doublets.

3.2.3.5 REP doublet structure

Analysis of the organization of REP doublets shows that in the majority of cases, pairs of REPs (93 % of all 430 REP doublets) – of either the GI, GII, or GIII types – are organized as two inverted REP sequences that overlap the most abundant 16-mer (Figure 3.8A & B). While the spacer region between REPs shows less conservation than evident in the REPs themselves, secondary structure predictions for ssDNA shows that the conserved bases on each side pair resulting in a hairpin (Figure 3.8E). Thus, while selection appears to favour highly conserved nucleotide arrangements for REP and adjacent sequences, the critical features of the intervening sequence would appear to be length, and capacity to form a hairpin. Indeed, compensatory changes on either side of the predicted hairpin are common (Figure 3.8A).

3.2.3.6 Evidence of REP doublet excision (mobility) in Illumina sequencing data

Finally, if the assertion that the doublet defines a replicative entity is correct, then evidence of movement could in principle come from population sequencing. To this end 55,768,706 paired-end Illumina reads (36-76 bp long) obtained from sequencing

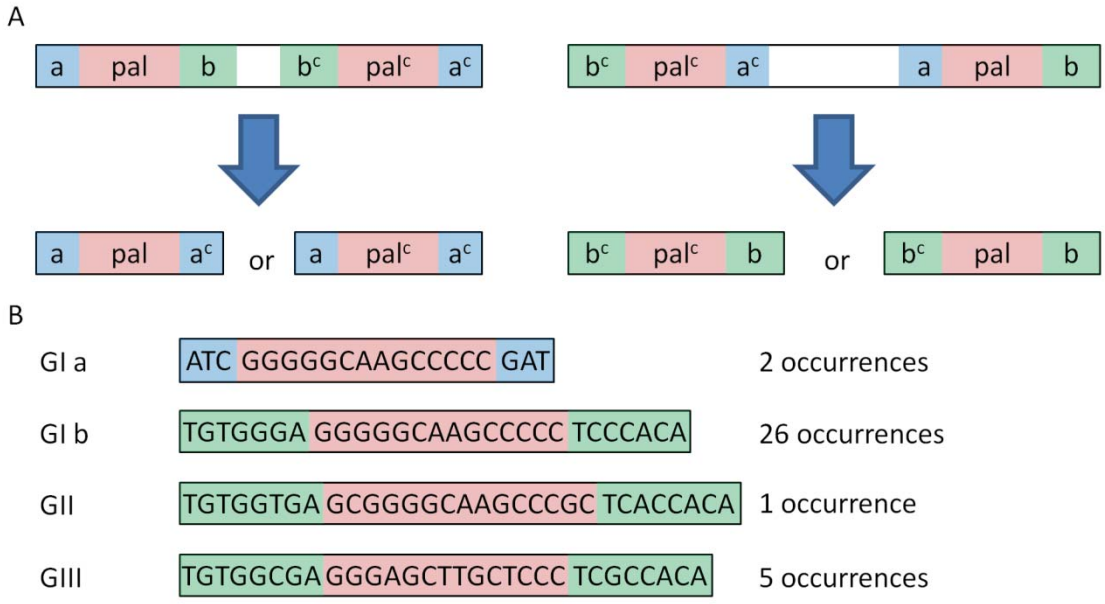


Figure 3.7. Unusual long palindromic GI, GII and GIII sequences and their potential evolution from REP doublets found in the SBW25 genome. (*A*) Shown are all four long palindromic GI, GII and GIII sequences together with their frequency found in the SBW25 genome. Note that two configurations are found for GI sequences and only one for both GII and GIII sequences. (*B*) Shows how long palindromic REP singlets could arise from REP doublets through the excision of the central sequence. Hence, REP doublets found in AA-TT orientation would produce *a-palindrome-a*^{c} REPs (*left*) and REP doublets found in TT-AA orientation *b*^{c}*-palindrome-b* REPs (*right*).

DNA extracted from 5 x 10e9 SBW25 cells, were interrogated for evidence of insertion and excision events. A total of 18 putative insertions were detected, however, the possibility of false positives could not be discounted. A similar search for excision events proved more profitable: three single reads were identified which mapped to three different locations on the genome, each corresponding to unique sequences flanking a GI REP doublet (Figure 3.8C and Figure A1.2). However, the expected doublet was absent from all sequence reads leading us to conclude that these sequences were from DNA molecules from which the doublet had excised. Additionally, 200 individual sequence reads were observed spanning a GII REP doublet indicating its excision from the entire population (Figure A1.2). That these events could result from machine and / or chemistry error is improbably low. Furthermore, a search for evidence of REP singlet deletions from the ~56 million Illumina reads failed to find evidence of a single such event (see Methods).

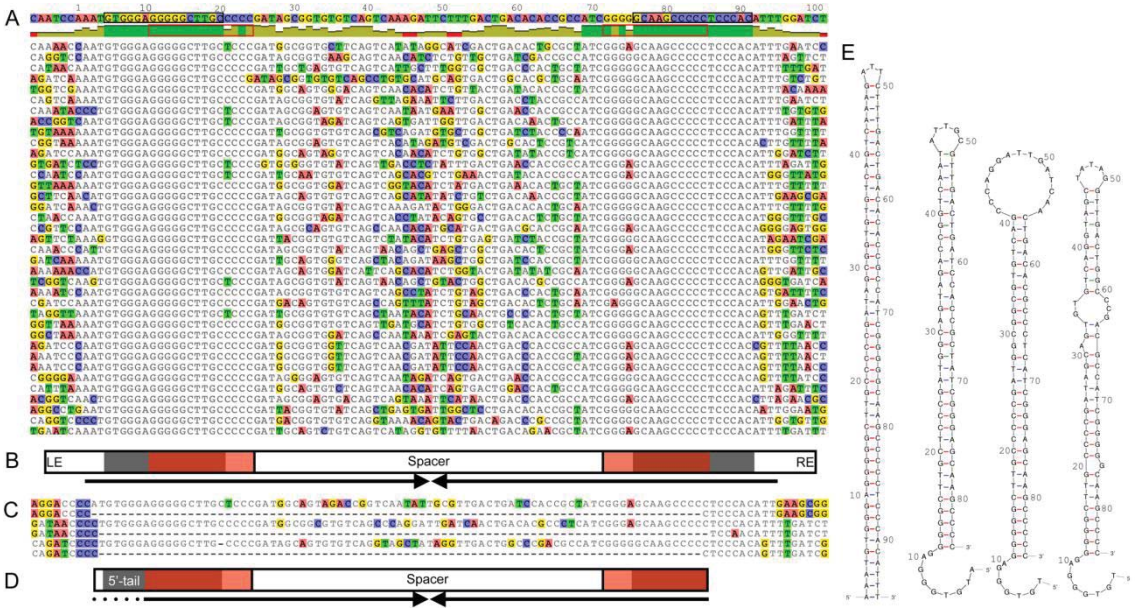


Figure 3.8. General organization and predicted secondary structure of REP doublets forming hairpins (REPINs). (*A*) Alignment of 101 GI REP doublets from SBW25 (37 are shown) shows a symmetrical (palindromic) organization comprised of two highly conserved regions separated by a spacer. Top line shows the consensus sequence followed by a graph displaying identity to the consensus (green denotes 100% identity). Two invariant regions of 16 bp are found in the left and right ends (LE, RE). These sequences are organized as inverted repeats and define the most abundant 16-mer in the SBW25 genome (black box). Each 16-mer overlaps a GI REP sequence (red box). (*B*) General REPIN features including LE and RE, each comprised of a highly conserved 16-mer (black) overlapping a REP sequence (red), with the two ends separated by a spacer. For a GI doublet the distance between the first residues of the two invariant 16-mers is 71 bp. Complementary bases permit formation of a hairpin structure (arrows). (*C*) Three excision events detected from Solexa sequencing reads reveal a putative transposition intermediate. Full length sequences show three genomic regions located between 2,577,312-2,577,231, 3,857,520-3,857,439 and 5,683,545-5,683,624 bp on the SBW25 genome each of

which contains a REPIN. The partial sequences below each genomic region are Solexa reads from which the REPIN has been excised (see also Figure A1.2). (*D*) Cartoon of the excised region indicating putative transposition intermediate. Note the 5′-tail which generates an asymmetrical sequence. (*E*) Secondary structure prediction for the consensus GI REPIN shows that the conserved bases on each side can pair resulting in a long hairpin (*E*, left). Predictions for transposition intermediates in the same order as the alignments in (*C*): the second, third and fourth hairpin correspond to the first, second and third alignment. The single stranded 5′-tail is free to pair with a complementary sequence.

Details of the excised doublets are shown in Figure 3.8C & D. Of particular interest is the asymmetrical nature of the deleted sequence: in both instances it begins (in the lefthand (5′) end (Figure 3.8B)) at the start of the invariant sequence defined by the most conserved 16-mer and extends through the spacer region into the second REP sequence. However, rather than finish at the end of the conserved 16-mer as expected, the deletion truncates at the 3′-end of the righthand REP sequence leaving the last ~6 bp of invariant sequence intact (Figure 3.8C).

Secondary structure predictions show a hairpin structure with a 5′-single strand tail. Although the structures of the two hairpins are not identical (due to differences in the sequence of the space region) the 5′-tail is a feature of the excised entity in both instances (Figure 3.8E). It is likely that the excised sequences define the transposition intermediate.

Additionally to the excision of a whole REP doublet, the excision of the central sequence of a REP doublet, leaving a long palindromic REP sequence behind, was identified in population sequencing data (Figure 3.9). The sequence was cut at the 3′ end of the 5′ palindrome and at the 3′ end of the 3′ palindrome leaving a long palindromic REP sequence behind. This excision is a symmetric cut (cut on 3′ end of both palindromes) as opposed to the REP doublet excision in Figure 3.8, which is asymmetric (cut on the 5′ end of the 5′ REP and on the 3′ end of the 3′ REP). The effect of these events is entirely unclear. Although one could speculate that it is simply an alternative way of transposing REP doublets. However, this immediately raises the question, why there are two ways of REP doublet transposition. Alternatively it could be a way to reduce the numbers and activity of REP doublets within the genome (single REP sequences are predicted to be immobile) without losing their functionality as transcription terminator (long palindromes are able to form long hairpin structures and are still flanked by runs of ‘A’s and ‘T’s). Nevertheless, the diversity of different REP doublet structures and the observation of different excision events are a testimony for the complexity of REP doublet biology within the SBW25 genome.

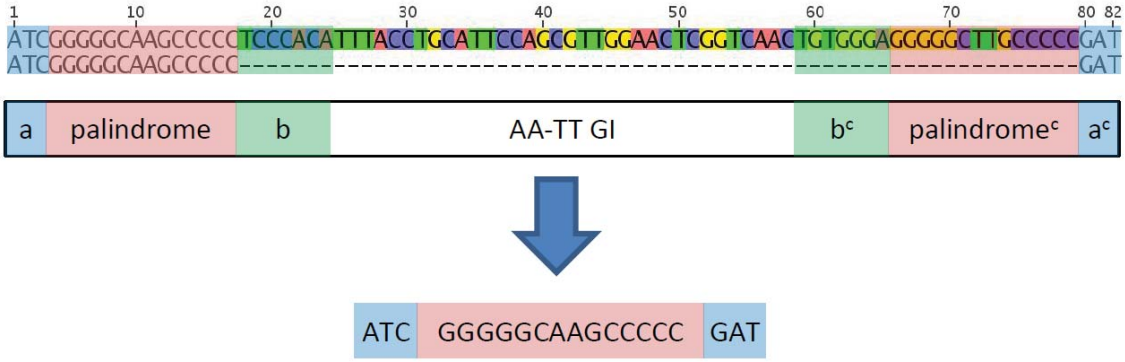


Figure 3.9. Incomplete symmetric excision event of a REP doublet detected in Illumina sequencing data. The first line of the alignment shows the genomic sequence of SBW25 from position 598,553 to position 598,634. The second line of the alignment shows part of the sequence read that maps perfectly to the corresponding genome sequence apart from the excision in the centre of the read. The cartoon below the alignment shows the general composition of the GI REP doublet. The last line in the picture shows the remaining REP sequence found in the sequence read. It only contains flanking sequence (a), the central palindrome and flanking sequence (a^c).

Together the above analyses implicate REP doublets as a unit of selection: a family of mobile DNA that has, until now, eluded recognition. Although REP doublets have previously been noted as one of many different higher order arrangements of REPs, they have not before been implicated as replicative entities [58, 59, 130-132]. Furthermore, in previous discussions of higher order arrangements it has been assumed that the singlet is the basic building block. In contrast, the presented data supports the view that REP singlets are defunct remnants of once functional REPINs. Because of their likely evolutionary relevance, a label that defines the replicative entity appears warranted. Henceforth REP doublets forming hairpins will be referred to as REPINs.

3.2.4 Higher order arrangements of REP sequences

3.2.4.1 REPIN clusters

While the majority of REPINs exist as singlets, some higher order arrangements are apparent (above and Table 3.4). These are of two main types: those showing a distinctive ordering and those with no apparent structure.

REPINs occurring in ordered clusters are typically arranged as tandem repeats of nearly identical REPINs – including the flanking sequences (Figure 3.10). With 16 such clusters distributed throughout the genome, these arrays are the most common higher

order arrangement of REPINs in SBW25. The largest cluster consists of four REPINs (plus an additional REP sequence) with a total length of over 700 bp.

Figure 3.10. A sketch of a typical tandemly repeated REPIN cluster. The cluster comprises two tandem repeat units. Each unit consists of a 5′ flanking sequence (f1) followed by a REPIN and ends with a second shorter flanking sequence (f2). The two units are usually separated by a short stretch of DNA that is not repeated.

REPINs in clusters lacking obvious organization are found in five regions of the genome and typically consist of two unrelated REPINs. Close inspection suggests that these clusters are formed by insertion of REPINs into, or next to, existing REPINs.

3.2.4.2 Tandemly repeated REP sequences

REPs also form higher order arrangements. These are of two distinct types: the first involves highly organized tandem arrays of GI and GIII REP sequences: GI REPs are separated from GIII REPs by 112 bp; GIII REPs are separated from GI REPs by 72 bp. Five such tandem arrays are located at ~2 Mbp all of which are found in forward orientation, six are found ~4 Mbp in reverse orientation (at a distance of ~2Mbp from the origin of replication). The two largest tandem arrays both contain 12 GI and GIII sequences, one found at ~4.1 Mbp the other at ~2.5 Mbp (Figure 3.11). These two arrays are almost identical copies of each other, but found in opposite orientations on opposite sides of the genome. The second type of tandemly organized REP sequences consists solely of evenly spaced GI sequences found at two positions in the genome.

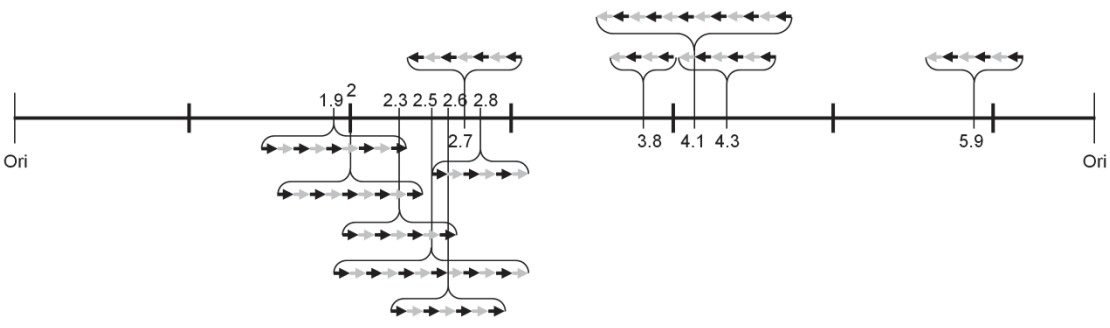


Figure 3.11. Approximate positions of the two largest tandem repeat clusters in the genome of SBW25. The tandem repeats are formed by sequences from GI and GIII. The gray and black arrows indicate different sequence lengths.

Similar to the GI-GIII tandem arrays one GI tandem array is found in forward and the other one in reverse orientation.

3.3 Discussion

3.3.1 Short repetitive sequences

Short interspersed repetitive sequences are widely distributed in bacteria, but past studies have shed little light on their evolutionary origins. The study of the abundance of short sequences in *P. fluorescens* SBW25 together with comparisons against a random (null) model, and subsequently against the data from the close relative *P. fluorescens* Pf0-1, revealed the presence over-represented short sequences, thus indicating that natural selection is a primary driver of their evolution. Moreover, these short repetitive sequences are shown to fall into three distinct groups (GI, GII and GIII), each bearing characteristics typical of REP sequences, that is, they are repetitive, extragenic and palindromic.

3.3.2 The replicative unit

A critical issue is the nature of the entity upon which selection acts. Evidence that this entity comprises a doublet of REP sequences – a REP doublet forming a hairpin structure (REPIN) – came firstly from analysis of the distribution of REPs in extragenic space. The striking departure from a random model shown in Figure 3.4, along with clear bias toward specific distances between REPs, pointed to the REPIN as the replicative entity. The hypothesis was further tested by examining the distribution of REP doublets in extragenic space, by measuring nucleotide diversity in singlets versus doublets, and by analysis of the conserved features of REPINs. Finally, the existence of REPINs as actively mobile entities was bolstered through the discovery of four excision events that may define putative transposition intermediates (Figure 3.8).

A previous analysis of the SBW25 genome using various repetitive DNA finding algorithms [100] revealed numerous repeat families. Two of these, the so named R0 and R2 repeats have characteristics similar to REPINs; indeed, a comparison (Table 3.5) shows a correspondence between REPINs and the R0 and R2 repeats. In general R0 repeats map to GI REPINs, while R2 repeats correspond to a mixture of both GII and GIII REPINs.

Table 3.6 Correlation between REPINs and repeat families previously detected in SBW25.

	GI REPINs[a]	GII REPINs[a]	GIII REPINs[a]
R0[b]	152	0	3
R2[b]	3	85	51
others	37[c]	1	0

[a]Only exact matches of GI, GII and GIII 16-mers were considered when searching for REPINs. [b]Repeat families detected in SBW25 by Silby et al. [100]. [c]The high number of others for GI doublets is attributable to the presence of two different REP orientations within REPINs (TT-AA is found in R0, AA-TT is not).

At first glance the footprints differ from expectations based on bioinformatic analyses in that they do not encompass the full extent of the conserved REPIN (Figure 3.8B): the lefthand end is complete, but the righthand end stops at the end of the right REP sequence (see Results and Figure 3.8C & D). This is curious given that the REPIN as defined by bioinformatic analyses is symmetrical (Figure 3.8A & B). One possibility is that the footprint has nothing to do with REPIN movement, but this seems unlikely. Alternatively, the asymmetry defined by the putative intermediate may provide clues as to a possible mechanism of transposition.

Assuming the footprint left by the excised DNA is a genuine intermediate in REPIN movement then a key issue is the reformation of the symmetrical REPIN. This could happen if REPIN transposition occurred via a single stranded intermediate where the 5′-tail was able to pair with complementary sequence. In this regard it is of interest to note that the 5′-tail is complementary to the 3′-end absent from the putative intermediate. Moreover, secondary structure predictions show that the tail is unlikely to form part of the hairpin (Figure 3.8E). It is possible that the 5′-tail is involved in recognition of the complementary sequence (the target) and that via this recognition event, integrates back into DNA leading to the formation of a new REPIN.

Apart from the asymmetric REPIN excisions, one symmetric excision was detected (both cleavage events at 3′ end of the REP palindrome), leaving a symmetric REP singlet behind (Figure 3.8 and Figure 3.9). However, there are a number of questions that remain. What role does this event play for the distribution of REPINs within the genome? Are the excised sequences integrated back into the genome? Can excisions of

that kind be adaptive or is it just a by-product of REPIN dissemination? Understanding the mechanism of these events could provide great insight into REPIN dynamics and genome evolution.

While the argument for REPINs as replicative entities is supported by substantive data, REP singlets are nonetheless a notable feature of the SBW25 genome. The presented data – particularly the significantly lower pairwise identity of REP singlets compared to REP doublets – suggests that these singlets are non-functional remnants of REPINs. But this does not explain why REP singlets are common. A close analysis of REP singletons reveals several possible routes for single REP sequences to emerge from REPINs. One possibility stems from limitations of our sequence search algorithms. When REPINs evolve neutrally successive acquisition of point mutations naturally leads to one REP becoming more decayed than the partner. If the less decayed REP is only just on the verge of recognition by the sequence search, then it is likely that the more decayed REP partner sequence will escape detection. A biologically plausible possibility is that singlets arise from insertion of DNA into REPINs. Indeed, earlier studies have noted that REP sequences are targets for certain insertion sequences [100, 139, 140]. REP singlets could also arise by deletion of the sequence between two REPs within a single REPIN leading to a long palindromic structure that contains only a single REP sequence: precisely such events can be seen in the genome of SBW25 (Figure 3.8 and Figure 3.9). A further possibility is that selection may act to preserve individual REP sequences because of specific functional consequences [130, 137].

3.3.3 Higher order arrangements of REPs and REPINs

A finding of note is the existence of several higher order arrangements of REPs and REPINs within the SBW25 genome, indeed, several such clusters occurred at a frequency above that expected from the null model (Table 3.3 and Table 3.4). Interestingly the majority of these clusters – at least those containing more than three REP sequences or REPINs – were arranged as highly ordered tandemly repeated units. This and the fact that higher order arrangements were not found in all REPIN containing genomes (see Table A2.2 and see section 4.2.3) indicates a second mechanism for REP/REPIN cluster formation and suggests specific functional roles for these structures.

3.3.4 Concluding comment

Finally, the evolutionary approach for the analysis of short repeats and discovery of REPINs may prove useful for elucidating the origins of different kinds of short, repetitive, interspersed palindromic sequences such as NEMISs [51], ERICs [50] and small dispersed repeats (SDR) [141]. Indeed, REPINs themselves could conceivably constitute the building blocks for a range of more complex repetitive structures. For example, REPINs that incorporate DNA beneficial to a host bacterium are likely to have an advantage over standard REPINs. In this regard it is possible that CRISPRs [89] and related mosaic entities are derived from REPIN-like elements.

Chapter 4:
The cause of REPIN dissemination

Based on: Bertels F, Rainey PB (2011) Within-Genome Evolution of REPINs: a New Family of Miniature Mobile DNA in Bacteria. PLoS Genet 7: e1002132. doi: 10.1371/journal.pgen.1002132. (attached to the end of this thesis)

Also published in: Bertels F, Rainey PB (2011) Curiosities of REPINs and RAYTs. Mob Genet Elements 1. (attached to the end of this thesis)

Contributions: Conceived and designed the experiments: FB PBR. Performed the experiments: FB. Analyzed the data: FB. Wrote the paper: FB PBR.

4.1 Introduction

The work in Chapter 3 resulted in the identification of the replicative unit for REP sequence, which was termed REPIN and consists of two inverted REP sequence (REP doublet) and a short spacer sequence, which together are predicted to form a long hairpin structure in single stranded DNA (see section 3.2.3). This chapter focusses on how REPINs are dispersed within the genome; specifically, the cause of REPIN dissemination will be investigated. This will also provide some insight into the probable mechanistic bases of REPIN dispersal in SBW25.

4.1.1 The importance of transposases in REPIN dissemination

While it is possible that REPINs disseminate by an entirely novel mechanism, it is probable that the causal and mechanistic bases will show at least some level of similarity to those of previously-characterized classes of selfish genetic elements. Due to their short length it seems unlikely that REPINs encode a transposase that allows autonomous transposition. Rather, the collective evidence presented in Chapter 3 strongly suggests that REPINs are non-autonomous mobile genetic elements (see

section 3.2.3). REPIN composition (two repeat sequences (REPs) in inverted orientation) is reminiscent of non-autonomous mobile genetic elements (*e.g.* MITEs) [20]. Discussed in sections 1.2.2 and 3.1.2, MITEs are non-autonomous transposable elements that consist of two inverted repeats. For transposition MITEs rely entirely on the transposase function encoded by an autonomous element that is flanked by the same inverted repeats. If REPIN dispersal indeed resembles that of non-autonomous selfish genetic elements, one might expect to find an autonomous transposase flanked by REPs encoded in the SBW25 genome. Hence, the work in the first part of this chapter concentrates on searching for candidate transposases.

4.1.2 Linkage of REPINs and a novel class of transposases (RAYTs)

While the studies in this chapter were underway, an independent paper was published by Nunvar et al. that recognized an association between REP sequences and a new class of genes in *Stenotrophomonas maltophilia* and a selection of other bacterial genomes [101]. The proteins encoded by these genes show similarities to IS*200* transposases. Since transposases encoded by IS*200* elements are part of the tyrosine transposase family, the gene class was named REP-associated tyrosine transposases (RAYTs). Through comparative studies of RAYTs and IS*200* sequences - and some evidence of co-evolution between REP sequences and RAYTs - the authors concluded that RAYTs are a likely causative agent for the dissemination of REP sequences. However, this paper does not recognize the significance of REPINs for REP sequence dispersal within the genome (since REPs are likely immobile remnants of REPINs see section 3.2.3.2), nor is it clear how the authors made the connection between the highly dissimilar sequence classes of IS*200* transposases and RAYTs (see section 5.2.2).

Independently of the work published by Nunvar et al., the work presented in this chapter ascertains a functional connection between IS*200* insertion sequences and RAYTs. Furthermore, the importance of REPINs for REP sequence dispersal is shown by: (1) identifying REP sequences through their association with RAYTs in 18 bacterial genomes including *E. coli* K-12 and *Nostoc punctiforme*, and (2) analysing higher order arrangements of the associated REP sequences within the respective genomes.

4.1.3 Aims

The overall aim of this chapter is to investigate the possible cause(s) of REPIN dissemination within the SBW25 genome. Specifically, the aims are:

(1) To clearly explain how a functional connection between RAYTs, REPINs and IS*200* insertion sequences was established.

(2) To summarize the similarities between the REPIN–RAYT system and IS*200* sequences.

(3) To study higher order arrangements of RAYT-associated REP sequences in a selection of 18 bacterial genomes.

4.2 Results

4.2.1 Detection of RAYTs, a class of genes linked to REPINs in SBW25

To identify genes linked to REPINs that could be responsible for their within-genome dispersal, the SBW25 genome was searched for genes that are flanked on either side by REPs (the inverted repeats that REPINs consist of). This was achieved using the Artemis genome browser [103]. Particular attention was given to REP-flanked genes previously annotated as transposases [100]. A similar search was performed by Silby et al, and both searches resulted in the identification of two candidate genes: *pflu4572A* and *pflu5832* [100]. Depicted in Figure 4.1, the genomic organization of these two (unlinked) genes is virtually identical; both are flanked on either side by inverted repeats and REP sequences. The organization of the two genes shows high similarity to that of IS*481*, a family autonomous insertion sequences that use two inverted repeats as recognition sequences [142] as opposed to the palindromic recognition sequences (REPs) that would be expected if the corresponding transposases were responsible for the spread of REPINs.

Instead of being the cause for REPIN dispersal the two genes are more likely to have targeted a REPIN for insertion. Insertion into a REPIN would put the insertion sequence including the flanking inverted repeats inside of the REPIN inbetween the REP sequences. This is exactly what is observed in the SBW25 genome (Figure 4.1). The two genes are immediately flanked by two inverted repeats, which in turn are flanked by REP sequences. This indicates that each of the two genes targeted and destroyed a REPIN through insertion as opposed to being the autonomous elements that enable REPINs to move. Insertion sequences have also been reported to target REPs in other bacteria [139, 140].

Figure 4.1 Depiction of *pflu5832* and *pflu4572A* and their flanking sequences. Both *pflu4572A* and *pflu5832* are found inbetween two flanking REP sequences. However, the REP sequences are found outside the flanking inverted repeats (IR/IR' probably used as recognition sequences for the encoded transposase), which indicates that the insertion sequence (of the IS*481* family) inserted into a REPIN, rather than recognizing and transposing REP sequences.

The search performed for this thesis resulted in the identification of two further candidate genes: *pflu2165* and *pflu4255* (annotated as *yafM*). These were found respectively embedded in GIII and GII REPIN clusters. The amino acid sequences encoded by these genes share 64 % identity, indicating that they share a recent common ancestor. A third gene, *pflu3939*, which is similar to *pflu2165* and *pflu4255* was identified in the centre of two GI REPINs.

Upon publication of the SBW25 genome [100], each of these three genes was annotated merely as putative conserved protein, demonstrating that while their function was unknown, their conservation had been recognised. In order to determine whether these genes could encode a transposase, the sequences of the three encoded proteins were searched against an insertion sequence database (*www-is.biotoul.fr*) using BLASTP (basic local alignment search tool protein) [102]. As expected from the lack of annotations in the SBW25 genome, all hits were relatively insignificant (*e*-Value > 0.004). However, the majority of these hits were annotated as insertion sequences of the well-described IS*200* family (see section 4.2.2). This suggested that the three identified genes may share a common motif with IS*200* transposases. As mentioned previously (section 4.1.2), an independent study named the family to which the three candidate genes belong 'RAYTs'. Thus, henceforth these genes will be referred to as RAYTs.

4.2.2 Similarities between IS*200* transposases and RAYTs

In order to identify possible motifs shared by IS*200* transposases and the three RAYT genes identified in the section above, an amino acid sequence alignment of the three SBW25 RAYTs and other RAYT proteins found in a range of different genomes was constructed (Figure 4.2). ISHp*608* (the IS*200*/IS*605* protein for which the transposition mechanism was elucidated [48, 143-145]) was then added to the alignment to enable the comparison of functional features. The alignment in Figure 4.2 shows a number of sites

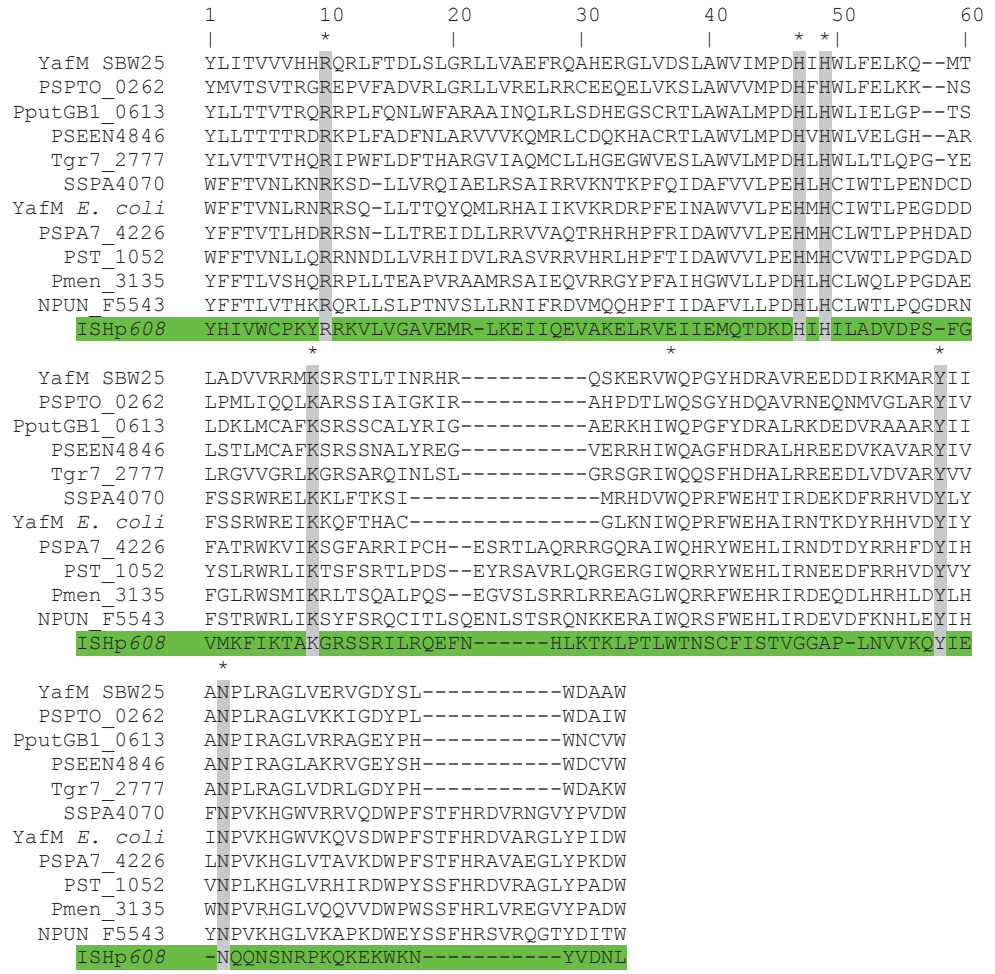


Figure 4.2. Multiple amino acid sequence alignment of REPIN associated proteins (RAYTs) and ISHp608 (green). Stars denote fully conserved amino acid positions. Amino acids in grey are found to be conserved in the IS200 family [48]. The beginning and the end of the alignment are not displayed due to space restrictions and do not contain any conserved motifs. Protein sequences were extracted from the following genomes: YafM SBW25 from P. fluorescens SBW25 [100], PSPTO_0262 from P. syringae pv. tomato DC3000 [108], PputGB1_0613 from P. putida GB-1 (NC_010322.1), PSEEN4846 from P. entomophila L48 [109], Tgr7_2777 from Thioalkalivibrio sp HL-EbGR7 (NC_011901.1), SSPA4070 from Salmonella enterica serovar Paratyphi A AKU_12601 [115], YafM E. coli from E. coli K-12 DH10B [116], PSPA7_4226 from P. aeruginosa PA7 [112], PST_1052 from P. stutzeri A1501) [114], Pmen_3135 from P. mendocina ymp (NC_009439.1), NPUN_F5543 from Nostoc punctiforme PCC 73102 (NC_010628.1) and ISHp608 from Helicobacter pylori (AF357224.1).

conserved in both ISHp*608* and the RAYT protein family. In particular, the HUH motif (histidine, hydrophobic amino acid, histidine) and the 3′ tyrosine have been shown to be essential for ISHp*608* transposition and are conserved in all IS*200*/IS*605* family proteins [48, 143-145]. These are conserved in all RAYTs investigated.

For IS*200*/IS*605* the HUH and the tyrosine motif have been shown to be located in the active site of the dimeric protein complex. During cleavage they form a complex with a divalent metal ion, which allows the tyrosine to perform a nucleophilic attack. This subsequently results in the covalent binding of the tyrosince residue to the DNA and the later release of the DNA when ligating it into the target site [145].

In addition to the functional motifs, there are other similarities between RAYTs and IS*200* genes. The most striking similarity with regard to REPIN mobility is that IS*200* genes are flanked by two short palindromic recognition sequences and REPINs consist of two inverted short palindromes. For ISHp*608*, it has been shown that the palindromes flanking the transposase are recognized and bound by the transposase, and as such are essential for both the excision and the insertion processes during transposition [48]. The high conservation level of the two palindromes contained within a REPIN suggests they carry an equally important transposition function.

Another interesting parallel between IS*200* and RAYTs is the asymmetry of the transposition intermediate (sequence excised from the genome). For ISHp*608* it has been shown that the distance from the 5′ end to the 5′ palindrome of the intermediate is 20 bp compared to 10 bp for the distance from the 3′ palindrome to the 3′ end [145]. Intriguingly, the putative transposition intermediate that was identified for REPINs has similar characteristics. The distance from the 5′ end to the 5′ palindrome is 7 bp compared to a distance of 0 bp from the 3′ palindrome to the 3′ end (see Figure 3.8). As for the majority of insertion sequences, the insertion of ISHp*608* sequences into novel DNA is targeted. ISHp*608* sequences contain a short (4 bp) sequence that pairs with the target during insertion [145]. Such base pairing is possible because the transposition process occurs *via* a single stranded transposition intermediate. If REPINs are also transposed in single stranded form then it is likely that the intermediate forms a long hairpin structure with a short, single stranded tail (see Figure 3.8). This tail could guide the intermediate to a target DNA motif complementary to the tail sequence. In contrast to other insertion sequences, ISHp*608* transposition does not result in target site deletion

or insertion [48]. This may also be the case for REPINs, for which target site deletions or insertions are not apparent in the consensus sequence. A further characteristic of IS*200* transposition is that only the top strand of the ISHp*608* DNA has been shown to transpose [145]. In this case sequence analysis provides little insight into whether the top or bottom strand of the REPIN is preferentially transposed, although due to almost perfect symmetry between the top and bottom strands, one might expect that both strands are equally likely to be transposed.

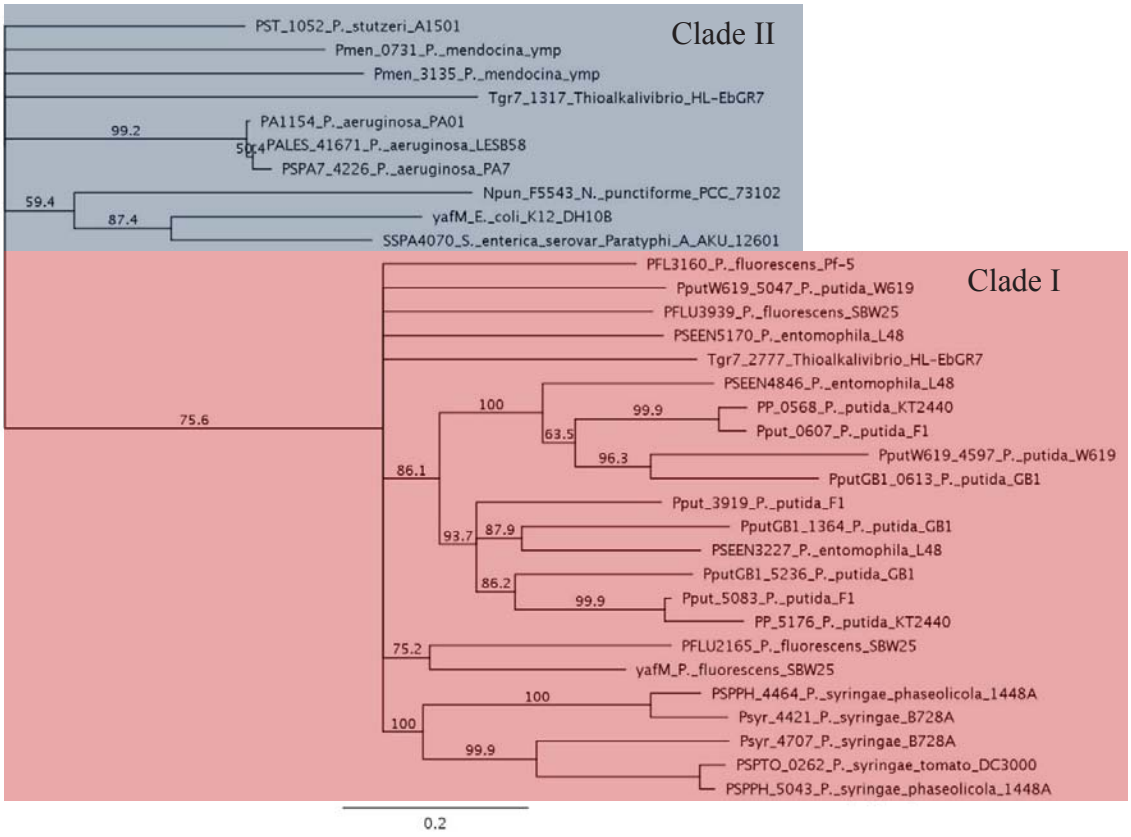


Figure 4.3. RAYT bootstrap neighbour joining tree. Two distinct phylogenetic groups are formed shown as Clade I and Clade II. The tree is based on a translated nucleotide alignment. The first part of the branch tip description denotes the gene name and the second part the name of the host organism. Alignment was performed with ClustalW2 in Genious [119, 120]. The tree was resampled 1000 times.

4.2.3 Association between RAYTs and REPINs in other genomes

If the hypothesis that RAYTs are responsible for REPIN dissemination is true, then RAYTs should also be associated with REPs or REPINs in other bacterial genomes. Hence, RAYTs were identified through BLAST searches and selected from 18 different bacterial strains including all fully sequenced *Pseudomonas* genomes, the genomes of *E. coli* K-12 DH10B and *Salmonella enterica* serovar Paratyphi A AKU 12601 (both

chosen because of their significance for REP research) and the genomes of *Thioalkalivibrio* HL-EbGR7 and *Nostoc punctiforme* PCC73102 (chosen due to their distant relationship to *Pseudomonas*). A phylogenetic analysis of the RAYTs was firstly undertaken (Figure 4.3). Notably, RAYTs from these strains form two distinct evolutionary lineages (clade I and II) with evidence of multiple independent introductions. For example, the genus *Pseudomonas* is separated into two sets of species defined by the presence of either 'clade I' or 'clade II' RAYTs. The genome of *Thioalkalivibrio* contains one clade I and one clade II RAYT. Several other genomes, in addition to SBW25, contain more than a single RAYT, but these almost never form phylogenetic clusters within strains; instead phylogenetic clusters are frequently found at species level, indicating ancient gene duplication events following vertical gene transfer and speciation. Together, the distribution of RAYTs is consistent with vertical transmission and rare incidents of lateral gene transfer.

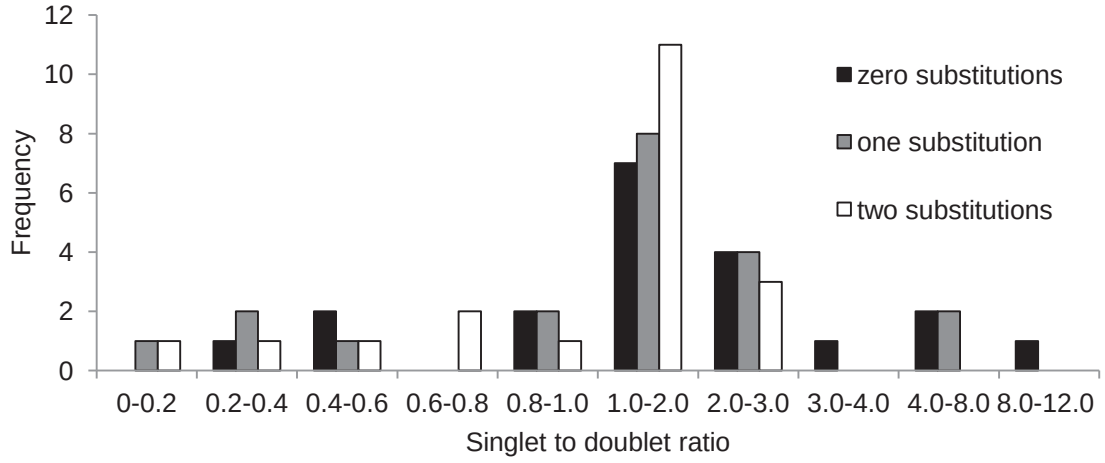


Figure 4.4. REP singlet to doublet ratios for REP sequences from bacterial genomes. Data are the most abundant 16-mers found within the flanking non-coding DNA of 20 RAYT genes from ten different genomes. In order to include related 16-mers, a set of degenerate sequences was produced by allowing up to two substitutions per 16-mer.

To test the association of RAYT genes with REPs or REPINs (identified both in this work and [101]), the non-coding DNA flanking each of the 30 RAYTs found in 18 different genomes was interrogated for 16-mers that were repetitive, extragenic and palindromic (*i.e.* are REPs). In each instance a REP was identified (Table A2.1). Subsequently, the hypothesis that REPs are organized as REPINs in order to be disseminated was tested. In order to form a REPIN, two REPs are required to be arranged as an inverted repeat. Such organisation will be apparent in the distribution of

REPs across the genome. Thus, the distribution of all REPs (allowing for up to two polymorphisms) was analysed for each genome as described in sections 2.2.6 and 3.2.3.1 (*i.e.* by measuring the distance between REP sequences (16-mers) the number of each higher order arrangement (singlet, doublet, triplet *etc.*) could be determined). Results were expressed as the ratio of REP singlets to doublets, where ratios greater than two indicate that REPs occur predominantly as singlets and ratios less than two mean that REPs occur predominantly as doublets. Figure 4.4 shows a histogram of singlet to doublet ratios for REP sequences associated with clade I RAYTs. Of the 20 REP sequence types (*i.e.* specific REP sequence groups that are associated with 20 clade I RAYTs from ten different genomes; one associated with each RAYT; some genomes contain more than one RAYT e.g., SBW25), 17 gave singlet to doublet ratios of less than two, indicating that most REPs occur as doublets. The majority of doublets contained REPs as inverted pairs (Table A2.2) as would be expected for REPINs.

A similar investigation for REP sequences associated with clade II RAYTs did not return conclusive results, which is probably due to different structures formed by clade II REPINs in comparison to clade I REPINs. Since clade I REPINs presumably co-evolve with clade I RAYTs and clade II REPINs with clade II RAYTs, different REPIN structures between the two clades are not surprising considering the distant relationship between clade I and clade II RAYTs (Figure 4.3). Hence, for each clade II RAYT an associated REP sequence candidate was manually tested for the formation of REPIN like structures. In all instances the general REPIN composition was found to hold (two inverted REP sequences separated by a short stretch of DNA and forming a hairpin, Figure 3.8), with the exception of REP sequences found in *P. stutzeri*: interestingly no REPINs were identified in this genome.

Higher order arrangements for REP sequences associated with clade I RAYTs were also analyzed, but these were not present in all genomes. Such higher order arrangements were predominantly found in *P. syringae* and *P. fluorescens*, although two such REP sequence classes were also detected in *P. putida* (Table A2.2). No correlation was found between the singlet to doublet ratio and cluster formation.

Taken together, the systematic cluster analysis of clade I REP sequences and secondary structure prediction of a selection of clade II REP sequences suggest that the organization of REP sequences into REPINs is a necessary condition for REP sequence

distribution and supports the hypothesis that RAYTs are a causative agent for REPIN dispersal.

4.3 Discussion

4.3.1 Overview of the discovery of REPIN-RAYT systems in SBW25

The REPIN-RAYT system in SBW25 was discovered due to the co-localization of specific REPIN clusters to specific RAYT (then conserved hypothetical) genes (section 4.2.1). Subsequently, the connection between RAYTs and the IS*200*/IS*605* family of transposases was realized when BLASTP searching the protein sequences against an insertion sequence database (www-is.biotoul.fr). Although the sequence hits were not highly significant, most hits obtained were to insertion sequences of the IS*200*/IS*605* family. Closer inspection showed that the reason for the repeated identification of members of the IS*200*/IS*605* family was a short motif conserved in both the IS200/IS605 and RAYT families. The realization of this distant relationship sparked a closer analysis of similarities between IS*200*/IS*605* and the REPIN-RAYT system.

4.3.2 Summary of the similarities between the REPIN-RAYT system and IS*200*/IS*605* insertion sequences

As discovered in section 4.2.2, a plethora of parallels exist between the IS*200*/IS*605* family and the REPIN-RAYT system. The most striking conserved features are: (1) the functional HUH and 3′ tyrosine motifs, (2) palindromic recognition sites in flanking sequences, and (3) the asymmetric transposition intermediate that was analysed in great detail for ISHp*608* [48, 143-145] and the asymmetric putative transposition intermediate identified for REPINs in Figure 3.8. Based on these parallels, the hypothesis that REPINs are dispersed by RAYTs was formulated. This hypothesis was tested by analysing higher order arrangements of REPs associated to RAYTs in 18 bacterial genomes.

4.3.3 Analysis of higher order arrangements of REPs in different bacterial genomes

Higher order arrangements of REPs were analysed in a selection of 18 RAYT-containing bacterial genomes (section 4.2.3). These studies included the analysis of the

immediate non-coding DNA flanking all RAYT genes identified (through TBLASTN searches) in each of the 18 genomes (a total of 30 RAYTs) for over-represented 16-mers. Interestingly, in each case a 16-mer was identified that was not only over-represented in its host genome, but also palindromic and almost exclusively extragenic (Table A2.1).

However, the most abundant 16-mer identified flanking the RAYT in *Pseudomonas stutzeri* was present only four times, with a *P*-Value of 0.0029 (proportion of 16-mers in the *P. stutzeri* genome that occur four or more times), and thus only just met the criteria to be considered over-represented (this 16-mer was also palindromic and extragenic). Interestingly, a secondary structure analysis showed that no REPINs are formed by these four borderline significant sequences in *P. stutzeri*. Together with the fact that the 16-mers (REPs) were found exclusively in the extragenic spaces immediately flanking the RAYT gene, this finding further supports the hypothesis that REPIN formation is a prerequisite for REP dispersal (REP singlets are immobile see section 3.2.3.2).

Aside from the REPs identified in *P. stutzeri,* all REP sequences identified in other bacterial genomes have been shown to form REPINs either through a systematic analysis (REPs associated to clade I RAYTs, Table A2.2) or through manual secondary structure predictions (REPs associated to clade II RAYTs, Figure A2.1). This finding greatly bolsters the conjecture that REPINs are a unit of selection and that RAYTs are the causative agent for REPIN dispersal. In addition, the apparently general nature of the association between REPINs and RAYTs, combined with substantial diversity among the elements themselves, suggests that the diversity of REPINs (REPs) and RAYTs is a consequence of longstanding co-evolution between RAYTs and their respective REPINs.

4.3.4 Concluding comments

While the case for REPINs as widely distributed replicative entities is strong, there remains much to be discovered, particularly regarding the mechanism of transposition and the relationship between REPINs and RAYTs. A further unknown is the origin of the REPINs themselves. One possibility is that REPINs are derived from the imperfect palindromic (REP) sequences flanking an ancestral IS*200*-like element in a manner

analogous to the evolution of MITEs and other non-autonomous elements [21], but with a twist. Whereas MITEs can exploit the transposase of extant transposons, the transposons they parasitize remain capable of autonomous replication. Conversely, RAYTs appear to be incapable of self-mobilization and exist as single copy entities: in those genomes harbouring more than a single RAYT each RAYT is distinctive and present as a single copy. This suggests that REPINs evolved a means of parasitizing an IS*200*-like ancestor that not only caused divergence of RAYTs from an IS*200*-like precursor, but did so in such a way as to enslave the RAYTs.

Chapter 5:
Evolutionary characterization of RAYTs, a novel class of REP and REPIN-associated genes

5.1 Introduction

Introduced in Chapter 4, REP-associated tyrosine transposases (RAYTs [101]), are putative transposases that are found in association with REPINs in a wide range of bacterial genomes. Given that RAYTs share a number of key characteristics with IS*200* transposases, they are likely themselves to encode transposases. As such, RAYTs are the probable cause of REPIN dispersal within the genome (discussed below in section 5.1.1). Since little is known about the RAYT family, the work in this chapter concentrates on the systematic characterization of RAYT distribution, phylogeny and sequence and consequently tries to address questions about the evolutionary history of RAYTs. Each of these objectives is discussed in more detail in the following sections.

5.1.1 Molecular characteristics of RAYTs and IS*200* transposases

The RAYT family shares some essential features with the IS*200* transposase family. The transposition of ISHp*608*, a member of the IS*200* family (see section 5.1.2.1.1), has been studied extensively, and the mechanism of transposition has been determined in detail [48, 143-146]. Two amino acid motifs are essential for ISHp*608* transposition: (1) the HUH (histidine, hydrophobic amino acid, histidine) motif at position 64, and (2) a 3' tyrosine residue at position 127 (which makes it a tyrosine transposase) [48, 143-145]. Both of these motifs are also found in almost all RAYT and IS*200* proteins identified to date [48, 101]. It is possible that the few RAYTs and IS*200* proteins that do not contain these conserved motifs are inactive as a consequence.

Genes encoding IS*200* transposases are flanked on either side by short (~20 bp) palindromes, each of which has to be recognized and bound by an IS*200* transposase

before transposition is possible [143]. The transposase-palindrome association as well as the dimerization of the two palindrome bound transposases is required for both the excision of the IS*200* sequence and its subsequent insertion elsewhere in the genome [48, 144].

Similar to IS*200* sequences, genes encoding RAYTs are usually flanked by REPINs, which consist of two palindromic REP sequences (see section 3.2.3). In section 4.2.3, the co-localization of RAYTs and REPINs was shown to hold in a selection of 18 different bacterial genomes, each of which contained up to three different RAYT genes. However, in contrast to IS*200* sequences and REPINs, RAYTs appear to be transposed only rarely: no RAYT duplicates, a feature of recent transposition events, were found in the genomes surveyed in Chapter 4 (section 4.2.3). Presumably, the putative transposase encoded by a RAYT gene recognizes the palindromes within REPINs and hence is able to transpose REPINs. It is possible that while evolving the propensity to transpose REPINs, RAYTs lost the ability to transpose themselves.

Hence, the transposition of REPINs is an *in trans* activity. This means that RAYTs transpose sequences (REPINs) that do not encode the transposase. Conversely, IS*200* and other insertion sequences predominantly have an *in cis* activity, meaning they transpose the gene from which the transposase was expressed.

While there are many parallels between IS*200* sequences and RAYTs, the amino acid conservation between IS*200* transposases and RAYTs is confined to only six residues (among which are the essential HUH and the Y motif), which are dispersed over the protein (see Figure 4.2 and section 4.2.2). Given this limited but marked pattern of conservation, the evolutionary origins of the two families remain unclear; either the conserved motifs evolved once in a common ancestor and the families subsequently diverged, or the shared sequence motifs arose multiple times in evolutionary distinct lineages (*i.e.* by convergent evolution). One of the aims of the work in this chapter is to provide insight into which of these scenarios is more likely. This will be achieved through further genomic and phylogenetic analysis of the RAYT gene family.

5.1.2 Genomic distribution of housekeeping genes versus insertion sequences

There are a number of characteristics that differentiate housekeeping genes from insertion sequences. One of these characteristics is the genomic distribution of a gene family. The genomic distribution can be defined as the position and frequency of genes on different genomes or plasmids. Typical characteristics that describe the genomic distribution of a gene family are for example duplication rate, gene frequency on plasmids or taxonomic distribution. Such characteristics are likely to be different between insertion sequences and housekeeping genes (essential genes). This expectation is derived from the different "life styles" of insertion sequences and housekeeping genes. Insertion sequences are selfish genetic elements that avoid host selection by moving horizontally (one genome to the next) within the gene pool [18, 147]. This leads to a characteristic genomic distribution where insertion sequences are frequently found as duplicates or on plasmids. In contrast, housekeeping genes are predominantly transferred vertically (one generation to the next); hence they are rarely found on plasmids or as duplicates.

5.1.2.1 Comparisons between different gene families

A central issue of the study of RAYTs is their function. Evidence to date is controversial. Sequence studies show that their composition and conserved sequence motifs are typical for IS*200* sequences (section 4.2.2). However, RAYT transposition seems to be rare. A systematic analysis of their genomic distribution compared to that of typical insertion sequences and housekeeping genes may shed light on this issue. Hence, the genomic distribution of RAYTs will be analysed and subsequently compared to that of three separate gene families: (1) IS*200* insertion sequences, (2) IS*110* insertion sequences, and (3) *def*, a housekeeping gene encoding a peptide deformylase. Each of these is discussed in more detail below.

*5.1.2.1.1 IS*200 *sequences*

Insertion sequences of the IS*200* family were first discovered by Roth and Lam in 1983 [148]. While characterizing loss-of-function mutants in the histidine operon, these authors found the insertion of an IS*200* sequence in the open reading frame of *hisD*.

According to Beuzon *et al*., the next 20 years brought only two further reports of loss of function phenotypes resulting from IS*200* insertions – even assays specifically designed to capture jumping IS*200* sequences were unfruitful [149]. Then in 1998, Kersulyte et al. [150] discovered a new insertion sequence that they named IS*605*. IS*605* is a chimera of two open reading frames, one encoding a gene named *tnpA* (similar to the ORF encoded in IS*200* sequences) and the other named *tnpB* (similar to the ORF encoded in IS*1341*). The same group later showed that the transposition activity of ISHp*608* (an IS*605* family member) is dependent only on the presence of the IS*200*-like *tnpA*; deletion of *tnpB* did not affect transposition, while deletion of *tnpA* abolished transposition [146]. Further investigation of ISHp*608* led to detailed elucidation of the transposition mechanism that by logical extension, is also thought to apply to IS*200* (see section 5.1.1).

5.1.2.1.2 IS110 insertion sequences

In addition to comparing the distribution of RAYTs among bacterial genomes to that of IS*200* sequences a member of the tyrosine transposase family; it is also desirable to perform a comparison with a member of the most abundant class of insertion sequences: DDE transposases (named after the enzymatically active amino acids aspartate, aspartate and glutamate) [18, 151]. Members of almost all insertion sequence families contain an active DDE transposase; with notable exceptions being the IS*200*, IS*91*, IS*607* and IS*1595* families (see www-is.biotoul.fr). Hence, there is an abundance of different DDE insertion sequence families to choose from. The IS*110* family was chosen, as members of this family are present in both *P. fluorescens* (the organism in which the initial REPIN analysis was undertaken) as well as *E. coli* (a popular model organism).

5.1.2.1.3 def, the housekeeping gene family of peptide deformylases

Finally, by way of control, the distribution of RAYTs will be compared to a housekeeping gene family. For this purpose the *def* gene family was chosen. The *def* gene encodes a peptide deformylase, the enzyme that removes the formyl group from the N-terminal methionine of a protein after translation. The *def* gene was chosen for a number of reasons. Firstly it is present in both *P. fluorescens* and *E. coli*, the two most important model organisms for RAYT and REP/REPIN studies. Secondly, the length of

the Def protein (169 residues) is similar to that of IS*200* and RAYT proteins (typically between 140 and 200 residues). Thirdly, *def* is a true housekeeping gene in that it is essential for survival; deletion from the *E. coli* genome is lethal [152]. Due to the essential nature and wide distribution of *def*, inhibitors of peptide deformylases have been developed for use as antibiotics [153, 154]. This underlines the usability of *def* as a representative housekeeping gene.

5.1.2.2 Genomic distribution characteristics

Characteristics that are expected to differ between housekeeping genes and insertion sequences will be introduced in the following sections.

5.1.2.2.1 Family size at varying levels of relationships as indicator for gene conservation

Characterizing a gene family is not trivial for a number of reasons. A major hurdle is to sensibly define genes that are part of a family and those that are not. Naively, one could say that all genes that share a common ancestor are part of the same gene family. However, this definition may not be very useful since presumably all known genes evolved from a single common ancestor or at least a very small group of ancestral genes [155]; hence a more restrictive definition is needed. Alternatively, and more usefully, one could sort all genes into families based on functionality, where genes of the same family also share a common ancestor not shared by genes with other functions. Since determining the function of a gene is inherently difficult and itself requires a significant amount of experimental work, it is generally assumed that two proteins that share a similar amino acid sequence also share a similar function. Since this assumption does not always hold [156], structural comparisons are also used to classify proteins [157]. However, determining the structure of a protein experimentally is again a time consuming effort, and so a multitude of structure prediction and comparisons have been developed with the aim of making more accurate predictions regarding gene functionality [158]. In order to be able to comprehensively analyse all known members of a certain gene family, here gene families are solely defined as related genes that are found through BLAST searches. This means that in the first instance gene families are defined to contain sequences that share certain similarities. Although in later sections,

RAYT subfamilies are also analysed for characteristics pertinent to genomic distribution as introduced in sections 5.1.2.2.2 to 5.1.2.2.6.

Gene family size at varying levels of relationship can also provide clues about conservation and sequence diversity, which is likely to be different between insertion sequences and housekeeping genes. The conservation of a gene family can be determined by the proportion of family members found at specific sequence similarity thresholds (or are identified for a particular e-Value threshold). This is because for strongly conserved sequence families a great proportion of the gene family is found at high levels of sequence similarity. In contrast, for sequence families that are weakly conserved a great proportion of the sequence family is found at lower sequence similarities. Generally there are three different processes that cause this distribution: negative (purifying) selection, positive selection and genetic drift [159]. Negative selection is evident when the ratio of synonymous changes (no change in the amino acid sequence) to non-synonymous changes (change in amino acid sequence) is less than expected by chance. This leads to high conservation of the protein sequences, which means that one expects to find genes with similar amino acid sequence in relatively distantly related genomes. Housekeeping genes that provide an important cellular function are expected to be under negative selection and hence show low levels of diversity [160, 161]. The opposite of negative selection is positive selection. Positive selection occurs when the ratio between synonymous and non-synonymous mutations is greater than expected by chance. Neutral evolution can also lead to changes in the amino acid sequence, if the change does not have an impact on the gene's fitness. Therefore genes whose evolution is predominantly governed by weak negative selection as well as drift and positive selection show dissimilar amino acid sequences even in relatively closely related genomes. Insertion sequences are an example for such a class of genes since they generally are not conserved to provide an important cellular function [162]. Since TBLASTN searches (protein query against nucleotide database) identify sequences with similar (conserved) amino acid sequence the family size at different e-Values (different levels of similarity) can provide information about the impact of negative selection on the evolution of the different gene families. For example, for a highly conserved gene family most members would be identified at very low e-Values and the family size would grow very little with increasing e-Values. In contrast, for a

highly diverse gene family (high levels of positive selection or drift), only few family members would be identified at low e-Values, but the family size would increase quickly for higher e-Values.

5.1.2.2.2 Duplication rate

The number of copies of a given gene can provide valuable information. For example, copy number can help to distinguish between insertion sequences and housekeeping genes.

One would expect the copy number to be low for housekeeping genes (except for rRNA or tRNA genes), as their persistence depends on fulfilling a host function, which is usually dosage sensitive. Dosage sensitivity means that additional copies can have toxic effects [163] since they affect the relative abundance of the protein product within the cell (which is presumably optimal under normal conditions and tightly regulated on various different levels). However, under specific circumstances the amplification of housekeeping genes can be favoured by selection in order to cope with specific environmental stress [164, 165]. Such amplifications are usually quickly retracted and are not conserved, except in cases where the copies diversify and acquire new functions [8, 166, 167].

In contrast, insertion sequences are present in higher copy numbers per genome. Their persistence within the gene pool relies on frequent duplication events [37, 168]. This means the gene does not increase in frequency within the gene pool as a result of the beneficial effects it provides for the organism (although in some cases insertion sequences have shown to be beneficial for the host [17]), but due to its capability to move within and between genomes.

5.1.2.2.3 Number of homologous genes per replicon

The number of homologous genes (genes of the same family) found per replicon (either plasmid or chromosome) is similar to the proportion of duplicates per replicon since every duplicate is also a homologous gene (but not every homologous gene is a duplicate). However, instead of only capturing very recent duplications or invasions the number of homologous genes also includes events that occurred in the more distant past, or invasions by more distantly related members of the same gene family. Similar to

duplication events, multiple homologous genes per replicon are expected for insertion sequences [168]. In contrast, for housekeeping genes fewer homologous genes are expected to be present per replicon. The divide between the two classes is unlikely to be as great as for duplication events, since diversification can lead to the acquisition of new beneficial functions for housekeeping genes and hence to multiple homologues per chromosome [6].

5.1.2.2.4 Gene occurrences on plasmids

Whether genes of a certain family can be found on plasmids can provide information about the nature of the gene family. Since plasmids greatly facilitate horizontal gene transfer [169], it is more likely to find genes on plasmids that rely on horizontal transfer for their persistence within the gene pool (*e.g.* selfish genetic elements) [147, 170]. For example, insertion sequences are frequently found on plasmids [18], whereas housekeeping genes are very rarely found on plasmids [65, 171]. Hence, the rate at which genes of a certain family are found on plasmids can allow predictions concerning the degree of selfishness of the gene family.

5.1.2.2.5 Distribution over taxonomic classes

The distribution of a gene family over a wide range of taxonomic classes can be the result of vertical transmission prior to the taxonomic division, lateral transfer or a combination of the two. Ancient housekeeping genes are likely to be present in almost all bacterial classes. Highly successful insertion sequences could also be present in a wide range of taxonomic classes. Hence, distinguishing between highly successful selfish genetic elements and highly conserved housekeeping genes on a taxonomic level may be problematic. Nevertheless, a taxonomic analysis provides a useful overview of the taxonomic distribution of gene families.

5.1.2.2.6 Frequency of most abundant short sequences in flanking non-coding DNA

Interestingly, the frequency of the most abundant short sequences (within the respective genome) found in the extragenic space immediately flanking a gene is also a characteristic that distinguishes housekeeping genes from insertion sequences. This is mainly due to two reasons. Firstly, insertion sequences consist of a transposase gene and two flanking inverted repeats, which are located in extragenic space [18]. The

repetitiveness of the flanking repeats increases the average within-genome frequency of the most abundant flanking short sequence. Secondly, duplication of insertion sequences not only affects the gene but also the two flanking inverted repeats, which further increases within-genome frequency of the flanking short sequences. In contrast, housekeeping genes are not flanked by over-represented short sequences. Hence the frequency of short sequences flanking housekeeping genes should not be higher than what is expected by chance.

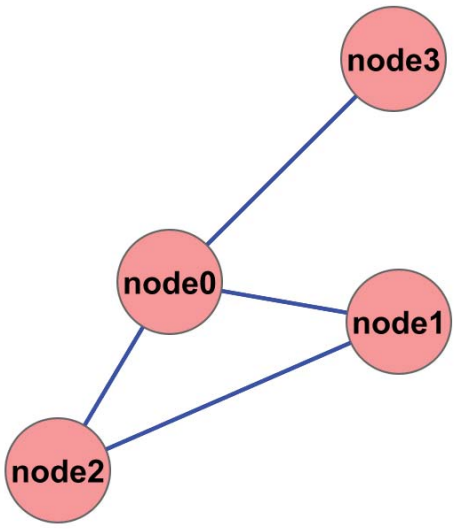


Figure 5.1. Example of a small phylogenetic map. In this example the proteins 0 to 2 that are represented by nodes 0 to 2 all share a pairwise identity greater than a certain defined threshold, indicated by the edges connecting the nodes. The protein represented by node 3 is only related to protein 0 and does not share a significant sequence similarity with the other proteins.

5.1.3 Phylogenetic methodology

Relationships between proteins are classically analysed by building multiple sequence alignments and based on these alignments, the construction of phylogenetic trees. Currently the most popular methods to construct phylogenetic trees are bayesian [172, 173] and maximum likelihood methods [174]. However, distance based methods (*e.g.* neighbour-joining [121]) are also frequently applied due to their lower complexity and therefore ability to cope with larger datasets in smaller timeframes. However, for large, distantly related datasets, where the exact phylogeny of closely related proteins plays a minor role, it can be useful to display phylogenetic relationships as a network. Such approaches have been successfully applied earlier to analyse, for example the relationship between different insertion sequences [175, 176], where protein networks are generated by applying a Markov cluster algorithm [177]. Alternatively one can apply a more transparent approach where nodes are represented as proteins and the connection between nodes (edges) represent pairwise identities above a set threshold. The clustering into protein families can then be performed based solely on the connectivity of the graph. The results of such an approach are easily interpretable and parameters (*e.g.* pairwise identity threshold) can easily be manipulated.

5.1.4 Aims

The aims of this chapter are:

(1) To define the RAYT gene family and gather information about the genomic distribution of RAYTs. This will include a comparison of genomic distribution with the insertion sequence families IS*200* and IS*110*, and the housekeeping gene family *def*.

(2) To analyse the evolutionary relationship between RAYTs and IS*200* sequences on the basis of phylogenetic data.

(3) To investigate evolutionary relationships within the RAYT gene family and characterize the genomic distribution of RAYT subfamilies. Furthermore for each subfamily the consensus sequence and at least one duplication event are analysed.

5.2 Results

5.2.1 Comparison of the genomic distribution of four gene families: RAYTs, IS*200*, IS*110* and *def*

In order to determine whether RAYTs show characteristics similar to insertion sequences or housekeeping genes, their genomic distribution will be analysed and compared to two other families of insertion sequences (IS*200* and IS*110*), and one housekeeping gene family (*def*). To perform the analysis, each of the four gene families must first be defined (section 5.2.1.1). The analyses in the following sections (5.2.1.1-5.2.1.6) investigate five aspects of genomic distribution: (1) proportion of family members for which a duplicate is found in the same genome or plasmid; (2) average number of homologues per replicon (chromosome or plasmid); (3) proportion of the family members found on plasmids; (4) gene distribution over taxonomic classes; and (5) the average frequency of the most abundant 16-mers found in the extragenic space immediately flanking each gene. These properties distinguish insertion sequences from housekeeping genes and hence can provide information about how the genomic distribution of RAYTs compares to that of housekeeping genes and insertion sequences.

5.2.1.1 Defining the RAYT, IS*200*, IS*110* and *def* gene families

Before the description of a gene family is possible, the members of the family need to be identified. One method of finding related family members is to perform a BLAST (basic local alignment search tool) search. When given a query sequence (*e.g.* the sequence of a protein or gene) a BLAST search finds all sequences similar to the query sequence within the sequences of a BLAST database (*e.g.* genome(s) or plasmids). In the event of finding a similar sequence in the database, an e-Value is provided. The e-Value correlates with the probability that a similar sequence could be present in the database merely by chance: the lower the e-Value, the less likely that the perceived similarity is due to chance [102]. This raises the question of how similar the query and the database sequences need to be in order to be defined as part of the same family. To circumvent this problem, each BLAST analysis was performed at four e-Value thresholds (1e-20, 1e-14, 1e-8 and 1e-2). In general, larger e-Values are expected to

contain progressively greater sequence numbers, as they will include all members from lower e-Values plus additional search results.

It is possible that the BLAST search will identify genes that are not of the same gene family as the query protein. These are either sequences where the similarity emerged through parallel evolution or distantly related genes that assumed new functions (especially at higher e-Values). However, it is difficult to identify such genes without extensive studies of sequence function and phylogeny (which will be done for RAYTs). Hence, to simplify matters the identified genes at all e-Values will be addressed by the name of the query genes' family, although the function of some members may not be the same.

To conduct the BLAST searches two query proteins were selected from each of the gene families described above. Where possible, for each sequence family a member from *P. fluorescens* and a member from *E. coli* was selected, as *E. coli* is probably the most important bacterial model organism and *P. fluorescens* is of particular importance for REPIN and hence also RAYT research (see Chapter 3). For the RAYT gene family, YafM from both *P. fluorescens* SBW25 and *E. coli* K-12, were selected as query proteins, one of each of the two phylogenetic RAYT groups described in Chapter 4 (see Figure 4.2). For the IS*200* family, IS*609* was selected from *E. coli* O157:H7 and *tnpA* from ISHp*608* of *Helicobacter pylori* (the protein for which the transposition mechanism has been elucidated, no IS*200* member could be identified in *P. fluorescens*). For the IS*110* family, ISPfl*1* from *P. fluorescens* Pf0-1 and ISEc*32* from *E. coli* S88 plasmid pECOS88 were selected. For the *def* gene family, a member from both *P. fluorescens* SBW25 and from *E. coli* K-12 was selected.

The above query sequences were searched against all available fully sequenced bacterial genomes (09/03/2011, ftp://ftp.ncbi.nih.gov/genomes/Bacteria/, 1398 chromosomes and 1015 plasmids) using TBLASTN. This is a BLAST variation that allows a protein sequence to be searched against nucleotide databases (*e.g.* genome databases). The results are depicted in Figure 5.2, where the gene family sizes increase with increasing e-Value. Interestingly at an e-Value of 1e-2, IS*110* is the largest gene family and not the *def* family. The IS*200* and the *def* gene family are of comparable size, whereas the RAYT family is the smallest at an e-Value of 1e-2.

A

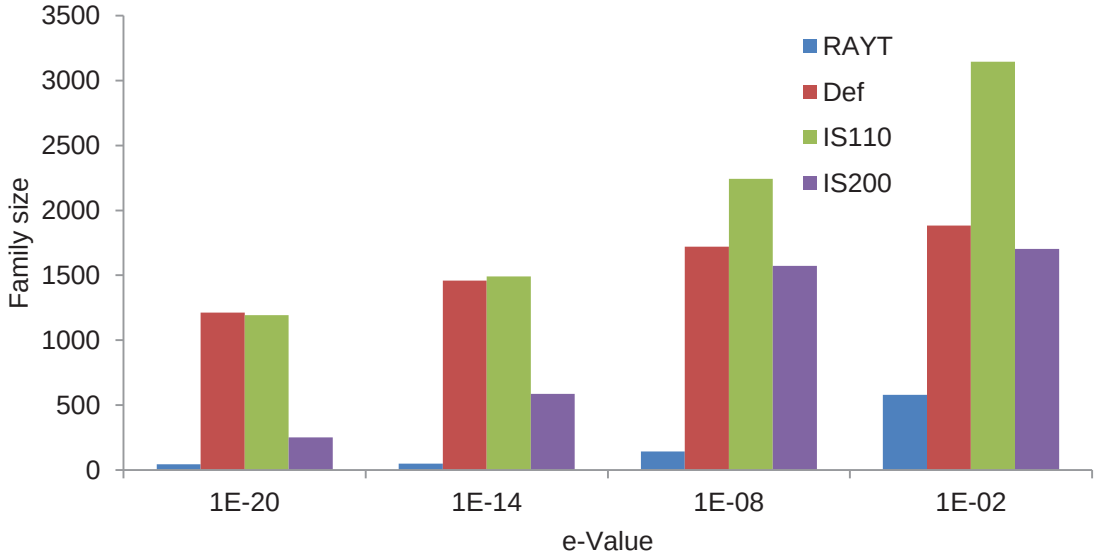


Figure 5.2. Gene family size of IS*200*, IS*110*, RAYTs and peptide deformylases (*def*). The gene family sizes were determined through BLAST searches by selection of two query proteins for each family. These were searched against all fully sequenced bacterial genomes available. The individual gene family sizes are the numbers of all genes that were identified below the e-Values indicated on the x-axis.

As predicted (see section 5.1.2.2.1) the *def* gene family is highly conserved. This is evident from the gene family sizes determined at different e-Values. For the least stringent e-Value of 1e-2 (which includes very distantly related *def* genes) a total of 1883 *def* genes were identified. At the most stringent e-Value (1e-20, at which only relatively closely related *def* genes are identified), 1212 *def* genes were identified. Thus, at an e-Value of 1e-20 64.4% of all *def* genes were identified. This proportion is large compared to the remaining gene families: for IS*200* and IS*110*, 14.7% and 37.9% of all IS*200* and IS*110* genes identified at 1e-2 were identified at 1e-20, respectively.

Surprisingly, only 7.6% of all RAYTs identified at 1e-2 (least stringent) were identified at 1e-20 (most stringent), which indicates great sequence diversity. The causes for the observed sequence diversity will be investigated more closely in section 5.2.3.

5.2.1.2 Analysis 1: investigation of gene duplication events within each family

Gene duplications are an important characteristic that distinguishes housekeeping genes from insertion sequences. Here duplicated genes are genes for which a second gene can be found on the same chromosome or plasmid with a pairwise nucleotide sequence identity of greater than 95%.

The results from the gene duplication analyses (Figure 5.3) are in accordance with the expectations above (see section 5.1.2.2.2). At an e-Value of 1e-2, for almost 70% of all IS*200* and IS*110* genes a duplicate is found within the same genome or plasmid. For the housekeeping gene *def*, in contrast, no duplicates could be identified. Interestingly, duplicates are found for only about 4% to 7% of all RAYTs. The proportion of RAYT duplicates is highest for the highest e-Value (*i.e.* when including genes that are very distantly related to the two query proteins). Thus, the proportion of RAYT duplicates is much smaller than that observed for IS*200* and IS*110*, but higher than for the housekeeping gene family *def*.

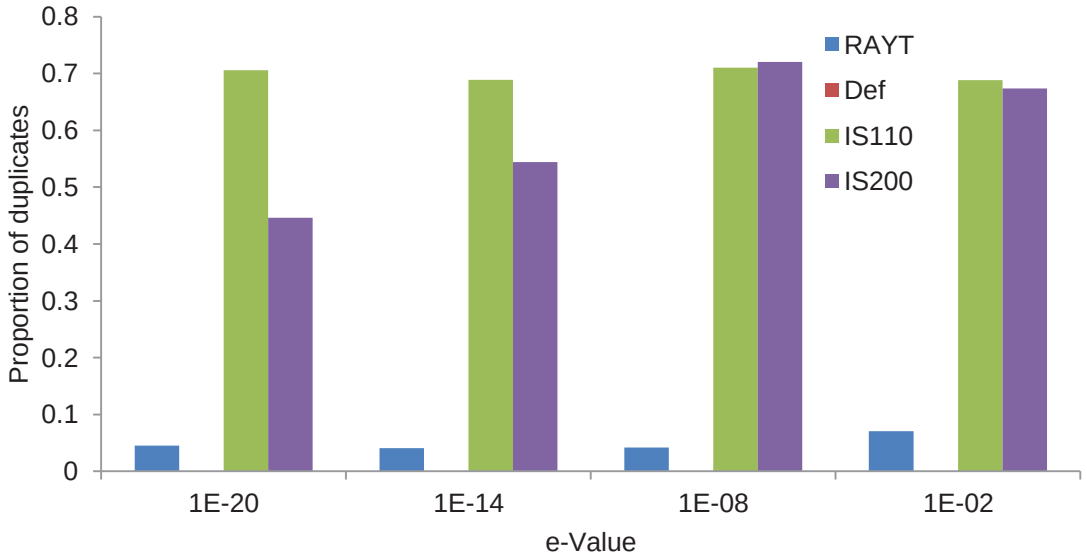


Figure 5.3. Proportion of gene family members for which duplicates were identified. Duplicates are genes that share more than 95% nucleotide sequence identity with a gene that is found on the same plasmid or chromosome. The data is gene family specific. The gene family is defined by the number of genes that were found below a certain e-Value threshold indicated on the x-axis. No duplicates were found for *def* genes.

5.2.1.3 Analysis 2: Investigation of the number of homologues present per chromosome or plasmid (replicon)

In line with the prediction (see section 5.1.2.2.3), the average number of homologues per replicon is highest for IS*110* genes (4.2 – 5.1) and IS*200* genes (2.1 – 4.6) (Figure 5.4). The differences between the two gene families are insignificant for higher e-Values (1e-8 and 1e-2) and therefore greater family sizes. The differences between the IS*200*/IS*110* and *def*/RAYT are significant for all observed e-Values. As expected the *def* gene family shows the lowest numbers of average homologues per replicon (1.3 – 1.6). Similar to the results for the proportion of duplicates, there are more homologous

RAYTs (1.5 – 2.1) found per chromosome or plasmid than was observed for *def* genes. However, the differences between the values for RAYT and *def* families are insignificant for e-Values below 1e-2. Only at an e-Value of 1e-2 there is a significant difference between RAYTs and *def* genes in the average number of homologues identified per replicon.

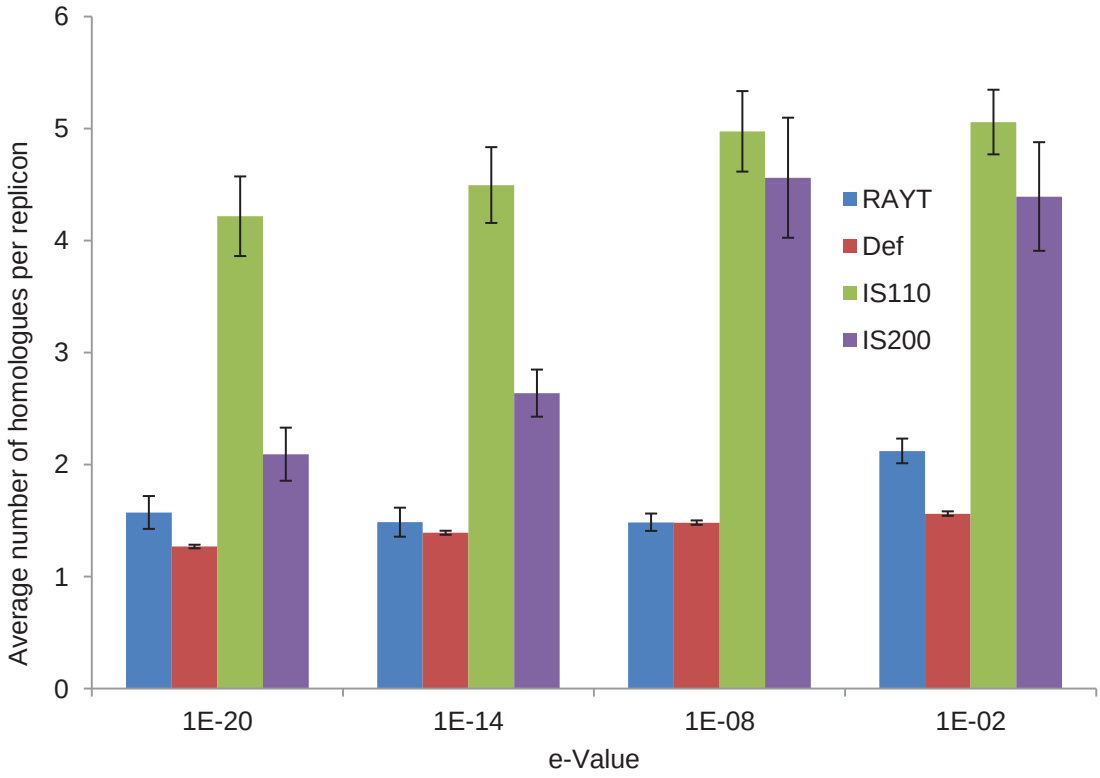


Figure 5.4. Average number of homologous genes per chromosome or plasmid (replicon). This data is the number of identified genes per gene family divided by the number of plasmids/chromosomes they were found on. All data is specific for the number of genes identified below a certain e-Value (x-axis). Error bars show standard error. The *P*-values for the gene family pairs are (pairs not shown have a *P*-value of <1e-5): **e-Value 1e-20:** RAYT def: 0.01398; IS*200* RAYT: 0.00491; IS*200* def:<1e-5; IS*110* IS*200*: <1e-5; **e-Value 1e-14:** RAYT def: 0.03189; IS*200* RAYT: <1e-5; **e-Value 1e-8:** RAYT def: 0.48397; IS*200* IS*110*: 0.21339; **e-Value 1e-2:** RAYT def: <1e-5; IS*200* IS*110*: 0.09115.

5.2.1.4 Analysis 3: Investigation of the presence of gene family members on plasmids

Since both genomes and plasmids were searched for homologous genes it was easy to determine whether a gene was identified on a plasmid or a bacterial genome. Analyses show that between 14.5% and 20.2% of all identified IS*200* genes and between 22% and 29% of all IS*110* genes are found on plasmids, whereas only 0.2% to 0.6% of all *def* genes are found on plasmids (Figure 5.5). Interestingly, the proportion of RAYT genes

found on plasmids is comparable to the number obtained for *def* genes. Only above an e-Value of 1e-8 were RAYT genes identified on plasmids with a maximum of 2.9% of all RAYT genes found at an e-Value of 1e-2.

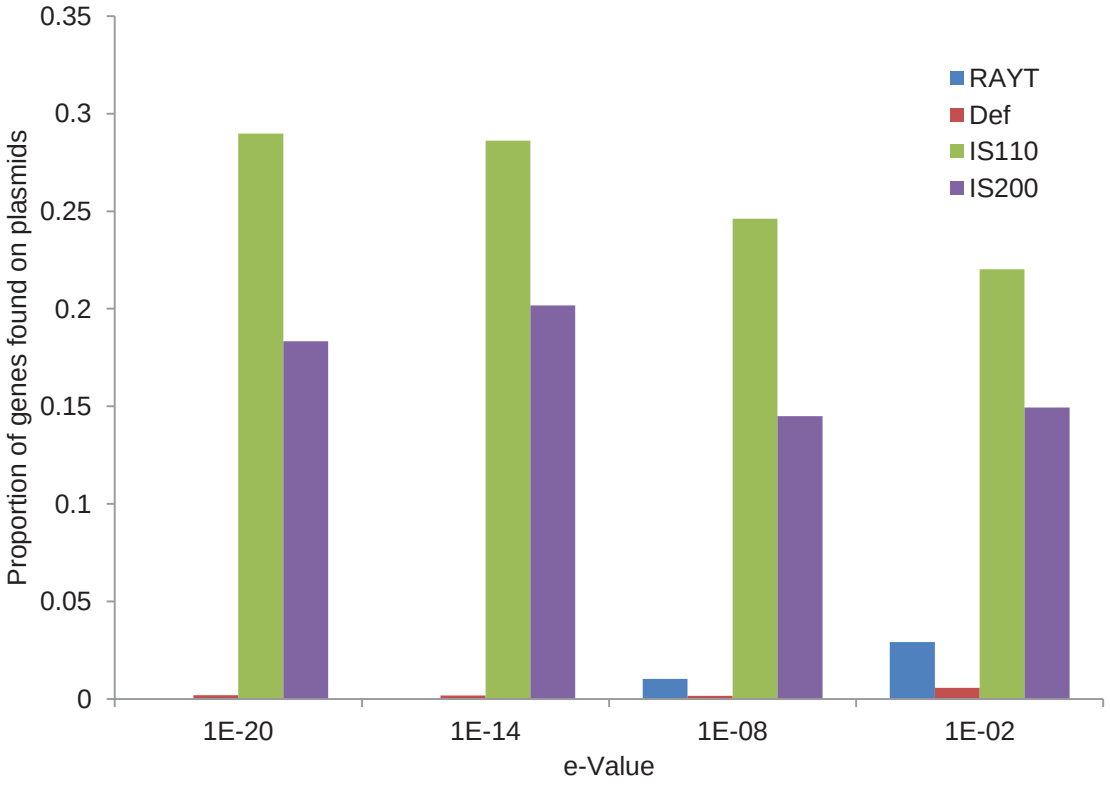


Figure 5.5. Proportion of genes found on plasmids. Query proteins from each gene family were searched against all available fully sequenced bacterial genomes and plasmids. The proportion of plasmid encoded genes below a certain e-Value (x-axis) was determined by dividing the number of genes found on plasmids by the total number of identified genes. No RAYT genes were found to be present on plasmids at 1e-20 and 1e-14.

5.2.1.5 Analysis 4: Investigation of gene distribution over bacterial taxonomic classes

The graph in Figure 5.6 shows that *def* is the most widely distributed (found in 1,199 of 1,398 (85.5%) bacterial genomes and in 48 of a total of 62 classes at an e-Value of 1e-2) of the four gene families. This indicates that the ancestor of all bacteria already possessed a *def* gene. The two insertion sequence families are also found surprisingly widely distributed (36 (IS*200*) and 35 (IS*110*) of a total of 62 classes). The RAYT family is the least widely distributed especially for lower e-Values. At an e-Value of 1e-2 the distribution of RAYTs is comparable to that of the two other insertion sequence families (found in 31 classes of a total of 62 classes).

Similar to the analyses for family size (see section 5.2.1.1) one can analyse the increase in distribution from an e-Value of 1e-20 to an e-Value of 1e-2 and express the increase as a proportion. As observed for the family size, at an e-Value of 1e-20 *def* genes are already identified in 89.6% of all taxonomic classes identified at an e-Value of 1e-2. This proportion is smaller for IS*200* and IS*110* genes with 50% and 40% respectively. Interestingly, in the case of insertion sequences the data is opposite to what was expected from the family size data (IS*110* genes show greater family size as well as greater conservation). However, this may just be an effect of the biased distribution of fully sequenced bacterial genomes among bacterial taxonomic classes. RAYTs again show the greatest increase and the smallest proportion of taxonomic classes identified at an e-Value of 1e-20 with only 12.9%.

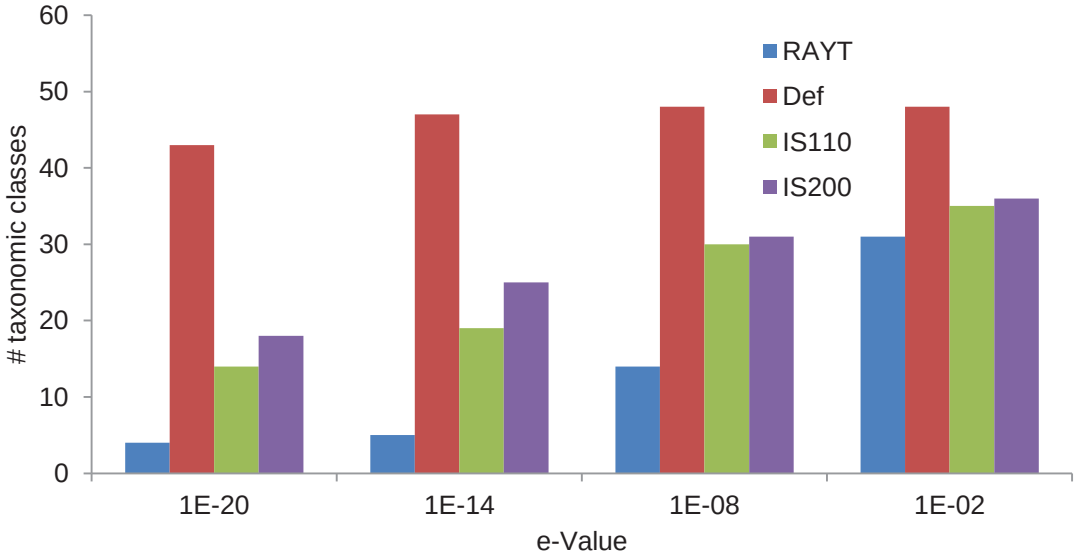


Figure 5.6. Number of bacterial taxonomic classes the individual gene family members are found in. Each genome and plasmids on which the individual genes were found was classified into a taxonomic class. The number of unique taxonomic classes was determined for each gene family below a certain e-Value (x-axis).

5.2.1.6 Analysis 5: Investigation of the DNA regions flanking members of each gene family

The analysis of the most abundant 16-mers (sequences of length 16 were chosen based on analyses performed in section 3.2.1) in the flanking extragenic space of the studied gene families allow a distinction being made between housekeeping genes and insertion sequences, but also between RAYT and non-RAYT gene families, given that typical RAYTs are flanked by highly abundant 16-mers (see section 4.2.3). This analysis also

provides insight into how 'RAYT-ness' correlates with the results from previous sections.

Figure 5.7 shows that, as expected (see section 5.1.2.2.6), *def* genes are flanked by the least abundant 16-mers (4.9 at 1e-2). IS*110* and IS*200* genes are flanked by significantly more frequent 16-mers (27.1 and 29.7 respectively). For low e-Values (closely related genes), RAYTs are associated with the most abundant 16-mers (at 1e-20 121.7), which is significantly more than what is observed for *def* (4.4), IS*200* (10.3) and IS*110* (26.6) genes. However, mirroring the results of previous sections at an e-Value of 1e-2 the average frequency of flanking 16-mers (31.6) is not significantly different from the average frequency of the most abundant 16-mers flanking IS*200* and IS*110* genes.

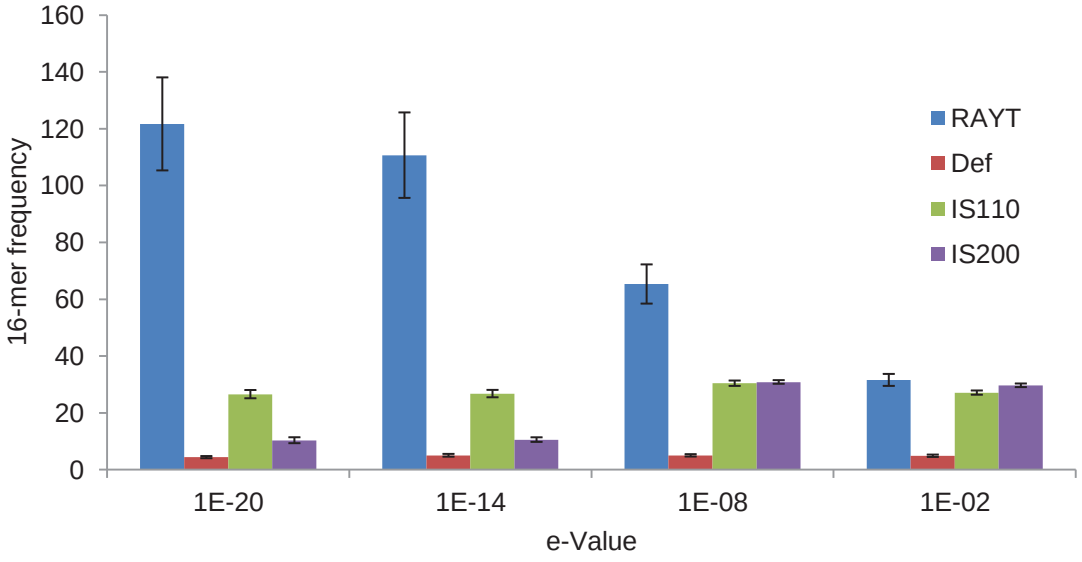


Figure 5.7. Average frequencies of the most abundant 16-mers found in the immediate extragenic space flanking the corresponding gene family. The frequencies of all 16-mers in all immediate extragenic spaces of each member of a gene family were determined. Of these frequencies the most abundant were used to calculate the average and standard errors in the above figure. The *P*-values for the gene family pairs are: **e-Value 1e-2:** IS*200* RAYT: 0.18745; IS*110* RAYT: 0.01369; IS*200* IS*110*: 4.8E-4; **e-Value 1e-8:** IS*200* IS*110*: 0.31333; IS*200*/IS*110* RAYT: <1e-5. All differences are significant (*P*-value<1e-5) for 1e-14 and 1e-20.

5.2.2 Phylogenetic comparisons between IS*200* and RAYT proteins

The limited number of shared sites between IS*200* sequences and RAYTs, in addition to the difference between the likely selfish lifestyle of IS*200* genes and the possible non-selfish lifestyle of RAYTs, raises the possibility that the two sequence classes evolved

through convergent evolution (*i.e.* independently from distinct common ancestors). To shed light on this issue, a phylogenetic analysis of the two protein families is performed in the following two sections. The analysis consists of two parts. Firstly, the pairwise identity of three members of each of the two sequence families was determined and compared to a null (random) model (section 5.2.2.1). The purpose of this is to assess whether the pairwise identity between the different RAYT and IS*200* proteins is significantly higher than what is expected under a random model. Secondly, the pairwise identity of all IS*200* and RAYT proteins is calculated and displayed as a phylogenetic map (Figure 5.1), in order to understand the relationship between the two families on a larger scale (section 5.2.2.2).

5.2.2.1 Assessment of the pairwise identity of individual RAYT and IS*200* members compared to null models

The evolutionary relationship between the RAYT and IS*200* protein families was investigated by comparing the pairwise identity of the four query proteins from the analyses in section 5.2.1 (YafM from *E. coli* K-12 and *P. fluorescens* SBW25 (RAYTs); IS*609* from *E. coli* and ISHp*608* from *H. pylori* (IS*200*)). The first two RAYT and two IS*200* proteins selected for this analysis.

In addition, the two most closely related proteins of the two families were selected. The two protein families overlap slightly at an e-Value of 1e-2. This means some proteins occur in both the IS*200* and RAYT sequence family. Hence, the most closely related proteins are proteins that occur in both families. But this does not answer the question of whether there is a significant sequence similarity between IS*200* and RAYT proteins. Hence, the most closely related proteins of the RAYT and IS*200* protein families were selected at an e-Value of 1e-8. At this e-Value there is no overlap between the two families. The two genes were identified by performing a BLAST search of the RAYT family against the IS*200* family and selecting the genes with the lowest e-Value from the search results. In the following paragraphs these will be referred to as "IS*200* 1e-8" and "RAYT 1e-8".

For all pairs of the six sequences mentioned above (three RAYT proteins and three IS*200* proteins) the pairwise identity was calculated. To determine the probability that the similarity between the proteins is greater than expected by chance, each pairwise

identity was compared to a random model. For the random model, the two protein sequences in question were shuffled 10,000 times and for each of the 10,000 shuffled protein pairs, the pairwise identity was determined. The number of times the pairwise identity of the shuffled sequence pair exceeded the pairwise identity of the two original proteins was counted and used to calculate a *P*-value. A high *P*-value indicates that the pairwise identity between the two proteins is not significant. Conversely, a low *P*-value indicates that the similarity between the two proteins is not due to chance alone. The results are presented in Table 5.1.

Table 5.1. Pairwise identities and *P*-Values for different protein pairs [a].

	IS*200* 1e-8	IS*609*	YafM SBW25	ISHp*608*	RAYT 1e-8
YafM *E. coli*	18.7 0.0241	14.6 0.4319	19.6 0.0149	12.7 0.6542	21.5 7.00E-04
IS*200* 1e-8		24.5 < 1e-4	17.8 0.0545	31.8 < 1e-4	27.4 < 1e-4
IS*609*			18.6 0.0714	21.1 3.00E-04	17.4 0.0616
YafM SBW25				13.6 0.4578	38.9 < 1e-4
ISHp*608*					20.6 0.0023

First line in each cell denotes the pairwise identity in percent. Second line shows the probability that a pair of shuffled sequences achieves a higher or equal pairwise identity. ISHp*608* and IS*609* are both from the IS*200* sequence family. Both YafM proteins are part of the RAYT family. The two most closely related members between the IS*200* and RAYT family at an e-Value of 1e-8 are RAYT 1e-8 (*cps_1489* from *Colwellia psychrerythraea* 34H) and IS200 1e-8 (*rma_1120* from *Rickettsia massiliae* MTU5). Cell in red indicates a close (significantly greater than expected by change) relationship between an IS*200* transposase and a RAYT protein.

Interestingly, there is no significant relationship between the query RAYT and IS*200* sequences. However, the pairwise identities between "IS*200* 1e-8" and "RAYT 1e-8" (the most closely related proteins from each sequence family), is significantly higher than expected by chance. This indicates that "IS*200* 1e-8" and "RAYT 1e-8" could be evolutionary links between the RAYT and IS*200* gene family; in other words their sequence identity has been preserved (presumably under purifying selection) since

RAYTs split from the IS*200* gene family. This finding supports the notion that IS*200* and RAYT proteins indeed share a common ancestry; nevertheless convergent evolution cannot be entirely ruled out since "IS*200* 1e-8" and "RAYT 1e-8" could be the result of recombination between RAYT and IS*200* genes.

5.2.2.2 Visualization of the relationship between the IS*200* and RAYT protein families

In order to better understand the evolutionary relationship between IS*200* and RAYT protein families the relationships between the individual RAYT and IS*200* proteins were visualized on a phylogenetic map (Figure 5.8). In this map, each protein is represented by a single node. An edge is drawn between two nodes if the pairwise identity is greater than a set threshold (see Figure 5.1). The distances between nodes are then displayed by cytoscape (www.cytoscape.org [123]), which calculates the distances between nodes based on the number of edges connecting the nodes. This layout option is called organic layout in cytoscape. For e-Values below 1e-2 (Figure 5.8A) and 1e-8 (Figure 5.8B) the relationship between RAYT and IS*200* proteins was visualized (see family sizes in Figure 5.2). Edges are drawn if the pairwise identity between two proteins exceeds 28%. The threshold of 28% was selected because it is slightly greater than the pairwise identity observed between the most closely related proteins from the IS*200* and RAYT family at an e-Value of 1e-8, which was considered highly significant (Table 5.1). Although a very conservative threshold, it ensures that connections between nodes are highly unlikely to be the result of chance. This is of particular importance as more than 1,000 proteins (more than 10^6 comparisons) are involved in this analysis.

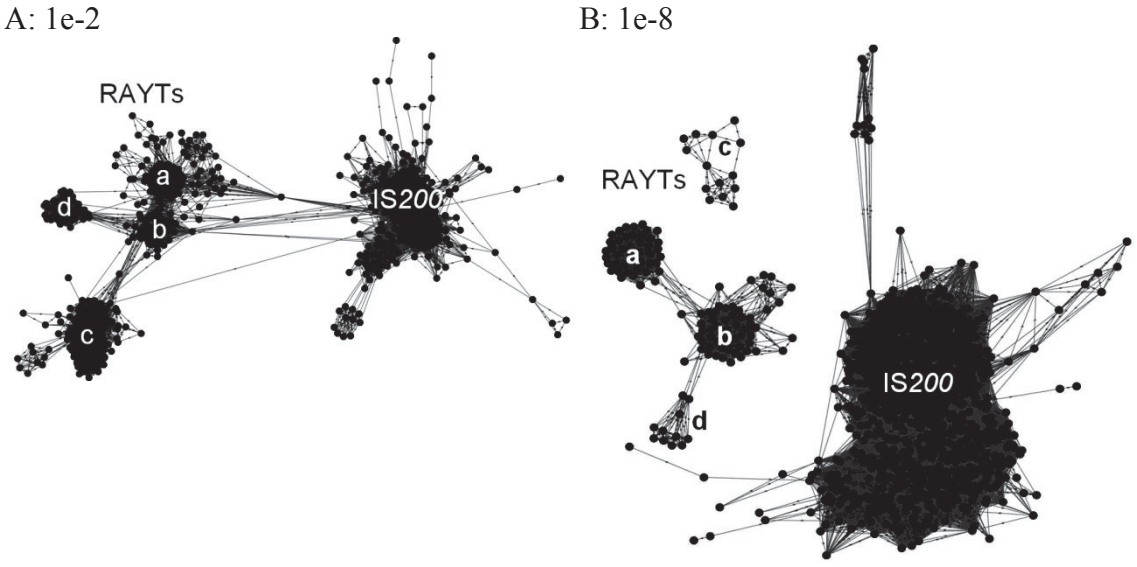


Figure 5.8. Phylogenetic clusters formed by IS*200* and RAYT proteins. Each node represents a protein found either by performing a BLAST search with two RAYT proteins (left) or two IS*200* proteins (right). Edges between nodes are drawn if the pairwise identity is higher than 28%. (*A*) Proteins shown that were found above an e-Value of 1e-2. Four separate RAYT groups are formed ((a), (b), (c) and (d)). (*B*) Proteins shown that were found above an e-Value of 1e-8.

Both Figure 5.8A and B show that the IS*200* family forms a reasonably homogenous cluster. In contrast, the RAYT gene family forms four separate clusters, named (a), (b), (c) and (d). As expected by the selection of the identity threshold, there are no connections between the IS*200* and RAYT family at an e-Value of 1e-8 (Figure 5.8B). However, for the less stringent e-Value of 1e-2, several proteins are added that connect the RAYT with the IS*200* gene cluster. Connections are formed between IS*200* and RAYT sequence clusters (a), (b) and (c). Another noteworthy observation is that cluster (c) and (d) below an e-Value of 1e-8 (Figure 5.8B) are very small in comparison to cluster (c) and (d) below an e-Value of 1e-2 (Figure 5.8A). This means the increase in RAYT family size below an e-Value of 1e-2 noted in section 5.2.1.1 (Figure 5.2) is almost entirely attributable to the addition of genes to the sequence clusters (c) and (d). Each of the clusters is investigated in more detail in the following section.

5.2.3 The four phylogenetic RAYT clusters and their characteristics

The phylogenetic maps in the section above show that RAYTs are not a homogenous gene family at an e-Value of 1e-2; instead the RAYT family consists of four separate subfamilies (clusters (a), (b), (c) and (d) in Figure 5.8A B). Furthermore in section 5.2.1 at an e-Value of 1e-2 the RAYT family also showed a change in genomic

distribution. The distribution at an e-Value of 1e-2 was more similar to that of IS*200* and IS*110* insertion sequences than RAYT characteristics observed at lower e-Values[1]. It is possible that the new RAYT members (especially members added to cluster (c) and (d) Figure 5.8) that were added at an e-Value of 1e-2 have insertion sequence characteristics and hence cause the observed shift in genomic distribution. If this is so then it could indicate that the propensity to transpose in *cis* (transpose itself) was lost and regained during the course of RAYT evolution. To test this hypothesis the characteristics for each individual RAYT subfamily at an e-Value of 1e-2 were studied below.

5.2.3.1 The phylogenetic map of the RAYT family

To study the individual characteristics of the four RAYT subfamilies the RAYT family alone was visualized in a phylogenetic map below an e-Value of 1e-2 (Figure 5.9). Figure 5.9 clearly shows the individual RAYT clusters. Cluster (c) is obviously the largest followed by cluster (a), (b) and (d). The difference in cluster size for clusters (a), (b) and (d) may merely be the result of a bias in the availability of fully sequenced bacterial genomes. For example, cluster (a) contains all RAYT homologues found in *E. coli*. Due to the large number of sequenced *E. coli* genomes this inflates the size of cluster (a). It is less likely that the size of cluster (c) is inflated due to its wide taxonomic distribution, as this would tend to deflate rather than inflate its size (Table 5.2). Clusters (a) and (b) are the most highly interconnected of the four clusters. This high connectivity indicates that there is a closer relationship between (a) and (b) than between any of the others. As expected from the phylogenetic analysis in Chapter 4 (Figure 4.3), YafM from *E. coli* and YafM from *P. fluorescens* SBW25 are found in the most closely related clusters, clusters (a) and (b) respectively. Cluster (b) also contains the IS*200*-RAYT hybrid protein ("RAYT 1e-8") identified in Table 5.1 and marked

[1] These are: (1) proportion of duplicates (increased from 4.2% to 7.1%), (2) average number of homologous genes per genome/plasmid (increased from 1.5 to 2.1), (3) proportion of genes found on plasmids (increased from 1% to 2.9%), (4) number of different taxonomic classes for which RAYTs were identified (increased from 14 to 31) and (5) the average frequency of the most abundant 16-mers flanking RAYT (decreased from 65 to 31).

with a red arrow in Figure 5.9. As indicated in Table 5.1 this protein connects and therefore is similar to both the IS*200* and RAYT sequence cluster.

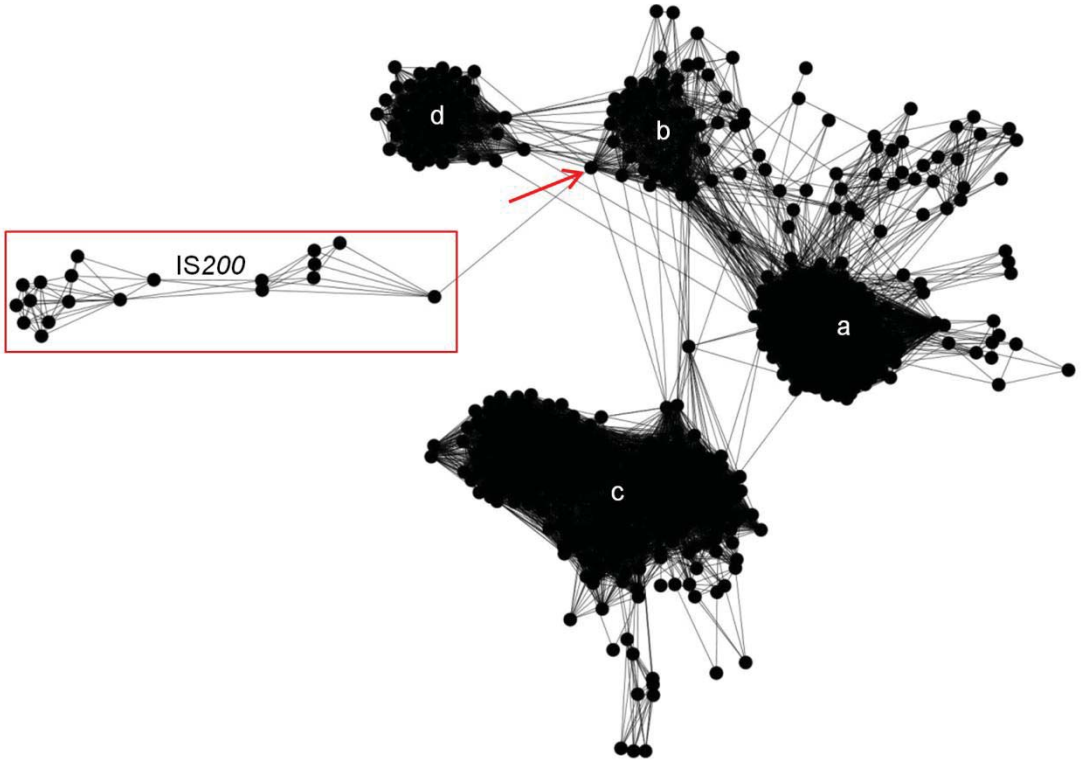


Figure 5.9. Phylogenetic RAYT clusters. Nodes represent RAYT proteins found through BLAST searches of two RAYTs below an e-Value of 1e-2. Edges between two nodes are drawn if the pairwise identity is greater than 28%. Groups are labelled according to Figure 5.8. The query proteins for the BLAST search are found in cluster (a) (YafM *E. coli*) and cluster (b) (YafM *P. fluorescens*). Red box indicates members of the IS*200* family. Red arrow indicates the hybrid protein CPS_1489 from Table 5.1.

5.2.3.2 Genomic distribution of the individual RAYT clusters

To be able to study the properties of the individual sequence clusters and thereby test the hypothesis that the change in RAYT characteristics is due to the addition of new genes to clusters (c) and (d), all four RAYT sequence clusters were manually extracted from the map and the same five characteristics as in section 5.2.1 were determined for each cluster (Table 5.2). In support of the hypothesis, cluster (c) and cluster (d) have the most unique properties, which will be discussed in detail in the next paragraph. In contrast, clusters (a) and (b) as expected show RAYT-like properties with a low propensity of *cis* transposition activity (copying the gene from which the transposase was expressed, evident by low gene duplication rate) but a high propensity for in *trans* transposition activity (copying sequences other than the gene from which the

transposase was expressed, evident from the high frequency of the associated 16-mers). This means that cluster (a) and (b) genes are more likely to copy associated REPINs than copy themselves.

Table 5.2. Characteristics of RAYT sequence clusters from Figure 5.9.

	Cluster (a)	Cluster (b)	Cluster (c)	Cluster (d)
# genes	131	59	205	52
# duplicates[a]	4	2	26	2
# plasmids[b]	0	0	2	0
# per replicon[c]	1.3±0.06	1.8±0.13	1.9±0.18	1.7±0.17
16-mer frequency[d]	49.5±4.3	92.7±13.2	10.9±1.2	7.9±1
# taxonomic classes	7	5	19	8

[a]Number of genes, for which a homologue with a pairwise identity of more than 95% is found in the same genome/plasmid. [b]Number of times the respective gene was found on a plasmid. [c]Average and standard error shown. [d]Average and standard error of the most abundant 16-mers found in the immediate extragenic space flanking the respective gene. The *P*-values for the average number of homologues per replicon are: A-B: 4e-5; A-C: <1e-5; A-D: 0.0011; B-C: 0.24743; B-D: 0.36731; C-D: 0.15247. Only the differences between cluster A and the other clusters are significant. The *P*-values for the average frequency of the most abundant flanking 16-mers are: B-A: 7e-5; B-C: <1e-5; B-D: <1e-5; A-C: <1e-5; A-D: <1e-5; C-D: 0.00127. All differences are significant.

The large increase in the number of duplicates above an e-Value of 1e-8 is almost entirely attributable to duplication events occurring in cluster (c), which shows a transposition activity that falls between the values of insertion sequences and RAYTs. Cluster (c) also contains the only genes that are located on plasmids (although at an e-Value of 1e-2 there are also genes that are not connected to any of the clusters that are found on plasmids). This indicates higher rates of horizontal transfer and hence suggests a greater degree of selfishness; however one could argue that this is solely due to the higher number of sampled genes. A higher degree of selfishness is also reflected in the larger number of homologues per replicon. Atypical for RAYTs, the most abundant 16-mers in flanking extragenic space occur at relatively low frequencies (10.9, comparable to that of *def* at 4.9), indicating the absence of REPs/REPINs. Together the data indicate that cluster (c) genes are more selfish and similar to insertion sequences than the rest of the RAYT sequence group.

In contrast, cluster (d) RAYTs, show characteristics similar to clusters (a) and (b) with the exception of association with over-represented 16-mers. Hence, cluster (d) RAYTs neither show signs of *cis* or *trans* transposition, which poses an even greater challenge for the elucidation of their function within the bacterium.

To compare the different RAYT clusters to each other and to the IS*200* sequence family on a molecular level, the consensus sequence for each cluster was calculated and aligned (Figure 5.10). A comparison of the conserved motifs present in the consensus sequence alignment of the four RAYT clusters with those of the IS*200* gene family further supports the conjecture that RAYTs are a sequence group distinct from IS*200* (RAYTs share more motifs within the gene family than they share with IS*200*, see Figure 5.10).

Figure 5.10. ClustalW2 [120] consensus sequence alignment of the four RAYT and the IS*200* protein families. 50% consensus sequences were extracted from multiple alignments of clusters from Figure 5.9 and all IS*200* proteins found below an e-Value of 1e-2. Xs are shown in the consensus sequence if less than 50% of the amino acids at the particular position agree. The consensus sequence alignment was refined manually.

5.2.3.3 Characterization of RAYT duplication events

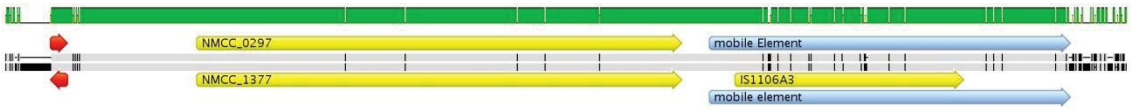
RAYT gene duplications (defined as genes found on the same plasmid/genome that share more than 95% nucleotide sequence identity) provide a wealth of information about the transposition process. Most importantly, a recent duplication event with intact flanking sequences can help to identify the flanking repeats that were required for transposition. Therefore, at least one duplication event will be analysed for each of the four clusters in the following sections.

5.2.3.3.1 Duplication events in cluster (a)

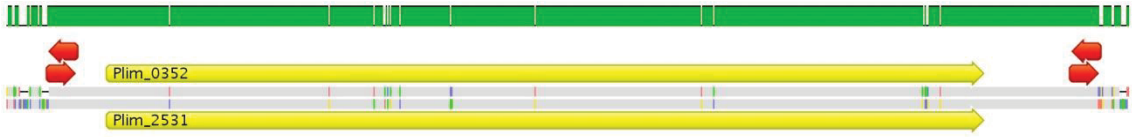
Two duplication events were observed in cluster (a): one in *Neisseria meningitides* 053442 and one in *Planctomyces limnophilus* DSM 3776.

The duplication event in *Neisseria* involves the genes *nmcc_0297* and *nmcc_1377*. The duplication is delimited at the 5′ end by the most abundant 16-mer in the flanking extragenic space, which occurs 315 times and forms a short hairpin (grey arrow Figure 5.11). This means a short REP sequence was required for transposition at the 5′ end of the duplication, probably recognized by the RAYT encoded by either *nmcc_0297* or *nmcc_1377*. In other parts of the genome the 16-mer (REP) is found as part of REPINs (Figure 5.12A). This is predicted by the analyses performed in Chapter 3, where REPINs have been shown to be the prerequisite for REP sequence dispersal throughout the genome. Interestingly, the duplication also involves a truncated version of the insertion sequence IS*1106A3* (including the terminal inverted repeat at the 3′ end) and no corresponding copy of the palindromic 16-mer from the 5′ end of the duplication. The fact that the 5′ end of the duplication event consists of a REP sequence (typically associated with RAYTs) and the 3′ end consists of the 3′ end of the insertion sequence IS*1106A3* raises the possibility that the transposition event involved both the transposase encoded by IS*1106A3* and the RAYT protein. However, the pairwise identity between the two truncated IS*1106A3* elements is only 94.9% whereas the pairwise identity between the RAYT genes is about 98.8%. This could be due to two processes: either the RAYT gene was inserted into an existing IS*1106A3* copy (there are numerous IS*1106A3* copies in the genome) at exactly the same position; or the RAYT gene and the truncated IS*1106A3* copy transposed as a unit and the higher conservation of the RAYT gene is the result of purifying selection. Purifying selection was suggested earlier as the selective process that preserves RAYT genes within genomes in the absence of horizontal transfer events (section 5.2.3.2). In order to be preserved by selection RAYTs are predicted to perform an unknown function for the host bacterium. Hence, it seems more likely that this duplication event is a result of cooperation between a RAYT and an IS*1106A3* insertion sequence and not two independent insertions into IS*1106A3*.

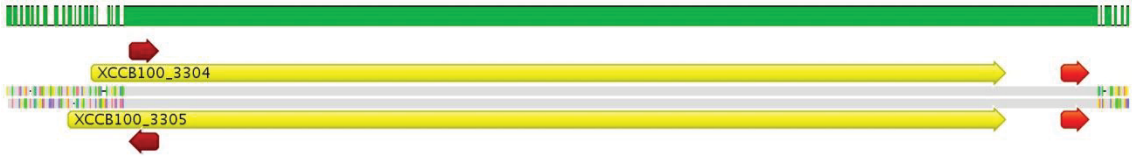
A: Duplication in cluster (a): *Neisseria meningitides* 053442



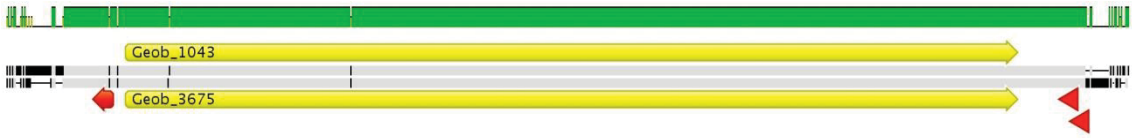
B: Duplication in cluster (a): *Planctomyces limnophilus* DSM 3776



C: Duplication in cluster (b): *Xanthomonas campestris* B100



D: Duplication in cluster (c): *Geobacter sp. FRC-32*



E: Duplication in cluster (d): *Prosthecochloris aestuarii* DSM 271

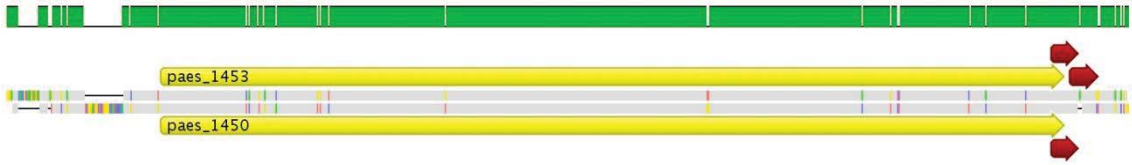


Figure 5.11. Alignment of duplicated RAYT regions in the four different RAYT clusters. Most abundant 16-mers are red. Open reading frames are yellow. First line in each figure is conservation of individual nucleotide sites. (*A*) Truncated IS*1106A3* is shown in light blue.

The duplication event in *Planctomyces limnophilus* DSM 3776 involves the genes *plim_0352* and *plim_2531* (Figure 5.11B). The duplication is delimited on both sides by the most abundant 16-mer in the flanking extragenic space, which occurs 16 times in the genome and can form a short hairpin structure. Of the remaining 12 (four are in the duplication) 16-mers, six are found in doublet conformation, of which two are arranged as inverted repeats and one as direct repeat. However, the structure that is formed by the doublets in inverted orientation is quite different from REPINs described in Chapter 3. They are larger (201 bp and 225 bp compared to only about 100 – 150 bp for

REPINs in SBW25) and contain a long insert, which is not part of the hairpin (Figure 5.12B). REPINs in *P. limnophilus* are quite different from other REPINs described that were found in association with cluster (a) RAYTs. However, REPINs associated with cluster (a) RAYTs have been shown to be more diverse than REPINs associated with cluster (b) RAYTs, so it is not overly surprising that there is yet another REPIN structure.

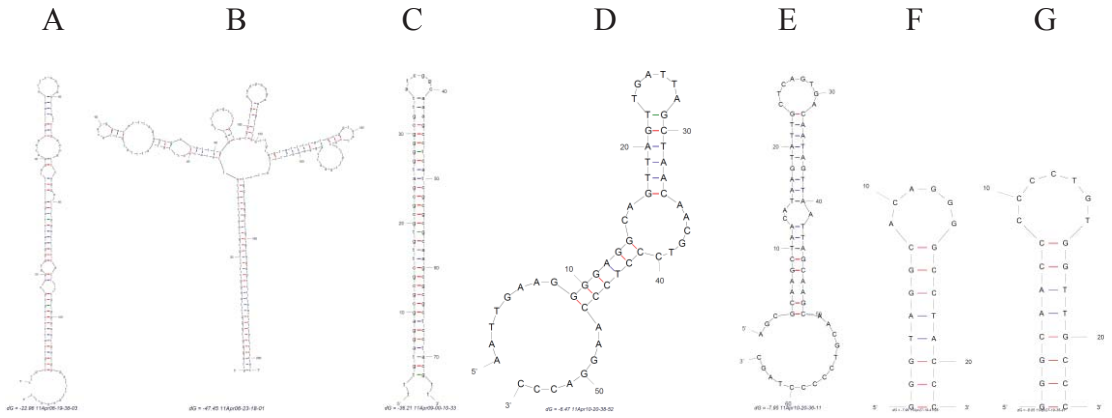


Figure 5.12. Secondary structures formed by REPs and REPINs. Secondary structure formed by inverted REP doublets found in (*A) Neisseria meningitides* 053442 (cluster (a)), *(B*) *Planctomyces limnophilus* DSM 3776 (cluster (a)), (*C*) *Xanthomonas campestris* B100 (cluster (b)). Structures formed by (*D*) 3' and (*E*) 5' flanking sequences of RAYT duplicates in *Geobacter sp FRC-32* (cluster (c)). (*F*) and (*G*) show palindromes that were found in the 5' and 3' flanking sequences of *paes_1450* and *paes_1453* from *Prosthecochloris aestuarii* DSM 271 delimiting the duplication (cluster (d)).

5.2.3.3.2 Duplication events in cluster (b)

Only one duplication event was identified for genes from cluster (b). The duplication occurred in *Xanthomonas campestris* B100 and involves two adjacent genes in inverted orientation called *xccb100_3304* and *xccb100_3305* (Figure 5.11C). The duplication event is delimited by the most abundant 16-mer (occurs 120 times) of the flanking DNA on one side and a highly abundant 16-mer (occurs 50 times, differs from the most abundant 16-mer by one nucleotide) on the other side. Typical REPINs as the ones described for SBW25 are formed throughout the genome (Figure 5.12C).

5.2.3.3.3 Duplication events in cluster (c)

As noted above, genes from cluster (c) have very different properties compared to genes from cluster (a) and cluster (b). This is again apparent when analysing duplication events. The duplication event involving the most closely related genes to the two query sequences occurred in *Geobacter sp. FRC-32* (*geob_1043* found at position 1146819-

1147784 and *geob_3675* found at position 4080757-4079792) (Figure 5.11D). Interestingly, two truncated RAYT genes are found in the vicinity of *geob_1043* and *geob_3675* respectively that are almost identical to the two genes (Figure 5.13). One gene called *geob_3667* is found at position 4070369-4070048 and matches to the start (1-322) of *geob_1043* and *geob_3675* and the second gene called *geob_1038* is found at position 1142229-1142794 and matches to the end (404-969). Interestingly, *geob_1043* faces the 5′ end towards *geob_1038*, of which only the 3′ end is left, and *geob_3675* faces the 3′ end towards *geob_3667,* of which only the 5′ end is left. An additional RAYT homologue is found in between *geob_1038* and *geob_1043* called *geob_1040.* This arrangement is strongly reminiscent of composite transposons. The genes encoded in between the truncated RAYTs and the full length homologue, are mostly YD repeat genes predicted to encode cell surface proteins (typically transferred horizontally) [178]. Additionally, between *geob_3667* and *geob_3675*, two genes are found that encode for abortive infection proteins implicated in host defence against phage infection [84].

It is difficult to determine the flanking repeats that were required for the duplication. At the 5′ and 3′ end of the duplicated genes (*geob_1043* and *geob_3675*) a conserved sequence is found, but is not located at the exact end of the duplicated region. Two slightly different hairpin structures are formed by the 5′ and 3′ end (Figure 5.12D and E); again no REP or REPIN sequences were identified within the genome. Finding no repeated palindromic sequences immediately flanking the duplication may either indicate that the transposition mechanism is different from other IS*200* or RAYT

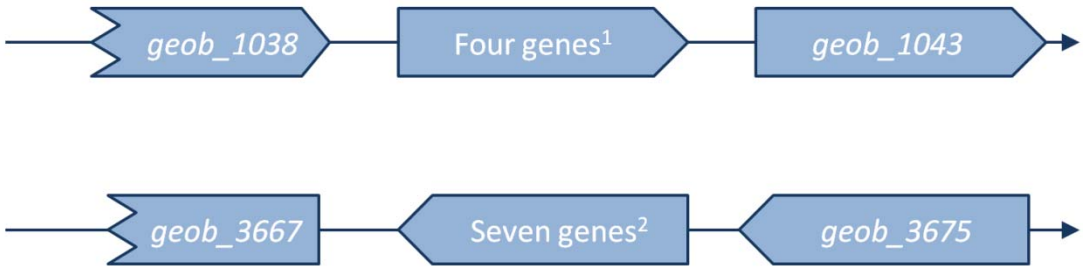


Figure 5.13. The genomic region that contains the two cluster (c) RAYT duplicates *geob_1043* and *geob_3675* (969 bp) as well as the truncated versions of the two genes *geob_1038* (404-969 bp) and *geob_3667* (1-322 bp). [1]Two of the four genes encode for YD repeat proteins. [2]Three of the seven genes encode for YD repeat proteins and two for abortive infection proteins. YD repeat proteins or *rhs* genes have been implicated in O-antigen variation in *E. coli* and have a different evolutionary history to the core genome, which indicates horizontal gene transfer [178]. Abortive infection proteins have been implicated in immunity to phage infection [84] as well as a means to confer self-immunity to bacteriocins [179].

proteins or that the exact ends of the duplication were lost.

5.2.3.3.4 Duplication events in cluster (d)

For cluster (d) only one duplication event was identified. It involved the genes *paes_1450* and *paes_1453* from *Prosthecochloris aestuarii* DSM 271 (Figure 5.11E). The event was delimited by two similar short palindromes (Figure 5.12F and G). Interestingly, for *paes_1453* the palindrome found at the 5' end of the gene is deleted from the duplication. This could be due to selection for a decreased transposition rate.

As expected from the above analysis, no REPs or REPINs involving the short palindromes could be identified within the genome. Nevertheless, the protein coding region of the duplication showed a higher pairwise identity (97.1%) than the 5' (89.7%) and 3' (87%) flanking regions, which indicates that selection has acted to preserve the gene.

5.3 Discussion

5.3.1 Overview of the results

The overarching aim of this chapter was to systematically characterize the RAYT family of proteins. Specifically, this involved characterizing the genomic distribution of RAYTs in bacteria (section 5.2.1), and investigating the evolutionary origins and history of RAYTs (sections 5.2.2 & 5.2.3). The comparison of features concerning the genomic distribution between RAYTs, the IS*200* and IS*110* insertion sequence families and the housekeeping gene family *def* showed that RAYTs are in most characteristics more similar to housekeeping genes than to insertion sequences. However, for family members identified below an e-Value of 1e-2 (family includes very distantly related proteins) the characteristics became more similar to insertion sequences. To determine what caused this change RAYTs had to be analysed in more detail. Hence a phylogenetic analysis of RAYTs was conducted. The analysis showed that the RAYT family is extremely diverse and consists of four discrete clusters at the least stringent definition of the sequence family (all BLAST search results below an e-Value of 1e-2 were considered). Two of the clusters showed characteristics typical for RAYTs (no duplications, not found on plasmids, highly abundant flanking 16-mers), one showed similar characteristics to RAYTs but without the association to highly abundant 16-mers, the last RAYT cluster in contrast showed characteristics similar to insertion sequences rather than RAYT genes. These results will be discussed in the following sections.

5.3.2 The genomic distribution of the RAYT gene family

For RAYTs identified at lower e-Values (1e-8, 1e-14 and 1e-20) the proportion of duplicates, number of homologues per replicon and gene occurrences on plasmids are similar to numbers observed for the housekeeping gene family *def* and greatly differ from numbers observed for the IS*200* and IS*110* insertion sequence families. At an e-Value of 1e-2 the same characteristics are significantly different from that of the *def* gene family but also still different from the two insertion sequence families.

The gene family size as well as the distribution among bacterial taxonomic classes is much smaller for RAYTs than that of IS*200* and IS*110* insertion sequences or *def* genes. However, at an e-Value of 1e-2 there is a great increase in family size as well as an increase in distribution among taxonomic classes. This effect, together with data of a greater duplication rate and a strong increase in gene occurrences on plasmids, is probably the result of a higher horizontal transfer rate of RAYTs that are added at an e-Value of 1e-2. The proposed higher rate of horizontal transfer of RAYTs added at an e-Value of 1e-2 is hence supported by all genomic distribution characteristics. Conversely the results also support a low horizontal transfer rate of RAYTs that are identified at lower e-Values.

Since the rate of horizontal transfer is tightly linked not only to insertion sequence [147, 180] activity but is also a hallmark for addictive selfish genetic elements [22, 170, 181, 182], it seems possible that RAYTs that are identified for low e-Values provide a beneficial function, whereas most RAYTs that are added at an e-Value of 1e-2 are genetic elements with more selfish characteristics that require horizontal transfer to persist. This hypothesis is also supported by the analysis of the genomic distribution of RAYT subfamilies (see section 5.3.4).

5.3.3 The relationship between the RAYT and the IS*200* family

The pairwise identity between a selection of RAYT and IS*200* proteins showed no differences from pairwise identities of the corresponding shuffled sequences (see section 5.2.2). This means that there is no significant direct evolutionary relationship between those proteins. However, the most closely related proteins from the two sequence groups each showed similarity to both IS*200* sequences and RAYTs. The existence of such hybrid sequences was somewhat surprising however these sequences may indicate that IS*200* and RAYT genes did not emerge through convergent evolution. Rather they indicate that RAYTs and IS*200* genes emerged independently and the observed hybrid is the result of recombination between members of the two sequence groups.

5.3.4 RAYT subfamilies and their genomic distribution

Closer investigation of the RAYT gene family showed the formation of four separate phylogenetic groups (see section 5.2.3). Analyses of these groups revealed that each showed very distinct properties.

Clusters (a) and (b) contained the query sequences from *E. coli* and SBW25 respectively. In accordance with findings from Chapter 4, both RAYT groups are associated with REPINs. Despite being the most closely related RAYT groups, the considerable phylogenetic distance between the two groups is reflected in different REPIN structures (Chapter 4 and Figure A2.1) and other characteristics such as the average number of homologues per replicon, or frequency of flanking 16-mers.

RAYT clusters (c) and (d) gave rise to some of the most interesting findings of this study. Both groups have very different characteristics and very distantly related to the other two RAYT clusters (almost as distantly as IS*200* sequences).

Cluster (c) shows frequent duplication events and is the only RAYT group for which members were found on plasmids. Furthermore, cluster (c) is considerably larger than the other three groups. As indicated by the low frequency of flanking 16-mers and the analysis of one duplication event, this cluster is unlikely to be associated with REPINs. Together, these data suggest that cluster (c) genes more closely resemble insertion sequences than the other three RAYT groups. The sequence alignment in Figure 5.10 shows that the consensus sequence of cluster (c) members is more similar to the other RAYT groups than it is to IS*200* sequences.

Assuming the connection between cluster (b) and the IS*200* family shown in Figure 5.9 reflects a real evolutionary relationship and is not the result of recombination then cluster (b) RAYTs evolved from IS*200* sequences. The sequence and phylogenetic data further suggests that the other three RAYT subfamilies diversified from cluster (b) into clusters (a), (c) and (d). Since cluster (b) RAYTs show little *cis* transposition activity (copying themselves), this model suggests that RAYTs evolved from the IS*200* sequence family by losing the ability for *cis* transposition. Interestingly, cluster (c) RAYTs show an unusually high propensity for *cis* transposition compared to the other RAYT subfamilies. This indicates that cluster (c) RAYTs re-evolved insertion

sequence like activities and probably lost (at least partially) the unknown but inferred ability of RAYTs to provide a certain beneficial function to the host. However, due to the limited sequence similarity the possibility remains that RAYTs did not evolve from IS*200* sequences but through convergent evolution. However, even this scenario still suggests that insertion sequence like activities can evolve from single copy host genes.

While cluster (d) genes show very similar characteristics to RAYT genes from cluster (a) and (b) a notable difference is that cluster (d) genes are not physically associated with REPs or REPINs (based on the low frequency of flanking 16-mers (similar to *def*) and analysis of the observed duplication event: section 5.2.3.3.4). Sequence comparison of the genes involved in the duplication event revealed a higher level of sequence conservation within the gene than that observed in the extragenic space (section 5.2.3.3.4). The fact that selection is acting to preserve the gene sequence strongly suggests cluster (d) genes fulfill a beneficial function in the bacterium. The nature of this function remains to be seen, and as such is an area of great interest for future experiments.

5.3.5 Conclusion

Further research is needed to reveal more about the evolutionary history of RAYTs, which could provide a general insight into the evolution of selfish genetic elements. Specifically, research on the evolutionary history of RAYTs could shed light onto questions such as how beneficial functions are gained and lost (IS*200* to RAYTs, RAYTs to cluster (c)). Although the graph in Figure 5.9 indicates that first cluster (b) evolved from IS*200* sequences, which diversified into clusters (a), (c) and (d), there are too few and some contradictory connections (possibly a result of recombination, drift or selection) to strongly support this hypothesis.

The above analyses make very clear that RAYTs mainly spread vertically (from one generation to the next) in gene pools rather than horizontally, with the possible exception of cluster (c) RAYTs, which show higher duplication rate and are found on plasmids. In order to preserve vertically transmitted genes within the bacterial genome, those genes have to confer a benefit to the host. Especially considering the strong deletional bias (strong propensity of bacteria to lose surplus DNA) observed for

bacterial genomes [183, 184]. The split of the RAYT gene family into distinct clusters and the existence of RAYT hybrids suggest that it is unlikely that RAYTs possess a universal function. However, elucidating the functions for different RAYT genes experimentally is likely to deliver highly interesting results and will show how it is possible for insertion sequences to switch from a horizontal to a vertical transmission mode and may even provide an explanation of how insertion sequences initially evolved from single copy genes.

Chapter 6:
Evolutionary characterization of two repetitive sequence classes in the genome of SBW25

6.1 Introduction

6.1.1 Regulatory antisense RNA in bacteria

Across all domains of life non-coding RNA has been shown to account for a major proportion of total transcripts within the cell [185]. These RNA molecules play an important catalytic role (ribozymes [186] *e.g.* ribosomal RNA [187], Group I [188] or II introns [38]), but are also involved in the regulation of gene expression, specifically as non-coding antisense RNAs (asRNAs) [185, 189, 190]. Non-coding RNAs can be divided into two groups: *cis*-acting asRNAs and *trans*-acting asRNAs.

6.1.1.1 Cis-acting antisense RNAs

Cis-acting asRNAs are transcribed on the opposite strand of the protein-encoding gene that is the target of regulation. Regulation can be achieved at either the transcriptional or translational level [185].

6.1.1.1.1 Regulation at the transcriptional level

There is a wide range of mechanisms for transcriptional regulation of gene expression by asRNA. The most common mechanisms involve transcriptional termination or transcriptional interference.

To the author's knowledge there are only two examples where transcription termination has been confirmed to be caused by the presence of asRNA. The first example involves the siderophore synthesis operon *fat* in *Virbio anguillarum*. Here, the authors show through *in vitro* experiments that transcription termination is dependent upon the

presence of asRNAs, and not a result of transcriptional interference. However, the exact mechanism of this type of transcription termination remains unknown [191]. The second example is the regulation of the virulence gene *icsA* through the asRNA RNAg. Here the evidence suggests that the binding of asRNA to the mRNA transcript prevents the formation of an antitermination structure, which leads to the termination of transcription and the dissociation of the RNA polymerase from the DNA [192].

The second mechanism for gene regulation at the transcriptional level involves transcriptional interference. Unlike the mechanisms presented above, asRNA is the effect rather than the cause of transcriptional interference. The cause for transcriptional interference is usually an oppositely oriented promoter. There are three interference mechanisms: collision, promoter occlusion and sitting duck. Collision interference describes the process of transcription termination due to the interference of two convergently transcribing RNA polymerases [193]. Promoter occlusion occurs when the transcription from a strong promoter prevents formation of the transcription initiation complex on the opposite strand [185, 194]. Sitting duck interference refers to a process where an open RNA polymerase complex is removed by a convergently transcribing polymerase [194].

6.1.1.1.2 Regulation at the translation level

Translation of the mRNA can be regulated through *cis*–acting asRNA by occupying the ribosome binding site (or a region close to the site) and hence preventing the initiation of translation or/and by reducing the mRNA half life by recruiting RNases for degradation [185].

A prominent example for the inhibition of translation by *cis*-acting asRNA is the SymE/SymR TA system [195]. The authors show that SymR is a *cis*-acting asRNA that suppresses the translation of SymE mRNA. SymE is a protein that leads to "reduced colony formation, decreased protein synthesis as well as significant decreases in the levels of several RNAs" [195]. A surprising finding of the SymE/SymR study by Kawano et al. was that in contrast to other type I TA systems (where the addiction is dependent on a stable toxin and an unstable antitoxin) the antitoxic asRNA SymR is surprisingly stable. This together with the fact that SymE expression is induced by the

SOS response led to the hypothesis that the SymE/SymR system confers a benefit to the cell. The authors propose that under stressful conditions the protein could possibly aid the cell in the recycling of damaged potentially toxic RNA molecules.

6.1.1.2 Trans-acting antisense RNAs

Trans-acting asRNAs have been shown to regulate gene expression at the translational level, although transcriptional regulation is possible in theory no examples have been reported to the author's knowledge. Translational regulation is achieved through almost the same mechanisms as for *cis*-acting asRNA. However, *trans*-acting asRNAs have also been shown to have positive regulatory effects on translation. The binding of mRNA by asRNA can for example resolve inhibitory complexes and therefore activate translation by allowing the ribosome to bind the mRNA [189].

The largest difference between *trans* and *cis*-acting RNA is the length of the interaction (base pairing) between asRNA and mRNA. *Trans*-acting asRNAs usually bind relatively short stretches of mRNA (~10-25 bp), whereas *cis*-acting asRNAs interact with mRNA regions of lengths between 100 and 7,000 bp [185]. Shorter regions of complementarity result in a broader target specificity. That means that multiple genes can be regulated by a single asRNA. This allows the regulation of global gene expression as a response to environmental cues. For example, the asRNA RhyB regulates the global use of iron in *E. coli* by catalyzing the degradation of the mRNA of various iron binding proteins [196].

Well studied *trans*-acting asRNAs are type I TA systems such as TisB/IstR-1 (which is covered in the introduction see section 1.4.1.1.2) and ShoB/OhsC [197]. The ShoB toxin is a 26 amino acid long hydrophobic protein that is encoded upstream of *ohsC*. The toxicity of ShoB is supposedly due to an effect on the cell membrane, which was inferred from a gene expression analysis.

6.1.2 Computational approaches for identifying non-coding RNAs within bacterial genomes

Currently there are three ways to predict genomic regions that are likely to encode non-coding RNAs. These are: (1) similarity searches of RNA structures; (2) the analysis of local base compositions; and (3) comparative genomics [198].

Initially, it was proposed that analysis of the free energy of RNA secondary structures is sufficient to identify non-coding RNA regions within the genome [199]. However, it has been shown that the difference between the predicted free energy of randomly assembled sequences and non-coding DNA sequences are comparable to the free energy predicted for non-coding RNA sequences [200]. Nevertheless, RNA secondary structure predictions can still be used for the identification of RNA coding DNA sequences by comparing the predicted RNA secondary structure to known secondary structures or RNA secondary structures obtained from related DNA sequences [201, 202]. If the structures are similar then it is likely that the DNA sequence encodes RNA.

The identification of non-coding RNAs by analysing local base composition (*e.g.*, GC or dinucleotide content) is based on the assumption that non-coding RNAs form stable secondary structure in order to be functional [199]. The stability of RNA structures is greatly enhanced in regions with high GC content since G-C pairings are more stable than A-T pairings. Thus DNA sequences that show an increased GC content are more likely to code for non-coding RNA [200]. The difference between the average GC content of the genome and the GC content of non-coding RNA seems to be greatest for genomes with low GC content as well as bacteria that live in high temperature environments [200, 203].

Non-coding RNAs can also be identified through comparative genomics [202]. For such studies the search for transcription initiation and termination signals outside protein coding regions can provide candidate DNA sequences that potentially encode RNA [198]. Supporting evidence that these candidates encode RNA can be provided by sequence comparisons with closely related bacterial genomes. If the candidate sequence encodes an RNA, which means that it is functional, then selection can act to preserve structural motifs (hairpins or stems) within the sequence through compensatory mutations [202]. Hence, due to the large number of available bacterial genome sequences it is relatively easy to identify strong candidates for non-coding RNA sequences.

6.1.3 Repetitive sequence analysis in the SBW25 genome

Silby et al. [100] identified four major repeat families in the genome of *Pseudomonas fluorescens* SBW25: R0, R2, R178 and R200. R0 and R2 repeats are REPINs or parts of REPINs (Table 3.6) and have been studied in detail in Chapter 3. The work in this chapter focuses on the R178 and R200 repeats. Sequence analyses are applied in order to investigate the evolution and potential function of R178 and R200 repeats. Additionally, the notable association of R200 repeats with REPs/REPINs will be examined by analysing sequence identities and phylogenies in the different REP/REPIN backgrounds.

6.1.4 Aims

The aims of this chapter are:

(1) To determine sequence properties of the R178 and R200 repeats, and thereby gain insight into the evolutionary history of these repeat classes.

(2) To characterize the relationship between REPs/REPINs and the R200 repeats.

6.2 Results

Of the four repetitive sequence classes identified in the SBW25 genome [100], R178 and R200 remain to be characterized. These occur 18 and 47 times, respectively, within the extragenic space of the SBW25 genome. Each of the 18 R178 repeats is about 110 bp long, while the 47 R200 repeats range in length from 128 bp to 329 bp. When aligning all R200 repeats only a 110 bp long segment in the centre of the alignment is shared by all R200 repeats (see Figure 6.8). The characteristics and evolution of the two repeat sequences are studied in the following sections.

6.2.1 Characterization of R178 repeat sequences

6.2.1.1 Evolutionary origins of R178 repeats

There are several questions concerning the evolution of R178 repeats that can be addressed with computational analyses. The initial question concerns the evolution of the repetitiveness of R178 sequences. There are several competing hypotheses that need to be considered. Firstly, that repetitiveness could be the result of random chance. This is very unlikely as it was already shown in Chapter 3 (section 3.2.1) that even 16-mers do not occur repetitively by chance and R178 repeats are about 111 bp long. Secondly, R178 repeats could have emerged as the result of similar selective pressures at different positions within the genome. This would require that over long periods of evolutionary time, the sequences in all 18 sequence backgrounds acquired and preserved similar mutations. This process depends on constant and strong mutational pressure and hence one would expect the sequences to be preserved in closely related strains. However, Silby et al. [100] found that *P. fluorescens* Pf-5 contains nine, *P. fluorescens* Pf0-1 28 and *P. fluorescens* SBW25 18 R178 copies. The fact that R178 repeats are not conserved among closely related genomes together with the fact that they are unlikely to arise by chance indicates that R178 repeats are likely to be the result of a duplicative evolutionary process.

A duplicative process could be driven by either the R178 repeat itself, which means that it encodes a protein (as for example insertion sequences [18]) or an RNA molecule (as in Group I introns [188]) that copies R178 sequences (autonomous transposition).

Alternatively, R178 repeats may encode non-autonomous duplicative elements that are copied by a transposase or another protein that is encoded somewhere else in the genome [52].

To first test whether it is possible that R178 repeats encode autonomous duplicative elements, the sequences of all R178 repeats were analysed for the capability to code for proteins by searching for a conserved open reading frame. However, this search was unfruitful; no conserved open reading frame could be identified for all 18 R178 repeats. To test the possibility that R178 repeats encode an autocatalytic RNA molecule, the secondary structure was predicted for all 18 sequences (Figure 6.1A). The predicted RNA secondary structures are relatively similar, which indicates that selection may act to preserve their structure. Whether selection acts on preserving a transposable element that encodes an autocatalytic RNA function is unclear at this point.

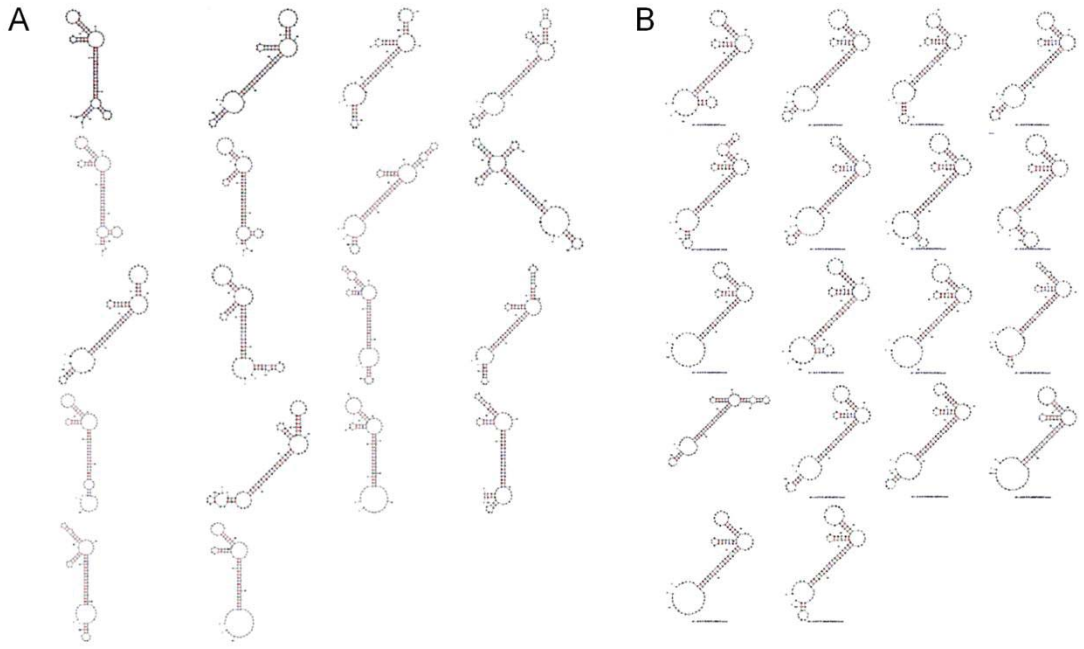


Figure 6.1. Predicted secondary structures of R178 sequences calculated by the mfold web server [104]. (*A*) RNA and (*B*) ssDNA secondary structures of all R178 repeats found in the SBW25 genome.

Alternatively R178 repeats could encode for non-autonomous duplicative elements as, for example, REPINs (see Chapter 3). Non-autonomous REPINs are thought to be transposed by RAYTs, which show sequence similarities with IS*200* transposases. IS*200* transposases have been shown to transpose their targets as single strands. Hence, the single stranded structure that is formed by IS*200* genes as well as REPINs is likely to be important for the transposition process. REPINs form a highly conserved

secondary structure in ssDNA (see section 3.2.3.5). The conservation suggests that the structure is functional and could for example affect the transposition process. It is possible that a conserved structure is also formed by non-autonomous R178 sequences. Hence, the secondary structure in ssDNA was predicted for all 18 R178 sequences (Figure 6.1B). Interestingly, the structure is highly conserved in all 18 sequences (even more so than observed for RNA secondary structures), and contains four distinct loops. This conservation is likely to be reflected within the nucleotide sequences of R178 repeats as, for example, complementary pairs of conserved nucleotides as well as complementary mutations in stem regions of the secondary structure. Such characteristics can be observed in a multiple sequence alignment, an analysis which will be performed in the following sections. This also includes analyses on the relationship between secondary structure and polymorphic regions within the alignment.

6.2.1.2 DNA sequence alignment of R178 repeats

To analyse polymorphic and conserved R178 regions, a multiple sequence alignment of all 18 identified DNA sequences using ClustalW2 [120] was performed (Figure 6.2). Most regions of the ~110 bp R178 repeat are highly conserved. However there are a few notable exceptions. In order to determine polymorphic or non-conserved sequence sites, the sequence diversity was calculated for each nucleotide within the alignment. A common measure for sequence diversity is the Shannon Entropy *H* [204], where high entropy values indicate high diversity and low values low diversity. For the four letter nucleotide alphabet the highest possible entropy is two bit. This is the case when at a certain position in an alignment all four nucleotides occur at the same frequency (*i.e.* 25%). That means if the aligned sequences are randomly assembled and have a GC content of 50% the maximum entropy for each position is two bit. In comparison, the maximum entropy for aligned sequences from the SBW25 genome with a GC content of 60.5% is 1.97 bit. In contrast to positions with maximum entropy, an entropy of zero bit is observed when a position within an alignment is completely conserved. If at a position two nucleotides occur at the same frequency the entropy is one bit. The entropy of one bit can be used as a threshold to distinguish between conserved and polymorphic sequence positions within an alignment. For the R178 alignment, this means that there are roughly seven polymorphic regions (red boxes Figure 6.2; entropy values for each alignment position are found in Table A3.1). The longest polymorphic

section extends over 13 bp from position 46 to position 59 (site 3, this region includes position 48 which is conserved). In total 34 of 111 nucleotide positions are highly conserved in the R178 alignment.

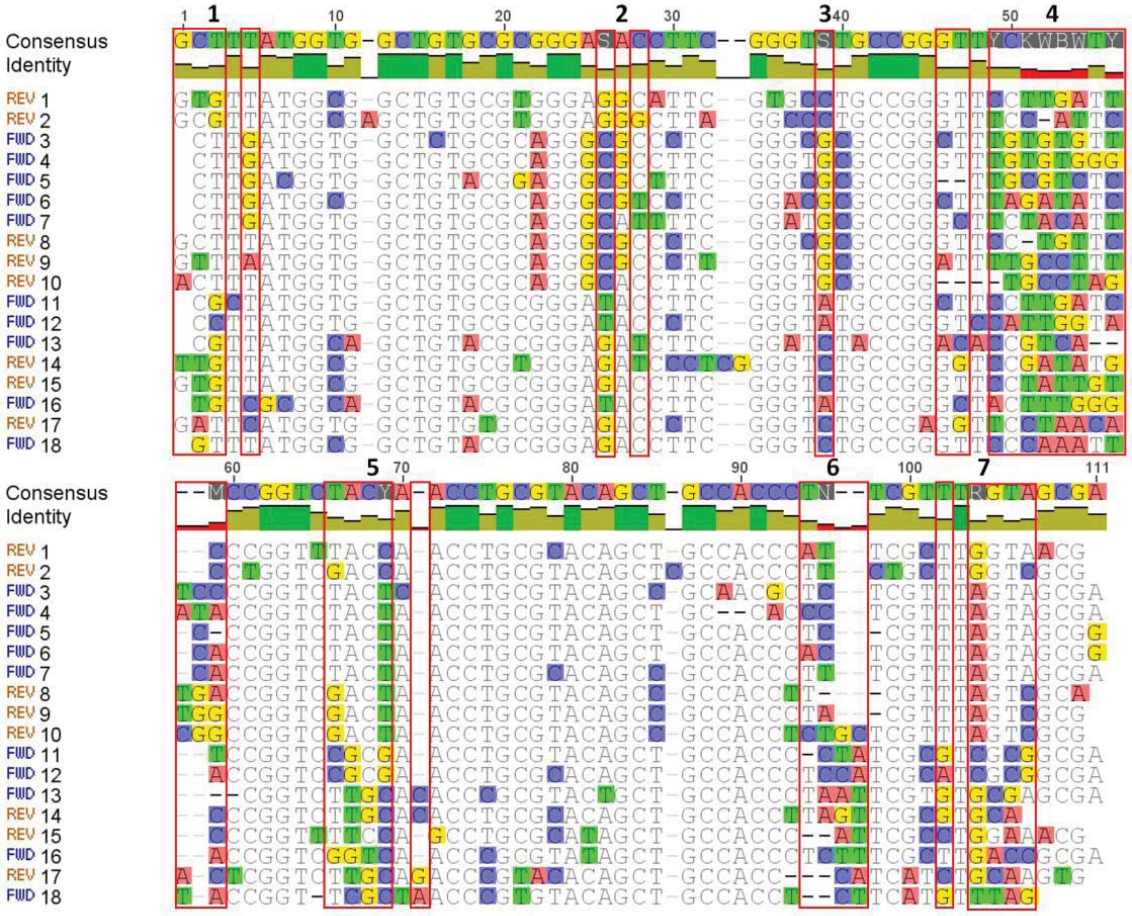


Figure 6.2. ClustalW2 [120] alignment of R178 sequences. Regions boxed in red show polymorphic sites with entropy of > 1 (see text). Annotations on the left side indicate sequence number and orientation within the genome. First row of the alignment shows consensus sequence. Second row shows conservation. Coloured nucleotides in the alignment differ from the consensus sequence.

6.2.1.3 Relationship between ssDNA secondary structures and the polymorphic regions in the R178 sequence alignment

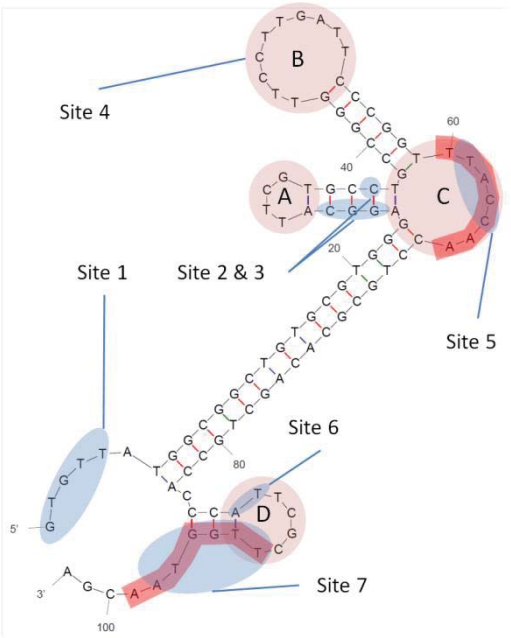


Figure 6.3. Relationship between polymorphisms and ssDNA secondary structure. Detailed characteristics of the predicted DNA secondary structure of R178. Structure contains four loops (A-D) shown in pink. Red region shows similarity between loop C and the end of the repeat. Blue regions correspond to polymorphic regions in the sequence alignment of Figure 6.2.

Figure 6.3 shows that the polymorphic sites identified in Figure 6.2 overlap with distinct features of the ssDNA secondary structure. Figure 6.3 shows that while polymorphic sites 2 and 3 are found in a minor stem region, sites 1 and 4-6 are found in loop regions.

Polymorphic site 1 (5 bp) is found in a loop region at the start of the R178 repeat. It is possible that the evolution of this polymorphic region is governed largely by genetic drift. Polymorphic site 2 (3 bp) corresponds to the stem preceding loop A and is complementary to (and hybridizes with) polymorphic site 3 (1 bp). Unusually, it is not the stem but the single stranded loop that is highly conserved. This may indicate that loop A acts as a recognition site for a conserved protein or complex. At 13 bp, site 4 is the longest polymorphic region and corresponds to loop B (see section 6.2.1.4). A polymorphic loop could indicate two different things: (1) evolution of the loop region is determined largely by drift and selection acts only to preserve the size of the loop or (2) the sequence in the loop co-evolves with a complementary sequence (binding site) in a different part of the genome. Site 5 (4 bp) corresponds to loop C, and is partially complementary to site 7 (but the two sites do not hybridize, which may indicate the existence of a less stable alternative structure, see section 6.2.1.5). Site 6 (2 bp) corresponds to the beginning of loop D, while site 7 (6 bp) corresponds to the 3′ end of the R178 repeat (see section 6.2.1.5). Furthermore the nucleotide alignment in Figure 6.2 clearly shows regions where compensatory mutations occur to preserve the secondary structure. Prominent examples are the positions 26 and 39 as well as neighbouring regions (sites 2 and 3) and the

positions 69 and 105 and neighbouring regions (sites 5 and 7). This data also strongly suggests that selection is acting to preserve the secondary structure.

If selection is acting to preserve the secondary structure then it means the secondary structure is functionally significant. The function of the structure could simply be to modulate the frequency of transposition by an unknown transposase. Alternatively and not necessarily mutually exclusive, the function of the element may enhance the persistence of the element within the genome or the genetic region as observed for TA systems [205]. Functional elements within secondary structures are stems that stabilize the structure and single stranded loops that are free to bind other single stranded sequences. It is possible that the loops observed within the R178 secondary structures bind to neighbouring sequences while the DNA is in a single stranded state; that is during transcription, replication or transposition.

If the structure in single stranded DNA is important during transposition it has been shown that the efficiency of transposition depends on whether the template is present on the leading or lagging strand [206]. If this is also true for R178 repeats one could imagine finding a skew in R178 distribution. However, analyses show that seven R178 repeats are found on the leading strand and 11 on the lagging strand. If it is equally likely to move R178 into the leading or lagging strand then the probability of observing the achieved distribution is about 24%. Hence it seems unlikely that transposition of R178 is affected by leading/lagging strand dynamics.

Alternatively, despite the seemingly lower conservation of the RNA secondary structure it is possible that the functional role of R178 is performed within RNA rather than ssDNA as observed for type I TA systems. Hence detailed analyses of the observed structural elements are performed in the following sections.

6.2.1.4 The binding sites of loop B (polymorphic site 3)

The polymorphisms observed in loop B could be the result of at least two evolutionary processes; they could either be generated by random genetic drift, or they could be shaped by selection (or a combination of both). Neutral evolution could be the main driver of sequence diversity if, for example, the function of the loop is entirely structural and selection acts only on maintaining its length. However, if the loop functions to bind a target DNA or RNA sequence (to *e.g.* inhibit or enhance ssDNA transcription or RNA

translation), then diversity would be driven by selection to match the target sequence. Diversity could also be driven by both neutral evolution and selection if the function of the two sites is linked. For example, if one site acquires a neutral mutation (drift) a compensatory mutation is subsequently selected for in the corresponding site (selection).

Figure 6.4. A typical R178 repeat and surrounding regions in the SBW25 genome. In most cases, an R30 repeat is found upstream of R178, while the downstream region encodes a short peptide. Red lines indicate the sequence of loop B (Figure 6.3) and its complementary sequences. Green regions indicate loop C and its complementary sequences.

Interestingly, a search for complementary sequences to the R178 B loop (between 7 and 13 bp long) in the vicinity (~1kb in either direction) of the R178 repeat showed that in 15 of the 18 cases, the B loop sequence matched to a sequence immediately downstream of the R178 repeat. Each of these 15 sequences was found on the template strand of the promoter of a short gene (each encoding a peptide of ~100 residues; Figure 6.4 and Table A3.2). That this arrangement occurred by chance is improbably small. This indicates some functional significance. For example, if loop B binds to the promoter of a short peptide then this could change the availability of the promoter for transcription. This raises the possibility that the R178 repeat has an impact on the expression of the downstream peptide.

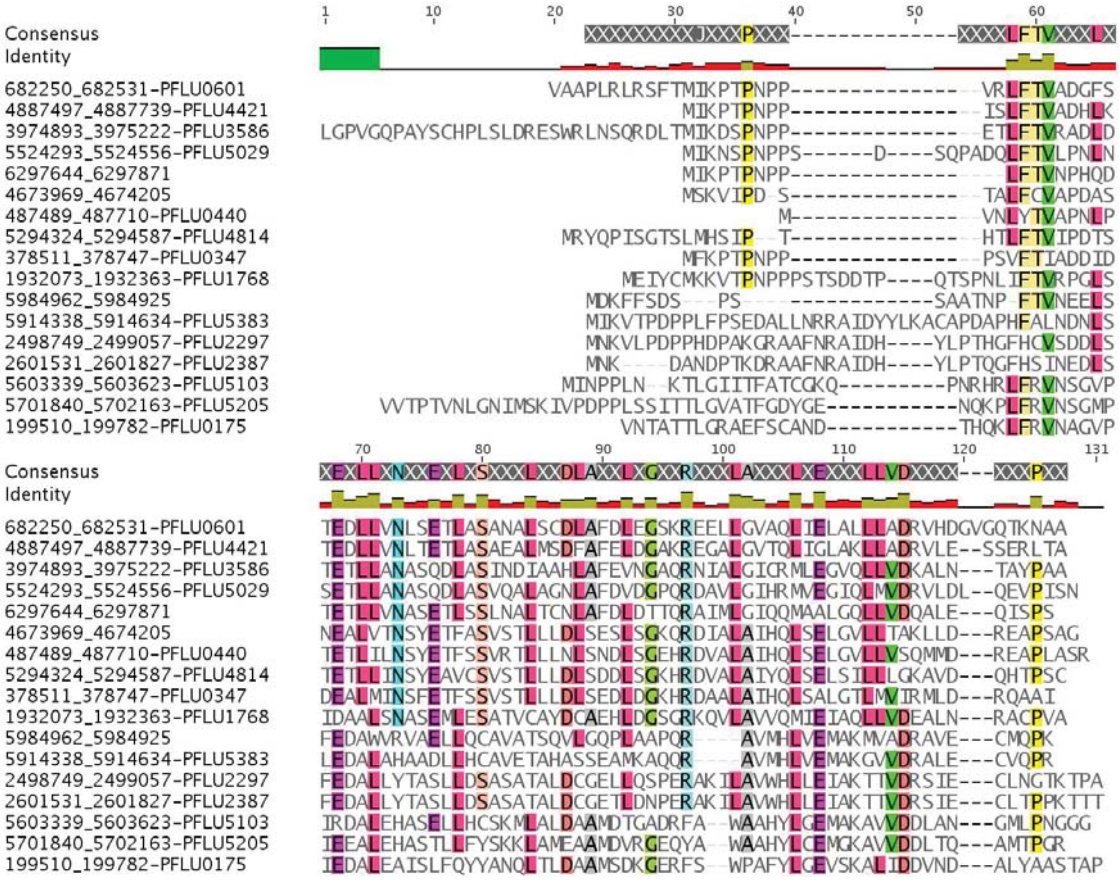


Figure 6.5. Alignment of peptides found at the 3′ end of R178 repeats. Coloured amino acids are conserved in more than 50% of the sequences. Name consists of position and locus tag if available.

Peptides were found downstream of 17 of the 18 R178 repeats, for 15 of which complementarity was observed between the promoter region and the loop B region of the R178 repeat. The 17 peptides appear to be highly diverse, ranging in length from about 73 to 102 residues (Figure 6.5). The diversity poses an obvious question as to whether the peptides evolved independently or share a recent common ancestor – and the related question of whether each independently co-opted its R178 repeat. The hypothesis that all the peptides share a recent common ancestor is supported by a number of conserved sites present in the amino acid sequence alignment of Figure 6.5. To further support this hypothesis, a more comprehensive analysis of the downstream peptides was performed. For all possible 136 pairwise peptide combinations, the pairwise identity was calculated, and the resulting data was displayed as a graph in Figure 6.6. The figure shows each peptide represented as a node, and pairwise identities above 28% represented as lines connecting the two peptides in question (see Figure 5.1 for a more detailed explanation of this graphing technique). As can be seen in Figure 6.6, all 17 peptides are connected to each other, suggesting that they all diverged from a

recent common ancestor and therefore may have a common function. Within the *P. fluorescens* SBW25 genome, homologues of the peptides are almost exclusively found downstream of the R178 repeat, except for one instance where a peptide is part of *pflu3894*, a conserved hypothetical protein.

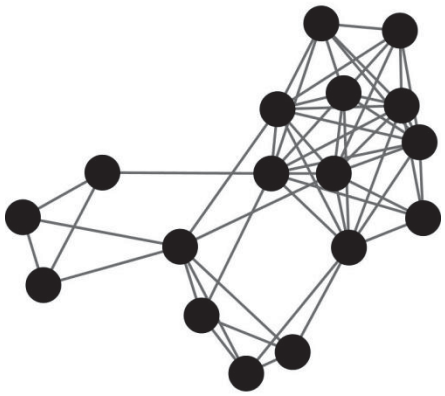


Figure 6.6. Relationship between peptides encoded directly downstream of R178 repeats. Each node represents a peptide. Edges are drawn if the pairwise identity between peptides is greater than 28%.

In conclusion, there are two adjacently located sequence groups one encoding for a peptide and one potentially for ssDNA that is predicted to form a conserved secondary structure. The two sequence groups are linked by a short oligomer that matches the B loop of the R178 repeat and the complement of the promoter of the downstream peptide. These data suggest that the function of R178 and the downstream peptide are tightly linked. The connection between the two genetic elements may have resulted in the spread of R178 and the downstream peptide as a single unit. This hypothesis is supported by the high degree of congruence between the phylogenetic trees of R178 and the downstream peptide (Figure 6.7A).

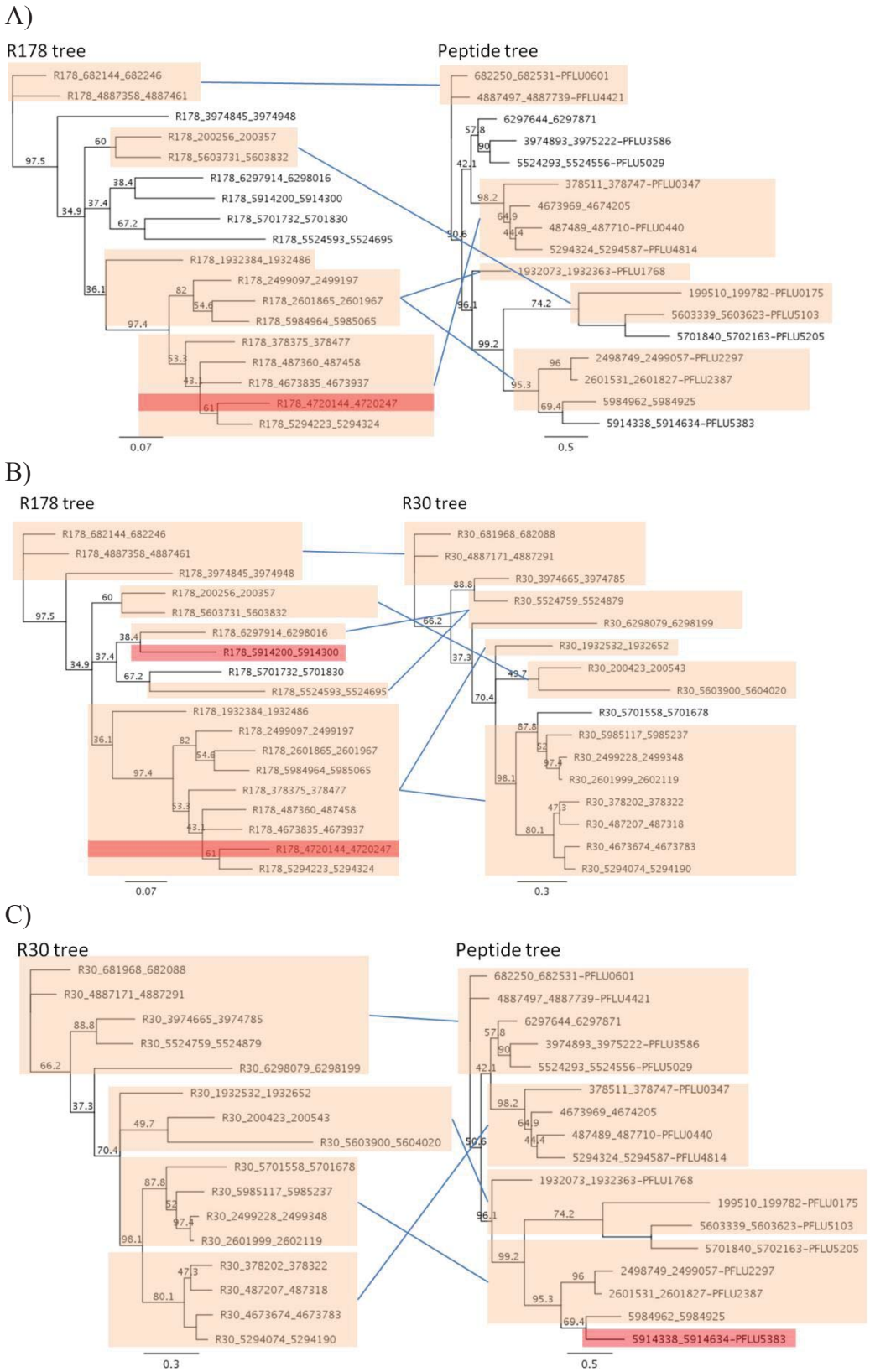


Figure 6.7. Congruence between the phylogenetic trees of the genetic elements found in the vicinity of R178 repeats. Clades that are boxed in orange show congruent phylogeny. Clades boxed in dark pink are not found in the corresponding tree. Phylogenetic trees were constructed based on ClustalW2 [120] alignments and by applying the neighbour-joining [121] method. Trees were re-sampled 1000 times (bootstrap method) and displayed in Geneious [119].

6.2.1.5 The binding sites of loop C (polymorphic sites 4 and 6)

As mentioned above, the sequence of loop C is complementary to the 3′ end of the R178 repeat. In 16 of the 18 R178 repeats, the sequence also matches to a DNA region ~50 bp upstream of R178. Of the 16 matching regions four instances overlap with the 5′ end of the R30 repeat (another class of repeat identified by Silby et al [100] (Table A3.2)). All four R30 repeats that were identified in the SBW25 genome are located upstream of an R178 repeat. This raises the possibility that R30 repeats are associated with all R178 repeats but that the sequence conservation of the remaining 12 possible R30 repeats was too low to be identified by the repeat finder applied by Silby et al. [100]. Armed with knowledge about both the size of the R30 repeat and the position of the loop C binding site within the R30 repeat, it is possible to identify less conserved R30 repeat regions upstream of 12 of the remaining 14 R178 repeats. The four R30 repeats and the newly identified 12 R30 repeat regions were aligned and a phylogenetic tree was built (Figure 6.7B and C). The R30 sequence alignment shows high sequence diversity as well as a few conserved regions. The phylogenetic tree built from R30 sequences is similar to the phylogenetic trees of both R178 repeats and the downstream peptide (Figure 6.7). This suggests that R30 is yet another part of the genetic unit comprised of R178 and the downstream peptide.

The ssDNA and RNA secondary structures formed by each R30 repeat were also predicted; however no conserved structure could be identified, indicating that, unlike R178 repeats, the function that led to the sequence preservation of the R30 repeat is not a consequence of its secondary structure formed in ssDNA or RNA. Hence, the information upon which selection acts, is likely to be encoded in the nucleotide sequences alone, similar to gene promoters.

6.2.1.6 The R178 composite genetic element

The above analyses suggest that the complete genetic element comprises not only R178, but also two flanking genetic elements: an R30 repeat and a peptide. These three elements are linked through the central R178 repeat. The predicted ssDNA (and to a degree also RNA) structure of the R178 repeat contains short oligomers (up to 13 bp long) in two loop structures (loop B and C). The loop B and C sequences are complementary to the promoter of the downstream peptide and the 5′ end of the R30

repeat, respectively. This indicates that the function of the three genetic elements is linked.

6.2.2 R200 repeat sequences

In the genome of SBW25, 47 R200 repeats have been identified [100]. They range in size from 129 bp to 380 bp, with only a 110 bp region shared by all R200 repeats (Figure 6.8). The large range in sequence length indicates that in some cases part of the repeat may have been lost by decay.

The analysis of R200 repeats can be approached in a similar as the analysis of R178 repeats. Applying the same argumentation as in section 6.2.1.1 it seems unlikely that R200 repeats arose by chance or as a result of similar selective pressures in different genetic backgrounds (no R200 repeats are found in the closely related *P. fluorescens* Pf0-1 strain). Hence it seems likely that R200 repeats are the result of a duplicative process. Again, one can ask the question whether R200 sequences are transposed autonomously or non-autonomously. If they are transposed autonomously then one would find a conserved open reading frame (transposase) or alternatively if a catalytically active RNA molecule is encoded a conserved RNA secondary structure. There is no evidence of a conserved open reading frame across R200 repeats, however, the consensus sequence over 380 bp is predicted to form a conserved secondary structure RNA using the mfold web server [104] (Figure 6.9). This raises the possibility that R200 repeats are amplified by a catalytically active RNA. However, alternative possibilities are possible. For example the Ibs/Sib type I TA family that was discovered in *E. coli* MG1655 encodes a toxin and a RNA antitoxin and is found in up to five copies within the genome [73]. This indicates that despite being repetitive and encoding a conserved protein as well as a conserved RNA the TA system was transposed *in trans*. Hence to test an alternative hypothesis, the R200 sequence and conserved RNA secondary structure were compared to type I TA systems.

One group of type I TA systems contains a conserved CCAG motif (indicated by a red arrow in Figure 6.7); conserved in 40 of the 47 R200 repeats, this motif is also found in three *E. coli* type I TA systems: TisB/IstR-1, ShoB/OhsC and SymE/SymR [73]. Notably, the CCAG motif is located in a stem region in the predicted 380 bp R200

secondary structure (red arrow in Figure 6.9). However, if the RNA secondary structure is predicted for only a part of the R200 repeat, so that the position of the CCAG motif aligns with that in the IstR-1, OhsC and SymR antitoxins, the resulting secondary structure not only resembles that of the type I TA systems but the conserved CCAG motif is located in a loop region (Figure 6.10 and [73]). This suggests that the segment of the R200 repeats that show homology to the three *E. coli* type I TA antitoxins may also encode an antitoxin (Figure 6.9 green region).

Figure 6.8. DNA sequence alignment of all R200 repeats found in the genome of SBW25. Only a 110 bp region is present in all R200 sequences. Coloured nucleotides are polymorphic regions that occur in less than 50% of the sequences. Sequences were aligned using ClustalW2 [120].

If part of the R200 repeat indeed encodes an antitoxic RNA structure, then one might reasonably expect a nearby open reading frame (ORF) to encode a corresponding toxin. Indeed, only a few nucleotides downstream of the putative antitoxin, a short highly conserved ORF is encoded. Although this ORF commences with a conserved methionine, it has not previously been annotated in SBW25. This is likely to be due to the ORF's short length of only 126 nucleotides.

The toxic peptides TisB and ShoB are predicted to form a short transmembrane helix. A short transmembrane helix is also predicted by TMpred [124] within the peptide encoded by the R200 repeat sequence (Figure 6.11). However, the TMpred score of the putative transmembrane helix – which positively correlates with the likelihood that the peptide contains a transmembrane helix - is relatively low; the 42 residue long R200 consensus protein sequence received a transmembrane helix score of 256 (a score of 500 is considered significant). Nevertheless, the substitution of only one amino acid - a polar arginine with an aliphatic isoleucine (R28L; highlighted in red in Figure 6.9) - results in a significant score (1141) similar to that of the TisB toxin (1302), the corresponding toxin to the IstR-1 antitoxin in *E. coli*. It is possible that this change is an adaptive response to the amplification of the R200 repeat; leaky expression of one TA system probably does not affect the organisms' fitness whereas leaky expression of 47 intact toxins may seriously impair growth.

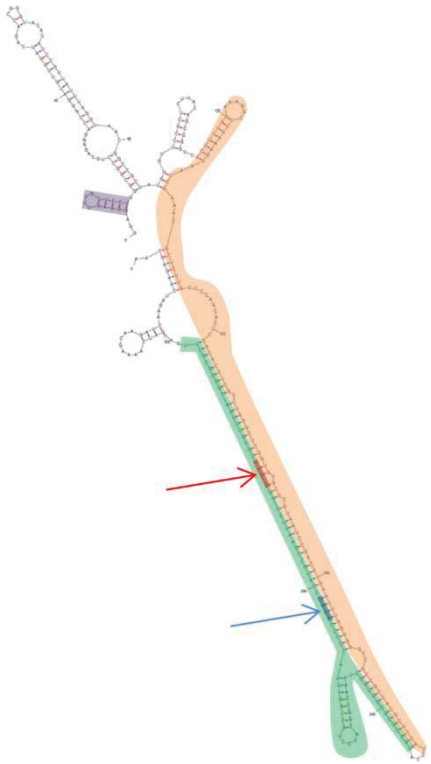


Figure 6.9. RNA secondary structure prediction of R200 consensus sequence. The sequence in orange encodes for putative toxic protein. The sequence in green forms a putative antitoxin shown in Figure 6.10. The purple box indicates a GIII REP sequence. Red and blue arrows indicate the CCAG and AAAU motifs respectively. Both sequences are bound to a complementary region. In contrast, when predicting the RNA secondary structure of the sequence underlined in green both sequences are found as part of a loop (see also Figure 6.10). RNA secondary structure prediction was performed using the mfold web server [104].

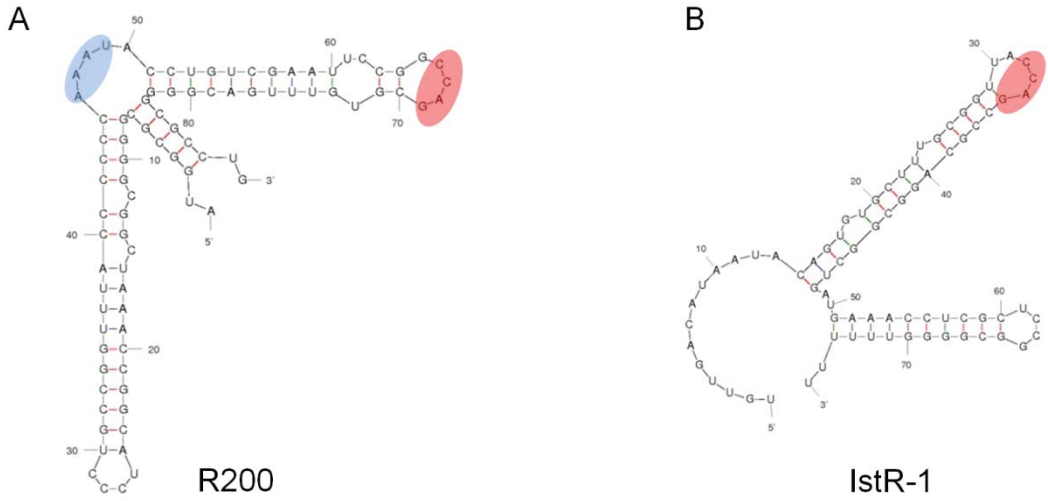


Figure 6.10. RNA secondary structure prediction for part of a R200 sequence as well as the antitoxin IstR-1. (*A*) The RNA secondary structure was predicted from part of an R200 sequence found at position 1,720,610 in the SBW25 genome. The motif in red is conserved as part of a loop in IstR-1, SymR and OhsC antitoxins [73]. Note that the AAAU (blue) motif present in the loop is bound to a complement when predicting the whole R200 secondary structure (Figure 6.9) (*B*) The IstR-1 sequence was extracted from the genome of *E. coli* O111:H str. 11128. The secondary structure predicted for IstR-1 is similar to the one predicted for the partial R200 sequence. RNA secondary structure predictions were performed using the mfold web server [104].

Antitoxins such as IstR-1 are predicted to repress the expression of the toxin by competing with the ribosome for the ribosome binding site [73]: when IstR-1 binds to the TisB (toxin) mRNA, the mRNA is cleaved by RNase III, while binding of the ribosome leads to mRNA translation and thus expression of toxic TisB. A similar mechanism could lead to inhibition of R200 toxin expression. Figure 6.9 shows that the putative R200 antitoxin (Figure 6.10A) is found immediately upstream of the putative R200 toxin. The putative R200 toxin ribosome binding site in RNA is complementary to and hence can be bound by the putative R200 antitoxin. Therefore, similar to the TisB/IstR-1 mechanism of inhibition, competition between the antitoxic RNA and the ribosome for the ribosome binding site could feasibly prevent the translation of the R200 toxin.

MARGGGPPNSCRITGTPSLSEVPSVGARAFWLLLGLSKSDPL

Figure 6.11. Consensus sequence of putative toxin found in R200 repeats. The putative toxin contains a predicted low scoring (257, below 500 transmembrane status is considered unsure) transmembrane helix (in grey, http://www.ch.embnet.org/software/TMPRED_form.html [124]). However, when the arginine (red) is replaced with an isoleucine (codon change from AGA to ATA) the score becomes highly significant (1141), as is the case for toxins such as TisB or ShoB from *E. coli*.

6.2.3 Association between R200 repeats and REPs/REPINs

Within the SBW25 genome, R200 repeats are frequently co-localized with REPs, REPINs or tandemly repeated REPs and REPINs; Figure 6.12 shows that only three out of 47 R200 repeats are unlinked to REPs or REPINs. Most strikingly, all 11 GI-GIII REP tandem repeats (see section 3.2.4.2) are linked at the 3′ end to an R200 repeat. Furthermore, 20 R200 repeats were found as doublets, 14 of which form inverted repeats together with a central REPIN (six occurrences) or REPIN doublet (one occurrence). The remaining six R200 doublets are found as tandem repeats flanking tandemly repeated REP sequences. That these arrangements are the result of chance will be considered below.

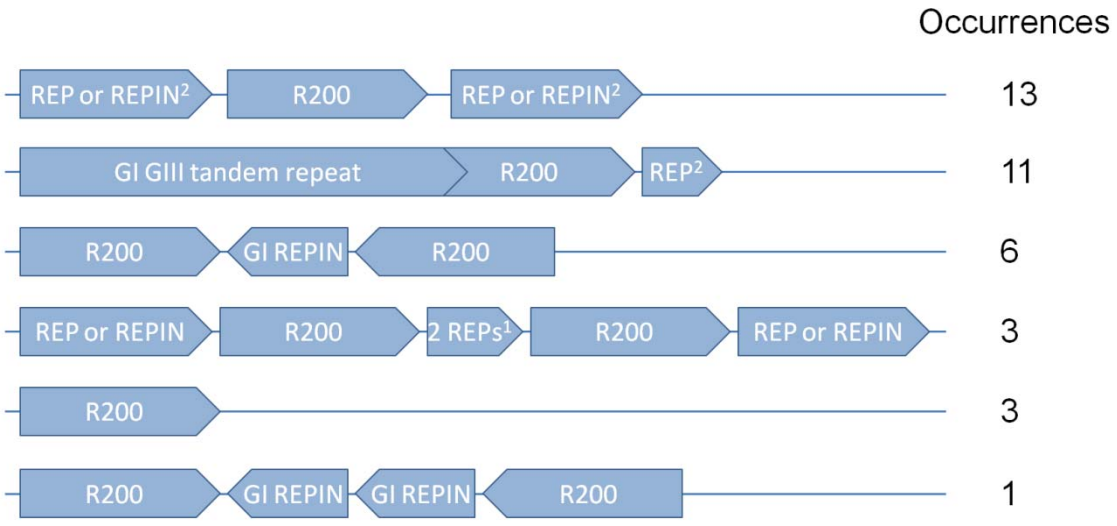


Figure 6.12. R200 and REPs/REPINs in the genome of SBW25. There is a total of 47 R200 repeats in the SBW25 genome. 44 of 47 R200 repeats are linked to REPs or REPINs. [1]Two REP sequences in the same orientation. [2]Zero or more occurrences.

The one-to-one association between 11 GI-GIII tandem repeats and 11 R200 sequences is so conserved that the repeat recognition program applied by Silby et al. [100] recognized a GIII sequence as part of the R200 repeat (see purple box in Figure 6.9). Interestingly, the location of the GIII REP in the R200 repeat is not only conserved for the 11 R200 repeats associated with GI-GIII tandem repeats but also for four tandem doublets and ten R200 singlets linked to REPs. The two remaining tandem doublets are linked to GII REPs instead of GIII REPs. Another argument against this being a chance result is that all inverted R200 doublets flank a REPIN or tandem REPIN (inverted REP

sequences), whereas all six tandem doublets flank tandemly repeated REP sequences. If the association between the R200 and the different REP structures is not due to chance, it is possible that the association has an impact on R200 sequence evolution. To investigate this possibility, the pairwise identity of the R200 sequences in each of the different REP/REPIN backgrounds was determined (Figure 6.13). R200 sequences that are found near GI-GIII REP tandem repeats are the most conserved. Such high conservation could be the result of recent amplification of R200 repeats in the context of tandemly repeated GI and GIII sequences. Alternatively, low sequence diversity could be due to frequent recombination events, a possibility that will be discussed further in section 6.3.4. The second most highly conserved sequence group are R200 doublets flanked by tandemly repeated REP sequences. These tandem repeats are likely the result of recent local amplification, which is supported by the phylogenetic tree in Figure 6.14.

Not only does the sequence diversity of R200 repeats differ depending on the association with REPs/REPINs, but it appears that the underlying evolutionary process that produced the current set of R200 repeats also varies. This hypothesis is supported by an analysis of the different REP/REPIN-dependent R200 phylogenies. R200 repeats of each association group (Figure 6.12) were aligned and for each alignment a

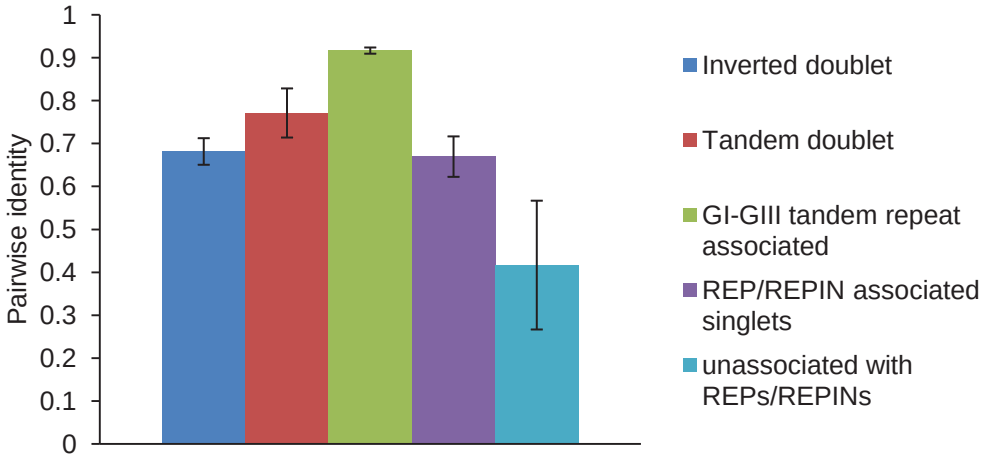


Figure 6.13. Average pairwise identities of R200 sequences in each of the 'association' groups from Figure 6.12. Error bars show one standard deviation. The differences between R200 repeats found associated with GI-GIII tandem repeats and all other groups (except for R200 found as tandem doublets) are significant. The difference between the pairwise identities of R200 repeats found as tandem doublets and R200 repeats with no association to REPs is significant. Differences are considered significant if all means acquired by sampling without replacement exceed the maximum mean of the group of comparison.

neighbour-joining tree was calculated. Interestingly, the phylogenetic trees show three different topologies. This indicates that three different processes shape the R200 sequence evolution.

For R200 sequences found in inverted doublets and R200 singlets localized near REPs/REPINs the same topology was observed (Figure 6.14, blue box). Both phylogenetic trees are reminiscent of trees that are produced by the master copy model [207-209] – a model that assumes only one sequence copy can actively spread and that new copies are immobile. The same tree topology could be produced by a process where old copies are rendered immobile and only the newest can spread.

In contrast, the phylogenetic tree for tandemly repeated R200 sequences strongly suggests that the repeats are formed by local amplification (the most closely related R200 sequences are found in the same tandem repeat). This is similar to observations from tandemly repeated REPINs (see section 3.2.4).

The most conserved class of R200 repeats (linked to GI-GIII tandem repeats) shows the highest level of phylogenetic uncertainty (more than two branch points at a certain level of the phylogenetic tree). This uncertainty could be explained by rapid repeat expansion. If no mutations occur during the entire amplification process, determination of the precise phylogeny would be impossible. Alternatively, frequent recombination between the different repeats could lead to a similar result.

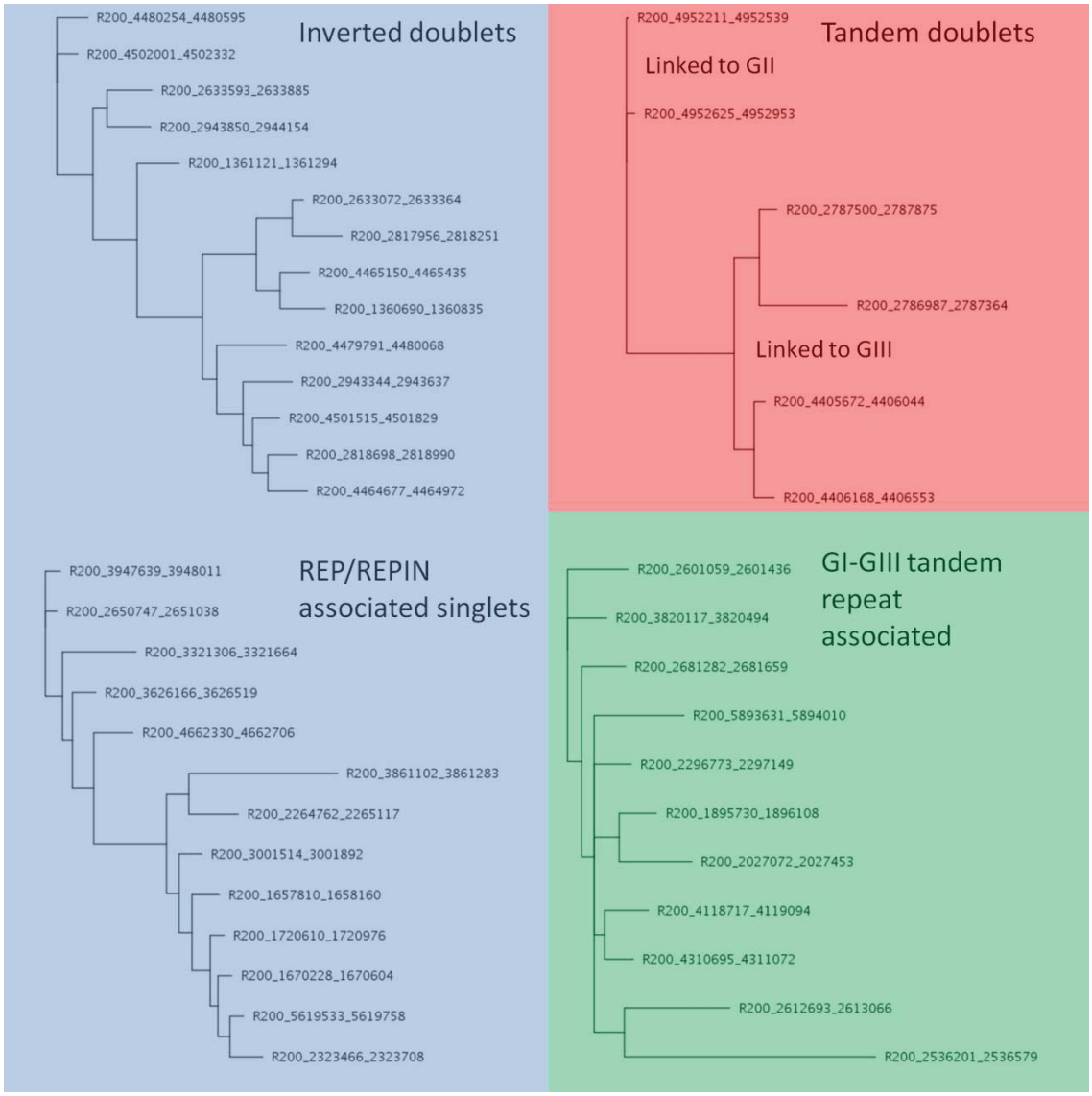


Figure 6.14. Neighbour joining trees for different R200 groups. The phylogenetic trees in blue suggest that either a single R200 copy amplifies, or only the newly formed copy can amplify. The phylogenetic tree in red supports the duplication of R200 repeats within the repeat. The main branch point divides R200 sequences into sequences linked to GII and sequences linked to GIII REPs. The high uncertainty observed in the green tree can be a result of two different processes. It can reflect rapid sequence amplification or frequent recombination events. Phylogenetic tree of R200 sequences without link to REPs could not be built due to the low sample size of three. Trees are based on 1000 bootstraps. Branch points are supported by at least 30% of the samples.

6.3 Discussion

6.3.1 Overview of the results

The results show that R178 repeats form a conserved secondary structure and co-evolve with the flanking R30 repeat and an upstream ORF. The arrangement is reminiscent of TA systems. Interestingly, R200 repeats also consist of at least two parts. The first part forms a highly conserved secondary structure and the second encodes a short protein or peptide. Since the secondary structure and the peptide resemble certain TA systems found in *E. coli* it is possible that R200 repeats encode TA systems. Analyses of the association of R200 repeats and REPINs/REPs raise the possibility that R200 repeats take advantage of the REPIN/REP amplification mechanism. This could represent an example of cooperation between chromosomally encoded addictive and duplicative selfish genetic elements.

6.3.2 Cooperation of selfish genetic elements

As introduced in Chapter 1.2, selfish genetic elements are DNA sequences that "*are vertically transmitted genetic entities that manipulate their "host" so as to promote their own spread*" [13]. There are two main classes of selfish genetic elements: (1) duplicative elements that increase their frequency within the population through spread within and between genomes (*e.g.* insertion sequences, see Chapter 1.3), and (2) addictive elements that increase their copy number within the gene pool by killing of cells that do not contain a copy of the gene (e.g. toxin-antitoxin systems, see Chapter 1.4). One way for a selfish genetic element to increase its evolutionary success is to be linked to another selfish element. If both elements benefit from such linkage it can be considered a form of cooperation.

It is not difficult to envisage that the persistence and spread of duplicative and addictive selfish genetic elements could be aided through cooperation. Cooperation (mutual benefit) can be achieved by physical linkage of a duplicative and an addictive element; the addictive element could then be spread by the duplicative element when (under certain circumstances) the duplicative element transposes not only itself but also

flanking DNA (*e.g.* composite transposons [210]). Conversely, maintenance of the duplicative element is bolstered by the presence of the addictive element. For example, if the flanking DNA of a duplicative element harbours an addictive genetic element then it could greatly benefit from the interaction by, for example, being transferred from the genome to a plasmid, which might help the element to spread to other bacterial genomes. Duplicative genetic elements may in turn benefit from the increased stability of the DNA flanking the addictive element (any large scale deletion around the addictive element is prevented since the loss of the addictive element leads to cell death *e.g.* [205]). Hence, the closer the linkage between duplicative and addictive elements, the greater the potential benefit. Indeed, most addictive selfish genetic elements show signatures of frequent horizontal transfers, which indicate cooperation with duplicative elements [181, 182, 211, 212].

Prominent examples of potentially cooperating duplicative and addictive selfish genetic elements include: plasmids (which are themselves a type of duplicative selfish genetic element) that contain TA systems or bacteriocins, where the association increases the plasmid's persistence within the host bacterium as well as increasing the competitive advantage of the host plasmid to other plasmids [22, 213]; composite transposons (cooperative systems comprised of two insertion sequences and a cassette of other selfish genetic elements, such as antibiotic resistance genes) [214]; and phages (which may also be considered a type of duplicative selfish genetic element) containing TA systems to prevent co-infection by other phage [215]. Although there are a plethora of duplicative and addictive selfish genetic elements localized on bacterial chromosomes, documented examples of cooperation between chromosomal elements are rare. Rather than reflecting a lack of cooperative interactions; this may be due to a lack of research in this area. For both R200 and to a lesser degree R178 repeats REPINs were observed directly flanking the repeats. It is possible that this co-localization reflects some kind of co-operation between the genetic elements.

6.3.3 R178 repeats

As mentioned above, R178 and R30 are likely to be part of the same genetic element together with a short protein coding sequence. The functional significance of each of

these elements is unclear but is linked through the central R178 repeat. Motifs within the repeat are complementary to both the putative promoter of the protein coding region and the beginning of the R30 repeat. This raises the possibility that R178 mediates the expression of the protein coding region. In turn, the expression of R178 could be regulated by motifs found in R30 (*e.g.* R30 as promoter region). However, to shed light on this issue, and to determine the functional significance of R30, experimental analyses are required.

Nevertheless, it is possible to propose a hypothesis regarding the function(s) of R178 and the associated peptide. The composition of the genetic element is reminiscent of chromosomally encoded RNA (type I) TA systems (see section 1.4.1.1). If the element encodes a TA system then R178 probably encodes an antitoxin that tightly regulates the expression of the associated toxic peptide. This hypothesis is supported by the presence of a binding site of the R178 loop B immediately upstream of the associated peptide's promoter. The co-evolution between R178 and the downstream peptide further supports the notion of strongly linked functions. It is possible that the R178 secondary structure forms during transcription and by binding to the nearby ribosome binding site prevents the binding of the ribosome and therefore the translation of the downstream peptide gene. Conversely it is possible that translation of the downstream peptide gene is activated when the R178 loop C binds to a motif found at the 5′ end of the R30 repeat and hence makes it impossible for loop B to bind the ribosome binding site of the peptide.

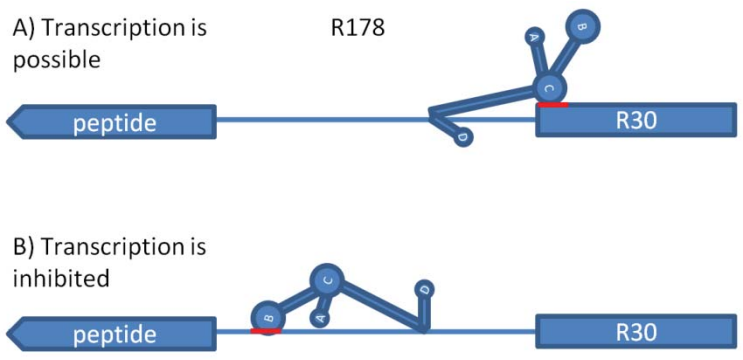


Figure 6.15. Proposed mechanism for the transcriptional regulation of a peptide through the R178 repeat. (*A*) The promoter of the peptide is available for transcription when loop C is bound to the R30 repeat upstream of R178. (*B*) The transcription of the peptide is repressed when loop B is bound to the promoter of the peptide.

REPs, REPINs and higher order REPIN arrangements were observed directly flanking the putative TA system in ten out of 18 cases. A possible explanation for this association is that in some instances the mechanism leading to the dispersal of REPINs

also increases the copy number of R178 genetic elements. However, 28 R178 repeats are found in *P. fluorescens* Pf0-1, a closely related strain, which does not contain REPs or REPINs. One can imagine that spread in SBW25 and Pf0-1 was the result of cooperation between the R178 sequence and a range of duplicative elements such as insertion sequences (see section 6.1). Evidence of this cooperation could have been lost through excision mediated by the encoded transposase or a simple deletion event. Alternatively, REPINs could have an effect on the expression of the system by, for example, enhancing mRNA half-life [217]. If this hypothesis is correct, then these associations represent an example of weak cooperation between two chromosomally encoded selfish genetic elements.

6.3.4 R200 repeats

R200 repeats consist of a DNA region that is predicted to encode a highly conserved non-coding RNA and a short protein coding sequence. Interestingly, both the predicted secondary structure of part of the R200 sequence and the encoded protein (peptide) show similarities to type I TA systems, such as TisB/IstR1 in *E. coli* [73]. Hence, it is possible that the expression of the putative R200 toxin is regulated in a similar manner. IstR-1 binds to the Shine-Dalgarno sequence of TisB encoding mRNA, thereby preventing TisB translation. This regulatory mechanism might also control the expression of the R200 system, based on the sequence similarities between putative toxin and antitoxin. However, the similarities between the putative R200 toxin mRNA and its corresponding antitoxin are more extensive than for TisB/IstR-1. The length of the complementary sequence between TisB mRNA and IstR-1 RNA is 23 nucleotides [73] compared to a total of about 60 complementary nucleotides for putative R200 toxin and antitoxin (Figure 6.9). This long complementarity may allow an even tighter repression of the translation of the putative toxin's mRNA by the putative R200 antitoxin.

The putative R200 toxin is short and predicted to contain a transmembrane helix. However, the score of the prediction is considered insignificant and much lower than that predicted for TisB (the transmembrane toxin associated with IstR-1). Notably, the substitution of an arginine by an isoleucine, a change that requires the mutation of only

a single nucleotide, changes the score of the prediction to a highly significant value (Figure 6.11). One could imagine that this change is the (adaptive) result of R200 sequence amplification. Amplification of a TA system could lead to detrimental effects to the organism as a result of leaky expression. This may have caused a single base change to alleviate the effect. However, this raises a new question: if the putative toxin is no longer toxic, then what is the function of the putative non-toxic "toxin-antitoxin system"? It is possible that the peptide is still toxic but needs to be present in much greater concentrations to cause cell death.

6.3.5 Association between R200 repeats and REP/REPIN structures

The association between R200 repeats and REPs/REPINs is more pronounced than that observed for R178 repeats. Only a small proportion of all R200 repeats (three out of 47) show no association with REPs or REPINs. Interestingly, R200 repeats are not present in Pf0-1, which contains neither REPs/REPINs nor RAYTs, but are present in Pf-5, which contains a (potentially inactive) RAYT copy and a large number of REPs/REPINs [100]. In the vicinity of different REP/REPIN structures, R200 repeats show different properties. Tandem repeats of R200 sequences flank tandem repeats of REPs; inverted R200 doublets flank REPINs (inverted REP repeats). R200 sequences even show different phylogenies depending on the type of association with REPs/REPINs (Figure 6.14). For example, tandemly repeated GI-GIII REPs, and their associated R200 sequences, are highly conserved. This could be the result of rapid repeat expansion or frequent recombination events between R200 sequences. Such recombination events could potentially occur during replication. When the two replication forks commence DNA replication from the origin of replication, they could switch template strands at two inverted R200 repeats associated with GI-GIII tandem repeats. This would lead to gene conversion as well as reversing the orientation of the intervening DNA. The similar distance of GI-GIII tandem repeats to the origin of replication (*e.g.* the two largest GI-GIII tandem repeats are found at position ~2.5 Mbp and ~4.2 Mbp in the SBW25 genome and are both located at a distance of ~2.5 Mbp from the origin of replication given that the SBW25 genome is ~6.7 Mbp long) could increase the chances of such recombination events since the two replication forks arrive at the repeat at approximately the same time.

The results presented indicate strong cooperation between the REP/REPIN system and the putative R200 toxin-antitoxin system. It seems likely that R200 repeats are copied by the REP/REPIN system. Given that the function of R200 repeats remains unknown (although R200 repeats possess TA characteristics), the benefits for REPs/REPINs are harder to infer. However, assuming that R200 repeats encode an addictive selfish genetic element, and based on knowledge of other cooperative associations between duplicative and addictive selfish genetic elements, it seems likely that the R200 sequence aids the persistence of the REP/REPIN system within the genome.

6.3.6 Concluding comments

TA systems are a present in most bacterial genomes [212, 218]. Their evolutionary success is not only a result of their addictive properties, but probably also by the host's ability to co-opt them for a diverse range of cellular functions, such as adaptation to nutritional stress or the production of persister cells through arresting cell growth [78, 219, 220]. Considering the wide range of potential host functions for which TA systems can be co-opted, as well as their wide spread throughout bacterial genomes, it is important to identify them, and understand their evolution as well as their function. Here the first steps in identifying and characterizing their evolution have been performed. Furthermore this study suggests that TA systems may enhance their evolutionary success by cooperating with other selfish genetic elements.

Finally, the study presented here shows that although the cooperation between the R178 system and REPs/REPINs is at best weak, cooperation between R200 repeats and REPs/REPINs is more apparent and is possibly involved in amplification of R200 repeats. Examples of cooperation between chromosomally encoded duplicative and addictive selfish genetic elements may be greater than currently appreciated. However, questions concerning the cause and effect of this interaction remain mainly unanswered and require further investigation.

Chapter 7:
Discussion

7.1 Overview of the results

7.1.1 Summary of Chapter 3: Within-genome evolution of REPINs

Evolutionary analyses of short repetitive sequences in the genome of *P. fluorescens* SBW25 marked the start of this thesis. The initial aim of this study was to provide an unbiased analysis of short, repetitive sequences in the SBW25 genome. All short sequences (10-20 bp) that occurred at frequencies above a certain threshold were selected for further analyses. The threshold was determined through comparisons to short sequence frequencies obtained from randomly assembled genomes and subsequently from the genome of the closely related strain *P. fluorescens* Pf0-1. These comparisons led to the conclusion that the short sequences selected for further analysis were shaped by selection and did not simply arise by chance.

Interestingly, the 96 different short sequences of sequences selected for further analysis could each be categorized into one of three groups (GI, GII or GIII, see section 2.2.4 and 3.2.1). Sequences from each of the three sequence groups were found to be repetitive, palindromic and predominantly extragenic, and were therefore labelled REP sequences.

Analyses of next-neighbour distances showed that the majority of the sequences within the individual groups occur at specific distances from one another. This observation strongly deviates from what is expected under a random model, which predicts that it is unlikely to observe REP sequences that share the same next-neighbour distance. This led to the hypothesis that REPs are part of a larger genetic element, consisting of two or more REP sequences separated by a spacer of a specific length. The number of REP sequences that are involved in the formation of the new genetic element was determined by analysing higher order arrangements of REP sequences (formation of REP clusters).

The REP sequence cluster data obtained from SBW25 was compared to expectations from a randomly generated null model. Based on the null model, two thirds of all REP sequences are expected to occur as singlets. Only one third of all REP sequences are found as singlets in the SBW25 genome, but two thirds were found as doublets. Hence, REP doublets were proposed to be the main replicative unit.

This hypothesis was tested and confirmed various ways. First, the distribution of REP doublets in the genome of SBW25 is comparable to what is expected under a randomly generated null model. In contrast to the distribution of singlets does not conform to a random model. Second, REP sequences found as part of REP doublets show a higher level of DNA sequence conservation than REP sequences found as singlets. This suggests that REP doublets are under selection (and therefore functional) as opposed to singlets, which are probably non-functional remnants of REP doublets. Third, evidence of REP doublet excisions was observed in SBW25 whole genome sequencing data, while no evidence of REP singlet excisions was found. This finding not only supports the hypothesis that the REP doublet is an individual genetic element, but also indicates that REP doublets are actively moving in the SBW25 genome. The sequence of the excision events enabled a hypothesis to be formed regarding a possible transposition mechanism. Hence, REP doublets are a new class of mobile bacterial DNA, which was named REP doublets forming hairpins (REPINs).

In addition to REPINs, other higher order REP arrangements were observed above the frequencies that would be expected by chance. These were either highly organized, tandemly repeated REPINs or tandemly repeated REP sequences. The evolutionary and functional significance of such structures remains unclear.

7.1.2 Summary of Chapter 4: Cause of within-genome REPIN dispersal

In 2010, Nunvar et al. [101] proposed that RAYTs (REP-associated tyrosine transposases) are the cause for REP sequence dispersal in bacterial genomes. This conclusion was reached in parallel during the course of the research in this thesis. Given that a publication about the association between REPs and RAYTs already exists, the chapter about the cause for REPIN dispersal within bacterial genomes was kept relatively short and focused on points not covered by Nunvar et al., such as how the

connection between IS*200* transposases and the very distantly related RAYT family was made, and the importance of REPIN formation for REP sequence dispersal.

To find genes that could be the cause of REP sequence dispersal, REPIN sequence clusters were analysed. Three genes of particular note were found. Each of these was located within a specific cluster of one of the three REPIN groups identified in chapter 3. In the original genome annotation the three genes were predicted to encode conserved hypothetical proteins, and no connection to IS*200* proteins had been made [100]. In an attempt to elucidate the function of these REPIN-associated genes, BLASTP searches were performed, but as expected there were no significant matches to any known gene family (databases are updated now). However, a BLASTP search against an insertion sequence database revealed that the majority of hits were to IS*200* proteins. Further investigations showed that IS*200* proteins share a highly conserved motif with this new class of proteins (named REP-associated tyrosine transposases (RAYTs) by Nunvar et al. [101]).

IS*200* proteins transpose by binding to short palindromes flanking the transposase gene. Since REPINs, like IS*200* genes, contain two flanking palindromes, RAYTs are a likely candidate causative basis for their transposition and dispersal. Further analyses show that each of the three RAYTs discovered in SBW25 is associated to one (and only one) of the three REP sequence classes identified in Chapter 3. This not only further supports the hypothesis that RAYTs are responsible for REPIN dispersal, but also enabled the systematic identification of different REP sequence classes in different bacterial genomes. Once REP sequence classes were identified, cluster analysis could be performed that showed that the formation of REPINs is not only a prerequisite for REPIN dispersal in SBW25, but in all 18 bacterial genomes analysed.

7.1.3 Summary of Chapter 5: Characterization of the RAYT family

In Chapter 5, RAYTs were characterized more comprehensively in order to determine: (1) whether RAYTs have characteristics similar to housekeeping genes or insertion sequences, (2) the relationship between RAYTs and IS*200* sequences, and (3) the composition of the RAYT family. Answering these questions first required the identification of RAYTs in bacterial genomes. Hence, all available, fully-sequenced

bacterial genomes and plasmids were searched *via* BLAST for relatives to a representative RAYT protein from each of *P. fluorescens* SBW25 and *E. coli* K-12. The identified genes were then analysed for characteristics that differentiate insertion sequences from housekeeping genes, such as presence on plasmids and duplication rate.

For comparative purposes, the same features were determined for the IS*200* and IS*110* families of insertion sequences and the housekeeping gene family of peptide deformylases (*def*). Interestingly, for most characteristics RAYTs show more similarity to housekeeping genes than to insertion sequences.

Together with the fact that RAYTs and IS*200* sequences share very low sequence similarity, the above finding led to the question of whether RAYTs and IS*200* sequences share a recent common ancestor or whether the two sequence classes arose through convergent evolution. The pairwise comparison of RAYT and IS*200* sequences showed the existence of hybrids: genes that share significant sequence similarity to both IS*200* and RAYT genes. Although the existence of hybrid genes indicates that a recent common ancestor may have existed, alternative explanations such as emergence through recombination (or even selection/neutral evolution) could not be ruled out.

Closer analysis of the RAYT family showed that RAYTs are not a homogenous gene family; instead, RAYTs were found to form four separate and only distantly related sequence clusters. Each of these clusters was shown to have very unique characteristics. The two most closely related clusters ((a) and (b)) contain the RAYTs from clade I and clade II of the phylogenetic tree built in Chapter 4 (Figure 4.3). RAYTs from these two clusters and cluster (d) show very low duplication frequencies and are not found on plasmids (similar to housekeeping genes). Conversely, RAYTs from cluster (c) are found on plasmids and show relatively high duplication frequencies. Thus, they are more similar to insertion sequences than to housekeeping genes. Interestingly, RAYTs from clusters (a) and (b) are linked to over-represented 16-mers (REPs), while those from clusters (c) and (d) are not.

7.1.4 Summary of Chapter 6: Novel repetitive elements in the genome of SBW25

The final results chapter was a study of the remaining two groups of repetitive sequences that were identified in the genome of *P. fluorescens* SBW25 by Silby et al. [100]. First, a repeat family named R178 was analysed. Similar to the earlier analysis of short repetitive sequences in the SBW25 genome, the R178 repetitiveness can be the result of several different processes. The possibilities that R178 repeats emerged by chance or as a result of similar selective pressures in different genetic backgrounds (convergent evolution) were ruled out. The alternative, that R178 are duplicative elements was further analysed. To determine whether the duplicative process is driven by a protein or catalytic RNA that is encoded by R178 repeats (autonomous) or driven by an element encoded at a different position within the genome (non-autonomous) the R178 element was analysed for conserved open reading frames (ORFs) and RNA secondary structures. No conserved ORFs could be identified. However, the predicted RNA secondary structure showed some structural conservation among the 18 R178 repeats. The observed conservation could be due to a function that is performed by RNA. Alternatively the conservation could also be a result of functional secondary structures that are formed when the DNA is found in a single stranded state (*e.g.* secondary structures that are formed during transposition, as shown for IS*200* sequences [48, 143-146]). Hence, I also predicted the secondary structure of ssDNA. Interestingly, all 18 R178 sequences are also predicted to form highly similar secondary structures in ssDNA, leaving the possibility for both, functional RNA and ssDNA. Conserved loops within these structures also show overlaps with polymorphic regions in a multiple alignment of R178 sequences. Complementary counterparts of these regions were found in the vicinity of almost all R178 sequences. One loop was found to be complementary to a region downstream of almost all R178 repeats, found only a few bases before the start codon of a conserved short protein-coding gene. Another loop is complementary to a site within the R178 repeat as well as a site about 50 bp upstream of the R178 repeat. In four instances, this site is also the 5′ end of the R30 repeat, which is exclusively found directly upstream of R178 repeats.

Sequence analyses of R178 upstream regions showed that they all share similarities with R30 repeat sequences. This led to the prediction that R178 is the central sequence of a larger genetic element consisting of a 5′ R30 sequence, an R178 sequence, and a 3′ peptide. If the three elements are part of the same genetic element then the three individual parts are expected to co-evolve. Evidence for this hypothesis comes from the comparison of the three phylogenetic trees built from the individual parts of the genetic element, which are highly similar. The association between peptide and conserved RNA secondary structure strongly reminds of type I TA systems. A weak association between R178 repeats and REP/REPIN structures was observed. However, the impact and cause of this association remains obscure.

The second part of the chapter analyzed the composition and evolution of R200 repeats. R200 repeats are quite different from R178 repeats. The length of the two sequence groups differs considerably. R200 sequences are longer and range in length from 129 bp to 380 bp. R178 sequences in contrast range in length from 98 bp to 102 bp. R200 repeats are also far more frequent than R178 repeats; found 48 times within the genome compared to 18 R178 copies. Similar to the analysis of R178 repeats, the investigation into the cause of replication suggested that R200 repeats are the result of a replicative process rather than the product of chance or independent local selective processes. To elucidate whether the replicative process is likely to be driven by a product encoded by the R200 sequences or by a product that is encoded in *trans*, the sequence was analysed for conserved RNA secondary structures (necessary for autonomous RNA transposases such as group I introns [188]) and ORFs (necessary for replication of autonomous transposons). No ORF that spanned the whole R200 sequence could be identified; however, a conserved RNA secondary structure was predicted. Since a conserved RNA secondary structure does not necessarily mean that a replicative RNA is encoded, alternative explanations were considered. One alternative is that R200 repeats encode type I TA systems. They are commonly found within bacterial genomes and consist of a non-coding RNA (with conserved secondary structure) and a toxic peptide. Furthermore they have been found replicated within the genome of *E. coli* [73]. A literature search showed that a 5′-CCAG-3′ motif is shared by a number of different type I RNA TA system, which is also highly conserved in R200 sequences. This prompted further analyses which revealed that the R200 sequence consists of at least

two parts: a 5′ ORF that was predicted to encode a transmembrane toxin and a 3′ region potentially coding for an RNA molecule acting as antitoxin. This hypothesis was supported by similarities shared with the IstR-1/TisB TA system in *E. coli*. The secondary structure of the putative R200 antitoxin is similar to that of the IstR-1 antitoxin, and the short peptide encoded by R200 is similar in length to the TisB toxin, which encodes a transmembrane helix. Interestingly, the R200-encoded peptide shows a weak tendency to form a transmembrane helix, according to the online prediction tool TMpred [124]. However, the change of a single nucleotide can lead to a strong predicted membrane association. It is possible a mutation occurred during the in *trans* transposition of R200 sequences, without which even leaky expression may lead to cell death.

Another curious feature of R200 repeats is the strong linkage with REP/REPIN structures. Analyses show a correlation between REP/REPIN-associated repeats and differences in phylogenetic tree topology and sequence diversity. This may indicate that the association between R200 sequences and REPINs could have led to the amplification of R200 sequences.

7.2 Evaluation of the implications

7.2.1 Technological advances that made this work possible

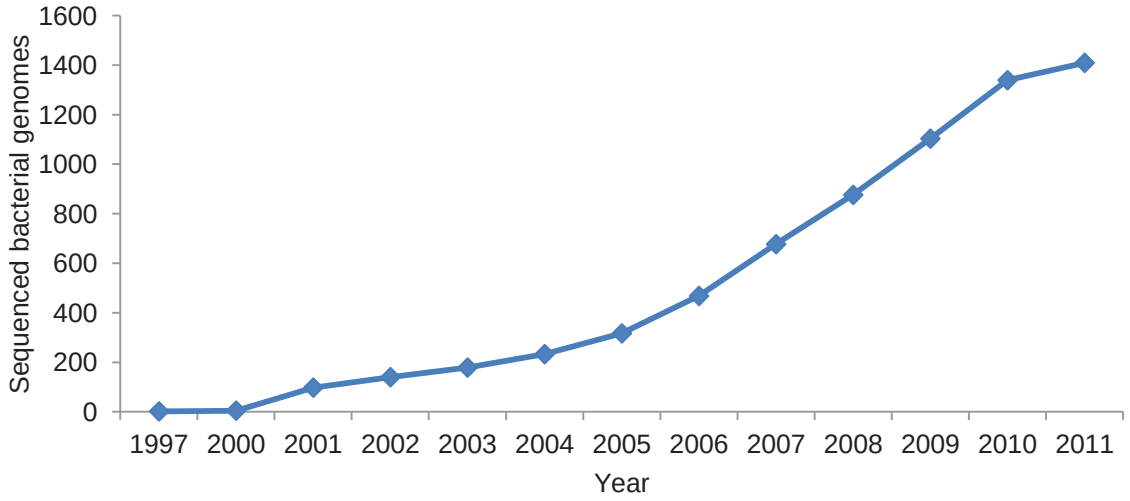


Figure 7.1. The number of fully sequenced bacterial genomes available at NCBI from August 1997 to May 2011.

The "dawn of the genomic era" was announced when the complete human genome sequence was published in 2001 [221, 222]. However, a significant reduction in cost and time per sequenced base pair was only achieved in 2005 with the development of high throughput (3rd or "next-generation") sequencing technologies [223, 224]. Those technologies enable *de novo* assemblies of bacterial genomes and rapid re-sequencing of any eukaryotic genome for a fraction of the price and time required with the original Sanger sequencing. Hence, the yearly number of fully sequenced bacterial genomes deposited on the NCBI website significantly increased from 2005 onwards compared to previous years (Figure 7.1). The great variety of available genome sequences presents a wealth of information that greatly enhances the formulation and testing of hypotheses across a diverse set of genomes. Furthermore, new approaches to analyze not only the resulting consensus sequences, but also the raw short sequence reads from the sequencing run are constantly being developed. Interesting examples include the determination of genome conformation within the cell or nucleus [225], and analyses to detect rare mutations or amplifications within bacterial populations [226]. The analyses and approaches presented in this thesis make use of both sequence data generated by next-generation sequencing technologies and the existing fully sequenced bacterial genomes.

7.2.2 Relevance of the developed approaches to the field

A "top-down" (analysing a complex system by characterizing the system as a whole first and subsequently smaller and smaller components of it) analysis of repetitive sequences in the genome of *P. fluorescens* SBW25 was performed prior to this thesis [100]. It defined a set of repetitive sequences with varying lengths that were found in the three sequenced *P. fluorescens* strains. Although this approach allowed the positions of repetitive sequences to be marked with reasonable accuracy, it did not provide information about the structure or characteristics of the sequences and hence did not classify them into evolutionarily meaningful sequence groups. The work presented in this thesis is based on a "bottom-up" (analysing a complex system by characterizing its smallest parts first in order to understand the formation of larger components) approach to identify and describe repetitive sequences. Instead of trying to define the longest possible repetitive DNA sequences, properties of short repetitive sequences of

10 bp to 20 bp - which appear to be the most conserved building blocks of larger repeats - were analysed. Since this approach proved to be successful, it may be useful for the identification, classification and description of repetitive sequences in other genomes or possibly also for less abundant sequences that still occur more frequently than expected by chance in the genome of SBW25 and other microbial and eukaryotic genomes.

Furthermore, the application of next-generation sequencing (NGS) is likely to not only be useful in showing REPIN activity, but as a simple first mobility test for any genetic element that is proposed to amplify through transposition (a small proportion of the reads should show insertion or deletions of mobile DNA sequences). It is also noteworthy that some non-matching sequences from NGS data are not necessarily erroneous sequences, but could be the results of real biological processes.

In contrast to Chapter 3, Chapter 5 pursued a comparative "top-down" approach to describe RAYTs. Questions concerning gene family characteristics were first addressed on the highest level of complexity (top level) and compared to characteristics observed for other gene families. On the highest level the only knowledge about the sequence families is that the identified sequences share a certain similarity to the two query sequences. Since these similarities could be the result of different matching regions within the gene (Figure 7.2), the sequences were analysed on a lower level (pairwise sequence comparison) by applying phylogenetic maps. These maps visualize phylogenetic relationships by displaying proteins as nodes and pairwise identities above a certain threshold as lines. A visualization algorithm then determines the length of each line based on the number of connections within a certain group of nodes, which in turn leads to the formation of phylogenetic clusters. These clusters were further

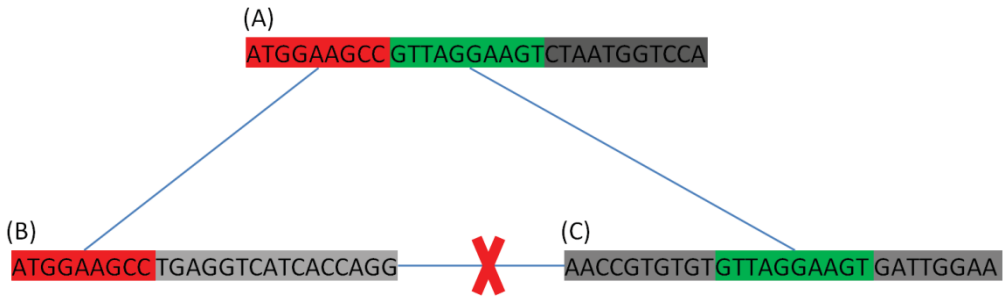


Figure 7.2. Evolution of sequence clusters. The sequence pairs AB and AC share an 8 bp sequence motif, which is not shared by BC. If selection preserves the red and green sequence motifs in all three sequences, this results in the formation of three separate sequence clusters (A, B and C) that are connected through A.

analyzed for characteristics of interest. Characteristics can be analysed on a lower level by increasing the pairwise identity threshold, which leads to smaller clusters with closer relationships. This approach could not only be useful for the analysis of RAYTs, but could help to understand the evolution of most proteins in bacteria. However, it might be particularly useful for the preliminary analysis of a new family of proteins.

In Chapter 6, the remaining two repetitive sequence classes were analysed mostly by applying knowledge about repetitive selfish genetic elements acquired in Chapter 3. In both cases the recognition of a highly conserved ssDNA/RNA secondary structure was the key for further description of the genetic elements. Simple phylogenetic and sequence analyses helped to further understand the elements and formulate testable hypotheses.

7.2.3 Relevance of the described results to the field

In *P. fluorescens* SBW25 REPINs and higher order organizations comprise more than one percent of the genome. Hence understanding the dynamics and causes of REPIN dispersal and evolution is important for our understanding of the ecology, evolution and function of *P. fluorescens* SBW25. However, understanding the dynamics between REPINs and their associated transposases (RAYT) is not straightforward. In plants and other eukaryotes non-autonomous transposons seem to simply exploit an also repetitive autonomous transposon until either the transposon evolves to prevent exploitation (thereby rendering the non-autonomous transposon non-functional) or the autonomous transposon goes extinct [227]. The widespread REPIN-RAYT system in bacteria appears to follow different rules. If genomes contain both RAYTs and REPINs then one RAYT is associated with one specific group of repetitive REPINs. If the system were entirely selfish then RAYTs are expected to rapidly go extinct due to random drift unless there is a high rate of horizontal transfer. However, both the phylogenetic tree shown in Chapter 4 (Figure 4.3) and the non-existence of RAYTs on plasmids shown in Chapter 5 suggest that horizontal transfer of RAYTs is very rare. Although initially the REPIN-RAYT system may have been entirely selfish, the inferred vertical mode of transmission suggested that the system was co-opted by the host to perform a beneficial

function. Determining the beneficial function in turn could help to understand other aspects of bacterial ecology.

The results presented in Chapter 5 not only show that RAYT characteristics are more similar to host gene characteristics than to insertion sequences, but also that insertion sequence characteristics can be regained (RAYT cluster (c) showed insertion sequence characteristics and closer relationship to cluster (b), which did not show insertion sequence characteristics than to IS*200*). In hindsight this finding does not seem surprising, especially considering that transposing foreign DNA sequences (*trans* transposition *e.g.* REPINs) is likely to be similar to transposing its own DNA sequence (*cis* transposition *e.g.* the encoded RAYT). However, comparable examples of losing *cis* transposition capabilities and regaining *trans* transposition capabilities have not been reported to my knowledge. This may be due to a number of reasons. One is probably that the necessary data (thousands of fully sequenced bacterial genomes) to do similar studies has only been available for the last few years. Another reason could be that interests in specific protein families are mostly constrained to either a detailed insight into enzyme mechanisms (*e.g.* [48]) or a superficial overview of general protein families without interest in specific gene characteristics (*e.g.* [228]). Studies similar to the one conducted in Chapter 5 seem to be rare.

Another interesting finding in Chapter 5 was that one RAYT subfamily was neither linked to REPINs nor did it show any insertion sequence characteristics. As for the REPIN-RAYT system, this group of RAYTs presumably confers a benefit to the host to prevent loss through random genetic drift. The nature of this benefit is elusive, although it is likely to be interesting, since most reported domesticated transposases are the result of hybridization with a host protein [44-47] and not the potentially slow stepwise acquisition of a new beneficial function as inferred for the RAYT protein family.

The association between TA systems encoded by R200 repeats and the REPIN-RAYT system in Chapter 6 suggests that even within bacterial chromosomes addictive and duplicative selfish genetic elements cooperate. The cause and effect of this cooperation are elusive; with the possible exception of the amplification of R200 elements as an effect of close association with REPs/REPINs. However, one could speculate that cooperation between R200 and the REPIN-RAYT system represents a mutualism similar to plasmid-TA systems, leading to increased vertical or horizontal transmission

of both the REPIN-RAYT system and the R200 repeats. It is not clear what led to the replication of R178 repeats. It is possible that within genome recombination or an autonomous transposase is the cause of R178 amplification.

Another interesting aspect of R178 repeats is the curious combination of conserved ssDNA/RNA secondary structures together with a conserved protein coding region. This combination is similar to type I TA systems. However, it seems unlikely that ssDNA structures regulate gene expression. It is possible that the observed conserved ssDNA structures are artefacts of the applied prediction programs. Hence, the regulation of the downstream protein coding region may be achieved through the transcription of R178 into a regulatory non-coding RNA (similar to type I TA systems). Either way, resolution of this problem could help to improve secondary structure prediction algorithms, or alternatively reveal that ssDNA is able to affect gene expression through the formation of secondary structures.

In general understanding the evolution of repetitive elements is also required to obtain insight into the mechanisms of bacterial genome evolution. This is of particular importance, since repetitive sequences enable recombination within and between bacterial genomes [211, 229]. This is an effect that probably contributes to the lack of synteny in many bacterial genomes and is enhanced by the great numbers of repetitive elements that are found within bacterial genomes [60, 100, 230].

7.3 Future directions

The work presented in this thesis shows that computational biology is an invaluable tool for understanding the biology of bacterial selfish genetic elements. Wet lab experiments are needed to test predictions and hypotheses. Nevertheless, there are a number of computational studies that could be performed to deepen the understanding of the evolution of selfish genetic elements in bacterial genomes.

7.3.1 REPINs and their associated RAYTs

There are many open questions concerning the REPIN-RAYT system. Whether REPINs are mobilized by RAYTs is probably the most basic hypothesis to test; based

on computational evidence this hypothesis is likely to be confirmed. However, the multitude of cluster types found in the genome of SBW25 raises the possibility that RAYTs are not the only cause for REPIN amplification and that there may be a secondary mechanism that leads to REPIN and/or REP amplification.

Another interesting avenue for future research involves the predicted beneficial function that RAYTs (possibly in conjunction with REPINs) provide to the bacterium in order to be preserved. REPIN-RAYT systems could be involved in DNA repair or the regulation of gene expression based on their presumed ability to bind, cut and ligate DNA. Some hypotheses are currently being tested in the lab (XX Zhang, AP Lind, F Bertels, PB Rainey) and preliminary results indicate that, under certain conditions (changing environments), there is a small fitness effect when all RAYT genes are deleted from the SBW25 genome. Further evidence of REPIN movement is expected to be observed in a one year mutation accumulation experiment that includes the SBW25 wild type strain as well as the RAYT deletion strain (XX Zhang, PB Rainey).

The presence of highly organized REPIN and REP structures within the genome of *P. fluorescens* SBW25 and their linkage with R200 and R178 repeats suggests that there may be more than one function for REPINs and REPs. Given that relatives of the TA system encoded by R200 elements are involved in the SOS response, it is possible that REPs/REPINs are also indirectly involved in a similar process in SBW25. Investigating the function of R178 and R200 repeats together with the effect of the association with REPs/REPINs could provide insight into the predicted cooperative nature of the connection.

REPs/REPINs have been found to be associated with other selfish genetic elements in other bacteria (*e.g.* the REPIN-RAYT system in *Neisseria meningitides* (Chapter 5)), which is also likely to affect hypotheses regarding the functional significance of the system. This genetic diversity poses great challenges for future research. However, one could imagine that greater mechanistic insight into both amplification and dispersal processes could help to predict the functional significance of REPIN-RAYT systems in different genetic backgrounds and in association with different selfish genetic elements. Furthermore, the mechanism for REPIN dispersal and amplification is likely to differ between the different RAYT clusters, indicated by the high RAYT sequence diversity (Chapter 5) and large structural differences between associated REPINs (Chapter 4).

In conclusion, it is probably safe to assume that elucidating the different effects the RAYT-REPIN system has on the bacterium will provide sufficient research opportunities for many years to come.

7.3.2 Research opportunities arising from studying cluster (c) and (d) RAYTs

The discovery of RAYT subfamilies in Chapter 5 that were not associated with REPs/REPINs was somewhat surprising and led to a multitude of future research questions.

7.3.2.1 Cluster (c) RAYTs

Cluster (c) represents the largest RAYT subfamily, which is probably the result of regaining a high rate of self-replication and horizontal transfer (insertion sequence properties). However, whether these properties were re-acquired from a group of essentially single copy genes (RAYTs) or whether cluster (c) genes are simply a less prolific offshoot of the IS*200* gene cluster needs further testing. But no matter what the final answer to this question, it is obvious that cluster (c) RAYTs are very different (in sequence and characteristics) from both other RAYT groups and IS*200* sequences. Furthermore, based on the above prediction that RAYTs are likely to have a beneficial function, it would be interesting to first theoretically and then experimentally investigate if that needs to be/is the case for cluster (c) RAYTs, or if the observed replication and horizontal transfer rate is sufficient to explain persistence and conservation among genomes. This analysis would be of particular interest as cluster (c) RAYTs are an intermediate class of genes that lies in between single copy genes and typical insertion sequences.

7.3.2.2 Cluster (d) RAYTs

Cluster (d) is another curious RAYT subfamily. It shows no signs of REPIN association but neither does it show insertion sequence-like characteristics. Cluster (d) RAYTs are found as conserved single copy genes within genomes and therefore are also expected to provide a beneficial function to the bacterium. One could imagine that cluster (d) RAYTs evolved from cluster (b) RAYTs by losing the ability to transpose

REPINs as well as either gaining a new beneficial function or changing/enhancing the beneficial function it provided in conjunction with REPINs. Testing this hypothesis through computational analyses as well as wet lab experiments could provide insights into the evolution of insertion sequences and host genes.

7.3.3 R178 and R200 repeats

The study of R200 and R178 repeats showed that short repetitive elements are not necessarily non-autonomous transposons derived from functional insertion sequences. Instead Chapter 6 shows that putative addictive selfish genetic elements can take advantage of the replicative properties of unrelated duplicative selfish genetic elements. Future research could include: (1) a test of the hypothesis that R200 and R178 repeats encode TA systems, (2) an investigation of the function of individual components of the system, and (3) a study of the causes and mechanism of R178 and R200 amplification.

Analogous approaches could also be applied to characterize similar repetitive elements in other genomes. Such analyses could unveil whether (and potentially why) cooperation between MITEs (or other duplicative selfish genetic elements) and TA systems (or other addictive selfish genetic elements) is commonly found on bacterial chromosomes.

7.4 Final comment

Selfish genetic elements are found in almost all genomes. Even highly streamlined bacterial genomes contain a diverse range of addictive and duplicative selfish genes. Analyses conducted in this study describe four new classes of selfish genetic elements; hence, the number of selfish genetic elements known to be present in bacterial genomes is likely to rise as more such studies are conducted. However, many selfish genetic elements - including chromosomally encoded TA systems, restriction modification systems or CRISPRs - have been co-opted by the bacterium to perform beneficial functions, to the point where it is hard to justify the term ‘selfish’. Similar processes may have led to the evolution of the REPIN-RAYT system or the reported R200 and R178 repeats, for which beneficial functions are predicted. Another major point of this

thesis is that persistence of selfish genetic elements within genomes may be enhanced through cooperation with other selfish genetic elements. However, the nature of this cooperation and whether it also leads to enhanced benefits for the host remains unclear. Nevertheless, the abundance and diversity of REPINs, their associated RAYTs, other RAYT families and R200/R178 repeats within and between genomes suggests that they play an important role in bacterial evolution and ecology as well as performing certain cellular functions.

References

1. Taylor M, Campbell N, Reece J (2007) Biology. Pearson/Benjamin Cummings.

2. Painter TS (1934) The Morphology of the X Chromosome in Salivary Glands of Drosophila Melanogaster and a New Type of Chromosome Map for This Element. Genetics 19: 448–469.

3. Guay PS, Guild GM (1991) The ecdysone-induced puffing cascade in Drosophila salivary glands: a Broad-Complex early gene regulates intermolt and late gene transcription. Genetics 129: 169–175.

4. Mayer VW, Aguilera A (1990) High levels of chromosome instability in polyploids of Saccharomyces cerevisiae. Mutat Res 231: 177–186.

5. Kellis M, Birren BW, Lander ES (2004) Proof and evolutionary analysis of ancient genome duplication in the yeast Saccharomyces cerevisiae. Nature 428: 617–624. doi: 10.1038/nature02424.

6. Crow KD, Wagner GP, Investigators SMBETNY (2006) Proceedings of the SMBE Tri-National Young Investigators' Workshop 2005. What is the role of genome duplication in the evolution of complexity and diversity? Mol Biol Evol 23: 887–892.

7. Sémon M, Wolfe KH (2007) Consequences of genome duplication. Curr Opin Genet Dev 17: 505–512. doi: 10.1016/j.gde.2007.09.007.

8. Soskine M, Tawfik DS (2010) Mutational effects and the evolution of new protein functions. Nat Rev Genet 11: 572–582. doi: 10.1038/nrg2808.

9. Chandler M, Galas DJ (1983) IS1-mediated tandem duplication of plasmid pBR322. Dependence on recA and on DNA polymerase I. J Mol Biol 165: 183–190.

10. Petes TD, Hill CW (1988) Recombination between repeated genes in microorganisms. Annu Rev Genet 22: 147–168. doi: 10.1146/annurev.ge.22.120188.001051.

11. Shapiro J (1983) Mobile genetic elements. Orlando, FL, USA: Academic Press.

12. Waldor MK, Friedman DI (2005) Phage regulatory circuits and virulence gene expression. Curr Opin Microbiol 8: 459–465. doi: 10.1016/j.mib.2005.06.001.

13. Hurst LD, Atlan A, Bengtsson BO (1996) Genetic conflicts. Q Rev Biol 71: 317–364.

14. Ohno S (1970) Evolution by gene duplication. London: George Alien & Unwin Ltd. Berlin, Heidelberg and New York: Springer-Verlag.

15. Lynch M, Conery JS (2000) The evolutionary fate and consequences of duplicate genes. Science 290: 1151–1155.

16. Cordaux R, Batzer MA (2009) The impact of retrotransposons on human genome evolution. Nat Rev Genet 10: 691–703. doi: 10.1038/nrg2640.

17. Schneider D, Lenski RE (2004) Dynamics of insertion sequence elements during experimental evolution of bacteria. Res Microbiol 155: 319–327. doi: 10.1016/j.resmic.2003.12.008.

18. Mahillon J, Chandler M (1998) Insertion sequences. Microbiol Mol Biol Rev 62: 725–774.

19. Parks AR, Peters JE (2009) Tn7 elements: engendering diversity from chromosomes to episomes. Plasmid 61: 1–14. doi: 10.1016/j.plasmid.2008.09.008.

20. Wessler SR, Bureau TE, White SE (1995) LTR-retrotransposons and MITEs: important players in the evolution of plant genomes. Curr Opin Genet Dev 5: 814–821.

21. Burt A, Trivers R (2006) Genes in Conflict: The Biology of Selfish Genetic Elements. Cambridge, Massachusetts: Belknap Press of Harvard University Press.

22. Mochizuki A, Yahara K, Kobayashi I, Iwasa Y (2006) Genetic addiction: selfish gene's strategy for symbiosis in the genome. Genetics 172: 1309–1323. doi: 10.1534/genetics.105.042895.

23. Kobayashi I (2004) Plasmid Biology, ASM Press, Washington, D.C., chapter 6: Genetic addiction - a principle in symbiosis of genes in a genome, pages 105–144.

24. Feschotte C, Pritham EJ (2007) DNA transposons and the evolution of eukaryotic genomes. Annu Rev Genet 41: 331–368. doi: 10.1146/annurev.genet.40.110405.090448.

25. Hsia AP, Schnable PS (1996) DNA sequence analyses support the role of interrupted gap repair in the origin of internal deletions of the maize transposon, MuDR. Genetics 142: 603–618.

26. Dewannieux M, Esnault C, Heidmann T (2003) LINE-mediated retrotransposition of marked Alu sequences. Nat Genet 35: 41–48. doi: 10.1038/ng1223.

27. Hartl DL, Lozovskaya ER, Lawrence JG (1992) Nonautonomous transposable elements in prokaryotes and eukaryotes. Genetica 86: 47–53.

28. Abrusán G, Krambeck HJ (2006) Competition may determine the diversity of transposable elements. Theor Popul Biol 70: 364–375. doi: 10.1016/j.tpb.2006.05.001.

29. Feschotte C, Osterlund MT, Peeler R, Wessler SR (2005) DNA-binding specificity of rice mariner-like transposases and interactions with Stowaway MITEs. Nucleic Acids Res 33: 2153–2165. doi: 10.1093/nar/gki509.

30. Callinan PA, Batzer MA (2006) Retrotransposable elements and human disease. Genome Dyn 1: 104–115. doi: 10.1159/000092503.

31. Deininger PL, Batzer MA (1999) Alu repeats and human disease. Mol Genet Metab 67: 183–193. doi: 10.1006/mgme.1999.2864.

32. Hata K, Sakaki Y (1997) Identification of critical CpG sites for repression of L1 transcription by DNA methylation. Gene 189: 227–234.

33. Eickbush TH, Jamburuthugoda VK (2008) The diversity of retrotransposons and the properties of their reverse transcriptases. Virus Res 134: 221–234. doi: 10.1016/j.virusres.2007.12.010.

34. Malik HS, Eickbush TH (2001) Phylogenetic analysis of ribonuclease H domains suggests a late, chimeric origin of LTR retrotransposable elements and retroviruses. Genome Res 11: 1187–1197. doi: 10.1101/gr.185101.

35. Malik HS, Henikoff S, Eickbush TH (2000) Poised for contagion: evolutionary origins of the infectious abilities of invertebrate retroviruses. Genome Res 10: 1307–1318.

36. Havecker ER, Gao X, Voytas DF (2004) The diversity of LTR retrotransposons. Genome Biol 5: 225. doi: 10.1186/gb-2004-5-6-225.

37. Charlesworth B, Sniegowski P, Stephan W (1994) The evolutionary dynamics of repetitive DNA in eukaryotes. Nature 371: 215–220. doi: 10.1038/371215a0.

38. Matsuura M, Saldanha R, Ma H, Wank H, Yang J, et al. (1997) A bacterial group II intron encoding reverse transcriptase, maturase, and DNA endonuclease activities: biochemical demonstration of maturase activity and insertion of new genetic information within the intron. Genes Dev 11: 2910–2924.

39. Curcio MJ, Belfort M (1996) Retrohoming: cDNA-mediated mobility of group II introns requires a catalytic RNA. Cell 84: 9–12.

40. Cousineau B, Lawrence S, Smith D, Belfort M (2000) Retrotransposition of a bacterial group II intron. Nature 404: 1018–1021. doi: 10.1038/35010029.

41. Pace JK, Feschotte C (2007) The evolutionary history of human DNA transposons: evidence for intense activity in the primate lineage. Genome Res 17: 422–432. doi: 10.1101/gr.5826307.

42. Ray DA, Pagan HJT, Thompson ML, Stevens RD (2007) Bats with hATs: evidence for recent DNA transposon activity in genus Myotis. Mol Biol Evol 24: 632–639. doi: 10.1093/molbev/msl192.

43. Mclintock B (1950) The origin and behavior of mutable loci in maize. Proc Natl Acad Sci U S A 36: 344–355.

44. Sarkar A, Sim C, Hong YS, Hogan JR, Fraser MJ, et al. (2003) Molecular evolutionary analysis of the widespread piggyBac transposon family and related "domesticated" sequences. Mol Genet Genomics 270: 173–180. doi: 10.1007/s00438-003-0909-0.

45. Zhou L, Mitra R, Atkinson PW, Hickman AB, Dyda F, et al. (2004) Transposition of hAT elements links transposable elements and V(D)J recombination. Nature 432: 995–1001. doi: 10.1038/nature03157.

46. Cordaux R, Udit S, Batzer MA, Feschotte C (2006) Birth of a chimeric primate gene by capture of the transposase gene from a mobile element. Proc Natl Acad Sci U S A 103: 8101–8106. doi: 10.1073/pnas.0601161103.

47. Feschotte C (2008) Transposable elements and the evolution of regulatory networks. Nat Rev Genet 9: 397–405. doi: 10.1038/nrg2337.

48. Ton-Hoang B, Guynet C, Ronning DR, Cointin-Marty B, Dyda F, et al. (2005) Transposition of ISHp*608*, member of an unusual family of bacterial insertion sequences. EMBO J 24: 3325–3338. doi: 10.1038/sj.emboj.7600787.

49. Galimand M, Sabtcheva S, Courvalin P, Lambert T (2005) Worldwide disseminated armA aminoglycoside resistance methylase gene is borne by composite transposon Tn1548. Antimicrob Agents Chemother 49: 2949–2953. doi: 10.1128/AAC.49.7.2949-2953.2005.

50. Hulton CS, Higgins CF, Sharp PM (1991) ERIC sequences: a novel family of repetitive elements in the genomes of *Escherichia coli*, *Salmonella typhimurium* and other enterobacteria. Mol Microbiol 5: 825–834.

51. Correia FF, Inouye S, Inouye M (1988) A family of small repeated elements with some transposon-like properties in the genome of *Neisseria gonorrhoeae*. J Biol Chem 263: 12194–12198.

52. Delihas N (2008) Small mobile sequences in bacteria display diverse structure/function motifs. Mol Microbiol 67: 475–481. doi: 10.1111/j.1365-2958.2007.06068.x.

53. Oggioni MR, Claverys JP (1999) Repeated extragenic sequences in prokaryotic genomes: a proposal for the origin and dynamics of the RUP element in *Streptococcus pneumoniae*. Microbiol 145: 2647–2653.

54. Levinson G, Gutman GA (1987) Slipped-strand mispairing: a major mechanism for DNA sequence evolution. Mol Biol Evol 4: 203–221.

55. Myers RS, Stahl FW (1994) Chi and the RecBC D enzyme of *Escherichia coli*. Annu Rev Genet 28: 49–70. doi: 10.1146/annurev.ge.28.120194.000405.

56. Bigot S, Saleh OA, Lesterlin C, Pages C, Karoui ME, et al. (2005) KOPS: DNA motifs that control *E. coli* chromosome segregation by orienting the FtsK translocase. EMBO J 24: 3770–3780. doi: 10.1038/sj.emboj.7600835.

57. Hendrickson H, Lawrence JG (2006) Selection for chromosome architecture in bacteria. J Mol Evol 62: 615–629. doi: 10.1007/s00239-005-0192-2.

58. Stern M, Ames G, Smith N, Robinson E, Higgins C (1984) Repetitive extragenic palindromic sequences: a major component of the bacterial genome. Cell 37: 1015–1026.

59. Gilson E, Clément JM, Brutlag D, Hofnung M (1984) A family of dispersed repetitive extragenic palindromic DNA sequences in *E. coli*. EMBO J 3: 1417–1421.

60. Treangen TJ, Abraham AL, Touchon M, Rocha EPC (2009) Genesis, effects and fates of repeats in prokaryotic genomes. FEMS Microbiol Rev 33: 539–571.

61. de Bruijn FJ (1992) Use of repetitive (repetitive extragenic palindromic and enterobacterial repetitive intergeneric consensus) sequences and the polymerase chain reaction to fingerprint the genomes of *Rhizobium meliloti* isolates and other soil bacteria. Appl Environ Microbiol 58: 2180–2187.

62. Versalovic J, Koeuth T, Lupski JR (1991) Distribution of repetitive DNA sequences in eubacteria and application to fingerprinting of bacterial genomes. Nucleic Acids Res 19: 6823–6831.

63. Higgins CF, McLaren RS, Newbury SF (1988) Repetitive extragenic palindromic sequences, mRNA stability and gene expression: evolution by gene conversion? A review. Gene 72: 3–14.

64. Siguier P, Filée J, Chandler M (2006) Insertion sequences in prokaryotic genomes. Curr Opin Microbiol 9: 526–531. doi: 10.1016/j.mib.2006.08.005.

65. Eberhard WG (1990) Evolution in bacterial plasmids and levels of selection. Q Rev Biol 65: 3–22.

66. Meinhardt F, Kempken F, Kämper J, Esser K (1990) Linear plasmids among eukaryotes: fundamentals and application. Curr Genet 17: 89–95.

67. Futcher B, Reid E, Hickey DA (1988) Maintenance of the 2 micron circle plasmid of Saccharomyces cerevisiae by sexual transmission: an example of a selfish DNA. Genetics 118: 411–415.

68. Kado CI (1998) Origin and evolution of plasmids. Antonie Van Leeuwenhoek 73: 117–126.

69. Norman A, Hansen LH, Sørensen SJ (2009) Conjugative plasmids: vessels of the communal gene pool. Philos Trans R Soc Lond B Biol Sci 364: 2275–2289. doi: 10.1098/rstb.2009.0037.

70. Melderen LV (2010) Toxin-antitoxin systems: why so many, what for? Curr Opin Microbiol doi: 10.1016/j.mib.2010.10.006.

71. Gerdes K, Wagner EGH (2007) RNA antitoxins. Curr Opin Microbiol 10: 117–124. doi: 10.1016/j.mib.2007.03.003.

72. Unoson C, Wagner EGH (2008) A small SOS-induced toxin is targeted against the inner membrane in Escherichia coli. Mol Microbiol 70: 258–270. doi: 10.1111/j.1365-2958.2008.06416.x.

73. Fozo EM, Hemm MR, Storz G (2008) Small toxic proteins and the antisense RNAs that repress them. Microbiol Mol Biol Rev 72: 579–89, Table of Contents. doi: 10.1128/MMBR.00025-08.

74. Dörr T, Vulić M, Lewis K (2010) Ciprofloxacin causes persister formation by inducing the TisB toxin in Escherichia coli. PLoS Biol 8: e1000317. doi: 10.1371/journal.pbio.1000317.

75. Gazit E, Sauer RT (1999) The Doc toxin and Phd antidote proteins of the bacteriophage P1 plasmid addiction system form a heterotrimeric complex. J Biol Chem 274: 16813–16818.

76. Tian QB, Ohnishi M, Murata T, Nakayama K, Terawaki Y, et al. (2001) Specific protein-DNA and protein-protein interaction in the hig gene system, a plasmid-borne proteic killer gene system of plasmid Rts1. Plasmid 45: 63–74. doi: 10.1006/plas.2000.1506.

77. Hazan R, Sat B, Engelberg-Kulka H (2004) Escherichia coli mazEF-mediated cell death is triggered by various stressful conditions. J Bacteriol 186: 3663–3669. doi: 10.1128/JB.186.11.3663-3669.2004.

78. Gerdes K, Christensen SK, Løbner-Olesen A (2005) Prokaryotic toxin-antitoxin stress response loci. Nat Rev Microbiol 3: 371–382. doi: 10.1038/nrmicro1147.

79. Critchlow SE, O'Dea MH, Howells AJ, Couturier M, Gellert M, et al. (1997) The interaction of the F plasmid killer protein, CcdB, with DNA gyrase: induction of

DNA cleavage and blocking of transcription. J Mol Biol 273: 826–839. doi: 10.1006/jmbi.1997.1357.

80. Robson J, McKenzie JL, Cursons R, Cook GM, Arcus VL (2009) The vapBC operon from Mycobacterium smegmatis is an autoregulated toxin-antitoxin module that controls growth via inhibition of translation. J Mol Biol 390: 353–367. doi: 10.1016/j.jmb.2009.05.006.

81. Pedersen K, Zavialov AV, Pavlov MY, Elf J, Gerdes K, et al. (2003) The bacterial toxin RelE displays codon-specific cleavage of mRNAs in the ribosomal A site. Cell 112: 131–140.

82. Rotem E, Loinger A, Ronin I, Levin-Reisman I, Gabay C, et al. (2010) Regulation of phenotypic variability by a threshold-based mechanism underlies bacterial persistence. Proc Natl Acad Sci U S A 107: 12541–12546. doi: 10.1073/pnas.1004333107.

83. Fineran PC, Blower TR, Foulds IJ, Humphreys DP, Lilley KS, et al. (2009) The phage abortive infection system, ToxIN, functions as a protein-RNA toxin-antitoxin pair. Proc Natl Acad Sci U S A 106: 894–899. doi: 10.1073/pnas.0808832106.

84. Anba J, Bidnenko E, Hillier A, Ehrlich D, Chopin MC (1995) Characterization of the lactococcal abiD1 gene coding for phage abortive infection. J Bacteriol 177: 3818–3823.

85. Gilson L, Mahanty HK, Kolter R (1990) Genetic analysis of an MDR-like export system: the secretion of colicin V. EMBO J 9: 3875–3884.

86. Cascales E, Buchanan SK, Duché D, Kleanthous C, Lloubès R, et al. (2007) Colicin biology. Microbiol Mol Biol Rev 71: 158–229. doi: 10.1128/MMBR.00036-06.

87. Naito T, Kusano K, Kobayashi I (1995) Selfish behavior of restriction-modification systems. Science 267: 897–899.

88. Wilson GG, Murray NE (1991) Restriction and modification systems. Annu Rev Genet 25: 585–627. doi: 10.1146/annurev.ge.25.120191.003101.

89. Ishino Y, Shinagawa H, Makino K, Amemura M, Nakata A (1987) Nucleotide sequence of the *iap* gene, responsible for alkaline phosphatase isozyme conversion in *Escherichia coli*, and identification of the gene product. J Bacteriol 169: 5429–5433.

90. Jansen R, van Embden JDA, Gaastra W, Schouls LM (2002) Identification of genes that are associated with DNA repeats in prokaryotes. Mol Microbiol 43: 1565–1575.

91. Mojica FJM, Díez-Villaseñor C, García-Martínez J, Soria E (2005) Intervening sequences of regularly spaced prokaryotic repeats derive from foreign genetic elements. J Mol Evol 60: 174–182. doi: 10.1007/s00239-004-0046-3.

92. Bolotin A, Quinquis B, Sorokin A, Ehrlich SD (2005) Clustered regularly interspaced short palindrome repeats (CRISPRs) have spacers of extrachromosomal origin. Microbiology 151: 2551–2561. doi: 10.1099/mic.0.28048-0.

93. Pourcel C, Salvignol G, Vergnaud G (2005) CRISPR elements in Yersinia pestis acquire new repeats by preferential uptake of bacteriophage DNA, and provide additional tools for evolutionary studies. Microbiology 151: 653–663. doi: 10.1099/mic.0.27437-0.

94. Makarova KS, Grishin NV, Shabalina SA, Wolf YI, Koonin EV (2006) A putative RNA-interference-based immune system in prokaryotes: computational analysis of the predicted enzymatic machinery, functional analogies with eukaryotic RNAi, and hypothetical mechanisms of action. Biol Direct 1: 7. doi: 10.1186/1745-6150-1-7.

95. Horvath P, Barrangou R (2010) CRISPR/Cas, the immune system of bacteria and archaea. Science 327: 167–170. doi: 10.1126/science.1179555.

96. Stern A, Sorek R (2011) The phage-host arms race: shaping the evolution of microbes. Bioessays 33: 43–51. doi: 10.1002/bies.201000071.

97. Rainey PB, Bailey MJ (1996) Physical and genetic map of the Pseudomonas fluorescens SBW25 chromosome. Mol Microbiol 19: 521–533.

98. Rainey PB, Travisano M (1998) Adaptive radiation in a heterogeneous environment. Nature 394: 69–72. doi: 10.1038/27900.

99. Beaumont HJE, Gallie J, Kost C, Ferguson GC, Rainey PB (2009) Experimental evolution of bet hedging. Nature 462: 90–93. doi: 10.1038/nature08504.

100. Silby M, Cerdeno-Tarraga A, Vernikos G, Giddens S, Jackson R, et al. (2009) Genomic and genetic analyses of diversity and plant interactions of *Pseudomonas fluorescens*. Genome Biol 10: R51. doi: 10.1186/gb-2009-10-5-r51.

101. Nunvar J, Huckova T, Licha I (2010) Identification and characterization of repetitive extragenic palindromes (REP)-associated tyrosine transposases: implications for REP evolution and dynamics in bacterial genomes. BMC Genomics 11: 44. doi: 10.1186/1471-2164-11-44.

102. Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ (1990) Basic local alignment search tool. J Mol Biol 215: 403–410. doi: 10.1006/jmbi.1990.9999.

103. Rutherford K, Parkhill J, Crook J, Horsnell T, Rice P, et al. (2000) Artemis: sequence visualization and annotation. Bioinformatics 16: 944–945.

104. Zuker M (2003) Mfold web server for nucleic acid folding and hybridization prediction. Nucleic Acids Res 31: 3406–3415.

105. Paulsen IT, Press CM, Ravel J, Kobayashi DY, Myers GSA, et al. (2005) Complete genome sequence of the plant commensal *Pseudomonas fluorescens* Pf-5. Nat Biotechnol 23: 873–878. doi: 10.1038/nbt1110.

106. Joardar V, Lindeberg M, Jackson RW, Selengut J, Dodson R, et al. (2005) Whole-genome sequence analysis of *Pseudomonas syringae* pv. *phaseolicola* 1448A reveals divergence among pathovars in genes involved in virulence and transposition. J Bacteriol 187: 6488–6498. doi: 10.1128/JB.187.18.6488-6498.2005.

107. Feil H, Feil WS, Chain P, Larimer F, DiBartolo G, et al. (2005) Comparison of the complete genome sequences of *Pseudomonas syringae* pv. *syringae* B728a and pv. *tomato* DC3000. Proc Natl Acad Sci U S A 102: 11064–11069. doi: 10.1073/pnas.0504930102.

108. Buell CR, Joardar V, Lindeberg M, Selengut J, Paulsen IT, et al. (2003) The complete genome sequence of the *Arabidopsis* and tomato pathogen *Pseudomonas syringae* pv. *tomato* DC3000. Proc Natl Acad Sci U S A 100: 10181–10186. doi: 10.1073/pnas.1731982100.

109. Vodovar N, Vallenet D, Cruveiller S, Rouy Z, Barbe V, et al. (2006) Complete genome sequence of the entomopathogenic and metabolically versatile soil bacterium *Pseudomonas entomophila*. Nat Biotechnol 24: 673–679. doi: 10.1038/nbt1212.

110. Nelson KE, Weinel C, Paulsen IT, Dodson RJ, Hilbert H, et al. (2002) Complete genome sequence and comparative analysis of the metabolically versatile *Pseudomonas putida* KT2440. Environ Microbiol 4: 799–808.

111. Stover CK, Pham XQ, Erwin AL, Mizoguchi SD, Warrener P, et al. (2000) Complete genome sequence of *Pseudomonas aeruginosa* PAO1, an opportunistic pathogen. Nature 406: 959–964. doi: 10.1038/35023079.

112. Roy PH, Tetu SG, Larouche A, Elbourne L, Tremblay S, et al. (2010) Complete genome sequence of the multiresistant taxonomic outlier *Pseudomonas aeruginosa* PA7. PLoS One 5: e8842. doi: 10.1371/journal.pone.0008842.

113. Winstanley C, Langille MGI, Fothergill JL, Kukavica-Ibrulj I, Paradis-Bleau C, et al. (2009) Newly introduced genomic prophage islands are critical determinants of in vivo competitiveness in the Liverpool Epidemic Strain of *Pseudomonas aeruginosa*. Genome Res 19: 12–23. doi: 10.1101/gr.086082.108.

114. Yan Y, Yang J, Dou Y, Chen M, Ping S, et al. (2008) Nitrogen fixation island and rhizosphere competence traits in the genome of root-associated *Pseudomonas stutzeri* A1501. Proc Natl Acad Sci U S A 105: 7564–7569. doi: 10.1073/pnas.0801093105.

115. Holt KE, Thomson NR, Wain J, Langridge GC, Hasan R, et al. (2009) Pseudogene accumulation in the evolutionary histories of *Salmonella enterica* serovars Paratyphi A and Typhi. BMC Genomics 10: 36. doi: 10.1186/1471-2164-10-36.

116. Durfee T, Nelson R, Baldwin S, Plunkett G, Burland V, et al. (2008) The complete genome sequence of *Escherichia coli* DH10B: insights into the biology of a laboratory workhorse. J Bacteriol 190: 2597–2606. doi: 10.1128/JB.01695-07.

117. Vorhölter FJ, Schneiker S, Goesmann A, Krause L, Bekel T, et al. (2008) The genome of Xanthomonas campestris pv. campestris B100 and its use for the reconstruction of metabolic pathways involved in xanthan biosynthesis. J Biotechnol 134: 33–45. doi: 10.1016/j.jbiotec.2007.12.013.

118. Warburton PE, Giordano J, Cheung F, Gelfand Y, Benson G (2004) Inverted repeat structure of the human genome: the X-chromosome contains a preponderance of large, highly homologous inverted repeats that contain testes genes. Genome Res 14: 1861–1869. doi: 10.1101/gr.2542904.

119. Drummond A, Ashton B, Cheung M, Heled J, Kearse M, et al. (2009). Geneious v4.8. Available from http://www.geneious.com/.

120. Chenna R, Sugawara H, Koike T, Lopez R, Gibson TJ, et al. (2003) Multiple sequence alignment with the Clustal series of programs. Nucleic Acids Res 31: 3497–3500.

121. Saitou N, Nei M (1987) The neighbor-joining method: a new method for reconstructing phylogenetic trees. Mol Biol Evol 4: 406–425.

122. Needleman SB, Wunsch CD (1970) A general method applicable to the search for similarities in the amino acid sequence of two proteins. J Mol Biol 48: 443–453.

123. Cline MS, Smoot M, Cerami E, Kuchinsky A, Landys N, et al. (2007) Integration of biological networks and gene expression data using Cytoscape. Nat Protoc 2: 2366–2382. doi: 10.1038/nprot.2007.324.

124. Hofmann K, Stoffel W (1993) TMbase - A database of membrane spanning proteins segments. Biol Chem Hoppe-Seyler 374: 166.

125. Gregory TR (2005) The Evolution of the Genome. Burlington, Massachusetts: Elsevier Academic Press.

126. Woods CR, Versalovic J, Koeuth T, Lupski JR (1992) Analysis of relationships among isolates of *Citrobacter diversus* by using DNA fingerprints generated by repetitive sequence-based primers in the polymerase chain reaction. J Clin Microbiol 30: 2921–2929.

127. Higgins CF, Ames GF, Barnes WM, Clement JM, Hofnung M (1982) A novel intercistronic regulatory element of prokaryotic operons. Nature 298: 760–762.

128. Gilson E, Saurin W, Perrin D, Bachellier S, Hofnung M (1991) Palindromic units are part of a new bacterial interspersed mosaic element (BIME). Nucleic Acids Res 19: 1375–1383.

129. Lupski JR, Weinstock GM (1992) Short, interspersed repetitive DNA sequences in prokaryotic genomes. J Bacteriol 174: 4525–4529.

130. Wilson LA, Sharp PM (2006) Enterobacterial repetitive intergenic consensus (ERIC) sequences in *Escherichia coli*: Evolution and implications for ERIC-PCR. Mol Biol Evol 23: 1156–1168. doi: 10.1093/molbev/msj125.

131. Bachellier S, Clément JM, Hofnung M (1999) Short palindromic repetitive DNA elements in enterobacteria: a survey. Res Microbiol 150: 627–639.

132. Aranda-Olmedo I, Tobes R, Manzanera M, Ramos J, Marques S (2002) Species-specific repetitive extragenic palindromic (REP) sequences in *Pseudomonas putida*. Nucleic Acids Research 30: 1826–1833.

133. Tobes R, Pareja E (2005) Repetitive extragenic palindromic sequences in the *Pseudomonas syringae* pv. *tomato* DC3000 genome: extragenic signals for genome reannotation. Res Microbiol 156: 424–433. doi: 10.1016/j.resmic.2004.10.014.

134. Barrangou R, Fremaux C, Deveau H, Richards M, Boyaval P, et al. (2007) CRISPR provides acquired resistance against viruses in prokaryotes. Science 315: 1709–1712. doi: 10.1126/science.1138140.

135. Nakazaki T, Okumoto Y, Horibata A, Yamahira S, Teraishi M, et al. (2003) Mobilization of a transposon in the rice genome. Nature 421: 170–172. doi: 10.1038/nature01219.

136. Csurös M, Noé L, Kucherov G (2007) Reconsidering the significance of genomic word frequencies. Trends Genet 23: 543–546. doi: 10.1016/j.tig.2007.07.008.

137. Espéli O, Moulin L, Boccard F (2001) Transcription attenuation associated with bacterial repetitive extragenic BIME elements. J Mol Biol 314: 375–386. doi: 10.1006/jmbi.2001.5150.

138. Rocco F, Gregorio ED, Nocera PPD (2010) A giant family of short palindromic sequences in *Stenotrophomonas maltophilia*. FEMS Microbiol Lett 308: 185–192. doi: 10.1111/j.1574-6968.2010.02010.x.

139. Clément JM, Wilde C, Bachellier S, Lambert P, Hofnung M (1999) IS*1397* is active for transposition into the chromosome of *Escherichia coli* K-12 and inserts

specifically into palindromic units of bacterial interspersed mosaic elements. J Bacteriol 181: 6929–6936.

140. Tobes R, Pareja E (2006) Bacterial repetitive extragenic palindromic sequences are DNA targets for Insertion Sequence elements. BMC Genomics 7: 62.

141. Elhai J, Kato M, Cousins S, Lindblad P, Costa JL (2008) Very small mobile repeated elements in cyanobacterial genomes. Genome Res 18: 1484–1499. doi: 10.1101/gr.074336.107.

142. McLafferty MA, Harcus DR, Hewlett EL (1988) Nucleotide sequence and characterization of a repetitive DNA element from the genome of Bordetella pertussis with characteristics of an insertion sequence. J Gen Microbiol 134: 2297–2306.

143. Ronning DR, Guynet C, Ton-Hoang B, Perez ZN, Ghirlando R, et al. (2005) Active site sharing and subterminal hairpin recognition in a new class of DNA transposases. Mol Cell 20: 143–154. doi: 10.1016/j.molcel.2005.07.026.

144. Guynet C, Hickman AB, Barabas O, Dyda F, Chandler M, et al. (2008) In vitro reconstitution of a single-stranded transposition mechanism of IS*608*. Mol Cell 29: 302–312. doi: 10.1016/j.molcel.2007.12.008.

145. Barabas O, Ronning DR, Guynet C, Hickman AB, Ton-Hoang B, et al. (2008) Mechanism of IS*200*/IS*605* family DNA transposases: activation and transposon-directed target site selection. Cell 132: 208–220. doi: 10.1016/j.cell.2007.12.029.

146. Kersulyte D, Velapatiño B, Dailide G, Mukhopadhyay AK, Ito Y, et al. (2002) Transposable element ISHp*608* of *Helicobacter pylori*: nonrandom geographic distribution, functional organization, and insertion specificity. J Bacteriol 184: 992–1002.

147. Bichsel M, Barbour AD, Wagner A (2010) The early phase of a bacterial insertion sequence infection. Theor Popul Biol doi: 10.1016/j.tpb.2010.08.003.

148. Lam S, Roth JR (1983) IS*200*: a *Salmonella*-specific insertion sequence. Cell 34: 951–960.

149. Beuzón CR, Chessa D, Casadesús J (2004) IS*200*: an old and still bacterial transposon. Int Microbiol 7: 3–12.

150. Kersulyte D, Akopyants NS, Clifton SW, Roe BA, Berg DE (1998) Novel sequence organization and insertion specificity of IS*605* and IS*606*: chimaeric transposable elements of *Helicobacter pylori*. Gene 223: 175–186.

151. Fayet O, Ramond P, Polard P, Prère MF, Chandler M (1990) Functional similarities between retroviruses and the IS3 family of bacterial insertion sequences? Mol Microbiol 4: 1771–1777.

152. Mazel D, Pochet S, Marlière P (1994) Genetic characterization of polypeptide deformylase, a distinctive enzyme of eubacterial translation. EMBO J 13: 914–923.

153. Giglione C, Pierre M, Meinnel T (2000) Peptide deformylase as a target for new generation, broad spectrum antimicrobial agents. Mol Microbiol 36: 1197–1205.

154. Clements JM, Beckett RP, Brown A, Catlin G, Lobell M, et al. (2001) Antibiotic activity and characterization of BB-3497, a novel peptide deformylase inhibitor. Antimicrob Agents Chemother 45: 563–570. doi: 10.1128/AAC.45.2.563-570.2001.

155. Koonin EV (2003) Comparative genomics, minimal gene-sets and the last universal common ancestor. Nat Rev Microbiol 1: 127–136. doi: 10.1038/nrmicro751.

156. Alexander PA, He Y, Chen Y, Orban J, Bryan PN (2007) The design and characterization of two proteins with 88% sequence identity but different structure and function. Proc Natl Acad Sci U S A 104: 11963–11968. doi: 10.1073/pnas.0700922104.

157. Ponting CP, Russell RR (2002) The natural history of protein domains. Annu Rev Biophys Biomol Struct 31: 45–71. doi: 10.1146/annurev.biophys.31.082901.134314.

158. Yang AS, Honig B (2000) An integrated approach to the analysis and modeling of protein sequences and structures. I. Protein structural alignment and a quantitative measure for protein structural distance. J Mol Biol 301: 665–678. doi: 10.1006/jmbi.2000.3973.

159. Nielsen R (2005) Molecular signatures of natural selection. Annu Rev Genet 39: 197–218. doi: 10.1146/annurev.genet.39.073003.112420.

160. Kimura M (1983) The neutral theory of molecular evolution. Cambridge University Press.

161. Cooper GM, Brown CD (2008) Qualifying the relationship between sequence conservation and molecular function. Genome Res 18: 201–205. doi: 10.1101/gr.7205808.

162. Kalia A, Mukhopadhyay AK, Dailide G, Ito Y, Azuma T, et al. (2004) Evolutionary dynamics of insertion sequences in Helicobacter pylori. J Bacteriol 186: 7508–7520. doi: 10.1128/JB.186.22.7508-7520.2004.

163. Veitia RA, Bottani S, Birchler JA (2008) Cellular reactions to gene dosage imbalance: genomic, transcriptomic and proteomic effects. Trends Genet 24: 390–397. doi: 10.1016/j.tig.2008.05.005.

164. Romero D, Palacios R (1997) Gene amplification and genomic plasticity in prokaryotes. Annu Rev Genet 31: 91–111. doi: 10.1146/annurev.genet.31.1.91.

165. Andersson DI, Slechta ES, Roth JR (1998) Evidence that gene amplification underlies adaptive mutability of the bacterial *lac* operon. Science 282: 1133–1135.

166. Bergthorsson U, Andersson DI, Roth JR (2007) Ohno's dilemma: evolution of new genes under continuous selection. Proc Natl Acad Sci U S A 104: 17004–17009. doi: 10.1073/pnas.0707158104.

167. Conant GC, Wolfe KH (2008) Turning a hobby into a job: How duplicated genes find new functions. Nat Rev Genet 9: 938–950. doi: 10.1038/nrg2482.

168. Sawyer SA, Dykhuizen DE, DuBose RF, Green L, Mutangadura-Mhlanga T, et al. (1987) Distribution and abundance of insertion sequences among natural isolates of Escherichia coli. Genetics 115: 51–63.

169. Ochman H, Lawrence JG, Groisman EA (2000) Lateral gene transfer and the nature of bacterial innovation. Nature 405: 299–304. doi: 10.1038/35012500.

170. Goddard MR, Burt A (1999) Recurrent invasion and extinction of a selfish gene. Proc Natl Acad Sci U S A 96: 13880–13885.

171. Koch AL (1981) Evolution of antibiotic resistance gene function. Microbiol Rev 45: 355–378.

172. Yang Z, Rannala B (1997) Bayesian phylogenetic inference using DNA sequences: a Markov Chain Monte Carlo Method. Mol Biol Evol 14: 717–724.

173. Mau B, Newton MA, Larget B (1999) Bayesian phylogenetic inference via Markov chain Monte Carlo methods. Biometrics 55: 1–12.

174. Felsenstein J (1981) Evolutionary trees from DNA sequences: a maximum likelihood approach. J Mol Evol 17: 368–376.

175. Siguier P, Gagnevin L, Chandler M (2009) The new IS1595 family, its relation to IS1 and the frontier between insertion sequences and transposons. Res Microbiol 160: 232–241. doi: 10.1016/j.resmic.2009.02.003.

176. Gourbeyre E, Siguier P, Chandler M (2010) Route 66: investigations into the organisation and distribution of the IS66 family of prokaryotic insertion sequences. Res Microbiol 161: 136–143. doi: 10.1016/j.resmic.2009.11.005.

177. Enright AJ, Dongen SV, Ouzounis CA (2002) An efficient algorithm for large-scale detection of protein families. Nucleic Acids Res 30: 1575–1584.

178. Hill CW, Sandt CH, Vlazny DA (1994) Rhs elements of Escherichia coli: a family of genetic composites each encoding a large mosaic protein. Mol Microbiol 12: 865–871.

179. Kjos M, Snipen L, Salehian Z, Nes IF, Diep DB (2010) The abi proteins and their involvement in bacteriocin self-immunity. J Bacteriol 192: 2068–2076. doi: 10.1128/JB.01553-09.

180. Touchon M, Rocha EPC (2007) Causes of insertion sequences abundance in prokaryotic genomes. Mol Biol Evol 24: 969–981. doi: 10.1093/molbev/msm014.

181. Jeltsch A, Pingoud A (1996) Horizontal gene transfer contributes to the wide distribution and evolution of type II restriction-modification systems. J Mol Evol 42: 91–96.

182. Godde JS, Bickerton A (2006) The repetitive DNA elements called CRISPRs and their associated genes: evidence of horizontal transfer among prokaryotes. J Mol Evol 62: 718–729. doi: 10.1007/s00239-005-0223-z.

183. Mira A, Ochman H, Moran NA (2001) Deletional bias and the evolution of bacterial genomes. Trends Genet 17: 589–596.

184. Kuo CH, Ochman H (2009) Deletional bias across the three domains of life. Genome Biol Evol 1: 145–152. doi: 10.1093/gbe/evp016.

185. Georg J, Hess WR (2011) cis-antisense RNA, another level of gene regulation in bacteria. Microbiol Mol Biol Rev 75: 286–300. doi: 10.1128/MMBR.00032-10.

186. Scott WG (2007) Ribozymes. Curr Opin Struct Biol 17: 280–286. doi: 10.1016/j.sbi.2007.05.003.

187. Cech TR (2000) Structural biology. The ribosome is a ribozyme. Science 289: 878–879.

188. Haugen P, Simon DM, Bhattacharya D (2005) The natural history of group I introns. Trends Genet 21: 111–119. doi: 10.1016/j.tig.2004.12.007.

189. Waters LS, Storz G (2009) Regulatory RNAs in bacteria. Cell 136: 615–628. doi: 10.1016/j.cell.2009.01.043.

190. Gripenland J, Netterling S, Loh E, Tiensuu T, Toledo-Arana A, et al. (2010) RNAs: regulators of bacterial virulence. Nat Rev Microbiol 8: 857–866. doi: 10.1038/nrmicro2457.

191. Stork M, Lorenzo MD, Welch TJ, Crosa JH (2007) Transcription termination within the iron transport-biosynthesis operon of Vibrio anguillarum requires an antisense RNA. J Bacteriol 189: 3479–3488. doi: 10.1128/JB.00619-06.

192. Giangrossi M, Prosseda G, Tran CN, Brandi A, Colonna B, et al. (2010) A novel antisense RNA regulates at transcriptional level the virulence gene icsA of Shigella flexneri. Nucleic Acids Res 38: 3362–3375. doi: 10.1093/nar/gkq025.

193. Crampton N, Bonass WA, Kirkham J, Rivetti C, Thomson NH (2006) Collision events between RNA polymerases in convergent transcription studied by atomic force microscopy. Nucleic Acids Res 34: 5416–5425. doi: 10.1093/nar/gkl668.

194. Sneppen K, Dodd IB, Shearwin KE, Palmer AC, Schubert RA, et al. (2005) A mathematical model for transcriptional interference by RNA polymerase traffic in Escherichia coli. J Mol Biol 346: 399–409. doi: 10.1016/j.jmb.2004.11.075.

195. Kawano M, Aravind L, Storz G (2007) An antisense RNA controls synthesis of an SOS-induced toxin evolved from an antitoxin. Mol Microbiol 64: 738–754. doi: 10.1111/j.1365-2958.2007.05688.x.

196. Massé E, Vanderpool CK, Gottesman S (2005) Effect of RyhB small RNA on global iron use in Escherichia coli. J Bacteriol 187: 6962–6971. doi: 10.1128/JB.187.20.6962-6971.2005.

197. Fozo EM, Kawano M, Fontaine F, Kaya Y, Mendieta KS, et al. (2008) Repression of small toxic protein synthesis by the Sib and OhsC small RNAs. Mol Microbiol 70: 1076–1093. doi: 10.1111/j.1365-2958.2008.06394.x.

198. Pichon C, Felden B (2008) Small RNA gene identification and mRNA target predictions in bacteria. Bioinformatics 24: 2807–2813. doi: 10.1093/bioinformatics/btn560.

199. Le SV, Chen JH, Currey KM, Maizel JV (1988) A program for predicting significant RNA secondary structures. Comput Appl Biosci 4: 153–159.

200. Rivas E, Eddy SR (2000) Secondary structure alone is generally not statistically significant for the detection of noncoding RNAs. Bioinformatics 16: 583–605.

201. Dandekar T, Hentze MW (1995) Finding the hairpin in the haystack: searching for RNA motifs. Trends Genet 11: 45–50.

202. Rivas E, Eddy SR (2001) Noncoding RNA gene detection using comparative sequence analysis. BMC Bioinformatics 2: 8.

203. Hurst LD, Merchant AR (2001) High guanine-cytosine content is not an adaptation to high temperature: a comparative analysis amongst prokaryotes. Proc Biol Sci 268: 493–497. doi: 10.1098/rspb.2000.1397.

204. Shannon C (1951) Prediction and entropy of printed English. Bell System Technical Journal 30: 50–64.

205. Szekeres S, Dauti M, Wilde C, Mazel D, Rowe-Magnus DA (2007) Chromosomal toxin-antitoxin loci can diminish large-scale genome reductions in the absence of selection. Mol Microbiol 63: 1588–1605. doi: 10.1111/j.1365-2958.2007.05613.x.

206. Ton-Hoang B, Pasternak C, Siguier P, Guynet C, Hickman AB, et al. (2010) Single-stranded DNA transposition is coupled to host replication. Cell 142: 398–408. doi: 10.1016/j.cell.2010.06.034.

207. Shen MR, Batzer MA, Deininger PL (1991) Evolution of the master Alu gene(s). J Mol Evol 33: 311–320.

208. Deininger PL, Batzer MA, Hutchison CA, Edgell MH (1992) Master genes in mammalian repetitive DNA amplification. Trends Genet 8: 307–311.

209. Tachida H (1996) A population genetic study of the evolution of SINEs. II. Sequence evolution under the master copy model. Genetics 143: 1033–1042.

210. Kleckner N (1981) Transposable elements in prokaryotes. Annu Rev Genet 15: 341–404. doi: 10.1146/annurev.ge.15.120181.002013.

211. Frost LS, Leplae R, Summers AO, Toussaint A (2005) Mobile genetic elements: the agents of open source evolution. Nat Rev Microbiol 3: 722–732. doi: 10.1038/nrmicro1235.

212. Leplae R, Geeraerts D, Hallez R, Guglielmini J, Drèze P, et al. (2011) Diversity of bacterial type II toxin-antitoxin systems: a comprehensive search and functional analysis of novel families. Nucleic Acids Res 39: 5513–5525. doi: 10.1093/nar/gkr131.

213. Cooper TF, Paixão T, Heinemann JA (2010) Within-host competition selects for plasmid-encoded toxin-antitoxin systems. Proc Biol Sci 277: 3149–3155. doi: 10.1098/rspb.2010.0831.

214. Wagner A (2006) Cooperation is fleeting in the world of transposable elements. PLoS Comput Biol 2: e162. doi: 10.1371/journal.pcbi.0020162.

215. Chopin MC, Chopin A, Bidnenko E (2005) Phage abortive infection in lactococci: variations on a theme. Curr Opin Microbiol 8: 473–479. doi: 10.1016/j.mib.2005.06.006.

216. Wadkins RM (2000) Targeting DNA secondary structures. Curr Med Chem 7: 1–15.

217. Merino E, Becerril B, Valle F, Bolivar F (1987) Deletion of a repetitive extragenic palindromic (REP) sequence downstream from the structural gene of

Escherichia coli glutamate dehydrogenase affects the stability of its mRNA. Gene 58: 305–309.

218. Fozo EM, Makarova KS, Shabalina SA, Yutin N, Koonin EV, et al. (2010) Abundance of type I toxin-antitoxin systems in bacteria: searches for new candidates and discovery of novel families. Nucleic Acids Res 38: 3743–3759. doi: 10.1093/nar/gkq054.

219. Schumacher MA, Piro KM, Xu W, Hansen S, Lewis K, et al. (2009) Molecular mechanisms of HipA-mediated multidrug tolerance and its neutralization by HipB. Science 323: 396–401. doi: 10.1126/science.1163806.

220. Yamaguchi Y, Inouye M (2011) Regulation of growth and death in Escherichia coli by toxin-antitoxin systems. Nat Rev Microbiol 9: 779–790. doi: 10.1038/nrmicro2651.

221. Venter JC, Adams MD, Myers EW, Li PW, Mural RJ, et al. (2001) The sequence of the human genome. Science 291: 1304–1351. doi: 10.1126/science.1058040.

222. Lander ES, Linton LM, Birren B, Nusbaum C, Zody MC, et al. (2001) Initial sequencing and analysis of the human genome. Nature 409: 860–921.

223. Margulies M, Egholm M, Altman WE, Attiya S, Bader JS, et al. (2005) Genome sequencing in microfabricated high-density picolitre reactors. Nature 437: 376–380. doi: 10.1038/nature03959.

224. Schuster SC (2008) Next-generation sequencing transforms today's biology. Nat Methods 5: 16–18. doi: 10.1038/nmeth1156.

225. Rodley CDM, Bertels F, Jones B, O'Sullivan JM (2009) Global identification of yeast chromosome interactions using Genome conformation capture. Fungal Genet Biol 46: 879–886. doi: 10.1016/j.fgb.2009.07.006.

226. Kugelberg E, Kofoid E, Andersson DI, Lu Y, Mellor J, et al. (2010) The tandem inversion duplication in *Salmonella enterica*: selection drives unstable precursors to final mutation types. Genetics 185: 65–80. doi: 10.1534/genetics.110.114074.

227. Jiang N, Feschotte C, Zhang X, Wessler SR (2004) Using rice to understand the origin and amplification of miniature inverted repeat transposable elements (MITEs). Curr Opin Plant Biol 7: 115–119. doi: 10.1016/j.pbi.2004.01.004.

228. Bateman A, Coin L, Durbin R, Finn RD, Hollich V, et al. (2004) The Pfam protein families database. Nucleic Acids Res 32: D138–D141. doi: 10.1093/nar/gkh121.

229. Kofoid E, Bergthorsson U, Slechta ES, Roth JR (2003) Formation of an F' plasmid by recombination between imperfectly repeated chromosomal Rep sequences: a closer look at an old friend (F'(128) *pro lac*). J Bacteriol 185: 660–663.

230. Casjens S (1998) The diverse and dynamic structure of bacterial genomes. Annu Rev Genet 32: 339–377. doi: 10.1146/annurev.genet.32.1.339.

Appendices

A1 Appendix material from chapter 3

A1.1 Figures

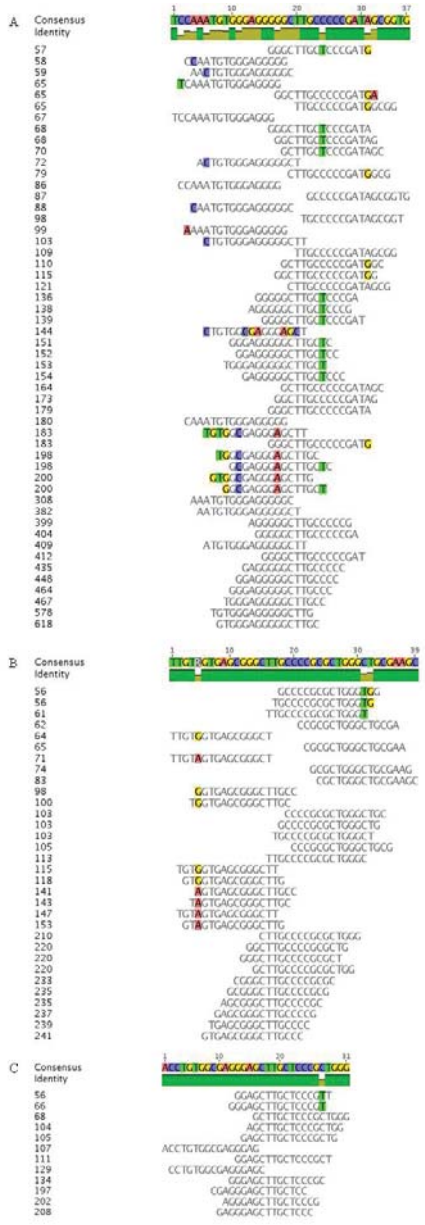


Figure A1.1. Alignments of the most abundant sequence groups in SBW25. GI sequences are shown in (A), GII sequences in (B) and GIII sequences in (C). The consensus sequence contains the respective palindromic cores (framed in red). Numbers to the left of the alignment denote the frequency of the respective 16-mer (e.g. the first 16-mer in (A) GGGCTTGCTCCCGATG occurs 57 times). Coloured nucleotides within the alignment denote differences to the consensus sequence**.**

A

```
Read 1
@2_21_784_925
CGGCAGAGGTGATACTGGATGCGGTGGAGAGTGATGTGCTGGTGCTCAAGCCGCAGGCGTTGATGGATCAGCTCGN
+
AA?@BB<@@=;>@A1>?>@B>1=8..71<49'@4:>6A=;BA;A4>42?;,5=566####################
Read 2
@2_21_784_925
TTCCAACTGACAACGGATAACCCCTCCAACATTTTGATCTCCATCGTTCATAAGGTTGGATCAGGGCTTGNTNNNN
+
BC;BBCBBCCABC:BBBBBB?B>AB>BAB7B>ABBBABAB@?BB:C=B(>B==BA4@><@>9@<A###########
Read 1
@1_82_1308_1969
TACTACAAGGTCAACGACAAAGTGCGCCTGAACCTCGACGTGAAGAACCTGTTCAACCGCGAGTATGAAGAACGCG
+
6@AABCA>B>>;@?8>?@?A<4;>B<9?@8;A7)=B>A:B6BA@>B:'03>-<:>>?-<B@?A?<>=6@A53&7;@
Read 2
@1_82_1308_1969
ATACCCAGGACCCCCTCCCACATTGAAGCGGTGTACGCCGTCATAGCGTGTAGGCGAATGCGATGTGGACGGGCAN
+
AAAA99>A5>5:5?5@3<5>=AB@CB@:<A@/=7?99773&1,##################################
Read 1
@2_70_1540_677
GACATAACCACCAATCACCACAAAGGCCGTAACAGCGTCACCTGACACAACGCCGATCAAACTGTGGGAGGGGGAT
+
BCBCCCCBBCBBCCCCBBBCBBB?=AA;B?BB>BC@AABCBAB@BBCBBBBBBB@@=?=>@/BA<;AB@B;A@A29
Read 2
@2_70_1540_677
ATTATTGTGATTGCACCGTTTGGATATTCACGCAGATCCCCCTCCCACAGTTTGATCGGCGTTGTGTCAGGTGAAG
+
BCCCCCBCBBCCCBCBCBBCCCCCBBCBBBACACBCA@A@>@B@A@A=A@8=A@@=6?>AB@;;;:==><@/B9##
Read 1
@1_13_1051_777
CTGTCACCACACGCACAGGTGACGGGTGGTCGAGAGTAAAATCGTTCGCAAGCTACTTATCTATTTGCGACGCGCA
+
BA@A<A;A@:??A?6<9:/7;>?9'3,'82;A9?=A4177>>9@A?9=:4>207?5<<@?5;//67/33<@#####
Read 2
@1_13_1051_777
GCGGGCAGCGCAGTAGAGTGTTGAAGACTGTGTAGTGAGCGCCGGAGCGAAAAGCGCCACCTTTTGGCGCGGTGNN
+
B9AB9==@'@67BBB?:B>AABB6BB<16;2>,=;3.:(,B16A==;,5:366#########################
Read 1
@1_12_1235_507
GCAGTTCAGGGCCTAGTATCGCCTGTACCGACCTCATCGGGGGCAAGCCCCCGATAGCGGTTTCAGCGAGATATGA
+
BA>A?A?:;>9A:A??;?A443<:;7769<4357:;?4992494331303 2#########################
Read 2
@1_12_1235_507
CGCTGAAACCGCTATCGGGGGCTTGCCCCCGATGAGGTCGGTACAGGCGATACTAAGGCCCGAAATGCANATNNNN
+
BCBCCCCCBBCBCCBA@AABAAA?B?>?=AA?AB3><*=A=####################################
```

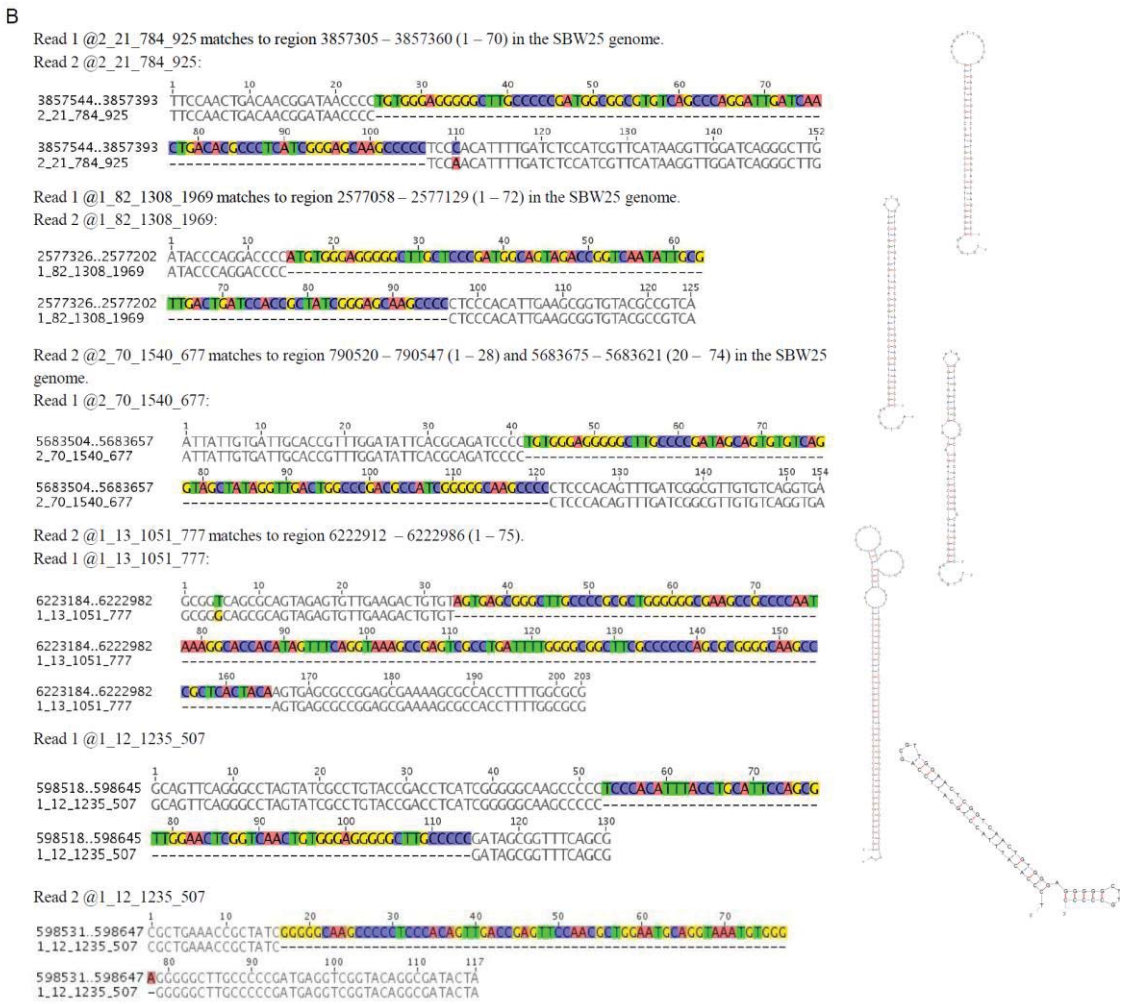


Figure A1.2. Excision events detected in Illumina sequencing data. (*A*) Shows fastq formatted raw Illumina sequences for the excision events and their corresponding paired ends or 'mates'. Quality scores are the last line of each fastq entry. (*B*) In all cases Read 1 matches to a position close to the corresponding Read 2 as expected for paired end reads. The alignments show the match between the sequence reads (second line in the alignment) and the SBW25 genome (first line in the alignment). Colored nucleotides show differences between genome and sequence read. Secondary structure predictions of the excised sequences are shown on the right. For the fourth excision a total of 200 sequence reads were found showing the same event, indicating that the entire REPIN was excised from the genome.

A1.2 Tables

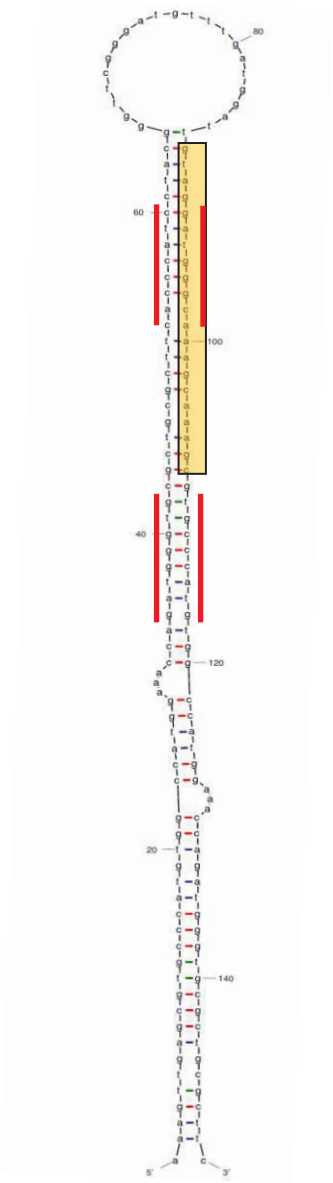
Table A1.1 Dinucleotide frequencies in *P. fluorescens* Pf0-1 and SBW25.

Di-nucleotides	Pf0-1	SBW25	Difference to Pf0-1
AA	0.047167435	0.046317477	2%
AC	0.055368846	0.054939965	1%

AG	0.051895004	0.052577167	-1%
AT	0.043838038	0.042889754	2%
CA	0.070204044	0.072515172	-3%
CC	0.077093329	0.081420588	-6%
CG	0.104089616	0.095726792	8%
CT	0.051886151	0.052746299	-2%
GA	0.063428452	0.056363832	11%
GC	0.107768012	0.109277776	-1%
GG	0.076447517	0.081610249	-7%
GT	0.054290163	0.055384886	-2%
TA	0.017469547	0.021527881	-23%
TC	0.063042953	0.056770672	10%
TG	0.069502162	0.072722386	-5%
TT	0.046504227	0.047209105	-2%

A2 Appendix material from chapter 4

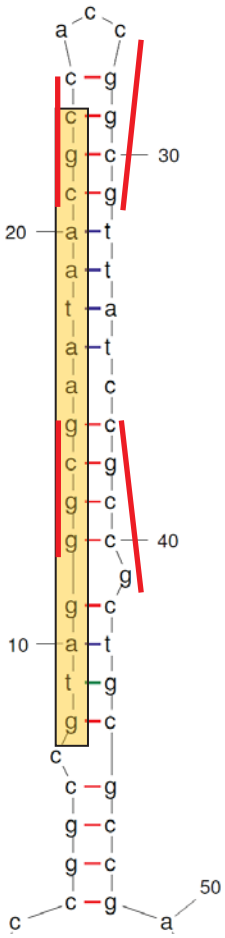
A2.1 Figures



Thioalkalivibrio sp. HL-EbGR7

Position: 1417740..417890

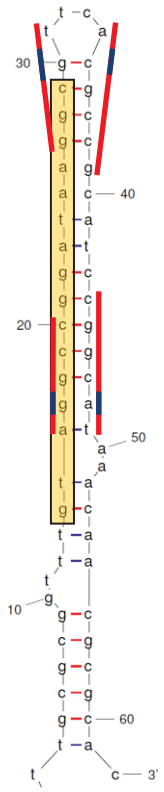
16-mer found adjacent to *tgr7_1317*



Pseudomonas aeruginosa PA7

Position: 878990..879046

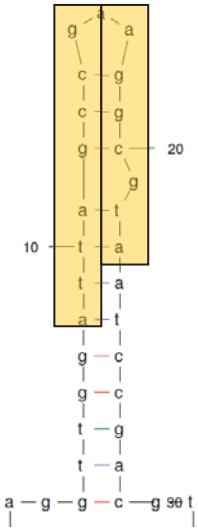
16-mer found adjacent to *pspa7_4226*



Escherichia coli K-12 DH10B

Position: 868786..868847

16-mer found adjacent to *yafM*

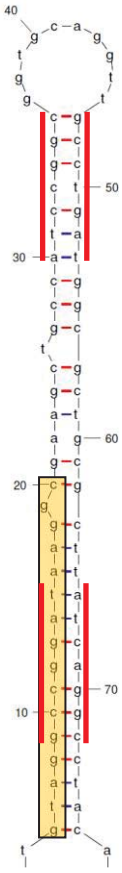


Pseudomonas stutzeri A1501

Position: 1162127_1162158

16-mer found adjacent to *pst_1052*

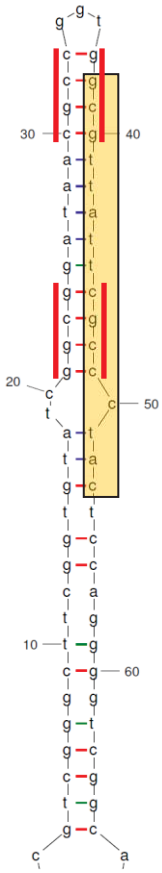
No typical REPIN formation, only found adjacent to *pst_1052*. Perhaps no dissemination possible.



Salmonella enterica serovar Paratyphi A AKU 12601

Position: 298766..298843

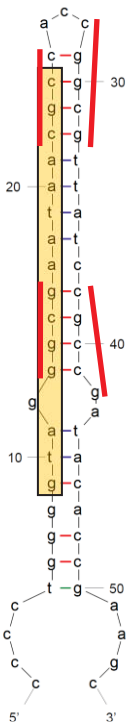
16-mer found adjacent to *sspa4070*

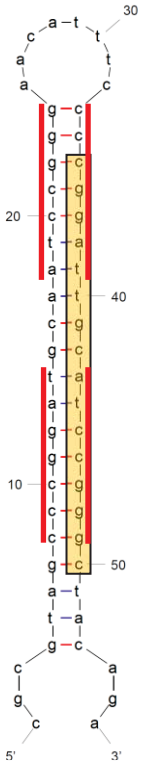


Pseudomonas aeruginosa PAO1 PA1154

Position: 264851..264919

16-mer found adjacent *pa1154*





Pseudomonas aeruginosa LESB58

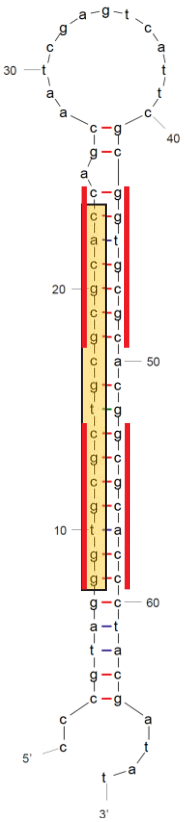
Position: 257386..257439

16-mer found adjacent to *pales_41671*

Pseudomonas mendocina ymp

Position: 213188..213243

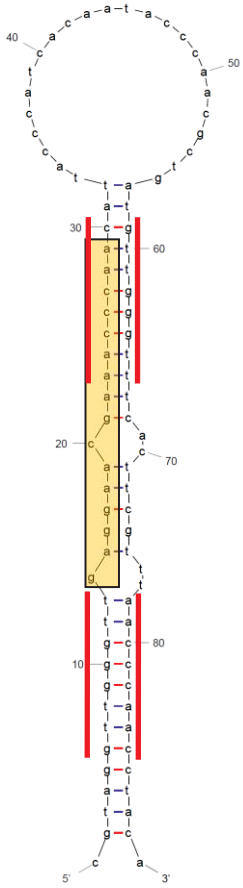
16-mer found adjacent to *pmen0731*



Pseudomonas mendocina ymp

Position: 58161..58228

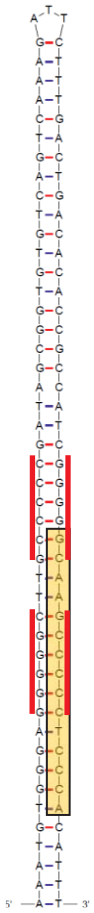
16-mer found adjacent to *pmen3135*



Nostoc punctiforme PCC 73102

Position: 684772400..6847813

16mer found adjacent to *npunF5543*



Pseudomonas fluorescens SBW25

GI consensus structure

Figure A2.1. REPIN secondary structures found in different genomes predicted by the mfold webserver (http://mfold.rna.albany.edu/). Red bars show palindromic parts of the structure. The yellow box indicates the most abundant 16-mer found in the non-coding flanking DNA of the respective RAP. The GI consensus sequence from *Pseudomonas fluorescens* SBW25 is the only REPIN shown from RAYT clade I (Figure 4.3), all other REPINs are associated to RAPs from clade II.

A2.2 Tables

Table A2.1. Most abundant words in the non-coding DNA flanking RAYTs.

Organism	Name of RAYThomologue	Pos[a]	Most abundant 16-mer in non-coding DNA flanking RAP	Freq	p-Value[b]	Palindromes (all 16-mers within the palindrome occur at least twice within the genome)
P. fluorescens SBW25	*yafM*	5′	gggcaagcccgctcac	241	2.00E-06	gcgggcaagcccgc
P. fluorescens SBW25	*yafM*	3′	gggcaagcccgctcac	241	2.00E-06	gggctgcttcgcagcc c
P. fluorescens	*pflu2165*	5′	gagggagcttgctccc	208	3.55E-06	gggagcttgctccc

SBW25						
P. fluorescens SBW25	*pflu2165*	3′	gagggagcttgctccc	208	3.55E-06	gggagcttgctccc
P. syringae phaseolicola 1448A	*pspph_4464*	3′	gcaagctcgctcccac	28	4.57E-04	gcgagcaagctcgc
P. syringae phaseolicola 1448A	*pspph_4464*	5′	acgatgcgactttgcc	1	1	none
P. syringae syringae B728a	*psyr_4421*	3′	tcgcgagcaagctcgc	88	4.41E-06	gcgagcaagctcgc
P. syringae syringae B728a	*psyr_4421*	5′	atgtgattgtgatctc	1	1	none
P. syringae syringae B728a	*psyr_4707*	3′	ttcgcgaacaagttcg	201	3.40E-07	gcgaacaagttcgc
P. syringae syringae B728a	*psyr_4707*	5′	gtgtcgttgcgcaatg	1	1	none
P. syringae phaseolicola 1448A	*pspph_5043*	3′	acggcgtgccactgcg	1	1	none
P. syringae phaseolicola 1448A	*pspph_5043*	5′	ggagcggacttgtccg	42	2.40E-04	gcggacttgtccgc
P. syringae tomato DC3000	*pspto_0262*	3′	gcgtgccgctgcgcaa	3	0.004	none
P. syringae tomato DC3000	*pspto_0262*	5′	gagcggacttgtccgc	39	1.81E-04	gcggacttgtccgc
P. fluorescens SBW25	*pflu3939*	5′	gcaagccccctcccac	618	1.54E-07	gggggcaagccccc
P. fluorescens SBW25	*pflu3939*	3′	gtgggaggggcttgc	618	1.54E-07	gggggcttgccccc
P. fluorescens Pf-5	*pfl_3160*	5′	tcgccggcaagccggc	358	1.48E-07	gccggcaagccggc
P. fluorescens Pf-5	*pfl_3160*	3′	tcgccggcaagccggc	358	1.48E-07	gccggcaagccggc
P. entomophila L48	*pseen5170*	5′	gtaggagccagcttgc	95	1.62E-05	gccagcttgctggcg
P. entomophila L48	*pseen5170*	3′	aacactttatccacag	2	0.04	none
T. sp HL-EbGR7	*tgr7_2777*	3′	tcggcctgaaggccga	50	1.59E-05	gtcggcctgaaggccgac
T. sp HL-EbGR7	*tgr7_2777*	5′	tcgggctgaagcccga	80	6.01E-07	gtcgggctgaagcccgac
P. putida W619	*pputw619_5047*	5′	gatcgccggcaagccg	20	2.66E-05	cggcaagccg
P. putida W619	*pputw619_5047*	3′	tcgccggcaagccggc	229	1.79E-07	ggcaagccggcttgcc
P. entomophila L48	*pseen4846*	5′	caaggccgctcccaca	181	4.51E-06	gggccgctgtgcggccc

P. entomophila L48	*pseen4846*	3′	caaggccgctcccaca	181	4.51E-06	gcgacacaaggccgc
P. putida KT2440	*pp_0568*	5′	tgtgggagcggccttg	54	9.65E-06	gcggccttgcgtcgc
P. putida KT2440	*pp_0568*	3′	caaggccgctcccaca	54	9.65E-06	gcgacacaaggccgc
P. putida F1	*pput_0607*	5′	atgagggcgaagccct	2	0.03	tgagggcgaagccctc a
P. putida F1	*pput_0607*	3′	caaggccgctcccaca	140	3.50E-06	cgctcccacagggacc gcg
P. putida W619	*pputw619_4597*	3′	gcggccttgtgtcgcg	148	1.07E-06	ggggctgccttgcagc cc
P. putida W619	*pputw619_4597*	5′	caaggccgctcctaca	119	3.21E-06	ccgctcctacagggg
P. putida GB1	*pputgb1_0613*	5′	tcgcgacacaaggccg	235	1.71E-07	ggggccgctttgcggc cc
P. putida GB1	*pputgb1_0613*	3′	tcgcgacacaaggccg	235	1.71E-07	cgcgacacaaggccgc tcctacagggatcgcg
P. putida GB1	*pputgb1_5236*	5′	aacccgctcccacagg	62	6.32E-06	gcgggtgaacccgc
P. putida GB1	*pputgb1_5236*	3′	tcgcgggtaaacccgc	90	3.07E-06	gcgggtaaacccgc
P. putida KT2440	*pp_5176*	5′	agcccgcgaagaggcc	26	3.20E-05	ctcttcgcgggcgagc ccgcgaagag
P. putida KT2440	*pp_5176*	3′	cctgtgggagcgggcg	86	5.25E-06	gcgggcgtgcccgc
P. putida F1	*pput_5083*	5′	cgggcgagcccgcgaa	33	4.07E-05	ggcctcttcgcgggcg agcccgcgaagaggcc
P. putida F1	*pput_5083*	3′	gcccgctcccacaggg	70	1.19E-05	gcgggcatgcccgc
P. putida GB1	*pputgb1_1364*	5′	gccgcccgcgcggcgc	35	1.33E-05	agcgccgcccgcgcgg cgct
P. putida GB1	*pputgb1_1364*	3′	gccgcccgcgcggcgc	35	1.33E-05	gcgccgcccgcgcggc gc
P. entomophila L48	*pseen3227*	5′	gcggattcatccgcga	151	6.50E-06	gcggattcatccgc
P. entomophila L48	*pseen3227*	3′	tcgcggatgaatccgc	151	6.50E-06	gcggatgaatccgc
P. putida F1	*pput_3919*	5′	cgggtttacccgcgaa	404	1.75E-07	gcgggtttacccgc
P. putida F1	*pput_3919*	3′	cctcaccccagccgcg	2	0.03	None
P. mendocina ymp	*pmen_3135*	5′	ggtgcgcacggcgcac	198	2.08E-07	ggtgcgcacggcgcac c
P. mendocina ymp	*pmen_3135*	3′	ggtgcgcacggcgcac	198	2.08E-07	ggtgcgcacggcgcac c
T. sp. HL-EbGR7	*tgr7_1317*	5′	gtaggatgggcaaagc	14	1.43E-04	atgggcaaagcgatag cgtgcccat
T. sp. HL-EbGR7	*tgr7_1317*	3′	gtaggatgggcaaagc	14	1.43E-04	atgggcaaagcaacgc gtgcccat
N. punctiforme PCC 73102	*npun_f5543*	5′	gaggaacgaaacccaa	13	4.39E-04	gttgggttgaggaacg aaacccaac
N. punctiforme PCC 73102	*npun_f5543*	3′	atgttgggtttcgttc	13	4.39E-04	gttgggtttcgttcct caacccaac

P. mendocina ymp	*pmen_0731*	5′	cggattgcatccgggc	93	3.12E-06	cccggattgcatccgggg
P. mendocina ymp	*pmen_0731*	3′	cggattgcatccgggc	93	3.12E-06	cccggattgcatccgggg
P. stutzeri A1501	*pst_1052*	5′	attagccgaaggcgta	4	0.0029	tggattagccgaaggcgtaatccg
P. stutzeri A1501	*pst_1052*	3′	aaacgacggaagcgcc	2	0.03	None
S. enterica serovar Paratyphi A AKU 12601	*sspa4070*	5′	cgcttaccgggcctac	18	7.57E-06	gcccggtggcgcttcgcttaccgggc
S. enterica serovar Paratyphi A AKU 12601	*sspa4070*	3′	gtaggccggataaggc	57	2.23E-07	aggccggataaggcgt
E coli K-12 DH10B	*yafM*	5′	tgcctgatgcgacgct	77	2.26E-06	gcctgatgcgacgctggcgcgtcttatcatgc
E. coli K-12 DH10B	*yafM*	3′	gtaggccggataaggc	106	2.26E-07	aggccggataaggcgt
P. aeruginosa PAO1	*pa1154*	5′	gcgttattcgccctac	30	1.02E-06	gcgttattcgc
P. aeruginosa PAO1	*pa1154*	3′	gcgttattcgccctac	30	1.02E-06	gcgttattcgc
P. aeruginosa PA7	*pspa7_4226*	5′	gtagggcgaataacgc	7	3.15E-04	gcgaataacgc
P. aeruginosa PA7	*pspa7_4226*	3′	gtagggcgaataacgc	7	3.15E-04	gcgaataacgc
P. aeruginosa LESB58	*pales_41671*	5′	gtagggcgaataacgc	26	8.11E-07	gcgaataacgc
P. aeruginosa LESB58	*pales_41671*	3′	gtagggcgaataacgc	26	8.11E-07	gcgaataacgc

All homologues are flanked by at least one 16-mer that is unusually over-represented within the respective genome of the bacterium. In all cases the 16-mer contains or is part of a palindrome or inverted repeat. Letters in red denote complementary base pairs. [a] Denotes whether the 16-mer was found in the extragenic space flanking the RAYT on the 5′ or 3′ side. [b] Proportion of different words that occur equally or more often than the most abundant 16-mer from the non-coding DNA flanking the RAYT homologue.

Table A2.2. Details concerning the analysis of REP sequences in other bacterial genomes. Please download table under:

http://www.plosgenetics.org/article/fetchSingleRepresentation.action?uri=info:doi/10.1371/journal.pgen.1002132.s014

A3 Appendix materials for chapter 6

A3.1 Figures

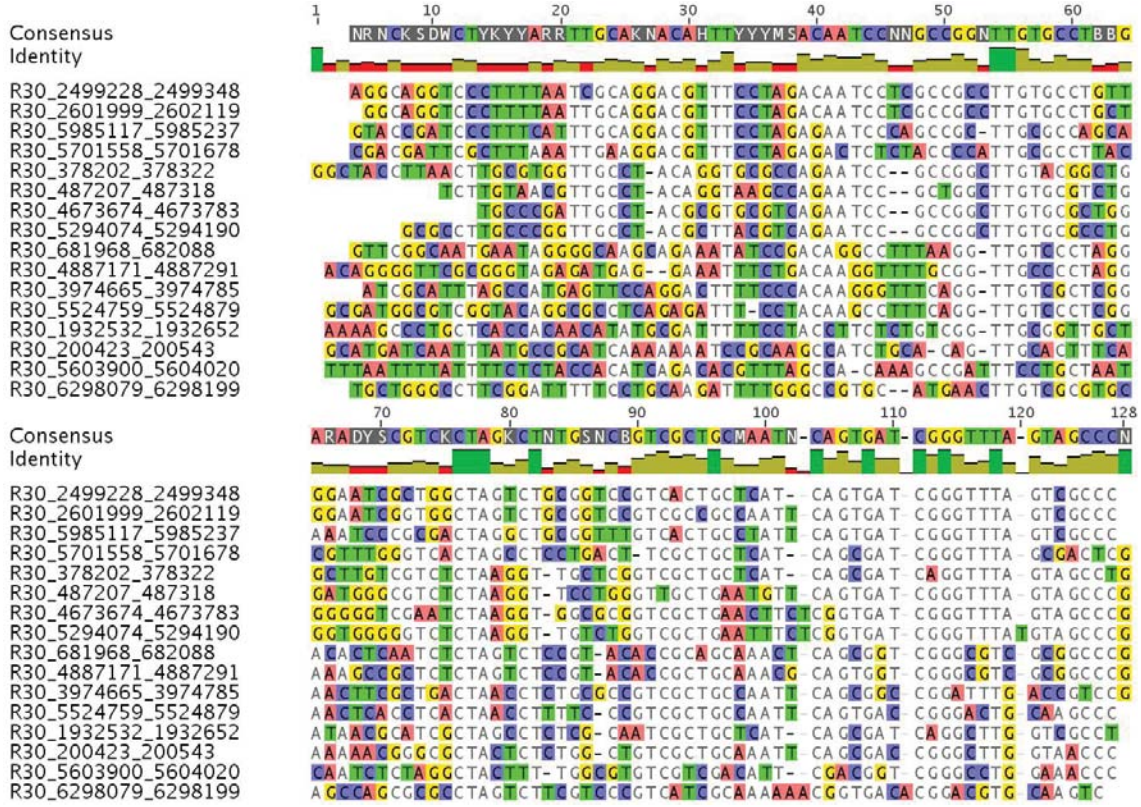


Figure A3.1. Alignment of R30 sequences found at the 5′ end of R178 repeats. Name shows start and end of sequence in SBW25. Uncoloured nucleotides are conserved in more than 50% of the sequences.

A3.2 Tables

Table A3.1. The entropy observed for each position in the nucleotide alignment of R178 (see Figure 6.2).

Alignment Position	Entropy
1	1.061278
2	1.410848
3	1.23266
4	0.309543
5	1.568318
6	0.309543
7	0.503258
8	0
9	0
10	0.991076

11	0.503258
12	0
13	0
14	0
15	0
16	0.309543
17	0
18	0.764205
19	0.309543
20	0
21	0.944489
22	0.991076
23	0
24	0
25	0.991076
26	1.480682
27	0.991076
28	1.052941
29	0
30	0
31	0.803072
32	0.918296
33	0.309543
34	0.914183
35	0
36	0.309543
37	0.944489
38	0.918296
39	1.480682
40	0.991076
41	0.309543
42	0
43	0
44	0
45	0.309543
46	1.19946
47	1.19946
48	0.640206
49	1.263933
50	1.569445
51	1.448816
52	1.923795
53	1.573989
54	1.925127
55	1.530125
56	1.545152
57	1.384432
58	1.903968

59	1.677421
60	0.322757
61	0.309543
62	0
63	0
64	0
65	0.522559
66	1.299737
67	1.615805
68	1.052941
69	1.392147
70	0.614369
71	1.5
72	0.309543
73	0
74	0
75	0.764205
76	0
77	0.503258
78	0.309543
79	0.918296
80	0
81	0.503258
82	0.309543
83	0
84	0
85	0.852405
86	0
87	0
88	0
89	0.322757
90	0
91	0
92	0.614369
93	0.696212
94	1.198184
95	1.556657
96	1.974938
97	1.419737
98	0.834347
99	0.309543
100	0.503258
101	0.964079
102	1.500272
103	0
104	1.633731
105	1.233225
106	1.615805

107	1.446648
108	0.543564
109	0.33729
110	0.33729
111	0.764205

Table A3.2. Binding sites of R178 B and C loop sequences (see Figure 6.1A).

Position of R178	B loop sequence	Binds to promoter of	Distance to start codon of associated gene	C loop sequence	Distance to R178 5′ end
200256	TCCTTGATT	3′ peptide not annotated	18-9	TTACCAA	69-76
378375	TTCTACATTC	PFLU0347 3′ peptide	11-1	CTACTAA	56-63 (5′ end of R30)
487360	CGTCTCC	PFLU0440 3′ peptide	11-4	CTACTAA	45-52 (5′ end of R30)
682144	TCCTTGATCT Binds to TCCTTAATCT	PFLU0601 3′ peptide (wrong start)	18-8	CCGCGA	59-65
1932384	TCCATTC	PFLU1768 3′ peptide (wrong start)	17-10	No proper C loop TTGGTCG But proper D loop: CGACCAA	49-56
2499097	TTCCTGTTCTGA Binds to: TTCCGTG-CTGA	PFLU2297 3′ peptide	18-8	CGACTAA	34-41
2601865	CCTTTGG	PFLU2387 3′ peptide	18-10	CGACTAA	35-42
3974845	CTTTGGGA	PFLU3586 3′ peptide (wrong start)	18-10	CGGTCAA	64-70
4673835	AGATATCC	3′ peptide (not annotated)	11-3	CTACTAA	55-62 (5′ of R30)

4720144	CTTTGTGTGTTTCC	No associated peptide or other promoter found that motif binds to		CTACTCA Only partially found within repeat: AGTAG	none
4887358	TCCATTGGTAA	PFLU4421 3′ peptide	19-8	CCGCGA	70-76
5294223	TTTGTGTGGGA	Binds 3′ of 3′ peptide (233 bp away) it also seems to bind to Ala tRNA promoters	16-5 (for tRNA)	CTACTAA Not found in 5′ ex space, however R30 found and CTACATAA	36-44
5524593	TCCTAACAA	PFLU5029 3′ peptide	17-8	CTTGCAG	67-74
5603731	TCCTATTG	cysK	14-6	TTTCCAG	71-77
5701732	TTCCCAAATTTAC	PFLU5205 3′ peptide (wrong start)	18-5	TCGCTAA	57-64
5914200	B loop only consists of three bases			GTGTGC	none
5984964	CTAGCGG	3′ peptide (not annotated)	18-11	CGACTAA	55-62
6297914	TCCGATATG	3′ peptide (not annotated)	19-10	TTGCAC	66-72

A4 Source code

A4.1 Generation of randomized genomes

```
//Write random genome with given dinucleotide sequence.
public static String generateSequence(HashMap<Character,HashMap<Character,Double>> hm,double GC,int length){
          StringBuffer seq=new StringBuffer();
          seq.append(GC<Math.random()?0.5<Math.random()?'A':'T':0.5<Math.random()?'C':'G');
          for(int i=1;i<length;i++){
                    double rand=Math.random();
                    HashMap<Character,Double> temp=hm.get(seq.charAt(i-1));
                    double A=temp.get('A')/temp.get('S');
                    double T=A+temp.get('T')/temp.get('S');
                    double C=T+temp.get('C')/temp.get('S');
                    if(rand<A){
                              seq.append('A');
                    }else if(rand<T)seq.append('T');
                    else if(rand<C)seq.append('C');
                    else seq.append('G');
          }
          return seq.toString();
}
//Read Dinucleotide sequence from a given genome.
private static HashMap<Character,HashMap<Character,Double>> readDinuc(File in){
          HashMap<Character,HashMap<Character,Double>> hm=new HashMap<Character, HashMap<Character,Double>>();
          try{
                    BufferedReader br=new BufferedReader(new FileReader(in));
                    String line="";
                    ArrayList<Character> list=new ArrayList<Character>();
                    while((line=br.readLine())!=null){
                              String[] split=line.split("\\s+");
                              split[0]=split[0].toUpperCase();
                              char first=split[0].charAt(0);
                              char sec=split[0].charAt(1);
                              double p=Double.parseDouble(split[1]);
                              if(hm.containsKey(first))hm.get(first).put(sec, p);
                              else{
                                        list.add(first);
                                        HashMap<Character,Double> temp=new HashMap<Character, Double>();
                                        temp.put(sec,p);
                                        hm.put(first,temp);
                              }
                    }
                    br.close();
                    for(int i=0;i<list.size();i++){
                              double sum=0;
                              for(int j=0;j<list.size();j++){
                                        sum+=hm.get(list.get(i)).get(list.get(j));
                              }
                              hm.get(list.get(i)).put('S',sum);
                    }

          }catch(IOException e){
                    e.printStackTrace();
          }
          return hm;
}
```

A4.2 Frequency determination of most abundant oligonucleotides

```
//Determine frequency of all short sequences (words) within a specific sequence (e.g. genome).
private static void writeWords(int start,int wl,String genome,File out){

          HashMap<BitSet,Integer> wordsBitSet=new HashMap<BitSet, Integer>();

          for(int i=start;i<=wl;i++){
                    for(int j=0;j<genome.length()-1-i;j++){
                              String key=genome.substring(j, j+i);
                              BitSet key2BitSet=new BitSet();
                              key2BitSet=DNAmanipulations.codeDNA(key.toUpperCase());
                              if(key2BitSet==null){
                                        continue;
                              }
                              BitSet complement=DNAmanipulations.reverse(key2BitSet);
                              if(wordsBitSet.containsKey(key2BitSet) ){
                                        wordsBitSet.put(key2BitSet,wordsBitSet.get(key2BitSet)+1);
                              }else if( wordsBitSet.containsKey(complement)){
                                        wordsBitSet.put(complement,wordsBitSet.get(complement)+1);
                              }else{
                                        wordsBitSet.put(key2BitSet,1);
                              }
                    }
                    write(wordsBitSet,out,i);
                    wordsBitSet.clear();
```

```
		}
}

//Determine the most abundant short sequence as well as the average short sequence frequency.
private static void getStats(File in,File max,File avg){
		HashMap<Integer,Integer> Max=new HashMap<Integer, Integer>();
		HashMap<Integer,Double> Avg=new HashMap<Integer, Double>();
		HashMap<Integer,String> MaxWord=new HashMap<Integer, String>();


		try{
			BufferedReader br=new BufferedReader(new FileReader(in));
			String line="";
			int sum=0;
			int count=0;
			int oldlength=0;
			while((line=br.readLine())!=null){
				String[] split=line.split("\\s+");
				String key=split[0];
				int number=Integer.parseInt(split[1]);

				if(Max.containsKey(key.length())){
					if(Max.get(key.length())<number){
						Max.put(key.length(),number);
						MaxWord.put(key.length(), key);
					}
				}else{
					if(oldlength>0){
						Avg.put(oldlength, (sum*1.0)/count);
					}

					oldlength=key.length();
					sum=0;
					count=0;
					Max.put(key.length(),number);
					MaxWord.put(key.length(),key);
				}
				count++;
				sum+=number;
			}
			if(oldlength>0){
				Avg.put(oldlength, (sum*1.0)/count);
			}
		}catch(IOException e){
			System.err.println(e.toString());
		}
		write(printHashWord(Max,MaxWord),max);
		write(printHash(Avg),avg);
}
```

A4.3 Grouping of highly abundant oligonucleotides in SBW25

```
public static void main(String args[]){
		File genome=new File(args[0]);
		File searchStringFile=new File(args[1]);
		String outputFolder=args[2];
		int flanking=Integer.parseInt(args[3]);
		HashMap<String,StringBuilder> genomeFasta=ReadFasta.readFasta(genome);
		//readFile, needs input from SelectOverrepresentedWords.java
		//pull out most abundant+flanking sequence
		//concatenated the sequences separated by |
		//write everything in a file which doesnt match+frequency
		//write everything in a file which does match+frequency
		//iterate process
		int i=0;
		while(searchStringFile.length()>1){
			String word=getWord(searchStringFile);
			System.out.println(word);
			Iterator<Entry<String,StringBuilder>> it=genomeFasta.entrySet().iterator();
			StringBuilder wordSequence=new StringBuilder();
			while(it.hasNext()){
				Entry<String,StringBuilder> e=it.next();
				wordSequence.append(getSequences(word,e,flanking));
			}
			searchStringFile=writeFiles(searchStringFile,wordSequence.toString(),outputFolder,i);
			i++;
		}
}

private static File writeFiles(File searchStringFile,String wordSequence,String path,int group){
		File newSearchStringFile=new File(path+"/"+"GroupNotFound"+group+".out");
		newSearchStringFile.deleteOnExit();
		try{
			File found=new File(path+"/"+"Group"+group+".out");
			BufferedReader br=new BufferedReader(new FileReader(searchStringFile));
			BufferedWriter bwNotFound=new BufferedWriter(new FileWriter(newSearchStringFile));
			HashMap<String,Integer> foundHash=new HashMap<String, Integer>();
			String line="";
			while((line=br.readLine())!=null){
				String[] split=line.split("\\s+");
				String word=split[0];
				String revWord=DNAmanipulations.reverse(word);
```

```
                                int freq=Integer.parseInt(split[1]);
                                if(wordSequence.contains(word) || wordSequence.contains(revWord) ){
                                        if(foundHash.containsKey(revWord)){
                                                foundHash.put(revWord, foundHash.get(revWord)+freq);
                                        }else{
                                                foundHash.put(word, freq);
                                        }
                                }else{
                                        bwNotFound.write(line+"\n");
                                }
                        }
                        bwNotFound.close();
                        Histogram.write(foundHash,found);
                }catch(IOException e){
                        System.err.println(e.toString());
                }

                return newSearchStringFile;
}

private static String getWord(File in){
                String word="";
                try{
                        BufferedReader br=new BufferedReader(new FileReader(in));
                        String line="";
                        int max=0;
                        while((line=br.readLine())!=null){
                                String[] split=line.split("\\s+");
                                int freq=Integer.parseInt(split[1]);
                                if(max<freq) {
                                        word=split[0];
                                        max=freq;
                                }
                        }
                }catch(IOException e){
                        System.err.println(e.toString());
                }
                return word;
}


private static String getSequences(String word,Entry<String,StringBuilder> e,int flanking){
                StringBuilder sb=buildString(word,e,flanking,true);
                sb.append(buildString(word,e,flanking,false));
                return sb.toString();

}

private static StringBuilder buildString(String word,Entry<String,StringBuilder> e,int flanking,boolean reverse){
                int i=0;
                StringBuilder result=new StringBuilder();
                word=word.toUpperCase();
                String
sequence=reverse?DNAmanipulations.reverse(e.getValue().toString().toUpperCase()):e.getValue().toString().toUpperCase()
;
                while((i=sequence.indexOf(word, i))!=-1){
                        String value=sequence.length()>i+word.length()+flanking && i-flanking>0?sequence.substring(i-
flanking,i+word.length()+flanking):sequence.substring(i,i+word.length());
                        result.append("|"+value);
                        i++;
                }
                return result;
}
```

A4.4 Extending REP sequence groups and identifying the frequency of false positives

```
//Count the occurrences of a set of mutated sequences within the extragenic space of a genome as well as the shuffled
extragenic space of a genome.
public class MutatedSequenceOccurrences {

                ArrayList<ArrayList<Integer>> statistics;
                ArrayList<ArrayList<Integer>> exSpace;
                private int maxMut;
                private int number=0;

                public static void main(String args[]){
                        HashMap<String,StringBuilder> rf=ReadFasta.readFasta(new File(args[0]));
                        String genome=rf.values().toArray(new StringBuilder[0])[0].toString();
                        File artemis=new File(args[1]);
                        int maxmutations=Integer.parseInt(args[2]);
                        File summary=new File(args[3]);
                        int maxsimulations=Integer.parseInt(args[4]);
                        File wordFasta=new File(args[5]);

                        try{
                                wordFasta.createNewFile();
                                summary.createNewFile();
                        }catch(IOException e){
```

```
                        System.err.println(e.toString());
                        System.exit(1);
                }
                int printMutations=Integer.parseInt(args[6]);

                ArrayList<BitSet> words=new ArrayList<BitSet>();


                int size=args[7].length();

                for(int i=7;i<args.length;i++){
                        if(args[i].length()!=size){
                                System.err.println("Words have to have the same size!");
                                System.exit(1);
                        }
                        words.add(DNAmanipulations.codeDNA(args[i]));
                }
                size=words.get(0).length();
                System.out.println("Start...");
                GenerateExtragenicSequences ge=new GenerateExtragenicSequences(genome,artemis);//Normal
genome data (SBW25).
                System.out.println("Extragenic sequence generation done.");
                GenerateMutatedSequences gm=new GenerateMutatedSequences(words,maxmutations);//Generate
mutated query sequences
                System.out.println("Original data...");
                ArrayList<String> seqs=ge.getSequences();
                MutatedSequenceOccurrences mso=new MutatedSequenceOccurrences(maxmutations); );//Determine
occurrences of mutated query sequence in original genome (SBW25)
                BitSetIndexHash bsih=new BitSetIndexHash(DNAmanipulations.toBitSet(seqs),size,false);
                mso.makeStatistics(bsih,size,gm);
                mso.write(summary,false);
                mso.writeWordsWithOccurrence(wordFasta,gm,bsih,printMutations,ge.getMap());
                System.out.println("Random data...");
                SimulateExtragenicSequences ses=new SimulateExtragenicSequences(seqs);//Randomize
extragenic space
                mso=new MutatedSequenceOccurrences(maxmutations);
                for(int i=0;i<maxsimulations;i++){
                        ArrayList<String> randomSeqs=ses.simulate();
                        bsih=new BitSetIndexHash(DNAmanipulations.toBitSet(randomSeqs),size,false);
                        mso.makeStatistics(bsih,size,gm);
                }
                mso.write(summary,true);
        }

        public void writeWordsWithOccurrence(File out,GenerateMutatedSequences gm,BitSetIndexHash bsih,int
mut,HashMap<Integer,Integer> map){

                try{
                        BufferedWriter bw=new BufferedWriter(new FileWriter(out));

                        ArrayList<BitSet> words=gm.getList(mut);

                        for(int i=0;i<words.size();i++){
                                ArrayList<SequencePositions> sq;
                                if((sq=bsih.getPos(words.get(i)))!=null){
                                        bw.write(">"+DNAmanipulations.decodeDNA(words.get(i))+sq.size()+"
Positions:");
                                        for(int j=0;j<sq.size();j++)

        bw.write(map.get(sq.get(j).sequence)+sq.get(j).position+";");
                                        bw.write("\n"+DNAmanipulations.decodeDNA(words.get(i))+"\n");
                                }
                        }

                        bw.close();
                }catch(IOException e){
                        System.err.println(e.toString());

                }

        }

        public MutatedSequenceOccurrences(int maxmutations){
                maxMut=maxmutations;
                statistics=new ArrayList<ArrayList<Integer>>();
                exSpace=new ArrayList<ArrayList<Integer>>();
                for(int i=0;i<=maxMut;i++){
                        statistics.add(new ArrayList<Integer>());
                        statistics.get(i).add(0);
                        exSpace.add(new ArrayList<Integer>());
                        exSpace.get(i).add(0);

                }
        }

        public int getOccurrences(int mut){
                return statistics.get(mut).get(0);
        }

        public int getOccupiedExSpaces(int mut){
                return exSpace.get(mut).get(0);
        }

        public void makeStatistics(BitSetIndexHash bsih,int size,GenerateMutatedSequences gm){
                number++;
                ArrayList<BitSet> newWords=gm.getList(0);
```

```
                    System.out.println("Trial "+number);
                    for(int i=0;i<=maxMut;i++){
                              System.out.println("\t"+i+" mutations. Number of sequences: "+newWords.size());
                              ArrayList<Integer> stats=checkOverLap(bsih.getPos(newWords),size);
                              //int occurrences=bsih.getNumber(newWords);
                              int occurrences=stats.get(0);
                              statistics.get(i).add(occurrences);
                              statistics.get(i).set(0,statistics.get(i).get(0)+occurrences);
                              int spaces=stats.get(1);
                              exSpace.get(i).add(spaces);
                              exSpace.get(i).set(0,exSpace.get(i).get(0)+spaces);
                              if(i<maxMut){
                                        newWords=gm.getList(i+1);
                              }
                    }
          }

          private static ArrayList<Integer> checkOverLap(ArrayList<SequencePositions> pos,int size){
                    HashMap<Integer,ArrayList<Integer>> hm=new HashMap<Integer,ArrayList<Integer>>();
                    ArrayList<Integer> stats=new ArrayList<Integer>();
                    HashMap<Integer,Boolean> seqHM=new HashMap<Integer,Boolean>();
                    int sum=0;
                    for(int i=0;i<pos.size();i++){
                              int seq=pos.get(i).sequence;
                              seqHM.put(seq,true);
                              int posi=pos.get(i).position;
                              if(hm.containsKey(seq)){
                                        hm.get(seq).add( posi);
                              }else{
                                        ArrayList<Integer> al=new ArrayList<Integer>();
                                        al.add(posi);
                                        hm.put(seq, al);
                              }
                    }
                    Iterator<Entry<Integer,ArrayList<Integer>>> it=hm.entrySet().iterator();
                    while(it.hasNext()){
                              Entry<Integer,ArrayList<Integer>> e=it.next();
                              TreeMap<Integer,Boolean> tm=new TreeMap<Integer,Boolean>();
                              for(int i=0;i<e.getValue().size();i++){
                                        tm.put(e.getValue().get(i),true);

                              }
                              sum+=checkOverLap(tm,size);
                    }
                    stats.add(sum);
                    stats.add(seqHM.size());
                    return stats;

          }

          private static int checkOverLap(TreeMap<Integer,Boolean> tm,int size){
                    Integer[] pos=tm.keySet().toArray(new Integer[0]);
                    int number=0;

                    for(int i=0;i<pos.length-1;i++){

                              if(pos[i]+(size/2)<pos[i+1]){
                                        number++;
                              }
                    }
                    number++;
                    return number;
          }

          public ArrayList<Integer> getStatistics(int mutation){
                    return statistics.get(mutation);
          }

          public void write(File Summary,boolean append){
                    try{
                              BufferedWriter bw=new BufferedWriter(new FileWriter(Summary,append));
                              bw.write("\tSummary\t");
                              for(int j=1;j<statistics.get(0).size();j++){
                                        bw.write("Trial "+j+"\t");
                              }
                              bw.write("\n");
                              for(int i=0;i<statistics.size();i++){
                                        bw.write(i+" mutations:\t");
                                        bw.write((int)((statistics.get(i).get(0)*1.0)/number)+"\t");
                                        for(int j=1;j<statistics.get(i).size();j++){
                                                  bw.write(statistics.get(i).get(j)+"\t");
                                        }
                                        bw.write("\n");
                              }
                              bw.close();

                    }catch(IOException e){
                              System.err.println(e.toString());
                              System.exit(1);
                    }
          }
}

//Randomize extragenic space
public SimulateExtragenicSequences(ArrayList<String> original){
          calcParameters(original);
}

private void calcParameters(ArrayList<String> original){
```

```
          ATGCProbs=new ArrayList<ArrayList<EntryExpanded<Double,Character>>>();
          lengths=new ArrayList<Integer>();
          for(int i=0;i<original.size();i++){
                    String seq=original.get(i);
                    lengths.add(seq.length());
                    ATGCProbs.add(convertToProbabilitySet(getATGCContent(seq)));
          }
}

public ArrayList<String> simulate(){
          ArrayList<String> simulation=new ArrayList<String>();
          for(int j=0;j<lengths.size();j++){
                    int seqLength=lengths.get(j);
                    simulation.add(shuffle(ATGCProbs.get(j),seqLength));
          }
          return simulation;
}

private HashMap<Character,Double> getATGCContent(String sequence){
          HashMap<Character,Double> ATGC=new HashMap<Character, Double>();
          sequence=sequence.toUpperCase();
          double part=1.0/sequence.length();
          for(int i=0;i<sequence.length();i++){
                    Character c=sequence.charAt(i);
                    if(!ATGC.containsKey(c)){
                              ATGC.put(c, part);
                    }else{
                              ATGC.put(c, ATGC.get(c)+part);

                    }
          }
          return ATGC;
}

private ArrayList<EntryExpanded<Double,Character>> convertToProbabilitySet(HashMap<Character,Double> ATGC){
          Iterator<Entry<Character,Double>> it=ATGC.entrySet().iterator();
          ArrayList<EntryExpanded<Double, Character>> BaseProbabilities=new
ArrayList<EntryExpanded<Double,Character>>();
          double sum=0;
          while(it.hasNext()){
                    Entry<Character,Double> e=it.next();
                    sum+=e.getValue();
                    EntryExpanded<Double, Character> ee=new EntryExpanded<Double, Character>(sum,e.getKey());
                    BaseProbabilities.add(ee);
          }
          return BaseProbabilities;
}

private String shuffle(ArrayList<EntryExpanded<Double,Character>> ATGC,int length){
          StringBuilder sequence=new StringBuilder();
          for(int i=0;i<length;i++){
                    double rand=Math.random();
                    int j=0;
                    while(rand>ATGC.get(j).getKey() && j<ATGC.size()){
                              j++;
                    }
                    sequence.append(ATGC.get(j).getValue());

          }
          return sequence.toString();
}
```

A4.5 Distribution simulation

```
//Random distribution simulation

public class RandomDistributionSimulation {
          BitSet genomeInter;
          BitSet genomeIntra;
          ArrayList<Integer> posList;

          public RandomDistributionSimulation(BitSet sequence,int numberOfElements,int length,double percentIntra){

                    genomeInter=(BitSet)sequence.clone();
                    genomeIntra=(BitSet)sequence.clone();
                    posList=setElementsRandomly(numberOfElements,length,percentIntra);
          }

          public ArrayList<Integer> getPosList(){
                    return posList;
          }

          private ArrayList<Integer> setElementsRandomly(int nOE,int length,double percentIntra){
                    ArrayList<Integer> posList=new ArrayList<Integer>();
                    int i=nOE;
                    Random r=new Random();
                    while(i>0){

                              int genomePos=r.nextInt(genomeInter.size()-length);
                              boolean intra=r.nextDouble()<=percentIntra;
                              if(isFree(genomePos,length,intra)){
                                        posList.add(genomePos);
                                        setOccupied(genomePos,length,intra);
```

```
                                        i--;
                                }
                        }
                        return posList;
                }


                private boolean isFree(int pos,int length,boolean intra){
                        for(int i=pos;i<pos+length;i++){
                                if(intra?!genomeIntra.get(i):genomeInter.get(i)){
                                        return false;
                                }
                        }
                        return true;
                }

                private void setOccupied(int pos,int length,boolean intra){
                        for(int i=pos;i<pos+length;i++){

                                if(intra)genomeIntra.set(i,false);
                                else genomeInter.set(i,true);

                        }

                }

                public static ArrayList<Integer> createFreeSpaces(boolean genome[]){
                        ArrayList<Integer> free=new ArrayList<Integer>();
                        for(int i=0;i<genome.length;i++){
                                if(!genome[i]){
                                        free.add(i);
                                }
                        }
                        return free;
                }


}

//Actual REP sequence distribution in SBW25

public class DistributionMutatedSequences {
        public static void main(String args[]){
                File genome=new File(args[0]);
                String fasta=ReadFasta.readFasta(genome).values().toArray(new StringBuilder[0])[0].toString();
                File artemis=new File(args[1]);
                ArrayList<BitSet> words=new ArrayList<BitSet>();
                int mutations=Integer.parseInt(args[2]);
                File out=new File (args[3]);
                File enclosed=new File(args[4]);//directory in which the enclosed sequences are put out
                File cluster=new File(args[5]);
                File clusterSeqs=new File(args[6]);
                File doublets=new File(args[7]);
                int maxDist=Integer.parseInt(args[8]);
                int singletonSize=Integer.parseInt(args[9]);
                int notMut1=Integer.parseInt(args[10]);
                int notMut2=Integer.parseInt(args[11]);
                String centralMotif=args[12].toUpperCase();
                for(int i=13;i<args.length;i++){
                        words.add(DNAmanipulations.codeDNA(args[i]));
                }
                int size=words.get(0).length();
                //for extragenic space only
                GenerateExtragenicSequences ge=new GenerateExtragenicSequences(fasta,artemis);
                BitSetIndexHash bsih=new
BitSetIndexHash(DNAmanipulations.toBitSet(ge.getSequences()),size,ge.getMap(),false);
                ArrayList<String> fas=new ArrayList<String>();
                fas.add(fasta);
                HashMap<Integer,Integer> fakePosMap=new HashMap<Integer, Integer>();
                fakePosMap.put(0, 0);
                GenerateMutatedSequences gm=new GenerateMutatedSequences(words,mutations,notMut1,notMut2);
                //Determine sequence positions of query sequences
                ArrayList<Integer> seqPositions=subtractOverLaps(bsih.getPosMap(gm.getList(mutations)),size/2);
                //Determine distance between query sequences
                Integer[] distance=DistanceAndSorting.calcDistance(seqPositions.toArray(new Integer[0]));
                //Calculate histogram of distances
                Histogram<Integer> h=new Histogram<Integer>(distance);
                h.write(out,"std");
                //Determine frequency of different cluster sizes
                HashMap<Integer,String>
clusterDescription=writeCluster(seqPositions,cluster,maxDist,fasta,clusterSeqs,doublets,centralMotif);

        writeEnclosedSequences(h.getHistogram(),seqPositions,fasta,enclosed,args[6].length(),maxDist,clusterDescript
ion);
                writeSingletons(seqPositions,singletonSize,enclosed,400,fasta,centralMotif,args[13].length());

        writeSingletonsFromPairs(seqPositions,singletonSize,enclosed,400,fasta,centralMotif,args[13].length(),18,22)
;

        writeSingletonsFromPairs(seqPositions,singletonSize,enclosed,400,fasta,centralMotif,args[13].length());
        }

        public static void writeSingletonsFromPairs(ArrayList<Integer> seqpos,int size,File out,int dist,String
genome,String centMot,int wordSize){
                try{
                        BufferedWriter bwPair=new BufferedWriter(new
FileWriter(out+"/../singletonsFromPairs"+size+"bp.fas"));
```

```
                        BufferedWriter bwCluster=new BufferedWriter(new
FileWriter(out+"/../singletonsFromCluster"+size+"bp.fas"));
                        BufferedWriter bwSingle=new BufferedWriter(new
FileWriter(out+"/../singletons"+size+"bp.fas"));
                        BufferedWriter bwAll=new BufferedWriter(new FileWriter(new
File(out+"/../singletonsAll"+size+"bp.fas")));
                        ArrayList<String> clusterSeqs=new ArrayList<String>();
                        ArrayList<Integer> clusterPos=new ArrayList<Integer>();
                        for(int i=1;i<seqpos.size();i++){
                                int pos=seqpos.get(i);
                                String seq=genome.substring(pos-size,pos+wordSize+size);
                                if (seq.toUpperCase().contains(centMot))seq=DNAmanipulations.reverse(seq);
                                bwAll.write(">"+pos+"\n"+seq+"\n");
                                if(i>1){
                                        if(seqpos.get(i)-seqpos.get(i-1)<=dist){
                                                clusterPos.add(pos);

                                                clusterSeqs.add(seq);
                                        }else{
                                                if(clusterPos.size()==2){
                                                        write(clusterPos,clusterSeqs,bwPair);
                                                }else if (clusterPos.size()>2){
                                                        write(clusterPos,clusterSeqs,bwCluster);
                                                }else{
                                                        write(clusterPos,clusterSeqs,bwSingle);
                                                }
                                                clusterPos=new ArrayList<Integer>();
                                                clusterSeqs=new ArrayList<String>();
                                                clusterPos.add(pos);
                                                clusterSeqs.add(seq);
                                        }
                                }else{
                                        clusterPos.add(pos);

                                        clusterSeqs.add(seq);
                                }
                        }
                        bwPair.close();
                        bwCluster.close();
                        bwSingle.close();
                        bwAll.close();
                }catch(IOException e){
                        e.printStackTrace();
                }
        }
        public static void writeSingletonsFromPairs(ArrayList<Integer> seqpos,int size,File out,int dist,String
genome,String centMot,int wordSize,int minSize,int maxSize){
                try{
                        BufferedWriter bw=new BufferedWriter(new
FileWriter(out+"/../singletonsFrom_"+minSize+"_to_"+maxSize+"_"+size+"bp.fas"));
                        ArrayList<String> clusterSeqs=new ArrayList<String>();
                        ArrayList<Integer> clusterPos=new ArrayList<Integer>();
                        for(int i=1;i<seqpos.size();i++){
                                int pos=seqpos.get(i);
                                String seq=genome.substring(pos-size,pos+wordSize+size);
                                if (seq.toUpperCase().contains(centMot))seq=DNAmanipulations.reverse(seq);
                                if(i>1){
                                        if(seqpos.get(i)-seqpos.get(i-1)<=maxSize&&seqpos.get(i)-
seqpos.get(i-1)>=minSize){
                                                clusterPos.add(pos);

                                                clusterSeqs.add(seq);
                                        }else{
                                                if(clusterPos.size()==2){
                                                        write(clusterPos,clusterSeqs,bw);
                                                }
                                                clusterPos=new ArrayList<Integer>();
                                                clusterSeqs=new ArrayList<String>();
                                                clusterPos.add(pos);
                                                clusterSeqs.add(seq);
                                        }
                                }else{
                                        clusterPos.add(pos);

                                        clusterSeqs.add(seq);
                                }
                        }
                        bw.close();
                }catch(IOException e){
                        e.printStackTrace();
                }
        }
        public static void write(ArrayList<Integer> pos,ArrayList<String> seqs,BufferedWriter bw){
                try{
                        for(int i=0;i<seqs.size();i++){
                                bw.write(">"+pos.get(i)+"\n"+seqs.get(i)+"\n");
                        }
                }catch(IOException e){
                        e.printStackTrace();
                }
        }

        public static void writeSingletons(ArrayList<Integer> seqpos,int size,File out,int dist,String genome,String
centMot,int wordSize){
                try{
                        BufferedWriter bwFlank=new BufferedWriter(new
FileWriter(out+"/../singletonsFlank"+size+"bp.fas"));
                        for(int i=1;i<seqpos.size()-1;i++){
                                int pos=seqpos.get(i);
```

```
                                        int prev=seqpos.get(i-1);
                                        int after=seqpos.get(i+1);
                                        String seqFlank5=genome.substring(pos-12-size,pos-size);
                                        String seqFlank3=genome.substring(pos+wordSize+size,pos+wordSize+size+12);

        bwFlank.write(">flank5prime_"+i+"_"+pos+"\n"+seqFlank5+"\n>flank3prime"+i+"_"+pos+"\n"+seqFlank3+"\n");
                                        if(pos-prev>dist&&after-pos>dist){
                                                String seq=genome.substring(pos-size,pos+wordSize+size);
                                                if (seq.toUpperCase().contains(centMot)){
                                                        seq=DNAmanipulations.reverse(seq);

                                                }
                                        }
                                }
                                bwFlank.close();
                        }catch(IOException e){
                                e.printStackTrace();
                        }
        }

        //returns a list of cluster sizes, input for the class Histogram
        public  static Integer[] getCluster(Integer[] pos,int maxDist){
                ArrayList<Integer> clusters=new ArrayList<Integer>();
                        int c=0;
                        for(int i=0;i<pos.length;i++){
                                int dist=i==pos.length-1?maxDist+1:pos[i+1]-pos[i];

                                c++;
                                if(dist>maxDist){


                                        clusters.add(c);
                                        c=0;
                                }


                        }

                return clusters.toArray(new Integer[0]);

        }

        //writes cluster histogram (how many time a cluster of a certain size is observed)
        //also returns a hash map that contains for each position a String like the following: c[cluster
number]_[cluster size] if the cluster is larger than 2
        public  static HashMap<Integer,String> writeCluster(ArrayList<Integer> pos,File cluster,int maxDist,String
genome, File sequences, File doublets,String centMot){
                HashMap<Integer,String> clusterDescription=new HashMap<Integer, String>();
                try{
                        BufferedWriter bw=new BufferedWriter(new FileWriter(cluster));
                        BufferedWriter bwseqs=new BufferedWriter(new FileWriter(sequences));
                        BufferedWriter doubBw=new BufferedWriter(new FileWriter(doublets));
                        File artemisdoublets=new File(doublets+".tab");
                        ArrayList<Info> artDoublets=new ArrayList<Info>();
                        int c=0;
                        HashMap<Integer,Integer> clusterHash=new HashMap<Integer, Integer>();
                        ArrayList<Integer> clusterPositions=new ArrayList<Integer>();
                        int clusterNumber=0;
                        int direct=0;
                        int inverted=0;
                        int others=0;
                        int counter=0;
                        String centMotrev=DNAmanipulations.reverse(centMot).toUpperCase();
                        System.out.println(pos.size());
                        for(int i=0;i<pos.size();i++){
                                if(pos.get(i)>4253700&&pos.get(i)<4253800){
                                        System.out.println(c+"HHHHHHHHHHHHHHHHHHHHH");
                                }
                                int dist=i==pos.size()-1?maxDist+1:pos.get(i+1)-pos.get(i);
                                clusterPositions.add(pos.get(i));
                                c++;
                                if(dist>maxDist){
                                        if(c>3){
                                                for(int j=0;j<clusterPositions.size();j++){
                                                        String
seq1=genome.substring(clusterPositions.get(j),clusterPositions.get(j)+16).toUpperCase();
                                                        int distance=j>0?clusterPositions.get(j)-
clusterPositions.get(j-1):clusterPositions.get(j);
                                                        System.out.println(seq1+" "+distance);
                                                }
                                                System.out.println("_____________");
                                        }
                                        if(c==2){
                                                String
seq1=genome.substring(clusterPositions.get(0),clusterPositions.get(0)+16).toUpperCase();
                                                String
seq2=genome.substring(clusterPositions.get(1),clusterPositions.get(1)+16).toUpperCase();

                                                if
((seq1.contains(centMot)&&seq2.contains(centMot))||(seq1.contains(centMotrev)&&seq2.contains(centMotrev))){
                                                        direct++;
                                                }else
if((seq1.contains(centMot)&&seq2.contains(centMotrev))||(seq1.contains(centMotrev)&&seq2.contains(centMot))){
                                                        inverted++;
                                                }else{
                                                        others++;
                                                }
                                                int
add=(seq1.contains(centMotrev)&&seq2.contains(centMot))?8:0;
```

```
                    doubBw.write(">doublet_"+clusterPositions.get(0)+"\n"+genome.substring(clusterPositions.get(0)-
add,clusterPositions.get(1)+16+add)+"\n");
                                                                artDoublets.add(new Info(clusterPositions.get(0)-
add+1,clusterPositions.get(1)+16+add,"REPIN"));
                                                      }
                                                      if(c>3){
                                                                counter++;
                                                                for(int j=0;j<clusterPositions.size()-1;j++){
                                                                          int start=clusterPositions.get(j);
                                                                          int end=clusterPositions.get(j+1)+16;

                    bwseqs.write(">c"+c+"."+counter+"_"+start+"_"+end+"\n"+genome.substring(start,end)+"\n");

                                                                }

                                                      }
                                                      if(c>2){
                                                                clusterNumber++;
                                                                for(int j=0;j<clusterPositions.size();j++){
                                                                          clusterDescription.put(clusterPositions.get(j),
"c"+clusterNumber+"_"+c);

                                                                          //System.out.println(clusterPositions.get(j)+"
"+genome.substring(clusterPositions.get(j),clusterPositions.get(j)+16)+"\t"+c);
                                                                }
                                                      }

                                                      if(clusterHash.containsKey(c)){
                                                                clusterHash.put(c, clusterHash.get(c)+1);
                                                      }else{
                                                                clusterHash.put(c,1);
                                                      }
                                                      c=0;
                                                      clusterPositions=new ArrayList<Integer>();
                                                      WriteArtemis.write(artDoublets, artemisdoublets);
                                            }


                                  }

                                  System.out.println("Direct: "+direct+"\nInverted: "+inverted+"\nOthers: "+others);
                                  System.out.println("Clusters larger than three:"+counter);
                                  bw.write(Histogram.write(clusterHash, 1));
                                  bw.close();
                                  bwseqs.close();
                                  doubBw.close();
                        }catch(IOException e){
                                  System.err.println(e.toString());
                        }
                        return clusterDescription;

          }

          public static ArrayList<Integer> subtractOverLaps(ArrayList<Integer> pos,int size){
                    SortPositions sa=new SortPositions(pos);
                    ArrayList<Integer> temp=sa.getList();
                    ArrayList<Integer> positions=new ArrayList<Integer>();
                    for(int i=0;i<temp.size()-1;i++){

                                  if(temp.get(i)+size<temp.get(i+1)){
                                            positions.add(temp.get(i));
                                  }
                    }
                    if(temp.size()>0)positions.add(temp.get(temp.size()-1));
                    return positions;
          }

}
```

A4.6 Singlet decay

```
public class PairwiseAlignmentWithoutReplacement {
          ArrayList<Integer> list=new ArrayList<Integer>();
          public static void main(String args[]){
                    PairwiseAlignmentWithoutReplacement pwawr=new PairwiseAlignmentWithoutReplacement();
                    File folder=new File(args[0]);
                    int repetitions=Integer.parseInt(args[1]);
                    File[] files=folder.listFiles();
                    try{
                                  BufferedWriter bwResults=new BufferedWriter(new FileWriter(new
File(folder+"/resultsWithoutReplacement.txt")));
                                  bwResults.write("Average\tStdev\tStderr\tmax\tmin\n");
                                  for(int k=0;k<files.length;k++){
                                            if(!files[k].getAbsolutePath().endsWith("fas"))continue;
                                            System.out.println(files[k]);
                                            ArrayList<Fasta> fas1=Fasta.readFasta(files[k]);
                                            ArrayList<Double> pw=new ArrayList<Double>();
                                            double min=Integer.MAX_VALUE;
                                            double max=Integer.MIN_VALUE;

                                            BufferedWriter bw=new BufferedWriter(new FileWriter(files[k]+".out"));
```

```
                                        for(int i=0;i<repetitions;i++){
                                                  double sum=0;
                                                  int j=0;
                                                  pwawr.fillList(fas1.size());
                                                  while(pwawr.getSize()>1){
                                                            j++;
                                                            int rand1=(int)(Math.random()*pwawr.getSize());

                                                            Fasta seq1=fas1.get(pwawr.getItem(rand1));
                                                            int rand2=(int)(Math.random()*pwawr.getSize());
                                                            Fasta seq2=fas1.get(pwawr.getItem(rand2));
                                                            double
pwIdent=NeedlemanWunsch.getPairwiseIdentity(seq1.getSequence().toUpperCase(), seq2.getSequence().toUpperCase());

                                                                      String
rev=DNAmanipulations.reverse(seq1.getSequence());
                                                            double
          pwIdent2=NeedlemanWunsch.getPairwiseIdentity(rev.toUpperCase(), seq2.getSequence().toUpperCase());

                                                            sum+=Math.max(pwIdent,pwIdent2);
                                                  }

                                                  double avg=sum/j;
                                                  bw.write(avg+"\r\n");
                                                  pw.add(avg);
                                                  if(min>avg)min=avg;
                                                  if(max<avg)max=avg;
                                        }
                                        bw.close();

                                        Stats stats=new Stats(pw);
                                        bwResults.write(files[k].getName()+"\t"+stats.getAverage());
                                        bwResults.write("\t"+stats.getStandardDeviation());
                                        bwResults.write("\t"+stats.getStandardError());
                                        bwResults.write("\t"+min);
                                        bwResults.write("\t"+max+"\n");


                              }
                              bwResults.close();
                    }catch(IOException e){
                              e.printStackTrace();
                              System.exit(-1);
                    }
          }
          private int getItem(int index){
                    int item=list.get(index);
                    list.remove(index);
                    return item;
          }
          private void fillList(int number){
                    list=new ArrayList<Integer>();
                    for(int i=0;i<number;i++){
                              list.add(i);
                    }
          }
          private int getSize(){
                    return list.size();
          }

}
```

A4.7 Testing for excision of REP singlets

```
//given a fasta file and a solexa sequence file, population sequencing reads that contain both flanking sequences at a
distance of less than what is expected are returned
//reads that were returned were manually tested for excision to verify the event
public class FindExcissions {
          public static void main(String args[]){
                    File in=new File(args[0]);
                    ArrayList<Fasta> words=Fasta.readFasta(new File(args[1]));
                    File out=new File(args[2]);
                    int distance=Integer.parseInt(args[3]);

                    Fasta.write(getSequencesFastQ(in, words,distance),out);
          }

          public static ArrayList<Fasta> getSequencesOneLineFasta(File in,ArrayList<Fasta> words){
                    ArrayList<Fasta> seqs=new ArrayList<Fasta>();
                    try{
                              BufferedReader br=new BufferedReader(new FileReader(in));
                              String line="";
                              int i=0;
                              String ident="";
                              while((line=br.readLine())!=null){
                                        if(i%100000==0)System.out.println(i+" lines read and checked.");
                                        if(line.startsWith(">")){
                                                  ident=line.substring(1);
                                        }else {
                                                  line=line.toUpperCase();
                                                  for(int j=0;j<words.size();j+=2){
                                                            String word1=words.get(j).getSequence();
```

```
                                                                String word2=words.get(j+1).getSequence();
                                                                String reverse1=DNAmanipulations.reverse(word1);
                                                                String reverse2=DNAmanipulations.reverse(word2);
                                                                if((line.contains(word1.toUpperCase())
&&line.contains(word2.toUpperCase()))||(line.contains(reverse1.toUpperCase())&&line.contains(reverse2.toUpperCase())))
{
                                                                        System.out.println("Found one:");
                                                                        System.out.println(">"+ident+"\n"+line);
                                                                        System.out.println("Found with:");

          System.out.println(words.get(j)+"\n"+words.get(j+1));
                                                                        seqs.add(new Fasta(ident,line));
                                                                        break;
                                                                }
                                                        }
                                                }
                                                i++;
                                        }

                                        br.close();
                                }catch(IOException e){
                                        e.printStackTrace();
                                }

                                return seqs;
          }
          public static ArrayList<Fasta> getSequencesFastQ(File in,ArrayList<Fasta> words,int dist){
                                ArrayList<Fasta> seqs=new ArrayList<Fasta>();
                                try{
                                        BufferedReader br=new BufferedReader(new FileReader(in));
                                        String line="";
                                        int i=0;
                                        String ident="";
                                        while((line=br.readLine())!=null){

                                                if(i%4==0){
                                                        ident=line.substring(1);
                                                }else if(i%4==1) {
                                                        line=line.toUpperCase();
                                                        for(int j=0;j<words.size();j+=2){
                                                                String word1=words.get(j).getSequence();
                                                                String word2=words.get(j+1).getSequence();

                                                                if(contains(line,word1,word2,dist)){
                                                                        System.out.println("Found one:");
                                                                        System.out.println(">"+ident+"\n"+line);
                                                                        System.out.println("Found with:");

          System.out.println(words.get(j)+"\n"+words.get(j+1));
                                                                        seqs.add(new Fasta(ident,line));
                                                                        break;
                                                                }
                                                        }
                                                }
                                                i++;
                                        }

                                        br.close();
                                }catch(IOException e){
                                        e.printStackTrace();
                                }

                                return seqs;
          }
          public static boolean contains(String line,String word1,String word2,int dist){
                                String reverse1=DNAmanipulations.reverse(word1);
                                String reverse2=DNAmanipulations.reverse(word2);
                                if (dist ==-1){
                                        return (line.contains(word1.toUpperCase())
&&line.contains(word2.toUpperCase()))||(line.contains(reverse1.toUpperCase())&&line.contains(reverse2.toUpperCase()));
                                }else{
                                        if(line.contains(word1.toUpperCase()) &&line.contains(word2.toUpperCase())){
                                                return dist>=Math.abs(line.indexOf(word1.toUpperCase())-
line.indexOf(word2.toUpperCase()));
                                        }else if(line.contains(reverse1.toUpperCase())&&line.contains(reverse2.toUpperCase())){
                                                return dist>=Math.abs(line.indexOf(reverse1.toUpperCase())-
line.indexOf(reverse2.toUpperCase()));
                                        }else return false;
                                }

          }
}
```

A4.8 REP sequence selection in other genomes

```
//this program computes the p-Value and word occurrence of all sequences in a fasta file that are of a specified
length (atm 16bp)
//it uses the program GetFrequencyBelowPvalue.java for calculating the pValue

//as input are required: folder for word frequency files
//a fasta file where the first word in the description has to be the same as the filename before the extension in the
wordFrequency folder
```

```
//words that are longer than 20bp are chopped into 16bp long words and are subsequently analysed

public class PValueWords {
          public static void main(String args[]){
                    File fasta=new File(args[0]);
                    File wordFreqs=new File(args[1]);
                    int wordlength=Integer.parseInt(args[2]);
                    File out=new File(args[3]);
                    //HashMap<BitSet,Integer> Occurrences=null;
                    ArrayList<String> ids=new ArrayList<String>();
                    ArrayList<String> words=new ArrayList<String>();
                    readFasta(fasta,ids,words,wordlength);
                    GetFrequencyBelowPvalue gp = null;
                    File frequencyE=new File("");
                    try{
                              BufferedWriter bw=new BufferedWriter(new FileWriter(out));
                              for(int i=0;i<ids.size();i++){
                                        File frequency=new File(wordFreqs+"/"+ids.get(i).split("\\s+")[0]+".out");

                                        if(gp==null || !frequency.equals(frequencyE)){
                                                  //Occurrences=null;
                                                  gp=new GetFrequencyBelowPvalue(frequency,wordlength);
                                                  //Occurrences=readMap(frequency,wordlength);
                                        }
                                        int occ=gp.getFreq(words.get(i));
                                        double pValue=gp.getpValue(occ, words.get(i).length());
                                        bw.write(ids.get(i)+"\t"+words.get(i)+"\t"+occ+"\t"+pValue+"\n");
                                        System.out.println(ids.get(i)+"\t"+words.get(i)+"\t"+occ+"\t"+pValue);
                                        frequencyE=frequency;
                              }
                              bw.close();
                    }catch(IOException e){
                              e.printStackTrace();
                    }

          }

          public static void readFasta(File fas,ArrayList<String> ids,ArrayList<String> words,int wl){
                    try{
                              BufferedReader br = new BufferedReader(new FileReader(fas));
                              String line="";
                              String id="";
                              BufferedWriter bw=new BufferedWriter(new FileWriter(new
File("/home/frederic/auckland/fastaSeqs/palindromesMorethan20bp.fas")));
                              while((line=br.readLine())!=null){
                                        if(line.startsWith(">")){
                                                  id=line.substring(1);
                                        }else {
                                                  line=line.replace("\n","");
                                                  line=line.replace("\r","");
                                                  if(line.length()==wl){
                                                            ids.add(id);
                                                            words.add(line);
                                                  }else{
                                                            bw.write(">"+id+"\n"+line+"\n");
                                                            chopWord(line,id,ids,words,wl);
                                                  }
                                        }
                              }
                              bw.close();
                    }catch(IOException e){
                              e.printStackTrace();
                    }

          }

          public static void chopWord(String word,String id,ArrayList<String> ids,ArrayList<String> words,int wl){
                    for(int i=1;i<=word.length()-wl-1;i++){
                              ids.add(id+"."+i);
                              words.add(word.substring(i-1,i+wl-1));
                              //System.out.println(word.substring(i-1,i+15)+" "+word);
                    }
          }

          public static int getOccurrences(HashMap<BitSet,Integer> Occ,String word){
                    BitSet code=DNAmanipulations.codeDNA(word);
                    BitSet rev=DNAmanipulations.reverse(code);
                    if(Occ.containsKey(code)){
                              return Occ.get(code);
                    }else if(Occ.containsKey(rev)){
                              return Occ.get(rev);
                    }else return 0;
          }
          public static HashMap<BitSet,Integer> readMap(File in,int wl){
                    HashMap<BitSet,Integer> hm=new HashMap<BitSet, Integer>();
                    try{
                              BufferedReader br=new BufferedReader(new FileReader(in));
                              String line="";
                              while((line=br.readLine())!=null){
                                        String[] split=line.split("\\s+");
                                        if(split[0].length()!=wl)continue;
                                        BitSet coded=DNAmanipulations.codeDNA(split[0]);
                                        hm.put(coded, Integer.parseInt(split[1]));
                              }

                    }catch(IOException e){
                              e.printStackTrace();
                    }
                    return hm;
```

```
	}
}
```

A4.9 BLAST search

```
//given a blast database, a folder containing associated gbk, fna and word frequency (.wfr) files, a file with query
protein sequences the program
//and e-Value, program returns the AA and Nuc sequence of genes that overlap with blast hits, it also returns the
flanking sequences and the associated most abundant
//n-mer
//wordFrequency files are created if not present
public class GetRelatedAASeqNucSeq5PAnd3PFlankSeq {
	public static void main(String args[]){
		File InputFolder=new File(args[0]);
		File OutputFolder=new File(args[1]);
		String eValue=args[2];
		File query=new File(args[3]);
		File[] genomes=InputFolder.listFiles();
		ArrayList<Info> info=new ArrayList<Info>();
		for(int i=0;i<genomes.length;i++){
			if(genomes[i].getAbsolutePath().endsWith(".fna")){
				info.addAll(blastQuery(genomes[i],query,OutputFolder,eValue));
			}
		}
		ArrayList<String> eValues=geteValues(OutputFolder);
		writeAANuc(info,InputFolder,OutputFolder,eValues);
	}

	public static ArrayList<String> geteValues(File out){
		ArrayList<String> temp=new ArrayList<String>();
		File[] files=out.listFiles();
		for(int i=0;i<files.length;i++){
			if(files[i].getName().contains("OutputFolder")){
				String split[]=files[i].getName().split("_");
				temp.add(split[1]);
			}
		}
		return temp;
	}

	public static void writeAANuc(ArrayList<Info> intervals,File InFolder,File OutFolder,ArrayList<String>
eValues){
		HashMap<String,Boolean> hm=new HashMap<String,Boolean>();
		//initialize
		for(int i=0;i<eValues.size();i++){
			File AA=new File(OutFolder+"/OutputFolder_"+eValues.get(i)+"/AA.faa");
			File Nuc=new File(OutFolder+"/OutputFolder_"+eValues.get(i)+"/Nuc.fna");
			File FiveFlank=new File(OutFolder+"/OutputFolder_"+eValues.get(i)+"/FiveFlank.fna");
			File ThreeFlank=new File(OutFolder+"/OutputFolder_"+eValues.get(i)+"/ThreeFlank.fna");
			AA.delete();
			Nuc.delete();
			FiveFlank.delete();
			ThreeFlank.delete();
		}
		try{


			int number=0;
			ReadGenbank rgb=null;
			String oldFileID="";
			for(int i=0;i<intervals.size();i++){
				String split[]=intervals.get(i).info.split("\\||\\.");
				String id=split[3];
				double eValue=Double.parseDouble(intervals.get(i).info.split("---")[1]);
				String fileID=InFolder+"/"+id;

				//File wfr=new File(fileID+".wfr");
//				if(!wfr.exists()){
//					PrintWordFrequencyBothStrands.main(new String[]{fileID+".fna",
wl+"", wl+"" ,fileID+".wfr"});
//				}

				if(rgb==null||!oldFileID.equals(fileID))rgb=new ReadGenbank(new
File(fileID+".gbk"));
				oldFileID=fileID;
				//System.out.println(intervals.get(i).getStart()+"
"+intervals.get(i).getEnd());
				ArrayList<Info> genes=new ArrayList<Info>();
				InfoTree it=rgb.getInfoTree("CDS");
				it.search(intervals.get(i), genes);
				if(genes.size()>1)genes=filterGenes(intervals.get(i),genes);
				if(genes.size()==1){
					Info pre=it.predecessor(genes.get(0));
					Info suc=it.successor(genes.get(0));
					String genome=Fasta.readFasta(new
File(fileID+".fna")).get(0).getSequence();
					Info flank5=pre!=null?new
Info(pre.getEnd(),genes.get(0).getStart(),"flank5_"+genes.get(0).info):new
Info(0,genes.get(0).getStart(),"flank5_"+genes.get(0).info);
					Info flank3=suc!=null?new
Info(genes.get(0).getEnd(),suc.getStart(),"flank3_"+genes.get(0).info):new
Info(genes.get(0).getEnd(),genome.length(),"flank3_"+genes.get(0).info);
					int start=genes.get(0).getStart()-1;
```

```
                                                  int end=genes.get(0).getEnd();
                                                  String hmID=id+"_"+start+"_"+end;
                                                  if(!hm.containsKey(hmID)){
                                                            number++;
                                                            hm.put(hmID,true);

                                                            String nucSeq=genome.substring(start,end);
                                                            if(genes.get(0).info.contains("complement")){
                                                                      nucSeq=DNAmanipulations.reverse(nucSeq);
                                                                      int temp=start;
                                                                      start=end;
                                                                      end=temp;
                                                            }
                                                            int f5start=flank5.getStart();
                                                            int f5end=flank5.getEnd();
                                                            int f3start=flank3.getStart();
                                                            int f3end=flank3.getEnd();
                                                            String
f5seq=f5start<f5end?genome.substring(f5start,f5end):"";
                                                            String
f3seq=f3start<f3end?genome.substring(flank3.getStart(),flank3.getEnd()):"";
                                                            String
AASeq=DNAmanipulations.translate(nucSeq,DNAmanipulations.code());

                                                            for(int j=0;j<eValues.size();j++){

                                                                      if(eValue<=Double.parseDouble(eValues.get(j))){
                                                                                File AA=new
File(OutFolder+"/OutputFolder_"+eValues.get(j)+"/AA.faa");
                                                                                File Nuc=new
File(OutFolder+"/OutputFolder_"+eValues.get(j)+"/Nuc.fna");
                                                                                File FiveFlank=new
File(OutFolder+"/OutputFolder_"+eValues.get(j)+"/FiveFlank.fna");
                                                                                File ThreeFlank=new
File(OutFolder+"/OutputFolder_"+eValues.get(j)+"/ThreeFlank.fna");
                                                                                BufferedWriter bwNuc=new
BufferedWriter(new FileWriter(Nuc,true));
                                                                                BufferedWriter bwAA=new
BufferedWriter(new FileWriter(AA,true));
                                                                                BufferedWriter bw5F=new BufferedWriter
(new FileWriter(FiveFlank,true));
                                                                                BufferedWriter bw3F=new
BufferedWriter(new FileWriter(ThreeFlank,true));

          bwNuc.write(">"+number+"_"+id+"_"+start+"_"+end+"_"+genes.get(0).info+"\n"+nucSeq+"\n");

          bwAA.write(">"+number+"_"+id+"_"+start+"_"+end+"_"+genes.get(0).info+"\n"+AASeq+"\n");

          bw5F.write(">"+number+"_"+id+"_"+flank5.getStart()+"_"+flank5.getEnd()+"\n"+f5seq+"\n");

          bw3F.write(">"+number+"_"+id+"_"+flank3.getStart()+"_"+flank3.getEnd()+"\n"+f3seq+"\n");
                                                                                bwNuc.close();
                                                                                bwAA.close();
                                                                                bw5F.close();
                                                                                bw3F.close();
                                                                      }
                                                            }
                                                  }

                                        }else{
                                                  if(genes.size()==0){
                                                            System.err.println(id+" "+intervals.get(i).getStart()+"
"+intervals.get(i).getEnd()+": Zero overlapping genes found...possibly a pseudogene?");
                                                  }else{
                                                            System.err.println(id+" "+intervals.get(i).getStart()+"
"+intervals.get(i).getEnd()+": Too many ("+ genes.size()+") genes found.");
                                                  }
                                        }

                              }

                    }catch(IOException e){
                              e.printStackTrace();
                    }
          }
          public static ArrayList<Info> filterGenes(Info interval,ArrayList<Info> genes){
                    ArrayList<Info> temp=new ArrayList<Info>();
                    int max=0;
                    Info maxInfo=new Info(0,0,"");
                    for(int i=0;i<genes.size();i++){
                              int o=getOverlap(genes.get(i),interval);
                              if(max<o){
                                        max=o;
                                        maxInfo=genes.get(i);
                              }
                    }
                    temp.add(maxInfo);
                    //System.out.println(maxInfo);
                    return temp;
          }

          public static int getOverlap(Info gene,Info interval){
                    int start=0;
                    int end=0;
                    int intervalStart=Math.min(interval.getStart(),interval.getEnd());
                    int intervalEnd=Math.max(interval.getStart(),interval.getEnd());
                    //System.out.println("START: "+gene+"\n"+interval);
                    if(intervalStart>gene.getStart()){
                              start=intervalStart;
```

```
                    }else{
                              start=gene.getStart();
                    }
                    if(intervalEnd<gene.getEnd()){
                              end=intervalEnd;
                    }else{
                              end=gene.getEnd();
                    }

                    return end-start;
          }

          public static ArrayList<Info> blastQuery(File db, File query,File outFolder,String e){
                    File out=new File(outFolder+"/temp.txt");
                    ArrayList<Info> blastIntervals=new ArrayList<Info>();
                    //out.deleteOnExit();
                    PerformBlast.blast("tblastn", Double.parseDouble(e), out, query, db, true,false,true);
                    ReadBlast rb=new ReadBlast(out);
                    for(int i=0;i<rb.getDatabase().size();i++){
                              int start=rb.getStartDB().get(i);
                              int end=rb.getEndDB().get(i);
                              int temp=start;
                              start=start<end?start:end;
                              end=end>start?end:temp;
                              blastIntervals.add(new Info(start,end,rb.getDatabase().get(i)+"---
"+rb.getEvalue().get(i)));
                    }
                    return blastIntervals;
          }
}
```

A4.10 Identifying duplications

```
public static void checkIdentity(HashMap<String,ArrayList<Fasta>> chr,File out,File matrix,double threshold){
                    try{
                              BufferedWriter bw=new BufferedWriter(new FileWriter(out));
                              Iterator<Entry<String, ArrayList<Fasta>>> it=chr.entrySet().iterator();
                              while(it.hasNext()){
                                        Entry<String,ArrayList<Fasta>> e=it.next();
                                        ArrayList<Fasta> genes=e.getValue();
                                        for(int i=0;i<genes.size();i++){
                                                  //System.out.println(genes.get(i).getIdent());
                                                  for(int j=i+1;j<genes.size();j++){
                                                            //System.out.println("     "+genes.get(j).getIdent());

                                                            NeedlemanWunsch nw=new
NeedlemanWunsch(genes.get(i).getSequence(), genes.get(j).getSequence(), NeedlemanWunsch.readSimilarityMatrix(matrix),
6, 1);
                                                            //System.out.println(nw.getAlignments());

                                                            double pw=nw.getPairwiseIdentity();
                                                            //System.out.println(pw);
                                                            if(pw>threshold)bw.write(genes.get(i)+" "+genes.get(j)+"
"+pw+"\n");
                                                  }

                                        }
                              }
                              bw.close();
                    }catch(IOException e){
                              e.printStackTrace();
                    }
          }

          public static HashMap<String,ArrayList<Fasta>> sortChromosomes(ArrayList<Fasta> seqs){
                    HashMap<String,ArrayList<Fasta>> chr=new HashMap<String, ArrayList<Fasta>>();
                    for(int i=0;i<seqs.size();i++){
                              String id=seqs.get(i).getIdent();
                              String[] split=id.split("\\_");
                              String chromo=split[1]+"_"+split[2];
                              if(chr.containsKey(chromo)){
                                        chr.get(chromo).add(seqs.get(i));
                              }else{
                                        ArrayList<Fasta> temp=new ArrayList<Fasta>();
                                        temp.add(seqs.get(i));
                                        chr.put(chromo, temp);
                              }
                    }
                    return chr;
          }
```

A4.11 Taxonomy information

```
public class TaxonomicTree {
          public static void main(String args[]){
                    File taxInfoFolder=new File(args[0]);
                    File taxonomy=new File(args[1]);
                    String rank=args[2];
```

```
                File name=new File(args[3]);
                HashMap<String,String> taxTree=read(taxonomy,1);
                HashMap<String,String> level=read(taxonomy,2);
                HashMap<String,String> namehash=readName(name,1);
                File[] folders=taxInfoFolder.listFiles();
                for(int i=0;i<folders.length;i++){
                        if(folders[i].isDirectory()){
                                File in=new File(folders[i]+"/geneInfo.txt");
                                if(in.exists()){
                                        File out=new File(folders[i]+"/"+rank+".txt");
                                        HashMap<String,String> tax=readTaxInfo(in);
                                        //System.out.println(in);
                                        writeRank(taxTree,level,rank,tax,namehash,out);
                                }
                        }
                }
        }

        public static void writeRank(HashMap<String,String> tree,HashMap<String,String> level,String
rank,HashMap<String,String> taxNCBI,HashMap<String,String> name,File out){
                try{
                        BufferedWriter bw=new BufferedWriter(new FileWriter(out));
                        Iterator<Entry<String,String>> it = taxNCBI.entrySet().iterator();
                        while(it.hasNext()){
                                Entry<String,String> e=it.next();
                                String current=e.getKey();
                                String old="";
                                //System.out.println(name.get(current));
                                //System.out.println(current);
                                while(!level.get(current).equals(rank)&&!current.equals(old)){
                                        old=current;
                                        current=tree.get(current);
                                        if(rank.equals("class")&&level.get(current).equals("phylum")){
                                                break;
                                        }
                                        //System.out.println(current);

                                }

        bw.write(e.getKey()+"\t"+current+"\t"+name.get(current)+"\t"+e.getValue()+"\n");
                                        //System.out.println(name.get(current));
                        }
                        bw.close();
                }catch(IOException e){
                        e.printStackTrace();
                }
        }
        public static HashMap<String,String> readName(File in,int i){
                HashMap<String,String> taxTree=new HashMap<String, String>();
                try{
                        BufferedReader br=new BufferedReader(new FileReader(in));
                        String line="";
                        while((line=br.readLine())!=null){
                                String[] split=line.split("\\|");
                                if(split[3].trim().equals("scientific name"))taxTree.put(split[0].trim(),
split[i].trim());
                        }

                        br.close();
                }catch(IOException e){
                        e.printStackTrace();
                }
                return taxTree;
        }
        public static HashMap<String,String> read(File in,int i){
                HashMap<String,String> taxTree=new HashMap<String, String>();
                try{
                        BufferedReader br=new BufferedReader(new FileReader(in));
                        String line="";
                        while((line=br.readLine())!=null){
                                String[] split=line.split("\\|");
                                if(!taxTree.containsKey(split[0].trim()))taxTree.put(split[0].trim(),
split[i].trim());
                        }

                        br.close();
                }catch(IOException e){
                        e.printStackTrace();
                }
                return taxTree;
        }


        public static HashMap<String,String> readTaxInfo(File tax){
                HashMap<String,String> taxes=new HashMap<String, String>();
                try{
                        BufferedReader br=new BufferedReader(new FileReader(tax));
                        String line="";
                        while((line=br.readLine())!=null){
                                String[] split=line.split("\t");
                                taxes.put(split[1],split[0]);
                        }
                        br.close();
                }catch(IOException e){
                        e.printStackTrace();
                }
                return taxes;
        }
}
```

```
  Frequency determination of flanking 16-mers
public static String getMaxWordAndFreq(String sequence,HashMap<BitSet,Integer> wf,int length){
          int max=0;
          String maxWord="";
          for(int i=0;i<sequence.length()-length;i++){
                    BitSet word=DNAmanipulations.codeDNA(sequence.substring(i,i+length).toUpperCase());
                    if(word==null)continue;
                    int freq=0;
                    String wordTxt=DNAmanipulations.decodeDNA(word);
                    if(wf.containsKey(word)){
                              freq=wf.get(word);
                    }else if(wf.containsKey(DNAmanipulations.reverse(word))){
                              freq=wf.get(DNAmanipulations.reverse(word));

                    }else{
                              System.err.println(wordTxt+" not found in word frequency file!");
                    }
                    if(freq>max){
                              max=freq;
                              maxWord=wordTxt;
                    }
          }
          return maxWord+" "+max;
}
```

A4.12 Calculating the significance of differences for genomic distribution characteristics

```
//n (length of distribution) members of a given distribution are randomly chosen and the mean of these n members is
calculated
//this procedure is repeated <rep> (100,000) times
//returns what proportion (P-value) of the <rep> (100,000) repetitions exceed the average of a distribution that is to
be compared

public static double produceAndCompare(ArrayList<Double> original,int rep,double average){
          int exceed=0;
          for(int i=0;i<rep;i++){
                    double sum=0;
                    for(int j=0;j<original.size();j++){
                              int rand=(int)(Math.random()*original.size());
                              sum+=original.get(rand);
                    }
                    if((sum/original.size())>average){
                              exceed++;
                    }
          }
          return (exceed*1.0)/rep;
}
```

A4.13 Calculating the pairwise identity for amino acid sequences and its significance

```
public class PairwiseAlign {
          public static void main(String args[]){
                    File f1=new File(args[0]);
                    File matrix=new File(args[1]);
                    double GapOpen=Double.parseDouble(args[2]);
                    double GapC=Double.parseDouble(args[3]);
                    ArrayList<Fasta> fas1=Fasta.readFasta(f1);
                    ArrayList<Double> pw=new ArrayList<Double>();
                    HashMap<Character,HashMap<Character,Integer>> subMat=NeedlemanWunsch.readSimilarityMatrix(matrix);
                    for(int i=0;i<fas1.size();i++){
                              for(int j=i+1;j<fas1.size();j++){
                                        String seq1=fas1.get(i).getSequence().toUpperCase();
                                        String seq2=fas1.get(j).getSequence().toUpperCase();
                                        NeedlemanWunsch nw=new NeedlemanWunsch(seq1,seq2,subMat,GapOpen,GapC);
                                        double pw1=nw.getPairwiseIdentity();
                                        nw=new
NeedlemanWunsch(DNAmanipulations.reverse(seq1).toUpperCase(),seq2,subMat,GapOpen,GapC);
                                        double pw2=nw.getPairwiseIdentity();
                                        System.out.println(pw1+" "+pw2);
                                        double identity=Math.max(pw1,pw2);
                                        pw.add(identity);
                                        System.out.println(identity);
                                        System.out.println(nw.getAlignments());
                              }
                    }
                    Stats stats=new Stats(pw);

                    System.out.println("Average: "+stats.getAverage());
                    System.out.println("Standard deviation: "+stats.getStandardDeviation());
                    System.out.println("Standard error: "+stats.getStandardError());
```

```
	}
}
```

A4.14 Calculating phylogenetic clusters

```
public class BuildProteinRelationGraph {
	public static void main(String args[]){
		double threshold=Double.parseDouble(args[0]);
		File matrix=new File(args[1]);
		ArrayList<File> seqFiles=new ArrayList<File>();
		File out=new File(args[2]);
		for(int i=3;i<args.length;i++){
			seqFiles.add(new File(args[i]));
		}
		ArrayList<Fasta> seqs=createFasta(seqFiles);
		calculateAndWriteIdentity(seqs,out,NeedlemanWunsch.readSimilarityMatrix(matrix),10,1,threshold);
	}
	private static void writeYEDNodesSimple(HashMap<String,HashMap<String,Double>> graph,BufferedWriter
bw,double threshold,String[] nodes){

		try{
			int num=0;
			for(int i=0;i<nodes.length;i++){

					//if(connectTest(minConnect,e.getKey()) &&
interactionTest(minInteraction,e.getKey()))
						bw.write("node\n[id "+nodes[i].split("\\s+")[0]+"\nlabel
\"Segment "+nodes[i].split("\\s+")[1]+"\"\ngraphics\n[\nx "+0+"\ny "+0+"\nw "+(10)+"\nh 10\n]\n]\n");
					num++;


			}

		}catch(IOException e){
			e.printStackTrace();
		}
	}
	public static void writeYEDGraphSimple(HashMap<String,HashMap<String,Double>> graph,File out,double
threshold,String nodes[]){
		try{
			BufferedWriter bw=new BufferedWriter(new FileWriter(out));
			bw.write("Creator \"FredeGraph\"\ngraph\n[\nhierarchic      0\nlabel   \"\"\ndirected
0\n");
			writeYEDNodesSimple(graph,bw,threshold,nodes);

			Iterator<Entry<String,HashMap<String, Double>>> it=graph.entrySet().iterator();
			while(it.hasNext()){
				Entry<String,HashMap<String, Double>> e=it.next();
				String i=e.getKey();
				HashMap<String, Double> hm2=e.getValue();
				Iterator<Entry<String,Double>> it2=hm2.entrySet().iterator();

				while (it2.hasNext()){
					Entry<String,Double> e2=it2.next();
					String j=e2.getKey();

					String node1=i;
					String node2=j;
					if((graph.get(node1).get(node2)>=threshold ))
					{
						bw.write("edge\n[\nsource "+i.split("\\s+")[0]+"\ntarget
"+j.split("\\s+")[0]+"\ngraphics\n[\nwidth "+graph.get(i).get(j)+"\n]\n]\n");

					}
				}
			}
			bw.write("]");
			bw.close();
		}catch(IOException e){
			e.printStackTrace();
		}
	}
	public static void calculateAndWriteIdentity(ArrayList<Fasta> seqs,File
out,HashMap<Character,HashMap<Character,Integer>> subMat,double gapOpen,double gapCont,double t){
			HashMap<String,HashMap<String,Double>> graph=new HashMap<String,
HashMap<String,Double>>();
			HashMap<String,Boolean> nodesHM=new HashMap<String, Boolean>();
			for(int i=0;i<seqs.size();i++){
				String id1=i+" "+seqs.get(i).getIdent().split("\\s+")[0];
				String seq1=seqs.get(i).getSequence();
				nodesHM.put(id1,true );
				System.out.println("Sequence "+i+" of "+seqs.size());
				for(int j=i+1;j<seqs.size();j++){
					String id2=j+" "+seqs.get(j).getIdent().split("\\s+")[0];
					String seq2=seqs.get(j).getSequence();
					//System.out.println(id1+" "+id2);
					NeedlemanWunsch nw=new NeedlemanWunsch(seq1, seq2,subMat ,
gapOpen,gapCont);
					double pwi=nw.getPairwiseIdentity();
					nodesHM.put(id2, true);
					if(pwi>t){
						insert(id1,id2,pwi,graph);
					}
```

```
                                        }
                                }
                                String[] nodes=nodesHM.keySet().toArray(new String[0]);
                                writeYEDGraphSimple(graph, out, t,nodes);
                                writeGraph(graph,new File(out+".dat"));

        }
        private static void writeGraph(HashMap<String,HashMap<String,Double>> graph,File out){
                try{
                        BufferedWriter bw=new BufferedWriter(new FileWriter(out));
                        Iterator<Entry<String,HashMap<String,Double>>> it=graph.entrySet().iterator();
                        while(it.hasNext()){
                                Entry<String,HashMap<String,Double>> e1=it.next();
                                Iterator<Entry<String,Double>> it2=e1.getValue().entrySet().iterator();
                                while(it2.hasNext()){
                                        Entry<String,Double> e2=it2.next();
                                        bw.write(e1.getKey()+"\t"+e2.getKey()+"\t"+e2.getValue()+"\n");
                                }
                        }
                        bw.close();
                }catch(IOException e){
                        e.printStackTrace();
                }
        }
        public static void insert(String id1,String id2,double value,HashMap<String,HashMap<String,Double>> graph){
                if(graph.containsKey(id1)){
                        graph.get(id1).put(id2, value);
                }else if(graph.containsKey(id2)){
                        graph.get(id2).put(id1, value);
                }else{
                        HashMap<String,Double> temp=new HashMap<String, Double>();
                        temp.put(id2, value);
                        graph.put(id1,temp);
                }
        }

        public static ArrayList<Fasta> createFasta(ArrayList<File> seqFiles){
                ArrayList<Fasta> seqs=new ArrayList<Fasta>();
                for(int i=0;i<seqFiles.size();i++){
                        ArrayList<Fasta> seqtemp=Fasta.readFasta(seqFiles.get(i));
                        for(int j=0;j<seqtemp.size();j++){
                                seqs.add(new
Fasta(i+"_"+seqtemp.get(j).getIdent(),seqtemp.get(j).getSequence()));
                        }
                }
                return seqs;
        }
}
```

A4.15 Pairwise identities for R200 sequences

```
public class PairwiseAlignmentWRAlign {
        ArrayList<Integer> list=new ArrayList<Integer>();
        public static void main(String args[]){
                PairwiseAlignmentWRAlign pwawr=new PairwiseAlignmentWRAlign();
                File folder=new File(args[0]);
                int repetitions=Integer.parseInt(args[1]);
                File matrix=new File(args[2]);
                double GapOpen=Double.parseDouble(args[3]);
                double GapC=Double.parseDouble(args[4]);
                File[] files=folder.listFiles();
                try{
                        BufferedWriter bwResults=new BufferedWriter(new FileWriter(new
File(folder+"/resultsWithoutReplacement.txt")));
                        bwResults.write("Average\tStdev\tStderr\tmax\tmin\n");
                        ArrayList<Double> average=new ArrayList<Double>();
                        ArrayList<Double> unexplained=new ArrayList<Double>();
                        ArrayList<Integer> ni=new ArrayList<Integer>();
                        for(int k=0;k<files.length;k++){

        if(!files[k].getAbsolutePath().endsWith("fas")&&!files[k].getAbsolutePath().endsWith("fasta"))continue;
                                System.out.println(files[k]);
                                HashMap<String,HashMap<String,Double>>
pwi=getPWI(files[k],matrix,GapOpen,GapC);
                                ArrayList<Fasta> fas1=Fasta.readFasta(files[k]);
                                ArrayList<Double> pw=new ArrayList<Double>();
                                double min=Integer.MAX_VALUE;
                                double max=Integer.MIN_VALUE;

                                BufferedWriter bw=new BufferedWriter(new FileWriter(files[k]+".out"));

                                for(int i=0;i<repetitions;i++){
                                        double sum=0;
                                        int j=0;
                                        pwawr.fillList(fas1.size());
                                        while(pwawr.getSize()>1){
                                                j++;
                                                int rand1=(int)(Math.random()*pwawr.getSize());

                                                Fasta seq1=fas1.get(pwawr.getItem(rand1));
                                                int rand2=(int)(Math.random()*pwawr.getSize());
                                                Fasta seq2=fas1.get(pwawr.getItem(rand2));
```

```
                                                  double
pwIdent=pwi.get(seq1.getIdent()).get(seq2.getIdent());

                                                  sum+=pwIdent;
                                        }

                                        double avg=sum/j;
                                        bw.write(avg+"\r\n");
                                        pw.add(avg);
                                        if(min>avg)min=avg;
                                        if(max<avg)max=avg;
                                }
                                bw.close();

                                Stats stats=new Stats(pw);
                                average.add(stats.getAverage());
                                unexplained.add(stats.unexplaineVar());
                                ni.add(fas1.size()*(fas1.size()+1)/2);
                                bwResults.write(files[k].getName()+"\t"+stats.getAverage());
                                bwResults.write("\t"+stats.getStandardDeviation());
                                bwResults.write("\t"+stats.getStandardError());
                                bwResults.write("\t"+min);
                                bwResults.write("\t"+max);
                                bwResults.write("\t"+stats.unexplaineVar()+"\n");


                        }
                        bwResults.close();
                }catch(IOException e){
                        e.printStackTrace();
                        System.exit(-1);
                }
        }
        private int getItem(int index){
                int item=list.get(index);
                list.remove(index);
                return item;
        }
        private void fillList(int number){
                list=new ArrayList<Integer>();
                for(int i=0;i<number;i++){
                        list.add(i);
                }
        }
        private int getSize(){
                return list.size();
        }
        public static HashMap<String,HashMap<String,Double>> getPWI(File in,File matrix,double GapOpen,double GapC){
                HashMap<String,HashMap<String, Double>> pwi=new HashMap<String, HashMap<String,Double>>();
                ArrayList<Fasta> fas=Fasta.readFasta(in);
                HashMap<Character,HashMap<Character,Integer>> submat=NeedlemanWunsch.readSimilarityMatrix(matrix);
                for(int i=0;i<fas.size();i++){
                        for(int j=i+1;j<fas.size();j++){
                                String seq1=fas.get(i).getSequence().toUpperCase();
                                String seq2=fas.get(j).getSequence().toUpperCase();
                                NeedlemanWunsch nw=new NeedlemanWunsch(seq1,seq2,submat,GapOpen,GapC);
                                double pw1=nw.getPairwiseIdentity();
                                nw=new
NeedlemanWunsch(DNAmanipulations.reverse(seq1).toUpperCase(),seq2,submat,GapOpen,GapC);
                                double pw2=nw.getPairwiseIdentity();
                                System.out.println(pw1+" "+pw2);
                                double identity=Math.max(pw1,pw2);
                                if(!pwi.containsKey(fas.get(i).getIdent())){
                                        HashMap<String,Double> temp1=new HashMap<String, Double>();
                                        temp1.put(fas.get(j).getIdent(),identity );
                                        pwi.put(fas.get(i).getIdent(), temp1);
                                }else{
                                        pwi.get(fas.get(i).getIdent()).put(fas.get(j).getIdent(),identity );
                                }
                                if(!pwi.containsKey(fas.get(j).getIdent())){
                                        HashMap<String,Double> temp1=new HashMap<String, Double>();
                                        temp1.put(fas.get(i).getIdent(),identity );
                                        pwi.put(fas.get(j).getIdent(), temp1);
                                }else{
                                        pwi.get(fas.get(j).getIdent()).put(fas.get(i).getIdent(),identity );
                                }

                        }
                }
                return pwi;
        }

}
```

OPEN ACCESS Freely available online

PLoS GENETICS

Within-Genome Evolution of REPINs: a New Family of Miniature Mobile DNA in Bacteria

Frederic Bertels[1]*, **Paul B. Rainey**[1,2]

1 New Zealand Institute for Advanced Study and Allan Wilson Centre for Molecular Ecology and Evolution, Massey University at Albany, Auckland, New Zealand, **2** Max Planck Institute for Evolutionary Biology, Plön, Germany

Abstract

Repetitive sequences are a conserved feature of many bacterial genomes. While first reported almost thirty years ago, and frequently exploited for genotyping purposes, little is known about their origin, maintenance, or processes affecting the dynamics of within-genome evolution. Here, beginning with analysis of the diversity and abundance of short oligonucleotide sequences in the genome of *Pseudomonas fluorescens* SBW25, we show that over-represented short sequences define three distinct groups (GI, GII, and GIII) of repetitive extragenic palindromic (REP) sequences. Patterns of REP distribution suggest that closely linked REP sequences form a functional replicative unit: REP doublets are over-represented, randomly distributed in extragenic space, and more highly conserved than singlets. In addition, doublets are organized as inverted repeats, which together with intervening spacer sequences are predicted to form hairpin structures in ssDNA or mRNA. We refer to these newly defined entities as REPINs (REP doublets forming hairpins) and identify short reads from population sequencing that reveal putative transposition intermediates. The proximal relationship between GI, GII, and GIII REPINs and specific REP-associated tyrosine transposases (RAYTs), combined with features of the putative transposition intermediate, suggests a mechanism for within-genome dissemination. Analysis of the distribution of REPs in a range of RAYT–containing bacterial genomes, including *Escherichia coli* K-12 and *Nostoc punctiforme*, show that REPINs are a widely distributed, but hitherto unrecognized, family of miniature non-autonomous mobile DNA.

Citation: Bertels F, Rainey PB (2011) Within-Genome Evolution of REPINs: a New Family of Miniature Mobile DNA in Bacteria. PLoS Genet 7(6): e1002132. doi:10.1371/journal.pgen.1002132

Editor: David S. Guttman, University of Toronto, Canada

Received December 16, 2010; **Accepted** May 2, 2011; **Published** June 16, 2011

Copyright: © 2011 Bertels, Rainey. This is an open-access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Funding: FB acknowledges a post-graduate scholarship from the Allan Wilson Centre; PBR is supported by a James Cook Research Fellowship from the Royal Society of New Zealand. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing Interests: The authors have declared that no competing interests exist.

* E-mail: f.bertels@massey.ac.nz

Introduction

Short repetitive sequences are a feature of most genomes and have consequences for genome function and evolution [1,2]. Often attributable to the proliferation of selfish elements [3,4], short repeats also arise from amplification processes, such as replication slippage [5] and *via* selection on genome architecture [6–8].

Repetitive DNA in bacterial genomes is less prominent than in eukaryotes, nonetheless, an over abundance of short oligomers is a hallmark of almost every microbial genome [9]. Known generically as interspersed repetitive sequences, these elements have a history of exploitation as signatures of genetic diversity (e.g., [10–12]), but their evolution, maintenance and mechanism of within- and between-genome dissemination are poorly understood [9,13–16].

Interspersed repetitive sequences fall into several broad groups each sharing short length (individual units range from ~20 to ~130 bp), extragenic placement, and palindromic structure [9,17]. REPs (repetitive extragenic palindromic sequences) – also known as PUs (palindromic units) – range from ~20 to ~60 bp in length, possess an imperfect palindromic core, are widespread among bacteria, and occur hundreds of times per genome [13,18–23]. While often existing as singlets, REPs also form a range of complex higher order structures termed BIMEs (bacterial interspersed mosaic elements) [14]. CRISPRs (clustered regularly interspaced short palindromic repeats) are a further, higher order composite of REP-like sequences that are formed from direct repeats of short (~30 bp) palindromic sequences interspersed by similar size unique non-repeated DNA ([24]; reviewed in [25]). Recent work shows that the unique sequences are often phage derived and that CRISPRs, along with associated proteins, confer resistance to phage by targeting viral DNA [25,26].

Non-autonomous DNA transposons form a more distinct family of repetitive sequences defined by their size (~100 to ~400 bp) and presence of terminal inverted repeats. Also known generically as MITEs (miniature inverted repeat transposable elements), non-autonomous transposons depend on transposase activity encoded by co-existing autonomous transposons for dissemination [4]. Identified initially in plants [27], where evidence of active transposition has been obtained [28], recent bioinformatic analyses suggest that they also occur in bacteria [29,30]. For example, ERICs (enterobacterial repetitive intergenic consensus) – found in a range of enteric bacteria including *Escherichia coli*, *Salmonella* and *Yersinia* [31] – and NEMISs (*Neisseria* miniature insertion sequences) in pathogenic neisseriae [32] are thought to be non-autonomous transposons (MITEs).

Scenarios for the origins and functional significance of non-autonomous elements, and to a lesser extent CRISPRs, can be envisaged, but this is not so for the majority of short interspersed repetitive sequences. Nonetheless, studies of specific elements in particular genetic contexts have uncovered evidence of functional roles ranging from transcription termination and control of

Author Summary

DNA sequences that copy themselves throughout genomes, and make no specific contribution to reproductive success, are by definition "selfish." Such DNA is a feature of the genomes of all organisms and evident by virtue of its repetitive nature. In bacteria the predominant repetitive sequences are short (~20 bp), extragenic, and palindromic. These so-called REP sequences may occur many hundreds of times per genome, but their origins and means of dissemination have been a longstanding mystery. We show that REPs are components of higher-order replicative entities termed REPINs, which are themselves thought to be derived from REP sequences that flanked an ancestral autonomous selfish element. In this ancestral state the REP sequences were likely to have been critical for the movement of the selfish element, but were devoid of any capacity to replicate independently. REPINs, on the other hand, have evolved to have a life of their own, albeit one that exploits—even enslaves—a genetic element upon which their existence depends. REPINs are the ultimate non-autonomous, super-streamlined, selfish element and are widespread among bacteria.

mRNA stability, to binding sites for DNA polymerase I (reviewed in [9]). However, the fact that the distribution and abundance of elements show substantial among-strain diversity [16,22] suggests that the range of functional roles is incidental, arising from, for example, co-option or genetic accommodation [31].

Differences in the distribution and abundance of repetitive elements among closely related strains carries additional significance in that it suggests that the evolution of these elements is independent of the core genome. This is particularly apparent from comparisons of closely related strains. For example, *Pseudomonas fluorescens* isolates SBW25 and Pf0-1 are closely related and yet highly dissimilar in terms of the nature, abundance and distribution of interspersed repetitive elements [22], even, as we show here, at the level of REPs. While this may reflect unequal rates of element loss, an alternative possibility is independent acquisition. Implicit in this suggestion is the notion that repetitive elements are genetic parasites [13,31,33].

The idea that REPs are selfish elements is not new [13,31,33]; however, there is little evidence – either direct or indirect – to support such an assertion. Indeed, the small size of REPs makes a mechanism for autonomous replication difficult to envision, however, the recent discovery of a proximal association between REPs and IS*200*-like elements, termed RAYTs (REP-associated tyrosine transposases) [23], raises interesting possibilities and suggests shared ancestry between RAYTs and certain REP families.

Evolutionary approaches to the analysis of sequence motifs can be highly informative [34]. While there is a ready tendency to assume that motifs recognized by search algorithms have functional significance, this need not be so. Neutral evolutionary processes alone (nothing more than random chance) ensure that short sequences will occur multiple times within any given genome. Thus, before concluding functional significance, it is necessary to test the null hypothesis of chance. Should this hypothesis be rejected, then the conclusion that over-abundance of short sequences is attributable – at least in part – to natural selection is sound. Moreover, evidence for selection justifies the assumption of functional significance. A key issue, however, is the level of biological organization at which functionality has been selected. There are two distinct possibilities: short repeats may have evolved because of selective benefits conferred on the cell, but alternatively, they may deliver benefits at the level of the gene – more specifically, at the level of a genetic element, of which the repeat sequence is a component. Distinguishing between these two alternatives is possible, although not necessarily straightforward. Indeed, whereas on initial emergence, selection is likely to operate exclusively at one level, over time, it is likely to shift to encompass multiple levels [4,16].

Here, we take a fresh and unbiased look at bacterial genome sequences in order to analyze the frequency and nature of short sequence repeats. Our approach is informed by evolutionary theory and begins free of assumptions regarding functional significance. Accordingly, the null hypothesis that short sequence repeats are no more frequent than expected by chance is the initial focus. We begin by interrogating the *P. fluorescens* SBW25 genome. Using suitable null models we show that over-abundant oligomers – which cannot be accounted for by chance alone – fall into three separate groups, each with characteristics typical of REPs. Highly significant differences in patterns of REP abundance and diversity between SBW25 and a second closely related *P. fluorescens* strain led us to question the hypothesis that the causes of REP diversity are linked to cellular function. This prompted a search for a replicative unit, which, based on patterns of REP distribution, we argue is a REP doublet. We refer to these entities as REPINs (REP doublets forming hairpins) and provide evidence from population sequencing for the existence of a putative transposition intermediate. Finally, extension to a range of RAYT-containing bacterial genomes including *E. coli* K-12 and *Nostoc punctiforme* indicate that REP sequences, organized as REPINs, define a class of hitherto unrecognized miniature non-autonomous mobile DNA.

Results

Oligonucleotide frequencies in *P. fluorescens* SBW25 and comparison to null models

Defining repetitive DNA on the basis of short sequences ranging from 10–20 nucleotides is simple and can be done logically without invoking heuristics and approximations (for longer sequences exact repetitions are rare). Figure 1 shows that the *P. fluorescens* SBW25 genome harbors numerous repetitive sequences: the most common 10-mer occurs 832 times; the most common 20-mer occurs 427 times. While these numbers appear significant, it is possible that they are no more than expected by random chance. To test this hypothesis, 100 random genomes were generated, with the same dinucleotide content, replication bias and length, as the SBW25 genome. The frequency of the most abundant oligonucleotides was determined from both leading and lagging strands. Figure 1 shows that the most abundant 10-mer from the randomly generated genomes occurs 304 times. For longer sequence lengths this number rapidly decreases (four instances in the case of 20-mers): the number of repeats expected by chance alone is thus much lower than observed. In total, there are 108 different 10-mers and 14,351 different 20-mers that occur significantly more often in the *P. fluorescens* genome than the most abundant oligonucleotides from randomly generated genomes ($P<0.01$, Figure S1). While compelling evidence for the existence of over-representation of short sequences, gene duplications could in part account for these findings [35]. We therefore sought an alternative null model.

P. fluorescens Pf0-1, one of the closest relatives of SBW25, shares the same GC-content and has a highly similar dinucleotide content (Table S1); coding density differs by 1.7% and the genome length differs by 4% (6,722,539 bp for SBW25 and 6,438,405 bp for Pf0-1, [22]). The close similarity means that any bias in the

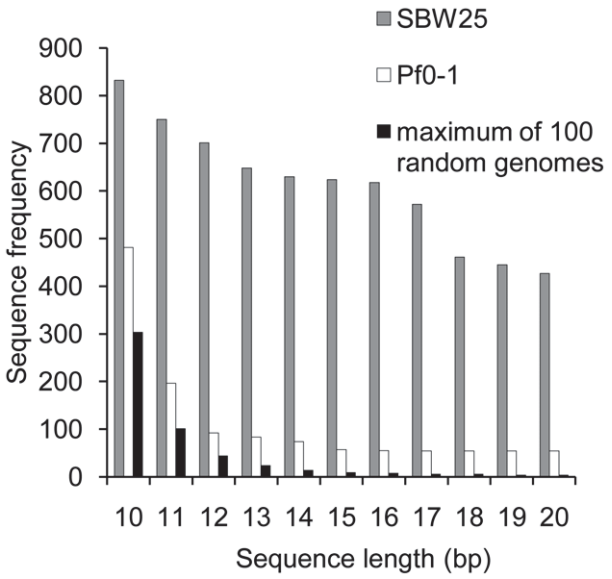


Figure 1. Frequency of common oligonucleotides in the genome of *P. fluorescens* SBW25. Data shows comparisons to both a random model, and to the closely related *P. fluorescens* Pf0-1 genome. The random model is based on 100 genomes generated with the same dinucleotide content, replication bias and length as the SBW25 genome. *P. fluorescens* Pf0-1 shares the same GC-content as SBW25 and has a highly similar dinucleotide content (Table S1); coding density differs by 1.7% and the genome length differs by 4%.
doi:10.1371/journal.pgen.1002132.g001

representation of short sequences due to duplicative evolutionary processes, or other selective mechanisms, should be similar in both genomes.

As in SBW25, over-represented short sequences in Pf0-1 are more frequent than expected by chance (Figure 1), however, a considerable difference in short sequence frequency is apparent. The difference between SBW25 and Pf0-1 is greatest at a sequence length of 16, where the most abundant sequence in SBW25 occurs 618 times – over 11 times more frequently than the most abundant 16-mer in Pf0-1 (Figure S2). On the basis of comparisons to both the random null model and the Pf0-1 genome we deemed all SBW25 16-mers occurring more than 55 times (the frequency of the most abundant 16-mer in Pf0-1) to be over-represented. This led us to reject the null hypothesis that chance alone explains the occurrence of short repetitive sequences in the SBW25 genome. Accordingly, we attribute over-representation of oligonucleotides to selective processes.

Short repetitive sequences in *P. fluorescens* SBW25 are synonymous with REPs

The collection of over-represented 16-mers together encompasses 96 different sequences; however, a cursory glance suggested that many share similarity. Using a grouping method designed to detect overlapping subsets of sequences (Methods and Figure S3), the 96 sequences were found to be members of just three separate sequence groups (GI, GII and GIII (Figure S4)), each containing an imperfect palindrome (the palindrome overlaps the most abundant 16-mer in GI and GII, but is part of the most abundant 16-mer in GIII (Table 1)). The most abundant 16-mers of each group together occur 1,067 times. The majority of these sequences are extragenic; only 14 16-mers overlap with genes. Together these data show that the three groups of 16-mers are over-represented in the SBW25 genome, contain an imperfect palindromic core and are primarily extragenic. Possessing the hallmarks of repetitive extragenic palindromic (REP) sequences, we conclude that the three groups of 16-mers are, for all intents and purposes, synonymous with REPs.

Determining REP sequence family size

In order to accommodate the possibility of related family members, we generated a pool of sequences that differed to GI, GII and GIII sequences by up to four bases. This generated 488,373 different 16-mers of which 1,861 were located in extragenic space. To define the proportion of false positives the search was repeated by interrogating randomly generated extragenic space (with the same dinucleotide content and length of each individual extragenic space) for matches to the 488,373 different 16-mers. This showed that 12% of all sequences with up to four substitutions are false positives (sequences unrelated to GI, GII or GIII). Repeating the analysis with the subset of sequences, which differ firstly by three and subsequently, two substitutions showed that 2% and 0.2% of matches are false positive, respectively. For two substitutions the false positive rate is low enough to conclude that the described repetitive sequence families consist of at least 1,422 members (Table 2). The precise number of members belonging to each of the GI, GII and GIII groups cannot be determined because with a degeneracy of two, some sequences fall into more than one group.

The distribution of REP sequences in the genome of SBW25

The selective causes for the prevalence of GI, GII and GIII sequences in the SBW25 genome are of considerable interest. Although implicit in many studies is the notion that REP-like sequences have evolved because of their selective benefit to the cell (as transcription binding sites, termination signals and the like [20,36,37]), it is also possible that selection has favored their evolution as a consequence of benefits delivered to a genetic (parasitic) element, of which the repeat sequence is a component. The highly significant differences in the frequency, nature and genomic location of short repetitive sequences in SBW25, compared to Pf0-1 make a compelling case for the latter.

If the prevalence of GI, GII and GIII sequences is a consequence of gene-level selection, then this implies the existence of a replicative entity – a genetic element that has the capacity to reproduce within the genome. The distribution of REP sequences is likely to provide some information. One way to quantify the distribution is to measure distances between neighboring REP sequences and compare these to distances between REPs generated by a null (random) model. If individual REPs are randomly distributed then this would suggest the individual REP

Table 1. Short repetitive sequence groups in the SBW25 genome.

Group[a]	Sequence[b]	Occurrences	Palindromic core[c]
I	GTGGGAGGGGGCTTGC	618	GGGGGCTTGCCCCC
II	GTGAGCGGGCTTGCCC	241	GCGGGCTTGCCCCGC
III	GAGGGAGCTTGCTCCC	208	GGGAGCTTGCTCCC

[a]16-mers were sorted into three groups (GI, GII and GIII) using a grouping algorithm (Figure S3 and Figure S4).
[b]Sequence of the most common 16-mer from each group.
[c]Each GI, GII and GIII sequence either contains, or overlaps, an imperfect palindrome (the palindromic core).
doi:10.1371/journal.pgen.1002132.t001

Table 2. Frequency of GI, GII, and GIII 16-mers in the extragenic space of the SBW25 genome.

	Number of occurrences	
Number of 16-mers[a]	**Extragenic space**	**Randomly assembled extrangenic space[b]**
0 substitutions (3 sequences)	1053	<0.01
1 substitution (147 sequences)	1249	0.13±0.33
2 substitutions (3,387 sequences)	1422	2.24±1.41
3 substitutions (48,707 sequences)	1560	31.18±5.18
4 substitutions (488,373 sequences)	1861	264.74±15.87

[a]In order to identify closely related members of each GI, GII and GIII sequence family extragenic space was searched for all possible sequences that differed by up to four substitutions. The number in brackets is the number of variant sequences: e.g., with no substitutions there are just the three sequences (Table 1); allowing one substitution there are 147 different sequences, and so forth. The number found in extragenic space was compared to a null (random) model based on randomly assembled extragenic space (see text).

[b]Data are means and standard deviation from 100 independent extragenic space randomizations.

doi:10.1371/journal.pgen.1002132.t002

as replicative unit. If the distance between adjacent REPs is non-random, then this may suggest the evolving entity is some higher order arrangement of REPs.

To construct the null model, 1,053 (the number of invariant GI, GII and GIII sequences in extragenic space) non-overlapping 16 bp segments were positioned at random within the extragenic space of the SBW25 genome. This process was repeated 10,000 times and the average occurrence of the distance between neighboring elements calculated. Equivalent data for the 1,053 over-represented REPs is shown in Figure 2. A comparison between the two histograms reveals marked differences in the distributions of distances between next-neighbors. Most striking is the strong bias toward specific inter-element distances. This marked skew shows that REPs are not independently distributed and is suggestive of an underlying copying mechanism involving at least two REP sequences. Of note is the fact that doublets typically comprise pairs of identical GI, GII or GIII sequences and are rarely mixed (although some exceptions are discussed below) (Figure 2).

The replicative unit

To explore the possibility that the replicative unit is an entity comprised of two REP elements (a REP doublet) we determined the number of singlets, doublets, triplets and higher order arrangements of REPs (REP clusters) by examining the 400 bp flanking either side of each REP for the presence of REP sequences (Figure S5). Once again, the results of this analysis were compared to the null (random) model used above.

According to the random model, 58% of all REP sequences are expected to occur as singlets, whereas data from SBW25 shows that just 18% are singlets. In contrast, 61% of all REPs are organized as doublets, which is significantly greater than the 17% expected by chance (Table 3). Interestingly, REP triplets are rarer than expected, whereas several higher order arrangements of REPs, including two sets of twelve (see below), are more frequent than expected (Table 3).

The highly significant over-representation of REP doublets suggests that the doublet defines an appropriate replicative unit. If true, then the distribution of doublets across extragenic space should be unaffected by neighboring REP elements and should thus conform approximately to a null (random) model.

To test this hypothesis, random distributions of REP doublets over extragenic space were compared to actual REP clusters found in SBW25 (Table 4). However, because the distance between REPs (in the doublet conformation) varies (Figure 2), two random models were generated based on the two most common inter-REP spacings: 71 bp (a doublet of GI REPs) and 110 bp (a doublet of GII REPs). Simulations were based on the random assignment of 560 REP doublets (corresponding to the sum of REP clusters (of two or more) in Table 3) to extragenic space and were repeated 10,000 times. Although the two segments differ significantly in size, simulations for each family gave remarkably similar results (Table 4). Together these data show that the observed number resembles that predicted if the doublets are randomly distributed.

A further prediction concerns evolutionary processes affecting doublets vs. singlets. If REP doublets are the replicative unit, then singlets are likely to derive from doublets, either by decay (divergence) of the neighboring element, or by destruction of the doublet through insertion or deletion. In either case the REP singlet is expected to be non-functional (immobile) and thus subject to random genetic drift. REP doublets on the other hand – being (according to our hypothesis) functional and potentially mobile – are expected to be shaped by selection: genetic diversity of REP singlets should thus be greater than doublets. To test this hypothesis we extracted GI, GII and GIII sequences from the SBW25 genome plus all related sequences that varied by up to two positions. Since only two nucleotide differences distinguish GII and GIII sequences from a GI sequence, GII and GIII sequences were defined by two fixed (invariant) positions (GII: 2T, 6C; GIII: 6A, 13T). After extraction, sequences from each group were divided into a set of 16-mers obtained from singlets, a set of 16-mers from doublets and a set of 16-mers obtained from clusters (where a cluster contains three or more REPs). For all nine sequence groups (three from each GI, GII and GIII group) the pairwise identity was calculated (Figure 3, see Methods for details). The average pairwise identity of 16-mers obtained from REP doublets is significantly greater than the average pairwise identity of 16-mers obtained from REP singlets: this is true for comparisons within each of the REP groups ($P<1e\text{-}10$ for GI; $P<1e\text{-}8$ for GII and GIII).

Analysis of the organization of REP doublets shows that in the majority of cases, pairs of REPs (93% of all 430 REP doublets) – of

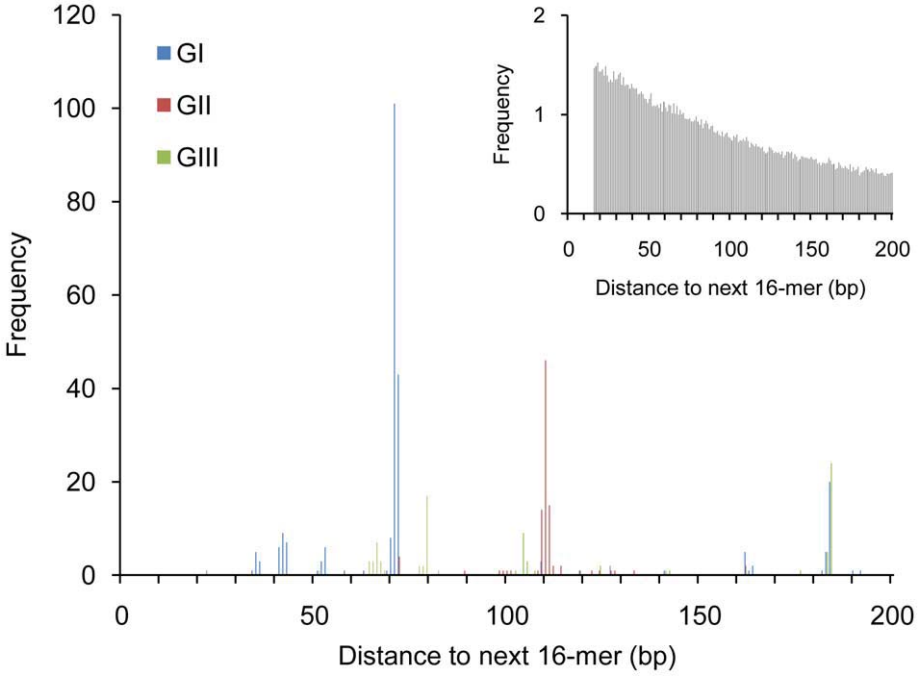


Figure 2. Frequency of next-neighbor distances for GI, GII, and GIII sequences in the genome of *P. fluorescens* SBW25. Data are next-neighbor distances for 1,053 GI, GII and GIII sequences in extragenic space, compared to a random model (inset). The peaks at 71 and 110 bp correspond to doublets of GI and GII sequences, respectively. The peak at 184 bp corresponds to GI–GIII tandem repeat clusters (see text). No significant deviation from the random model was noted for next-neighbor distances above 200 bp. The next-neighbor distances of 16-mers randomly assigned to extragenic space is the average of 10,000 simulations (inset).
doi:10.1371/journal.pgen.1002132.g002

Table 3. Frequency of REP clusters within the SBW25 genome.

Cluster Size	Number of occurrences		*P*-Value	
	Observed[a]	Expected (random model)[b]	≤[c]	≥[d]
1	267	832±22.24	1	0
2	431	181.4±11.12	0	1
3	26	44.3±6.1	0.9998	0.0009
4	12	13.1±3.42	0.6658	0.4537
5	1	4.38±1.96	0.9893	0.0615
6	6	1.67±1.03	0.0070	0.9989
7	5	0.66±0.65	0.0007	0.9999
8	5	0.31±0.46	0	1
9	3	0.14±0.35	0.0006	1
10	0	0.07±0.25	1	0.9364
11	0	0.04±0.18	1	0.9658
12	2	0.02±0.14	0	1
Sum	1422	1421.76		

Data are the number of REPs occurring as clusters (from singlets to clusters of 12) in extragenic space compared to expectations from a null model based on the random assignment of 1,422 16-mers (to extragenic space) (see text).
[a]Observed occurrences from the SBW25 genome.
[b]Expected values (means and standard deviation) based on 10,000 simulations.
[c]The proportion of times the observed frequency was less than or equal to the expected value.
[d]The proportion of times the observed frequency was greater than or equal to the expected value.
doi:10.1371/journal.pgen.1002132.t003

either the GI, GII, or GIII types – are organized as two inverted REP sequences that overlap the most abundant 16-mer (Figure 4A and 4B). While the spacer region between REPs shows less conservation than evident in the REPs themselves, secondary structure predictions for ssDNA shows that the conserved bases on each side pair resulting in a hairpin (Figure 4E). Thus, while selection appears to favor highly conserved nucleotide arrangements for REP and adjacent sequences, the critical features of the intervening sequence would appear to be length, and capacity to form a hairpin. Indeed, compensatory changes on either side of the predicted hairpin are common (Figure 4A).

Finally, if our assertion that the doublet defines a replicative entity is correct, then evidence of movement could in principle come from population sequencing. To this end we interrogated 55,768,706 paired-end Illumina reads (36–76 bp long) obtained from sequencing DNA extracted from 5×10e9 SBW25 cells, for evidence of insertion and excision events. A total of 18 putative insertions were detected, however, the possibility of false positives could not be discounted. A similar search for excision events proved more profitable: three single reads were identified which mapped to three different locations on the genome, each corresponding to unique sequences flanking a GI REP doublet (Figure 4C and Figure S6). However, the expected doublet was absent from all sequence reads leading us to conclude that these sequences were from DNA molecules from which the doublet had excised. Additionally, we observed 200 individual sequence reads spanning a GII REP doublet indicating its excision from the entire population (Figure S6). That these events could result from machine and/or chemistry error is improbably low. Furthermore, a search for evidence of REP singlet deletions from the ~56 million Illumina reads failed to find evidence of a single such event (see Methods).

Table 4. Frequency of REP doublets within the SBW25 genome.

Segment length	Cluster size	Number of occurrences		*P*-Value	
		Observed[a]	Expected (random model)[b]	≤[c]	≥[d]
71 bp	2	457	434.76±12.9	0.0990	0.9144
	4	13	46.3±5.75	1	0
	6	11	7.69±2.6	0.0832	0.9575
	8	8	1.63±1	0.0001	1
	10	0	0.4±0.5	1	0.7323
	12	2	0.12±0.3	0.0023	0.9999
	14	0	0.03±0.18	1	0.9787
	16	0	0.01±0.1	1	0.9932
	18	0	0.002±0.06	1	0.9980
Sum		560	559.98		
110 bp	2	457	419.2±13	0.0167	0.9874
	4	13	49.1±5.9	1	0
	6	11	9.4±2.8	0.2112	0.8715
	8	8	2.2±1.2	0.0001	1
	10	0	0.7±0.6	1	0.6112
	12	2	0.2±0.4	0.0078	0.9998
	14	0	0.09±0.25	1	0.9553
	16	0	0.02±0.16	1	0.9834
	18	0	0.02±0.1	1	0.9944
Sum		560	560.07		

Data are the frequency of REP clusters (from doublets to cluster of 18 REPs) found in extragenic space compared to a null model based on the random assignment of 560×71 bp and 560×110 bp segments (to extragenic space). REP clusters containing an uneven number of REP sequences are included in the next lower cluster size (REP singlets are omitted).

[a]Observed occurrences from the SBW25 genome.

[b]Expected values (means and standard deviation) based on 10,000 simulations.

[c]The proportion of times the observed frequency was less than or equal to the expected value.

[d]The proportion of times the observed frequency was greater than or equal to the expected value.

doi:10.1371/journal.pgen.1002132.t004

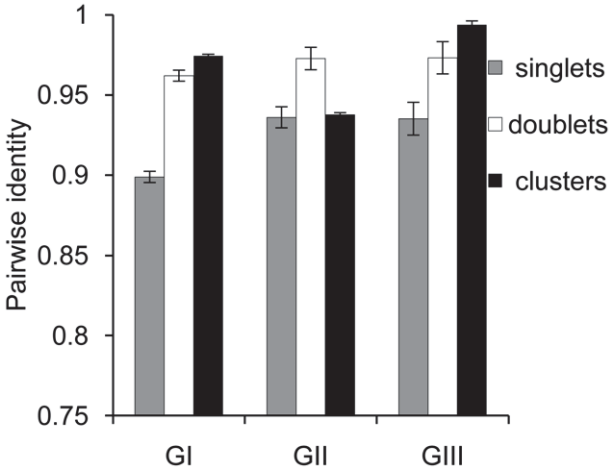


Figure 3. Average pairwise identity of REP sequences found in singlets, doublets, and clusters. Data are average pairwise identity of REPs found as singlets, doublets and clusters (clusters contain more than three REPs). Error bars show standard deviation. Statistical testing (jackknife) shows the average pairwise identity of 16-mers from REP doublets (and clusters for GI and GIII, *P*-value<1e-10) to be significantly greater than the average pairwise identity of 16-mers obtained from REP singlets: this is true for comparisons within each of the REP groups (*P*<1e-10 for GI; *P*<1e-8 for GII and GIII).

doi:10.1371/journal.pgen.1002132.g003

Details of the three excised GI doublets are shown in Figure 4C and 4D. Of particular interest is the asymmetrical nature of the deleted sequence: in all instances it begins (in the left-hand (5′) end (Figure 4B)) at the start of the invariant sequence defined by the most conserved 16-mer and extends through the spacer region into the second REP sequence. However, rather than finish at the end of the conserved 16-mer, the deletion truncates at the 3′-end of the right-hand REP sequence, leaving the last ~6 bp of invariant sequence intact (Figure 4C).

Secondary structure predictions show a hairpin structure with a 5′-single strand tail. Although the structures of the hairpins are not identical (due to differences in the sequence of the space region) the 5′-tail is a feature of the excised entity in all instances (Figure 4E). It is possible that the excised sequences define a putative transposition intermediate.

Together the above analyses implicate REP doublets as a unit of selection: a family of mobile DNA that has, until now, eluded recognition. Although REP doublets have previously been noted as one of many different higher order arrangements of REPs, they have not before been implicated as replicative entities [16–20]. Furthermore, in previous discussions of higher order arrangements it has been assumed that the singlet is the basic building block. In contrast, our data supports the view that REP singlets are defunct remnants of once functional REPINs. Because of their likely evolutionary relevance, a label that defines the replicative entity appears warranted. Henceforth we refer to REP doublets forming hairpins as REPINs.

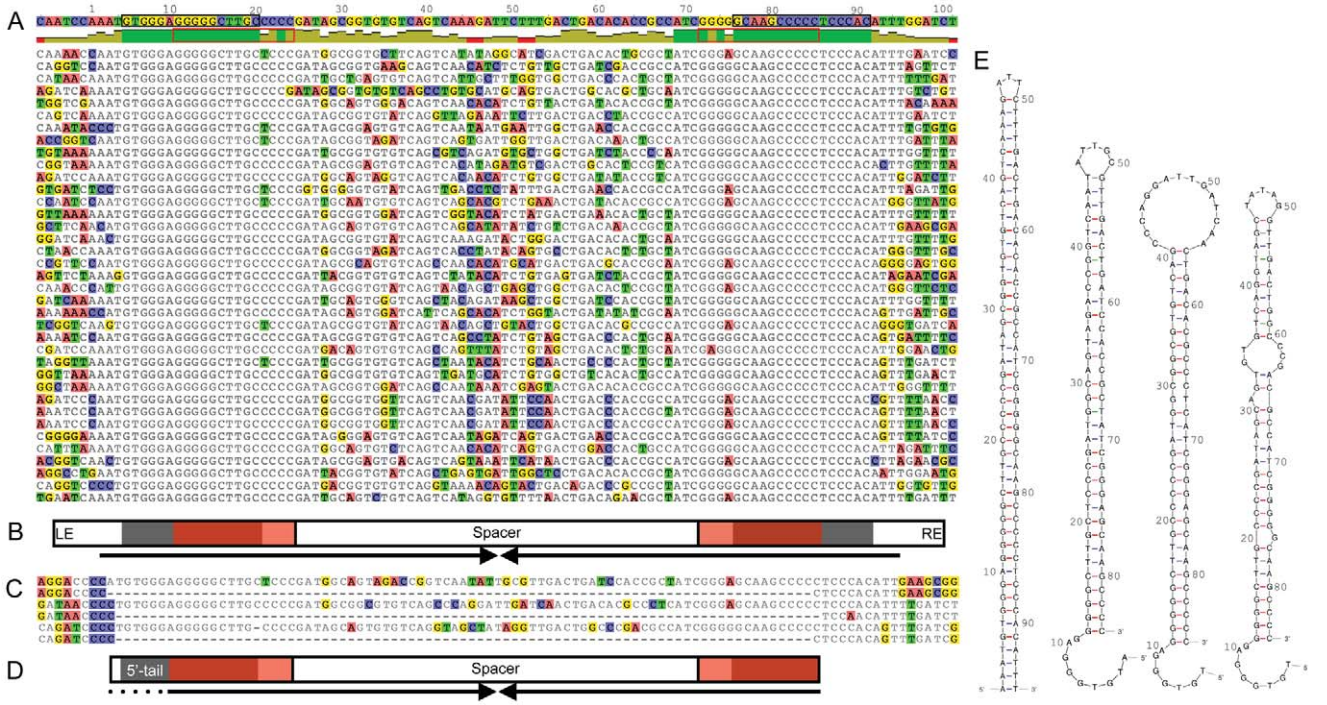


Figure 4. General organization and predicted secondary structure of REPINs. (*A*) Alignment of 101 GI REP doublets forming hairpins (REPINs) from SBW25 (37 are shown) shows a symmetrical (palindromic) organization comprised of two highly conserved regions separated by a spacer. Top line shows the consensus sequence followed by a graph displaying identity to the consensus (green denotes 100% identity). Two invariant regions of 16 bp are found in the left and right ends (LE, RE). These sequences are organized as inverted repeats and define the most abundant 16-mer in the SBW25 genome (black box). Each 16-mer overlaps a GI REP sequence (red box). (*B*) General REPIN features including LE and RE, each comprised of a highly conserved 16-mer (black) overlapping a REP sequence (red), with the two ends separated by a spacer. For a GI doublet the distance between the first residues of the two invariant 16-mers is 71 bp. Complementary bases permit formation of a hairpin structure (arrows). (*C*) Three excision events detected from Illumina sequencing reads reveal a putative transposition intermediate. Full-length sequences show three genomic regions located between 2,577,312–2,577,231, 3,857,520–3,857,439 and 5,683,545–5,683,624 bp on the SBW25 genome, each of which contains a REPIN. The partial sequences below each genomic region are Illumina reads from which the REPIN has been excised (see also Figure S6). (*D*) Cartoon of the excised region indicating putative transposition intermediate. Note the 5′-tail, which generates an asymmetrical sequence. (*E*) Secondary structure prediction for the consensus GI REPIN shows that the conserved bases on each side can pair resulting in a long hairpin (*E*, left). Predictions for transposition intermediates in the same order as the alignments in (*C*): the second, third and fourth hairpin correspond to the first, second and third alignment. The single stranded 5′-tail is free to pair with a complementary sequence.
doi:10.1371/journal.pgen.1002132.g004

REPIN clusters

While the majority of REPINs exist as singlets, some higher order arrangements are apparent (above and Table 4). These are of two main types: those showing a distinctive ordering and those with no apparent structure.

REPINs occurring in ordered clusters are typically arranged as tandem repeats of nearly identical REPINs – including the flanking sequences (Figure S7). With 16 such clusters distributed throughout the genome, these arrays are the most common higher order arrangement of REPINs in SBW25. The largest cluster consists of four REPINs (plus an additional REP sequence) with a total length of over 700 bp.

Three higher order REPIN clusters are of particular note: one from each of the three distinctive REPIN groups (GI, GII and GIII) each located adjacent to one of the three recently identified REP-associate tyrosine transposases (RAYTs, [23]) (*pflu3939*, *pflu4255* and *pflu2165*). The fact that a different REPIN cluster is located beside each of the RAYTs, combined with the fact that REPINs (and REPs) in SBW25 come in three distinct flavors, raises the possibility that RAYTs are intimately linked to REPIN mobilization (Figure 5).

REPINs in clusters lacking obvious organization are found in five regions of the genome and typically consist of two unrelated REPINs. Close inspection suggests that these clusters are formed by insertion of REPINs into, or next to, existing REPINs.

Tandemly repeated REP sequences

REPs also form higher order arrangements. These are of two distinct types: the first involves highly organized tandem arrays of GI and GIII REP sequences: GI REPs are separated from GIII REPs by 112 bp; GIII REPs are separated from GI REPs by

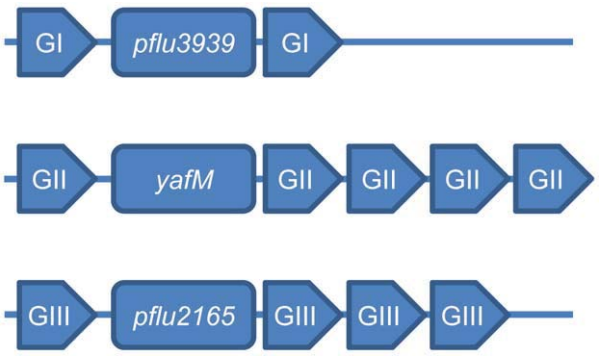


Figure 5. Proximity of GI, GII, and GIII REPIN clusters to RAYT genes in the *P. fluorescens* SBW25 genome. The RAYT genes in SBW25 are *pflu3939*, *yafM* and *pflu2165*.
doi:10.1371/journal.pgen.1002132.g005

72 bp. Five such tandem arrays are located at ~2 Mbp all of which are found in forward orientation, six are found ~4 Mbp in reverse orientation (at a distance of ~2 Mbp from the origin of replication). The two largest tandem arrays both contain 12 GI and GIII sequences, one found at ~4.1 Mbp the other at ~2.5 Mbp (Figure S8). These two arrays are almost identical copies of each other, but found in opposite orientations on opposite sides of the genome. The second type of tandemly organized REP sequences consists solely of evenly spaced GI sequences found at two positions in the genome. Similar to the GI–GIII tandem arrays one GI tandem array is found in forward and the other one in reverse orientation.

REP sequence organization in other genomes

REPIN dissemination could occur *via* the exploitation of a functional transposase encoded separately within the genome. Non-autonomous DNA transposons (MITEs) do precisely this and typically consist of two inverted repeats. REPINs also consist of two inverted repeats (REP sequences) and, as mentioned above, may exploit the putative transposase encoded by RAYTs. If REP sequences in other genomes are components of REPINs – and disseminate *via* RAYT-encoded transposase activity – then, given the broad distributions of RAYTs [23], REPINs are likely to be a common feature of bacterial genomes; they are also likely to share common ancestry.

Although a fully comprehensive among-genome analysis is beyond the scope of this paper we nonetheless analyzed REP sequence clusters in a variety of genomes containing RAYTs. To this end REP sequences were selected from 18 different bacterial strains including all fully sequenced *Pseudomonas* genomes, the genomes of *E. coli* K-12 DH10B and *Salmonella enterica* serovar Paratyphi A AKU 12601 (chosen because of their significance for REP research) and the genomes of *Thioalkalivibrio* HL-EbGR7 and *N. punctiforme* PCC73102 (chosen because of their distant relation to *Pseudomonas*). A phylogenetic analysis of the RAYTs was firstly undertaken (Figure S9). Notably, RAYTs from these strains form two distinct evolutionary lineages with evidence of multiple independent introductions. For example, the genus *Pseudomonas* is separated into two sets of species defined by the presence of either 'clade I' or 'clade II' RAYTs. The genome of *Thioalkalivibrio* contains one clade I and one clade II RAYT. Several other genomes, in addition to SBW25, contain more than a single RAYT, but these almost never cluster. In fact the most closely related RAYTs are found in different genomes. Overall the distribution of RAYTs among distantly related organisms shows evidence of lateral gene transfer; however, at the species level, lateral gene transfer does not seem to occur frequently as evident by the fact that RAYT phylogeny is largely congruent with the relationship among species (Figure S9).

Since REP sequences have been shown to be associated with RAYT genes (this work and [23]), we interrogated non-coding DNA flanking each RAYT for 16-mers that were repetitive, extragenic and palindromic, that is, are REPs. In each instance a REP was identified (Table S2). To test the hypothesis that REPs are organized as REPINs an analysis of the distribution of REPs was performed on each genome as described above (also see Methods) and included all REP sequences that differed from the consensus by up to two nucleotides. Results were expressed as the ratio of REP singlets to doublets, where ratios greater than two indicate that REPs occur predominantly as singlets. Ratios less than two mean that REPs occur predominantly as doublets. Figure 6 shows a histogram of singlet to doublet ratios for REP sequences associated with clade I RAYTs. Of the 20 REP sequence classes (one associated with each RAYT, some genomes contain more than one RAYT e.g., SBW25) 17 had singlet to doublet ratios of less than two, indicating that most REPs occur as doublets. The majority of doublets contained REPs as inverted pairs (Table S3) as expected of REPINs.

Our simple search method did not return conclusive results for clade II REP sequences. One possibility is that the REPIN structure in these genomes is less conserved. To this end we performed a secondary structure prediction on a sample of REP sequences. In all instances we found the general REPIN composition to hold (two inverted REP sequences separated by a short stretch of DNA and forming a hairpin, Figure S10), with the exception of REP sequences found in *P. stutzeri*: interestingly no REPINs were identified in this genome.

We also analyzed higher order arrangements for clade I REP sequences, but these were not present in all analyzed genomes. They were predominantly found in *P. syringae* and *P. fluorescens*, although two REP sequence classes were also detected in *P. putida* (Table S3). No correlation was found between the singlet to doublet ratio and cluster formation.

Taken together, the systematic cluster analysis of clade I REP sequences and secondary structure prediction of a selection of clade II REP sequences suggest that the organization of REP sequences into REPINs is a necessary condition for REP sequence distribution.

Discussion

Short interspersed repetitive sequences are widely distributed in bacteria, but past studies have shed little light on their evolutionary origins. We began by examining the abundance and distribution of short sequences in *P. fluorescens* SBW25 and showed, by comparison against a random (null) model, and subsequently against Pf0-1, that short sequences are over-represented. Moreover, we found that short repetitive sequences fall into three distinct groups (GI, GII and GIII), each bearing characteristics typical of REP sequences, that is, they are repetitive, extragenic and palindromic.

In order to discount the possibility that REP sequences are the product of mutation pressure (a possibility already called into doubt by comparison to the random model) we took advantage of the closely related Pf0-1 genome. Comparisons using this null model – based upon a genome likely to have been shaped by similar underlying evolutionary processes – allowed us to emphatically reject the possibility that REP evolution can be explained by drift. Our data thus indicate natural selection as the primary driver of REP sequence evolution.

A critical issue is the nature of the entity upon which selection acts. Evidence that this entity comprises a doublet of REP sequences – a REP doublet forming a hairpin structure (REPIN) – came firstly from analysis of the distribution of REPs in extragenic space. The striking departure from a random model shown in Figure 2, along with clear bias toward specific distances between REPs, pointed to the REPIN as the replicative entity. The hypothesis was further tested by examining the distribution of REP doublets in extragenic space, by measuring nucleotide diversity in singlets versus doublets, and by analysis of the conserved features of REPINs. Finally, the existence of REPINs as actively mobile entities was bolstered through the discovery of four deletion events that may define putative transposition intermediates (Figure 4).

A previous analysis of the SBW25 genome using various repetitive DNA finding algorithms [22] revealed numerous repeat families. Two of these, the so named R0 and R2 repeats have characteristics similar to REPINs; indeed, a comparison (Table S4) shows a correspondence between REPINs and the R0 and R2

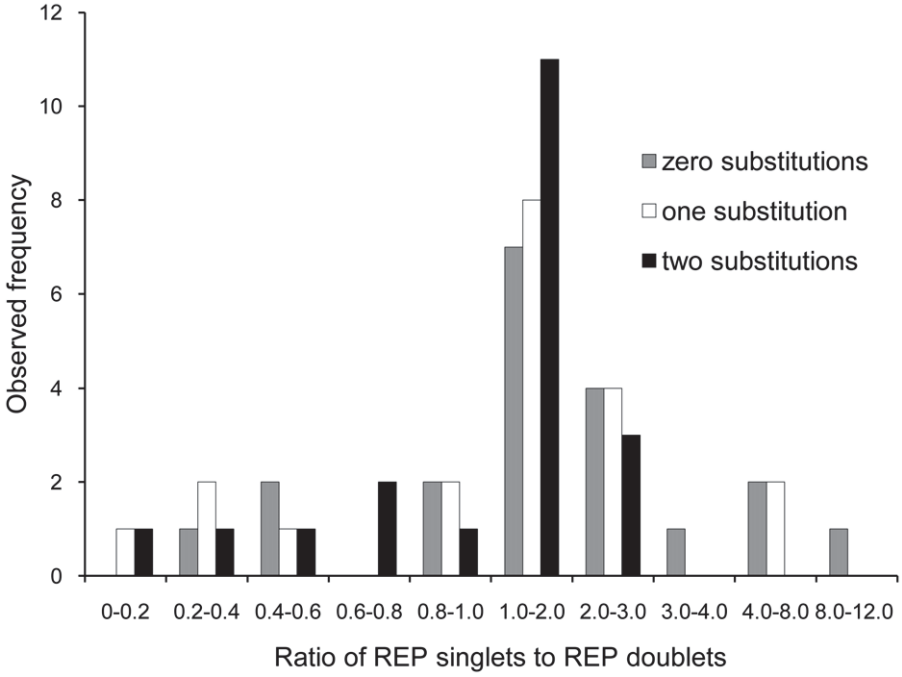


Figure 6. REP singlet to doublet ratios for REP sequences from bacterial genomes. Data are the most abundant 16-mers found within the flanking non-coding DNA of RAYT genes from 18 genomes. In order to include related 16-mers, a set of degenerate sequences was produced by allowing up to two substitutions per 16-mer.
doi:10.1371/journal.pgen.1002132.g006

repeats. In general R0 repeats map to GI REPINs, while R2 repeats correspond to a mixture of both GII and GIII REPINs.

The mechanism by which REPINs are disseminated is a central, but unresolved issue. Recently, a hypothesis concerning REP sequence distribution was put forward [23]. The authors proposed that REP movement is effected by RAYTs – so named Y1 transposases – that are distantly related to the IS*200*/IS*605* family of insertion sequences. Integral to the transposition of IS*608* (a member of the IS*200*/IS*605* family) are two imperfect (REP-like) palindromes that flank either side of the insertion sequence and which are recognized by the transposase [38]. Whereas Nunvar et al. [23] suggested that REPs are moved by RAYTs, our data leads us to predict that it is the REPIN (and not the REP) that is mobilized *via* the RAYT: REPINs could be transposed by a RAYT dimer encoded in trans that recognizes the REP doublet. This mechanism would result in the strong conservation of the two REP sequences that define a REPIN (Figure S11).

The suggestion that RAYTs are integral to REPIN movement is given additional support by the discovery of excision events that appear to define the transposition intermediate. At first glance the footprints differ from expectations given that they do not encompass the full extent of the conserved REPIN (Figure 4B). However the asymmetrical nature of the putative intermediate is telling, particularly in light of the unusual mechanism of IS*608* transposition. IS*608* transposes *via* a single stranded intermediate and exploits singled stranded DNA at the replication fork; moreover, the intermediate involves pairing of asymmetric ends [38–40].

Assuming the excised DNA (Figure 4C and 4D) is a transposition intermediate then a key issue is re-establishment of the symmetrical REPIN. This could happen if the 5′-tail was involved in target recognition and paired with complementary sequence. In this regard it is of interest to note that the 5′-tail of the putative intermediate, which secondary structure predictions show is unlikely to form part of the hairpin (Figure 4E), is complementary to the 3′-end of the REPIN. It is possible that a recognition event involving pairing between complementary sequences, perhaps mediated *via* the RAYT, integrates back into DNA leading to the formation of a new REPIN. Although further insight requires molecular investigations, there exist a number of striking parallels with the mechanism of transposition of the IS*200*/IS*605* family of insertion sequences to which RAYTs – and their associated REPINs – are related.

While the argument for REPINs as replicative entities is supported by substantive data, REP singlets are nonetheless a notable feature of the SBW25 genome. Our data – particularly the significantly lower pairwise identity of REP singlets compared to REP doublets – suggests that these singlets are non-functional remnants of REPINs. But this does not explain why REP singlets are common. A close analysis of REP singletons reveals several possible routes for single REP sequences to emerge from REPINs. One possibility stems from limitations of our sequence search algorithms. When REPINs evolve neutrally successive acquisition of point mutations naturally leads to one REP becoming more decayed than the partner. If the less decayed REP is only just on the verge of recognition by our sequence search, then it is likely that the more decayed REP partner sequence will escape detection. A biologically plausible possibility is that singlets arise from insertion of DNA into REPINs. Indeed, earlier studies have noted that REP sequences are targets for certain insertion sequences [22,41,42]. REP singlets could also arise by deletion of the sequence between two REPs within a single REPIN leading to a long palindromic structure that contains only a single REP sequence: precisely such events can be seen in the genome of SBW25 (F. Bertels and P. B. Rainey, unpublished). A further possibility is that selection may act to preserve individual REP sequences because of specific functional consequences [16,36].

A finding of note is the existence of several higher order arrangements of REPs and REPINs within the SBW25 genome, indeed, several such clusters occurred at a frequency above that expected from the null model (Table 3 and Table 4). Interestingly the majority of these clusters – at least those containing more than three REP sequences or REPINs – were arranged as highly ordered tandemly repeated units. This, combined with the fact that higher order arrangements were not found in all REPIN containing-genomes (Table S3), indicates a second mechanism for REP/REPIN cluster formation and suggests specific functional roles for these structures.

Extension of our analysis to a set of related (*Pseudomonas*) and unrelated (*E. coli*, *S. enterica*, *N. puctiforme* and *Thioalkalivibrio*) genomes each known to contain RAYTs showed that REPs in these bacteria are present in the immediate vicinity of RAYTs: moreover, in accord with predictions, these REPs are organized as REPINs. This finding greatly bolsters our conjecture that REPINs are a unit of selection, are RAYT associated, and widely distributed. In addition, the apparently general nature of the association between REPINs and RAYTs, combined with substantial diversity among the elements themselves, suggests that the diversity of REPINs (REPs) and RAYTs is a consequence of longstanding co-evolution between RAYTs and their respective REPINs.

The case for REPINs as widely distributed replicative entities is strong, but there remains much to be discovered, particularly regarding the mechanism of transposition, and the relationship between REPINs and RAYTs. A further unknown is the evolution of the entities themselves. One possibility is that REPINs are derived from the imperfect palindromic (REP) sequences flanking an ancestral IS*200*-like element – thus becoming non-autonomous elements [4] – but with a twist. Whereas non-autonomous elements exploit the transposase of extant transposons, the transposons they parasitize remain capable of autonomous replication. In contrast, RAYTs appear to be incapable of self-mobilization and exist as single copy entities (in those genomes harboring more than a single RAYT each RAYT is distinctive and present as just a single copy). This suggests that REPINs evolved a means of parasitizing an IS*200*-like ancestor that not only caused divergence of RAYTs from an IS*200*-like precursor, but did so in such a way as to enslave the RAYT. Just what keeps this association from extinction is among the more intriguing questions for future research, but suggests the existence of either an addiction system that ensures death of any cell that loses RAYT functionality, or a functional role for the RAYT in cell physiology that is somehow linked to REP function.

Finally, our evolutionary approach to the analysis of short repeats and discovery of REPINs and their associated RAYTs may prove useful for elucidating the origins of different kinds of short, repetitive, interspersed palindromic sequences such as NEMISs [32], ERICs [31] and small dispersed repeats (SDR) [43]. Indeed, REPINs themselves could conceivably constitute the building blocks for a range of more complex repetitive structures. For example, REPINs that incorporate DNA beneficial to a host bacterium are likely to have an advantage over standard REPINs. In this regard it is possible that CRISPRs [24] and related mosaic entities are derived from REPIN-like elements.

Methods

Generation of randomized genomes

100 genomes with the same dinucleotide content of the leading/lagging strand and length as the genome of *P. fluorescens* SBW25 were generated by randomly choosing nucleotides according to their occurrence probability based on the preceding nucleotide. To account for dinucleotide skew in the leading or lagging strand of the SBW25 genome, the dinucleotide content of the top strand was determined for the first half of the genome and of the bottom strand for the second half of the genome [22].

Frequency determination of most abundant oligonucleotides

Sequence frequencies for all oligonucleotides of length 10 to 20 were determined using a sliding window with a step size of one for leading and lagging strand separately. The most abundant oligonucleotide for each sequence length was determined. This analysis was conducted for randomly generated genomes as well as for *P. fluorescens* SBW25 and Pf0-1.

Grouping of highly abundant oligonucleotides in SBW25

All oligonucleotides of the chosen sequence length that occur more often in SBW25 than in Pf0-1 were ordered into groups using the following algorithm: 1, Select the most abundant 16-mer from the list of 16-mers that occur more frequently than the most abundant 16-mer in Pf0-1; 2, interrogate the SBW25 genome; 3, extract all occurrences including 20 bp of flanking DNA; 4, concatenate, separating each sequence by a vertical bar (a symbol that is not part of the genomic alphabet); 5, search all remaining 16-mers from the list against the generated string; 6, remove from the list of 16-mers all those sequences found within the generated string and place into the same group as the query; 7, repeat until the list of 16-mers is empty (Figure S3).

Extending REP sequence groups and identifying the frequency of false positives

The genome was searched for related elements by introducing base pair substitutions into the most abundant sequence of each group to a maximum of four. The newly generated sequences, as well as the most abundant sequence of each group, were then used to interrogate the genome and the number of occurrences was counted. In order to determine the false positive rate, a simulation program was written to determine the number of sequences found in randomly generated extragenic space (with the same dinucleotide content).

Distribution simulation

In order to produce a null model against which the observed next-neighbor distances could be compared, 1,053 segments of length 16 were randomly assigned to the extragenic space of SBW25. The simulation was repeated 10,000 times and for each simulation the distances to neighboring segments were determined. Additionally, the formation of clusters by GI, GII and GIII sequences with up to two mismatches (1,422 sequences) was measured. A cluster of REP sequences was defined as a group of REP sequences where each REP sequence has two neighboring REP sequences within the group that are separated by less than 400 bp (the next-neighbor distances showed no significant deviations from randomly expected distances above 400 bp) and a maximum of two REP elements that have only one neighbor within the group which is separated by less than 400 bp.

The same method was applied when distributing doublets randomly over the genome. Instead of 1,422 16 bp long segments, 560×71 bp and 560×110 bp long segments respectively, were randomly assigned. The number of REP doublets was determined by only counting doublets and clusters of doublets. For clusters that contain an odd number of REP sequences, only the even proportion was counted, thus excluding singlets.

Singlet decay

To compare the rate of decay between REP singlets and REP sequences that are part of clusters, REP sequences were divided into their respective groups and then subdivided depending on whether they are found in clusters, or as singlets. In order to include related sequences, the 16-mers were allowed to vary at up to two positions. Since GI 16-mers differ from GII and GIII 16-mers by only two nucleotides, GII and GIII sequences also had to have two group-specific bases (GII: 2T, 6C; GIII: 6A, 13T).

The significance of the singlet decay data was tested using a permutation test. Nine different REP sequence pools were created. Three sequence pools for each sequence group, one of which contained REP singlets, one REP doublets and one greater REP cluster sequences. Two sequences were randomly drawn without replacement from a specific sequence pool and their pairwise identity (the number of sites that are identical between the two sequences divided by the total number of sites) was calculated. This procedure was repeated until the sequence pool was empty. The whole process was repeated 100,000 times for each sequence pool, resulting in the calculation of 100,000 average pairwise identities (mean). For GI sequences the maximum mean calculated for REP singlets never exceeded the minimum mean for REP sequences arranged as doublets. For GII and GIII sequences the maximum mean of REP singlets did exceed the minimum mean of REP sequences from doublets when more than 1,000 means were produced, hence the lower significance of 1e-8. Additionally, for GI and GIII sequences the maximum mean for singlets also never exceeds the minimum mean for clusters (*P*-value 1e-10). The average of the calculated means and the standard deviation are displayed in Figure 3.

REP sequence selection in other genomes

Since REP sequences have been shown to be associated with RAYT genes [23], we looked for 16-mers that were repetitive, extragenic and palindromic in the non-coding DNA flanking RAYT genes. The most frequent 16-mers found within the flanking DNA were also part of or contained a palindrome and were found predominantly in extragenic space, thereby fulfilling all REP sequence prerequisites (Table S2). These 16-mers were then used for a subsequent cluster analysis (flanking clade I RAYTs) or a sample DNA secondary structure prediction (flanking clade II RAYTs).

Bioinformatics and phylogenies

Blast searches were performed using NCBI Blast [44]. The genome was browsed using Artemis [45]. Inverted repeats were identified using Repeat Finder [46]. The multiple alignments in Figure 4 were displayed with Geneious [47] (due to the perfectly conserved distances between the 16-mers, the sequences were aligned after extraction from the genome, no alignment method was needed). DNA secondary structures were predicted using the mfold web server [48]. The RAYT phylogenetic tree was based on a translation alignment (ClustalW2 [49]) as implemented within Geneious [47]. The tree was constructed using a neighbor-joining [50] bootstrap analysis (1000 replicates) also embedded in Geneious.

Genomes used in our analysis

Pseudomonas fluorescens SBW25 (NC_012660.1) [22]
Pseudomonas fluorescens Pf0-1 (NC_007492.2) [22]
Pseudomonas fluorescens Pf-5 (NC_004129.6) [51]
Pseudomonas syringae phaseolicola 1448A (NC_005773.3) [52]
Pseudomonas syringae syringae B728a (NC_007005.1) [53]
Pseudomonas syringae tomato DC3000 (NC_004578.1) [54]
Pseudomonas entomophila L48 (NC_008027.1) [55]
Pseudomonas putida W619 (NC_010501.1)
Pseudomonas putida KT2440 (NC_002947.3) [56]
Pseudomonas putida F1 (NC_009512.1)
Pseudomonas putida GB-1 (NC_010322.1)
Pseudomonas aeruginosa PAO1 (NC_002516.2) [57]
Pseudomonas aeruginosa PA7 (NC_009656.1) [58]
Pseudomonas aeruginosa LESB58 (NC_011770.1) [59]
Pseudomonas mendocina ymp (NC_009439.1)
Pseudomonas stutzeri A1501 (NC_009434.1) [60]
Salmonella enterica serovar Paratyphi *A* AKU_12601 (NC_011147.1) [61]
Escherichia coli K-12 DH10B (NC_010473.1) [62]
Thioalkalivibrio sp HL-EbGR7 (NC_011901.1)
Nostoc punctiforme PCC 73102 (NC_010628.1)

Population sequencing

Pure genomic DNA was isolated from a single SBW25 colony using a combination of chloroform, CTAB and column (Qiagen DNeasy Blood & Tissue Kit) purification techniques. The genomic DNA was sheared to ~400 bp and 76 bp paired-end were sequenced on two channels of an Illumina GA-II flowcell using standard protocols. Raw data were filtered to generate a set of sequences no less than 36 bp in length. After mapping short reads to the SBW25 genome using the Mosaik software suite (http://bioinformatics.bc.edu/marthlab/Mosaik), reads that could not be mapped were screened for REPIN excisions. The screening was accomplished in two steps: 1, for each REPIN present in the SBW25 genome 12 bp of the 5′ and 3′ flanking sequences were extracted; 2, since all reads are shorter than 76 bp, none of the extracted flanking sequences should occur within one read, hence reads containing both 5′ and 3′ REPIN flanking sequences contain an excision. Details of the sequences from which REPINs were excised are given in Figure S6.

Testing for excision of REP singlets

In order to identify excisions of short palindromic sequences it was necessary to define a seed sequence. The GI and GII sequences described above do not overlap the palindromic region and hence are not suitable for this purpose (Table 1). We therefore used an 18-mer containing the palindrome of the GI REP as the seed sequence (GGGGGCTTGCCCCCTCCC). From this seed sequence we generated a set of 18-mers with up to five mismatches. These sequences matched a total of 1376 positions in the SBW25. This set of 1376 sequences encompassed all three GI, GII and GIII REP sequence groups and their relatives. In addition, to allow for the possibility of inexact excisions of palindromes, we allowed the excision to include three additional base pairs on each side of the seed sequence. Armed with this set of sequences we interrogated the ~56 million Illumina-generated sequence reads for evidence of excision events.

Supporting Information

Figure S1 Number of different oligonucleotides in the genome of *P. fluorescens* SBW25 that occur more often than the most frequent oligonucleotides from randomly assembled genomes.
(PDF)

Figure S2 Ratio between the most abundant oligonucleotides from SBW25 and Pf0-1.
(PDF)

Figure S3 Flowchart for grouping over-represented 16-mers. The algorithm sorts all 16-mers that occur more frequently in SBW25 than the most abundant 16-mer in Pf0-1 into groups.
(PDF)

Figure S4 Alignments of the most abundant sequence groups in SBW25. GI sequences are shown in (*A*), GII sequences in (*B*) and GIII sequences in (*C*). The consensus sequence contains the respective palindromic cores (framed in red). Numbers to the left of the alignment denote the frequency of the respective 16-mer (e.g. the first 16-mer in (*A*) GGGCTTGCTCCCGATG occurs 57 times). Colored nucleotides within the alignment denote differences to the consensus sequence.
(PDF)

Figure S5 Process of REP sequence cluster determination. REP sequences are blue boxes. Red arrows indicate a sequence length of 400 bp. The algorithm starts with the position of the first REP sequence (a) and adds it to cluster 1. It then checks the distance to the next REP sequence. The distance to REP sequence (b) is less than 400 bp, hence, the size of cluster 1 increases by one. The distance from (b) to the next REP sequence (c) is greater than 400 bp, therefore, the final size of cluster 1 is two and a new cluster of size one is created called cluster 2. The distance from REP sequence (c) to the next REP sequence is greater than 400 bp; hence, cluster 2 is closed.
(PDF)

Figure S6 Excision events detected in Illumina sequencing data. (*A*) Shows fastq formatted raw Illumina sequences for the excision events and their corresponding paired ends or 'mates'. Quality scores are the last line of each fastq entry. (*B*) In all cases Read 1 matches to a position close to the corresponding Read 2 as expected for paired end reads. The alignments show the match between the sequence reads (second line in the alignment) and the SBW25 genome (first line in the alignment). Colored nucleotides show differences between genome and sequence read. Secondary structure predictions of the excised sequences are shown on the right. For the fourth excision a total of 200 sequence reads were found showing the same event, indicating that the entire REPIN was excised from the genome.
(PDF)

Figure S7 Schematic representation of a typical tandemly repeated REPIN cluster. The cluster comprises two tandem repeat units. Each unit consists of a 5′ flanking sequence (f1) followed by a REPIN and ends with a second shorter flanking sequence (f2). The two units are usually separated by a short stretch of DNA that is not repeated.
(PDF)

Figure S8 Approximate positions of the tandem repeat clusters in the genome of SBW25. The tandem repeats are formed by sequences from GI and GIII. The gray and black arrows indicate different module lengths.
(PDF)

Figure S9 RAYT neighbor joining tree. Two distinct phylogenetic groups are present (Clade I and Clade II). The tree is based on a translated nucleotide alignment. The first part of the branch tip description denotes the gene name and the second part the name of the host organism.
(PDF)

Figure S10 REPIN secondary structures found in different genomes predicted by the mfold web server (http://mfold.rna.albany.edu/). Red bars show palindromic parts of the structure. The yellow box indicates the most abundant 16-mer found in the non-coding flanking DNA of the respective RAYT. The GI consensus sequence from *Pseudomonas fluorescens* SBW25 is the only REPIN shown from RAYT clade I (Figure 4), all other REPINs are associated to RAYTs from clade II.
(PDF)

Figure S11 Two different REPIN folds and their potential susceptibility for transposition by a RAYT dimer. According to our hypothesis the more stable hairpin structure formed by REPINs (left) is unlikely to be recognized by RAYTs and may be a mechanism to reduce the frequency of transposition within the genome. In contrast, the less stable "clover" configuration (right) is likely to be recognized in an *IS200* like manner and may lead to the excision of an asymmetric transposition intermediate.
(PDF)

Table S1 Dinucleotide frequencies in *P. fluorescens* Pf0-1 and SBW25.
(PDF)

Table S2 Short sequence composition of the non-coding DNA flanking RAYTs.
(PDF)

Table S3 Details concerning the analysis of REP sequences in other bacterial genomes.
(XLSX)

Table S4 Correlation between REPINs and repeat families previously detected in SBW25.
(PDF)

Acknowledgments

We thank Bernhard Haubold, Heather Hendrickson, Jenna Gallie, and Ben Kerr for valuable feedback on drafts of the manuscript; John Roth, Justin O'Sullivan, and Allen Rodrigo for stimulating discussion. We also thank David Guttman and three anonymous referees for critical appraisals.

Author Contributions

Conceived and designed the experiments: FB PBR. Performed the experiments: FB. Analyzed the data: FB. Wrote the paper: FB PBR.

References

1. Gregory TR (2005) The Evolution of the Genome. Burlington, Massachusetts: Elsevier Academic Press.
2. Cordaux R, Batzer MA (2009) The impact of retrotransposons on human genome evolution. Nat Rev Genet 10: 691–703. doi: 10.1038/nrg2640.
3. Charlesworth B, Sniegowski P, Stephan W (1994) The evolutionary dynamics of repetitive DNA in eukaryotes. Nature 371: 215–220. doi: 10.1038/371215a0.
4. Burt A, Trivers R (2006) Genes in Conflict: The Biology of Selfish Genetic Elements. CambridgeMassachusetts: Belknap Press of Harvard University Press.
5. Levinson G, Gutman GA (1987) Slipped-strand mispairing: a major mechanism for DNA sequence evolution. Mol Biol Evol 4: 203–221.
6. Myers RS, Stahl FW (1994) Chi and the RecBC D enzyme of *Escherichia coli*. Annu Rev Genet 28: 49–70. doi: 10.1146/annurev.ge.28.120194.000405.
7. Bigot S, Saleh OA, Lesterlin C, Pages C, Karoui ME, et al. (2005) KOPS: DNA motifs that control *E. coli* chromosome segregation by orienting the FtsK translocase. EMBO J 24: 3770–3780. doi: 10.1038/sj.emboj.7600835.
8. Hendrickson H, Lawrence JG (2006) Selection for chromosome architecture in bacteria. J Mol Evol 62: 615–629. doi: 10.1007/s00239-005-0192-2.
9. Treangen TJ, Abraham AL, Touchon M, Rocha EPC (2009) Genesis, effects and fates of repeats in prokaryotic genomes. FEMS Microbiol Rev 33: 539–571.
10. de Bruijn FJ (1992) Use of repetitive (repetitive extragenic palindromic and enterobacterial repetitive intergeneric consensus) sequences and the polymerase chain reaction to fingerprint the genomes of *Rhizobium meliloti* isolates and other soil bacteria. Appl Environ Microbiol 58: 2180–2187.
11. Versalovic J, Koeuth T, Lupski JR (1991) Distribution of repetitive DNA sequences in eubacteria and application to fingerprinting of bacterial genomes. Nucleic Acids Res 19: 6823–6831.
12. Woods CR, Versalovic J, Koeuth T, Lupski JR (1992) Analysis of relationships among isolates of *Citrobacter diversus* by using DNA fingerprints generated by repetitive sequence-based primers in the polymerase chain reaction. J Clin Microbiol 30: 2921–2929.

13. Higgins CF, Ames GF, Barnes WM, Clement JM, Hofnung M (1982) A novel intercistronic regulatory element of prokaryotic operons. Nature 298: 760–762.
14. Gilson E, Saurin W, Perrin D, Bachellier S, Hofnung M (1991) Palindromic units are part of a new bacterial interspersed mosaic element (BIME). Nucleic Acids Res 19: 1375–1383.
15. Lupski JR, Weinstock GM (1992) Short, interspersed repetitive DNA sequences in prokaryotic genomes. J Bacteriol 174: 4525–4529.
16. Wilson LA, Sharp PM (2006) Enterobacterial repetitive intergenic consensus (ERIC) sequences in *Escherichia coli*: Evolution and implications for ERIC-PCR. Mol Biol Evol 23: 1156–1168. doi: 10.1093/molbev/msj125.
17. Bachellier S, Clément JM, Hofnung M (1999) Short palindromic repetitive DNA elements in enterobacteria: a survey. Res Microbiol 150: 627–639.
18. Gilson E, Clément JM, Brutlag D, Hofnung M (1984) A family of dispersed repetitive extragenic palindromic DNA sequences in *E. coli*. EMBO J 3: 1417–1421.
19. Stern M, Ames G, Smith N, Robinson E, Higgins C (1984) Repetitive extragenic palindromic sequences: a major component of the bacterial genome. Cell 37: 1015–1026.
20. Aranda-Olmedo I, Tobes R, Manzanera M, Ramos J, Marques S (2002) Species-specific repetitive extragenic palindromic (REP) sequences in *Pseudomonas putida*. Nucleic Acids Research 30: 1826–1833.
21. Tobes R, Pareja E (2005) Repetitive extragenic palindromic sequences in the *Pseudomonas syringae* pv. *tomato* DC3000 genome: extragenic signals for genome reannotation. Res Microbiol 156: 424–433. doi: 10.1016/j.resmic.2004.10.014.
22. Silby M, Cerdeno-Tarraga A, Vernikos G, Giddens S, Jackson R, et al. (2009) Genomic and genetic analyses of diversity and plant interactions of *Pseudomonas fluorescens*. Genome Biol 10: R51. doi: 10.1186/gb-2009-10-5-r51.
23. Nunvar J, Huckova T, Licha I (2010) Identification and characterization of repetitive extragenic palindromes (REP)-associated tyrosine transposases: implications for REP evolution and dynamics in bacterial genomes. BMC Genomics 11: 44. doi: 10.1186/1471-2164-11-44.
24. Ishino Y, Shinagawa H, Makino K, Amemura M, Nakata A (1987) Nucleotide sequence of the *iap* gene, responsible for alkaline phosphatase isozyme conversion in *Escherichia coli*, and identification of the gene product. J Bacteriol 169: 5429–5433.
25. Horvath P, Barrangou R (2010) CRISPR/Cas, the immune system of bacteria and archaea. Science 327: 167–170. doi: 10.1126/science.1179555.
26. Barrangou R, Fremaux C, Deveau H, Richards M, Boyaval P, et al. (2007) CRISPR provides acquired resistance against viruses in prokaryotes. Science 315: 1709–1712. doi: 10.1126/science.1138140.
27. Wessler SR, Bureau TE, White SE (1995) LTR-retrotransposons and MITEs: important players in the evolution of plant genomes. Curr Opin Genet Dev 5: 814–821.
28. Nakazaki T, Okumoto Y, Horibata A, Yamahira S, Teraishi M, et al. (2003) Mobilization of a transposon in the rice genome. Nature 421: 170–172. doi: 10.1038/nature01219.
29. Delihas N (2008) Small mobile sequences in bacteria display diverse structure/function motifs. Mol Microbiol 67: 475–481. doi: 10.1111/j.1365-2958.2007.06068.x.
30. Oggioni MR, Claverys JP (1999) Repeated extragenic sequences in prokaryotic genomes: a proposal for the origin and dynamics of the RUP element in *Streptococcus pneumoniae*. Microbiol 145: 2647–2653.
31. Hulton CS, Higgins CF, Sharp PM (1991) ERIC sequences: a novel family of repetitive elements in the genomes of *Escherichia coli*, *Salmonella typhimurium* and other enterobacteria. Mol Microbiol 5: 825–834.
32. Correia FF, Inouye S, Inouye M (1988) A family of small repeated elements with some transposon-like properties in the genome of *Neisseria gonorrhoeae*. J Biol Chem 263: 12194–12198.
33. Higgins CF, McLaren RS, Newbury SF (1988) Repetitive extragenic palindromic sequences, mRNA stability and gene expression: evolution by gene conversion? A review. Gene 72: 3–14.
34. Haubold B, Wiehe T (2006) Introduction to Computational Biology: An Evolutionary Approach. Basel: Birkhauser.
35. Csurös M, Noé L, Kucherov G (2007) Reconsidering the significance of genomic word frequencies. Trends Genet 23: 543–546. doi: 10.1016/j.tig.2007.07.008.
36. Espéli O, Moulin L, Boccard F (2001) Transcription attenuation associated with bacterial repetitive extragenic BIME elements. J Mol Biol 314: 375–386. doi: 10.1006/jmbi.2001.5150.
37. Rocco F, Gregorio ED, Nocera PPD (2010) A giant family of short palindromic sequences in *Stenotrophomonas maltophilia*. FEMS Microbiol Lett 308: 185–192. doi: 10.1111/j.1574-6968.2010.02010.x.
38. Ton-Hoang B, Guynet C, Ronning DR, Cointin-Marty B, Dyda F, et al. (2005) Transposition of ISHp*608*, member of an unusual family of bacterial insertion sequences. EMBO J 24: 3325–3338. doi: 10.1038/sj.emboj.7600787.
39. Barabas O, Ronning DR, Guynet C, Hickman AB, Ton-Hoang B, et al. (2008) Mechanism of IS*200*/IS*605* family DNA transposases: activation and transposon-directed target site selection. Cell 132: 208–220. doi: 10.1016/j.cell.2007.12.029.
40. Ton-Hoang B, Pasternak C, Siguier P, Guynet C, Hickman AB, et al. (2010) Single-stranded DNA transposition is coupled to host replication. Cell 142: 398–408. doi: 10.1016/j.cell.2010.06.034.
41. Clément JM, Wilde C, Bachellier S, Lambert P, Hofnung M (1999) IS*1397* is active for transposition into the chromosome of *Escherichia coli* K-12 and inserts specifically into palindromic units of bacterial interspersed mosaic elements. J Bacteriol 181: 6929–6936.
42. Tobes R, Pareja E (2006) Bacterial repetitive extragenic palindromic sequences are DNA targets for Insertion Sequence elements. BMC Genomics 7: 62.
43. Elhai J, Kato M, Cousins S, Lindblad P, Costa JL (2008) Very small mobile repeated elements in cyanobacterial genomes. Genome Res 18: 1484–1499. doi: 10.1101/gr.074336.107.
44. Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ (1990) Basic local alignment search tool. J Mol Biol 215: 403–410. doi: 10.1006/jmbi.1990.9999.
45. Rutherford K, Parkhill J, Crook J, Horsnell T, Rice P, et al. (2000) Artemis: sequence visualization and annotation. Bioinformatics 16: 944–945.
46. Warburton PE, Giordano J, Cheung F, Gelfand Y, Benson G (2004) Inverted repeat structure of the human genome: the X-chromosome contains a preponderance of large, highly homologous inverted repeats that contain testes genes. Genome Res 14: 1861–1869. doi: 10.1101/gr.2542904.
47. Drummond A, Ashton B, Cheung M, Heled J, Kearse M, et al. (2009) Geneious v4.8. Available from http://www.geneious.com/.
48. Zuker M (2003) Mfold web server for nucleic acid folding and hybridization prediction. Nucleic Acids Res 31: 3406–3415.
49. Chenna R, Sugawara H, Koike T, Lopez R, Gibson TJ, et al. (2003) Multiple sequence alignment with the Clustal series of programs. Nucleic Acids Res 31: 3497–3500.
50. Saitou N, Nei M (1987) The neighbor-joining method: a new method for reconstructing phylogenetic trees. Mol Biol Evol 4: 406–425.
51. Paulsen IT, Press CM, Ravel J, Kobayashi DY, Myers GSA, et al. (2005) Complete genome sequence of the plant commensal *Pseudomonas fluorescens* Pf-5. Nat Biotechnol 23: 873–878. doi: 10.1038/nbt1110.
52. Joardar V, Lindeberg M, Jackson RW, Selengut J, Dodson R, et al. (2005) Whole-genome sequence analysis of *Pseudomonas syringae* pv. *phaseolicola* 1448A reveals divergence among pathovars in genes involved in virulence and transposition. J Bacteriol 187: 6488–6498. doi: 10.1128/JB.187.18.6488-6498.2005.
53. Feil H, Feil WS, Chain P, Larimer F, DiBartolo G, et al. (2005) Comparison of the complete genome sequences of *Pseudomonas syringae* pv. *syringae* B728a and pv. *tomato* DC3000. Proc Natl Acad Sci U S A 102: 11064–11069. doi: 10.1073/pnas.0504930102.
54. Buell CR, Joardar V, Lindeberg M, Selengut J, Paulsen IT, et al. (2003) The complete genome sequence of the *Arabidopsis* and tomato pathogen *Pseudomonas syringae* pv. *tomato* DC3000. Proc Natl Acad Sci U S A 100: 10181–10186. doi: 10.1073/pnas.1731982100.
55. Vodovar N, Vallenet D, Cruveiller S, Rouy Z, Barbe V, et al. (2006) Complete genome sequence of the entomopathogenic and metabolically versatile soil bacterium *Pseudomonas entomophila*. Nat Biotechnol 24: 673–679. doi: 10.1038/nbt1212.
56. Nelson KE, Weinel C, Paulsen IT, Dodson RJ, Hilbert H, et al. (2002) Complete genome sequence and comparative analysis of the metabolically versatile *Pseudomonas putida* KT2440. Environ Microbiol 4: 799–808.
57. Stover CK, Pham XQ, Erwin AL, Mizoguchi SD, Warrener P, et al. (2000) Complete genome sequence of *Pseudomonas aeruginosa* PAO1, an opportunistic pathogen. Nature 406: 959–964. doi: 10.1038/35023079.
58. Roy PH, Tetu SG, Larouche A, Elbourne L, Tremblay S, et al. (2010) Complete genome sequence of the multiresistant taxonomic outlier *Pseudomonas aeruginosa* PA7. PLoS ONE 5: e8842. doi: 10.1371/journal.pone.0008842.
59. Winstanley C, Langille MGI, Fothergill JL, Kukavica-Ibrulj I, Paradis-Bleau C, et al. (2009) Newly introduced genomic prophage islands are critical determinants of in vivo competitiveness in the Liverpool Epidemic Strain of *Pseudomonas aeruginosa*. Genome Res 19: 12–23. doi: 10.1101/gr.086082.108.
60. Yan Y, Yang J, Dou Y, Chen M, Ping S, et al. (2008) Nitrogen fixation island and rhizosphere competence traits in the genome of root-associated *Pseudomonas stutzeri* A1501. Proc Natl Acad Sci U S A 105: 7564–7569. doi: 10.1073/pnas.0801093105.
61. Holt KE, Thomson NR, Wain J, Langridge GC, Hasan R, et al. (2009) Pseudogene accumulation in the evolutionary histories of *Salmonella enterica* serovars Paratyphi A and Typhi. BMC Genomics 10: 36. doi: 10.1186/1471-2164-10-36.
62. Durfee T, Nelson R, Baldwin S, Plunkett G, Burland V, et al. (2008) The complete genome sequence of *Escherichia coli* DH10B: insights into the biology of a laboratory workhorse. J Bacteriol 190: 2597–2606. doi: 10.1128/JB.01695-07.