Development of workflow for picornavirus genome sequence analysis

Mrunalini Lotankar
Master's Thesis
Master's Degree Programme in Digital Health and Life Sciences
Department of Future Technologies
University of Turku
June 2020

ABSTRACT

UNIVERSITY OF TURKU
Department of Future Technologies/Faculty of Science and Engineering

LOTANKAR, MRUNALINI: Development of workflow for picornavirus genome sequence analysis

Master’s Thesis, 53 p, 14 p. appendices
Bioinformatics
June 2020

Picornaviruses are small, non-enveloped, icosahedral, positive stranded RNA viruses and among the most common human pathogens. Some of the clinically important genera for humans are *Enterovirus*, *Hepatovirus*, *Parechovirus* and *Cardiovirus*. The symptoms for the picornaviral infections range from mild, asymptomatic to fatal disease. Threats posed to human health by these viruses is observed in the constant outbreaks of enteroviruses and parechoviruses in the different parts of the world. Next generation sequencing provides an efficient way to detect and identify known or novel micro-organisms. Advantages of NGS are rapid sequencing methods, high-throughput process and affordable costs. On the other hand, NGS also requires advanced technical and computational skills, and creates a bottleneck owing to necessity of standardization of bioinformatic tools. It is therefore imperative to optimize and determine parameters, which provide accuracy in every stage of NGS workflow.

The aim of this thesis was to develop a rapid and straightforward, user-friendly workflow for the assembly and analysis of picornaviral genomes. Chipster platform was chosen as the primary test platform. The workflow involved use of automated analysis pipelines (VirusDetect and A5 assembly pipeline), and alternative approaches, which included pre-processing of raw data, and reference-mapping or *de novo* assembly (Velvet and SPAdes) of picornavirus sequences. Except for *de novo* assembly and validation and quality assessment of final outputs, all steps were performed in Chipster. Of these approaches, VirusDetect and reference-mapping were not successful. A5 pipeline for microbial genome assembly was found to be very suited for picornavirus identification. Velvet and SPAdes also performed well, but Velvet assembler was found to more computationally exhaustive and time consuming. Quality assessment suggested that performance of SPAdes was relatively better than the performance of A5 or Velvet. As A5 pipeline does not require any parameter settings, it can be used as initial identification and contig/scaffold generation method for picornaviral sequences. Together with implementation of *de novo* assembler(s) on Chipster platform a novel, user-friendly NGS workflow for picornavirus sequence assembly can be established.

Keywords
Picornaviridae, next generation sequencing, *de novo* assembly, Chipster, A5 microbial genome assembly pipeline, Velvet, SPAdes

Table of contents

List of Figures

List of Tables

List of Tables for appendices

List of abbreviations

A5	Andrew And Aaron's Awesome Assembly pipeline
BAM	Binary alignment/Map
BLAST	Basic Local Alignment Search Tool
BWA	Burrows wheeler transform
BWA-MEM	Burrows wheeler transform-maximal exact matches
cDNA	Complementary DNA
CSC	Centre for Scientific Computing
DNA	Deoxyribonucleic acid
dNTPs	Deoxyribonucleotide phosphates
EM	Electron microscope
GC	Guanine-cytosine
HIV	Human immunodeficiency virus
HTS	High-throughput sequencing
IDBA-UD	Iterative de Bruijn Graph *de Novo* Assembler for Short Reads Sequencing data with Highly Uneven Sequencing Depth
IRES	Internal ribosome entry site
NCBI	National Center for Biotechnology Information
NGS	Next genereation sequencing
OLC	Overlap layout consensus
ORF	Open reading frame

PCR	Polymerase chain reaction
QC	Quality control
QUAST	quality assessment tool
RdRp	RNA-dependent RNA polymerase enzyme
RNA	Ribonucleic acid
RT-qPCR	Reverse transcriptase quantitative polymerase chain reaction
SAM	Sequence alignment/Map
SBS	sequencing by synthesis
SIB	Swiss Institute of Bioinformatics
SNP	Single nucleotide polymorphism
SPAdes	St. Petersburg genome assembler
UTR	Untranslated regions
WHO	World health organization
VP1-4	Viral protein 1-4

“Be ready to revise any system, scrap any method, abandon any theory, if the success of the job requires it.”

— Henry Ford

1 Introduction

The family *Picornaviridae* (Figure 1.1 below) in the order *Picornavirales* comprises of 63 genera with 147 species (http://www.picornaviridae.com/, March 2020). Picornaviruses are small, non-enveloped, icosahedral, positive-stranded RNA viruses (Zell, 2018). They play a vital role as viral pathogens in humans and animals. While most picornavirus infections are asymptomatic or cause mild illness, some virus types cause serious infections of central nervous system, respiratory and gastrointestinal tract, skeletal muscle system, heart, liver and eyes (Yin-Murphy & Almond, 1996; Zell, 2018).

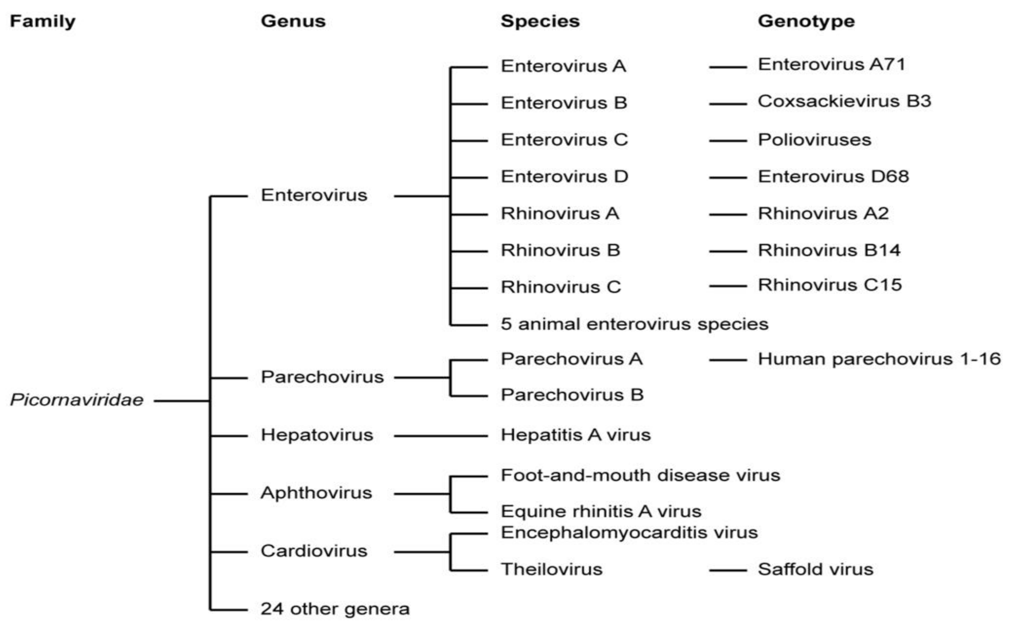


Figure 1.1 Classification of *Picornaviridae* family
Depiction of clinically important genera, with selected examples of species and genotypes (van der Linden et al., 2015) (Note: the numbers in the figure are according to the classification until 2015).

In addition, picornavirus family constitutes of etiological agents, which have global prevalence, exhibit a wide range of illnesses and play pivotal role in human health impact (Table 1.1 below). In the lieu of poliovirus eradication, the circulation of non-polio picornaviruses poses a possible threat to the human health. This is observed in the constant outbreaks of enteroviruses and parechoviruses in the different parts of the world (Wolthers et al., 2019).

Table 1.1 Diseases associated with human picornaviruses
adapted from (Santti et al., 1999). Nomenclature for the viruses as indicated in brackets as per (Simmonds et al., 2020).

Disease	**Virus**
Poliomyelitis	Polioviruses (PV-1-3)
Paralytic disease	Coxsackievirus A7 (CVA7), enterovirus 70 and 71 (EV-D70, EV-A71)
Meningitis/encephalitis	Several enterovirus serotypes (EV)
Myocarditis	Coxsackie B viruses (CVB)
Neonatal infections	Coxsackie B viruses (CVB), echoviruses (EV)
Pleurodynia	Coxsackie B viruses (CVB)
Herpangina	Coxsackie A viruses (CVA)
Hand-foot-and-mouth disease	Coxsackie A16 (CVA16), enterovirus 71 (EV-A71)
Acute haemorrhagic conjunctivitis	Coxsackie A24 (CVA24) and enterovirus 70 (EV-D70)
Respiratory infections	Many enteroviruses (EV), parechoviruses (PeV)
Common cold	Rhinoviruses (RV)
Gastroenteritis	Parechovirus 1(PeV-A1)
Acute hepatitis	Hepatitis A virus

1.1 Picornavirus virion

Picornavirus virion comprises icosahedral, non-enveloped and small (diameter 22-30 nm) protein particle, which encloses single-stranded RNA genome. Icosahedral particle symmetry is a characteristic feature of the members within picornavirus family. Viral capsid is densely packed with 60 protomers. Each protomer consists of external VP1, VP2, VP3 surface proteins and internal VP4 protein (Figure 1.2). Successful transmission and tropism of virus is largely dependent on capsid structure. Capsid helps in host cell recognition, virus attachment and release of viral RNA into the host cells. Additionally, capsid also aids the virus to evade host immune system (Cifuente & Moratorio, 2019; Yin-Murphy & Almond, 1996; Zell, 2018; ViralZone, SIB Swiss Institute of Bioinformatics, 2018).

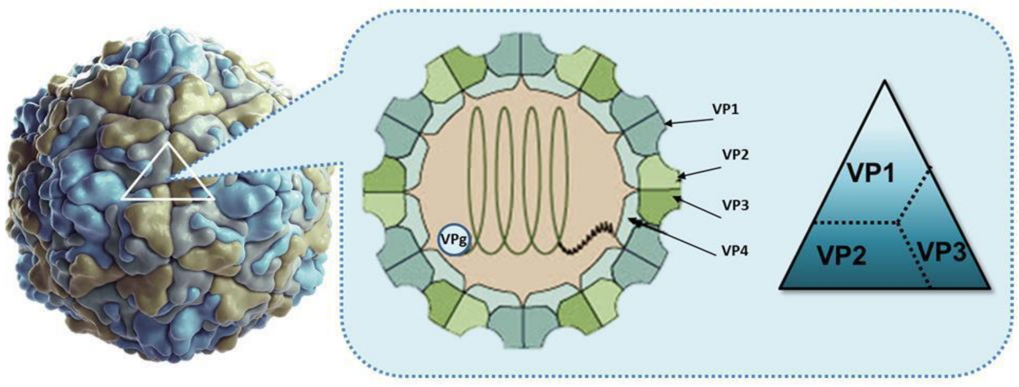


Figure 1.2 Structure of picornavirus virion
Viral RNA genome is enclosed in a protein capsid. Capsid is composed of four structural proteins: VP1, VP2, VP3 and VP4. Icosahedral symmetry of a capsid is a characteristic feature within picornavirus family (Source: https://www.creative-biolabs.com/vaccine/vaccines-for-virus-from-picornaviridae-family.htm).

Viral genome is monopartite (that is, all viral genes reside in a single molecule), positive-stranded and linear RNA molecule of 7.1-8.9 kb in size. Genome has covalently bound VPg (viral protein, genome linked; a non-capsid protein) in the 5’ end and poly-A tail in the 3’ end. Genomic RNA is infectious and functions similarly to mammalian mRNA. Generally, the genome organisation pattern is common across the picornaviruses (Figure 1.3). A single ORF (open reading frame) encodes a polyprotein with UTRs (untranslated regions) in 5’ and 3’ ends. An internal ribosome entry site (IRES), which resides in the 5’ UTR region, leads the translation of the polyprotein. Viral polyprotein is divided into three regions, P1, P2 and P3, which encode structural proteins and non-structural proteins essential in genome replication. Short 3’ UTR plays a role in the synthesis of negative-strand. A leader protein (L) is encoded in some genera (Boros et al., 2012; Zell, 2018; ViralZone Expasy, SIB Swiss Institute of Bioinformatics, 2018).

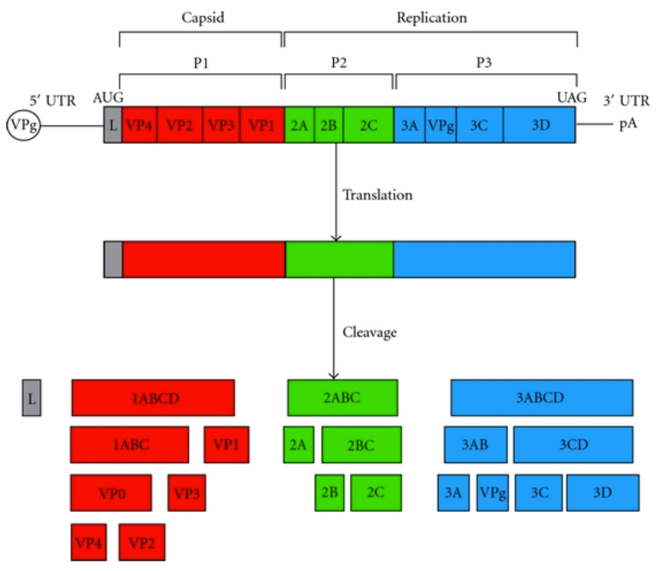


Figure 1.3 Genome organization of picornavirus
The viral mRNA is translated into a single polyprotein, which is then cleaved into functional proteins (Source: Kerkvliet et al., 2010).

1.2 Picornavirus identification, typing and genome-sequencing

Picornaviruses are typed based on the capsid protein VP4/2 and VP1 sequences, which are the most variable gene regions and define the immunogenicity of the virus types (www.picornaviridae.com). Typing is essential in monitoring and controlling the emerging epidemics caused by specific picornavirus types and in determining changes in their pathogenesis and disease-causing potential. However, typing regions are fairly short and therefore, there is a lack of genetic and evolutionary information regarding the overall changes in virus genome. The changes in virus genome are results of high mutation rates due to low fidelity of RNA-dependent RNA polymerase enzyme (RdRp). These mutations, along with recombination (that is, two different viral strains infect same host cell and give rise to a new strain) are responsible for genetic diversification of picornaviruses. Next generation sequencing (NGS) offers an avenue to identify and characterise viral diversification (Joffret et al., 2018; Posada-Cespedes et al., 2017). While genome sequencing helps in virus typing, it also contributes to evolutionary analysis and determination of pathogenic variants. In addition, full length or near-full length genomic sequence information is also useful in generating cDNA viral clones, which are used in the analyses of viral functions and in the development of viral vectors for gene and oncolytic virotherapy. This approach of NGS has been successfully reported in studies of poliovirus, enterovirus A71 and enterovirus species C (Bessaud et al., 2016; Montmayeur et al., 2017; Sahoo et al., 2017, Tan et al., 2015).

1.3 Limitations of traditional virus identification methods

One of the hallmarks of disease diagnostics is the ability to identify the causal agent(s). Conventional methods in virus identification include electron microscopy (EM), culture-based methods (virus culture) and detection of antibodies against viruses with serological testing (serology). Virus culture accompanied with morphological changes in cells, i.e. cytopathic effect caused by viruses, enables enrichment of virus particles in clinical sample for use in further genetic analysis. Previously, viruses were sequenced using overlapping PCR-amplified viral fragments and Sanger sequencing. However, these methods are time consuming and tedious, require experienced personnel and have a risk of failure. Sanger sequencing results in a single sequence, which is complicated if the primers bind and amplify several target sequences. Clinical isolates often fail to grow in cell lines, and if they do, continuous virus cultivation is likely to increase the risk of changes in the viral genomes, which may affect the integrity of the sequence and data interpretation (Chun et al., 2018; Datta, 2015). To overcome these shortcomings, metagenomic approaches, which use the means of next generation sequencing (NGS), have been developed (Chun et al., 2018; Lim & Brown, 2018).

1.4 Next generation sequencing in virology

The advent of NGS or high-throughput sequencing technology (HTS) has revolutionized the field of viral genomics enabling rapid and economical sequencing and further assembly of vast number of viral genomes. HTS plays crucial role in answering biological questions related to viruses such as, intra-host diversity and nature of quasispecies, virus transmission and tropism, antiviral resistance and vaccine evasion. High rate of evolution, short generation time and low fidelity of polymerases contribute to high viral diversity. Genomic changes also allow viruses to escape from host immune system and make them resistant to antiviral therapy. For example, WHO has established global strategies to monitor human immunodeficiency virus (HIV) drug resistance and the Global Influenza Surveillance and Response System to monitor, prevent and assess the viral evolution regarding antiviral-therapy (Radford et al., 2012).

Culture-independent methods to characterize genomic information directly from the sample, allow unbiased analysis of the viruses. They have key roles in virus discovery (influenza virus, Greninger et al., 2010; Schmallemberg virus of ruminants, Hoffmann et al., 2012; Day et al., 2010) and in understanding virus ecology in a wide range of environmental habitats (faeces : Donaldson et al., 2010, Reyes et al., 2010; sewage : Cantalupo et al., 2011; water : Lopez-Bueno et al., 2009; Rodriguez-Brito et al., 2010; vaccines : Victoria et al., 2010). Measurement of relative mRNA expression levels by NGS, i.e. RNA-seq, provides insight into differences in viral genome expression (Epstein- Barr virus: Lin et al., 2010; Cytomegalovirus: Gatherer et al., 2011) in healthy and diseased individuals (Orton et al., 2016; Radford et al., 2012).

1.5 NGS sequencing platforms

Variety of NGS sequencing technologies are available in the market, which have different amplification and sequencing methodologies. These NGS platforms differ in costs, capacities (read lengths, run time and errors), chemistries and applications. Although modifications and advancements are observed on routine basis in these technologies, the core principle remains the same (Metzker, 2010). The current sequencing platforms include Illumina (Illumina), PacBio (Pacific Biosciences), Ion Torrent (Life Technologies) and MinIon (Oxford Nanopore Technologies). Illumina offers versatile platforms like MiSeq, Hiseq, GAIIx, MiSeqDx, NextSeq, NovaSeq, MiniSeq, and iSeq, which cater the needs regarding capacities and costs. Ion Torrent/Ion S5 platform is also affordable and easy to use, albeit have higher error rate than Illumina. Pacific Biosciences (PacBio) with its two platforms, PacBioRS/RSII and the Sequel offer high throughput with long reads (average read length 10kb). MinIon (compact sized single molecule sequencer) streams data in real time so that analysis can be performed during the experiment. MinIon workflows are fully versatile and include PromethION and GridION platforms (high-throughput platforms for parallel sequencing with stacked multiple flow cells) (Maljkovic Berry et al., 2020). The methodology regarding each platform and comparative studies regarding their performances for different genomes are described in literature elsewhere (Datta, 2015; Hodzic et al., 2017; Kulski, 2016; Liu et al., 2012; Metzker, 2010; Pereira et al., 2020; Radford et al., 2012). The choice of the platform depends on the experimental needs. Considerations should be given to the genome size, GC content and depth, and sequencing coverage. Illumina is found to be consistent in its leading application amongst NGS market (J & G, 2016; Bentely et al., 2008;

https://www.genengnews.com/a-lists/top-10-sequencing-companies-2/, 2018), and is thus commonly used in viral metagenomics.

1.6 Illumina sequencing platform

Illumina is one of the most preferred sequencing platforms in viral genomic studies (Goya et al., 2018; Huang et al., 2019; Radford et al., 2012) (Virus detection and research reviews, Illumina, 2013). The services, the basic principles and the workflows with respective tutorials can be found on the Illumina online site (https://emea.illumina.com/science/technology/next-generation-sequencing.html). The sequences used in this thesis were from the samples sequenced with Illumina HiSeq 3000 instrument. The Illumina workflow is described in Figure 1.4 (Adapted from Illumina introduction manual, www.illumina.com).

Illumina sequencing is based on sequencing by synthesis chemistry (SBS). During the DNA synthesis cycle, fluorescently-labelled deoxyribonucleotide phosphates (dNTPs) are incorporated into a DNA template by DNA polymerase. The nucleotides are detected by fluorophore excitation at the time of incorporation. This process is followed in massive parallel fashion. The sample preparation and sequencing steps include library preparation, cluster generation and sequencing.

1.6.1 Library preparation

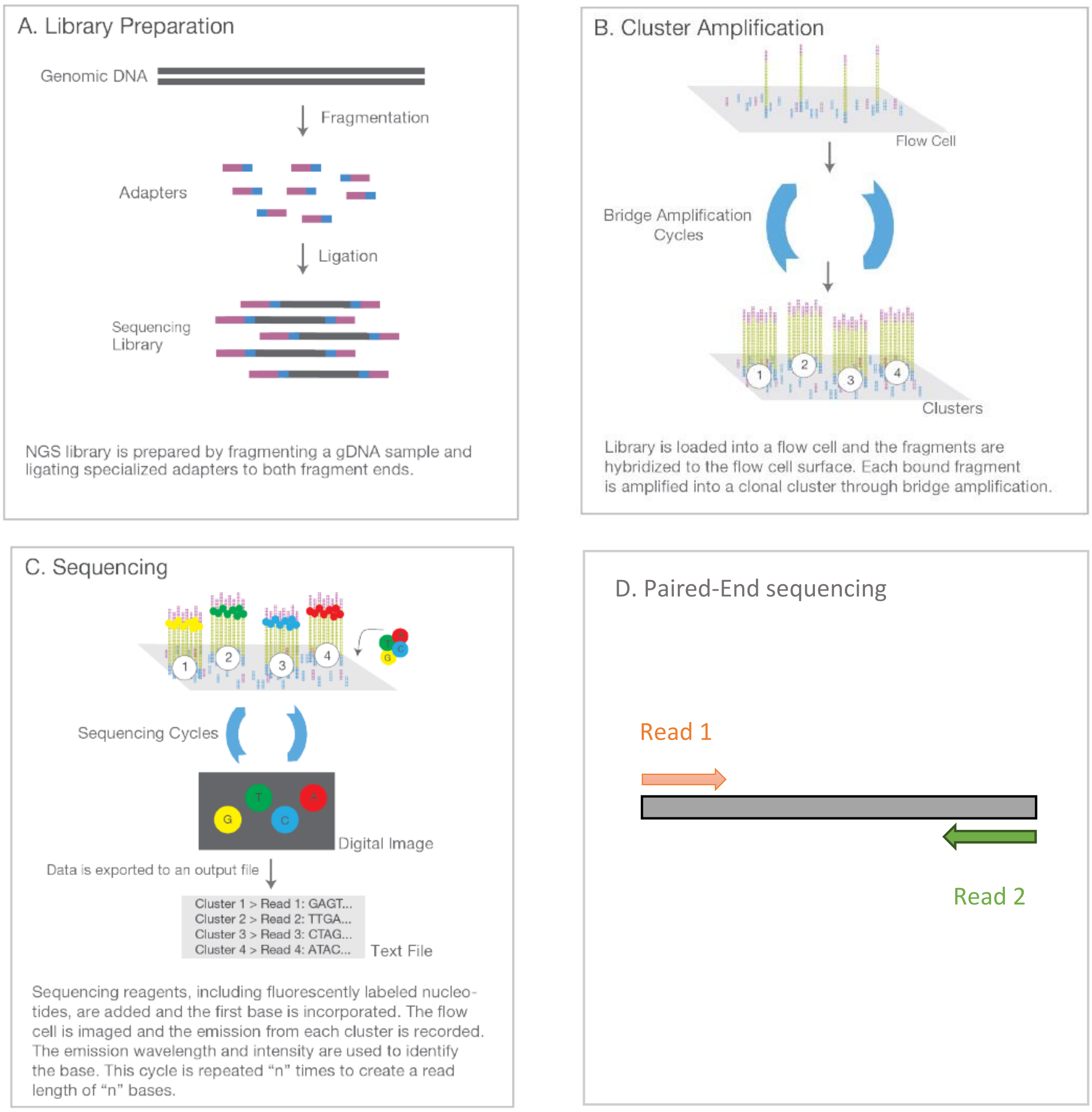
Library preparation means a step, in which sample nucleic acid is fragmented into smaller units and in which specific adapters are attached on both ends to allow identification of the fragments. The first step in sample processing is fragmentation of nucleic acid, which is often an automated and optimized process and mediated by mechanical shearing or enzymatic treatment. If the sample is RNA, it is converted to a complementary DNA, cDNA, before fragmentation. This step is followed by 5' and 3' adapter ligation. Sequences from the pooled sample library are identified in the analysis stage based on the unique adapter indices introduced to the sample during the library preparation step.

1.6.2 Cluster generation

To prepare the fragment library to the actual NGS sequencing step, the library is loaded into a flow cell. A flow cell is a glass support, which has patterned nanowells with attached DNA probes (oligos). These surface-bound oligos are complementary to the library adapters enabling hybridization of fragments to the oligos on the glass surface. A polymerase is used to create a complementary strand using the hybridised fragment as a template. The double-stranded molecule is then denatured, and the original template strand is washed away. For bridge amplification, a single-stranded molecule folds over and hybridizes to an adjacent primer, which forms a bridge. A polymerase synthesizes the reverse strand, which forms the double stranded bridge. Denaturation of this bridge results in the formation of single stranded copies of the molecule tethered to the flow cell. This process is repeated several times and occurs simultaneously for millions of clusters, hence clonally amplifying all the fragments. After the bridge amplification, the reverse strands are cleaved and washed off, which leaves only the forward strands. Blocking of 3'ends prevents unwanted priming. When cluster generation is complete, the templates are ready for sequencing.

1.6.3 Sequencing

The SBS technology of Illumina employs the reversible terminator-based method, which detects single bases as they are incorporated into DNA template strands. During sequencing, fluorescently labelled nucleotides are added to the strand. When the flow cell is imaged, each base is identified by the emission of unique wavelength intensity. Illumina HiSeq3000 sequencer uses a four-channel sequencing. This type of chemistry utilises unique fluorescent label for each of the nucleotide bases i.e. adenine, cytosine, guanine and thymine. Each intensity is captured with four images by the instrument. Sequencing begins with the extension of the first sequencing primer to produce the first read. With each cycle, fluorescently tagged nucleotides are competitively added to the growing chain and only one base is added based on the template sequence. After addition of each base the clusters are excited by a light source and a characteristic fluorescence is emitted. The length of the read is determined by the number of the cycles. The base call is determined by the emission wavelength and the fluorescence intensity. The clusters are sequenced in massive parallel way. The read product is washed away after completion of the read process. In this step, index1 read primer is introduced and hybridised to the template. The read is generated similar to the first read. After completion of index read, the read product is washed off, and 3'ends of the template are deprotected. The template folds over and binds the adjacent oligo on the flow cell. Index2 is read in the same manner as index1. Polymerases extend the second flow cell oligo forming a double stranded bridge. This double stranded DNA is then linearized and 3'ends are blocked. The original forward strand is cleaved off and washed away leaving only the reverse strand. Read two begins with the introduction of read2 sequencing primer. Sequencing steps are repeated as in read1 until the desired read length is achieved. The read2 product is then washed away. This entire process generates millions of the reads representing all the fragments. Sample reads with similar stretches of base calls are locally clustered. Forward and reverse reads are paired, creating contiguous sequences.



NGS library is prepared by fragmenting a gDNA sample and ligating specialized adapters to both fragment ends.

Library is loaded into a flow cell and the fragments are hybridized to the flow cell surface. Each bound fragment is amplified into a clonal cluster through bridge amplification.

```
Cluster 1 > Read 1: GAGT...
Cluster 2 > Read 2: TTGA...
Cluster 3 > Read 3: CTAG...
Cluster 4 > Read 4: ATAC...
```

Sequencing reagents, including fluorescently labeled nucleotides, are added and the first base is incorporated. The flow cell is imaged and the emission from each cluster is recorded. The emission wavelength and intensity are used to identify the base. This cycle is repeated "n" times to create a read length of "n" bases.

Figure 1.4 Illumina NGS sequencing workflow
A., B., C. are the basic steps of sequencing workflow, while D. shows the paired end read (Adapted from Illumina introduction manual, www.illumina.com)

1.6.4 Paired-end sequencing

The paired-end sequencing involves sequencing both ends of the DNA fragment (as shown in Figure 1.4-D). In a library, forward strand is sequenced, and then complementary strand is kept while cleaving the template strand away. Similar step is performed on the reverse strand. These sequenced ends produced from the original fragments are then aligned as forward and reverse reads, forming a read pair. Paired-end sequencing improves the ability to identify the relative positions of the reads with respect to the genome and offer more accurate read alignment as compared to single-end sequencing, where sequencing is performed from only one end. Hence, the paired-end sequencing is more preferable for *de novo* assemblies as it raises the overall confidence of the result by providing read overlap in the otherwise low-quality areas.

2 Bioinformatic methods

Staying up-to-date with the technologies and current research is “a must” in modern virus diagnostics. Rapid and accurate detection of the emerging pathogens and markers responsible for virulence plays important roles during prevention and control of the worldwide outbreaks. Ability to detect a pathogen without *a priori* knowledge, cost effectiveness, availability of various platforms and high-throughput output have made NGS the preferred method for virus detection in a sample. NGS approach results in raw sequence data of gigabases in size for each experiment run, and therefore computational analysis of the data is necessary. Bioinformatic analysis of NGS data is a multistep process where multiple algorithms and programs are used, and hence it may be useful to define optimal workflow with suitable analysis programs and quality check criteria to minimize errors and to obtain reliable results. A number of NGS data analysis workflows have been described (Ekblom & Wolf, 2014; Lambert et al., 2018; Orton et al., 2016). In general, the workflows have the following generic steps: pre-processing of the reads, assembly of the reads using either reference-based or de novo approaches and further downstream analyses. These steps are described below, along with the respective commonly used programs. Different terms and file formats required for NGS analysis are included tables in appendix A (Table A.1).

2.1 Pre-processing of the reads

Quality control is the first step in every NGS analysis. The output of the sequence run consists of millions of reads, which is most commonly in FASTQ format. The FASTQ file contains the sequence and its quality score. Poor base quality due to base miscalls and primer or adapter contamination are considered as common sequencing errors or artefacts of the NGS reads. To ensure accuracy of final results quality check is important. The commonly applied tool for quality check is FastQC (https://www.bioinformatics.babraham.ac.uk/projects/fastqc/). FastQC produces summary statistics, which include sequence quality and distribution, overall GC content, length and duplicates in the reads. For paired-end reads the quality check is performed on both forward and reverse reads.

Adapter removal depends on the protocol used during the library preparation of sequencing. This step is important as adapter sequence can interfere with the read mapping as well as other analyses like SNP (single nucleotide polymorphism) or variant calling. Trimmomatic

(Bolger et al., 2014) and Cutadapt (Martin, 2011) are the most common applications for adapter removal.

Biological samples will predominantly contain host-derived sequences, which in some cases may overshadow reads for putative pathogen. Hence, mapping the sequencing reads to host genome sample (host sequence subtraction) and using the remaining unmapped reads for the assembly is common practice. The common mapping algorithms used are CLC read mapper (https://resources.qiagenbioinformatics.com/white-papers/White_paper_on_CLC_read_mapper.pdf), BWA (Li & Durbin, 2009) and Bowtie2 (Langmead & Salzberg, 2013).

2.2 Assembly of the reads

NGS equipment generate a large number of reads, covering different parts of the genome. With the help of assembly software, these reads are combined into large contigs (contiguous linear stretches of DNA or RNA consensus sequence, *in silico*, constructed by aligning a number of smaller overlapping sequencing reads) or scaffolds (two or more contigs joined together using read-pair information). There are basically two methods that are used for genome assembly: reference-based mapping and *de novo* assembly (Daly et al., 2015; Orton et al., 2016). Reference-based mapping is preferred method for most of the NGS experiments, if closely related and well-annotated reference genome is available for the target. Reference-based mapping is performed by indexing a reference genome and aligning the sequence reads to this reference index. SAMtools (http://samtools.sourceforge.net.) is a bioinformatic tool that provides utilities to manipulate the alignments in SAM/BAM formats. SAM (Sequence alignment/Map) is used to store the mapped reads and BAM (Binary alignment/Map) is a form of SAM, but the data is stored in binary format. SAMtools allows manipulation of alignment in per-position format by indexing and sorting. The mapping tools used include Mosaik (W. P. Lee et al., 2014), Stampy (Lunter & Goodson, 2011), BWA and Bowtie2.

In the absence of known or related sequence, a *de novo* approach is used. The most popular assemblers are based on de Bruijn graph. It is considered as an anti-intuition algorithm (Ramana M. Idury and Michael S. Waterman, 1995). It works by first cutting reads into shorter k-mers, using all k-mers to capture the de Bruijn graph, which captures the overlaps between the k-mers for k-1 length and genome sequence can be inferred on this de Bruijn graph. Short-read assembly programs based on this approach include Velvet (Zerbino &

Birney, 2008), SOAPdenovo (Luo et al., 2012), IDBA-UD (Peng et al., 2012), SPAdes (Bankevich et al., 2012) and ABySS (Simpson et al., 2009).

Overlap layout consensus (OLC) (Staden, 1980) is another assembly approach and works in three steps; first by finding the overlaps among all reads, then forming the layout of all the reads and last by overlapping the information on the graph and determining the final consensus sequence (Ekblom & Wolf, 2014; Liu et al., 2012; Orton et al., 2016). OLC assemblers include MIRA (Chevreux, 2018) (Chevreux B., Wetter T. & Suhai S., 1999) and Edena (Hernandez et al., 2008).

2.3 Quality assessment and validation

After successful assembly, it is important to check the quality of the assembled reads. In absence of an optimized process for the given target, it is advisable to test and compare different programs. If more than one type of assembler is used, then the different assemblers can be compared. Comparisons are based on several evaluation metrics like N50 (the shortest sequence length at 50% of the genome), number of contigs generated and contig length etc. (Kremer et al., 2017; Lischer & Shimizu, 2017; Song et al., 2019). The bioinformatic tool commonly used for this purpose is QUAST (quality assessment tool) (Gurevich et al., 2013). QUAST can be used to evaluate the assemblies and make decision about using the suitable assembler. BLAST (Basic Local Alignment Search Tool) (Altschul et al., 1990) is a robust tool to check the validity of the contigs or scaffolds. BLAST uses National Center for Biotechnology Information (NCBI) sequence database. Both tools are described in brief as follows.

QUAST employs the Nucmer aligner from MUMmer v3.23 (Kurtz et al., 2004) for reference guided assemblies. It can also compute and evaluate *de novo* sequence assemblies. The various metrics evaluated by QUAST include contig sizes, misassemblies and structural variations, genome functional elements i.e. GC content, duplicate ratios, indels and mismatches etc., and N50 variations. The different plots like cumulative plots, GC content plots and contig alignment plots are used to present statistics. Comparative histograms of different parameters are also generated. Some of the output files are mentioned in the table. QUAST supports the file formats like PNG and PDF. The QUAST interface is easy to use and visualize, is reasonably fast and accepts multiple assemblies to compare. QUAST can be run on Linux and macOS platforms. It is also available as a web interface application. More details about instructions and output files can be found at http://quast.sourceforge.net/docs/manual.html.

BLAST performs a sequence similarity search. It compares the sequence of interest to sequence databases and calculates the statistical significance of the resultant matches. The BLAST results are generated quickly, and it helps to make a decision about a given alignment with 'expect value' calculation. This provides an idea about probability of matches at a given score. The BLAST algorithm is based on the modular nature of the proteins. Proteins have one or more functional domains in them, and different species may have the same domains of these proteins. The algorithm finds these domains of similarity. Hence, BLAST can also be used to find evolutionary and functional relationships and identify with the members of the given family. Two frequently used BLAST algorithm types are BLASTN (nucleotide query against nucleotide database) and BLASTX (nucleotide query against protein database) (Madden, 2013). NCBI Virus is the recently developed community portal, which provides resources for sequence data and related information concerning viruses (https://www.ncbi.nlm.nih.gov/labs/virus/vssi/#/, Hatcher et al., 2017).

2.4 Various available platforms and software

Several commercial and non-commercial laboratories are involved in developing various viral NGS as well as metagenomic analyses services. Bioinformatic analysis pipelines have been designed for various purposes and offer options at each analysis point. Various pipelines are explained in the literature elsewhere (Lambert et al., 2018; Orton et al., 2016), however, basic information about commercially available platforms and NGS analysis tools is included in appendix A (Table A.2 and Table A.3). The choice of the pipeline depends on the adequate sensitivity provided for the detection of the virus and the cost.

Despite the availability of tools in viral NGS analysis, lack of computational skills (necessity to work in Unix environment and programming skills) often pose the first bottleneck for wider use. In addition, parameter optimization (different viruses may require different parameters setup), output format handling and data processing, and large data storage capacities may prove to be challenging. For example, VSEARCH, a powerful tool for metagenomic, is an open source and freely available, but it requires command line knowledge.

Considering the need for user-friendly bioinformatic tools and pipelines, Chipster (http://chipster.csc.fi/) was selected as the primary analysis platform to be evaluated in this thesis. Various analysis and visualization tools available in Chipster platform are listed in: (https://chipster.csc.fi/features.shtml). Pipelines in Chipster and external assemblers (not implemented in Chipster environment) for picornavirus sequence assembly are described below.

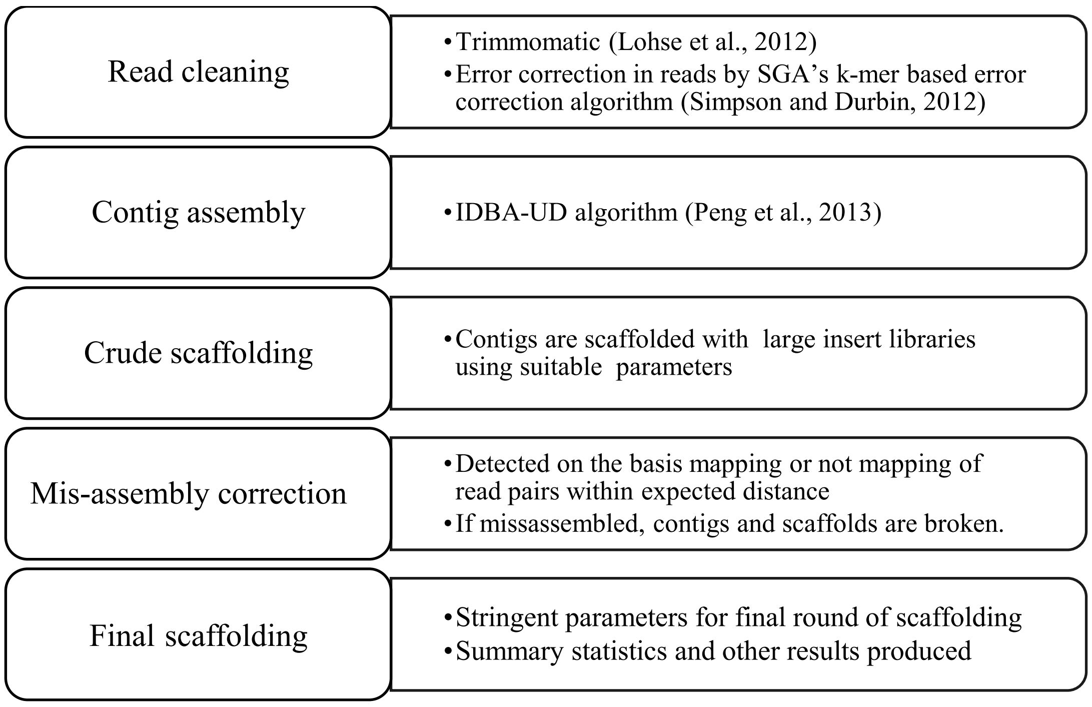
2.5 Chipster

Chipster maintained at Centre for Scientific Computing (Espoo, Finland) (Kallio et al., 2011), provides an easy access platform and intuitive graphical user interface for non-programming biologists to analyse and integrate different kinds of data generated with high-throughput technologies. It has a collection of data analysis methods and integration tools, which allows the user to perform multiple analysis in a consecutive manner and save the performed tasks as reproducible and automatic workflows. It facilitates research by collaboration and sharing the workflows and data analysis sessions. Chipster is an open source, versatile and extendable platform. It supports integration of command line tools, which can be included manually or with the help of CSC help service support. Being a client-server system, it allows tasks to be performed on a laptop or on a single server. Recent version of Chipster is also available as a web application (https://chipster.rahtiapp.fi/home), which can be accessed by registration.

There are two basic automated pipelines available in Chipster: VirusDetect (both reference-mapping and *de novo* assembly) and A5 assembly pipeline for microbial genomes (*de novo* assembly).

VirusDetect (Zheng et al., 2017), is an automated bioinformatic pipeline that can detect both known and novel viruses from small RNA (sRNA) datasets, which has been reported useful in plant and animal virus detection (Kreuze et al., 2009; Wu et al., 2010, 2015). It performs reference-guided assembly through mapping sRNA sequence to a curated virus reference database and *de novo* assembly. For both approaches automated parameter optimization and the option of host sRNA subtraction is used. These assembled contigs are then compared with a reference virus database for virus identification. Details about outputs generated can be found at https://chipster.csc.fi/manual/virusdetect.html.

A5 assembly pipeline (Coil et al., 2015) for microbial genomes is a revised A5-miseq pipeline. It is developed by replacing original A5 assembly components with new software modules to process and improve assemblies, to allow assembly of Illumina long reads. This pipeline consists of the following steps:



Read cleaning
•Trimmomatic (Lohse et al., 2012)
•Error correction in reads by SGA's k-mer based error correction algorithm (Simpson and Durbin, 2012)
Contig assembly
•IDBA-UD algorithm (Peng et al., 2013)
Crude scaffolding
•Contigs are scaffolded with large insert libraries using suitable parameters
Mis-assembly correction
•Detected on the basis mapping or not mapping of read pairs within expected distance
•If missassembled, contigs and scaffolds are broken.
Final scaffolding
•Stringent parameters for final round of scaffolding
•Summary statistics and other results produced

Figure 2.1 Schematic representation of A5-miseq pipeline
The pipeline follows five steps as described in the figure. No parameter optimization is necessary.

The use of A5 pipeline is easy, as it does not require any parameter optimization. All the above-mentioned steps are automated and can be performed on a laptop computer.

2.6 External assembly programs

Although Chipster platform itself does not have a separate *de novo* assembler, CSC offers a variety of assemblers in its server. Two of them, Velvet and SPAdes are shortly described here.

Velvet (Zerbino & Birney, 2008) can be used as a *de novo* assembler to build contigs or gapped assemblies of contigs into scaffolds for short-read datasets. Velvet is a de Bruijn algorithm-based assembler, which builds the reads on the de Bruijn graph, removes the errors and attempts to resolve repeats if given, paired-end read and long read information. Final output includes the assembled reads with related statistics. Velvet uses paired-end FASTA and FASTQ datasets, either as a single merged file or two separate files with reads paired by their ordering. Velveth and velvetg are the main two programs on which Velvet is based on. Both these programs represent the basic two steps of a single Velvet assembly: hashing and graph building respectively. Velveth reads sequence files, builds a dictionary of all k-mers of length k. Here, k is defined by the user. This parameter defines the exact local alignments between the reads. These alignments are used to build a de Bruijn graph by velvetg. Velvetg also removes errors, simplifies the graph and resolves repeats based on user-defined parameters. The hash length or value of k is the most important parameter for a Velvet run. To define the hash length, the following three technical constraints are a must:

1. k must be an odd number to avoid palindromes.
2. Hash length must be below or equal to MAXKMERHASH length, as Velvet requires more memory to store longer words.
3. Hash length must be lower than the read length, as no overlap between the reads will be observed for larger number.

The hash length can be automated with VelvetOptimiser, a script written by Simon Gladman and Torsten Seeman (https://github.com/tseemann/VelvetOptimiser). The script automatically scans the specified parameter range to produce the best possible assembly. Details about application of this script are given in chapter 4 materials and methods. The final outputs in FASTA file are actually scaffolds, i.e. contigs are constructed into scaffolds by adding N's to the corresponding estimated gap length.

SPAdes (St. Petersburg genome assembler) (Bankevich et al., 2012) is an open source *de novo* genome assembler. It is primarily designed to assemble small genomes from standard bacterial sequenced data sets and single cell genomics data sets, which are difficult to assemble due to highly non-uniform read coverage and high levels of sequencing error and chimeric reads. SPAdes supports unpaired, paired-end and mate-pairs reads. Important point to remember while using this assembler is the size of the genome of interest. It is not designed for large genomes like mammalian genomes. To generate scaffolds from contigs SPAdes uses two basic methods. Using the read pairs, first the gap size separating the contigs is determined and second step relies on the assembly graph, i.e. if a complex unresolvable repeat is observed between two contigs, SPAdes joins these contigs with a fixed gap size of 100bp. The final output does not have any N sequences.

3 Aim of the study

The aim of this thesis was to develop a rapid, straightforward and user-friendly workflow for the assembly and analysis of picornaviral genomes. The specific aim was to generate consensus near full-length viral genomic sequences from the pool of the short contigs.

In total, 17 picornaviral sequences generated by Illumina HiSeq 3000 (paired end, 2 x 75 bp in length) were analyzed in the study using programs implemented in Chipster platform and external assembly programs such as Velvet and SPAdes.

4 Materials and methods

4.1 Material

Seventeen (17) picornaviral genomic sequence sets representing five picornavirus types (Table 4.1) and generated by Illumina HiSeq 3000 (paired end, 2 x 75 bp in length) were used in the study. The sequence reads were in compressed (.gz) format and used as such in the analyses. The virus samples represented interesting clinical isolates from different study cohorts. Viruses were originally diagnosed as entero- or parechoviruses using ENRI-RT-qPCR protocol followed by genetic typing using VP1 Sanger sequencing. The type name was used to obtain corresponding reference viral genome from GenBank for reference mapping (RefMap analysis). The reference genomes are shown in Table 4.1.

Table 4.1 Reference genomes used in the study for RefMap analysis
The following table includes virus nomenclature as per recommended by Simmonds et al., 2020, the NCBI accession id for the virus genome and its respective genome length.

	Virus	**NCBI Accession ID**	**Definition**	**Length**
1	EV-D68	NC_038308.1	Human enterovirus 68 strain Fermon, complete genome	7367 bp
2	E18	MN749146.1	Echovirus E18 strain 12J3, complete genome	7422 bp
3	PeV-A3	KM986843.1	Human parechovirus 3 strain VGHKS-2007, complete genome	7349 bp
4	PeV-A1	FM178558.1	Human parechovirus 1, complete genome	7380 bp
5	E13	AY302539.1	Human echovirus 13 strain Del Carmen complete genome	7410 bp

4.2 Description of workflow

Chipster was originally chosen as test platform since it was considered user-friendly and applicable for picornavirus sequence analysis. To create consensus near full length picornaviral sequences, Chipster with automated pipelines and other methods were used to pre-process, assemble and analyse the same sequence set. Figure 4.1 represents the schematics for the various steps carried out during this study.

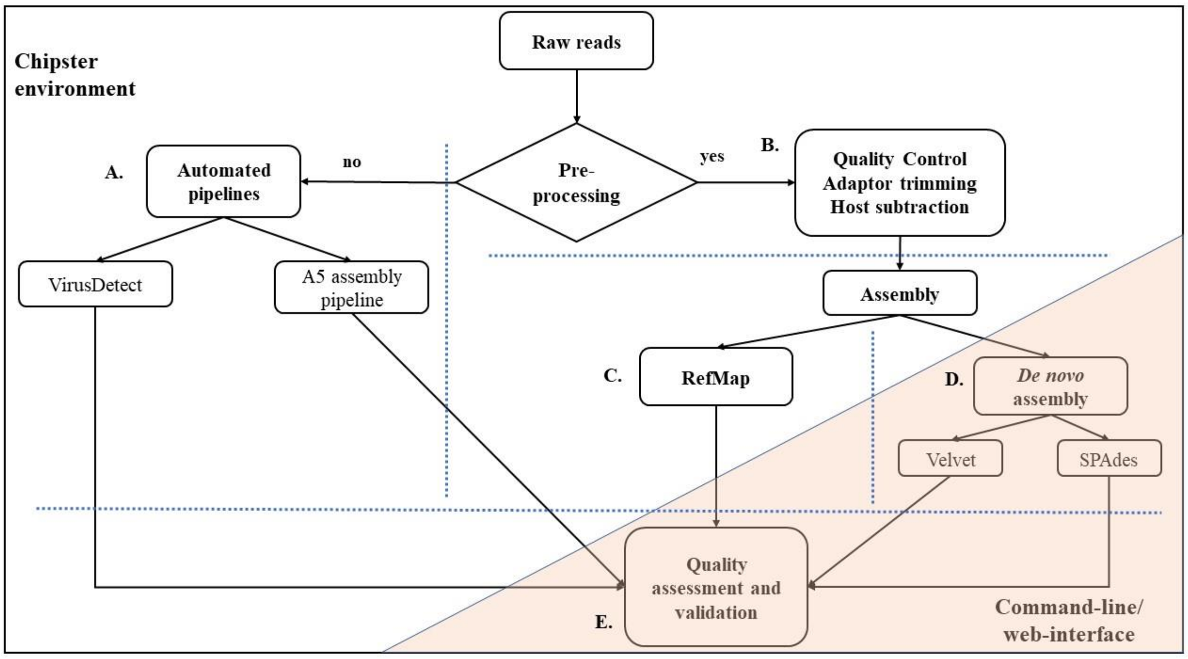


Figure 4.1 Schematic representation of various bioinformatic approaches
A. Application of default pipelines without any pre-processing of data. B. Pre-processing steps required to carry out analysis other than automated pipelines. C. Testing reference-mapping approach for sequence assembly. D. *De novo* approach with external assembly programs. E. Quality assessment and validation of outputs generated by all approaches. Steps A. to C. were carried out in Chipster environment (white background), while steps D and E were implemented outside Chipster (light orange background). The dotted blue lines represent division of work during study.

The work was divided into different parts and tests based on implementation of default pipelines (VirusDetect and A5 assembly pipeline for microbial genome assembly), pre-processing raw sequence reads to try different approach for assembly than automated pipelines, assembly of the pre-processed reads with reference-mapping and *de novo* assembly programs and finally, validation and quality assessment of the outputs generated (contigs/scaffolds) from these different methods. The analysis steps were carried out within as well as outside Chipster environment as shown in Figure 4.1.

The user first logs in to the Chipster platform using login id and password. User accounts are freely available upon request to CSC helpdesk. Illumina paired-reads are imported with "import files" option of the file menu. Here, the primary analysis is done with pipelines available in the Chipster under "utilities" menu. VirusDetect and A5 assembly pipelines for microbial genomes are two automated assembly programs in Chipster and do not require any parameter optimization. Thus, in its simplest manner, genome analysis can be performed only in a few steps in Chipster without pre-processing steps.

For the sequence assembly using approaches other than those included the pipelines implemented in Chipster, pre-processing of the raw reads is required. Such pre-processing steps can be performed in Chipster and are as follows: Quality analysis of reads is performed with FastQC, trimming of reads with Trimmomatic and host sequence subtraction with Bowtie2. The processed reads are then used for assembly.

Sequence assembly was performed using two approaches: reference-mapping (RefMap) and *de novo* assembly. RefMap was done within Chipster with Bowtie2. *De novo* assembly was performed outside Chipster environment and with Velvet (VelvetOptimiser) and SPAdes programs.

Long contigs were submitted to National Center for Biotechnology Information (NCBI) for BLAST analysis. Quality of contigs/scaffolds produced was assessed with QUAST. Details about steps and parameters are given below.

4.3 Automated analysis pipelines implemented in Chipster

For VirusDetect pipeline, the input data file should be a FASTQ file. The assembled contigs are compared to the reference viral genome for identification with the help of BLASTN and BLASTX. During this study, parameters to set were reference virus database and host organism (if unavailable in drop down menu, own genome could be uploaded). Other parameters were used with default settings. VirusDetect produces a large number of files, and all files can be stored in a single .tar file from the "option".

A5 assembly pipeline for microbial genomes is suitable for constructing small genomes based on Illumina MiSeq data, but it is essential that the input data file contains paired-end reads.

As mentioned earlier, this pipeline does not require pre-processing of the raw reads or any parameter settings.

4.4 Alternative approaches for sequence analysis

4.4.1 Pre-processing of raw sequence reads

All the steps included in the pre-processing step were performed in Chipster platform. Quality of the sequence reads was checked with FastQC (https://www.bioinformatics.babraham.ac.uk/projects/fastqc/). The option of "MultiQC" format allowed a single quality report output file for multiple sequences.

In the following step, the reads were trimmed with the 'Trimmomatic' (Bolger et al., 2014), with parameters:

Adapter set= none,
Quality scale used in the fastq file= phred + 33,
sliding window trimming parameters = 20:30,
minimum length of reads to keep = 50,
write a log file = yes.

In the final step of pre-processing, the host sequences were removed with 'Bowtie2 for paired-end reads' (Langmead & Salzberg, 2013) with two or more read files and options:

strategy = Very Sensitive,
Put unaligned reads to a separate file = yes.

The unaligned reads were collected into separate file. For the future convenience, the fastq file were compressed using .gzip ("NGS"-"Utilities" menu of Chipster).

The final output files from pre-processing stage served as input files for the assembly stage. Two approaches were utilised for assembly.

4.4.2 RefMap

Trimmed reads were mapped with Bowtie2 paired end reads and own genome, as the required viral reference genomes were not available in the Chipster. The reference genomes were uploaded in the fasta file format. The parameters set were strategy = Very Sensitive, Put unaligned reads to a separate file = no. The mapped reads were then visualized with built-in genome browser.

The genome browser uses the .bam file from the mapping step. As the reference genomes are not indexed, they were indexed in Chipster with option “Index FASTA”, available under “utilities”.

For visualization process, one chooses the .bam file of the reads and fasta format for reference sequence. This selection makes the genome browser option visible. After selecting the correct reference genome from drop down menu, reference genome and mapped reads alignment can be explored.

4.4.3 *De novo* assembly

Currently, there are no separate *de novo* assembly tools available on Chipster platform, although one can avail them by contacting CSC. Two tools were successfully employed – Velvet assembler and SPAdes assembly tool. As both were command line tools, the assembly run was performed in the Linux environment, Ubuntu 18.4 distribution.

For *de novo* assembly, first program used was VelvetOptimiser, which is designed as wrapper script for Velvet assembler, which provides the best possible assembly with hash length optimisation. The information specified included the working directory name, the hash length, and a list of filenames, with file format and read type. The steps performed for assembly were as follows:

1. A range of k-mers (hash length) was determined. For this study, the range of k-mers was given from 21 to 77.
2. The appropriate file description line for velveth was determined, viz. '-shortPaired -fastq.gz’. This line described the input reads as a short read of paired-end data. As the paired-end reads were in separate file, this information was also provided along with the respective file names.
3. VelvetOptimiser was run with the script below.
4. Output data was collected in the working directory. Upon exiting, VelvetOptimiser printed out the directory in which it left the output of its final Velvet assembly. This directory contained the standard Velvet output files.

VelvetOptimiser script

```
velvetoptimiser.pl -s 21 -e 75 -f '-shortPaired -fastq.gz -separate reads1.fq.gz reads2.fq.gz'
```

Where, -s = starting or lower hash value;

-e = end or higher hash value;

-f = velvethfiles (file format options = fastq.gz; read type = shortPaired)

-separate = separate files for forward and reverse reads of a paired end library

Other program used for *de novo* assembly was SPAdes. SPAdes supports paired-end, mate-pairs as well as unpaired reads. As it was initially designed for small genomes, it was chosen for picornavirus sequence reads. The general command line script was as follows, the file names and output directories were specified per sequence. The paired-end fastq files (forward and reverse reads) were provided. The option 'careful' was recommended for small genome assemblies, as it reduces the number of mismatches and short indels. The SPAdes output files were stored in the output directories specified.

SPAdes script

```
spades.py -1 illumina_R1.fastq.gz -2 illumina_R2.fastq.gz --careful -o R_spades
```

where, R defined the name of the sequence under assembly

-1 = input file of forward reads,

-2 = input file of reverse reads,

--careful = minimized mismatches and short indels,

-o = the output directory.

4.5 Quality analysis and validation of contigs/scaffolds

Longest contigs/scaffolds were identified for putative viral hits with the help of BLAST. This option is also available in Chipster, but only 10 sequences can be run at a time. The BLAST analyses were done manually for each contig/scaffold for each sequence run. The additional setting of BLASTN was used. The hits identified as viral hits, were saved separately in a file.

To compare the contigs/scaffold generated form the applied pipelines and assemblers QUAST tool was used. It is python-language based tool and the script used was as follows.

QUAST accepted assemblies and reference genomes in FASTA format. The output files were generated into specified directory.

QUAST script

```
quast.py seq_name_A5.fasta  seq_name_SPAdes.fasta  seq_name_velvet.fasta -r ref.fasta -o
name_quast
```

where, input files = contigs/scaffolds in fasta format for each assembly;

r = reference sequence in fasta format (not mandatory)

o = name for output directory

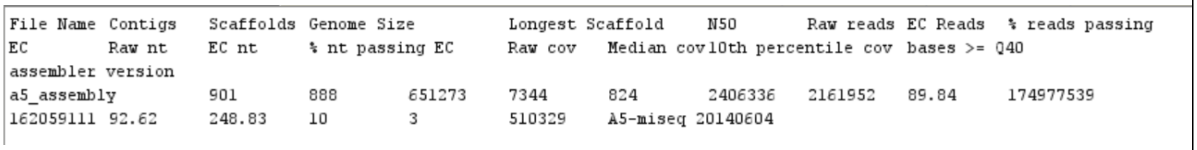
5 Results

The primary goal of this thesis was to develop a straightforward and user-friendly workflow for picornaviral genome sequencing and sequence analysis. The basic idea of the thesis was to test the feasibility of Chipster platform for picornavirus genome analysis. The details about software used and their respective output files are included in appendix B.

5.1 Automated analysis pipelines implemented in Chipster

Two pipelines implemented in Chipster were tested with the dataset. VirusDetect pipeline failed to produce any results, while A5 pipeline was successfully applied to the available sequence sets. The output files obtained from A5 pipeline are listed in appendix B (Table B.1). The summary statistics of assembly are provided in a tab separated file and provide information of total number of contigs and scaffolds generated, longest scaffold, N50 etc. The fasta files for final contigs and final scaffolds contain assembled sequences as contigs and scaffolds respectively. Both file formats are shown in Figure 5.1.

A. Summary statistics of assembly



```
File Name Contigs   Scaffolds Genome Size         Longest Scaffold    N50       Raw reads EC Reads  % reads passing
EC        Raw nt    EC nt     % nt passing EC     Raw cov   Median cov10th percentile cov bases >= Q40
assembler version
a5_assembly         901       888       651273    7344      824       2406336   2161952   89.84     174977539
162059111 92.62     248.83    10        3         510329    A5-miseq 20140604
```

B. Fasta file for final contig

```
>scaffold_0
TTTTTTTTGTGGGGGGTGTCTTTGGGGTTTGGTTGGTTCGGGGTATGGGGTTAGCAGCGGTGTG
TGTGTGCTGGGTAGGATGGGCGAGGGTTGTATTGATGAGATTAGTAGTATGGGAGTGGGAGGG
GAAAATAATGTGTTAGTTGGGGGGTGACTGTTAAAAGTGCATACCGCCAAAAGATAAAATTTG
AAATCTGGTTAGGCTGGTGTTAGGGTTCTTTGTTTTTGGGGTTTGGCAGAGATGTGTTTAAGTG
CTGTGGCCAGAAGCGGGGGGAGGGGGGGTTTGGTGGAAATTTTTTGTTATGATGTCTGTGTGGA
```

Figure 5.1 Output files for A5 assembly pipeline
A. shows a tab separated file with important statistics such as number of contigs and scaffolds, longest scaffold and N50. B. A fasta format file for final contig, note: all sequences are not shown, the single file contains all the contigs or scaffolds generated which are serially numbered from 0.

Results for A5 assembly pipeline output analysis are shown in Table 5.1, which include the longest scaffolds generated for given sequences. Viral hits of comparatively shorter lengths are not mentioned in the result here.

Table 5.1 Results of A5 assembly pipeline output analysis
The table provides details about sequence ID, expected hit with Sanger typing, number of scaffolds generated, longest scaffold generated for a given sequence and respective putative viral hits identified with BLASTN. The viral hits are mentioned with abbreviated virus name. The full names of virus with their respective abbreviations are provided in appendix C (Table C.1).

Sequence	Sanger typed as	Total scaffolds	Scaffold number	length in bp	Putative Viral hit
FFGC_VIR_1_S104	EV-D68_19	18829	Scaffold_17	6950	E18
FFGC_VIR_2_S105	EV-D68_21	17826	Scaffold_26	5918	E18
FFGC_VIR_3_S106	EV-D68_23	17166	Scaffold_8	7356	E18
FFGC_VIR_4_S107	EV-D68 Fermon(type strain).	68	Scaffold_0	7376	EV- D68
			Scaffold_1	7107	E18
FFGC_VIR_5_S108	E-18	586	Scaffold_1	7135	E18
FFGC_VIR_6_S109	E-18	104	Scaffold_1	7458	E18
FFGC_VIR_7_S110	E-18	2836	Scaffold_1	7401	E18
FFGC_VIR_8_S115	E-18	146	Scaffold_0	7409	E18
FFGC_VIR_9_S114	E-13	46	Scaffold_0	5774	E13
FFGC_VIR_10_S113	etiology by unknown pathogen	114	Scaffold_13	775	EV-B
FFGC_VIR_11_S112	EV-D68	221	Scaffold_0	7339	EV-D68
FFGC_VIR_13_S111	EV-D68	901	Scaffold_0	7340	EV-D68
FFGC_VIR_18_S116	PeV-A3/Vi988	505	**None**		
FFGC_VIR_19_S117	PeV-A3/152037	22	Scaffold_0	6749	PeV-A3
FFGC_VIR_20_S118	PeV-A3/145-8	167	Scaffold_2	3053	PeV-A1, PeV-A3, PeV-A4

FFGC_VIR_21_S119	PeV-A1/101-17	14949	Scaffold_8	7335	PeV-A1
FFGC_VIR_22_S120	PeV-A1/103-2	16333	Scaffold_10	7314	PeV-A1

Considering the longest lengths of scaffolds generated for each paired-end sequence reads, the longest scaffold was 7458bp (FFGC_VIR_6_S109, Scaffold_1), while shortest was 775bp (FFGC_VIR_10_S113, Scaffold_13). It was interesting to note that for one sequence (FFGC_VIR_4_S107) two near full length scaffolds of lengths 7376bp (Scaffold_0) and 7107bp (Scaffold_1) were generated. Both scaffolds showed two different putative viral hits for EV-D68 (expected based on Sanger typing) and E18. For a sequence (FFGC_VIR_20_S118) with scaffold of intermittent length (3053bp) multiple viral hits were observed. Only one sequence (FFGC_VIR_18_S116) (shown in bold) failed to produce any viral hits for A5 pipeline. It should be noted that PCR Ct values were not available for the samples, and therefore it cannot be concluded whether differences in lengths are due to differences in the copy number, which ultimately affect the quality of sequenced library.

5.2 Alternative approaches for sequence reads analysis

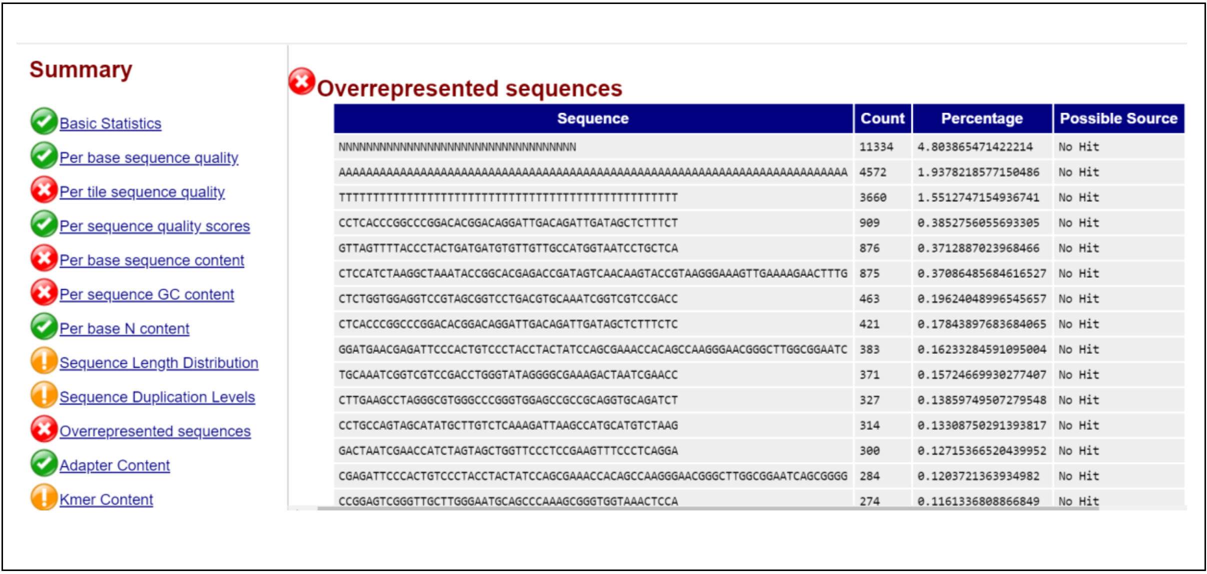
5.2.1 Pre-processing of raw sequence reads

Pre-processing of raw reads consists of quality control, adapter removal and host genome subtraction from the given sequence datasets. All three steps were performed in Chipster.

5.2.1.1 Quality control with FastQC

FastQC analysis generated two output files for each read- a html file of FastQC analysis report of individual sequence read and a mqc file, to combine all FastQC reports in a single file. The report included tables and graphical plots for general statistics, sequence counts, sequence quality histograms, per sequence quality score, per base sequence content, per sequence GC content, per base N content, sequence length distribution, sequence duplication levels, overrepresented sequences an adapter content.

FastQC analysis indicated absence of poor-quality bases and any adapter contamination for all sequences in dataset. For each module run in FastQC analysis, success or warning is indicated with a colored symbol. For a given module, green tick indicates that module is ok, orange exclamation mark indicates it is slightly unusual and can be further investigated and red cross indicates it is very unusual and might be wrong. Figure 5.2 shows a part of FastQC analysis report, with warning for overrepresented sequences for sequence, FFGC_VIR_18_S116. Assembly of this sequence with A5 pipeline failed to generate any viral contigs. Later in the study, after implementation of other assembly programs, some of the sequences resulted into either smaller length contigs or with no contigs of interest at all. After comparing the FastQC reports, it was observed that all these sequences showed red cross warning i.e. high percentage of overrepresented sequences as compared to other sequence reads.


Summary
Basic Statistics
Per base sequence quality
Per tile sequence quality
Per sequence quality scores
Per base sequence content
Per sequence GC content
Per base N content
Sequence Length Distribution
Sequence Duplication Levels
Overrepresented sequences
Adapter Content
Kmer Content

Overrepresented sequences

Sequence	Count	Percentage	Possible Source
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN	11334	4.80386547142214	No Hit
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA	4572	1.9378218577150486	No Hit
TTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTT	3660	1.5512747154936741	No Hit
CCTCACCCGGCCCGGACACGGACAGGATTGACAGATTGATAGCTCTTTCT	909	0.385275605569305	No Hit
GTTAGTTTTACCCTACTGATGATGTGTTGTTGCCATGGTAATCCTGCTCA	876	0.371288702396846	No Hit
CTCCATCTAAGGCTAAATACCGGCACGAGACCGATAGTCAACAAGTACCGTAAGGGAAAGTTGAAAAGAACTTTG	875	0.37086485684616527	No Hit
CTCTGGTGGAGGTCCGTAGCGGTCCTGACGTGCAAATCGGTCGTCCGACC	463	0.19624048996545657	No Hit
CTCACCCGGCCCGGACACGGACAGGATTGACAGATTGATAGCTCTTTCTC	421	0.17843897683684065	No Hit
GGATGAACGAGATTCCCACTGTCCCTACCTACTATCCAGCGAAACCACAGCCAAGGGAACGGGCTTGGCGGAATC	383	0.16233284591095004	No Hit
TGCAAATCGGTCGTCCGACCTGGGTATAGGGGCGAAAGACTAATCGAACC	371	0.15724669930277407	No Hit
CTTGAAGCCTAGGGCGTGGGCCCGGGTGGAGCCGCCGCAGGTGCAGATCT	327	0.13859749507279548	No Hit
CCTGCCAGTAGCATATGCTTGTCTCAAAGATTAAGCCATGCATGTCTAAG	314	0.13308750291393817	No Hit
GACTAATCGAACCATCTAGTAGCTGGTTCCCTCCGAAGTTTCCCTCAGGA	300	0.12715366520439952	No Hit
CGAGATTCCCACTGTCCCTACCTACTATCCAGCGAAACCACAGCCAAGGGAACGGGCTTGGCGGAATCAGCGGGG	284	0.120372136393498	No Hit
CCGGAGTCGGGTTGCTTGGGAATGCAGCCCAAAGCGGGTGGTAAACTCCA	274	0.116133680886849	No Hit

Figure 5.2 Part of FastQC analysis report (FFGC_VIR_18_S116)
The left-hand panel, under title summary indicates the modules run for FastQC analysis and colored symbols indicating success or warning. Warning for overrepresented sequences was observed as common for all sequences which either resulted into smaller contigs or no viral contigs at all. Sequence, FFGC_VIR_18_S116, failed to generate any viral contigs for A5 pipeline as well as other assemblers.

5.2.1.2 Adapter removal and trimming with Trimmomatic

FastQC was followed by application of Trimmomatic and Bowtie2. With the Trimmomatic, the sequence reads are trimmed and filtered. Parameters setting is an essential step for Trimmomatic. The encoding determines the offset of quality scores. Here, phred+33 encoding was set, which suggests ASCII 33 offset (that is, the quality characters in FASTQ file are equal to their quality plus 33). Adapter clipping parameters trim the adapter set used during library processing. These details can be found in the sequencing summary reports. Adapter sets are available in Chipster and if not, the user can upload own set of adapters for trimming. For the given dataset, adapters were removed beforehand, hence adapter set trimming was not required. Sliding window parameters trim the low-quality sequences in the reads. The window size determines the average quality within the window and sequences are trimmed when the quality falls below this threshold. Setting a window considers quality of multiple bases, which helps to avoid removal of high-quality bases surrounding a single poor-quality base. The minimum read length parameter decides the minimum length of the read to retain after clipping. For the study, different settings of the parameters for both window size and minimum length were tried as suggested in various citations. Current settings considerably reduced the number of reads than other setting combinations and hence, were decided to use.

5.2.1.3 Host sequence subtraction with Bowtie2

The trimmed reads were then aligned with human genome as a host with Bowtie2. The ‘very sensitive’ option is more sensitive and accurate. This resulted in the host sequence depletion and further decreasing the number of reads. The most significant host sequence depletion was observed in 10 out of 17 sequences as represented in Figure 5.3. The reduction in number due to trimming and host subtraction is shown as follows (Table 5.2).

Table 5.2 Reduced number of reads with trimming and host subtraction

Sequence	**Input read pairs**	**Trimmed reads**	**Host subtraction**
FFGC_VIR_1_S104	3649440	3223444	788577
FFGC_VIR_2_S105	2872004	2593354	648896
FFGC_VIR_3_S106	2876908	2556630	678412
FFGC_VIR_4_S107	975519	599085	297354
FFGC_VIR_5_S108	2590348	2339714	2157011
FFGC_VIR_6_S109	1908565	1705746	1654351
FFGC_VIR_7_S110	3150044	2861061	2454296
FFGC_VIR_8_S115	732111	616362	613755
FFGC_VIR_9_S114	1668916	1487498	1450294
FFGC_VIR_10_S113	838109	448550	11363
FFGC_VIR_11_S112	855702	508013	118503
FFGC_VIR_13_S111	1203168	968805	660489
FFGC_VIR_18_S116	235935	183996	50599
FFGC_VIR_19_S117	3855605	3480990	3473068
FFGC_VIR_20_S118	194498	176036	59875
FFGC_VIR_21_S119	3306931	2997088	996686
FFGC_VIR_22_S120	3449781	3118051	920137

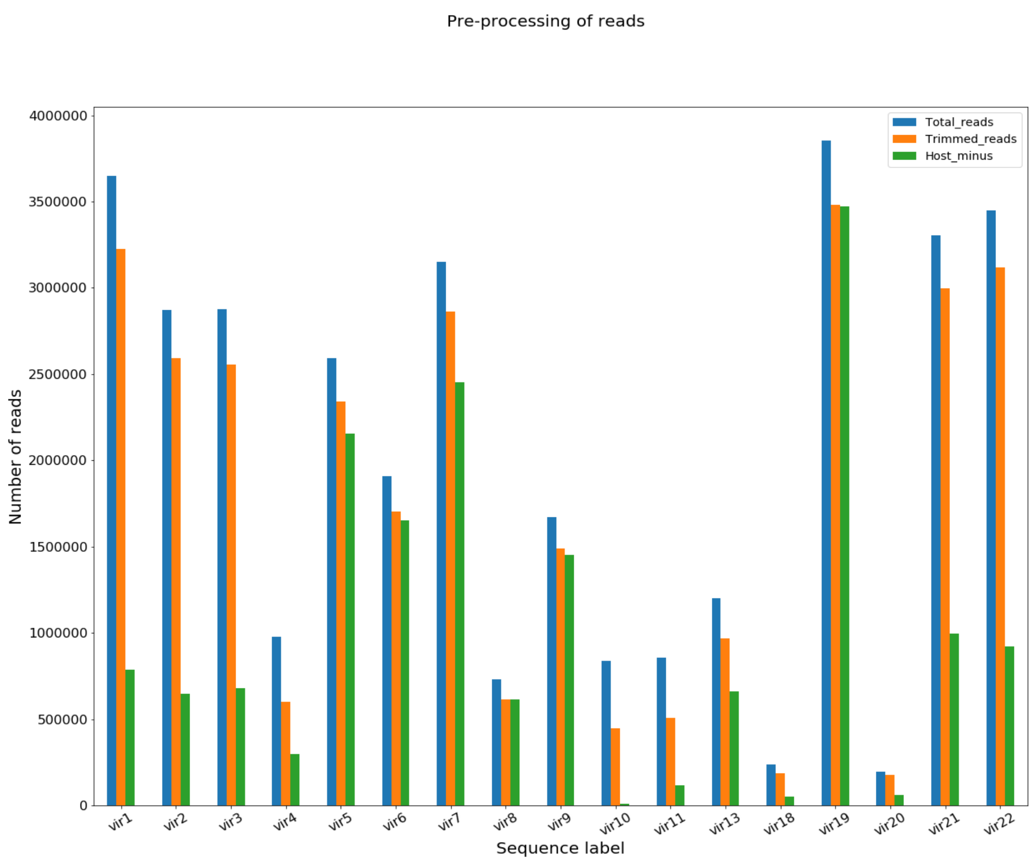


Figure 5.3 Effect of trimming and host sequence subtraction on total sequence reads

As shown above, the reduction in number of reads was not uniform across the sequences. The unmapped reads were collected in separate output files and were then used for next step of assembly.

5.2.2 RefMap analysis

As the sequences were typed and provided ideas about the related sequences, mapping with reference genome approach was used. The reference sequences are mentioned before (Table 4.1). The results were visualized using IGV, available in Chipster. As the sequence vir10 (FFGC_VIR_10) was typed as pathogen without known etiology, no reference mapping was performed with this sequence. The results for mapping were wide-ranging. Sequence coverage of all ranges (low, intermittent and high) was observed. Figure 5.4 shows coverage observed with reference mapping for selected sequences.

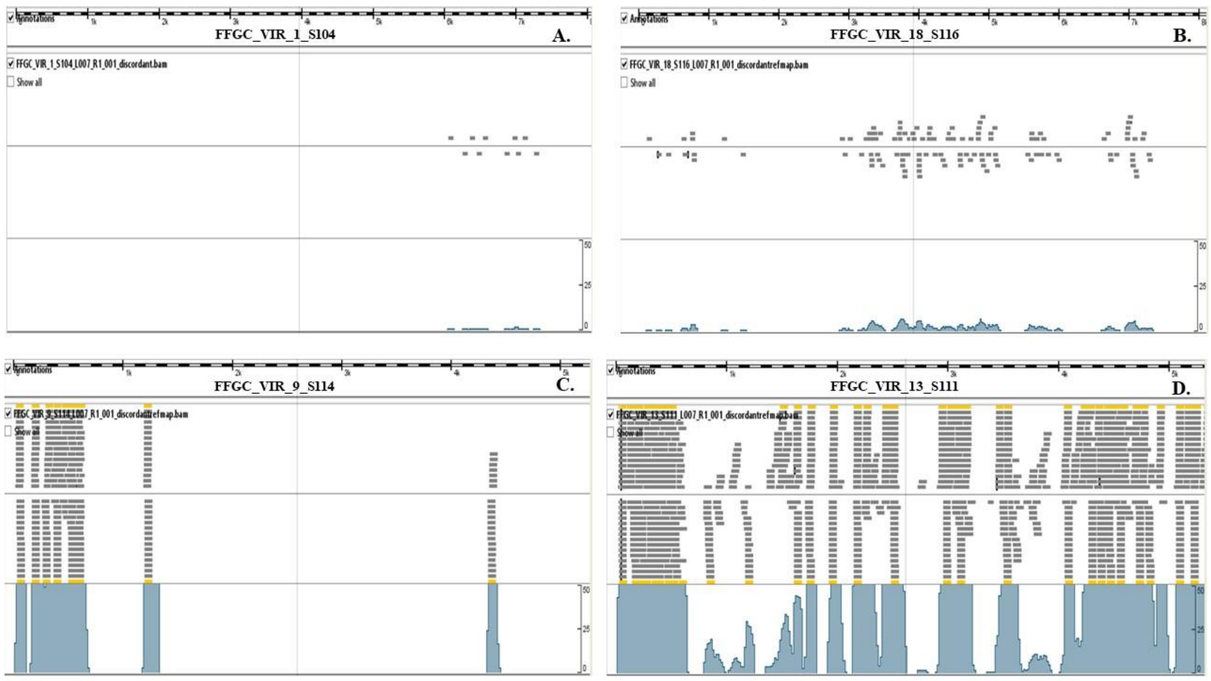

Annotations
FFGC_VIR_1_S104
A.
FFGC_VIR_1_S104_L007_R1_001_discordant.bam
Show all
Annotations
FFGC_VIR_18_S116
B.
FFGC_VIR_18_S116_L007_R1_001_discordantrefmap.bam
Show all
Annotations
FFGC_VIR_9_S114
C.
FFGC_VIR_9_S114_L007_R1_001_discordantrefmap.bam
Show all
Annotations
FFGC_VIR_13_S111
D.
FFGC_VIR_13_S111_L007_R1_001_discordantrefmap.bam
Show all

Figure 5.4 Sequence coverage observed with RefMap analysis
Figures A. to D. show the different ranges of coverage achieved in reference mapping for selected sequence. A. showed very low coverage, B. with slightly more coverage as compared to A, C. with intermittent coverage, while D. shows the highest coverage as compared to other three.

5.2.3 *De novo* assembly

Until this stage, all previous work- application of automated pipelines (A5 and VirusDetect), alternative workflow to these pipelines, which included quality control of reads with FastQC, trimming of reads with Trimmomatic, host sequence subtraction with Bowtie2 and RefMap (that is, reference-mapping) with Bowtie2 were implemented in Chipster. However, Chipster platform itself does not have any specific assembly program for *de novo* assembly. Hence, the assemblers Velvet and SPAdes were used outside Chipster. As both assemblers are command line tools, the work was carried out in Linux environment (Ubuntu 18.04). Trimmed and host depleted sequence reads (final output of steps mentioned in 5.2.1.3) were exported from Chipster and were utilised as inputs for *de novo* assembly programs.

5.2.3.1 VelvetOptimiser

The most important parameter for Velvet is a k-mer or hash length. To obtain a successful assembly, it is suggested to have k-mer value between half to 2/3 of a read length. To ensure that appropriate k-mer value is chosen, VelvetOptimiser was used in the study. This program takes a range of values instead of a single value and provides most optimal k-value used to assemble sequence reads. Considering the variable lengths of the reads in the dataset, lower limit of 21 (default is 19) and higher limit of 77 was chosen. The resultant optimal k-mer values for each sequence are given in Table 5.3

When finished with VelvetOptimiser run, the files found in output directory were Contigs file, Graph, PreGraph and Graph2 files, log file, sequences and stats file. The details about the output files are provided in the appendix B (Table B.2). The important files to consider were contigs file, log and stats file. The contigs file provided the information about contigs with k-mer length and k-mer coverage. The assembly with this tool was time-consuming and required more computational storage and memory. The final output file consisted of the contigs for the most optimal k-mer value and content format for a file is described below.

```
>NODE_1_length_7202_cov_271.608429
AAGATTTTTACATTAACCCCATGCCTGGTCTCCACTAGTTGAAGGCAGCTTGCAATAAAA
TGAGTGGGAACAAGACGCTTAAAGCATGGTGTAAATTAACTTTTCTAACTCACACTTTGT
GTGGGGTGGCAGGTGGCGTGCCATAATTCTATTAGTGAGATACCACGCTTGTGGATCTTA
TGCTCACACAGCCATCCTCTAGTAAGTTTGTGAGACGTCTGGTGACGTGTGGGAACTTAT
TGGAAACAACATTTTGCTGTAAAGCATCCTATTGCCAGCGGATTAACACCTGGTAACAGG
```

Here, 1 is contig number, 7202 is k-mer length in base pairs and 271.608429 represents k-mer coverage for a given contig.

Velvet assembler failed to provide putative viral hits for five sequences (FFGC_VIR_1, 2, 3, 18, 20) out of 17 (shown in the Table 5.3). It also failed to identify the pathogen of unknown etiology (FFGC_VIR_10) (shown in bold).

Table 5.3 Final output for Velvet assembler
The table shows the k-mer value, number of contigs, length of the contig and putative viral hit for the given sequence.

Sequence	k-mer value	Number of contigs	Length of longest contig	Putative viral hits
FFGC_VIR_1	67	687	-	**No**
FFGC_VIR_2	67	574	-	**No**
FFGC_VIR_3	65	1248	-	**No**
FFGC_VIR_4	65	6	3326	EV-D68

FFGC_VIR_5	73	2	7384	E18
FFGC_VIR_6	71	1	7189	E18
FFGC_VIR_7	73	1	7199	E18
FFGC_VIR_8	73	16	7066	E6, E11
FFGC_VIR_9	71	2	7361	E13
FFGC_VIR_10	55	20	-	**No**
FFGC_VIR_11	61	11	6987	EV-D68
FFGC_VIR_13	71	6	7344	EV-D68
FFGC_VIR_18	59	28	-	**No**
FFGC_VIR_19	73	4	6719	PeV-A3
FFGC_VIR_20	71	11	-	**No**
FFGC_VIR_21	73	2	7276	PeV-A, PeV-A5, PeV-A4
FFGC_VIR_22	73	2	7274	PeV-A,PeV-A1, PeV-A3, PeV-A4

Taking into account longest contigs generated by Velvet, longest contig was observed for sequence FFGC_VIR_5, contig size 7384bp and smallest with FFGC_VIR_4, contig size 3326bp. Optimised k-mer values were in different ranges.

After completion of analysis with Velvet program, same sets of sequence reads were assembled with SPAdes.

5.2.3.2 SPAdes

SPAdes assembler was relatively easier to use and required less time than VelvetOptimiser. SPAdes does not use a single k-value (unlike Velvet) and a range k-mer value is selected automatically depending on the read length. Another assembly parameter used in SPAdes is ‘careful’ and is recommended for small genome assemblies. Parameter “careful” minimizes the errors produced by mismatches and indels. As picornaviruses have small genomes, this option was selected.

The final output files consisted of fasta files for final contigs and scaffolds. All output files and their respective uses are given in appendix B (Table B.3) Contigs/scaffolds names in SPAdes output FASTA files had the following format-

```
>NODE_1_length_7647_cov_1644.037408
CTCCTAGAGAGCTTGGCCGTTGGGCCTTATACCCCAACTTGCCGAGCTTCTCTAGGAGAG
TCCCTTTCCCAGCCCTGAGGCGGCTGGTTAATAAAAGCCTCAACTGTAACAAACATCTAA
GATTTTTACATTAACCCCATGCCTGGTCTCCACTAGTTGAAGGCAGCTTGCAATAAAATG
AGTGGGAACAAGACGCTTAAAGCATGGTGTAAATTAACTTTTCTAACTCACACTTTGTGT
GGGGTGGCAGGTGGCGTGCCATAATTCTATTAGTGAGATACCACGCTTGTGGATCTTATG
```

Here, 1, 7647 and 1644.037408 represent the number of the contig/scaffold, the sequence length and the k-mer coverage for the last (largest) k value used respectively.

The resultant scaffolds were in decreasing order in the length. This made identifying the putative viral hits moderately simple. Viral assembly results are shown in Table 5.4.

Table 5.4 Final output for SPAdes assembler
The table shows number of scaffolds, length of the longest scaffold and putative viral hit for the given sequence.

Sequence	Number of scaffolds	Length of longest scaffold	Putative viral hit
FFGC_VIR_1	13609	7213	E6, CVB2
FFGC_VIR_2	13309	5102	E18
FFGC_VIR_3	13333	7367	E18
FFGC_VIR_4	36	6904	EV-D68
FFGC_VIR_5	1099	1017	CVB4, E27
FFGC_VIR_6	329	1224	E18
FFGC_VIR_7	3966	-	**No**
FFGC_VIR_8	305	7424	E18
FFGC_VIR_9	14	7418	E13
FFGC_VIR_10	78	2277	E30, CVB1
FFGC_VIR_11	99	7341	EV-D68

FFGC_VIR_13	585	7474	EV-D68
FFGC_VIR_18	99	-	**No**
FFGC_VIR_19	840	5037	PeV-A3
FFGC_VIR_20	80	3092	PeV-A1, PeV-A3, PeV-A4
FFGC_VIR_21	6687	7340	PeV-A1
FFGC_VIR_22	7333	7647	PeV-A, PeV-A5, PeV-A4

SPAdes assembler failed to generate viral contigs for two sequences, FFGC_VIR_7 and FFGC_VIR_18 (shown in bold). Considering the length of longest scaffolds produced, longest scaffold size was observed for FFGC_VIR_22 (7647bp) and smallest size observed for FFGC_VIR_5 (1017bp).

The final results for A5 assembly, Velvet and SPAdes assemblers for successful picornavirus sequence reads assembly into longer contigs/scaffold is given in Table 5.5.

Table 5.5 Assemblers and their suitability for picornavirus genome
Table shows whether assemblers were useful in generating picornaviral contigs from the sequence sets

Sequence ID	A5 assembly	Velvet	SPAdes
FFGC_VIR_1_S104	Yes	**No**	Yes
FFGC_VIR_2_S105	Yes	**No**	Yes
FFGC_VIR_3_S106	Yes	**No**	Yes
FFGC_VIR_4_S107	Yes	Yes	Yes
FFGC_VIR_5_S108	Yes	Yes	Yes
FFGC_VIR_6_S109	Yes	Yes	Yes
FFGC_VIR_7_S110	Yes	Yes	**No**
FFGC_VIR_8_S115	Yes	Yes	Yes
FFGC_VIR_9_S114	Yes	Yes	Yes
FFGC_VIR_10_S113	Yes	**No**	Yes
FFGC_VIR_11_S112	Yes	Yes	Yes
FFGC_VIR_13_S111	Yes	Yes	Yes
FFGC_VIR_18_S116	**No**	**No**	**No**

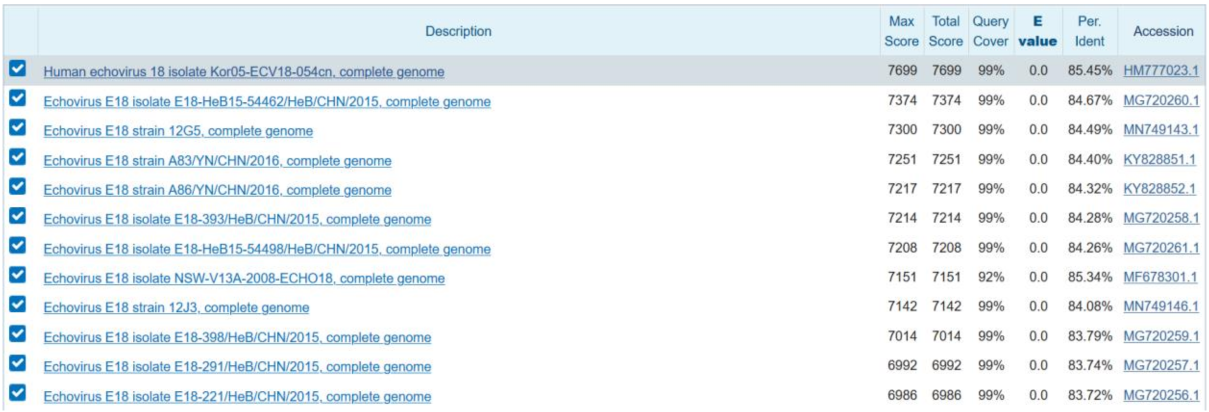
FFGC_VIR_19_S117	Yes	Yes	Yes
FFGC_VIR_20_S118	Yes	No	Yes
FFGC_VIR_21_S119	Yes	Yes	Yes
FFGC_VIR_22_S120	Yes	Yes	Yes

A5 assembly program was simplest to use, followed by Spades. VelvetOptimiser was the most computationally exhaustive program amongst three. As compared to VelvetOptimiser, A5 assembly and SPAdes assembly programs were quite successful in regards with successful virus genome assembly for number of sequences. One sequence, FFGC_VIR_18 failed to generate any viral contigs/scaffolds with each program.

The details of comparison between results of A5 pipeline, VelvetOptimiser and SPAdes for picornavirus sequence assembly is shown in appendix C, Table C.2

5.3 Quality analysis and validation of contigs/scaffolds

The final outputs i.e. final contigs/scaffolds generated by all assembly programs (A5 pipeline, VelvetOptimiser and SPAdes) were checked with BLAST for validation of viral hits.

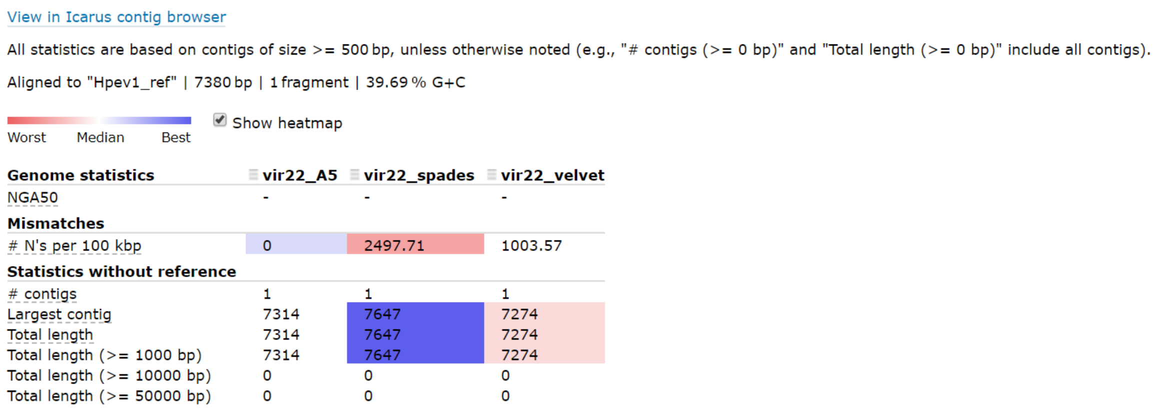


	Description	Max Score	Total Score	Query Cover	E value	Per. Ident	Accession
✓	Human echovirus 18 isolate Kor05-ECV18-054cn, complete genome	7699	7699	99%	0.0	85.45%	HM777023.1
✓	Echovirus E18 isolate E18-HeB15-54462/HeB/CHN/2015, complete genome	7374	7374	99%	0.0	84.67%	MG720260.1
✓	Echovirus E18 strain 12G5, complete genome	7300	7300	99%	0.0	84.49%	MN749143.1
✓	Echovirus E18 strain A83/YN/CHN/2016, complete genome	7251	7251	99%	0.0	84.40%	KY828851.1
✓	Echovirus E18 strain A86/YN/CHN/2016, complete genome	7217	7217	99%	0.0	84.32%	KY828852.1
✓	Echovirus E18 isolate E18-393/HeB/CHN/2015, complete genome	7214	7214	99%	0.0	84.28%	MG720258.1
✓	Echovirus E18 isolate E18-HeB15-54498/HeB/CHN/2015, complete genome	7208	7208	99%	0.0	84.26%	MG720261.1
✓	Echovirus E18 isolate NSW-V13A-2008-ECHO18, complete genome	7151	7151	92%	0.0	85.34%	MF678301.1
✓	Echovirus E18 strain 12J3, complete genome	7142	7142	99%	0.0	84.08%	MN749146.1
✓	Echovirus E18 isolate E18-398/HeB/CHN/2015, complete genome	7014	7014	99%	0.0	83.79%	MG720259.1
✓	Echovirus E18 isolate E18-291/HeB/CHN/2015, complete genome	6992	6992	99%	0.0	83.74%	MG720257.1
✓	Echovirus E18 isolate E18-221/HeB/CHN/2015, complete genome	6986	6986	99%	0.0	83.72%	MG720256.1

Figure 5.5 BLAST search for a scaffold.
Only few hits are shown here. The percent identity of more than 80% was selected for putative viral hits.

Although Chipster platform offers option of BLAST analysis, as number is limited to 10 jobs at a time, Blast searches were performed outside Chipster. The contigs/scaffolds identified with a viral hit, were then saved separately as a fasta file. These files were used for QUAST analysis.

Quality assessment with QUAST produced a number of output files. The final result was generated in pdf format and also available with Icarus viewer. The QUAST report consisted of table for genome statistics, mismatches and statistics without reference, with heatmap indicating worst and best statistics. The example output is shown as follows:



View in Icarus contig browser

All statistics are based on contigs of size >= 500 bp, unless otherwise noted (e.g., "# contigs (>= 0 bp)" and "Total length (>= 0 bp)" include all contigs).

Aligned to "Hpev1_ref" | 7380 bp | 1 fragment | 39.69 % G+C

Worst Median Best

✓ Show heatmap

Genome statistics	vir22_A5	vir22_spades	vir22_velvet
NGA50	-	-	-
Mismatches			
# N's per 100 kbp	0	2497.71	1003.57
Statistics without reference			
# contigs	1	1	1
Largest contig	7314	7647	7274
Total length	7314	7647	7274
Total length (>= 1000 bp)	7314	7647	7274
Total length (>= 10000 bp)	0	0	0
Total length (>= 50000 bp)	0	0	0

Figure 5.6 Sample QUAST report for sequence FFGC_VIR_22
Figure shows general statistics for contigs/scaffolds generated by all three assemblers, suggesting best assembly output by SPAdes in this scenario.

From the QUAST reports, overall quality of the assembly outputs for SPAdes was relatively better than other two programs.

From the results obtained from the analysis of different approaches (outlined before in Figure 4.1 Schematic representation of various bioinformatic approaches), a workflow is suggested for picornavirus genome assembly. As each assembler has its own pros and cons, using more than one program is helpful. Currently the workflow uses programs available in Chipster as well as outside Chipster. In future, with addition of *de novo* assembly programs in Chipster environment, one can perform all the tasks within Chipster.

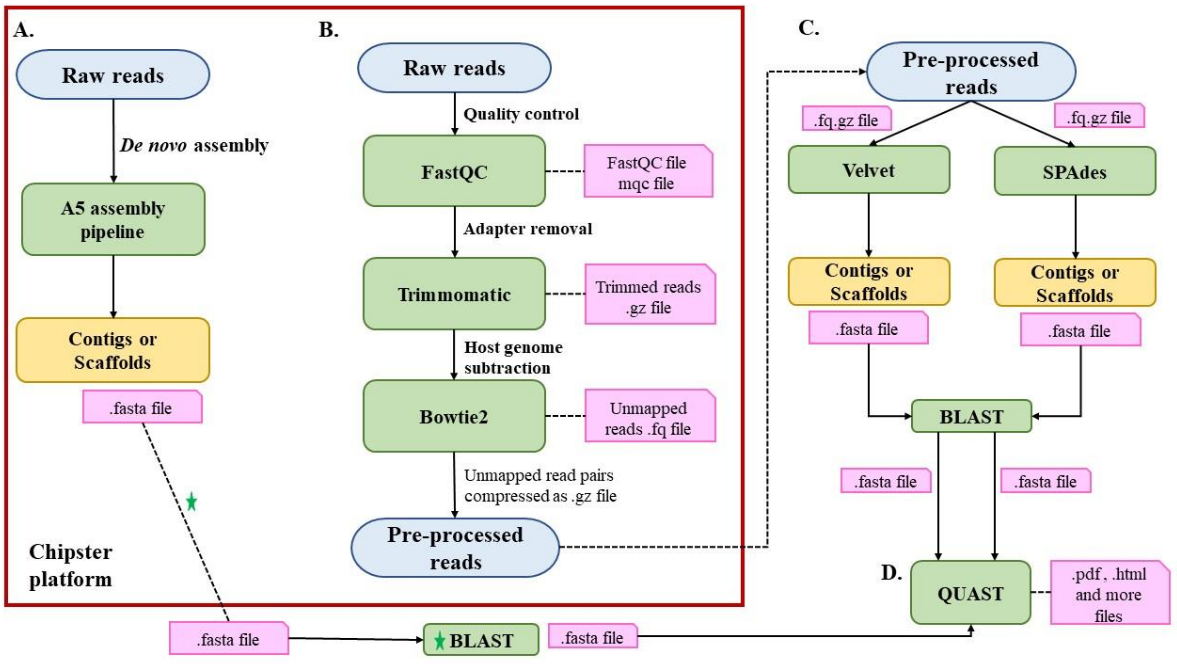


Figure 5.7 Schematic representation of workflow for picornavirus sequence analysis
The workflow can be divided into three parts. A. Represents preliminary analysis with default pipeline in Chipster. B. Provides stepwise guidance for approaches other than automated pipeline. A and B can be performed in Chipster (red box). C. Suggests the application of external *de novo* assembly programs. Validation of outputs from both A and C can be performed with BLAST and quality assessment with QUAST shown as D. Green star suggests that the step of BLAST analysis can also be done in Chipster.

6 Discussion

Picornaviridae is a large family of vertebrate viruses with small genome sizes that produce both clinically asymptomatic infections and mild and fatal disease. The genus Enterovirus comprises seven species infecting humans (enterovirus A-D and rhinovirus A-C). This genus contains poliovirus (PV), coxsackieviruses A/B (CVA/B), enteroviruses (EV), echoviruses (E), and rhinoviruses (RV). Serological classification has indicated existence of more serotypes. Hepatitis A virus (HAV) is the sole human-virus species in the genus Hepatovirus. Other human picornaviruses include members of the genera Cardiovirus (Saffold virus—SAFV), Cosavirus (CoSV), Parechovirus (Ljungan virus—LV), Kubovirus (Aichi virus—AiV), and Salivirus (Salivirus A—SaVA) (Nielsen et al., 2013).

Next generation sequencing (NGS) allied with bioinformatic analysis provides an efficient way to detect and identify known or novel micro-organisms. It helps to generate more accurate sequences than conventional methods. The genome sequence assembly and its downstream analysis help to determine etiological agents, identify and track outbreak origins and transmissions of a disease. Its efficiency along with affordable cost for application makes NGS an quality tool for research. NGS is a powerful, multifaceted tool, which also poses few challenges during its applications (Ekblom & Wolf, 2014; Lambert et al., 2018; Maljkovic Berry et al., 2020; Orton et al., 2016; Pereira et al., 2020; Sutton et al., 2019; White et al., 2017).

The outcome of NGS approaches is dependent on the sample quality and the amount of the target in it. While it is easy to determine the quantity of human DNA for NGS, it is much more difficult to determine viral load. In case of picornaviruses, the main criteria is viral copy number. In most cases this means the Ct value obtained in RT-qPCR diagnostic method. This relative value can be used to compare the sample to each other and to the sequencing results, that is, the number of viral sequences in the overall sequence pool and the quality of the sequences to obtain full genome-length contigs. Quality of reads and coverage depth play a deciding role in final genome assembly. Both these factors affect the correctness and overall completeness of assembled genome. This quality will also be dependent on the heuristics applied for the assembly. Reference-based assembly and *de novo* assembly are two commonly used approaches for genome assembly. Reference-based assembly provides an efficient tool for assembling known genomes in time sensitive manner and in limited computational environment. In order to obtain accurate reference mapping, a closely related reference genome should be selected for genome of interest. Sample reads are mapped to reference genome depending on the best match and alignment. BWA (Burrows wheeler transform), BWA-MEM (maximal exact matches) (Li & Durbin, 2009) and Bowtie2 (Langmead & Salzberg, 2013) are the most commonly used tools for reference-mapping (Lambert et al., 2018; H. C. Lee et al., 2012; Nooij et al., 2018). However, reference-mapping can pose difficulties when assembling a novel organism or variable organisms, like RNA viruses, which show high genetic diversity owing to their high mutation rates and recombination tendencies (Kasibhatla et al., 2016; Orton et al., 2016). In general, RNA viruses also contain conserved and non-conserved regions that pose a problem for the function of reference-mapping techniques. The actual parameters or percentage difference range where Refmap methods work optimally await to be determined.

De novo assembly is useful in absence of a quality reference genome and is commonly used to assemble and identify novel organisms by means of metagenomics. It involves connecting short overlapping sequence reads into longer contigs or scaffolds. Developments in sequencing technologies have led to advancements in the algorithms for assemblers used. For example, the overlap layout consensus (OLC) algorithm (Pevzner et al., 2001) was quite successful with capillary reads (sequencing by capillary electrophoresis, Karger and Guttman, 2009), as it produced high quality genome assemblies. However, use of this algorithm with short reads assembly was found to be computationally expensive (Zerbino, 2010). This led to the development of assemblers based on the de Bruijn graph, such as ALLPATHS (Butler et al., 2008), ABySS (Simpson et al., 2009), SOAPdenovo (Li et al., 2009) and Velvet (Zerbino and Birney, 2008), which comparatively reduced computational time. Some of the frequently used *de novo* assemblers are MIRA, Velvet, ABySS, SOAPdenovo and SPAdes (Blawid et al., 2017; Ibrahim et al., 2018; H. C. Lee et al., 2012; White et al., 2017).

Although most of the NGS programs are command-line tools, they can be accessed with commercial platforms (for example, Geneious and CLC Genomics workbench), open source graphical user interfaces or web-based tools (such as Galaxy platform and EDGE) or command-line based pipelines (like ngs_mapper and GATK). However, the limited computational skills is one of the bottlenecks faced during NGS analysis by non-bioinformaticians. Accessing command line tools through user friendly interfaces provides a way for researchers, who do not have any technical knowledge, to perform NGS related bioinformatics analysis successfully (Lambert et al., 2018; Maljkovic Berry et al., 2020).

Chipster (http://chipster.csc.fi/) at Centre for Scientific Computing (Espoo, Finland) is a platform, which allows user to explore multiple tools at a single access. Chipster brings a powerful collection of data analysis methods including NGS within the reach of bioscientists via its intuitive graphical user interface (Kallio et al., 2011). It can be used a Java-web start desktop application (Chipster v3.16.3) or a web-application through browser (Chipster v4). It is an open source and easy-to-use analysis software, allows user to save the analysis steps as a workflow and share those workflows with the peers. Chipster handles different file formats easily and has a provision for converting files to necessary and needful formats (for example, indexing FASTA files, compressing files to .tar etc). Handling of data through servers and cloud system provides sufficient memory for computation and storage capacity. However, Chipster does not have any independent *de novo* assembler within its environment, unlike other platforms like Galaxy and Geneious, which offer options such as Velvet, SOAPdenovo, Spades, MIRA, ABySS and IDBA-UD. It also does not have any quality assessment tool to compare different assembler outputs. Although Chipster allows addition of tools, it will be more helpful if it has some options to start with.

In the current study, Chipster was used for picornavirus genome sequence analysis. The workflows included primary analysis of picornavirus sequence data with automated pipelines (VirusDetect and A5) within Chipster, and if approaches other than automated pipelines were used, the pre-processing of the reads followed by subsequent assembly of these trimmed reads with reference-mapping or *de novo* methods were used. As Chipster does not have independent *de novo* tools, external programs Velvet and SPAdes were used.

Although reference-mapping and application of VirusDetect pipeline was not successful, one should consider the reasons behind this lack of success. Reference mapping is based on aligning the query reads with known viral genome. RNA viruses are known for their high evolution rate and low fidelity of RNA-dependent RNA polymerase (RdRp), which can lead to the viral sequence diversity. VirusDetect pipeline was originally developed for identifying viruses based on small RNAs, which might act as a limiting factor for its success in picornavirus analysis.

A5 pipeline for microbial genome assembly was found to be quite successful for picornavirus identification. With automated steps, A5-miseq could produce nearly complete genome assemblies. A5-miseq offered automated adapter trimming, reconstruction of reads into longer sequences, outputs which could be readily uploaded to NCBI and with quality reports (.qvl files). Being automated and computationally efficient, A5-miseq would be helpful for researcher with limited computational and bioinformatics experience. Hence, A5 pipeline can be used as initial identification and contig/scaffold generation method for picornaviruses.

Quality analysis helps to screen sequence reads for any quality issues and flags the reads with questionable quality. Trimming of reads removes primers and adapters and also filters the reads with ambiguous lengths. RNA viruses pose difficulty in NGS due to their relatively low abundance as compared to host sequence. Hence, it is necessary to subtract host sequences from the total sequence reads. Successful application of these pre-processing steps to sequence datasets can facilitate viral genome assembly (Daly et al., 2015; Guo et al., 2013; Kruppa et al., 2018; Marston et al., 2013; Montmayeur et al., 2017). The steps related to quality control, sequence trimming and cleaning, alignments to subtract host genome sequence can be easily performed in the Chipster environment. The final trimmed sequences then can be exported and be used for assembly programs.

Velvet (or VelvetOptimiser) and SPAdes, both were successfully used for sequence assembly. However, application of Velvet poses two challenges. First, Velvet utilises quite a lot of memory, hence, without adequate memory it either slows down, or stops working completely. Second, Velvet attempts to connect contigs into scaffolds by default. It tries to scaffold contigs which cannot quite connect and adds the N's sequences in the estimated gap between these neighbouring contigs. This results in stretches of Ns in the contigs.fa file output. SPAdes performed well, was simple to employ and generated both contigs and scaffolds.

Final assemblies by SPAdes and A5-miseq have been reported to be competitive with their other counterparts (Magoc et al., 2013; Lambert et al., 2018), while most of the times producing more promising results. This was found true for the current data analysis. All three assemblers could produce near-full length contigs/scaffolds for most of the picornavirus sequences. The exception was sequence, FFGC_VIR_18, which failed to generate sequence assembly with all three programs. This sequence, along with sequences which produced comparatively smaller contigs/scaffolds were observed to have warning for overrepresented sequences in FastQC analysis report.

QUAST assessed and compared the assemblies generated by different programs and helped to understand their efficiency.

To summarise, next generation sequencing (NGS) have provided a breakthrough for the viral genomics and bioinformatics. Applications of NGS in virology include analysis of intra-host viral diversity, study of quasispecies, viral transmission and resistance to antiviral therapy and vaccines. It is considered as a powerful tool for virus discovery and metagenomic with its ability to detect and identify known and emerging viruses with or without *a priori* knowledge. Although various tools and algorithms are available in the market, these tools differ in regards with ease of use, speed, quality control, output format and affordability (Ekblom & Wolf, 2014; Orton et al., 2016; White et al., 2017). The selection of parameter settings determines the performance and sensitivity of the bioinformatic tools. The choice of the tools and assemblers depend on the source and biological relevance of the data. Assessing the impact of different workflows and parameter settings can affect the quality of the assembled genome. Combination of different assemblies can help in establishing the optimized workflow. Different assemblers are based different algorithms and differ in their accuracy, speed, computational skills and requirements. Hence, it is judicious to get acquainted with the various algorithms available and then selecting a particular, suitable approach (Caicedo-Montoya et al., 2019; Datta, 2015).

Finally, Chipster with its comprehensive collection of analysis tools can provide a way to efficiently perform pre-processing of data and initial analysis with the help of A5 assembly pipeline for picornaviruses. Alternatively, external assemblers can be used. The quality of the sequences defines which method is the most optimal for designated virus. For this purpose, one needs to compare the sequences to larger pool of NGS sequences of the same type. Inclusion of the external assembler within Chipster environment can help to establish a straightforward and user-friendly workflow for next generation sequencing of picornaviruses.

7 References

Altschul, S. F., Gish, W., Miller, W., Myers, E. W., & Lipman, D. J. (1990). Basic local alignment search tool. *Journal of Molecular Biology*, *215*(3), 403–410. https://doi.org/https://doi.org/10.1016/S0022-2836(05)80360-2

Bankevich, A., Nurk, S., Antipov, D., Gurevich, A. A., Dvorkin, M., Kulikov, A. S., Lesin, V. M., Nikolenko, S. I., Pham, S., Prjibelski, A. D., Pyshkin, A. V., Sirotkin, A. V., Vyahhi, N., Tesler, G., Alekseyev, M. A., & Pevzner, P. A. (2012). SPAdes: A new genome assembly algorithm and its applications to single-cell sequencing. *Journal of Computational Biology*, *19*(5), 455–477. https://doi.org/10.1089/cmb.2012.0021

Blawid, R., Silva, J. M. F., & Nagata, T. (2017). Discovering and sequencing new plant viral genomes by next-generation sequencing: description of a practical pipeline. *Annals of Applied Biology*, *170*(3), 301–314. https://doi.org/10.1111/aab.12345

Bolger, A. M., Lohse, M., & Usadel, B. (2014). Trimmomatic: A flexible trimmer for Illumina sequence data. *Bioinformatics*, *30*(15), 2114–2120. https://doi.org/10.1093/bioinformatics/btu170

Boros, Á., Nemes, C., Pankovics, P., Kapusinszky, B., Delwart, E., & Reuter, G. (2012). Identification and complete genome characterization of a novel picornavirus in Turkey (Meleagris gallopavo). *Journal of General Virology*, *93*(PART 10), 2171–2182. https://doi.org/10.1099/vir.0.043224-0

Caicedo-Montoya, C., Pinilla, L., Toro, L. F., Yepes-García, J., & Ríos-Estepa, R. (2019). Comparative analysis of strategies for de novo transcriptome assembly in prokaryotes: Streptomyces clavuligerus as a case study. *High-Throughput*, *8*(4). https://doi.org/10.3390/ht8040020

Chevreux, B. (2018). *Sequence assembly and mapping with MIRA 5 The Definitive Guide*.

Chun, S., Muthu, M., Gopal, J., Paul, D., Kim, D. H., Gansukh, E., & Anthonydhason, V. (2018). The unequivocal preponderance of biocomputation in clinical virology. *RSC Advances*, *8*(31), 17334–17345. https://doi.org/10.1039/c8ra00888d

Cifuente, J. O., & Moratorio, G. (2019). Evolutionary and Structural Overview of Human Picornavirus Capsid Antibody Evasion. *Frontiers in Cellular and Infection Microbiology*, *9*(August), 1–11. https://doi.org/10.3389/fcimb.2019.00283

Coil, D., Jospin, G., & Darling, A. E. (2015). A5-miseq: An updated pipeline to assemble microbial genomes from Illumina MiSeq data. *Bioinformatics*, *31*(4), 587–589. https://doi.org/10.1093/bioinformatics/btu661

Daly, G. M., Leggett, R. M., Rowe, W., Stubbs, S., Wilkinson, M., Ramirez-Gonzalez, R. H., Caccamo, M., Bernal, W., & Heeney, J. L. (2015). Host subtraction, filtering and assembly validations for novel viral discovery using next generation sequencing data. *PLoS ONE*, *10*(6), 1–28. https://doi.org/10.1371/journal.pone.0129059

Datta, S. (2015). Next-generation sequencing in clinical virology: Discovery of new viruses. *World Journal of Virology*, *4*(3), 265. https://doi.org/10.5501/wjv.v4.i3.265

Ekblom, R., & Wolf, J. B. W. (2014). A field guide to whole-genome sequencing, assembly and annotation. *Evolutionary Applications*, *7*(9), 1026–1042. https://doi.org/10.1111/eva.12178

Goya, S., Valinotto, L. E., Tittarelli, E., Rojo, G. L., Nabaes Jodar, M. S., Greninger, A. L., Zaiat, J. J., Marti, M. A., Mistchenko, A. S., & Viegas, M. (2018). An optimized methodology for whole genome sequencing of RNA respiratory viruses from nasopharyngeal aspirates. *PLoS ONE*, *13*(6), 1–15. https://doi.org/10.1371/journal.pone.0199714

Guo, Y., Ye, F., Sheng, Q., Clark, T., & Samuels, D. C. (2013). Three-stage quality control strategies for DNA re-sequencing data. *Briefings in Bioinformatics*, *15*(6), 879–889. https://doi.org/10.1093/bib/bbt069

Gurevich, A., Saveliev, V., Vyahhi, N., & Tesler, G. (2013). QUAST: Quality assessment tool for genome assemblies. *Bioinformatics*, *29*(8), 1072–1075. https://doi.org/10.1093/bioinformatics/btt086

Hodzic, J., Gurbeta, L., Omanovic-Miklicanin, E., & Badnjevic, A. (2017). Overview of Next-generation Sequencing Platforms Used in Published Draft Plant Genomes in Light of Genotypization of Immortelle Plant (Helichrysium Arenarium). *Medical Archives (Sarajevo, Bosnia and Herzegovina)*, *71*(4), 288–292. https://doi.org/10.5455/medarh.2017.71.288-292

Huang, B., Jennsion, A., Whiley, D., McMahon, J., Hewitson, G., Graham, R., De Jong, A., & Warrilow, D. (2019). Illumina sequencing of clinical samples for virus detection in a public health laboratory. *Scientific Reports*, *9*(1), 1–8. https://doi.org/10.1038/s41598-019-41830-w

Ibrahim, B., McMahon, D. P., Hufsky, F., Beer, M., Deng, L., Mercier, P. Le, Palmarini, M., Thiel, V., & Marz, M. (2018). A new era of virus bioinformatics. *Virus Research*, *251*, 86–90. https://doi.org/10.1016/j.virusres.2018.05.009

J, S., & G, S. (2016). Next Generation Sequencing-Current Status. *Journal of Next Generation Sequencing & Applications*, *3*(1), 1–2. https://doi.org/10.4172/2469-

9853.1000e107

Joffret, M.-L., Polston, P. M., Razafindratsimandresy, R., Bessaud, M., Heraud, J.-M., & Delpeyroux, F. (2018). Whole Genome Sequencing of Enteroviruses Species A to D by High-Throughput Sequencing: Application for Viral Mixtures. *Frontiers in Microbiology*, *9*(September), 1–10. https://doi.org/10.3389/fmicb.2018.02339

Kallio, M. A., Tuimala, J. T., Hupponen, T., Klemelä, P., Gentile, M., Scheinin, I., Koski, M., Käki, J., & Korpelainen, E. I. (2011). Chipster: User-friendly analysis software for microarray and other high-throughput data. *BMC Genomics*, *12*(1), 507. https://doi.org/10.1186/1471-2164-12-507

Kasibhatla, S. M., Waman, V. P., & Kale, Mohan M. and Kulkarni-Kale, U. (2016). Analysis of Next-generation Sequencing Data in Virology - Opportunities and Challenges. *Intech*, *i*(tourism), 13. https://doi.org/http://dx.doi.org/10.5772/57353

Kerkvliet, J., Edukulla, R., & Rodriguez, M. (2010). Novel Roles of the Picornaviral 3D Polymerase in Viral Pathogenesis. *Advances in Virology*, *2010*. https://doi.org/10.1155/2010/368068

Kremer, F. S., McBride, A. J. A., & Pinto, L. da S. (2017). Approaches for in silico finishing of microbial genome sequences. *Genetics and Molecular Biology*, *40*(3), 553–576. https://doi.org/10.1590/1678-4685-gmb-2016-0230

Kruppa, J., Jo, W. K., van der Vries, E., Ludlow, M., Osterhaus, A., Baumgaertner, W., & Jung, K. (2018). Virus detection in high-throughput sequencing data without a reference genome of the host. *Infection, Genetics and Evolution*, *66*, 180–187. https://doi.org/https://doi.org/10.1016/j.meegid.2018.09.026

Kulski, J. K. (2016). Next-Generation Sequencing — An Overview of the History, Tools, and “Omic” Applications. In J. K. Kulski (Ed.), *Next Generation Sequencing*. IntechOpen. https://doi.org/10.5772/61964

Lambert, C., Braxton, C., Charlebois, R. L., Deyati, A., Duncan, P., La Neve, F., Malicki, H. D., Ribrioux, S., Rozelle, D. K., Michaels, B., Sun, W., Yang, Z., & Khan, A. S. (2018). Considerations for optimization of high-throughput sequencing bioinformatics pipelines for virus detection. *Viruses*, *10*(10). https://doi.org/10.3390/v10100528

Langmead, B., & Salzberg, S. (2013). Bowtie2. *Nature Methods*, *9*(4), 357–359. https://doi.org/10.1038/nmeth.1923.Fast

Lee, H. C., Lai, K., Lorenc, M. T., Imelfort, M., Duran, C., & Edwards, D. (2012). Bioinformatics tools and databases for analysis of next-generation sequence data. *Briefings in Functional Genomics*, *11*(1), 12–24. https://doi.org/10.1093/bfgp/elr037

Lee, W. P., Stromberg, M. P., Ward, A., Stewart, C., Garrison, E. P., & Marth, G. T. (2014). MOSAIK: A hash-based algorithm for accurate next-generation sequencing short-read mapping. *PLoS ONE*, *9*(3). https://doi.org/10.1371/journal.pone.0090581

Li, H., & Durbin, R. (2009). Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics*, *25*(14), 1754–1760. https://doi.org/10.1093/bioinformatics/btp324

Lim, C. S., & Brown, C. M. (2018). Know your enemy: Successful bioinformatic approaches to predict functional RNA structures in viral RNAs. *Frontiers in Microbiology*, *8*(JAN). https://doi.org/10.3389/fmicb.2017.02582

Lischer, H. E. L., & Shimizu, K. K. (2017). Reference-guided de novo assembly approach improves genome reconstruction for related species. *BMC Bioinformatics*, *18*(1), 1–12. https://doi.org/10.1186/s12859-017-1911-6

Liu, L., Li, Y., Li, S., Hu, N., He, Y., Ray, P., Lin, D., Lu, L., & and Law, M. (2012). Comparison ofNext-Generation Sequencing Systems Lin. *Journal of Biomedicine and Biotechnology*, *2012*, 11. https://doi.org/10.1201/b16568

Luo, R., Liu, B., Xie, Y., Li, Z., Huang, W., Yuan, J., He, G., Chen, Y., Pan, Q., Liu, Y., Tang, J., Wu, G., Zhang, H., Shi, Y., Liu, Y., Yu, C., Wang, B., Lu, Y., Han, C., … Wang, J. (2012). SOAPdenovo2: An empirically improved memory-efficient short-read de novo assembler. *GigaScience*, *1*, 1. https://doi.org/10.1186/s13742-015-0069-2

Madden, T. (2013). The BLAST sequence analysis tool. *The BLAST Sequence Analysis Tool*, *Md*, 1–17. http://www.ncbi.nlm.nih.gov/books/NBK153387/

Maljkovic Berry, I., Melendrez, M. C., Bishop-Lilly, K. A., Rutvisuttinunt, W., Pollett, S., Talundzic, E., Morton, L., & Jarman, R. G. (2020). Next Generation Sequencing and Bioinformatics Methodologies for Infectious Disease Research and Public Health: Approaches, Applications, and Considerations for Development of Laboratory Capacity. *The Journal of Infectious Diseases*, *221*(3), S292–S307. https://doi.org/10.1093/infdis/jiz286

Marston, D. A., McElhinney, L. M., Ellis, R. J., Horton, D. L., Wise, E. L., Leech, S. L., David, D., de Lamballerie, X., & Fooks, A. R. (2013). Next generation sequencing of viral RNA genomes. *BMC Genomics*, *14*(1). https://doi.org/10.1186/1471-2164-14-444

Martin, M. (2011). Cutadapt Removes Adapter Sequences From High-Throughput Sequencing Reads. *EMBnet.Journal*, *17*(1), 10–12.

Metzker, M. L. (2010). Sequencing technologies the next generation. *Nature Reviews Genetics*, *11*(1), 31–46. https://doi.org/10.1038/nrg2626

Montmayeur, A. M., Ng, T. F. F., Schmidt, A., Zhao, K., Magaña, L., Iber, J., Castro, C. J., Chen, Q., Henderson, E., Ramos, E., Shaw, J., Tatusov, R. L., Dybdahl-Sissoko, N., Endegue-Zanga, M. C., Adeniji, J. A., Oberste, M. S., & Burns, C. C. (2017). High-throughput next-generation sequencing of polioviruses. *Journal of Clinical Microbiology*, *55*(2), 606–615. https://doi.org/10.1128/JCM.02121-16

Nielsen, A. C. Y., Gyhrs, M. L., Nielsen, L. P., Pedersen, C., & Böttiger, B. (2013). Gastroenteritis and the novel picornaviruses aichi virus, cosavirus, saffold virus, and salivirus in young children. *Journal of Clinical Virology*, *57*(3), 239–242. https://doi.org/https://doi.org/10.1016/j.jcv.2013.03.015

Nooij, S., Schmitz, D., Vennema, H., Kroneman, A., & Koopmans, M. P. G. (2018). Overview of virus metagenomic classification methods and their biological applications. *Frontiers in Microbiology*, *9*(APR). https://doi.org/10.3389/fmicb.2018.00749

Orton, R. J., Gu, Q., Hughes, J., Maabar, M., Modha, S., Vattipally, S. B., Wilkie, G. S., & Davison, A. J. (2016). Bioinformatics tools for analysing viral genomic data. *OIE Revue Scientifique et Technique*, *35*(1), 271–285. https://doi.org/10.20506/rst.35.1.2432

Peng, Y., Leung, H. C. M., Yiu, S. M., & Chin, F. Y. L. (2012). IDBA-UD: A de novo assembler for single-cell and metagenomic sequencing data with highly uneven depth. *Bioinformatics*, *28*(11), 1420–1428. https://doi.org/10.1093/bioinformatics/bts174

Pereira, R., Oliveira, J., & Sousa, M. (2020). Bioinformatics and Computational Tools for Next-Generation Sequencing Analysis in Clinical Genetics. *Journal of Clinical Medicine*, *9*(1), 132. https://doi.org/10.3390/jcm9010132

Posada-Cespedes, S., Seifert, D., & Beerenwinkel, N. (2017). Recent advances in inferring viral diversity from high-throughput sequencing data. *Virus Research*, *239*, 17–32. https://doi.org/10.1016/j.virusres.2016.09.016

Radford, A. D., Chapman, D., Dixon, L., Chantrey, J., Darby, A. C., & Hall, N. (2012). Application of next-generation sequencing technologies in virology. *Journal of General Virology*, *93*, 1853–1868. https://doi.org/10.1099/vir.0.043182-0

Santti, J., Raija, V., & Hyypiä, T. (1999). Molecular detection and typing of human picornaviruses. *Virus Research*, *62*, 177–183.

Simmonds, P., Gorbalenya, A. E., Harvala, H., Hovi, T., Knowles, N. J., Lindberg, A. M., Oberste, M. S., Palmenberg, A. C., Reuter, G., Skern, T., Tapparel, C., Wolthers, K. C., Woo, P. C. Y., & Zell, R. (2020). Recommendations for the nomenclature of enteroviruses and rhinoviruses. *Archives of Virology*, *0123456789*. https://doi.org/10.1007/s00705-019-04520-6

Song, G., Lee, J., Kim, J., Kang, S., Lee, H., Kwon, D., Lee, D., Lang, G. I., Michael Cherry, J., & Kim, J. (2019). Integrative meta-assembly pipeline (IMAP): Chromosome-level genome assembler combining multiple de novo assemblies. *PLoS ONE*, *14*(8), 1–15. https://doi.org/10.1371/journal.pone.0221858

Sutton, T. D. S., Clooney, A. G., Ryan, F. J., Ross, R. P., & Hill, C. (2019). Choice of assembly software has a critical impact on virome characterisation. *Microbiome*, *7*(1), 1–15. https://doi.org/10.1186/s40168-019-0626-5

van der Linden, L., Wolthers, K. C., & van Kuppeveld, F. J. M. (2015). Replication and inhibitors of enteroviruses and parechoviruses. *Viruses*, *7*(8), 4529–4562. https://doi.org/10.3390/v7082832

White, D. J., Wang, J., & Hall, R. J. (2017). Assessing the impact of assemblers on virus detection in a de novo metagenomic analysis pipeline. *Journal of Computational Biology*, *24*(9), 874–881. https://doi.org/10.1089/cmb.2017.0008

Wolthers, K. C., Susi, P., Jochmans, D., Koskinen, J., Landt, O., Sanchez, N., Palm, K., Neyts, J., & Butcher, S. J. (2019). Progress in human picornavirus research: New findings from the AIROPico consortium. *Antiviral Research*, *161*(November 2018), 100–107. https://doi.org/10.1016/j.antiviral.2018.11.010

Yin-Murphy, M., & Almond, J. W. (1996). *Chapter 53 Picornaviruses*.

Zell, R. (2018). Picornaviridae—the ever-growing virus family. *Archives of Virology*, *163*(2), 299–317. https://doi.org/10.1007/s00705-017-3614-8

Zerbino, D. R. (2010). Using Velvet *de novo* assembler for short-reading sequence technologies. *Current Protocols Bioinformatics*, *31*(11), 1–13. https://doi.org/10.1002/0471250953.bi1105s31.Using

Zerbino, D. R., & Birney, E. (2008). Velvet: Algorithms for de novo short read assembly using de Bruijn graphs. *Genome Research*, *18*(5), 821–829. https://doi.org/10.1101/gr.074492.107

Zheng, Y., Gao, S., Padmanabhan, C., Li, R., Galvez, M., Gutierrez, D., Fuentes, S., Ling, K. S., Kreuze, J., & Fei, Z. (2017). VirusDetect: An automated pipeline for efficient virus discovery using deep sequencing of small RNAs. *Virology*, *500*(August 2016), 130–138. https://doi.org/10.1016/j.virol.2016.10.017

8 Acknowledgements

To work sincerely and to best of one’s ability is the imperative for a student. But work without any proper direction can be in vain. Hence, I would like to express my sincere gratitude to Dr. Petri Susi, for acting as my supervisor for this thesis and his continued guidance, support and useful critiques throughout the study. I am deeply grateful to Dr. Martti Tolvanen for his insights and enthusiastic encouragement throughout the duration of the studies. Discussions with Petri and Martti have always been illuminating and beneficial.

I would like to thank Eero Hietanen for his valuable inputs and help in thesis work. I warmly thank Juho Heimonen for his guidance and support throughout the MSc program. I would also like to thank CSC-Chipster team for their suggestions.

Finally, I would like to thank my family and friends for their unwavering support, encouragement and confidence in me, which has helped me throughout the journey of my life.

Appendix A. Next generation sequencing

Table A.1 Glossary for NGS

Term	Meaning
Adapters	Short sequence-specific oligos ligated to the 5′and 3′ ends of each DNA fragment in a sequence library
Alignment	Matching sequencing reads to a reference genome
Burrows-Wheeler transformation (BWT)	BWT is a method of indexing (and compressing) a reference genome into a graph data structure of overlapping substrings, known as a suffix tree. It requires a single computational effort to build this graph for a particular reference genome, then it can be stored and reused when mapping multiple NGS data sets to this genome. The BWT method is particularly efficient when the data contain runs of repeated sequences
Consensus sequence	When two or more DNA sequences are aligned, the overlapping portions can be combined to create a single consensus sequence. In positions where all overlapping sequences have the same base (a single column of the multiple alignment), that base becomes the consensus.
Contigs	A contiguous linear stretch of DNA or RNA consensus sequence, *in silico*, constructed by aligning a number of smaller overlapping sequencing reads
Coverage (depth/level)	The average number of sequenced bases that align to known reference bases
de Bruijn graph (DBG)	This is a graph theory method for assembling a long sequence (like a genome) from overlapping fragments (like sequence reads). The de Bruijn graph is a set of unique substrings (words) of a fixed length (a *k*-mer) that contain all possible words in the data set exactly once.
De novo sequencing	Sequencing of genetic material with no reference sequence available
DNA fragment	A small piece of DNA, often produced by a physical or chemical shearing of larger DNA molecules.
Illumina sequencing	The NGS sequencing method developed by the Solexa company, then acquired by Illumina Inc.
k-mer	Short, unique element of DNA sequence of length k, used by many assembly algorithms

Library	Collection of DNA (or RNA) fragments modified in a way that is appropriate for downstream analyses, such as high-throughput sequencing in this case
Mapping	A term routinely used to describe alignment of short sequence reads to a longer reference sequence
N50	A statistic of a set of contigs (or scaffolds). It is defined as the length for which the collection of all contigs of that length or longer contains at least half of the total of the lengths of the contigs.
Next-generation sequencing (NGS)	DNA sequencing technologies that simultaneously determine the sequence of DNA bases from many thousands (or millions) of DNA templates in a single biochemical reaction volume.
Oligonucleotide (oligo)	A short DNA or RNA sequence
Paired-end sequencing	A process of sequencing from both ends of a DNA fragment in the same run
Quality score	A metric in NGS that predicts or estimates the probability of an error in base calling
Read	Data output (Short base-pair sequence inferred) from the analysis of a single fragment or sequence of DNA/RNA template by sequencing
Reference genome	A fully sequenced and assembled genome; A curated consensus sequence for all of the DNA in the genome (all of the chromosomes) of a species of organism.
Scaffold	Two or more contigs joined together using read-pair information
Sequence assembly	Computational reconstruction of a longer sequence from smaller sequence reads by finding overlaps of identical (or nearly identical) strings of letters among a set of sequence fragments and iteratively joining them together
Throughput	The amount of data produced by a next-generation sequencing instrument

Table A.2 Common file formats used in NGS analysis

File format	Description
FASTA	Text file representing nucleotide or protein sequences

FASTQ*	Text file storing raw sequence and its corresponding quality score for each nucleotide in ASCII code
SAM	Output of short-read sequence aligners, storage of sequence alignments and their mapping coordinates
BAM	Binary form of SAM
BED	Used for viewing alignments in a genome browser as annotation track.

* typically, *gzip* compressed, extension. *fastq.gz* or *fq.gz*

SAM- Sequence Alignment/Map

BAM- Binary Alignment/Map

BED- Browser Extensible Data

Table A.3 Commercial platforms available for bioinformatic analysis

Platform	**Link**	**Application platform**	**Cost**
CLC Genomics Workbench	http://www.clcbio.com/download	Command line interface, Graphical user interface	Paid
Galaxy	https://usegalaxy.org/	Web-based	Free but Registration required
Geneious	https://www.geneious.com/	Graphical user interface	Paid

Table A.4 Various pipelines and tools available for virus NGS analysis

Tool	**Reference**	**Source**	**Purpose**	**Platform**
ABySS	Simpson et al, 2009	https://codeload.github.com/bcgsc/abyss/tar.gz/2.0.2	*De novo* assembly of short reads	Command line interface
ALLPATHS-LG	Gnerre et al, 2010	http://software.broadinstitute.org/allpaths-lg/blog/	*De novo* assembly of short reads	Command line interface

drVM	Hsin-Hung Lin and Yu-Chieh Liao, 2017	https://sourceforge.net/projects/sb2nhri/files/drVM/	Rapid viral read identification, , *de novo* assembly	Command line interface, Graphical user interface
GLUE	Singer et al, 2018	http://glue-tools.cvr.gla.ac.uk/#/home	A flexible software system support for rapid development of virus sequence data	Command line interface
IDBA-UD	Peng et al, 2012	http://www.cs.hku.hk/~alse/idba_ud	*De novo* assembler for single-cell and metagenomic sequencing data	Command line interface
IVA	Hunt et al, 2015	http://sanger-pathogens.github.io/iva	*De novo* assembler designed to assemble virus genomes that have no repeat sequences	Command line interface
Kaiju	Menzel et al, 2016	http://kaiju.binf.ku.dk	Taxonomic classification of metagenomic NGS reads	Command line interface
Kraken	Davis et al, 2013	http://www.ebi.ac.uk/research/enright/software/kraken	Suite of tools for quality control and	Command line interface

			analysis of NGS datasets	
Metavir 2	Roux et al, 2014	http://metavir-meb.univ-bpclermont.fr	Viral metagenome comparison and assembled virome analysis	Web user interface
MIRA	Chevreux, 2018	http://mira-assembler.sourceforge.net/docs/DefinitiveGuideToMIRA.html	*De novo* assembly of short reads	Command line interface
MOSAIK	Lee et al, 2014	http://gkno.me; https://github.com/wanpinglee/MOSAIK/wiki	Reference-guided aligner	Command line interface
PANDAseq	Masella et al, 2012	https://github.com/neufeld/pandaseq	Assembles paired-end reads	Command line interface
PRICE	Ruby et al, 2013	derisilab.ucsf.edu/software/price/ or sourceforge.net/projects/pricedenovo/	*De novo* assembly of short gun metagenomic data	Command line interface
RIEMS	Scheuch et al, 2015	No working link available	Taxonomic classification of all individual reads in metagenomics sequence datasets	Command line interface
RINS	Bhaduri et al, 2012	http://khavarilab.stanford.edu/tools-1#tools	Workflow for identification of non-human sequences in	Command line interface

			metagenomic datasets	
SOAPdenovo 2	Luo R et al, 2012	https://github.com/aquaskyline/SOAPdenovo2	*De novo* assembly of short reads	Command line interface
SPAdes	Bankevich et al, 2013	http://cab.spbu.ru/files/release3.10.1/manual.html	*De novo* assembly of short reads and single cell assembler	Command line interface
SSAKE	Warren et al, 2007	http://www.bcgsc.ca/bioinfo/software/ssake	*De novo* assembly of short reads	Command line interface
SURPI	Naccache et al, 2014	https://github.com/chiulab/surpi	Pipeline for pathogen identification from complex metagenomic NGS data from clinical samples	Command line interface
Trinity	Grabherr et al, 2011	https://github.com/trinityrnaseq/trinityrnaseq/wiki	*De novo* transcriptomes reconstruction from RNA-seq data	Command line interface
Velvet	Zerbino and Birney, 2008	https://www.ebi.ac.uk/~zerbino/velvet/	*De novo* assembly of short reads	Command line interface
V-GAP	Nakamura et al, 2015	No link available	An automated pipeline for *de novo* assembly and rapid	-

			genome typing of viruses	
VICUNA	Yang et al, 2012	http://www.broadinstitute.org/scientific-community/science/projects/viral-genomics/viral-genomics-analysis-software	*De novo* assembly	Command line interface
VIP	Li et al, 2016	https://github.com/keylabivdc/VIP	Virus identification and discovery metagenomic NGS data	Command line interface
Vipie	Lin J et al, 2017	https://binf.uta.fi/vipie; https://sourceforge.net/projects/vipie/	*De novo* assembly, taxonomic classification of viruses as well as sample analyses	Web user interface, Registration required
VirFinder2	Wang Q et al, 2015	https://bioinfo.uth.edu/VirusFinder/	Intra-host viruses in NGS data	Command line interface
VIROME	Wommack et al, 2012	http://virome.dbi.udel.edu/	Bioinformatics pipeline for metagenomic analysis	Web user interface, Free registration required
VirSorter	Roux et al, 2015	https://github.com/simroux/VirSorter	Designed and optimized for detection of bacterial and archaeal viruses	Web user interface

VirusHunter	Zhao G et al, 2013	http://www.ibridgenetwork.org/wustl/virushunter	Detects both novel and known sequence of microbial origin	Command line interface
VirusSeeker	Zhao et al, 2017	https://github.com/guoyanzhao/VirusSeeker-Virome	Pipeline for both novel virus discovery and virome composition analysis	Command line interface
VirusSeq	Chen et al, 2013	http://odin.mdacc.tmc.edu/~xsu1/VirusSeq.html	Detecting known viruses and their integration sites in the human genome using NGS data	Command line interface
VirusTAP	Yamashita et al, 2016	https://gph.niid.go.jp/cgi-bin/virustap/index.cgi/.	Assembles virus genomes from NGS reads	Web user interface (Firefox only)
Vy-PER	Michael Forster et al, 2015	https://www.ikmb.uni-kiel.de/vy-per/	Detection of virus integration	Command line interface

drVM-detect and reconstruct known viral genomes from metagenomes

GLUE-Genes Linked by Underlying Evolution

IVA-Iterative Virus Assembler

NGS-next generation sequencing

PRICE- paired-read iterative contig extension

RIEMS-Reliable Information Extraction from Metagenomic Sequence datasets

RINS-Rapid identification of non-human sequences

SURPI-sequence-based ultra-rapid pathogen identification

V-GAP-Viral genome assembly pipeline

VIP-Virus Identification Pipeline

VIROME- Viral Informatics Resource for Metagenome Exploration

Virus TAP-Virus genome-Targeted Assembly Pipeline

Vy-PER-Virus integration detection bY Paired End Reads

Appendix B. Software and their output files

Chipster

Java- web start desktop application (Chipster v3.16.3)

A5 assembly pipeline

Table B.1 A5 assembly output files

a5_assembly.contigs.fasta	Contigs
a5_assembly.final.scaffolds.fasta	Final scaffolds (checked for misassemblies, and re-scaffolded)
a5_assembly.final.scaffolds.fastq	Final scaffolds in FastQ format with base call qualities
a5_assembly.final.scaffolds.qvl	Quality values for the final scaffolds in QVL format for submission to NCBI
a5_assembly_stats.csv	A tab-separated file with assembly summary statistics

VelvetOptimiser

VelvetOptimiser-2.2.5

Dependencies:

- Velvet => 0.7.51
- Perl => 5.8.8
- BioPerl >= 1.4
- GNU utilities: grep sed free cut

Table B.2 VelvetOptimiser output files and contents

Contigs.fa	Fasta file with sequences of the contigs
Graph	a textual representation of the contig graph
Graph2	Detailed representation of the de Bruijn graph
Log	commands you typed to get this assembly result, for reproducing results later on
PreGraph	--

Sequences	Sequences used as input
stats	Simple tab-limited description of the nodes

SPAdes

SPAdes requires a 64-bit Linux system or Mac OS and Python (supported versions are Python2: 2.4–2.7, and Python3: 3.2 and higher) to be pre-installed on it.

Table B.3 SPAdes output

SPAdes stored all output files in the output directory specified at the start of the run.

corrected/	directory contains reads corrected by BayesHammer in fastq.gz files;
scaffolds.fasta	contains resulting scaffolds
contigs.fasta	contains resulting contigs
assembly_graph.gfa	contains SPAdes assembly graph and scaffolds paths in GFA 1.0 format
assembly_graph.fastg	contains SPAdes assembly graph in FASTG format
contigs.paths	contains paths in the assembly graph corresponding to contigs.fasta
scaffolds.paths	contains paths in the assembly graph corresponding to scaffolds.fasta
params.txt	information about SPAdes parameters in this run
spades.log	SPAdes log
dataset.info	internal configuration file
input_dataset.yaml	internal YAML data set file

QUAST

QUAST can be run on Linux (64-bit and 32-bit with slightly limited functionality) or macOS (OS X).

Its default pipeline requires:

- Python2 (2.5 or higher) or Python3 (3.3 or higher)
- GCC 4.7 or higher
- Perl 5.6.0 or higher

- GNU make and ar
- zlib development files

Table B.4 QUAST output files

report.txt	assessment summary in plain text format
report.tsv	tab-separated version of the summary
report.tex	LaTeX version of the summary
icarus.html	Icarus main menu with links to interactive viewers
report.pdf	all other plots combined with all tables
report.html	HTML version of the report with interactive plots
contigs_reports/	generated only if a reference genome is provided

Appendix C. Supplementary tables for results

Table C.1 Virus names and their abbreviations

The following table includes virus nomenclature as per recommended by Simmonds et al., 2020

Virus name	Abbreviation of virus name
Coxsackievirus A9	CVA9
Coxsackievirus B1	CVB1
Coxsackievirus B2	CVB2
Coxsackievirus B3	CVB3
Coxsackievirus B4	CVB4
Coxsackievirus B5	CVB5
Echovirus E13	E13
Echovirus E18	E18
Echovirus E27	E27
Echovirus E30	E30
Echovirus E30	E30
Echovirus E6	E6
Echovirus E9	E9
Enterovirus D68	EV-D68
Hu-Parechovirus 1	PeV-A1
Hu-Parechovirus 3	PeV-A3
Hu-Parechovirus 4	PeV-A4
Hu-Parechovirus 5	PeV-A5

Table C.2 Comparison of final assembly outputs for A5, Velvet and SPAdes programs

The table informs about sequence and virus it was typed for (Sanger typed), longest scaffold produced by all three assembly programs and their respective BLASTN viral hits.

Sequence	**Typed as**	**A5**		**Velvet**		**Spades**	
		Length	Putative viral hit	Length	Putative viral hit	Length	Putative viral hit
FFGC_VIR_1	EV-D68_19	6950	E18	None		7213	E6, CVB2
FFGC_VIR_2	EV-D68_21	5918	E18	None		5102	E18
FFGC_VIR_3	EV-D68_23	7356	E18	None		7367	E18
FFGC_VIR_4	EV-D68Fermon (type strain).	7376	EV-D68	3420	EV-D68	6904	EV-D68
		7107	E18				

FFGC_VIR_5	A2269 (E-18)	7135	E18	7383	E18	1017	CVB4, E27
FFGC_VIR_6	A2326 (E-18)	7458	E18	7189	E18	1224	E18
FFGC_VIR_7	A2262 (E-18)	7401	E18	7310	E18	None	
FFGC_VIR_8	A2283 (E-18)	7409	E18	7273	E18	7424	E18
FFGC_VIR_9	B3612 (E-13)	5774	E13	7309	E13	7418	E13
FFGC_VIR_10	1S AFP etiology by unknown pathogen	775	EV-B, CVB3, CVB4	None		2277	E30, CVB1
						2223	EV-D68
FFGC_VIR_11	CB103 (EV-D68)	7339	EV-D68	6994	EV-D68	7341	EV-D68
FFGC_VIR_13	TU44 (EV-D68)	7340	EV-D68	7344	EV-D68	7474	EV-D68
FFGC_VIR_18	PeV-A3/Vi988	None		None		None	
FFGC_VIR_19	PeV-A3/152037	6749	PeV-A3	6719	PeV-A3	5034	PeV-A3
		4344	PeV-A4	4132	PeV-A, PeV-A4	1428	PeV-A3, PeV-A4
FFGC_VIR_20	PeV-A3/145-8	3053	PeV-A1, PeV-A3, PeV-A4	None		3092	PeV-A1, PeV-A3, PeV-A4
FFGC_VIR_21	PeV-A1/101-17	7335	PeV-A1	7274	PeV-A, PeV-A1, PeV-A3, PeV-A4	7340	PeV-A1
FFGC_VIR_22	PeV-A1/103-2	7314	PeV-A1	7276	PeV-A, PeV-A1, PeV-A5, PeV-A4	7647	PeV-A, PeV-A5, PeV-A4