EZcount: an All-in-one Software for microRNA Expression Quantification from NGS Sequencing data*

Filippo Geraci[a,*,1], Giovanni Manzini[b,a,*,1]

[a]*Institute for Informatics and Telematics, CNR, Pisa, 56124, Italy*
[b]*Department of Computer Science, University of Pisa, Pisa 56127, Italy*

ARTICLE INFO

Keywords:
MicroRNA
Algorithms
Transcriptomics
Next-generation sequencing

ABSTRACT

MicroRNAs (miRNAs) are short endogenous molecules of RNA that influence cell regulation by suppressing genes. Their ubiquity throughout all branches of the tree of life has suggested their central role in many cellular functions. Nowadays, several personalized medicine applications rely on miRNAs as biomarkers for diagnoses, prognoses, and prediction of drug response. The increasing ease of sequencing miRNAs contrasts with the difficulty of accurately quantifying their concentration. The use of general purpose aligners is only a partial solution as they have limited possibilities to accurately solve ambiguous mapping due to the short length of these sequences.

We developed *EZcount*, an all-in-one software that, with a single command, performs the entire quantification process: from raw fastq files to read counts. Experiments show that *EZcount* is more sensitive and accurate than methods based on sequence alignment, independently of the library preparation protocol and sequencing machine. The parallel architecture of *EZcount* makes it fast enough to process a sample in minutes using a standard workstation.

EZcount runs on all of the most common operating systems (Linux, Windows and MacOS) and is freely available for download at `https://gitlab.com/BioAlgo/miR-pipe`

A detailed description of the datasets, the raw experimental results, and all the scripts used for testing are available as supplementary material.

1. Introduction

MicroRNAs (miRNAs) belong to a class of small non-coding RNAs, with length ranging approximately between 18 and 26 nucleotides, involved in the regulatory process as they are capable of inhibiting translation and, consequently, silencing genes.

The massive presence of this class of RNAs in several functional pathways [4] and their ubiquity among different organisms (currently miRBase [8], the largest specialized database, includes microRNA from about 270 organisms), have attracted the attention of the research community which agrees to consider miRNA analysis as a tool that will become fundamental in the diagnosis and treatment of several complex diseases. For example, applications in oncology have already demonstrated that miRNA expression profiles can be used successfully as non-invasive biomarkers [31, 23] when quantified from circulating tumor cells collected by means of *liquid biopsy* [22]. In addition, the constant improvement of small-RNA sequencing protocols has made the measurement of the expression level of the entire miRNAome an inexpensive routine.

Quantifying the expression profile of miRNAs from raw reads, however, poses several algorithmic challenges. Because they are short, these sequences can occur in multiple locations throughout the reference genome, causing an aligner to make erroneous or ambiguous placements. The need to accept imperfect matching either to align polymorphic miRNAs [19], or to handle substitutions due to A-to-I editing [10], or to recover sequencing errors, as well as the high degree of pairwise similarity among some groups of miRNAs, make the problem more complicated. A further complication is the fact that some miRNAs may also be subsequences of others.

Setting stringent alignment parameters and filtering out ambiguous or imperfect reads can be useful in reducing read/miRNA misassignments, but have the disadvantage of potentially discarding perfectly legal reads and, as a result,

* Postprint version; final publication available on ScienceDirect `https://doi.org/10.1016/j.compbiomed.2021.104352`

*Corresponding author
`filippo.geraci@iit.cnr.it` (F. Geraci); `giovanni.manzini@unipi.it` (G. Manzini)
ORCID(s): `0000-0001-6993-6761` (F. Geraci); `0000-0002-5047-0196` (G. Manzini)

[1] Equal contribution, ordered alphabetically.

underestimating (or even altering) the expression profile. This, in turn, would cause a drop in sensitivity by making moderately expressed miRNAs undetectable.

A typical example in this regard is that of isoMIRs [21] where either the 5' or 3' endpoint can be trimmed or slightly modified [29]. Treating the clipped endpoints as errors may result in the loss of valid reads. An overly permissive alignment, on the other hand, may map reads on multiple miRNAs, therefore only postponing the problem of deciding on the correct assignment. At the user level, miRNA counting can be a complex and time-consuming activity due to the numerous processing steps involved and to the number of parameters that need to be set. Typical pipelines include: an initial check to remove low-quality reads, a pre-processing in which adapter sequences are trimmed, the alignment of reads against a reference genome, and, finally, the quantification of the number of reads supporting each miRNA. Although bioinformatics tools for each of the above steps typically accept and return standard file formats, thereby making counting pipelines modular and easy to modify, each individual step requires careful design choices dependent on the specific library preparation and sequencing protocol. In order to simplify the design of the pipeline, some authors have conducted extensive comparative experiments (see for example [17] and [32]) and others have made available integrated pipelines (see for example miRDeep2 [7], *COMPSRA* [12] and *sRNAbench* [1]).

In this paper, we introduce *EZcount*, an all-in-one software specifically designed for microRNA expression quantification. The main novelty of *EZcount* is that of inverting the read/miRNA matching process: instead of aligning each read against (a portion of) the reference genome and then deriving the corresponding miRNAs, we seek the miRNA sequences within the reads. This approach offers several advantages. First, although at a higher computational cost, the read/miRNA match remains feasible even when the adapter sequence is not found. Second, it is more accurate than sequence alignment, especially in the presence of Unique Molecular Identifiers (UMIs) [26]. In fact, if not trimmed, aligning UMI sequences to the reference would introduce spurious mismatches at the endpoints, potentially causing the read to be rejected. Finally, our approach enables us to correctly associate miRNAs also to those reads that would have been discarded as low quality even though the majority of sequencing errors are accumulated outside the miRNA sequence. As reported in [24], this case is rather common as errors are not evenly distributed across positions. For example, the increasing probability of out-of-phase reads, typical of the sequencing-by-synthesis process, tends to concentrate errors at the end of the read, and, as a consequence, on the 3' adapter.

In addition to the algorithmic aspects, bioinformatics tools must also be user-friendly and easy to install in order to be useful [11].

Although bioinformatics experts are familiar with long, complex and computationally demanding pipelines, simpler solutions with a clean interface can be beneficial [25]. For example, as shown in [5], aggregating the pre-processing steps contributes to limiting system requirements, reduces execution time by optimizing I/O operations as well as to simplifying the pipeline. A second aspect of usability is the self-tuning of parameters according to the input characteristics. This aspect is important in those situations where the user needs to process data coming from different sources.

EZcount covers all these usability aspects by offering an all-in-one solution that, using very little system resources (as little as 100MB of RAM per CPU thread), allows the user to perform miRNA quantification from raw reads with a single command without the need for any pre-processing or data cleaning. Moreover, by combining our strategy of seeking for miRNAs within reads with the adaptive algorithm to detect the adapter sequence, *EZcount* does not need any a-priori knowledge about the sequencing protocol to perform counting and, consequently, it does not require any adjustment of the input parameters.

In order to demonstrate the effectiveness of *EZcount*, we conducted an exhaustive comparison with several commonly used methods on two publicly available datasets (SRP199350 and GSE123627) which altogether include five different library preparation protocols and three sequencing machines. For comparison, we tested three pipelines based on general purpose aligners (*BWA*, *Bowtie 2* and *Subread*) as well as three integrated solutions (*mirdeep2* [7], *sRNAbench* [1] and *COMPSRA* [12])

Results show that, due to the ability to match reads with indels, *EZcount* consistently compares favourably with the other methods in terms of the number of correctly matched reads. This higher number of matches translates into a higher sensitivity that allows our method to quantify the expression level also of miRNAs with a small number of supporting reads.

Although in terms of running time *EZcount* is slower than most of the tested methods, its highly parallel architecture allowed us to process the largest sample in few minutes using a standard workstation.

2. Material and Methods

From an algorithmic point of view, the miRNA counting problem can be modeled as follows. We are given a set X of miRNA sequences and a collection of sequencing reads. Our basic assumption is that, disregarding errors in the sequencing, each read has the form $r = \alpha s_r \beta a \gamma$, where: $s_r \in X$, a is the adapter sequence, α, β are short random UMI sequences, and γ is arbitrary. Note that α, β and γ can be empty, while the adapter sequence a is the same for all reads but it is not always known beforehand.

Our task is to identify the miRNA sequence s_r for each read r, therefore *assigning* the read r to the sequence s_r. When all reads have been processed, we provide a counting profile giving for each sequence $s \in X$ the number of reads assigned to it. The task is made harder by sequencing errors in the reads and by the similarities in the miRNA sequences, since they can differ in only a few bases and some of them are substrings one of another. Whenever possible, we make use of the per nucleotide quality scores to assign reads in the ambiguous cases.

2.1. Keyword tree for exact string matching

Given the set X of miRNAs, our first step is to compute the subset $X' \subseteq X$ of *minimal* sequences. Formally, X' is the largest subset of X such that if $s \in X'$ then no proper substring of s is also in X'.

We then build the keyword tree with failure links $T(X')$ for the set X', as defined in [9, Sect. 3.4]. Given a read r, using $T(X')$ we can find whether r contains a substring identical to a sequence $s \in X'$ and, if such is the case, the identity of s and the position where s matches inside r. This computation takes $\mathcal{O}(|r|)$ time, where $|r|$ is the length of the read, and is therefore relatively fast since it does not depend on the size of X'. Note that the definition of X' implies that if r contains a substring identical to a sequence in X, it also contains a substring matching a sequence in X'.

2.2. Computation of the adapter

Our next step is to identify the adapter sequence from the set of input reads. Our algorithm requires no prior knowledge of the adapter: our only assumption is that a large enough fraction of the input reads contains a substring exactly matching one of the sequences in X', and that the adapter sequence begins shortly after this match.

We start by collecting statistics on the top k most frequent strings following sequences in X'. This is achieved by means of a slightly modified version of the *space-saving* data structure described in [18]. We scan reads from the input file and, for each read r, we use the keyword tree $T(X')$ to find whether a sequence in X' exactly matches a substring s of r. If this is the case, we partition r as $r = \alpha s \tau$.

Then we check if a (fixed size) prefix of τ is among the list of monitored keywords in the space-saving data structure and update it accordingly. In order to handle indels, we extended the space-saving data structure by including as keywords also some (large) subsequences of τ. For efficiency purposes, we do not store these subsequences directly but use rolling hashing. Two critical parameters for the space saving to be accurate are: the number of reads to process and the number k of sequences to monitor. We do not set a fixed value for the number of reads, but we stop processing when the empirical probability of changing the top keywords becomes lower than 5% (experimentally, this happens after inspecting 3 to 10 thousand reads). As for k, instead, we observed that it must be greater than $|\Sigma|^{|\beta|}$ where Σ is the alphabet and $|\beta|$ is the average length of UMI sequences. By setting $k = 1000$ we can handle UMIs up to 5 nucleotides long.

After collecting the frequency statistics, we derive, by means of a global multiple alignment of the k most common sequences, a consensus string to be used as a *candidate adapter*. To limit the computational cost, instead of aligning all the possible $k^2/2$ pairs of sequences, we use the most common one as a reference and align the others with it. Then, we derive the candidate adapter by majority vote. If a given position does not have a clearly dominant character we mark it with N. A stretch of Ns at the beginning of the candidate adapter is interpreted as UMI and is trimmed.

In our experiments, we observed that some reads do not contain the complete adapter, indeed it can be truncated in the sequencing process. To avoid discarding such reads from the analysis, we shorten the candidate adapter as much as possible. More precisely, our *working adapter* will be the shortest prefix of the candidate adapter which is at edit distance greater than d from all sequences in X. By default it is $d = 2$ but *EZcount* enables the user to set this value from the command line. The rationale for this strategy is that we want our working adapter to be as short as possible, but still sufficiently different from any miRNA sequence.

2.3. Search of miRNA sequences in reads

After having computed the working adapter a, we begin the actual process of trying to assign each read to a single miRNA sequence in X.

For each read r in the input file we first search r for a substring q within edit distance d, the same parameter as above, from our working adapter a. This search is done using the Shift-Or algorithm [2] which is extremely fast when the pattern is shorter than the length of a computer word and the number of errors is relatively small, which is precisely our setting.

Unless otherwise specified in the command line, if the search fails, i.e. the adapter a is not found within the given edit distance, the read is discarded. If a string q within edit distance d from a is found, we partition r as $r = \sigma q \tau$, discard $q\tau$ and proceed searching the miRNA sequences in σ as follows (note that with the notation introduced at the beginning of this section it is $\sigma = \alpha s_r \beta$). Initially, we use the Keyword Tree $T(X')$ to find if a substring of σ exactly matches a sequence s in X'. If this is the case, s will be our candidate sequence.

If the exact search using the Keyword Tree fails then, for each sequence $s \in X'$, we search σ for a substring with Hamming distance at most m from s. The search is done, again, using the Shift-Or algorithm and we use Hamming (instead of Edit) distance to reduce the computational cost since the search is done for each $s \in X'$. By default we use $m = 2$, but this value can be modified via command line.

At the end of this exhaustive search we are left with possibly more than one sequence with a minimal number of mismatches with respect to the trimmed read σ. Let $S = \{s_1, \dots, s_k\} \subset X'$ be the set of such sequences. The set S, however, does not necessarily contain all the candidate miRNAs at minimal distance. In fact, by definition of X', there may exist one or more sequences $t \in X/X'$ such that: s_i is subsequence of t and both s_i and t have the same number of mismatches with respect to σ. If this is the case, we extend S including t.

To assign the read r to the correct miRNA, we need to decide which element of S is best suited. Since all elements in S have the same number of mismatches with respect to σ, we use their relative similarities, measured as the number of matches, for comparison. Since a longer sequence corresponds to a higher similarity, we can narrow S to contain only the longest elements. In most cases, this filtering causes S to contain only one miRNA which, being the most similar, is matched to the read r. If it is not possible to unambiguously assign the read on the basis of the sequence alone, we break ties by leveraging on the per-base quality score. Assuming that mismatches are the effect of sequencing errors, their probability must be inversely proportional to quality. Consequently, we can apply a further filtering to S retaining only the sequences with lowest overall quality on their mismatched positions in r. Again, if only one element remains in S the read can be assigned to it, otherwise the ambiguity cannot be broken, the read is marked as ambiguous and we proceed with the next read.

3. Results

We experimentally tested *EZcount* in order to assess its ability to perform accurate read counting and to automatically identify and trim the 3' adapter independently of any a-priori knowledge about the library preparation kit and sequencing protocols. In addition, we performed a comparison with state of the art counting pipelines and integrated suites in order to show the benefits of our approach. Given the wide variety of tools and options for each step of the quantification process, a exhaustive comparison would be rather difficult. Thus, we leveraged on comparative studies in the literature as well as on usage popularity as selection criteria. In particular, for aligner based pipelines we choose *cutadapt* [16] for trimming, and *featureCounts* [15] for read counting as they are two of the most commonly used tools for these tasks. For the choice of the aligners we followed the guidelines provided by [32]. In that work, the authors showed that *BWA*, *Bowtie 1* and *Bowtie 2* rank as the most suitable aligners for small RNAs and suggested specific command line options for their use. Specifically, they recommended *Bowtie 2* to be run setting the `--very-sensitive-local` option, while *Bowtie 1* was recommended to be run using the `--best --strata` options. We chose not to run *Bowtie 1* standalone but we replaced it with *miRDeep2* [7] and *sRNAbench* [1] which are more popular and make internal use of *Bowtie 1* with the `--best --strata` options enabled. In the case of *sRNAbench* we did not have to optimize library specific parameters but we could use those recommended by the authors in the sRNAtoolbox website[2]. Although not rated in [32], we included in the comparison the *Subread* [14] aligner since it is part of the same software package of *featureCounts*. Finally, we set *BWA* parameters according to the recommendations in [28].

As for integrated suites, despite the vast literature on the subject, only a limited number of options are available in practice. Some pipelines are no longer maintained or use deprecated libraries. An example of unmaintained software

[2] https://arn.ugr.es/srnatoolbox/srnabench/

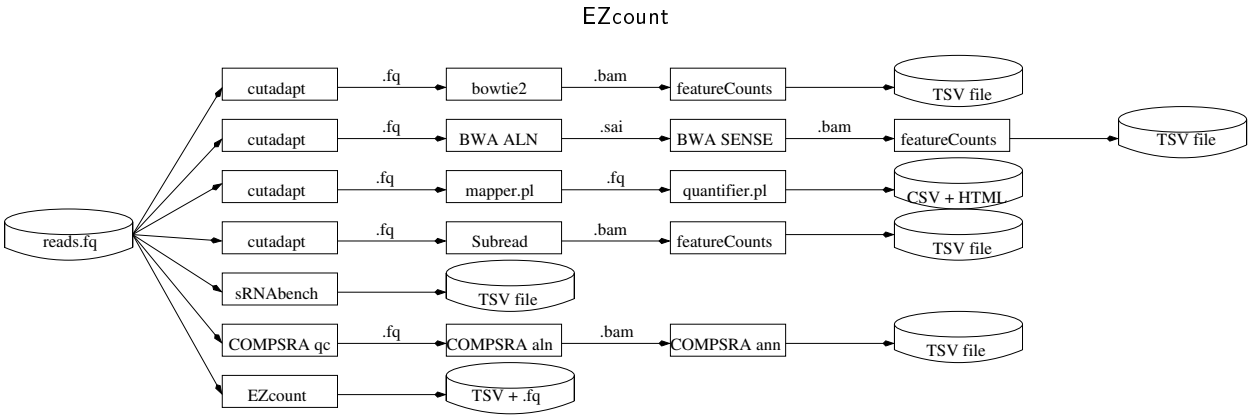

Figure 1: Graphical representation of the pipelines of the tools considered in this paper. For simplicity in the following the tools will be denoted as (from top to bottom): *bowtie2*, *BWA mirdeep2*, *Subread*, *sRNAbench*, *COMPSRA* and *EZcount*. Additionally, we will use *EZcount** to denote *EZcount* with options `-M 3 -S 2 -R`.

is *CAP-miRSeq* [27] which is packaged in a old Ubuntu virtual machine with a very old version of *cutadapt* that does not support the `-u` option required to handle the Clontech SMARTer smRNA-Seq library preparation kits. Examples of software based on deprecated libraries are: *miRge* [3] and *UEA small RNA Workbench* [20]. *miRge* depends on very specific versions of libraries while *UEA small RNA Workbench* depends on an outdated version of the Java runtime environment. At the end of our investigation we identified only three integrated suites: *sRNAbench*, *mirdeep2* and *COMPSRA* [12]. Figure 1 graphically depicts the compared pipelines, while in the SUPPLEMENTARY MATERIAL we provide the exact scripts we used. We tested two versions of the *EZcount* tool: one with the default settings and another, henceforth referred to as *EZcount**, with options `-M 3 -S 2 -R` whose main difference is to allow three mismatches, instead of two as in *EZcount*, between the read and the miRNA. Notice that, due to the fact that the Clonetech adapter is a subsequence of the miRNA *hsa-miR-3613-3p*, in 5 cases *EZcount* failed to detect it, and thus we had to provide it via command line (see SUPPLEMENTARY MATERIAL for details).

With regard to data, we downloaded the following datasets from NCBI (see SUPPLEMENTARY RESULTS for a detailed description) representative of the most common library preparation kits:

- a collection of 61 runs described in [30], from NCBI trace (study accession number SRP199350[3]), consisting of the same human brain tissue, in various concentrations, prepared with 4 different protocols: 19 samples using the Clontech SMARTer smRNA-Seq Kit for Illumina (Clontech), 19 samples using the Bioo Scientific NEXTflex Illumina Small RNA Sequencing Kit v3 (NEXTflex), 10 samples using the Illumina TruSeq Small RNA Library Prep Kit (TruSeq), and 13 samples using the New England BioLabs Next Multiplex small RNA kit (NEB). All the samples have been sequenced with an Illumina HiSeq-3000;
- a collection of 9 samples from NCBI GEO datasets (accession number GSE123627[4]), described in [13], belonging to two common human cell lines: HEK293T and HeLa, processed using *AQ-seq* (6 samples) and *TruSeq* (3 samples) as library preparation protocols and sequenced with an Illumina HiSeq-2500 (7 samples) and Illumina MiSeq (2 samples).

We also created a synthetic dataset using a procedure detailed in Section 3.2. The dataset is available in our website[5].

3.1. Sensitivity quantification

In miRNA counting pipelines, a non negligible fraction of reads is usually not assigned to any miRNA sequence. This is sometimes due to reads containing sequences other than miRNAs, but more often the software fails to assign the reads due to sequencing errors. Indeed, some of these errors cause two or more miRNAs to approximately match the read making the assignment not straightforward: an important feature of our tool is to use quality scores to assign even reads with errors to the correct miRNA.

[3] https://trace.ncbi.nlm.nih.gov/Traces/sra/sra.cgi?study=SRP199350
[4] https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE123627
[5] http://bioinformatics.iit.cnr.it/ezcount/synthetic.tar.bz2

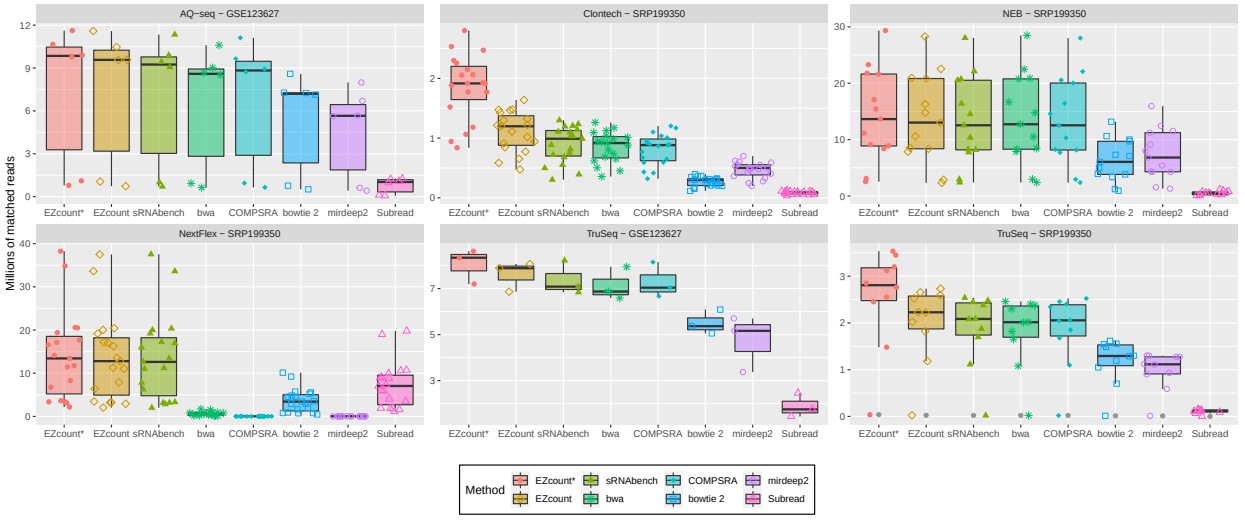

Figure 2: Comparison of counting pipelines on different sequencing technologies

Although it is conceivable that unassigned reads do not affect miRNA frequency ranking, as they are proportional to the expression level (the higher the expression the higher the probability of missing reads), these reads can bias differential analysis causing the exclusion of significant miRNAs from subsequent studies. Moreover, as demonstrated experimentally in [17], assigning more reads improves sensitivity of differential expression algorithms allowing even under-expressed miRNAs to be analysed.

For the above reasons, in this section we report the outcome of two experiments aimed at 1) quantifying the overall number of matched reads and 2) measuring sensitivity to lowly expressed miRNAs for the different pipelines.

Figure 2 reports, for each library preparation protocol tested, the distribution of the number of matched reads for all pipelines (see SUPPLEMENTARY RESULTS for details). Note that *featureCounts* was set to assign ambiguous reads to (only) one miRNA; *EZcount*, instead, is more conservative in that it discards reads that it was unable to disambiguate and never assigns a read to more than one sequence.

Figure 2 shows that in general *Bowtie 2*, *mirdeep2* and *Subread* are less effective than the others tools. *BWA* and *COMPSRA* worked well in most of the cases but they showed a dramatic drop in performance dealing with NextFlex library. We cannot fully explain the reasons for this drop but we can exclude that it depends on the read preprocessing. In fact, *Subread* takes as input the same trimmed reads as *BWA* and returns higher counts for this dataset. From Figure 2, we see that *EZcount* outperforms *sRNAbench* which in turns is superior to the other tools. We also see that for certain inputs *EZcount** performs even better. The Clontech data, for example, has a non negligible fraction of isoMIRs with the last two nucleotides trimmed. In this case, the additional mismatch allowed by *EZcount** nearly doubled the number of assigned reads. Similarly, on TruSeq samples *EZcount** was able to match approximately one million more reads per sample.

To evaluate the sensitivity of the counting pipelines detecting low expression, Table 1 reports for each method/library the number of miRNAs with at least 10 supporting reads. In general, to the higher volume of reads that *EZcount* and *sRNAbench* are able to match, corresponds an increase in the number of these miRNAs. In this case, however, the advantage of *EZcount** over the other methods is more evident. *BWA* and *COMPSRA* are marginally less sensitive, while *Bowtie 2*, *mirdeep2* and *Subreads* are generally less capable of quantifying low signals.

3.2. Accuracy assessment

Assessing the correctness of the assignments of the reads to individual miRNAs would require the availability of annotated reads that are impossible to produce with current sequencing protocols. As discussed in [17], comparing with qPCR quantification is also problematic. In fact, due to the logarithmic scale of the cycles to threshold measure, qPCR is much less sensitive than NGS to small variations of the expression level. In the absence of a gold standard, we used synthetically generated samples for which the read/miRNA association was known. By tracking the correspondence

Table 1
Average number of miRNAs with at least 10 supporting reads.

	GSE123627		SRP199350			
Algorithm	AQ-seq	TruSeq	Clontech	NEB	NEXTflex	TruSeq
EZcount*	1039	816	890	1041	1214	628
EZcount	881	661	542	780	906	470
BWA	767	555	485	739	315	430
Bowtie 2	461	715	318	522	506	339
mirdeep2	730	524	452	693	116	406
Subread	243	267	153	241	597	157
COMPSRA	696	508	436	680	76	413
sRNAbench	877	622	556	1250	849	455

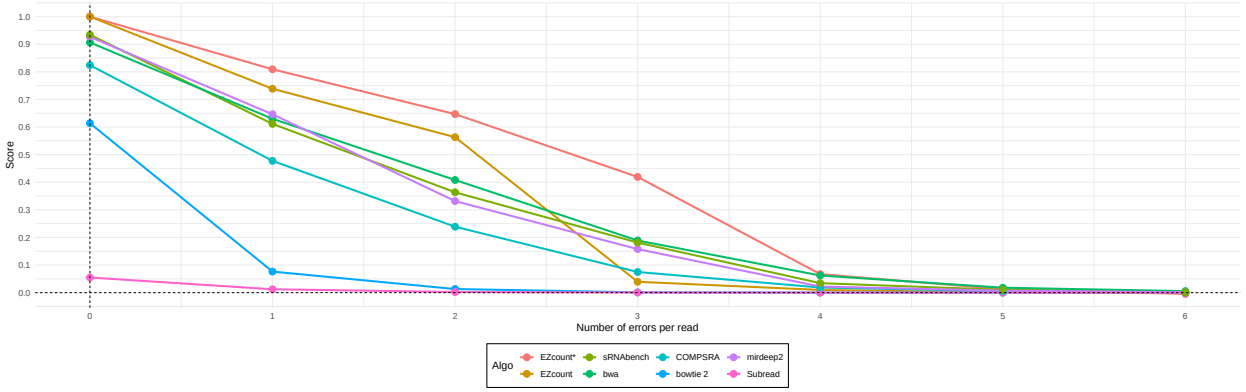

Figure 3: Assessment of the Accuracy of the counting pipelines on a synthetic dataset

of the aligned reads with the corresponding miRNA counts we were able to evaluate the accuracy of the methods.

We started with the set of human miRNA sequences annotated in mirBase v22.1 and removed elements with different names but exactly the same sequence (e.g. *hsa-miR-527* and *hsa-miR-518a-5p*). Synthetic samples were produced creating 100 replicates of each sequence and applying a random perturbation to them. Let $H_d(x)$ and $E_d(x)$, be functions that generate a random string respectively at Hamming/edit distance d from x, and let $T_t(x)$ be the function that trims the last t characters from x. As a perturbation function we used

$$P_{h,e,t}(sequence) = H_h(E_e(T_t(sequence)))$$

with h, e, t pre-defined integer constants. By repeating the above procedure for every possible assignment of the parameters h, e, t in the range $[0, 2]$, we obtained a dataset of 27 synthetic samples (containing each 100 copies of the initial set of miRNAs perturbed with the same random perturbation function).

Given such synthetic samples, where the correct assignment of each read is known by construction, we measure the accuracy of the tools by considering the proportion of correctly assigned reads minus the proportion of wrongly assigned ones. This score spans the range $[1, -1]$ achieving value 1 when all the reads have correctly been assigned, value -1 when all the reads have been assigned to the wrong miRNA and value 0, for example, when no reads have been assigned. The rationale for this measure is that incorrectly assigned reads introduce errors in the overall counts and therefore they should be penalized more than unassigned reads.

Figure 3 shows a chart comparing the average score of the different tools on the synthetic dataset. The x axis reports the overall number of errors per read, that is, the sum of the constants h, e and t (hence $x = 1$ reports the average for sample obtained with the perturbation functions $P_{1,0,0}$, $P_{0,1,0}$, and $P_{0,0,1}$). With the exception of *Subread* and *Bowtie 2* which are not designed to align miRNAs, all tools work fairly well as long as the error rate is small. Interestingly, the use of edit distance does not yield any significant advantage to BWA compared with *sRNAbench* and *mirdeep2* which

use Hamming distance being based on Bowtie. *COMPSRA*, which is based on the STAR aligner [6], is slightly less accurate than the other aligners. This is not surprising considering that STAR was specifically designed for transcripts and its main feature is the handling of splicing points. Our tool shows a different profile than the other methods with consistently higher scores. Both *EZcount* and *EZcount** scores show a sharp drop only when the number of errors per reads exceeds the number of allowed mismatches (2 for *EZcount*, 3 for *EZcount**). We also see that for less than three errors *EZcount** provides only a marginal advantage over *EZcount*. Detailed results of each tool for all the 27 synthetic samples are available in the SUPPLEMENTARY RESULTS.

3.3. Running time

Although running time is a secondary feature for counting pipelines, to enable the processing of large datasets it is important that the entire process terminates in a reasonable amount of time.

To evaluate the computational efficiency of the tested pipelines, we report in Table 2 their running times expressed as thousands of processed nucleotides per second using a single CPU core (in our experiments we use an Intel(R) Xeon(R) CPU E5-4620 v2 @ 2.60GHz). The reported running times do not include the cost of preliminary operations, such as the construction of the indices or of the supporting data structures, which are performed only once. Notice that, while *EZcount* setup took about 20 seconds to process mirbase v 22.1 on the human genome, for other methods, in particular the integrated pipelines such as *COMPSRA* and *sRNAbench*, setting up the necessary environment and indexing data structures can be a complicated and slow process, sometimes requiring manual intervention.

Table 2
Average processing speeds in kbp/s.

	GSE123627		SRP199350			
Algorithm	AQ-seq	TruSeq	Clontech	NEB	NEXTflex	TruSeq
EZcount*	31.1	78.4	92.8	43.5	40.8	39.6
EZcount	99.7	387.0	411.6	158.4	136.0	185.9
BWA	13.6	9.5	141.8	67.7	10.1	98.8
Bowtie 2	916.8	921.9	1653.9	1028.2	908.8	945.1
mirdeep2	472.5	505.6	410.9	524.9	313.7	388.9
Subread	1059.7	1127.6	880.3	1053.1	835.6	877.4
COMPSRA	389.1	447.3	869.9	397.8	448.9	1904.7
sRNAbench	6138.7	8462.7	7388.1	8415.0	2232.6	4449.2

The results in Table 2 show that most methods run in time of the same order of magnitude. Notable exceptions are *BWA*, which sacrificies performance for greater accuracy, and *sRNAbench* which runs much faster than all the others. This performance comes from a clever construction of the index which includes only the regions surroundings the miRNAs. Our tool ranks between *BWA* and the other methods, with *EZcount* closer to the other methods and *EZcount** closer to *BWA*.

4. Discussion

miRNA profiles are studied under a variety of experimental conditions that require different protocols for preparing libraries and performing sequencing. The subsequent processing of raw data must take into account the different characteristics of these protocols to accurately quantify the expression profiles. This, in turns, forces professionals in bioinformatics to continuously tweak, when not change, processing pipelines.

We claim that automatically recognizing experiment-dependent parameters can reduce the need of these tweaks making processing pipelines easier to maintain. An example in this direction is the *fastp* tool [5] that, among its many features, has gained popularity for its ability to automatically detect the 3' adapter. At the same time, tools should be able to guarantee the same accuracy in different experimental scenarios minimizing the effort required for optimizing the counting process. *EZcount* was designed to fulfil both these requirements of user friendliness and effectiveness.

Our experiments showed that,with the exception of 5 CloneTech samples, *EZcount* was always able to derive the 3' adapter from input reads. Failures were due to the fact that the CloneTech protocol introduces a poly(A) sequence before the Illumina adapter, and such sequence, being at Hamming distance 1 from the miRNA *hsa-miR-3613-3p*,

makes automatic recognition of the adapter much more difficult. For such extreme cases, *EZcount* supports the use of a user-supplied adapter.

The accuracy and sensitivity measurements showed that *EZcount* is more independent than the other methods to library-specific characteristics of data. In fact, it achieved good results both with the GSE123627 dataset, where spurious insertions are unlikely, and with the SRP199350 dataset, where indels are more frequent. When allowed to admit 3 mismatches, *EZcount* was the only tool able to match trimmed reads of the Clontech samples doubling the counts of *sRNAbench* which was the second best software. *Bowtie 2*, *mirdeep2* and *Subread* showed in general to suffer the presence of indels, while *BWA* and *COMPSRA* failed counting NEXTflex samples.

On the user's perspective, the actual running time, and in turn the computational sustainability, depends not only on samples sizes, CPU parallelism, and disk speed, but especially on the cost of tuning the pipeline and preparing the supporting data structures. Making deployment straightforward was one of the drives for the development of *EZcount* which depends only on two standard libraries: zlib and pthreads. Moreover, the creation of the supporting data structures is done with a single command on standard files (the reference and the GFF of the miRNA db) and takes as few as 20 seconds. These features make *EZcount* much easier to use than all the others tools which either require to spend time creating indices or installing third-party software. From the computational point of view, being an all-in-one solution, *EZcount* has the advantage of reading the input file only once and not producing large intermediate BAM files. Moreover, in a multithreaded environment, CPU utilization is optimized by means of a simple consumer/producer model in which a central thread loads the input in blocks and distributes the reads to consumer threads; each consumer thread then performs independently the entire sequence of adapter trimming, miRNA matching and counting. As a result, *EZcount* took on average 2 minutes per sample using 64 threads and less than 10 minutes per sample narrowing to 16 threads.

5. Conclusion

MiRNAs are small endogenous molecules of RNA that are known to participate in several cellular functions by suppressing genes. Their short length (ranging from 18 to 26 nucleotides) and the similarity among some of their sequences make the expression quantification based on the alignment between reads and miRNAs a challenging algorithmic problem. At the same time, accurate quantification is crucial for subsequent analyses. For example, personalized medicine relies on the miRNA expression profiles for prognosis, diagnosis and prediction of drug response. Computing these profiles on extracellular miRNAs, which are released in small concentrations, requires not only highly accurate but also highly sensitive algorithms.

In this work, we presented *EZcount*, a new tool that deviates from the existing pipelines by inverting the read/miRNA matching process. Instead of aligning each read against the reference genome and then deriving the corresponding miRNAs, *EZcount* searches for miRNA sequences within the reads. This new strategy allows *EZcount* to be more sensitive than pipelines based on sequence alignment, without any loss in its accuracy. Moreover, *EZcount* performance is completely independent of the sequencing parameters. Tests on about 70 samples, 5 library preparation protocols, and 3 sequencing machines have proven the effectiveness of *EZcount*. In addition to being more effective, *EZcount* was designed to improve the user-experience. An easy to use single-step command line interface, and the ability to automatically identify the adapter sequence are two features of *EZcount* that greatly reduce the effort for deploying a counting pipeline making it more reusable.

Funding

G.M. was partially supported by PRIN grant 2017WR7SHH, by INdAM-GNCS Project 2020 *MFAIS-IoT*.
F.G. was partially supported by the *Algorithms for RNA-seq data analysis* (RNAlgo) Project from the IIT-CNR.

References

[1] Aparicio-Puerta, E., Lebrón, R., Rueda, A., Gómez-Martín, C., Giannoukakos, S., Jaspez, D., Medina, J.M., Zubkovic, A., Jurak, I., Fromm, B., et al., 2019. sRNAbench and sRNAtoolbox 2019: intuitive fast small RNA profiling and differential expression. Nucleic acids research 47, W530–W535.

[2] Baeza-Yates, R.A., Gonnet, G.H., 1992. A new approach to text searching. Commun. ACM 35, 74–82.

[3] Baras, A.S., Mitchell, C.J., Myers, J.R., Gupta, S., Weng, L.C., Ashton, J.M., Cornish, T.C., Pandey, A., Halushka, M.K., 2015. MiRge: A multiplexed method of processing small RNA-Seq data to determine microRNA entropy. PloS one 10, e0143066.

[4] Bartel, D.P., 2009. MicroRNAs: target recognition and regulatory functions. cell 136, 215–233.

[5] Chen, S., Zhou, Y., Chen, Y., Gu, J., 2018. fastp: an ultra-fast all-in-one FASTQ preprocessor. Bioinformatics 34, i884–i890.
[6] Dobin, A., Davis, C.A., Schlesinger, F., Drenkow, J., Zaleski, C., Jha, S., Batut, P., Chaisson, M., Gingeras, T.R., 2013. Star: ultrafast universal RNA-seq aligner. Bioinformatics 29, 15–21.
[7] Friedländer, M.R., Mackowiak, S.D., Li, N., Chen, W., Rajewsky, N., 2012. miRDeep2 accurately identifies known and hundreds of novel microRNA genes in seven animal clades. Nucleic acids research 40, 37–52.
[8] Griffiths-Jones, S., Grocock, R.J., Van Dongen, S., Bateman, A., Enright, A.J., 2006. miRBase: microRNA sequences, targets and gene nomenclature. Nucleic acids research 34, D140–D144.
[9] Gusfield, D., 1997. Algorithms on Strings, Trees, and Sequences - Computer Science and Computational Biology. Cambridge University Press.
[10] Ha, M., Kim, V.N., 2014. Regulation of microRNA biogenesis. Nature reviews Molecular cell biology 15, 509–524.
[11] Javahery, H., Seffah, A., Radhakrishnan, T., 2004. Beyond power: making bioinformatics tools user-centered. Communications of the ACM 47, 58–63.
[12] Jiang, L., Kho, A.T., Chase, R.P., Lorena, P., Leanna, F., Amr, S.S., et al., 2020. COMPSRA: a comprehensive platform for small RNA-Seq data analysis. Scientific Reports (Nature Publisher Group) 10.
[13] Kim, H., Kim, J., Kim, K., Chang, H., You, K., Kim, V.N., 2019. Bias-minimized quantification of microRNA reveals widespread alternative processing and 3' end modification. Nucleic acids research 47, 2630–2640.
[14] Liao, Y., Smyth, G.K., Shi, W., 2013. The subread aligner: fast, accurate and scalable read mapping by seed-and-vote. Nucleic acids research 41, e108–e108.
[15] Liao, Y., Smyth, G.K., Shi, W., 2014. featureCounts: an efficient general purpose program for assigning sequence reads to genomic features. Bioinformatics 30, 923–930.
[16] Martin, M., 2011. Cutadapt removes adapter sequences from high-throughput sequencing reads. EMBnet. journal 17, 10–12.
[17] Metpally, R.P.R., Nasser, S., Malenica, I., Courtright, A., Carlson, E., Ghaffari, L., Villa, S., Tembe, W., Keuren-Jensen, V., et al., 2013. Comparison of analysis tools for miRNA high throughput sequencing using nerve crush as a model. Frontiers in genetics 4, 20.
[18] Metwally, A., Agrawal, D., El Abbadi, A., 2005. Efficient computation of frequent and top-k elements in data streams, in: International Conference on Database Theory, Springer. pp. 398–412.
[19] Mishra, P.J., Bertino, J.R., 2009. MicroRNA polymorphisms: the future of pharmacogenomics, molecular epidemiology and individualized medicine. Pharmacogenomics 10, 399–416.
[20] Mohorianu, I., Stocks, M.B., Applegate, C.S., Folkes, L., Moulton, V., 2017. The UEA small RNAworkbench: a suite of computational tools for small RNA analysis, in: MicroRNA Detection and Target Identification. Springer, pp. 193–224.
[21] Morin, R.D., OâĂŹConnor, M.D., Griffith, M., Kuchenbauer, F., Delaney, A., Prabhu, A.L., Zhao, Y., McDonald, H., Zeng, T., Hirst, M., et al., 2008. Application of massively parallel sequencing to microRNA profiling and discovery in human embryonic stem cells. Genome research 18, 610–621.
[22] Palmirotta, R., Lovero, D., Cafforio, P., Felici, C., Mannavola, F., Pellè, E., Quaresmini, D., Tucci, M., Silvestris, F., 2018. Liquid biopsy of cancer: a multimodal diagnostic tool in clinical oncology. Therapeutic advances in medical oncology 10, 1758835918794630.
[23] Pant, N., Rakshit, S., Paul, S., Saha, I., 2019. Genome-wide analysis of multi-view data of miRNA-seq to identify miRNA biomarkers for stomach cancer. Journal of Biomedical Informatics 97.
[24] Schirmer, M., DâĂŹAmore, R., Ijaz, U.Z., Hall, N., Quince, C., 2016. Illumina error profiles: resolving fine-scale variation in metagenomic sequencing data. BMC bioinformatics 17, 125.
[25] Seemann, T., 2013. Ten recommendations for creating usable bioinformatics command line software. GigaScience 2, 2047–217X.
[26] Smith, T., Heger, A., Sudbery, I., 2017. UMI-tools: modeling sequencing errors in unique molecular identifiers to improve quantification accuracy. Genome research 27, 491–499.
[27] Sun, Z., Evans, J., Bhagwate, A., Middha, S., Bockol, M., Yan, H., Kocher, J.P., 2014. CAP-miRSeq: a comprehensive analysis pipeline for microRNA sequencing data. BMC genomics 15, 1–10.
[28] Tam, S., Tsao, M.S., McPherson, J.D., 2015. Optimization of miRNA-seq data preprocessing. Briefings in bioinformatics 16, 950–963.
[29] Westholm, J.O., Ladewig, E., Okamura, K., Robine, N., Lai, E.C., 2012. Common and distinct patterns of terminal modifications to mirtrons and canonical microRNAs. Rna 18, 177–192.
[30] Wright, C., Rajpurohit, A., Burke, E.E., Williams, C., Collado-Torres, L., Kimos, M., Brandon, N.J., Cross, A.J., Jaffe, A.E., Weinberger, D.R., et al., 2019. Comprehensive assessment of multiple biases in small RNA sequencing reveals significant differences in the performance of widely used methods. BMC genomics 20, 513.
[31] Wu, L., Qu, X., 2015. Cancer biomarker detection: recent achievements and challenges. Chemical Society Reviews 44, 2963–2997.
[32] Ziemann, M., Kaspi, A., El-Osta, A., 2016. Evaluation of microRNA alignment techniques. Rna 22, 1120–1138.