Round #1

by Gavin Douglas, 12 Oct 2022 20:08
Manuscript: **https://www.biorxiv.org/content/10.1101/2022.09.02.506364v1**

Revisions required

To Dr. Abby and colleagues,

Two reviewers have examined your manuscript and they both agree it is interesting and is of value for the field, which I also agree with. Their overlapping comments pertain to clarifying certain points of confusion, through adding additional text and figures where needed, and also clarifying how metagenome-assembled genomes could be used as input.

I also have some comments in addition to what the reviewers brought up, mainly related to adding additional clarification. Please see below.

I think after addressing all of our combined comments that your manuscript will be clearer, particularly for users new to the MacSyFinder framework. Please let me know if you need clarification on any requested changes.

All the best,

Gavin Douglas

Dear Dr. Douglas, thanks a lot for the positive assessment of our work, and for your relevant comments and suggestions that helped us to improve our manuscript. In the following, we made our best to reply to the comments and suggestions made by you and the two reviewers. We hope you will now find our manuscript suitable for recommendation at PCI Genomics.

Recommender's comments

Major

I think reviewer #2 had great suggestions for future development. I think they are beyond the scope of this specific manuscript though, with the exception of providing some example input and output files. It would be helpful for users to have these example files so they can try running the tool themselves and confirm they are getting the right output. Ideally, this would be for all common types of input formats.

One simple way of doing this would be to upload some example input/output files to FigShare and then add some example usage commands using these files to the tool README.

We have now added an example genome on Figshare and provide the expected output files when running MacSyFinder with TXSScan using the two most common mode “ordered_replicon” and “unordered”: https://doi.org/10.6084/m9.figshare.21581280. Another, more comprehensive set of examples and genomes is available here: https://doi.org/10.6084/m9.figshare.21716426.v1. In addition, references to these datasets and examples were included in the README.md of the MacSyFinder Github repository, and in the Documentation, following the Quick start section.

How long does the tool take to run on typical input files and how much memory does it use? Can it be run on multiple cores, and if so, roughly what is the relative increase in run-time (e.g., is it linear)?

We ran several new analyses and clarified these two points in a new paragraph and accompanying figures in the main text (section II of Results). We provide run times and memory usage for the analysis of several datasets with increasing number of genomes (Figure 5C&D). We also discuss more thoroughly the behaviour (in terms of resources) of the different parts of the MacSyFinder algorithm. MacSyFinder usually process a genome within a few seconds. And yes, MacSyFinder can be ran on multiple cores, the HMM similarity search can be parallelized (--worker option). The relative decrease in run-time would be roughly linear for the HMM annotation part (and of course depends first on the number of HMM profiles to run), as it can be fully parallelized. This adds to a flat, irreducible time that corresponds to the search for the best solution. This latter step depends heavily on the number of hits and clusters that are under analysis. This is now documented in the new dedicated section in Results, where it is also shown that run time heavily depends on the complexity of the macsy-models being used. We also added the description of a workflow to run MacSyFinder in parallel on many genomes.

I think the manuscript would be greatly improved if a brief discussion was added on the following three general areas. I think a short section could be added to the discussion/results to talk about these caveats (if they are indeed caveats).

First, how sensitive are the results are to different default values for the scores (e.g., as mentioned at L320). Does tweaking these parameters typically make much difference? Also, how were the default scores chosen? It sounds like they were selected to produce reasonable results on datasets where specific systems are known to be present, but that’s not totally clear. As a potential user I would be worried that these default values would create biases when applied to genomes with different characteristics than in the training datasets. I’m guessing this kind of investigation was important when developing the first version of the tool, or for one of the other related manuscripts, but either way it would be good to quickly refer to any past validations for readers unfamiliar with the earlier work.

These scores were introduced in the new version, and the default values were chosen so that higher scores would be obtained for non-redundant, more complete systems – as explained in the main text. Also, the choice was made in order to give priority to mandatory over accessory components, and to give priority to the main listed gene over the ones listed as “exchangeables”. Here the relative order of the values of the weights are more important than the absolute values. We attempted to clarify in the text. We used TXSScan to show the differences between V1 and V2 (see also below) and demonstrate the algorithm works much better with the new scoring system. In the case of systems with many specific and conserved elements in single loci, the program would probably be quite insensitive to small changes in the scoring/weighing parameters. For other types of systems, these parameters may have to be explored and tested to evaluate the biological relevance of the systems detected, as part of the modelling process. All these values can indeed be parametrized by the modeler (and shipped with the models via dedicated “model_conf.xml” files) if the behavior needs to be changed. For instance in the case of CasFinder, since discrimination between types or subtypes is mainly based on gene content, mandatory, accessory and exchangeable components were assigned the same weight to enable an efficient detection and typing of the Cas loci.

Second, and similarly, how confident can you be in any definitions of co-localization for specific systems across different lineages? My concern is that these co-localization relationships may be well-described in certain lineages, but that operon structure may be less in others, and so these tools might not work as well. Do you think this is a concern? And if so, what should users watch out for?

The recommender is right in the sense that biological expertise is a pre-requisite to enable the modelling of the systems. It also depends on how confident the user is to search for distant variants of existing systems. There is usually a strong bias in our knowledge of the systems, and applying any annotation tool to very distant lineages should always go together with a more careful exploration of the results. Because developing the models is a work in itself, we have made several tutorial-like publications describing the modelling process (cited in the main text), as well as developed a thorough Modeller section in the Documentation of v2.

Last, how many systems are affected by the choice of whether components are allowed to overlap across systems are not (e.g., L330 and the examples later)? Knowing this would help give users an idea of whether it’s worth exploring the multi_system/multi_model options. It’s also not totally clear to me how much one might expect the overall results to change when using those options.

This is an integral part of the modelling process. The decision to use these options must rely on biological knowledge. There is thus no general answer to that. In the case of CasFinder, and as detailed in main text, section III, it was known that tandem systems could share the adaptation module (composed of several genes). It was therefore important to allow these components to be present (shared) in several tandem systems in order to detect overlapping systems.

Minor

The tool's link should be added https://github.com/gem-pasteur/macsyfinder to the end of the abstract

Done

"Nanomachines" is not a commonly used term in this context (to my knowledge at least), so I recommend that the authors define it or use a simpler term

Yes you're right, furthermore, it was used only once in the manuscript. The term was replaced with the more commonly used "machineries".

In the intro it is implied that "components" is synonymous with "proteins". If this is the case, then I think the authors should explicitly explain why they opted for this term over simply saying proteins. I'm thinking that perhaps the authors actually mean that components can be non-protein coding genes and perhaps noncoding elements as well, which would explain why this more general term is used. Either way, this should be explained (but based on the CRISPR-Cas section I don't think this is the case).

Yes, it is true that it is here unnecessary to introduce the term "components" in place of genes and proteins, even though in the future we might consider adding noncoding elements to the modeling design. We changed the term to "proteins" or "genes" depending on the context.

L33 – rather than "performing function of interest" I would reword to be "performing the same function as this system"

We changed the text.

L34 – rather than "coherent" I would say "distinct, non-co-localized" perhaps to be clearer? (If that is what you mean there)

Using "conversely" was misleading here and probably a mistake. We changed the sentences to clarify our meaning.

L67 – I think the limitations should be in a separate paragraph, as they are the key motivation for making the new version, so you want to make readers don't miss them.

Yes, thanks for this suggestion, we have now the limitations in a separate paragraph.

L70 – Unclear what “component-specific filtering criteria” refer to here, without looking at methods. It would be nice to see a quick example.

We attempted to clarify the sentence, it now reads “protein-specific criteria to filter the HMMER hits when annotating the genes for the systems detection”, however adding an example at this stage of the manuscript (end of introduction) seemed a bit early.

L77 – Instead of “novel” I think you mean “new”

Done

L82 – should be “takes” rather than “gets”

Done

L83 – “fasta” should be “FASTA”, upon all usages

Done

L102 – change “have” to “had”

Done

L120, L476 – change “…” to “, etc.”

Done

L158: A quick description of what GA scores are based on (or an example) would be useful

Yes, thanks for the suggestion, it was missing. A brief description of GA scores is now included.

E-values are well known, but I haven’t seen the term “i-evalue” before. Maybe I missed where it was defined, but if not this should be defined and distinguished from normal E-values.

The “i-evalue” (independent e-value) is an internal statistical value computed by HMMER. We now add its meaning upon first appearance of the term.

L328 – should be “are” rather than “were” for both instances (i.e., use present tense when describing what the tool does in general, and not just what it did on a single occasion)

Yes thanks a lot for pointing this out, we corrected these mistakes.

L335-336: It’s not clear to me what the edges are based on in this case (i.e., do they

represent the binary relationship of whether the systems can be compatible or not? Or can they be weighted?). I think it would be useful to add a sentence clarifying that for potential users who aren't familiar with the clique search approach.

The sentence was modified to clarify it. It now reads: "The program builds a graph where each node represents a system with its associated score (as a weight), and where only compatible systems are connected with an edge."

L393 – Capitalize "archaea" and "bacteria" (here and elsewhere) to be consistent with earlier usage

Done

L468 – rather than "we present its application" I would say "we apply it"

Done

L499 – Make "Multi_model" lowercase (since I'm guessing the module is case sensitive?)

Yes, thanks! Done

L524: "that the same cluster is" should be "for the same cluster to be"

Indeed, thanks! Done

Reviews

Reviewed by Max Emil Schön, 26 Sep 2022 09:34

First of all congrats to the authors for a very nice tool and an interesting corresponding paper! I do not have many comments, as I think the usefulness and impact of the tool is apparent and the paper presents its main points clearly and concisely. Importantly, all described updates and improvements are well justified and will likely help in spreading the application of MacSyFinder and improve its predictions. The addition of a dedicated structure and automatic handling of models is also very handy and should significantly lower the threshold for new users, as will the intuitive installation process via pip, conda or docker. The available online documentation is very extensive, detailed and well structured. The application of GA scores (which I did not know about before) is very elegant and should reduce false positives. The ability to re-analyze runs using macsyprofile could be very useful for in-depth searches for e.g. novel variations of molecular systems.

Below are my comments on the paper, please consider them suggestions. I would be very interested to see your comments on these issues, but I think the paper could be considered complete even without theses additions.

First of all, we thank the reviewer for the very positive assessment of our work and the relevant suggestions. We made our best to address the different suggestions and comments.

general comments

While the authors describe the improvements that were made between v1 and v2, I think it would be great to see some visualizations/data substantiating this. I could imagine e.g. an additional figure showing the discovery of a system that was not detected in v1 but now is found by v2. Information on the number of false positive/false negative detections, on the gene or system level, for both versions on one or several example genomes could be interesting as well. Showing this in a figure would help the reader appreciate at a glance why the development of such a major update was indeed necessary.

Figures 6 and 7 (formerly 4 and 5) and the sections III and IV illustrate some cases where version 1 would have produced less biologically relevant results. In addition to that, we have now included a new paragraph in Results based on new analyses that give comparative statistics between v1 and v2 results, based on the run of the models for TXSScan on the same set of genomes. The results are displayed on the new Figure 4.

Have you assessed the impact of fragmented assemblies (e.g. incomplete MAGs) on the performance of MacSyFinder v2? I think it could be worthwhile to many readers how this affects the prediction, since many people will likely apply it on genomes that are, to a certain degree, incomplete or fragmented. I am for example thinking of systems that are separated by contig boundaries and therefore potentially not in the correct order as they are on the genome.

We have now included a few sentences in the main text in the Materials & Methods (section “Input & Output files”) on the use of MacSyFinder on fragmented assemblies or incomplete genomes. We also created a “How to” section in the User guide from MacSyFinder’s documentation:

“Of note, recommendations on how to use MacSyFinder on incomplete or fragmented genomes are included in the “How To” section of the User guide. In a nutshell and depending on the level of assembly and completeness of the genome, we recommend to run MacSyFinder with the “ordered_replicon” mode, which can be complemented by the results of an “unordered” run. Results using the “ordered replicon” option on draft genomes have to be considered with care.”

Here is the link to the “How to” section in the Documentation: https://macsyfinder.readthedocs.io/en/latest/user_guide/FAQ.html#how-to-deal-with-fragmented-genomes-mags-sags-draft-genomes

We are also investigating the possibility to create a dedicated search mode for a future version of MacSyFinder.

specific comments

abstract: I think it's worth mentioning here that the target organisms are only bacteria and archaea, but not eukaryotic microbes.

We left is as is, as one could also consider using MacSyFinder to detect viruses or prokaryotic elements embedded in eukaryotic genomes. Moreover, some fungi have biosynthetic gene clusters that could also be explored using MacSyFinder (see for example a review by Gills & Gloer here: https://doi.org/10.1128/microbiolspec.FUNK-0009-2016).

L 15-16: I find this sentence a bit hard to parse, consider clarifying it.

Done, we shortened it. It now reads:

"Finally, we have updated and improved MacSyFinder popular models: TXSScan to identify protein secretion systems, TFFscan to identify type IV filaments, CONJscan to identify conjugative systems, and CasFinder to identify CRISPR associated proteins."

L 109: Is there some sort of XML schema for validating the model definition in the new macsy-model packages? This could be useful for future modellers.

The "*macsydata check*" subcommand is actually dedicated to that (see Table 2). On top of this, we have now implemented and added to MacSyFinder v2 the "*macsydata init*" subcommand that will create a template macsy-model package for the modellers, as suggested by the second reviewer.

L 172-177: A somehwat difficult/unclear paragraph that could perhaps be clarified.

We attempted to clarify by explaining what are the GA scores and by rewriting the paragraph:

"Many of the HMM protein profiles used in MacSyFinder models already include GA thresholds because they were retrieved from PFAM or TIGRFam, which systematically use them (Sonnhammer et al., 1997; Haft et al., 2003). Yet, some other profiles lacked GA thresholds. To remediate this limitation, we modified these profiles to include the threshold GA scores. We did this for CasFinder, TXSScan, CONJScan, and TFFscan profiles (see Table 1)."

L 219: There's something wrong with this sentence I think.

Upon this reviewer and reviewer #2 request, we changed it to "The code was ported to Python 3".

Table 1: Consider changing ‘Nb’ to ‘No.’

Done

L 385-389: This sentence could be rephrased for more clarity.

This was done, it now reads:

“To illustrate the interest of the novel file architecture, we created a new version of “TXSScan” (v1.1.0) that gathers the models for the type IV filament super-family (“TFF-SF”) and for the protein secretion systems (former “TXSScan”, v1.0.0) (Abby et al., 2016; Denise et al., 2019). These models were also ported to the grammar of MacSyFinder v2.”

Macsy-models github organization: link to https://github.com/macsy-models/.github/blob/CONTRIBUTING.md is broken

Thanks a lot for pointing this out, the link has now been fixed to point to the following: https://github.com/macsy-models/.github/blob/main/CONTRIBUTING.md

Reviewed by Kwee Boon Brandon Seah, 12 Oct 2022 11:14

This manuscript (https://doi.org/10.1101/2022.09.02.506364) describes a new version (v2) of the tool MacSyFinder, which searches for gene clusters in microbial genomes (e.g. coding for macromolecular complexes) by first using HMMs to identify individual protein components, and then user-defined models of essential and accessory components to annotate the clusters. In this version, the code was updated to Python 3, the modeling and search engine were improved, and new tools were added to make it easier to distribute and install models.

The previous version of MacSyFinder (https://doi.org/10.1371/journal.pone.0110726) and tools in which it has been integrated, such as CrisprCasFinder, appear to be popular and widely adopted. The updates should improve the user-friendliness of the software.

Specifically, the software is straightforward to install via different distribution channels (Conda, pip, Docker container). The macsydata tool and the use of Git repositories to distribute models is convenient for both users and modellers, and is a good idea for getting more community participation and to make the tool more extensible. Extensive documentation for users, modellers, and developers is available online. Several of the most often used models such as those for CRISPR-Cas systems have been ported to MacSyFinder v2.

We thank the reviewer for the positive assessment of our work and manuscript, and for the relevant comments. We made our best to address the different suggestions and comments.

Major comments

Comparison to other tools

Could the authors briefly discuss the intended use cases for MacSyFinder and how they differ from other tools for pathway and gene cluster annotation, such as KEGG Mapper, Pathway Tools, and Antismash (secondary metabolite gene clusters)? For example, the logical expressions used to define KEGG Modules has similarities to the model definitions in MacSyFinder, but don't incorporate information about collocation, as far as I know. This could help give some context for readers and users.

It is true that some context was missing. We now added a paragraph in the introduction section to give some context regarding other annotation tools. The main difference between the mentioned tools and MacSyFinder, is that MacSyFinder is a generic framework allowing to model annotation rules for any macromolecular system of interest. Moreover, the program relies both on gene content AND co-localization rules to predict the presence of a given macromolecular system, and provides a yes/no to the question of the presence of the system. Co-localization rules are used in AntiSmash but not in KEGG Mapper. The content and co-localization rules of AntiSmash are probably well-suited for the detection of biosynthetic gene clusters, but they do not offer the range of possibilities that MacSyFinder grammar offers (specify gene specific features, have sets of genes listed that fulfil a same function, the possibility to have loner and forbidden genes, multi-loci or single locus systems…). Examples of intended use cases are given all along the Results section with the depiction of MacSyFinder most popular models.

Input data formats

It appears to be only possible to analyse either complete closed genomes where the gene order is known (in ordered replicon mode), or treat each gene independently (unordered mode), which in the v1 paper was suggested for the analysis of metagenomes. However it is now quite common to work with draft genomes and metagenome assembled genomes (MAGs) where the gene order is only partly known. Is it possible to exploit this partial information? In the user guide, a third input option, "gembase" is offered as a solution for analyzing multiple genomes at once, but this is not mentioned in the paper. Would it be appropriate to use this input mode for draft genomes?

Given this comment and that of reviewer #1, we clarified in the text the different input types and what they entail. This is now done in the dedicated, 1st section of the Materials & Methods. The “unordered” replicon mode does indeed ignore the gene order, but is not limited to an individual gene treatment, since the number of each different genes found is computed, and the quorum of genes applies, providing an idea of the “completeness” of the potential macromolecular system found in the genome. We also added a paragraph on the use of MacSyFinder on draft genomes and metagenome assembled genomes (see reply to reviewer #1).

As for the “gembase” format, it consists in the analysis of multiple genomes at once (one FASTA file with all sets of proteins combined), and it is now advised to use the Nextflow workflow provided to run such a task. It is thus not well-suited to analyse

contigs of MAGs. We hope this is now clearer with the updated paragraph on input types.

Improvements to search engine

My expertise is not in algorithm design so I can't comment too much on the method itself, but could the authors give an example of a suboptimal solution found by v1 compared to the improved results from v2 (see lines 431-465)? It would be helpful to have a concrete example of what these look like in v1 so that we can see how v2 produces better results.

We have now added a new paragraph and figures (Fig. 4) dedicated to a systematic comparison between the v1 and v2 versions when using TXSScan on the same set of genomes. We also exemplify how the behaviours of the two versions differ in the Results section III and IV and on Figure 6 (formerly 4).

I also found it difficult to understand the description of the heuristic used by v2 to simplify out-of-cluster components (lines 454-456); perhaps a diagram might be helpful here. The results shown in Figure 4 appear to be the output from v2 only.

We attempted to clarify the text, and introduced a new supplementary figure explaining the rationale behind the proposed heuristics (Figure S2) that is referred to in the main text.

Predicting complete vs. incomplete systems

An incomplete system could be missing one or more of its components, whether mandatory or accessory. In the example shown in Figure 5C (dCONJ typeF), one of the three original mandatory genes (traI) is still mandatory even though the model is incomplete. However, isn't it possible to have an incomplete copy of the T4SS where only traI is missing? Could one define a model for incomplete systems simply by changing all genes that are mandatory to "accessory"?

It is possible to have an incomplete copy of T4SS where only traI (the relaxase) is missing. However, we wanted to distinguish conjugative T4SS from other non-conjugative T4SS. The latter are generally devoid of relaxase in their complete forms. This is why we have set the relaxase as mandatory. In the current version of MacSyfinder, at least one gene per model needs to be mandatory. However, the minimum number of mandatory genes needed to reach the quorum can be set to 0 using min_mandatory=0. In that case, the proposition of the reviewer would work.

Suggestions for future development

Finally I have a few suggestions for future versions of the software that the authors could consider.

- Bundle some example data for users to test the software after installation, and also to help them learn how to use the various options.

Yes, thanks a lot for this suggestion, we have now included links to example files in the README file of MacSyFinder repo, in main text (Input/output section of Materials & Methods) and in the Documentation (see also our reply to the recommender). They are available here: https://doi.org/10.6084/m9.figshare.21581280 and here https://doi.org/10.6084/m9.figshare.21716426.v1. They come in addition to all the functional tests that are automatically ran upon installation to check the success of the procedure.

- In addition to the model validator macsydata check , how about something like macsydata init to set up the directory structure and template files to get started with a model?

We thank the reviewer for the very relevant suggestion, we could implement this new feature to ease the design of new macsy-models. This feature is now available in the new MacSyFinder release 2.1 as the “macsydata init” subcommand.

- Allow annotated nucleotide sequences as input too, e.g. feature tables + nucleotide Fasta + genetic code, or EMBL/Genbank files. This would simplify the specification of feature coordinates, e.g. when we are working with draft genomes where a complete replicon is not available, only individual contigs.

We thank the reviewer for this suggestion. While we acknowledge that this is a very interesting feature, it is for now beyond the scope of this paper, but it is definitely on our TO DO list for the release of a future version of MacSyFinder.

Minor comments

- Figure 1. Panels 3.1 and 3.2 "exam" should probably be "examination" or "examine"

  Yes, thank you, it is done.

- Line 219. I think the authors mean "ported to Python 3" instead of "carried under Python 3".

  Thanks a lot, it is much better indeed! Done

- Line 295. Suggest to change "xm" and "xa" to use subscripts for m and a, to avoid giving the impression that these are products x times m or x times a.

  Thanks for the suggestion, this was done.

- Lines 338-339. Could the authors clarify what "this" in "this is the definition of a clique" refers to? Is it referring to the set of connected systems or the connections between them (or are these equivalent?).

  Yes we agree this was unclear. It now reads: “A sub-graph where all nodes are inter-connected is the definition of a “clique”.

- Line 446. Which panel in Figure 4 is being referred to?

We now specify it is Fig. 4D (now 6D).

- Line 583. The word "degenerate" here appears to be used here in the sense of "decayed", "degraded", or "incomplete". However it can easily be confused with the technical meaning of "having multiple elements that correspond to a single element", especially in the context of defining a model. I suggest sticking to "complete" vs. "incomplete" or similar terminology used elsewhere in the manuscript.

We have now replaced the words “degenerated” and “degenerate” by “decayed”.

- The software is archived at the Software Heritage Archive, but this isn't mentioned in the manuscript. I suggest also citing the archive DOI.

We have added the permalink to the Software Heritage snapshot of the MSF v2.0 release: https://archive.softwareheritage.org/swh:1:dir:4a5136d45e82edfd4d06ce93cd3892195230e1d8;origin=https://github.com/gem-pasteur/macsyfinder;visit=swh:1:snp:344cf013fc7a3d44b87a722da3fe87d33f8c07bc;anchor=swh:1:rev:86781d479c3361cb0728161bc8ab23e4adca6c28. And release 2.1 resulting from this round of revision is available on Github and on Figshare. We will provide the permalink on Software Heritage to the version 2.1 when available.