Development of forensic genomics research toolkits by the use of Massively Parallel Sequencing

Gaag, K.J. van der

Chapter 5

FDSTools: A software package for analysis of massively parallel sequencing data with the ability to recognise and correct STR stutter and other PCR or sequencing noise

Forensic Science Intenational Genetics 2017 Mar;27:27-40
Published online November 26, 2016

Jerry Hoogenboom
Kristiaan J van der Gaag
Rick H de Leeuw
Titia Sijen
Peter de Knijff
Jeroen F.J. Laros

Supplementary material is available via https://www.fsigenetics.com

Abstract

Massively parallel sequencing (MPS) is on the advent of a broad scale application in forensic research and casework. The improved capabilities to analyse evidentiary traces representing unbalanced mixtures is often mentioned as one of the major advantages of this technique. However, most of the available software packages that analyse forensic short tandem repeat (STR) sequencing data are not well suited for high throughput analysis of such mixed traces. The largest challenge is the presence of stutter artefacts in STR amplifications, which are not readily discerned from minor contributions. FDSTools is an open-source software solution developed for this purpose. The level of stutter formation is influenced by various aspects of the sequence, such as the length of the longest uninterrupted stretch occurring in an STR. When MPS is used, STRs are evaluated as sequence variants that each have particular stutter characteristics which can be precisely determined. FDSTools uses a database of reference samples to determine stutter and other systemic PCR or sequencing artefacts for each individual allele. In addition, stutter models are created for each repeating element in order to predict stutter artefacts for alleles that are not included in the reference set. This information is subsequently used to recognise and compensate for the noise in a sequence profile. The result is a better representation of the true composition of a sample. Using Promega Powerseq™ Auto System data from 450 reference samples and 31 two-person mixtures, we show that the FDSTools correction module decreases stutter ratios above 20% to below 3%. Consequently, much lower levels of contributions in the mixed traces are detected. FDSTools contains modules to visualise the data in an interactive format allowing users to filter data with their own preferred thresholds.

Introduction

Analysis of Short Tandem Repeats (STRs) has been a successful forensic tool in the past two decades. The comparison of STR profiles from forensic DNA evidentiary traces with reference samples and DNA databases has provided essential information in many forensic cases. [1] Standard practice is to use Capillary Electrophoresis (CE) to analyse STR length variation. In recent years, Massively Parallel Sequencing (MPS) was introduced as a new method to analyse STRs and other forensic DNA markers [2,3]. MPS enables the simultaneous detection of both length and sequence variation of STRs, which increases the discriminatory value substantially [4,5,6]. The output of CE consists of peaks reflecting fluorescent signal intensities with their own respective shapes and peak heights. The output of MPS data analysis consists simply of read counts of the observed sequences. Both methods can suffer from the occurrence of PCR artefacts such as STR stutters [7]. This especially complicates the analysis of STR profiles coming from multiple contributors, which is common in forensic evidentiary traces. [8] The

level of stutter formation depends on a number of distinct aspects of the sequence, including the A/T content of the repeat unit and the number of consecutive repeat units occurring in an STR [9]. Since any specific STR length identified by CE can consist of multiple different sequences, these CE-identified length variants show a larger variation in measured stutter percentage than individual sequences analysed through MPS. This decreased variation in stutter percentage for MPS STR data may aid in the interpretation of mixtures [2], as it allows for a better prediction of stutter behaviour, which can be used to filter the data for stutter products. Existing software packages for the analysis of STR sequencing data [10,11,12] do not support extensive filtering and correction of systemic PCR and/or sequencing errors and therefore seem less suited for analysis of mixed DNA samples. This prompted us to develop a software package that harbours the following features: 1) characterisation and correction of noise in the sequencing data caused by PCR stutter or other systemic PCR and/or sequencing errors; 2) visualisation of sequencing data as comprehensive profiles; 3) filtering of data in graphs and tables with user definable thresholds and 4) open-source accessibility. Forensic DNA Sequencing Tools (FDSTools) is available via the Python Package Index (either by manual installation or by using the command 'pip install fdstools'). We assess the performance of FDSTools on 31 two-person mixtures genotyped via the Promega Powerseq™ Auto System for which we first generated a reference dataset of 450 samples.

Material and Methods

Sample preparation

PCR products and sequencing libraries were prepared as described previously [2] using a prototype Promega Powerseq™ Auto System containing 23 STRs and amelogenin. A set of 450 Dutch samples [13] and 31 two-person mixtures were amplified and sequenced. The mixtures consisted of three combinations of two donors selected randomly from a pool of unrelated individuals, which were mixed in different ratios. The minor components in the mixtures contributed 0.5% (six mixtures), 1% (six mixtures), 5% (four mixtures), 10% (six mixtures), 20% (six mixtures) and 50% (three mixtures).

Since the mixtures were used to test the performance of the software and also to determine analysis thresholds that are fit for purpose, we balanced the influence of varying DNA inputs in the PCR and increased drop-out due to low DNA input. This was achieved by the use of a minimum of the minor component of 60 pg in the 0.5%, 1% and 5% mixtures, resulting in a total DNA input of 12 ng, 6 ng and 1.2 ng, respectively (60 pg resulted in less than 20% drop-out in the validation of Powerplex 6C [14]). The same total DNA input of 1.2 ng was used for the 5%, 10%, 20% and

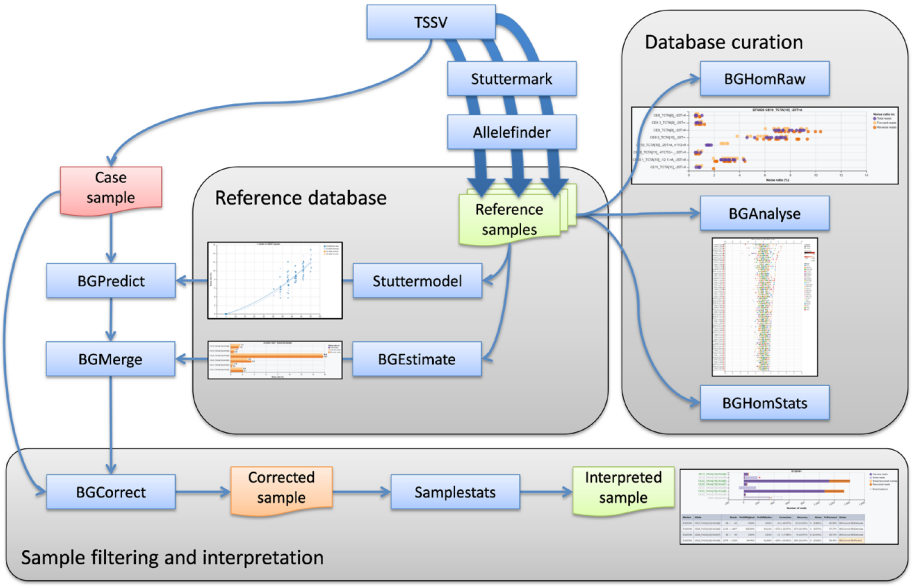
50% mixtures (resulting in 120 pg and 240 pg of the minor components in the 10% and 20% mixtures, respectively). The DNA input was 0.5 ng for single donor samples.

The genotypes of the donors used in the mixtures were known, which enables the identification of drop-in and drop-out allele calls. Paired-end sequencing data of all amplicons was generated using the MiSeq® Sequencer (Illumina).

Initial data processing

In Figure 1, the main tools of the FDSTools package and their role in the data analysis pipeline are displayed. The tools can be split into three functional groups: tools for reference database creation, tools for reference database curation (data quality assessment) and tools for case sample filtering and data interpretation. In addition, the package contains initial data processing tools such as TSSV [10] that are common to reference database samples and case samples.

Figure 1. Flow chart of the analysis process, showing the main tools of FDSTools



Flow chart showing the main tools (blue rectangles) of the FDSTools package and their roles in the data analysis pipeline. The output of each tool can be visualised using the Vis tool (not shown).

Chapter 5

Paired-end read merging

Using paired-end sequencing, forward and reverse strand molecules of each amplicon were sequenced from both ends. The first ~300 nucleotides from either end were obtained. These read pairs were merged into a consensus read by aligning the read pair such that the largest possible overlap is obtained while allowing for up to 33% mismatches in the overlapped region. Most amplicons were about 300 base pairs in length and provided fully complementary read pairs.

With STR amplicons that are longer than 300 bp, a problem may occur when both reads end in the middle of the STR structure and the pair may be merged into a truncated STR sequence. A modified version of FLASH 1.2.11 [15] (available via github.com/Jerrythafast/FLASH-lowercase-overhang) was used to mark the bases that were not in the overlapped region in lower case in the consensus read. This enables detection of truncated STR sequences in downstream analysis.

Linking reads to loci and alleles

The merged reads are linked to specific loci and alleles by the TSSV tool, which is a wrapper around a simplified version of the TSSV [10] program called TSSV-Lite. TSSV links reads to loci by scanning the reads for the sequences flanking the STR loci used. The flanking sequences of each locus, that usually represent the most 5' nucleotides of the primers, are provided to FDSTools in a library file, together with various other details about the loci used. Supplementary File 1 represents the library file used in this study. The file contains a description for the contents of each section.

Each read is scanned for these flanking sequences by computing alignments. In this study, the flanking sequences were 18 nucleotides in length and two substitutions (or two inserted or deleted bases) per flank were allowed in the alignment. Reads are categorised as 'unrecognised' if no flanking sequence is found. Furthermore, both flanking sequences are required to have at least one upper case letter, which ensures that overlapped reads that are potentially truncated are categorised as 'unrecognised' as well. Reads in which only one flanking sequence is found with at least one upper case letter get linked to a locus but flagged as 'no start' or 'no end' depending on whether the left or right flank is missing, respectively (optionally, these reads can be written to separate fasta or fastq files).

The main output of TSSV is a text file with tab-separated values. The file contains one line for every unique sequence of each locus. The columns include the name of the locus, the sequence, and the number of reads carrying this particular sequence. Read counts are given separately for the forward and reverse strand.

TSSV includes additional options for filtering sequences that are seen too few times and sequences with a length outside a given range (e.g., primer-dimers). This range can be specified separately for each locus. Furthermore, filtered sequences can be aggregated into a single 'other sequences' category for each locus. In this project, only singletons (i.e., sequences with only one read) were aggregated to the 'other sequences' category.

Building a reference database

One function of FDSTools is the building of a reference database. Such a database can be used to obtain estimates of recurring allele-specific systemic noise. Here, 'noise' refers to the complete collection of sequences observed in a sample, except the sample's true allelic sequences. Noise includes any artefact deriving from the PCR as well as the sequencing (such as PCR stutter or single-nucleotide errors). Additionally, based on the reference data a statistical model can be derived that aims to predict stutter ratios for alleles not present in the reference set.

The creation of a reference database involves various tools included in the FDSTools package, which will be discussed in the next sections. In addition to these separate tools, FDSTools offers the Pipeline tool, which conveniently integrates the entire data analysis pipeline. Users are advised to use Pipeline as it removes the complexity of having to run several separate tools and to combine their output. Pipeline takes a simple configuration file containing the analysis parameters and automatically runs the appropriate tools.

Building a reference database is a two-phase process. In the first phase, the reference samples are analysed in a global manner to identify their alleles and reject those samples in which the alleles are not readily identified. In the second phase, the systemic noise of each of these alleles is analysed in detail.

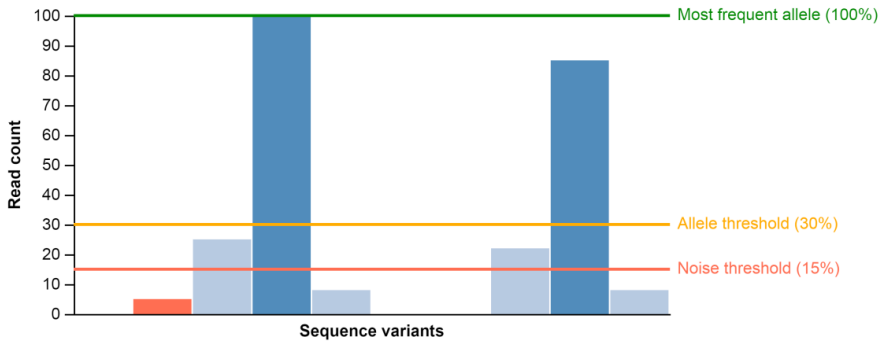
Allele calling for reference samples

Determining the alleles of single donor reference samples is a fairly straightforward process because these generally represent the one or two most abundant sequences for any locus. FDSTools includes Allelefinder to call alleles this way. It is applied after Stuttermark, which is described below. A number of thresholds are used to guard against including alleles of potential low-level contaminations, which are outlined in Figure 2. For heterozygous loci, a second allele is only called if it passes the allele threshold, which is defined in this project as 30% of the read count of the most frequent allele at the same locus. As we expect no stutter above 30% [2], this threshold separates the alleles from noise. No alleles are called at a locus when additional sequences occur that have a read count below the allele threshold but above the noise threshold (which is

Chapter 5

defined as 15% of the most frequent allele in this project) or if a third sequence passes the allele threshold. If more than two loci in the same sample fail to give a result for these reasons, the overall quality of the sample is considered too poor to report any alleles. Additionally, Allelefinder can be configured to call at most one allele at haploid loci.

Figure 2. Thresholds used by Allelefinder to call alleles in reference samples



Sequence variants with a read count above the allele threshold are called as alleles. The four lighter-shaded bars represent stutter variants (as recognised by Stuttermark), which are ignored by Allelefinder.

The three potential pitfalls are 1) PCR stutter artefacts that exceed the noise threshold; 2) strong read count imbalance for heterozygous alleles, which may be the result of e.g., primer-site sequence variants and 3) autosomal trisomy, which is rare. To deal with the problem of stutter, each sample was analysed with Stuttermark [2] before calling alleles. With Stuttermark, sequences that are in a stutter position of another sequence while having a read count below a user-supplied percentage with respect to the other sequence are marked as 'stutter'. Sequences that have a read count that is too high to be explained by stutter alone will not be marked as 'stutter', as they may coincide with a genuine allele. The thresholds used here were 30% for −1 stutter (loss of a repeat unit) and 10% for +1 stutter (gain of a repeat unit). For −2 stutter products, a 30% threshold of the −1 stutter product is used. Sequences that are marked as 'stutter' are completely ignored by Allelefinder.

Allelefinder produces the list of alleles and a report detailing for which samples and loci allele calling is rejected and for which reasons.

Estimating average allele-specific systemic noise

For each allele, a profile of recurring systemic noise, including PCR stutter products as well as any other 'side products', can be generated based on the reference data.

Noise profiles are always computed separately for forward and reverse reads, because strand bias may exist in the sequencing technology used. Profiles are also computed separately for each locus, under the assumption that noise production is not influenced by alleles of other loci. The level of noise is expressed as the number of noise reads as a percentage of the number of reads of the parent allele. In the context of PCR stutter analysis, this quantity is often referred to as the 'stutter ratio', despite the representation as a percentage of the parent allele. We use the generalised term 'noise ratio' (also represented as a percentage of the parent allele) to account for all other systemic noise as well.

$$\text{Noise ratio} = \frac{\text{Noise reads}}{\text{Allele reads}} \times 100\%$$

In homozygous samples, the noise ratio can be calculated by dividing the number of reads of a non-allelic sequence by the number of reads of the allele. Allele-specific noise profiles are readily computed from homozygous samples carrying this allele by scaling the read counts in each sample such that the parent allele is 100 and averaging the noise ratios for each noise sequence. These per-allele noise statistics and other statistics, such as the standard deviations of the noise ratios can be obtained using BGHomStats. In heterozygous samples the extraction of noise sequences is more complex, because it has to be determined which proportions each allele contributed to the observed noise sequences. We assume that noise in heterozygous samples corresponds to the sum of the noise profiles of the two alleles, after the application of a scaling correction to account for differences in the amount of each allele amplified. This is needed as even for heterozygous allele pairs, PCR efficiency may vary due to primer binding site sequence variation or STR length. [16] To extract noise from heterozygous reference samples an iterative approach was taken and implemented in the BGEstimate tool in FDSTools.

In essence, the algorithm, which is discussed in more detail in Supplementary Text 1, seeks a non-negative least squares solution to the matrix equation $A\,P = C$. In this equation, C is an $N \times M$ matrix of constants derived from the read counts in the reference samples, A is an $N \times N$ matrix summarising the allele balance in the samples, and P is an $N \times M$ matrix containing the estimated profiles of systemic noise. N is the number of unique genuine alleles among the reference samples and thus also the number of profiles produced and M is the total number of unique sequences observed.

Matrix C is computed once at the start of the algorithm. Each row in C corresponds to one allele and contains the sum of the read counts of all samples that have that particular allele, after scaling the allele to 100 reads for homozygous samples and 50 reads for heterozygotes. The noise profiles in P are initialised with the assumption that no systemic noise is present, i.e., all elements are set to 0, except for the elements that correspond to the actual alleles, which are set to 100.

Chapter 5

The algorithm then proceeds by repeatedly re-estimating the allele balance matrix A while reading cross-contributions between the alleles from the current profiles P and subsequently re-estimating P by finding a non-negative least squares optimal solution to A P = C. The values thus obtained in P are the average noise ratios of all observed systemic noise for all alleles (i.e., each row in P contains the noise profile of one allele).

To avoid noise from one allele being incorporated in the noise profile of another allele, a minimum of three different heterozygous genotypes per allele was used in this study. A threshold can be set for the minimal read count of noise to consider and the minimal percentage (we used 80%) of reference samples with the same allele which should contain the same noise before it is included in the noise profile. Each of these parameters can be set using various options of the 'fdstools bgestimate' command.

Relating the amount of stutter to repeat length

With the methods outlined above, profiles of systemic noise were obtained for each allele present in the reference set. However, one would also like to be able to filter and correct the noise originating from alleles that are not (yet) included in the reference set, as case samples may be encountered that contain alleles for which no reference sample was available. For this purpose, we developed a method to predict the sequence and corresponding amount of PCR stutter artefacts that would be produced for any allele of a given locus. Note that this method does not predict noise other than noise resulting from STR stutter or single nucleotide stretches.

Previous studies have shown that the amount of stutter is strongly correlated with the length of the repeated sequence [17] and even more so with the number of consecutive repeat units [2,18]. The FDSTools tool Stuttermodel seeks to fit polynomial functions to the repeat length and stutter ratio in homozygous reference samples. Stuttermodel scans each of the alleles for all positions where a particular repeat unit (e.g., the sequence 'AGAT') is repeated and records the length of this repeat, as the number of nucleotides, including incomplete repeats at the beginning or end of the repeated stretch. For each sample with this allele, the number of noise reads that lack exactly one repeat is counted. Reads that combine the loss of one repeat with one or more other differences (e.g., substitutions, or stutter in another stretch of repeats in the same allele) are included in this count. The counts thus obtained are used to compute the noise ratios of individual stutter sites and a polynomial function is fitted to quantify the relationship between the length of the repeat and the stutter ratio.

This analysis is repeated for each unique repeat unit of a length between one and a configurable maximum number of nucleotides (inclusive), treating cyclically equivalent units (e.g., 'ATAG' and 'AGAT') and their respective reverse complements (e.g., 'CTAT' and 'ATCT') synonymously. The amount of +1 stutter, −2 stutter etc. is analysed the same way.

Because different loci behave different in stutter formation, a separate function is fitted for each locus. Additionally, a polynomial function is fitted to all data at once,

which is used to predict stutter in alleles of loci for which insufficient reference data was available to fit a locus-specific function. Separate functions are fitted for the forward and reverse strands.

For each fitted function, Stuttermodel also determines the lower bound of the repeat length for which the function gives meaningful results. This lower bound is defined as the lowest repeat length for which the function produces a nonnegative result and the function is non-decreasing. Below this threshold, and in any other points where the function value would be negative, the function value is set to zero.

The quality of fit is assessed by computing the coefficient of determination,

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$$

where

$$\hat{y}_i = \begin{cases} f_i & f_i \geq 0 \\ 0 & f_i < 0 \end{cases}$$

with y_i the noise ratios of the reference samples, $\bar{y}$ the mean, f_i the polynomial function's estimate of the noise ratio of sample i, and $\hat{y}_i$ the modified function value. The R^2 score will be close to one when the function is a good fit and lower otherwise.

Stuttermodel supports fitting polynomial functions of any degree. To prevent over-fitting while still allowing a non-linear relationship, second-degree polynomials (with a minimum R^2 score) were used. In cases where the fit for one strand has an R^2 score above the threshold while the fit for the other strand scores below the threshold, both fits are rejected to prevent unintended introduction of strand bias by filtering stutter on only one strand.

Curating the reference database

To make sure all reference samples were of good quality and all alleles were called correctly, they were put through the same analysis pipeline as case samples, thereby performing noise filtering and correction on the reference samples. It is important to note that these reference samples were previously genotyped by us in great detail using CE [13]. The remaining amounts of noise in each sample were assessed using BGAnalyse (described below) to identify potentially unsuitable reference samples that still passed the thresholds of Allelefinder. Any sample with a notably higher amount of remaining background was manually removed from the set of reference samples to prevent pollution of the noise profiles.

BGAnalyse was developed and employed to analyse the remaining noise after

Chapter 5

correction. For each locus and each sample, this tool calculates the least frequent (this can be a negative value because of over-correction), most frequent, and total noise as a percentage of the number of reads of the highest allele at each locus. These results are subsequently visualised to easily identify potentially problematic samples. In the visualisation, samples can be sorted by any of the calculated values or by coverage (total number of reads). Samples were subjected to manual inspection and any sample that exhibited non-stutter products with corrected read counts above 4% of the most frequent allele or above 2% of the total reads was rejected.

Analysing case samples

The analysis of mock case samples was performed in a three-step process which is described in the following sections.

1. A prediction was made for the amount of stutter for each sequence in the sample, using the fitted polynomial functions obtained from running Stuttermodel on the reference samples. These predictions are used to extend the allele-specific noise profiles obtained from running BGEstimate on the reference samples.
2. The extracted noise profiles are used to filter and correct the noise in the case sample.
3. Alleles are called and the sample is subjected to manual interpretation.

Similar to the creation of a reference database, analysing case samples involves multiple tools discussed in the following sections. Pipeline offers a convenient way to automatically analyse a case sample with all tools discussed.

Predicting stutter amounts for unknown alleles

Because case samples may contain alleles that are not present in the reference samples, noise profiles for these alleles need to be predicted. FDSTools includes the BGPredict tool, which uses a previously created Stuttermodel file to predict the amounts of stutter artefacts for alleles not present in the reference data. BGPredict finds all sequences in the analysed case sample in which a particular repeat unit is repeated. The expected amount of stutter in this repeat is then computed using the corresponding fitted polynomial function from the Stuttermodel file. All possible combinations of stutter are taken into consideration when the frequencies of each stutter artefact are computed. The noise profiles created in this way are used to extend the noise profiles in the previously created BGEstimate file (a tool called BGMerge is included in FDSTools for this purpose).

Noise filtering and correction in case samples

To be able to filter systemic noise in case samples, one first needs to determine which alleles are likely present in the sample. To this end, the algorithm of BGEstimate is essentially reversed, i.e., the goal is now to solve for **a** in $\mathbf{a}\,\mathbf{P} = \mathbf{c}$, where c is a row vector with the sample's read counts for the *M* sequences in the noise profiles and **a** is a row vector with the estimated amount of each of the *N* profiles in the sample. **P** is the $N \times M$ matrix of noise profiles obtained from BGEstimate, extended with the predictions obtained from BGPredict. Solving for **a** is done in a non-negative least squares sense as before, giving estimated allele contributions that best fit the various sequences – alleles as well as noise – present in the sample.

Background-corrected read counts can then be computed by first subtracting the scaled profiles from the sample's read counts

$$\mathbf{d} \leftarrow \mathbf{c} - \mathbf{a}\,\mathbf{P}$$

and then adding the total size of each profile to the corresponding allele, i.e.,

$$\mathbf{d}_n \leftarrow \mathbf{d}_n + \mathbf{a}_n \sum_{m=1}^{M} \mathbf{P}_{n,m}, \quad \forall n \in [1 \ldots M]$$

Note that **d** may have negative elements if the sample contains a lower amount of a certain sequence than was predicted by the profiles of its dominant alleles.

FDSTools offers *BGCorrect* to filter and correct background noise following the procedure outlined above. Given a sample data file (obtained from *TSSV* for example) and a file containing noise profiles, *BGCorrect* produces a copy of the sample data with additional columns giving the amounts of each sequence attributed to noise and the amounts of each sequence that would be recovered by noise correction (i.e., adding the noise to the originating allele). These values are given separately for the forward and reverse strand. Although the method by which *BGCorrect* computes them results in non-integer values, it was decided not to round these numbers to avoid unnecessary loss of precision. If necessary, these numbers can be rounded to integer values, thereby easing the interpretation as 'read counts' when presented in a graph or table in a report.

Allele calling for case samples

The naïve method of calling alleles that *Allelefinder* uses is not appropriate for case samples, since these may contain alleles of multiple contributors in different quantities. Therefore, calling alleles in case samples is done by computing various statistics based on the information of the detected sequences and subsequently setting interpretation thresholds on these statistics. For this, *Samplestats* was developed, which operates on

Chapter 5

and adds various columns to the output of *BGCorrect*. *Samplestats* automatically marks sequences as 'allele' using the thresholds outlined in Table 1.

Alleles can also be called while visualising the sample data, hence, FDSTools includes the *Samplevis* visualisation. By means of the interactive graphical user interface of *Samplevis*, the same set of thresholds as depicted in Table 1 are available to filter the visible sequences and to automatically call alleles. Thresholds can be specified separately for the graphs and for the tables. While the table displays the called alleles, less conservative settings may be used for the filtering of the corresponding graph to ensure visibility of alleles just below the allele-calling threshold. The results of changing the thresholds are immediately visible. Clicking a sequence in any of the graphs toggles its 'allele' status. This allows the user to manually add alleles to and remove alleles from the profile. A note is added to manually added alleles, stating that the allele is 'User-added'. Similarly, if the user removes any alleles, the allele remains visible but a 'User-removed' note is added. In this way it remains easy to trace back exactly which alleles meet the thresholds and which ones were manually added and removed.

Samplestats can also be used to filter sequences using the same types of thresholds (albeit with more stringent threshold values than used for allele calling, as potential alleles should not be filtered out) and (optionally) aggregate the filtered sequences per locus to a single line categorised 'other sequences'.

Table 1. Interpretation thresholds for case samples in Samplestats and Samplevis.

Threshold	Description	Allele calling default	Filtering default
Total reads	Minimum number of reads per allele. Non-systemic (and thus unfilterable) sequence errors occur sporadically. This threshold ensures that a minimal amount of amplified product is present to support the allele call.	30	5
Reads per strand	Minimum number of reads per allele for both strands. This threshold can be used to exclude low template sequences with strong strand bias.	1	0
Percentage of most frequent	The number of reads as a percentage of the number of reads of the most frequent allele at the locus. This threshold sets a limit to the mixture proportions that can be analysed in mixed samples or to the allele balance in samples with a single contributor.	2%	0.5%
Percentage of locus	Each allele contributes at least this percentage to the total number of reads of the locus. With this threshold, a minimum contribution percentage can be enforced.	1.5%	0%
Percentage correction	This percentage derives from the number of reads after noise correction minus the number of reads before correction, which is divided by the number of reads before correction. Consequently, the percentage correction is negative if noise correction resulted in a reduction of the read count of a sequence. Therefore, with this threshold set to 0%, any sequence representing noise will not be called as an allele. To be able to detect alleles of minor contributors that coincide with noise products for the major contributor's alleles, the 'percentage recovery' threshold described below is allowed to overrule this threshold.	0%	0%
Percentage recovery	The number of reads added by noise correction as a percentage of the total number of reads after noise correction. After noise correction, at least this percentage of reads must have originated from corrected noise. The rationale behind this threshold is that only allelic sequences will have substantial amounts of recovered reads. When an allele of a minor contribution coincides with the stutter of an allele of the major contributor, noise will be extracted and added to the major contributor's parent allele resulting in a negative percentage correction. Yet, since the minor contributor's contribution to the reads also results in noise products that are corrected, the allele will receive recovered reads and a percentage recovery >0%. To allow the calling of alleles for which no noise profile exists (or no noise was detected) in the reference database the threshold is set at 0% by default.	0%	0%

Sequences that meet either the 'Percentage correction' or 'Percentage recovery' threshold (or both) as well as all the other thresholds will be marked as 'allele'. These threshold values are evaluated after noise correction. The 'Allele calling default' column lists the default threshold values for calling alleles. The 'Filtering default' column lists the default values used for filtering displayed sequences in Samplevis graphs.

Chapter 5

Visualisation

For visualisation of the data, FDSTools makes use of the JavaScript graphing library Vega [19]. Vega graphs can be embedded on a web page, exposing a JavaScript programming interface that allows for updating the graphs based on the user's interaction with the web page. Vega can also run on Node.js, which allows it to be included in automated analysis pipelines to generate (static) image files.

FDSTools comes with Vega graph specifications and accompanying interactive web pages (HTML files) to visualise the output of each tool. The Vis tool can be used to obtain self-contained HTML files containing visualisations of various types of data files generated by the other tools. For example, Samplevis visualises a sample data file as a sequence profile and Profilevis visualises background noise profiles obtained from BGEstimate or BGPredict. A description of each visualisation can be found in Supplementary Table 1. When viewed in a web browser, the web page provides additional controls that allow the user to filter the data, switch between linear and logarithmic scales, or select different subsets of the data to visualise. The default values for the settings on the web page can be set when the HTML file is generated by the Vis tool.

The web pages also offer the option to save the displayed graphs as a Scalable Vector Graphics (SVG) or rasterised Portable Network Graphics (PNG) image, so that they can be imported into documents. Alternatively, the Vis tool can supply a raw Vega graph specification file (either with or without embedded data), which can then be used by Vega to generate SVG or PNG images directly on the command line.

Results and discussion

We developed FDSTools, a software package containing a suite of tools that can be used for the analysis of forensic MPS data. With these tools, FDSTools provides detailed insight in the quality of a sample and the noise profile of a certain allele (or sequence variant). In Supplementary Table 1, an overview of all tools currently available in the package is provided, of which a selection was described in more details in Section 2.

To enhance the analysis of mixed samples, FDSTools identifies, extracts and corrects for PCR or sequencing noise such as stutter from a reference database with the aim to discern low mixture proportions. Different STR amplification assays and different amplification protocols could result in different noise. It is therefore important to base the database for noise correction on references generated by a method that is representable for the casework samples to be analysed.

Note that it is not possible to correct all noise completely as the level of noise shows variation between samples.

Reference database

Our reference samples were sequenced with an average coverage of 65,000 reads and a mode of about 45,000 reads. For the present study, a minimum coverage of 6,000 reads per sample was required, which relates to an average of 250 reads per locus as 24 loci were co-amplified. For heterozygous loci, less than 250 reads per locus is not sufficient to quantify low amounts of noise accurately.

Reference sample curation

Since the reference database is used to filter and correct noise in case samples, it is essential that the reference samples contain no contaminants and reference alleles are called correctly. Although all other steps can be performed automatically by FDSTools, a manual curation of samples in the reference database is needed. BGAnalyse was developed to facilitate this process by visualising potential outliers.

Allelefinder automatically rejected two out of the initial 450 samples which were clearly contaminated and three samples that had too low coverage to detect alleles reliably. Manual inspection of samples with a notably higher amount of remaining noise after correction in BGAnalyse resulted in the rejection of an additional 16 samples. Reasons for rejection were low-level contamination, low coverage and low sequencing quality. The interactive BGAnalyse visualisations displaying the remaining noise for the reference samples are available in Supplementary File 2a (before database curation) and 2b (after curation). For the majority of samples, the highest remaining noise variant in the complete profile did not exceed 3% of the number of reads of the highest allele at the locus while without correction STR stutters can represent over 20%. For the remaining 429 samples, no drop-in or drop-out was observed when calling alleles using Allelefinder with the settings described in Section 2.3.1.

Extending noise profiles for noise correction

As described in Section 2.4.1, case samples may contain alleles which are not present in the reference database. In such cases, FDSTools resorts to noise prediction instead of noise estimation. A column in the output file of *BGCorrect* marks if correction has been performed using data obtained from *BGEstimate* (if the allele was available in the reference database) or by using *BGPredict* (if not available in the reference database).

From the results from *Stuttermodel* it becomes evident that for simple STRs consisting of a single repeating element or for long stretches of a specific repeating element within a complex STR, only few reference samples are needed to reliably fit a stutter model. However, when complex repeats consist of several repeating elements of which

some show little length variation, correction using the stutter model is suboptimal as exemplified by the predictions for D12S391. This STR locus consists of two repeat units; an AGAT repeat stretch of highly variable length and an ACAG repeat that is repeated 6 to 8 times for most individuals. Since *Stuttermodel* predicts the amount of stutter based on the repeat length, at least four different repeat lengths need to be available in homozygous reference samples to obtain a reliable fit. However, the set of reference samples used in this study only contained homozygotes with 6 to 8 repeats of ACAG, which is not sufficiently variable to obtain a reliable fit. Consequently, ACAG is omitted from the stutter model for D12S391, even though this repeat stutters up to 9% for the longer repeats (8 repeat units, data not shown). When *BGEstimate* does not obtain a background noise profile, *BGPredict* will not correct stutter in this repeat and thus stutters will remain present. As a last resort, *BGPredict* offers the possibility to use a stutter model based on data from all loci that have the same repeat unit sequence if no locus-specific fit is available. Supplementary Figure 1 displays the stutter model obtained from the set of 429 reference samples, including the individual observations on which the model was based.

Combining *BGEstimate* and *BGPredict* (by using *BGMerge*) instead of using *BGPredict* alone is expected to reduce the noise remaining after correction, as the combined correction also corrects for noise other than stutters . This is confirmed when we determine the percentage of remaining noise (the reads representing remaining noise as a percentage of the reads for the most frequent allele at the locus) and plot the highest percentage and various percentiles (90^{th}, 95^{th} and 99^{th}) (Supplementary Figure 2a–b). The percentiles illustrate how often samples exhibit outlying noise sequence variants and when the 99th percentile is regarded, *BGPredict* alone retains on average 2.6% noise and the combined correction 2.4%. Also, the combined correction results in less overcorrected variants.

Thus, *BGPredict* can be used without *BGEstimate* with a slightly reduced accuracy in correction. Note that *BGEstimate* should not be used without *BGPredict* since alleles not included in the reference database will not be corrected, which can result in a combination of corrected and uncorrected alleles and remaining noise for the uncorrected alleles.

Reference database size and coverage

To test the effect of the sample size and type from which the reference database is built, we used the complete curated reference database of 429 samples and a random selection of 100 samples (both with combined *BGEstimate* and *BGPredict* correction, which was found to be slightly better as described in Section 3.1.2). Supplementary

Figures 2c–d display an overview of the most frequent and the total remaining noise at each locus after correction. The different percentiles of the reference samples are given to illustrate how often samples exhibit outlying noise sequence variants.

When comparing the results for the complete database with the results for the subset of 100 samples, the difference in remaining noise seems surprisingly small (Supplementary Figure 2c–d). However, with a smaller database, less alleles will fit the criteria to create a *BGEstimate* noise profile and more alleles rely on noise prediction by *BGPredict*. Indeed, for the reference set of 429 samples, only 3.5% of the alleles are corrected using *BGPredict*. This percentage increases to 10.2% when the correction is based on the subset of 100 samples.

In a larger reference database more alleles will be observed. Supplementary Figure 3 displays the alleles observed in the reference databases of 429 and 100 samples. To fit the criteria to create a *BGEstimate* noise profile, alleles need to be present as a homozygous genotype or be available as part of shared genotypes with at least three other alleles that must also fit these criteria. For the stutter model, only the homozygous genotypes are used. In the larger 429 database, more alleles fit these criteria than in the smaller 100 sample set database.

To examine the effect of read coverage of the reference samples on noise profile analysis, we generated two subsets comprising samples with high or low coverage, which is specified as a total read count between 82,000 and 350,000 or 8,000 and 44,000 respectively. The high coverage set comprised 71 samples; the low coverage set 70. We noticed that in the low-coverage noise profiles, strand bias can occur especially for the low-percentage noise that is due to single-strand drop-out of this noise. This is illustrated by the *BGEstimate* noise profiles for the CE10_TCTA[10]_-20T>A allele for locus D7S820 in Supplementary Figure 4, in which forward and reverse reads are in good or reasonable balance for all seven noise sequences in the high coverage sample set while good balance is only seen for the two main noise sequences in the low coverage set.

Since the most abundant noise after correction in a sample is usually in the range of 0.5–3% (for STR analysis), we recommend a coverage of at least 1,000 reads per locus (which relates to a 24,000 total read coverage for our 24 loci amplification kit) for the samples of the reference database to obtain the most accurate noise estimates.

Infrequent alleles

Depending on the composition of the reference database, occasionally alleles will be encountered that are not included in the database. *BGPredict* can predict the noise from stutter or other repeating elements but correction of other types of noise (like low level SNPs caused by sequence errors) is not possible for these infrequent alleles.

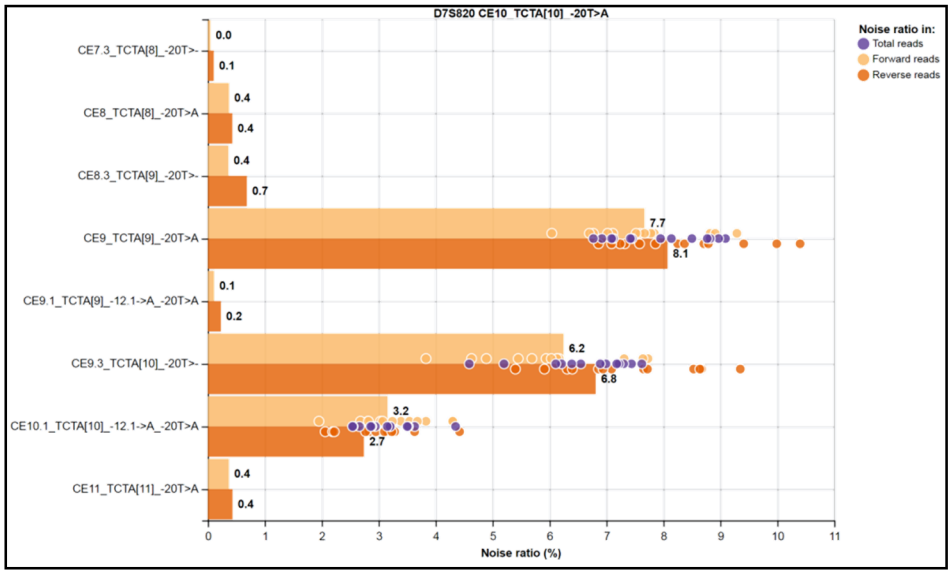
We therefore recommend to obtain *BGEstimate* noise profiles for as many alleles as possible, while retaining good quality of these noise profiles. Several filtering criteria can be applied, such as the minimum number of different heterozygous genotypes per allele, the minimum number of samples per allele and the minimum number of homozygous samples per allele. The effect of increasing the stringency on the filtering criteria on the number of retrieved *BGEstimate* noise profiles for our 429 reference set is shown in Supplementary Table 2. The settings selected for use in this study are: at least two samples per allele (which ensures noise is not based on a single sample as that could be an outlier) that present at least three different heterozygous or at least one homozygote genotype (i.e., the samples can be three different heterozygotes or two homozygotes or one homozygote and one heterozygote).

When an allele at a heterozygous locus fails the criteria, the complete locus carrying this allele cannot be used for establishment of a noise profile since the noise cannot be attributed to any of the two alleles. Thus, for both alleles at a heterozygous locus no noise profile is extracted.

Accuracy of noise reference database and stutter model

To verify the accuracy of the noise profiles obtained through *BGEstimate* and *BGPredict*, it can be useful to compare the average noise ratios with the noise ratios observed in individual homozygous samples. The noise ratios of all noise in all homozygous reference samples can easily be collected using the *BGHomRaw* tool. These data points can be plotted on top of a noise profile to inspect the consistency and variation in the noise ratios of various types of noise for each allele. In Figure 3, the noise profile of the most frequent allele of D7S820 (CE10_TCTA[10]_-20T>A) is displayed, which has foremost a –1 stutter (CE10_TCTA[9]_-20T>A) in addition to a –1 nt slippage product at the A-stretch (CE9.3_TCTA[10]_-20T>-). The individual observations for the homozygous samples coincide nicely with the estimated noise profile ratios.

Figure 3. Noise profile of D7S820 allele CE10_TCTA[10]_-20T>A



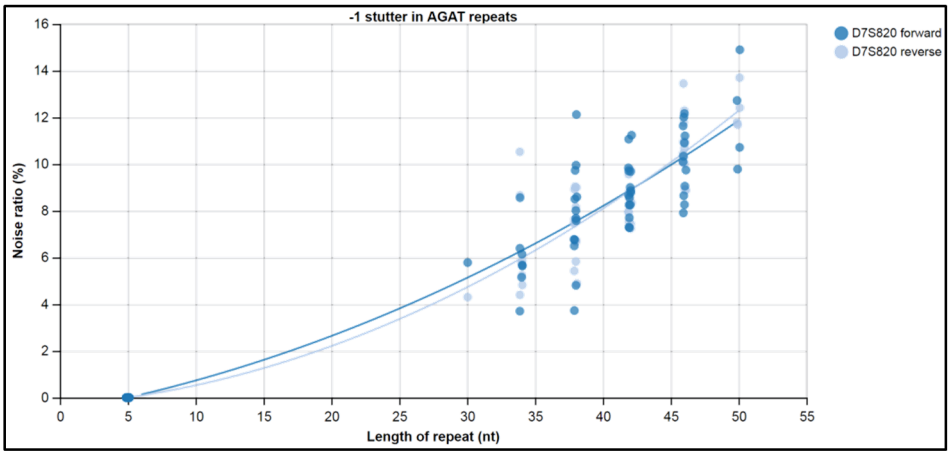
The noise ratio is shown for each systemic noise sequence observed with a noise ratio of 0.1% or higher. Individual observations in homozygous samples (above 0.5%) are displayed as circles. As expected, the most frequently observed noise sequence is the −1 stutter, but since the allele contains a single-nucleotide stretch of 9 A nucleotides, a considerable portion of the noise consists of sequences with slippage at this A-stretch (or a combination of the two).

Similarly, it is useful to compare the functions fitted by *Stuttermodel* to the data points to which they were fitted. *Stuttermodel* includes an option to write the raw data points to a separate output file, which can be visualised together with the fitted model as shown in Figure 4 for D7S820. This example shows that the homozygous calls and the Stuttermodel estimation follow the same trend and that there is no discrepancy between forward and reverse reads. The same holds for the A-stretch (data not shown).

In the stutter model, fits with an R2 score below 0.75 were rejected. Although this may seem a very low R2 score, we obtained better results by including more fits than by excluding them, which would result in the inability of the stutter model to be used to filter and correct stutter for the respective repeat units at all.

Chapter 5

Figure 4. Stutter model for the –1 stutter of D7S820



On the x-axis the length of the repeat is displayed (in nucleotides) and on the y-axis the –1 stutter noise ratio (as percentage of reads of the parent allele) is displayed. Each homozygous reference sample is displayed as a dot and the lines display the fitted functions used for calculating the expected stutter of each allele.

Sample analysis

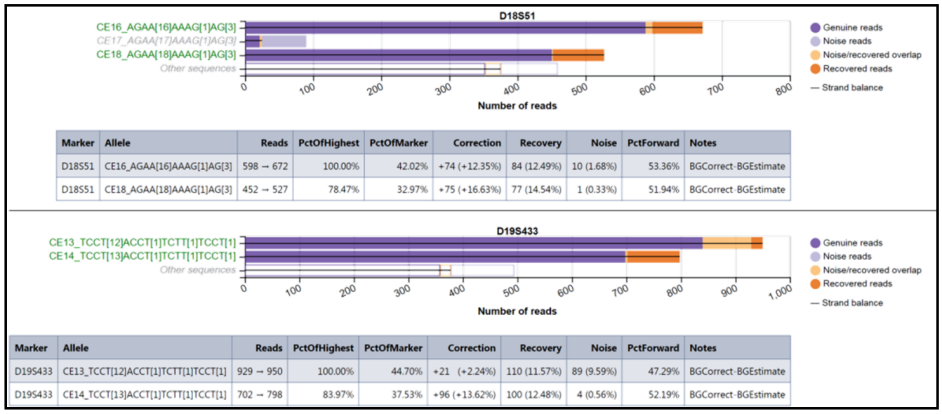
Allele calling, interpretation and visualisation

When a reference database has been created, one can proceed with the analysis of samples. FDSTools analyses sequencing data, calls alleles and interprets the data by correction for noise as inferred from the reference database. Results can be represented as a graphical sequence profile output and as an interactive profile report.

In Figure 5, an example of a sequence profile of two loci of a single-source sample is displayed (generated by the command 'fdstools vis sample'). A sequence profile displays the read counts before and after correction and visualises the effects of noise filtering and noise correction. A more detailed explanation of the interpretation of a sequence profile can be found in Supplementary Figure 5.

The interactive sequence profile reports provide separate filtering options for the graphs and tables displayed (see Section 2.4.3). In the graphs, all alleles that are hidden by the filtering options are (optionally) aggregated as a separate bar (displaying the cumulative numbers of reads) with the label 'other sequences'. In addition, we aggregate all singleton reads into 'other sequences' already in the first step of the analysis (using 'fdstools tssv --minimum 2 --aggregate-filtered') which has the additional benefits of speeding up subsequent analysis and decreasing data storage demand.

Figure 5. Sequence profile of a single-source sample



Marker	Allele	Reads	PctOfHighest	PctOfMarker	Correction	Recovery	Noise	PctForward	Notes
D18S51	CE16_AGAA[16]AAAG[1]AG[3]	598 → 672	100.00%	42.02%	+74 (+12.35%)	84 (12.49%)	10 (1.68%)	53.36%	BGCorrect-BGEstimate
D18S51	CE18_AGAA[18]AAAG[1]AG[3]	452 → 527	78.47%	32.97%	+75 (+16.63%)	77 (14.54%)	1 (0.33%)	51.94%	BGCorrect-BGEstimate

Marker	Allele	Reads	PctOfHighest	PctOfMarker	Correction	Recovery	Noise	PctForward	Notes
D19S433	CE13_TCCT[12]ACCT[1]TCTT[1]TCCT[1]	929 → 950	100.00%	44.70%	+21 (+2.24%)	110 (11.57%)	89 (9.59%)	47.29%	BGCorrect-BGEstimate
D19S433	CE14_TCCT[13]ACCT[1]TCTT[1]TCCT[1]	702 → 798	83.97%	37.53%	+96 (+13.62%)	100 (12.48%)	4 (0.56%)	52.19%	BGCorrect-BGEstimate

Sequence profile of loci D18S51 and D19S433 of a single-source sample. A sequence profile displays the read count before correction (in purple bars) and shows the effects of noise filtering (light purple for the reads that are removed) and noise correction (with the noise reads added to the parent alleles in dark orange). When performing correction, it is possible that an allele gains reads because the noise reads originating from this allele are added, but loses reads at the same time since the noise of another allele in the profile includes reads of this allele. This overlapping part of added and removed reads is marked separately in light orange. This means that the original read count of an allele before correction is the combination of the purple and the light orange bar. The lines in the bars indicate the strand balance; the line is drawn near the top of the bar if the majority of reads of a sequence is on the forward strand, near the bottom of the bar if the majority of reads is on the reverse strand, and in the middle of the bar in the absence of strand bias. Sequences displayed in green in the graphs are the alleles that the software infers to be genuine alleles in the sample. These are also displayed in the table.

Improving heterozygote balance through noise correction

The amplification of long STR alleles in the PCR is generally less efficient than shorter alleles and, in addition, long STR alleles suffer from a higher degree of stutter resulting in reduced heterozygote balance between the two alleles. [2] Since FDSTools determines which 'noise reads' are derived from which parent alleles, these reads can (optionally) be added to the read counts of the parent alleles, which theoretically will improve the heterozygote balance. When we examine the heterozygote allele balance in the 429 single-source reference samples, an improved heterozygote balance is indeed observed when the stutter reads are added to the read counts of the parent alleles (Table 2). Heterozygote balance was determined per locus by dividing the read counts for the less frequent alleles by those for the more frequent alleles, and taking the average of all 429 samples.

Table 2. Heterozygote balance for original, filtered and corrected datasets

Dataset \ locus	Amel	CSF1PO	D10S1248	D12S391	D13S317	D16S539	D18S51	D19S433	D1S1656	D21S11	D22S1045	D2S1338	D2S441	D3S1358	D5S818	D7S820	D8S1179	FGA	PentaD	PentaE	TH01	TPOX	vWA
Uncorrected data	0.83	0.88	0.82	0.79	0.88	0.87	0.85	0.84	0.88	0.89	0.85	0.77	0.90	0.88	0.90	0.88	0.89	0.85	0.87	0.78	0.88	0.88	0.85
Filtered data without noise reads added to allele read count	0.83	0.90	0.87	0.80	0.89	0.89	0.87	0.88	0.89	0.90	0.88	0.78	0.90	0.90	0.91	0.89	0.90	0.87	0.87	0.78	0.88	0.89	0.88
Corrected data with noise reads added to allele read count	0.83	0.91	0.89	0.84	0.90	0.91	0.89	0.90	0.91	0.91	0.93	0.80	0.91	0.91	0.91	0.91	0.91	0.89	0.88	0.81	0.89	0.90	0.90

The read counts for the less frequent alleles are divided by those for the more frequent alleles, and the average for all 429 single-source reference samples is taken.

Mixture analysis

For the analysis mixtures, noise correction may assist in identifying the alleles of a low minor contributor. We used 31 two-person mixtures with minor contributions of 50%, 20%, 10%, 5%, 1% and 0.5% to assess this expectation.

We varied the 'percentage of locus' threshold (Table 1) for calling alleles, which sets a limit to the mixture proportion. When no noise correction was applied the threshold was varied between 5.0% and 1.5%; when noise correction was applied, a lower threshold could be used, varying between 3.0% to 0.5%. We compared the various methods by calculating percentage missed alleles (a.k.a. drop-out) and the number of erroneous allele calls (a.k.a. drop-in). The percentage drop-out was calculated by dividing the number of donor alleles not called by the total number of possible alleles (homozygous and shared alleles are counted as one, Amel is included), and the percentage was averaged for the mixtures with the same mixture ratio. Drop-in is presented in the average number occurring in profiles with the same mixture ratio. In Table 3, the results of these analyses are displayed and it is obvious that without correction more drop-in alleles occur that mostly represent stutters. Consequently, the threshold for calling an allele can be lower when correction is applied, as less stutters remain in the corrected profile that can be wrongfully called as an allele. As expected, the percentage of drop-out depends largely on the 'percentage of locus' threshold for allele calling (and hardly on the application of noise correction); drop-out is more frequent with a higher (more stringent) threshold. When the threshold for corrected data is decreased below 1.5%, the number of drop-ins rapidly increases for all ratios. Not surprisingly, the drop-out percentage for the mixtures with 1% and 0.5% is very high when using a threshold that is higher than the minor component. Therefore, the data from the 5% and 10% minor contribution was used to determine the optimal threshold for allele calling.

In Figure 6, we show the relation between the 'percentage of locus' allele-calling threshold, drop-out and drop-in for the mixtures with a 5% or 10% minor contribution. In the used dataset, a threshold of 1.5% appears to be the most effective for calling

genuine alleles in mixtures with minimal erroneous calling of remaining noise in the mixtures. With mixture ratios down to 10%, no drop-out and only minimal drop-in is observed with this threshold, whereas with contributions smaller than 10% an optimal balance between drop-out and drop-in is achieved (Table 3). When investigating the noise that is erroneously called using this threshold it is apparent that the drop-in alleles are rarely resulting from stutter but almost exclusively consist of PCR hybrids [20]. In Table 3b, the percentage of drop-out when using the 1.5% allele-calling threshold is categorised and illustrates that drop-out alleles consist mostly of heterozygous minor alleles. Most of these drop-out alleles represent minor contributions on stutter positions (where stutter ratio of the major contributor was lower than the average observed in the set of reference samples, thereby causing over-correction) and long alleles that suffer from heterozygote imbalance. In Figure 6 the trends from Table 3 are confirmed: drop-out is hardly and drop-in is largely affected by the use of noise correction. Thus, calling of genuine alleles is not negatively influenced by noise correction.

Note that the number of drop-ins may be reduced further by applying additional thresholds from Table 1, but the effects of varying additional threshold values were not studied in depth.

Table 3. Average number of drop-in alleles and average drop-out percentage per sample for different 'percentage of locus' allele-calling thresholds

a) Summary of drop-in and drop-out rates for various allele-calling thresholds

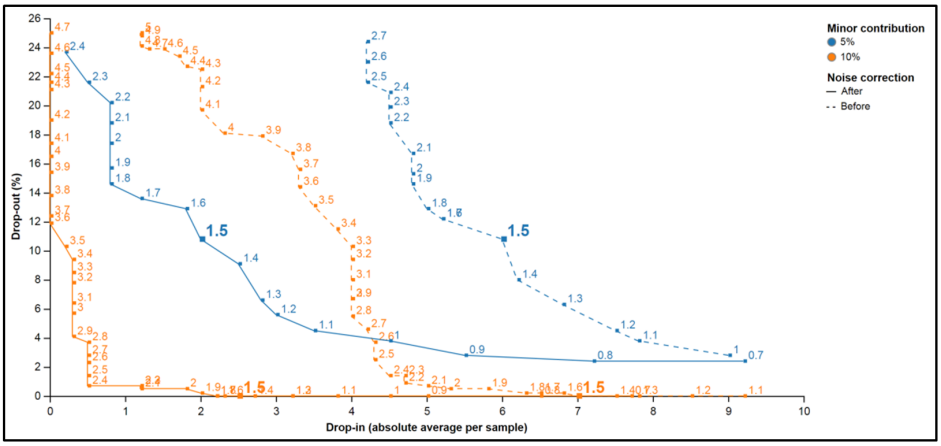
Minor contribution	50%: 600 pg	20%: 240 pg	10%: 120 pg	5%: 60 pg	1%: 60 pg	0.5%: 60 pg
Analysis method and threshold	# drop-in / % drop-out	# drop-in / % drop-out	# drop-in / % drop-out	# drop-in / % drop-out	# drop-in / % drop-out	# drop-in / % drop-out
Without correction, ≥ 5.0%	0.7 / 0.0%	0.8 / 2.1%	1.2 / 25.0%	1.0 / 33.1%	1.7 / 39.9%	0.8 / 40.8%
Without correction, ≥ 2.5%	4.3 / 0.0%	3.8 / 0.0%	4.3 / 2.5%	4.2 / 21.6%	3.5 / 38.3%	2.3 / 40.4%
Without correction, ≥ 2.0%	5.3 / 0.0%	5.3 / 0.0%	5.3 / 0.5%	4.8 / 15.3%	4.2 / 38.1%	2.8 / 40.1%
Without correction, ≥ 1.5%	7.7 / 0.0%	7.7 / 0.0%	7.0 / 0.0%	6.0 / 10.8%	6.0 / 37.4%	4.7 / 39.7%
With correction, ≥ 3.0%	0.0 / 0.0%	0.3 / 0.0%	0.3 / 5.7%	0.0 / 30.3%	0.7 / 41.1%	0.3 / 41.1%
With correction, ≥ 2.5%	0.3 / 0.0%	0.3 / 0.0%	0.5 / 1.4%	0.0 / 25.1%	0.7 / 40.6%	0.3 / 41.1%
With correction, ≥ 2.0%	0.7 / 0.0%	1.2 / 0.0%	1.8 / 0.5%	0.8 / 17.4%	0.8 / 40.6%	1.2 / 40.8%
With correction, ≥ 1.5%	1.7 / 0.0%	2.3 / 0.0%	2.5 / 0.0%	2.0 / 10.8%	2.7 / 39.9%	2.3 / 40.8%
With correction, ≥ 1.0%	4.0 / 0.0%	5.2 / 0.0%	4.5 / 0.0%	4.5 / 3.8%	5.7 / 37.8%	3.2 / 40.6%
With correction, ≥ 0.5%	16.3 / 0.0%	17.3 / 0.0%	14.0 / 0.0%	15.5 / 2.4%	14.0 / 30.3%	11.0 / 37.4%

b) Categorised drop-out rates when using 1.5% allele-calling threshold (with correction)

Minor contribution	20%: 240 pg	10%: 120 pg	5%: 60 pg	1%: 60 pg	0.5%: 60 pg
Alleles unique to the minor (homozygous)	0.0%	0.0%	0.0%	91.3%	100.0%
Alleles unique to the minor (heterozygous)	0.0%	0.0%	31.6%	98.1%	99.4%
Alleles unique to minor	0.0%	0.0%	27.0%	97.2%	99.4%
All alleles of the minor	0.0%	0.0%	18.5%	67.7%	69.3%
All alleles of the major	0.0%	0.0%	0.0%	0.0%	0.0%

Chapter 5

Figure 6. Average number of drop-in and percentage of drop-out per sample for different 'percentage of locus' allele-calling thresholds



Effect of different 'percentage of locus' allele-calling thresholds on the drop-in and drop-out rates. The numbers next to the points display allele-calling thresholds. The position of each point illustrates the number of drop-ins and percentage of drop-out for the corresponding threshold in mixtures with a ratio of 90:10 (orange) and 95:5 (blue). Points connected by dashed lines correspond to results obtained without noise correction, points connected by solid lines correspond to results obtained after noise correction. The 1.5% 'percentage of locus' allele-calling threshold that appears most optimal is indicated in bold.

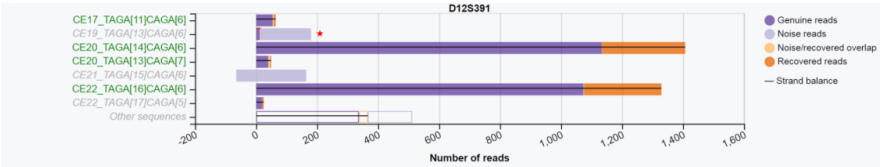
In Figure 7a–b, the effect of noise correction on allele calling is shown for a highly unbalanced mixture (95:5 mixture ratio) in which the alleles of the minor contributor have a similar or lower read count than the stutter products of the alleles of the major contributor. Without noise correction the four most frequent sequence variants are the major contributor's alleles and the corresponding –1 stutters and interpretation of the less frequent sequence variants becomes intractable; after noise correction, the stutter products and other PCR artefacts are filtered out and four sequence variants meet the 'percentage of locus' allele-calling threshold of 1.5%, which correspond to the four alleles of the two heterozygous donors. Also, the alleles of both the major and minor contributor have gained recovered reads.

Figure 7. Interpretation of a mixed sequence profile before and after correction

a) Sequence profile of locus D12S391 of a mixed sample with a ratio of 95:5, without noise filtering and correction

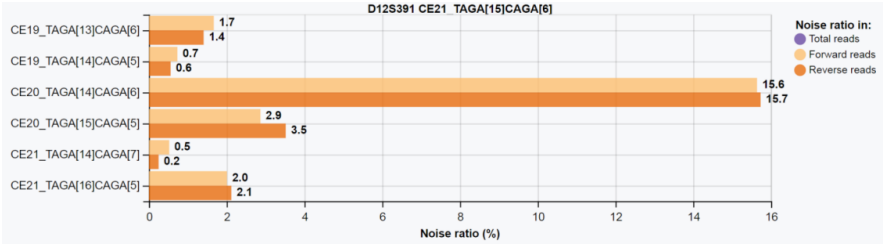
Marker	Allele	Reads	PctOfHighest	PctOfMarker	Correction	Recovery	Noise	PctForward	Notes
D12S391	CE17_TAGA[11]CAGA[6]	58	5.11%	1.82%	+0 (+0.00%)	0 (0.00%)	0 (0.00%)	62.07%	
D12S391	CE19_TAGA[13]CAGA[6]	180	15.86%	5.64%	+0 (+0.00%)	0 (0.00%)	0 (0.00%)	59.44%	
D12S391	CE20_TAGA[14]CAGA[6]	1135	100.00%	35.58%	+0 (+0.00%)	0 (0.00%)	0 (0.00%)	59.21%	
D12S391	CE21_TAGA[15]CAGA[6]	164	14.45%	5.14%	+0 (+0.00%)	0 (0.00%)	0 (0.00%)	56.10%	
D12S391	CE22_TAGA[16]CAGA[6]	1075	94.71%	33.70%	+0 (+0.00%)	0 (0.00%)	0 (0.00%)	61.40%	

b) Sequence profile of locus D12S391 of a mixed sample with a ratio of 95:5, with noise filtering and correction applied



Marker	Allele	Reads	PctOfHighest	PctOfMarker	Correction	Recovery	Noise	PctForward	Notes
D12S391	CE17_TAGA[11]CAGA[6]	58 → 64	4.56%	2.01%	+6 (+10.67%)	10 (14.93%)	3 (5.86%)	60.59%	BGCorrect·BGEstimate
D12S391	CE20_TAGA[14]CAGA[6]	1135 → 1407	100.00%	44.11%	+272 (+23.97%)	273 (19.39%)	1 (0.07%)	57.37%	BGCorrect·BGEstimate
D12S391	CE20_TAGA[13]CAGA[7]	46 → 49	3.50%	1.54%	+3 (+7.08%)	9 (19.07%)	6 (13.34%)	60.73%	BGCorrect·BGEstimate
D12S391	CE22_TAGA[16]CAGA[6]	1075 → 1329	94.45%	41.66%	+254 (+23.62%)	256 (19.24%)	2 (0.16%)	59.45%	BGCorrect·BGPredict

c) Noise profile of D12S391 allele CE21_TAGA[15]CAGA[6]



Sequence profile of locus D12S391 of a mixed sample with a ratio of 95:5, A without and B with applying noise filtering and correction. The table displays all sequence variants with at least 1.5% of the reads of the locus (the 'percentage of locus' allele-calling threshold used), which are also marked in green in the graph. A note in the table in panel B warns that no noise profile was available for the major CE22_TAGA[16]CAGA[6] allele and a stutter prediction has been used instead. An additional variant CE22_TAGA[17]CAGA[5], which is derived from the major CE22 allele, remains visible in the sequence profile (although not marked green as it does not meet the 1.5% threshold). C Noise profile of a similar allele, showing a non-stutter PCR artefact with a noise ratio of about 2%.

This example also displays a pitfall of the interpretation of a mixed DNA profile where the major contributor has an infrequent allele for which no *BGEstimate* noise profile is available. The noise correction of allele CE22_TAGA[16]CAGA[6] is only based on the stutter model (using BGPredict), which fails to correct for the CE22_TAGA[17]CAGA[5] PCR artefact. Looking at the noise profile of the most resembling allele in the reference database, CE21_TAGA[15]CAGA[6], we find a similar PCR artefact CE21_TAGA[16]CAGA[5] that represents a C to T substitution at the first CAGA repeat unit , with a noise ratio of about 2% (Figure 7c). This suggests that the CE22_TAGA[17]CAGA[5] artefact would be properly corrected if a noise profile for the CE22_TAGA[16]CAGA[6] allele would be available. Thus, additional inspection of the applied method of correction (*BGPredict* or *BGEstimate*) may be useful when infrequent alleles occur.

Analysis time and computer demand

To indicate the required time and computer memory demand, five samples with different numbers of reads (15,169–318,403 total read pairs) were analysed and the time and peak memory usage for each separate tool was registered (Supplementary Figure 6). With the used 2.0 GHz processor, the analysis time is mostly consumed by TSSV (≈75% of the total analysis time) and the complete analysis only takes up to 13:30 minutes for a sample with 318,403 reads. BGCorrect shows the highest peak memory usage but does not exceed 200 MB for the largest sample (of the five tested samples). Both the required time and memory increase more or less linearly when the read count of the analysed samples is increased.

Conclusions

We developed FDSTools for the analysis of forensic MPS data. FDSTools can determine systemic PCR and/or sequencing noise from the data of reference samples, build a database from this data and use it to correct for systemic noise in case samples. The software is also able to predict the noise caused by stutter for alleles not included in the reference database and uses this information in the correction of case samples.

With automatic threshold-based allele calling, noise correction reduces the occurrence of drop-in and drop-out substantially and improves the balance between alleles of a heterozygote pair. This decreases the detection limits of minor contributions in mixtures. STR stutter variants are no longer the most frequent remaining noise as PCR hybrid artefacts now generally exceed the corrected read counts of stutters.

Although reliable noise correction can already be obtained from a database of 100 samples, a larger database is preferred as a larger number of alleles can be corrected by

the use of a complete noise profile instead of relying on noise predictions based on the stutter model. This will also reduce manual inspection of retained non-stutter noise for infrequent alleles. When building the database, it is important to use an amplification kit representative for the kit used for the samples. Although not extensively tested, we anticipate that noise prediction will be less precise when less DNA is used and more stochastic PCR effects occur. Also, more strand bias will occur during the massively parallel sequencing. These effects are intrinsic to low-level DNA typing and probabilistic genotyping software have been developed that accommodate drop-in and dropout during profile interpretation [21,22,23,24,25,26,27,28]. Such software are not yet straightforwardly able to deal with MPS data, but the necessary adaptations are feasible and include nomenclature for sequence variants, allele frequencies databases and read counts replacing peak heights in continuous models (not required for semi-continuous models) . In CE-based analysis, PCR replicates are often used to reduce profiling uncertainty [7]; replicates can be entered in probabilistic genotyping software or used to prepare a consensus profile [7,8]. A future version of FDSTools will feature a consensus-based analysis method alike those used with CE data [8]. Besides, export options for DNA database systems such as CODIS will be added.

FDSTools has been validated following recommendations for software validation [29,30] and is already implemented in the ISO17025 certified environment of the LUMC for forensic casework. The validation for use and performance of the software in casework was a separate study which was not based on the data described in this manuscript. By providing tools to evaluate the performance of noise correction in reference samples FDSTools facilitates the determination of analysis thresholds that are fit for purpose.

The application of FDSTools is not limited to the analysis of STRs. FDSTools has already been applied successfully to the analysis of multiplex assays of SNP fragments (manuscript in preparation) and complete mtDNA data (Weiler et al., submitted). Note that the minimum number of required reference samples for loci other than STRs will depend on the amount of variation observed in these loci.

Acknowledgements

This study was supported by a grant from the Netherlands Genomics Initiative / Netherlands Organization for Scientific Research (NWO) within the framework of the Forensic Genomics Consortium Netherlands. The authors wish to thank Arie Koppelaar (Netherlands Forensic Institute) for providing help and expertise in setting up the computer cluster for the analyses. We also thank Martin Ensenberger, Cynthia Sprecher and Douglas R. Storts (Promega Corporation) for their contributions to designing and providing the PowerSeq™ Auto System.

References

1. M.A. Jobling, P. Gill. Encoded evidence: DNA in forensic analysis. Nat. Rev. Genet. 5 (10) (2004) 739–751.
2. K.J. van der Gaag, R.H. de Leeuw, J. Hoogenboom, J. Patel, D.R. Storts, J.F.J. Laros, P. de Knijff. Massively parallel sequencing of short tandem repeats-Population data and mixture analysis results for the PowerSeq™ System. Forensic Sci. Int. Genet. 24 (2016) 86-96.
3. K.B. Gettings, K.M. Kiesler, S.A. Faith, E. Montano, C.H. Baker, B.A. Young, R.A. Guerrieri, P.M. Vallone. Sequence variation of 22 autosomal STR loci detected by next generation sequencing. Forensic Sci. Int. Genet. 21 (2016) 15–21.
4. C. Gelardi, E. Rockenbauer, S. Dalsgaard, C. Børsting, N. Morling. Second generation sequencing of three STRs D3S1358, D12S391 and D21S11 in Danes and a new nomenclature for sequenced STR alleles. Forensic Sci. Int. Genet. 12 (2014) 38–41.
5. M. Scheible, O. Loreille, R. Just, J. Irwin. Short tandem repeat typing on the 454 platform: Strategies and considerations for targeted sequencing of common forensic markers. Forensic Sci. Int. Genet. 12 (2014) 104–119.
6. X. Zeng, J.L. King, M. Stoljarova, D.H. Warshauer, B.L. LaRue, A. Sajantila, J. Patel, D.R. Storts, B. Budowle. High sensitivity multiplex short tandem repeat loci analyses with massively parallel sequencing. Forensic Sci. Int. Genet. 16 (2015) 38–47.
7. C.C. Benschop, C.P. van der Beek, H.C. Meiland, A.G. van Gorp, A.A. Westen, T. Sijen. Low template STR typing: effect of replicate number and consensus method on genotyping reliability and DNA database search results. Forensic Sci. Int. Genet. 5 (2011) 316–328.
8. C.C. Benschop, T. Sijen. LoCIM-tool: An expert's assistant for inferring the major contributor's alleles in mixed consensus DNA profiles. Forensic Sci. Int. Genet. 11 (2014) 154–165.
9. C. Brookes, J.A. Bright, S. Harbison, J. Buckleton. Characterising stutter in forensic STR multiplexes. Forensic Sci. Int. Genet. 6 (2012) 58–63.
10. S.Y. Anvar, K.J. van der Gaag, J.W. van der Heijden, M.H. Veltrop, R.H. Vossen, R.H. de Leeuw, C. Breukel, H.P. Buermans, J.S. Verbeek, P. de Knijff, J.T. den Dunnen, J.F.J. Laros. TSSV: a tool for characterization of complex allelic variants in pure and mixed genomes. Bioinformatics 30 (2014) 1651–1659.
11. D.H. Warshauer, J.L. King, B. Budowle. STRait Razor v2.0: the improved STR Allele Identification Tool--Razor. Forensic Sci. Int. Genet. 14 (2015) 182–186.
12. S.L. Friis, A. Buchard, E. Rockenbauer, C. Børsting, N. Morling. Introduction of the Python script STRinNGS for analysis of STR regions in FASTQ or BAM files and expansion of the Danish STR sequence database to 11 STRs. Forensic Sci. Int. Genet., 21 (2016) 68–75.
13. A.A. Westen, T. Kraaijenbrink, E.A. Robles de Medina, J. Harteveld, P. Willemse, S.B. Zuniga, K.J. van der Gaag, N.E. Weiler, J. Warnaar, M. Kayser, T. Sijen, P. de Knijff. Comparing six commercial autosomal STR kits in a large Dutch population sample. Forensic Sci. Int. Genet. 10 (2014) 55–63.
14. M.G. Ensenberger, K.A. Lenz, L.K. Matthies, G.M. Hadinoto, J.E. Schienman, A.J. Przech, M.W. Morganti, D.T. Renstrom, V.M. Baker, K.M. Gawrys, M. Hoogendoorn, C.R. Steffen, P. Martín, A. Alonso, H.R. Olson, C.J. Sprecher, D.R. Storts. Developmental validation of the PowerPlex® Fusion 6C System. Forensic Sci. Int. Genet. 21 (2016) 234–144.
15. T. Magoc, S.L. Salzberg. FLASH: fast length adjustment of short reads to improve genome

assemblies. Bioinformatics 27 (2011) 2957–2963.
16. B.C. Hendrickson, B. Leclair, S. Forrest, J. Ryan, B.E. Ward, D. Petersen, T.D. Kupferschmid, T. Scholl. Accurate STR allele designations at the FGA and vWA loci despite site polymorphisms. J. Forensic Sci. 49 (2) (2004) 250–254.
17. D. Shinde, Y. Lai, F. Sun, N. Arnheim. Taq DNA polymerase slippage mutation rates measured by PCR and quasi-likelihood analysis: (CA/GT)n and (A/T)n microsatellites. Nucleic Acids Res. 31 (2003) 974–980.
18. M. Klintschar, P. Wiegand. Polymerase slippage in relation to the uniformity of tetrameric repeat stretches. Forensic Sci. Int. 135 (2) (2003) 163–166.
19. A. Satyanarayan, R. Russell, J. Hoffswell, J. Heer. Reactive Vega: A Streaming Dataflow Architecture for Declarative Interactive Visualization. IEEE Trans. Vis. Comput. Graphics. 22 (2016) 659–668.
20. A.R. Shuldiner, A. Nirula, J. Roth. Hybrid DNA artifact from PCR of closely related target sequences. Nucleic Acids Res. 17 (11) (1989) 4409.
21. H. Haned, P. Gill. Analysis of complex DNA mixtures using the Forensim package. Forensic Sci. Int. Genet. Supp. Series 3 (2011) e79–e80.
22. H. Swaminathan, A. Garg, C.M. Grgicak, M. Medard, D.S. Lun. CEESIt: A computational tool for the interpretation of STR mixtures. Forensic Sci. Int. Genet. (2016) 149–160.
23. D.J. Balding. Evaluation of mixed-source, low-template DNA profiles in forensic science. Proc. Natl. Acad. Sci. U. S. A. 110 (30) (2013) 12241–12246.
24. D. Taylor, J.A. Bright, J. Buckleton. The interpretation of single source and mixed DNA profiles. Forensic Sci. Int. Genet. 7 (2013) 516–528.
25. R.G. Cowell, T. Graversen, S.L. Lauritzen, J. Mortera. Analysis of forensic DNA mixtures with artefacts. Appl. Stat. 64 (1) (2015) 1–32.
26. O. Bleka, G. Storvik, P. Gill. EuroForMix: An open source software based on a continuous model to evaluate STR DNA profiles from a mixture of contributors with artefacts. Forensic Sci. Int. Genet. 21 (2016) 35–44.
27. M.W. Perlin, M.M. Legler, C.E. Spencer, J.L. Smith, W.P. Allan, J.L. Belrose, B.W. Duceman. Validating TrueAllele® DNA Mixture Interpretation. J. Forensic Sci. 56 (2011) 1430–1447.
28. R. Puch-Solis, L. Rodgers, A. Mazumder, S. Pope, I.W. Evett, J. Curran, D. Balding. Evaluating forensic DNA profiles using peak heights, allowing for multiple donors, allelelic dropout and stutters. Forensic Sci. Int. Genet. 7 (2013) 555–563.
29. H. Haned, P. Gill, K. Lohmueller, K. Inman, N. Rudin. Validation of probabilistic genotyping software for use in forensic DNA casework: Definitions and illustrations. Sci. Justice 56 (2) (2016) 104–108.
30. E.J. Rykiel. Testing ecological models: the meaning of validation. Ecol. Model. 90 (1996) 229–244.

Chapter 5

Supplementary materials

Supplementary Text 1 - Allele-centric systemic noise estimation (BGEstimate)

The BGEstimate tool of FDSTools does the computation of a background noise profile for each allele found among a set of reference samples. Computing background noise profiles from a set of homozygous samples is straightforward, whereas for heterozygous samples this becomes more complicated as the alleles of one sample in general appear in different amounts, thereby systematically contributing to a different amount of the same background noise sequence.

Algorithm 1, which enables the computation of these background noise profiles from heterozygous samples, is implemented in BGEstimate. In testing, the best results were obtained if for each allele at least one homozygous sample or at least three different heterozygous samples were available. With default settings, BGEstimate will ensure these conditions are met before executing Algorithm 1.

In essence Algorithm 1 seeks a non-negative least squares solution to the matrix equation $AP = C$. In this equation, C is an N×M matrix of constants derived from the observed read counts in the reference samples (see Figure 1), A is an N×N matrix in which the estimated allele balance in the samples is summarised and P is an N×M matrix containing the estimated profiles of systemic noise. N is the number of unique alleles among the observed reference samples and therefore also the number of profiles produced and M is the number of unique sequences observed. It is possible to include additional sequences beyond the N alleles of the samples if this is deemed appropriate. With default settings, BGEstimate will include all sequences that appear in at least 80% of the samples with any particular allele, since these sequences are probably the result of systemic noise. In any case, the first N columns in P and C correspond to the N alleles of the samples and the order of the rows and columns is the same (i.e., row n and column n in both P and C correspond to the same sequence).

The input of Algorithm 1 consists of a K×M matrix S which contains the observed number of reads of each of the M sequences in each of the K samples. The genotype of each sample is provided as a set of indices $g_k \forall g_k \leq N$.

Any element $P_{n,m}$ of P can be interpreted as the amount of sequence m that is observed, on average, for every 100 reads of sequence n. Therefore, the algorithm initialises P to a diagonal N×M matrix with the elements on its major diagonal set to 100. The number 100 was chosen for practical reasons since it directly results in noise ratios expressed as percentages of the actual allele.

Algorithm 1 Systemic noise profile estimation.

Require: $K \times M$ matrix $\mathbf{S}$ containing the K samples in the rows
Require: number of alleles $N \leq M$, corresponding to the first N columns of $\mathbf{S}$
Require: list g of sets g_k: the genotypes (indices of alleles $\leq N$) of samples $\mathbf{S}_k$
Return: $N \times M$ matrix $\mathbf{P}$ containing the profiles

function EstimateBackgroundProfiles($\mathbf{S}, g, N$)
  Initialisation:
  $\mathbf{P}_{n,m} \leftarrow 0, \quad 1 \leq n \leq N, \quad 1 \leq m \leq M$
  $\mathbf{P}_{n,n} \leftarrow 100, \quad 1 \leq n \leq N$ ▷ Set actual allele to 100.
  $\mathbf{C}_{n,m} \leftarrow 0, \quad 1 \leq n \leq N, \quad 1 \leq m \leq M$
  for $k \leftarrow 1$ **to** K **do**
    $\mathbf{C}_{i,:} \leftarrow \mathbf{C}_{i,:} + \mathbf{S}_{k,:} \times \frac{100}{|g_k|} / \mathbf{S}_{k,i}, \quad \forall i \in g_k$ ▷ Compute separately for each i.
  end for
  Optimisation:
  repeat
    Estimate allele balance:
    $\mathbf{A}_{i,j} \leftarrow 0, \quad \forall i, j \in [1 \dots N]$
    for $k \leftarrow 1$ **to** K **do**
      $\mathbf{Q}_{i,j} \leftarrow \mathbf{P}_{g_{k,i}, g_{k,j}}, \quad \forall i, j \in [1 \dots |g_k|]$
      $\mathbf{r}_i \leftarrow \mathbf{S}_{k, g_{k,i}}, \quad \forall i \in [1 \dots |g_k|]$
      $\mathbf{B} \leftarrow \left(\frac{100}{|g_k|} / \mathbf{r}^T \right) \times \text{NNLS}(\mathbf{Q}^T, \mathbf{r}^T)^T$
      $\mathbf{A}_{g_{k,i}, g_{k,j}} \leftarrow \mathbf{A}_{g_{k,i}, g_{k,j}} + \mathbf{B}_{i,j}, \quad \forall i, j \in [1 \dots |g_k|]$
    end for
    Update profiles:
    $\mathbf{E} \leftarrow \mathbf{A}^T \mathbf{A}$
    $\mathbf{F} \leftarrow \mathbf{A}^T \mathbf{C}$
    repeat
      for $n \leftarrow 1$ **to** N **do**
        $\mathbf{P}_{n,:} \leftarrow (\mathbf{F}_{n,:} - \mathbf{E}_{n,\setminus n} \mathbf{P}_{\setminus n,:}) / \mathbf{E}_{n,n}$
        $\mathbf{P}_{n,:} \leftarrow \text{MAX}(\mathbf{P}_{n,:}, 0)$ ▷ Set negative elements to 0.
        $\mathbf{P}_{n,n} \leftarrow 100$ ▷ Keep actual allele at 100.
      end for
    until stop condition is met
  until stop condition is met
  return $\mathbf{P}$
end function

Chapter 5

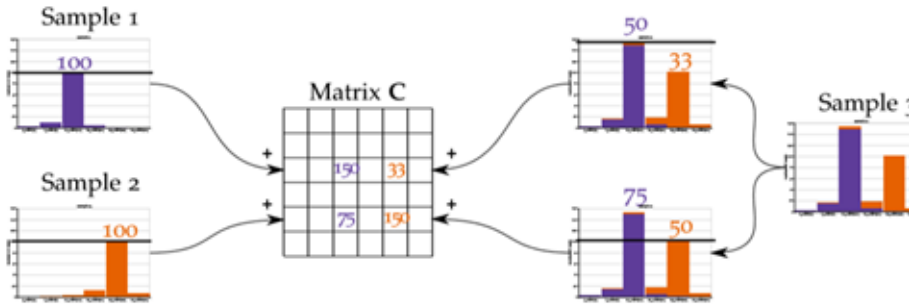


Figure 1. Construction of matrix C in Algorithm 1 by summing scaled read counts of the samples that share the same alleles. Left: Samples 1 and 2 are homozygous for the third and fifth allele respectively. The read counts of both samples are scaled such that their true allele is equal to 100, after which they are added to the third and fifth row of matrix C respectively. Right: Sample 3 is heterozygous, having both the third and fifth allele. Therefore, it is added to both the third and fifth row of matrix C, with its read counts scaled such that the third or fifth allele is equal to 50 respectively.

Similarly, the elements Cn,m of C can be interpreted as the total amount of sequence m that is observed in all samples with allele n. Line 6 in Algorithm 1 initialises C. For homozygous samples, it scales the read counts of each sample Sk such that its allele $S_{k,i}, i \in g_k$ is equal to 100 and then adds the scaled counts to row Ci of C. Heterozygous samples are treated likewise twice — once for each of their alleles — except that the allele is scaled to 50 instead of 100 to compensate for the fact that the sample is added to two rows in C, as compared to just one for homozygous samples.[1]

Each row Ci of C thus contains the sum of the read counts of all samples that have allele i, with the read counts of each sample scaled such that allele $S_{k,i}, i \in g_k$ becomes $100/|g_k|$.[2] After matrices P and C are initialised, the algorithm enters its main loop wherein it alternately estimates the allele balance in the samples (matrix A) and refines the least squares fit of the profiles P. The main loop is exited and the profiles are returned when the sum of the squared errors,

$$\sum_{n=1}^{N}\sum_{m=1}^{M}(\mathbf{D}_{n,m})^2, \quad \mathbf{D} = \mathbf{C} - \mathbf{AP}$$

is reduced by less than 0.01% in one iteration. This stopping condition is generally met within 20 iterations.

1 One may also say that the samples are added once for each allele, adding them to the same row twice
2 Interestingly, because Algorithm 1 scales read counts to 100 divided by the number of alleles in the sample, it handles samples with more than two alleles without problems. Since Algorithm 1 makes no assumptions about the number of alleles each sample can have, it is possible to use mixed samples to compute systemic noise profiles as well. This has not been tested, however.

Estimation of the allele balance is done for every sample in isolation. At line 11, the elements in P that correspond to cross-contributions between the alleles $i,j \in [1...|g_k|]$ of sample k are extracted. Similarly, the corresponding elements from Sk are extracted at line 12. For heterozygous samples, this expands to (shortening $g_{k,i}$ to i for brevity):

$$\mathbf{Q} \leftarrow \begin{bmatrix} \mathbf{P}_{i,i} & \mathbf{P}_{i,j} \\ \mathbf{P}_{j,i} & \mathbf{P}_{j,j} \end{bmatrix} = \begin{bmatrix} 100 & \mathbf{P}_{i,j} \\ \mathbf{P}_{j,i} & 100 \end{bmatrix}$$

$$\mathbf{r} \leftarrow \begin{bmatrix} \mathbf{S}_{k,i} & \mathbf{S}_{k,j} \end{bmatrix}$$

At line 13, a non-negative least squares algorithm is employed to estimate the allele balance within the sample. The nnls function can be any algorithm that solves JK = L for K subject to K≥0 in the least squares sense, e.g., [1]. Line 13 of Algorithm 1 uses this function to solve bQ = r for b (by solving QTbT = rT), which gives an estimation of the proportions in which the alleles are present in the sample. The resulting row vector b is left-multiplied by a column vector with the same scaling factors as previously calculated at line 6. The result is, for heterozygotes, a 2×2 matrix B.[3] Finally, at line 14, the elements of B are added to their corresponding elements of A.

With the allele balance estimates all added to A, the second step in the main loop of Algorithm 1 is to update the profiles P such that they form a non-negative least squares solution to the equation AP = C subject to the additional requirement that the elements on the diagonal of P must be 100. Lines 16–24 implement nnls(A, C) with this additional requirement enforced on line 22. Indeed, with the omission of line 22, lines 16–24 are a general purpose implementation of the nnls function. This implementation is based on [1].

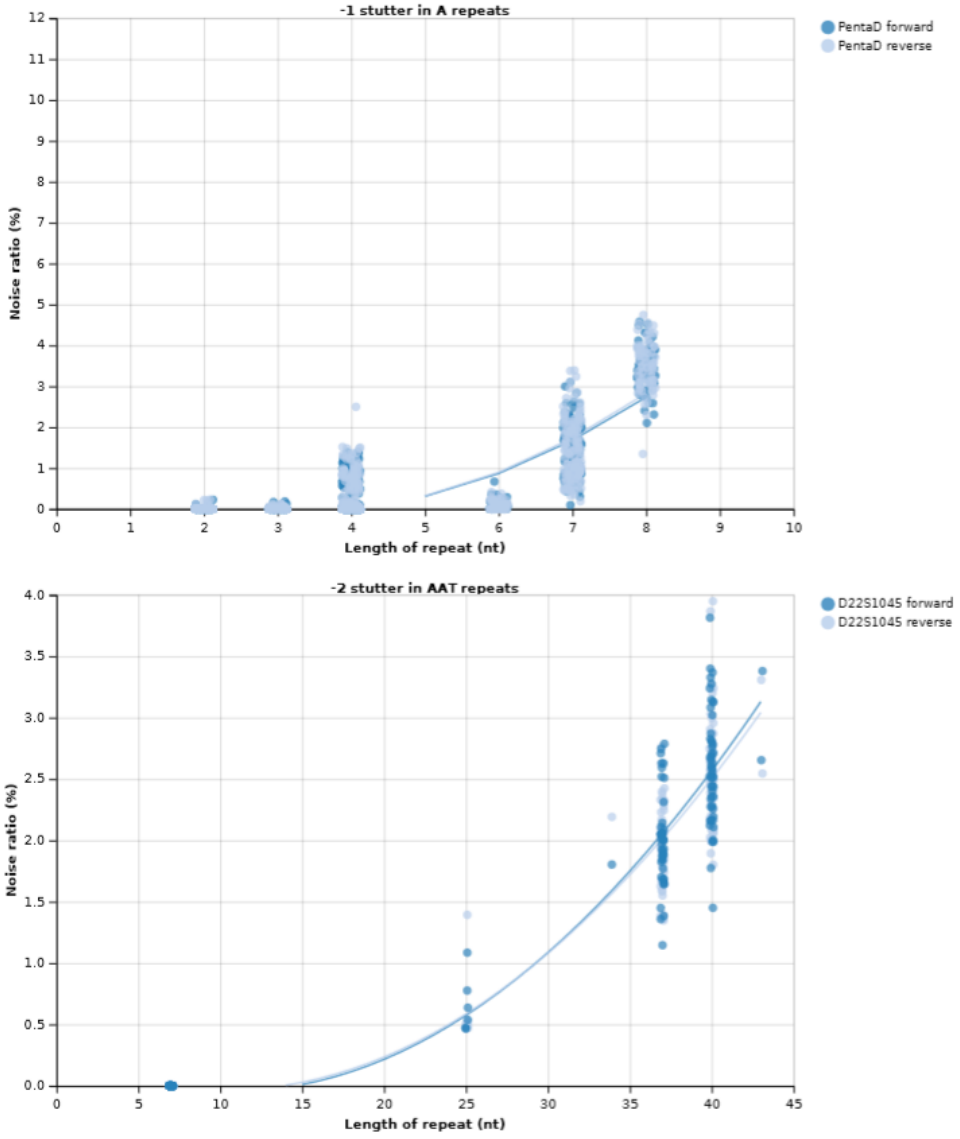
Separate profiles for the numbers of forward and reverse reads can be constructed by doubling the number of columns in P and C, where the left half corresponds to the forward stand and the right half to the reverse strand. This ensures that the same allele balance matrix A is used for both strands.

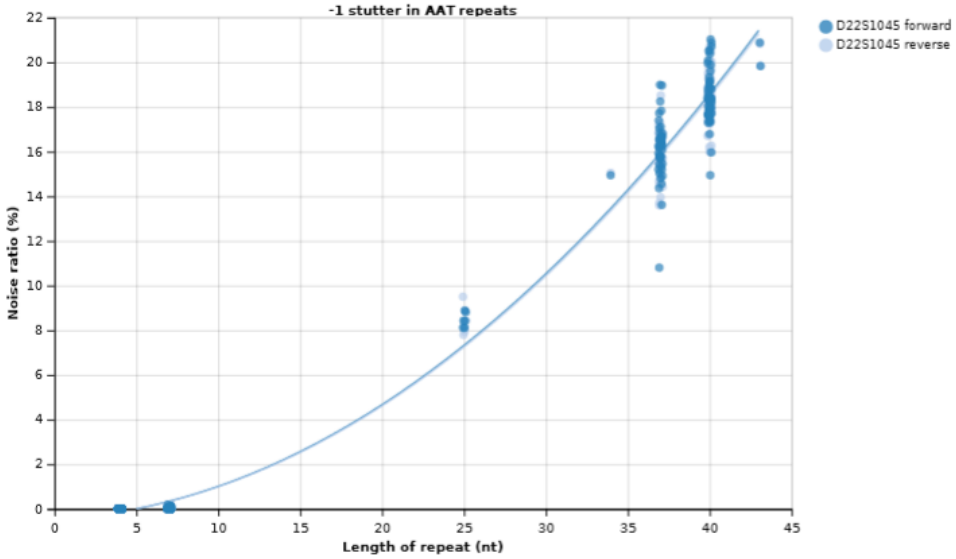
References

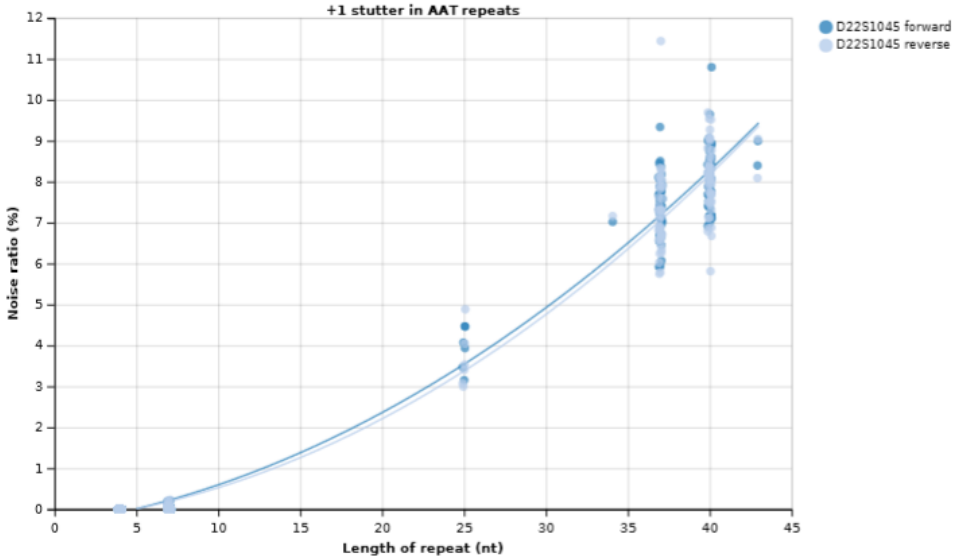
1. M. N. Schmidt, O. Winther, and L. K. Hansen. Bayesian non-negative matrix factorization. In: Independent Component Analysis and Signal Separation. Springer, 2009, pp. 540–547.

Chapter 5

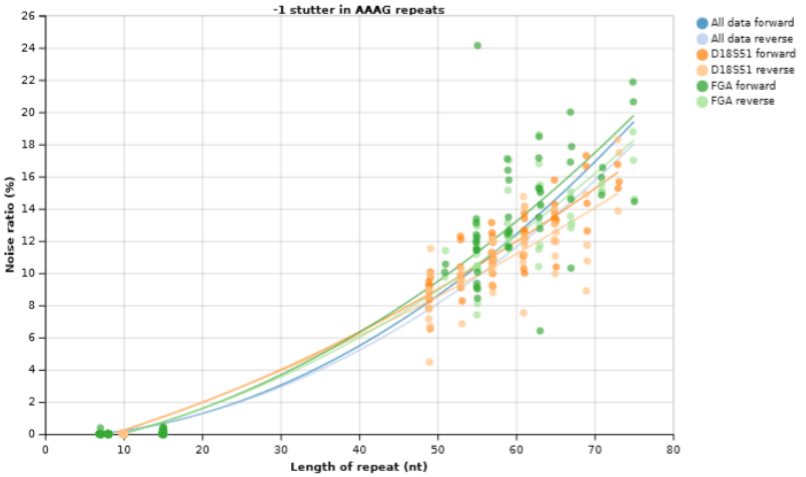
Supplementary Figure 1 – Stutter model based on a reference database of 429 samples

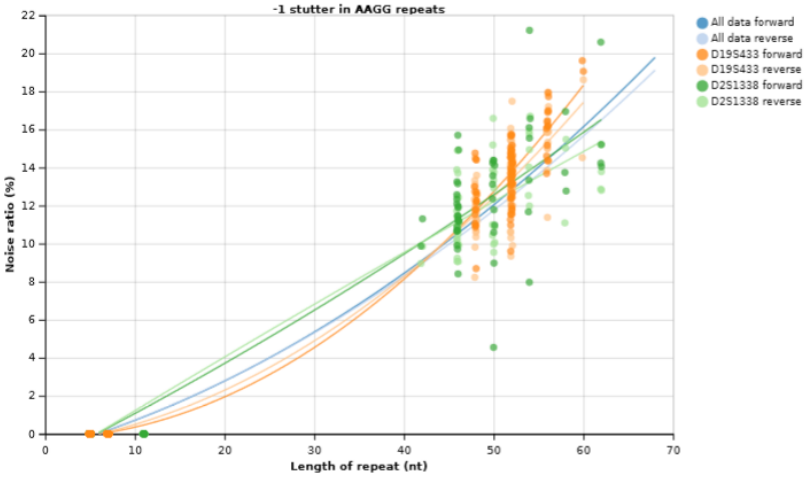


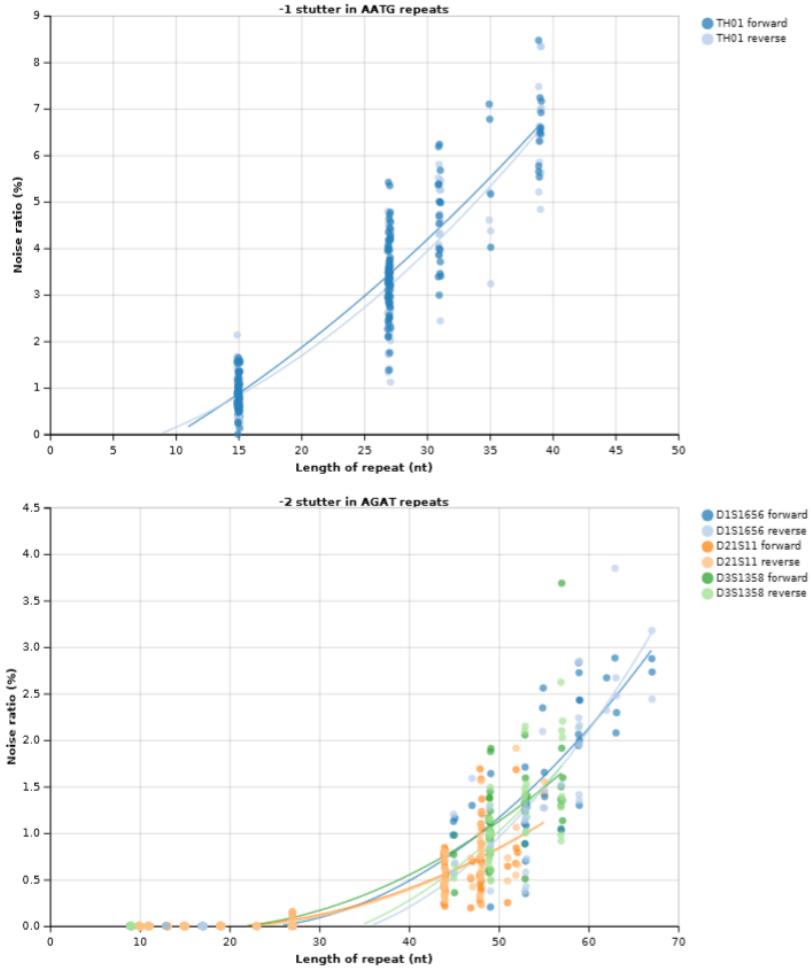

-1 stutter in AAT repeats
Noise ratio (%)
Length of repeat (nt)
D22S1045 forward
D22S1045 reverse


+1 stutter in AAT repeats
Noise ratio (%)
Length of repeat (nt)
D22S1045 forward
D22S1045 reverse

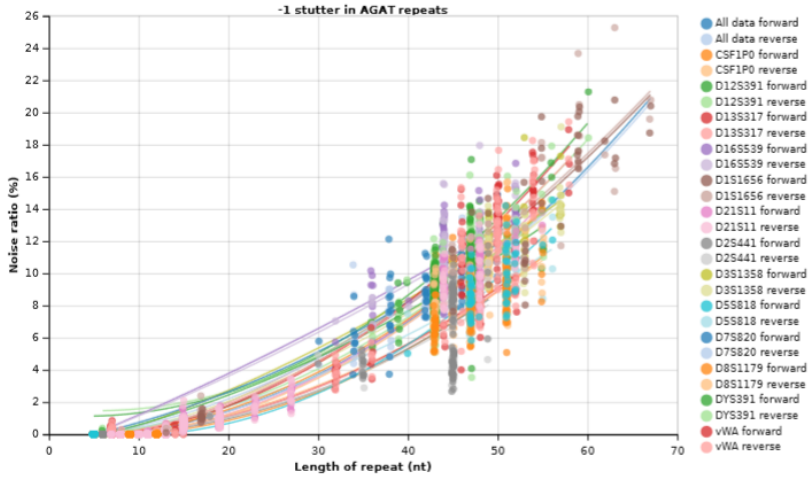
Chapter 5

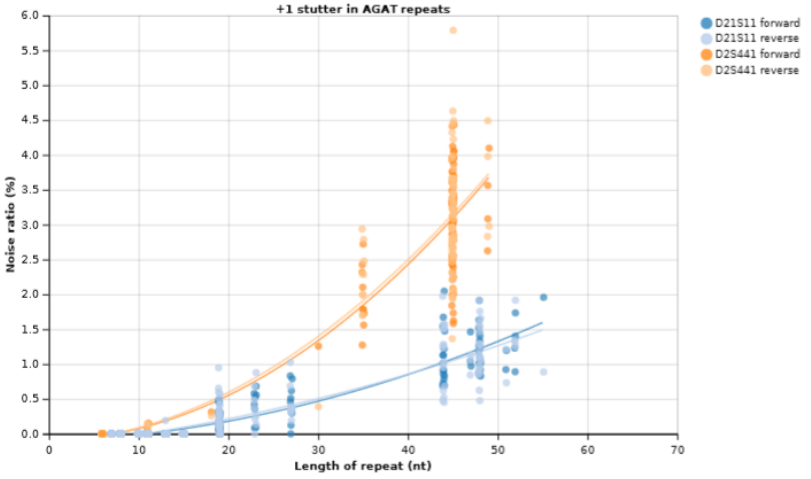

-1 stutter in AAAG repeats
Noise ratio (%)
Length of repeat (nt)
All data forward
All data reverse
D18S51 forward
D18S51 reverse
FGA forward
FGA reverse

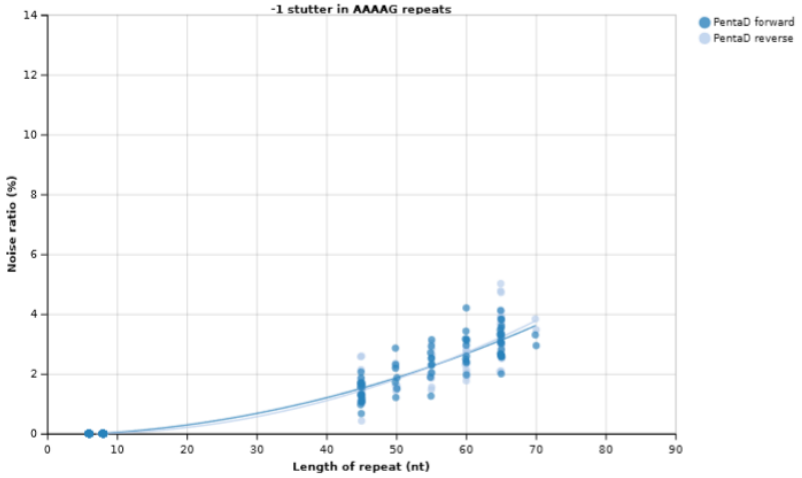

-1 stutter in AAGG repeats
Noise ratio (%)
Length of repeat (nt)
All data forward
All data reverse
D19S433 forward
D19S433 reverse
D2S1338 forward
D2S1338 reverse


-1 stutter in AATG repeats
Noise ratio (%)
Length of repeat (nt)
TH01 forward
TH01 reverse
-2 stutter in AGAT repeats
Noise ratio (%)
Length of repeat (nt)
D1S1656 forward
D1S1656 reverse
D21S11 forward
D21S11 reverse
D3S1358 forward
D3S1358 reverse

Chapter 5

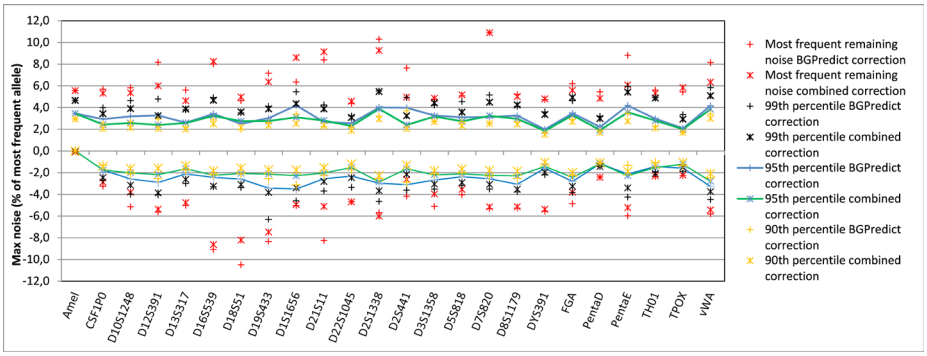

-1 stutter in AGAT repeats
Noise ratio (%)
Length of repeat (nt)
All data forward
All data reverse
CSF1P0 forward
CSF1P0 reverse
D12S391 forward
D12S391 reverse
D13S317 forward
D13S317 reverse
D16S539 forward
D16S539 reverse
D1S1656 forward
D1S1656 reverse
D21S11 forward
D21S11 reverse
D2S441 forward
D2S441 reverse
D3S1358 forward
D3S1358 reverse
D5S818 forward
D5S818 reverse
D7S820 forward
D7S820 reverse
D8S1179 forward
D8S1179 reverse
DYS391 forward
DYS391 reverse
vWA forward
vWA reverse


+1 stutter in AGAT repeats
Noise ratio (%)
Length of repeat (nt)
D21S11 forward
D21S11 reverse
D2S441 forward
D2S441 reverse



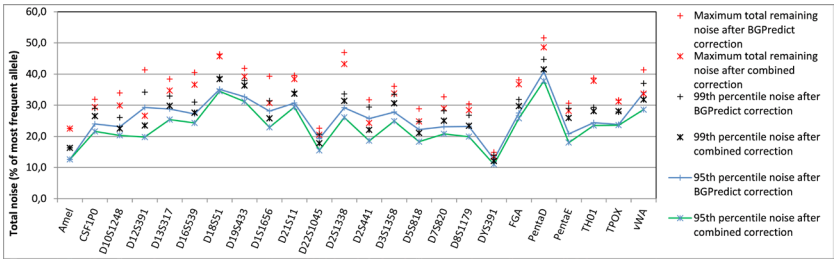
Supplementary Figure 2 – Comparison of remaining noise using different correction settings

a) Most frequent noise variant for each locus after correction in 429 reference samples, BGPredict correction vs combined BGEstimate and BGPredict correction

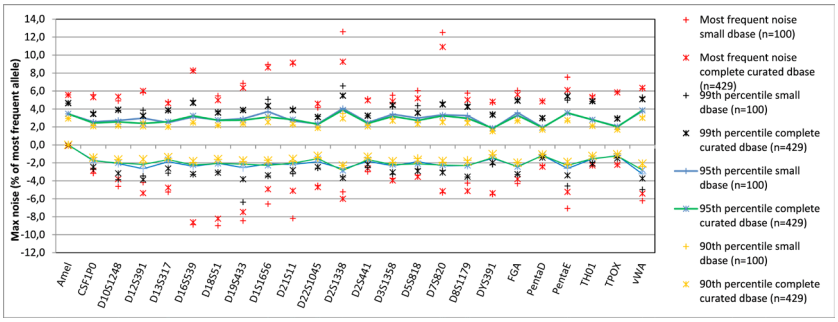


Chapter 5

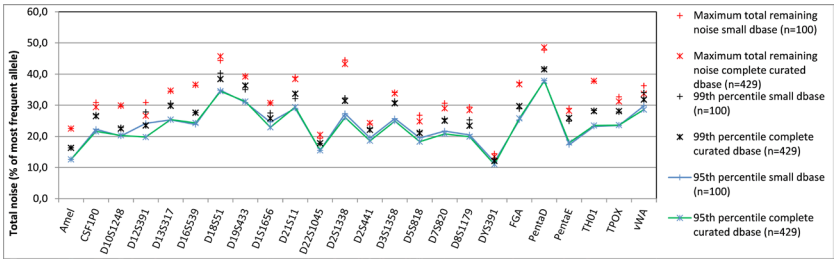
b) Total remaining noise for each locus after correction in 429 reference samples, BGPredict correction vs combined BGEstimate and BGPredict correction



c) Most frequent noise variant for each locus after combined correction in 429 reference samples, correction based on 100 vs 429 reference samples



d) Total remaining noise for each locus after correction in 429 reference samples, correction based on 100 vs 429 reference samples



a) The dotplot displays the most frequently observed noise that remained after correction in any of the 429 analysed reference samples when performing the correction using BGPredict only (based on the stutter model) or a combination of BGEstimate and BGPredict. In addition, the 99th and the 95th percentile are plotted to illustrate the variation in remaining noise.
b) The dotplot displays the highest total observed noise (cumulative percentage of the reads of the most frequent allele) that remained after correction in any of the 429 analysed reference samples when performing the correction using BGPredict only (based on the stutter model) or a combination of BGEstimate and BGPredict. In addition, the 99th and the 95th percentile are plotted to illustrate the variation in remaining noise.
c) The dotplot displays the most frequently observed noise that remained after correction in any of the 429 analysed reference samples when performing the correction based on all 429 samples or only 100 randomly selected samples from this database. In addition, the 99th, 95th, and 90th percentile are plotted to illustrate the variation in remaining noise.
d) The dotplot displays the highest total observed noise (cumulative percentage of the reads of the most frequent allele) that remained after correction in any of the 429 analysed reference samples when performing the correction based on all 429 samples or only 100 randomly selected samples from this database. In addition, the 99th and 95th percentile are plotted to illustrate the variation in remaining noise.

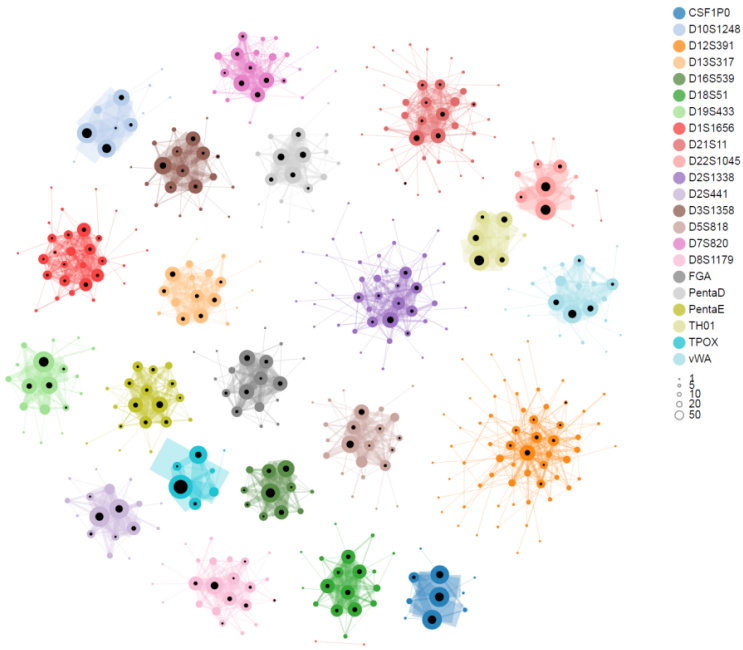
Supplementary Figure 3 - Visualisation of the frequency of each allele and the frequency of co-occurence with other alleles as heterozygous genotypes

The graphs depict every allele among the reference samples as a circle. The size of the circle corresponds to the number of samples with that particular allele. A black inner circle depicts the number of homozygotes.

The circles of two alleles are connected by a line whenever samples exist that have a combination of the two connected alleles. The thickness of the line corresponds to the number of heterozygotes with that particular combination of alleles.

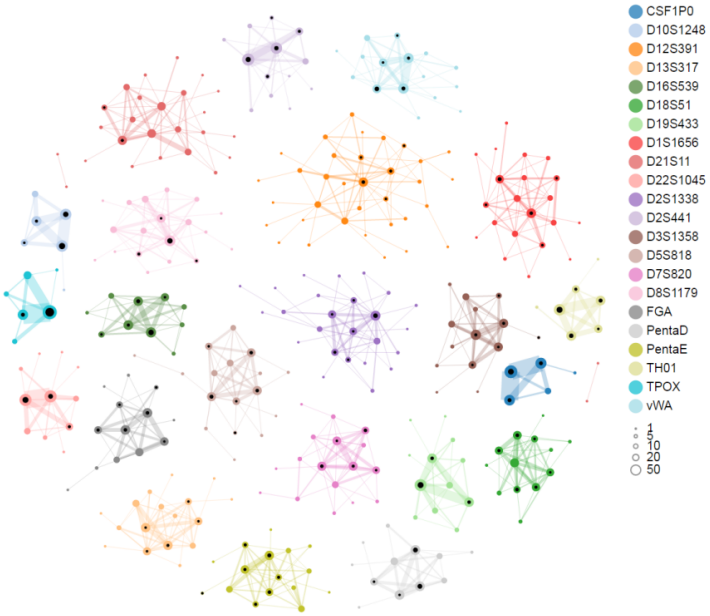
To fit the criteria to create a BGEstimate noise profile, alleles need to be present as a homozygous genotype (displayed as a black circle) or be connected with at least three other alleles that must also fit these criteria.
These figures were generated using the command 'fdstools vis allele'.

a) Allele visualisation of 429 reference samples

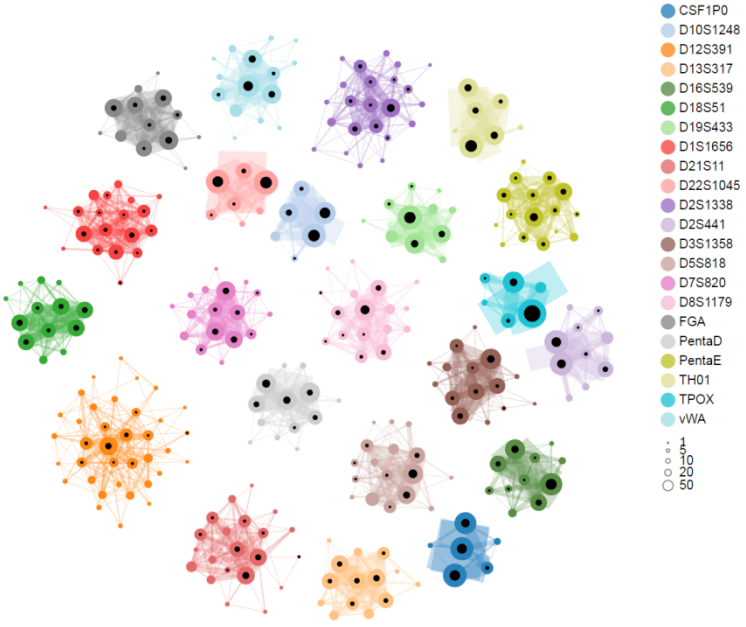


Chapter 5

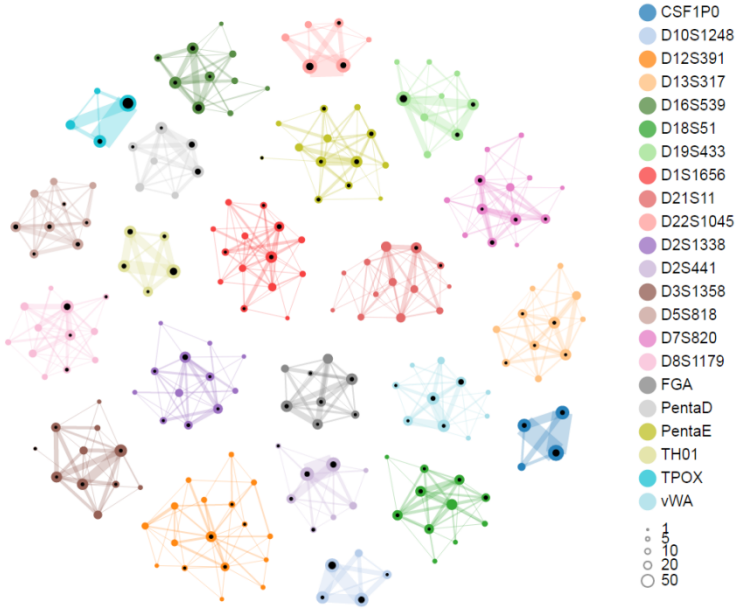
b) Allele visualisation of 100 reference samples



c) Allele visualisation of 429 reference samples after removing alleles that do not meet thresholds for determining a reliable noise profile

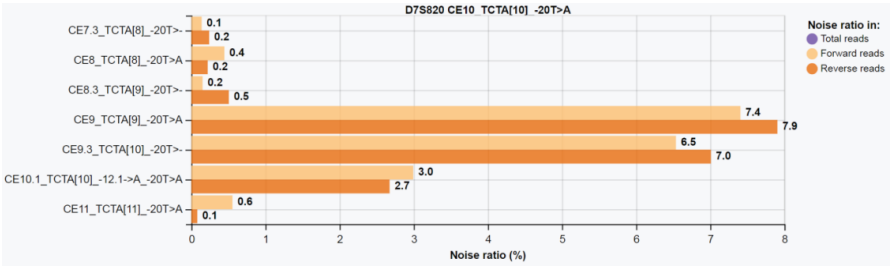


d) Allele visualisation of 100 reference samples after removing alleles that do not meet thresholds for determining a reliable noise profile

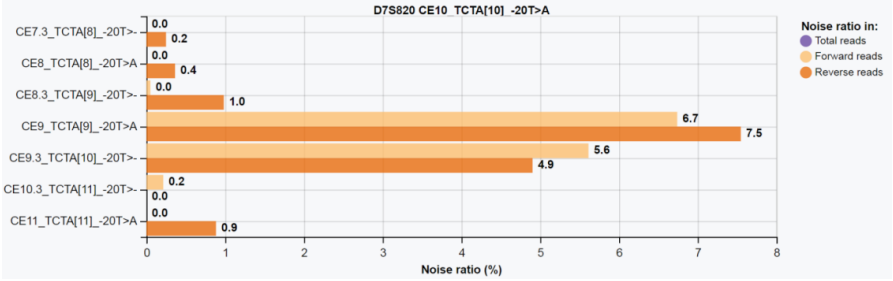


Supplementary Figure 4 – Noise profiles of D7S820 allele CE10_TCTA[10]_-20T>A estimated from high and low-coverage samples

a) Noise profile of D7S820 allele CE10_TCTA[10]_-20T>A estimated from high-coverage samples



b) Noise profile of D7S820 allele CE10_TCTA[10]_-20T>A estimated from low-coverage samples

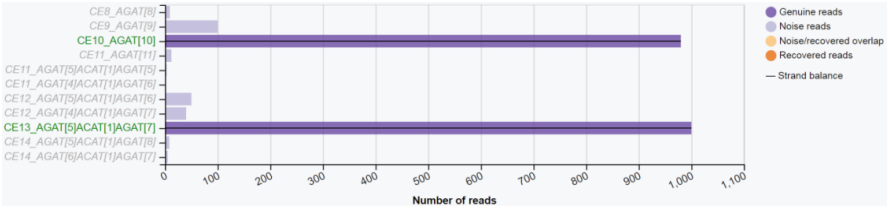


Noise profiles created with BGEstimate based on a selection of a) 71 high-coverage samples (82,000–350,000 total reads) and b) 70 low-coverage samples (8,000–44,000 total reads). The noise ratio is shown for each systemic noise sequence observed. It is clear that for the low-percentage noise in the low-coverage noise profile, more strand bias is introduced due to single-strand drop-out of this noise caused by insufficient coverage of the reference samples.

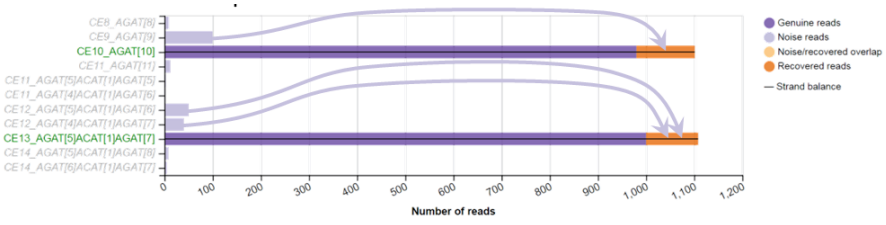
Supplementary Figure 5a – Explanation of a sequence profile for raw, filtered and corrected data

Raw data

Noise reads filtered out



Noise reads added to the parent alleles

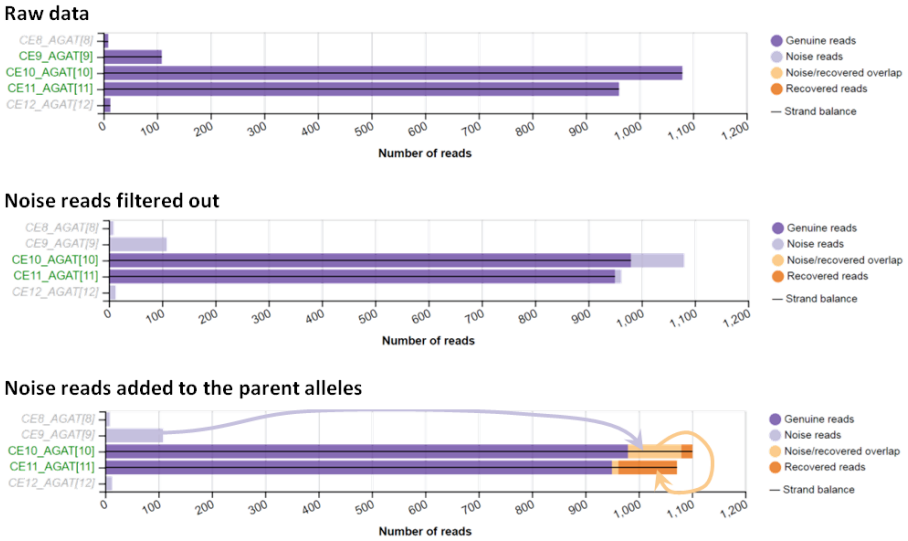


Sequence profiles for three different stages of the analysis for a simple reference sample without overlap between stutter and genuine alleles.

On top, *raw read counts are displayed for each observed variant.*
In the middle*, noise reads are filtered out (based on the observed reproducible noise for each allele in the reference database). Filtered noise reads are displayed in light purple.*
At the bottom*, filtered reads are added to the parent allele (as determined by the noise profiles) as recovered reads marked in dark orange.The lines in the bars indicate the strand balance; the line is drawn near the top of the bar if the majority of reads of a sequence is on the forward strand, near the bottom of the bar if the majority of reads is on the reverse strand, and in the middle of the bar in the absence of strand bias. Sequences displayed in green in the graphs are the alleles that the software infers to be genuine alleles in the sample, based on a threshold of 1.5% of the total number of reads of the locus.*

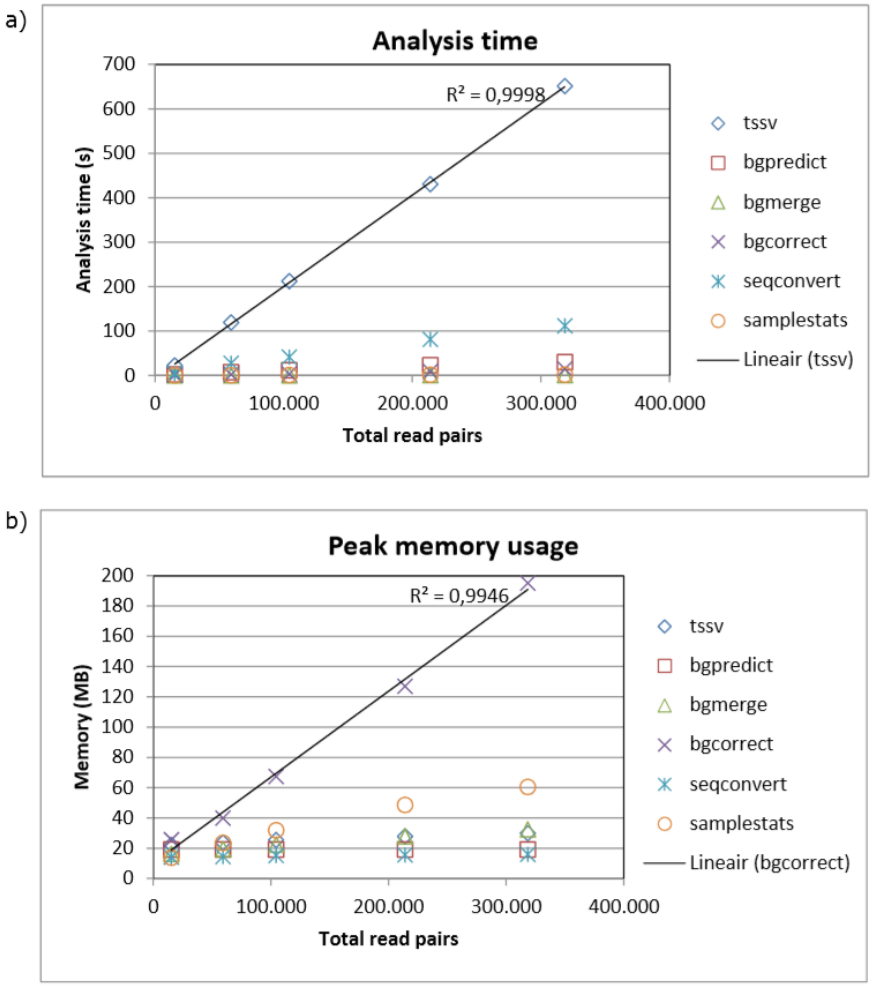
Chapter 5

Supplementary Figure 5b – Explanation of a sequence profile for raw, filtered and corrected data



Sequence profiles for three different stages of the analysis for a reference sample with overlap between stutter and genuine alleles.
***On top**, raw read counts are displayed for each observed variant.*
***In the middle**, noise reads are filtered out (based on the observed reproducible noise for each allele in the reference database). Filtered noise reads are displayed in light purple. Note that part of allele CE10 is filtered out as noise from allele CE11.*
***At the bottom**, filtered reads are added to the parent allele (as determined by the noise profiles) as recovered reads marked in dark orange. For allele CE10, part of the reads are removed as noise, but some reads are recovered as well. The overlap of this filtered noise and recovered reads is marked in light orange. The lines in the bars indicate the strand balance; the line is drawn near the top of the bar if the majority of reads of a sequence is on the forward strand, near the bottom of the bar if the majority of reads is on the reverse strand, and in the middle of the bar in the absence of strand bias. Sequences displayed in green in the graphs are the alleles that the software infers to be genuine alleles in the sample, based on a threshold of 1.5% of the total number of reads of the locus.*

Supplementary Figure 6 – Required time and memory for the analysis of case samples



The dot plots display the registered time (a) and the peak memory usage (b) of analysis for five samples for each tool of the standard casework analysis pipeline. The analysis was performed using a single core of an Intel(R) Xeon(R) E5-2620 processor at 2.00 GHz. The analysis time is mostly consumed by TSSV and the highest memory demand is measured for the tool BGCorrect . Both the analysis time and memory increase more or less linearly when the coverage of a sample is increased.

Supplementary Table 1 – Currently available tools and visualisations in FDSTools

General	
Pipeline	Automatically run complete, predefined analysis pipelines using the other tools in the package. Recommended starting point for new users.
TSSV	Link raw reads in a FastA or FastQ file to markers and count the number of reads for each unique sequence. Wrapper around the TSSV-Lite program.
Vis	Create a data visualisation web page or Vega graph specification.
Seqconvert	Convert between raw sequences, TSSV-style sequences (shortened notation of STRs), and allele names.
BGMerge	Merge multiple files containing background noise profiles. Used to extend the output of BGEstimate with output of BGPredict.
Library	Create an empty FDSTools library file.
Libconvert	Convert between TSSV and FDSTools library file formats. Primarily useful for users migrating from the older, standalone TSSV programme to FDSTools.
Reference sample analysis	
Stuttermark	Mark potential stutter products by assuming a fixed maximum percentage of stutter product with respect to the parent sequence. Used to mask stutter products for Allelefinder.
Allelefinder	Find true alleles in reference samples and detect possible contaminations.
BGHomRaw	Compute noise ratios for all noise detected in homozygous reference samples.
BGHomStats	Compute allele-centric statistics for background noise in homozygous reference samples (min, max, mean, sample variance).
BGEstimate	Estimate allele-centric background noise profiles (means) from reference samples.
Stuttermodel	Train a stutter prediction model using homozygous reference samples.
BGAnalyse	Find contaminated or otherwise unsuitable reference samples.
Case sample analysis	
BGPredict	Predict background profiles of new alleles based on a model of stutter occurrence obtained from Stuttermodel.
BGCorrect	Match background noise profiles to samples to filter and correct for systemic noise.
FindNewAlleles	Mark all sequences that are not in a list of known (e.g., previously encountered) allelic sequences.
Samplestats	Compute various statistics for each sequence in a given sample data file. Can also call alleles and filter sequences using these statistics.

Visualisations	
Samplevis	Visualise and interpret sample data files. The web page version of Samplevis offers many features to help interpreting the sample, such as automatically marking sequences that match given criteria as 'allele'.
Profilevis	Visualise background noise profiles obtained with BGEstimate, BGHomStats, and BGPredict.
BGRawvis	Visualise raw background noise data obtained with BGHomRaw.
Stuttermodelvis	Visualise models of stutter obtained from Stuttermodel.
Allelevis	Visualise the allele list obtained from Allelefinder as a graph in which the nodes correspond to alleles and the edges correspond to heterozygous samples combining the connected alleles.
BGAnalysevis	Visualise remaining background noise levels as obtained from BGAnalyse.

Supplementary Table 2 – Effects of criteria for admission of alleles to noise profile estimation

Number of alleles for which a BGEstimate noise profile can be obtained in our 429 sample reference set when applying different criteria. These comprise the minimum number of different heterozygous genotypes per allele, minimum number of samples per allele and minimum number of homozygous samples per allele. When a criterion is varied, the other criteria are kept at the minimum value possible which is at least 1 heterozygous genotype per allele, 1 sample per allele and 0 homozygous samples per allele. The criterion of the minimum number of different heterozygotes per allele does not apply if the allele is present in at least one homozygote.

When the settings are more stringent, BGEstimate noise profiles are obtained for fewer alleles. The results for the settings selected for this study are indicated in the rightmost column labelled 'used settings' and represent at least 3 different heterozygous genotypes per allele or, if a homozygote is available, at least 2 samples per allele.
Note that three different alleles have been detected for the gender locus Amel, which is due to the detection of 2 sequence variants for the X allele.

Criteria	Heterozygous genotypes per allele (or one homozygote available)					Samples per allele (homozygous or heterozygous)					Homozygous samples per allele (situation for *Stuttermodel*)					Used settings
	1+	2+	**3+**	4+	5+	1+	**2+**	3+	4+	5+	1+	2+	3+	4+	5+	
Locus	Alleles															
Amel	3	1	1	1	1	3	3	3	3	3	1	1	1	1	1	1
CSF1P0	10	8	6	5	4	10	8	6	6	6	4	4	4	4	3	6
D10S1248	11	7	6	6	6	11	7	6	6	6	5	4	4	4	3	6
D12S391	61	44	36	29	27	61	45	36	31	28	10	5	3	2	2	36
D13S317	18	14	14	14	13	18	14	14	14	14	7	7	7	6	6	14
D16S539	11	11	11	11	11	11	11	11	11	11	8	4	4	4	4	11
D18S51	18	15	13	13	12	18	15	14	14	14	7	7	7	6	5	13
D19S433	18	15	11	10	10	18	16	14	13	10	5	4	3	3	3	11
D1S1656	29	22	19	17	17	29	22	20	18	17	13	7	4	3	2	19
D21S11	45	28	21	18	15	45	29	22	18	17	9	7	3	3	3	21
D22S1045	11	9	7	7	6	11	11	8	8	8	5	4	3	3	3	7
D2S1338	42	28	23	22	17	42	28	24	22	22	10	8	5	4	3	23
D2S441	17	13	10	10	9	17	14	12	10	10	6	4	4	3	3	10
D3S1358	19	16	15	14	11	19	16	15	14	13	8	5	5	3	3	15
D5S818	23	18	17	14	13	23	18	17	16	15	8	5	4	3	3	17
D7S820	21	19	17	16	15	21	19	17	16	16	8	6	5	5	4	17
D8S1179	23	19	17	14	14	23	19	18	18	16	9	5	5	4	4	17
DYS391	6	6	6	6	6	6	5	5	4	4	6	5	5	4	4	5
FGA	18	15	13	10	9	18	15	14	11	9	7	7	6	4	4	13
PentaD	18	14	13	10	9	18	14	13	13	11	6	6	5	5	5	13
PentaE	18	16	16	15	14	18	16	16	15	14	11	9	8	7	4	16
TH01	7	7	7	7	5	7	7	7	7	7	5	5	5	5	4	7
TPOX	7	6	6	6	6	7	6	6	6	6	4	3	3	3	2	6
vWA	22	16	15	11	9	22	16	15	14	12	5	5	5	3	3	15