

ANALYSIS OF THE EPSTEIN-BARR VIRUS LYTIC TRANSCRIPTOME  
USING HIGH-THROUGHPUT SEQUENCING METHODS


AN ABSTRACT

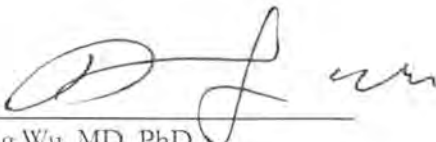
SUBMITTED ON THE FOURTH DAY OF MARCH 2016  
TO THE DEPARTMENT OF PATHOLOGY  
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS  
OF THE GRADUATE SCHOOL OF TULANE UNIVERSITY  
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY


BY

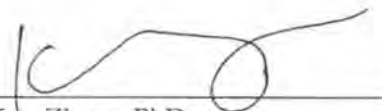
  
CHRISTINA MARIE O'GRADY


APPROVED BY:

  
Erik K. Flemington, PhD  
PhD Advisor

  
Tong Wu, MD, PhD

  
Victoria Belancio, PhD

  
Kun Zhang, PhD

  
Prescott Deininger, PhD

## **ABSTRACT**

The lytic replicative cascade in Epstein-Barr virus and other herpesviruses has traditionally been understood to involve the ordered expression of sets of protein-coding genes. Recent experiments using tiling microarrays and next-generation sequencing, however, have indicated extensive transcription outside of known coding regions. In this study, strand-specific Illumina RNA-Seq reveals abundant antisense and intergenic transcription of the EBV genome during lytic replication. Both polyadenylated and non-polyadenylated transcripts are shown to arise from nearly the entire genome. However, the complex and overlapping nature of these transcripts confounds attempts to resolve their structures with short-read RNA-Seq alone. In order to resolve the structures of polyadenylated transcripts on a global level, the Transcriptome Resolution by Integration of Multi-platform Data (TRIMD) method was developed. This method combines data from Pacific Biosciences long-read SMRT sequencing (Iso-Seq protocol) with data from Illumina RNA-Seq and deepCAGE. Using TRIMD we identify nearly 300 unannotated transcripts in replicating Epstein-Barr virus. These transcripts illustrate multiple strategies by which the virus achieves its remarkable level of transcript diversity, including alternative promoter usage, alternative splicing, readthrough transcription and intergenic splicing. The TRIMD method is simple and flexible, and scripts have been developed to facilitate its application to other genomes.

ANALYSIS OF THE EPSTEIN-BARR VIRUS LYTIC TRANSCRIPTOME  
USING HIGH-THROUGHPUT SEQUENCING METHODS

A DISSERTATION

SUBMITTED ON THE FOURTH DAY OF MARCH 2016

TO THE DEPARTMENT OF PATHOLOGY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS


OF THE GRADUATE SCHOOL OF TULANE UNIVERSITY


FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

BY

  
CHRISTINA MARIE O'GRADY

APPROVED BY:

  
Erik K. Flemington, PhD  
PhD Advisor

  
Tong Wu, MD, PhD

  
Victoria Belancio, PhD

  
Kun Zhang, PhD

  
Prescott Deininger, PhD



## **ACKNOWLEDGMENT**

I am profoundly grateful to my mentor, Dr. Erik Flemington. He is an insightful, astute and accomplished researcher who is nevertheless endlessly supportive of his students. I consider myself extremely lucky to have found a home in his lab. Also, I couldn't ask for a more intelligent, creative, capable group of scientists to model myself after than my committee: Drs. Victoria Belancio, Prescott Deininger, Tong Wu and Kun Zhang. I'm delighted to have worked with the smart, friendly and funny members of the Flemington Lab: Melody Baddoo, Subing Cao, Monica Concha, Tom Laskow, Zhen Lin, Hani Nakhoul, Claire Roberts, Mike Strong, Xia Wang, Qinyan Yin and Yi Yu. Likewise I have learned so much from my friends and colleagues in the BMS Program, especially our Study Group: Emily Carron, Van Hoang, Grace Jairo, Nicki Kikendall, Narae Lee, Courtney Standlee and Manish Ranjan. I am grateful that much of my time as a PhD student has been supported by the National Cancer Institute in the form of a Ruth L. Kirschstein NRSA F31 fellowship. I am indebted to faculty members I have learned from at The Catholic University of America, especially Drs. Venigalla Rao, James Greene and Michael Mullins, and the University of New Orleans, especially Drs. Carla Penz and Phil DeVries, for encouraging me to pursue a PhD and a career in research. Finally, I am thankful for my friends and family near and far. Lindsay Falke, Leah Levkowicz, Gerard Mangusso, Matt Morrin and the Purple Lightning troublemakers of Burgundy St. for filling my non-lab hours and acting appropriately impressed when I talk about science. Mom and Dad and aunts and uncles for providing an environment of both curiosity and diligence. And of course my boys, Phillip and Sid.

## TABLE OF CONTENTS

ACKNOWLEDGMENT	.....	ii
LIST OF TABLES	.....	iv
LIST OF FIGURES	.....	v
CHAPTER 1: Introduction	.....	1
CHAPTER 2: Global investigation of unannotated lytic EBV transcription	.....	10
CHAPTER 3: Global determination of novel EBV transcript structures	.....	29
CHAPTER 4: Development of an algorithm to automate transcript structure determination and annotation	.....	52
CHAPTER 5: Materials and methods	.....	62
CHAPTER 6: Discussion and future directions	.....	81
LIST OF REFERENCES	.....	91
APPENDICES		
Appendix 1: TRIMD_start_validator.pl	.....	104
Appendix 2: TRIMD_junction_validator.pl	.....	125
Appendix 3: TRIMD_end_validator.pl	.....	136
Appendix 4: TRIMD_transcript_validator.pl	.....	154
Appendix 5: TRIMD_README.txt	.....	169
Appendix 6: Validated EBV transcription start sites	.....	177
Appendix 7: Validated EBV splice junctions	.....	185
Appendix 8: Validated EBV polyadenylation sites	.....	192
Appendix 9: Validated novel EBV transcripts	.....	195
Appendix 10: Updated EBV-Akata annotation	.....	205

## LIST OF TABLES

I. Latency programs in EBV	.....	3
II. RNA-Seq in uninduced and induced Akata cells	.....	13
III. RNA-Seq using ribodepleted RNA in a reactivation time course	.....	13
IV: 5' and 3' RACE Primers	.....	75
V: RT-PCR primers	.....	77
VI. cDNA and PCR primers for strand-specific qRT- PCR	.....	78

## LIST OF FIGURES

1: Reactivation and sequencing strategy	12
2: RNA-Seq sense and antisense read coverage of cellular genes	14
3: RNA-Seq read coverage of the Akata EBV genome at 24 hours postinduction	15
4: Novel transcription of the Akata genome 24 hours postinduction	16
5: Antisense transcription at known genes and gene classes	17
6: Sense and antisense transcription at the EBNA2 locus	19
7: Sense and antisense transcription at the EBNA3A locus	20
8: Sense and antisense transcription at the EBNA3B locus	21
9: A novel antisense ORF in the EBNA3C locus	23
10: Genomic location of Stellaris FISH probes	24
11: Sequence motif conservation in EBNA antisense transcripts	25
12: Splice junctions in the EBNA3A region	26
13: Splice junctions at 24 hours postinduction	28
14: Long-read sequence mapping data and transcript identification and validation strategy	31
15: Length distribution of Iso-Seq CFLs from different size fractions	32



16: CFLs mapping to the cellular gene RNF167	.....	33
17: Validation of EBV transcript features	.....	35
18: Annotated EBV 5' start sites validated by TRIMD	.....	36
19: Start sites identified by deepCAGE	.....	37
20: Novel validated viral transcripts	.....	39
21: Novel intergenic transcripts	.....	41
22: Readthrough transcription and intergenic splicing at the BZLF2 locus	.....	43
23: Validated transcripts at the EBNA loci	.....	45
24: Programmed exon skipping in the W repeat region	.....	48
25: Complex lytic promoter usage for LMP2 transcripts	.....	51
26: Strand specific qRT-PCR	.....	79
27: TRIMD-validated BRLF1 and BZLF1 isoforms	.....	84
28: Length distribution of detected and predicted polyadenylated EBV isoforms	.....	88

## **CHAPTER 1**

### Introduction

Epstein-Barr Virus (EBV) is a ubiquitous human herpesvirus that causes disease worldwide. EBV was first discovered in 1964 in a cell line derived from a Burkitt lymphoma (BL) tumor in an African patient<sup>1</sup>. Since then links have been established between EBV and a number of other cancers, including Hodgkin lymphoma<sup>2</sup>, nasopharyngeal carcinoma<sup>3</sup> and gastric cancers<sup>4</sup>.

Like most other herpesviruses, EBV can exist in both a latent and a lytic phase. On initial infection of a new host EBV productively infects epithelial cells in the oropharynx, using the lytic cascade to produce new virus particles<sup>5</sup>. These newly replicated virions then establish a latent infection in oropharyngeal B cells<sup>6,7</sup>. The latent infection commences with a gene expression program known as type III latency, in which eight viral proteins and several viral noncoding RNAs are expressed (Table I)<sup>8</sup>. EBV-infected cells in latency phase III are detectable by the human immune system and largely eliminated, however infected cells that enter the latency 0 program do not express viral proteins and can escape immune control<sup>9</sup>. Latent infection of some B cells generally persists for the life of the host. Latently infected B cells *in vivo* are largely resistant to entering a phase of productive EBV replication, however lytic reactivation in B cells does occur and is an essential component of the virus shedding that can occur in the saliva of even asymptomatic EBV carriers<sup>10,11</sup>.

**Table I. Latency programs in EBV**

<b>Latency Program</b>	<b>Viral Proteins Expressed</b>	<b>Viral ncRNAs expressed</b>
<b>III</b>	EBNA1, EBNA2, EBNA3A, EBNA3B, EBNA3C, EBNA-LP, LMP1, LMP2	EBER1, EBER2, BARTs
<b>II</b>	LMP1, LMP2, EBNA1	EBER1, EBER2, BARTs
<b>I</b>	EBNA1	EBER1, EBER2, BARTs
<b>0</b>	EBNA1 (only during cell division)	EBER1, EBER2

### **The EBV lytic cascade**

When lytic reactivation is triggered, e.g. by B-cell differentiation or by acute stress<sup>12</sup>, the immediate early (IE) genes BZLF1 and BRLF1 are expressed to produce the proteins Zta (also known as ZEBRA<sup>13</sup>, and EB1<sup>14</sup>) and Rta. These transactivators function together to activate downstream genes in the lytic cascade<sup>15</sup>; Zta by binding to AP1-like sites in the promoters of both viral and cellular genes<sup>16</sup>, Rta by binding to Rta-responsive elements in promoters<sup>17,18</sup> and by inducing cellular phosphatidylinositol-3 (PI3) kinase signaling<sup>19</sup>. Many of the early (E) viral genes activated by Zta and Rta function in host cell control and viral DNA replication: for example the exonuclease BGLF5 and the DNA polymerase BALF5<sup>20,21</sup>. After viral DNA replication, the late (L) genes are transcribed by cellular RNA Polymerase II under the control of the viral pre-initiation complex<sup>22</sup>. The IE, E and L genes overlap temporally in expression, however viral DNA replication is required for L gene expression and this class of genes can be inhibited using the viral DNA polymerase inhibitor phosphonoacetic acid (PAA)<sup>23,24</sup>. Many L genes are involved in virion packaging and assembly, such as the packaging protein BDRF1, the capsid protein BDLF1 and the envelope glycoprotein BLLF1<sup>24</sup>.

## Pervasive transcription in herpesviruses

Our current understanding of lytic reactivation and viral replication is largely based on the ordered synthesis of mRNA transcripts that are translated into proteins, which in turn function both to carry out the necessary processes of virus production and to regulate the transcriptional cascade itself. Recently though, an even more complicated view of herpesviral lytic replication has begun to emerge as studies have revealed extensive lytic transcription that is not accounted for by currently annotated open reading frames.

An early indication that the herpesvirus lytic transcriptome is much larger than expected came from a large-scale cDNA cloning project using the betaherpesvirus Human Cytomegalovirus (HCMV)<sup>25</sup>. This study examined *de novo* HCMV infection of fibroblasts, during which time the virus is actively replicating, and detected extensive transcription of putative intergenic regions and a large number of transcripts antisense to known open reading frames. A subsequent study using Illumina RNA-Seq confirmed these findings<sup>26</sup>, with both studies noting that much of the novel transcription likely produces noncoding RNA. When ribosomal profiling was applied to the same system it was found that the HCMV proteome as well as the transcriptome was much larger than previously appreciated, with novel proteins being produced from unannotated short open reading frames as well as from truncated or frameshifted variations of known open reading frames<sup>27</sup>.

Pervasive transcription of the gammaherpesvirus Murine gammaherpesvirus 68 (MHV68) during productive infection of fibroblasts was first detected using a high-density tiling microarray<sup>28</sup>. The same method later showed similar widespread transcription of unannotated regions during viral reactivation in a latently infected B-cell line<sup>29</sup>. Several of

these transcripts were shown to be functional by targeting them for expression knockdown with antisense oligonucleotides, which resulted in changed expression levels of a viral protein<sup>30</sup>.

Abundant antisense and intergenic transcription has also been identified during replication in the gammaherpesvirus Kaposi sarcoma-associated herpesvirus (KSHV), first by tiling microarray in *de novo* infection of endothelial cells<sup>31</sup> and reactivated B cells<sup>32</sup>, then by RNA-Seq after lytic reactivation in an engineered epithelial cell line<sup>33</sup>. As seen in HCMV, the KSHV proteome is also significantly more complex than traditional annotation indicates, as revealed by both ribosomal profiling<sup>33</sup> and liquid chromatography/tandem-mass spectrometry (LC-MS/MS)<sup>32</sup>.

In EBV, Dresang *et al.* used tiling microarray and LC-MS/MS to identify novel peptides and transcripts antisense to annotated open reading frames after induction of a B-cell line coinfecting with EBV and KSHV<sup>32</sup>. Concha *et al.* used RNA-seq to investigate transcription during viral reactivation in Akata cells and found intergenic viral transcripts and complex splicing<sup>34</sup>. Furthermore, Concha's 5' and 3' RACE experiments in a transcribed intergenic region revealed bidirectional transcription. These findings provided the first indication that transcription of the EBV genome may be more complex than previously thought.

### **The EBV genome and transcriptome**

EBV's double-stranded DNA genome is approximately 171 kilobases long and densely packed with open reading frames, with over 70% of the genome annotated as being transcribed in at least one direction<sup>35</sup>. The genome is linear when packaged in the viral

capsid<sup>36</sup> and circularizes upon infection to persist within the host cell as a stable episome<sup>37</sup>, typically in multiple copies. Each end of the linear genome consists of repeat sequence known as the terminal repeats: these terminal repeat ends are joined to form the circular chromosome<sup>38</sup>. In GenBank and other sequence repositories EBV genome sequences are typically represented in their linear form, with terminal repeat sequences at each end. However, the LMP2 gene spans the terminal repeats on the circular genome<sup>39</sup>. A linear representation of the EBV genome split at the terminal repeats impedes analysis of LMP2 RNA sequence data and so this study and others use a representation of the EBV genome split between the BBRF3 and BGLF3 genes, a region of relatively low transcription<sup>34</sup>.

Initial EBV genome sequencing was based on the common laboratory strain B95-8<sup>40</sup>. As this strain contains a large genomic deletion a composite reference sequence was later developed by integrating sequence from another strain, Raji<sup>41,42</sup>. Subsequently, the genomic sequences of many more EBV strains have been determined and made available<sup>43-53</sup>. Transcripts have been annotated based on a mix of empirical evidence (e.g., Northern blots and Sanger sequencing of cDNA) and sequence analysis (i.e., the presence of open reading frames, TATA boxes and polyadenylation signals)<sup>35,40</sup>. Regions of the EBV genome are described in terms of BamHI restriction fragments and most transcript names reflect this: e.g. the transcripts BZLF1 (BamHI fragment Z, Leftward reading Frame 1) and BHRF1 (BamHI fragment H, Rightward reading Frame 1)<sup>40</sup>.

At the genomic sequence level strains of EBV differ from each other largely based on variations in the latency genes<sup>53</sup>. In particular, variations in EBNA2 and the EBNA3A/B/C genes separate EBV strains into type I and type 2 groups (also known as types A and

B)<sup>47,54-56</sup>. Different virus strains are associated with different geographical regions, have different phenotypic properties and may have different disease-causing properties<sup>53</sup>.

### **EBV-infected cell lines**

Many EBV-positive cell lines exist, including naturally infected lines derived from cancers, experimentally infected lymphoblastoid cell lines (LCLs) or epithelial cell lines, and cell lines carrying virus artificially engineered to exhibit particular properties. Data used in this dissertation is derived from naturally infected B-cell lines and LCLs, as described below.

The Akata cell line is an EBV-positive B-cell line that was established from a Burkitt lymphoma tumor recovered from a 4-year-old Japanese girl<sup>57</sup>. Unlike the virus strains present in many cell lines, the Akata strain does not contain large genomic deletions relative to other EBV strains<sup>43,57</sup>. Akata cells generally express a type I latency program (see Table I), but can be synchronously induced to produce infectious virus by crosslinking the B-cell receptors (BCR) with anti-IgG antibodies<sup>57-59</sup>. The full genome sequence of the Akata strain, a type 1 strain of the virus is available<sup>43</sup>.

Similarly, Mutu cells exist in latency type I and can be induced to lytic replication by BCR crosslinking, in this case using anti-IgM antibodies<sup>60</sup>. The Mutu cell line was established from a biopsy from a Burkitt lymphoma patient in a region of Kenya with a high level of endemic EBV-positive BL<sup>60</sup>. Mutu is a type 1 strain of the virus, and its complete genome sequence is available<sup>43</sup>. The AG876 cell line was also established from a Burkitt lymphoma tumor, in this case from an 8-year-old boy in Ghana<sup>61</sup>. Unlike the Akata and Mutu cell lines, the EBV strain in AG876 cells is type 2. The genomic sequence AG876 strain is available<sup>45</sup>.



While the Akata, Mutu and AG876 cell lines were derived from naturally EBV-infected tumors, many other cell lines have been created by *in vitro* EBV infection. The common laboratory strain B95-8 was created by infecting tamarin (*Saguinus oedipus*) B cells with EBV that was isolated from a cell line derived from a patient with mononucleosis<sup>62,63</sup>. B95-8 cells exist in type III latency with a relatively high level of spontaneous reactivation (up to 5% of cells)<sup>64,65</sup>. For many years B95-8 was the predominant laboratory strain because of this constitutive virus production<sup>66</sup>.

The B95-8 strain of EBV is frequently used to immortalize primary B cells and create lymphoblastoid cell lines that generally persist in type III latency<sup>67</sup>. Many lymphoblastoid cell lines have been established and used in both EBV and cellular research, including by large-scale efforts such as HapMap, ENCODE and the 1000 Genomes Project<sup>68-70</sup>. Though usually originally generated with cellular analysis in mind, cell lines and the associated data generated by these projects can also be used to gain insight into EBV and its interactions with host cells<sup>71</sup>. The cell lines JY, X50-7, GM12878, GM12892 and GM12891, used in Chapter 3, are lymphoblastoid cell lines immortalized with B95-8<sup>66,68,72</sup>.

## Specific Aims

The experiments and analysis described in this dissertation are guided by three specific aims:

*Aim 1:* Globally investigate unannotated lytic EBV transcription

*Aim 2:* Globally determine novel EBV transcript structures

*Aim 3:* Develop an algorithm to automate transcript structure determination and annotation

The discovery of antisense and intergenic transcription in EBV, combined with indications of widespread unannotated transcription in related herpesviruses, indicated the need for detailed global investigation of unannotated transcription of the EBV genome during viral replication (Aim 1). This investigation, undertaken primarily using strand-specific Illumina short-read sequencing, is reported in Chapter 2 and a recent Journal of Virology publication<sup>73</sup>. In order to globally determine the structures of novel EBV transcripts (Aim 2), a new method was developed using data from Pacific Biosciences Iso-Seq long-read sequencing<sup>74</sup>, Illumina short read sequencing and deepCAGE<sup>75</sup>. This method, Transcript Resolution by Integration of Multi-platform Data (TRIMD) is described in Chapter 3. The efficacy of this method for structure resolution in the complex lytic transcriptome of EBV suggests its usefulness for annotating other gene-dense genomes. To this end, flexible, user-friendly scripts were developed that can be used to apply TRIMD to other genomes (Aim 3). The TRIMD software is described in Chapter 4 and reproduced in Appendices 1-5 and is freely available at available at <https://github.com/flemingtonlab/public>.

## **CHAPTER 2**

Global investigation of unannotated lytic EBV transcription

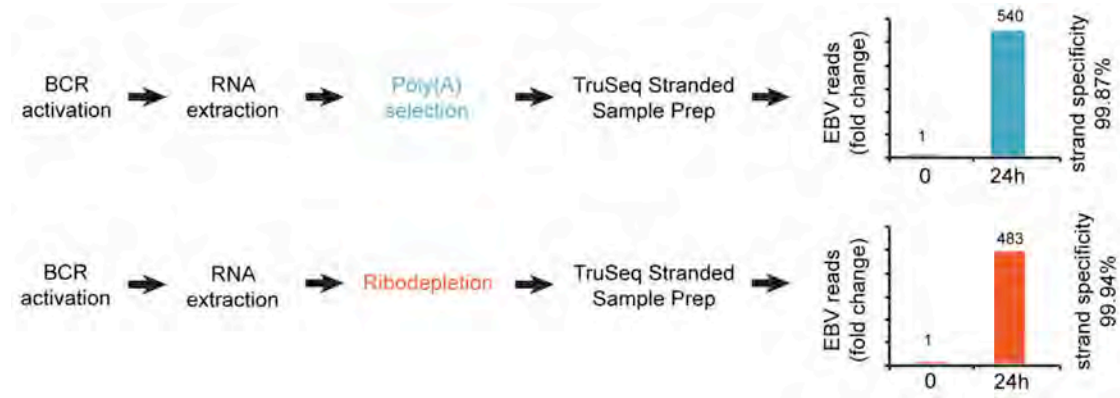
The discovery by Concha *et al.*<sup>34</sup> of complex bidirectional transcription in an intergenic region of the EBV genome raised the intriguing possibility of further bidirectional transcription upon lytic reactivation of EBV. To explore this possibility we took advantage of strand-specific Illumina RNA-Seq technology to globally investigate transcription of the EBV genome during viral replication. Much of the data in this chapter is reported in the publication<sup>73</sup>

O'Grady *et al.* (2014) Global bidirectional transcription of the Epstein-Barr virus genome during reactivation. *Journal of Virology* 88, 1604-1616.

### **Strand-specific RNA-Seq of the lytic EBV transcriptome**

To investigate the lytic transcriptome of EBV we used the Akata cell line, inducing lytic reactivation by crosslinking the B-cell receptors with an anti-IgG antibody. RNA was extracted from cells for RNA-Seq before and after BCR crosslinking. We subjected the RNA to poly(A)-selection to enrich for mRNA and polyadenylated noncoding RNA. Additionally, we extracted RNA and subjected it to ribodepletion in order to remove ribosomal RNA but retain other cellular and viral transcripts that are not polyadenylated, as functional transcripts without poly(A) tails are known in both eukaryotic cells and viruses<sup>76,77</sup>. RNA samples were prepared for sequencing using Illumina TruSeq Stranded sample prep kits to allow assignment of sequence reads to the DNA strand from which the transcript arose. Samples were sequenced on an Illumina HiSeq 2000 instrument and the reads mapped to the human (hg19 assembly) and EBV Akata (KC207813.1<sup>43</sup>) genomes. The dramatic increase in reads

mapping to the EBV genome after BCR crosslinking indicated robust induction of viral lytic reactivation 24 hours after BCR crosslinking (Figure 1 and Tables II & III).



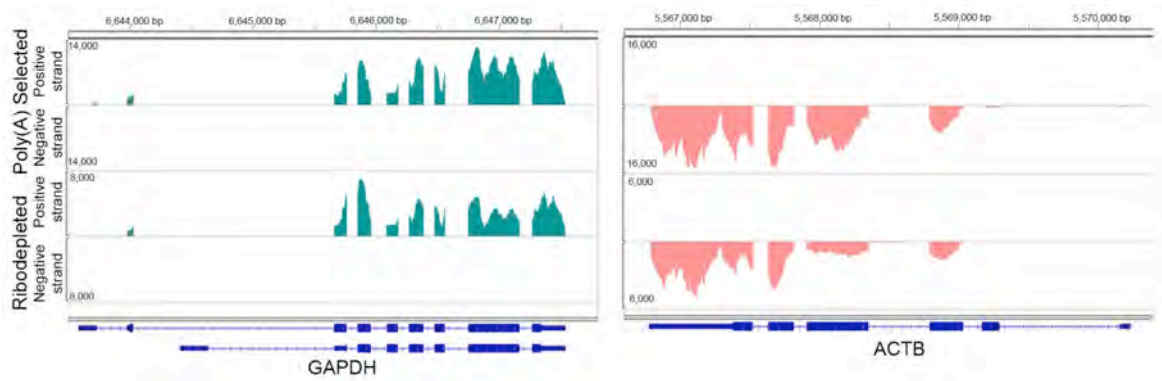
**Figure 1.** Reactivation and sequencing strategy. Column charts represent fold change in the number of RNA-Seq reads mapped to the EBV genome, excluding EBER regions, at 0 and 24 hours. For the calculation of strand specificity, see Materials and Methods.

As accurate assignment of RNA-Seq reads to the appropriate genomic strand is crucial to the analysis of antisense transcription, we investigated the level of strand specificity obtained with the TruSeq method. Using a set of well-characterized, highly expressed cellular genes with no known antisense transcription (see Chapter 5, *Materials and methods*), we calculated an average background expression level that we could use to distinguish true antisense transcription from spurious background reads. We found strand specificity to be high, above 99.8% in all samples (Figure 2 and Tables II & III).

### Extensive bidirectional transcription of the lytic EBV genome

Mapping strand-specific RNA-Seq reads to the EBV genome showed coverage across both strands of nearly the entire genome, in both the poly(A)-selected and the ribodepleted datasets (Figure 3). We used the degree of strand specificity calculated above to identify





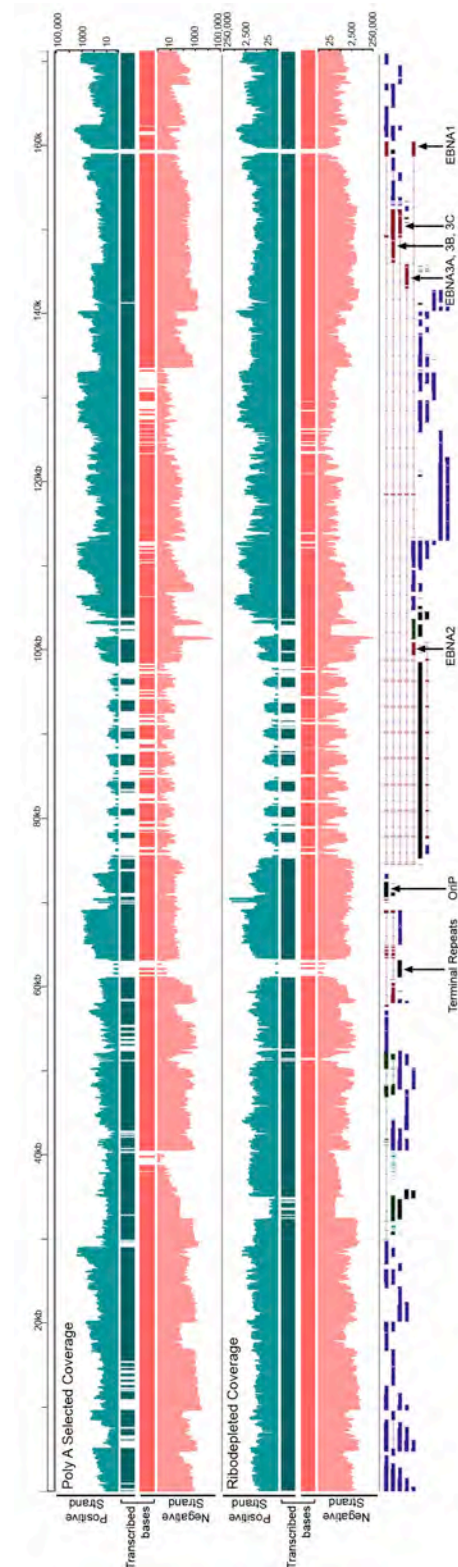
**Figure 2.** RNA-Seq sense and antisense read coverage of cellular genes GAPDH and ACTB.

regions of coverage that exceeded the expected background coverage plus four standard deviations. Genomic coordinates meeting this stringent coverage requirement were categorized as being transcribed (dark tracks in Figure 3).

The EBV genome is known to be gene-dense, with annotated exons covering over 70% of the genome<sup>43</sup> (Figures 3 & 4A). However, only 4% of the genome is annotated as having exons on both strands<sup>43</sup>. We observe RNA-Seq reads from poly(A)-selected RNA covering both strands over 65% of the genome. Even more striking, we observe reads from ribodepleted RNA covering both strands over 80% of the genome (Figure 4A). At least 50% of read coverage from each dataset was in regions not annotated as being exons.

Transcription of unannotated regions did not appear to be incidental, low-level transcription but instead occurred at high levels, with a substantial proportion of transcribed bases being covered at levels comparable to highly expressed cellular genes (Figure 4B). This extensive, high-depth coverage of unannotated EBV genomic regions suggests the existence of previously unidentified viral genes. Given the number of bases outside of previously

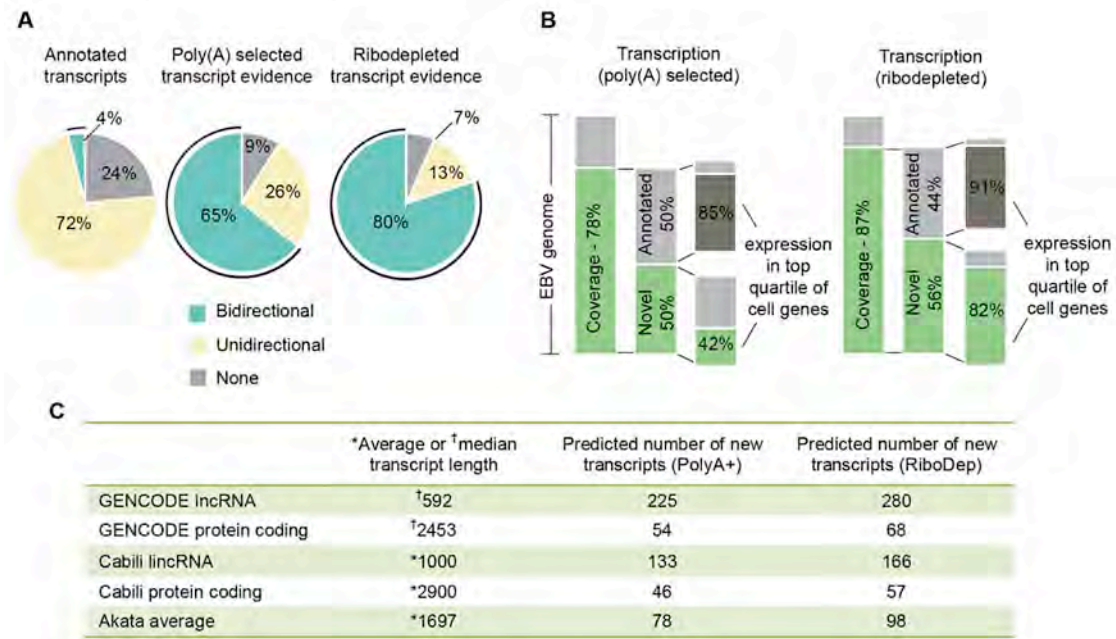
described exons that we identify here as being transcribed, we roughly estimated potential numbers of new genes. The average length of an annotated gene in the Akata genome is 1,697 bases; dividing the number of bases showing novel transcription by this length yields an estimated 78 novel genes in the poly(A)-selected dataset, and 98 in the ribodepleted dataset (Figure 4C). Using published average transcript lengths of cellular genes<sup>78-80</sup> we obtain comparable estimated gene numbers. Because many of these novel transcripts may be noncoding (see below), we also estimated the number of possible novel genes using published catalogs of noncoding genes<sup>78,79</sup>. As noncoding genes are, on average, shorter than coding genes this calculation yields estimated numbers of unannotated genes ranging from 133 to 280 (Figure 4C). These estimates do not take into account the potential for transcripts



**Figure 3.** RNA-Seq read coverage of the Akata EBV genome at 24 hours postinduction. The scales are logarithmic. Transcribed bases tracks display bases with at least five reads with coverage above background levels (see Materials and Methods).



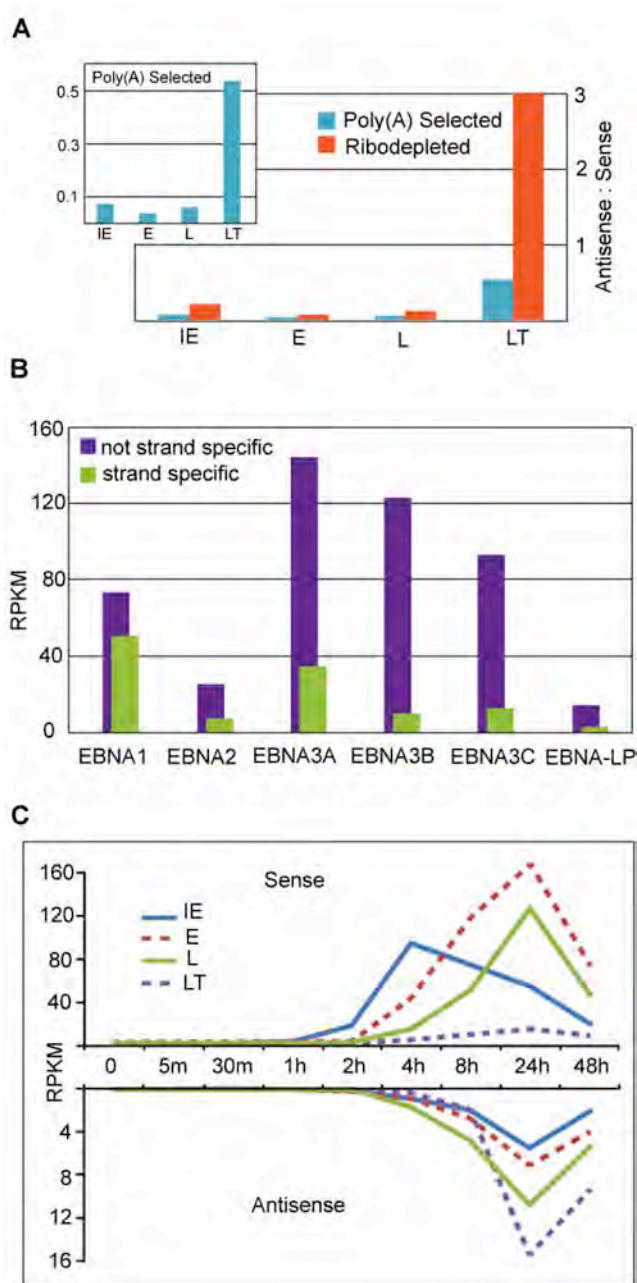
overlapping on the same strand or for alternatively spliced transcripts. Given that both of these phenomena are common in lytic EBV<sup>34</sup>, these numbers are likely underestimates of undiscovered genes. The more extensive coverage of the viral genome using ribodepleted data suggests that many unannotated EBV transcripts are not polyadenylated.



**Figure 4.** Novel transcription of the Akata genome 24 hours postinduction. (A) Percentage of the genome covered by annotated genes and by RNA-seq reads from poly(A) selected and from ribodepleted RNA. (B) Percentage of novel transcribed genome regions and their expression relative to cellular gene expression. Coverage percentage is based on the full length of both strands of genomic DNA, i.e., 342,646 potentially transcribed bases. (C) Predicted numbers of novel genes based on the number of transcribed bases divided by average of or median transcript lengths from published gene catalogs

### Antisense transcription at known EBV gene loci

While we observed high levels of transcription in regions previously believed to be intergenic, we also observed extensive transcription antisense to known genes. Antisense transcription was particularly high at latency-associated loci, with antisense coverage in the ribodepleted dataset far exceeding sense coverage (Figure 5A). While this raises the exciting



**Figure 5.** Antisense transcription at known genes and gene classes. (A) Ratio of antisense to sense RPKM values at 24 h postinduction for annotated gene classes using RNA-Seq coverage from poly(A) selected (blue) and ribodepleted (orange) RNA. Transcription at EBER loci are excluded. (B) RPKM values for EBNA genes calculated using an unstranded (purple) or strand-specific (green) method on RNA-Seq data from poly(A) selected RNA (C) Sense and antisense RPKM values for annotated gene classes at nine time points after reactivation

possibility of functional antisense transcripts, it is also important to note for studies of the known genes in these regions, as antisense reads erroneously assumed to be sense reads seriously distort abundance estimates for latency-associated genes (Figure 5B).

RNA-Seq data from Akata cells harvested at multiple time points after BCR crosslinking provides a more detailed picture of the lytic cascade. Reads aligning to annotated exons in the sense direction showed the expected pattern of expression, with Immediate Early genes rising in abundance first, followed by Early and Late genes. Latent

genes also show deeper coverage after BCR crosslinking. The pattern of expression antisense to known genes differs, with the depth of reads antisense to all classes of known genes peaking at 24 hours, late in the lytic cycle (Figure 5C).

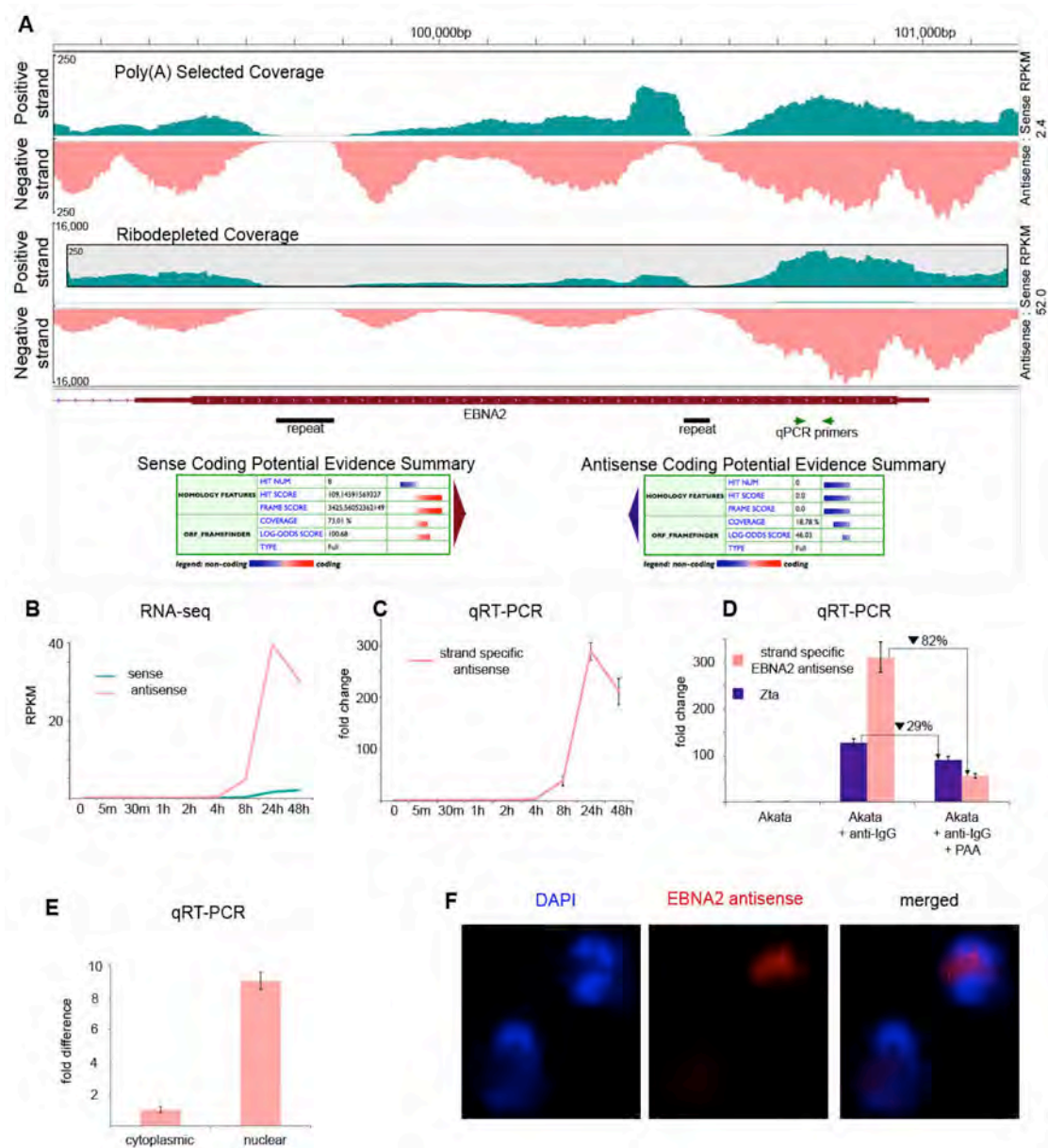
Latency type III gene expression has been detected previously in reactivated EBV, both at the mRNA and protein levels<sup>23,24,81,82</sup>. Our findings of high levels of antisense transcription suggest a more complicated role for latency-associated loci during lytic reactivation than has previously been understood.

### **Extensive antisense transcription at latency loci: an analysis of the EBNAs**

The ratio of antisense to sense transcription is especially high at the Epstein-Barr virus Nuclear Antigen (EBNA) 2 and EBNA3 loci, with antisense read coverage substantially exceeding sense read coverage (Figures 6A, 7A & 8A). EBNA2 and EBNA3A/B/C encode transcription factors that control a range of cellular genes during viral latency<sup>83-85</sup>. EBNA2, EBNA3A and EBNA3B are critical for the transformation of primary B cells into type III latency lymphoblastoid cell lines<sup>86,87</sup>, but the role of the EBNA genes in lytic reactivation is not yet understood.

### **Temporal dynamics of EBNA antisense transcription**

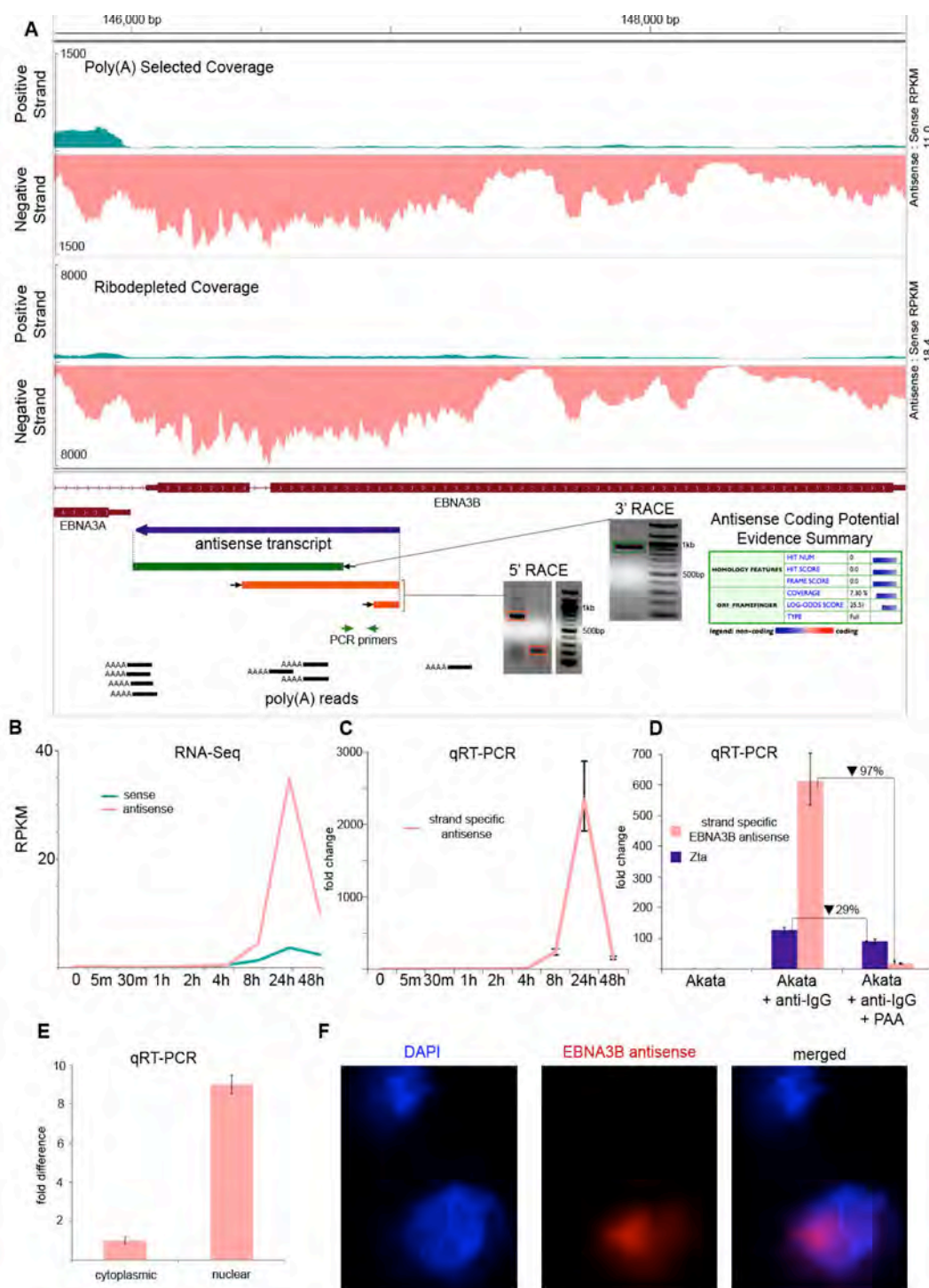
Analysis of RNA-Seq read abundance across multiple post-induction time points shows that antisense read coverage at EBNA2, EBNA3A and EBNA3B loci peaks at 24 hours, with the same timing as annotated Late genes (Figures 6B, 7B & 8B). Strand-specific qRT-PCR of RNA extracted at the same time points shows the same pattern (Figures 6C & 8C). To



**Figure 6.** Sense and antisense transcription at the EBNA2 locus (A) RNA-seq read coverage from poly(A) selected and ribodepleted RNA, and coding potential evidence summary from the Coding Potential Calculator. Note the inset with a zoomed scale for the positive (sense) strand in ribodepleted coverage. (B) RPKM values for RNA-seq time course from ribodepleted RNA (C) Relative abundance of antisense transcription from 0 to 48 h measured by strand-specific qRT-PCR. Primer placement is shown in panel A. (D) Relative abundance of antisense transcription at 24 h after treatment with anti-IgG, anti-IgG plus PAA, or nothing (control). Measured by strand-specific qRT-PCR. (E) Relative abundance of antisense transcription in the nucleus and cytoplasm at 24 h after treatment with anti-IgG. Measured by strand-specific qRT-PCR. (F) FISH of EBNA2 antisense transcripts.

**Figure 7.** Sense and antisense transcription at the EBNA3A locus. (A) RNA-Seq read coverage from poly(A) selected and ribodepleted RNA (top tracks). Blue annotation tracks represent consensus from RACE fragments from 2 or more primers. Yellow circles indicate RACE fragments that end within 20 bases of another RACE fragment. Orange dashed ends indicate matching splice acceptors with different splice donors. Colored boxes on RACE gel images indicate bands corresponding to pictured fragments of the same color. Poly(A) reads track illustrates RNA-Seq reads mapping partially to poly(A) tails. Coding potential evidence summary is from the Coding Potential Calculator. (B) RPKM values from RNA-Seq reads from ribodepleted RNA time course (C) FISH of antisense EBNA3A transcripts.





**Figure 8.** Sense and antisense transcription at the EBNA3B locus (A) RNA-seq read coverage from poly(A) selected and ribodepleted RNA. Blue annotation tracks represents consensus from RACE fragments. Green annotation track represents a 3' RACE fragment (band indicated by green box on gel). Orange annotation tracks indicate 5' RACE fragments (bands indicated by boxes on gel). Poly(A) reads track illustrates RNA-seq reads mapping partially to poly(A) tails. Coding potential evidence summary is from the Coding Potential Calculator. (B) RPKM values for RNA-seq time course using ribodepleted RNA. (C) Relative abundance of antisense transcription from 0 to 48 h measured by strand-specific qRT-PCR. Primer placement is shown in panel A. (D) Relative abundance of antisense transcription at 24 h after treatment with anti-IgG, anti-IgG plus PAA, or nothing (control). Results were obtained by strand-specific qRT-PCR. (E) Relative abundance of antisense transcription in the nucleus and cytoplasm at 24 h after treatment with anti-IgG. Results were obtained by strand-specific qRT-PCR. (F) FISH of antisense transcripts.

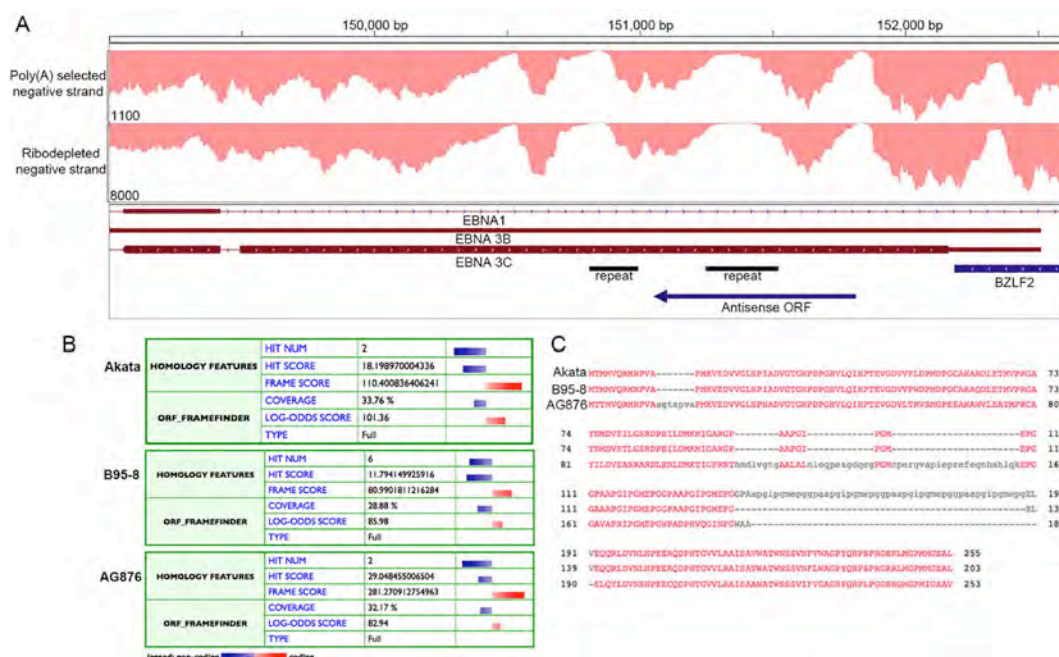
further investigate the potential classification of these antisense transcripts we treated Akata cells with PAA before induction. PAA specifically inhibits the viral DNA polymerase, preventing viral genome replication and the expression of Late genes. After BCR crosslinking, PAA treatment inhibited expression of transcripts antisense to EBNA2 and EBNA3B much more strongly than it inhibited expression of the Immediate Early gene Zta, as measured by strand-specific qRT-PCR (Figures 6D & 8D).

### **Coding Potential of EBNA antisense transcripts**

To facilitate functional analysis of these late antisense transcripts we interrogated the genomic sequence for possible open reading frames, known functional motifs and domains homologous to other genes. The Coding Potential Calculator<sup>88</sup> reports an open reading frame opposite EBNA3C (Figure 9A & B). This open reading frame is conserved in the genomic sequence of both the B95-8 and AG876 strains of the virus, supporting the possibility that an unreported protein-coding gene exists antisense to the EBNA3C open reading frame (Figure 9C). In contrast, no reliable open reading frames are detected antisense to EBNA2, EBNA3A or EBNA3B, and the Coding Potential Calculator reports a high likelihood that transcripts in this region are noncoding RNA (Figures 6A, 7A & 8A).

### **Subcellular localization of EBNA antisense transcripts**

The subcellular localization of biological molecules can provide indications of their function, and so we investigated the location of the noncoding EBNA2, EBNA3A and EBNA3B antisense transcripts. Strand-specific qRT-PCR of RNA isolated from the nuclear and cytoplasmic fractions of 24-hour induced Akata cells indicates strong nuclear enrichment of



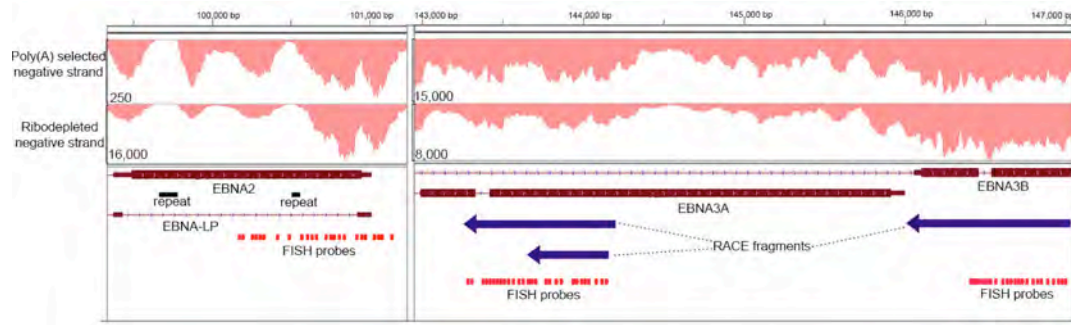
**Figure 9.** A novel antisense ORF in the EBNA3C locus. (A) Genomic location and RNA-Seq read coverage of a novel antisense ORF. (B) Coding Potential Calculator output for corresponding regions in the EBV strains Akata, B95-8 and AG876. (C) BLAST multiple sequence alignment of translated ORFs from Akata, B95-8 and AG876 sequences.

EBNA2 and EBNA3B antisense transcripts (Figures 6E & 8E). Fluorescent *in situ* hybridization (FISH) using a series of probes targeted to the antisense transcripts also supports nuclear localization of transcripts antisense to EBNA2, EBNA3A and EBNA3B (Figures 6F, 7C, 8F & 10). We conclude that genes antisense to EBNA2, EBNA3A and EBNA3B produce nuclear long noncoding RNA transcripts with Late gene kinetics.

### EBNA3 antisense transcript structures

RNA-Seq data has been used to infer the structures of cellular transcripts<sup>89-91</sup>. However, the apparent abundance of overlapping transcripts and the resulting deep and extensive RNA-





**Figure 10.** Genomic location of Stellaris FISH probes relative to known transcripts, RACE fragments and RNA-Seq read coverage.

Seq read coverage of the EBV genome confounds this type of analysis (compare Figure 2 to Figures 6A, 7A & 8A). A previous study used 5' and 3' Rapid Amplification of cDNA Ends (RACE) to glean information about EBV transcript structure and revealed a more complex transcriptional architecture than RNA-Seq coverage made apparent<sup>34</sup>. We applied RACE to the EBNA3A and EBNA3B regions (Figure 7A & 8A). Antisense to EBNA3B, we detected a 5' end and a 3' end that appear to correspond to a polyadenylated transcript that arises antisense to the final exon of EBNA3B and terminates between the first coding exon of EBNA3B and the final exon of EBNA3A (Figure 8A). The 5' start site is supported by two separate RACE primers, and the 3' end is supported by one RACE primer and a pileup of RNA-Seq reads that contain partial poly(A) tails. The 3' end is further supported by the presence of a canonical polyadenylation signal (AATAAA) in the genomic sequence. This polyadenylation signal is fully conserved in both the B95-8 and AG876 strains of the virus (Figure 11B).

Transcription is more complex at the EBNA3A locus (Figure 7A). 5' RACE reveals two 5' start sites antisense to EBNA3A's final exon. These two start sites are both supported by

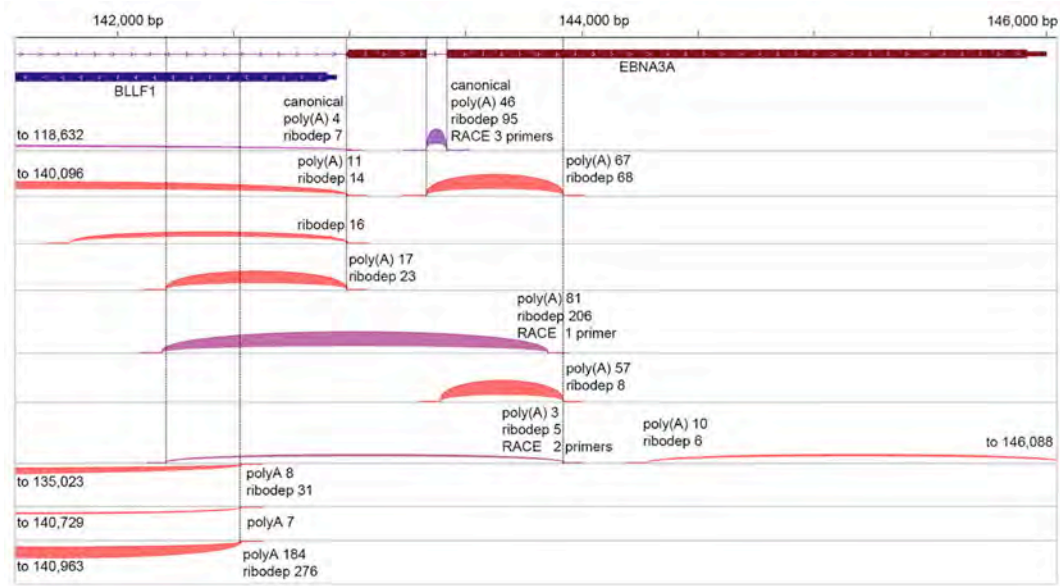


**Figure 11.** Sequence motif conservation. (A) TATA boxes for EBNA3A antisense transcripts. Red boxes indicate the locations of TATA motifs. (B) Polyadenylation signal for EBNA3B antisense transcript. Red boxes indicate AATAAA signal and downstream GT-rich element.

TATA box motifs that are conserved in the B95-8 and AG876 viral strains (Figure 11A). 3' RACE located a 3' end site supported by poly(A)-tail containing RNA-Seq reads downstream of the 5' start sites. However, a 5' RACE fragment generated from randomly primed cDNA extends beyond that polyadenylation site. The same fragment was not generated from poly(A)-primed cDNA, demonstrating the presence of a longer, non-polyadenylated transcript that arises from the more upstream of the 5' start sites (Figure 7).

Surprisingly, 5' RACE in the sense direction revealed previously unannotated EBNA3A-overlapping transcripts, which use unannotated transcription start sites and splice junctions (Figure 7A). One novel transcription start site is located in an intron upstream of the EBNA3A open reading frame. Multiple distinct RACE fragments corresponded to this transcription start site, including several that used the final annotated EBNA3A splice junction, one that contained the full intron sequence of that splice junction, and two that contained a novel, much longer splice junction. A second novel transcription start site near the final annotated EBNA3A splice donor was also detected with multiple primers (Figure 7A).

Further analysis of splicing at the EBNA3A locus revealed unreported splice junctions on both strands. Many of these were spanned by RNA-Seq reads from both poly(A)-selected and ribodepleted RNA, as well as RACE fragments. In many cases, novel splice junctions are supported by as many RNA-Seq reads as annotated splice junctions (Figure 12).

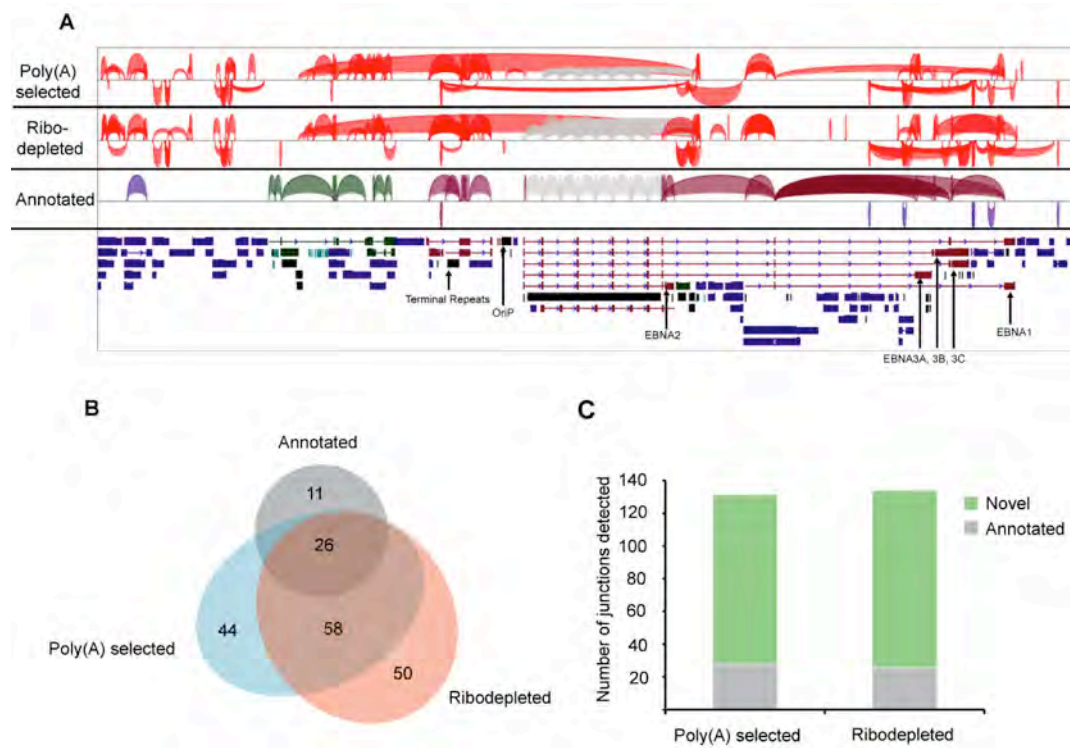


**Figure 12.** Splice junctions in the EBNA3A region that are annotated, detected in fragments from at least 2 RACE primers, or detected in RNA-Seq reads by TopHat analysis (at least 5 reads from poly(A)-selected RNA or 10 reads from ribodepleted RNA).

It should be noted that RNA-Seq read coverage extends beyond the start and end sites detected by RACE at both the EBNA3A and EBNA3B loci (Figures 7A & 8A). Apparently, more overlapping transcripts are present that were not identified by this method. The presence of complex overlapping transcripts at the EBNA genes supports a role in reactivation for these loci that goes beyond a simple recapitulation of their latency-associated functions as protein-coding genes.

### **Extensive novel splicing in the lytic EBV transcriptome**

Splicing is not thought to be common in EBV genes under lytic conditions<sup>35</sup>. Of 37 annotated splice junctions in the Akata genome, only 7 are in lytic genes (Figure 13A). Surprisingly, we observe 178 splice junctions, almost 5 times as many as have been previously annotated in latent and lytic genes combined. 84 of these are reported in both the poly(A)-selected and ribodepleted datasets, including all 7 annotated lytic junctions and 19 junctions annotated in latency-associated transcripts. A further 94 junctions are detected in one dataset or the other, but not both (Figure 13B). Most of the junctions detected in each dataset are unannotated (Figure 13C). We do not detect all of the annotated latency-associated splice junctions despite observing RNA-Seq read coverage at all latency-associated loci: this may be indicative of alternative splicing at latency loci during lytic reactivation, as observed for EBNA3A, above (Figures 7A & 12).



**Figure 13.** Splice junctions at 24 hours postinduction (A) Splice junctions supported by at least five RNA-Seq reads from poly(A) selected RNA or by at least ten RNA-Seq reads from ribodepleted RNA. Annotated splice junctions are color coded: blue, lytic genes; maroon, latent genes; and green, noncoding transcripts. (B) Venn diagram indicating numbers of junctions that are annotated, the number that are detected in poly(A) RNA data, and the number detected in ribodepleted RNA data. (C) Annotated and novel splice junctions detected in poly(A) selected or ribodepleted RNA.

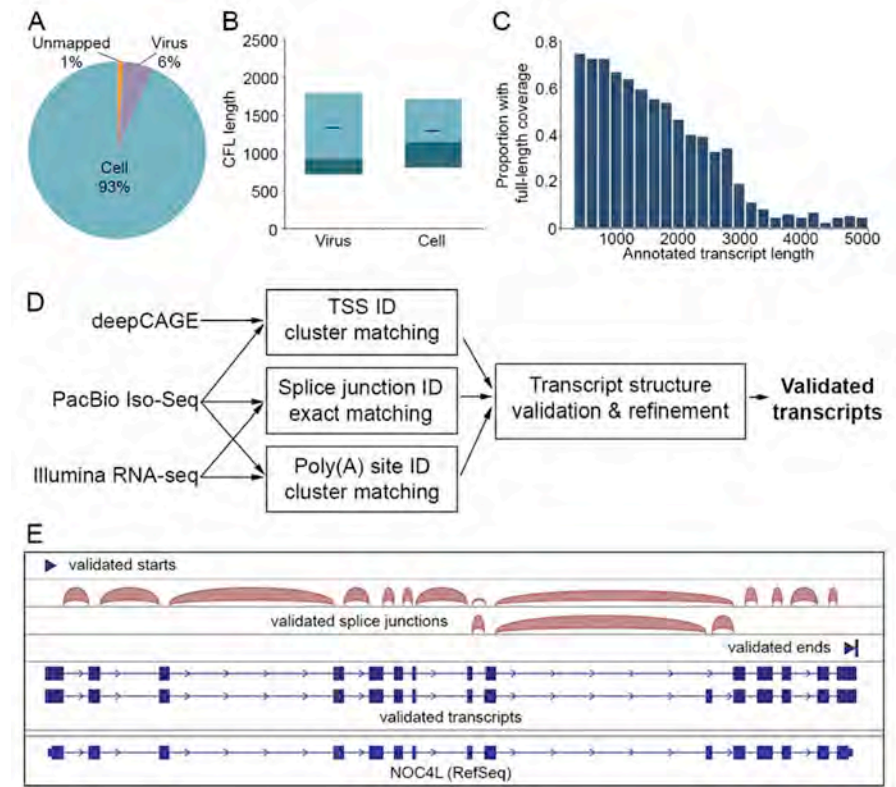
## **CHAPTER 3**

Global determination of novel EBV transcript structures

As seen in Chapter 2, the pervasiveness and complexity of EBV lytic transcription has defied efforts to fully define novel transcripts using short-read sequencing methods. New technologies for long-read sequencing have the potential to resolve many of these difficulties. Pacific Bioscience's Iso-Seq protocol uses Single-Molecule Real-Time (SMRT) long-read sequencing of cDNA to obtain sequences of full-length RNA transcripts<sup>74</sup>. Here, we integrate Iso-Seq data with Illumina RNA-Seq and deepCAGE data to globally elucidate viral transcript structure with a high degree of confidence.

### **Iso-Seq interrogation of the lytic EBV transcriptome**

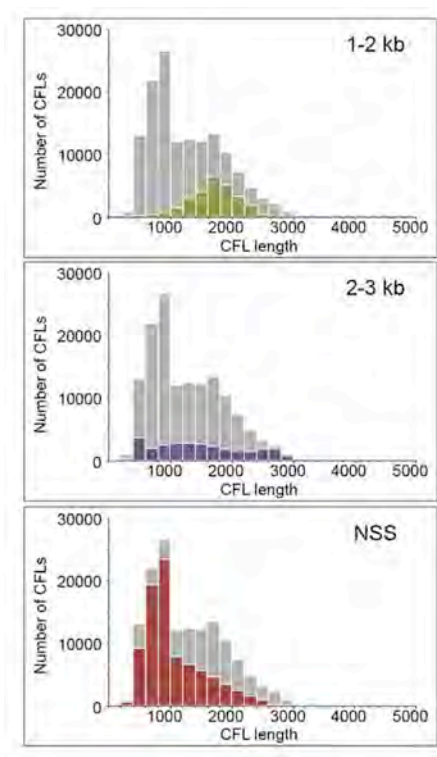
We crosslinked the B-cell receptors of Akata cells using anti-IgG to induce lytic reactivation using the same method as in Chapter 2. RNA was harvested at 20 and 24 hours, pooled and subjected to the Iso-Seq protocol. Raw SMRT read data was processed to obtain a set of consensus “full-length” isoforms (CFLs) using RS\_IsoSeq on SMRTPortal version 1. The CFLs were mapped to both the human (hg 19 assembly) and EBV Akata (KC207813.1<sup>43</sup>) genomes. Approximately 6% of the CFLs mapped to the EBV genome, consistent with results obtained with Illumina RNA-Seq reads (Figure 14A, Tables II & III)<sup>34,73</sup>. CFLs were substantially longer than the 101-base Illumina RNA-Seq reads, with mapped CFL length ranging from 300 to 16,430 bases. The median length of mapped CFLs was 1,335 bases, and the length distribution was similar between CFLs mapped to the EBV genome and CFLs mapped to the cellular genome (Figure 14B). Size fractionation helped to reduce the bias toward sequencing smaller transcripts (Figure 15).



**Figure 14.** Long-read sequence mapping data and transcript identification and validation strategy. See also Figures S1, S2 & S3. (A) Percentages of consensus full-length isoforms mapped to cellular and EBV genomes. (B) Length distributions of consensus full-length isoforms mapped to cellular and EBV genomes. Blue boxes represent second and third quartiles, horizontal black lines indicate mean. (C) Proportion of expressed annotated transcripts that are represented by full-length sequenced isoforms as a function of transcript length. (D) TRIMD data integration/transcript feature validation strategy. (E) Example of TRIMD validated cellular transcripts.

The Iso-Seq protocol was developed to sequence full-length RNA molecules<sup>74</sup>. To estimate the extent to which full-length transcripts are captured by the CFL dataset, we used a set of well-characterized cellular transcripts (RefSeq transcripts with Reviewed or Validated status). We considered a cellular transcript to have full-length Iso-Seq coverage if a CFL's 5' and 3' ends matched the annotated transcript's 5' and 3' ends. We considered a cellular transcript to have partial Iso-Seq coverage if a CFL's 3' end matched the annotated transcript's 3' end but





**Figure 15.** Length distribution of Iso-Seq CFLs from different size fractions. 1-2 kb and 2-3 kb fractions were sequenced on two SMRT cells each. NSS = Non-size-selected fraction, sequenced on four SMRT cells.

the CFL's 5' end did not match the annotated transcript's 5' end, and the annotated transcript had Illumina RNA-Seq reads mapping near its 5' end. We observed that the majority of transcripts shorter than 1,000 bases were represented by full-length CFL coverage, but that the proportion of transcripts with full-length coverage decreased with increasing transcript length (Figure 14C).

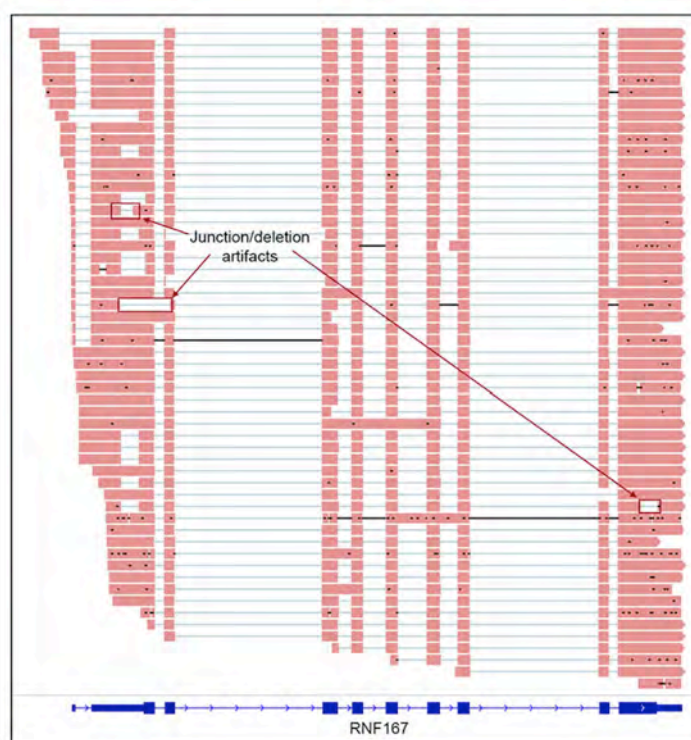
Visual inspection of CFLs mapped to both the cellular and EBV genomes revealed CFLs that closely matched annotated genes and CFLs that appeared to represent novel isoforms (Figure

16). However, other CFLs had 5' ends that mapped progressively downstream of the annotated 5' end, suggesting that they represent the artifactual result of strand invasion during cDNA synthesis<sup>92</sup>. We also observed some CFLs with apparent splice junctions that were unannotated and noncanonical, and were only supported by a single SMRT read. We concluded that additional information from alternate platforms was necessary in order to determine with high confidence which CFLs could be interpreted as accurate full-length representations of transcripts.

## Transcriptome Resolution through Integration of Multi-platform Data (TRIMD)

In order to identify 5' ends, splice junctions and 3' ends with more certainty than can be obtained with Iso-Seq alone, we developed a method to integrate information from three distinct platforms: Iso-Seq, Illumina RNA-Seq and deepCAGE<sup>75</sup>. As described in the following sections, we validated transcript features (5' ends, 3' ends and splice junctions), each using multiple data sources, then used these sets of validated features to identify CFLs that represent true transcripts (Figure 14D). When applied to cellular transcripts this method performed well,

identifying CFLs that closely match annotated transcripts as well as novel splice isoforms and isoforms using alternative transcription start sites and transcription termination sites (Figure 14E, F & G). The development of software to implement TRIMD is described in Chapter 4.



**Figure 16.** CFLs mapping to the cellular gene RNF167. CFLs appear to represent the annotated transcript and possible isoform variants, as well as splice junctions and 5' truncations that are likely artifacts.

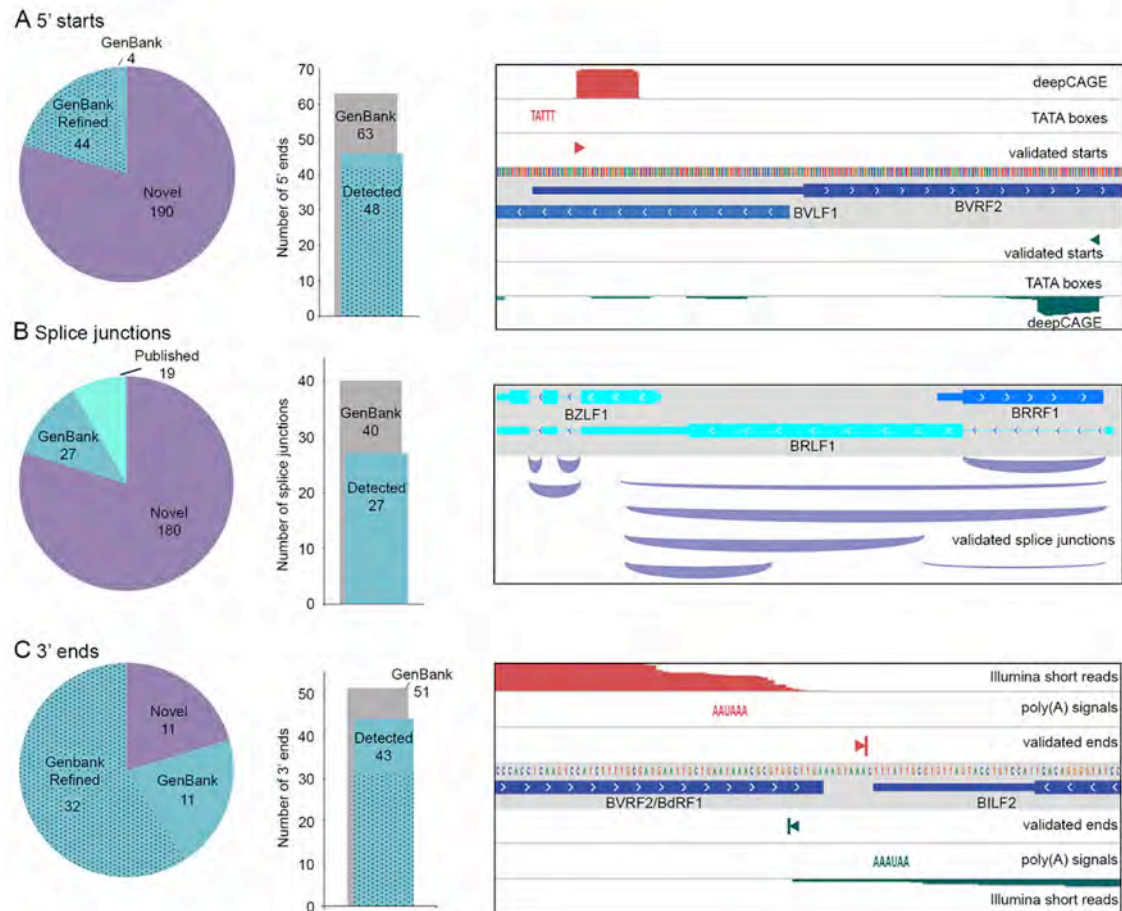
## Identification of transcription start sites in the lytic EBV genome

To gain more information about transcription start sites, we used deepCAGE (Cap Analysis of Gene Expression)<sup>75,93</sup>. This method makes use of the CAP-trapper protocol<sup>94</sup>, a method that acquires RNA fragments near their 5' caps. Iso-Seq, in contrast uses SMART (Switching Mechanism at 5' End of RNA Template) cDNA synthesis<sup>95</sup> to identify transcript 5' ends.

The substantially different mechanisms employed by deepCAGE and Iso-Seq lends confidence to start sites identified by both methods. Notably, the Iso-Seq 5' ends that appear to represent truncation artifacts are not typically supported by deepCAGE (data not shown).

To identify 5' start sites supported by both Iso-Seq and deepCAGE, we began by analyzing data from each method independently. We treated Iso-Seq 5' ends mapping within 8 bases of each other on the genome as start site clusters and calculated the chromosomal coordinate average of each cluster, weighted by SMRT read depth, to determine consensus Iso-Seq start sites. For deepCAGE data, we used the Paraclu algorithm<sup>96</sup> to identify clusters then calculated chromosomal coordinate averages weighted by CAGE tag depth to identify CAGE start sites. Iso-Seq consensus start sites that were supported by CAGE start sites within 3 bases were considered validated transcript 5' start sites.

238 transcription start sites in the EBV genome are supported by both Iso-Seq and deepCAGE, a number more than three times higher than the number of EBV start sites annotated in GenBank (Figure 17A and Appendix 6). The majority (191) of these validated start sites are unannotated. We also validate 47 GenBank-annotated start sites. Many of these were annotated originally based on the location of TATA boxes in the genomic

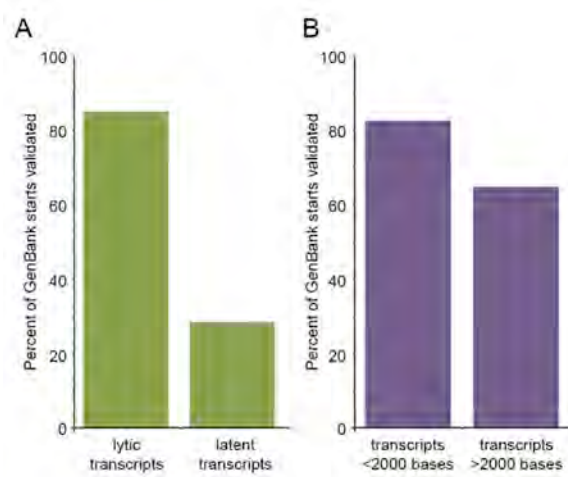


**Figure 17.** Validation of EBV transcript features. See also Figures S4 & S5. (A) Validation of 5' starts. Pie chart indicates annotation status of validated 5' starts. "GenBank Refined" includes start sites previously annotated at TATA boxes and more accurately identified in this study. Bar chart indicates the number of GenBank-annotated 5' starts validated in this study (stippled = refined). Genome browser panel shows examples of validated 5' starts for the GenBank annotated BVR2 and BVL1 genes. (B) Validation of splice junctions. Pie chart indicates annotation status of validated splice junctions. Bar chart indicates the number of GenBank-annotated splice junctions validated in this study. Genome browser panel shows example validated splice junctions (thickness of splice junction features represents combined depth of Illumina RNA-Seq reads and SMRT circular consensus sequence reads). (C) Validation of 3' ends. Pie chart indicates annotation status of validated 3' ends. "GenBank Refined" includes end sites annotated at canonical polyadenylation signals that are more accurately identified in this study. Bar chart indicates the number of GenBank-annotated 3' ends validated in this study (stippled = refined). Genome browser panel shows example validated 3' ends for the GenBank annotated BVR2/BdRF1 and BILF2 genes.

sequence<sup>43,97</sup>; using this empirical evidence we update the annotation to provide more accurate start sites (for updated annotation see Appendix 10). Also, several of the novel transcription start sites identified here appear to be associated with annotated open reading frames for which transcription start sites could not previously be identified because of the

absence of canonical TATA boxes immediately upstream (see Figure 17A, right panel).

Although this study substantially increases the number of known EBV transcription start sites, not all start sites were detected by this method. The start sites of annotated transcripts that are very long or that are latency-associated were not as reliably detected in this study as those from shorter, lytic transcripts (Figure 18). Many deepCAGE-identified 5' ends did not correspond to Iso-Seq identified 5' ends (Figure 19); these likely represent 5' ends of longer and/or non-polyadenylated transcripts, which are not well detected by the Iso-Seq protocol.



**Figure 18.** Annotated EBV 5' start sites validated by TRIMD. (A) Percentage of GenBank-annotated start sites for lytic- and latency-associated transcripts validated by TRIMD (B) Percentage of GenBank-annotated start sites for transcripts under or over 2000 bases long validated by TRIMD.

### Identification of splice junctions in the lytic EBV genome

We used Illumina RNA-Seq data in conjunction with Iso-Seq CFLs to investigate splice junctions in the EBV transcriptome. Information about detected splice junctions was compiled from multiple sets of RNA-Seq data derived from Akata cells at various time points before or after BCR crosslinking. Splice junctions detected in both one or more Iso-Seq CFLs and one or more Illumina RNA-Seq reads were considered validated. As in the Illumina RNA-Seq analysis in Chapter 2, we identified a much higher number of splice junctions than is annotated: a total of 226 splice junctions were supported by both Iso-Seq and Illumina RNA-Seq (Figure 17B and Appendix 7). This confirms 19 splice junctions that



**Figure 19.** Start sites identified by deepCAGE. (A) 5' start sites detected by deepCAGE only, and by both deepCAGE and Iso-Seq. DeepCAGE-identified start sites in genomic repeat regions are excluded. (B) Examples of deepCAGE peaks both supported by Iso-Seq and unsupported by Iso-Seq.

have been reported using Illumina RNA-Seq or other methods<sup>34,73,98-101</sup>. 27 GenBank-annotated splice junctions were validated, including all 7 junctions associated with annotated lytic transcripts and 20 junctions annotated in latency-associated transcripts. 180 novel splice junctions were validated, many at sequencing depths comparable to annotated splice junctions (see Figure 17B, right panel).

### Identification of polyadenylation sites in the lytic EBV genome

As with 5' ends, we treated Iso-Seq 3' ends mapping within 8 bases of each other on the genome as single putative polyadenylation sites and calculated chromosomal coordinate averages weighted by SMRT read depth to identify consensus polyadenylation sites. To locate putative polyadenylation sites in Illumina RNA-Seq data, we used a modification of the approach used in Chapter 2 to identify partial poly(A) tails captured in sequencing reads. Clusters of poly(A) tail-containing reads mapping to within 8 bases of each other were identified and weighted average chromosomal coordinates calculated as for Iso-Seq ends. Iso-Seq consensus 3' ends within 8 bases of Illumina polyadenylation sites were considered validated 3' ends.

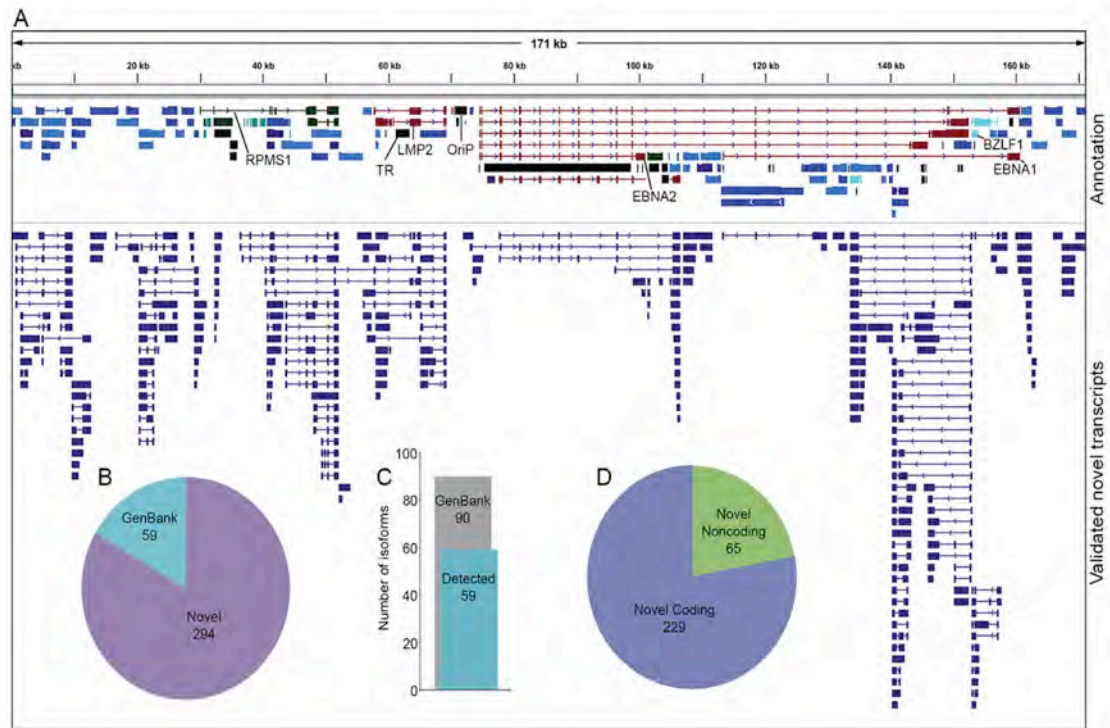
This approach validated 54 polyadenylation sites (Appendix 8). In contrast to validated transcription start sites and splice junctions, most of the validated polyadenylation sites corresponded to 3' ends already annotated in GenBank (Figure 17C). Similarly to transcription start sites, most 3' ends are annotated in GenBank based on genomic sequence motifs, in this case canonical AATAAA or AAUAAA polyadenylation signals<sup>43,97</sup>. We are now able to provide refined polyadenylation sites based on empirical evidence for 32 annotated GenBank 3' ends. Additionally, some of the unannotated polyadenylation sites are downstream of open reading frames for which transcription termination sites could not previously be estimated because of the lack of immediate downstream canonical polyadenylation signals (see Figure 17C, right panel for an example).

### **Isoform structure determination and annotation**

We used the sets of validated transcription start sites, splice junctions and polyadenylation sites to determine which CFLs could be treated as fully validated transcript structures. At this step, 7,906 CFLs that mapped to the EBV genome were filtered and condensed to a final list of 353 distinct isoforms (Figure 20A: see Chapter 4 for more details). 59 of these correspond to GenBank annotated transcripts (Figure 20B & C). Most annotated transcripts that were not validated are either very long or are latency-associated.

294 of the validated isoform structures were novel (Figure 20B and Appendix 9). These include splice variants of annotated transcripts, isoforms that extend or truncate annotated transcripts, chimeric transcripts produced by readthrough transcription and intergenic





**Figure 20.** Novel validated viral transcripts. See also Figures S6 & S7. (A) Top track displays EBV-Akata GenBank annotated gene and structural features. Bottom track displays novel validated EBV transcripts identified in this study. (B) Annotation status of TRIMD-validated transcripts. (C) Number of GenBank-annotated transcripts validated in this study. (D) Coding potential of novel EBV transcripts as determined by CPAT.

splicing, and wholly novel transcripts that arise from a novel transcription start site, span a putative intergenic region and terminate at a novel polyadenylation site. Nearly one quarter of the novel transcripts are likely noncoding (Figure 20D).

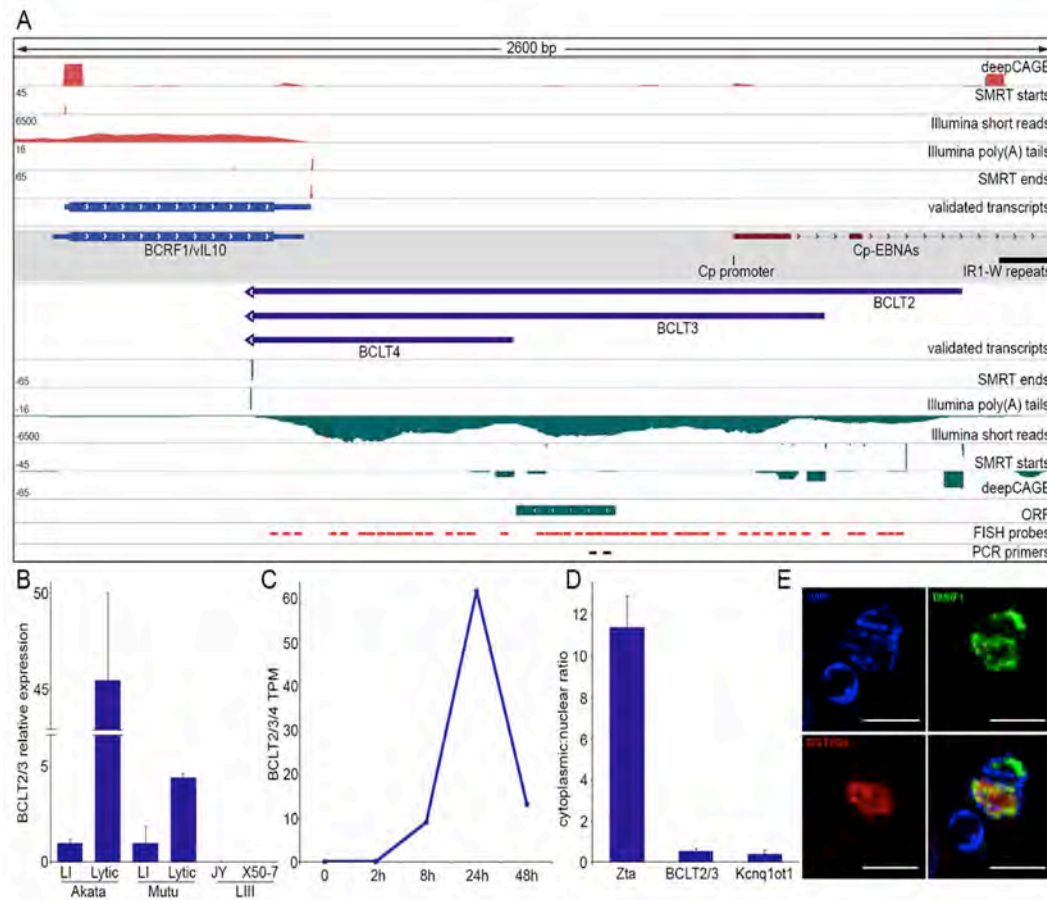
We provide a naming scheme for the novel transcripts based on the existing naming scheme for EBV genes<sup>40</sup>. The first two letters correspond to the genomic BamHI restriction fragment in which transcription initiates: e.g., BC is BamHI fragment C and Ba is BamHI fragment a. The next letter is R for Rightward transcripts (i.e., annotated on the plus strand) or L for Leftward transcripts (i.e., annotated on the minus strand). The final letter in our



naming scheme is T for Transcript: the original naming scheme is based on protein-coding genes and uses F for reading Frame. We begin numbering our validated transcripts where the GenBank annotation numbering ends for each fragment. For example, BBRF3 is an annotated gene and we present here novel transcripts we designate BBRT4, BBRT5, etc.

### **Novel intergenic transcription of the lytic EBV genome**

BCLT2, 3 and 4 are a set of overlapping transcripts that do not share any sequence with annotated EBV transcripts. They arise from three novel transcription start sites in the vicinity of, but antisense to the latency-associated Cp promoter, are transcribed through a putative intergenic region, and terminate at a shared polyadenylation site antisense to the viral IL10 homolog BCRF1 (Figure 21A). To gain insight into the possible functions of these transcripts we investigated the timing and context of their expression, their coding potential and their subcellular localization. Strand-specific qRT-PCR using primers to an overlapping region of BCLT2 and BCLT3 showed substantially higher abundance after BCR crosslinking in both Akata and Mutu cells (Figure 21B). Little or no BCLT2/3 expression was detected in the type III latency cell lines JY and X50-7, supporting lytic-specific induction of these transcripts. Also, no evidence of BCLT2-4 expression was detected by SMRT long-read sequencing of the type III latency cell lines GM12878, GM12891 and GM12892<sup>102</sup> (data not shown). Quantification of RNA-Seq reads from poly(A)-selected RNA harvested from Akata cells at multiple time points showed Illumina RNA-Seq read coverage in this region peaking at 24 hours after induction, consistent with Late gene expression (Figure 21C). Sequence analysis of BCLT2-4 using the Coding Potential Assessment Tool (CPAT)<sup>103</sup> indicated all three transcripts are likely noncoding, though a small open reading frame is present in BCLT3 and BCLT4 (Figure 21A). Strand-specific qRT-PCR of RNA extracted



**Figure 21.** Novel intergenic transcripts. (A) Genome browser visualization of BCLT2-4 transcripts and supporting evidence. Grey shaded track displays GenBank-annotated features. (B) Strand-specific qRT-PCR of BCLT2/3 in Akata, Mutu, JY and X50-7 cells. LI = type I latency, LIII = type III latency (C) Normalized Illumina RNA-Seq read counts of BCLT2/3/4 at multiple time points after induction. TPM = transcripts per million. (D) Strand-specific qRT-PCR of nuclear and cytoplasmic fractions of induced Akata cells (24 h). (E) FISH and immunofluorescence of BCLT2/3/4 and EBV nuclear protein BMRF1.

from nuclear and cytoplasmic fractions of 24-hour induced Akata cells indicated strong nuclear enrichment of these transcripts (Figure 21D). This was confirmed by FISH using a set of fluorescently labeled probes targeting all three transcripts (Figure 21E). Simultaneous immunolabeling of the viral nuclear protein BMRF1 showed some degree of overlap with the BCLT2-4 transcripts. BMRF1 is known to be associated with newly synthesized viral

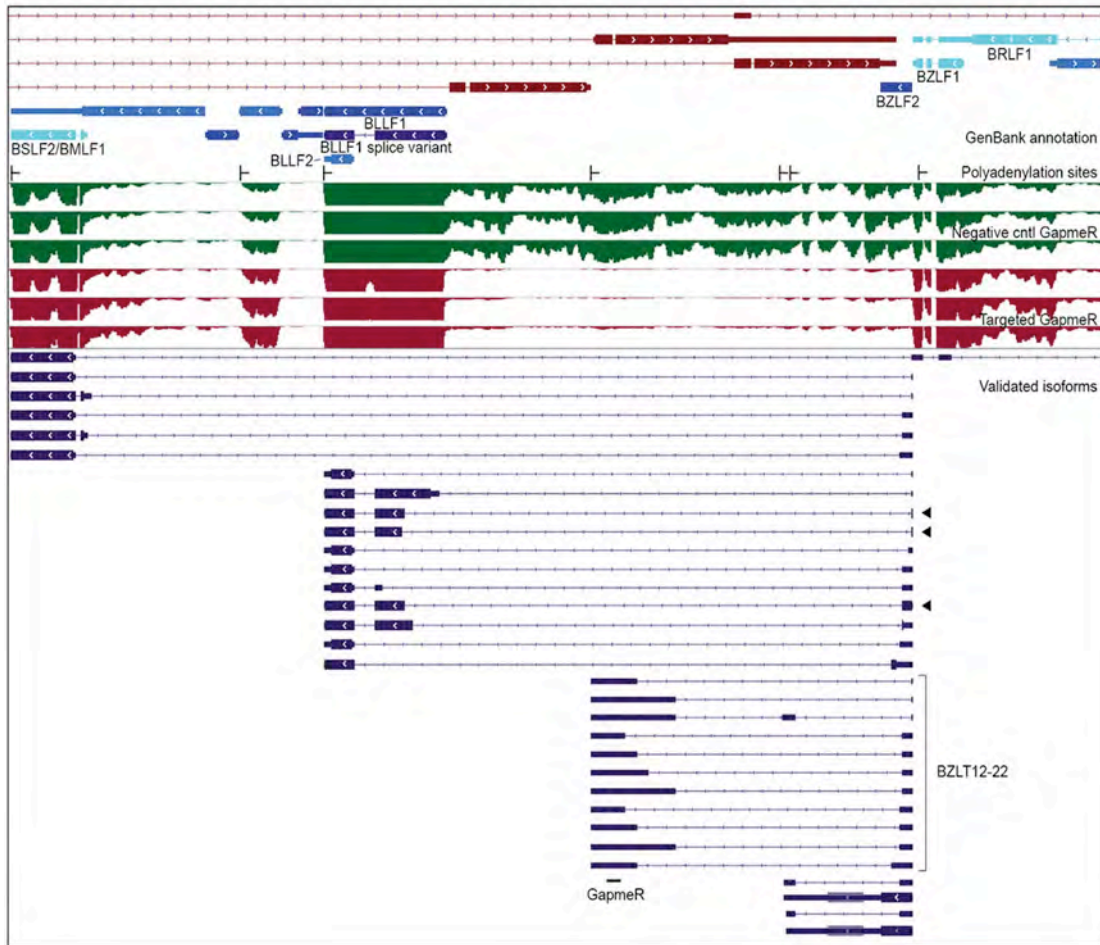
genomes<sup>104,105</sup>: colocalization with this protein suggests that BCLT2-4 transcripts may also associate with newly replicated viral DNA.

### **Readthrough transcription**

Transcription start sites and polyadenylation sites shared by multiple transcripts are common in the EBV transcriptome (see, e.g., Figure 20A). The transcription start site upstream of the BZLF2 open reading frame is a striking example of this, giving rise to over 30 distinct transcripts (Figure 22). This transcription start site is not supported by a canonical TATA box and as such was not previously annotated; nevertheless the SMRT read and CAGE tag depth for this site is greater than that of any other EBV transcription start site detected in this study.

Only two of the transcript structures originating at this site contain the full BZLF2 open reading frame. These transcripts are unspliced and terminate at a pair of polyadenylation sites 23 bases apart, approximately 2 kb downstream of the transcription start site (Figure 22). Interestingly, both also contain the novel open reading frame antisense to EBNA3C that was described in Chapter 2 (Figure 9). This open reading frame begins approximately 350 bases downstream of the BZLF2 translation stop codon and is larger than BZLF2. It is possible that these transcripts, like others previously identified in EBV<sup>106,107</sup>, are bicistronic.

Most other isoforms arising from this locus are the result of transcriptional readthrough that bypasses both of the BZLF2-proximal polyadenylation sites. Several transcripts continue to bypass further polyadenylation sites, with some finally terminating nearly 19 kilobases downstream at a polyadenylation site shared with the BSLF2/BMLF1 mRNA. This



**Figure 22.** Readthrough transcription and intergenic splicing at the BZLF2 locus. From top: GenBank gene annotation, TRIMD-validated polyadenylation sites, Illumina short-read coverage of induced Akata cells with negative control GapmeR (green tracks) and induced Akata cells with GapmeR targeting BZLT12-22 (red tracks), novel validated isoforms (blue transcript features). Black arrows indicate transcripts whose largest ORF is an in-frame fusion.

transcriptional readthrough allows for intergenic splicing including, intriguingly, splicing that preserves partial or full open reading frames of annotated genes. Three isoforms (indicated by black arrows in Figure 22) are candidate mRNAs coding for chimeric proteins that contain protein structure from both BZLF2 and BLLF1.

A group of overlapping BZLF2-readthrough transcripts terminates at the unannotated polyadenylation site antisense to EBNA3B that was described by 3' RACE in Chapter 2

(Figure 8A). We designed a GapmeR antisense oligonucleotide targeted to an overlapping region of these transcripts and treated Akata cells with either this GapmeR or a negative control GapmeR prior to BCR crosslinking. Illumina short read RNA-Seq of poly(A)-selected RNA harvested from these cells 24 hours after crosslinking revealed an interesting effect on transcripts from this locus: short-read coverage was substantially decreased over a large region of the EBV genome, much more extensive than that covered by the targeted isoforms (Figure 22, green and red coverage tracks). The pattern of coverage suggests that the GapmeR is strongly inhibiting a 12 kilobase, unspliced transcript that arises from the BZLF2-associated transcription start site and terminates at the BLLF1-associated polyadenylation site. Given the decrease in full-length CFL coverage for long transcripts (Figure 14C), a transcript of this length is unlikely to be detected by the Iso-Seq method. Interestingly, very long transcripts antisense to latency loci have also been reported in KSHV<sup>31,33</sup>.

### **EBNA3 antisense transcript structures**

Despite relatively low Iso-Seq coverage across the EBNA loci, we were able to determine the isoform structures of several of the transcripts detected by RACE in Chapter 2 (Figures 7A, 8A and 23). The novel transcript BELT4 arises from a RACE-identified transcription start site antisense to EBNA3A (Figure 23A). It does not terminate at the pileup of poly(A)-containing Illumina short reads depicted in Figure 7A but extends further downstream, utilizes an unannotated splice junction reported by Concha *et al.*<sup>34</sup> and terminates at the BLLF1/2-associated polyadenylation site (Figure 23A and data not shown). BELT4 contains the full BLLF2 open reading frame. Two additional transcription start sites in the region give rise to additional isoforms (BLLT4-9). Interestingly, all of the isoforms arising antisense to



**Figure 23.** TRIMD-validated transcripts at the EBNA loci. Transcripts indicated with black arrows were partially described in Chapter 2, Figures 7 & 8. (A) Validated splice junctions and transcripts at the EBNA3A locus. Splice junctions with \* were detected in Chapter 2 and depicted in Figure 12. (B) Validated transcripts at the EBNA3B locus.

the coding exons of EBNA3A contain open reading frames: this locus appears to be a collection of 5' UTR-variant mRNAs coding for BLLF1 (gp350/220) and BLLF2.

Antisense to the coding exons of EBNA3B we identify several overlapping transcripts that originate at the BZLF2-associated transcription start site as discussed above (Figure 22), and also several shorter, unspliced transcripts (Figure 23B). 5' RACE and 3' RACE in Chapter 2 revealed a transcription initiation site and a polyadenylation site (Figure 8A); TRIMD confirms that the transcript BELT1 uses both of these sites (Figure 23B). Another transcript, BELT2, also arises from this start site and is spliced nearly 11 kilobases downstream to the BSLF2 open reading frame. A further transcription start site downstream of the first gives rise to the overlapping BELT3 transcript.

### **EBNA3 sense transcript structures**

EBNA3A, EBNA3B and EBNA3C transcripts are annotated as arising from the Cp promoter, containing a repeating set of exons denoted as W1 and W2, then a pair of non-repeat exons known as Y1 and Y2, and finally terminating with their individual coding exons<sup>108,109</sup>. Under type I latency conditions in the Akata cell line the EBNA3 transcripts are not expressed (Table I). Expression of these genes has been reported by others during viral reactivation<sup>23,24</sup>, and we see Illumina short read coverage across the expected exons (data not shown). However, no Iso-Seq CFLs contained full-length annotated transcripts of the EBNA genes, likely because these transcripts are very long, ranging from 4,814 to 8,265 bases.

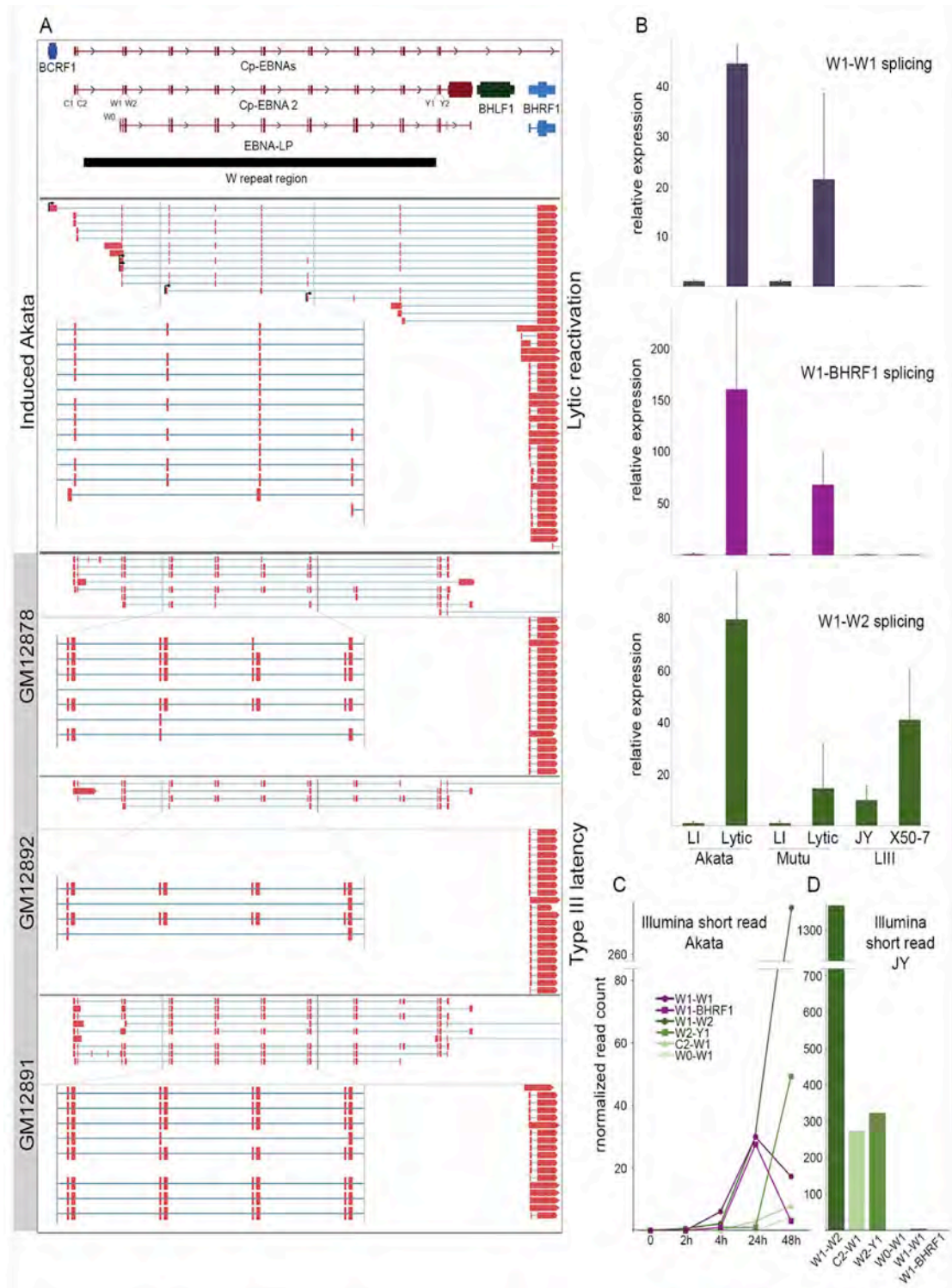
5' RACE experiments in Chapter 2 (Figure 7A) supported the existence of novel transcripts overlapping EBNA3A in the sense direction. Using TRIMD, we determined the full structure of one such transcript partially described by RACE: BLRT5 (Figures 23A and 7A). Interestingly, BLRT5 and four other overlapping isoforms (BLRT4, BERT1, BERT2 and BERT3) contain truncated versions of the EBNA3A open reading frame (Figure 23A). Truncated versions of EBNA2 and EBNA3C were also identified (data not shown).

### **Programmed exon skipping in W-repeat transcripts**

While full-length annotated EBNA transcripts were not identified by Iso-Seq in reactivated Akata cells, many Iso-Seq CFLs do contain exons transcribed from the W repeat region (Figure 24A). Unexpectedly, these transcripts uniformly exclude W2 exons and do not splice to the Y1 or Y2 exons or the unique EBNA exons. Instead, each CFL contains multiple W1 exons and terminates with an exon encompassing the BHRF1 open reading frame. This is distinct from transcript structures detected by long-read sequencing in the type III latency LCLs GM12878, GM12892 and GM12891, which contain the expected W1-W2 splicing pattern and show no evidence of upstream splicing in the annotated BHRF1 transcripts (Figure 24A)<sup>102</sup>.

BHRF1 transcripts containing both W1 and W2 exons have been reported under latency conditions<sup>110-112</sup>, suggesting that the exclusion of W2 exons is a phenomenon associated with viral replication. To further investigate this possibility we performed qRT-PCR using primers spanning the W1-W1, W1-W2 and W1-BHRF1 splice junctions in type I latency (Akata and Mutu cells), viral reactivation (BCR-crosslinked Akata and Mutu cells) and type III latency (JY and X50-7 cells – Figure 24B). As expected, little or no W1-W1 and W1-BHRF1 splicing





**Figure 24.** Programmed exon skipping in the W repeat region. (A) Genome browser visualization of CFLs mapping to the W repeat region and/or BHRF1 gene in induced Akata cells and lymphoblastoid cell lines. (B) qRT-PCR using primers spanning the indicated splice junctions in Akata, Mutu, JY and X50-7 cells. LI = type I latency, LIII = type III latency, Lytic refers to 24 or 48 h induction in Akata and Mutu cells. (C) Time course analysis of splice junction reads in polyA+ RNA from Akata cells. (D) Splice junction reads detected in polyA+ RNA from the type III latency cell line, JY.

was detected during either type of latency, but the abundance of both splice junctions was substantially increased during reactivation. Canonical W1-W2 splicing was detected at high levels in the type III latency cells relative to the type I latency cells. A substantial increase in W1-W2 splicing was also detected during viral reactivation by qRT-PCR, despite its absence from Iso-Seq data during reactivation. This is consistent with W1-W2 splicing being present in very long EBNA transcripts, which are unlikely to be detected by Iso-Seq.

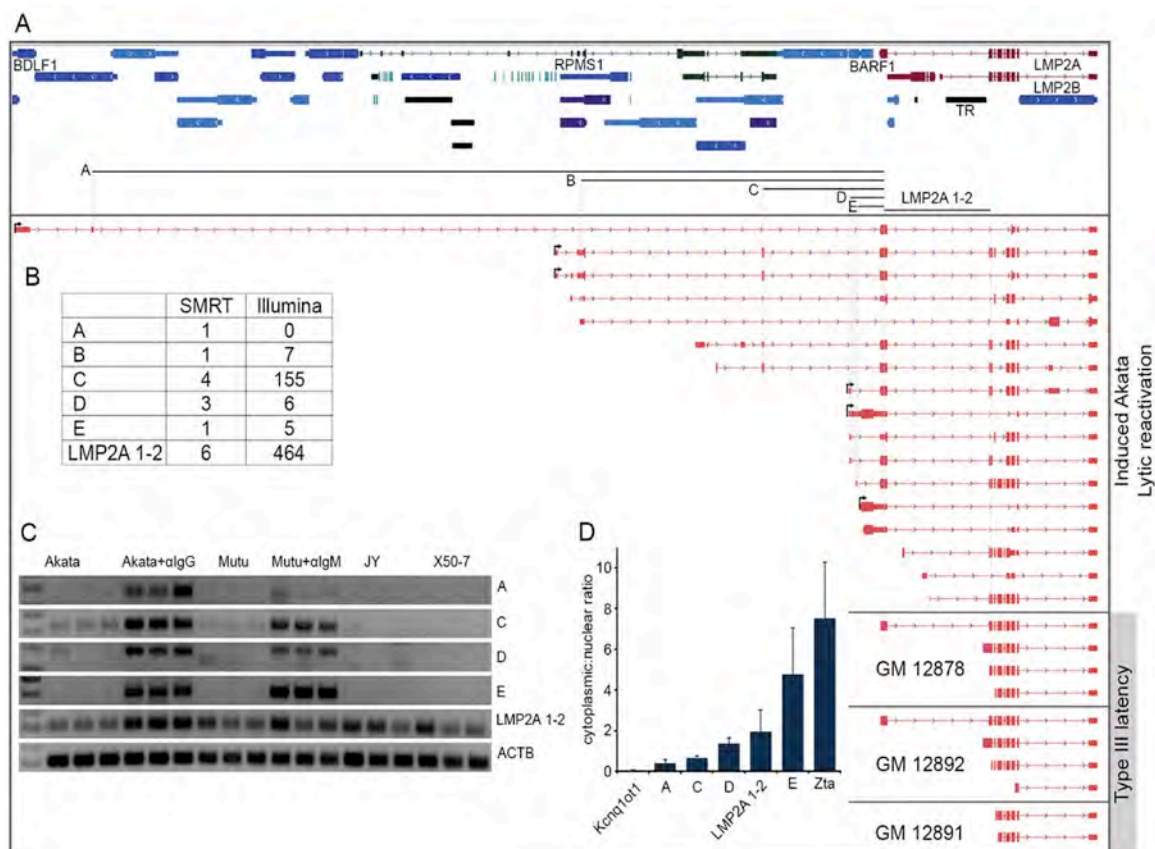
Illumina short read sequencing from multiple time points following BCR crosslinking in Akata cells provides more information about the temporal dynamics of splicing in this region (Figure 24C). The annotated EBNA-associated splice junctions W1-W2, W2-W1, C2-W1 and W0-W1 increase steadily throughout reactivation, even as the levels of viral transcription drop by 48 hours. Reactivated Akata cells have been shown to remain viable for at least 72 hours after BCR crosslinking<sup>23</sup> and this evidence suggests that at least some Akata cells enter the type III latency phase after BCR crosslinking, as reported by Rowe *et al.*<sup>82</sup> In contrast, the depth of coverage for the novel W1-W1 and W1-BHRF1 splice junctions increases after crosslinking to a peak at 24 hours, then decreases. Short-read sequencing from the type III latency JY cell line provides evidence supporting only the expression of annotated EBNA transcripts, with abundant splicing from W1-W2, C2-W1 and W2-Y1, and little or no splicing from W1-W1 or W1-BHRF1 (Figure 24D). Taken together, these results demonstrate distinct usage of the latency-associated W exons during viral reactivation.

### Alternative promoter usage in LMP2 isoforms

The latency-associated gene LMP2 also undergoes significant transformations in its splicing pattern during viral reactivation, as demonstrated with Illumina short read sequencing<sup>34</sup>. Long-read sequencing with Iso-Seq confirms the presence of unannotated splice junctions and also reveals a previously undetected layer of complexity: that of alternative promoter usage. (Figure 25A). Strikingly, no Iso-Seq CFLs from BCR-crosslinked Akata cells are consistent with transcription initiation at the annotated transcription start site. No deepCAGE-derived transcription start sites are present at that location either (data not shown). In contrast, SMRT sequence reads from the type III latency LCLs GM12878, GM12892 and GM12891 are all consistent with transcript initiation occurring at the annotated LMP2A transcription start site<sup>102</sup> (Figure 25A).

Most CFLs that contain LMP2 exons initiate from one of several locations upstream of the annotated LMP2 transcription start site, and many also exhibit novel splicing upstream of the annotated LMP2 exons. Most of these novel splice junctions are supported by Illumina short reads (Figure 25B). We validated expression of several junctions by qRT-PCR using junction-spanning primers and found that their expression is limited to cells undergoing viral replication (i.e., BCR-crosslinked Akata and Mutu cells – Figure 25C). We next performed qRT-PCR on RNA from nuclear and cytoplasmic fractions of BCR-crosslinked Akata cells (Figure 25D). We found the upstream junction associated with the only CFL that contained the full LMP2 reading frame (E in Figure 25) to be the most strongly enriched in the cytoplasm, suggesting that this isoform is exported to the cytoplasm for translation while others are retained in the nucleus for other functions. These findings demonstrate the remarkably complex nature of the lytic LMP2 locus, where alternative promoter usage and

alternative splicing lead to the production of a diverse group of LMP2 isoforms that occupy different cellular locations.



**Figure 25.** Complex lytic promoter usage for LMP2 transcripts. (A) Genome browser visualization of CFLs mapping to the LMP2 exons in induced Akata cells and lymphoblastoid cell lines. Arrows positioned at the beginning of reads signify those with validated 5' ends. (B) Splice junction read depth for SMRT circular consensus and Illumina short-read sequencing. Labels A through E refer to junctions indicated below GenBank-annotated gene track in (A). (C) PCR using junction-spanning primers in Akata, Mutu, JY and X50-7 cells. Akata+αIgG and Mutu+αIgM refer to Akata and Mutu cells induced for 24 and 48 h, respectively. (D) qRT-PCR of nuclear and cytoplasmic fractions of induced Akata cells (24 hours).

## **CHAPTER 4**

Development of an algorithm to automate transcript structure determination and annotation

Efficiently parsing and integrating multiple data types at the genome scale requires automation. We implemented the TRIMD method with Perl scripts for our analysis of the lytic EBV genome (Figure 14D). Because the TRIMD concepts can be generalized to apply to other genomes the scripts were developed to be flexible and customizable, with easily adjustable parameters to accommodate the particularities of other genomes and datasets. The full suite of TRIMD scripts and documentation will be available under the GNU General Public License<sup>113</sup> at <https://github.com/flemingtonlab/public> and is reproduced here as Appendices 1-5.

The TRIMD scripts were developed assuming that Iso-Seq, deepCAGE and Illumina short-read RNA-Seq data are processed according to the steps described in Chapter 5 (*Materials and methods*). Other programs and parameters may be used providing the output files are in the same format.

### **Transcription start site identification using *TRIMD\_start\_validator.pl***

The script *TRIMD\_start\_validator.pl* accepts as input a SAM file of mapped Iso-Seq CFLs, a SAM file of mapped CAGE tags, and a BED file of annotated polyadenylated transcripts.

Using the SAM file of mapped Iso-Seq CFLs, the script identifies CFLs that map to the user-specified chromosome and that are not softclipped at their 5' end. CFLs mapping to each strand are processed separately. The 5' start site of each qualifying CFL is identified and

its supporting SMRT read depth is extracted from the CFL name field. The number of SMRT reads supporting CFLs that start at each genomic coordinate are summed to produce a BedGraph file of CFL start sites. The script then uses this BedGraph file to identify clusters of CFL start sites, defining start sites within a user-specified distance of each other (default: 8) as single putative start sites and producing a temporary BED file of CFL start site clusters. Next, the script uses the cluster information in the temporary BED file and the start site read depth information in the BedGraph file to calculate an average of the genomic coordinates in each cluster, weighted by the depth of SMRT read starts at each position within the cluster. This weighted average is taken to be the Iso-Seq consensus start site for that cluster and a BED file of consensus Iso-Seq start sites is generated. The name field of each feature in this BED file includes the genomic coordinates of the entire cluster and the total SMRT read start site depth of the cluster. The coordinates reported in the BED file's chrStart and chrEnd fields represent the weighted average consensus start site and are zero-based, half-open relative to the genome while cluster coordinates in the name field are zero-based for plus strand start sites and one-based for minus strand start sites.

To identify putative start sites in the deepCAGE data *TRIMD\_start\_validator.pl* incorporates the clustering algorithm Paraclu<sup>96</sup>. The script first processes mapped CAGE tags from the CAGE SAM file and formats the data for Paraclu by identifying tag start sites in the same way as for Iso-Seq CFLs (above). The script also collates this start site information into a BedGraph file that can be directly visualized on a genome browser. The Paraclu function of the script generates start site clusters and *TRIMD\_start\_validator.pl* filters the Paraclu output according to user-specified parameters (see Appendix 5, *TRIMD\_README.txt*). A temporary BED file of CAGE start site clusters is generated and a weighted genomic

coordinate average calculated for each cluster as for Iso-Seq 5' ends (above). These weighted averages are taken as putative start sites detected by deepCAGE and a BED file is generated as for Iso-Seq start sites above.

*TRIMD\_start\_validator.pl* then compares each feature in the BED file of Iso-Seq identified start sites with the features in the BED file of deepCAGE-identified start sites. Iso-Seq 5' start sites that are supported by a deepCAGE 5' start site within a user-specified distance (default: 3 bases) are considered validated 5' start sites and are printed to a BedDetail file, with information about the range and depth of the Iso-Seq start site cluster included in the sixth BedDetail field.

Finally, the list of annotated start sites is extracted from the user-supplied annotation file and each validated start site is compared to the annotated start sites. Validated start sites within a user-specified distance of annotated start sites (default: 10) are noted as annotated in the name field of the output BED file, others are noted as novel (see Appendix 6 for an example of the format). The script also provides the user with counts of total, annotated and novel 5' start sites validated, both in the terminal window and a separately generated text file.

Many parameters in *TRIMD\_start\_validator.pl* are adjustable; for example greater SMRT read or CAGE tag depth can be required for datasets that appear to contain a large number of background reads. The BedGraph files of Iso-Seq and CAGE start sites, as well as the BED files of start sites identified in either or both datasets, are useful for researchers to visually inspect the data and select suitable parameter values.



### Splice junction identification using *TRIMD\_junction\_validator.pl*

The script *TRIMD\_junction\_validator.pl* accepts as input a file of splice junctions identified in Iso-Seq CFLs (in the format output by the GMAP aligner<sup>114</sup>), a file of splice junctions identified in Illumina RNA-Seq reads (in the format output by the STAR aligner<sup>115</sup>), a BED file of annotated polyadenylated transcripts and, optionally, a BED file of genomic regions to ignore in the analysis (e.g., repeat regions).

The script first compiles and reformats information about splice junctions detected by Iso-Seq. Splice junctions corresponding to the user-specified chromosome are identified and the total SMRT read depth for each of these junctions is determined. This information is included in a BED file of Iso-Seq-identified junctions, in which chrStart and chrEnd coordinates of each junction feature refer to the first and last bases of the excised intron. If the user has supplied a BED file of genomic regions that are to be ignored, splice junctions with donors and/or acceptors in those regions are removed. Next an analogous process is carried out using the Illumina splice junctions file, converting relevant junctions to BED format and removing those in user-excluded regions.

After the BED files are generated containing lists of introns from Iso-Seq and from Illumina RNA-Seq, the script compares the junctions in the two files. Junctions that are detected by each platform with at least the user-specified depth requirements (default: 1 read from each platform) are considered validated. Lastly, annotated splice junctions are extracted from the user-specified annotation file and compared to the validated splice junctions. Annotated junctions are indicated as such in the BED output file and other junctions are indicated as

novel. The script also provides the user with counts of total, annotated and novel splice junctions validated, both in the terminal window and as a separately generated file.

### **Polyadenylation site identification using *TRIMD\_end\_validator.pl***

The script *TRIMD\_end\_validator.pl* accepts as input a SAM file of mapped Iso-Seq CFLs, a SAM file of mapped Illumina RNA-Seq reads, and a BED file of annotated polyadenylated transcripts.

The script first extracts 3' ends of Iso-Seq CFLs using a method analogous to that used by *TRIMD\_start\_validator.pl* to extract 5' CFL start sites (see above). A BedGraph file of Iso-Seq 3' ends is generated and used to identify clusters of 3' ends within a user-specified distance of each other (default: 8 bases). Consensus end sites within the clusters are determined in the same way as for consensus start sites: by calculating an average of genomic coordinates for each cluster weighted by SMRT read depth. A BedDetail file of Iso-Seq consensus 3' ends is generated, with the chrStart and chrEnd fields representing the consensus polyadenylation site (zero-based, half open) and the sixth field containing information about the range and depth of the full cluster.

To identify transcript 3' ends in Illumina RNA-Seq data, *TRIMD\_end\_validator.pl* first identifies RNA-Seq reads that map to the user-specified chromosome and end with a run of the user-specified number of As (or start with the user-specified number of Ts – default 5), with at least a user-specified number of bases that do not match the genomic sequence (default: 2). These reads, which contain putative poly(A) tails, are added to a SAM file that is then sorted by genomic coordinate. The script then extracts the polyadenylation site (defined

as the last base of the read that aligns to the genome before the terminal mismatches) from each read in this SAM file and produces a BedGraph file. The BedGraph file is then used to extract clusters and calculate consensus polyadenylation sites in the same way as for Iso-Seq 3' and 5' ends.

Next, *TRIMD\_end\_validator.pl* compares the consensus 3' ends from Iso-Seq and from Illumina RNA-Seq. Iso-Seq consensus 3' ends that are supported by Illumina RNA-Seq consensus 3' ends within a user-specified number of bases (default: 4 upstream or 10 downstream) are considered validated 3' ends. Validated 3' ends are added to a BedDetail file that has a sixth field containing information about the range and depth of the Iso-Seq cluster. Because Illumina RNA-Seq 3' end clusters were usually downstream of Iso-Seq 3' end clusters in our datasets, Illumina RNA-Seq 3' end coordinates are used as the validated 3' end coordinates (chrStart and chrEnd in the BedDetail file).

Lastly, annotated 3' end sites are extracted from the user-supplied annotation file and compared to the validated ends. Validated ends within the user-specified distance of annotated ends (default: 10 bases) are indicated as annotated in the output BED file, others are indicated as novel. Additionally, the script provides users with a count of total, annotated and novel validated 3' ends, both in the terminal window and as a separately generated text file.

Many *TRIMD\_end\_validator.pl* parameters are adjustable, e.g. minimum SMRT read or Illumina RNA-Seq poly(A)-tail read depth. Output files useful for researchers to select new parameter values, if necessary, include BedGraph files of Iso-Seq and Illumina RNA-Seq

polyadenylation sites, a SAM file of poly(A) tail-containing Illumina RNA-Seq reads, and BED files of 3' ends detected with each platform.

### **Transcript structure validation with *TRIMD\_transcript\_validator.pl***

The final script in the TRIMD suite, *TRIMD\_transcript\_validator.pl*, uses the BedDetail files output by *TRIMD\_start\_validator.pl*, *TRIMD\_junction\_validator.pl* and *TRIMD\_end\_validator.pl* to interrogate the SAM file of Iso-Seq CFLs. A BED file of annotated polyadenylated transcripts is also used as input, to determine the annotation status of validated transcripts.

First, the script converts CFLs in the SAM file to BED format. The script then uses this BED file to obtain the 5' start site for each CFL, and reads the BED file of validated starts to determine whether the CFL's start site is contained within a cluster that was identified by *TRIMD\_start\_validator.pl* as representing a validated 5' start site. CFLs with validated 5' start sites are stored in an array along with the genomic coordinate of the validated consensus start site.

*TRIMD\_transcript\_validator.pl* then extracts the 3' end of each CFL that has been determined to have a validated 5' end. Each end is compared to the coordinate ranges in the supplied file of validated 3' ends, and CFLs with validated 5' start sites and 3' end sites are stored in a subsequent array, along with the genomic coordinates of the validated start and end sites.

The script next investigates splice junctions in CFLs whose starts and ends are validated. CFLs with validated starts and ends that do not contain splice junctions are considered fully validated and added to a temporary BED file. Splice junctions in CFLs that contain validated

starts and ends are compared to validated splice junctions from the user-supplied file. If all splice junctions in a CFL with a validated start and end are validated, the CFL is considered fully validated and is added to the temporary BED file of validated transcripts.

At this point the BED file output contains transcripts whose structures are validated, however the start and end sites of the CFLs may need to be adjusted by a few bases to match the validated consensus start and end sites. Also, the temporary BED file likely contains multiple validated transcripts that represent the same transcript, but differ by a few bases at the CFL level in their 5' starts or 3' ends, or have minor sequence variation caused by sequencing error. *TRIMD\_transcript\_validator.pl* thus adjusts the start and end sites when necessary and compares the transcripts to each other. Transcripts whose splice junctions match each other, whose start sites arise from the same cluster of validated start sites and whose end sites arise from the same cluster of validated end sites are considered to represent the same transcript structure. Their SMRT read depth is summed and they are represented in the output BED file as a single transcript feature. The name of one of the contributing CFLs is retained as the transcript name, and the score is the sum of all SMRT reads contributing to the transcript.

Finally, the BED file of validated transcripts is compared to the user-supplied annotation file of known polyadenylated transcripts. Transcripts that match annotated transcripts within user-supplied parameters are noted as such, with the name of the annotated transcript prepended to the CFL name of the transcript. Annotated transcripts are assigned the display color for that transcript in the annotation file. *TRIMD\_transcript\_validator.pl* also provides the

user with the number of total, novel and annotated transcripts validated, both in the terminal window and as a separately generated text file.

Using a genome browser to visually inspect the BED file of validated transcripts in conjunction with the BED files of validated starts, ends and splice junctions can help the user in troubleshooting and identification of possible false negatives and false positives.

Additionally, the source code of *TRIMD\_transcript\_validator.pl* can be manipulated to generate intermediate files (a BED file of CFLs with validated 5' starts, a BED file of CFLs with validated 5' starts and 3' ends, and/or a BED file of fully validated CFLs before coordinate refinement) Lines of code that should be “uncommented” in order to produce these intermediate files are identified with comments including the word “uncomment”.

## **CHAPTER 5**

Materials and methods

## **Cell culture**

Akata, Mutu, JY and X50-7 cells were cultured in RPMI 1640 medium (Thermo Scientific, catalog no. SH30027) supplemented with 10% fetal bovine serum (FBS; Invitrogen-Gibco, catalog no. 16000), and 0.5% penicillin-streptomycin (pen/strep; Invitrogen-Gibco, catalog no. 15070), in a humidified incubator at 37°C and 5% CO<sub>2</sub>.

## **Lytic cycle induction**

Near-saturation cell cultures were diluted with equal volumes of fresh RPMI 1640 (with 10% FBS and 0.5% pen/strep) one day prior to induction. The next day, cells were pelleted and resuspended at a concentration of 10<sup>6</sup> cells/ml in fresh RPMI 1640 (with 10% FBS and 0.5% pen/strep) plus 10 µg/ml of anti-IgG (for Akata cells - Sigma-Aldrich, catalog no. I2136) or 10 µg/ml of anti-IgM (for Mutu I cells - Sigma-Aldrich, catalog no. I0759). For Illumina RNA-seq, Akata cells were harvested at 0 minutes (uninduced), 5 minutes, 30 minutes, 1 hour, 2 hours, 4 hours, 8 hours, 24 hours, and 48 hours after induction. Mutu cells were harvested at 0 hours (uninduced) and 24 hours. For Pacific Biosciences Iso-Seq, Akata cells were harvested at 20 hours and 24 hours. For deepCAGE Akata cells were harvested at 24 hours.



For phosphonoacetic acid (PAA) experiments, Akata cells were resuspended in media that contained either 200 µg of PAA/ml or no PAA in addition to anti-IgG. Cells were harvested 24 hours after treatment.

### **Transcript knockdown with GapmeR antisense oligonucleotides**

GapmeRs targeted to the BZLT12-22 transcripts (sequence: TTTGGCCAGTCTTAAT) were designed and ordered from Exiqon. For knockdown, Akata cells were pelleted and resuspended in RPMI 1640 medium supplemented with 10% FBS (no antibiotic) and maintained in antibiotic-free medium for at least 2 days. For transfection,  $3 \times 10^6$  cells per treatment were pelleted and resuspended in 100 µl Nucleofector Solution R (Lonza catalog no. VVCA-1001) with 600 pmol targeted GapmeR or negative control GapmeR A (Exiqon, catalog no. 300613-04). Cells were electroporated using program G-16 and transferred to a 6-well plate containing warm RPMI 1640 + 10% FBS. On the following day an equal volume of RPMI 1640 + 10% FBS, with 10 µg/ml anti-IgG or no anti-IgG was added to each well. 24 hours later the cells were harvested and RNA extracted (below).

### **RNA extraction**

RNA was extracted with TRIzol reagent (Life Technologies, catalog no. 15596-018) according to the vendor's protocol. Nuclear and cytoplasmic RNA isolation for qRT-PCR was carried out using a cytoplasmic and nuclear RNA purification kit from Norgen Biotek (catalog no. 2100).

### **Illumina RNA-seq**

RNA samples were treated with RNase-free DNase (Qiagen, catalog no. 79254 or Ambion, catalog no. AM1906) according to the vendor's protocol, then either poly(A)-selected or ribodepleted (Ribo-Zero; Epicentre, catalog no. MRZH11124) and prepared using the TruSeq stranded protocol (Illumina, catalog no. RS-930-2001). Ribodepleted samples underwent 101 base single-end sequencing using an Illumina HiSeq 2000 instrument. Poly(A)-selected samples underwent  $2 \times 101$  base paired-end sequencing using an Illumina HiSeq 2000 instrument. RNA samples from the GapmeR knockdown experiment were poly(A)-selected and underwent 101 base single-end sequencing. Poly(A) selection, library preparation and sequencing were performed by the University of Wisconsin Biotechnology Center, Madison, Wisconsin, USA.

### **Pacific Biosciences Iso-Seq**

For Iso-Seq, polyadenylated RNA was first selected using a Poly(A)Purist MAG kit (Life Technologies, catalog no. AM1922). 7 ug of poly(A) RNA from the 20 hour induction time point and 3.3 ug of polyA RNA from the 24 hour induction time points were pooled. Library preparation and sequencing were performed according to the Pacific Biosciences Iso-Seq protocol by the Johns Hopkins Deep Sequencing and Microarray Core Facility, Baltimore, Maryland, USA. Eight SMRT cells were used: two with a 1-2 kb RNA fraction, two with a 2-3 kb RNA fraction and four with non-size-selected RNA. Raw data was processed using RS\_IsoSeq on SMRTPortal version 1<sup>74</sup> to obtain full-length consensus isoforms.

## deepCAGE

For deepCAGE, nAnT-iCAGE libraries<sup>116</sup> were prepared from RNA extracted from two parallel samples of induced Akata cells. From each sample a portion of the RNA was treated with DNase (Ambion AM1906) and a portion not treated, for four total samples. Samples were subjected to 50-base single-end sequencing using an Illumina HiSeq 2500 instrument. Library preparation and sequencing were performed by DNAform, Yokohama, Japan.

## Data acquisition

Pacific Biosciences SMRT sequence data for type III latency lymphoblastoid cell lines was downloaded from NCBI SRA, accession number SRP036136<sup>102</sup>. RNA-Seq data for JY cells was downloaded from NCBI SRA, accession number SRR364065<sup>34,117</sup>.

## Sequence Alignments

Illumina RNA-Seq reads were aligned using indexes containing both the human (hg19 assembly) and the Akata EBV (KC207813.1<sup>43</sup>) genomes. For these analyses, the circular EBV Akata genome was split between the BBRF3 and BGLF3 genes (between positions 107954 and 107955) rather than the terminal repeats to allow for the detection of LMP2 transcripts, which span the terminal repeats. Alignments were performed as noted for different analyses using Novoalign version 2.08.02 (Novocraft; -o SAM -r R, default options), Bowtie<sup>118</sup> version 2 (-library-type fr-firststrand, default options) and STAR<sup>115</sup> version 2.3.01 (default options unless otherwise noted). For analyses that used both paired-end and single-end RNA-Seq data (see Chapter 2), only the first read of the paired-end sequencing data was analyzed in order to maintain consistency.

Pacific Biosciences Iso-Seq full-length consensus isoforms (CFLs) were aligned and mapped with GMAP<sup>114</sup> release 2014-07-21 to the human (hg 19 assembly) and Akata EBV (KC207813.1<sup>43</sup>) genomes. The circular EBV Akata genome was split as above between positions 107954 and 107955. Full-length isoforms unpolished by Quiver were used in these analyses as we observed that Quiver polishing sometimes obscured introns and prevented discrimination of overlapping transcripts in the gene-dense EBV genome. Only reads mapping to a single location were retained (argument `-n 1`).

deepCAGE tags were mapped with STAR version 2.3.01 (`--outFilterMultimapNmax 100 --outSAMprimaryFlag AllBestScore`, to allow detection of potential start sites in repeat regions).

For Pacific Biosciences SMRT sequence data from type III latency lymphoblastoid cell lines, reads were first oriented using their poly(A) tails. Reads ending with AAAAAAAAA and reads beginning with TTTT\*TTT\* were extracted. Reads beginning with TTTT\*TTT\* and their quality scores were reversed to produce fastq files of “sense” oriented RNA. These reads were then aligned with GMAP<sup>114</sup> release 2014-07-21 to the Akata EBV (KC207813.1<sup>43</sup>) genome, split as above between positions 107954 and 107955.

### **Strand specificity calculation for Illumina RNA-Seq**

Strand specificity was determined using a set of highly-expressed cellular genes without known antisense transcription (GAPDH, ACTB, RPL8, EEF2, RPS6, RPLP1, GNB2L1 and PFN1). To confirm the absence of antisense transcription, Bowtie2/TopHat aligned reads

were visualized on the IGV genome browser<sup>119,120</sup> and each gene was visually inspected for clusters of antisense reads that might represent previously unannotated antisense transcripts. No likely antisense transcript was discovered for these genes. To calculate strand-specificity, coverage files were generated using IGVtools<sup>119,120</sup> from Bowtie2/TopHat aligned data, containing the number of reads covering each genomic coordinate for each strand of each gene. The gene coverage files were converted to exon coverage files using the BedTools command intersectBed<sup>121</sup> and an exon bed file from the hg19 assembly of the human genome. At all nucleotide positions with 200 or more sense reads, the number of antisense reads was divided by the number of sense reads and multiplied by 100 to obtain the percent background antisense reads. The mean and standard deviation were then calculated to determine the average level and variability of antisense background.

### **Calculation of EBV-mapping reads and induction level**

The number of single-end or first-in-pair reads aligned to the EBV genome by Bowtie2/TopHat with a primary alignment (SAM FLAG code 0 or 16) was divided by the total number of reads with a primary alignment on either genome and multiplied by 100. To allow better comparison between poly(A)-selected and ribodepleted datasets, reads overlapping the highly-expressed, non-polyadenylated EBER genes were removed in both directions prior to calculating the percentage of reads mapped to EBV. The fold change between induced and uninduced conditions was calculated by dividing the percentage of reads mapped to EBV at 24 hours post-induction by the percentage of reads mapped to EBV at 0 minutes post-induction.

### **Determination of transcribed EBV genome loci**

Coverage files containing the number of reads covering each EBV genomic coordinate were generated with IGVtools for each strand of the EBV genome from Bowtie2/TopHat aligned files. For the purposes of this analysis a nucleotide position was considered to be transcribed if it met both of the following criteria: 1) the number of reads mapping to the respective base was greater than 4, and 2) the number of reads mapping to that base was higher than the expected antisense background from opposite strand reads (i.e., the opposite strand read numbers multiplied by the average antisense background) plus 4 standard deviations. Transcription was considered to be “known” if the base was contained within a GenBank-annotated exon (KC207813.1<sup>43</sup>).

### **Quantification of gene expression using strand information**

For the analysis in Chapter 2, expression levels of known EBV and cellular genes were quantified from Bowtie2/TopHat aligned files using SAMMate<sup>122</sup>. Quantification was made strand-specific by using a separate annotation file for each strand in conjunction with a SAM file containing only reads aligning to the strand matching the annotation file. To allow for direct comparison of RPKM (reads per kilobase of transcript per million mapped reads) values generated for different strands, these RPKM values were multiplied by the ratio of the sum of read counts for that strand (as determined by SAMMate) to the total number of mapped reads for both strands of the two genomes (as determined by Bowtie2/TopHat). Levels of antisense expression to known genes were quantified using the annotation file for one strand together with a SAM file containing only reads aligning to the opposite strand. Antisense RPKM values were corrected as described above. Non-strand specific expression

values for previously known genes were obtained in the poly(A)-selected dataset by combining sense and antisense read counts for each gene from strand-specific SAMMate output and dividing by the gene's transcript length in thousands and by the number of million mapped reads.

For the analysis in Chapter 3, transcript abundance estimates were generated from Illumina RNA-Seq reads using RSEM<sup>123</sup> with an annotation file including the human genome GRCh38 assembly and GenBank Akata annotation (KC207813.1<sup>43</sup>), plus transcript coordinates representing the novel transcript BCLT2.

### **Comparison of novel EBV gene expression to cellular transcript levels**

Prior to the determination of the structures or the novel EBV transcripts, RPKM or TPM values could not be calculated to measure abundance at the transcript level. To compare EBV transcription levels with cellular transcription levels for the analysis in Chapter 2, per-base EBV read counts were first normalized by dividing by the total number of million reads mapped by Bowtie2/TopHat to either genome. This value is the Reads per Million mapped reads (RPM) at each position. To determine the expression level of the top quartile of cellular genes, cellular gene expression was quantified with SAMMate<sup>122</sup> as described in the previous section. The strand-specific RPKMs for the plus and minus strand cellular genes were combined, genes with fewer than two reads aligning were removed, and the top quartile expression level was determined. To allow comparison of per-base EBV read levels to the top quartile of cellular genes, the RPKM value representing the 75<sup>th</sup> percentile of cellular genes was converted to an RPM value by multiplying by 1000 and dividing by the read length (101).

## **Determination of Iso-Seq and Illumina RNA-Seq coverage of cellular genes**

An annotation file of human (hg 19 assembly) RefSeq<sup>124</sup> RNA transcripts with either Reviewed or Validated status was downloaded using the UCSC Table Browser<sup>125</sup>. A transcript was considered to have full-length Iso-Seq coverage if an Iso-Seq CFL's 5' and 3' ends mapped within 50 and 20 bp respectively of the annotated transcript's 5' and 3' ends. A transcript was considered to have partial coverage if an Iso-Seq CFL's 3' end aligned within 20 bp of the annotated transcript's 3' end, its 5' end did not map within 50 bases of the annotated transcript's 5' end, and at least five Illumina RNA-Seq reads mapped by STAR to the first exon within 100 bp of the 5' end (this reduces false calls of "partial coverage" for non-expressed transcripts that share a 3' end with expressed transcripts).

## **Identification of transcription start sites**

Iso-Seq full-length consensus isoform 5' ends mapping without softclipping to within 8 bp of each other on the genome were considered a single candidate transcription start site. The consensus transcription start sites were determined by calculating weighted averages of the start coordinates, with weighting based on the number of SMRT reads starting at each coordinate.

Clusters of start sites in mapped deepCAGE data were extracted using Paraclu<sup>96</sup>. Clusters were required to be less than 20 bases long, contain at least 15 CAGE tags and have a relative density fold change of at least 2. Consensus transcription start sites were determined by calculating weighted averages of the start coordinates, with weighting based on the number of CAGE tags starting at each coordinate. Consensus start sites appearing within 2



bases of each other in at least three of the four CAGE samples were used to validate SMRT consensus transcription start sites.

SMRT consensus transcription start sites were considered validated if they were within 3 bases of CAGE consensus transcription start sites.

### **Identification of splice junctions**

For the analysis in Chapter 2 splice junctions were identified using TopHat<sup>126</sup> version 2.0.6 (default options). Splice junctions in the poly(A)-selected RNA dataset were reported if they were supported by at least 5 reads. Splice junctions in the ribodepleted dataset were reported if they were supported by at least 10 reads, because approximately twice as many reads from this dataset mapped to the EBV genome (see Table 2). Because 101-bp reads cannot be assigned definitively to specific splice junctions within the repeats, junctions with a donor and/or acceptor in the W-repeat region (bases 75265-98628 on the inverted Akata genome) were ignored in this analysis.

For the analysis in Chapter 3 splice junctions were identified by GMAP (argument `-f` introns) for Iso-Seq CFLs and by STAR version 2.3.01 for Illumina reads. Splice junctions were considered validated if they were detected by both Iso-Seq and Illumina RNA-Seq datasets. To find Illumina reads mapping to splice junctions in the IR1 W repeat region (bases 75265-98628 on the inverted Akata genome), the STAR `outFilterMismatchNmax` argument was set to 100 to report alignments for reads that mapped up to 100 times. A repeat splice junction was considered to have been detected by Illumina RNA-Seq if any of the set of possible alignments was reported by STAR. Illumina RNA-Seq read depth for

repeat splice junctions was normalized by dividing by the number of equivalent genomic alignments possible.

### **Identification of polyadenylation sites**

For the analysis in Chapter 2 Illumina RNA-Seq reads with runs of 5 or more 'T's at their 5' end were extracted from Novoalign-generated SAM alignment files from ribodepleted RNA isolated from cells treated with anti-IgG for 24 hours. All reads with 5' poly(T)s were then mapped to the Akata genome using BLAST version 2.2.28+<sup>127</sup>. Reads with mismatches at the first two or more positions of the read were identified as candidate poly(A) tail reads. Mapping data for this set of reads were then used to generate a BED file for visualization on a genome browser.

For the analysis in Chapter 3 a combined Iso-Seq and Illumina RNA-Seq approach was used. To identify 3' ends represented in the Iso-Seq data, full-length consensus isoform 3' ends aligning within 8 bp of each other on the genome were considered a single candidate polyadenylation site. The consensus polyadenylation sites were determined by calculating weighted averages of the end coordinates, with weighting based on the number of SMRT reads ending at each coordinate. Next, Illumina RNA-Seq reads containing putative poly(A) tails were extracted from STAR-generated SAM alignment files using the following criteria: reads identified by FLAG code as being first-of-pair (for paired-end sequencing) that end with a run of at least 5 As, at least 2 of which are softclipped (plus strand) or that start with a run of at least 5 Ts, at least 2 of which are softclipped (minus strand). Illumina reads with putative poly(A) tails were extracted from 22 different RNA-Seq datasets representing multiple time points relative to anti-IgG induction, and both poly(A)-selected and

ribodepleted RNA preparations. The alignment position of the softclipped-adjacent bases was taken to represent a candidate polyadenylation site, with sites situated within 8 bases of each other considered single candidate polyadenylation sites. The Illumina consensus polyadenylation site was determined using a read-end-depth weighted average as for the Iso-Seq isoform 3' ends (above). Candidate polyadenylation sites were considered validated if they were supported by at least 5 SMRT reads and the presence of an Illumina candidate polyadenylation site on the same strand within 4 bases upstream or 10 bases downstream. As Illumina consensus polyadenylation sites were almost always downstream of Iso-Seq consensus polyadenylation sites, the coordinate of the Illumina consensus polyadenylation site was considered to be the validated end coordinate.

### **Transcript validation**

Each Iso-Seq CFL was examined to determine whether its 5' end, 3' end and splice junctions (if any) met the criteria described above for validation. When reporting transcripts, 5' and 3' ends that formed part of validated consensus transcription start sites and polyadenylation sites were adjusted to match the validated consensus sites, if necessary. Finally, Iso-Seq CFLs that had matching validated transcription start sites, polyadenylation sites and splice junctions (if any) were collapsed into "validated transcripts".

### **Calculation of Coding Potential**

For the analysis in Chapter 2 the coding potential of known and novel transcripts was calculated using the Coding Potential Calculator<sup>88</sup>. For transcripts with ambiguous 5' or 3'

ends, several sequences of varying length were used as input. Representative results are shown.

For the analysis in Chapter 3 sequences of validated isoforms were analyzed for coding potential and the presence of open reading frames with the Coding Potential Assessment Tool<sup>103</sup>.

### 5' and 3' RACE

5' and 3' RACE (Rapid Amplification of cDNA ends) was performed using the SMARTer RACE cDNA Amplification Kit (Clontech catalog number 634924). cDNA was prepared with Primer A or with Random Primer Mix to detect polyadenylated and non-polyadenylated transcripts, respectively. Thermal cycling was performed according to the manufacturer's Program 2. RACE PCR products were cloned using a TOPO TA cloning kit (Invitrogen catalog number K4575) and sequenced using the Sanger method.

**Table IV: 5' and 3' RACE Primers**

EBNA3A		
5' RACE primer 1	CCGGCGGCCAGGGT	TTCAGTCTCCA
5' RACE primer 2	ACGTGACACCTACGGCCACCTGTGCA	
5' RACE primer 3	GCTCTCCGCGTCCTCACTTCTTCCCG	
5' RACE primer 4	TGCCCTGTTCCGTTCGTTTGCCCGCT	
5' RACE primer 5	TGCACAGGTGGCCGTAGGTGTCACGT	
5' RACE primer 6	ACACCGATCACCAGACGACTCCCAC	
5' RACE primer 7	TCCCACCCAGCCGGATCTCCCT	
EBNA3B		
5' RACE primer 1	TGTGAACCCAACGCAGGCTCCAGTGA	
5' RACE primer 2	CACGTCGTGCTAGGTCACTTTCGGCAGA	
3' RACE primer 1	GCCAGCACTGTACGTTGTTGCATGCCG	

## qRT-PCR

For the analysis in Chapter 2, cDNA was synthesized from RNA extracted from PAA-treated cells at the 24 hour time point using the Superscript III First Strand Synthesis System (Invitrogen catalog number 18080-051) with oligo(dT) primers. qRT-PCR reactions were carried out using iQ SYBR Green Supermix (Bio-Rad catalog number Cat No: 170-8882) on a Bio-Rad CFX96 instrument as follows: 1 µl cDNA product was denatured for 3 minutes at 95°C and amplified for 40 cycles of 15 seconds denaturation at 95°C and 1 minute annealing/extension at 60°C. Transcript abundance was quantified using the Comparative  $C_T$  method ( $2^{-\Delta\Delta C_T}$ ).

For the analysis in Chapter 3, cDNA was synthesized from RNA using an iScript cDNA synthesis kit (Bio-Rad catalog no. 170-8891) according to the vendor's protocol. Quantitative PCR was performed using iQ SYBR green Supermix (Bio-Rad, catalog no. 170-8882) on a Bio-Rad CFX96 instrument. 1 µl of cDNA product was denatured for 3 min at 95°C and amplified for 40 cycles of 15-s denaturation at 95°C and 1-min annealing/extension at 58°C. Total RNA transcript abundance was quantified using the comparative  $C_T$  method ( $2^{-\Delta\Delta C_T}$ ) normalized to ACTB. Nuclear to cytoplasmic ratios were calculated as  $2^{-\text{NuclearCt}-\text{CytoplasmicCt}}$ .

**Table V: RT-PCR primers**

ACTB	CACTCTTCCAGCCTTCCTTC	GTACAGGTCCTTGCGGATGT
Zta (Ch. 2)	GAAGCCACCCGATTCTTGTAT	CGACGTACAAGGAAACCACTAC
Zta (Ch. 3)	CACGACGTACAAGGAAACCA	GAAGCCACCTCACGGTAGTG
W1-W1	TCGGGCCAGAGCCTAGGG	TGGTCCAGGGACTTCACTTC
W1-BHRF1	AGGGGAGACCGAAGTGAAGT	CCCTTGTGTGAATAGGCCATC
W1-W2	AGGGGAGACCGAAGTGAAGT	CCTTCTACGGACTCGTCTGG
LMP2A 1-2	CCTACTCTCCACGGGATGAC	CGGTGTCAGCAGTTTCCTTT
Junction A	GCAGGTCAGACTTGGTGCTT	GAGTTGTTTCCGCCATCGT
Junction C	GCCCGAGGAGCTGTAGACC	GAGTTGTTTCCGCCATCGT
Junction D	CGATAGAGGGCCAGGTAGTG	GAGTTGTTTCCGCCATCGT
Junction E	GCAAAGGCAGGTCTTTCTCA	GAGTTGTTTCCGCCATCGT

**Strand-specific qRT-PCR**

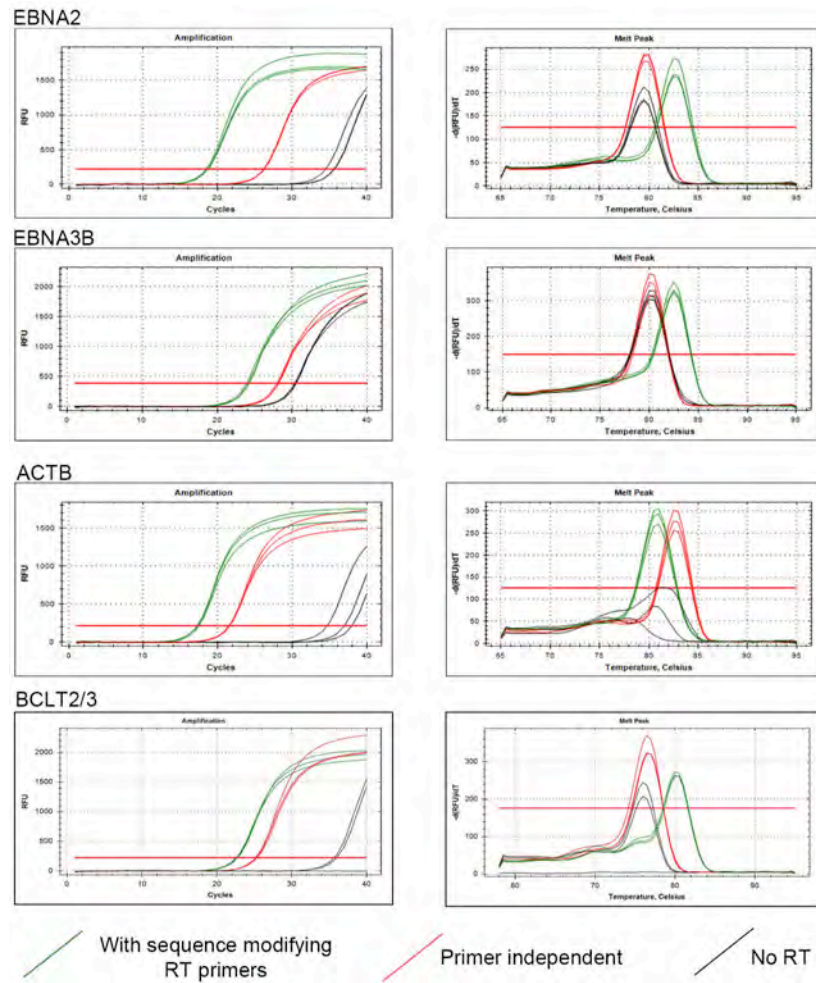
The method of Feng *et al.*<sup>128</sup> was used for strand-specific quantitative reverse-transcription PCR. cDNA was synthesized from RNA at 65°C for 50 minutes using gene-specific sequence modifying primers (or non-sequence modifying reverse primers for Zta and Kcnq1ot1) and ThermoScript reverse transcriptase (Life Technologies, catalog no. 12236-022) according to the manufacturer's protocol. Quantitative PCR was performed using iQ SYBR green Supermix (Bio-Rad, catalog no. 170-8882) on a Bio-Rad CFX96 instrument. 1 µl of cDNA product was denatured for 3 min at 95°C and amplified for 40 cycles of 15-s denaturation at 95°C and 1-min annealing/extension at 58°C. Melting-curve analysis was performed from 58 to 95°C with a ramp of 0.5°C/5 s to confirm strand specificity (Figure 26). Total RNA transcript abundance was quantified using the comparative CT method ( $2^{-\Delta\Delta CT}$ ) normalized to ACTB. Nuclear to cytoplasmic ratios were calculated as

$$2^{-\text{NuclearCt}-\text{CytoplasmicCt}}$$

**Table VI. cDNA and PCR primers for strand-specific qRT-PCR**

EBNA2		
cDNA primer	GCAACCCCTAACGT <sup>T</sup> TCACC <sup>gggcCgg</sup> GAACCGG*	
PCR primers	GCAACCCCTAACGT <sup>T</sup> TCACC CGGGGAAGAGAATGGGAGC	
Zta		
cDNA primer	CACGACGTACAAGGAAACCA	
PCR primers	CACGACGTACAAGGAAACCA GAAGCCACCTCACGGTAGTG	
EBNA3B		
cDNA primer	TGGCAT <sup>T</sup> TGTACAGATACCACGA <sup>gcggCg</sup> GACCAAAAC*	
PCR primers	TGGCAT <sup>T</sup> TGTACAGATACCACGA CCGAAAGTGACCTAGCACGA	
BCLT2/3		
cDNA primer	GTTCAGTTCGTCGAGTGCT <sup>cgCggc</sup> GGAACAG*	
PCR primers	GTTCAGTTCGTCGAGTGCT CGCCAACAAGGT <sup>T</sup> CAAT <sup>T</sup> TT <sup>T</sup>	
ACTB		
cDNA primer	GTACAGGTCT <sup>T</sup> TTGCGGATGT <sup>ttAtaTa</sup> ACACT <sup>T</sup> TCATG*	
PCR primers	GTACAGGTCT <sup>T</sup> TTGCGGATGT CACTCT <sup>T</sup> TCCAGCCTTCCTTC	
Kcnq1ot1		
cDNA primer	GCTGATAAAGGCACCGGAAGGAAA	
PCR primers	GCTGATAAAGGCACCGGAAGGAAA TACCGGATCCAGGT <sup>T</sup> TTGCAGTACA	

\*Lower case letters indicate sequence-modifying bases



**Figure 26.** Strand specific qRT-PCR. Representative Ct curves and melting curves are shown for each set of primers. Sequence modifying RT primers increase (EBNA2 and EBNA3B) or decrease (ACTB) the melting temperature of the PCR amplicons relative to the unmodified amplicon.

## FISH and immunofluorescence

For the analysis in Chapter 2, fluorescence *in situ* hybridization (FISH) was performed with custom Stellaris RNA FISH probes (Biosearch Technologies) using CAL Fluor Red 610, according to the vendor's protocol.  $10 \times 10^6$  24 hour induced or uninduced Akata cells were used per treatment. Images were captured on a Leica DMRXA2 Deconvolution upright



microscope. 3D imaging of Akata cells was acquired using a 100x/1.35 oil objective on a motorized XYZ-stage with a Cooke SensiCAM camera using Slidebook software.

For the analysis in Chapter 3, immunolabeling was performed simultaneously with FISH using a modified version of the Stellaris protocol with mouse anti-EBV EA-D-p52/50 antibody (EMD Millipore catalog no. MAB8186) and Alexa Fluor 488 goat anti-mouse secondary antibody (Life Technologies catalog no. A11001). Briefly,  $10 \times 10^6$  24 hour induced or uninduced Akata cells were washed, fixed in freshly made fixation buffer and permeabilized for approximately 24 h. Cells were hybridized overnight at 37°C using freshly made hybridization buffer with 50 nM FISH probe or no probe, then incubated with the primary antibody (diluted 1:200) for 3 hours at room temperature, washed, and incubated for 30 minutes with the secondary antibody (diluted 1:500) and 5 ng/ml DAPI at 37°C in the dark. Cells were then washed a final time, mounted on slides with Prolong Diamond mounting medium (Life Technologies catalog no. P36961) and cured in the dark for two days. Imaging was performed using a Zeiss Axioplan 2 upright microscope and Z-stacks were deconvolved using Slidebook software, version 6 (Intelligent Imaging Innovations).

## **CHAPTER 6**

Discussion and future directions

### **Increasing the functional capacity of the genome**

Our findings of hundreds of new transcripts arising from pervasive transcription of the EBV genome indicate that the functional capacity of the genome extends far beyond a set of canonical open reading frames situated between TATA boxes and polyadenylation signals. Using our long-read sequencing based approach to identify new transcript structures we have revealed many different types of transcripts, including wholly novel transcripts, antisense transcripts, chimeric transcripts that fuse multiple annotated genes, and novel isoforms that feature alternative splicing, extended or truncated untranslated regions, and even truncated reading frames of known genes.

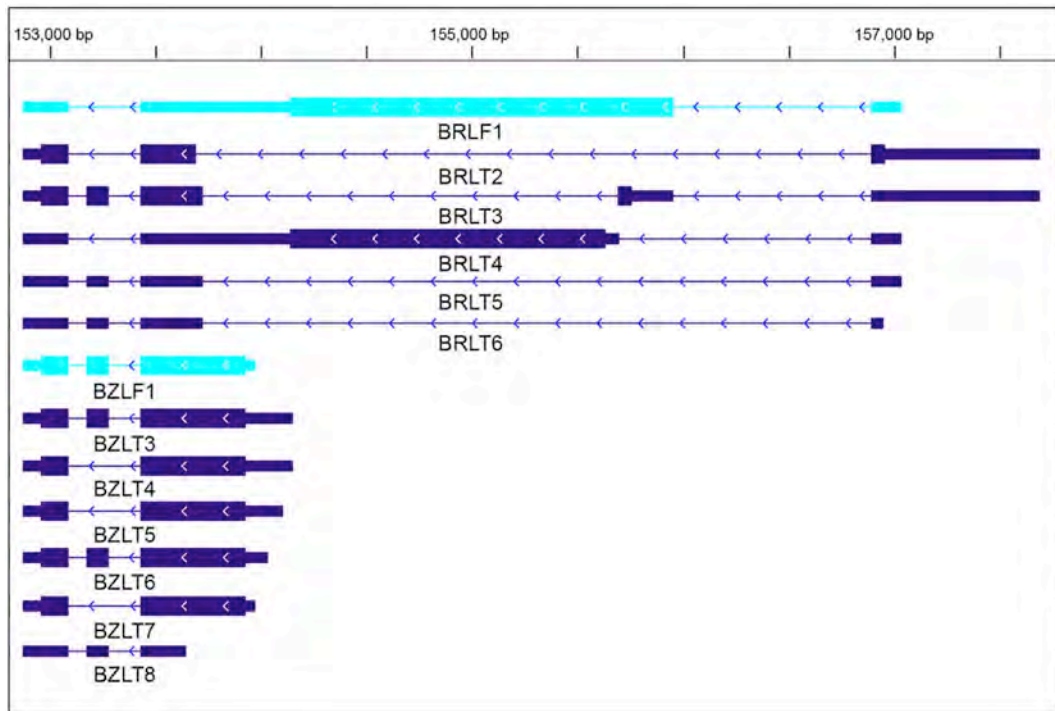
Many of the novel transcripts contain annotated open reading frames but extended or truncated untranslated regions (UTRs), especially 5' UTRs. Untranslated regions play major roles in mRNA regulation, often providing binding sites for regulatory proteins or RNAs that control mRNA cellular localization, translation or degradation<sup>129</sup>. Eukaryotes are known to use UTR variations to differentially control mRNA in different cell types and differentiation states<sup>130,131</sup> and even between individual cells of the same type<sup>132</sup>. The abundance of related isoforms in our dataset suggest that the virus is exploiting UTR variations as a mechanism of mRNA modulation.

Alternative transcription start site usage sometimes goes beyond altering just the gene's 5' UTR and affects the open reading frame. For example, a truncated form of the LMP1

protein arising from alternative promoter usage during lytic replication in some EBV strains negatively regulates LMP1 signaling pathways and promotes degradation of full-length, latency-associated LMP1<sup>133,134</sup>. Many of the novel transcripts reported here contain similar smaller, in-frame ORFs of annotated genes. Informatic analysis of many of these transcripts indicates that they are likely coding mRNA (data not shown). While we are unable to verify in this study whether or not these isoforms are translated, ribosomal profiling after treatment with harringtonine reveals pileups of initiating ribosomes at many downstream in-frame start codons within known ORFs of KSHV<sup>33</sup>, HCMV<sup>27</sup> and humans<sup>135</sup>. The resulting N-terminal truncated proteins can play roles in cellular stress-response signaling<sup>136</sup> and impact protein localization<sup>137</sup>.

In addition to truncated reading frames, some novel transcripts contain reading frames that are extended through alternative splicing (e.g. see Figures 25A and 27) and even chimeric open reading frames (e.g. see Figure 22). The EBV fusion protein Raz, generated from an alternatively spliced transcript that produces a chimeric Rta-Zta protein, regulates Rta<sup>138</sup>. The novel transcripts presented here raise the possibility that the phenomenon of fusion proteins is more widespread in the EBV proteome.

Ribosomal profiling in KSHV<sup>33</sup> and HCMV<sup>27</sup> has also revealed extensive translation of previously unannotated ORFs. Often those ORFs were not previously identified because they are shorter than traditional ORFs, encoding 100 or fewer amino acids. Short proteins are also abundant in the human proteome<sup>139</sup>, and both viral and eukaryotic small peptides have been shown to be functional<sup>140,141</sup>. In some cases these short ORFs are translated from putative noncoding transcripts like KSHV's PAN<sup>33</sup>, raising the possibility that some of the



**Figure 27.** TRIMD-validated BRLF1 and BZLF1 isoforms. Light blue = GenBank-annotated isoforms. Dark blue = novel isoforms.

newly-identified EBV transcripts that appear to be noncoding may in fact be translated. In other cases the short ORFs are present in the putative 5' UTR of mRNA. It has long been known that upstream ORFs regulate at least one HCMV gene<sup>142</sup> and ribosomal profiling suggests the mechanism may be widespread in HCMV<sup>27</sup>. Functional upstream ORFs are also prevalent in the human genome, regulating mRNA translation and degradation<sup>143-145</sup>. Several of the novel EBV transcripts with extended 5' UTRs contain upstream ORFs (e.g., BZLT3/4/5: Figure 27), which may impact the translation or stability of those transcripts.

In addition to the many transcripts that are novel isoforms of known protein-coding genes with altered ORFs or UTRs, many of the novel transcripts are predicted to be noncoding.

This finding of substantial numbers of noncoding transcripts parallels findings from the human transcriptome, in which next-generation sequencing has revealed many long noncoding transcripts<sup>78,146</sup>. Some of these appear to be noncoding isoforms of coding genes (e.g. some LMP2 isoforms – Figure 25 and BZLT8 – Figure 27). Others represent novel transcription of putative intergenic regions (e.g. BCLT2-4 – Figure 21). Some are antisense to known protein coding genes (e.g. BZLT12-22 – Figures 22 & 23): these are particularly abundant at latency loci. While some of the putative noncoding transcripts may encode novel short ORFs, those that we tested are mostly localized to the nucleus (Figures 6, 7, 8, 21 & 25), suggesting noncoding roles. Several cellular lncRNAs that localize to the nucleus have been determined to function as transcriptional regulators. Interestingly, knockdown of the noncoding RNA arising from EBV's OriP locus leads to a widespread repression of viral transcription<sup>147</sup>, and knockdown of several different noncoding RNAs in MHV68 leads to altered expression of a lytic viral protein<sup>30</sup>.

### **Novel isoforms of latency-associated transcripts during reactivation**

While many of the novel EBV transcripts are structural variations of lytic genes, latency loci are remarkable for the abundance of novel isoforms relative to annotated isoforms. LMP2 is particularly striking: neither LMP2A or LMP2B is detected in its annotated form in Iso-Seq CFLs, but multiple isoforms arising from alternative promoters and using alternative splicing are present (Figure 25). Isoforms of EBNA2, EBNA3A and EBNA3C arising from downstream transcription start sites are also present in our dataset (Figure 23 and data not shown). Full-length annotated forms of the EBNA transcripts are not detected by Iso-Seq, though that is possibly due to their length (see Chapter 3). Both LMP1 and EBNA1 are known to use alternate promoters during viral replication<sup>148-150</sup>; potentially other latency-

associated genes use a similar mechanism to provide an additional level of transcriptional and translational control under different gene expression programs. While transcription and translation of LMP2 and EBNA2/3A/3B/3C have been observed by others using microarrays and Western blots<sup>23,24,81,82</sup>, this level of structural detail was not observable with those technologies.

### **Mechanics of pervasive transcription**

The mechanisms by which EBV achieves this high diversity of transcripts have yet to be determined. Many of the novel transcripts appear with the same timing as viral Late genes (Figures 6, 7, 8, 21 & 24) which, unlike the Immediate Early or Early genes are under the transcriptional control of the viral pre-initiation complex (vPIC)<sup>22,151</sup>. While the vPIC is known to recognize the Late-gene associated genomic TATT motif, possibly it can also initiate transcription at alternate sites late in reactivation. Recent reports about cellular transcriptional control may offer some clues about readthrough transcription. Rutkowski *et al.* report massive transcriptional readthrough in the cellular genome of fibroblasts in response to HSV-1 infection<sup>152</sup>, while Vilborg *et al.* make a comparable observation in neural cells in response to osmotic stress<sup>153</sup>. These similar observations in very different systems hint at the possibility of a broadly used mechanism, especially considering that the osmotic stress-induced readthrough was dependent on intracellular Ca<sup>2+</sup> release, a process critical to both HSV-1 infection<sup>154</sup> and EBV reactivation<sup>155</sup>.

## **Refinement of EBV genome annotation**

In addition to resolving the isoform structures of nearly 200 novel transcripts, using TRIMD we are able to refine the annotation for nearly two thirds of annotated transcripts. This assigns genomic start and end coordinates based on experimental evidence from multiple platforms, refining transcript ends that had previously been annotated based on the presence of TATA boxes and polyadenylation signal motifs in the genomic sequence<sup>43,97</sup>. Most refined ends correspond as expected with annotated ends, with transcription start sites occurring 25 to 35 bases downstream of TATA boxes and polyadenylation sites occurring 10 to 30 bases downstream of polyadenylation signals. In addition, we are able to determine 5' and 3' transcript ends for many genes that do not have canonical TATA boxes or polyadenylation signals near their open reading frames. The updated annotation is available as Appendix 10 and at <https://github.com/flemingtonlab/public>.

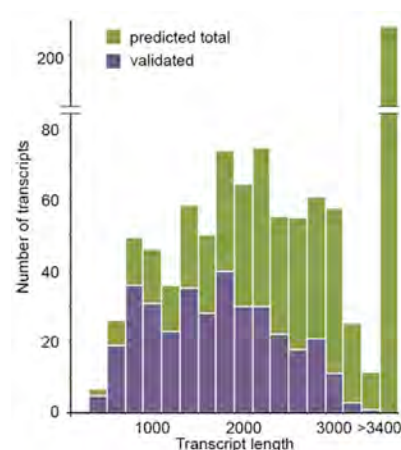
## **Completeness of annotation**

While the set of isoforms we identified in Chapter 3 more than quadruples the size of the catalog of known EBV transcripts, we believe that further transcripts exist that were undetectable in this study. A sizable proportion of expressed cellular transcripts was not represented in our Iso-Seq dataset, with longer transcripts especially lacking representation (Figure 14C). To estimate how many polyadenylated EBV transcripts may remain undiscovered, we used information about the lengths of validated EBV isoforms and the proportion of annotated cellular transcripts of each length with full-length Iso-Seq coverage (Figure 14C). Further, we used the proportion of annotated EBV transcripts longer than 3,233 bases (the length of the longest EBV isoform validated by TRIMD) to estimate how



many longer transcripts might be missing from our dataset. Using these measures, we estimate that over 900 polyadenylated transcripts are expressed during EBV reactivation (Figure 28). Some of these transcripts were partially captured by Iso-Seq and TRIMD, as indicated by features in our sets of validated 5' starts, splice junctions and 3' ends that do not correspond to fully validated isoforms (see for example Figure 20).

It is also notable that we observed deeper and more extensive Illumina short-read coverage of the genome from ribodepleted RNA than from poly(A)-selected RNA (Figure 4). This indicates that there are likely many additional lytic EBV transcripts that are not polyadenylated, and therefore not detectable by the Iso-Seq method.



**Figure 28.** Length distribution of detected and predicted polyadenylated EBV isoforms. Number of predicted isoforms is calculated based on the proportion of full-length cellular transcripts captured (Figure 1C) and the percentage of annotated EBV transcripts longer than the longest validated isoform.

## Future directions

As indicated above, capturing the structures of extremely long transcripts and of non-polyadenylated transcripts will be necessary to create a complete annotation of the EBV transcriptome. New technologies will need to be developed and implemented to accomplish this. While this will be extremely helpful for transcript quantification purposes and the interpretation of genome-modification experiments, possibly more important than complete transcriptome annotation is identification of the function of these novel transcripts.

Knockdown of some noncoding transcripts in both EBV<sup>147</sup> and MHV68<sup>30</sup> have had a global

impact on the viral lytic cycle; further investigation is necessary to elucidate the functions and mechanisms of other transcripts.

Important clues to the function of many transcripts may be gained by determining their coding status using ribosomal profiling. This method has revealed many novel and truncated reading frames in both KSHV<sup>33</sup> and HCMV<sup>27</sup>, and can indicate the coding status of EBV transcripts in a more rigorous and accurate way than sequence analysis. This could provide valuable insight into, for example, the alternative promoter usage at the LMP2 locus that appears to give rise to a collection of both nuclear noncoding and cytoplasmic coding transcripts (Figure 25).

Valuable information may also be gleaned from recently developed single-cell sequencing technologies. Though induced reactivation in Akata cells is believed to be largely synchronous there is always a small percentage of cells undergoing spontaneous reactivation, and single-cell imaging methods have shown that some cells respond differently to BCR-crosslinking, delaying their entry into the lytic cycle or causing a switch to a different form of latency<sup>82,156</sup>. This obscures the temporal dynamics of Immediate Early, Early and Late genes as observed in bulk sequencing (e.g., Figure 5), based on which we inferred that most novel transcripts detected here are Late. There is further the possibility that different viral or cellular transcripts are expressed in different cells at the same stage of reactivation, which can only be determined by analysis at the single-cell level.

This work is based largely on the analysis of transcripts present in EBV-infected cells at different stages of reactivation. Lytic genes are also known to be expressed upon *de novo*

infection of cells<sup>157-159</sup>, though it is not currently known whether the pervasive transcription associated with lytic reactivation occurs. Viral and cellular RNA is also known to be packaged into EBV virions and exosomes to facilitate infection<sup>157,160</sup>, though a global examination of the transcripts included has not been undertaken.

## **LIST OF REFERENCES**

- 1 Epstein, M. A., Achong, B. G. & Barr, Y. M. Virus Particles in Cultured Lymphoblasts from Burkitt's Lymphoma. *Lancet* **1**, 702-703 (1964).
- 2 Murray, P. & Young, L. Hodgkin's lymphoma: molecular pathogenesis and the contribution of the Epstein-Barr virus. in *Epstein-Barr Virus* (ed E. Robertson) (Norfolk: Caister Academic Press, 2005).
- 3 Raab-Traub, N. Epstein-Barr virus in the pathogenesis of NPC. *Semin Cancer Biol* **12**, 431-441 (2002).
- 4 Takano, Y. *et al.* The role of the Epstein-Barr virus in the oncogenesis of EBV (+) gastric carcinomas. *Virchows Archiv* **434**, 17-22 (1999).
- 5 Sixbey, J. W., Nedrud, J. G., Raab-Traub, N., Hanes, R. A. & Pagano, J. S. Epstein-Barr virus replication in oropharyngeal epithelial cells. *N Engl J Med* **310**, 1225-1230 (1984).
- 6 Karajannis, M. A., Hummel, M., Anagnostopoulos, I. & Stein, H. Strict lymphotropism of Epstein-Barr virus during acute infectious mononucleosis in nonimmunocompromised individuals. *Blood* **89**, 2856-2862 (1997).
- 7 Anagnostopoulos, I., Hummel, M., Kreschel, C. & Stein, H. Morphology, immunophenotype, and distribution of latently and/or productively Epstein-Barr virus-infected cells in acute infectious mononucleosis: implications for the interindividual infection route of Epstein-Barr virus. *Blood* **85**, 744-750 (1995).
- 8 Niedobitek, G. *et al.* Epstein-Barr virus (EBV) infection in infectious mononucleosis: virus latency, replication and phenotype of EBV-infected cells. *J Pathol* **182**, 151-159 (1997).
- 9 Hochberg, D. *et al.* Demonstration of the Burkitt's lymphoma Epstein-Barr virus phenotype in dividing latently infected memory cells in vivo. *Proc Natl Acad Sci U S A* **101**, 239-244 (2004).
- 10 Yao, Q. Y., Ogan, P., Rowe, M., Wood, M. & Rickinson, A. B. Epstein-Barr virus-infected B cells persist in the circulation of acyclovir-treated virus carriers. *Int J Cancer* **43**, 67-71 (1989).
- 11 Yao, Q. Y., Rickinson, A. B. & Epstein, M. A. A re-examination of the Epstein-Barr virus carrier state in healthy seropositive individuals. *Int J Cancer* **35**, 35-42 (1985).

- 12 Laichalk, L. L. & Thorley-Lawson, D. A. Terminal differentiation into plasma cells initiates the replicative cycle of Epstein-Barr virus in vivo. *J Virol* **79**, 1296-1307 (2005).
- 13 Grogan, E. *et al.* Transfection of a rearranged viral DNA fragment, WZhet, stably converts latent Epstein-Barr viral infection to productive infection in lymphoid cells. *Proc Natl Acad Sci U S A* **84**, 1332-1336 (1987).
- 14 Chevallier-Greco, A. *et al.* Both Epstein-Barr virus (EBV)-encoded trans-acting factors, EB1 and EB2, are required to activate transcription from an EBV early promoter. *EMBO J* **5**, 3243-3249 (1986).
- 15 Feederle, R. *et al.* The Epstein-Barr virus lytic program is controlled by the co-operative functions of two transactivators. *EMBO J* **19**, 3080-3089 (2000).
- 16 Farrell, P. J., Rowe, D. T., Rooney, C. M. & Kouzarides, T. Epstein-Barr virus BZLF1 trans-activator specifically binds to a consensus AP-1 site and is related to c-fos. *EMBO J* **8**, 127-132 (1989).
- 17 Gruffat, H. & Sergeant, A. Characterization of the DNA-binding site repertoire for the Epstein-Barr virus transcription factor R. *Nucleic Acids Res* **22**, 1172-1178 (1994).
- 18 Heilmann, A. M., Calderwood, M. A., Portal, D., Lu, Y. & Johannsen, E. Genome-wide analysis of Epstein-Barr virus Rta DNA binding. *J Virol* **86**, 5151-5164 (2012).
- 19 Darr, C. D., Mauser, A. & Kenney, S. Epstein-Barr virus immediate-early protein BRLF1 induces the lytic form of viral replication through a mechanism involving phosphatidylinositol-3 kinase activation. *J Virol* **75**, 6135-6142 (2001).
- 20 El-Guindy, A., Heston, L. & Miller, G. A subset of replication proteins enhances origin recognition and lytic replication by the Epstein-Barr virus ZEBRA protein. *PLoS Pathog* **6**, e1001054 (2010).
- 21 Fixman, E. D., Hayward, G. S. & Hayward, S. D. Replication of Epstein-Barr virus oriLyt: lack of a dedicated virally encoded origin-binding protein and dependence on Zta in cotransfection assays. *J Virol* **69**, 2998-3006 (1995).
- 22 Aubry, V. *et al.* Epstein-Barr virus late gene transcription depends on the assembly of a virus-specific preinitiation complex. *J Virol* **88**, 12825-12838 (2014).
- 23 Yuan, J., Cahir-McFarland, E., Zhao, B. & Kieff, E. Virus and cell RNAs expressed during Epstein-Barr virus replication. *J Virol* **80**, 2548-2565 (2006).
- 24 Lu, C. C. *et al.* Genome-wide transcription program and expression of the Rta responsive gene of Epstein-Barr virus. *Virology* **345**, 358-372 (2006).
- 25 Zhang, G. *et al.* Antisense transcription in the human cytomegalovirus transcriptome. *J Virol* **81**, 11267-11281 (2007).

- 26 Gatherer, D. *et al.* High-resolution human cytomegalovirus transcriptome. *Proc Natl Acad Sci U S A* **108**, 19755-19760 (2011).
- 27 Stern-Ginossar, N. *et al.* Decoding human cytomegalovirus. *Science* **338**, 1088-1093 (2012).
- 28 Johnson, L. S., Willert, E. K. & Virgin, H. W. Redefining the genetics of murine gammaherpesvirus 68 via transcriptome-based annotation. *Cell Host Microbe* **7**, 516-526 (2010).
- 29 Cheng, B. Y. *et al.* Tiled microarray identification of novel viral transcript structures and distinct transcriptional profiles during two modes of productive murine gammaherpesvirus 68 infection. *J Virol* **86**, 4340-4357 (2012).
- 30 Canny, S. P. *et al.* Pervasive transcription of a herpesvirus genome generates functionally important RNAs. *MBio* **5**, e01033-01013 (2014).
- 31 Chandriani, S., Xu, Y. & Ganem, D. The lytic transcriptome of Kaposi's sarcoma-associated herpesvirus reveals extensive transcription of noncoding regions, including regions antisense to important genes. *J Virol* **84**, 7934-7942 (2010).
- 32 Dresang, L. R. *et al.* Coupled transcriptome and proteome analysis of human lymphotropic tumor viruses: insights on the detection and discovery of viral genes. *BMC Genomics* **12**, 625 (2011).
- 33 Arias, C. *et al.* KSHV 2.0: a comprehensive annotation of the Kaposi's sarcoma-associated herpesvirus genome using next-generation sequencing reveals novel genomic and functional features. *PLoS Pathog* **10**, e1003847 (2014).
- 34 Concha, M. *et al.* Identification of new viral genes and transcript isoforms during Epstein-Barr virus reactivation using RNA-Seq. *J Virol* **86**, 1458-1467 (2012).
- 35 Farrell, P. J. Epstein-Barr virus genome. in *Epstein-Barr Virus* (ed E. S. Robertson) 263-287 (Caister Academic Press, 2005).
- 36 Pritchett, R. F., Hayward, S. D. & Kieff, E. D. DNA of Epstein-Barr virus. I. Comparative studies of the DNA of Epstein-Barr virus from HR-1 and B95-8 cells: size, structure, and relatedness. *J Virol* **15**, 556-559 (1975).
- 37 Lindahl, T. *et al.* Covalently closed circular duplex DNA of Epstein-Barr virus in a human lymphoid cell line. *J Mol Biol* **102**, 511-530 (1976).
- 38 Kintner, C. R. & Sugden, B. The structure of the termini of the DNA of Epstein-Barr virus. *Cell* **17**, 661-671 (1979).
- 39 Laux, G., Perricaudet, M. & Farrell, P. J. A spliced Epstein-Barr virus gene expressed in immortalized lymphocytes is created by circularization of the linear viral genome. *EMBO J* **7**, 769-774 (1988).

- 40 Baer, R. et al. DNA sequence and expression of the B95-8 Epstein-Barr virus genome. *Nature* **310**, 207-211 (1984).
- 41 de Jesus, O. *et al.* Updated Epstein-Barr virus (EBV) DNA sequence and analysis of a promoter for the BART (CST, BARF0) RNAs of EBV. *J Gen Virol* **84**, 1443-1450 (2003).
- 42 Parker, B. D., Bankier, A., Satchwell, S., Barrell, B. & Farrell, P. J. Sequence and transcription of Raji Epstein-Barr virus DNA spanning the B95-8 deletion region. *Virology* **179**, 339-346 (1990).
- 43 Lin, Z. *et al.* Whole-genome sequencing of the Akata and Mutu Epstein-Barr virus strains. *J Virol* **87**, 1172-1182 (2013).
- 44 Tsai, M. H. *et al.* Spontaneous lytic replication and epitheliotropism define an Epstein-Barr virus strain found in carcinomas. *Cell Rep* **5**, 458-470 (2013).
- 45 Dolan, A., Addison, C., Gatherer, D., Davison, A. J. & McGeoch, D. J. The genome of Epstein-Barr virus type 2 strain AG876. *Virology* **350**, 164-170 (2006).
- 46 Kwok, H. *et al.* Genomic sequencing and comparative analysis of Epstein-Barr virus genome isolated from primary nasopharyngeal carcinoma biopsy. *PLoS One* **7**, e36939 (2012).
- 47 Kwok, H. *et al.* Genomic diversity of Epstein-Barr virus genomes isolated from primary nasopharyngeal carcinoma biopsy samples. *J Virol* **88**, 10662-10672 (2014).
- 48 Lei, H. *et al.* Identification and characterization of EBV genomes in spontaneously immortalized human peripheral blood B lymphocytes by NGS technology. *BMC Genomics* **14**, 804 (2013).
- 49 Liu, P. *et al.* Direct sequencing and characterization of a clinical isolate of Epstein-Barr virus from nasopharyngeal carcinoma tissue by using next-generation sequencing technology. *J Virol* **85**, 11291-11299 (2011).
- 50 Santpere, G. *et al.* Genome-wide analysis of wild-type Epstein-Barr virus genomes derived from healthy individuals of the 1,000 Genomes Project. *Genome Biol Evol* **6**, 846-860 (2014).
- 51 Tso, K. K. *et al.* Complete genomic sequence of Epstein-Barr virus in nasopharyngeal carcinoma cell line C666-1. *Infect Agent Cancer* **8**, 29 (2013).
- 52 Zeng, M. S. *et al.* Genomic sequence analysis of Epstein-Barr virus strain GD1 from a nasopharyngeal carcinoma patient. *J Virol* **79**, 15323-15330 (2005).
- 53 Palser, A. L. *et al.* Genome diversity of Epstein-Barr virus from multiple tumor types and normal infection. *J Virol* **89**, 5222-5237 (2015).



- 54 Dambaugh, T., Hennessy, K., Chamnankit, L. & Kieff, E. U2 region of Epstein-Barr virus DNA may encode Epstein-Barr nuclear antigen 2. *Proc Natl Acad Sci U S A* **81**, 7632-7636 (1984).
- 55 Rowe, M. *et al.* Distinction between Epstein-Barr virus type A (EBNA 2A) and type B (EBNA 2B) isolates extends to the EBNA 3 family of nuclear proteins. *J Virol* **63**, 1031-1039 (1989).
- 56 Sample, J. *et al.* Epstein-Barr virus types 1 and 2 differ in their EBNA-3A, EBNA-3B, and EBNA-3C genes. *J Virol* **64**, 4084-4092 (1990).
- 57 Takada, K. *et al.* An Epstein-Barr virus-producer line Akata: establishment of the cell line and analysis of viral DNA. *Virus Genes* **5**, 147-156 (1991).
- 58 Takada, K. & Ono, Y. Synchronous and sequential activation of latently infected Epstein-Barr virus genomes. *J Virol* **63**, 445-449 (1989).
- 59 Takada, K. Cross-linking of cell surface immunoglobulins induces Epstein-Barr virus in Burkitt lymphoma lines. *Int J Cancer* **33**, 27-32 (1984).
- 60 Gregory, C. D., Rowe, M. & Rickinson, A. B. Different Epstein-Barr virus-B cell interactions in phenotypically distinct clones of a Burkitt's lymphoma cell line. *J Gen Virol* **71** ( Pt 7), 1481-1495, doi:10.1099/0022-1317-71-7-1481 (1990).
- 61 Pizzo, P. A., Magrath, I. T., Chattopadhyay, S. K., Biggar, R. J. & Gerber, P. A new tumour-derived transforming strain of Epstein-Barr virus. *Nature* **272**, 629-631 (1978).
- 62 Miller, G., Shope, T., Lisco, H., Stitt, D. & Lipman, M. Epstein-Barr virus: transformation, cytopathic changes, and viral antigens in squirrel monkey and marmoset leukocytes. *Proc Natl Acad Sci U S A* **69**, 383-387 (1972).
- 63 Blacklow, N. R., Watson, B. K., Miller, G. & Jacobson, B. M. Mononucleosis with heterophil antibodies and EB virus infection. Acquisition by an elderly patient in hospital. *Am J Med* **51**, 549-552 (1971).
- 64 Bernasconi, M. *et al.* Quantitative profiling of housekeeping and Epstein-Barr virus gene transcription in Burkitt lymphoma cell lines using an oligonucleotide microarray. *Virol J* **3**, 43 (2006).
- 65 Gradoville, L., Kwa, D., El-Guindy, A. & Miller, G. Protein kinase C-independent activation of the Epstein-Barr virus lytic cycle. *J Virol* **76**, 5612-5626 (2002).
- 66 Skare, J., Edson, C., Farley, J. & Strominger, J. L. The B95-8 isolate of Epstein-Barr virus arose from an isolate with a standard genome. *J Virol* **44**, 1088-1091 (1982).
- 67 Frisan, T., Levitsky, V. & Masucci, M. Generation of lymphoblastoid cell lines (LCLs) in *Epstein-Barr Virus Protocols* 125-127 (Springer, 2001).

- 68 International HapMap, C. *et al.* Integrating common and rare genetic variation in diverse human populations. *Nature* **467**, 52-58 (2010).
- 69 Consortium, E. P. *et al.* Identification and analysis of functional elements in 1% of the human genome by the ENCODE pilot project. *Nature* **447**, 799-816 (2007).
- 70 Siva, N. 1000 Genomes project. *Nature biotechnology* **26**, 256-256 (2008).
- 71 Arvey, A. *et al.* An atlas of the Epstein-Barr virus transcriptome and epigenome reveals host-virus regulatory interactions. *Cell Host Microbe* **12**, 233-245 (2012).
- 72 Wilson, G. & Miller, G. Recovery of Epstein-Barr virus from nonproducer neonatal human lymphoid cell transformants. *Virology* **95**, 351-358 (1979).
- 73 O'Grady, T. *et al.* Global bidirectional transcription of the Epstein-Barr virus genome during reactivation. *J Virol* **88**, 1604-1616 (2014).
- 74 Pacific Biosciences of California, I. *PacBio*. <http://www.pacb.com/>.
- 75 Kurosawa, J., Nishiyori, H. & Hayashizaki, Y. Deep cap analysis of gene expression. *Methods Mol Biol* **687**, 147-163 (2011).
- 76 Rosa, M. D., Gottlieb, E., Lerner, M. R. & Steitz, J. A. Striking similarities are exhibited by two small Epstein-Barr virus-encoded ribonucleic acids and the adenovirus-associated ribonucleic acids VAI and VAII. *Mol Cell Biol* **1**, 785-796 (1981).
- 77 Yang, L., Duff, M. O., Graveley, B. R., Carmichael, G. G. & Chen, L. L. Genomewide characterization of non-polyadenylated RNAs. *Genome Biol* **12**, R16 (2011).
- 78 Cabili, M. N. *et al.* Integrative annotation of human large intergenic noncoding RNAs reveals global properties and specific subclasses. *Genes Dev* **25**, 1915-1927 (2011).
- 79 Derrien, T. *et al.* The GENCODE v7 catalog of human long noncoding RNAs: analysis of their gene structure, evolution, and expression. *Genome Res* **22**, 1775-1789 (2012).
- 80 Harrow, J. *et al.* GENCODE: the reference human genome annotation for The ENCODE Project. *Genome Res* **22**, 1760-1774 (2012).
- 81 Zetterberg, H., Stenglein, M., Jansson, A., Ricksten, A. & Rymo, L. Relative levels of EBNA1 gene transcripts from the C/W, F and Q promoters in Epstein-Barr virus-transformed lymphoid cells in latent and lytic stages of infection. *J Gen Virol* **80**, 457-466 (1999).

- 82 Rowe, M., Lear, A. L., Croom-Carter, D., Davies, A. H. & Rickinson, A. B. Three pathways of Epstein-Barr virus gene activation from EBNA1-positive latency in B lymphocytes. *J Virol* **66**, 122-131 (1992).
- 83 Cohen, J. I. & Kieff, E. An Epstein-Barr virus nuclear protein 2 domain essential for transformation is a direct transcriptional activator. *J Virol* **65**, 5880-5885 (1991).
- 84 Chen, A., Zhao, B., Kieff, E., Aster, J. C. & Wang, F. EBNA-3B- and EBNA-3C-regulated cellular genes in Epstein-Barr virus-immortalized lymphoblastoid cell lines. *J Virol* **80**, 10139-10150 (2006).
- 85 Cludts, I. & Farrell, P. J. Multiple functions within the Epstein-Barr virus EBNA-3A protein. *J Virol* **72**, 1862-1869 (1998).
- 86 Cohen, J. I., Wang, F., Mannick, J. & Kieff, E. Epstein-Barr virus nuclear protein 2 is a key determinant of lymphocyte transformation. *Proc Natl Acad Sci U S A* **86**, 9558-9562 (1989).
- 87 Tomkinson, B., Robertson, E. & Kieff, E. Epstein-Barr virus nuclear proteins EBNA-3A and EBNA-3C are essential for B-lymphocyte growth transformation. *J Virol* **67**, 2014-2025 (1993).
- 88 Kong, L. *et al.* CPC: assess the protein-coding potential of transcripts using sequence features and support vector machine. *Nucleic Acids Res* **35**, W345-349 (2007).
- 89 Trapnell, C. *et al.* Differential gene and transcript expression analysis of RNA-seq experiments with TopHat and Cufflinks. *Nat Protoc* **7**, 562-578 (2012).
- 90 Haas, B. J. *et al.* De novo transcript sequence reconstruction from RNA-seq using the Trinity platform for reference generation and analysis. *Nat Protoc* **8**, 1494-1512 (2013).
- 91 Pertea, M. *et al.* StringTie enables improved reconstruction of a transcriptome from RNA-seq reads. *Nat Biotechnol* **33**, 290-295 (2015).
- 92 Tang, D. T. *et al.* Suppression of artifacts and barcode bias in high-throughput transcriptome analyses utilizing template switching. *Nucleic Acids Res* **41**, e44 (2013).
- 93 Shiraki, T. *et al.* Cap analysis gene expression for high-throughput analysis of transcriptional starting point and identification of promoter usage. *Proc Natl Acad Sci U S A* **100**, 15776-15781 (2003).
- 94 Carninci, P. *et al.* High-efficiency full-length cDNA cloning by biotinylated CAP trapper. *Genomics* **37**, 327-336 (1996).

- 95 Chenchik, A. *et al.* Generation and use of high-quality cDNA from small amounts of total RNA by SMART PCR. in *Gene cloning and analysis by RT-PCR Biotechniques Molecular Laboratory Methods Series* (eds Paul D. Siebert & James W. Larrick) (Biotechniques Books, 1998).
- 96 Frith, M. C. *et al.* A code for transcription initiation in mammalian genomes. *Genome Res* **18**, 1-12 (2008).
- 97 Farrell, P. J. Epstein-Barr virus. The B95-8 strain map. *Methods Mol Biol* **174**, 3-12 (2001).
- 98 Cao, S. *et al.* High-throughput RNA sequencing-based virome analysis of 50 lymphoma cell lines from the Cancer Cell Line Encyclopedia project. *J Virol* **89**, 713-729 (2015).
- 99 Edwards, R. H., Marquitz, A. R. & Raab-Traub, N. Epstein-Barr virus BART microRNAs are produced from a large intron prior to splicing. *J Virol* **82**, 9094-9106 (2008).
- 100 Sadler, R. H. & Raab-Traub, N. Structural analyses of the Epstein-Barr virus BamHI A transcripts. *J Virol* **69**, 1132-1141 (1995).
- 101 Smith, P. R. *et al.* Structure and coding content of CST (BART) family RNAs of Epstein-Barr virus. *J Virol* **74**, 3082-3092 (2000).
- 102 Tilgner, H., Grubert, F., Sharon, D. & Snyder, M. P. Defining a personal, allele-specific, and single-molecule long-read transcriptome. *Proc Natl Acad Sci U S A* **111**, 9869-9874 (2014).
- 103 Wang, L. *et al.* CPAT: Coding-Potential Assessment Tool using an alignment-free logistic regression model. *Nucleic Acids Res* **41**, e74 (2013).
- 104 Daikoku, T. *et al.* Architecture of replication compartments formed during Epstein-Barr virus lytic replication. *J Virol* **79**, 3409-3418 (2005).
- 105 Sugimoto, A. *et al.* Different distributions of Epstein-Barr virus early and late gene transcripts within viral replication compartments. *J Virol* **87**, 6693-6699 (2013).
- 106 Manet, E. *et al.* Epstein-Barr virus bicistronic mRNAs generated by facultative splicing code for two transcriptional trans-activators. *EMBO J* **8**, 1819-1826 (1989).
- 107 Wang, F., Petti, L., Braun, D., Seung, S. & Kieff, E. A bicistronic Epstein-Barr virus mRNA encodes two nuclear proteins in latently infected, growth-transformed lymphocytes. *J Virol* **61**, 945-954 (1987).
- 108 Bodescot, M., Perricaudet, M. & Farrell, P. J. A promoter for the highly spliced EBNA family of RNAs of Epstein-Barr virus. *J Virol* **61**, 3424-3430 (1987).

- 109 Sample, J., Hummel, M., Braun, D., Birkenbach, M. & Kieff, E. Nucleotide sequences of mRNAs encoding Epstein-Barr virus nuclear proteins: a probable transcriptional initiation site. *Proc Natl Acad Sci U S A* **83**, 5096-5100 (1986).
- 110 Austin, P. J., Flemington, E., Yandava, C. N., Strominger, J. L. & Speck, S. H. Complex transcription of the Epstein-Barr virus BamHI fragment H rightward open reading frame 1 (BHRF1) in latently and lytically infected B lymphocytes. *Proc Natl Acad Sci U S A* **85**, 3678-3682 (1988).
- 111 Bodescot, M. & Perricaudet, M. Epstein-Barr virus mRNAs produced by alternative splicing. *Nucleic Acids Res* **14**, 7103-7114 (1986).
- 112 Pearson, G. R. *et al.* Identification of an Epstein-Barr virus early gene encoding a second component of the restricted early antigen complex. *Virology* **160**, 151-161 (1987).
- 113 Free Software Foundation. *GNU General Public License*, <<http://www.gnu.org/licenses/gpl-3.0.en.html>> (2007).
- 114 Wu, T. D. & Watanabe, C. K. GMAP: a genomic mapping and alignment program for mRNA and EST sequences. *Bioinformatics* **21**, 1859-1875 (2005).
- 115 Dobin, A. *et al.* STAR: ultrafast universal RNA-seq aligner. *Bioinformatics* **29**, 15-21 (2013).
- 116 Murata, M. *et al.* Detecting expressed genes using CAGE. *Methods Mol Biol* **1164**, 67-85 (2014).
- 117 Lin, Z. *et al.* Detection of murine leukemia virus in the Epstein-Barr virus-positive human B-cell line JY, using a computational RNA-Seq-based exogenous agent detection pipeline, PARSES. *J Virol* **86**, 2970-2977 (2012).
- 118 Langmead, B. & Salzberg, S. L. Fast gapped-read alignment with Bowtie 2. *Nat Methods* **9**, 357-359 (2012).
- 119 Robinson, J. T. *et al.* Integrative genomics viewer. *Nat Biotechnol* **29**, 24-26 (2011).
- 120 Thorvaldsdottir, H., Robinson, J. T. & Mesirov, J. P. Integrative Genomics Viewer (IGV): high-performance genomics data visualization and exploration. *Brief Bioinform* **14**, 178-192 (2013).
- 121 Quinlan, A. R. & Hall, I. M. BEDTools: a flexible suite of utilities for comparing genomic features. *Bioinformatics* **26**, 841-842 (2010).
- 122 Xu, G. *et al.* SAMMate: a GUI tool for processing short read alignments in SAM/BAM format. *Source Code Biol Med* **6**, 2, (2011).

- 123 Li, B. & Dewey, C. N. RSEM: accurate transcript quantification from RNA-Seq data with or without a reference genome. *BMC Bioinformatics* **12**, 323 (2011).
- 124 Pruitt, K. D. *et al.* RefSeq: an update on mammalian reference sequences. *Nucleic Acids Res* **42**, D756-763 (2014).
- 125 Karolchik, D. *et al.* The UCSC Table Browser data retrieval tool. *Nucleic Acids Res* **32**, D493-496 (2004).
- 126 Kim, D. *et al.* TopHat2: accurate alignment of transcriptomes in the presence of insertions, deletions and gene fusions. *Genome Biol* **14**, R36 (2013).
- 127 Zhang, Z., Schwartz, S., Wagner, L. & Miller, W. A greedy algorithm for aligning DNA sequences. *J Comput Biol* **7**, 203-214 (2000).
- 128 Feng, L. *et al.* Technique for strand-specific gene-expression analysis and monitoring of primer-independent cDNA synthesis in reverse transcription. *Biotechniques* **52**, 263-270 (2012).
- 129 Hughes, T. A. Regulation of gene expression by alternative untranslated regions. *Trends Genet* **22**, 119-122 (2006).
- 130 Nagasaki, H., Arita, M., Nishizawa, T., Suwa, M. & Gotoh, O. Species-specific variation of alternative splicing and transcriptional initiation in six eukaryotes. *Gene* **364**, 53-62 (2005).
- 131 Carninci, P. *et al.* The transcriptional landscape of the mammalian genome. *Science* **309**, 1559-1563 (2005).
- 132 Velten, L. *et al.* Single-cell polyadenylation site mapping reveals 3' isoform choice variability. *Mol Syst Biol* **11**, 812 (2015).
- 133 Erickson, K. D. & Martin, J. M. The late lytic LMP-1 protein of Epstein-Barr virus can negatively regulate LMP-1 signaling. *J Virol* **74**, 1057-1060 (2000).
- 134 Pandya, J. & Walling, D. M. Oncogenic activity of Epstein-Barr virus latent membrane protein 1 (LMP-1) is down-regulated by lytic LMP-1. *J Virol* **80**, 8038-8046 (2006).
- 135 Van Damme, P., Gawron, D., Van Crielinge, W. & Menschaert, G. N-terminal proteomics and ribosome profiling provide a comprehensive view of the alternative translation initiation landscape in mice and men. *Mol Cell Proteomics* **13**, 1245-1261 (2014).
- 136 Sasaki, A. *et al.* The N-terminal truncated isoform of SOCS3 translated from an alternative initiation AUG codon under stress conditions is stable due to the lack of a major ubiquitination site, Lys-6. *J Biol Chem* **278**, 2432-2436 (2003).

- 137 Shalak, V., Kaminska, M. & Mirande, M. Translation initiation from two in-frame AUGs generates mitochondrial and cytoplasmic forms of the p43 component of the multisynthetase complex. *Biochemistry* **48**, 9959-9968 (2009).
- 138 Furnari, F. B., Zacny, V., Quinlivan, E. B., Kenney, S. & Pagano, J. S. RAZ, an Epstein-Barr virus transdominant repressor that modulates the viral reactivation mechanism. *J Virol* **68**, 1827-1836 (1994).
- 139 Frith, M. C. *et al.* The abundance of short proteins in the mammalian proteome. *PLoS Genet* **2**, e52 (2006).
- 140 Jaber, T. & Yuan, Y. A virally encoded small peptide regulates RTA stability and facilitates Kaposi's sarcoma-associated herpesvirus lytic replication. *J Virol* **87**, 3461-3470 (2013).
- 141 Andrews, S. J. & Rothnagel, J. A. Emerging evidence for functional peptides encoded by short open reading frames. *Nat Rev Genet* **15**, 193-204 (2014).
- 142 Geballe, A. P. & Mocarski, E. S. Translational control of cytomegalovirus gene expression is mediated by upstream AUG codons. *J Virol* **62**, 3334-3340 (1988).
- 143 Calvo, S. E., Pagliarini, D. J. & Mootha, V. K. Upstream open reading frames cause widespread reduction of protein expression and are polymorphic among humans. *Proc Natl Acad Sci U S A* **106**, 7507-7512 (2009).
- 144 Wittmann, J., Hol, E. M. & Jack, H. M. hUPF2 silencing identifies physiologic substrates of mammalian nonsense-mediated mRNA decay. *Mol Cell Biol* **26**, 1272-1287 (2006).
- 145 Spriggs, K. A., Bushell, M. & Willis, A. E. Translational regulation of gene expression during conditions of cell stress. *Mol Cell* **40**, 228-237 (2010).
- 146 Kapranov, P. *et al.* RNA maps reveal new RNA classes and a possible function for pervasive transcription. *Science* **316**, 1484-1488 (2007).
- 147 Cao, S. *et al.* New Noncoding Lytic Transcripts Derived from the Epstein-Barr Virus Latency Origin of Replication, oriP, Are Hyperedited, Bind the Paraspeckle Protein, NONO/p54nrb, and Support Viral Lytic Transcription. *J Virol* **89**, 7120-7132 (2015).
- 148 Schaefer, B. C., Strominger, J. L. & Speck, S. H. The Epstein-Barr virus BamHI F promoter is an early lytic promoter: lack of correlation with EBNA 1 gene transcription in group 1 Burkitt's lymphoma cell lines. *J Virol* **69**, 5039-5047 (1995).
- 149 Nonkwelo, C., Skinner, J., Bell, A., Rickinson, A. & Sample, J. Transcription start sites downstream of the Epstein-Barr virus (EBV) Fp promoter in early-passage Burkitt lymphoma cells define a fourth promoter for expression of the EBV EBNA-1 protein. *J Virol* **70**, 623-627 (1996).

- 150 Hudson, G. S., Farrell, P. J. & Barrell, B. G. Two related but differentially expressed potential membrane proteins encoded by the EcoRI Dhet region of Epstein-Barr virus B95-8. *J Virol* **53**, 528-535 (1985).
- 151 Serio, T. R., Cahill, N., Prout, M. E. & Miller, G. A functionally distinct TATA box required for late progression through the Epstein-Barr virus life cycle. *J Virol* **72**, 8338-8343 (1998).
- 152 Rutkowski, A. J. *et al.* Widespread disruption of host transcription termination in HSV-1 infection. *Nat Commun* **6**, 7126 (2015).
- 153 Vilborg, A., Passarelli, M. C., Yario, T. A., Tycowski, K. T. & Steitz, J. A. Widespread Inducible Transcription Downstream of Human Genes. *Mol Cell* **59**, 449-461 (2015).
- 154 Cheshenko, N. *et al.* Herpes simplex virus triggers activation of calcium-signaling pathways. *J Cell Biol* **163**, 283-293 (2003).
- 155 Faggioni, A. *et al.* Calcium modulation activates Epstein-Barr virus genome in latently infected cells. *Science* **232**, 1554-1556 (1986).
- 156 Chiu, Y. F., Sugden, A. U. & Sugden, B. Epstein-Barr viral productive amplification reprograms nuclear architecture, DNA replication, and histone deposition. *Cell Host Microbe* **14**, 607-618 (2013).
- 157 Jochum, S., Ruiss, R., Moosmann, A., Hammerschmidt, W. & Zeidler, R. RNAs in Epstein-Barr virions control early steps of infection. *Proc Natl Acad Sci U S A* **109**, E1396-1404 (2012).
- 158 Kalla, M., Schmeinck, A., Bergbauer, M., Pich, D. & Hammerschmidt, W. AP-1 homolog BZLF1 of Epstein-Barr virus has two essential functions dependent on the epigenetic state of the viral genome. *Proc Natl Acad Sci U S A* **107**, 850-855 (2010).
- 159 Wen, W. *et al.* Epstein-Barr virus BZLF1 gene, a switch from latency to lytic infection, is expressed as an immediate-early gene after primary infection of B lymphocytes. *J Virol* **81**, 1037-1042 (2007).
- 160 Pegtel, D. M. *et al.* Functional delivery of viral miRNAs via exosomes. *Proc Natl Acad Sci U S A* **107**, 6328-6333 (2010).



## **APPENDIX 1**

TRIMD\_start\_validator.pl

```
#!/usr/bin/perl

#Accepts a SAM file of Iso-Seq fl data, a SAM file of CAGE data, and a
#bed file of annotated polyadenylated transcripts. Counts the number of
#non-clipped Iso-Seq reads with 5' starts at each genomic position and
#estimates consensus locations of clusters of 5' starts. Uses Paraclu to
#identify clusters of 5' starts in the CAGE data. Output includes a
#bedgraph file of Iso-Seq 5' starts, a bed file of the weighted centers
#of Iso-Seq start clusters, a bedgraph file of CAGE tag 5' starts, a bed
#file of the weighted centers of Paraclu-identified CAGE 5' start
#clusters, and a bed file of Iso-seq 5' starts supported by the CAGE
#data, with their annotation status noted.

#USAGE:
# perl <PATH/TRIMD_start_validator.pl> </PATH/Iso-Seq_sam_file>
# </PATH/CAGE_file> </PATH/Annotation_bed_file>

use warnings;
use strict;

die "USAGE: 'perl <PATH/TRIMD_start_validator.pl> </PATH/Iso-Seq_sam_file> </PATH/CAGE_file> </PATH/Annotation_bed_file>'" unless @ARGV == 3;

my ($SMRT_file, $CAGE_file, $ann_file) = @ARGV;

print "Enter name of viral chromosome (e.g. chrEBV_Akata_inverted): ";
my $viral_chr = <STDIN>;
chomp $viral_chr;

my $distance_between_SMRT_peaks;
my $min_tags;
my $min_dens;
my $min_length;
my $max_length;
my $dist_SMRT_CAGE;
my $min_SMRT;
my $ann_dist;

print "Use default parameters [y/n]? ";
my $answer = <STDIN>;
chomp $answer;

if ($answer eq "y") {
    $distance_between_SMRT_peaks = 8;
    $min_tags = 15;
    $min_dens = 2;
    $min_length = 1;
}
```

```

    $max_length = 20;
    $dist_SMRT_CAGE = 3;
    $min_SMRT = 1;
    $ann_dist = 10;
}
else {
    print "Enter desired window for collapsing Iso-Seq 5' starts (e.g.
8): ";
    $distance_between_SMRT_peaks = <STDIN>;
    chomp $distance_between_SMRT_peaks;

    print "Enter minimum tags per CAGE cluster (e.g. 15): ";
    $min_tags = <STDIN>;
    chomp $min_tags;

    print "Enter minimum relative density for CAGE clusters (e.g. 2):
";
    $min_dens = <STDIN>;
    chomp $min_dens;

    print "Enter minimum CAGE cluster length (e.g. 1): ";
    $min_length = <STDIN>;
    chomp $min_length;

    print "Enter maximum CAGE cluster length (e.g. 20): ";
    $max_length = <STDIN>;
    chomp $max_length;

    print "Enter desired maximum allowable distance between Iso-Seq and
CAGE 5' starts (e.g. 3): ";
    $dist_SMRT_CAGE = <STDIN>;
    chomp $dist_SMRT_CAGE;

    print "Enter minimum number of SMRT reads to report a 5' start
(e.g. 1): ";
    $min_SMRT = <STDIN>;
    chomp $min_SMRT;

    print "Enter maximum distance in bp from an annotated start to be
called as 'annotated' (e.g. 10): ";
    $ann_dist = <STDIN>;
    chomp $ann_dist;
}

print "-----\n";

#####-----SMRT FILE PROCESSING-----#####
system("awk '$3==\"$viral_chr\"' \Q$SMRT_file\E \|| sort -k 4,4n >
\Q$SMRT_file\E.sorted.temp");
system("awk '$2==0' \Q$SMRT_file\E.sorted.temp >
\Q$SMRT_file\E.sorted.plus.sam.temp");
system("awk '$2==16' \Q$SMRT_file\E.sorted.temp >
\Q$SMRT_file\E.sorted.minus.sam.temp");
system("rm \Q$SMRT_file\E.sorted.temp");

```

```

#processing of PLUS SMRT sam file
open(INF, "<$SMRT_file.sorted.plus.sam.temp") or die "couldn't open
file";
open(OUT, ">$SMRT_file.sorted.plus.sam.read_starts.bedgraph") or die
"couldn't open file";

my $previous_coordinate=1;
my $count=0;
my $previous_chr = "start";
print "Processing Iso-Seq plus strand reads...\n";

while (my $line = <INF>) {
    chomp($line);
    my @cols = split("\t", $line);
    next if ($cols[5] =~ m/^\d+S/); #skips reads clipped at the 5' end
    my @split_id = split("\/", $cols[0]); #extracts the read depth for
this putative isoform from its id
    if (($cols[2] eq $previous_chr) and ($cols[3] ==
$previous_coordinate)) {
        $count = $count + $split_id[1]; #increases the count by the
read depth for the putative isoform
    }
    else {
        if ($previous_chr eq "start") { #doesn't print out the
placeholder first line.
            $previous_chr = $cols[2];      #sets the previous
chromosome, previous coordinate and count values
            $previous_coordinate = $cols[3];
            $count = $split_id[1];
        }
        else {
            print OUT $previous_chr, "\t", $previous_coordinate-1,
"\t", $previous_coordinate, "\t", $count, "\n"; #prints to output file,
converting to chrStart 0-based bedgraph coordinate
            $previous_chr = $cols[2];
            $previous_coordinate = $cols[3];
            $count = $split_id[1];
        }
    }
}

print OUT $previous_chr, "\t", $previous_coordinate-1, "\t",
$previous_coordinate, "\t", $count, "\n"; #prints the last start
coordinates to output file
close(INF);
close(OUT);

system("rm \Q$SMRT_file\E.sorted.plus.sam.temp");

#processing of MINUS SMRT sam file
open(INF, "<$SMRT_file.sorted.minus.sam.temp") or die "couldn't open
file";
open(OUT, ">$SMRT_file.sorted.minus.sam.read_starts.bedgraph.temp") or
die "couldn't open file";

```

```

my @CIGAR_dist;
my $sum;
my %minus_starts;
print "Processing Iso-Seq minus strand reads...\n";

while (my $line = <INF>) {
    chomp($line);
    my @cols = split("\t", $line);
    next if ($cols[5] =~ m/\d+S$/); #skips reads soft-clipped at the 5'
end
    while ($cols[5] =~ /(\d+)[DMNX=]/g) { #these lines use the CIGAR
string to determine the downstream coordinate
        push (@CIGAR_dist, $1);
    }
    $sum += $_ for @CIGAR_dist;
    my $start_coord = $cols[3] + $sum - 1; #subtract 1 to account for
start/end inclusion
    my $chr_start_coord = "$cols[2]\:$start_coord"; #combines the
chromosome and 5' end coordinate into a key to use for the hash
    $sum = 0;
    @CIGAR_dist = ();
    my @split_id = split("\/", $cols[0]); #extracts the read depth for
this putative isoform from its id
    if (exists $minus_starts{$chr_start_coord}) { #if the key is
already in the hash, increases the value (count) by the read depth for
that putative isoform
        $minus_starts{$chr_start_coord} =
$minus_starts{$chr_start_coord} + $split_id[1];
    }
    else {
        $minus_starts{$chr_start_coord} = $split_id[1]; #if the key is
not already in the hash, adds it with a value (count) of the read depth
for that putative isoform
    }
}

foreach my $chr_start_coord (sort keys %minus_starts) { #prints out a(n
inadequately) sorted temporary bedgraph file
    my @split_keys = split(":", $chr_start_coord);
    print OUT $split_keys[0], "\t", $split_keys[1]-1, "\t",
$split_keys[1], "\t-", $minus_starts{$chr_start_coord}, "\n"; #prints
to output file, converting chrStart to 0-based bedgraph coordinates
}
close(INF);
close(OUT);

system("sort -k 1,1 -k 2,2n
\Q$SMRT_file\E.sorted.minus.sam.read_starts.bedgraph.temp >
\Q$SMRT_file\E.sorted.minus.sam.read_starts.bedgraph");

system("cat \Q$SMRT_file\E.sorted.plus.sam.read_starts.bedgraph
\Q$SMRT_file\E.sorted.minus.sam.read_starts.bedgraph.temp | sort -k2,3n
> \Q$SMRT_file\E.\Q$viral_chr\E.read_starts.bedgraph.noheader");

system("rm \Q$SMRT_file\E.sorted.minus.sam.read_starts.bedgraph.temp");

```

```

system("rm \Q$SMRT_file\E.sorted.minus.sam.read_starts.bedgraph");
system("rm \Q$SMRT_file\E.sorted.minus.sam.temp");
system("rm \Q$SMRT_file\E.sorted.plus.sam.read_starts.bedgraph");

#add header to bedgraph file
open(INF, "<$SMRT_file.$viral_chr.read_starts.bedgraph.noheader") or die "couldn't open file";
open(OUT, ">$SMRT_file.$viral_chr.read_starts.bedgraph") or die "couldn't open file";

print OUT "track type=bedgraph
name=\"\Q$SMRT_file.$viral_chr.read_starts.bedgraph\" description=\"5'
starts of SMRT reads from start_finder_sam_to_bed.pl\"\n";
while (my $line = <INF>) {
    print OUT $line;
}
close(OUT);
close(INF);

system("rm
\Q$SMRT_file\E.\Q$viral_chr\E.read_starts.bedgraph.noheader");

#make a bed file from the SMRT bedgraph file:
open(INF, "<$SMRT_file.$viral_chr.read_starts.bedgraph") or die "couldn't open file";
open(OUT, ">$SMRT_file.starts.temp.bed") or die "couldn't open file";

print "Combining Iso-Seq 5' starts within $distance_between_SMRT_peaks
of each other and calculating consensus 5' starts...\n";
collapse_bedgraph($distance_between_SMRT_peaks);

close(INF);
close(OUT);

system("sort -k 1,1 -k 2,2n \Q$SMRT_file\E.starts.temp.bed >
\Q$SMRT_file\E.starts.bed.noheader");
system("rm \Q$SMRT_file.starts.temp.bed\E");

#add header to bed file
open(INF, "<$SMRT_file.starts.bed.noheader") or die "couldn't open file";
open(OUT, ">$SMRT_file.$viral_chr.SMRT_starts.bed") or die "couldn't open file";

print OUT "track type=bed
name=\"\Q$SMRT_file.$viral_chr.SMRT_starts.bed\" description=\"consensus
5' starts of Iso-Seq reads within $distance_between_SMRT_peaks bp
collapsed to weighted center from start_finder_sam_to_bed.pl\"\n";
while (my $line = <INF>) {
    print OUT $line;
}
close(OUT);
close(INF);

system("rm \Q$SMRT_file\E.starts.bed.noheader");

```

```
#####-----PROCESSING CAGE DATA-----#####

print "Preparing CAGE file...\n";

system("awk '\$3==\"$viral_chr\"' \Q$CAGE_file\E \|| sort -k 4,4n >
\Q$CAGE_file\E.sorted.temp");
system("awk '\$2==0 \|| \2==81 \|| \2==83 \|| \2==89 \||
\2==137 \|| \2==161 \|| \2==163' \Q$CAGE_file\E.sorted.temp >
\Q$CAGE_file\E.sorted.plus.sam.temp");
system("awk '\$2==16 \|| \2==73 \|| \2==97 \|| \2==99 \||
\2==145 \|| \2==147 \|| \2==153' \Q$CAGE_file\E.sorted.temp >
\Q$CAGE_file\E.sorted.minus.sam.temp");

#processing of plus CAGE sam file

open(INF, "<$CAGE_file.sorted.plus.sam.temp") or die "couldn't open
file";
open(OUT, ">$CAGE_file.read_starts.txt") or die "couldn't open file";
open(OUT2, ">$CAGE_file.starts.bedgraph.temp");

my $prev_coord=0.5;
my $start_count=0;

while (my $line = <INF>) {
    chomp($line);
    next if ($line =~ m/^@/); #skips header lines
    my @cols = split("\t", $line);
    if ($cols[3] == $prev_coord) {
        $start_count++; #increases the count by 1
    }

    else {
        if ($prev_coord == 0.5) { #doesn't print out the placeholder
first line.
            $prev_coord = $cols[3];
            $start_count = 1;
        }

        else {
            print OUT $viral_chr, "\t+\t", $prev_coord, "\t",
$start_count, "\n"; #prints to output txt file for Paraclu
            print OUT2 $viral_chr, "\t", $prev_coord-1, "\t",
$prev_coord, "\t", $start_count, "\n"; #prints to output bedgraph file
            $prev_coord = $cols[3];
            $start_count = 1;
        }
    }
}

print OUT "$viral_chr\t+\t$prev_coord\t$start_count\n"; #prints the
last start coordinates to output file
close(INF);

system("rm \Q$CAGE_file\E.sorted.plus.sam.temp");
```

```

#processing of MINUS CAGE sam file
open(INF, "<$CAGE_file.sorted.minus.sam.temp") or die "couldn't open
file";

my @read_dist;
my $dist_sum;
my %minus_start;

while (my $line = <INF>) {
    chomp($line);
    my @cols = split("\t", $line);
    while ($cols[5] =~ /\(\d+\)[DMNX=]/g) { #these lines use the CIGAR
string to determine the downstream coordinate
        push (@read_dist, $1);
    }
    $dist_sum += $_ for @read_dist;
    my $start_coord = $cols[3] + $dist_sum - 1; #subtract one to
account for start/end inclusion
    $dist_sum = 0;
    @read_dist = ();
    if (exists $minus_start{$start_coord}) { #if the key is already in
the hash, increases the value (count) by the read depth for that
putative isoform
        $minus_start{$start_coord} = $minus_start{$start_coord} + 1;
    }
    else {
        $minus_start{$start_coord} = 1; #if the key is not already in
the hash, adds it with a value (count) of the read depth for that
putative isoform
    }
}
foreach my $start_coord (sort keys %minus_start) { #prints out a(n
inadequately) sorted file. Doesn't need to be sorted because Paraclu
will do that anyways.
    print OUT "$viral_chr\t-
\t$start_coord\t$minus_start{$start_coord}\n";
    print OUT2 $viral_chr, "\t", $start_coord-1, "\t", $start_coord,
"\t-", $minus_start{$start_coord}, "\n"; #prints to output bedgraph
file
}
close(INF);
close(OUT);
close(OUT2);

system("rm \Q$CAGE_file\E.sorted.minus.sam.temp");
system("rm \Q$CAGE_file\E.sorted.temp");
system("sort -k2,3n \Q$CAGE_file\E.starts.bedgraph.temp >
\Q$CAGE_file\E.starts.bedgraph.noheader");
system("rm \Q$CAGE_file\E.starts.bedgraph.temp");

#add header to bedgraph file
open(INF, "<$CAGE_file.starts.bedgraph.noheader") or die "couldn't open
file";
open(OUT, ">$CAGE_file.$viral_chr.starts.bedgraph") or die "couldn't
open file";

```



```

print OUT "track type=bedgraph
name=\"\$CAGE_file.\$viral_chr.starts.bedgraph\" description=\"5' starts
of CAGE tags from start_finder_sam_to_bed.pl\"\\n\";
while (my $line = <INF>) {
    print OUT $line;
}
close(INF);
close(OUT);

system("rm \\Q\$CAGE_file\\E.starts.bedgraph.noheader");

#Running Paraclu to define clusters

open(INF, "<\$CAGE_file.read_starts.txt") or die "couldn't open file";
open(OUT, ">\$CAGE_file.paraclu.txt.temp");

# paraclu.pl: perform parametric clustering of data attached to
sequences

# Written by Martin C Frith 2006
# Genome Exploration Research Group, RIKEN GSC and
# Institute for Molecular Bioscience, University of Queensland

# This program reads in a list of numeric values attached to positions
# in sequences. The list should have four tab- (or space-) separated
# columns containing: the sequence name, the strand, the position, and
# the value. (Multiple values for the same sequence/strand/position
# will be summed.) It outputs the clusters as eight tab-separated
# columns: sequence name, strand, start, end, number of values, sum of
# values, min d, max d. See below for the meaning of "d".

# An example line of input:
# chr1      +      17689      3
# Clustering is performed separately for different strands (as if each
# strand were a completely different sequence). It does not matter
# whether the position uses 0-based or 1-based coordinates: the
# program does not care, and the output will be consistent with the
# input.

# The clusters are defined as follows. A cluster is a maximal scoring
# segment, where the score of any segment is: the sum of the values in
# the segment minus d times the size of the segment. Large values of d
# give smaller, tighter clusters and small values of d give larger,
# looser clusters. The program finds all possible clusters for any
# value of d, and annotates each cluster with the maximum and minimum
# values of d that produce it. The ratio max d / min d provides a
# measure of the cluster's "stability".

# The output will include two types of obvious/trivial/degenerate
# clusters: those that cover single positions, and those that cover
# all of the positions in a sequence. For many purposes, it would be
# best to ignore these cases.

use strict;

```

```

use List::Util qw(min max);

my %data;

#warn "reading...\n";

while (<INF>) {
    chomp;
    s/#.*//; # ignore comments
    next unless /\S/; # skip blank lines

    my ($seq, $strand, $pos, $value) = split;
    my $key = "$seq $strand";
    push @{$data{$key}}, [ $pos, $value ];
}

warn "Clustering CAGE data...\n";

print OUT "# sequence, strand, start, end, sites, sum of values, min d,
max d\n";

for my $key (sort keys %data) { # iterate over sequences / strands
    my ($seq, $strand) = split " ", $key;
    my $sites = $data{$key};

    @$sites = sort { $$a[0] <=> $$b[0] } @$sites; # sort by position

    my $clusters = all_clusters($sites);

    for my $c (@$clusters) {
        my ($beg, $end, $tot, $sit, $min, $max) = @$c;
        my $beg_pos = $$sites[$beg][0];
        my $end_pos = $$sites[$end][0];
        printf OUT
"$seq\t$strand\t$beg_pos\t$end_pos\t$sit\t$tot\t%.3g\t%.3g\n",
            $min, $max;
    }
}

### Generic code to find clusters in a sparse sequence of values: ###

sub all_clusters {
    our $inf = 1e100; # hopefully much bigger than any value in the
input
    our $sites = shift; # input: reference to array of site locations
& values
    our $clusters = []; # output: reference to array of clusters
    get_clusters(0, $$sites, -$inf);
    return $clusters;
}

# get clusters of sites between beg and end with density > min_density
sub get_clusters {
    our ($clusters, $inf);
    my ($beg, $end, $min_density) = @_;

```

```

my ($prefix, $pmin, $ptot, $psit) = weakest_prefix($beg, $end);
my ($suffix, $smin, $stot, $ssit) = weakest_suffix($beg, $end);
$ptot == $stot and $psit == $ssit or die "internal error!";
my $max_density = min $pmin, $smin;

unless ($max_density == $inf) {
    my $break = $pmin < $smin ? $prefix + 1 : $suffix;
    my $new_min = max $min_density, $max_density;
    get_clusters($beg, $break-1, $new_min);
    get_clusters($break, $end, $new_min);
}

push @$clusters, [ $beg, $end, $ptot, $psit, $min_density,
$max_density ]
    if $max_density > $min_density;
}

# get least dense prefix (and total of values & sites)
sub weakest_prefix {
    our ($sites, $inf);
    my ($beg, $end) = @_;

    my $beg_pos = $$sites[$beg][0];
    my $min_density = $inf;
    my $min_prefix = $end;
    my $tot = 0;
    my $sit = 0;

    for (my $i = $beg; $i < $end; ++$i) {
        $tot += $$sites[$i][1];
        next if $$sites[$i][0] == $$sites[$i+1][0]; # idiot-proofing
        ++$sit;
        my $dist = $$sites[$i+1][0] - $beg_pos;
        my $density = $tot / $dist;
        if ($density < $min_density) {
            $min_prefix = $i;
            $min_density = $density;
        }
    }

    $tot += $$sites[$end][1];
    ++$sit;
    return ($min_prefix, $min_density, $tot, $sit);
}

# get least dense suffix (and total of values & sites)
sub weakest_suffix {
    our ($sites, $inf);
    my ($beg, $end) = @_;

    my $end_pos = $$sites[$end][0];
    my $min_density = $inf;
    my $min_suffix = $beg;
    my $tot = 0;

```

```

my $sit = 0;

for (my $i = $end; $i > $beg; --$i) {
    $tot += $$sites[$i][1];
    next if $$sites[$i][0] == $$sites[$i-1][0]; # idiot-proofing
    ++$sit;
    my $dist = $end_pos - $$sites[$i-1][0];
    my $density = $tot / $dist;
    if ($density < $min_density) {
        $min_suffix = $i;
        $min_density = $density;
    }
}

$tot += $$sites[$beg][1];
++$sit;
return ($min_suffix, $min_density, $tot, $sit);
}

close(INF);
close(OUT);

system("sort -k2,2 -k3,3n -k4,4rn \Q$CAGE_file\E.paraclu.txt.temp >
\Q$CAGE_file\E.paraclu.txt");
system("rm \Q$CAGE_file\E.paraclu.txt.temp");

#filtering clusters:
print "Extracting CAGE clusters containing $min_tags tags, density fold
change at least $min_dens, from $min_length to $max_length bp long\n";

my $length;
my $dens;
my $prev_start = 0;
my $prev_end = 0;

open(INF, "<$CAGE_file.paraclu.txt") or die "couldn't open file";
open(OUT,
">$CAGE_file.clusters.$min_tags.$min_dens.$min_length.$max_length.bed")
or die "couldn't open file";

while (my $line = <INF>) { #extracts clusters meeting the criteria.
    Excludes subclusters.
    chomp($line);
    next if ($line =~ /^#/); #skips the header line
    my @cols = split("\t", $line);
    next if ($cols[5] < $min_tags);
    $length = $cols[3] - $cols[2] + 1;
    if (($length >= $min_length) and ($length <= $max_length)) {
        $dens = $cols[7] / $cols[6];
        if ($dens >= $min_dens) {
            next if (($cols[2] >= $prev_start) and ($cols[2] <=
$prev_end));
            if ($dens < 100) { #keep everything one-based (like sam)
for now; will convert to zero-based in next step
                printf OUT "%s\t%d\t%d\t%d%s%.1f\t%d\t%s\n", $cols[0],

```

```

$cols[2], $cols[3], $cols[5], ":", $dens, $cols[5], $cols[1]; #limits
the density output to 1 decimal place, but doesn't change huge numbers
to exponents
    }
    else {
        printf OUT "%s\t%d\t%d\t%d%s%.1e\t%d\t%s\n", $cols[0],
$cols[2], $cols[3], $cols[5], ":", $dens, $cols[5], $cols[1]; #changes
large numbers to exponents
    }
    $prev_start = $cols[2];
    $prev_end = $cols[3];
}
}
}
close(INF);
close(OUT);

system("rm \Q$CAGE_file\E.paraclu.txt");

#getting weighted averages of Paraclu clusters:

my $rangeStart_CAGE;
my $rangeEnd_CAGE;
my $strand_CAGE;
my $CAGE_weighted_sum = 0;
my $CAGE_weighted_average;

open(INF,
"<$CAGE_file.clusters.$min_tags.$min_dens.$min_length.$max_length.bed")
or die "couldn't open file";
open(OUT, ">$CAGE_file.$viral_chr.CAGE_starts.temp") or die "couldn't
open file";

while (my $line = <INF>) {
    chomp($line);
    my @cols = split("\t", $line);
    $rangeStart_CAGE = $cols[1];
    $rangeEnd_CAGE = $cols[2];
    $strand_CAGE = $cols[5];
    open(INF2, "<$CAGE_file.read_starts.txt") or die "couldn't open
file";
    while (my $line2 = <INF2>) {
        chomp($line2);
        my @cols2 = split("\t", $line2);
        if ((($cols2[2]) >= $rangeStart_CAGE) and (($cols2[2]) <=
$rangeEnd_CAGE) and ($cols2[1] eq $strand_CAGE)) {
            $CAGE_weighted_sum = $CAGE_weighted_sum +
($cols2[2]*$cols2[3]);
        }
    }
    $CAGE_weighted_average = sprintf("%.0f",
($CAGE_weighted_sum/$cols[4]));
    if ($strand_CAGE eq "+") {
        print OUT $cols[0], "\t", $CAGE_weighted_average-1, "\t",
$CAGE_weighted_average, "\t", $rangeStart_CAGE-1, ":", $rangeEnd_CAGE-

```

```

1, ":", $cols[3], "\t", $cols[4], "\t", $strand_CAGE, "\n"; #prints
output, converting chrStart and range to 0-based
}
    elsif ($strand_CAGE eq "-") {
        print OUT $cols[0], "\t", $CAGE_weighted_average-1, "\t",
        $CAGE_weighted_average, "\t", $rangeStart_CAGE, ":", $rangeEnd_CAGE,
        ":", $cols[3], "\t", $cols[4], "\t", $strand_CAGE, "\n"; #prints
output, converting chrStart to 0-based but keeping range 1-based
(because these are all chrEnds)
    }
    $CAGE_weighted_sum = 0;
    close(INF2);
}

close(INF);
close(OUT);

system("sort -k2,2n -k 3,3n
\Q$CAGE_file\E.\Q$viral_chr\E.CAGE_starts.temp >
\Q$CAGE_file\E.\Q$viral_chr\E.CAGE_starts.noheader");
system("rm \Q$CAGE_file\E.\Q$viral_chr\E.CAGE_starts.temp");

#add header to bed file

open(INF, "<$CAGE_file.$viral_chr.CAGE_starts.noheader") or die
"couldn't open file";
open(OUT, ">$CAGE_file.$viral_chr.CAGE_starts.bed");

print OUT "track type=bed
name=\"\$CAGE_file.$viral_chr.CAGE_starts.bed\" description=\"weighted
averages of CAGE clusters between $min_length and $max_length bases
long with at least $min_tags tags and relative density of at least
$min_dens from start_finder_sam_to_bed.pl and paraclu\""\n";
while (my $line = <INF>) {
    print OUT $line;
}
close(OUT);
close(INF);

system("rm \Q$CAGE_file\E.\Q$viral_chr\E.CAGE_starts.noheader");
system("rm \Q$CAGE_file\E.read_starts.txt");
system("rm
\Q$CAGE_file\E.clusters.$min_tags.$min_dens.$min_length.$max_length.bed
");

#####-----SEEKING CAGE SUPPORT FOR SMRT STARTS-----#####

open(INF, "<$CAGE_file.$viral_chr.CAGE_starts.bed" ) or die "couldn't
open file";

print "Extracting Iso-Seq 5' starts within $dist_SMRT_CAGE bases of
CAGE clusters...\n";

my %features_CAGE;
my $key_combo_CAGE;

```

```

while(my $line = <INF> ) {
    chomp($line);
    next if ($line =~ /^track/); #skips the track definition line
    my @cols = split("\t", $line);
    if ($cols[5] eq "+") { #for each line in the CAGE bed file, creates
a key for the hash combining coordinate and strand. Selects chrStart
for starts on the plus strand and chrEnd for starts on the minus
strand.
        $key_combo_CAGE = "$cols[1]:$cols[5]";
    }
    if ($cols[5] eq "-") {
        $key_combo_CAGE = "$cols[2]:$cols[5]";
    }
    $features_CAGE{$key_combo_CAGE} = $cols[4]; #enters a count value
for the key into the hash
}

close(INF);

open(INF, "<$SMRT_file.$viral_chr.SMRT_starts.bed" ) or die "couldn't
open file";
open(OUT,
">$SMRT_file.$viral_chr.SMRT_starts.bed.CAGE_support.bed.temp");

my $match_count;
my $lower_limit;
my $upper_limit;

while(my $line = <INF>) {
    chomp($line);
    next if ($line =~ /^track/); #skips the track definition line
    my @SMRT_cols = split("\t", $line);
    next if (abs $SMRT_cols[4] < $min_SMRT); #skips starts without
enough SMRT support
    foreach my $key_combo_CAGE (keys %features_CAGE) {
        my @CAGE_cols = split(":", $key_combo_CAGE);
        if ($SMRT_cols[5] eq "+") {
            $lower_limit = $SMRT_cols[1]-$dist_SMRT_CAGE;
            $upper_limit = $SMRT_cols[1]+$dist_SMRT_CAGE;
        }
        if ($SMRT_cols[5] eq "-") {
            $lower_limit = $SMRT_cols[2]-$dist_SMRT_CAGE;
            $upper_limit = $SMRT_cols[2]+$dist_SMRT_CAGE;
        }
        if (($SMRT_cols[5] eq $CAGE_cols[1]) and ($CAGE_cols[0] >=
$lower_limit) and ($CAGE_cols[0] <= $upper_limit)) {
            if ($match_count) { #if more than one CAGE start matches
the SMRT start, selects the CAGE end with the most tags
                if ($features_CAGE{$key_combo_CAGE} > $match_count) {
                    $match_count = $features_CAGE{$key_combo_CAGE};
                }
            }
        }
        else {
            $match_count = $features_CAGE{$key_combo_CAGE};
        }
    }
}

```

```

    }
  }
}
if ($match_count) {
  my $name = "$SMRT_cols[4].IsoSeq_$match_count.CAGE";
  my $count = $match_count + $SMRT_cols[4];
  print OUT
"$SMRT_cols[0]\t$SMRT_cols[1]\t$SMRT_cols[2]\t$name\t$count\t$SMRT_cols
[5]\t$SMRT_cols[3]\n";
  undef($match_count);
}
else {
  my @range_cols = split (":", $SMRT_cols[3]);
  print OUT
"$SMRT_cols[0]\t$SMRT_cols[1]\t$SMRT_cols[2]\t$range_cols[2].IsoSeq\t$range_cols[2]\t$SMRT_cols[5]\t$SMRT_cols[3]\n";
}
}

close(OUT);
close(INF);

#####-----COMPARING TO ANNOTATED STARTS-----#####
open(INF, "<$ann_file" ) or die "couldn't open file";

print "Processing annotation file...\n";

#extract 5' starts from the annotation file:
#annotation file must be sorted by chrStart then chrEnd!
my @annotated_starts;
my $plus_prev_coord = 0;
my $minus_prev_coord = 0;

while(my $line = <INF>) {
  chomp($line);
  next if ($line =~ /^track/); #skips the track definition line
  my @ann_cols = split("\t", $line);
  next if $ann_cols[0] ne $viral_chr; #skip lines that aren't viral
  if ($ann_cols[5] eq "+") {
    if ($ann_cols[1] != $plus_prev_coord) {
      push (@annotated_starts, "$ann_cols[1]:$ann_cols[5]");
      $plus_prev_coord = $ann_cols[1];
    }
  }
  elsif ($ann_cols[5] eq "-"){
    if ($ann_cols[2] != $minus_prev_coord) {
      push (@annotated_starts, "$ann_cols[2]:$ann_cols[5]");
      $minus_prev_coord = $ann_cols[2];
    }
  }
}

my $annotated = scalar @annotated_starts;

```



```

close(INF);

#compare starts in the altered SMRT starts file (that already has info
about CAGE starts) with annotated starts

open(INF,
"<$SMRT_file.$viral_chr.SMRT_starts.bed.CAGE_support.bed.temp" ) or die
"couldn't open file";
open(OUT, ">$SMRT_file.$viral_chr.validated_starts.bed");

print "Comparing Iso-seq starts to annotated starts...\n";

print OUT "track type=bedDetail
name=\"\$SMRT_file.$viral_chr.validated_starts.bed\"
description=\"consensus Iso-Seq 5' starts of collapse value 8 supported
by at least $min_SMRT read(s) within $dist_SMRT_CAGE bp of CAGE
clusters or within $ann_dist bp of annotated starts. From
start_finder_sam_to_bed.pl\"\"";

my $annotated_found_by_SMRT = 0;
my $novel_found_by_SMRT_CAGE = 0;
my $SMRT_annotated = 0; #this is different than
$annotated_found_by_SMRT because depending on input parameters two SMRT
starts may correspond to a single annotated start or vice versa.

while(my $line = <INF>) {
    chomp($line);
    my @SMRT_cols = split("\t", $line);
    my $found_flag=0;
    foreach my $ann_start (@annotated_starts) {
        my @ann_cols = split(":", $ann_start);
        my $lower_limit = $ann_cols[0]-$ann_dist;
        my $upper_limit = $ann_cols[0]+$ann_dist;
        if ($SMRT_cols[5] eq "+") {
            if (($SMRT_cols[5] eq $ann_cols[1]) and
($SMRT_cols[1]>=$lower_limit) and ($SMRT_cols[1]<=$upper_limit)) {
                if ($found_flag == 0) {
                    print OUT
"$SMRT_cols[0]\t$SMRT_cols[1]\t$SMRT_cols[2]\tann_$SMRT_cols[5]_SMRT_c
ols[3]\t$SMRT_cols[4]\t$SMRT_cols[5]\t$SMRT_cols[6]\n";
                    $found_flag = 1;
                    $annotated_found_by_SMRT++; #counts multiple
annotated starts near SMRT starts
                    $SMRT_annotated++; #only counts one annotated start
per SMRT start
                }
            }
            elsif ($found_flag == 1) {
                $annotated_found_by_SMRT++;
            }
        }
    }
    if ($SMRT_cols[5] eq "-") {
        if (($SMRT_cols[5] eq $ann_cols[1]) and
($SMRT_cols[2]>=$lower_limit) and ($SMRT_cols[2]<=$upper_limit)) {
            if ($found_flag == 0) {

```

```

        print OUT $SMRT_cols[0], "\t", $SMRT_cols[1], "\t",
$SMRT_cols[2], "\tann_", $SMRT_cols[5], "_", $SMRT_cols[3], "\t",
abs($SMRT_cols[4]), "\t", $SMRT_cols[5], "\t", $SMRT_cols[6], "\n";
        $found_flag = 1;
        $annotated_found_by_SMRT++;
        $SMRT_annotated++;
    }
    elsif ($found_flag == 1) {
        $annotated_found_by_SMRT++;
    }
}

}

}
if ($found_flag == 0) {
    if ($SMRT_cols[3] =~ /.+IsoSeq_.+CAGE/) {
        print OUT
"$SMRT_cols[0]\t$SMRT_cols[1]\t$SMRT_cols[2]\tnov_$SMRT_cols[5]_SMRT_c
ols[3]\t$SMRT_cols[4]\t$SMRT_cols[5]\t$SMRT_cols[6]\n";
        $novel_found_by_SMRT_CAGE++;
    }
}
}

my $total_found = $SMRT_annotated + $novel_found_by_SMRT_CAGE;

close(INF);
close(OUT);

print "-----\n";

open(OUT, ">${viral_chr}_validated_starts_stats.txt");

if ($total_found > 0) {
    if ($SMRT_annotated != $annotated_found_by_SMRT) {
        print "$total_found 5' starts found. $novel_found_by_SMRT_CAGE
are novel, $SMRT_annotated are annotated. $annotated_found_by_SMRT out
of $annotated total annotated 5' starts are found.\nNote that two
annotated starts may be within $ann_dist bp of a single Iso-Seq start
or vice versa.\n";
        print OUT "$viral_chr\n$total_found 5'
starts\n\t$novel_found_by_SMRT_CAGE novel\n\t$SMRT_annotated
annotated\n$nannotated starts in annotation
file\n\t$annotated_found_by_SMRT detected by Iso-Seq\n\ninput
files:\n\t$SMRT_file\n\t$CAGE_file\n\t$ann_file\n";
    }
    else {
        print "$total_found 5' starts found. $novel_found_by_SMRT_CAGE
are novel, $SMRT_annotated are annotated (out of a total of $annotated
annotated 5' starts).\n";
        print OUT "$viral_chr\n\n$total_found 5'
starts\n\t$novel_found_by_SMRT_CAGE novel\n\t$SMRT_annotated
annotated\n$nannotated starts in annotation file\n\ninput
files:\n\t$SMRT_file\n\t$CAGE_file\n\t$ann_file\n";
    }
}

```

```

}
else {
    print "No validated starts found.\n";
    print OUT "No validated starts found.\n\ninput
files:\n\t$SMRT_file\n\t$CAGE_file\n\t$ann_file\n";
}

close(OUT);

system("rm
\Q$SMRT_file\E.\Q$viral_chr\E.SMRT_starts.bed.CAGE_support.bed.temp");

#####
sub collapse_bedgraph {
    my ($distance_between_peaks) = shift;
    my $prev_coord_plus = 1;
    my $prev_coord_minus = 1;
    my $count_sum_plus = 0;
    my $count_sum_minus = 0;
    my $weighted_coordinate_sum_plus = 0;
    my $weighted_coordinate_sum_minus = 0;
    my $weighted_average_plus;
    my $weighted_average_minus;
    my $first_plus = 1;
    my $first_minus = 1;
    my @coords_plus;
    my @coords_minus;
    my $chrStart_plus;
    my $chrEnd_plus;
    my $chrStart_minus;
    my $chrEnd_minus;

    while (my $line = <INF>) {
        chomp($line);
        next if ($line =~ /^track/); #skips the track definition line
        my @cols = split("\t", $line);
        if ($cols[3] > 0) { #if this coordinate has a positive count...
            if ($cols[1] <= $prev_coord_plus +
($distance_between_peaks)) { #if the coordinate is within the specified
number of bp of the previous coordinate
                $count_sum_plus = $count_sum_plus + $cols[3]; #adds to
the sums to eventually calculate the weighted average
                $weighted_coordinate_sum_plus =
$weighted_coordinate_sum_plus + ($cols[1]*$cols[3]);
                push (@coords_plus, $cols[1]);
                $prev_coord_plus = $cols[1]; #sets the current
coordinate as the "previous coordinate" before moving on
            }
            else { #if the present coordinate is not within the
specified number of bp of the previous coordinate, need to print out a
feature
                if ($first_plus == 1) { #"first" flag avoids wonkiness
if the first coordinate is far from coordinate 1 (don't need to print
out a feature yet)
                    $count_sum_plus = $cols[3];

```

```

        $weighted_coordinate_sum_plus = $cols[1]*$cols[3];
        $prev_coord_plus = $cols[1];
        push (@coords_plus, $cols[1]);
        $first_plus = 0;
    }
    else {
        $weighted_average_plus = sprintf("%.0f",
($weighted_coordinate_sum_plus/$count_sum_plus)); #calculates weighted
average
        $chrStart_plus = $coords_plus[0];
        $chrEnd_plus = pop(@coords_plus);
        print OUT $viral_chr, "\t", $weighted_average_plus,
"\t", $weighted_average_plus+1, "\t", $chrStart_plus, ":",
$chrEnd_plus, ":", $count_sum_plus, "\t", $count_sum_plus, "\t+\n";
#prints out weighted average for plus strand features. Use printf to
round the weighted average.
        @coords_plus = ($cols[1]);
        $count_sum_plus = $cols[3]; #sets "previous
coordinate", count and sum of counts for the current coordinate
        $weighted_coordinate_sum_plus = $cols[1]*$cols[3];
        $prev_coord_plus = $cols[1];
    }
}
}
elseif ($cols[3] < 0) { #if this coordinate has a negative
count...
    if ($cols[2] <= $prev_coord_minus +
($distance_between_peaks)) { #if the coordinate is within the specified
number of bp of the previous coordinate
        $count_sum_minus = $count_sum_minus + $cols[3]; #adds
to the sums to eventually calculate the weighted average
        $weighted_coordinate_sum_minus =
$weighted_coordinate_sum_minus + ($cols[2]*$cols[3]);
        push (@coords_minus, $cols[2]);
        $prev_coord_minus = $cols[2]; #sets the current
coordinate as the "previous coordinate" before moving on
    }
    else { #if the present coordinate is not within the
specified number of bp of the previous coordinate, need to print out a
feature
        if ($first_minus == 1) { #"first" flag avoids wonkiness
if the first coordinate is far from coordinate 1 (don't need to print
out a feature yet)
            $count_sum_minus = $cols[3];
            $weighted_coordinate_sum_minus = $cols[2]*$cols[3];
            $prev_coord_minus = $cols[2];
            push (@coords_minus, $cols[2]);
            $first_minus = 0;
        }
        else {
            $weighted_average_minus = sprintf("%.0f",
($weighted_coordinate_sum_minus/$count_sum_minus)); #calculates
weighted average.
            $chrStart_minus = $coords_minus[0];
            $chrEnd_minus = pop(@coords_minus);

```

```

        print OUT $viral_chr, "\t",
$weighted_average_minus-1, "\t", $weighted_average_minus, "\t",
$chrStart_minus, ":", $chrEnd_minus, ":", $count_sum_minus, "\t",
abs($count_sum_minus), "\t-\n";
        @coords_minus = ($cols[2]);
        @coords_minus = ($cols[2]);
        $count_sum_minus = $cols[3]; #sets "previous
coordinate", count and sum of counts for the current coordinate
        $weighted_coordinate_sum_minus = $cols[2]*$cols[3];
        $prev_coord_minus = $cols[2];
    }
}
}

    if ($count_sum_plus > 0) {#calculates and prints out weighted
average for the last feature (plus strand)
        $weighted_average_plus = sprintf("%.0f",
($weighted_coordinate_sum_plus/$count_sum_plus));
        $chrStart_plus = $coords_plus[0];
        $chrEnd_plus = pop(@coords_plus);
        print OUT $viral_chr, "\t", $weighted_average_plus, "\t",
$weighted_average_plus+1, "\t", $chrStart_plus, ":", $chrEnd_plus,
":", $count_sum_plus, "\t", $count_sum_plus, "\t+\n"; #prints out
weighted average for plus strand features. Use printf to round the
weighted average.
    }

    if ($count_sum_minus < 0) {#calculates and prints out weighted
average for the last feature (minus strand)
        $weighted_average_minus = sprintf("%.0f",
($weighted_coordinate_sum_minus/$count_sum_minus));
        $chrStart_minus = $coords_minus[0];
        $chrEnd_minus = pop(@coords_minus);
        print OUT $viral_chr, "\t", $weighted_average_minus-1, "\t",
$weighted_average_minus, "\t", $chrStart_minus, ":", $chrEnd_minus,
":", $count_sum_minus, "\t", abs($count_sum_minus), "\t-\n";
    }
}

```

## **APPENDIX 2**

TRIMD\_junction\_validator.pl

```
#!/usr/bin/perl

#Accepts a junctions files from GMAP/Iso-Seq (generated with the -f
introns argument), an SJ.out.tab files from STAR/Illumina and an
annotation file. Returns 3 bed files: one of SMRT introns, one of
Illumina introns and one of introns detected by both methods.
Annotation status of validated introns is noted.

#USAGE:
# perl <PATH/TRIMD_junction_validator.pl> </PATH/Iso-Seq_introns_file>
</PATH/Illumina_SJ.out.tab_file> </PATH/transcript_annotation_bed_file>
<coordinates_to_ignore_bed_file(optional)>

use warnings;
use strict;

my ($SMRT_jfile, $ill_jfile, $ann_file, $ig_file) = @ARGV;

print "Enter name of viral chromosome (e.g. chrEBV_Akata_inverted): ";
my $viral_chr = <STDIN>;
chomp $viral_chr;

my $min_SMRTj;
my $min_illj;

print "Use default parameters [y/n]? ";
my $answer = <STDIN>;
chomp $answer;

if ($answer eq "y") {
    $min_SMRTj = 1;
    $min_illj = 1;
}
else {
    print "Enter minimum Iso-Seq read depth to report a splice junction
(e.g. 1): ";
    $min_SMRTj = <STDIN>;
    chomp $min_SMRTj;

    print "Enter minimum Illumina read depth to report a splice
junction (e.g. 1): ";
    $min_illj = <STDIN>;
    chomp $min_illj;
}

print "-----\n";

####-----GMAP/SMRT FILE CONVERSION-----#####
```

```

open(INF, "<$SMRT_jfile");
open(OUT, ">$SMRT_jfile.temp");

print "Processing Iso-Seq splice junctions...\n";

while(my $line = <INF> ) {
    chomp($line);
    my ($id) = $line =~ /\>(.)\./i;
    my ($chr) = $line =~ /\s(.+):/;
    my ($score) = $line =~ /\>.+\/(\d+)\//;
    my ($donor, $acceptor) = $line =~ /:(\d+)\.\.(\d+)/;
    next if $chr ne $viral_chr;
    if ($acceptor > $donor) {
        print OUT $chr, "\t", $donor, "\t", $acceptor - 1, "\t", $id,
"\t", $score, "\t+\n";
    }
    else {
        print OUT $chr, "\t", $acceptor, "\t", $donor - 1, "\t", $id,
"\t", $score, "\t-\n";
    }
}
close(OUT);
close(INF);

system("sort -k2,3n \Q$SMRT_jfile\E.temp >
\Q$SMRT_jfile\E.sorted.temp"); #sorts so that duplicate introns will be
next to each other.

open(INF, "<$SMRT_jfile.sorted.temp" ) or die "couldn't reopen file";
open(OUT, ">$SMRT_jfile.bed.temp");

my $plus_previous_chr = "start";
my $plus_count = 0;
my $plus_previous_start = 0;
my $plus_previous_end = 0;
my $minus_previous_chr = "start";
my $minus_count = 0;
my $minus_previous_start = 0;
my $minus_previous_end = 0;

while (my $line = <INF>) {
    chomp($line);
    my @cols = split("\t", $line);
    if ($cols[5] eq "+") { #plus and minus need to be treated
separately in case of introns with the same starts and ends annotated
on opposite strands
        if (($cols[0] eq $plus_previous_chr) and ($cols[1] ==
$plus_previous_start) and ($cols[2] == $plus_previous_end)) { #checks
to see if the intron matches the previous intron
            $plus_count = $plus_count + $cols[4];
        }
        else {
            if ($plus_previous_chr eq "start") { #prevents the initial
placeholder value from printing out as a line, and sets the values of
the first intron

```



```

        $plus_previous_chr = $cols[0];
        $plus_previous_start = $cols[1];
        $plus_previous_end = $cols[2];
        $plus_count = $cols[4];
    }
    else {
        print OUT
"$plus_previous_chr\t$plus_previous_start\t$plus_previous_end\t$plus_co
unt\t$plus_count\t+\n";
        $plus_previous_chr = $cols[0];
        $plus_previous_start = $cols[1];
        $plus_previous_end = $cols[2];
        $plus_count = $cols[4];
    }
}
}
if ($cols[5] eq "-") {
    if (($cols[0] eq $minus_previous_chr) and ($cols[1] ==
$minus_previous_start) and ($cols[2] == $minus_previous_end)) {
        $minus_count = $minus_count + $cols[4];
    }
    else {
        if ($minus_previous_chr eq "start") {
            $minus_previous_chr = $cols[0];
            $minus_previous_start = $cols[1];
            $minus_previous_end = $cols[2];
            $minus_count = $cols[4];
        }
        else {
            print OUT
"$minus_previous_chr\t$minus_previous_start\t$minus_previous_end\t$minu
s_count\t$minus_count\t-\n"; #prints out in bed format
            $minus_count = $cols[4];
            $minus_previous_chr = $cols[0];
            $minus_previous_start = $cols[1];
            $minus_previous_end = $cols[2];
        }
    }
}
}
}

print OUT
"$plus_previous_chr\t$plus_previous_start\t$plus_previous_end\t$plus_co
unt\t$plus_count\t+\n"; #adds the last plus strand feature
print OUT
"$minus_previous_chr\t$minus_previous_start\t$minus_previous_end\t$minu
s_count\t$minus_count\t-\n"; #adds the last plus strand feature
close(OUT);
close(INF);

system("sort -k2,3n \Q$SMRT_jfile\E.bed.temp >
\Q$SMRT_jfile\E.\Q$viral_chr\E.bed.sorted.temp");
system("rm \Q$SMRT_jfile\E.bed.temp");
system("rm \Q$SMRT_jfile.sorted\E.bed.temp");
system("rm \Q$SMRT_jfile\E.bed.temp");

```

```

#if an annotation file of regions to be ignored is supplied, remove the
SMRT junctions with a donor or acceptor in those regions:
if (defined $ig_file) {
    open(INF, "<$ig_file");
    print "Removing Iso-Seq junctions with donor or acceptor in ignored
region...\n";
    my @ig_coords;
    while(my $line = <INF>) {
        chomp($line);
        my @cols = split("\t", $line);
        my $ig_coord = "$cols[1]:$cols[2]";
        push (@ig_coords, $ig_coord);
    }
    close(INF);

    open(INF, "<$SMRT_jfile.$viral_chr.bed.sorted.temp") or die
"couldn't open file";
    open(OUT, ">$SMRT_jfile.$viral_chr.bed.noheader");

    while(my $line = <INF>) {
        chomp($line);
        my @cols = split( "\t", $line );
        my $found_flag=0;
        foreach my $ig_coord (@ig_coords) {
            my ($ig_start, $ig_end) = split (":", $ig_coord);
            if ((($cols[1] >= $ig_start) and ($cols[1] <= $ig_end)) ||
(($cols[2] >= $ig_start) and ($cols[2] <= $ig_end))) {
                $found_flag = 1;
                last;
            }
        }
        if ($found_flag == 0) {
            print OUT $line, "\n";
        }
    }

    close(INF);
    close(OUT);
}

#add header to bed file
if (defined $ig_file) {
    open(INF, "<$SMRT_jfile.$viral_chr.bed.noheader") or die "couldn't
open file";
}
else {
    open(INF, "<$SMRT_jfile.$viral_chr.bed.sorted.temp") or die
"couldn't open file";
}
open(OUT, ">$SMRT_jfile.$viral_chr.bed") or die "couldn't open file";

print OUT "track type=bed name=\"$SMRT_jfile.$viral_chr.bed\"
description=\"Iso-Seq introns from splice_junction_matcher.pl\"\n";
while (my $line = <INF>) {

```

```

    print OUT $line;
}
close(OUT);
close(INF);

if (defined $ig_file) {
    system("rm \Q$SMRT_jfile\E.\Q$viral_chr\E.bed.noheader");
}
system("rm \Q$SMRT_jfile\E.\Q$viral_chr\E.bed.sorted.temp");

#####-----STAR/ILLUMINA FILE CONVERSION-----#####

open(INF, "<$ill_jfile" ) or die "couldn't open file";
open(OUT, ">$ill_jfile.$viral_chr.bed");

print "Processing Illumina splice junctions...\n";
print OUT "track type=bed name=\"$ill_jfile.$viral_chr.bed\"
description=\"Illumina STAR introns from
splice_junction_matcher.pl\""\n";

while(my $line = <INF> ) {
    chomp($line);
    my @cols = split( "\t", $line );
    tr/12/+-/ foreach ($cols[3]); #change the numeric strand
    indicators to + or -
    next if $cols[0] ne $viral_chr; #skip lines that aren't viral
    my $chrStart = $cols[1] - 1; #changes the start coordinate to 0-
    based for bed
    print OUT
"$cols[0]\t$chrStart\t$cols[2]\t$cols[4]\t$cols[6]\t$cols[3]\n";
}

close(OUT);
close(INF);

#if an annotation file of regions to be ignored is supplied, remove the
Illumina junctions with a donor or acceptor in those regions:
if (defined $ig_file) {
    open(INF, "<$ig_file");
    print "Removing Illumina junctions with donor or acceptor in
ignored region...\n";
    my @ig_coords;
    while(my $line = <INF>) {
        chomp($line);
        my @cols = split("\t", $line);
        my $ig_coord = "$cols[1]:$cols[2]";
        push (@ig_coords, $ig_coord);
    }
    close(INF);

    open(INF, "<$ill_jfile.$viral_chr.bed") or die "couldn't open
file";
    open(OUT, ">$ill_jfile.$viral_chr.no_ignored.bed");

```

```

    print OUT "track type=bed
name=\"$ill_jfile.$viral_chr.no_ignored.bed\" description=\"Illumina
STAR introns from splice_junction_matcher.pl\"\\n";

    while(my $line = <INF>) {
        chomp($line);
        next if ($line =~ /^track/); #skips the track definition line
        my @cols = split( "\\t", $line );
        my $found_flag=0;
        foreach my $ig_coord (@ig_coords) {
            my ($ig_start, $ig_end) = split(":", $ig_coord);
            if ((($cols[1] >= $ig_start) and ($cols[1] <= $ig_end)) ||
                (($cols[2] >= $ig_start) and ($cols[2] <= $ig_end))) {
                $found_flag = 1;
                last;
            }
        }
        if ($found_flag == 0) {
            print OUT $line, "\\n";
        }
    }

    close(INF);
    close(OUT);
    system("rm \\Q$ill_jfile\\E.\\Q$viral_chr\\E.bed");
}

#####-----GMAP/ILLUMINA COMPARISON-----#####

if (defined $ig_file) {
    open(INF, "<$ill_jfile.$viral_chr.no_ignored.bed" ) or die
"couldn't open file";
}
else {
    open(INF, "<$ill_jfile.$viral_chr.bed" ) or die "couldn't open
file";
}

print "Checking for matching splice junctions...\\n";

my %ill_junctions;

while(my $line = <INF> ) {
    chomp($line);
    next if ($line =~ /^track/); #skips the track definition line
    my @cols = split("\\t", $line);
    next if ($cols[4] < $min_illj);
    my $ill_key_combo = "$cols[0]$cols[1]$cols[2]$cols[5]"; #for each
line in the Illumina file, creates a key for the hash combining
chromosome, start coordinate, end coordinate and strand
    $ill_junctions{$ill_key_combo} = $cols[4]; #enters a count value
for the key into the hash
}

close(INF);

```

```

open(INF, "<$SMRT_jfile.$viral_chr.bed" ) or die "couldn't open file";
open(OUT, ">$SMRT_jfile.$viral_chr.illumina_support.bed.temp");

while(my $line = <INF>) {
    chomp($line);
    next if ($line =~ /^track/); #skips the track definition line
    my @cols = split("\t", $line);
    next if ($cols[4] < $min_SMRTj);
    my $SMRT_key_combo = "$cols[0]$cols[1]$cols[2]$cols[5]"; #for each
line in the SMRT file, creates a variable/key combining chromosome,
start coordinate, end coordinate and strand
    if (exists $ill_junctions{$SMRT_key_combo}) { #checks to see if the
key exists in the Illumina hash: if so, prints it out
        my $junction_depth = $cols[4] +
$ill_junctions{$SMRT_key_combo};
        print OUT
"$cols[0]\t$cols[1]\t$cols[2]\t$cols[4].IsoSeq_$ill_junctions{$SMRT_key
_combo}.Ill\t$junction_depth\t$cols[5]\n";
    }
    else {
        print OUT
"$cols[0]\t$cols[1]\t$cols[2]\t$cols[3].IsoSeq\t$cols[4]\t$cols[5]\n";
    }
}
close(INF);
close(OUT);

#####-----ANNOTATION FILE COMPARISON-----#####

#First extract intron coordinates from the annotation file
open(INF, "<$ann_file");

print "Processing annotation file...\n";

my @intron_start;
my @intron_end;
my %ann_intron_coord_pair;
my $start;
my $end;

while (my $line = <INF>) {
    chomp($line);
    next if ($line =~ /^track/); #skips the track definition line
    my @cols = split("\t", $line);
    next if $cols[0] ne $viral_chr; #skip lines that aren't viral
    my $intron_number = $cols[9] - 1;
    next if ($intron_number == 0);
    my @block_sizes = split(",", $cols[10]);
    my @block_starts = split(",", $cols[11]);
    for (my $i = 0; $i < $intron_number; $i = $i + 1) { #for the
transcript currently in the "while" loop, creates an array of intron
start sites relative to the genome
        $start = $cols[1] + $block_sizes[$i] + $block_starts[$i];
        push(@intron_start, $start);
    }
}

```

```

    }
    for (my $i2 = 1; $i2 < $cols[9]; $i2 = $i2 + 1) { #for the
transcript currently in the "while" loop, creates an array of intron
end sites relative to the genome
        $end = $cols[1] + $block_starts[$i2];
        push(@intron_end, $end);
    }
    for (my $i3 = 0; $i3 < $intron_number; $i3 = $i3 + 1) { #for the
transcript currently in the "while" loop, matches up intron start and
end sites to create a hash of complete intron coordinates relative to
the genome
        my $intron_coords =
"$cols[0]:$intron_start[$i3]:$intron_end[$i3]:$cols[5]";
        if (exists $ann_intron_coord_pair{$intron_coords}) {
            $ann_intron_coord_pair{$intron_coords} =
$ann_intron_coord_pair{$intron_coords} + 1; #if the intron is already
in the hash (from another transcript), increase the count
        }
        else {
            $ann_intron_coord_pair{$intron_coords} = 1; #if the intron
is not already in the hash, adds it with a value of 1
        }
    }
    @intron_start = ();
    @intron_end = (); #intron starts and ends have been assigned to the
%ann_intron_pair hash; empty them for the next transcript
}

my $ann_count = 0;

if (defined $ig_file) {
    open(INF, "<$ig_file");
    my @ig_coords;
    while(my $line = <INF>) {
        chomp($line);
        my @cols = split("\t", $line);
        my $ig_coord = "$cols[1]:$cols[2]";
        push (@ig_coords, $ig_coord);
    }
    close(INF);

    foreach my $ann_intron_coord_pair (keys %ann_intron_coord_pair) {
        my ($ann_chr, $ann_start, $ann_end, $ann_strand) = split(":",
$ann_intron_coord_pair);
        my $found_flag = 0;
        foreach my $ig_coord (@ig_coords) {
            my ($ig_start, $ig_end) = split(":", $ig_coord);
            if ((($ann_start >= $ig_start) and ($ann_start <= $ig_end))
|| (($ann_end >= $ig_start) and ($ann_end <= $ig_end))) {
                $found_flag = 1;
                last;
            }
        }
        if ($found_flag == 0) {
            $ann_count++;
        }
    }
}

```

```

    }
  }
}
else {
    $ann_count = scalar (keys %ann_intron_coord_pair);
}

close(INF);

#Compare introns in the altered (with Illumina data) SMRT file to
annotated introns

open(INF, "<$SMRT_jfile.$viral_chr.illumina_support.bed.temp");
open(OUT, ">$SMRT_jfile.$viral_chr.validated_introns.bed");

print "Comparing Iso-Seq junctions to annotation file...\n";

print OUT "track type=bed
name=\"$SMRT_jfile.$viral_chr.validated_introns.bed\"
description=\"Introns detected by Iso-Seq with read depth at least
$min_SMRTj supported by Illumina-detected junctions with read depth at
least $min_illj and/or annotation. From
splice_junction_matcher.pl\"";

my $val_SMRT_count = 0;
my $ann_SMRT_count = 0;
my $nov_SMRT_count = 0;

while (my $line = <INF>) {
    chomp($line);
    my @SMRT_cols = split("\t", $line);
    my $SMRT_intron_coords =
"$SMRT_cols[0]:$SMRT_cols[1]:$SMRT_cols[2]:$SMRT_cols[5]"; #creates a
key to search the has of annotated introns
    if (exists $ann_intron_coord_pair{$SMRT_intron_coords}) { #if the
intron matches an annotated intron, notes that and prints out the line
        print OUT
"$SMRT_cols[0]\t$SMRT_cols[1]\t$SMRT_cols[2]\tann_$SMRT_cols[5]_SMRT_c
ols[3]\t$SMRT_cols[4]\t$SMRT_cols[5]\n";
        $ann_SMRT_count++;
        $val_SMRT_count++;
    }
    else {
        if ($SMRT_cols[3] =~ /.+IsoSeq_.+Ill/) { #if the intron doesn't
match an annotated intron but does have Illumina support, notes that
and prints out the line
            print OUT
"$SMRT_cols[0]\t$SMRT_cols[1]\t$SMRT_cols[2]\tnov_$SMRT_cols[5]_SMRT_c
ols[3]\t$SMRT_cols[4]\t$SMRT_cols[5]\n";
            $nov_SMRT_count++;
            $val_SMRT_count++;
        }
    }
}
close(OUT);

```

```

close(INF);

print "-----\n";

open(OUT, ">${viral_chr}_validated_introns_stats.txt");

if ($val_SMRT_count > 0) {
    print "$val_SMRT_count validated junctions detected in the Iso-Seq
file. $nov_SMRT_count are novel and $ann_SMRT_count are annotated (out
of $ann_count annotated junctions).\n";
    if (defined $ig_file) {
        print OUT "$viral_chr\n\n$val_SMRT_count validated
junctions\n\t$nov_SMRT_count novel\n\t$ann_SMRT_count
annotated\n$ann_count junctions in annotation file\n\ninput
files:\n\t$SMRT_jfile\n\t$ill_jfile\n\t$ann_file\n\t$ig_file\n";
    }
    else {
        print OUT "$viral_chr\n\n$val_SMRT_count validated
junctions\n\t$nov_SMRT_count novel\n\t$ann_SMRT_count
annotated\n$ann_count junctions in annotation file\n\ninput
files:\n\t$SMRT_jfile\n\t$ill_jfile\n\t$ann_file\n";
    }
}
else {
    print "No validated junctions found.\n";
    if (defined $ig_file) {
        print OUT "No validated junctions found.\n\ninput
files:\n\t$SMRT_jfile\n\t$ill_jfile\n\t$ann_file\n\t$ig_file\n";
    }
    else{
        print OUT "No validated junctions found.\n\ninput
files:\n\t$SMRT_jfile\n\t$ill_jfile\n\t$ann_file \n";
    }
}

close(OUT);

system ("rm \Q$SMRT_jfile\E.\Q$viral_chr\E.illumina_support.bed.temp");

```



### **APPENDIX 3**

TRIMD\_end\_validator.pl

```
#!/usr/bin/perl

#Accepts a SAM file using Iso-Seq fl data, a SAM file using Illumina
data, and a bed file of annotated polyadenylated transcripts. Counts
the number of non-clipped Iso-Seq reads with 3' ends at each genomic
position and estimates consensus locations of clusters of 3' ends.
Extracts Illumina reads containing apparent polyA tails and estimates
consensus locations of clusters of polyadenylation sites. Output
includes bedgraph files of all 3' ends, bed files of the weighted
centers of end clusters, a sam file of reads with polyA tails and a bed
file of Iso-Seq 3' ends supported by either the annotation or the
Illumina data.

#USAGE:
# perl <PATH/TRIMD_end_validator.pl> </PATH/Iso-Seq_sam_file>
</PATH/Illumina_sam_file> </PATH/Annotation_bed_file>

use warnings;
use strict;

die "USAGE: 'perl <PATH/TRIMD_end_validator.pl> </PATH/Iso-
Seq_sam_file> </PATH/Illumina_sam_file> </PATH/Annotation_bed_file>'"
unless @ARGV == 3;

my ($SMRT_file, $ill_file, $ann_file) = @ARGV;

print "Enter name of viral chromosome (e.g. chrEBV_Akata_inverted): ";
my $viral_chr = <STDIN>;
chomp $viral_chr;

my $distance_between_SMRT_peaks;
my $min_As;
my $min_softclip;
my $distance_between_ill_peaks;
my $dist_SMRT_ill_d;
my $dist_SMRT_ill_u;
my $min_SMRT;
my $min_ill;
my $ann_dist;

print "Use default parameters [y/n]? ";
my $answer = <STDIN>;
chomp $answer;

if ($answer eq "y") {
    $distance_between_SMRT_peaks = 8;
    $min_As = 5;
    $min_softclip = 2;
    $distance_between_ill_peaks = 8;
}
```

```

    $dist_SMRT_ill_d = 10;
    $dist_SMRT_ill_u = 4;
    $min_SMRT = 5;
    $min_ill = 1;
    $ann_dist = 10;
}
else {
    print "Enter desired window for collapsing Iso-Seq 3' ends (e.g.
8): ";
    $distance_between_SMRT_peaks = <STDIN>;
    chomp $distance_between_SMRT_peaks;

    print "Enter minimum number of As for Illumina poly(A) tails (e.g.
5): ";
    $min_As = <STDIN>;
    chomp $min_As;

    print "Enter minimum number of mismatches for Illumina poly(A)
tails (e.g. 2): ";
    $min_softclip = <STDIN>;
    chomp $min_softclip;

    print "Enter desired window for collapsing Illumina 3' ends (e.g.
8): ";
    $distance_between_ill_peaks = <STDIN>;
    chomp $distance_between_ill_peaks;

    print "Enter number of bases downstream of Iso-Seq ends to look for
Illumina support (e.g. 10): ";
    $dist_SMRT_ill_d = <STDIN>;
    chomp $dist_SMRT_ill_d;

    print "Enter number of bases upstream of Iso-Seq ends to look for
Illumina support (e.g. 4): ";
    $dist_SMRT_ill_u = <STDIN>;
    chomp $dist_SMRT_ill_u;

    print "Enter minimum number of Iso_seq reads to report a 3' end
(e.g. 5): ";
    $min_SMRT = <STDIN>;
    chomp $min_SMRT;

    print "Enter minimum number of Illumina poly(A) tails to support a
3' end (e.g. 1): ";
    $min_ill = <STDIN>;
    chomp $min_ill;

    print "Enter maximum distance in bp from an annotated end to be
called as 'annotated' (e.g. 10): ";
    $ann_dist = <STDIN>;
    chomp $ann_dist;
}

print "-----\n";

```

```
#####-----SMRT FILE PROCESSING-----#####
system("awk '\$3==\"$viral_chr\"' \Q$SMRT_file\E \| sort -k 4,4n >
\Q$SMRT_file\E.sorted.temp");
system("awk '\$2==0' \Q$SMRT_file\E.sorted.temp >
\Q$SMRT_file\E.sorted.plus.sam.temp");
system("awk '\$2==16' \Q$SMRT_file\E.sorted.temp >
\Q$SMRT_file\E.sorted.minus.sam.temp");
system("rm \Q$SMRT_file\E.sorted.temp");

#processing of PLUS sam file
open(INF, "<$SMRT_file.sorted.plus.sam.temp") or die "couldn't open
file";
open(OUT, ">$SMRT_file.sorted.plus.sam.read_ends.bedgraph.temp") or die
"couldn't open file";

my @dist;
my $sum;
my %plus_ends;
print "Processing Iso-Seq plus strand reads...\n";

while (my $line = <INF>) {
    chomp($line);
    my @cols = split("\t", $line);
    next if ($cols[5] =~ m/\d+S$/); #skips reads soft-clipped at the 3'
end
    while ($cols[5] =~ /(\d+)[DMNX=]/g) { #these lines use the CIGAR
string to determine the downstream coordinate
        push (@dist, $1);
    }
    $sum += $_ for @dist;
    my $end_coord = $cols[3] + $sum - 1; #subtract 1 to account for
start/end inclusion
    my $chr_end_coord = "$cols[2]\:$end_coord"; #combines the
chromosome and 3' end coordinate into a key to use for the hash
    $sum = 0;
    @dist = ();
    my @split_id = split("/", $cols[0]); #extracts the read depth for
this putative isoform from its id
    if (exists $plus_ends{$chr_end_coord}) { #if the key is already in
the hash, increases the value (count) by 1
        $plus_ends{$chr_end_coord} = $plus_ends{$chr_end_coord} +
$split_id[1];
    }
    else {
        $plus_ends{$chr_end_coord} = $split_id[1]; #if the key is not
already in the hash, adds it with a value (count) of the read depth
    }
}

foreach my $chr_end_coord (sort keys %plus_ends) { #prints out a(n
inadequately) sorted temporary bedgraph file
    my @split_keys = split(":", $chr_end_coord);
    print OUT $split_keys[0], "\t", $split_keys[1]-1, "\t",
$split_keys[1], "\t", $plus_ends{$chr_end_coord}, "\n"; #prints to
```

```

output file, converting chrStart to 0-based bedgraph coordinates
}
close(INF);
close(OUT);

system("rm \Q$SMRT_file\E.sorted.plus.sam.temp");

#processing of MINUS sam file
open(INF, "<$SMRT_file.sorted.minus.sam.temp") or die "couldn't open
file";
open(OUT, ">$SMRT_file.sorted.minus.sam.read_ends.bedgraph.temp") or
die "couldn't open file";

my $previous_coordinate=1;
my $count=0;
my $previous_chr = "start";
print "Processing Iso-Seq minus strand reads...\n";

while (my $line = <INF>) {
    chomp($line);
    my @cols = split("\t", $line);
    next if ($cols[5] =~ m/^\d+S/); #skips reads soft-clipped at the 3'
end
    my @split_id = split("\/", $cols[0]); #extracts the read depth for
this putative isoform from its id
    if (($cols[2] eq $previous_chr) and ($cols[3] ==
$previous_coordinate)) {
        $count = $count + $split_id[1]; #increases the count by the
read depth for the putative isoform
    }
    else {
        if ($previous_chr eq "start") { #doesn't print out the
placeholder first line.
            $previous_chr = $cols[2];      #sets the previous
chromosome, previous coordinate and count values
            $previous_coordinate = $cols[3];
            $count = $split_id[1];
        }
        else {
            print OUT $previous_chr, "\t", $previous_coordinate-1,
"\t", $previous_coordinate, "\t-", $count, "\n"; #prints to output
file, converting chrStart to 0-based bedgraph coordinates
            $previous_chr = $cols[2];
            $previous_coordinate = $cols[3];
            $count = $split_id[1];
        }
    }
}

print OUT $previous_chr, "\t", $previous_coordinate-1, "\t",
$previous_coordinate, "\t-", $count, "\n"; #prints the last start
coordinates to output file
close(INF);
close(OUT);

```

```

system("cat \Q$SMRT_file\E.sorted.plus.sam.read_ends.bedgraph.temp
\Q$SMRT_file\E.sorted.minus.sam.read_ends.bedgraph.temp | sort -k2,3n >
\Q$SMRT_file\E.\Q$viral_chr\E.read_ends.bedgraph.noheader");

system("rm \Q$SMRT_file\E.sorted.plus.sam.read_ends.bedgraph.temp");
system("rm \Q$SMRT_file\E.sorted.minus.sam.read_ends.bedgraph.temp");
system("rm \Q$SMRT_file\E.sorted.minus.sam.temp");

#add header to bedgraph file
open(INF, "<$SMRT_file.$viral_chr.read_ends.bedgraph.noheader") or die
"couldn't open file";
open(OUT, ">$SMRT_file.$viral_chr.read_ends.bedgraph") or die "couldn't
open file";

print OUT "track type=bedGraph
name=\"\$SMRT_file.$viral_chr.read_ends.bedgraph\" description=\"3' ends
of Iso-Seq reads from end_finder_sam_to_bed.pl\"\n";
while (my $line = <INF>) {
    print OUT $line;
}
close(OUT);
close(INF);

system("rm \Q$SMRT_file\E.\Q$viral_chr\E.read_ends.bedgraph.noheader");

#make a bed file from the SMRT bedgraph file:
open(INF, "<$SMRT_file.$viral_chr.read_ends.bedgraph") or die "couldn't
open file";
open(OUT, ">$SMRT_file.ends.temp.bed") or die "couldn't open file";

print "Combining Iso-Seq 3' ends within $distance_between_SMRT_peaks of
each other and calculating consensus 3' ends...\n";
collapse_bedgraph($distance_between_SMRT_peaks);

close(INF);
close(OUT);

system("sort -k 1,1 -k 2,2n \Q$SMRT_file\E.ends.temp.bed >
\Q$SMRT_file\E.ends.bed.noheader");
system("rm \Q$SMRT_file.ends.temp.bed\E");

#add header to bed file
open(INF, "<$SMRT_file.ends.bed.noheader") or die "couldn't open file";
open(OUT, ">$SMRT_file.$viral_chr.SMRT_ends.bed") or die "couldn't open
file";

print OUT "track type=bed name=\"\$SMRT_file.$viral_chr.SMRT_ends.bed\"
description=\"consensus 3' ends of Iso-Seq reads within
$distance_between_SMRT_peaks bp collapsed to weighted center from
end_finder_sam_to_bed.pl\"\n";
while (my $line = <INF>) {
    print OUT $line;
}
close(OUT);
close(INF);

```

```

system("rm \Q$SMRT_file\E.ends.bed.noheader");

#####-----ILLUMINA FILE PROCESSING-----#####

open(INF, "<$ill_file") or die "couldn't open input file";
open(OUT, ">$ill_file.polyA_ends.temp") or die "couldn't open output
file";

print "Extracting Illumina reads with at least $min_As As and at least
$min_softclip mismatches...\n";

while (my $line = <INF>) {
    chomp($line);
    my @cols = split("\t", $line);
    next if ($cols[0] eq "@HD" || $cols[0] eq "@PG" || $cols[0] eq
"@SQ"); #skips SAM file header lines
    next if $cols[2] ne $viral_chr;
    if ($cols[1] == 81 || $cols[1] == 83 || $cols[1] == 89 || $cols[1]
== 16) { #selects reads with FLAG codes indicating they are first in
pair on the plus strand
        if (($cols[5] =~ m/\d+S$/) and ($cols[9] =~ m/A{$min_As}$/)) {
# selects reads with softclipping and a run of As at the end
            my ($softclips) = $cols[5] =~ m/(\d+)S/; #pulls out the
number of softclipped bases
            if ($softclips > $min_softclip) { #selects reads with at
least the specified number of softclipped bases
                print OUT $line, "\n";
            }
        }
    }
    elsif ($cols[1] == 73 || $cols[1] == 97 || $cols[1] == 99 ||
$cols[1] == 0) { #selects reads with FLAG codes indicating they are
first in pair on the minus strand
        if (($cols[5] =~ m/^\d+S/) and ($cols[9] =~ m/^\T{$min_As}$/)) {
#selects reads with softclipping and a run of Ts at the beginning
            my ($softclips) = $cols[5] =~ m/^\(\d+)S/; #pulls out the
number of softclipped bases
            if ($softclips > $min_softclip) { #selects reads with at
least the specified number of softclipped bases
                print OUT $line, "\n";
            }
        }
    }
}

close(INF);
close(OUT);

system ("sort -k 4,4n \Q$ill_file\E.polyA_ends.temp >
\Q$ill_file\E.polyA_ends.sam");
system ("rm \Q$ill_file\E.polyA_ends.temp");

open(INF, "<$ill_file.polyA_ends.sam") or die "couldn't open file";
open(OUT, ">$ill_file.polyA_sites.temp") or die "couldn't open file";

```

```

print "Processing Illumina reads with polyA tails...\n";
#create a file with the coordinates corresponding to the polyA ends of
the reads, and sort it by those coordinates

my $cigar_sum;
my $cigar_calc;
my @plus_ends;
my @read_dist;

while (my $line = <INF>) {
    chomp($line);
    my @cols = split("\t", $line);
    if ($cols[1] == 73 || $cols[1] == 97 || $cols[1] == 99 || $cols[1]
== 0) { #minus strand
        print OUT "$cols[2]\t$ccols[3]\t0\n";
    }
    elsif ($cols[1] == 81 || $cols[1] == 83 || $cols[1] == 89 ||
$ccols[1] == 16) { #plus strand
        while ($cols[5] =~ /\d+([DMNX=])/g) { #these lines use the CIGAR
string to determine the downstream coordinate
            push (@read_dist, $1);
        }
        $cigar_sum += $_ for @read_dist;
        $cigar_calc = $cols[3] + $cigar_sum - 1; #subtract one to account
for start/end inclusion
        $cigar_sum = 0;
        @read_dist = ();
        print OUT "$cols[2]\t$cigar_calc\t1\n";
    }
}
close(INF);
close(OUT);

system("sort -k 1,1 -k 2,2n \Q$ill_file.polyA_sites.temp\E >
\Q$ill_file.polyA_sites.temp\E.sorted");

#create a bedgraph file from the sorted coordinates file

open(INF, "<$ill_file.polyA_sites.temp.sorted") or die "couldn't open
file";
open(OUT, ">$ill_file.polyA_sites.temp.bedgraph") or die "couldn't open
file";

my $chrom_minus;
my $previous_coordinate_m=0;
my $count_m=0;
my $chrom_plus;
my $previous_coordinate_p=0;
my $count_p=0;

while (my $line = <INF>) {
    my @cols = split("\t", $line);

```



```

#reads on the plus strand:
if ($cols[2] == 1) {
    if ($chrom_plus) { #if $chrom_plus has been defined (i.e. there
is a previous plus strand read)
        if (($cols[0] eq $chrom_plus) and ($cols[1] ==
$previous_coordinate_p)) {
            $count_p++;
        }
        else {
            print OUT $chrom_plus, "\t", $previous_coordinate_p-
1, "\t", $previous_coordinate_p, "\t", $count_p, "\n"; #prints to
output file, converting chrStart to 0-based bedgraph coordinates
            $previous_coordinate_p = $cols[1];
            $count_p = 1;
        }
    }
    else { #if $chrom_plus has not been defined (i.e. there is no
previous plus strand read)
        $chrom_plus = $cols[0];
        $previous_coordinate_p = $cols[1];
        $count_p = 1;
    }
}

#reads on the minus strand:
elsif ($cols[2] == 0) {
    if ($chrom_minus) {
        if (($cols[0] eq $chrom_minus) and ($cols[1] ==
$previous_coordinate_m)) {
            $count_m++;
        }
        else {
            print OUT $chrom_minus, "\t", $previous_coordinate_m-
1, "\t", $previous_coordinate_m, "\t-", $count_m, "\n"; #prints to
output file, converting chrStart to 0-based bedgraph coordinates
            $chrom_minus = $cols[0];
            $previous_coordinate_m = $cols[1];
            $count_m = 1;
        }
    }
    else {
        $chrom_minus = $cols[0];
        $previous_coordinate_m = $cols[1];
        $count_m = 1;
    }
}

}

#prints to output file, converting chrStart to 0-based bedgraph
coordinates
print OUT $chrom_plus, "\t", $previous_coordinate_p-1, "\t",
$previous_coordinate_p, "\t", $count_p, "\n";
print OUT $chrom_minus, "\t", $previous_coordinate_m-1, "\t",
$previous_coordinate_m, "\t-", $count_m, "\n";

close(INF);

```

```

close(OUT);

system("sort -k 1,1 -k 2,2n \Q$ill_file\E.polyA_sites.temp.bedgraph >
\Q$ill_file\E.polyA_sites.bedgraph.noheader");
system("rm \Q$ill_file\E.polyA_sites.temp.bedgraph");
system("rm \Q$ill_file\E.polyA_sites.temp.sorted");
system("rm \Q$ill_file\E.polyA_sites.temp");

#add header to bedgraph file
open(INF, "<$ill_file.polyA_sites.bedgraph.noheader") or die "couldn't
open file";
open(OUT, ">$ill_file.$viral_chr.polyA_sites.bedgraph") or die
"couldn't open file";

print OUT "track type=bedGraph
name=\"\Q$ill_file.$viral_chr.polyA_sites.bedgraph\" description=\"polyA
sites in Illumina reads with at least 5As and at least 2 mismatches
from end_finder_sam_to_bed.pl\"\n";
while (my $line = <INF>) {
    print OUT $line;
}
close(OUT);
close(INF);

system("rm \Q$ill_file\E.polyA_sites.bedgraph.noheader");

#make a bed file from the Illumina bedgraph file:
open(INF, "<$ill_file.$viral_chr.polyA_sites.bedgraph") or die
"couldn't open file";
open(OUT, ">$ill_file.$viral_chr.polyA_sites.temp.bed") or die
"couldn't open file";

print "Combining Illumina polyA tails within
$distance_between_ill_peaks of each other and calculating consensus 3'
ends...\n";
collapse_bedgraph($distance_between_ill_peaks);

close(INF);
close(OUT);

system("sort -k 1,1 -k 2,2n
\Q$ill_file\E.\Q$viral_chr\E.polyA_sites.temp.bed >
\Q$ill_file\E.\Q$viral_chr\E.polyA_sites.bed.noheader");
system("rm \Q$ill_file\E.\Q$viral_chr\E.polyA_sites.temp.bed");

#add header to bed file
open(INF, "<$ill_file.$viral_chr.polyA_sites.bed.noheader") or die
"couldn't open file";
open(OUT, ">$ill_file.$viral_chr.polyA_sites.bed") or die "couldn't
open file";

print OUT "track type=bed name=\"\Q$ill_file.$viral_chr.polyA_sites.bed\"
description=\"consensus polyA sites of Illumina reads with tails of 5
As with 2 mismatches within $distance_between_ill_peaks bp collapsed to
weighted centers from end_finder_sam_to_bed.pl\"\n";

```

```

while (my $line = <INF>) {
    print OUT $line;
}
close(OUT);
close(INF);

system("rm \Q$ill_file\E.\Q$viral_chr\E.polyA_sites.bed.noheader");

#####-----SEEKING ILLUMINA SUPPORT FOR SMRT ENDS-----
#####

open(INF, "<$ill_file.$viral_chr.polyA_sites.bed" ) or die "couldn't
open file";

print "Extracting Iso-Seq 3' ends with Illumina polyA tails within
$dist_SMRT_ill_d bases downstream or $dist_SMRT_ill_u upstream...\n";

my %features_ill;
my $key_combo_ill;

while(my $line = <INF> ) {
    chomp($line);
    next if ($line =~ /^track/); #skips the track definition line
    my @cols = split("\t", $line);
    if ($cols[5] eq "+") { #for each line in the Illumina polyA reads
        bed file, creates a key for the hash combining coordinate and strand.
        Selects chrEnd for ends on the plus strand and chrStart for ends on the
        minus strand.
        $key_combo_ill = "$cols[2]:$cols[5]";
    }
    if ($cols[5] eq "-") {
        $key_combo_ill = "$cols[1]:$cols[5]";
    }
    $features_ill{$key_combo_ill} = $cols[4]; #enters a count value for
    the key into the hash
}

close(INF);

open(INF, "<$SMRT_file.$viral_chr.SMRT_ends.bed" ) or die "couldn't
open file";
open(OUT, ">$SMRT_file.$viral_chr.ends.bed.illumina_support.bed.temp");

my $ill_coord;
my $match_count;
my $lower_limit;
my $upper_limit;

while(my $line = <INF>) {
    chomp($line);
    next if ($line =~ /^track/); #skips the track definition line
    my @SMRT_cols = split("\t", $line);
    next if (abs $SMRT_cols[4] < $min_SMRT);
    foreach my $key_combo_ill (keys %features_ill) {
        my @ill_cols = split(":", $key_combo_ill);
    }
}

```

```

        next if (abs $features_ill{$key_combo_ill} < $min_ill);

        if ($SMRT_cols[5] eq "+") { #sets boundaries for plus strand
support
            $lower_limit = $SMRT_cols[2]-$dist_SMRT_ill_u;
            $upper_limit = $SMRT_cols[2]+$dist_SMRT_ill_d;
        }
        if ($SMRT_cols[5] eq "-") { #sets boundaries for minus strand
support
            $lower_limit = $SMRT_cols[1]-$dist_SMRT_ill_d;
            $upper_limit = $SMRT_cols[1]+$dist_SMRT_ill_u;
        }
        if (($SMRT_cols[5] eq $ill_cols[1]) and ($ill_cols[0] >=
$lower_limit) and ($ill_cols[0] <= $upper_limit)) {

            if ($match_count) { #if more than one Illumina end matches
the SMRT end, selects Illumina end with the most reads
                if ($features_ill{$key_combo_ill} > $match_count){
                    $match_count = $features_ill{$key_combo_ill};
                    $ill_coord = $ill_cols[0];
                }
            }
            else {
                $match_count = $features_ill{$key_combo_ill};
                $ill_coord = $ill_cols[0];
            }
        }

        if ($match_count) {
            if ($SMRT_cols[5] eq "+") {
                my $name = "$SMRT_cols[4].IsoSeq_$match_count.Ill";
                my $count = $match_count + $SMRT_cols[4];
                print OUT $SMRT_cols[0], "\t", $ill_coord-1, "\t",
$ill_coord, "\t", $name, "\t", $count, "\t", $SMRT_cols[5], "\t",
$SMRT_cols[3], "\n"; #prints to output, adjusting chrStart to 0-based
                undef($match_count);
            }
            if ($SMRT_cols[5] eq "-") {
                my $name = "$SMRT_cols[4].IsoSeq_$match_count.Ill";
                my $count = $match_count + $SMRT_cols[4];
                print OUT $SMRT_cols[0], "\t", $ill_coord, "\t",
$ill_coord+1, "\t", $name, "\t", $count, "\t", $SMRT_cols[5], "\t",
$SMRT_cols[3], "\n"; #prints to output, adujsting chrEnd
                undef($match_count);
            }
        }
        else {
            my @range_cols = split (":", $SMRT_cols[3]); #includes SMRT
ends that are not supported by Illumina in this temporary file
            print OUT
"$SMRT_cols[0]\t$SMRT_cols[1]\t$SMRT_cols[2]\t$range_cols[2].IsoSeq\t$r
ange_cols[2]\t$SMRT_cols[5]\t$SMRT_cols[3]\n";
        }
    }
}

```

```

close(OUT);
close(INF);

#####-----COMPARING TO ANNOTATED ENDS-----#####
open(INF, "<$ann_file" ) or die "couldn't open file";

print "Processing annotation file...\n";

#extract 3' ends from the annotation file:
#annotation file must be sorted by chrStart then chrEnd!
my @annotated_ends;
my $plus_prev_coord = 0;
my $minus_prev_coord = 0;

while(my $line = <INF>) {
    chomp($line);
    next if ($line =~ /^track/); #skips the track definition line
    my @ann_cols = split("\t", $line);
    next if $ann_cols[0] ne $viral_chr; #skip lines that aren't viral
    if ($ann_cols[5] eq "+") {
        if ($ann_cols[2] != $plus_prev_coord) {
            push (@annotated_ends, "$ann_cols[2]:$ann_cols[5]");
#creates an array with chrEnd and strand
            $plus_prev_coord = $ann_cols[2];
        }
    }
    elsif ($ann_cols[5] eq "-"){
        if ($ann_cols[1] != $minus_prev_coord) {
            push (@annotated_ends, "$ann_cols[1]:$ann_cols[5]");
#creates an array with chrStart and strand
            $minus_prev_coord = $ann_cols[1];
        }
    }
}

my $annotated = scalar @annotated_ends;

close(INF);

#compare ends in the altered SMRT ends file (that already has info
about Illumina ends) with annotated ends

open(INF, "<$SMRT_file.$viral_chr.ends.bed.illumina_support.bed.temp" )
or die "couldn't open file";
open(OUT, ">$SMRT_file.$viral_chr.validated_ends.bed");

print "Comparing Iso-Seq ends to annotated ends...\n";

print OUT "track type=bedDetail
name=\"$SMRT_file.$viral_chr.SMRT_ends.bed.illumina_support.bed\"
description=\"validated ends supported by at least $min_SMRT Iso-Seq
read ends within $distance_between_SMRT_peaks bp, with an Illumina
polyA site within $dist_SMRT_ill_d bp downstream or $dist_SMRT_ill_u bp

```

upstream, or within \$ann\_dist bp of an annotated end. Illumina polyA sites have at least \$min\_ill reads with \$min\_As As and \$min\_softclip mismatches, within \$distance\_between\_ill\_peaks bp of each other. From end\_finder\_sam\_to\_bed.pl\"n";

```
my $annotated_found_by_SMRT = 0;
my $novel_found_by_SMRT_ill = 0;
my $SMRT_annotated = 0; #this is different than
$annotated_found_by_SMRT because depending on input parameters two SMRT
ends may correspond to a single annotated end or vice versa.
```

```
while(my $line = <INF>) {
    chomp($line);
    my @SMRT_cols = split("\t", $line);
    my $found_flag=0;
    foreach my $ann_end (@annotated_ends) {
        my @ann_cols = split(":", $ann_end);
        my $lower_limit = $ann_cols[0]-$ann_dist;
        my $supper_limit = $ann_cols[0]+$ann_dist;
        if ($ann_cols[1] eq "+") {
            if (($SMRT_cols[5] eq $ann_cols[1]) and
($SMRT_cols[2]>=$lower_limit) and ($SMRT_cols[2]<=$supper_limit)) {
                if ($found_flag == 0) {
                    print OUT
"$SMRT_cols[0]\t$SMRT_cols[1]\t$SMRT_cols[2]\tann_$SMRT_cols[5]_$SMRT_c
ols[3]\t$SMRT_cols[4]\t$SMRT_cols[5]\t$SMRT_cols[6]\n";
                    $found_flag = 1;
                    $annotated_found_by_SMRT++;
                    $SMRT_annotated++;
                }
                elsif ($found_flag == 1) {
                    $annotated_found_by_SMRT++;
                }
            }
        }
        if ($ann_cols[1] eq "-") {
            if (($SMRT_cols[5] eq $ann_cols[1]) and
($SMRT_cols[1]>=$lower_limit) and ($SMRT_cols[1]<=$supper_limit)) {
                if ($found_flag == 0) {
                    print OUT $SMRT_cols[0], "\t", $SMRT_cols[1], "\t",
$SMRT_cols[2], "\tann_", $SMRT_cols[5], "_", $SMRT_cols[3], "\t",
abs($SMRT_cols[4]), "\t", $SMRT_cols[5], "\t", $SMRT_cols[6], "\n";
                    $found_flag = 1;
                    $annotated_found_by_SMRT++;
                    $SMRT_annotated++;
                }
                elsif ($found_flag == 1) {
                    $annotated_found_by_SMRT++;
                }
            }
        }
    }
}
if ($found_flag == 0) {
    if ($SMRT_cols[3] =~ /.+IsoSeq_.+Ill/) {
        print OUT
```

```

"$SMRT_cols[0]\t$SMRT_cols[1]\t$SMRT_cols[2]\tnov_$SMRT_cols[5]_SMRT_c
ols[3]\t$SMRT_cols[4]\t$SMRT_cols[5]\t$SMRT_cols[6]\n";
    $novel_found_by_SMRT_ill++;
}
}
}

my $total_found = $SMRT_annotated + $novel_found_by_SMRT_ill;

close(INF);
close(OUT);

print "-----\n";

open(OUT, ">${viral_chr}_validated_ends_stats.txt");

if ($total_found > 0) {
    if ($SMRT_annotated != $annotated_found_by_SMRT) {
        print "$total_found 3' ends found. $novel_found_by_SMRT_ill are
novel, $SMRT_annotated are annotated. $annotated_found_by_SMRT out of
$annotated total annotated 3' ends are found.\nNote that two annotated
ends may be within $ann_dist bp of a single Iso-Seq end or vice
versa.\n";
        print OUT "$viral_chr\n\n$total_found 3'
ends\n\t$novel_found_by_SMRT_ill novel\n\t$SMRT_annotated
annotated\n\n$annotated 3' ends in annotation\n\t$annotated_found_by_SMRT
detected by Iso-Seq\n\ninput
files:\n\t$SMRT_file\n\t$ill_file\n\t$ann_file\n";
    }
    else {
        print "$total_found 3' ends found. $novel_found_by_SMRT_ill are
novel, $SMRT_annotated are annotated (out of a total of $annotated
annotated 3' ends).\n\n";
        print OUT "$viral_chr\n\n$total_found 3'
ends\n\t$novel_found_by_SMRT_ill novel\n\t$SMRT_annotated
annotated\n\n$annotated 3' ends in annotation\n\ninput
files:\n\t$SMRT_file\n\t$ill_file\n\t$ann_file\n";
    }
}
else {
    print "No 3' ends validated\n";
    print OUT "No validated 3' ends found\n\ninput
files:\n\t$SMRT_file\n\t$ill_file\n\t$ann_file\n";
}

close(OUT);

system("rm
\Q$SMRT_file\E.\Q$viral_chr\E.ends.bed.illumina_support.bed.temp");

#####
sub collapse_bedgraph {
    my ($distance_between_peaks) = shift;
    my $prev_coord_plus = 0;
    my $prev_coord_minus = 0;

```

```

my $count_sum_plus = 0;
my $count_sum_minus = 0;
my $weighted_coordinate_sum_plus = 0;
my $weighted_coordinate_sum_minus = 0;
my $weighted_average_plus;
my $weighted_average_minus;
my $first_plus = 1;
my $first_minus = 1;
my @coords_plus;
my @coords_minus;
my $chrStart_plus;
my $chrEnd_plus;
my $chrStart_minus;
my $chrEnd_minus;

while (my $line = <INF>) {
    chomp($line);
    next if ($line =~ /^track/); #skips the track definition line
    my @cols = split("\t", $line);
    if ($cols[3] > 0) { #if this coordinate has a positive count...
        if ($cols[2] <= $prev_coord_plus +
($distance_between_peaks)) { #if the coordinate is within the specified
number of bp of the previous coordinate
            $count_sum_plus = $count_sum_plus + $cols[3]; #adds to
the sums to eventually calculate the weighted average
            $weighted_coordinate_sum_plus =
$weighted_coordinate_sum_plus + ($cols[2]*$cols[3]);
            push (@coords_plus, $cols[2]);
            $prev_coord_plus = $cols[2]; #sets the current
coordinate as the "previous coordinate" before moving on
        }
        else { #if the present coordinate is not within the
specified number of bp of the previous coordinate, need to print out a
feature
            if ($first_plus == 1) { #"first" flag avoids wonkiness
if the first coordinate is far from coordinate 1 (don't need to print
out a feature yet)
                $count_sum_plus = $cols[3];
                $weighted_coordinate_sum_plus = $cols[2]*$cols[3];
                $prev_coord_plus = $cols[2];
                push (@coords_plus, $cols[2]);
                $first_plus = 0;
            }
            else {
                $weighted_average_plus = sprintf("%.0f",
($weighted_coordinate_sum_plus/$count_sum_plus)); #calculates weighted
average
                $chrStart_plus = $coords_plus[0];
                $chrEnd_plus = pop(@coords_plus);
                print OUT $viral_chr, "\t", $weighted_average_plus-
1, "\t", $weighted_average_plus, "\t", $chrStart_plus, ":",
$chrEnd_plus, ":", $count_sum_plus, "\t", $count_sum_plus, "\t+\n";
                #prints out weighted average for plus strand features. Use printf to
round the weighted average.
                @coords_plus = ($cols[2]);
            }
        }
    }
}

```



```

        $count_sum_plus = $cols[3]; #sets "previous
coordinate", count and sum of counts for the current coordinate
        $weighted_coordinate_sum_plus = $cols[2]*$cols[3];
        $prev_coord_plus = $cols[2];
    }
}
}
elseif ($cols[3] < 0) { #if this coordinate has a negative
count...
    if ($cols[1] <= $prev_coord_minus +
($distance_between_peaks)) { #if the coordinate is within the specified
number of bp of the previous coordinate
        $count_sum_minus = $count_sum_minus + $cols[3]; #adds
to the sums to eventually calculate the weighted average
        $weighted_coordinate_sum_minus =
$weighted_coordinate_sum_minus + ($cols[1]*$cols[3]);
        push (@coords_minus, $cols[1]);
        $prev_coord_minus = $cols[1]; #sets the current
coordinate as the "previous coordinate" before moving on
    }
    else { #if the present coordinate is not within the
specified number of bp of the previous coordinate, need to print out a
feature
        if ($first_minus == 1) { #"first" flag avoids wonkiness
if the first coordinate is far from coordinate 1 (don't need to print
out a feature yet)
            $count_sum_minus = $cols[3];
            $weighted_coordinate_sum_minus = $cols[1]*$cols[3];
            $prev_coord_minus = $cols[1];
            push (@coords_minus, $cols[1]);
            $first_minus = 0;
        }
        else {
            $weighted_average_minus = sprintf("%1.0f",
($weighted_coordinate_sum_minus/$count_sum_minus)); #calculates
weighted average
            $chrStart_minus = $coords_minus[0];
            $chrEnd_minus = pop(@coords_minus);
            print OUT $viral_chr, "\t",
$weighted_average_minus, "\t", $weighted_average_minus+1, "\t",
$chrStart_minus, ":", $chrEnd_minus, ":", $count_sum_minus, "\t",
abs($count_sum_minus), "\t-\n";
            @coords_minus = ($cols[1]);
            $count_sum_minus = $cols[3]; #sets "previous
coordinate", count and sum of counts for the current coordinate
            $weighted_coordinate_sum_minus = $cols[1]*$cols[3];
            $prev_coord_minus = $cols[1];
        }
    }
}
}

if ($count_sum_plus > 0) {#calculates and prints out weighted
average for the last feature (plus strand)
    $weighted_average_plus = sprintf("%1.0f",

```

```

($weighted_coordinate_sum_plus/$count_sum_plus));
    $chrStart_plus = $coords_plus[0];
    $chrEnd_plus = pop(@coords_plus);
    print OUT $viral_chr, "\t", $weighted_average_plus-1, "\t",
$weighted_average_plus, "\t", $chrStart_plus, ":", $chrEnd_plus, ":",
$count_sum_plus, "\t", $count_sum_plus, "\t+\n";
}

    if ($count_sum_minus < 0) {#calculates and prints out weighted
average for the last feature (minus strand)
        $weighted_average_minus = sprintf("%1.0f",
($weighted_coordinate_sum_minus/$count_sum_minus));
        $chrStart_minus = $coords_minus[0];
        $chrEnd_minus = pop(@coords_minus);
        print OUT $viral_chr, "\t", $weighted_average_minus, "\t",
$weighted_average_minus+1, "\t", $chrStart_minus, ":", $chrEnd_minus,
":", $count_sum_minus, "\t", abs($count_sum_minus), "\t-\n";
    }
}

```

## **APPENDIX 4**

TRIMD\_structure\_validator.pl

```

#!/usr/bin/perl
#Takes a sam file of Iso-Seq fl isoforms and compares them to a list of
validated 5' ends, 3' ends and introns to create a list of validated
transcript structures, which are compared to an annotation file.

use warnings;
use strict;

die "USAGE: 'perl <PATH/TRIMD_transcript_validator.pl> </PATH/Iso-
Seq_sam_file> </PATH/validated_starts_file> </PATH/validated_ends_file>
</PATH/validated_introns_file> </PATH/Annotation_bed_file>' " unless
@ARGV == 5;

my ($test_file, $valid_starts_file, $valid_ends_file,
    $valid_introns_file, $ann_file) = (@ARGV);

print "Enter maximum distance from an annotated 5' start to be called
annotated (e.g. 10): ";
my $start_dist = <STDIN>;
chomp $start_dist;

print "Enter maximum distance from an annotated 3' end to be called
annotated (e.g. 10): ";
my $end_dist = <STDIN>;
chomp $end_dist;

#Convert SMRT sam file to bed:

print "-----\nReformatting
Iso-Seq file...\n";

open(INF, "<$test_file") or die "couldn't open input file";
open(OUT, ">$test_file.bed") or die "couldn't open output file";

while (my $line = <INF>) {
    $line =~ s/\r//g;
    chomp($line);
    next if ($line =~ m/\@/); #skips SAM header lines
    my @cols = split("\t", $line);
    my @split_id = split("\/", $cols[0]);
    my $strand;
    my $chr = $cols[2];
    my $chr_start = $cols[3] - 1;
    my $chr_end = 0;
    my $feature_name = $cols[0];
    my $score = $split_id[1];
    my $color = "133,0,33";
    if ($cols[1] == 0) {
        $strand = "+";
    }
}

```

```

}
elseif ($cols[1] == 16) {
    $strand = "-";
}
else {
    next; #skips isoforms that aren't mapped
}
my @split_CIGAR_temp = split(/(\d+\D)/, $cols[5]); #splits CIGAR
code into segments and puts segments into an array (but also the empty
values between them)
my @split_CIGAR;
foreach my $temporary(@split_CIGAR_temp) { #removes empty values
from the array
    if ($temporary =~ m/\d+\D/) {
        push(@split_CIGAR, $temporary);
    }
}
my $exon_sum = 0;
my @exon_lengths = ();
my @block_starts = (0);
my $count = 0;
foreach my $split_CIGAR(@split_CIGAR) {
    $count++;
    if (($count == 1) && (my ($five_prime_clipped_bases) =
$split_CIGAR =~ m/(\d+)S$/)) { #ignores soft clipping at the beginning
    }

    elseif (($count > 1) && (my ($three_prime_clipped_bases) =
$split_CIGAR =~ m/(\d+)S$/)) { #ignores soft clipping at the end
    }

    elseif ($split_CIGAR =~ m/N$/) { #if element is an intron...
        push(@exon_lengths, $exon_sum); #...adds the last value to
the exon sum...
        my ($intron_length) = $split_CIGAR =~ m/(\d+)/; #...gets
intron length...
        my $new_block_start = $exon_lengths[-1] + $block_starts[-1]
+ $intron_length; #...calculates new blockStart...
        push(@block_starts, $new_block_start); #...adds new
blockStart to array...
        $exon_sum = 0; #...and resets the exon sum
    }
    else {
        my ($value) = $split_CIGAR =~ m/(\d+)/;
        if ($split_CIGAR =~ m/I$/) { #ignores insertions
            $exon_sum = $exon_sum - 0;
        }
        else { #adds matches, mismatches and deletions to the exon
sum
            $exon_sum = $exon_sum + $value;
        }
    }
}
push(@exon_lengths, $exon_sum); #at the end of the CIGAR array,
push the last exon sum into the exon_lengths array
$chr_end = $chr_start + $block_starts[-1] + $exon_lengths[-1];

```

```

    my $exon_number = @exon_lengths;
    print OUT $chr, "\t", $chr_start, "\t", $chr_end, "\t",
$feature_name, "\t", $score, "\t", $strand, "\t", $chr_start, "\t",
$chr_end, "\t", $color, "\t", $exon_number, "\t", join("\t",
@exon_lengths), "\t", join("\t", @block_starts), "\n";
}
close(INF);
close(OUT);

#Create an array of validated start sites from the start sites input
file:
open(INF, "<$valid_starts_file") or die "couldn't open file";

my @valid_start;

while (my $line = <INF>) {
    chomp($line);
    next if ($line =~ /^track/); #skips the track definition line
    push (@valid_start, $line); #puts each line of the start sites file
into an array to be checked later
}
close(INF);

#Check each start site in the SMRT reads file against the array of
validated start sites:
open(INF, "<$test_file.bed") or die "couldn't open file";
#open(OUT, ">$test_file.valid_start.bed.temp"); #uncomment to print out
file of isoforms with validated starts

print "Checking start sites...\n";

my @good_start;
my $new_start_line;

while (my $line = <INF>) {
    chomp($line);
    next if ($line =~ /^track/); #skips the track definition line
    my ($chrom, $chromStart, $chromEnd, $name, $score, $strand,
    $thickStart, $thickEnd, $itemRgb, $blockCount, $blockSizes,
    $blockStarts) = split("\t", $line);
    if ($strand eq "+") {
        foreach my $valid_start (@valid_start) { #checks to see if the 5'
end of the (plus strand) SMRT transcript matches a range of possible
start site values from the list of validated start sites
            my @start_cols = split("\t", $valid_start);
            my ($range_start, $range_end, $SMRT_depth) = split(":",
$start_cols[6]);
            if (($chrom eq $start_cols[0]) and ($strand eq
$start_cols[5]) and ($chromStart >= $range_start) and ($chromStart <=
$range_end)) {
                $new_start_line = "$line\t$start_cols[1]"; #creates a
line for the read, changing the start site to the consensus start site
and adding an extra field with the original start site
                push (@good_start, $new_start_line); #if the start
site matches, pushes the line into a new array of SMRT transcripts with

```

```

validated 5' ends
    #print OUT $new_start_line, "\n"; #uncomment to print
out file of isoforms with validated starts
    last;
}
}
}
elseif ($strand eq "-"){
    foreach my $valid_start (@valid_start) { #checks to see if the 5'
end of the (minus strand) SMRT transcript matches a range of possible
start site values from the list of validated start sites
        my @start_cols = split("\t", $valid_start);
        my ($range_start, $range_end, $SMRT_depth) = split(":",
$start_cols[6]);
        if (($chrom eq $start_cols[0]) and ($strand eq
$start_cols[5]) and ($chromEnd >= $range_start) and ($chromEnd <=
$range_end)) {
            $new_start_line = "$line\t$start_cols[2]"; #creates a
line for the read, changing the start site to the consensus start site
and adding an extra field with the original start site
            push (@good_start, $new_start_line); #if the start
site matches, pushes the line into a new array of SMRT transcripts with
validated 5' ends
            #print OUT $new_start_line, "\n"; #uncomment to print
out file of isoforms with validated starts
            last;
        }
    }
}
}

my $good_start_number = scalar @good_start;

#close(OUT); #uncomment to print out file of isoforms with validated
starts and ends
close(INF); #have an array in memory of reads that have validated 5'
ends, and their newly estimated 5' ends. Can uncomment lines to have an
output a file of reads (in their original form) that have validated 5'
ends.

#Create an array of validated end sites from the end sites input file:
open(INF, "<$valid_ends_file") or die "couldn't open file";

my @valid_end;

while (my $line = <INF>) {
    chomp($line);
    next if ($line =~ /^track/); #skips the track definition line
    push (@valid_end, $line); #puts each line of the end sites file
into an array to be checked later
}

close(INF);

#open(OUT, ">$test_file.valid_start_and_end.bed.temp"); #uncomment to

```

```

print out file of isoforms with validated starts and ends

print "Checking end sites...\n";

my @good_start_and_end;
my $new_end_line;

foreach my $good_start (@good_start) { #starts with the array of SMRT
transcripts with validated 5' ends
    my ($chrom, $chromStart, $chromEnd, $name, $score, $strand,
    $thickStart, $thickEnd, $itemRgb, $blockCount, $blockSizes,
    $blockStarts, $new_coord) = split("\t", $good_start);
    if ($strand eq "+") { #determines 3' end of the SMRT transcript
        foreach my $valid_end (@valid_end) { #checks to see if the 3' end
of the SMRT transcript matches a range of possible 3' end values from
the list of validated start sites
            my @end_cols = split("\t", $valid_end);
            my ($range_start, $range_end, $SMRT_depth) = split(":",
$end_cols[6]);
            if (($chrom eq $end_cols[0]) and ($strand eq $end_cols[5])
and ($chromEnd >= $range_start) and ($chromEnd <= $range_end)) {
                $new_end_line = "$good_start\t$end_cols[2]";
                push (@good_start_and_end, $new_end_line);
                #print OUT $new_end_line, "\n"; #uncomment to print out
file of isoforms with validated starts and ends
                last;
            }
        }
    }
    elsif ($strand eq "-") {
        foreach my $valid_end (@valid_end) { #checks to see if the 3'
end of the SMRT transcript matches a range of possible 3' end values
from the list of validated start sites
            my @end_cols = split("\t", $valid_end);
            my ($range_start, $range_end, $SMRT_depth) = split(":",
$end_cols[6]);
            if (($chrom eq $end_cols[0]) and ($strand eq $end_cols[5])
and ($chromStart >= $range_start) and ($chromStart <= $range_end)) {
                $new_end_line =
"$chrom\t$chromStart\t$chromEnd\t$name\t$score\t$strand\t$thickStart\t$
thickEnd\t$itemRgb\t$blockCount\t$blockSizes\t$blockStarts\t$end_cols[1
]\t$new_coord";
                push (@good_start_and_end, $new_end_line);
                #print OUT $new_end_line, "\n"; #uncomment to print out
file of isoforms with validated starts and ends
                last;
            }
        }
    }
}

my $good_start_end_number = scalar @good_start_and_end;

#close(OUT); #uncomment to print out file of isoforms with validated
starts and ends

```



```

open(INF, "<$valid_introns_file") or die "couldn't open file";

my @valid_intron;

while (my $line = <INF>) {
    chomp($line);
    next if ($line =~ /^track/); #skips the track definition line
    push (@valid_intron, $line); #creates an array of valid splice
    junctions
}

close(INF);

open(OUT, ">$test_file.validated_unrefined.bed.temp");

print "Checking splice junctions...\n";

my $start;
my $end;
my @intron_start;
my @intron_end;
my @intron_coord_pair;
my @good_intron_counter;

foreach my $good_start_and_end (@good_start_and_end) { #starts with the
    array of SMRT transcripts with validated 5' and 3' ends
    my @cols = split("\t", $good_start_and_end);
    my $intron_strand = $cols[5];
    my $intron_number = $cols[9] - 1;
    if ($intron_number == 0) { #if a SMRT transcript has validated 5'
    and 3' ends and no introns, it is fully validated
        print OUT $good_start_and_end, "\n";
    }
    else {
        my @block_sizes = split(",", $cols[10]);
        my @block_starts = split(",", $cols[11]);
        for (my $i = 0; $i < $intron_number; $i = $i + 1) { #for the
        transcript currently in the "while" loop, creates an array of intron
        start sites relative to the genome
            $start = $cols[1] + $block_sizes[$i] + $block_starts[$i];
            push(@intron_start, $start);
        }
        for (my $i2 = 1; $i2 < $cols[9]; $i2 = $i2 + 1) { #for the
        transcript currently in the "while" loop, creates an array of intron
        end sites relative to the genome
            $end = $cols[1] + $block_starts[$i2];
            push(@intron_end, $end);
        }
        for (my $i3 = 0; $i3 < $intron_number; $i3 = $i3 + 1) { #for the
        transcript currently in the "while" loop, matches up intron start and
        end sites to create an array of complete intron coordinates relative to
        the genome
            my $intron_coords = "$intron_start[$i3]:$intron_end[$i3]";
            push (@intron_coord_pair, $intron_coords);
        }
    }
}

```

```

    }
    @intron_start = ();
    @intron_end = (); #intron starts and ends have been assigned to
the @intron_coords array; empty them for the next transcript
    foreach my $intron_coord_pair (@intron_coord_pair) { #goes
through each intron in the SMRT transcript
        my @coords = split(":", $intron_coord_pair); #allows
extraction of the start and end coordinates from each intron in the
SMRT transcript
        foreach my $valid_intron (@valid_intron) { #goes through
each intron in the array of validated introns
            my @valid_coords = split("\t", $valid_intron);
#allows extraction of the start and end coordinates from each validated
intron
            next if $cols[0] ne $valid_coords[0]; #enforces
chromosome matching
            next if $intron_strand ne $valid_coords[5]; #enforces
strand matching
            if (($coords[0] == $valid_coords[1]) and ($coords[1]
== $valid_coords[2])) {
                push(@good_intron_counter, $intron_coord_pair);
                #puts introns that are validated for this transcript into an array
(this really just functions as a counter)
            }
        }
    }
    @intron_coord_pair = (); #once each intron in the SMRT transcript
has been examined, empty the array for the next transcript
    if (@good_intron_counter == $intron_number) { #check to see if
all of the introns in the transcript are validated
        print OUT $good_start_and_end, "\n";
    }
    @good_intron_counter = (); #after checking to see if all the
introns in the transcript are validated, empties this array for the
next transcript
}
}

close(OUT);

#system("sort -k 2,2n -k 3,3n \Q$test_file\E.valid_start.bed.temp >
\Q$test_file\E.valid_start.bed"); #uncomment to print out file of
isoforms with validated starts
#system("rm \Q$test_file\E.valid_start.bed.temp"); #uncomment to print
out file of isoforms with validated starts

#system("sort -k 2,2n -k 3,3n
\Q$test_file\E.valid_start_and_end.bed.temp >
\Q$test_file\E.valid_start_and_end.bed"); #uncomment to print out file
of isoforms with validated starts and ends
#system("rm \Q$test_file\E.valid_start_and_end.bed.temp"); #uncomment
to print out file of isoforms with validated starts and ends

system("sort -k2,2n -k3,3n \Q$test_file\E.validated_unrefined.bed.temp

```

```

> \Q$test_file\E.validated_unrefined.bed");
system("rm \Q$test_file\E.validated_unrefined.bed.temp");

open(INF, "<$test_file.validated_unrefined.bed");
open(OUT, ">$test_file.validated_refined.temp");

my $new_block_size;
my @exon_start;
my @exon_end;
my @exon_coord_pair;
my $validated_count = 0;

while (my $line = <INF>) {
    chomp($line);
    my @cols = split("\t", $line);
    my $exon_number = $cols[9];
    if ($exon_number == 1) { #if a SMRT transcript has validated 5' and
        3' ends and no introns, it is fully validated. Just adjust the start
        and end to the consensus sites and fix the BlockSize accordingly
        $new_block_size = $cols[13]-$cols[12];
        print OUT
"$cols[0]\t$cols[12]\t$cols[13]\t$cols[3]\t$cols[4]\t$cols[5]\t$cols[12]
\t$cols[13]\t$cols[8]\t$cols[9]\t$new_block_size\t$cols[11]\n";
        $validated_count++;
    }
    else { #need to adjust the start and end sites and also the
        blockStarts and blockSizes
        my @block_sizes = split(",", $cols[10]);
        my @block_starts = split(",", $cols[11]);
        for (my $i = 0; $i < $exon_number; $i = $i + 1) { #for the
            transcript currently in the "while" loop, creates an array of exon
            start sites relative to the genome
            $start = $cols[1] + $block_starts[$i];
            push(@exon_start, $start);
        }
        for (my $i2 = 0; $i2 < $exon_number; $i2 = $i2 + 1) { #for the
            transcript currently in the "while" loop, creates an array of intron
            end sites relative to the genome
            $end = $cols[1] + $block_starts[$i2] + $block_sizes[$i2];
            push(@exon_end, $end);
        }
        shift(@exon_start); #removes the first exon start
        unshift(@exon_start, $cols[12]); #replaces the first exon start
        with the adjust chrStart value
        pop(@exon_end); #removes the last exon end
        push(@exon_end, $cols[13]); #replaces the last exon end with
        the adjusted chrEnd value
        for (my $i3 = 0; $i3 < $exon_number; $i3 = $i3 + 1) { #for the
            transcript currently in the "while" loop, matches up intron start and
            end sites to create an array of complete intron coordinates relative to
            the genome
            my $exon_coords = "$exon_start[$i3]:$exon_end[$i3]";
            push (@exon_coord_pair, $exon_coords);
        }
        @exon_start = ();
    }
}

```

```

        @exon_end = (); #intron starts and ends have been assigned to the
        @intron_coords array; empty them for the next transcript
        my @new_starts;
        my @new_sizes;
        #print $cols[3], "\t", @exon_coord_pair, "\n";
        foreach my $exon_coord_pair (@exon_coord_pair) { #goes through
        each exon in the SMRT transcript
            my @coords = split(":", $exon_coord_pair);
            my $blockStart = $coords[0] - $cols[12];
            my $blockSize = $coords[1] - $coords[0];
            push (@new_starts, $blockStart);
            push (@new_sizes, $blockSize);
        }
        @exon_coord_pair = ();
        shift(@new_starts); #removes the first value of the new_starts
        array, so we can replace it with 0 (it won't be 0 already if chrStart
        has been updated)
        my $assembled_starts = join(",", 0, @new_starts);
        my $assembled_sizes = join(",", @new_sizes);
        print OUT
        "$cols[0]\t$cols[12]\t$cols[13]\t$cols[3]\t$cols[4]\t$cols[5]\t$cols[12]
        ]\t$cols[13]\t$cols[8]\t$cols[9]\t$assembled_sizes\t$assembled_starts\n
        ";
        $validated_count++;
    }
}

```

#then need to add code to collapse the transcripts with identical structure into a single feature

```

close(INF);
close(OUT);

system("sort -k 2,2n -k 3,3n -k11,11 -k12,12 -k5,5n
\Q$test_file\E.validated_refined.temp >
\Q$test_file\E.validated_refined.bed");
system("rm \Q$test_file\E.validated_refined.temp");

#Collapsing matching transcripts into single isoforms

open(INF, "<$test_file.validated_refined.bed");
open(OUT, ">$test_file.isoforms.bed");

print "Collapsing matching transcripts into isoforms...\n";

my $prev_chr_plus = "start";
my $prev_chrStart_plus = 0;
my $prev_chrEnd_plus = 0;
my $prev_name_plus;
my $count_plus = 0;
my $prev_rgb_plus;
my $prev_blocks_plus;
my $prev_blockSizes_plus = "1,1";
my $prev_blockStarts_plus = "0,0";
my $prev_chr_minus = "start";

```

```

my $prev_chrStart_minus = 0;
my $prev_chrEnd_minus = 0;
my $prev_name_minus;
my $count_minus = 0;
my $prev_rgb_minus;
my $prev_blocks_minus;
my $prev_blockSizes_minus = "1,1";
my $prev_blockStarts_minus = "0,0";
my $iso_count = 0;

while(my $line = <INF>) {
    chomp($line);
    my @cols = split("\t", $line);
    if ($cols[5] eq "+") {
        if (($cols[0] eq $prev_chr_plus) and ($cols[1] ==
$prev_chrStart_plus) and ($cols[2] == $prev_chrEnd_plus) and ($cols[10]
eq $prev_blockSizes_plus) and ($cols[11] eq $prev_blockStarts_plus)) {
            $count_plus = $count_plus + $cols[4];
            $prev_chr_plus = $cols[0];
            $prev_name_plus = $cols[3];
            $prev_rgb_plus = $cols[8];
            $prev_blocks_plus = $cols[9];
        }
        else {
            if ($count_plus == 0) {
                $prev_chrStart_plus = $cols[1];
                $prev_chrEnd_plus = $cols[2];
                $prev_blockSizes_plus = $cols[10];
                $prev_blockStarts_plus = $cols[11];
                $count_plus = $cols[4];
                $prev_chr_plus = $cols[0];
                $prev_name_plus = $cols[3];
                $prev_rgb_plus = $cols[8];
                $prev_blocks_plus = $cols[9];
            }
            else {
                print OUT
"$prev_chr_plus\t$prev_chrStart_plus\t$prev_chrEnd_plus\t$prev_name_plu
s\t$count_plus\t+\t$prev_chrStart_plus\t$prev_chrEnd_plus\t$prev_rgb_p
lus\t$prev_blocks_plus\t$prev_blockSizes_plus\t$prev_blockStarts_plus\n
";
                $iso_count++;
                $prev_chrStart_plus = $cols[1];
                $prev_chrEnd_plus = $cols[2];
                $prev_blockSizes_plus = $cols[10];
                $prev_blockStarts_plus = $cols[11];
                $count_plus = $cols[4];
                $prev_chr_plus = $cols[0];
                $prev_name_plus = $cols[3];
                $prev_rgb_plus = $cols[8];
                $prev_blocks_plus = $cols[9];
            }
        }
    }
}

```

```

        elif ($cols[5] eq "-") {
            if (($cols[0] eq $prev_chr_minus) and ($cols[1] ==
$prev_chrStart_minus) and ($cols[2] == $prev_chrEnd_minus) and
($cols[10] eq $prev_blockSizes_minus) and ($cols[11] eq
$prev_blockStarts_minus)) {
                $count_minus = $count_minus + $cols[4];
                $prev_chr_minus = $cols[0];
                $prev_name_minus = $cols[3];
                $prev_rgb_minus = $cols[8];
                $prev_blocks_minus = $cols[9];
            }
            else {
                if ($count_minus == 0) {
                    $prev_chrStart_minus = $cols[1];
                    $prev_chrEnd_minus = $cols[2];
                    $prev_blockSizes_minus = $cols[10];
                    $prev_blockStarts_minus = $cols[11];
                    $count_minus = $cols[4];
                    $prev_chr_minus = $cols[0];
                    $prev_name_minus = $cols[3];
                    $prev_rgb_minus = $cols[8];
                    $prev_blocks_minus = $cols[9];
                }
                else {
                    print OUT
"$prev_chr_minus\t$prev_chrStart_minus\t$prev_chrEnd_minus\t$prev_name_
minus\t$count_minus\t\t-
\t$prev_chrStart_minus\t$prev_chrEnd_minus\t$prev_rgb_minus\t$prev_bloc
ks_minus\t$prev_blockSizes_minus\t$prev_blockStarts_minus\n";
                    $iso_count++;
                    $prev_chrStart_minus = $cols[1];
                    $prev_chrEnd_minus = $cols[2];
                    $prev_blockSizes_minus = $cols[10];
                    $prev_blockStarts_minus = $cols[11];
                    $count_minus = $cols[4];
                    $prev_chr_minus = $cols[0];
                    $prev_name_minus = $cols[3];
                    $prev_rgb_minus = $cols[8];
                    $prev_blocks_minus = $cols[9];
                }
            }
        }
    }
}

if ($count_plus > 0) {#prints out the last feature (plus strand)
    print OUT
"$prev_chr_plus\t$prev_chrStart_plus\t$prev_chrEnd_plus\t$prev_name_plu
s\t$count_plus\t\t+\t$prev_chrStart_plus\t$prev_chrEnd_plus\t$prev_rgb_p
lus\t$prev_blocks_plus\t$prev_blockSizes_plus\t$prev_blockStarts_plus\n
";
    $iso_count++;
}

if ($count_minus > 0) {#prints out the last feature (minus strand)
    print OUT

```

```

"$prev_chr_minus\t$prev_chrStart_minus\t$prev_chrEnd_minus\t$prev_name_
minus\t$count_minus\t\
\t$prev_chrStart_minus\t$prev_chrEnd_minus\t$prev_rgb_minus\t$prev_bloc
ks_minus\t$prev_blockSizes_minus\t$prev_blockStarts_minus\n";
    $iso_count++;
}

close(INF);
close(OUT);

my @ann;

open(INF, "<$ann_file") or die "couldn't open file";
while (my $line = <INF>) {
    chomp($line);
    next if ($line =~ /^track/); #skips the track definition line
    push (@ann, $line); #puts each line of the annotation file into an
array to be checked later
}
close(INF);

open(INF, "<$test_file.isoforms.bed") or die "couldn't open file";
open(OUT, ">$test_file.validated_transcripts.bed");

print "Checking for annotated isoforms...\n";

my $upper_limit_s;
my $lower_limit_s;
my $upper_limit_e;
my $lower_limit_e;
my $ann_count = 0;

print OUT "track type=bed name=\"$test_file.validated_transcripts.bed\"
description=\"validated transcript structures from
transcript_structure_validator.pl\"\n";

while (my $line = <INF>) {
    chomp($line);
    next if ($line =~ /^track/); #skips the track definition line
    my @val_cols = split("\t", $line);
    my $found_flag=0;
    foreach my $ann (@ann) {
        my $val_introns = 0;
        my $ann_introns = 0;
        my @ann_cols = split("\t", $ann);
        next if ($val_cols[5] ne $ann_cols[5]);
        next if ($val_cols[9] ne $ann_cols[9]);
        if ($val_cols[5] eq "+") {
            $upper_limit_s = $ann_cols[1] + $start_dist;
            $lower_limit_s = $ann_cols[1] - $start_dist;
            $upper_limit_e = $ann_cols[2] + $end_dist;
            $lower_limit_e = $ann_cols[2] - $end_dist;
        }
        if ($val_cols[5] eq "-") {
            $upper_limit_s = $ann_cols[1] + $end_dist;

```

```

        $lower_limit_s = $ann_cols[1] - $end_dist;
        $upper_limit_e = $ann_cols[2] + $start_dist;
        $lower_limit_e = $ann_cols[2] - $start_dist;
    }
    if (($val_cols[1] >= $lower_limit_s) and ($val_cols[1] <=
$upper_limit_s)) {
        if (($val_cols[2] >= $lower_limit_e) and ($val_cols[2] <=
$upper_limit_e)) {
            if ($val_cols[9] == 1) {
                print OUT $val_cols[0], "\t", $val_cols[1], "\t",
$val_cols[2], "\t", $ann_cols[3], "_", $val_cols[3], "\t",
$val_cols[4], "\t", $val_cols[5], "\t", $val_cols[6], "\t",
$val_cols[7], "\t", $ann_cols[8], "\t", $val_cols[9], "\t",
$val_cols[10], "\t", $val_cols[11], "\n";
                $ann_count++;
                $found_flag=1;
            }
            else {
                my $val_intron_number = $val_cols[9] - 1;
                my @val_block_sizes = split(",", $val_cols[10]);
                my @val_block_starts = split(",", $val_cols[11]);
                for (my $i = 0; $i < $val_intron_number; $i = $i +
1) { #for the transcript currently in the "while" loop, creates an
array of intron start sites relative to the genome
                    $start = $val_cols[1] + $val_block_sizes[$i] +
$val_block_starts[$i];
                    $val_introns = "$val_introns:$start";
                }
                for (my $i2 = 1; $i2 < $val_cols[9]; $i2 = $i2 + 1)
{ #for the transcript currently in the "while" loop, creates an array
of intron end sites relative to the genome
                    $end = $val_cols[1] + $val_block_starts[$i2];
                    $val_introns = "$val_introns:$end";
                }
                my $ann_intron_number = $ann_cols[9] - 1;
                my @ann_block_sizes = split(",", $ann_cols[10]);
                my @ann_block_starts = split(",", $ann_cols[11]);
                for (my $i = 0; $i < $ann_intron_number; $i = $i +
1) { #for the annotation currently in the "foreach" loop, creates an
array of intron start sites relative to the genome
                    $start = $ann_cols[1] + $ann_block_sizes[$i] +
$ann_block_starts[$i];
                    $ann_introns = "$ann_introns:$start";
                }
                for (my $i2 = 1; $i2 < $ann_cols[9]; $i2 = $i2 + 1)
{ #for the annotation currently in the "foreach" loop, creates an array
of intron end sites relative to the genome
                    $end = $ann_cols[1] + $ann_block_starts[$i2];
                    $ann_introns = "$ann_introns:$end";
                }
                if ($val_introns eq $ann_introns) {
                    print OUT $val_cols[0], "\t", $val_cols[1],
"\t", $val_cols[2], "\t", $ann_cols[3], "_", $val_cols[3], "\t",
$val_cols[4], "\t", $val_cols[5], "\t", $val_cols[6], "\t",
$val_cols[7], "\t", $ann_cols[8], "\t", $val_cols[9], "\t",

```



```

$val_cols[10], "\t", $val_cols[11], "\n";
                                $ann_count++;
                                $found_flag=1;
                                }
                            }
                        }
                    }
                }
            }
        if ($found_flag == 0){
            print OUT $line, "\n";
        }
    }

    close(INF);
    close(OUT);

    open(OUT, ">validated_isoforms_stats.txt");

    my $novel_count = $iso_count - $ann_count;
    print OUT "$iso_count validated transcripts\n\t$novel_count
    novel\n\t$ann_count annotated\n";

    close(OUT);

    print "-----
    \n$good_start_number sequences have validated start sites.\n";
    print "$good_start_end_number sequences have validated start and end
    sites.\n";
    print "$validated_count fully validated sequences collapse into
    $iso_count distinct isoforms.\n";
    print "$ann_count isoforms match annotated transcripts.\n";

    system("rm \Q$test_file\E.validated_refined.bed");
    system("rm \Q$test_file\E.validated_unrefined.bed");
    system("rm \Q$test_file\E.isoforms.bed");
    system("rm \Q$test_file\E.bed");

```

## **APPENDIX 5**

TRIMD\_README.txt

## TRIMD

Transcriptome Resolution by Integration of Multi-platform Data

Scripts included:

TRIMD\_start\_finder.pl  
 TRIMD\_junction\_matcher.pl  
 TRIMD\_end\_finder.pl  
 TRIMD\_isoform\_validator.pl

Notes for all scripts:

Defaults are set using the Epstein-Barr virus Akata strain as a model.

Annotation file should contain only features for polyadenylated transcripts.

Fasta files of Iso-Seq data must have names formatted as  
 putative\_isoform\_id/number\_of\_SMRT\_reads/length, as from the Iso-Seq pipeline.

TRIMD\_start\_finder.pl

USAGE: perl /PATH/TRIMD\_start\_finder.pl </PATH/SMRT\_sam\_file> </PATH/CAGE\_file>  
 </PATH/Annotation\_bed\_file>

Accepts a SAM file of Iso-Seq fl data, a SAM file of CAGE data, and a bed file of annotated polyadenylated transcripts. Counts the number of non-clipped SMRT reads with 5' starts at each genomic position and estimates consensus locations of clusters of 5' starts. Uses Paraclu to identify clusters of 5' starts in the CAGE data. Output includes BEDGRAPH files of all 5' starts, BED files of the weighted centers of start clusters and a BED file of Iso-Seq 5' starts supported by either the annotation or the CAGE data.

Paraclu was written by Martin C Frith 2006, Genome Exploration Research Group, RIKEN GSC and Institute for Molecular Bioscience, University of Queensland and distributed under the GNU General Public License.

## INPUT

1. SAM file of Iso-Seq fl isoforms: this script was developed using data aligned with GMAP (-f samse option). Other aligners may also be appropriate.
2. SAM file of CAGE reads: this script was developed using data aligned with STAR. Other aligners may also be appropriate.
3. BED file of annotated polyadenylated transcripts: the annotation file MUST be sorted by chrStart, then chrEnd. If your annotation file contains non-polyadenylated transcripts or other features (e.g. repeat regions, promoters) these should be removed to avoid false positives.

## PARAMETERS

-----

1. (Viral) chromosome name: the name of the chromosome under investigation. This must match between both SAM files (field 3) and the BED annotation file (field 1). Does not have to be viral, but for organisms with multiple chromosomes only one chromosome can be examined at a time.  
No default: must be entered at prompt
2. Window for collapsing Iso-Seq 5' starts: Iso-Seq 5' starts within this number of bases of each other will be considered to represent the same transcription start site. The consensus transcription start site is determined by calculating an average of the coordinates in the cluster, weighted by read depth at each coordinate.  
Default: 8
3. Minimum tags per CAGE cluster: the minimum number of CAGE tags in a cluster to be considered a potential transcription start site.  
Default: 15
4. Minimum relative density for CAGE clusters: a measure of change in tag density between the cluster and its surroundings. Higher number = bigger change. For more information, see the Paraclu paper referenced  
Default: 2
5. Minimum CAGE cluster length: the minimum cluster length, in base pairs, to be considered a potential transcription start site.  
Default: 1
6. Maximum CAGE cluster length: the maximum cluster length, in base pairs, to be considered a potential transcription start site.  
Default: 20
7. Maximum allowable distance between Iso-Seq and CAGE 5' starts: Maximum distance of a CAGE consensus start site from an Iso-Seq consensus start site to consider the start site validated.  
Default: 3
8. Minimum number of SMRT reads to report a 5' start: the minimum number of Iso-Seq 5' starts in a cluster to be considered a potential transcription start site  
Default: 1
9. Maximum distance in bp from an annotated start site to be called as "annotated"  
Default: 10

## OUTPUT

-----

1. A BED file of validated 5' starts: this file is in bedDetail format, using the first 6 standard bed fields and an additional field that is necessary for the TRIMD\_isoform\_validator.pl script. The coordinates are those of the Iso-Seq consensus 5' start. The name field is in the format [nov|ann]\_[+|-]\_123.IsoSeq\_456.CAGE and indicates whether the start site is novel or annotated, strand, the number of SMRT reads supporting it and the number of CAGE tags supporting it. The score is the number of SMRT reads and CAGE tags added together. The additional field indicates the range of the SMRT start cluster and number of supporting SMRT reads. Note that the validated starts file includes Iso-Seq starts that are supported by CAGE and/or annotation data. You may wish to filter the results to contain only starts that are supported by CAGE.
2. A text file with the number of total, novel and annotated 5' start sites validated, and a record of the input files.
3. BED file of Iso-Seq consensus 5' starts
4. BEDGRAPH file of all nonclipped Iso-Seq 5' starts

5. BED file of CAGE consensus 5' starts
6. BEDGRAPH file of all nonclipped CAGE 5' starts

=====

TRIMD\_junction\_matcher.pl

-----

USAGE: perl /PATH/TRIMD\_junction\_matcher.pl </PATH/SMRT\_introns\_file>  
 </PATH/Illumina\_SJ.out.tab\_file> </PATH/transcript\_annotation\_bed\_file>  
 <coordinates\_to\_ignore\_bed\_file(optional)>

Accepts a junctions files from GMAP/SMRT (generated with the -f introns argument) and an SJ.out.tab files from STAR/Illumina.

Returns 3 bed files: one of SMRT splice junctions, one of Illumina splice junctions and one of junctions detected by both methods. The coordinates in the output bed files correspond to the first and last bases of the introns. Optionally, splice junctions in repeat regions can be ignored by providing a bed file with the coordinates of those junctions.

## INPUT

-----

1. A file of Iso-Seq splice junctions data generated by GMAP (-f introns argument)
2. A file of Illumina splice junctions data generated by STAR (SJ.out.tab file in default STAR output).
3. BED file of annotated polyadenylated transcripts
4. (optional) BED file of genomic regions to ignore (e.g. repeat regions)

## PARAMETERS

-----

1. (Viral) chromosome name: the name of the chromosome under investigation. This must match between both junction files and the BED annotation file (field 1). Does not have to be viral, but for organisms with multiple chromosomes only one chromosome can be examined at a time.  
No default: must be entered at prompt

2. Minimum SMRT read depth to report a splice junction

3. Minimum Illumina RNA-seq read depth to report a splice junction

## OUTPUT

-----

1. BED file of validated splice junctions: this file is in BED format, using the first 6 standard bed fields. The coordinates correspond to the first and last base of the excised intron. The name field is in the format [nov|ann]\_123.IsoSeq\_456.CAGE and indicates whether the junction is novel or annotated, the number of SMRT reads supporting it and the number of Illumina reads supporting it. The score is the number of SMRT reads and Illumina reads added together.

2. A text file with the number of total, novel and annotated splice junctions validated, and a record of the input files.

3. BED file of splice junctions detected on the specified chromosome in the Iso-Seq data
4. BED file of splice junctions detected on the specified chromosome in the Illumina data

=====

TRIMD\_end\_finder.pl

-----

USAGE: perl /PATH/TRIMD\_end\_finder.pl </PATH/SMRT\_sam\_file> </PATH/Illumina\_sam\_file>  
</PATH/Annotation\_bed\_file>

Accepts a SAM file using Iso-Seq fl data, a SAM file using Illumina data, and a BED file of annotated polyadenylated transcripts. Counts the number of non-clipped SMRT reads with 3' ends at each genomic position and estimates consensus locations of clusters of 3' ends. Extracts Illumina reads containing apparent poly(A) tails and estimates consensus locations of clusters of polyadenylation sites. Output includes BEDGRAPH files of all 3' ends, BED files of the weighted centers of end clusters, a sam file of reads with polyA tails and a BED file of Iso-Seq 3' ends supported by either the annotation or the Illumina data.

SMRT fl read names must be formatted as putative\_isoform\_id/number\_of\_SMRT\_reads/length.

## INPUT

-----

1. SAM file of Iso-Seq fl isoforms: this script was developed using data aligned with GMAP (-f samse option). Other aligners may also be appropriate.
2. SAM file of Illumina RNA-seq data: Illumina libraries should have been prepared with stranded TruSeq or a similar protocol. Sequence data can be paired-end or single-end. This script was developed using data aligned with STAR. Other aligners may also be appropriate.
3. BED file of annotated polyadenylated transcripts: the annotation file MUST be sorted by chrStart, then chrEnd. If your annotation file contains non-polyadenylated transcripts or other features (e.g. repeat regions, promoters) these should be removed to avoid false positives.

## PARAMETERS

-----

1. (Viral) chromosome name: the name of the chromosome under investigation. This must match between both SAM files (field 3) and the BED annotation file (field 1). Does not have to be viral, but for organisms with multiple chromosomes only one chromosome can be examined at a time.  
No default: must be entered at prompt
2. Window for collapsing Iso-Seq 3' ends: Iso-Seq 3' ends within this number of bases of each other will be considered to represent the same polyadenylation site. The consensus Iso-Seq polyadenylation site is determined by calculating an average of the coordinates in the cluster, weighted by read depth at each coordinate.  
Default: 8
3. Minimum number of As for Illumina poly(A) tails: the number of As (or Ts, as appropriate) required in a read to indicate the presence of a poly(A) tail.  
Default: 5
4. Minimum number of mismatches for Illumina poly(A) tails: the number of terminal mismatches in a read relative to the genome sequence to indicate the presence of a poly(A) tail.  
Default: 2

5. Window for collapsing Illumina 3' ends: Illumina reads containing poly(A) tails within this number of bases of each other will be considered to represent the same polyadenylation site. The consensus Illumina polyadenylation site is determined by calculating an average of the coordinates in the cluster, weighted by read depth at each coordinate.

Default: 8

6. Number of bases downstream of Iso-Seq consensus 3' ends to look for Illumina support

Default: 10

7. Number of bases upstream of Iso-Seq consensus 3' ends to look for Illumina support

Default: 4

8. Minimum number of SMRT reads to report a 3' end

Default: 5

9. Minimum number of Illumina poly(A) tail reads to report a 3' end

Default: 1

10. Maximum distance in bp from an annotated end to be called as "annotated"

Default: 25

## OUTPUT

-----

1. BED file of validated 3' ends

This file is in bedDetail format, using the first 6 standard bed fields and an additional field that is necessary for the TRIMD\_isoform\_validator.pl script. The coordinates are those of the Illumina consensus 3' end. The name field is in the format [nov|ann]\_[+|-]\_123.IsoSeq\_456.CAGE and indicates whether the end is novel or annotated, strand, the number of SMRT reads supporting it and the number of Illumina poly(A) reads supporting it. The score is the number of SMRT reads and Illumina poly(A) reads added together. The additional field indicates the range of the SMRT end cluster and number of supporting SMRT reads. Note that the validated starts file includes SMRT ends that are supported by Illumina and/or annotation data. You may wish to filter the results to contain only starts that are supported by Illumina.

2. A text file with the number of total, novel and annotated 3' end sites validated, and a record of the input files.

3. BED file of Iso-Seq consensus 3' ends

4. BEDGRAPH file of nonclipped Iso-Seq 3' ends

5. BED file of Illumina consensus 3' ends

6. BEDGRAPH file of nonclipped Illumina 3' ends

7. SAM file of Illumina reads containing putative poly(A) tails

=====

TRIMD\_isoform\_validator.pl

-----

USAGE: perl /PATH/TRIMD\_isoform\_validator.pl </PATH/SMRT\_sam\_file>  
 </PATH/validated\_starts\_file> </PATH/validated\_ends\_file> </PATH/validated\_introns\_file>  
 </PATH/Annotation\_bed\_file>

Takes a SAM file of Iso-Seq fl isoforms and compares them to a list of validated 5' ends, 3' ends and introns to create a list of validated isoform structures, which are compared to an annotation file.

## INPUT

-----

1. SAM file of Iso-Seq fl isoforms
2. BedDetail file of validated starts: the output file from TRIMD\_start\_finder.pl that ends ".validated\_starts.bed"
3. BedDetail file of validated ends: the output file from TRIMD\_end\_finder.pl that ends ".validated\_ends.bed"
4. BedDetail file of validated splice junctions: the output file from TRIMD\_junction\_matcher.pl that ends ".validated\_introns.bed"
5. BED file of annotated polyadenylated transcripts: the annotation file MUST be sorted by chrStart, then chrEnd. If your annotation file contains non-polyadenylated transcripts or other features (e.g. repeat regions, promoters) these should be removed to avoid false positives.

Note that the chromosome names must match exactly between all of the input files.

## PARAMETERS

-----

1. Maximum distance from an annotated 5' start to be called annotated  
Default: 10
2. Maximum distance from an annotated 3' end to be called annotated  
Default: 10

Note that if there are overlapping transcripts, more than one may be called as the same "annotated" transcript, depending on how the distance parameters are set.

## OUTPUT

-----

1. BED file of validated isoform structures: this file is in bed format, using all 12 standard fields. The coordinate for the 5' start is taken from the Iso-Seq consensus 5' start site and the coordinate for the 3' end is taken from the Illumina poly(A) read consensus 3' end. The name is that of one of the Iso-Seq isoforms representing that transcript, prefixed by any matching annotated transcript. The score is the number of SMRT reads that support that transcript. ThickStart and thickEnd (fields 7 and 8) match chrStart and chrEnd (fields 2 and 3): no information about ORFs is inferred. If the transcript is called as annotated, the color (field 9) is imported from the annotation file.
2. A text file with the number of total, novel and annotated isoforms validated, and a record of the input files.

=====

## LICENSE

-----

Distributed under the GNU General Public License. For more, see License.txt

=====



## PLEASE CITE

-----

tba

and

Frith MC, Valen E, Krogh A, Hayashizaki Y, Carninci P, Sandelin A (2008) A code for transcription initiation in mammalian genomes" *Genome Research* 18(1):1-12.

=====

## CONTACT

-----

Erik Flemington (erik@tulane.edu)

Tina O'Grady (tmogradey@gmail.com)

## **APPENDIX 6**

Validated EBV transcription start sites

chrStart	chrEnd	ID	Read depth	Strand
599	600	nov_+_18.SMRT_369.CAGE	387	+
610	611	ann_-_43.SMRT_9828.CAGE	9871	-
1734	1735	nov_-_48.SMRT_765.CAGE	813	-
2567	2568	nov_-_11.SMRT_128.CAGE	139	-
3178	3179	nov_-_11.SMRT_1937.CAGE	1948	-
3193	3194	nov_-_21.SMRT_59.CAGE	80	-
3397	3398	ann_-_24.SMRT_3205.CAGE	3229	-
3500	3501	nov_-_1.SMRT_100.CAGE	101	-
3850	3851	nov_+_16.SMRT_320.CAGE	336	+
4383	4384	nov_+_19.SMRT_192.CAGE	211	+
4391	4392	ann_-_12.SMRT_231.CAGE	243	-
4450	4451	ann_-_3.SMRT_165.CAGE	168	-
5148	5149	nov_-_2.SMRT_154.CAGE	156	-
5174	5175	nov_-_1.SMRT_105.CAGE	106	-
6208	6209	ann_-_2550.SMRT_24906.CAGE	27456	-
7705	7706	ann_-_73.SMRT_1894.CAGE	1967	-
7765	7766	nov_+_51.SMRT_434.CAGE	485	+
8655	8656	nov_-_2.SMRT_2089.CAGE	2091	-
10381	10382	ann_-_1899.SMRT_20750.CAGE	22649	-
10648	10649	nov_-_5.SMRT_320.CAGE	325	-
10888	10889	nov_-_1.SMRT_178.CAGE	179	-
11441	11442	nov_-_18.SMRT_282.CAGE	300	-
11491	11492	nov_-_2.SMRT_224.CAGE	226	-
11606	11607	nov_+_1.SMRT_107.CAGE	108	+
11639	11640	nov_-_2.SMRT_59.CAGE	61	-
11753	11754	ann_-_431.SMRT_22227.CAGE	22658	-
12629	12630	nov_-_67.SMRT_41153.CAGE	41220	-
12659	12660	ann_-_340.SMRT_423.CAGE	763	-
13796	13797	nov_+_1.SMRT_96.CAGE	97	+
14503	14504	nov_-_17.SMRT_127.CAGE	144	-
14696	14697	nov_-_11.SMRT_113.CAGE	124	-
15102	15103	nov_-_36.SMRT_423.CAGE	459	-
15945	15946	nov_-_2.SMRT_70.CAGE	72	-
16557	16558	nov_+_3.SMRT_907.CAGE	910	+
16984	16985	ann_-_28.SMRT_15038.CAGE	15066	-
18317	18318	nov_+_6.SMRT_155.CAGE	161	+
19392	19393	nov_+_419.SMRT_762.CAGE	1181	+
20624	20625	nov_+_3.SMRT_212.CAGE	215	+

22563	22564	nov_-_858.SMRT_7056.CAGE	7914	-
22587	22588	ann_-_40.SMRT_970.CAGE	1010	-
22695	22696	nov_-_4.SMRT_211.CAGE	215	-
23149	23150	nov_-_10.SMRT_227.CAGE	237	-
23521	23522	nov_+_6.SMRT_275.CAGE	281	+
23919	23920	nov_+_151.SMRT_9868.CAGE	10019	+
24396	24397	nov_+_52.SMRT_113.CAGE	165	+
24424	24425	ann_-_5.SMRT_2217.CAGE	2222	-
24470	24471	nov_+_39.SMRT_349.CAGE	388	+
24642	24643	ann_+_12.SMRT_1546.CAGE	1558	+
25612	25613	nov_+_2.SMRT_62.CAGE	64	+
27060	27061	ann_+_45.SMRT_10982.CAGE	11027	+
27459	27460	nov_-_13.SMRT_224.CAGE	237	-
27960	27961	ann_+_80.SMRT_10972.CAGE	11052	+
28287	28288	nov_+_20.SMRT_74.CAGE	94	+
28508	28509	nov_+_4.SMRT_241.CAGE	245	+
28682	28683	nov_+_1.SMRT_698.CAGE	699	+
29574	29575	nov_-_5.SMRT_237.CAGE	242	-
29614	29615	nov_-_3.SMRT_265.CAGE	268	-
29845	29846	ann_-_555.SMRT_42730.CAGE	43285	-
29880	29881	ann_-_2.SMRT_141.CAGE	143	-
30286	30287	nov_-_21.SMRT_813.CAGE	834	-
30587	30588	nov_-_3.SMRT_72.CAGE	75	-
30679	30680	nov_-_41.SMRT_1350.CAGE	1391	-
30818	30819	nov_-_10.SMRT_282.CAGE	292	-
31049	31050	nov_-_29.SMRT_59.CAGE	88	-
32615	32616	nov_-_1.SMRT_1117.CAGE	1118	-
32701	32702	nov_-_1.SMRT_182.CAGE	183	-
32930	32931	nov_-_1.SMRT_307.CAGE	308	-
32957	32958	nov_-_1.SMRT_179.CAGE	180	-
33059	33060	nov_-_1.SMRT_260.CAGE	261	-
33097	33098	nov_-_1.SMRT_2115.CAGE	2116	-
33117	33118	nov_-_1.SMRT_138.CAGE	139	-
33204	33205	nov_-_1.SMRT_3133.CAGE	3134	-
33324	33325	nov_-_1.SMRT_174.CAGE	175	-
33569	33570	nov_-_1.SMRT_272.CAGE	273	-
34833	34834	nov_-_1.SMRT_2336.CAGE	2337	-
35009	35010	nov_-_26.SMRT_48315.CAGE	48341	-
36316	36317	nov_+_2.SMRT_830.CAGE	832	+

36715	36716	nov_+_1.SMRT_69.CAGE	70	+
39251	39252	nov_-_1.SMRT_59.CAGE	60	-
40390	40391	nov_+_3.SMRT_304.CAGE	307	+
41344	41345	nov_-_7.SMRT_275.CAGE	282	-
41597	41598	nov_-_41.SMRT_458.CAGE	499	-
41972	41973	ann_-_278.SMRT_258.CAGE	536	-
42093	42094	nov_-_37.SMRT_130.CAGE	167	-
42913	42914	nov_-_9.SMRT_510.CAGE	519	-
42960	42961	nov_-_95.SMRT_1776.CAGE	1871	-
43338	43339	nov_-_11.SMRT_2121.CAGE	2132	-
43362	43363	nov_-_4.SMRT_292.CAGE	296	-
43541	43542	nov_+_1.SMRT_128.CAGE	129	+
43592	43593	nov_+_7.SMRT_12372.CAGE	12379	+
48118	48119	nov_+_25.SMRT_365.CAGE	390	+
49310	49311	nov_-_46.SMRT_4758.CAGE	4804	-
49367	49368	nov_+_43.SMRT_187.CAGE	230	+
50466	50467	ann_-_1397.SMRT_11863.CAGE	13260	-
52764	52765	nov_-_13.SMRT_1011.CAGE	1024	-
53956	53957	nov_-_5.SMRT_158.CAGE	163	-
55122	55123	nov_+_5.SMRT_91.CAGE	96	+
55911	55912	ann_-_7.SMRT_8241.CAGE	8248	-
55967	55968	nov_+_32.SMRT_150.CAGE	182	+
56540	56541	ann_-_35.SMRT_1777.CAGE	1812	-
56600	56601	ann_+_47.SMRT_183.CAGE	230	+
56626	56627	ann_+_41.SMRT_1250.CAGE	1291	+
56870	56871	nov_-_26.SMRT_760.CAGE	786	-
58622	58623	ann_-_742.SMRT_36211.CAGE	36953	-
58730	58731	nov_-_22.SMRT_992.CAGE	1014	-
59835	59836	nov_-_8.SMRT_430.CAGE	438	-
59909	59910	nov_-_59.SMRT_96.CAGE	155	-
60067	60068	nov_-_12.SMRT_377.CAGE	389	-
60251	60252	nov_-_243.SMRT_8593.CAGE	8836	-
60600	60601	ann_-_2.SMRT_636.CAGE	638	-
63743	63744	nov_-_5.SMRT_173.CAGE	178	-
63852	63853	nov_+_2.SMRT_224.CAGE	226	+
63931	63932	nov_+_1.SMRT_133.CAGE	134	+
65094	65095	ann_+_47.SMRT_18054.CAGE	18101	+
71970	71971	nov_+_18.SMRT_166.CAGE	184	+
72926	72927	ann_+_27.SMRT_369.CAGE	396	+

74051	74052	nov_-_2.SMRT_61.CAGE	63	-
74831	74832	nov_-_15.SMRT_145.CAGE	160	-
75175	75176	nov_-_35.SMRT_192.CAGE	227	-
77648	77649	ann_+_2.SMRT_356.CAGE	358	+
80720	80721	nov_+_1.SMRT_356.CAGE	357	+
98915	98916	nov_+_1.SMRT_83.CAGE	84	+
100357	100358	nov_+_9.SMRT_214.CAGE	223	+
101661	101662	nov_-_1.SMRT_1113.CAGE	1114	-
101677	101678	nov_-_1.SMRT_128.CAGE	129	-
101764	101765	nov_-_1.SMRT_350.CAGE	351	-
101786	101787	nov_-_1.SMRT_1114.CAGE	1115	-
103581	103582	nov_-_1.SMRT_275.CAGE	276	-
103684	103685	nov_-_1.SMRT_123.CAGE	124	-
103765	103766	ann_-_126.SMRT_48592.CAGE	48718	-
104766	104767	ann_+_102.SMRT_42958.CAGE	43060	+
104920	104921	nov_+_36.SMRT_14276.CAGE	14312	+
105143	105144	nov_+_8.SMRT_379.CAGE	387	+
105294	105295	nov_-_1.SMRT_126.CAGE	127	-
105392	105393	nov_+_21.SMRT_997.CAGE	1018	+
105562	105563	nov_+_6.SMRT_87.CAGE	93	+
105598	105599	nov_+_7.SMRT_113.CAGE	120	+
105633	105634	nov_+_2.SMRT_66.CAGE	68	+
105654	105655	nov_+_4.SMRT_205.CAGE	209	+
105841	105842	nov_+_3.SMRT_244.CAGE	247	+
105936	105937	nov_+_9.SMRT_250.CAGE	259	+
106064	106065	nov_+_1.SMRT_165.CAGE	166	+
106348	106349	nov_+_4.SMRT_1786.CAGE	1790	+
108104	108105	nov_-_35.SMRT_652.CAGE	687	-
108606	108607	nov_-_22.SMRT_307.CAGE	329	-
108646	108647	nov_-_42.SMRT_106.CAGE	148	-
109041	109042	nov_-_2.SMRT_64.CAGE	66	-
109095	109096	nov_+_21.SMRT_10416.CAGE	10437	+
109184	109185	nov_-_1.SMRT_99.CAGE	100	-
109516	109517	ann_-_155.SMRT_2355.CAGE	2510	-
109554	109555	nov_+_1.SMRT_122.CAGE	123	+
109573	109574	nov_+_10.SMRT_5977.CAGE	5987	+
109727	109728	nov_-_17.SMRT_110.CAGE	127	-
109842	109843	ann_+_172.SMRT_18237.CAGE	18409	+
110486	110487	nov_+_5.SMRT_114.CAGE	119	+

112360	112361	nov_+_1813.SMRT_50392.CAGE	52205	+
113215	113216	nov_+_6.SMRT_6960.CAGE	6966	+
123054	123055	nov_-_3.SMRT_1507.CAGE	1510	-
125948	125949	ann_+_20.SMRT_16318.CAGE	16338	+
127102	127103	ann_+_336.SMRT_5770.CAGE	6106	+
128963	128964	nov_+_2.SMRT_88.CAGE	90	+
129735	129736	ann_+_74.SMRT_7531.CAGE	7605	+
130772	130773	ann_+_720.SMRT_6507.CAGE	7227	+
131714	131715	ann_+_92.SMRT_2103.CAGE	2195	+
131763	131764	nov_+_14.SMRT_1080.CAGE	1094	+
135227	135228	ann_-_28.SMRT_2944.CAGE	2972	-
135743	135744	nov_-_13.SMRT_99.CAGE	112	-
135865	135866	nov_-_3.SMRT_72.CAGE	75	-
136009	136010	nov_-_8.SMRT_61.CAGE	69	-
136283	136284	nov_-_58.SMRT_698.CAGE	756	-
136346	136347	nov_+_14.SMRT_297.CAGE	311	+
137815	137816	ann_+_34.SMRT_2374.CAGE	2408	+
138538	138539	nov_+_14.SMRT_77.CAGE	91	+
138807	138808	nov_-_1.SMRT_258.CAGE	259	-
139385	139386	ann_-_70.SMRT_3140.CAGE	3210	-
139443	139444	ann_+_593.SMRT_10469.CAGE	11062	+
139716	139717	nov_+_93.SMRT_498.CAGE	591	+
139796	139797	ann_+_162.SMRT_53017.CAGE	53179	+
140922	140923	ann_-_20.SMRT_2943.CAGE	2963	-
141133	141134	nov_-_1.SMRT_56.CAGE	57	-
141590	141591	nov_+_2.SMRT_51.CAGE	53	+
141643	141644	nov_-_3.SMRT_95.CAGE	98	-
141674	141675	nov_-_2.SMRT_81.CAGE	83	-
141799	141800	nov_+_1.SMRT_422.CAGE	423	+
142237	142238	nov_-_6.SMRT_62.CAGE	68	-
142908	142909	ann_-_1301.SMRT_39016.CAGE	40317	-
143363	143364	nov_-_12.SMRT_131.CAGE	143	-
143437	143438	nov_-_5.SMRT_323.CAGE	328	-
143867	143868	nov_+_2.SMRT_60.CAGE	62	+
144154	144155	nov_-_4.SMRT_535.CAGE	539	-
144255	144256	nov_+_58.SMRT_725.CAGE	783	+
145374	145375	nov_+_2.SMRT_101.CAGE	103	+
146927	146928	nov_-_4.SMRT_365.CAGE	369	-
147032	147033	nov_-_97.SMRT_1094.CAGE	1191	-

150164	150165	nov_+_2.SMRT_107.CAGE	109	+
152491	152492	nov_-_1.SMRT_857.CAGE	858	-
152867	152868	ann_-_203.SMRT_59254.CAGE	59457	-
153639	153640	nov_-_2.SMRT_305.CAGE	307	-
153968	153969	ann_-_7.SMRT_265.CAGE	272	-
154024	154025	nov_-_9.SMRT_297.CAGE	306	-
154100	154101	nov_-_1.SMRT_390.CAGE	391	-
154144	154145	nov_-_14.SMRT_522.CAGE	536	-
155814	155815	ann_+_43.SMRT_1311.CAGE	1354	+
156011	156012	nov_+_28.SMRT_1262.CAGE	1290	+
156860	156861	nov_+_11.SMRT_265.CAGE	276	+
156949	156950	ann_-_2.SMRT_287.CAGE	289	-
157036	157037	nov_-_5.SMRT_130.CAGE	135	-
157040	157041	ann_+_2078.SMRT_16853.CAGE	18931	+
157697	157698	nov_-_4.SMRT_315.CAGE	319	-
159860	159861	nov_+_1.SMRT_56.CAGE	57	+
160213	160214	nov_+_201.SMRT_13429.CAGE	13630	+
160263	160264	nov_-_2.SMRT_411.CAGE	413	-
160298	160299	nov_+_41.SMRT_50.CAGE	91	+
160572	160573	ann_+_297.SMRT_5954.CAGE	6251	+
160811	160812	nov_+_54.SMRT_2042.CAGE	2096	+
161000	161001	ann_+_3.SMRT_729.CAGE	732	+
161265	161266	nov_+_3503.SMRT_25085.CAGE	28588	+
161561	161562	nov_+_173.SMRT_4140.CAGE	4313	+
161726	161727	nov_+_4.SMRT_252.CAGE	256	+
161762	161763	ann_+_6.SMRT_167.CAGE	173	+
161795	161796	ann_+_6.SMRT_148.CAGE	154	+
161835	161836	nov_+_1.SMRT_341.CAGE	342	+
163056	163057	nov_-_3.SMRT_82.CAGE	85	-
163079	163080	nov_-_24.SMRT_2100.CAGE	2124	-
163333	163334	nov_-_7.SMRT_104.CAGE	111	-
164547	164548	nov_+_17.SMRT_5300.CAGE	5317	+
165000	165001	nov_-_7.SMRT_901.CAGE	908	-
165920	165921	nov_+_21.SMRT_232.CAGE	253	+
166431	166432	nov_+_22.SMRT_4923.CAGE	4945	+
166520	166521	nov_-_1.SMRT_123.CAGE	124	-
169341	169342	nov_-_35.SMRT_828.CAGE	863	-
169395	169396	nov_+_8.SMRT_62.CAGE	70	+
169527	169528	nov_-_7.SMRT_181.CAGE	188	-



169648	169649	ann+_1852.SMRT_7176.CAGE	9028	+
169676	169677	ann_-_26.SMRT_2365.CAGE	2391	-
169713	169714	nov_-_5.SMRT_149.CAGE	154	-
169766	169767	nov+_1959.SMRT_15063.CAGE	17022	+
171265	171266	nov+_3.SMRT_155.CAGE	158	+

## **APPENDIX 7**

Validated EBV splice junctions

chrStart	chrEnd	ID	Read Depth	Strand
833	1498	nov_8SMRT_278III	286	+
833	4820	nov_8SMRT_57III	65	+
833	4932	nov_1SMRT_1III	2	+
833	8523	nov_3SMRT_41III	44	+
1838	3856	nov_2SMRT_116III	118	+
1838	4820	nov_1SMRT_34III	35	+
1838	5006	nov_1SMRT_2III	3	+
1838	8523	nov_8SMRT_36III	44	+
1905	2004	nov_2SMRT_42III	44	-
1905	3569	nov_1SMRT_5III	6	-
1905	5020	nov_7SMRT_19III	26	-
3799	4535	nov_1SMRT_3III	4	+
5169	5685	nov_1SMRT_3III	4	-
5172	11224	nov_1SMRT_4III	5	-
5185	8523	ann_120SMRT_2331III	2451	+
5802	8523	nov_2SMRT_16III	18	+
6217	8523	nov_1SMRT_19III	20	+
6531	8523	nov_2SMRT_35III	37	+
7335	8523	nov_5SMRT_49III	54	+
7623	8523	nov_1SMRT_2III	3	+
7973	8523	nov_3SMRT_5III	8	+
8106	8523	nov_4SMRT_38III	42	+
8221	8523	nov_35SMRT_298III	333	+
8446	8523	nov_4SMRT_83III	87	+
9768	11217	nov_1SMRT_2III	3	-
9768	11224	nov_1SMRT_182III	183	-
10295	11224	nov_1SMRT_11III	12	-
11894	12392	nov_1SMRT_43III	44	-
12366	15626	nov_1SMRT_54III	55	+
15751	16548	nov_1SMRT_267III	268	+
16783	18577	nov_1SMRT_7III	8	+
16783	22834	nov_1SMRT_22III	23	+
19705	22834	nov_1SMRT_4III	5	+
20577	21444	nov_3SMRT_9III	12	-
20577	22381	nov_2SMRT_46III	48	-
20577	22702	nov_3SMRT_107III	110	-
20915	22182	nov_1SMRT_14III	15	+
21111	22702	nov_1SMRT_5III	6	-

21444	22381	nov_1SMRT_7III	8	-
21444	22702	nov_1SMRT_13III	14	-
21603	22381	nov_110SMRT_653III	763	-
21603	22702	nov_13SMRT_1490III	1503	-
22467	22834	nov_2SMRT_209III	211	+
22485	22702	nov_1SMRT_35III	36	-
22890	23119	nov_1SMRT_1III	2	-
22890	23541	nov_2SMRT_240III	242	-
22890	29078	nov_2SMRT_49III	51	-
23045	24077	nov_1SMRT_1III	2	+
23045	24099	nov_5SMRT_282III	287	+
23298	29078	nov_1SMRT_24III	25	-
23789	24099	nov_5SMRT_88III	93	+
24042	24266	nov_1SMRT_2III	3	-
24252	25247	nov_1SMRT_6III	7	+
24252	25455	nov_1SMRT_1III	2	+
26474	27513	nov_1SMRT_24III	25	+
26474	27601	nov_1SMRT_45III	46	+
29502	31025	nov_1SMRT_1III	2	-
36381	37821	nov_2SMRT_20III	22	+
36775	37821	nov_8SMRT_216III	224	+
36775	41171	nov_4SMRT_20III	24	+
37923	41171	ann_13SMRT_581III	594	+
40259	41171	nov_1SMRT_7III	8	+
40494	41171	nov_2SMRT_15III	17	+
40494	41513	nov_1SMRT_3III	4	+
40494	41827	nov_1SMRT_3III	4	+
40750	41171	nov_1SMRT_10III	11	+
40750	41513	nov_1SMRT_3III	4	+
40898	41141	nov_4SMRT_70III	74	-
40898	41585	nov_1SMRT_9III	10	-
41303	41513	nov_19SMRT_647III	666	+
41303	41827	nov_2SMRT_44III	46	+
41668	41827	nov_16SMRT_531III	547	+
41939	46854	ann_21SMRT_253III	274	+
41939	48114	nov_1SMRT_16III	17	+
41939	50212	nov_1SMRT_10III	11	+
41939	51368	nov_6SMRT_5III	11	+
41939	57650	nov_1SMRT_7III	8	+

43753	46854	nov_1SMRT_3III	4	+
43806	46854	nov_6SMRT_87III	93	+
43806	51368	nov_1SMRT_23III	24	+
44550	48114	nov_1SMRT_7III	8	+
45116	46854	nov_8SMRT_325III	333	+
45116	48114	nov_1SMRT_31III	32	+
46769	46854	nov_1SMRT_16III	17	+
46937	47908	nov_7SMRT_54III	61	+
46937	48114	nov_22SMRT_431III	453	+
46937	48433	nov_1SMRT_7III	8	+
46937	51368	nov_2SMRT_10III	12	+
47136	47908	nov_3SMRT_8III	11	+
47136	48114	nov_4SMRT_85III	89	+
47801	48114	nov_1SMRT_6III	7	+
48325	48433	ann_73SMRT_1665III	1738	+
48325	50212	nov_8SMRT_315III	323	+
48325	51368	nov_6SMRT_43III	49	+
48516	50212	ann_79SMRT_2081III	2160	+
48516	51368	nov_2SMRT_12III	14	+
48959	50212	nov_5SMRT_22III	27	+
49484	50212	nov_1SMRT_12III	13	+
49506	50212	nov_3SMRT_37III	40	+
49606	49748	nov_2SMRT_15III	17	+
49606	50212	nov_1SMRT_34III	35	+
49648	50212	nov_1SMRT_3III	4	+
49822	50212	nov_2SMRT_24III	26	+
49858	50212	nov_3SMRT_29III	32	+
50339	51368	ann_72SMRT_673III	745	+
50471	51368	nov_31SMRT_143III	174	+
50963	51368	nov_3SMRT_12III	15	+
51197	51368	nov_1SMRT_3III	4	+
51506	57650	nov_4SMRT_155III	159	+
56155	57650	nov_3SMRT_6III	9	+
56431	57650	nov_1SMRT_5III	6	+
58045	63426	ann_6SMRT_464III	470	+
58045	63728	nov_2SMRT_69III	71	+
58045	64648	nov_3SMRT_163III	166	+
58045	64942	nov_2SMRT_34III	36	+
58646	60126	nov_1SMRT_3III	4	-

58944	63426	nov_1SMRT_22III	23	+
59677	60126	nov_1SMRT_1III	2	-
60049	60126	ann_101SMRT_1818III	1919	-
60049	60291	nov_3SMRT_35III	38	-
60049	63466	nov_1SMRT_7III	8	-
60049	63767	nov_1SMRT_6III	7	-
60213	60291	ann_9SMRT_684III	693	-
60213	63466	nov_1SMRT_22III	23	-
60213	63657	nov_1SMRT_1III	2	-
60213	65533	nov_1SMRT_3III	4	-
63641	63728	ann_5SMRT_603III	608	+
63641	64394	nov_4SMRT_105III	109	+
63827	63908	ann_4SMRT_394III	398	+
63827	64394	nov_3SMRT_213III	216	+
64157	64239	ann_5SMRT_379III	384	+
64320	64394	ann_5SMRT_831III	836	+
64320	64648	nov_3SMRT_130III	133	+
64565	64648	ann_14SMRT_1630III	1644	+
64565	64840	nov_1SMRT_57III	58	+
64864	64942	ann_21SMRT_1283III	1304	+
64864	68776	nov_2SMRT_125III	127	+
65051	66638	nov_3SMRT_287III	290	+
65051	68776	ann_23SMRT_952III	975	+
65167	66638	nov_1SMRT_11III	12	+
65167	68776	nov_2SMRT_58III	60	+
65529	68776	nov_2SMRT_9III	11	+
66264	66638	nov_2SMRT_11III	13	+
66264	68776	nov_3SMRT_15III	18	+
66871	68776	nov_1SMRT_4III	5	+
67045	68776	nov_1SMRT_56III	57	+
67211	68776	nov_18SMRT_329III	347	+
67591	68776	nov_12SMRT_213III	225	+
68320	68776	nov_2SMRT_110III	112	+
103462	104870	nov_1SMRT_257III	258	-
104875	105314	nov_95SMRT_21300III	21395	+
104896	105314	ann_1SMRT_312III	313	+
104974	105314	nov_3SMRT_95III	98	+
105058	105314	nov_20SMRT_1234III	1254	+
113441	118460	ann_6SMRT_3455III	3461	+

118632	127509	nov_1SMRT_2III	3	+
118632	129072	nov_1SMRT_2III	3	+
118632	142986	ann_1SMRT_30III	31	+
135023	135129	ann_213SMRT_17391III	17604	-
135023	142536	nov_11SMRT_97III	108	-
135023	146657	nov_1SMRT_8III	9	-
135023	152589	nov_4SMRT_10III	14	-
135023	152631	nov_17SMRT_48III	65	-
135023	152838	nov_10SMRT_246III	256	-
135023	153422	nov_1SMRT_24III	25	-
135050	135129	nov_1SMRT_51III	52	-
135276	152631	nov_2SMRT_1III	3	-
135356	152838	nov_1SMRT_1III	2	-
136811	137844	nov_1SMRT_7III	8	+
140729	141401	nov_1SMRT_53III	54	-
140729	142536	nov_1SMRT_26III	27	-
140943	141401	nov_1SMRT_2III	3	-
140943	142536	nov_1SMRT_1III	2	-
140963	141401	ann_449SMRT_9735III	10184	-
140963	141930	nov_2SMRT_7III	9	-
140963	142060	nov_1SMRT_7III	8	-
140963	142536	nov_120SMRT_1821III	1941	-
140963	152403	nov_1SMRT_1III	2	-
140963	152589	nov_4SMRT_30III	34	-
140963	152631	nov_28SMRT_144III	172	-
140963	152766	nov_1SMRT_10III	11	-
140963	152838	nov_58SMRT_411III	469	-
141563	152631	nov_2SMRT_1III	3	-
141978	142536	nov_1SMRT_10III	11	-
141978	152838	nov_1SMRT_2III	3	-
142029	142536	nov_1SMRT_12III	13	-
142029	152631	nov_1SMRT_5III	6	-
142029	152838	nov_3SMRT_5III	8	-
142197	143852	nov_1SMRT_77III	78	+
142204	152631	nov_1SMRT_4III	5	-
142210	142986	nov_1SMRT_114III	115	+
142769	152838	nov_2SMRT_20III	22	-
143330	143418	ann_1SMRT_351III	352	+
143330	143920	nov_1SMRT_126III	127	+

143390	143920	nov_1SMRT_36III	37	+
146731	152589	nov_1SMRT_1III	2	-
146731	152631	nov_1SMRT_5III	6	-
146995	152403	nov_1SMRT_1III	2	-
146995	152589	nov_1SMRT_5III	6	-
146995	152631	nov_1SMRT_5III	6	-
146995	152838	nov_1SMRT_6III	7	-
147236	152631	nov_1SMRT_1III	2	-
147814	150084	nov_1SMRT_2III	3	-
147814	152403	nov_1SMRT_6III	7	-
147814	152589	nov_1SMRT_19III	20	-
147814	152631	nov_2SMRT_35III	37	-
147814	152838	nov_1SMRT_24III	25	-
149419	157904	nov_1SMRT_1III	2	+
150367	152589	nov_4SMRT_62III	66	-
150367	152631	nov_1SMRT_99III	100	-
150367	152838	nov_2SMRT_127III	129	-
153081	153166	ann_59SMRT_6716III	6775	-
153081	153422	nov_59SMRT_2462III	2521	-
153271	153422	ann_60SMRT_9985III	10045	-
153686	156893	nov_2SMRT_12III	14	-
153721	154690	nov_2SMRT_48III	50	-
153721	155693	nov_7SMRT_87III	94	-
153721	156893	nov_4SMRT_70III	74	-
154212	155303	nov_1SMRT_16III	17	+
155694	156893	nov_1SMRT_4III	5	-
155953	156893	ann_9SMRT_2566III	2575	-
157376	157748	nov_9SMRT_49III	58	+
157376	158709	nov_1SMRT_511III	512	+
160743	160875	nov_2SMRT_78III	80	+
161718	161810	nov_1SMRT_42III	43	+
168022	168150	ann_139SMRT_9112III	9251	-



## **APPENDIX 8**

Validated EBV polyadenylation sites

chrStart	chrEnd	ID	Read Depth	Strand
58	59	ann_-_150.SMRT_115.Ill	265	-
1466	1467	nov_-_222.SMRT_40.Ill	262	-
4775	4776	ann_-_2806.SMRT_60.Ill	2866	-
9646	9647	ann_-_3383.SMRT_411.Ill	3794	-
9674	9675	ann_+_216.SMRT_7.Ill	223	+
12599	12600	ann_-_422.SMRT_185.Ill	607	-
20188	20189	ann_-_1089.SMRT_225.Ill	1314	-
20304	20305	ann_+_495.SMRT_10.Ill	505	+
26205	26206	ann_-_69.SMRT_3.Ill	72	-
26496	26497	ann_+_300.SMRT_26.Ill	326	+
29046	29047	ann_-_918.SMRT_63.Ill	981	-
29058	29059	ann_+_467.SMRT_380.Ill	847	+
32284	32285	nov_-_35.SMRT_2219.Ill	2254	-
40597	40598	ann_-_775.SMRT_77.Ill	852	-
42963	42964	ann_-_33.SMRT_36.Ill	69	-
47810	47811	ann_-_1578.SMRT_48.Ill	1626	-
52124	52125	ann_+_170.SMRT_12.Ill	182	+
52118	52119	ann_-_126.SMRT_253.Ill	379	-
55952	55953	ann_-_90.SMRT_6.Ill	96	-
57315	57316	ann_+_140.SMRT_39.Ill	179	+
58049	58050	ann_-_1228.SMRT_143.Ill	1371	-
69232	69233	ann_+_217.SMRT_21.Ill	238	+
73394	73395	nov_-_126.SMRT_16.Ill	142	-
73545	73546	ann_+_79.SMRT_15.Ill	94	+
101010	101011	ann_+_11.SMRT_5.Ill	16	+
101276	101277	ann_-_110.SMRT_4492.Ill	4602	-
103134	103135	nov_-_6.SMRT_5.Ill	11	-
106523	106524	ann_+_325.SMRT_346.Ill	671	+
106943	106944	ann_-_470.SMRT_168.Ill	638	-
111702	111703	nov_+_174.SMRT_6.Ill	180	+
113031	113032	ann_-_45.SMRT_17.Ill	62	-
113074	113075	ann_+_1967.SMRT_60.Ill	2027	+
129779	129780	nov_-_5.SMRT_1.Ill	6	-
129808	129809	ann_+_659.SMRT_195.Ill	854	+
132677	132678	nov_+_5.SMRT_1.Ill	6	+
133100	133101	ann_+_1071.SMRT_459.Ill	1530	+
133630	133631	ann_-_373.SMRT_222.Ill	595	-
138492	138493	ann_-_125.SMRT_11.Ill	136	-

138524	138525	ann_+_58.SMRT_2.Ill	60	+
140304	140305	ann_-_1620.SMRT_115.Ill	1735	-
140333	140334	ann_+_917.SMRT_286.Ill	1203	+
145994	145995	ann_+_83.SMRT_13.Ill	96	+
146001	146002	nov_-_150.SMRT_8.Ill	158	-
150127	150128	nov_-_14.SMRT_5.Ill	19	-
150159	150160	nov_-_36.SMRT_10.Ill	46	-
152531	152532	ann_+_8.SMRT_4.Ill	12	+
152866	152867	ann_-_111.SMRT_196.Ill	307	-
156897	156898	ann_+_106.SMRT_33.Ill	139	+
158702	158703	ann_+_2160.SMRT_5.Ill	2165	+
162442	162443	ann_+_4736.SMRT_239.Ill	4975	+
162444	162445	ann_-_94.SMRT_15.Ill	109	-
167304	167305	ann_-_142.SMRT_108.Ill	250	-
167444	167445	ann_+_175.SMRT_10.Ill	185	+
171013	171014	ann_+_3878.SMRT_67.Ill	3945	+

## **APPENDIX 9**

Validated novel EBV transcripts

chrStart	chrEnd	ID	Strand	Block Count	BlockSize	BlockStart
599	9675	BBRT4	+	3	234,253,1152	0,4333,7924
599	9675	BBRT5	+	2	234,1152	0,7924
599	9675	BBRT6	+	3	234,365,1152	0,4221,7924
599	9675	BBRT7	+	4	234,340,179,1152	0,899,4407,7924
599	9675	BBRT8	+	3	234,340,1152	0,899,7924
599	9675	BBRT9	+	4	234,340,365,1152	0,899,4221,7924
4383	9675	BGRT2	+	2	802,1152	0,4140
7765	9675	BGRT3	+	2	681,1152	0,758
7765	9675	BGRT4	+	2	456,1152	0,758
7765	9675	BGRT5	+	2	208,1152	0,758
7765	9675	BGRT6	+	1	1910	0
7765	9675	BGRT7	+	2	341,1152	0,758
16557	20305	BcRT2	+	2	226,1728	0,2020
16557	26497	BcRT3	+	3	226,211,2398	0,6277,7542
19392	20305	BTRT2	+	1	913	0
20624	26497	BTRT3	+	5	291,285,211,153,1042	0,1558,2210,3475,4831
23521	26497	BXRT2	+	2	268,2398	0,578
23521	26497	BXRT3	+	1	2976	0
24396	26497	BVRT3	+	1	2101	0
24470	26497	BVRT4	+	1	2027	0
25612	26497	BVRT5	+	1	885	0
28287	29059	BdRT2	+	1	772	0
28508	29059	BIRT1	+	1	551	0
28682	29059	BIRT2	+	1	377	0
36316	52125	BIRT3	+	6	65,102,132,426,127,757	0,1505,4855,5197,13896,15052
36316	52125	BIRT4	+	5	65,102,132,426,757	0,1505,4855,5197,15052
36715	52125	BIRT5	+	9	60,102,132,155,112,83,211,83,1913	0,1106,4456,4798,5112,10139,11399,11718,13497
40390	52125	BIRT6	+	3	104,112,757	0,1437,10978
40390	69233	BIRT7	+	8	104,132,426,138,395,216,109,457	0,781,1123,10978,17260,24258,24552,28386
40390	69233	BIRT8	+	10	104,426,138,395,215,99,171,216,109,4	0,1123,10978,17260,23036,23338,24004,24

					57	258,24552,283 86
43541	52125	BIRT9	+	6	265,83,417,8 3,127,757	0,3313,4367,4 892,6671,7827
43592	52125	BIRT10	+	3	161,1471,757	0,3262,7776
43592	52125	BIRT11	+	6	214,282,417, 83,127,757	0,3262,4316,4 841,6620,7776
43592	52125	BIRT12	+	4	214,83,211,7 57	0,3262,4522,7 776
43592	52125	BIRT13	+	5	214,282,211, 259,757	0,3262,4522,6 620,7776
43592	52125	BIRT14	+	4	214,1471,259 ,757	0,3262,6620,7 776
43592	52125	BIRT15	+	5	214,83,211,2 59,757	0,3262,4522,6 620,7776
43592	52125	BIRT16	+	2	214,757	0,7776
48118	52125	BART2	+	4	207,83,259,7 57	0,315,2094,32 50
48118	52125	BART3	+	3	398,127,757	0,2094,3250
48118	52125	BART4	+	4	207,83,127,7 57	0,315,2094,32 50
49367	52125	BART5	+	1	2758	0
49367	52125	BART6	+	3	117,259,757	0,845,2001
49367	52125	BART7	+	4	239,110,259, 757	0,381,845,200 1
49367	52125	BART8	+	4	239,74,259,7 57	0,381,845,200 1
49367	52125	BART9	+	3	139,259,757	0,845,2001
55122	57316	BART10	+	1	2194	0
55967	57316	BART11	+	1	1349	0
55967	69233	BART12	+	3	2078,109,457	0,8975,12809
55967	69233	BART13	+	8	188,395,215, 171,216,109, 573,457	0,1683,7459,8 427,8681,8975 ,10671,12809
56600	57316	BART14	+	1	716	0
56626	69233	BART15	+	3	1419,109,457	0,8316,12150
63852	69233	BNRT2	+	4	468,216,109, 457	0,796,1090,49 24
63931	69233	BNRT3	+	6	226,81,171,2 16,109,457	0,308,463,717, 1011,4845
65094	69233	BNRT4	+	2	2117,457	0,3682
65094	69233	BNRT5	+	2	1170,457	0,3682
65094	69233	BNRT6	+	3	1170,573,457	0,1544,3682
65094	69233	BNRT7	+	2	2497,457	0,3682
65094	69233	BNRT8	+	2	1951,457	0,3682
65094	69233	BNRT9	+	2	435,457	0,3682
65094	69233	BNRT10	+	2	73,457	0,3682
71970	73546	BCRT3	+	1	1576	0

72926	106524	BCRT4	+	7	522,66,66,66, 66,66,1210	0,4893,7966,1 1039,14112,23 331,32388
77648	106524	BWRT2	+	4	237,66,66,12 10	0,9390,18609, 27666
77648	106524	BWRT3	+	7	237,66,66,66, 66,66,1210	0,3244,6317,9 390,12463,186 09,27666
96086	106524	BWRT4	+	2	237,1210	0,9229
98915	101011	BYRT1	+	1	2096	0
100357	101011	BHRT2	+	1	654	0
104766	106524	BHRT3	+	2	109,1210	0,548
104920	106524	BHRT4	+	2	54,1210	0,394
104920	106524	BHRT5	+	1	1604	0
104920	106524	BHRT6	+	2	138,1210	0,394
105143	106524	BHRT7	+	1	1381	0
105392	106524	BHRT8	+	1	1132	0
105562	106524	BHRT9	+	1	962	0
105598	106524	BHRT10	+	1	926	0
105633	106524	BHRT11	+	1	891	0
105654	106524	BHRT12	+	1	870	0
105841	106524	BFRT4	+	1	683	0
105936	106524	BFRT5	+	1	588	0
106064	106524	BFRT6	+	1	460	0
109573	111703	BFRT7	+	1	2130	0
109842	111703	BFRT8	+	1	1861	0
110486	111703	BFRT9	+	1	1217	0
113215	129809	BFRT10	+	3	226,172,2300	0,5245,14294
128963	129809	BaRT2	+	1	846	0
130772	132678	BMRT3	+	1	1906	0
131763	133101	BMRT4	+	1	1338	0
136346	138525	BSRT2	+	1	2179	0
137815	140334	BSRT3	+	1	2519	0
138538	140334	BSRT4	+	1	1796	0
139716	140334	BLRT3	+	1	618	0
141590	145995	BLRT4	+	3	620,344,2075	0,1396,2330
141799	145995	BLRT5	+	2	398,2143	0,2053
143867	145995	BERT1	+	1	2128	0
144255	145995	BERT2	+	1	1740	0
145374	145995	BERT3	+	1	621	0
150164	152532	BERT4	+	1	2368	0

156011	156898	BRRT3	+	1	887	0
156011	158703	BRRT4	+	1	2692	0
156860	158703	BRRT5	+	1	1843	0
157040	158703	BRRT6	+	2	336,955	0,708
159860	162443	BKRT5	+	1	2583	0
160213	162443	BKRT6	+	2	530,1568	0,662
160213	162443	BKRT7	+	1	2230	0
160298	162443	BKRT8	+	1	2145	0
161000	162443	BKRT9	+	1	1443	0
161265	162443	BKRT10	+	1	1178	0
161561	162443	BKRT11	+	2	157,633	0,249
161561	162443	BKRT12	+	1	882	0
161726	162443	BKRT13	+	1	717	0
161795	162443	BKRT14	+	1	648	0
161835	162443	BKRT15	+	1	608	0
165920	167445	BBRT10	+	1	1525	0
169395	171014	BBRT11	+	1	1619	0
169766	171014	BBRT12	+	1	1248	0
167304	169714	BBLT5	-	2	718,1564	0,846
167304	169528	BBLT6	-	2	718,1378	0,846
167304	169528	BBLT7	-	1	2224	0
167304	169342	BBLT8	-	1	2038	0
167304	169342	BBLT9	-	2	718,1192	0,846
162444	163334	BBLT10	-	1	890	0
162444	163080	BKLT1	-	1	636	0
162444	163057	BKLT2	-	1	613	0
152866	157698	BRLT2	-	3	215,264,805	0,556,4027
152866	157698	BRLT3	-	5	215,105,299, 260,805	0,300,556,282 7,4027
152866	157037	BRLT4	-	3	215,2272,144	0,556,4027
152866	157037	BRLT5	-	4	215,105,299, 144	0,300,556,402 7
152866	156950	BRLT6	-	4	215,105,299, 57	0,300,556,402 7
133630	156950	BRLT7	-	4	1393,243,264 ,57	0,19208,19792 ,23263
152866	154145	BZLT3	-	3	215,105,723	0,300,556
152866	154145	BZLT4	-	2	215,723	0,556
152866	154101	BZLT5	-	2	215,679	0,556
152866	154025	BZLT6	-	3	215,105,603	0,300,556
152866	153969	BZLT7	-	2	215,547	0,556



152866	153640	BZLT8	-	3	215,105,218	0,300,556
150159	152868	BZLT9	-	2	208,279	0,2430
150127	152868	BZLT10	-	1	2741	0
150127	152868	BZLT11	-	2	240,279	0,2462
146001	152868	BZLT12	-	2	994,465	0,6402
146001	152868	BZLT13	-	2	1813,279	0,6588
146001	152868	BZLT14	-	3	1813,283,30	0,4083,6837
146001	152868	BZLT15	-	2	1813,237	0,6630
146001	152868	BZLT16	-	2	1813,30	0,6837
146001	152868	BZLT17	-	2	730,279	0,6588
146001	152868	BZLT18	-	2	994,30	0,6837
146001	152868	BZLT19	-	2	994,279	0,6588
146001	152868	BZLT20	-	2	1235,237	0,6630
146001	152868	BZLT21	-	2	994,237	0,6630
146001	152868	BZLT22	-	2	730,237	0,6630
140304	152868	BZLT23	-	2	659,30	0,12534
140304	152868	BZLT24	-	2	659,237	0,12327
140304	152868	BZLT25	-	3	659,577,30	0,1097,12534
140304	152868	BZLT26	-	2	659,465	0,12099
140304	152868	BZLT27	-	3	659,628,237	0,1097,12327
140304	152868	BZLT28	-	2	659,102	0,12462
140304	152868	BZLT29	-	2	659,279	0,12285
140304	152868	BZLT30	-	3	659,803,237	0,1097,12327
140304	152868	BZLT31	-	3	659,162,237	0,1097,12327
140304	152868	BZLT32	-	3	659,1368,30	0,1097,12534
140304	152868	BZLT33	-	3	659,628,30	0,1097,12534
133630	152868	BZLT34	-	2	1393,237	0,19001
133630	152868	BZLT35	-	2	1393,279	0,18959
133630	152868	BZLT36	-	3	1393,227,30	0,1499,19208
133630	152868	BZLT37	-	2	1393,30	0,19208
133630	152868	BZLT38	-	3	1393,147,237	0,1499,19001
150159	152492	Be3LT1	-	1	2333	0
146001	147033	BELT1	-	1	1032	0
133630	147033	BELT2	-	2	1393,376	0,13027
146001	146928	BELT3	-	1	927	0
140304	144155	BELT4	-	2	659,1619	0,2232
140304	143438	BLLT4	-	2	659,2037	0,1097
140304	143438	BLLT5	-	2	659,902	0,2232
140304	143364	BLLT6	-	1	3060	0

140304	143364	BLLT7	-	2	659,828	0,2232
140304	143364	BLLT8	-	2	425,828	0,2232
140304	143364	BLLT9	-	2	659,1963	0,1097
140304	142909	BLLT10	-	2	659,373	0,2232
140304	142909	BLLT11	-	2	659,979	0,1626
140304	142909	BLLT12	-	2	639,1508	0,1097
140304	142909	BLLT13	-	2	659,849	0,1756
140304	142909	BLLT14	-	2	425,1508	0,1097
140304	142909	BLLT15	-	3	659,628,373	0,1097,2232
140304	142909	BLLT16	-	2	639,373	0,2232
133630	142909	BLLT17	-	2	1393,373	0,8906
140304	142238	BLLT18	-	2	659,837	0,1097
140304	142238	BLLT19	-	1	1934	0
140304	141675	BLLT20	-	2	659,274	0,1097
140304	141644	BLLT21	-	1	1340	0
140304	141644	BLLT22	-	2	659,243	0,1097
140304	141134	BLLT23	-	1	830	0
133630	136284	BSLT3	-	2	1393,1155	0,1499
133630	136284	BSLT4	-	1	2654	0
133630	136010	BSLT5	-	2	1420,881	0,1499
133630	136010	BSLT6	-	2	1393,881	0,1499
133630	136010	BSLT7	-	1	2380	0
133630	135866	BSLT8	-	2	1393,737	0,1499
133630	135744	BSLT9	-	1	2114	0
133630	135744	BSLT10	-	2	1393,615	0,1499
133630	135228	BSLT11	-	1	1598	0
106943	109728	BFLT3	-	1	2785	0
106943	109185	BFLT4	-	1	2242	0
106943	109042	BFLT5	-	1	2099	0
106943	108647	BFLT6	-	1	1704	0
106943	108607	BFLT7	-	1	1664	0
103134	103766	BHLT2	-	1	632	0
101276	101787	BHLT3	-	1	511	0
101276	101765	BHLT4	-	1	489	0
101276	101678	BHLT5	-	1	402	0
101276	101662	BHLT6	-	1	386	0
73394	75176	BCLT2	-	1	1782	0
73394	74832	BCLT3	-	1	1438	0
73394	74052	BCLT4	-	1	658	0

58049	63744	BNLT3	-	2	2000,278	0,5417
58049	63744	BNLT4	-	3	1628,87,87	0,2077,5608
58049	63744	BNLT5	-	2	2164,278	0,5417
58049	60252	BNLT6	-	2	2000,126	0,2077
58049	60252	BNLT7	-	1	2203	0
58049	60068	BNLT8	-	1	2019	0
58049	59910	BNLT9	-	1	1861	0
58049	59836	BNLT10	-	1	1787	0
58049	58731	BNLT11	-	1	682	0
55952	58623	BNLT12	-	1	2671	0
55952	56871	BALT6	-	1	919	0
52118	53957	BALT7	-	1	1839	0
52118	52765	BALT8	-	1	647	0
47810	49311	BALT9	-	1	1501	0
40597	43363	BILT3	-	1	2766	0
40597	43339	BILT4	-	2	301,2198	0,544
40597	42961	BILT5	-	1	2364	0
40597	42961	BILT6	-	2	301,1376	0,988
40597	42914	BILT7	-	1	2317	0
40597	42094	BILT8	-	1	1497	0
40597	41598	BILT9	-	2	301,457	0,544
40597	41598	BILT10	-	1	1001	0
40597	41345	BILT11	-	2	301,204	0,544
40597	41345	BILT12	-	1	748	0
32284	33570	BILT13	-	1	1286	0
32284	33325	BILT14	-	1	1041	0
32284	33205	BILT15	-	1	921	0
32284	33118	BILT16	-	1	834	0
32284	33098	BILT17	-	1	814	0
32284	33060	BILT18	-	1	776	0
32284	32958	BILT19	-	1	674	0
32284	32931	BILT20	-	1	647	0
32284	32702	BILT21	-	1	418	0
32284	32616	BILT22	-	1	332	0
29046	31050	BILT23	-	1	2004	0
29046	30819	BILT24	-	1	1773	0
29046	30680	BILT25	-	1	1634	0
29046	30588	BILT26	-	1	1542	0
29046	30287	BILT27	-	1	1241	0

29046	29881	BILT28	-	1	835	0
20188	29846	BILT29	-	3	1415,596,768	0,2514,8890
20188	29846	BILT30	-	3	923,188,768	0,2514,8890
20188	29846	BILT31	-	3	389,188,768	0,2514,8890
29046	29615	BILT32	-	1	569	0
29046	29575	BILT33	-	1	529	0
20188	24425	BVLT2	-	3	1256,188,884	0,2514,3353
20188	24425	BVLT3	-	2	389,2044	0,2193
20188	23150	BXLT3	-	4	389,159,188, 31	0,1256,2514,2 931
20188	23150	BXLT4	-	1	2962	0
20188	23150	BXLT5	-	2	1415,448	0,2514
20188	22696	BXLT6	-	2	1415,315	0,2193
20188	22696	BXLT7	-	1	2508	0
20188	22588	BXLT8	-	2	1415,207	0,2193
20188	22588	BXLT9	-	2	1256,207	0,2193
20188	22564	BXLT10	-	2	1415,183	0,2193
20188	22564	BXLT11	-	2	389,183	0,2193
20188	22564	BXLT12	-	2	389,1120	0,1256
20188	22564	BXLT13	-	3	389,159,183	0,1256,2193
12599	15103	BDLT5	-	1	2504	0
12599	14697	BDLT6	-	1	2098	0
12599	14504	BDLT7	-	1	1905	0
9646	12630	BDLT8	-	1	2984	0
9646	12630	BDLT9	-	2	649,1406	0,1578
9646	12630	BDLT10	-	2	122,1406	0,1578
9646	12630	BDLT11	-	2	2248,238	0,2746
9646	12630	BDLT12	-	2	122,1413	0,1571
4775	12630	BDLT13	-	2	397,1406	0,6449
9646	11492	BDLT14	-	1	1846	0
9646	11442	BDLT15	-	1	1796	0
9646	10889	BDLT16	-	1	1243	0
9646	10649	BDLT17	-	1	1003	0
1466	6209	BGLT6	-	2	439,1189	0,3554
1466	6209	BGLT7	-	3	439,149,524	0,3554,4219
4775	5175	BGLT8	-	1	400	0
4775	5149	BGLT9	-	1	374	0
1466	4451	BGLT10	-	2	439,882	0,2103
1466	4451	BGLT11	-	1	2985	0

1466	3194	BGLT12	-	1	1728	0
1466	2568	BGLT13	-	2	439,564	0,538
1466	2568	BGLT14	-	1	1102	0
58	2568	BGLT15	-	1	2510	0

## **APPENDIX 10**

Updated EBV-Akata annotation

chrStart	chrEnd	ID	Strand	block Count	blockStart	blockSize
58	611	BBLF1	-	1	553	0
58	1735	BGLF5	-	1	1677	0
1466	3179	BGLF4	-	1	1713	0
1466	3398	BGLF3.5	-	1	1932	0
1466	4392	BGLF3	-	1	2926	0
3850	9675	BGRF1/BDRF1	+	2	1335,1152	0,4673
4775	6209	BGLF2	-	1	1434	0
4775	7706	BGLF1	-	1	2931	0
4775	8363	BDLF4	-	1	3588	0
4775	8613	BDLF3.5	-	1	3838	0
9646	10382	BDLF3	-	1	736	0
9646	11754	BDLF2	-	1	2108	0
9646	12660	BDLF1	-	1	3014	0
12599	17019	BcLF1	-	1	4420	0
16669	20283	BcRF1	+	1	3614	0
18317	20305	BTRF1	+	1	1988	0
20188	22564	BXLF2	-	1	2376	0
20188	24443	BXLF1	-	1	4255	0
23919	26497	BXRF1	+	1	2578	0
24642	26497	BVRF1	+	1	1855	0
26205	27460	BVLF1	-	1	1255	0
27060	29059	BVRF2	+	1	1999	0
27960	29059	BdRF1	+	1	1099	0
29046	29846	BILF2	-	1	800	0
29953	52119	RPMS1	+	10	124,106,134,102, 132,155,112,147 1,83,1907	0,1085,2196,7868,1 1218,11561,11874, 16901,18480,20259
30543	30921	Repeat_region	+	1	378	0
30683	30704	ebv-miR-BART3*	+	1	21	0
30720	30742	ebv-miR-BART3	+	1	22	0
30819	30841	ebv-miR-BART4	+	1	22	0
30857	30880	ebv-miR-BART4*	+	1	23	0
30942	30966	ebv-miR-BART1- 5p	+	1	24	0
30978	31000	ebv-miR-BART1- 3p	+	1	22	0
31144	31166	ebv-miR-BART15	+	1	22	0
31267	31291	ebv-miR-BART5	+	1	24	0
31309	31327	ebv-miR-BART5*	+	1	18	0

31387	31411	ebv-miR-BART16	+	1	24	0
31507	31529	ebv-miR-BART17-5p	+	1	22	0
31545	31568	ebv-miR-BART17-3p	+	1	23	0
31625	31647	ebv-miR-BART6-5p	+	1	22	0
31663	31685	ebv-miR-BART6-3p	+	1	22	0
32161	35304	LF3	-	1	3143	0
32356	34874	Repeat_IR4_PstI	+	1	2518	0
34799	36037	oriLyt	+	1	1238	0
34864	35921	DRright_similar_to_40265..41308	+	1	1057	0
37106	37127	ebv-miR-BART21-5p	+	1	21	0
37140	37162	ebv-miR-BART21-3p	+	1	22	0
37590	37612	ebv-miR-BART18-3p	+	1	22	0
38027	38049	ebv-miR-BART7*	+	1	22	0
38063	38085	ebv-miR-BART7	+	1	22	0
38360	38382	ebv-miR-BART8	+	1	22	0
38395	38418	ebv-miR-BART8*	+	1	23	0
38547	38569	ebv-miR-BART9*	+	1	22	0
38585	38608	ebv-miR-BART9	+	1	23	0
38792	38815	ebv-miR-BART22	+	1	23	0
38910	38932	ebv-miR-BART10*	+	1	22	0
38945	38968	ebv-miR-BART10	+	1	23	0
39126	39150	ebv-miR-BART11-5p	+	1	24	0
39164	39185	ebv-miR-BART11-3p	+	1	21	0
39526	39548	ebv-miR-BART12	+	1	22	0
39805	39828	ebv-miR-BART19-5p	+	1	23	0
39844	39865	ebv-miR-BART19-3p	+	1	21	0
39929	39950	ebv-miR-BART20-5p	+	1	21	0
39964	39986	ebv-miR-BART20-3p	+	1	22	0
40116	40138	ebv-miR-BART13*	+	1	22	0
40153	40176	ebv-miR-BART13	+	1	23	0
40334	40356	ebv-miR-BART14*	+	1	22	0
40368	40390	ebv-miR-BART14	+	1	22	0



40597	41973	LF2	-	1	1376	0
40597	43339	LF1	-	1	2742	0
40597	44232	BILF1	-	1	3635	0
42962	47876	BALF5	-	1	4914	0
44337	44359	ebv-miR-BART2-5p	+	1	22	0
44337	44397	ebv-miR-BART2-3p	+	1	24	0
47136	52119	A73	+	4	1189,83,127,751	0,1297,3076,4232
47810	50467	BALF4	-	1	2657	0
47810	52496	BALF3	-	1	4686	0
50708	52124	BARF0	+	1	1416	0
52118	55944	BALF2	-	1	3826	0
55952	56541	BALF1	-	1	589	0
56626	57316	BARF1	+	1	690	0
57626	69233	LMP-2A	+	9	419,215,99,249,8 1,171,216,109,45 7	0,5800,6102,6282,6 613,6768,7022,731 6,11150
58049	58623	BNLF2a	-	1	574	0
58049	60601	LMP-1	-	3	2000,87,310	0,2077,2242
58049	58423	BNLF2b	-	1	374	0
59496	59658	Repeat_unit_range_167452..167484	+	1	162	0
60836	69233	LMP-2B	+	9	155,215,99,249,8 1,171,216,109,45 7	0,2590,2892,3072,3 403,3558,3812,410 6,7940
61183	63319	TR_repeat-unit_range_169138..169671	+	1	2136	0
65059	69233	BNRF1	+	1	4174	0
70004	70171	EBER1	+	1	167	0
70331	70505	EBER2	+	1	174	0
70691	72577	OriP	+	1	1886	0
70797	71307	FR_Repeats_EBN A1_Binding_sites_I	+	1	510	0
72285	72400	Dyad_Symmetry_EBNA1_Binding_site_II	72285	115	0	
72926	73546	BCRF1/IL10	+	1	620	0
74600	74601	Cp_Promoter	+	1	1	0
74600	101011	Cp-EBNA2	+	19	144,32,66,132,66 ,132,66,132,66,1 32,66,132,66,132 ,66,132,33,122,1 641	0,290,3219,3366,62 92,6439,9365,9512, 12438,12585,15511 ,15658,18584,1873 1,21657,21804,241 45,24262,24770



106235	106257	ebv-miR-BHRF1-3	+	1	22	0
106943	108105	BFLF2	-	1	1162	0
106943	109517	BFLF1	-	1	2574	0
109095	111703	BFRF1A	+	1	2608	0
109842	113075	BFRF1	+	1	3233	0
110486	113075	BFRF2	+	1	2589	0
112360	113075	BFRF3	+	1	715	0
113031	123091	BPLF1	-	1	10060	0
113031	126224	BOLF1	-	1	13195	0
113404	160578	Qp-EBNA1	+	3	37,172,1869	0,5056,45305
113412	113460	EBNA_1_Binding_site_III	+	1	48	0
120666	120913	Repeat_unit_range -57298..57348	+	1	247	0
121369	121414	Repeat_unit_range -58001..58015	+	1	45	0
125918	129809	BORF1	+	1	3891	0
127102	129809	BORF2	+	1	2707	0
129705	133101	BaRF1	+	1	3396	0
130772	133101	BMRF1	+	1	2329	0
131714	133101	BMRF2	+	1	1387	0
133220	133363	Repeat_unit_range -69852..69922	+	1	143	0
133630	135228	BSLF2/BMLF1	-	2	1393,99	0,1499
133630	137783	BSLF1	-	1	4153	0
134541	134631	Repeat_unit_range -71173..71181	+	1	90	0
137815	138525	BSRF1	+	1	710	0
138492	139386	BLLF3	-	1	894	0
139443	140334	BLRF1	+	1	891	0
139796	140334	BLRF2	+	1	538	0
140304	140923	BLLF2	-	1	619	0
140304	142909	BLLF1-splice_variant	-	2	659,1508	0,1097
140304	142909	BLLF1	-	1	2605	0
141051	141388	Repeat_unit_range -77683..77713	+	1	337	0
144956	145026	Repeat_family_type_A	+	1	70	0
145029	145055	Repeat_family_type_B	+	1	26	0
145055	145130	Repeat_family_type_C	+	1	75	0
145134	145160	Repeat_family_type_B	+	1	26	0

145160	145238	Repeat_family_type_C	+	1	78	0
145238	145309	Repeat_family_type_A	+	1	71	0
145319	145397	Repeat_family_type_C	+	1	78	0
145397	145468	Repeat_family_type_A	+	1	71	0
145644	145731	Repeat_family_type_D	+	1	87	0
145731	145818	Repeat_family_type_D	+	1	87	0
148270	148387	Repeat_unit_range -84902..84961	+	1	117	0
150159	152868	BZLF2	-	1	2709	0
150810	150993	Repeat_unit_range -87442..87456	+	1	183	0
151248	151521	Repeat_unit_range -87880..87918	+	1	273	0
152866	153969	BZLF1	-	3	215,105,547	0,300,556
152866	157037	BRLF1	-	3	215,2531,144	0,556,4027
153321	153420	Repeat_region	+	1	99	0
155814	156898	BRRF1	+	1	1084	0
157040	158703	BRRF2	+	1	1663	0
158984	159620	Repeat_family-IR3	+	1	636	0
160572	162443	BKRF2	+	1	1871	0
160811	162443	BKRF3	+	1	1632	0
161762	162443	BKRF4	+	1	681	0
162444	165001	BBLF4	-	1	2557	0
164547	167445	BBRF1	+	1	2898	0
166431	167445	BBRF2	+	1	1014	0
167304	169677	BBLF2/BBLF3	-	2	718,1527	0,846
169648	171014	BBRF3	+	1	1366	0

## **BIOGRAPHY**

Christina (Tina) O'Grady was born and raised in Kimberley, British Columbia. She attended the University of British Columbia in Vancouver to earn her Bachelor of Science in Biology.

She then enrolled at The Catholic University of America in Washington, DC, where she completed a dual Master's program in Cell & Microbial Biology and Library & Information Science. She worked as a librarian for several years at the University of New Orleans and at

Mount Sinai School of Medicine in New York, NY, all the while becoming increasingly interested in biological information and its potential to answer important questions. In 2011

she was admitted to the Biomedical Sciences program at Tulane University School of Medicine and soon received a Ruth L. Kirschstein NRSA F31 predoctoral fellowship to work in the lab of Dr. Erik Flemington. Upon completion of her PhD she will relocate to Belgium for a postdoctoral fellowship with Drs. Ingrid Struman and Franck Dequiedt at the University of Liège.