Short Read Alignment Based on Maximal Approximate Match Seeds

Wei Quan¹, Dengfeng Guan¹,²†, Guangri Quan¹, Bo Liu¹ and Yadong Wang¹*

¹ School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China, ² Institute of Zoology, Chinese Academy of Sciences, Beijing, China

Sequence alignment is a critical step in many critical genomic studies, such as variant calling, quantitative transcriptome analysis (RNA-seq), and metagenomic sequence classification. However, the alignment performance is largely affected by repetitive sequences in the reference genome, which extensively exist in species from bacteria to mammals. Aligning repeating sequences might lead to tremendous candidate locations, bringing about a challenging computational burden. Thus, most alignment tools prefer to simply discard highly repetitive seeds, but this may cause the true alignment to be missed. Using maximal approximate matches (MAMs) as seeds is an option, but MEMs seeds may fail due to sequencing errors or genomic variations in MEMs seeds. Here, we propose a novel sequence alignment algorithm, named MAM, which can efficiently align short DNA sequences. MAM first builds a modified Burrows-Wheeler transform (BWT) structure of a reference genome to accelerate approximate seed matching. Then, MAM uses maximal approximate matches (MAMs) seeds to reduce the candidate locations. Finally, MAM applies an affine-gap-penalty dynamic programming to extend MAMs seeds. Experimental results on simulated and real sequencing datasets show that MAM achieves better performance in speed than other state-of-the-art alignment tools. The source code is available at https://github.com/weiquan/mam.

Keywords: whole-genome resequencing, next-generation sequencing, repeats, sequence alignment, FM-index

1. INTRODUCTION

The development of next-generation sequencing (NGS) technologies has led to a rapid decline in the sequencing cost and had a tremendous impact on genomic research (Morozova and Marra, 2008; Reinert et al., 2015). There has been an intense effort in recent years to develop computational methods and applications to meet the increasing demands for sequencing data analysis (Flicek and Birney, 2009). One of these fundamental tasks is sequence alignment. Sequence alignment is one of the most critical steps necessary to process NGS data, and the alignment accuracy has a very large impact on downstream applications, such as variant calling (Dalca and Brudno, 2010), quantitative measurement of RNA-seq (Pepke et al., 2009), eQTL analysis (Wang et al., 2019), and metagenomic analysis (Breitwieser et al., 2019; Cheng et al., 2020). During the past decades, many alignment methods have been proposed to improve the efficiency and accuracy of sequence alignment, including but not limited to Maq (Li et al., 2008a), SOAP (Li et al., 2008b), Bowtie (Langmead et al., 2009), BWA (Li and Durbin, 2009), and mrsFAST (Hach et al., 2010). With these developments, the performance of alignment tools has been greatly improved with respect to speed, sensitivity and accuracy (Li et al., 2009; Schadt et al., 2010; Langmead and Salzberg, 2012; Xiao et al., 2017). However, aligning repetitive DNA sequences accurately to the reference genome remains a major issue.
Repetitive DNA sequences are multiple copies of sequences with high similarity that occur throughout the genome. Repetitive DNA sequences are highly abundant in a broad range of species, from bacteria to mammals. For example, nearly 40% of bacterial genomes and 50% of human genomes are composed of repetitive sequences (Treangen and Salzberg, 2012). These repetitive patterns could cause several computational challenges for sequence alignment, which would result in loss of information related to essential biological phenomena. From a computational perspective, repeats may bring out ambiguous candidate positions, which will decrease alignment speed. In recent years, several alignment tools use maximal exact matches (MEMs) or super maximal exact matches (SMEMs) to reduce the number of candidate positions (Liu and Schmidt, 2012; Li, 2013). However, these approaches may reduce the sensitivity in complex genomic regions.

In this article, we build a novel index, named MAM-index, for a reference genome based on a modified Burrows-Wheeler transform (BWT) structure. MAM-index combines the conventional FM-index and an auxiliary data structure, which accelerates approximate string matching. We propose a MAMs based seeding algorithm to calculate the candidate locations. We present a novel sequence alignment tool, named MAM, that can efficiently align highly repetitive DNA sequences by using maximal approximate matches (MAMs) as seeds. To evaluate the performance of MAM, we performed alignment experiments on both simulated and real datasets, and compared our results with those of Bowtie and BWA-MEM. Experiments on simulated and real sequencing datasets show that MAM achieves better performance in speed than the other alignment programs.

2. MATERIALS AND METHODS

2.1. Overview of MAM

MAM uses a hierarchical index (MAM-index), which consists of two types of indexes: a global index (global FM-index) and numerous local indexes (local FM-indexes). MAM follows the canonical seed-and-extend paradigm. In the seeding stage, it initially seeds an alignment with exact matches of length $l_{init}$ via the global FM-index. To reduce ambiguous candidate locations caused by highly repetitive seeds, we introduce maximal approximate matches as seeds to re-seed all initial seeds with more than $m$ occurrences. The overview of MAM seeding is shown in Figure 1: (1) Search a $l_{init} = 20$ bp initial seed based on the global FM-index, and obtain global suffix array interval $[bg, ed]$. (2) Find the local FM-index that corresponds to suffix array interval $[bg, ed]$. (3) Use local FM-index to obtain the $l_{ext} = 16$ bp extended seed. (4) Repeat 3 until the seed is not extensible. After the seeding stage, we use affine-gap-penalty dynamic programming (DP) to extend all MAMs seeds with no more than $m$ occurrences, and perform the best alignment.

2.2. Construction of the MAM-Index

The structure of the MAM-index consists of two parts: a global index (global FM-index) and numerous local indexes (local FM-indexes). A conventional FM-index is employed as the global index part of the MAM-index. The FM-index was proposed by Paolo Ferragina and Giovanni Manzini in 2000 (Ferragina and Manzini, 2000). The FM-index is a self-indexing index based on the Burrows-Wheeler transform (BWT), and it can be used to efficiently find the occurrences of a pattern in the case of low RAM. Below, we will focus on the construction of the local index.

The local index consists of a BWT-like data structure (sBWT) and a relation array (RA). The sBWT is a variation of the BWT, which is used to index multiple strings of equal length. Given a seed, we employ an sBWT to index predecessor sequences and successor sequences of the seed and a RA to store the relation information between successor sequences and predecessor sequences of the seed.

2.2.1. Construction of the sBWT

2.2.1.1. Building predecessor and successor sets

Given the $i$-th iteration seed $s_i$ which occurs more than $k$ times in the reference genome sequence, the $l_{ext}$ length predecessor sequences of $s_i$ are the $l_{ext}$ length predecessors of $s_i$ at all occurrences of $s_i$, denoted $pred(s_i, l_{ext})$. Similarly, the $l_{ext}$ length successor sequences of $s_i$ are the $l_{ext}$ length successors of $s_i$ at all occurrences of $s_i$, denoted $succ(s_i, l_{ext})$.

The predecessor sequence set construction algorithm is shown in Algorithm 1. Similarly, the successor sequence set construction algorithm can be implemented by modifying the code for fetching local sequences from the reference genome sequence. To better illustrate the algorithm, we use a 140 bp reference genome sequence as an example; see Figure 2.

In Figure 2, sequence CGACTA marked in red is an initial iteration seed with 7 occurrences in the 140 bp reference genome. The predecessor sequences and successor sequences of CGACTA are shown in Figure 3A. The L-Ext column lists the predecessor sequences of CGACTA, and the R-Ext column lists the successor sequences of CGACTA. The Pos sub-column lists the position of the sequence in the reference genome.
genome sequence. For duplicate sequences, only one copy of the duplicate sequences is stored in the predecessor sequence set; see Figure 3B. In Figure 3B, the local sequence 'TCAT' in L-Ext.

2.2.1.2. Building sBWT for predecessor and successor sets
The sBWT of predecessor and successor sequences is shown in Figures 4A, B, respectively. The construction process of sBWT is similar to that of the Burrows-Wheeler matrix. The $i$-th
Algorithm 2: buildSbwt

Input: A sorted set S of n sequences of equal length l
Output: sBWT
1 $n_{rot} \leftarrow l$
2 // S is sorted in dictionary order, and indexes of sequences in S are stored in SA
3 $SA^0 \leftarrow$ sort (S)
4 for $i \leftarrow 0$ to $n$
5    $sBWT^0[i] \leftarrow S[SA^0[i]]$
6 end
7 for $rot \leftarrow 1$ to $n_{rot}$ do
8    $SA^{rot} \leftarrow$ radixSort ($SA^{rot-1}$, S)
9    for $i \leftarrow 0$ to $n$
10       $sBWT^{rot}[i] \leftarrow S[SA^{rot}[i]]$
11 end
12 return sBWT

iteration transform is performed by sorting local sequences of $i$ rotations in lexicographic order. The L-Seq column in Figure 4A is the predecessor sequences set, and Rot0 is the 0-th iteration transform of L-Seq. The construction algorithm of the sBWT is shown in Algorithm 2.

2.2.2. Construction of the Relation Array
sBWT can be used to find the occurrences of predecessor and successor sequences. However, the searched predecessor and successor sequences may not be proper paired, which may lead to ambiguous seed extension. For this reason, we use a relation data structure (RA) to store relation information between the predecessor and successor sequences.

The RA consists of LtoRel and Rel. LtoRel[i] stores the initial index of the predecessor sequence in Rel. Given an $i$-th iteration seed $s_i$ and a next iteration seed $s_{i+1} = \text{pred}(s_i) + s_i + \text{succ}(s_i)$, if $s_{i+1}$ occurs in the reference genome sequence, then $\text{pred}(s_i)$ and $\text{succ}(s_i)$ can be searched via sBWT. Supposing that $\text{pred}(s_i)$ is $\text{rank}_{\text{pred}}$ ranked in all predecessor sequences and $\text{succ}(s_i)$ is $\text{rank}_{\text{succ}}$ ranked in all successor sequences, $\text{rank}_{\text{best}}$ should be a value of $\text{LtoRel}[j]$ for $j$ in the range $[\text{rank}_{\text{pred}}, \text{rank}_{\text{pred}} + 1)$. Rel[i] stores the pointer to $s_{i+1}$ in the MAM-index. With Rel[i], the seed jumps to the next iteration. The RA of seed CGACTA is shown in Figure 5.

2.3. Alignment With MAM
MAM follows the canonical seed-and-extend paradigm. MAM generates every $x$ nt exact match seeds in reads. If seeds occur more than $k$ times, MAM iteratively extends seeds by finding approximate matches in both directions until the occurrences of seeds are less than $k$ or seeds are not extensible. After seeding, MAM generates locations of seeds. All locations are sorted in ascending order, and all locations whose distances are less than $d$ are placed within the same chain. Then, all chains are sorted by seeds number in descending order. After chains are sorted, MAM performs a striped Smith-Waterman algorithm for the top $k$ chains. We choose the best score location as the best alignment.

The algorithm of alignment with MAM is shown in Algorithm 3. In Algorithm 3, the function backwardSearchGlobal is a conventional backward search algorithm, and the function backwardSearchLocal is shown in Algorithm 4. $\text{C}(s_{\text{bwt}}^i, c)$ is a function that contains the number of occurrences of lexically smaller characters in the $i$-th rotation sBWT. The function $\text{Occ}(s_{\text{bwt}}^i, k, c)$ is the number of occurrences of character $c$ in the $s_{\text{bwt}}^i[1:k]$.

3. RESULTS
We have implemented MAM to align short reads to a reference genome. The default output format is SAM format. MAM is distributed under the GNU General Public License (GPL). The source code is available at https://github.com/weiquan/mam.
The performance of MAM has been compared with those of the three most widely used alignment tools BWA-MEM (version 0.7.17) and BOWTIE2 (version 2.3.5). All aligners were tested on two simulated datasets and two high-throughput sequencing (HTS) datasets to assess their speed, sensitivity, and accuracy. All benchmarks were conducted on a desktop computer with 32 GB of RAM and a 3.30 GHz Intel i9-7900X processor with a total of 10 CPU cores running Linux Ubuntu 18.04.

### 3.1. Evaluation on Simulated Datasets

We simulated 4 million 100 and 150 bp Illumina-like reads from the human genome GRCh38 using Mason2 (Holtgrewe, 2010) with a 0.1% SNP mutation rate, a 0.02% indel mutation rate and a 2% sequencing base error rate. We ran MAM, BWA-MEM and BOWTIE2 with the default settings. A read is defined as a mapped read if the read is mapped with at least one alignment. An alignment is defined as a good alignment if the alignment position is the true position. The sensitivity and accuracy are defined as the following percentages:

\[
\text{sensitivity} = \frac{\#\text{mapped}}{\#\text{reads}} \times 100\%
\]

\[
\text{accuracy} = \frac{\#\text{good}}{\#\text{mapped}} \times 100\%
\]

Table 1 shows that BWA-MEM (100.00%) is more sensitive than MAM (99.90–99.97%) and BOWTIE2...
(99.01–99.39%) on both the sim-100 and sim-150 dataset. BWA-MEM(95.06–96.24%) is the most accurate on both the sim-100 dataset and the sim-150 dataset. With respect to speed, MAM is the fastest on both the sim-100 dataset and sim-150 dataset. On memory, BOWTIE2 uses (3.22–3.23 GB), smaller than BWA-MEM (5.26–5.39 GB), and MAM (16.54–16.59 GB).

### 3.2. Evaluation on HTS Datasets

We benchmarked all aligners on two HTS datasets to assess the performance on real datasets. We mapped 4 million 100 bp reads sequenced with the Illumina HiSeq 2000 (SRA ID: ERR037900) and 4 million 148 bp reads sequenced with the Illumina HiSeq 2000 (SRA ID: SRR1766443) to the human reference genome (GRCh38). All aligners were run with the
Algorithm 3: Alignment with MAM

**Input:** A MAM-index Index, read sequence Read, and parameter max_loc, the max locate number

**Output:** Suffix array interval

```plaintext
1 // Initialization
2 l0 ← initial seed length
3 l1 ← extend length
4 n ← len(Read)
5 gBWT ← gGetGlobal(Index)
6 for i ← 0 to n do
7     x ← i
8     y ← i + l0
9     s ← Read[x:y]
10    [bg, ed] ← backwardSearchGlobal (gBWT, s)
11   while ed − bg > max_loc and x − l1 > 0 and
12       y + l1 < n do
13       sBWT_pred, sBWTsucc ← getLocal(Index, s)
14       [bg0, ed0, bg1, ed1] ← backwardSearchLocal
15            (sBWT_pred, Read[x − l1:x])
16       [bg1, ed1] ← backwardSearchLocal
17            (sBWTsucc, Read[y:y + l1])
18   // check whether seed could be extensible in both
directions if checkPairing
19       (Index, bg0, ed0, bg1, ed1) then
20       x ← x − l1
21       y ← y + l1
22       s ← Read[x:y]
23   // convert local index interval to global index
24     interval
25       [bg, ed] ← localToGlobal (Index, s, bg0, ed0, bg1, ed1)
26 end
27 aln_len ← aln_len + 1
28 return [bg, ed]
```

Algorithm 4: backwardSearchLocal

**Input:** A sBWT sbwt of multiple sequences of equal
length l and sequence seq of length l

**Output:** Interval of sBWT

```plaintext
1 for i ← 0 to l do
2    rot = i
3    aln_len = 0
4   while bg < ed and aln_len < l do
5       c ← seq[l − 1 − rot]
6       bg ← (sbwtrot, c) + l ⊕ c (sbwtrot, bg, c)
7       ed ← (sbwtrot, c) + l ⊕ c (sbwtrot, ed, c)
8       rot ← (rot + 1) (mod l)
9       aln_len ← aln_len + 1
10 end
11 end
12 return [bg, ed]
```

TABLE 1 | Evaluation on simulated data.

| Program      | sim-100 | sim-150 | sim-100 | sim-150 |
|--------------|---------|---------|---------|---------|
|              | Sen (%) | Acc (%) | Time     | Mem     | Sen (%) | Acc (%) | Time     | Mem     |
|              |         |         |         |         |         |         |         |         |
| MAM          | 99.90   | 94.65   | 12m28s  | 16.54GB | 99.97   | 95.96   | 23m13s  | 16.59GB |
| BWA-MEM      | 100.00  | 95.06   | 16m20s  | 5.39GB  | 100.00  | 96.24   | 39m43s  | 5.26GB  |
| BOWTIE2      | 99.01   | 93.88   | 22m66s  | 3.22GB  | 99.39   | 95.28   | 25m46s  | 3.23GB  |

Sen, alignment sensitivity; Acc, alignment accuracy; Mem, the peak memory usage of the tool.

TABLE 2 | Evaluation on real data.

| Program      | real-100 | real-148 | real-100 | real-148 |
|--------------|----------|----------|----------|----------|
|              | Sen (%)  | Acc (%)  | Time     | Mem     | Sen (%)  | Acc (%)  | Time     | Mem     |
|              |          |          |          |         |          |          |          |         |
| MAM          | 99.83    | 95.89    | 8m59s    | 16.52GB | 98.85    | 96.21    | 24m37s   | 16.59GB |
| BWA-MEM      | 99.99    | 95.87    | 18m18s   | 5.39GB  | 99.51    | 96.27    | 21m57s   | 5.24GB  |
| BOWTIE2      | 99.62    | 94.54    | 13m37s   | 3.22GB  | 98.11    | 97.32    | 24m29s   | 3.23GB  |

Sen, alignment sensitivity; Acc, alignment accuracy; Mem, the peak memory usage of the tool.

default settings. As the true alignment locations of reads are
not known in real datasets, we define that an alignment is a
good alignment if the alignment score between the read and the
reference is higher than 85% of the highest score (e.g., if the
alignment score of a 100 bp read is higher than 85, it is regarded
as a perfect alignment).

The results are shown in Table 2. Table 2 shows that
MAM (95.89%) is the most accurate on the real-100 datasets,
and BOWTIE2 (97.32%) is the most accurate on the real-
148 datasets. BWA-MEM (99.51–99.99%) is the most
sensitive on both the real-100 and real-148 datasets. With
respect to speed, MAM (8m59s) is the fastest on the real-
100bp dataset, and BWA-MEM (21m57s) is the fastest on the
real-148 dataset. On memory, BOWTIE2 uses (3.22–
3.23 GB), smaller than BWA-MEM (5.24–5.39 GB), and
MAM (16.52–16.59 GB).

4. DISCUSSION

Enormous amounts of short read aligners have been developed
for fast and accurate alignment of reads to a reference genome.
However, aligning repetitive DNA sequences to the reference
genome is still a concern. Some of the aligners employ
MEMs seeds to reduce candidate locations in repetitive regions.
However, MEMs seeds may fail due to genomic variations and
sequencing errors. To this purpose, we use MAMs seeds to
filter candidate locations in the seeding stage. Although MAMs
seeds could be searched via the conventional FM-index, it
is ineffective.

Herein, we propose a variation FM-index (MAM-index) to
search MAMs seeds quickly and present a short read alignment
tool (MAM). We have demonstrated the performance of MAM
on aligning sequences to the human genome, and compared
it with the most widely used alignment tools, BWA-MEM and BOWTIE2. The result shows MAM is more efficient than BWA-MEM and BOWTIE2 with similar accuracy and sensitivity. In addition, accuracy and sensitivity of MAM could be improved by using shorter initial seed length, which means MAM has the potential to align sequences to complex genomic regions. Although MAM requires more memory than BWA-MEM and BOWTIE2, memory is not a practical concern on modern computer servers.

**DATA AVAILABILITY STATEMENT**

All datasets generated for this study are included in the article/supplementary material.

**REFERENCES**

Breitwieser, F. P., Lu, J., and Salzberg, S. L. (2019). A review of methods and databases for metagenomic classification and assembly. Brief. Bioinform. 20, 1125–1136. doi: 10.1093/bib/bbx120

Cheng, L., Qi, C., Zhuang, H., Fu, T., and Zhang, X. (2020). gutMDisorder: a comprehensive database for dysbiosis of the gut microbiota in disorders and interventions. Nucleic Acids Res. 48, D554–D560. doi: 10.1093/nar/gkz843

Dalca, A. V., and Brudno, M. (2010). Genome variation discovery with high-throughput sequencing data. Brief. Bioinform. 11, 3–14. doi: 10.1093/bib/bbp058

Ferragina, P., and Manzini, G. (2000). “Opportunistic data structures with applications,” in Proceedings 41st Annual Symposium on Foundations of Computer Science (Redondo Beach, CA: IEEE), 390–398.

Flicek, P., and Birney, E. (2009). Sense from sequence reads: methods for alignment and assembly. Nat. Methods 6, 56–512. doi: 10.1038/nmeth.1376

Hach, F., Hormozdiari, F., Alkan, C., Hormozdiari, F., Birol, I., Eichler, E. E., et al. (2010). mrsFAST: a cache-oblivious algorithm for short-read mapping. Nat. Methods 7:576. doi: 10.1038/nmeth0810-576

Holtgreve, M. (2010). Technical Report. Institut für Mathematik und Informatik, Freie Universität Berlin. Mason - A Read Simulator for Second Generation Sequencing Data.

Langmead, B., and Salzberg, S. L. (2012). Fast gapped-read alignment with bowtie 2. Nat. Methods 9, 357–359. doi: 10.1038/nmeth.1923

Langmead, B., Trapnell, C., Pop, M., and Salzberg, S. L. (2009). Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. Genome Biol. 10:R25. doi: 10.1186/gb-2009-10-3-r25

Li, H. (2013). Aligning sequence reads, clone sequences and assembly contigs with bwa-mem. arXiv preprint arXiv:1303.3997.

Li, H., and Durbin, R. (2009). Fast and accurate short read alignment with burrows-wheeler transform. Bioinformatics 25, 1754–1760. doi: 10.1093/bioinformatics/btp324

Li, H., Ruan, J., and Durbin, R. (2008a). Mapping short DNA sequencing reads and calling variants using mapping quality scores. Genome Res. 18, 1851–1858. doi: 10.1101/gr.078212.108

Li, R., Li, Y., Kristiansen, K., and Wang, J. (2008b). Soap: short oligonucleotide alignment program. Bioinformatics 24, 713–714. doi: 10.1093/bioinformatics/btn205

**AUTHOR CONTRIBUTIONS**

WQ and GQ designed the algorithm of indexing and seeding. WQ implemented the algorithm and performed in silico experiments. DG co-wrote the manuscript and co-performed the experiments. BL and YW revised the manuscript and provided funding support.

**FUNDING**

This work has been supported by the National Key Research and Development Program of China (Nos: 2017YFC1201201, 2018YFC0910504, and 2017YFC0907503).

Li, R., Yu, C., Li, Y., Lam, T.-W., Yu, S.-M., Kristiansen, K., et al. (2009). Soap2: an improved ultrafast tool for short read alignment. Bioinformatics 25, 1666–1677. doi: 10.1093/bioinformatics/btp336

Liu, Y., and Schmidt, B. (2012). Long read alignment based on maximal exact match seeds. Bioinformatics 28, i318–i324. doi: 10.1093/bioinformatics/bts414

Morozova, O., and Marra, M. A. (2008). Applications of next-generation sequencing technologies in functional genomics. Genomics 92, 255–264. doi: 10.1016/j.ygeno.2008.07.001

Pepke, S., Wold, B., and Mortazavi, A. (2009). Computation for ChIP-seq and RNA-seq studies. Nat. Methods 6, S22–S32. doi: 10.1038/nmeth.1371

Reinert, K., Langmead, B., Weese, D., and Evers, D. J. (2015). Alignment of next-generation sequencing reads. Annu. Rev. Genomics Hum. Genet. 16, 133–151. doi: 10.1146/annurev-genom-090413-025358

Schadt, E. E., Turner, S., and Kasarskis, A. (2010). A window into third-generation sequencing. Hum. Mol. Genet. 19, R227–R240. doi: 10.1093/hmg/ddq416

Treangen, T. J., and Salzberg, S. L. (2012). Repetitive DNA and next-generation sequencing: computational challenges and solutions. Nat. Rev. Genet. 13, 36–46. doi: 10.1038/nrg3117

Wang, T., Peng, Q., Liu, B., Liu, X., Liu, Y., Peng, J., et al. (2019). eQTLMAP: fast and accurate eQTL mediation analysis with efficient permutation testing approaches. Front. Genet. 10:1309. doi: 10.3389/fgene.2019.01309

Xiao, C.-L., Chen, Y., Xie, S.-Q., Chen, K.-N., Wang, Y., Han, Y., et al. (2017). MECAT: fast mapping, error correction, and de novo assembly for single-molecule sequencing reads. Nat. Methods 14:1072. doi: 10.1038/nmeth.4432

**Conflict of Interest:** The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Copyright © 2020 Quan, Guan, Quan, Liu and Wang. This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY). The use, distribution or reproduction in other forums is permitted, provided the original author(s) and the copyright owner(s) are credited and that the original publication in this journal is cited, in accordance with accepted academic practice. No use, distribution or reproduction is permitted which does not comply with these terms.