Amplikyzer: Automated methylation analysis of amplicons from bisulfite flowgram sequencing

Sven Rahmann [1*] Jasmin Beygo [2] Deniz Kanber [2]
Marcel Martin [1] Bernhard Horsthemke [2] Karin Buiting [2]

[1] Genome Informatics, Institute of Human Genetics,
University Hospital Essen, University of Duisburg-Essen
Hufelandstr. 55, 45147 Essen, Germany

[2] Institute of Human Genetics,
University Hospital Essen, University of Duisburg-Essen
Hufelandstr. 55, 45147 Essen, Germany

The Roche 454 GS Junior sequencing platform allows locus-specific DNA methylation analysis using deep bisulfite amplicon sequencing. However, bisulfite-converted DNA reads may contain long T homopolymers, and the main sources of errors on pyrosequencing platforms are homopolymer over- and undercalls. Furthermore, existing tools do not always meet the analysis requirements for complex assay designs with multiple regions of interest (ROIs) from multiple samples.

We have developed the *amplikyzer* software package to address the above challenges. It directly aligns the intensity sequences from standard flowgram files (SFF format) to given amplicon reference sequences, without converting to nucleotide FASTA format first, avoiding information loss by rounding flow intensities, and taking special measures to correctly process long homopolymers. It offers a variety of options to analyze complex multiplexed samples with several regions of interest and outputs useful statistics and publication-quality analysis plots without mandatory manual interaction. This allows our software to be used as part of automated pipelines as well as interactively.

The underlying analysis algorithms, using a novel hybrid flowgram-DNA sequence representation are described in detail. We also discuss configuration options and use cases of our open source *amplikyzer* software and present exemplary results. The software, including required libraries, is available at `https://bitbucket.org/svenrahmann/amplikyzer/downloads`. Contact: `Sven.Rahmann[at]uni-due.de`

*to whom correspondence should be addressed

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

1 Introduction

Motivation Locus-specific DNA methylation analysis is widely used in the field of genomic imprinting, related disorders and cancer research. To determine and quantify the methylation state at single nucleotide resolution, locus-specific bisulfite sequencing by Sanger sequencing of DNA clones was used as a gold-standard technique over a long time. However, this method was time consuming, so only a few clones were typically analysed, and biases in cloning might have led to a skewed ratio of methylated to unmethylated molecules. Nowadays massively parallel sequencing methods, in particular 454 sequencing, can be used for high-throughput sequencing of bisulfite PCR amplicons (Taylor et al., 2007; Beygo et al., 2013b,a; Berland et al., 2013). For example, using the Roche 454 GS Junior system, more than 100 000 sequence reads can easily be obtained in a single sequencing run without subcloning, thus eliminating cloning bias. This method allows to obtain highly quantitative DNA methylation patterns and to detect even slight methylation changes.

Currently, the International Human Epigenome Consortium (IHEC) is undertaking a multinational study to establish genome-wide basepair-level methylation maps, which are expected to yield new insights into human methylation patterns[1]. At present, more than 10 countries are a member of IHEC. In Germany, for example, DEEP (DEutsches Epigenom Programm[2]) will produce 70 reference epigenomes of selected human and murine cells involved in metabolic and inflammatory diseases. The validation of differentially methylated regions and more detailed studies require deep bisulfite amplicon sequencing from specifically selected loci (*regions of interest*, ROIs) with high coverage.

The Roche 454 sequencing technology is an appropriate one for deep amplicon sequencing, as it provides read lengths of up to 700 bp on the Roche 454 GS Junior device. We previously found, for instance, that several CpG islands (CGIs) on the X-chromosome are incompletely methylated, while on the autosomes, most CGIs are either completely or not at all methylated (Zeschnigk et al., 2009).

Problem Statement The *bisulfite-sequenced amplicon methylation analysis problem*, as considered in this work, consists of the following inputs: a set of bisulfite sequence reads, a set of reference ROIs with primers, a set of distinct samples given by multiplex identifier sequences (MIDs), and forward and reverse tag sequences. We specifically assume that a flowgram-based sequencing technology is used (i.e., Roche 454 or Ion Torrent) that produces reads as flowgrams (see Section 2.1) and outputs SFF files (standard flowgram format).

The task at hand is to identify, for each read, the corresponding MID and ROI, and to analyze the methylation status of each CpG in the ROI of the read. This is possible because sodium bisulfite treatment converts each unmethylated `C` into `T`, but leaves methylated `C`s unmodified. This means that by examining each nucleotide in a read that corresponds to the `C` of a `CG` dinucleotide in the reference, the methylation state can be inferred.

From such a detailed nucleotide-level analysis, statistical summaries (methylation rates of CpGs or entire ROIs) are to be generated and visualized.

Related Work *BiQ Analyzer* (Bock et al., 2005) and its successor *BiQ Analyzer HT* (Lutsik et al., 2011) are popular tools for bisulfite-sequenced amplicon analysis and widely used. *BiQ*

[1] `http://www.ihec-epigenomes.org/welcome/`
[2] `https://deutsches-epigenom-programm.de`

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

Analyzer HT offers a wide range of analysis functions, an interactive user interface and comprehensive visualization capabilities. However, the task of identifying MID and ROI for each read is not solved by BiQ Analyzer, but left to other SFF-capable read splitting and mapping tools, such as Roche's proprietary software, or any other commercial or free software able to separate reads into FASTA files according to barcode sequences (MIDs). Another tool for bisulfite amplicon analysis, also for repetitive ROI sequences, is *BISMA* (Rohde et al., 2010), which is also sequence-based (instead of flowgram-based). As it uses *CLUSTALW* (Larkin et al., 2007) for alignments, it does not allow high-throughput analysis with tens of thousands of reads. The *BDPC web server* (Rohde et al., 2008) takes *BISMA* or *BiQ Analyzer* results as input and provides additional aggregation and presentation capabilities.

Bismark (Krueger and Andrews, 2011) and *MethylCoder* (Pedersen et al., 2011) are combined bisulfite-aware read mappers and methylation callers. They rely on sequence-based read mappers (e.g, *Bowtie2* (Langmead and Salzberg, 2012)) and explicit simulated bisulfite conversion of the reference genome. As they work on the sequence level (rather than with flowgrams), they are more suitable for Illumina sequence data and exome-wide or genome-wide methylation analysis.

To summarize, *BiQ Analyzer HT* requires pre-processed FASTA sequence files separately for each MID; all tools provide sequence-based analysis. In contrast, a design goal of *amplikyzer* is to provide a unified analysis starting from raw flowgrams, while at the same time offering more customization options to the user.

Our Contributions Our *amplikyzer* software is an alternative to existing solutions with a number of distinguishing features that, to our knowledge, are not implemented in any of the tools described above. Most importantly, our analysis starts directly from the flowgrams in the raw SFF file without converting to FASTA first (see Section 2.1). No pre-processing of the SFF file is necessary; all steps from the raw data to publication-quality analysis figures are provided by our methods and software. Noteworthy features of *amplikyzer* include the separation of sequence reads based on single nucleotide polymorphisms (SNPs) present in the corresponding amplicon/ROI with the possibility to present the methylation data in separate plots for each allele. Furthermore, comparison plots enable the user to depict the methylation for separated alleles from one or more samples at a time. Different samples can be sorted automatically by methylation level or manually by specifying a MID order.

The *amplikyzer* software was designed to work within automated high-throughput workflows. This means that once the input data (reads, primers, ROI sequences, MIDs, etc.) and analysis parameters are specified in persistent editable configuration files, no further user interaction is necessary, and all analysis results are produced as textual reports and publication-quality image files in an output directory. This enables *amplikyzer* with all of its customization options to be part of larger workflows, such as provided by *Snakemake* (Köster and Rahmann, 2012). In the same spirit, *amplikyzer* is a standalone desktop application instead of a client/server system. An optional graphical user interface (GUI) is provided to specify the analysis parameters interactively if desired by the user.

Organization of this Article In Section 2, we summarize background knowledge on bisulfite sequencing experiments using the Roche 454 platform. This knowledge is required to understand the rationale behind *amplikyzer*'s algorithmic analysis approach, which is described in Section 3. A description of the software is given in Section 4. Section 5 describes an exemplary analysis and its results in comparison to other tools. A brief discussion concludes the paper.

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

2 Background

We provide background knowledge on flowgrams and SFF files in Section 2.1, discuss necessary filter adjustments to the Roche 454 GS Junior sequencer software for bisulfite sequencing in Section 2.2, and describe a typical library preparation protocol and the resulting structure of the sequence reads in Section 2.3.

2.1 Flowgram Analysis Challenges

The output of the Roche 454 sequencing technology (and similarly, of the Ion Torrent technology) does not consist of DNA sequences, but of flowgrams. A *flowgram* is a numeric sequence describing (normalized) measured light intensities during each sequencing cycle. The Roche 454 GS Junior system uses a fixed repeated nucleotide interrogation sequence (`TACG`) for up to two hundred cycles up to 800 flows. (Since the end of 2012, a more complex alternative interrogation sequence called flow pattern B can be used that leads to longer reads.) Each flow of a single nucleotide results in a measured intensity, approximately proportional to the number of nucleotides in the current homopolymer to be sequenced, where an intensity value of 1.0 corresponds to a single nucleotide. For example, assuming the interrogation sequence `TACG`, the flowgram $(1.02, 2.10, 0.15, 2.20, 3.07, 0.03, 2.58, 0.19)$ consisting of two full cycles (or eight flows) is also written as $\mathtt{T}^{1.02}\mathtt{A}^{2.10}\mathtt{C}^{0.15}\mathtt{G}^{2.20}\mathtt{T}^{3.07}\mathtt{A}^{0.03}\mathtt{C}^{2.58}\mathtt{G}^{0.19}$. By rounding, it can be translated ("base-called") into the DNA sequence `TAAGGTTTCCC`.

An SFF file output by the sequencer software contains both the raw flowgrams and the base-called DNA sequence for each read that passed the internal quality filters, together with quality and filtering information. The specific file format is documented at various locations[3].

In the example above, at least the decision for the flow $\mathtt{C}^{2.58}$ is not obvious: Does this intensity refer to `CC` or `CCC`? Theoretically, ambiguous intensities should not exist; in practice, they do occur. The problem becomes more prominent at higher intensities, where saturation effects occur and the measured intensity is not proportional to the number of nucleotides any more. For example, $\mathtt{A}^{5.32}$ might plausibly refer to a homopolymer of 5, 6, or 7 `A`s.

It follows that if the flowgram output of the sequencer is first base-called and only then aligned to a genomic reference sequence, there is a high chance that spurious insertions or deletions will be seen in the alignment because of wrong base calling decisions. To address these issues, we recently proposed an algorithm for directly aligning flowgrams to DNA references (Martin and Rahmann, 2013) without previous base-calling or alternatively converting the reference DNA into flowspace. A prototype implementation is available as the *FlowG* software[4].

In the context of sodium bisulfite sequencing, however, the problem is further aggravated by the fact that bisulfite treatment converts most `C`s into `T`s, such that there will be long `T`-homopolymers in many reads. For example, genomic `ATTCTCCTCGA` would become `ATTTTTTTTGA` (assuming an unmethylated CpG), which may plausibly lead to flows $\mathtt{A}^{1.10}\mathtt{T}^{6.48}\mathtt{G}^{0.92}\mathtt{A}^{1.02}$ (ignoring intermediate zero flows). The base calling step would interpret this as `ATTTTTTGA`, missing two `T`s, and yielding the alignment

```
genomic:  ATTCTCCTCGA
    read:  ATTTTTT--GA
```

where a gap is aligned to the `C` of the CpG dinucleotide, leaving us unable to infer its methylation state from the alignment.

[3] `http://www.ncbi.nlm.nih.gov/Traces/trace.cgi?cmd=show&f=formats&m=doc&s=format#sff`
[4] `http://www.rahmannlab.de/software`

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

The analysis method for *amplikyzer* has been specifically designed to work with flowgrams and avoid the illustrated problems, while still allowing an efficient analysis of a large number of reads.

2.2 Quality Filtering

As the homopolymer problem is well known, the manufacturer's software includes quality filters to alleviate the problem by discarding reads with too many ambiguous intensities. While this reduces the number of available reads, it increases the overall quality of base-called reads. As explained above, for sodium bisulfite sequencing, long homopolymer runs (and thus more ambiguous intensity values) cannot be avoided. The standard filter settings discard a large fraction of the reads, and so the settings must be adjusted to yield more reads (which are consequently harder to analyze using conventional methods). We recommend the following filter adjustments in the sequencer software:

1. setting `doValleyFilterTrimBack` to `True` (instead of its default value `False`). This choice instructs the system only to trim the low-quality end of (and not to entirely discard) reads with many ambiguous intensities.

2. increasing `vfBadFlowThreshold` to 10 from its default value 4. This parameter controls the number of allowed ambiguous flow values before the read is trimmed. (Possibly, even larger values could be attempted.)

2.3 Library Generation and Read Structure

Generation of Locus-Specific Amplicon Libraries To study DNA methylation at a specific locus in different individuals, amplicon libraries are generated using two consecutive PCRs. The first PCR on bisulfite-treated DNA (the ROI) is performed with locus-specific primers that match the bisulfite-treated DNA and contain universal primer tags (in *amplikyzer* abbreviated as FWD, REV) for forward and reverse primers. For the second PCR, primers are used which consist of the universal tags, sample-specific barcode sequences (MIDs), the 454 key sequence and the 454 sequencing system primers. Amplicon fragments are clonally amplified in an emulsion PCR (emPCR), beads carrying DNA are enriched, and the amplicon library is sequenced on the Roche 454 GS Junior system.

Structure of Sequenced Reads If the above library generation protocol is followed, each sequenced read (i.e., flowgram) consists of the following parts, which the *amplikyzer* algorithm needs to recognize in order to correctly classify the read:

1. the 454 key sequence (always `TCAG`), consuming two full `TACG` flow cycles and corresponding to the flow prefix $(1.00, 0.00, 1.00, 0.00, 0.00, 1.00, 0.00, 1.00)$. The key sequence is used to normalize observed flow intensities, such that the observed value of 1.00 on average corresponds to a flow of a single nucleotide.

2. a MID (multiplex ID), used to sequence amplicons from different samples in the same run. Each MID is a unique DNA sequence of length 10 nt.

3. a tag, for which presently only two possibilities exist, *forward* and *reverse*. The presence of the *forward* tag `CTTGCTTCCTGGCACGAG` indicates that the remainder of the read will start with a forward primer, continue with the corresponding region of interest, end with the

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

reverse primer, and that `C` $\mapsto$ `T` substitutions should be expected because of bisulfite treatment. In contrast, the *reverse* tag `CAGGAAACAGCTATGAC` indicates that the read continues with the reverse complements of a reverse primer, of the corresponding region of interest and of the forward primer, and that `G` $\mapsto$ `A` substitutions are expected.

4. locus sequence, consisting of either (f, i, r) or $(\bar{r}, \bar{i}, \bar{f})$, depending on the tag type, where f is the forward primer, i the region of interest (ROI), and r the reverse primer (as given in $5' \to 3'$ direction on the genomic reference), and $\bar{\cdot}$ denotes reverse complement. The locus sequence is the only read element that is bisulfite-treated.

The other tag and another copy of the MID follow, but we do not consider them during the analysis. Table 1 shows an overview of the read structure.

The *amplikyzer* software can be adapted to other read structures via configuration files; the description in this article, however, assumes the above structure.

3 Algorithms

This section describes the novel algorithmic analysis approach implemented by the *amplikyzer* software. As explained in Section 2.1, when working with flowgram-based data, it is advantageous to work directly with the flow intensities instead of converting to a DNA sequence first, in order to avoid information loss and spurious insertions or deletions in alignments. Methods to directly align flowgrams to DNA have been described previously (Vacic et al., 2008; Lysholm et al., 2011; Martin and Rahmann, 2013). None of them can efficiently make use of string-based indexing techniques and are thus much slower than state-of-the-art read mappers using hashing or an FM index based on the Burrows-Wheeler transform (Li and Durbin, 2009; Langmead and Salzberg, 2012). Therefore we introduce a hybrid representation between flowgram and DNA sequence (Section 3.1). It retains some of the flexibility of flow intensity values, but on the other hand allows to use the established FM index data structure and to construct a novel variant of the above-mentioned read mapping algorithms for rapid mapping of flowgram reads to ROIs (Section 3.2). Finally, we discuss the final alignment of the flowgram to the ROI sequence and subsequent methylation calling (Section 3.3).

The steps described in this subsection are the most time-consuming ones when analyzing an SFF file. The subsequent steps, such as aggregating methylation information and producing textual or graphical reports on methylation states and rates, can be performed quickly and flexibly, given the alignment results. They are discussed in Section 4.

3.1 Hybrid Flowgram-String Representation

We write a single *flow* as B^f, where $B \in \Sigma := \{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}\}$ is the *base* and $f \in \{0.00, 0.01, 0.02, \dots\}$ is the *intensity*. A *flowgram* is a sequence of flows $(B_1^{f_1}, B_2^{f_2}, \dots)$. Frequently, the B_i are a repeated fixed permutation of the DNA alphabet, but this is not necessary.

In order to use string-based indexing schemes and fast mapping algorithms, we convert a flowgram to a string over the extended alphabet $\Omega := \{\mathtt{A}, \mathtt{a}, \mathtt{C}, \mathtt{c}, \mathtt{G}, \mathtt{g}, \mathtt{T}, \mathtt{t}, \mathtt{+}\}$. We write b for the lower-case letter of $B \in \Sigma$ and c^n with integer n for the string representing the n-fold repetition of $c \in \Omega$. We write c^0 for the empty string.

Let $0 \le \mu \le 0.5$ be a parameter that describes how much a flow intensity may deviate from an integer value to be considered an *uncertain flow*; in practice, we use $\mu = 0.2$. That is, if flow intensity $f \in [n - \mu, n + \mu]$ for some integer n, we decide that flow B^f corresponds to the n-fold

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

element	length [bp]	orientation in read	bisulfite treated?	indexed sequence	req. match len L	C
key	4	fwd	no	fwd	4	4
MID	10	fwd	no	fwd	7	7
tag	17–18	fwd	no	fwd	10	12
locus	100+	fwd or rc	yes	fwd($\mathtt{C} \mapsto \mathtt{T}$) $\cup$ rc($\mathtt{G} \mapsto \mathtt{A}$)	30	50

Table 1: Read elements, properties, indexed sequence (fwd: forward; rc: reverse complement) and required exact match lengths to recognize an element; L: required exact match length at some starting position within the relevant part of the read; C: required cumulative match length over the relevant part of the read.

repetition of B exactly, but if $f \in]n+\mu,\ n+1-\mu[$, then we decide that B^f corresponds to n or $n+1$ repetitions of B and write this as $B^n b$ with an additional lower-case ("uncertain") base at the end. Because of saturation effects and limited discrimination power at high intensities, we limit this scheme at a given integer cut-off value N and convert B^f to $B^N\mathtt{+}$ for $f > N+\mu$, where the + indicates an unspecified additional number of Bs.

In other words, given parameters μ and N, the flow B^f is converted to its hybrid representation

$$h(B^f) := \begin{cases} B^n & \text{if } |f-n| \le \mu \text{ and } 0 \le n \le N, \\ B^n b & \text{if } n+\mu < f < n+1-\mu \text{ and } 0 \le n < N, \\ B^N\mathtt{+} & \text{if } f > N+\mu. \end{cases} \quad (1)$$

3.2 Rapid Read Mapping of Hybrid Flowgram-DNA Sequences

Preliminaries We describe how to efficiently find the correct MID and locus/ROI for each read, without generating the alignment yet. Each read is assumed to be a flowgram following the structure described in Section 2.3. For rapid read mapping, in order to quickly identify MID and ROI, each read is converted into the hybrid format, i.e., a string over alphabet Ω, described in Section 3.1.

For each modular part of a read (keys, MIDs, tags, and loci containing forward primer, ROI and reverse primer), we build a separate index data structure similar to an FM index (see below). Currently, there exists only one possible key sequence (TCAG) on the Roche 454 GS Junior, so indexing keys is not strictly necessary. For uniformity of analysis, however, we treat the key similar to the other read elements. The way the index is built differs for each element, according to its possible orientations in the read (forward only, or possibly as reverse complement) and its bisulfite treatment status (true or false). An overview is given in Table 1. The locus sequence may be oriented forward or reverse complementary in the read (according to the observed tag type) and is bisulfite-treated. Therefore, we index two variants of the locus sequence, each over a three-letter alphabet: the forward reference contains only A, G, T, while the reverse complementary reference contains only A, C, T. In order to precisely map the flowgram reads in the hybrid Ω-representation to the index, we use the same homopolymer length cut-off value N as in Section 3.1 and reduce longer homopolymers to length N (in practice, $N = 4$).

For technical reasons, as we use a variant of backward search for read mapping, but want to process the flowgrams from left to right, the *reversed* (not reverse complementary) sequences of the elements given in Table 1 are used. All reversed reference sequences of a structural element are terminated by a special character (traditionally written as $) and then concatenated. A $

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

appearing to the left of another `$` is considered a lexicographically smaller character. As an example, if there are three MIDs `ACGAGTGCGT`, `ACGCTCGACA`, `AGACGCACTC`, then the sequence to be indexed is $s =$ `TGCGTGAGCA$ACAGCTCGCA$CTCACGCAGA$`.

Index Data Structure The index consists of the suffix array `pos`, the longest common prefix array `lcp`, the Burrows-Wheeler transform (BWT) `bwt`, and two auxiliary arrays `less` and `occ` derived from the BWT. For completeness, we briefly review the basic definitions and otherwise refer the reader to Section 2 of the *BWA* article by Li and Durbin (2009) for details. While our indexing data structure is similar, our mapping algorithm has major differences to that of *BWA*, which we point out below.

Let $|s| = n$ be the length of the sequence to be indexed; $s = s[0], \ldots, s[n-1]$. For $0 \leq r < n$, the suffix array element $\texttt{pos}[r]$ is defined as the starting position in s of the r-th lexicographically smallest suffix of s. In other words, `pos` is the permutation of $\{0, \ldots, n-1\}$ that sorts the suffixes of s lexicographically. For $0 < r < n$, we define $\texttt{lcp}[r]$ as the length of the longest common prefix of the (lexicographically adjacent) suffixes starting at positions $\texttt{pos}[r-1]$ and $\texttt{pos}[r]$, whereas $\texttt{lcp}[0]$ is undefined. The BWT of s is defined as $\texttt{bwt}[r] := s[\texttt{pos}[r]-1]$ if $\texttt{pos}[r] > 0$ and $\texttt{bwt}[r] := s[n-1]$ otherwise. It follows that `bwt` is a particular permutation of the characters of s. We also define $\texttt{less}[c]$ for $c \in \Sigma$ as the number of occurrences of characters lexicographically smaller than c in s (or in `bwt`) and $\texttt{occ}[c, r]$ as the number of occurrences of c in $\texttt{bwt}[0], \ldots, \texttt{bwt}[r]$.

Read Mapping with Exact Backward Search and Branching on Uncertain Nucleotides The described data structure allows us to efficiently find all occurrences of any given pattern of length m within the indexed reference in $O(m)$ time, independently of the length of the reference sequence, as follows. All starting positions of the occurrence of a given pattern, such as $P =$ `AGC`, can be found adjacent to each other in an interval $[L, R]$ of the suffix array, i.e., as $\texttt{pos}[L], \texttt{pos}[L+1], \ldots, \texttt{pos}[R]$. For the empty pattern of length zero, we initially have $[L, R] = [0, n-1]$. The *Backward Search* algorithm tells us how to update L and R if we *prepend* another character to the existing pattern.

Lemma 1 (Backward Search (Ferragina and Manzini, 2000))**.** Let $P^+ := aP$ with $a \in \Sigma$; let $[L, R]$ be the known interval for P and $[L^+, R^+]$ the sought interval for P^+. Then

$$L^+ = \texttt{less}[a] + \texttt{occ}[a, L-1],$$
$$R^+ = \texttt{less}[a] + \texttt{occ}[a, R] - 1.$$

Since each update is done by two simple array lookups, computing the interval for a pattern of length m takes $O(m)$ time. For error-tolerant read mapping with up to k errors, one recursively branches into different sub-searches, not only searching for the exact read sequence, but taking possible substitutions, insertions and deletions at each position into account, potentially leading to a running time exponential in k.

Here we take an approach that allows us to use exact matching (i.e., $k = 0$), but uses a different kind of branching strategy. Recall that the reference is a string over $\Sigma \cup \{\$\}$, but the converted reads are strings over Ω, containing lower-case nucleotides and `+`, and that in both types of sequences, homopolymer lengths have been artificially limited to $N = 4$. In the mapping step, we ignore `+` characters in the reads. As we have indexed the reverse references, we can process the reads from left to right, first mapping against the keys, then (with the remainder of the read) against the MIDs, then against the tags, and finally against the loci.

Our mapping procedure processes the hybrid representation of a read (over Ω without `+`) character by character, updating the suffix array interval $[L, R]$ according to Lemma 1: If the

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

processed character is upper case (in Σ) then Lemma 1 is applied directly. If, however, the processed character is lower case, designating a potential but uncertain flow, the search branches into two cases. The first case corresponds to processing the character as if it were upper case; the second case simply skips the character. In order to avoid excessive branching due to many adjacent lower-case characters without informative upper-case characters in between, some uncertain (lower-case) nucleotides are converted to their certain (upper-case) counterparts using a greedy strategy, such that at most three uncertain nucleotides exist in each sliding window of length 20.

Application of Lemma 1 continues in each sub-branch until the suffix array interval $[L^+, R^+]$ becomes empty (i.e., $L^+ > R^+$), or until all read characters have been used. In the former case, we record the last valid $[L, R]$ interval; in the latter case, we record the final interval. In both cases, we also note the associated match length ℓ (number of matching characters).

Since we do not allow for errors when mapping (but do branch for uncertain characters), we re-start the mapping procedure at each position in the read, instead of only from the beginning. This results, for each starting position i within the hybrid read sequence, in a maximal match length ℓ_i and an associated suffix array interval $[L_i, R_i]$. The sequence of ℓ_i values is referred to as *matching statistics*. If ℓ_i is too small to indicate a significant match we treat it as zero. The required values for retaining ℓ_i are given in Table 1 (column L) for each read element.

Typically, for sufficiently long maximal matches, there is a single matching position within the indexed sequence, i.e., the final suffix array interval $[L_i, R_i]$ has length 1. In general, for each found reference position $\texttt{pos}[r]$, $L_i \leq r \leq R_i$, with maximal match length ℓ_i, we find the associated reference sequence k (e.g., MID or locus) and keep track of the length of the maximal match between the read and each individual reference sequence k. Additionally, we keep track of accumulated match lengths as follows. We say that there is a *jump* in matching statistics at read position i if either $i = 0$ or $\ell_i > \ell_{i-1} - 1$, i.e., the found maximally matching string is a different one than at the previous position (Rahmann, 2003). For each reference sequence, we add the match lengths at jumps to obtain cumulated match lengths.

Interpretation of Maximal and Cumulated Matching Statistics For each read part, the mapping procedure yields two numbers for each reference sequence k: the length $\ell^{(k)}$ of the longest match, and the cumulated length $c^{(k)}$ of longest matches after jumps. We say that a read potentially maps to reference k if $\ell^{(k)} \geq L$ or $c^{(k)} \geq C$ for thresholds L, C as given in Table 1.

The final decision against which references the current part of the read is precisely aligned is made on the basis of the two resulting candidate sets

$$S_{\mathrm{C}} := \{k \,|\, c^{(k)} \geq C\}, \quad S_{\mathrm{L}} := \{k \,|\, \ell^{(k)} \geq L\}$$

according to the following case distinction.

If $|S_{\mathrm{C}}| = |S_{\mathrm{L}}| = 0$, no suitable references have been found, and the read has not been mapped. If $|S_{\mathrm{C}}| = 0$ and $|S_{\mathrm{L}}| = 1$, the read is aligned against the unique sequence in S_{L}, even though there is only weak mapping evidence. If $|S_{\mathrm{C}}| = 0$ and $|S_{\mathrm{L}}| > 1$, no reasonable unique reference was identified, and the read is not aligned. If $|S_{\mathrm{C}}| = 1$ and $|SL| = 0$, the read is aligned against the unique sequence in S_{C}, even though there is only weak mapping evidence. If $|S_{\mathrm{C}}| = |S_{\mathrm{L}}| = 1$ and $S_{\mathrm{C}} = S_{\mathrm{L}}$, both criteria indicate the same reference, and the read is aligned against it; this is the ideal and most common case. If $|S_{\mathrm{C}}| = |S_{\mathrm{L}}| = 1$ and $S_{\mathrm{C}} \neq S_{\mathrm{L}}$, the criteria indicate different references, and the read is aligned against both, although only weak evidence exists for either. If $|S_{\mathrm{C}}| = 1$, $|S_{\mathrm{L}}| > 1$ and $S_{\mathrm{C}} \subset S_{\mathrm{L}}$, both criteria agree on the unique sequence in S_{C}, and the read is aligned against it. In the same case, if $S_{\mathrm{C}} \not\subset S_{\mathrm{L}}$, there is a contradiction between the criteria and the read is aligned against $S_{\mathrm{C}} \cup S_{\mathrm{L}}$; this is a rare case. If $|S_{\mathrm{C}}| > 1$ and $|S_{\mathrm{L}}| = 0$, no

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

	A	C	G	T	N	B	D	H	V	R	Y	S	W	K	M
A	10	−19	−19	−19	0	−19	2	2	2	5	−19	−19	5	−19	5
C	−19	10	−19	−19	0	2	−19	2	2	−19	5	5	−19	−19	5
G	−19	−19	10	−19	0	2	2	−19	2	5	−19	5	−19	5	−19
T	−19	−19	−19	10	0	2	2	2	−19	−19	5	−19	5	5	−19
a	10	−21	−21	−21	0	−21	2	2	2	5	−21	−21	5	−21	5
c	−21	10	−21	−21	0	2	−21	2	2	−21	5	5	−21	−21	5
g	−21	−21	10	−21	0	2	2	−21	2	5	−21	5	−21	5	−21
t	−21	−21	−21	10	0	2	2	2	−21	−21	5	−21	5	5	−21
+	−∞	−∞	−∞	−∞	−∞	−∞	−∞	−∞	−∞	−∞	−∞	−∞	−∞	−∞	−∞
A	10	−19	−19	−19	0	−19	2	1	1	5	−19	−19	5	−19	3
C	−19	10	−19	−19	−3	−1	−19	−1	−1	−19	2	2	−19	−19	2
G	−19	−19	10	−19	0	1	2	−19	1	5	−19	3	−19	5	−19
T	−19	10	−19	10	4	6	2	6	−1	−19	10	2	5	5	2
a	10	−21	−21	−21	0	−21	2	1	1	5	−21	−21	5	−21	3
c	−21	10	−21	−21	−3	−1	−21	−1	−1	−21	2	2	−21	−21	2
g	−21	−21	10	−21	0	1	2	−21	1	5	−21	3	−21	5	−21
t	−21	10	−21	10	4	6	2	6	−1	−21	10	2	5	5	2
+	−∞	−∞	−∞	−∞	−∞	−∞	−∞	−∞	−∞	−∞	−∞	−∞	−∞	−∞	−∞
A	10	−19	10	−19	4	−1	6	2	6	10	−19	2	5	2	5
C	−19	10	−19	−19	0	1	−19	2	1	−19	5	3	−19	−19	5
G	−19	−19	10	−19	−3	−1	−1	−19	−1	2	−19	2	−19	2	−19
T	−19	−19	−19	10	0	1	1	2	−19	−19	5	−19	5	3	−19
a	10	−21	10	−21	4	−1	6	2	6	10	−21	2	5	2	5
c	−21	10	−21	−21	0	1	−21	2	1	−21	5	3	−21	−21	5
g	−21	−21	10	−21	−3	−1	−1	−21	−1	2	−21	2	−21	2	−21
t	−21	−21	−21	10	0	1	1	2	−21	−21	5	−21	5	3	−21
+	−∞	−∞	−∞	−∞	−∞	−∞	−∞	−∞	−∞	−∞	−∞	−∞	−∞	−∞	−∞

Table 2: Amplikyzer's scoring function between the extended IUPAC DNA alphabet (genomic) and the hybrid flowgram-DNA alphabet (reads). Top: standard scoring; middle: scoring for bisulfite C ↦ T treatment; bottom: scoring for bisulfite G ↦ A treatment.

reasonable unique reference was identified, and the read is not aligned. If $|S_C| > 1$ and $|S_L| = 1$, the procedure is symmetric to case $|S_C| = 1, |S_L| > 1$. Finally, if $|S_C| > 1$ and $|S_L| > 1$, the read is aligned against the union $S_C \cup S_L$.

3.3 Aligning Flowgrams to DNA References

While the read mapping procedure described in Section 3.2 selects those references to which a given read may plausibly map, the alignment procedure produces a basepair-level alignment between the selected references and the read.

In principle, the flowgram-to-string aligment algorithm of Martin and Rahmann (2013) could and should be used. It directly aligns a flowgram to a DNA sequence using an elaborate scoring scheme $s(B^f, x)$ for each flow $B^f \equiv (B, f) \in \Sigma \times \{0.00, 0.01, 0.02, \dots\}$ and each potential substring $x \in \Sigma^*$ of the reference. Presently, however, only a relatively slow reference implementation of this algorithm exists. Therefore, a different method has been implemented in *amplikyzer*: We align the hybrid representation of the read (over alphabet Ω) to the selected references (over the DNA alphabet Σ) using a variation of the standard semi-global dynamic programming sequence alignment algorithm. Initial computational experiments indicated that this method is about ten times faster and produces similar alignments.

Thus the task at hand is to align a read $s \in \Omega^*$ in hybrid representation to a given DNA reference sequence r, where we allow that ambiguous IUPAC characters (any of RYWSKMBDHVN)

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

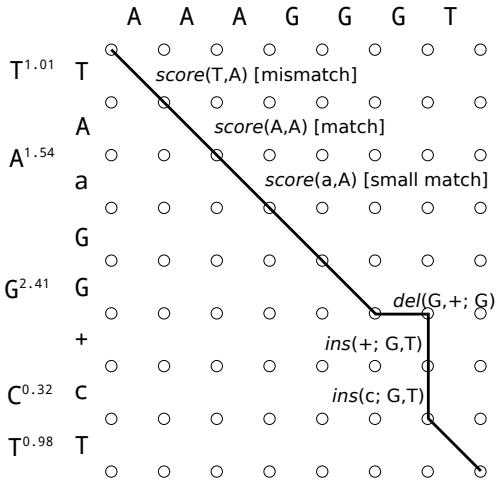

Figure 1: Alignment graph between a hybrid flowgram-DNA string (vertical) and a genomic reference (horizontal). Potentially, all horizontal, vertical and main diagonal edges between the nodes exist. Here the optimal alignment, a maximum-scoring path through the graph, has been marked. Scoring of diagonal edges is done using the scores from Table 2. Scoring of horizontal edges uses the *del* function described in Eq. (2), and scoring of vertical edges uses the *ins* function described in Eq. (3).

may exist in r to denote SNP positions and their possible realizations.

We use the three scoring schemes shown in Table 2: one for untreated (parts of) reads, one for forward tagged bisulfite-treated reads with expected `C` $\mapsto$ `T` substitutions, and one for reverse tagged bisulfite-treated reads with `G` $\mapsto$ `A` substitutions. The scores are computed as log-odds from expected frequencies of matches and mismatches in true alignments using *amplikyzer*'s scoring module. It is assumed that the frequency of matches in correct alignments is 95% for flows considered certain, allowing a reasonable margin for SNP positions and sequencing errors. For uncertain flows (lower-case letters), a match frequency of 96% is assumed, slightly higher than for upper-case letters. The rationale behind this constraint is that *if* an uncertain flow is used, there should be a good reason for it, i.e., a match in the alignment. The scores are scaled and rounded such that the standard match score achieves a value of 10.

Note that the `+` symbol that indicates a long homopolymer of unspecified length above a cut-off N in the read cannot be aligned directly to a genomic nucleotide and has to be aligned to a gap character.

Indeed, the key difference of the alignment algorithm in comparison to existing ones is its scoring of insertions and deletions. Recall that each alignment can be viewed as a path in an *alignment graph* (also *edit graph*) with horizontal, vertical and diagonal edges (oriented left-to-right and top-to-bottom) and that the score of an alignment can be written as the sum of the scores assigned to each edge of the path. The optimal alignment then corresponds to a maximum-scoring path in that graph (see Figure 1). To specify a scoring scheme, we specify how to score each edge, depending on its type and the annotated characters.

Let $del(h_1, h_2; g)$ be the (negative) score a horizontal edge (an additional genomic character g between two hybrid symbols h_1, h_2) and $ins(h; g_1, g_2)$ be the (negative) score for a vertical edge (an additional hybrid symbol h between two genomic characters g_1, g_2). Usually, such gap scores are assigned a constant value per edge, or using an affine function depending on the total length of the gap. The gap scoring in *amplikyzer* differs, since the score depends on the surrounding

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

context. In detail, we use

$$del(h_1, h_2; g) := \begin{cases} 0 & \text{if } h_2 \text{ does not exist,} \\ -25 & \text{if } h_2 \in \{\texttt{ACGTacgt}\} \text{ and } h_2 \text{ does not match } g, \\ -26 & \text{if } h_2 \in \{\texttt{ACGTacgt}\} \text{ and } h_2 \text{ matches } g, \\ -1 & \text{if } h_2 = \texttt{+} \text{ and } h_1 \text{ matches } g, \\ -\infty & \text{if } h_2 = \texttt{+} \text{ and } h_1 \text{ does not match } g. \end{cases} \quad (2)$$

$$ins(h; g_1, g_2) := \begin{cases} 0 & \text{if } h = \texttt{+}, \\ -5 & \text{if } h \in \{\texttt{acgt}\}, \\ -25 & \text{if } h \in \{\texttt{ACGT}\} \text{ and } g_1 \neq g_2, \\ -26 & \text{if } h \in \{\texttt{ACGT}\} \text{ and } g_1 = g_2. \end{cases} \quad (3)$$

The rationale is as follows. A standard gap score (e.g., as used in the nucleotide BLAST defaults) should be approximately two to three times the negative value of the match score, which has been fixed to +10, so a standard indel is assigned a score of −25. In order to disambiguate alignments, we penalize insertions within genomic runs of the same nucleotide or before a matching hybrid character slightly more (using −26 instead of −25).

Of course, consuming genomic characters g before a `+` that match the previous read character h_1 should not be penalized severely; hence a score of −1. We do not require $h_1 = g$, but only that they match, in the sense that a `G` in the read matches a genomic `R` or `N`. On the other hand, consuming non-matching characters before a `+` should not be possible.

On the other hand, consuming an optional nucleotide in the read should be inexpensive (−5), and consuming a `+` should not be penalized at all.

4 The Amplikyzer Software

This section describes the features of the *amplikyzer* software that distinguish it from existing solutions. The purpose of *amplikyzer* is to provide an automated analysis workflow, configurable by several parameters, from the SFF file output by the sequencer containing flowgram information to high-quality methylation plots. We describe the requirements to install and run the *amplikyzer* software (Section 4.1), explain how to prepare configuration files for an analysis (Section 4.2), and we discuss the basic workflow (Section 4.3), together with specialized options and adjustable parameters (Section 4.4).

4.1 Requirements

Amplikyzer is written in pure Python 3.2 and hence runs on every major operating system, in particular on Windows, MacOS X and Linux. It is an automation-friendly command-line application that does not require user interaction after specifying configuration files and analysis parameters.

Several parts of *amplikyzer* are provided as separate packages because they may be of independent interest in other projects. For example, an optional graphical user interface (GUI) exists on top of the software for convenient parameter entry (cf. Figure 8). The GUI is provided in a separate package `geniegui` and requires a current release of Tcl/Tk to be installed and the `ttk` extension of the `tkinter` package. This GUI package is not specific to amplikyzer, but will transform every command-line argument parser written using the standard Python library

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

```
[MIDS]
MID01 = ACGAGTGCGT
MID02 = ACGCTCGACA
...
[LABELS]
MID01 = Alice
MID02 = Bob
...
[TAGS]
FWD = CTTGCTTCCTGGCACGAG
REV = CAGGAAACAGCTATGAC
[LOCI]
MEST =
  CCGCTGCTGGCCAGCTCTGCACGGCT,
  GCGGGCTCTGCGGCGCCCGGTGCTCTGCAACGCT...GTGCG,
  GTGGGAACGAGGGGGTGTGGCTGG
...
```

Table 3: Abbreviated example of an *amplikyzer* configuration file (extension `.conf`), following the standard INI format. It is recommended to store the sections for MIDs, labels, tags and loci in different files, such as `mids.conf`, `labels.conf`. `tags.conf` and `loci.conf` in the same directory as the SFF file to be analyzed.

`argparse` package into a GUI that will launch the command-line program with the user-provided parameter values. Another package parses the SFF file format, while another one implements generic BWT-based read mapping algorithms in Python. All of these packages are collected into a single ZIP archive and should be installed together to obtain the full functionality. The `numpy` and `matplotlib` Python packages are required for graphical output.

4.2 Basic Configuration

We recommend to use a separate directory for each analysis. Such a directory should contain the SFF file and additional configuration files required for analysis. These configuration files must specify MID sequences, optionally human-readable labels to replace MIDs in the analysis output, forward and reverse tag sequences, and ROI sequences with primers, as they appear in the reference genome. The configuration files must adhere to the standard INI file format[5]; Table 3 shows an example. MIDs and tags are typically re-used in every analysis, as these sequences do not change, but the labels and analyzed loci are project-specific. The `[LABELS]` section is optional, but convenient if one wants to identify each sample in the analysis results not by `MID01`, but by a more human-readable alias. If no label is specified for any MID, the MID name is used (typically a string like `MID07`).

The `[LOCI]` section is the most complex one, as it contains each locus sequence, separated into forward primer, region of interest (ROI) and reverse primer, as a contiguous forward-strand genomic sequence (without simulated bisulfite treatment), with the three parts separated by commas. In other words, the format of a single locus entry is as follows.

[5] see `http://en.wikipedia.org/wiki/INI_file` for a description

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

```
GENE_NAME = FORWARD_PRIMER,REGION_OF_INTEREST,REVERSE_PRIMER
```

There should be no spaces between the commas; all parts must be valid IUPAC DNA sequences. In principle, one entry should be on a single line, but using the standard INI format indentation rules, an entry can be split across several lines. Ambiguous DNA characters (any of `NBDHVRYSWMK`) can be used to designate positions of known SNPs. Analysis options in *amplikyzer* allow to select only those reads with a specific allele at such positions. For efficiency, the `[LOCI]` section should only contain loci that are present in the run.

Comment lines starting with a `#` character can be added to any configuration file, for example to comment on SPNs or particular CpGs. The configuration files must be UTF-8 encoded; generally it is recommended to avoid non-ASCII characters.

4.3 Workflow

The basic workflow for a methylation analysis with *amplikyzer* consists of the following steps.

First, a separate analysis directory containing the SFF file and configuration files, as described in Section 4.2 must be prepared. The path to this directory is the only required option for each of the steps described below.

Usually, *amplikyzer* is first invoked with the `analyze` subcommand. This runs the read mapping and alignment algorithms described in Section 3 and creates an *amplikyzer* analysis file (extension `.akzr`) in the same directory. This file contains the identified MID, tag and ROI for each flowgram read of the SFF file in a human-readable and automatically parsable text format. This analysis is time-consuming (in comparison to the other steps) and may take ten minutes up to a few hours (depending on processor speed and number of cores). It only needs to be done once for each SFF file, unless one changes some of the analysis parameters (Section 4.4).

The `statistics` subcommand outputs analysis statistics, either on screen or into a separate text file (extension `.stats`). Statistics include the number of reads for each MID, tag (forward or reverse), locus, and most importantly, the number of reads for each successfully aligned MID/ROI pair.

The `align` subcommand collects the reads for each MID/ROI pair and produces multiple alignments in text or FASTA format, either for the whole ROI, or only for sites with variations, or only for CpGs. All alignments are saved into the `alignments/` subdirectory of the analysis directory. Different options control which reads are included and excluded. The resulting alignments can be viewed with most standard alignment editors.

The `methylation` subcommand internally creates multiple alignments at each CpG site, computes the methylation status of each read at each CpG and generates *individual methylation analyses*, displaying the state of each CpG in each read for a given MID/ROI combination in textual or graphical format, or *comparative methylation analyses*, displaying overall methylation rates of each CpG in a ROI across different samples identified by MID or label. This subcommand can be run several times while varying some of the options, including the stringency of filters, selection of alleles, etc. All textual methylation analysis reports and methylation plots are saved into the `methylation/` subdirectory of the analysis directory.

There exists another subcommand, `printreads`, that allows to output the reads of the SFF file in nucleotide FASTA, simple nucleotide text, hybrid flowgram-DNA (alphabet `ACGTacgt+`) text, or flowgram text format. It also allows to create histograms of the flow intensities in the SFF file (this requires the *gnuplot* software). All of these features are mainly useful for diagnostic purposes after a failed sequencing run and are not necessary during the normal *amplikyzer* workflow.

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

4.4 Adjustable Parameters

The *amplikyzer* software comes with built-in help, available using the `--help` option of each subcommand. For example, running

```
python3.2 -m amplikyzer analyze --help
```

displays analysis parameters and options. We only discuss the most important ones.

The `analyze` subcommand makes use of a modern multi-core system by running the analysis in parallel processes; the number of processes to be used can be specified using the `-j` option. The parameters μ and N of the hybrid flowgram-DNA representation described in Section 3.1, as used for read mapping, can be set using options `--certainflow` and `--maxflow`, respectively; their default values are $\mu = 0.2$ and $N = 4$. During the alignment phase, μ can be set differently using the option `--alignmaybeflow`; its default value is 0.35.

The `methylation` subcommand allows to specify the following parameters.

`--loci` with a space-separated list of locus names runs the analysis only on the specified loci. The default `*` is to iterate over each locus for which sufficiently many alignments exist.

`--mids` allows to restrict the analysis to given MIDs. Using the default `*` on both loci and MIDs provides a fully automatic analysis of each MID/ROI combination for which enough reads are available without further interaction.

`--alleles` with a string specifying the nucleotide values for each SNP position only selects reads with the given allele/haplotype. Again, `*` iterates over all possible haplotypes separately. Specifying `N` aggregates reads irrespective of SNP status at the corresponding position. For example, if a specific locus `XYZ` has four SNP positions, the option combination `--loci XYZ --alleles ANNA` performs methylation analysis using all reads that show an `A` at the first and fourth SNP position, aggregating over the second and third SNP position. To the best of our knowledge, this way to restrict the analysis to specific alleles is a unique feature of *amplikyzer*.

`--minreads` with a number (default: 20) specifies the number of required informative reads in order to produce an output file. Specifying a lower number than the default should be avoided, as methylation rates computed on less than 20 reads are of questionable value.

`--type` with an analysis type name (`individual`, `comparative` or `smart`) decides which type of analysis to perform: individual methylation analysis or comparative methylation analysis or both (automatically excluding uninteresting combinations).

`--conversionrate` with a rate between 0 and 1 (default: 0.95) specifies the required bisulfite conversion rate for each read to be considered for the analysis. Reads with a lower conversion rate, as measured by the number of remaining `C`s, are excluded from analysis.

`--badcpgs` with an integer number or fraction (< 1) specifies the allowed number of CpGs in a read with undetermined CpG status. The status of a CpG is determined if the corresponding genomic `C` is aligned to a `T` ("unmethylated") or `C` ("methylated") from the read, and undetermined otherwise. If the number of fraction of undetermined CpGs exceeds the given threshold (default: 2), the read is excluded from analysis.

`--sort` with a list of sort orders allows to sort reads (during individual analysis) or samples (during comparative analysis) using the given sorting criteria. The syntax of this option

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

is complex, as several sort orders may be applied in combination. Each sort order consists of a *prefix* specifying according to which criterion we want to sort; it must be from the set {`meth:`, `mids:`, `alleles:`}, followed (without spaces) by a corresponding argument. For example, `meth:down` and `meth:up` sort the samples by decreasing and increasing overall methylation, respectively. Using `mids:MID03,MID01,MID04` restricts the output to the given MIDs in the given order. Using `alleles:GA,GG,CA,CG` restricts the output to the given alleles (assuming two SNP positions in the selected loci) in the given order.

Additionally, there are parameters `--remark` to specify an arbitrary remark for the analysis, `--format` to specify the output format (PDF, PNG, SVG or text) and `--style` to specify `color` or `bw` for graphical color or black-and-white output.

There are more options controlling input and output files; and there are additional options for the `printreads`, `statistics` and `align` subcommands. As they are not crucial for the main workflow, we refer to the built-in help of the software.

5 Experiments

To illustrate the capabilities of the algorithmic approach taken with the *amplikyzer* software, we analyzed an exemplary dataset. The dataset is available as supplementary material as a single ZIP file (156 MB), including the SFF file and all configuration files; it can be directly used to test a new *amplikyzer* installation. It can be obtained from `https://bitbucket.org/svenrahmann/amplikyzer/downloads`.

Material and Methods We established amplicon libraries for six imprinted loci (*GRB10*, *MEST*, *KCNQ1OT1*, *H19-CTCF6*, *RB1*-CpG85, and *SNRPN*) and one non-imprinted locus (*LAMA3*).

Human blood samples were obtained after written informed consent. Control blood samples from blood donors were anonymised. The study was approved by the ethics committee of the University of Duisburg-Essen (approval number 08-3858). For *GRB10*, *MEST*, *KCNQ1OT1* and *RB1* we used blood DNA from individuals with known normal or abnormal methylation patterns for amplicon library preparation. For *H19-CTCF6* and *LAMA3* we used DNA from individuals heterozygous for a single nucleotide polymorphism (SNP), which can be used to separate the parental alleles. Furthermore, we used DNA from two patients with either Prader-Willi-syndrome (PWS) or Angelman syndrome (AS) to obtain a calibration curve for the *SNRPN* locus. Both patients have a deletion of the chromosomal region 15q11q13 including *SNRPN*. The PWS patient carries the deletion on his unmethylated paternal chromosome 15 resulting in complete methylation of the *SNRPN* differentially methylated region (DMR), whereas the patient with AS carries the deletion on his methylated maternal chromosome 15, resulting in absence of methylation at this DMR. Eleven different amplicon libraries for the calibration curve were set up by mixing different amounts of DNA aliquots from the PWS and AS patients.

All 33 amplicon libraries were pooled, clonally amplified and sequenced on the Roche 454 GS Junior system.

The initial SFF file analysis was run with default parameters and took less than 20 minutes on a quadcore multiprocessor PC with an Intel Core i7-2600 processor at 3.4 GHz and with 8 GB of RAM, using four parallel processes. We ran the methylation analysis using the default parameters with a minimum conversion rate of 95% and a maximum of two undetermined CpGs in a single read, generating single and comparative plots for each of the 33 MID/ROI combinations with more than 100 usable reads.

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

MID	Sample	Locus	*amplikyzer*		*BiQ Analyzer HT*	
			#Reads	Avg. meth.	#Reads	Avg. meth.
MID 01	*RB1* maternal deletion	*RB1*-CpG85	34885	3.0	15928	3.6
MID 03	*RB1* paternal deletion	*RB1*-CpG85	1125	92.9	771	96.0
MID 04	normal control 1	*RB1*-CpG85	1137	51.5	456	57.4
MID 05	normal control 2	*RB1*-CpG85	783	52.4	351	57.7
MID 17	0% AS / 100% PWS	*SNRPN*	318	97.2	295	97.2
MID 18	10% AS / 90% PWS	*SNRPN*	483	87.1	451	88.5
MID 19	20% AS / 80% PWS	*SNRPN*	1090	76.3	977	77.0
MID 20	30% AS / 70% PWS	*SNRPN*	792	65.8	678	66.6
MID 21	40% AS / 60% PWS	*SNRPN*	146	56.8	134	57.4
MID 22	50% AS / 50% PWS	*SNRPN*	1098	45.0	939	46.4
MID 24	60% AS / 40% PWS	*SNRPN*	787	41.0	686	41.2
MID 25	70% AS / 30% PWS	*SNRPN*	150	29.7	124	30.4
MID 26	80% AS / 20% PWS	*SNRPN*	601	18.1	527	18.7
MID 27	90% AS / 10% PWS	*SNRPN*	617	10.0	558	9.9
MID 28	100% AS / 0% PWS	*SNRPN*	1029	2.1	900	2.0
MID 09	normal control 1	*LAMA3*	670	61.6	576	61.1
MID 10	normal control 2	*LAMA3*	723	65.1	618	65.7
MID 08	normal control 1	*H19-CTCF6*	3319	45.0	2890	48.2
MID 17	normal control 2	*H19-CTCF6*	4938	49.6	4293	48.9
MID 25	normal control 3	*H19-CTCF6*	2486	52.5	2062	51.1
MID 17	normal control 1	*KCNQ10T*	3579	57.1	3743	57.5
MID 18	normal control 2	*KCNQ10T*	5667	63.3	5370	62.2
MID 19	normal control 3	*KCNQ10T*	5180	63.9	5152	63.9
MID 20	normal control 4	*KCNQ10T*	5703	56.2	5713	55.1
MID 06	BWS	*KCNQ10T*	2302	17.8	2265	16.8
MID 12	upd(7)mat	*GRB10*	1665	96.0	1322	96.2
MID 09	normal control 1	*GRB10*	3419	48.0	2460	52.1
MID 10	normal control 2	*GRB10*	7959	50.3	6147	49.8
MID 04	normal control 3	*GRB10*	3418	52.7	2746	47.4
MID 12	upd(7)mat	*MEST*	8832	97.1	8209	97.2
MID 10	normal control 1	*MEST*	6151	50.6	6066	51.6
MID 11	normal control 2	*MEST*	4627	53.7	4285	56.4
MID 13	normal control 3	*MEST*	9968	49.1	10099	49.8

Table 4: Overview of libraries and analysis results using both *amplikyzer* and *BiQ Analyzer HT*.

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

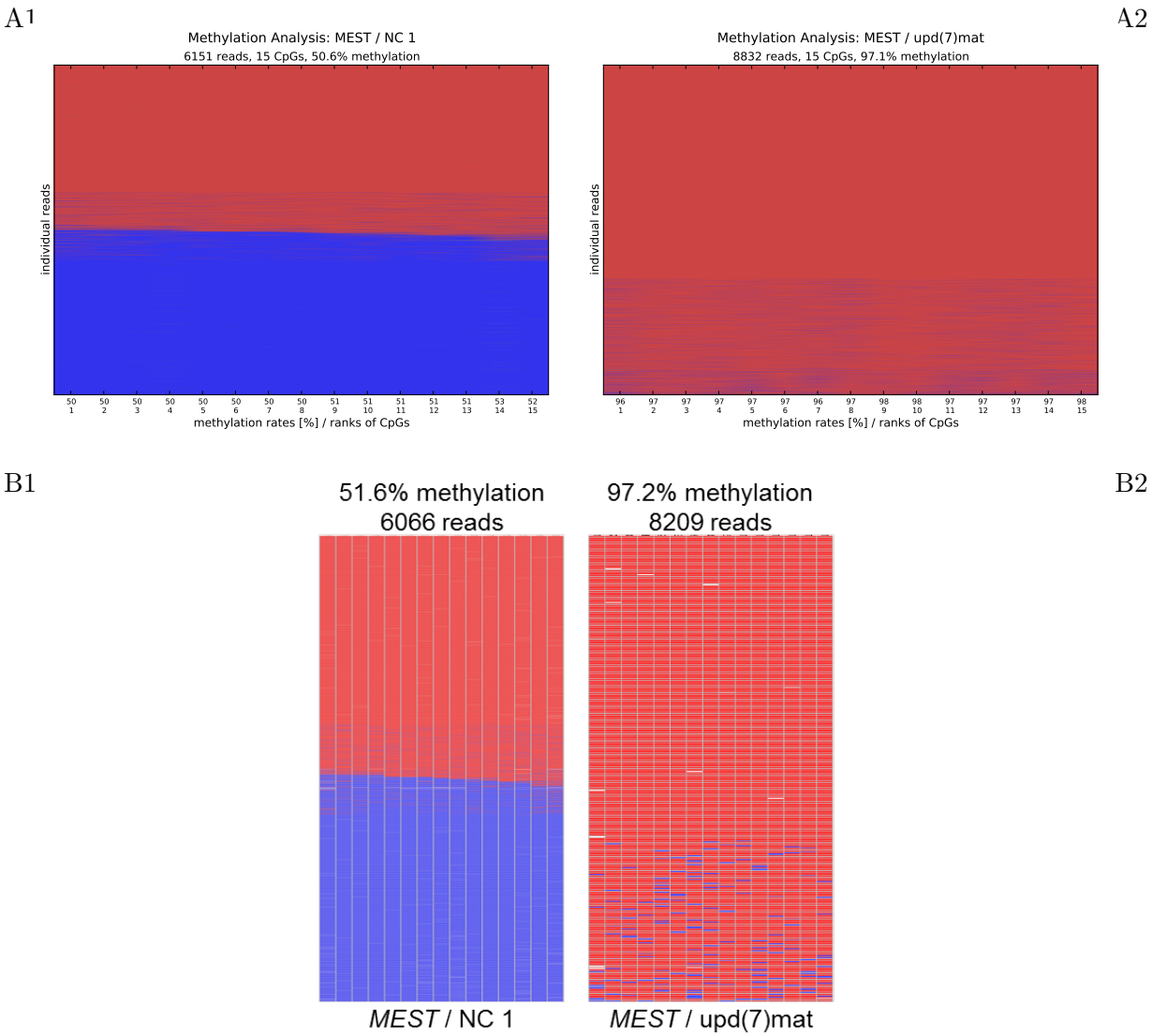

Figure 2: Methylation analysis of the imprinted *MEST* locus on human chromosome 7q32 by (A) *amplikyzer* and (B) *BiQ Analyzer HT*. In individual methylation plots, each row corresponds to an individual read and each column to a CpG. A red rectangle represents a methylated CpG, a blue rectangle an unmethylated CpG. Plots A1 and B1 show the methylation status of individual reads of a normal control (NC). The overall average methylation is approximately 50%, as expected for an imprinted locus. The reads can be separated into reads from the methylated maternal allele and reads from the unmethylated paternal allele. Plots A2 and B2 show an individual with hypermethylation for this locus. This hypermethylation is caused by a maternal uniparental disomy for chromosome 7 (upd(7)mat), i.e., the presence of two methylated maternal chromosomes 7 and the absence of an unmethylated paternal copy for chromosome 7.

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

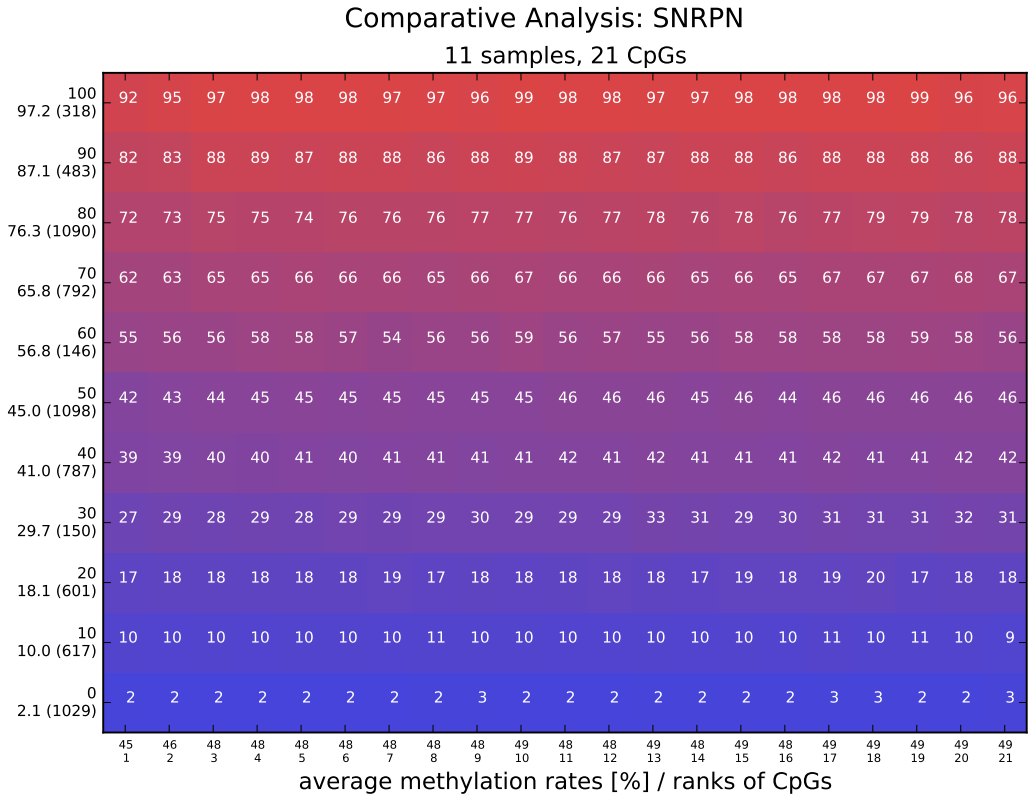

Figure 3: Comparative heatmap of the calibration curve for the *SNRPN* locus with methylation intervals of 10% obtained by mixing different amounts of an almost completely methylated DNA sample from a PWS patient with an almost completely unmethylated DNA sample of an AS patient. The expected percentage of methylation is shown in the upper part of each lane on the left side of the plot, whereas the obtained percentage of methylation is given below followed by the number of reads (in brackets).

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

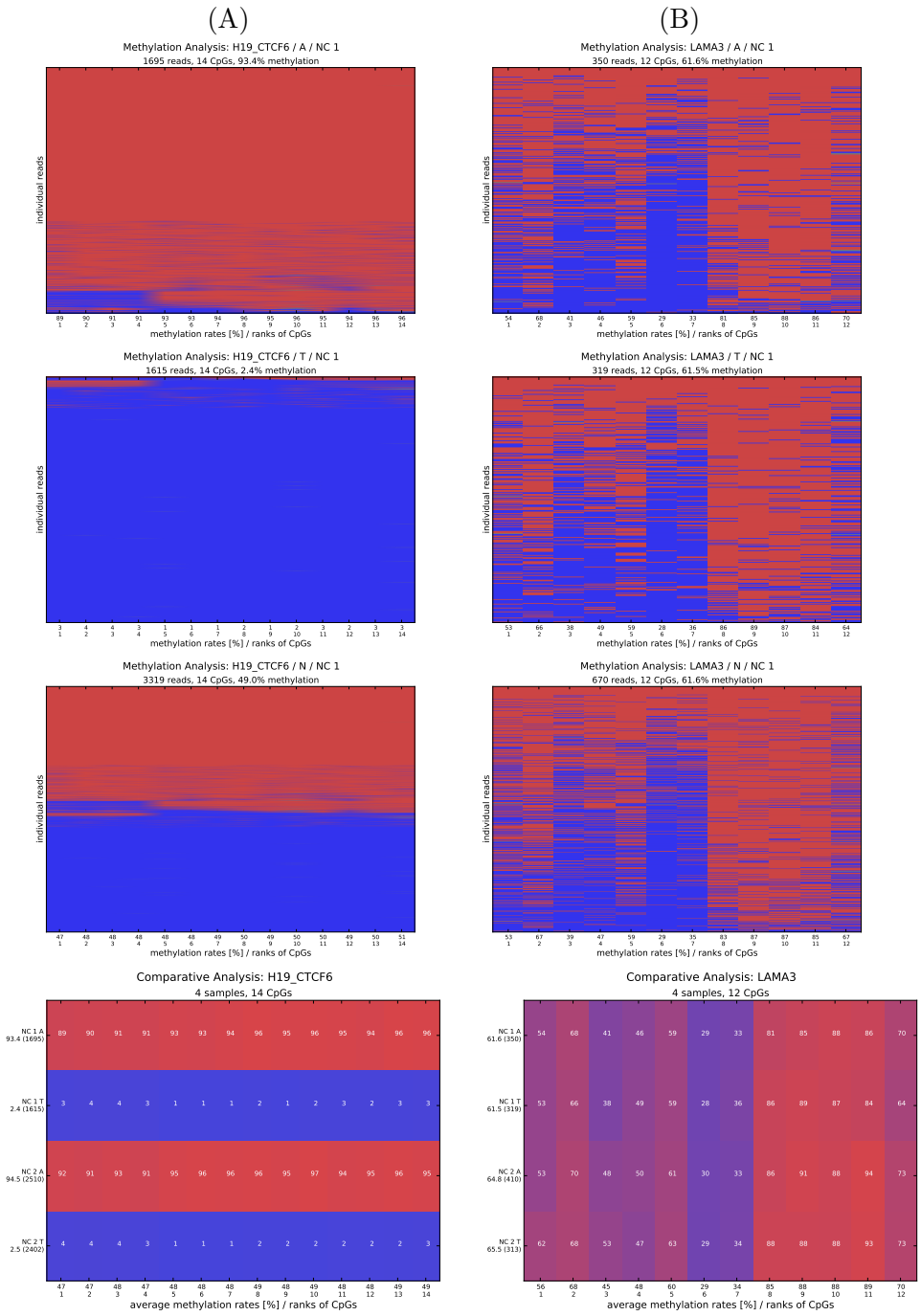

Figure 4: Allele separation. (A) Methylation analysis of the imprinted *H19-CTCF6* locus with separation of the parental alleles for a heterozygous normal control (NC1) with an `A/C` variant (rs2071094, `A/T` after bisulfite modification). The `A` allele represents the paternal methylated allele, whereas the `T` allele represents the unmethylated maternal allele. A combined plot (allele `N`) shows the methylation analysis for the same individual without separation. A comparative plot of two NCs (obtained by sorting by MIDs and alleles) clearly shows the distinct methylation patterns of the alleles. (B) Similar methylation analysis for the non-imprinted *LAMA3* locus of a heterogygous normal control (NC) with an `A/C` variant (rs1711451, `A/T` after bisulfite modification). Here separation of the alleles shows that both parental alleles have a similar methylation pattern, indicating that methylation at this locus is not allele or parent-of-origin specific.

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

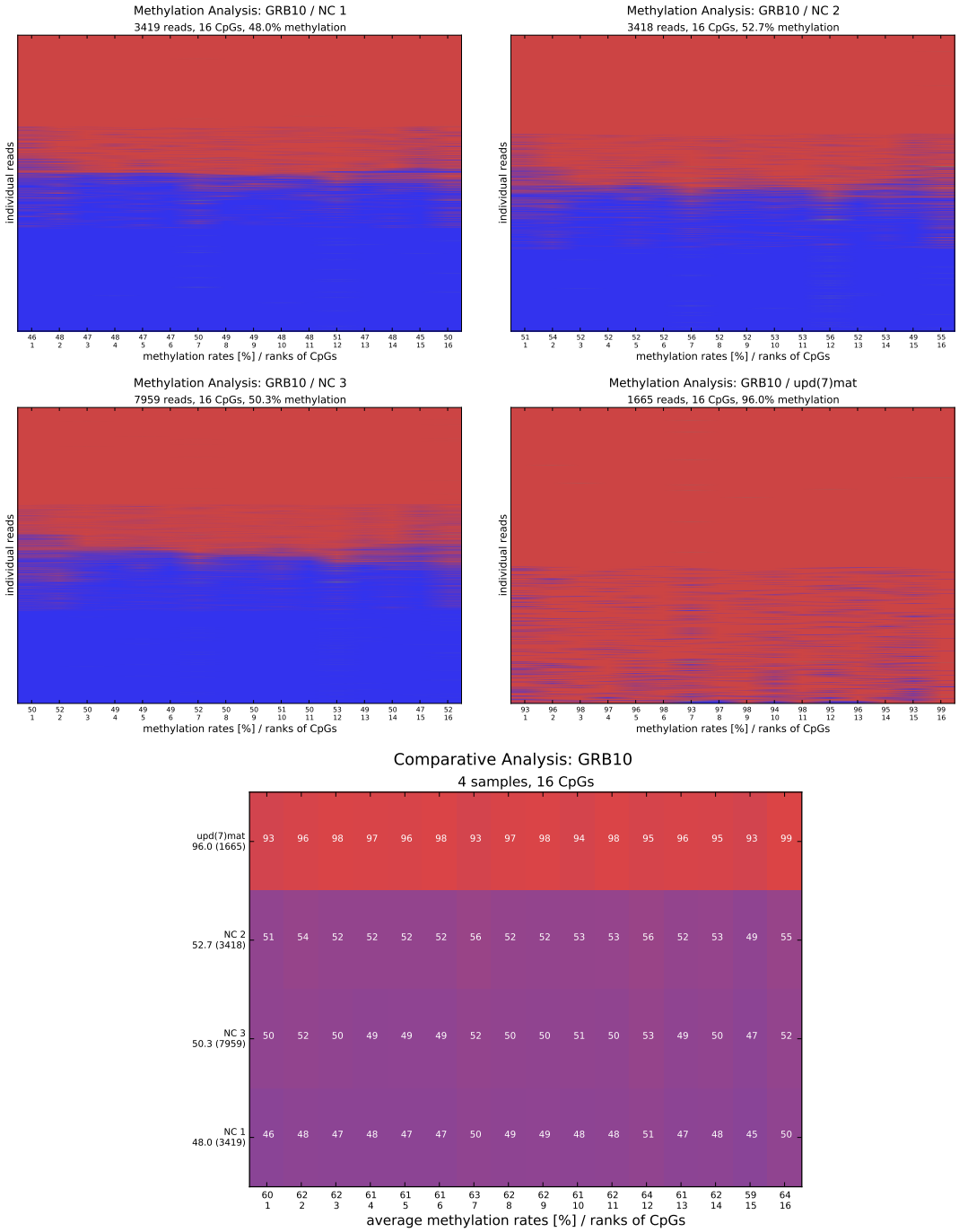

Figure 5: Methylation analysis of the *GRB10* locus on human chromosome 7p12 of three normal controls (NC1–3) with approximately 50% methylation, as expected for an imprinted locus, and an individual with hypermethylation of 96% caused by a maternal uniparental disomy for chromosome 7 (upd(7)mat). A comparative plot of all individuals is also shown.

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

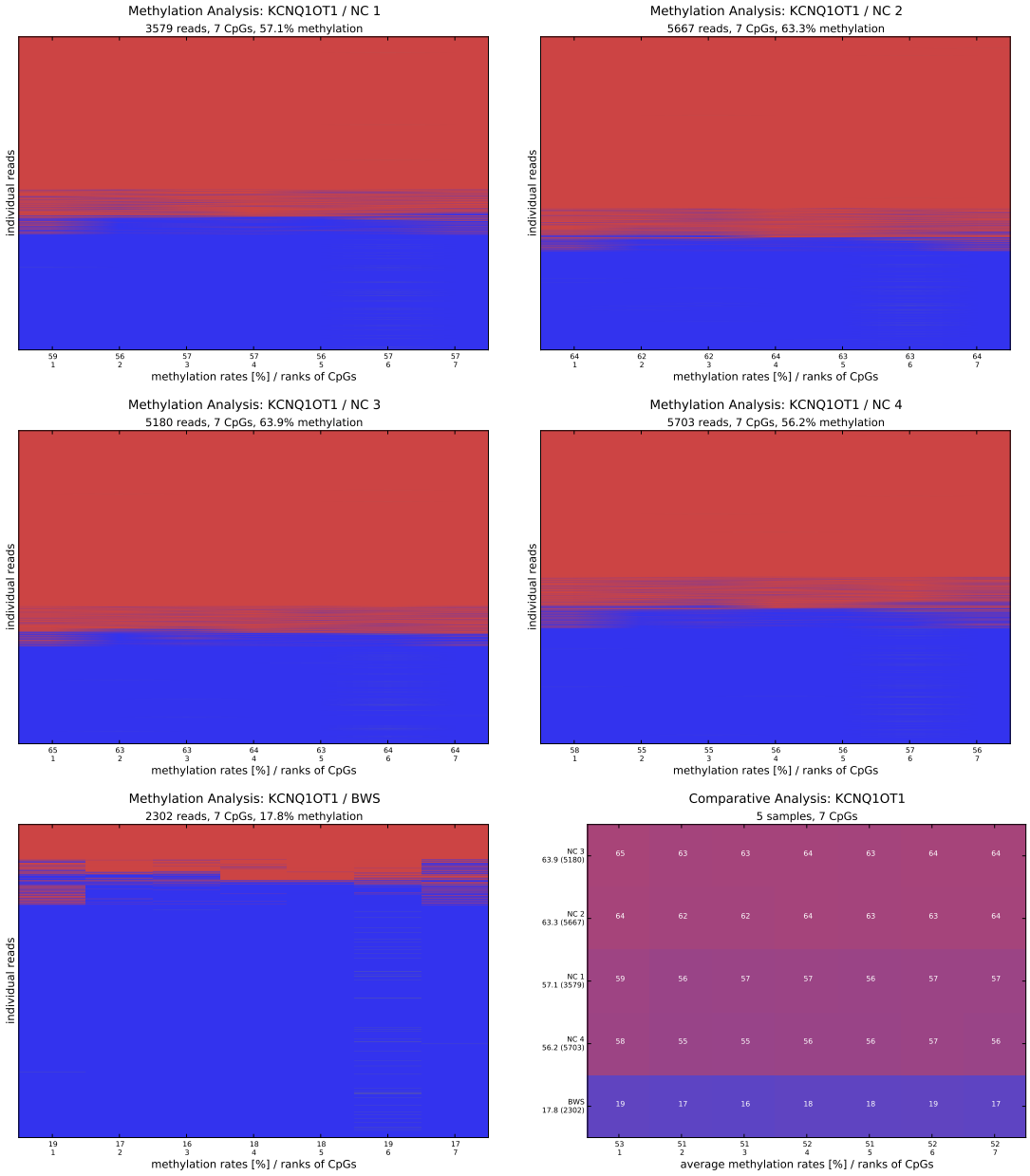

Figure 6: Methylation analysis of the *KCNQ1OT1* locus on human chromosome 11p15 of four normal controls (NC1–4) with average methylation rates between 56.2% and 63.9% and a patient with Beckwith-Wiedeman syndrome (BWS). The patient is a somatic mosaic for the methylation defect, meaning that he has normal methylated cells and cells with a methylation defect, resulting in hypomethylation of approximately 20%. A comparative plot of all individuals, sorted by overall methylation, is also shown.

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

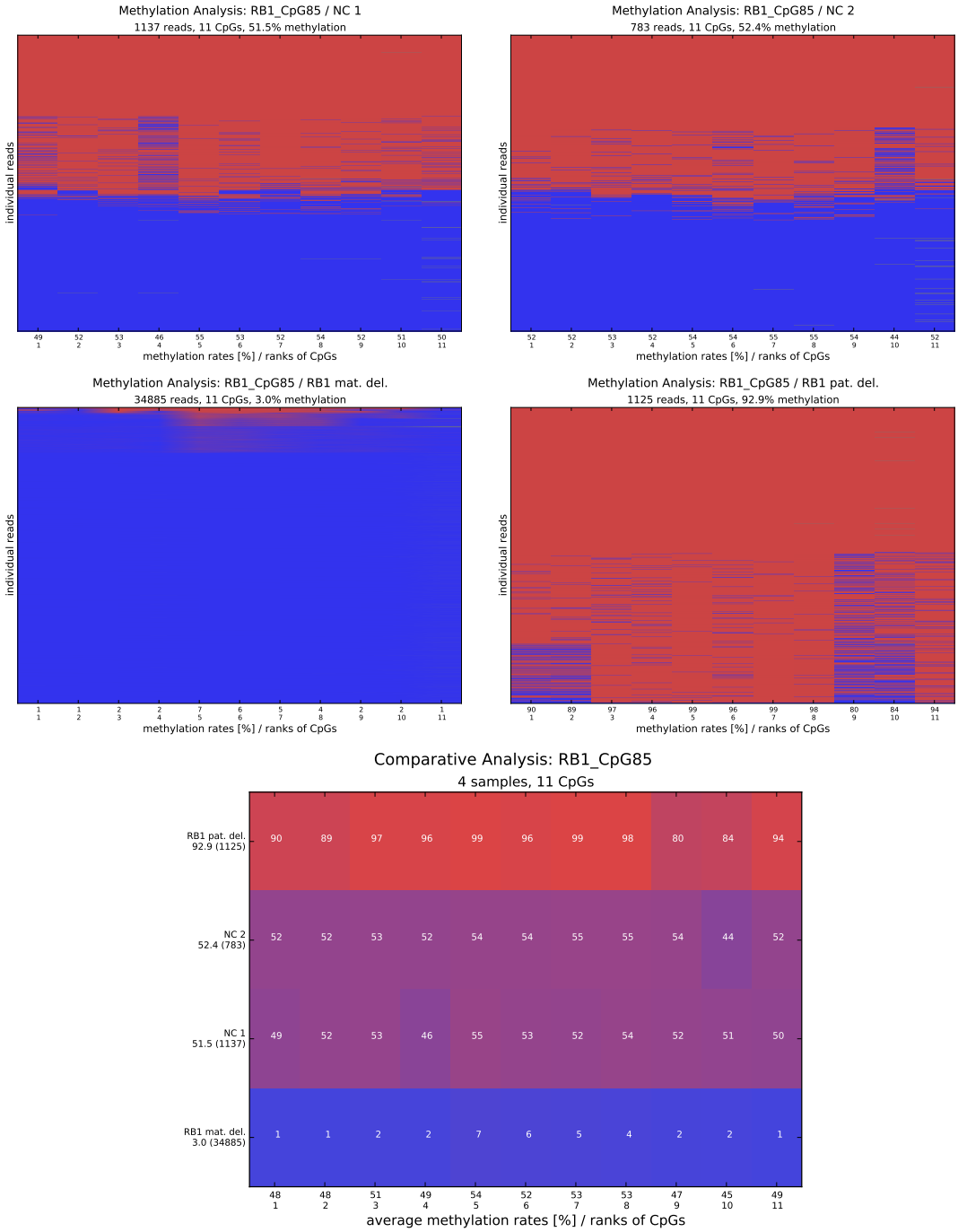

Figure 7: Methylation analysis for the Retinoblastoma 1 (*RB1*) locus at the CpG85 island in intron 2 on human chromosome 13 of two normal controls (NC1, NC2), a patient with a deletion affecting the *RB1* gene on the methylated maternal allele (*RB1* mat. del.) and a patient with a deletion affecting the unmethylated paternal allele (*RB1* pat. del.). A comparative plot of all individuals, sorted by overall methylation, is also shown.

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

Results The data set consists of 167 351 sequence reads. The number of sequence reads obtained for each MID/ROI combination are shown in Table 4.

A methylation analysis of individual reads, comparing *amplikyzer* with *BiQ Analyzer HT*, of an imprinted locus (*MEST*) for a normal control and a patient with abnormal methylation is shown in Figure 2. While *amplikyzer* finds and analyzes slightly more reads and has fewer unresolved CpGs (grey rectangles), the overall numbers and results are comparable between both tools and correspond to the expected results.

The results obtained for the calibration curve for the *SNRPN* locus are shown in Figure 3 as a comparative analysis plot. In contrast to an individual plot, where each row corresponds to a single reads, each row in a comparative plot corresponds to a sample. In both cases, a column corresponds to a CpG of the region of interest.

The *amplikyzer* software allows automatic allele separation when a SNP is defined in the genomic reference sequence for a ROI, resulting in specific methylation plots for each allele. It is also possible to get the union over both alleles in a single plot by specifying `N` for the corresponding SNP. Separation of the parental alleles for the imprinted and differentially methylated *H19-CTCF6* and the non-imprinted *LAMA3* locus is shown in Figure 4. For a comparative analysis of these loci between several individuals, *amplikyzer*'s `--sort` option allows to sort the samples explicitly by specifying the MID order in which they should be plotted from top to bottom. (By default, samples are sorted by average methylation level.)

Individual methylation plots and comparative heatmaps for loci *GRB10*, *KCNQ1OT1*, and *RB1*-CpG85 are shown in Figures 5, 6 and 7, respectively.

Results for the analysis of all samples investigated are listed in Table 4. We have analysed the described libraries with both *amplikyzer* and *BiQ Analyzer HT*. For the latter, the FASTA file generated by the sequencer with all sequence reads of the run was separated according to MIDs using the *Geneious* software (Biomatters). For methylation analysis we used filter parameters similar to the ones used by the amplikyzer with default settings (minimal conversion rate of 0.95; maximal fraction of unrecognized sites of 0.2). A genomic reference sequence in FASTA format was loaded for each locus into *BiQ Analyzer HT* together with the corresponding MID-separated FASTA files for each sample. The results of the methylation analyses obtained by *amplikyzer* and *BiQ Analyzer HT* showed only slight variation but no significant differences with regard to the average overall methylation and the number of reads for almost all loci (Table 4; see Figure 2 for a detailed example).

6 Discussion and Conclusion

In comparison to existing software, *amplikyzer* obtains similar results where applicable, but offers both more convenience and more analysis capabilities. It is more convenient, because *amplikyzer* generates all plots directly from the SFF file, whereas for *BiQ Analyzer HT*, the FASTA file generated by the sequencer has to be separated according to MIDs using third-party software. *Amplikyzer*'s unique analysis capabilities consist of automatic allele separation (or combination) and its extensive sorting options for comparative plots (according to a given order of MIDs and/or alleles). *Amplikyzer* is driven by configuration files and command line options that can be prepared before the analysis is started instead, and it does not require interaction during the analysis. This makes *amplikyzer* usable within larger automated pipelines. For convenience, a basic graphical user interface to enter command line parameters interactively is provided (Figure 8).

Methodologically, *amplikyzer* is the only tool based on flowgrams instead of base-called FASTA

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

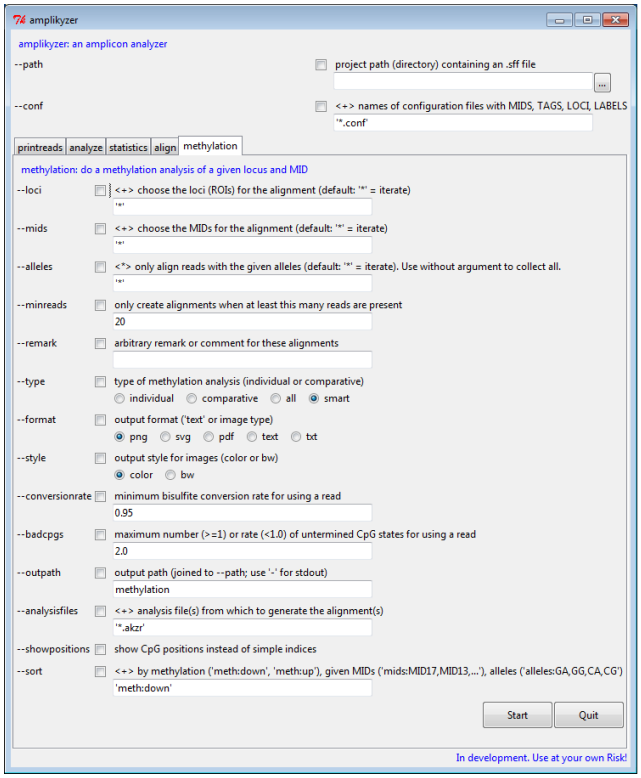

Figure 8: Part of the graphical user interface (GUI) of amplikyzer, showing the options for methylation analysis.

files. In the current version, a hybrid sequence representation is used for efficiency reasons. At the moment, this seems necessary during the mapping phase for quick MID and ROI identification. During alignment, however, we would like to switch to the pure flowgram-string alignment algorithm introduced by Martin and Rahmann (2013) in the future. However, this requires several speed-ups and implementation tricks; and initial tests do not indicate that the resulting alignments would be significantly different or better.

In summary, *amplikyzer* is a versatile and powerful tool for methylation analysis of deep bisulfite-sequenced amplicons. It can be obtained, together with the example dataset, from `https://bitbucket.org/svenrahmann/amplikyzer/downloads`.

Acknowledgements

We thank Katrin Rademacher and Nicholas Wagner for improving the bisulfite filter settings and Johanna Christina Czeschik for her helpful critical comments on the manuscript.

References

S. Berland, M. Appelbäck, O. Bruland, J. Beygo, K. Buiting, D. J. Mackay, I. K. Temple, and G. Houge. Evidence for anticipation in Beckwith-Wiedemann syndrome. *Eur. J. Hum. Genet.*, 2013. epub ahead of print.

J. Beygo, O. Ammerpohl, D. Gritzan, M. Heitmann, K. Rademacher, J. Richter, A. Caliebe,

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

R. Siebert, B. Horsthemke, and K. Buiting. Deep bisulfite sequencing of aberrantly methylated loci in a patient with multiple methylation defects. *PLoS ONE*, 8(10):e76953, 2013a.

J. Beygo, V. Citro, A. Sparago, A. De Crescenzo, F. Cerrato, M. Heitmann, K. Rademacher, A. Guala, T. Enklaar, C. Anichini, M. Cirillo Silengo, N. Graf, D. Prawitt, M. V. Cubellis, B. Horsthemke, K. Buiting, and A. Riccio. The molecular function and clinical phenotype of partial deletions of the IGF2/H19 imprinting control region depends on the spatial arrangement of the remaining ctcf binding sites. *Human Molecular Genetics*, 22(3):544–557, 2013b.

C. Bock, S. Reither, T. Mikeska, M. Paulsen, J. Walter, and T. Lengauer. BiQ Analyzer: visualization and quality control for DNA methylation data from bisulfite sequencing. *Bioinformatics*, 21:4067–4068, 2005.

P. Ferragina and G. Manzini. Opportunistic data structures with applications. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 390–398, 2000.

J. Köster and S. Rahmann. Snakemake: a scalable bioinformatics workflow engine. *Bioinformatics*, 28(19):2520–2522, 2012.

F. Krueger and S. R. Andrews. Bismark: a flexible aligner and methylation caller for bisulfite-seq applications. *Bioinformatics*, 27(11):1571–1572, 2011.

B. Langmead and S. Salzberg. Fast gapped-read alignment with Bowtie 2. *Nature Methods*, 9: 357–359, 2012.

M. Larkin, G. Blackshields, N. Brown, R. Chenna, P. McGettigan, H. McWilliam, F. Valentin, I. Wallace, A. Wilm, R. Lopez, J. Thompson, T. Gibson, and D. Higgins. Clustal W and Clustal X version 2.0. *Bioinformatics*, 23(21):2947–2948, 2007.

H. Li and R. Durbin. Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics*, 25(14):1754–1760, July 2009.

P. Lutsik, L. Feuerbach, J. Arand, T. Lengauer, J. Walter, and C. Bock. BiQ Analyzer HT: locus-specific analysis of DNA methylation by high-throughput bisulfite sequencing. *Nucleic Acids Research*, 39(suppl 2):W551–W556, 2011.

F. Lysholm, B. Andersson, and B. Persson. FAAST: Flow-space assisted alignment search tool. *BMC Bioinformatics*, 12:293, 2011.

M. Martin and S. Rahmann. Aligning flowgrams to DNA sequences. In *Proceedings of the German Conference on Bioinformatics (GCB) 2013*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2013. In press.

B. Pedersen, T.-F. Hsieh, C. Ibarra, and R. L. Fischer. MethylCoder: software pipeline for bisulfite-treated sequences. *Bioinformatics*, 27(17):2435–2436, 2011.

S. Rahmann. Fast and sensitive probe selection for DNA chips using jumps in matching statistics. In *Proceedings of the 2nd IEEE Computer Society Bioinformatics Conference (CSB 2003)*, pages 57–64. IEEE, 2003.

C. Rohde, Y. Zhang, T. P. Jurkowski, H. Stamerjohanns, R. Reinhardt, and A. Jeltsch. Bisulfite sequencing data presentation and compilation (BDPC) web server–a useful tool for DNA methylation analysis. *Nucleic Acids Research*, 36(5):e34, 2008.

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2

C. Rohde, Y. Zhang, R. Reinhardt, and A. Jeltsch. BISMA – fast and accurate bisulfite sequencing data analysis of individual clones from unique and repetitive sequences. *BMC Bioinformatics*, 11(1):230, 2010. ISSN 1471-2105.

K. H. Taylor, R. S. Kramer, J. W. Davis, J. Guo, D. J. Duff, D. Xu, C. W. Caldwell, and H. Shi. Ultradeep bisulfite sequencing analysis of DNA methylation patterns in multiple gene promoters by 454 sequencing. *Cancer Research*, 67(18):8511–8518, 2007.

V. Vacic, H. Jin, J.-K. Zhu, and S. Lonardi. A probabilistic method for small RNA flowgram matching. *Pacific Symposium on Biocomputing*, pages 75–86, 2008.

M. Zeschnigk, M. Martin, G. Betzl, A. Kalbe, C. Sirsch, K. Buiting, S. Gross, E. Fritzilas, B. Frey, S. Rahmann, and B. Horsthemke. Massive parallel bisulfite sequencing of CG-rich DNA fragments reveals that methylation of many X-chromosomal CpG islands in female blood DNA is incomplete. *Human Molecular Genetics*, 18(8):1439–1448, 2009.

PeerJ PrePrints | https://peerj.com/preprints/122v2/ | v2 received: 29 Nov 2013, published: 29 Nov 2013, doi: 10.7287/peerj.preprints.122v2