Copy Number Variation Patterns in DGRP

Ellison, Christopher[1a]; Rele, Chinmay P.[1b*]

Abstract

Transposable Elements (TEs) are a significant remodeller to mammalian and invertebrate genomes. They do this largely by manipulating gene expression in many ways – some of which involve activating and deactivating promoters and/or enhancers of particular genes. In order to study TE interaction in *Drosophila*, we did. Preliminary extrapolation and subdivided the TEs based on their copy number in the genomes of Drosophila Genetics Reference Panel (DGRP) flies. We explain some high copy number TEs as well as the trend that leads to this copy number. We have also started on calculating the correlations for each TE to each gene, but have not yet completed the code.

[1]Department of Genetics, Rutgers University, New Brunswick, New Jersey, USA
[a]Assistant Professor, Department of Genetics, Rutgers University
[b]Undergraduate Student, Department of Genetics, Rutgers University
**For more information or access to files, please contact chinmay.rele@rutgers.edu*

Introduction

Transposable elements (TEs) were first identified by McClintock in maize in the mid 50s [1], and since have been known to occur in both major kingdoms [2]. There are two major types of TEs that are segregated based on their mode of transposition [3]. Namely, retrotransposons and DNA transposons. Generally, both of these types can be defined as viruses without a shell. Retrotransposons move within the genome using an RNA intermediate and the enzyme transposase to move across the genome in a 'Ctrl-C + Ctrl-V' mechanism [4], where there is an increase in copy number of the element by 1 each time it transposes. DNA transposons transpose via a 'Ctrl-C + Ctrl-V' mechanism [5], and thus, we should not expect to see high copy number for these transposons. Furthermore, DNA transposons can be characterised into two more sub-types. Autonomous elements are those that do not need the enzyme (transposase) to function [6] as they code for their own transposase [7]. Non-autonomous elements need an external source of transposase, usually in the form of a mother element [8]. In this paper, we seek to explain the pattern of these TE insertions in *Drosophila melanogaster* populations using the Drosophila Genetics Reference Panel (DGRP). *D. melanogaster* has been a significant organism to study genetics and chromosomal loci since Morgan revolutionised the science with his white eyed mutant, and since Sturtevant and Bridges first pioneered chromosomal mapping [9]. They are also useful as they have a large number of offspring and a shorter generation time [10].

Due to the fact that TEs insert randomly into the genome [11], a majority of them are bound to be deleterious to the organism (if they insert into a functional gene) or neutral (if they insert within the introns or intergenic regions). However, if they are inserted into and help the promoter of a gene that is essential to the functioning of the individual enhance expression, these TEs will tend to have a much higher copy number at that locus across the population and will be defined as adaptive for the locus [12]. TEs are very important as the fate of the host cell is dependent on the mobility and insertion pattern of highly mobile TEs [13].

Previous methods of genome sequencing are inadequate when the need is to find TE copy number [14]. This is because most of these methods rely on reading short stretches of DNA and aligning them to a reference genome with some percent of polymorphisms being allowed and set by the experimenter [15]. DGRP data was sequenced using Illumina HiSeq [16]. Illumina HiSeq has a short read length (of upto 150bp) [17], and since the read length of an average TE is about 300bp [18], many TEs within the genome (at different loci) might get aligned to a single locus. This diminishes the copy number that is recorded from the actual copy number within the genome. In

order to rectify this bias, we choose to use Oxford Nanopore MinION™ sequencing whose average read length range from 250bp to 10kbp [19-21].

In this paper, we seek to explain the copy number variation in the DGRP strains based on particular gene expression, and the likelihood that this copy number is an artefact of other effects such as possible bias in Illumina sequencing and read-length bias. We know that several of the adaptive TEs act as enhancers based on the histone modification data and that the frequencies of many of the adaptive TEs in the DGRP population [22]. These results could help us understand the extent to which TEs influence gene expression and vice versa, and determine whether they are affected by gene expression of some genes, or are affecting expression of genes.

Pre-Computational Analysis

The copy numbers of five major classes of TEs were analysed in the DGRP strains whose data was uploaded to DGRP[1]. A brief summary of the information of these classes follow:

I. Foldback Elements: Members of this repetitive class of elements have terminal inverted repeats that flank a central region. These repeats are homologous across the whole class [23]. Due to their homologous flanking repeats, they also have a tendency to form chromosomal insertions [24].
II. IR Elements: Inverted repeat elements, also now termed as Miniature Inverted Transposable Elements (MITES) play important roles in genome evolution [25]. They have increased transposition activity which is facilitated by increasing mutation rate (μ) [26].
III. LTR: Long Terminal Repeat Elements are usually retroviral and share many similarities to retroviral elements [27]. They are a sub-family of retrotransposons in that they have the ability to migrate in the genome via an RNA intermediate. As the name suggests, they have long palindromic repeats that flank their core [28]. They make up about a tenth of mammalian genomes [29].
IV. Non-LTR: Non-Long Terminal Repeat Elements are the opposite of LTRs in only that they do not have long terminal repeats. They still are hotspots for recombination and chromosomal reorganisation [30].
V. SINEs: Short Interspersed Nuclear Elements can be found in high copy number as they are part of the retrotransposon group. They also have a high copy number in mammalian genomes [31].

Unsurprisingly, upon primary investigation, LTR and non-LTR transposons seem to have the highest copy number in the *Drosophila* genome, with DMRER1DM having the highest average number of copies. This abnormally high expression for this TE could be attributed to the levels of expression of TE suppressor genes, but this cannot be proven unless further analysis is done. Theoretically, TE copy number should be lower in strains that have high expression of TE suppressor genes [32].

Some strains that have DMRER1DM have a much higher frequency of this TE than other strains. One of the following reasons could explain this abnormal presence of TEs in the genome:

1. The genome is not good at protecting itself. This can be due to the fact that the expression of TE suppressor genes is reduced [33], which is primarily due to piRNAs [34]. Some DNA methylation can also be stochastic [35].

[1] http://dgrp2.gnets.ncsu.edu

2. The particular copy of the TE had a much higher mutation rate that enables it to be more active [36]. Retrotrasposons go through more replications than the genome does as they have to go through more rounds of replication than the latter [37]. They mutate at a whole order of magnitude higher than lytic RNA viruses and the host genome [38], and are thus termed as mutagenic and recombination hotspots [39]. This increase in μ could be due to an intrinsic factor, such as possibly a correlated SNP, or an extrinsic factor such as inefficient suppression gene expression.

For TEs with a high copy number, there is a large variation of the copy number housed by each individual strain.

We have found TEs the have a low copy number in one strains and a relatively high copy number in another strain [*Appendix 05*]. This will allow us to sequence both strains and see why there is differential expression.

TE_name	Common name	Highest RAL	Lowest RAL
DMRER1DM	R1A1-element	RAL-732	RAL-439
F	F-element	RAL-176	RAL-375

Table 01: Showing the strains with the highest and lowest copy number for the highest average copy number TEs.

Computational Analysis

This script was used to convert the male and female gene expression files {*the files are in Amarel under* `/projects/genetics/ellison_lab/chinmay` *(request access)*} to a convenient format with consistent naming for the strains. Many of the strains in the files had replicates, so in order to reduce confusion when computing, we had to combine those replicates into a single line and take the average. All of the strains annotated in the file had 2 replicates other than line_890, which only had a single replicate.

The list of tasks carried out in the file are summarised below:

1. A list of the male strains with replicates was made.
2. A dictionary with the ID of the strain as well as the index of the is made for further use.
3. If the strains were in the TE file, the replicate floats were averaged and were added to `002_male_condensed_expression.txt`.
4. If the TE strains were in the condensed expression file, they were added to another file — `TE_condensed.txt`.
5. Both of these files had the same number of columns, so no more computation was needed before finding the correlation. Spearman correlation was used as it accounted for outliers and removes them from correlation analysis.
6. After this, the highest and lowest correlation was isolated and the TE and the gene of interest were found.

The same tasks were carried out for the female expression data. The source code for male data is commented to further explain the reason for the particular line(s) of the script.

Spearman's correlation (ρ) was used as it is a non-parametric measure of rank between two variables [40]. It asses how well the relationship between two variables is monotonic [41]. It is less stringent than Pearson's correlation as it is not greatly affected by outliers. The code to find the correlation is given in the appendix and is annotated with commends [*Appendix 04*]. A .py of this file is available upon request.

Highest correlation might mean TE is affecting gene expression.

Lowest correlation might mean TE is reducing gene expression or gene is a TE suppressor.

Also, correlation of the Illumina read length and median coverage [*Appendix 01* + 02] were done. There should have been very low or no correlation between the two and the average copy number of the TE.

This was true for most TEs, but some TEs showed a relatively high correlation between 0.43 to -0.44 for both, the average read length achieved using Illumina sequencing as well as the median read length

Future Work

For those genes that show high correlation with TEs, we will look into those particular gene-TE pairs and see if one has an effect on the other. We will find TE-gene correlations, and then find the shuffled correlations. Δcorr. (= unaltered correlations - shuffled correlations) will show us if the correlations attained are purely by chance, or have some significance to them.

This can be easier done if we have better sequence data from Oxford Nanopore™ rather than Illumina as Oxford Nanopore™ has much longer reads than average, so it would negate the copy number bias that might happen with Illumina data. We will have to confirm this data again with computational means, but can then investigate particular pairs that have correlations.

Acknowledgements

We would like to thank Dr. Derek Gordon for introducing me to computational genetics as a career option.

We would also like to specially thank Weihuan Cao for constant input with improving procedures, preparing embryos, extracting nuclei and generally rearing the fly strains that have been used for the study.

Bibliography

1. McClintock, B., *Controlling elements and the gene.* Cold Spring Harbor symposia on quantitative biology, 1956. **21**: p. 197-216.
2. Feschotte, C., The contribution of transposable elements ot the evolution of regulatory networks. Nature Reviews Genetics, 2008. **9**: p. 397-405.
3. Ivics, Z., et al., Transposon-mediated Genome Manipulations in Vertebrates. Nature methods, 2009. **6**(6): p. 415-422.
4. Christensen, S., G. Pont-Kingdon, and D. Carroll, Target Specificity of the Endonuclease from the Xenopus laevis Non-Long Terminal Repeat Retrotransposon, Tx1L. Molecular and Cellular Biology, 2000. **20**(4): p. 1219-1226.
5. Muñoz-López, M. and J.L. García-Pérez, DNA Transposons: Nature and Applications in Genomics. Current Genomics, 2010. **11**(2): p. 115-128.
6. Wicker, T., et al., A unified classification system for eukaryotic transposable elements. Nature Reviews Genetics, 2007. **8**: p. 973.
7. Feschotte, C., N. Jiang, and S.R. Wessler, Plant transposable elements: where genetics meets genomics. Nature Reviews Genetics, 2002. **3**: p. 329.
8. Böhne, A., et al., Transposable elements as drivers of genomic and biological diversity in vertebrates. Chromosome Research, 2008. **16**(1): p. 203-215.
9. Sturtevant, A.H. and E. Novitski, The Homologies of the Chromosome Elements in the Genus Drosophila. Genetics, 1941. **26**(5): p. 517-541.
10. Morgan, T.H., C. Bridges, and A. Sturtevant, *The genetics of Drosophila melanogaster.* Biblphia genet, 1925. **2**: p. 1-262.
11. Ding, D. and H. Lipshitz, Spatially regulated expression of retroviruses-like transposons during Drosophila melanogaster embryogenesis. Genet. Res., 1994. **64**: p. 167.
12. González, J. and D.A. Petrov, The adaptive role of transposable elements in the Drosophila genome. Gene, 2009. **448**: p. 124-133.
13. de la Rosa, J., et al., A single-copy Sleeping Beauty transposon mutagenesis screen identifies new PTEN-cooperating tumor suppressor genes. Nature Genetics, 2017. **49**: p. 730.
14. Wheeler, D.A., et al., The complete genome of an individual by massively parallel DNA sequencing. nature, 2008. **452**(7189): p. 872-876.
15. Notredame, C. and D.G. Higgins, SAGA: sequence alignment by genetic algorithm. Nucleic acids research, 1996. **24**(8): p. 1515-1524.
16. Rahman, R., et al., Unique transposon landscapes are pervasive across Drosophila melanogaster genomes. Nucleic Acids Research, 2015. **43**(22): p. 10655-10672.
17. Quail, M.A., et al., A tale of three next generation sequencing platforms: comparison of Ion Torrent, Pacific Biosciences and Illumina MiSeq sequencers. BMC Genomics, 2012. **13**(1): p. 341.
18. Kazazian, H.H. and J.V. Moran, The impact of L1 retrotransposons on the human genome. Nature genetics, 1998. **19**(1): p. 19-24.
19. Branton, D., et al., The potential and challenges of nanopore sequencing. Nat Biotech, 2008. **26**(10): p. 1146-1153.
20. Goodwin, S., et al., Oxford Nanopore sequencing, hybrid error correction, and de novo assembly of a eukaryotic genome. Genome research, 2015. **25**(11): p. 1750-1756.
21. Jain, M., et al., The Oxford Nanopore MinION: delivery of nanopore sequencing to the genomics community. Genome biology, 2016. **17**(1): p. 239-239.
22. Schor, I.E., et al., Promoter shape varies across populations and affects promoter evolution and expression noise. Nature Genetics, 2017. **49**: p. 550.

23. Silber, J., et al., Distribution and conservation of the foldback transposable element in Drosophila. J Mol Evol, 1989. **28**(3): p. 220-4.
24. Marzo, M., M. Puig, and A. Ruiz, The Foldback-like element Galileo belongs to the P superfamily of DNA transposons and is widespread within the Drosophila genus. Proceedings of the National Academy of Sciences of the United States of America, 2008. **105**(8): p. 2957-2962.
25. Lu, C., et al., Miniature Inverted–Repeat Transposable Elements (MITEs) Have Been Accumulated through Amplification Bursts and Play Important Roles in Gene Expression and Species Diversity in Oryza sativa. Molecular Biology and Evolution, 2012. **29**(3): p. 1005-1017.
26. Shirasawa, K., et al., Characterization of active miniature inverted-repeat transposable elements in the peanut genome. TAG. Theoretical and Applied Genetics. Theoretische Und Angewandte Genetik, 2012. **124**(8): p. 1429-1438.
27. Copeland, C.S., et al., The Sinbad retrotransposon from the genome of the human blood fluke, Schistosoma mansoni, and the distribution of related Pao-like elements. BMC Evolutionary Biology, 2005. **5**: p. 20-20.
28. Godde, J.S. and A. Bickerton, The Repetitive DNA Elements Called CRISPRs and Their Associated Genes: Evidence of Horizontal Transfer Among Prokaryotes. Journal of Molecular Evolution, 2006. **62**(6): p. 718-729.
29. McCarthy, E.M. and J.F. McDonald, Long terminal repeat retrotransposons of Mus musculus. Genome Biology, 2004. **5**(3): p. R14.
30. Edelmann, W., et al., A recombination hotspot in the LTR of a mouse retrotransposon identified in an in vitro system. Cell, 1989. **57**(6): p. 937-946.
31. Kramerov, D.A. and N.S. Vassetzky, *Short Retroposons in Eukaryotic Genomes*, in *International Review of Cytology*. 2005, Academic Press. p. 165-221.
32. Ingelbrecht, I.L., J.E. Irvine, and T.E. Mirkov, Posttranscriptional Gene Silencing in Transgenic Sugarcane. Dissection of Homology-Dependent Virus Resistance in a Monocot That Has a Complex Polyploid Genome. Plant Physiology, 1999. **119**: p. 1187-1198.
33. Liu, M., et al., Two levels of protection for the B cell genome during somatic hypermutation. Nature, 2008. **451**: p. 841.
34. Houwing, S., et al., A role for Piwi and piRNAs in germ cell maintenance and transposon silencing in Zebrafish. Cell, 2007. **129**(1): p. 69-82.
35. Jaenisch, R. and A. Bird, Epigenetic regulation of gene expression: how the genome integrates intrinsic and environmental signals. Nature Genetics, 2003. **33**: p. 245.
36. Paquin, C.E. and V.M. Williamson, Temperature Effects on the Rate of Ty Transposition. Science, 1984. **226**(4670): p. 53-55.
37. Drake, J.W., et al., Rates of Spontaneous Mutation. Genetics, 1998. **148**(4): p. 1667.
38. Drake, J.W., *Rates of spontaneous mutation among RNA viruses.* Proceedings of the National Academy of Sciences, 1993. **90**(9): p. 4171-4175.
39. Arbeithuber, B., et al., Crossovers are associated with mutation and biased gene conversion at recombination hotspots. Proceedings of the National Academy of Sciences, 2015. **112**(7): p. 2109-2114.
40. Hess, A., H. Iyer, and W. Malm, Linear trend analysis: a comparison of methods. Atmospheric Environment, 2001. **35**(30): p. 5211-5222.
41. Smouse, P.E., J.C. Long, and R.R. Sokal, Multiple Regression and Correlation Extensions of the Mantel Test of Matrix Correspondence. Systematic Biology, 1986. **35**(4): p. 627-632.

Appendix

TE	Median_corr	pval	TE	Median_corr	pval
S2	0.49923579	1.22E-10	INVADER	-0.08397177	0.31192097
TRANSIB1	0.46879419	2.12E-09	TC1	-0.09020181	0.27724934
DOC5	0.44977033	1.10E-08	ROOA_LTR	-0.09038232	0.27628515
G4_DM	0.4496335	1.11E-08	ROXELEMENT	-0.11279804	0.17374325
TC3	0.44858852	1.21E-08	DOC4	-0.11590058	0.16213124
HOPPER2	0.4438221	1.80E-08	Q	-0.11780728	0.15528843
DM_ROO	0.42494216	8.11E-08	DMDM11	-0.12177454	0.14174935
DMCR1A	0.41373227	1.90E-07	JUAN	-0.1248365	0.13192839
DMIFACA	0.37029706	3.90E-06	GYPSY12	-0.12826269	0.12156722
TRANSIB3	0.36833194	4.43E-06	DMRER1DM	-0.12911834	0.11908062
DMBARI1	0.36133459	6.91E-06	GTWIN	-0.13323846	0.10765543
DMZAM	0.35251568	1.19E-05	DMW1DOC	-0.13811594	0.09526083
GYPSY7	0.35016055	1.38E-05	DM88	-0.14625819	0.07711671
DMTHB1	0.34897559	1.48E-05	DMIS297	-0.152146	0.06581969
INVADER3	0.3377036	2.88E-05	STALKER	-0.15223011	0.06566867
QBERT	0.32942669	4.62E-05	GYPSY3	-0.1547925	0.06120144
GYPSY10	0.32255244	6.77E-05	412	-0.16422212	0.04685536
DMREPG	0.3164146	9.45E-05	DMRTMGD1	-0.16897712	0.04075777
DMTOM1_LTR	0.30238742	0.00019727	G5_DM	-0.17082878	0.03857055
FW2	0.29012517	0.0003645	DIVER2	-0.1821781	0.02721284
ACCORD	0.2709121	0.00090378	OPUS	-0.18259887	0.0268536
DMHFL1	0.23725936	0.00380983	DOC3	-0.18681933	0.02346895
TC1-2	0.210878	0.01035238	DMLINEJA	-0.18798323	0.02260264
DMPOGOR11	0.19154139	0.02012271	DOC2	-0.19033362	0.02093677
GYPSY4	0.16792274	0.04204907	TARTC	-0.19677418	0.01690364
INVADER6	0.16711744	0.04305824	AY561850	-0.20068358	0.01480003
BS	0.14858355	0.07248113	INVADER2	-0.20704925	0.01186156
BS3	0.13411073	0.10535056	QUASIMODO	-0.20880658	0.01114643
CIRC	0.12288116	0.13813777	Tinker	-0.21292742	0.00961622
DM33463	0.12061223	0.14561954	DME010298	-0.21356537	0.00939669
G6_DM	0.11434871	0.16786487	G5A	-0.21796385	0.00799985
DMTRDNA	0.10239454	0.21715963	GYPSY2	-0.21936818	0.00759438
TIRANT	0.09967008	0.22970752	DME278684	-0.22355182	0.00649238
FROGGER	0.09423239	0.25625725	STALKER4	-0.22759912	0.00556415
F	0.09398491	0.25751383	DMTNFB	-0.2292216	0.00522664
DMIS176	0.0823853	0.32118335	OSV	-0.25506601	0.00182176
TRANSIB2	0.07508515	0.36606741	DMRER2DM	-0.26240326	0.00132382
DM14101	0.07009022	0.39890289	RT1C	-0.27210396	0.00085588
Beagle	0.06656269	0.42311477	DSRN	-0.27449949	0.00076658
GYPSY6	0.06577984	0.4286012	DMCOPIA	-0.27495757	0.00075051
Beagle2	0.06492082	0.43466833	INVADER4	-0.27814076	0.00064713
DMGYPF1A	0.06000312	0.47033174	DME9736	-0.28076933	0.00057181
McCLINTOCK	0.0570506	0.49248856	TABOR	-0.28201916	0.00053891
DMU89994	0.05392726	0.51651685	DMMDG3	-0.28210565	0.0005367
STALKER2	0.04495038	0.58877284	BAGGINS	-0.28588745	0.00044783
TAHRE	0.03466921	0.67676704	GYPSY8	-0.28668786	0.00043086
IVK	0.02805431	0.73588842	AF222049	-0.30196589	0.00020157
DMAURA	0.01873975	0.82175562	DME487856	-0.31197071	0.00011976
SPRINGER	0.01574712	0.84985338	DMTN1731	-0.31435054	0.00010552
Xanthias	0.00748036	0.92834951	RT1B	-0.31670535	9.30E-05
DM06920	-0.00444113	0.957424	BLOOD	-0.33410118	3.54E-05
DMBLPP	-0.01688679	0.83912873	G2	-0.36160331	6.80E-06
MARINER2	-0.01900095	0.81931337	1360	-0.37656405	2.59E-06
HEL	-0.0260178	0.75442569	STALKER3	-0.41659487	1.53E-07
PPI251	-0.0392844	0.63663108	INE1	-0.52575646	8.04E-12
ROVER	-0.04416801	0.59528263	NEOR1A	-0.55092538	4.83E-13
DM23420	-0.071229	0.39126648			

Appendix 01: Correlation of TE copy number with median coverage for each strain.

TE	Read_corr	pval	TE	Read_corr	pval
TRANSIB3	0.51258718	3.20E-11	DMTHB1	0.01491657	0.8576866
HOPPER2	0.44551972	1.56E-08	PPI251	0.01146636	0.8903669
DMCR1A	0.43490812	3.70E-08	DMCOPIA	0.01056407	0.8989462
TRANSIB1	0.42827733	6.26E-08	SPRINGER	0.01030243	0.9014362
Q	0.42698356	6.92E-08	DMPOGOR11	0.00977435	0.9064648
TC3	0.40933296	2.63E-07	G5A	0.00546826	0.9475899
QBERT	0.39837795	5.81E-07	AY561850	0.00227878	0.9781463
S2	0.39397597	7.93E-07	BS	0.00149534	0.9856586
G4_DM	0.39113003	9.67E-07	Tinker	0.00012303	0.99882
which	0.37016017	3.94E-06	INVADER	-0.00453753	0.9565007
FW2	0.35504306	1.02E-05	DMBLPP	-0.00496976	0.9523618
GYPSY7	0.33502984	3.36E-05	BLOOD	-0.00683521	0.9345147
DMTNFB	0.31059111	0.0001288	DOC3	-0.00768383	0.9264061
DOC5	0.28282726	0.0005186	ROXELEMENT	-0.02381181	0.7746646
DM33463	0.28042573	0.0005812	DMRTMGD1	-0.02527237	0.7612468
INVADER3	0.27403215	0.0007833	INE1	-0.03157683	0.7041866
DMZAM	0.26897441	0.0009869	BS3	-0.03648851	0.6608288
DMIFACA	0.25364641	0.0019358	F	-0.03952839	0.634537
DMBARI1	0.25073342	0.0021905	GYPSY6	-0.04610308	0.5792413
HEL	0.23739526	0.0037892	STALKER2	-0.04960889	0.5507019
GYPSY10	0.23693933	0.0038589	DME487856	-0.05012818	0.5465335
DMTRDNA	0.20947915	0.0108829	DMW1DOC	-0.05036793	0.5446142
CIRC	0.20511265	0.012696	DME278684	-0.0569245	0.493447
IVK	0.19428222	0.0183761	ROOA_LTR	-0.06747303	0.4167863
G6_DM	0.18322075	0.0263301	GYPSY12	-0.07298949	0.3796353
FROGGER	0.18226654	0.027137	DMMDG3	-0.07365817	0.3752732
DM_ROO	0.17303948	0.0360893	DMLINEJA	-0.08140605	0.3269884
DM23420	0.13763139	0.0964392	BAGGINS	-0.09075141	0.2743206
DMIS176	0.13391002	0.1058775	NEOR1A	-0.09197763	0.2678618
MARINER2	0.13258827	0.109399	GTWIN	-0.09488855	0.2529461
DOC4	0.11927427	0.1501731	TC1-2	-0.10945166	0.186945
GYPSY4	0.11231204	0.1756166	DM14101	-0.11032277	0.1834399
ACCORD	0.11156217	0.1785362	AF222049	-0.11250858	0.1748572
Beagle2	0.09864897	0.2345394	JUAN	-0.11545813	0.1637508
INVADER6	0.07808667	0.3471638	DMGYPF1A	-0.1183137	0.153508
TAHRE	0.07557974	0.3629095	INVADER4	-0.11976922	0.1484763
DMDM11	0.07158349	0.3889074	G2	-0.12360463	0.1358148
STALKER	0.07025562	0.3977882	McCLINTOCK	-0.12538494	0.1302258
Xanthias	0.06863377	0.4087978	DMTN1731	-0.14014307	0.0904546
DMIS297	0.0684297	0.4101957	OPUS	-0.14050191	0.0896243
DM88	0.06507335	0.4335875	GYPSY2	-0.14439631	0.0809982
DM06920	0.06199131	0.4557241	G5_DM	-0.15479147	0.0612032
Beagle	0.06194469	0.4560637	TABOR	-0.15587219	0.0593952
INVADER2	0.06181535	0.4570067	GYPSY3	-0.18301386	0.0265033
DMREPG	0.05849355	0.4815915	DMRER2DM	-0.18352412	0.0260779
TIRANT	0.05560326	0.5035489	1360	-0.21092697	0.0103342
STALKER4	0.05459859	0.511302	TRANSIB2	-0.21383597	0.0093049
DOC2	0.05002536	0.5473576	DMU89994	-0.24372483	0.0029321
DME9736	0.0499485	0.5479741	DME010298	-0.25981363	0.0014833
QUASIMODO	0.04972017	0.5498073	DMAURA	-0.28172231	0.0005466
TARTC	0.04448858	0.5926114	ROVER	-0.30097889	0.000212
DMHFL1	0.04409138	0.595922	RT1B	-0.30871183	0.0001422
TC1	0.04221226	0.6116962	DSRN	-0.35107926	1.30E-05
412	0.03950225	0.6347612	RT1C	-0.35772907	8.65E-06
STALKER3	0.03628871	0.6625719	OSV	-0.40885314	2.73E-07
DMRER1DM	0.02532803	0.7607368	GYPSY8	-0.42760349	6.59E-08
DIVER2	0.02046792	0.8056299			

Appendix 02: Correlation of TE copy number with read length for each strain.

TE_id	True Name	Family	TE_id	True Name	Family
DMRER1DM	R1A1-element	non-LTR	DMU89994	Burdock	LTR
F	F-element	non-LTR	DM06920	HeT-A	non-LTR
DMW1DOC	Doc	non-LTR	ROVER	rover	LTR
DMCR1A	Cr1a	non-LTR	GYPSY6	gypsy6	LTR
DM_ROO	roo	LTR	BS3	BS3	non-LTR
DMBARI1	Bari1	IR-elements	ACCORD	accord	LTR
RT1B	RT1B	non-LTR	SPRINGER	springer	springer
DMLINEJA	jockey	non-LTR	McCLINTOCK	McClintock	LTR
DMRER2DM	R2-element	non-LTR	DMPOGOR11	pogo	IR-elements
OPUS	opus	LTR	TRANSIB2	transib2	IR-elements
DME010298	GATE	LTR	DMHFL1	hobo	IR-elements
DOC3	DOC3	non-LTR	DMTHB1	HB	IR-elements
CIRC	Circe	LTR	PPI251	P-element	IR-elements
BAGGINS	baggins	non-LTR	STALKER2	Stalker2	LTR
DOC2	Doc2-element	non-LTR	INVADER3	invader3	LTR
DMIFACA	I-element	non-LTR	TC1-2	Tc1-2	IR-elements
G2	G2	non-LTR	MARINER2	mariner2	IR-elements
QUASIMODO	Quasimodo	LTR	G5_DM	G5	non-LTR
DMIS176	17.6	LTR	GYPSY2	gypsy2	LTR
DMIS297	297	LTR	TIRANT	Tirant	LTR
DME487856	Max-element	LTR	DOC5	Porto1	non-LTR
DMTN1731	1731	LTR	DMGYPF1A	gypsy	LTR
DMCOPIA	copia	LTR	G4_DM	G4	non-LTR
STALKER4	Stalker4	LTR	TAHRE	TAHRE	non-LTR
DME278684	Rt1a	non-LTR	DMAURA	aurora-element	LTR
DMRTMGD1	mdg1	LTR	STALKER3	Stalker3	LTR
STALKER	Stalker	LTR	QBERT	accord2	LTR
INVADER2	invader2	LTR	GYPSY12	gypsy12	LTR
GYPSY4	gypsy4	LTR	Q	Q-element	non-LTR
DMTRDNA	hopper	IR-elements	DMTOM1_LTR	Tom1	LTR
412	412	LTR	G5A	G5A	non-LTR
DMBLPP	flea	LTR	HEL	Helena	non-LTR
DM88	Dm88	LTR	GYPSY7	gypsy7	LTR
DME9736	Idefix	LTR	GYPSY10	gypsy10	LTR
Beagle	HMS-Beagle	LTR	DSRN	Dsim\ninja	LTR
DMDM11	micropia	LTR	GYPSY3	gypsy3	LTR
Tinker	diver	LTR	DOC4	Doc4-element	non-LTR
ROXELEMENT	X-element	non-LTR	GTWIN	gtwin	LTR
BLOOD	blood	LTR	HOPPER2	hopper2	IR-elements
DM23420	3S18	LTR	INVADER6	invader6	LTR
IVK	Ivk	non-LTR	DMZAM	ZAM	LTR
INVADER4	invader4	LTR	FW2	Fw2	non-LTR
DM33463	S-element	IR-elements	S2	S2	IR-elements
TC1	Tc1	IR-elements	AY561850	TART-A	non-LTR
DIVER2	diver2	LTR	FROGGER	frogger	LTR
INVADER	invader1	LTR	RT1C	Rt1c	non-LTR
Xanthias	Xanthias	LTR	DMTNFB	FB	Foldback elements
DMMDG3	mdg3	LTR	TRANSIB3	transib3	IR-elements
Beagle2	HMS-Beagle2	LTR	TC3	Tc3	IR-elements
AF222049	Transpac	LTR	OSV	Osvaldo	LTR
BS	BS	non-LTR	INE1	INE-1	SINE-like elements
G6_DM	G6	non-LTR	DM14101	TART-B	non-LTR
1360	1360	IR-elements	TRANSIB1	transib1	IR-elements
DMREPG	G-element	non-LTR	NEOR1A	Dnet\R1A	non-LTR
TABOR	Tabor	LTR	GYPSY8	gypsy8	LTR
JUAN	Juan	non-LTR	TARTC	TART-C	non-LTR
ROOA_LTR	rooA	LTR			

Appendix 03: Common names, true names and families of TEs used in study.

Appendix 04: Script for finding the correlations of the TEs and DGRP strains.

```
# .. General Guidelines for code:
# ..   All comments of lines would be in the line above the concerning script.
# ..       This is because Atom gives an error <= 80 chars per line.
# ..   All comments should be preceded by '..'.
# ..    This is because you might later silence particular parts of the script.
# ..        Which would comment them out.

# .. Imports
from scipy import stats
# import math
import numpy as np
# import pandas as pd
# from matplotlib import pyplot as plt
# import seaborn as sns

# ........................................................................
# ..................................MALE..................................
# ........................................................................
# .. THE CODE FOR THE MALE AND THE FEMALE CORRELATION DATA ARE THE SAME .......
# .. INCLUDING, BUT NOT LIMITED TO ALL VARIABLE NAMES. THE ONLY DISTINCT ......
# .. CHANGE IS THE INPUT FILES (MALE INPUT FILES ARE USED), THE OUPUT FILES, ..
# .. AND THE PRINT STATEMENTS. ..........................................
# ........................................................................

# ..  1     Open TE file to read order and strain_ids.
opendata = open('TE_inc.txt', 'r')

# .. Open first line of TE file.
firstTEline = opendata.readline()
# .. Read first line of TE file and split into a list.
TE_order = firstTEline.split()

# .. 2      Making dictionaries to average expression data.
# .. Open male expression file.
openmdata = open('male_expression.txt', 'r')
# .. Open male file to find length for loop later.
lenmdata = open('male_expression.txt', 'r')

# .. Assign variables to the first line of the data.
firstmline = openmdata.readline()

# .. Make a list of the first line of the data.
m_lst = firstmline.split()

# .. Make a strain list for all strains in the male data.
m_strain_list = []
# .. Add 'gene' to m_strain_list.
for item in m_lst:
    if ':' not in item:
        m_strain_list.append(item)

# .. Make an empty dictionary for male_expression.
# ..    This dictionary should contain male strains and the indexes that their
# ..    replicates are housed at.
m_columns = {}
# .. Go through items in first line of male_expression.
for i in m_lst:
    # ..If the item is 'gene', skip computation.
    if i == 'gene':
        continue
    # .. Assign variable to the index number to make it easier to add later.
    m_colnum = m_lst.index(i)
    # .. Assign variables to the strain_id and rep number by splitting the
    # .. string.
    (ids, rep) = i.split(':')
    # .. Assign num as an empty string to add the strain NUMBER to.
    num = ''
    # .. Go through each character in the id. 'line_xxx' in this case.
    for char in ids:
        # .. If the character in question is a digit (x), add to num as string.
```

```
            if char.isdigit():
                num += char
            # .. Because less than 1000 strains, all have 3 digit designations.
            # .. Thus, add '0' to 2x digit strains and nothing to 3x digit strains.
            # .. Assign this string as RAL.
            if len(num) == 2:
                RAL = 'RAL-0' + str(num)
            elif len(num) == 3:
                RAL = 'RAL-' + str(num)
        # .. Add strain_id to m_strain_list.
        m_strain_list.append(RAL)
        # .. If there is a key with the strain_id, add the colnum as another item
        # .. in list of values.
        if m_columns.get(RAL):
            m_columns[RAL].append(m_colnum)
        # .. If there is no key with the strain_id, make a key and add the colnum
        # .. as a list of a single value.
        else:
            m_columns[RAL] = [m_colnum]

    # .. Make a list of the keys of the male columns dictionary.
    m_keys = list(m_columns.keys())

    # .. Make an empty list to add the intersection of male and TE strains.
    intersection = sorted(list(set(TE_order) & set(m_keys)))

    # .. Make a list of indexes that should be used to float TE values.
    TE_float_index = []
    # .. Cycle through items in TE_order.
    for item in TE_order:
        # .. If item is the header, skip computation.
        if item == 'id':
            TE_float_index.append(TE_order.index(item))
        # .. Otherwise, add the index of the item to the list only if they are in
        # .. both, m_keys and f_keys.
        elif item in m_keys:
            indx = TE_order.index(item)
            TE_float_index.append(indx)

    # .. Have a variable for the nimber of lines of the expression files.
    endi = len(lenmdata.readlines())+1

    male_expr = []

    # .. Make a file to open and write to.
    with open('002_male_condensed_expression.txt', 'w') as f:
        # .. Print a tab delimated line of the intersection of male and TE strains.
        print('\t'.join(intersection), file=f)
        # .. Loop through all lines for range of length of expression data.
        for i in range(endi):
            # .. Mame temp lists for the average as well as the strains that have
            # .. already been averaged.
            temp = []
            temp_id = []
            # .. Open the row (as a string) and then convert it to a list.
            row_string = openmdata.readline()
            row = row_string.split()
            # .. Cycle through items in the list.
            for item in row:
                # .. If the item is the row header, append item.
                if 'FB' in item or 'XL' in item:
                    temp.append(item)
                # .. If the item is not header...
                else:
                    # .. Float item for computation.
                    float(item)
                    # .. Find index number of the item.
                    indx = row.index(item)
                    # .. Find the strain by looking in m_strain_list.
                    strain = m_strain_list[indx]
                    # .. If the strain is in TE_order, continue.
                    if strain in TE_order:
```

```
                        # .. If only appears once (lenght of values in column = 1),
                        # .. then simply add the item as a string to temp list and
                        # .. also add the strain_id to the temp_id list.
                        if len(m_columns[strain]) == 1:
                            temp.append(str(item))
                            temp_id.append(strain)
                        # .. If it has 2 replicates, then find average.
                        elif len(m_columns[strain]) == 2:
                            # .. Find the indexes of replicates.
                            colA = m_columns[strain][0]
                            colB = m_columns[strain][1]
                            # .. Find the expression of the gene, float it and
                            # .. average the two values.
                            avg = (float(row[colA])+float(row[colB]))/2
                            # If the strain is already in temp_id, do nothing.
                            if strain in temp_id:
                                continue
                            # .. If the strain is not in temp_id, add the strain
                            # .. to temp_id and also add average as a string.
                            else:
                                temp_id.append(strain)
                                temp.append(str(avg))
                        # .. If it has 3 replicates, then find average.
                        elif len(m_columns[strain]) == 3:
                            # .. Find the indexes of replicates.
                            colA = m_columns[strain][0]
                            colB = m_columns[strain][1]
                            colC = m_columns[strain][2]
                            # .. Find the expression of the gene, float it and
                            # .. average the two values.
                            avg = (float(row[colA])+float(row[colB]))/3
                            # If the strain is already in temp_id, do nothing.
                            if strain in temp_id:
                                continue
                            # .. If the strain is not in temp_id, add the strain
                            # .. to temp_id and also add average as a string.
                            else:
                                temp_id.append(strain)
                                temp.append(str(avg))
                # .. Add the temp list to the file as tab delimited.
                if len(temp) != 0:
                    temp2 = []
                    # .. Print the line to the condensed file for future reference.
                    print('\t'.join(temp), file=f)
                    # .. Now, we need to make a lists of lists (matrix) so we do not
                    # .. need to import the from the output file.
                    # .. If it is the header row, do not import.
                    if 'id' in temp:
                        continue
                    # .. If it is not the header row, then convert all the numbers to
                    # .. floats and let the names of the genes remain as strings.
                    else:
                        for item in temp:
                            if temp.index(item) != 0:
                                temp2.append(float(item))
                            else:
                                temp2.append(item)
                    male_expr.append(temp2)

# .. Make a list of lists that have the condensed TE data.
TEs = []
for item in TE_order:
    if item == 'id':
        TEs.append(item)
    if item in m_keys:
        TEs.append(item)

# .. Make a list that has the index of the strains in the TE file that appear
# .. in the DGRP files as well.
a = open('TE_inc.txt', 'r')
b = a.readline()
c = b.split()
```

```
TE_index = []
for item in TEs:
    if item in c:
        TE_index.append(c.index(item))

# .. Make the lists of lists of the TE condensed file.
a = open('TE_inc.txt', 'r')
lena = len(open('TE_inc.txt', 'r').readlines())
TE_condensed = []
for i in range(lena):
    temp = []
    b = a.readline()
    c = b.split('\t')
    if 'id' in c:
        continue
    for item in c:
        if c.index(item) == 0:
            temp.append(item)
        elif c.index(item) in TE_index:
            temp.append(float(item))
    while len(temp) < len(TE_index):
        temp.append(np.mean(temp[1:]))
    TE_condensed.append(temp)

# .. Write this list of lists into a file for easy transport.s
with open('TE_condensed.txt', 'w') as f:
    for item in TE_condensed:
        temp = []
        for it in item:
            temp.append(str(it))
        print('\t'.join(temp), file=f)

# .. Open a file to write the correlaitons to.
with open('male_corr.txt', 'w') as f:
    for i in range(len(TE_condensed)):
        ar1 = TE_condensed[i][1:]
        for j in range(len(male_expr)):
            ar2 = male_expr[j][1:]
            if len(ar1) == len(ar2):
                temp = []
                corr, pval = stats.spearmanr(ar1, ar2)
                temp.append(TE_condensed[i][0])
                temp.append(male_expr[j][0])
                temp.append(str(corr))
                temp.append(str(pval))
                            print('\t'.join(temp), file=f)
```