The WTX tumor suppressor interacts with the transcriptional corepressor TRIM28

Woo Jae Kim[1], Ben S. Wittner[1], Arnaud Amzallag[1], Brian W. Brannigan[1], David T. Ting[1], Sridhar Ramaswamy[1], Daniel A. Haber[1,2]

[1] Massachusetts General Hospital Cancer Center and Harvard Medical School, Charlestown, Massachusetts 02129, USA
[2] Howard Hughes Medical Institute, Chevy Chase, Maryland 20815, USA

Running Title: Interaction between WTX and TRIM28

Keywords: WTX, TRIM28, transcriptional repression

Address Correspondence to
Dr. Daniel Haber
MGH Cancer Center
CNY-7, Bldg 149, Charlestown, MA 02129
Tel 617 726 7805 fax 617 724 5919
Email Haber@helix.mgh.harvard.edu

SUPPLEMENTAL EXPERIMENTAL PROCEDURES

Bioinformatics.

DGE: To compute digital gene expression (DGE) from the Helicos sequence data for the RefSeq genes, we used the DGE pipeline of the HeliSphere 1.1.498.63 software using the Mouse.Txome reference files with default settings (1). We ignored DGE output corresponding to mitochondrial and ribosomal RNA and Helicos control spike-ins. To compute DGE for Repbase, in order to deal with the fact that reads of repetitive sequence will often align to multiple locations in the genome, we used a method described below called PASA (Probabilistic Assessment of Sequencing Alignments) that assigns weights to multiple alignments for a read.

Differentially Expressed Transcripts: DEGseq comparisons of DGE profiles were run using the DEGexp function of version 1.0.5 of the Bioconductor (www.bioconductor.org) DEGseq package. The “method” argument was set to “MARS” for the MA-plot-based method with Random Sampling (2). Family-wise error rate (FWER) was computed from the p-values output by DEGexp using the method of Holm (3) and a gene or repetitive element was considered differentially expressed if it had an FWER less than 0.05 and a fold-change greater than 2.

Genomic closeness of RefSeq genes and Repbase elements: For identities and genomic locations of genes we used the knownGene and knownToLocusLink tables of UCSC’s mm9 build of the mouse genome and for repetitive elements we used the rmsk table of that build. For any gene we considered the padded gene to be the portion of the gene’s chromosome that extends from 10kb upstream of the TSS to 10kb downstream of the end of the 3’ UTR. For a particular type of repetitive element (e.g., LINE) and a collection of genes, G, we computed for what percent of the genes in the collection does the padded gene intersect any of the repetitive elements of that type that were up-regulated in mouse ESCs harboring Wtx hairpins and in ESCs harboring Trim28 hairpins. Call that percentage p(G). Denote by G_a the collection of genes that were up-regulated in mouse ESCs harboring Wtx hairpins and in ESCs harboring Trim28 hairpins. We then created 10,000 random sets of genes of the same size as G_a, which we will call $G_{r,n}$ for n = 1, 2, …, 10,000. To make the figures, we made a histogram of the $p(G_{r,n})$ and marked $p(G_a)$ with a vertical red bar drawn on that histogram. We computed a p-value for the null hypothesis that there is no relation between the genomic location of the genes up-regulated by Wtx and Trim28 hairpins and the repetitive elements of that type up-regulated by Wtx and Trim28 hairpins as $(\#(p(G_{r,n}) \geq p(G_a)) + 1)/10{,}001$.

DGE for Repbase: Accurate alignment of sequencing reads to repetitive elements is challenging because of the high sequence similarity between DNA repeats**.** Accordingly repetitive element sequence will often align to multiple locations in the genome, potentially causing biases and errors in the genomic location of expressed repeats (4). To address this challenge, we performed the following steps:

The first step was to align the reads to the mouse genome so that if there were multiple alignments of a read with reasonable quality, all such alignments were included in the aligner's output. Specifically, we used the BASIC pipeline of the HeliSphere 1.1.498.63 software using the mm9-based Mouse.Genome reference files with default settings except for bestOnly = False and maxDelta = 0.3. (1).

The second step was to compute coverage using the PASA method described below, which apportions the coverage from each read to all its alignments in a probabilistically reasonable way. As input to the PASA method we gave the alignment computed in the previous step and the per-nucleotide error model for Helicos sequencing given by Table S6 in Pushkarev et al. (5).

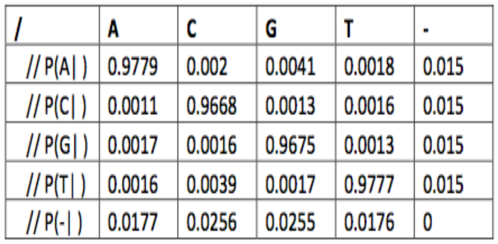
Step three: The coverage of each repetitive element in Repbase was computed by summing the coverage for every nucleotide in the element. For each repetitive element type, a total coverage was computed by summing the coverage for all the elements of that type. By "repetitive element type" we mean the three-tuple (repClass, repFamily, repName). An example is (LINE, L1, L1MDb). We downloaded Repbase from UCSC's table browser with the following settings: assembly = mm9, group = Variation and Repeats, track = RepeatMasker, table = rmsk, Create one BED record per = Whole Gene.

The fourth and final step was to divide the counts for all of the combinations in Repbase by the average read length (i.e., 34), so that the counts represented the number of reads falling in the regions rather than the number of nucleotides in the reads falling in the regions.

The PASA (Probabilistic Assessment of Sequencing Alignments) method

It is sometimes the case that a sequencing read aligns reasonably well to multiple genomic locations and there is no way to determine which genomic location was the true origin of the read. This occurs frequently in the sequencing of reads from repetitive elements. To address this situation, when computing coverage the PASA method apportions the read to the various genomic locations to which the read aligned according to the probability that the read originated in each of the genomic locations.

In order to do this, PASA makes use of a per-nucleotide error model, which must be supplied as an input to the method. The error model must give the probability that the sequencer will output an A, C, G, T, or nothing given that at that location the actual sequence contained an A, C, G, T, or nothing. An example of such a per-nucleotide error model is given by the following table from Pushkarev et al. (5), which says, for example, that the probability of the sequencer outputting a C when the actual sequence is an A is 0.0011.



/	A	C	G	T	-
// P(A|)	0.9779	0.002	0.0041	0.0018	0.015
// P(C|)	0.0011	0.9668	0.0013	0.0016	0.015
// P(G|)	0.0017	0.0016	0.9675	0.0013	0.015
// P(T|)	0.0016	0.0039	0.0017	0.9777	0.015
// P(-|)	0.0177	0.0256	0.0255	0.0176	0

The method then assumes independence in order to combine the per-nucleotide error probabilities into the probability of the sequencer outputting a sequence given the reference sequence to which the read aligned as follows:

$$\Pr(read \mid ref) = \prod_{i=1}^{readLength} \Pr(read_i \mid ref_i)$$

where $\Pr(read_i \mid ref_i)$ is gotten from the per-nucleotide error model, $read_i$ is the i^{th} nucleotide output by the sequencer and ref_i is the i^{th} nucleotide in the reference sequence that was aligned to the read.

The method then computes a posterior probability for the alignment by normalizing its likelihood to the likelihoods of all the other alignments produced by the same read, as given in the Bayes formula:

$$\Pr(ref \mid read) = \frac{\Pr(read \mid ref)}{\sum_{j=1}^{\Omega} \Pr(read / ref^{j})}$$

where ref^j is the reference sequence at the j^{th} alignment of that read and Ω is the number of alignments of the read. Note the method uses a uniform prior, which corresponds to absence of bias toward any specific genomic region.

An alignment probability, $\Pr(ref \mid read)$, can be viewed as the parameter of a Bernoulli variable modeling whether an alignment is correct or wrong. Hence, if we let *A(loc)* denote the actual number of reads that originate at a genomic location, then E(*A(loc)*), the expected value of *A(loc)*, is simply the sum of the alignment probabilities of all reads aligned to that location. Thus the method defines the coverage at a genomic location as

$$\text{Coverage}(loc) = \text{E}\big(\text{A}(loc)\big) = \sum_{k=1}^{M} \Pr(ref_k \mid read_k)$$

where $read_k$ is the k^{th} read that aligned to that location and ref_k is the reference sequence to which $read_k$ was aligned at that location and M is the number of reads that aligned to that location.

Note that the variance of *A(loc)* can be used to model the error around the coverage estimate due to ambiguous alignments, at the single sample level, and thus is useful when no replicates are available. It is given by the sum of the variances of each Bernoulli variable.

hGSEA: We performed hypergeometric gene set enrichment analysis (hGSEA) as follows. To determine whether gene set A was enriched in gene set B, we used the hypergeometric distribution to test whether the overlap of A and B was larger than would be expected by chance had A and B been drawn randomly from all the genes quantitated by the DGE. To account for multiple hypothesis testing, we applied the method of Benjamini and Hochberg (6) to the p-values so generated for each analysis, yielding false discovery rate (FDR) estimates. Gene sets evaluated were version 3.0 of MSigDB (7).

Target sequences for shRNAs.

Human hairpins
WTX-1 (5’-CCTCTGGAGAAGCGTTATGAA)
WTX-2 (5’-GCCCGTCTTAGAGTATCAGAT)
TRIM28 (5’-CCTGGCTCTGTTCTCTGTCCT)
Mouse hairpins
Wtx-1 (5’-GCCAGACATGCAAGAAGCAAA)
Wtx-2 (5’-CCAGACATGCAAGAAGCAAAT)
Trim28-1 (5’-CCGCATGTTCAAACAGTTCAA)
Trim28-2 (5’-CCACCAGTCTTCAAGGTCTTT)

Primer sequences

	Sequence (5' to 3')	Species	Usage	Ref
mGapdh F	TGGTGAAGCAGGCATCTGAG	Mouse	qPCR	
mGapdh R	TGCTGTTGAAGTCGCAGGAG	Mouse	qPCR	
mWtx F	GCGCAGCAGACAATAAACAG	Mouse	qPCR	

mWtx R	TCTTGGGCATCGCTAGAAGT	Mouse	qPCR	
mTrim28 F	GAACCACTTTGTGAGACCTG	Mouse	qPCR	
mTrim28 R	AAACTGGTACTGATGGTCCT	Mouse	qPCR	
IAP F	CGGGTCGCGGTAATAAAGGT	Mouse	qPCR	(8)
IAP R	ACTCTCGTTCCCCAGCTGAA	Mouse	qPCR	(8)
LINE F	TTTGGGACACAATGAAAGCA	Mouse	qPCR	(8)
LINE R	CTGCCGTCTACTCCTCTTGG	Mouse	qPCR	(7)
Mest F	CCAGAACCGCAGAATCAACC	Mouse	qPCR	
Mest R	GGAAAGATACCTCCATTCGACAG	Mouse	qPCR	
GAPDH F	TACTAGCGGTTTTACGGGCG	Human	ChIP-qPCR	
GAPDH R	TCGAACAGGAGGAGCAGAGAGCGA	Human	ChIP-qPCR	
GAL4DBS F	CACACAGGAAACAGCTATGAC	Human	ChIP-qPCR	
GAL5DBS R	GAATTCGCCAATGACAAGAC	Human	ChIP-qPCR	
Luc F	GCTATTCTGATTACACCCGA	Human	ChIP-qPCR	
Luc R	CCTCTTTGATTAACGCCCAG	Human	ChIP-qPCR	

SUPPLEMENTAL REFERENCES

1. Lipson, D., Raz, T., Kieu, A., Jones, D. R., Giladi, E., Thayer, E., Thompson, J. F., Letovsky, S., Milos, P., and Causey, M. (2009) *Nat Biotechnol* **27**, 652-658
2. Wang, L., Feng, Z., Wang, X., and Zhang, X. (2010) *Bioinformatics* **26**, 136-138
3. Holm, S. (1979) *Scand J Statist* **6**, 65-70
4. Treangen, T. J., and Salzberg, S. L. (2012) *Nat Rev Genet* **13**, 36-46
5. Pushkarev, D., Neff, N. F., and Quake, S. R. (2009) *Nat Biotechnol* **27**, 847-850
6. Benjamini Y, H. Y. (1995) *Journal of the Royal Statistical Society. Series B (Methodological)* **57**, 289-300
7. Rowe, H. M., Jakobsson, J., Mesnard, D., Rougemont, J., Reynard, S., Aktas, T., Maillard, P. V., Layard-Liesching, H., Verp, S., Marquis, J., Spitz, F., Constam, D. B., and Trono, D. (2010) *Nature* **463**, 237-240