Check for updates

SOFTWARE TOOL ARTICLE

MOAflow: how re-design a pipeline with Nextflow streamlines data analysis

[version 1; peer review: 2 approved with reservations, 1 not approved]

Jacopo Tartaglia[1], Mario Giorgioni[2], Luigi Cattivelli[1], Primetta Faccioli[1]

[1]CREA Research Centre for Genomics and Bioinformatics, Fiorenzuola d'Arda, Italy
[2]CREA Research Center for Forestry and Wood, Rome, Italy

v1 **First published:** 02 May 2026, **6**:143
https://doi.org/10.12688/openreseurope.23530.1
Latest published: 02 May 2026, **6**:143
https://doi.org/10.12688/openreseurope.23530.1

Abstract

Background

Advances in high-throughput DNA sequencing technologies have dramatically reduced the time and cost required to generate genomic data. As sequencing is no longer a limiting factor, increasing attention must be paid to optimizing the analyses of the large-scale datasets produced. Efficient processing of such data is essential to reduce computational time and operational costs. In this context, workflow management systems (WMSs) have become key instruments for orchestrating complex bioinformatic pipelines. Among these systems, Nextflow has emerged as one of the most widely adopted solutions in bioinformatics.

Methods

To improve scalability and computational efficiency, we employed Nextflow to re-design an already existing pipeline dedicated to the analysis of MNase-defined cistrome-Occupancy (MOA-seq) data. The re-engineering process focused on modularizing the workflow and integrating containerization technologies to ensure reproducibility and easier deployment across heterogeneous computing environments.

Results

The resulting workflow, named MOAflow, represents a modernized and fully containerized pipeline for MOA-seq data analysis. With only Docker and Nextflow required, the pipeline guarantees high portability and reproducibility. The data of the original article was

Open Peer Review

Approval Status ? ✗ ?

	1	2	3
version 1 02 May 2026	? view	✗ view	? view

1. **Joon-Yong An**, Department of Integrated Biomedical and Life Science, Korea University, Seoul, South Korea
2. **Johannes Loubser**, Stellenbosch University, Stellenbosch, South Africa
3. **Garima Mathur**, Sagar Institute of Science and Technology, Bhopal, India

Any reports and responses or comments on the article can be found at the end of the article.

used to benchmark the new pipeline. Its outputs closely match those of the original study with minor variations.

Conclusions

MOAflow demonstrates how the adoption of robust WMS can substantially enhance the performance and usability of pre-existing bioinformatic pipelines. By leveraging containerization and Nextflow, it ensures consistent results across platforms while minimizing setup complexity. This work highlights the value of modern WMS-driven approaches in meeting the computational demands.

Keywords
Nextflow, MOA-seq, high-performance computing, cloud computing, big data, workflow management systems



This article is included in the Horizon Europe gateway.

Corresponding authors: Jacopo Tartaglia (jacopo.tartaglia@crea.gov.it), Primetta Faccioli (primetta.faccioli@crea.gov.it)

Author roles: Tartaglia J: Conceptualization, Data Curation, Formal Analysis, Methodology, Validation, Writing – Original Draft Preparation, Writing – Review & Editing; **Giorgioni M**: Resources, Software, Writing – Review & Editing; **Cattivelli L**: Writing – Review & Editing; **Faccioli P**: Conceptualization, Project Administration, Supervision, Writing – Original Draft Preparation, Writing – Review & Editing

Competing interests: No competing interests were disclosed.

Grant information: This project has received funding from the Horizon Europe Framework Programme under grant agreement No [101081770]
The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Copyright: © 2026 Tartaglia J *et al*. This is an open access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

How to cite this article: Tartaglia J, Giorgioni M, Cattivelli L and Faccioli P. **MOAflow: how re-design a pipeline with Nextflow streamlines data analysis [version 1; peer review: 2 approved with reservations, 1 not approved]** Open Research Europe 2026, **6**:143 https://doi.org/10.12688/openreseurope.23530.1

First published: 02 May 2026, **6**:143 https://doi.org/10.12688/openreseurope.23530.1

Introduction

The progressive reduction in costs and the easier access to high-throughput sequencing technologies have led to massive data production in biology and in its omics disciplines. Accordingly, nowadays the bottleneck lies in the bioinformatic analysis of data and no longer in their production (Berger *et al.*, 2023). This situation has highlighted the need to increase the processing and storage capacity dedicated to data handling. However, meeting this challenge depends not only on implementing additional IT resources, but also on improving the efficiency with which they are used (Berger *et al.*, 2023). In this framework workflow management systems play a central role in streamlining data analysis. Nextflow (Di Tommaso *et al.*, 2017), which is a scientific workflow system based on the new DSL2 syntax, simplifies the writing of data-intensive computational pipelines. Nextflow has many bioinformatics applications, because it offers several advantages, including mature support for container technologies (e.g. Docker, Singularity …), compatibility with multiple source-code management systems, and a reactive programming model that enables highly efficient parallel execution and scalable workflows. Each process is organized as an independent module, promoting clarity and code reusability, while also allowing for easy implementation. The various processes are linked together in a workflow script, which can also be divided into sub-workflows, giving the code a transparent design that makes Nextflow pipelines easy to read, modify and extend.

Here we describe MOAflow, that is a bioinformatic pipeline for the analysis of MNase-defined cistrome-Occupancy (MOA-seq) data, developed on the previous work of Liang *et al.*, 2022a. The MOA-seq methodology was developed as a high-resolution, high-throughput, genome-wide method specifically designed to accurately identify putative Transcription Factor (TF) binding sites, called MOA-footprints (MFs), and to reveal Accessible Chromatin Regions (ACRs) in plants genomes (Savadel *et al.*, 2021). MOA-seq can generate an exceptionally high volume of data (Engelhorn *et al.*, 2025), especially when this technique is applied to species with very large genomes, making downstream analysis computationally demanding and complex particularly if based on separate scripts and standalone software. To overcome these constraints, the original pipeline has been re-designed using Nextflow, resulting in a modular, automated, and portable solution suitable for large-scale analyses. This new design ensures efficient resource allocation, faster execution, and reproducibility, while supporting collaborative use and workflow sharing. MOAflow was tested and benchmarked with the dataset of Liang *et al.*, 2022a, both in a local server and in a cloud-based environment (Microsoft Azure).

Materials and methods

Workflow overview

MOAflow as the original pipeline of Liang *et al.* 2022a is built upon a wide range of state-of-the-art software and data resources. The pipeline consists of three main phases: data pre-processing, reads alignment, and peak calling (see Figure 1). During the pre-processing step, quality control (QC) is performed on the raw PE reads (Paired Ends library is mandatory as input). FastQC (Andrews *et al.*, 2010) is used to check quality before and after the trimming process, which is performed by SeqPurge (Sturm 2025). Subsequently, overlapping PE reads are merged into single reads (SR) using FLASH (Magoč *et al.*, 2011) in order to improve quality and increase the fragment length. The alignment process begins with the generation of the genome index (if necessary) using STAR (Dobin *et al.*, 2013). Annotation file (gtf format mandatory) can be provided. Single End reads (SE) are aligned to the genome using STAR software. Subsequently, SAMtools (Danecek *et al.*, 2021) is used to filter the generated bam files, retaining reads that have a quality score (MAPQ) of 255 and are shorter than 80 bp. The effective genome length is calculated with the script unique-kmers.py from Khmers (Crusoe *et al.*, 2015), using either a fixed k-mer size or the average mapped fragment length (AMFL) calculated with SAMtools stat. bamCoverage function of deepTools (Ramírez *et al.*, 2016) is used to generate normalized bedgraph files (reads per genomic coverage, RPGC) of the complete read data. Finally, the MOA footprints (MFs) discovery process involves reducing reads to 20 bp (optional) and peak calling. To generate high-resolution maps, each read can be reduced to 20 bases centered on the midpoint. The shortening is performed using a simple awk command: in the case of reads with an odd number of bases, the central base is taken, and the sequence is extended by 10 bases on each side; in the case of reads with an even number of bases, one of the two central bases is chosen at random, and the sequence is extended by 10 bases on each side. Subsequently, the reads, shortened or kept at their original length, are used by MACS3 (Zhang *et al.*, 2008) for peak calling (callpeak sub-command). It should be noted that the analysis can be performed either by keeping the biological replicates separate (default case) or by merging the replicates to obtain a deeper coverage. However, if the replicates are merged, samples from different experimental conditions (e.g., treatment and control) cannot be analyzed at the same time.

Implementation in nextflow

The MOAflow pipeline is implemented in Nextflow, making use of the new DSL2 syntax. Nextflow is an extension of Groovy, which in turn is a superset of Java and simplifies the writing of data-intensive computational pipelines. This pipeline is composed by 13 different modules, each one representing a step in the analysis. Modules are linked together in the workflow script, called main.nf. The nextflow.config file defines the pipeline resources and default parameters, while

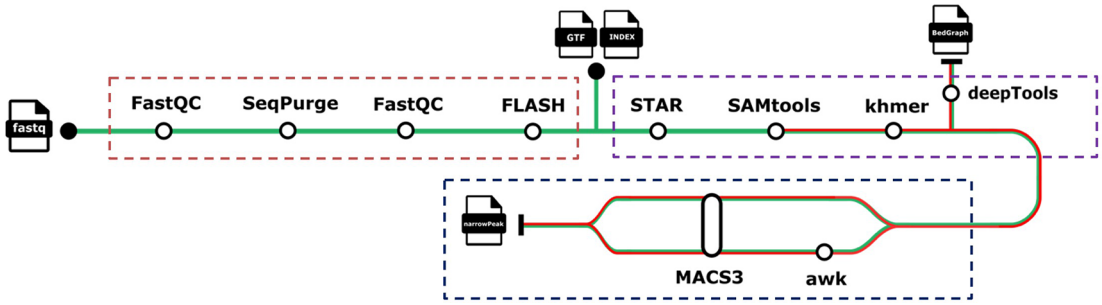

Figure 1. MOAflow workflow. The software used in the analysis are displayed in the metromap. The dotted boxes refer to the stages of analysis: pre-processing (orange), alignment (purple), and peak calling (blue). The green-red line starting from SAMtools means that replicates can be merged or not.

the params.json file is used to specify run-specific settings. Input samples are provided to the pipeline through a CSV file, in which the user defines each sample name along with the corresponding raw read files. To date, the software used in the MOAflow pipeline are run and containerized exclusively with Docker. The description of parameters and output files generated by MOAflow is given at the documentation page in the code repository.

Computing environments

MOAflow pipeline was tested on both local machine and in cloud environment (Microsoft Azure). The local system was a Windows Server 2019 Standard (Fujitsu Primergy RX900 S2) equipped with eight Intel Xeon CPU (E7–8860 2.27GHz CPUs, 80 cores, 160 threads), 224 GB of RAM, and a storage system consisting of a local unit of 272 GB and an external high-capacity storage unit of 21.26 TB. The run was performed in an Ubuntu v24.04.3 LTS virtual machine (made by Oracle VM VirtualBox v7.0) with 64 CPUs, 83.5GB of RAM and 2 TB of storage (vHD). The cloud environment was composed by a three-node cluster equipped with Intel Xeon Platinum processors, each node providing 96 vCPUs and 384 GB of RAM and using a 2 TB premium file share mounted via NFS on the Azure VMs to store raw data, working directories, and output files. The Open OnDemand v4.0.0 interface was used to simplify the use of Nextflow in the cloud environment.

MOAflow benchmarking

Since the main goal of this work was to redesign the pipeline of Liang *et al.* (2022a) in Nextflow while keeping its original structure unchanged, performance evaluation was conducted using various benchmark metrics and comparison strategies. The most direct analyses involved comparing the results reported in the original work with those obtained using MOAflow. The simplest comparisons concerned the number of input reads used for alignment, the number of MFs identified per sample, and the median length of full-length or 20 bp shortened MOA peaks. For a more extensive analysis based on the genomic regions shared between the MOAflow results and those of the original study, two distinct metrics were calculated: the Jaccard index and the F1-score. The Jaccard index was calculated using the jaccard function of Bedtools (v2.31.1, Quinlan *et al.*, 2010), using default parameters, on the different sets of genomic intervals to be compared. The calculation of Recall, Precision and F1-score ($F1 = 2 \times (Precision \times Recall) / (Precision + Recall)$) was performed using a combination of Bedtools intersect commands (with flags -f 0.5 -r -u) and awk commands (see f1_score_calc.sh script for more details).

Results

Pipeline execution

To test the pipeline performance in terms of biological results, MOAflow was run on the MOA-seq dataset from the work of Liang *et al.* (2022a), with the aim of reproduce their results. The input samples consisted of PE reads obtained from three biological replicates of B73 maize under control conditions and three other B73 biological replicates subjected to heat stress (4 h at 40 °C). The B73 AGPv4 assembly was employed as the maize reference genome for read mapping. The pipeline ran successfully and generated the expected output files. MOAflow was run with a single command, with fully automated installation and execution and an efficient task distribution across eleven different software tools to complete the analysis and provide quality control metrics, without any manual intervention required. The evaluation of pipeline performance was based on the following metrics: the number of inputs reads (with the percentage of uniquely mapped reads); the number of MOA peaks identified using full-length reads and reads trimmed to 20 bp (these called MOA footprints); and the median MOA peak length for both full-length and shortened reads. Table 1 summarizes the input read alignment results, benchmarked against the findings of Liang *et al.* (2022a). The results obtained with MOAflow, although not identical, closely correspond to those reported in the original study. The deviation in input reads per replicate

Table 1. Number of input merged reads per replicate (unique alignment rate).

Samples	Liang et al., 2022a	MOAflow
CONTROL1	126,939,893 (45%)	126,716,389 (46%)
CONTROL2	166,432,747 (44%)	166,101,382 (45%)
CONTROL3	138,693,816 (40%)	138,569,295 (45%)
STRESS1	158,734,914 (45%)	158,500,699 (49%)
STRESS2	131,979,910 (46%)	131,850,844 (50%)
STRESS3	160,081,276 (44%)	159,861,675 (47%)

Table 2. Number of MOA footprints (MFs, shortened peaks).

Samples	Liang et al., 2022a	MOAflow
CONTROL1	142,815	142,908
CONTROL2	184,536	184,539
CONTROL3	131,872	131,877
STRESS1	202,906	202,920
STRESS2	159,774	159,744
STRESS3	172,688	172,722

Table 3. F1-score statistics related to the MOA full length peaks per replicate. The MOA full length peaks of the original article were used as gold standard, while the MOAflow results as query.

Samples	Recall	Precision	F1-score
CONTROL1	0.93	0.94	0.94
CONTROL2	0.99	0.99	0.99
CONTROL3	0.97	0.96	0.97
STRESS1	0.97	0.96	0.96
STRESS2	0.97	0.96	0.97
STRESS3	0.99	0.99	0.99

ranged from 0.09% to a maximum of 0.2% fewer reads, and the unique alignment rates were also comparable. Table 2 shows the number of MFs per sample. The results were highly consistent with the original ones, with differences in peak counts ranging from 0.02% to 0.065% more MFs. Median MOA peak lengths were consistent: 180 bp for full-length peaks versus 179 bp reported in the article and 34 bp for MFs (shortened peaks), matching the original value.

Although these results are consistent from a merely numerical point of view, it was necessary to assess whether the MFs found by MOAflow overlapped the same genomic regions identified by the original work. Due to the lack of MFs coordinates files, we used the MOA peaks files (generated by full length reads) from the UCSC genome browser ("Availability of data and materials" section) from Liang *et al.* (2022a). The Jaccard index, which measures the ratio between the number of intersecting base pairs between two sets and the number of base pairs in the union of the two sets, as well as the precision, recall and F1-score statistics using the results of the original study as the "gold standard", were calculated as benchmarks. The Jaccard index, calculated with jaccard function of Bedtools (v 2.31.1), for each replicate in both conditions range from the lowest value of 0.92 (control 1) to the highest one of 0.99 (stress 3). Table 3 depicts the results of the F1-score statistic and related metrics. Subsequently, the results of the differential MOA footprints analysis between control and heat conditions (Additional file 2, Data S1: MOA-seq footprints of B73 detected under control or heat conditions) from Liang *et al.* (2022a) were used to assess the MFs resulted by MOAflow pipeline. The differential analysis was performed with the DiffBind (v2.12.0, Stark *et al.*, 2011) function in R on the results of the MOAflow pipeline. Again, the Jaccard index and F1-score statistics using the results of the original study as the "gold standard" were calculated as benchmarks. Table 4 summarizes differential MFs, and the distribution of those regions with increased

Table 4. Overall of differential MFs. Distribution of those regions in increased accessibility in heat stress and those with less accessibility to heat stress compared to controls.

MOA footprints	Liang et al., 2022a	This study
MFs in both conditions	13,792	12,613
MFs > Stress	11,140 (ca 80%)	11,362 (ca 81%)
MFs > Control	2,652 (ca 20%)	2,645 (ca 19%)

Table 5. F1-score statistics calculated on differential MFs.

MOA footprints	Recall	Precision	F1-score
MFs in both conditions	0.91	0.90	0.91
MFs > Stress	0.92	0.90	0.91
MFs > Control	0.91	0.91	0.91

accessibility in heat stress and those with less accessibility to heat stress compared to controls. The Jaccard index was calculated both on the overall set of differential MFs and separately for each experimental condition. For the total dataset, the Jaccard index was 0.89, while under heat stress and control conditions it was 0.88 and 0.90, respectively. The other metrics were calculated in the same manner and Table 5 summarizes the values. The high values of the Jaccard index as well as those of the F1-score indicate that the workflow performed as expected and it is consistent with the results of Liang *et al.* (2022a).

Computing environments comparison

The MOAflow pipeline was evaluated in two distinct computing environments: a local server and on a cloud-based platform (Microsoft Azure). The pipeline processed approximately 90 GB of input data, comprising raw sequencing reads and the maize reference genome, and generated roughly 229 GB of output data, including both intermediate and final files. Identical parameters, input datasets, and the execution command were used in both environments. On the local server, the run required 2 days and 4 hours, whereas the cloud ran successfully in 2 hours and 44 minutes. Total CPU usage amounted to 2,374.2 CPU-hours on the local machine, compared with 423.4 vCPU-hours in the cloud. In both cases, the number of tasks executed was 74.

Discussion

As shown in the previous section, the results obtained with MOAflow are highly consistent with, and in sometimes identical to, those reported in the original study. Minor discrepancies may essentially depend on stochastic initialization, meaning that results may not be identical. This is reflected in the Jaccard index and F1-score values when comparing full-length MOA peaks and MF differential analyses. The MOAflow pipeline was evaluated across heterogeneous computing environments, revealing significant differences in execution times. The comparison aimed to assess the robustness of the pipeline in different computing context and its ability to produce comparable, and in some cases identical, results, rather than to measure performance in terms of analysis speed. The results obtained confirm this behavior.

It should be emphasized that MOAflow was intentionally not implemented with integrated differential analysis (DiffBind) and motif discovery, because these analyses can be highly customizable and their underlying statistical approaches require careful evaluation of the outcomes; therefore, it is preferable to perform these steps manually. Nevertheless, the MOAflow pipeline is accompanied by the DiffBind script used to validate the results. This material is provided solely as a starting point for differential analysis; consequently, it is strongly recommend adapting them to everyone specific needs.

Declaration of AI use

Copilot (version number: 2.20260319.58.0) was used to improve the readability and language of the manuscript. The authors take full responsibility for the work.

Ethics and consent

Ethical approval and consent were not required.

This article derived from a pre-print version uploaded on bioRxiv pre-print server: https://doi.org/10.64898/2026.03.26.713914. (Tartaglia et al., 2026b).

Data and software availability

Source data

The data used to test the pipeline are available at https://www.ncbi.nlm.nih.gov/bioproject/?term=PRJNA849202 (Liang *et al.*, 2022b).

Software availability

Source code available at: https://github.com/Jacopot12/MOAflow.

Archived source code at time of publication: https://doi.org/10.5281/zenodo.18618374 (Tartaglia *et al.*, 2026a).

License: MIT License.

References

Andrews S: FastQC: **A Quality Control Tool for High Throughput Sequence Data**. 2010; **2**.

Berger B, Yu YW: **Navigating bottlenecks and trade-offs in genomic data analysis.** *Nat. Rev. Genet.* 2023; **24**(4): 235–250.
Publisher Full Text

Brown CT, *et al.*: **The khmer software package: enabling efficient nucleotide sequence analysis.** *F1000Res.* 2015; **4**: 900–900.

Crusoe, Michael R, *et al.*: **The khmer software package: enabling efficient nucleotide sequence analysis** *F1000Res.* 2015; **4**: 900.
Publisher Full Text

Danecek P, Bonfield JK, Liddle J, *et al.*: **Twelve years of SAMtools and BCFtools.** *Gigascience.* 2021; **10**(2): giab008.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

Di Tommaso P, *et al.*: **Nextflow enables reproducible computational workflows.** *Nat. Biotechnol.* 2017; **35**(4): 316–319.
Publisher Full Text

Dobin A, Davis CA, Schlesinger F, *et al.*: **STAR: ultrafast universal RNA-seq aligner.** *Bioinformatics.* 2013; **29**(1): 15–21.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

Engelhorn J, Snodgrass SJ, Kok A, *et al.*: **Genetic variation at transcription factor binding sites largely explains phenotypic heritability in maize.** *Nat. Genet.* 2025; **57**(9): 2313–2322.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

,
PubMed Abstract | **Publisher Full Text**

Liang Z, Myers ZA, Petrella D, *et al.*: **Mapping responsive genomic elements to heat stress in a maize diversity panel.** *Genome Biol.* 2022a; **23**(1): 234.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

Liang Z, *et al.*: **Mapping responsive genomic elements to heat stress in a maize diversity panel. The MOA-seq reads and associated RNA-seq reads, NCBI: BioProject: PRJNA849202.** 2022b. 2022. Accessed 29 Sept 2022.
Reference Source

Magoč T, Salzberg SL: **FLASH: fast length adjustment of short reads to improve genome assemblies.** *Bioinformatics.* 2011; **27**(21): 2957–2963.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

Quinlan AR, Hall IM: **BEDTools: a flexible suite of utilities for comparing genomic features.** *Bioinformatics.* 2010; **26**(6): 841–842.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

Ramírez F, *et al.*: **deepTools2: a next generation web server for deep-sequencing data analysis.** *Nucleic Acids Res.* 2016; **44**(Web Server issue): W160.

Savadel SD, Hartwig T, Turpin ZM, *et al.*: **The native cistrome and sequence motif families of the maize ear.** *PLoS Genet.* 2021; **17**(8): e1009689.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

Stark R, Brown G: **DiffBind: differential binding analysis of ChIP-Seq peak data.** *R package version 100.4.3.* 2011; 2–21.

Sturm M: **imgag/ngs-bits: 2025_03.** *Zenodo.* 2025.
Publisher Full Text

Tartaglia J, *et al.*: **MOAflow V1.0.** *Zenodo.* 2026a 30 marzo.
Publisher Full Text

Tartaglia J, *et al.*: **MOAflow: how re-design a pipeline with Nextflow streamlines data analysis.** *bioRxiv.* 2026b; 2026–03. (2026).

Zhang Y, Liu T, Meyer CA, *et al.*: **Model-based analysis of ChIP-Seq (MACS).** *Genome Biol.* 2008; **9**(9): R137.
Publisher Full Text

Open Peer Review

Current Peer Review Status: ? ✗ ?

Version 1

Reviewer Report 21 July 2026

https://doi.org/10.21956/openreseurope.25473.r76614

© **2026 Mathur G.** This is an open access peer review report distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

?

Garima Mathur
Sagar Institute of Science and Technology, Bhopal, Madhya Pradesh, India

The manuscript presents **MOAflow**, a Nextflow-based redesign of an existing MOA-seq analysis pipeline aimed at improving reproducibility, portability, and workflow automation. The topic is relevant and timely, and the authors successfully demonstrate that the redesigned pipeline produces results comparable to the original workflow while supporting execution on both local and cloud computing environments.
The manuscript is generally well written and technically sound; however, several important issues should be addressed before publication. The claims regarding improved computational efficiency, scalability, and storage optimization are not sufficiently supported by quantitative benchmarking. The authors should provide additional performance metrics, including CPU, memory, storage usage, and runtime comparisons with the original implementation. Reproducibility should be validated through repeated executions, and the discussion should include the limitations and trade-offs associated with the Nextflow implementation. Furthermore, testing the workflow on additional independent datasets and comparing it with other workflow management systems (e.g., Snakemake or CWL) would strengthen the manuscript.
Points that must be addressed to make the article scientifically sound:

1. Provide comprehensive computational benchmarking (runtime, CPU, memory, and storage).
2. Validate reproducibility through repeated executions and discuss output variability.
3. Moderate or support claims regarding computational efficiency and scalability.
4. Expand the discussion on workflow limitations and trade-offs.
5. Improve language and correct minor grammatical issues.

Overall Recommendation: Major Revision
The manuscript presents a useful software tool with practical value for the bioinformatics community. However, additional validation and more rigorous performance evaluation are required before the conclusions can be fully supported.

References
1. Mathur G, Pandey A, Goyal S. A review on blockchain for DNA sequence: security issues,

application in DNA classification, challenges and future trends. Multimed Tools Appl. 2023 May 25:1-23. doi: 10.1007/s11042-023-15857-1. Epub ahead of print. PMID: 37362738; PMCID: PMC10209554.
2. Mathur G, Pandey A, Goyal S. A comprehensive tool for rapid and accurate prediction of disease using DNA sequence classifier. J Ambient Intell Humaniz Comput. 2022 Jun 25:1-17. doi: 10.1007/s12652-022-04099-y. Epub ahead of print. PMID: 35789598; PMCID: PMC9243743.

Is the rationale for developing the new software tool clearly explained?
Partly

Is the description of the software tool technically sound?
Partly

Are sufficient details of the code, methods and analysis (if applicable) provided to allow replication of the software development and its use by others?
No

Is sufficient information provided to allow interpretation of the expected output datasets and any results generated using the tool?
Partly

Competing Interests: No competing interests were disclosed.

Reviewer Expertise: Bioinformatics, DNA, AI/ML, Security

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard, however I have significant reservations, as outlined above.

Reviewer Report 15 July 2026

https://doi.org/10.21956/openreseurope.25473.r74397

© 2026 Loubser J. This is an open access peer review report distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Johannes Loubser
Stellenbosch University, Stellenbosch, Western Cape, South Africa

Summary of the Article
This manuscript describes the redesign of an existing pipeline for the analysis of MNase-defined cistrome-Occupancy Analysis sequencing (MOA-seq) data. The authors successfully reimplemented the original workflow using Nextflow, a widely adopted workflow management system, and incorporated containerization to improve portability and reproducibility. The manuscript demonstrates execution of the pipeline on both local and cloud computing

environments and reports that the generated results are highly comparable to those obtained with the original implementation, with only minor discrepancies observed. The authors attribute these differences to stochastic initialization within machine learning components of the workflow, where random starting values are not fixed between analyses. Based on these findings, the authors conclude that the use of Nextflow and containerization improves the usability, portability, and reproducibility of the existing bioinformatics pipeline.

Overall, the work has practical value, as modernizing legacy bioinformatics workflows is important for improving reproducibility and long-term maintainability. However, several aspects of the manuscript require further clarification and discussion before the claims can be fully supported.

Major Comments
1. Claims regarding processing and storage capacity require further justification

One of the central motivations presented in the Introduction is the need to address limitations in processing and storage capacity associated with large-scale biological data analysis. However, I am not convinced that the presented reimplementation fully addresses these challenges, particularly with respect to storage requirements.

While Nextflow undoubtedly simplifies workflow orchestration, enhances reproducibility, and facilitates deployment across heterogeneous computing environments, it is also well known that Nextflow generates substantial intermediate data within its working directories. These intermediate files are critical for reproducibility, checkpointing, and workflow resumption, but they can significantly increase storage requirements during analysis.

Furthermore, despite improving workflow standardization, Nextflow pipelines generally require users to possess at least intermediate bioinformatics and computational skills for installation, configuration, execution, and troubleshooting. Therefore, while the reimplementation solves several important challenges, it may simultaneously introduce new limitations that deserve acknowledgement.

To strengthen the manuscript, the authors should:
- Quantify the storage requirements associated with the new pipeline.
- Report the size of the Nextflow work directory generated during the benchmarking analyses.
- Compare storage consumption between the original and redesigned implementations where possible.
- Discuss the implications of these storage requirements for resource-constrained research environments, particularly in low- and middle-income countries (LMICs).

I recognize that evaluating deployment in LMIC settings may not have been an explicit objective of this study. Nevertheless, because storage constraints are emphasized as a bottleneck in the Introduction, the manuscript should provide a more balanced discussion of both the advantages and disadvantages introduced by the Nextflow-based implementation.

This point should be addressed before publication, as the current framing may overstate the extent to which the redesign alleviates storage-related limitations.

2. Discussion of discrepancies between pipelines is insufficient

The explanation for the observed differences between the original and redesigned pipelines is currently limited to a brief reference to stochastic initialization. While this explanation may be valid, the manuscript requires a more detailed discussion of the mechanism and its implications.
In particular, the authors should clarify:

- Which specific component(s) of the workflow rely on stochastic processes.
- Whether random seeds can be fixed to improve reproducibility.
- The magnitude of the observed differences and whether they have any practical or biological significance.
- Whether repeated runs of the redesigned pipeline produce consistent results within an acceptable range of variation.

Providing this additional context would allow readers to better evaluate whether the discrepancies are expected, acceptable, and unlikely to affect downstream biological interpretation.
This point must be addressed to ensure the scientific soundness of the comparison between the original and redesigned workflows.

3. The Discussion should critically evaluate whether the redesign addresses the stated problem

The Discussion section currently emphasizes the advantages of containerization and workflow management but does not critically revisit the problem statement introduced earlier in the manuscript.
Specifically, the authors state:
“This situation has highlighted the need to increase the processing and storage capacity dedicated to data handling.”
The manuscript should explicitly discuss whether the redesigned pipeline actually contributes to solving this challenge.
From my perspective, the redesign clearly improves:

- Reproducibility;
- Portability;
- Workflow standardization;
- Scalability across computing environments.

However, the evidence presented does not demonstrate a reduction in storage requirements, and it remains unclear whether computational efficiency has been substantially improved relative to the original implementation.
Therefore, I recommend that the authors either:

- Provide evidence supporting claims related to processing and storage improvements, or
- Moderate the language used throughout the manuscript, particularly where storage-related benefits are implied.

A more balanced discussion would strengthen the manuscript and avoid overstating the impact of the redesign.

4. Additional benchmarking would strengthen the study

I would have liked to see a more direct comparison between the original implementation and the automated Nextflow workflow across both local and cloud computing environments.
Potential metrics could include:

- Runtime;
- CPU utilization;
- Memory consumption;
- Storage usage;
- Ease of deployment;
- Reproducibility metrics.

Such comparisons would provide a clearer assessment of the practical benefits and trade-offs associated with the redesign.
Although this may not be essential for publication, it would substantially increase the impact and usefulness of the study.

Minor Comments

- The manuscript would benefit from additional detail regarding the computational specifications of the local and cloud systems used for benchmarking.
- Consider including a short section outlining recommended hardware and storage requirements for prospective users.
- Clarify whether pipeline outputs are fully deterministic when fixed random seeds are supplied.
- A schematic comparing the original workflow architecture with the redesigned Nextflow implementation could help readers better understand the improvements introduced.

Overall Recommendation
The manuscript describes a valuable and timely effort to modernize a MOA-seq analysis workflow using Nextflow and containerization. The work has clear strengths in terms of reproducibility, portability, and workflow standardization. However, several of the broader claims regarding processing and storage limitations are not yet adequately supported by the presented results.

Points that must be addressed to ensure scientific soundness

1. Expand and strengthen the discussion of discrepancies between workflow outputs, including a more detailed explanation of stochastic initialization and its impact on reproducibility.
2. Quantify storage requirements generated by the Nextflow implementation, including reporting the size of the working directory used during benchmarking.
3. Reassess and potentially moderate claims that the redesign addresses storage-capacity limitations unless supporting evidence is provided.
4. Critically discuss the trade-offs introduced by the Nextflow implementation, including increased intermediate file generation and associated storage demands.

Addressing these points would substantially improve the manuscript and provide readers with a more balanced and rigorous evaluation of the redesigned pipeline.

References

1. Tartaglia J, Giorgioni M, Cattivelli L, Faccioli P: MOAflow: how re-design a pipeline with Nextflow streamlines data analysis. *bioRxiv*. 2026. Publisher Full Text

Is the rationale for developing the new software tool clearly explained?

Yes

Is the description of the software tool technically sound?
Yes

Are sufficient details of the code, methods and analysis (if applicable) provided to allow replication of the software development and its use by others?
Yes

Is sufficient information provided to allow interpretation of the expected output datasets and any results generated using the tool?
Yes

Competing Interests: No competing interests were disclosed.

Reviewer Expertise: Bioinformatics, Genomics, Expertise in Nextflow

I confirm that I have read this submission and believe that I have an appropriate level of expertise to state that I do not consider it to be of an acceptable scientific standard, for reasons outlined above.

Reviewer Report 13 July 2026

https://doi.org/10.21956/openreseurope.25473.r76610

© 2026 An J. This is an open access peer review report distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

?

Joon-Yong An
Department of Integrated Biomedical and Life Science, Korea University, Seoul, South Korea

I evaluated MOAflow as a potential end user attempting to benchmark the workflow on external data. The test was intentionally simple: use a standard Nextflow/Docker setup, use a public mammalian reference, use a public prebuilt STAR index, and run two small external mouse MNase-seq samples that were not part of the pipeline authors' original analysis. The samples were ERR2269362 and ERR2269363. The reference was mm10, with a public FASTA and public STAR index. This is the kind of reuse scenario I would expect many users to try before committing to a larger analysis. The core analysis path eventually ran, including trimming, read merging, STAR alignment, BAM filtering, read shortening, and MACS3 peak calling. The successful run produced mapped BAMs, filtered max80/MAPQ255 BAMs, shortened BED/BAM files, and MACS3 outputs. The run was informative because the main barriers were not the biological characteristics of the input data but the software and workflow assumptions around reference compatibility, auxiliary steps, parameter handling, and code-level validation. My comments below focus on those practical issues rather than on a conventional manuscript-level assessment.

Major comments

1. I could not run the stock workflow cleanly with a public prebuilt mm10 STAR index, even though this is a normal and reasonable user scenario. The public mm10 STAR index used in the benchmark was generated with STAR 2.7.1a, while MOAflow hard-coded STAR 2.7.9a in the STAR alignment module. STAR indices are version-sensitive, so the workflow stopped with a genome-version incompatibility error before the main analysis could proceed. This is a serious practical problem because many users working with standard genomes such as mm10 or hg38 will reasonably prefer a public prebuilt index over rebuilding a large STAR index from scratch. The problem is not solved by Docker alone, since the containerized STAR version still has to match the index version. In the code, the relevant issue is that the STAR container is fixed in the workflow modules rather than exposed as a profile or parameter. A more usable implementation would let users select a STAR container, provide matched reference profiles, or check the STAR version stored in `genomeParameters.txt` before alignment starts. This would turn a confusing runtime failure into a clear preflight message. This point also reflects current expectations for reusable Nextflow workflows, where standardized profiles, parameter validation, minimal test datasets, and traceable execution metadata are treated as part of pipeline usability rather than optional extras (refer to 1).

2. The main runtime bottlenecks in my benchmark were not the core MOA-seq processing steps but auxiliary workflow behavior. After the STAR version issue was addressed, the core steps were fairly fast. SeqPurge took about 16 to 23 seconds per sample, FLASH took about 15 to 32 seconds per sample, STAR alignment took about 32 to 50 seconds per sample, BAM processing took less than 2 seconds per sample, and MACS3 peak calling took about 3 to 4 seconds per sample. STAR alignment used about 25 GB RAM with the public mm10 index, which is large but expected for a mammalian genome. The slow or distracting parts were FastQC under the containerized execution environment, repeated whole-genome effective genome size calculation, and base-resolution `bamCoverage --binSize 1 --exactScaling`. These steps are useful for a full production analysis, but they made a small smoke test feel much heavier than the core analysis actually was. A user trying the pipeline for the first time needs a lean validation profile that tests trimming, merging, alignment, filtering, shortening, and peak calling without waiting on every QC and coverage product. The current default makes it harder to tell whether the pipeline itself works. For reproducible benchmarking, the tested setup, parameter choices, runtime, and evaluation criteria need to be explicit enough for another user to reproduce the comparison (refer to 2).

3. The effective genome size handling is both inefficient and confusingly named. The parameter `effective_genome_size_fixed` sounds as if it fixes the effective genome size value, but in the tested workflow it instead fixed the k-mer length and still launched a whole-genome `unique-kmers.py` calculation. That calculation is reference-level information, not sample-level information. Recomputing it for every sample is unnecessarily inefficient and makes multi-sample benchmarking slower than it needs to be. For standard references such as mm10 and hg38, the workflow would be more practical if users could supply a numeric effective genome size directly. It would also be easier to run and easier to document if it separated three concepts: a user-provided effective genome size value, a k-mer length used to estimate effective genome size, and a cached reference-level result reused across all samples. The current naming also makes it harder to understand what the pipeline did after execution, because the parameter does not describe the actual behavior. It also affects reproducibility because users may report that they used a fixed effective genome size when they actually triggered a genome-wide calculation. In my view, this

part of the code needs refactoring before the workflow is presented as broadly reusable.

4. The workflow does not validate derived fragment-length values before passing them into MACS3. In the external mouse samples, STAR alignment itself was successful: ERR2269362 had 75,852 uniquely mapped reads out of 91,820 merged reads, and ERR2269363 had 146,912 uniquely mapped reads out of 173,590 merged reads. The difficulty appeared after MOAflow's stringent `<81 bp` and MAPQ 255 filtering. Only 415 filtered mapped reads remained for ERR2269362, and only 353 remained for ERR2269363. With so few retained reads, one workflow branch produced an average fragment-length estimate of zero. That value was then passed directly to MACS3 as `--extsize 0` and `--min-length 0`, which caused MACS3 to fail. The code needs to check that the fragment length is a positive integer before constructing the MACS3 command. If too few reads survive filtering, the workflow ought to stop with a message that reports the number of retained reads and explains why peak calling cannot proceed. Another reasonable option would be to provide a documented fallback fragment size for smoke tests or sparse datasets. Recent recommendations for clinical bioinformatics emphasize validation, strict version control, and data integrity as practical requirements for reliable analysis operations, and the same principle applies here even though this workflow is not a clinical pipeline (refer to 3).

5. The benchmark result demonstrates that the computational path can run, but it does not support a biological performance claim. After local fixes, MOAflow did generate outputs from the two external mouse samples. The alignment rates were reasonable, and the core path reached MACS3 peak calling. The strict filtering left so few reads, however, that the resulting peak counts are not biologically meaningful. Full-length peak calling produced 3 peaks for ERR2269362 and 1 peak for ERR2269363. Shortened-read peak calling produced 4 and 5 peaks, respectively. I would therefore interpret this run as a software smoke test rather than a biological benchmark of MOAflow's peak-calling quality. A stronger public benchmark would need larger true MOA-seq or closely matched MNase-footprinting datasets with available processed outputs from the same study. Ideally, the comparison would include expected alignment counts, filtered read counts, peak counts, peak overlap, and signal-track or FRiP-like summaries. Providing such a benchmark would also make it much easier for users to distinguish a pipeline bug from an unsuitable input dataset.

Minor comments

1. The repository would benefit from a small public test dataset with expected outputs. The test would be more convincing if it did not rely solely on the authors' original data. A good example would include FASTQ accessions or bundled tiny FASTQs, a complete parameter file, reference/index instructions, expected STAR alignment counts, expected filtered read counts, expected MACS3 peak counts, and approximate runtime. For a workflow that is not species-specific, a mammalian example using mm10 or hg38 would be especially useful. A useful dataset would be small enough for a quick test but not so sparse that MACS3 receives near-empty input. This would make first-run troubleshooting much easier. It would also make it easier to test the workflow in continuous integration. Without such a dataset, each new user has to invent a benchmark before evaluating the tool.

2. The documentation needs to state explicitly that prebuilt STAR indices are version-sensitive. The required STAR version can be read from the STAR index metadata, so this needs to be documented and ideally checked automatically. Users would otherwise discover this only after a

failed alignment job. If the authors want to recommend rebuilding indices with the pipeline's STAR container, that point needs to be stated clearly. If public indices are supported, then compatible container versions need to be listed. The same section needs to explain the expected memory requirement for mammalian STAR alignment. In my test, the alignment step used roughly 25 GB RAM with the public mm10 index. That information is useful for users deciding whether to run the workflow locally or on a server.

3. Error messages need to report pipeline-level causes rather than leaving users to inspect downstream tool failures. The MACS3 failure caused by `--extsize 0` is a good example. MACS3 correctly rejected the command, but the real problem was upstream: too few reads survived filtering, and the average fragment-length estimate was invalid. The pipeline could catch this condition and print the sample name, filtered read count, computed fragment length, and suggested parameter changes. Similar checks would help for missing files, missing genome indices, incompatible STAR versions, empty FASTQ outputs after merging, and zero-size BAMs. None of these checks requires new biological methodology. They would make the existing workflow much easier to debug.

4. The output and parameter naming could be clearer. The term `effective_genome_size_fixed` is the main example because it does not mean that a fixed effective genome size value is being used. The code and documentation need separate names for a fixed numeric effective genome size, a fixed k-mer length, and a computed effective genome size. The same general point applies to smoke-test versus production outputs. Users need to be able to tell from the parameter file which outputs are required for the main analysis and which outputs are optional QC or visualization products. Clear naming matters because these values are often copied into methods sections or shared with collaborators. Ambiguous parameter names make the workflow harder to audit after execution.

5. The pipeline would be easier to adopt if the authors provided tested profiles for common environments. At minimum, I would expect profiles for a small local smoke test, a standard Docker run, and common mammalian references such as mm10 and hg38. If linux/amd64 containers are required, the documentation needs to say so and mention the expected behavior on Apple Silicon or other non-native architectures. Some of the slow behavior in the benchmark was tied to container execution rather than the biological algorithm. Users should be able to distinguish between MOAflow itself and the behavior of an emulated container stack. Clear environment profiles would also reduce the number of ad hoc edits users make to `nextflow.config`. That would improve reproducibility across labs.

References

1. Langer B, Amaral A, Baudement M, Bonath F, et al.: Empowering bioinformatics communities with Nextflow and nf-core. *Genome Biology*. 2025; **26** (1). Publisher Full Text
2. Li Z, M.Patel Z, Song D, Yasa S, et al.: Systematic benchmarking of computational methods to identify spatially variable genes. *Genome Biology*. 2025; **26** (1). Publisher Full Text
3. Lavrichenko K, Engdal E, Marvig R, Jemt A, et al.: Recommendations for bioinformatics in clinical practice. *Genome Medicine*. 2025; **17** (1). Publisher Full Text

Is the rationale for developing the new software tool clearly explained?

Yes

Is the description of the software tool technically sound?
Partly

Are sufficient details of the code, methods and analysis (if applicable) provided to allow replication of the software development and its use by others?
Partly

Is sufficient information provided to allow interpretation of the expected output datasets and any results generated using the tool?
Partly

Competing Interests: No competing interests were disclosed.

Reviewer Expertise: bioinformatics

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard, however I have significant reservations, as outlined above.