Title: VIGA: a sensitive, precise and automatic *de novo* VIral Genome Annotator.

Running head: *De novo* viral genome annotation

Authors: Enrique González-Tortuero[1], Thomas David Sean Sutton[1], Vimalkumar Velayudhan[1], Andrey Nikolaevich Shkoporov[1], Lorraine Anne Draper[1], Stephen Robert Stockdale[1,2], Reynolds Paul Ross[1,3], Colin Hill[1,3,*]

[1]APC Microbiome Ireland, University College Cork, Cork, Co. Cork, Ireland

[2]Teagasc Moorepark Food Research Centre, Fermoy, Co. Cork, Ireland

[3]School of Microbiology, University College Cork, Cork, Co. Cork, Ireland

*To whom correspondence should be addressed

Email addresses:

EGT: enrique.gonzaleztortuero@ucc.ie; enriquegleztortuero@gmail.com

TDSS: t.sutton@umail.ucc.ie

VV: mail@vimal.io

ANS: andrey.shkoporov@ucc.ie

LAD: l.draper@ucc.ie

SRS: stephen.stockdale@teagasc.ie

RPR: p.ross@ucc.ie

CH: c.hill@ucc.ie

Abstract

Viral (meta)genomics is a rapidly growing field of study that is hampered by an inability to annotate the majority of viral sequences; therefore, the development of new bioinformatic approaches is very important. Here, we present a new automatic *de novo* genome annotation pipeline, called VIGA, to annotate prokaryotic and eukaryotic viral sequences from (meta)genomic studies. VIGA was benchmarked on a database of known viral genomes and a viral metagenomics case study. VIGA generated the most accurate outputs according to the number of coding sequences and their coordinates, outputs also had a lower number of non-informative annotations compared to other programs.

Keywords: archaeal virus, bacteriophage, bioinformatics, *de novo* annotation, eukaryotic virus, genome annotation, metagenomics, viral genomics

Introduction

Virology is a diverse scientific discipline. While many researchers are interested in discovering and characterising pathogenic eukaryotic viruses [1], recently there has been an increased interest in revealing bacteria- and archaea-infecting viral communities [2]. The number of viral metagenomic studies is increasing due to the development of new sequencing technologies and the reduction in costs. However, due to the volume of information that these platforms generate and the large proportion of viral sequences sharing little or no homology to known viral genomes ('viral dark matter', [3]), new bioinformatic tools are required to examine viral contigs and genomes [4].

Viral annotation methods differ depending on the host organism. Bacteriophages and archaeal viruses are annotated using prokaryotic genome annotation software or web-servers such as RAST [5], Prokka [6] and RASTtk [7]. However, these bioinformatic tools are optimised for bacterial sequences, not viruses (despite the improvements in RASTtk to annotate phage sequences [8]). In contrast, eukaryotic viruses are annotated using close-reference based methods such as FLAN [9], VIGOR [10] and ViPR [11]. In a similar way, VirSorter [12] and VirusSeeker [13] were designed to predict putative prokaryotic viral contigs in metagenomic datasets. However, both programs predict viral contigs according to the presence of viral proteins using reference databases, and close-reference homology-based methods can underestimate true viral diversity due to database limitations [3,14]. Therefore, in this manuscript, we present a new modular and automatic *de novo* genome annotation bioinformatic pipeline, called VIGA (VIral Genome Annotator), to annotate viral sequences.

VIGA automatically detects open reading frames from a FASTA or multi-FASTA formatted file. VIGA then annotates protein sequences by detecting homologues in a BLAST ("Slow") or a DIAMOND ("Fast") protein database, with or without Hidden Markov Model (HMM) protein detection against a protein database. The different methodologies for annotating viral contigs and genomes allows the user to specify options that sacrifice annotation detail in exchange for increased speed, which is required when dealing with larger metagenomic datasets. In addition, VIGA also automatically detects (1) the topology of viral contigs, (2) the presence of rRNA, tRNA and tmRNA sequences, (3) potential CRISPR repeats and (4) tandem or inverted repeat sequences. Finally, VIGA outputs a FASTA file that includes user specified modifiers, a GenBank file and a five-column tab-delimited feature file to ease the upload of annotated contigs and genomes to various database repositories and genome visualisation platforms.

Results

Benchmarking of VIGA

The performance of VIGA, Prokka, RAST and RASTtk was tested using a benchmark database comprising 191 sequences belonging to 138 different viruses (52 bacteriophages, 72 eukaryotic and 10 archaeal viruses, and 4 virophages; Additional file 1). Of the 72 eukaryotic viruses, 11 have multipartite genomes. Experimental evidence is available for the coding sequences of 117 out of the 123 sequences of eukaryotic viruses, 28 out of 52 sequences of bacteriophages, 3 out of 10 sequences of archaeal viruses, and none of the 4 virophage sequences used. When bioinformatic methods were used to annotate these viral genomes in the original data, a wide variety of methods were employed, including GeneMark [15], GLIMMER [16], NCBI ORF Finder and the University of Wisconsin Genetics Computer Group [17]. The outputs of VIGA, Prokka, RAST and RASTtk were evaluated according to three different parameters: (1) number of coding sequences, (2) coordinates of coding sequences, and (3) power of prediction.

Firstly, the accuracy and the precision of the number of viral coding sequences were estimated using general linear models. Accuracy was measured by the slope, and precision was measured according to the coefficient of determination (R^2). To compare all these linear models, analysis of covariance (ANCOVA) was performed. In a general overview, the programs delivered different estimates of the number of coding sequences (ANCOVA: $p < 2\times10^{-16}$). In fact, although all programs tended to overestimate the number of genes, VIGA provided the most accurate predictions (i.e. accuracy is closest to one, Fig. 1A). Moreover, VIGA and Prokka had very similar values of precision (Table 1). When compared according to viral host, similar results were found only in the case of eukaryotic viruses (ANCOVA (Archaeal viruses): $p = 0.922$; ANCOVA (Bacteriophages): $p = 0.734$; ANCOVA (Eukaryotic viruses): $p = 1.560\times10^{-15}$; Figs. 1B-D). Interestingly, when bacteriophages were considered, only RASTtk tended to overestimate the number of coding sequences (Table 1).

Secondly, F_1 score, a measure that combines precision and sensitivity, was used to predict the quality of the coordinates of the viral coding sequences. Moreover, to evaluate the occurrence of false positives (i.e. false coordinates considered as true; type I error) and false negatives (i.e. true coordinates considered as false; type II error), false discovery rate (FDR) and false negative rate (FNR) were examined. VIGA scored very highly for both bacteriophages and eukaryotic viruses. In eukaryotic viruses the highest false discovery rate (FDR) was associated with RASTtk, while RAST had the highest false negative rate (FNR). For bacteriophages the highest FDR and FNR were

obtained for Prokka. In the case of archaeal viruses, VIGA again had the highest precision, while the highest sensitivity was noted in RASTtk (Table 2).

Finally, the power of prediction of all programs was measured by considering the number of non-informative annotations (i.e. all proteins classified as “hypothetical protein”, “uncharacterized protein”, “ORF”, “predicted protein”, “unnamed product protein” or “gp[Number]”). For these analyses, two different modes of running VIGA were considered – “Slow” (when BLAST and HMMER are used to annotate the genes) and “Fast” (when DIAMOND alone is used for annotation). Kruskal-Wallis (KW) test was performed to detect potential differences in the power of prediction of all three programs (including both variants of VIGA) and significant differences between the various programs were observed (KW test: $p = 1.683\times10^{-53}$). In all cases, no significant differences between VIGA-Slow and VIGA-Fast were found (Nemenyi test: $p = 0.853$). In fact, while RAST and RASTtk had the highest number of non-informative annotations, both VIGA modes had the smallest number (Fig. 2A). Additionally, there were significant differences among programs independently of the viral type (Table 3). In all cases, VIGA achieved optimal annotation, having always the smallest number of non-informative annotations. In contrast, Prokka had the highest amount of non-informative annotations in prokaryotic viruses (Figs. 2B-C) and RAST and RASTtk had the highest amount of non-informative descriptions in eukaryotic viruses (Fig. 2D).

Case study: healthy human gut phageome

To evaluate the performance of VIGA on a metagenomic dataset, VIGA, Prokka, RAST and RASTtk were run using a subset of 202 non-redundant contigs from the metavirome of healthy individuals [18]. VIGA was executed using 10 cores in two different ways: (1) using only DIAMOND (VIGA-Fast), and (2) using BLAST and HMMER (VIGA-Slow). These 202 contigs were composed of 65 short contigs (<15 kb), 99 medium-size contigs (15 – 70 kb), and 38 long contigs (>70 kb). Two different parameters were evaluated: (1) Speed of the program, and (2) power of prediction. Only RASTtk was unable to annotate these contigs.

To test the speed of VIGA-Slow and VIGA-Fast, both VIGA modes and Prokka were run in a local server (Lenovo x3650 M5, with 48 Intel Xeon 2.6GHz Processors, Ubuntu 14.04, 512 GB of RAM) using 10 processors. VIGA-Slow took 19,283 minutes (13 days 9 hours 23 minutes) to process all 202 contigs of this data set, while VIGA-Fast took 809 minutes (13 hours 29 minutes). In contrast, Prokka took 3 minutes to annotate all contigs. Unfortunately, we cannot estimate the time that RAST took to annotate these genomes due to be an external web server.

Finally, the power of prediction of all programs was evaluated by comparing the number of non-informative annotations as indicated above. Significant differences between the various programs were observed (KW test: $p = 2.121\times10^{-93}$). While Prokka had the highest percentage of non-informative descriptions, VIGA-Slow had the smallest number (Fig. 3A). In contrast to the benchmark, there were significant differences between VIGA-Slow and VIGA-Fast on a metagenomic dataset. VIGA-FAST had a higher percentage of non-informative descriptions than VIGA-Slow (Nemenyi test: $p = 3.900\times10^{-14}$). Surprisingly, no significant differences between VIGA-Fast and RAST were found (Nemenyi test: $p = 0.440$; Fig. 3A). When the different size of contigs were considered, significant differences between the non-informative annotations of the programs were found (KW test ("Short"): $p = 4.650\times10^{-24}$; KW test ("Medium"): $p = 3.731\times10^{-63}$; KW test ("Long"): $p = 8.708\times10^{-16}$). This is a similar pattern detected independently of the contig size (Figs. 3B-D).

Discussion

In this study, VIGA, a new bioinformatic pipeline for viral genome annotation, was tested against RAST, RASTtk and Prokka using a benchmark comprising of 138 viruses. In fact, this is the first genome annotation pipeline to be benchmarked using viral data, as previous validation of these programs tended towards the use of bacterial genomes [5,6]. When all these bioinformatic annotation pipelines were benchmarked, VIGA successfully outperformed the others in all test parameters. After validating VIGA, it was used to annotate the phages in a subset of the Manrique et al. healthy human gut phageome dataset [18]. This subset was based on the phages predicted by VirSorter [12], which could miss some viral contigs such as variants of crAssphage [19]. In that instance, this viral gene annotation is dependant on the proficiency of VirSorter.

When the benchmark of 138 viruses was performed to measure the accuracy and precision of the number of coding sequences, VIGA had the highest values of accuracy and precision in the general overview. The only differences in the number of coding sequences were shown in eukaryotic viruses. Additionally, when the quality of the coordinates of these coding sequences was analysed, RASTtk had the highest false discovery rate and RAST the highest false negative rate for eukaryotic viruses. All these observations strengthen the idea that all tested programs were developed for prokaryotic viruses. Although the most abundant viruses in the biosphere are bacteriophages [20], it was not possible to annotate around 80% of putative viral contigs in previous studies on viral diversity [14], indicating the extensive presence of 'viral dark matter'. The nature of

this 'viral dark matter' is related with the lack of knowledge in viral diversity, and due to the use of homology-search methods to classify and to annotate them [3]. In that sense, classification of viruses (independently of their hosts) currently should not only be performed using close-reference based homology searches because they could underestimate the real viral diversity based on the limitations of databases.

The quality of the coordinates of the coding sequences in the viral benchmark was higher using VIGA than with the other programs. Although this result suggests that VIGA is reliable, it is also important to note that there was only experimental evidence of the coding sequences in 68 of 74 sequences of eukaryotic viruses, 28 of 52 sequences of bacteriophages, and 3 of 10 sequences of archaeal viruses. In fact, although the development of automatic genomic pipelines such as RASTtk or VIGA can facilitate the prediction of genes in viral sequences, some features such as introns, morons or regulatory elements need manual refinement [8]. For this reason, all bioinformatic genome annotations are putative until validated using experimental procedures such as cDNA-gDNA hybridization [21–23], proteomics [24–26] or transcriptomics [27–29].

Analysis of the power of prediction of annotation pipelines showed that RAST and RASTtk tend to generate a higher number of non-informative annotations, while VIGA had the smallest number in all cases. Therefore, VIGA-Slow mode has the potential to provide more information on encoded viral genes than other genome annotation bioinformatic pipelines, which rely exclusively on homology-based methods such as BLAST, BLAT [30] or DIAMOND. Primarily because these methods increase the number of non-informative annotations, especially in novel viruses, as demonstrated in the described metagenomic case study. Viral dark matter [3], or the unknown fraction of the virome, is a prevalent hurdle in virome research and lack of homology to sequences in databases hampers most annotation methods. It is also important to note, that where annotations are available, many have been generated through bioinformatics and do not have supporting experimental evidence. It is therefore very important to consider the source of functional information for proteins when annotating new viruses unless empirical evidence is available [8,31].

Proteins related to viral function can have highly conserved sequences, such as the hepatitis B virus core protein [32], Dengue virus polyprotein [33] and the influenza A virus nucleoprotein [34], because non-synonymous mutations in these proteins could hamper viral function. For this reason, the use of HMMs was implemented to predict the putative function of these genes. Use of HHPred or InterProScan is suggested to increase the power of protein annotation predictions [31,35,36].

Although the implementation of these programs could be beneficial for VIGA and it will be implemented in future versions, HMM-based methods are slower than homology searches as noted in the case study. Another alternative to these HMM-based methods could be the implementation of homology-independent annotation methods such as iVIREONS [37] or VIRALpro [38]. All these methods use machine learning to predict structural phage proteins such as capsid, collar and tail proteins [8] and are also scheduled for implementation in future versions of VIGA. Finally, when the power of prediction of all genome annotation pipelines was analysed, a lack of criteria for gene annotations was found, making it difficult to compare between the outputs of the different programs. For this reason, the implementation of a standardised genome annotation system would ease the comparison between genomes [39,40] using some (alpha)numerical classifications such as the Enzyme Codes [41], Clusters of Orthologous Groups [42], KEGG Orthology [43] or the Prokaryotic Viral Orthologous Groups [44] which could be added in the genome annotation output.

Conclusions

The number of viral metagenomic studies is increasing as a consequence of the development of high throughput sequencing platforms and cost reductions. However, there are few software programs to annotate the viral sequences and never before have these programs been benchmarked against each other. In this study, we present VIGA, a new automatic *de novo* genome annotation bioinformatic pipeline to annotate prokaryotic and eukaryotic viral sequences from genomic and metagenomic studies. VIGA allows the most accurate, precise and sensitive annotation of viral genomes when benchmarked using 138 known viral genomes. VIGA can be executed using BLAST or DIAMOND to annotate proteins according to homology, with the option to also use HMMER to improve these annotations based on HMMs. The use of HMMs will enrich the annotation detail of the viral contigs, but will decrease the speed of the program. Where increased speed is required for example when dealing with larger metagenomics datasets.

Materials and methods

Workflow of the software

Overview. VIGA is an automatic *de novo* viral genome annotator implemented in Python 2.7 (requiring Biopython [45]) and designed to annotate complete and draft viral and phage genomes comprising single or multiple contigs (Fig. 4). As an input, VIGA accepts a DNA FASTA file with the (putative) viral contigs. These sequences are processed to predict the topology of the contigs (i.e. circular or linear). If the contig is circular, the prediction of the origin of replication is

performed according to cumulative GC skew and realignment of the contig from the putative origin of replication. Coding sequences (CDS) are predicted and, then, the function of these proteins is inferred based on homology using BLAST [46] or DIAMOND [47] and, optionally, using Hidden Markov Models (HMMER [48]). After that, a decision tree algorithm chooses the most reliable description of the protein (Fig. 5). Potential rRNA sequences are predicted using INFERNAL [49] with the use of the Rfam database [50], and tRNA and tmRNA sequences are predicted using ARAGORN [51]. Additionally, CRISPR, tandem and inverted repeats are predicted using PILER-CR [52], Tandem Repeats Finder [53] and Inverted Repeats Finder [54] respectively. Repeat sequences are related with the gene expression regulation, integration of the viral genome and, even, viral replication. Finally, the output of the program are a GenBank file, a FASTA file and a table (TBL) file suitable for GenBank submission (Fig. 4). Optionally, a General Feature Format (GFF) version 3 file can be generated.

Contig shape prediction. VIGA requires a FASTA file containing a single or multiple sequences of viral contigs. Before running the gene prediction, VIGA launches LASTZ [55] to predict the circularity of every contig. In this case, a contig is defined as circular when the similarity between the initial and terminal fragment of the contig (by default the first and last 101 bp) is more than 95% and the length of such alignment covers more than 40%. When the contig is predicted as a circular, the software will predict the origin of replication based on iREP [56], which predicts the origin and terminus according to the cumulative GC skew.

Gene prediction. To predict genes in the contig, its length is checked and the most suitable program is run. If a contig is larger than 100,000 bp, Prodigal [57] is executed to predict the genes. If not, MetaProdigal [58] is launched to predict the genes. In both cases, when there are linear contigs, the programs are optimised to avoid predicting genes in regions near the closed ends of the contig. After the gene prediction, the coordinates and the protein sequence are saved.

Function prediction. Protein sequences are analysed using BLASTP [59] to predict its function according to homology. By default, BLASTP is run with default parameters (except for the *e*-value, which has been changed to 10^{-5} by default). However, an exhaustive BLASTP search could be performed using very strict values (a word size of 2, a gap open of 8, a gap extend of 2, the PAM70 matrix instead of BLOSUM62 and no compositional based statistics) to accurately identify proteins [60]. Alternatively, DIAMOND [47] can be used to predict protein function according to homology. For a more accurate protein function prediction, HMMER [48] can be executed to predict functions

according to Hidden Markov Models with default parameters, except for the inclusion of an *e*-value cut-off of 0.001. To increase the protein function prediction speed, BLASTP can be launched using multiple threads and HMMER can run multiple jobs using GNU Parallel [61]. Both outputs are parsed independently according to identity, coverage, *e*-value and description to retrieve the protein function minimising the number of non-informative annotations as defined later.

Decision tree algorithm. If BLAST or DIAMOND were executed with HMMER to predict protein function, the BLAST/DIAMOND and HMMER outputs are processed using a decision tree to retrieve the description of every protein in the contig. For each protein, the existence of hits in both programs is checked. When the protein is detected in both BLAST and HMMER, non-informative annotations are detected searching for the expressions “hypothetical protein”, “uncharacterized protein”, “ORF”, “predicted protein”, “unnamed product protein” or “gp[Number]” in their BLAST and HMMER descriptions. If such a description is present in both proteins, the protein will be described as “hypothetical protein”. However if the “hypothetical protein” description is only present in BLAST, the consequent annotation retrieved by HMMER is considered as a valid one, and *vice versa*. In the scenario where the protein is not labelled as “hypothetical protein” in either BLAST or HMMER, it is checked if the percentage identity and coverage is higher in BLAST or in HMMER. Depending of these results, BLAST output or HMMER output is chosen accordingly (Fig. 5).

rRNA prediction. INFERNAL [49] is used altogether with the Rfam database [50] to predict the different ribosomal genes in every contig. In this case, INFERNAL hits are reported according to the gathering (GA) scores for every model.

tRNA prediction. ARAGORN [51] is launched to predict all tRNA and tmRNA sequences in every contig. After this step, the coordinates and the description of the tRNA are saved.

CRISPR, tandem and inverted repeats prediction. PILER-CR [52], Tandem Repeats Finder [53] and Inverted Repeats Finder [54] are used to detect CRISPR, direct tandem and inverted repeats in the contig, respectively.

Output files. After running all described steps, all saved information (contig shape, contig sequence, protein coordinates, protein sequences, protein descriptions, rRNA and tRNA coordinates, tRNA descriptions, and tandem and inverted repeats coordinates) is written to a GenBank file.

Additionally, the GenBank file is also converted to FASTA and TBL files after retrieving the metadata from a plain text file. The FASTA and the TBL files are suitable for GenBank submission. Optionally, a GFF file can also be created with this information.

Benchmarking of VIGA

Bioinformatic analysis. 138 different viruses (52 bacteriophages, 72 eukaryotic and 10 archaeal viruses, and 4 virophages) which comprises 191 sequences (Additional file 1) were used to validate VIGA. Additionally, these sequences were also submitted to Prokka [6], RAST [5] and RASTtk [7] to compare their performance with VIGA. In this case, VIGA was launched in two different ways. First, VIGA was executed using BLAST [46] and HMMER [48] to predict protein function in the VIGA-Slow mode and then, launched using only DIAMOND [47] as the VIGA-Fast mode to predict protein function. In both cases, *nr* and UniProt databases were considered for DIAMOND/BLAST and HMMER, respectively.

Statistical tests. To evaluate the performance of VIGA, three different analyses were done. Firstly, to infer the accuracy and the precision of the number of viral coding sequences, general linear models were used. All linear models were forced to have intercept zero. The slope was used to measure the accuracy, while the R^2 was used to measure the precision. Additionally, ANCOVA was used to compare the linear models. Secondly, the prediction quality of the coordinates of the viral coding sequences was evaluated by the F_1 score, the precision and sensitivity, defined as

$$F_1\,score = \frac{2 \times TP}{(2 \times TP + FP + FN)},$$

$$Precision = \frac{TP}{(TP + FP)},$$

$$Sensitivity = \frac{TP}{(TP + FN)},$$

where TP indicates the number of true positives, FP the number of false positives and FN the number of false negatives. FDR and FNR were considered to measure the type I (i.e. false coordinates were considered as true coordinates) and the type II (i.e. true coordinates were considered as false coordinates) errors, respectively. To evaluate differences in the power of prediction of all programs, Kruskal-Wallis test was performed. In case that there were differences between programs, *post-hoc* tests using Nemenyi tests were performed. All statistical tests were carried out at an alpha level of 0.05 and were performed in R v. 3.4.1 [62] using the *HH* [63] and the *PMCMR* [64] packages.

Case study: healthy human gut phageome

Bioinformatic analysis. VIGA was also tested on a metagenomic dataset using published data from the health human gut phageome [18]. This data set was downloaded from the SRA webpage (SRR codes: SRR4295172 – SRR4295175) and processed to retrieve contigs per sample. First, adapters were removed using Cutadapt 1.9.1 [65] and low-quality bases (lower than a PHRED score of 20 for a 4 bp sliding window) were trimmed using Trimmomatic [66]. All reads shorter than 30 bp were not considered for further analyses. All potential human reads were removed after being identified with Kraken v. 0.10.5 [67]. Contigs were assembled using metaSPAdes v. 3.10.0 [68] as recently the use of metaSPAdes was highly recommended to assemble metaviromes [69]. Assemblies of each sample were made non-redundant by an all-vs-all BLASTN [46] considering an *e*-value of 10^{-6}. A contig was deemed redundant when it is shared 90% of its identity over 90% of the contig length. In these cases, the longer of the two contigs was retained. Non-redundant contigs over 1,000bp were processed using VirSorter [12] to generate a final data set of viral metagenome sequences. These contigs were annotated using VIGA in the two different ways described in the 'Benchmarking of VIGA' subsection and Prokka using 10 cores. Time benchmarking was performed using the *time* command in Linux only for VIGA and Prokka, as RAST and RASTtk are online genome annotation services.

Statistical tests. To evaluate differences in the power of prediction of all programs, Kruskal-Wallis test and *post-hoc* tests using Nemenyi tests were performed as described before. Moreover, to discard the effect of the length size of contigs as a potential factor of the power of prediction, Kruskal-Wallis tests were performed after classifying the contigs in three groups: "short" (<15 kb), "medium" (15 – 70 kb), and "long" (>70 kb). All statistical tests were carried out at an alpha level of 0.05 and were performed in R v. 3.4.1 [62] using the *HH* [63] and the *PMCMR* [64] packages.

Declarations

Ethics approval and consent to participate. Not applicable.

Consent for publication. Not applicable.

Availability of data and material. Source code of VIGA (and the wrapper for the Galaxy platform) is available for download at https://github.com/EGTortuero/viga, implemented in Python 2.7, and supported on Linux, under the GPL3 licence. The program is also available as at Docker image (https://hub.docker.com/r/vimalkvn/viga/).

Competing interests. The authors declare that they have no competing interests.

Funding. This publication has emanated from research conducted with the financial support of Science Foundation Ireland (SFI) under Grant Number SFI/15/ERCD/3189. Author contributions were also made by individuals in receipt of the financial support of SFI under Grant Number SFI/12/RC/2273, a SFI's Spokes Programme which is co-funded under the European Regional Development Fund under Grant Number SFI/14/SP APC/B3032, and a research grant from Janssen Biotech, Inc.

Authors' contributions. LAD and SRS conceived the original idea. EGT and VV developed and wrote the VIGA software and the Galaxy wrapper. VV wrote the Docker integration of VIGA. EGT, LAD, SRS and CH designed the benchmark study. EGT and ANS tested VIGA against the validation benchmark. TDSS, EGT and ANS designed and run the case study. EGT, TDSS and SRS wrote the manuscript, with comments and editing by ANS, LAD, CH and RPR. All authors read and approved the final manuscript.

Acknowledgements. EGT wants to thank Dr. Aditya Upadrasta for help in improving the software and Dr. Andrei Sorin Bolocan, Dr. Adam Clooney and Dr. Feargal J. Ryan for discussions.

References

1. Miller RR, Montoya V, Gardy JL, Patrick DM, Tang P, Chambers S, et al. Metagenomics for pathogen detection in public health. Genome Med. 2013;5:81.

2. Simmonds P, Adams MJ, Benkő M, Breitbart M, Brister JR, Carstens EB, et al. Consensus statement: Virus taxonomy in the age of metagenomics. Nat. Rev. Microbiol. 2017;15:161–8.

3. Krishnamurthy SR, Wang D. Origins and challenges of viral dark matter. Virus Res. 2017;239:136–42.

4. Mitchell A, Bucchini F, Cochrane G, Denise H, Hoopen P ten, Fraser M, et al. EBI metagenomics in 2016 – an expanding and evolving resource for the analysis and archiving of metagenomic data. Nucleic Acids Res. 2016;44:D595–603.

5. Aziz RK, Bartels D, Best AA, DeJongh M, Disz T, Edwards RA, et al. The RAST Server: Rapid Annotations using Subsystems Technology. BMC Genomics 2008;9:75.

6. Seemann T. Prokka: rapid prokaryotic genome annotation. Bioinformatics 2014;30:2068–9.

7. Brettin T, Davis JJ, Disz T, Edwards RA, Gerdes S, Olsen GJ, et al. RASTtk: a modular and extensible implementation of the RAST algorithm for building custom annotation pipelines and annotating batches of genomes. Sci. Rep. 2015;5:8365.

8. McNair K, Aziz RK, Pusch GD, Overbeek R, Dutilh BE, Edwards R. Phage genome annotation using the RAST pipeline. In Clokie M, Kropinski A, Lavigne R, editors. Bacteriophages. Methods in molecular biology. New York: Humana Press; 2018. p. 231–8.

9. Bao Y, Bolotov P, Dernovoy D, Kiryutin B, Tatusova T. FLAN: A web server for influenza virus genome annotation. Nucleic Acids Res. 2007;35:W280–4.

10. Wang S, Sundaram JP, Stockwell TB. VIGOR extended to annotate genomes for additional 12 different viruses. Nucleic Acids Res. 2012;40:W186–92.

11. Pickett BE, Sadat EL, Zhang Y, Noronha JM, Squires RB, Hunt V, et al. ViPR: an open bioinformatics database and analysis resource for virology research. Nucleic Acids Res. 2012;40:D593-8.

12. Roux S, Enault F, Hurwitz BL, Sullivan MB. VirSorter: mining viral signal from microbial genomic data. PeerJ 2015;3:e985.

13. Zhao G, Wu G, Lim ES, Droit L, Krishnamurthy S, Barouch DH, et al. VirusSeeker, a computational pipeline for virus discovery and virome composition analysis. Virology 2017;503:21–30.

14. Aggarwala V, Liang G, Bushman FD. Viral communities of the human gut: metagenomic analysis of composition and dynamics. Mob. DNA 2017;8:12.

15. Besemer J, Borodovsky M. GeneMark: web software for gene finding in prokaryotes, eukaryotes and viruses. Nucleic Acids Res. 2005;33:W451-4.

16. Delcher AL, Harmon D, Kasif S, White O, Salzberg SL. Improved microbial gene identification with GLIMMER. Nucleic Acids Res. 1999;27:4636–41.

17. Devereux J, Haeberli P, Smithies O. A comprehensive set of sequence analysis programs for the VAX. Nucleic Acids Res. 1984;12:387–95.

18. Manrique P, Bolduc B, Walk ST, van der Oost J, de Vos WM, Young MJ. Healthy human gut phageome. Proc. Natl. Acad. Sci. U. S. A. 2016;113:10400–5.

19. Ren J, Ahlgren NA, Lu YY, Fuhrman JA, Sun F. VirFinder: a novel *k*-mer based tool for identifying viral sequences from assembled metagenomic data. Microbiome 2017;5:69.

20. Clokie MR, Millard AD, Letarov A V, Heaphy S. Phages in nature. Bacteriophage 2011;1:31–45.

21. Todd D, Weston JH, Mawhinney KA, Laird C. Characterization of the genome of avian encephalomyelitis virus with cloned cDNA fragments. Avian Dis. 1999;43:219–26.

22. Jiang D, Ghabrial SA. Molecular characterization of *Penicillium chrysogenum* virus: reconsideration of the taxonomy of the genus Chrysovirus. J. Gen. Virol. 2004;85:2111–21.

23. Chiba S, Salaipeth L, Lin Y-H, Sasaki A, Kanematsu S, Suzuki N. A novel bipartite double-stranded RNA Mycovirus from the white root rot Fungus *Rosellinia necatrix*: molecular and

biological characterization, taxonomic considerations, and potential for biological control. J. Virol. 2009;83:12801–12.

24. Kramer T, Greco TM, Enquist LW, Cristea IM. Proteomic characterization of pseudorabies virus extracellular virions. J. Virol. 2011;85:6427–41.

25. Lété C, Palmeira L, Leroy B, Mast J, Machiels B, Wattiez R, et al. Proteomic characterization of bovine herpesvirus 4 extracellular virions. J. Virol. 2012;86:11567–80.

26. Chan Y-W, Millard AD, Wheatley PJ, Holmes AB, Mohr R, Whitworth AL, et al. Genomic and proteomic characterization of two novel siphovirus infecting the sedentary facultative epibiont cyanobacterium *Acaryochloris marina*. Environ. Microbiol. 2015;17:4239–52.

27. Josset L, Zeng H, Kelly SM, Tumpey TM, Katze MG. Transcriptomic characterization of the novel avian-origin influenza A (H7N9) virus: specific host response and responses intermediate between avian (H5N1 and H7N7) and human (H3N2) viruses and implications for treatment options. MBio 2014;5:e01102-13.

28. Sun X, Wang Z, Gu Q, Li H, Han W, Shi Y. Transcriptome analysis of *Cucumis sativus* infected by Cucurbit chlorotic yellows virus. Virol. J. 2017;14:18.

29. Tombácz D, Balázs Z, Csabai Z, Moldován N, Szűcs A, Sharon D, et al. Characterization of the dynamic transcriptome of a herpesvirus with long-read single molecule real-time sequencing. Sci. Rep. 2017;7:43751.

30. Kent WJ. BLAT—The BLAST-Like Alignment Tool. Genome Res. 2002;12:656–64.

31. Aziz RK, Ackermann H-W, Petty NK, Kropinski AM. Essential steps in characterizing bacteriophages: Biology, taxonomy, and genome analysis. In Clokie M, Kropinski A, Lavigne R, editors. Bacteriophages. Methods in molecular biology. New York: Humana Press; 2018. p. 197–215.

32. Chain BM, Myers R. Variability and conservation in hepatitis B virus core protein. BMC Microbiol. 2005;5:33.

33. Khan AM, Miotto O, Nascimento EJM, Srinivasan KN, Heiny AT, Zhang GL, et al. Conservation and variability of dengue virus proteins: Implications for vaccine design. PLoS Negl. Trop. Dis. 2008;2:e272.

34. Babar MM, Zaidi N-SS. Protein sequence conservation and stable molecular evolution reveals influenza virus nucleoprotein as a universal druggable target. Infect. Genet. Evol. 2015;34:200–10.

35. Kuchibhatla DB, Sherman WA, Chung BYW, Cook S, Schneider G, Eisenhaber B, et al. Powerful sequence similarity search methods and in-depth manual analyses can identify remote homologs in many apparently “orphan” viral proteins. J. Virol. 2014;88:10–20.

36. Jones P, Binns D, Chang HY, Fraser M, Li W, McAnulla C, et al. InterProScan 5: Genome-scale protein function classification. Bioinformatics 2014;30:1236–40.

468 37. Seguritan V, Alves N, Arnoult M, Raymond A, Lorimer D, Burgin AB, et al. Artificial neural
469 networks trained to detect viral and phage structural proteins. PLoS Comput. Biol.
470 2012;8:e1002657.

471 38. Galiez C, Magnan CN, Coste F, Baldi P. VIRALpro: a tool to identify viral capsid and tail
472 sequences. Bioinformatics 2016;32:1405–7.

473 39. Klimke W, O'Donovan C, White O, Brister JR, Clark K, Fedorov B, et al. Solving the problem:
474 Genome annotation standards before the data deluge. Stand. Genomic Sci. 2011;5:168–93.

475 40. Tripp HJ, Sutton G, White O, Wortman J, Pati A, Mikhailova N, et al. Toward a standard in
476 structural genome annotation for prokaryotes. Stand. Genomic Sci. 2015;10:45.

477 41. McDonald AG, Tipton KF. Fifty-five years of enzyme classification: advances and difficulties.
478 FEBS J. 2014;281:583–92.

479 42. Tatusov RL, Galperin MY, Natale DA, Koonin E V. The COG database: a tool for genome-scale
480 analysis of protein functions and evolution. Nucleic Acids Res. 2000;28:33–6.

481 43. Kanehisa M, Sato Y, Kawashima M, Furumichi M, Tanabe M. KEGG as a reference resource
482 for gene and protein annotation. Nucleic Acids Res. 2016;44:D457–62.

483 44. Grazziotin AL, Koonin E V, Kristensen DM. Prokaryotic Virus Orthologous Groups (pVOGs): a
484 resource for comparative genomics and protein family annotation. Nucleic Acids Res.
485 2017;45:D491–8.

486 45. Cock PJA, Antao T, Chang JT, Chapman BA, Cox CJ, Dalke A, et al. Biopython: Freely
487 available Python tools for computational molecular biology and bioinformatics. Bioinformatics
488 2009;25:1422–3.

489 46. Camacho C, Coulouris G, Avagyan V, Ma N, Papadopoulos J, Bealer K, et al. BLAST+:
490 architecture and applications. BMC Bioinformatics. 2009;10:421.

491 47. Buchfink B, Xie C, Huson DH. Fast and sensitive protein alignment using DIAMOND. Nat.
492 Methods 2014;12:59–60.

493 48. Finn RD, Clements J, Eddy SR. HMMER web server: Interactive sequence similarity searching.
494 Nucleic Acids Res. 2011;39:W29–W37.

495 49. Nawrocki EP, Eddy SR. Infernal 1.1: 100-fold faster RNA homology searches. Bioinformatics
496 2013;29:2933–5.

497 50. Nawrocki EP, Burge SW, Bateman A, Daub J, Eberhardt RY, Eddy SR, et al. Rfam 12.0: updates
498 to the RNA families database. Nucleic Acids Res. 2015;43:D130–7.

499 51. Laslett D, Canback B. ARAGORN, a program to detect tRNA genes and tmRNA genes in
500 nucleotide sequences. Nucleic Acids Res. 2004;32:11–6.

501 52. Edgar RC. PILER-CR: fast and accurate identification of CRISPR repeats. BMC Bioinformatics
502 2007;8:18.

503 53. Benson G. Tandem repeats finder: a program to analyze DNA sequences. Nucleic Acids Res.
504 1999;27:573–80.

505 54. Warburton PE, Giordano J, Cheung F, Gelfand Y, Benson G. Inverted repeat structure of the
506 human genome: the X-chromosome contains a preponderance of large, highly homologous inverted
507 repeats that contain testes genes. Genome Res. 2004;14:1861–9.

508 55. Harris RS. Improved pairwise alignment of genomic DNA. The Pennsylvania State University;
509 2007.

510 56. Brown CT, Olm MR, Thomas BC, Banfield JF. Measurement of bacterial replication rates in
511 microbial communities. Nat. Biotechnol. 2016;34:1256–63.

512 57. Hyatt D, Chen G-L, Locascio PF, Land ML, Larimer FW, Hauser LJ. Prodigal: prokaryotic gene
513 recognition and translation initiation site identification. BMC Bioinformatics 2010;11:119.

514 58. Hyatt D, Locascio PF, Hauser LJ, Uberbacher EC. Gene and translation initiation site prediction
515 in metagenomic sequences. Bioinformatics. 2012;28:2223–30.

516 59. Altschul SF, Madden TL, Schäffer AA, Zhang J, Zhang Z, Miller W, et al. Gapped BLAST and
517 PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res.
518 1997;25:3389–402.

519 60. Fozo EM, Makarova KS, Shabalina SA, Yutin N, Koonin E V, Storz G. Abundance of type I
520 toxin-antitoxin systems in bacteria: searches for new candidates and discovery of novel families.
521 Nucleic Acids Res. 2010;38:3743–59.

522 61. Tange O. GNU Parallel – The Command-Line Power Tool. ;login USENIX Mag. 2011;36:42–7.

523 62. R Core Team. R: A Language and Environment for Statistical Computing R Found. Stat.
524 Comput. Vienna, Austria: R Foundation for Statistical Computing; 2015. http://www.r-project.org

525 63. Heiberger RM, Holland B. Statistical analysis and data display: An intermediate course with
526 examples in R. 2nd Ed. New York: Springer New York; 2015.

527 64. Pohlert T. PMCMR: Calculate Pairwise Multiple Comparisons of Mean Rank Sums. 2015.
528 http://cran.r-project.org/package=PMCMR

529 65. Martin M. Cutadapt removes adapter sequences from high-throughput sequencing reads.
530 EMBnet.journal 2011;17:10.

531 66. Bolger AM, Lohse M, Usadel B. Trimmomatic: a flexible trimmer for Illumina sequence data.
532 Bioinformatics 2014;30:2114–20.

533 67. Wood DE, Salzberg SL. Kraken: ultrafast metagenomic sequence classification using exact
534 alignments. Genome Biol. 2014;15:R46.

535 68. Nurk S, Meleshko D, Korobeynikov A, Pevzner PA. metaSPAdes: a new versatile metagenomic
536 assembler. Genome Res. 2017;27:824–34.

537 69. Roux S, Emerson JB, Eloe-Fadrosh EA, Sullivan MB. Benchmarking viromics: an *in silico*
538 evaluation of metagenome-enabled estimates of viral community composition and diversity. PeerJ
539 2017;5:e3817.

Tables

Table 1. Accuracy and precision in the number of coding sequences

Case	Program	Accuracy (Slope)	Precision (R^2)
General	*VIGA*	1.027	0.997
	Prokka	1.043	0.996
	RAST	1.118	0.979
	RASTtk	1.135	0.982
Archaeal viruses	*VIGA*	0.962	0.990
	Prokka	0.991	0.991
	RAST	0.821	0.936
	RASTtk	1.036	0.993
Bacteriophages	*VIGA*	0.997	0.997
	Prokka	0.983	0.995
	RAST	0.982	0.996
	RASTtk	1.015	0.997
Eukaryotic viruses	*VIGA*	1.031	0.997
	Prokka	1.050	0.997
	RAST	1.136	0.979
	RASTtk	1.151	0.982

Table 2. Accuracy, precision and sensitivity of the different programs. False Discovery Rate (FDR) and False Negative Ratio (FNR) are used to describe errors in the precision and sensitivity.

Case	Program	F1 Score	Precision	Sensitivity	FDR (Type I error)	FNR (Type II error)
General	*VIGA*	0.945	0.940	0.950	0.060	0.050
	Prokka	0.924	0.917	0.931	0.083	0.069
	RAST	0.853	0.844	0.862	0.156	0.138
	RASTtk	0.863	0.821	0.909	0.179	0.091
Archaeal viruses	*VIGA*	0.914	0.930	0.899	0.070	0.101
	Prokka	0.921	0.922	0.920	0.078	0.080
	RAST	0.819	0.918	0.739	0.082	0.261
	RASTtk	0.910	0.894	0.927	0.106	0.073
Bacteriophages	*VIGA*	0.952	0.958	0.947	0.042	0.053
	Prokka	0.909	0.921	0.897	0.079	0.103
	RAST	0.936	0.950	0.923	0.050	0.077
	RASTtk	0.934	0.929	0.939	0.071	0.061
Eukaryotic viruses	*VIGA*	0.942	0.930	0.954	0.070	0.046
	Prokka	0.933	0.914	0.952	0.086	0.048
	RAST	0.806	0.782	0.831	0.218	0.169
	RASTtk	0.820	0.760	0.889	0.240	0.111

Table 3. Kruskal-Wallis *p*-values for the comparison between all different pipelines considering the different viral types.

Case	*p*
Archaeal viruses	8.219×10^{-5}
Bacteriophages	5.596×10^{-28}
Eukaryotic viruses	1.348×10^{-46}

Figure legends

Figure 1. Correlation between the expected and observed number of coding sequences when considering (A) all known viral sequences, (B) archaeal viruses, (C) bacteriophages, and (D) eukaryotic viruses. Dotted line is a 1:1 line.

Figure 2. Percentage of non-informative annotations when processed in all programs for (A) all known viral sequences, (B) archaeal viruses, (C) bacteriophages, and (D) eukaryotic viruses. Dot indicates the average value of non-informative annotations and bars indicates the 95% confidence interval.

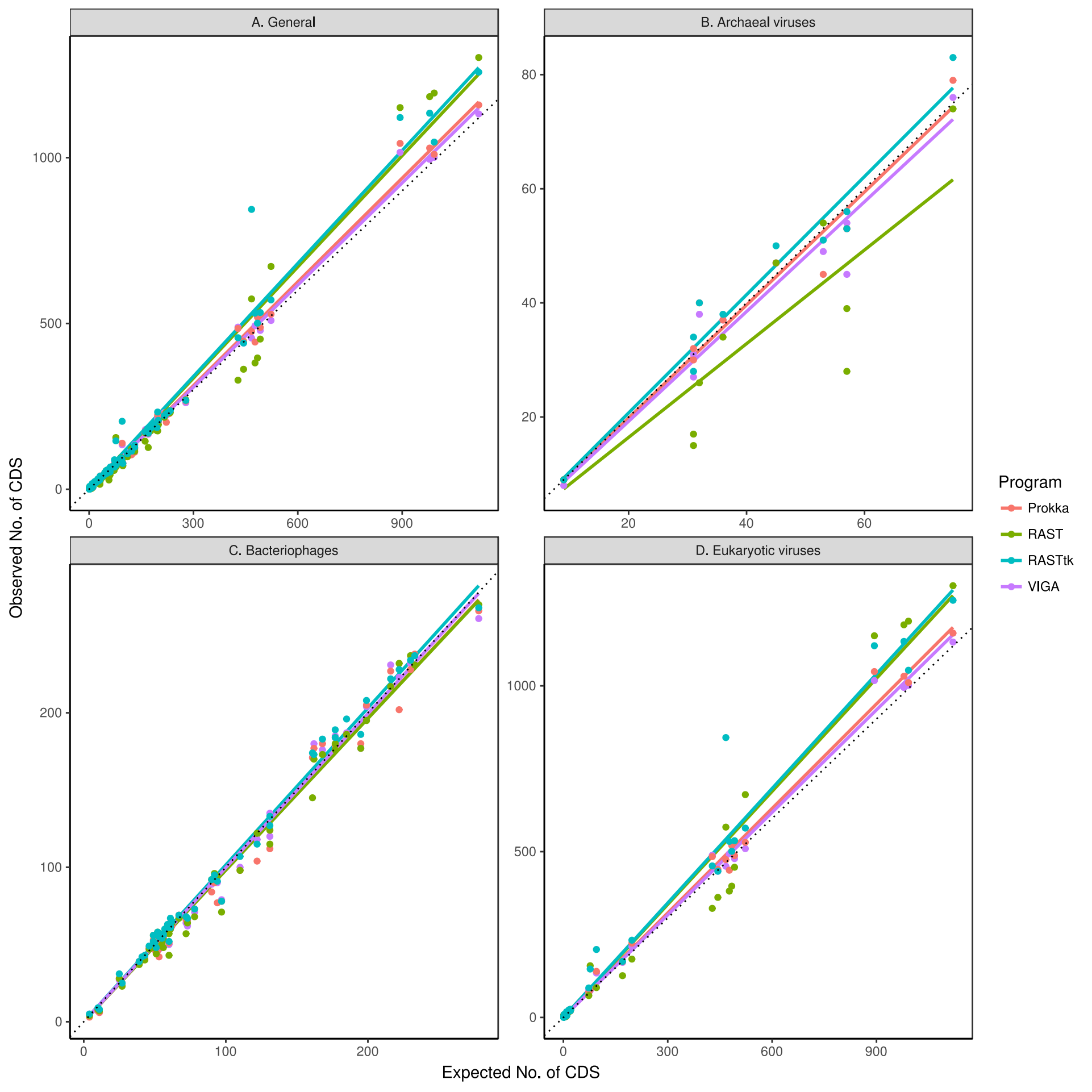
Figure 3. Percentage of non-informative annotations for the case study dataset when processed in all programs for (A) the case study dataset, (B) short contigs (<15 kb), (C) medium contigs (15 – 70 kb), and (D) long contigs (>70 kb). Dot indicates the average value of non-informative annotations and bars indicates the 95% confidence interval.

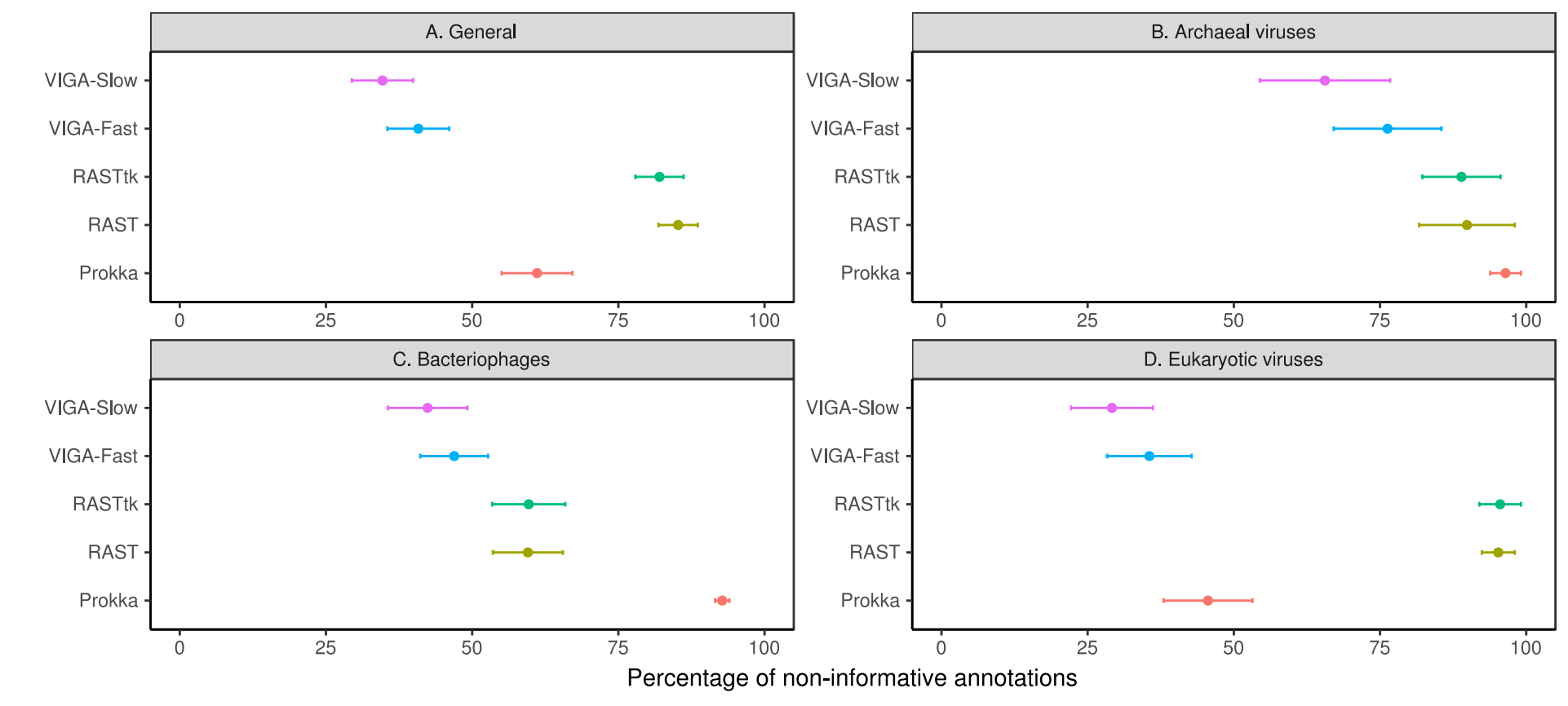
Figure 4. Flowchart of the VIGA pipeline. Orange rectangles represent the different steps of the program (among those, discontinuous-lined rectangles indicate optional steps; see main text). Red parallelograms indicate the relevant data that it is summarised in the output. Yellow rectangles with a wavy base stand for input and output files.

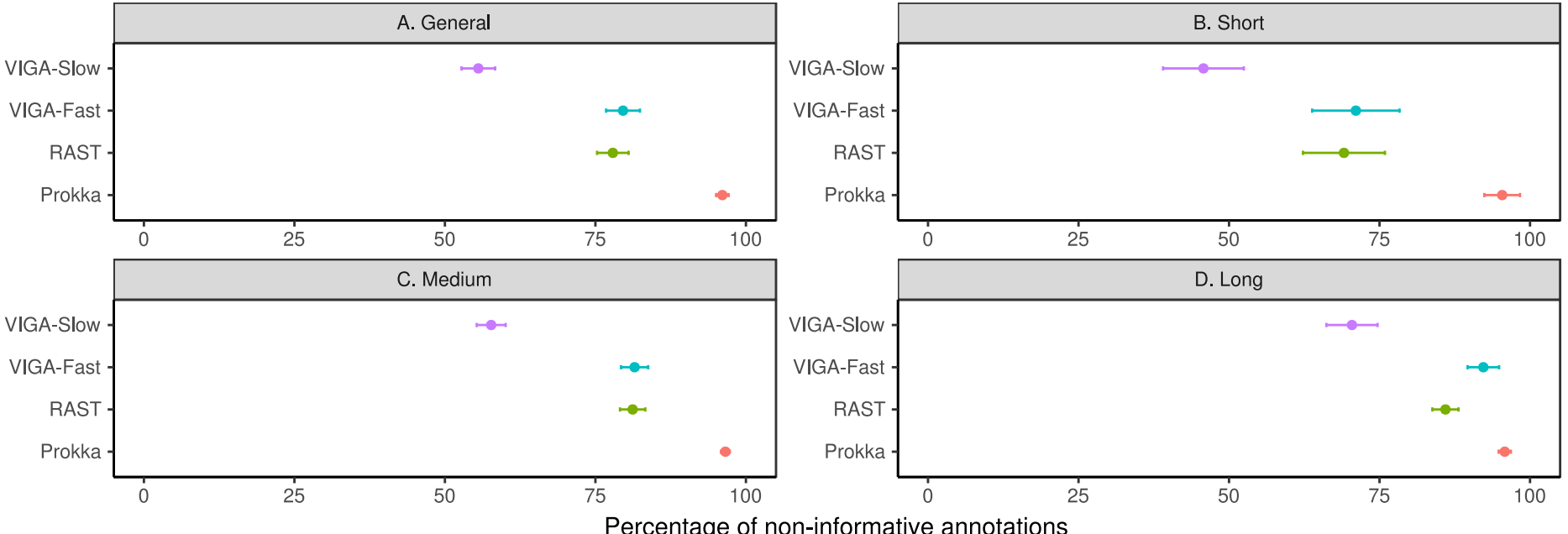
Figure 5. Flowchart of the decision tree algorithm. Blue rectangles represent steps in the decision tree. Orange and purple rectangles state optimal BLAST and HMMER solutions, respectively. Mustard coloured rectangles represent “hypothetical protein” decisions.

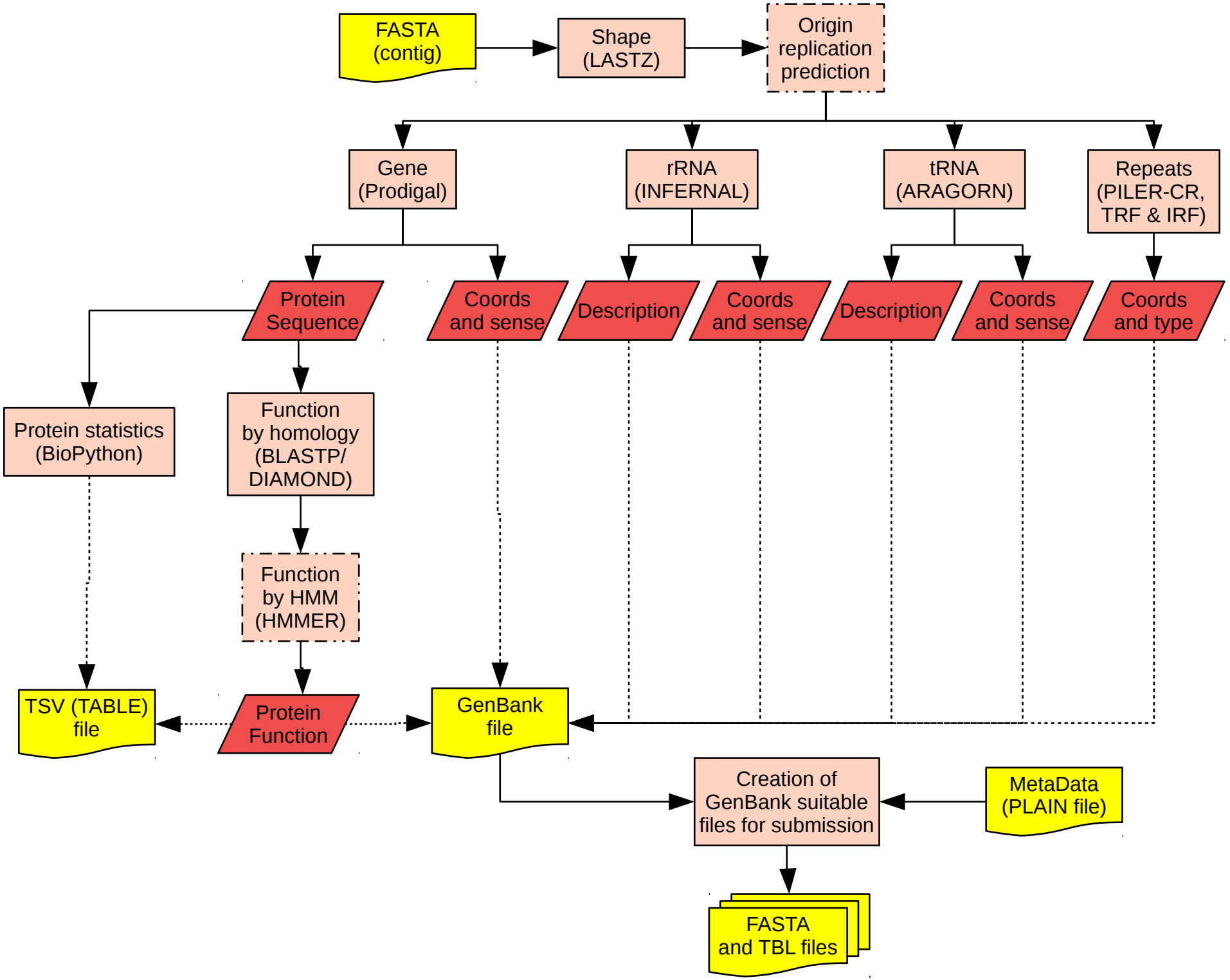
Additional files

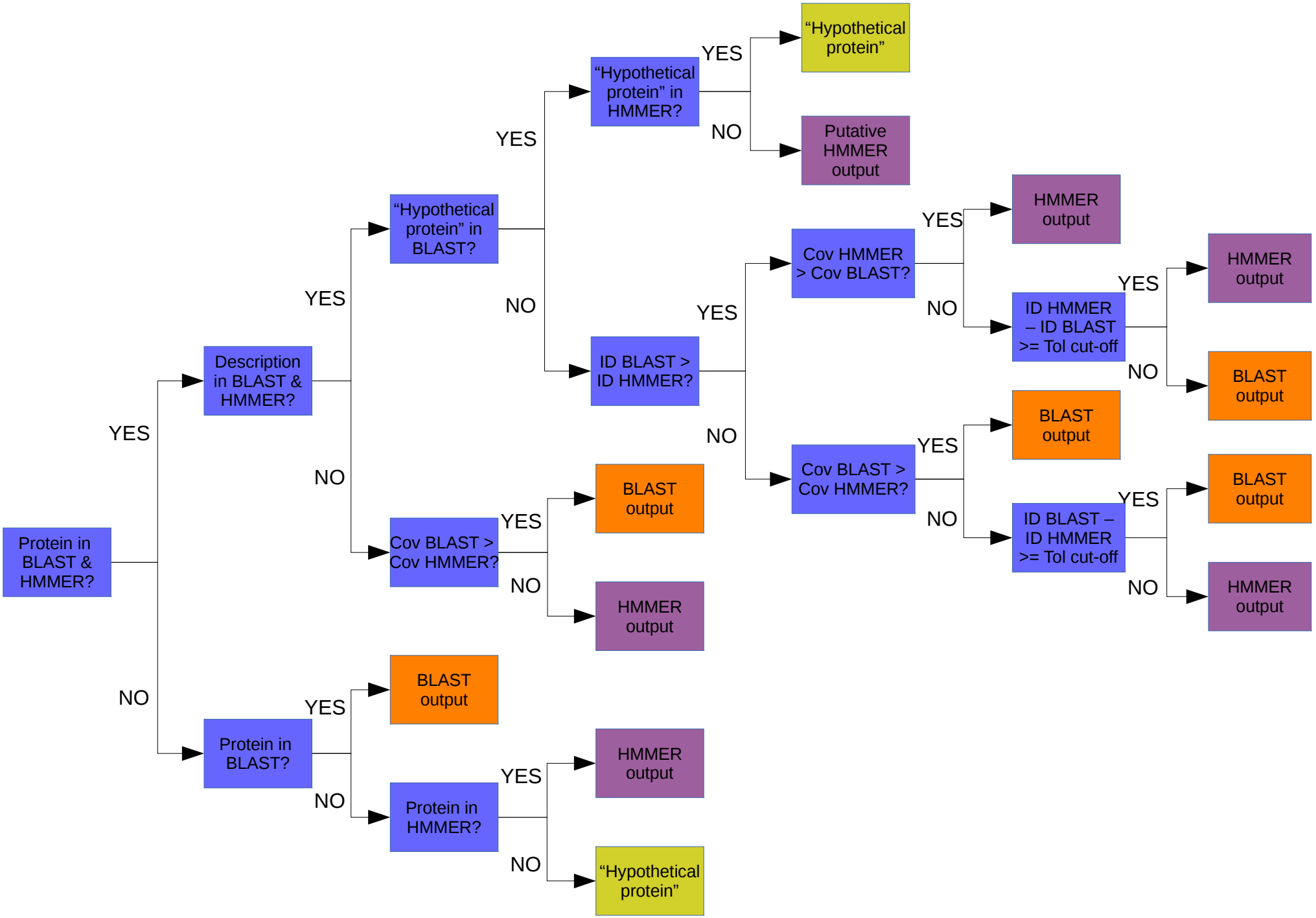
Additional file 1. List of the viruses used for the validation test (Excel file)


A. General
B. Archaeal viruses
C. Bacteriophages
D. Eukaryotic viruses
Observed No. of CDS
Expected No. of CDS
Program
Prokka
RAST
RASTtk
VIGA
0
300
600
900
500
1000
20
40
60
80
100
200


A. General
B. Archaeal viruses
C. Bacteriophages
D. Eukaryotic viruses
VIGA-Slow
VIGA-Fast
RASTtk
RAST
Prokka
0
25
50
75
100
Percentage of non-informative annotations


A. General
B. Short
C. Medium
D. Long
VIGA-Slow
VIGA-Fast
RAST
Prokka
0
25
50
75
100
Percentage of non-informative annotations


FASTA (contig)
Shape (LASTZ)
Origin replication prediction
Gene (Prodigal)
rRNA (INFERNAL)
tRNA (ARAGORN)
Repeats (PILER-CR, TRF & IRF)
Protein Sequence
Coords and sense
Description
Coords and sense
Description
Coords and sense
Coords and type
Protein statistics (BioPython)
Function by homology (BLASTP/ DIAMOND)
Function by HMM (HMMER)
TSV (TABLE) file
Protein Function
GenBank file
Creation of GenBank suitable files for submission
MetaData (PLAIN file)
FASTA and TBL files


Protein in BLAST & HMMER?
YES
Description in BLAST & HMMER?
YES
"Hypothetical protein" in BLAST?
YES
"Hypothetical protein" in HMMER?
YES
"Hypothetical protein"
NO
Putative HMMER output
NO
ID BLAST > ID HMMER?
YES
Cov HMMER > Cov BLAST?
YES
HMMER output
NO
ID HMMER – ID BLAST >= Tol cut-off
YES
HMMER output
NO
BLAST output
NO
Cov BLAST > Cov HMMER?
YES
BLAST output
NO
ID BLAST – ID HMMER >= Tol cut-off
YES
BLAST output
NO
HMMER output
NO
Cov BLAST > Cov HMMER?
YES
BLAST output
NO
HMMER output
NO
Protein in BLAST?
YES
BLAST output
NO
Protein in HMMER?
YES
HMMER output
NO
"Hypothetical protein"