CUSTOM MICROARRAY ANALYSIS PIPELINE
Perl scripts and file formats

Figure 1. Mapping probes to genes

This flowchart represents the processes involved in extracting information from genomic annotation files and integrating it with probe data. Two alternate approaches are available to map probes to genes, namely BLAT-mapping using `filter.pl` and position-mapping using `map_probes_to_exons.pl` together with `map_probes_to_genes.pl`.

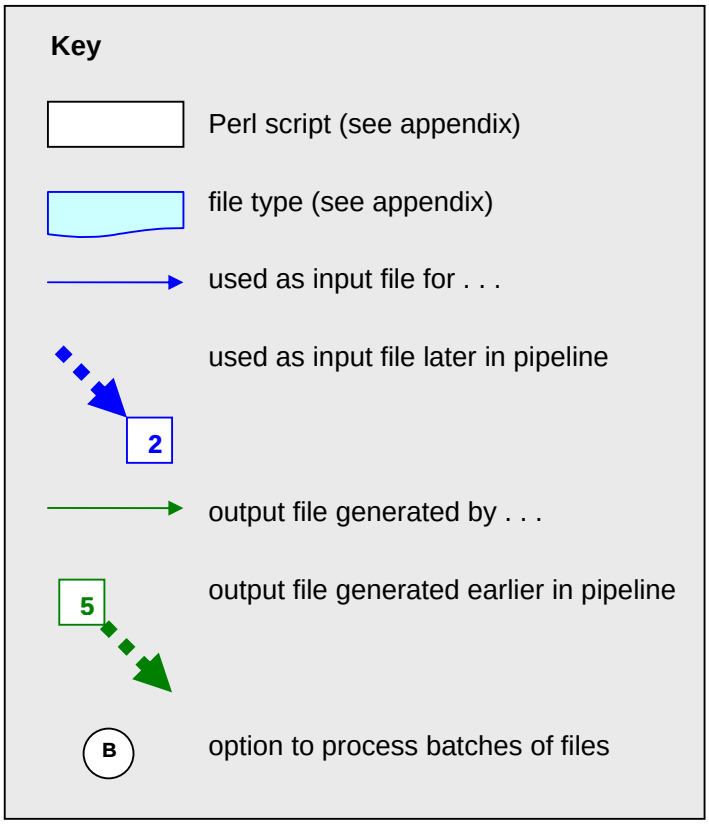


Figure 1. Mapping probes to genes

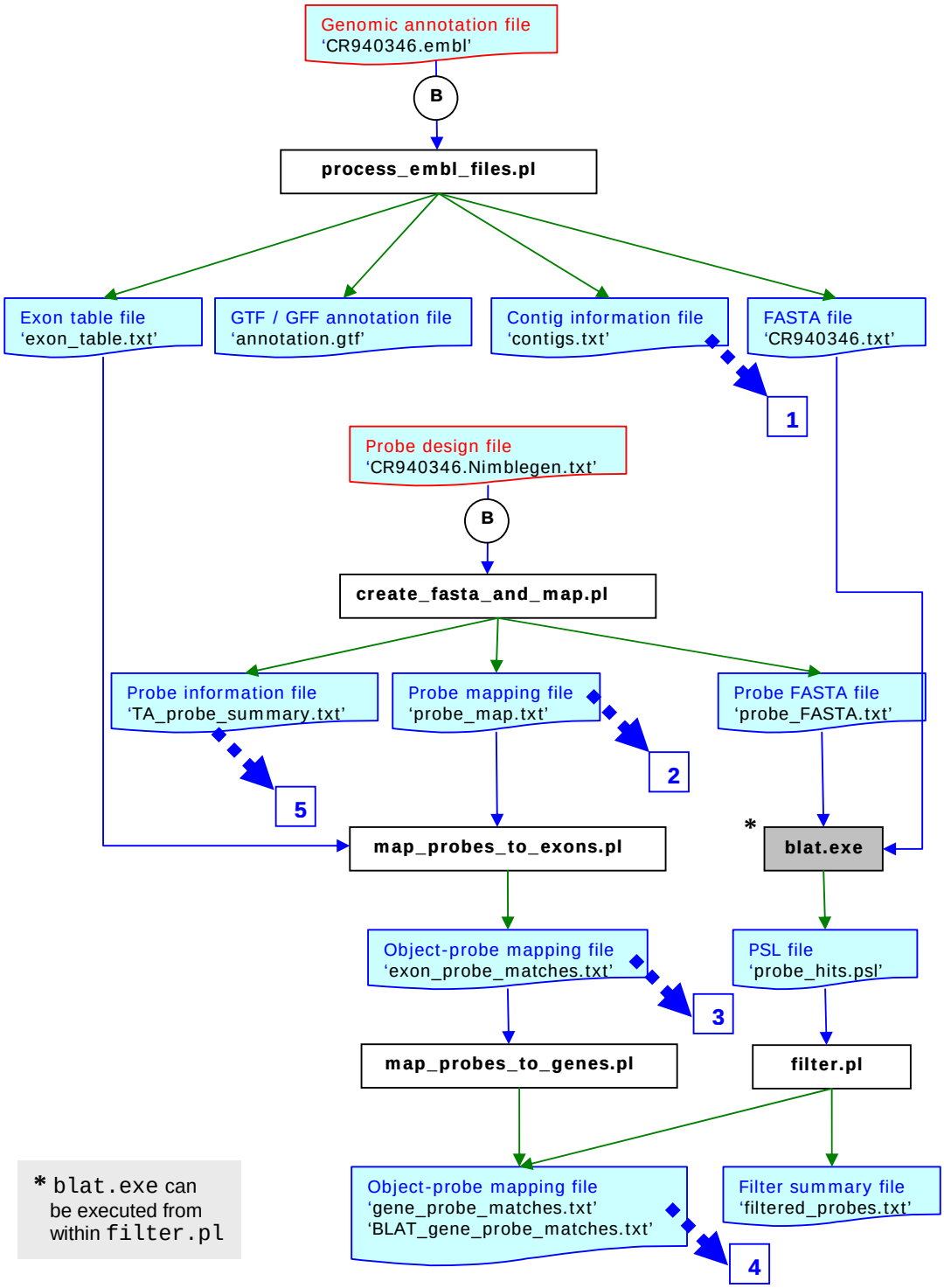


Figure 2. Determining gene expression

This flowchart represents the processes involved in identifying control probes from the microarray design and summarising the expression level of 'objects' (i.e. genes, exons etc.) using `abstract_intensities.pl`. In order to test whether an object has expression levels above background noise, `gc_wilcoxon.pl` is used.

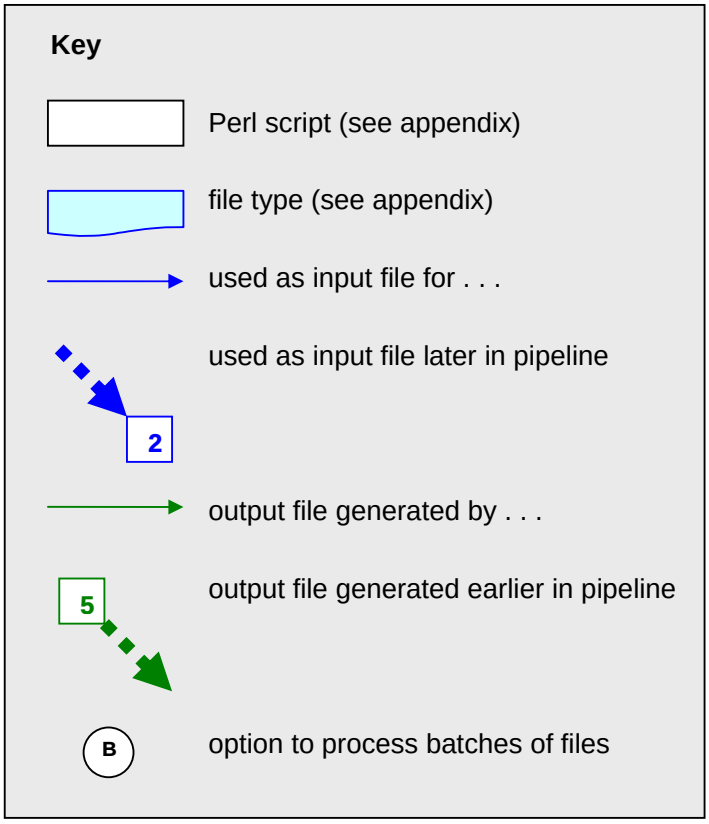


Figure 2. Determining gene expression

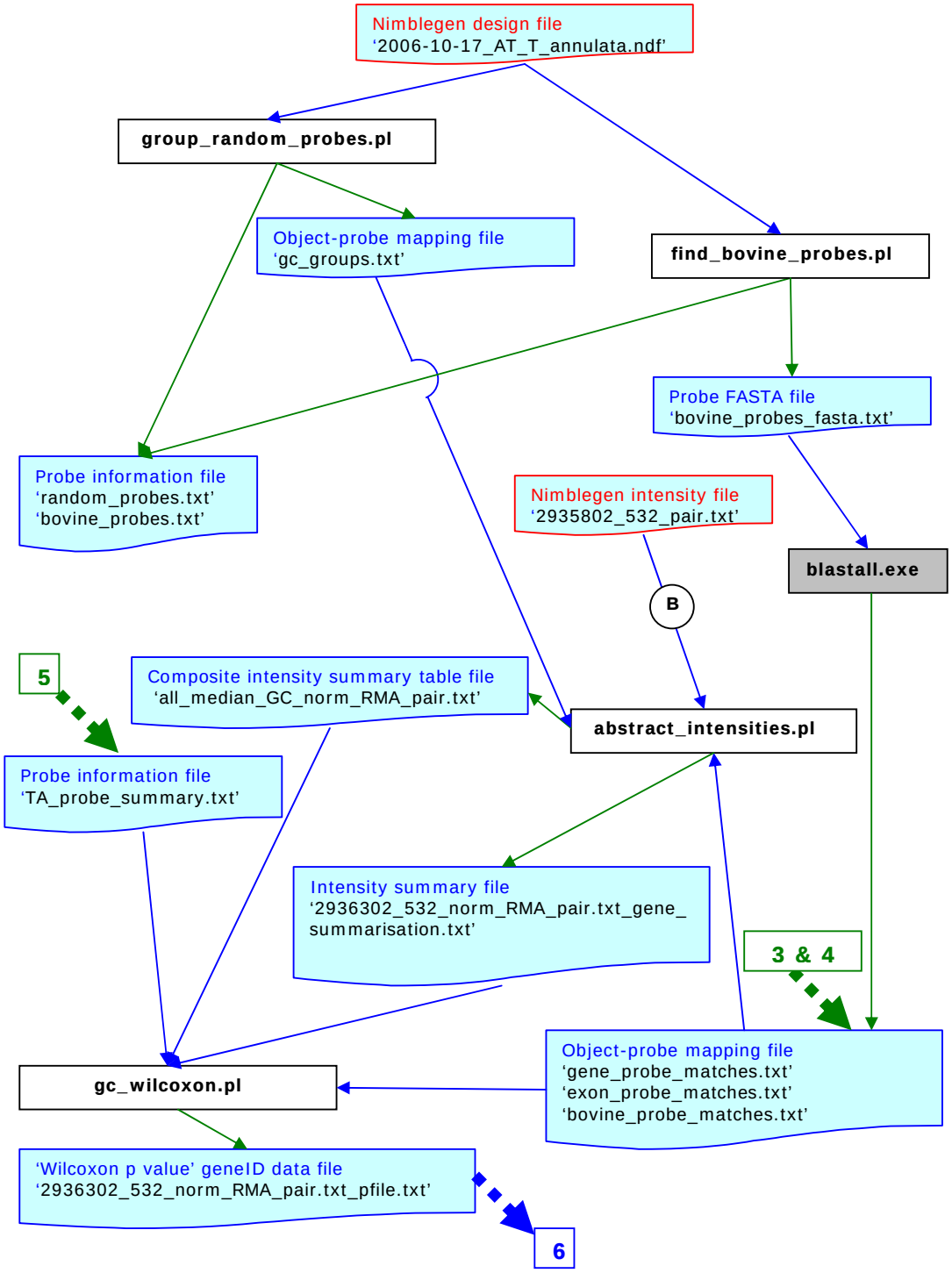


Figure 3. Sliding windows analysis

This flowchart represents the processes involved in performing sliding window and interval analysis (analogous to the Affymetrix application software, TAS). `sliding_window.pl` can perform both these functions, while `interval.pl` can be used to perform interval analysis on sequence graph files previously generated by `sliding_window.pl`. This chart also represents the creation of XYS files, used to investigate spatial trends on the array and the creation of `.sgr` files for viewing in the Integrated Genome Browser.

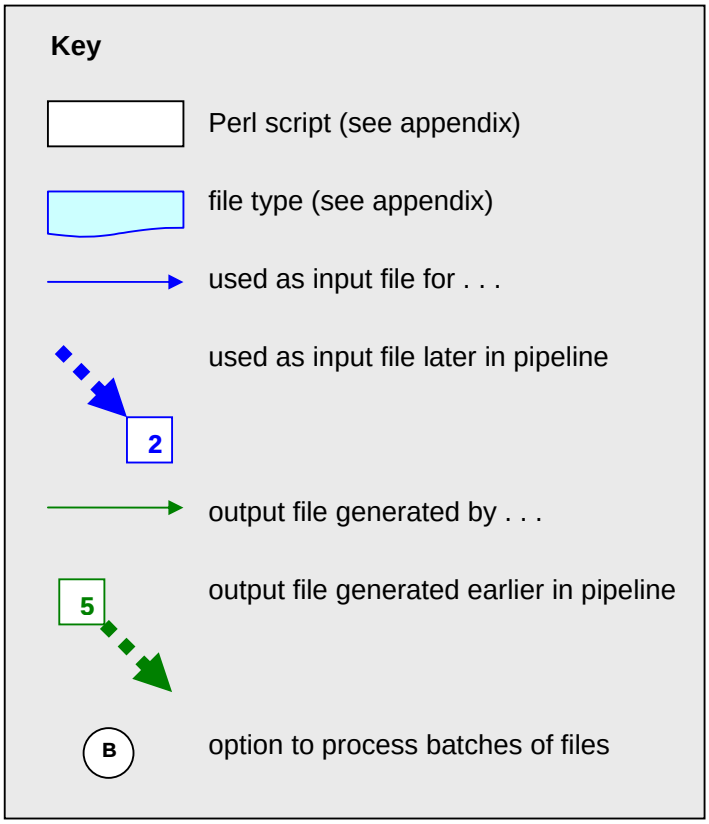
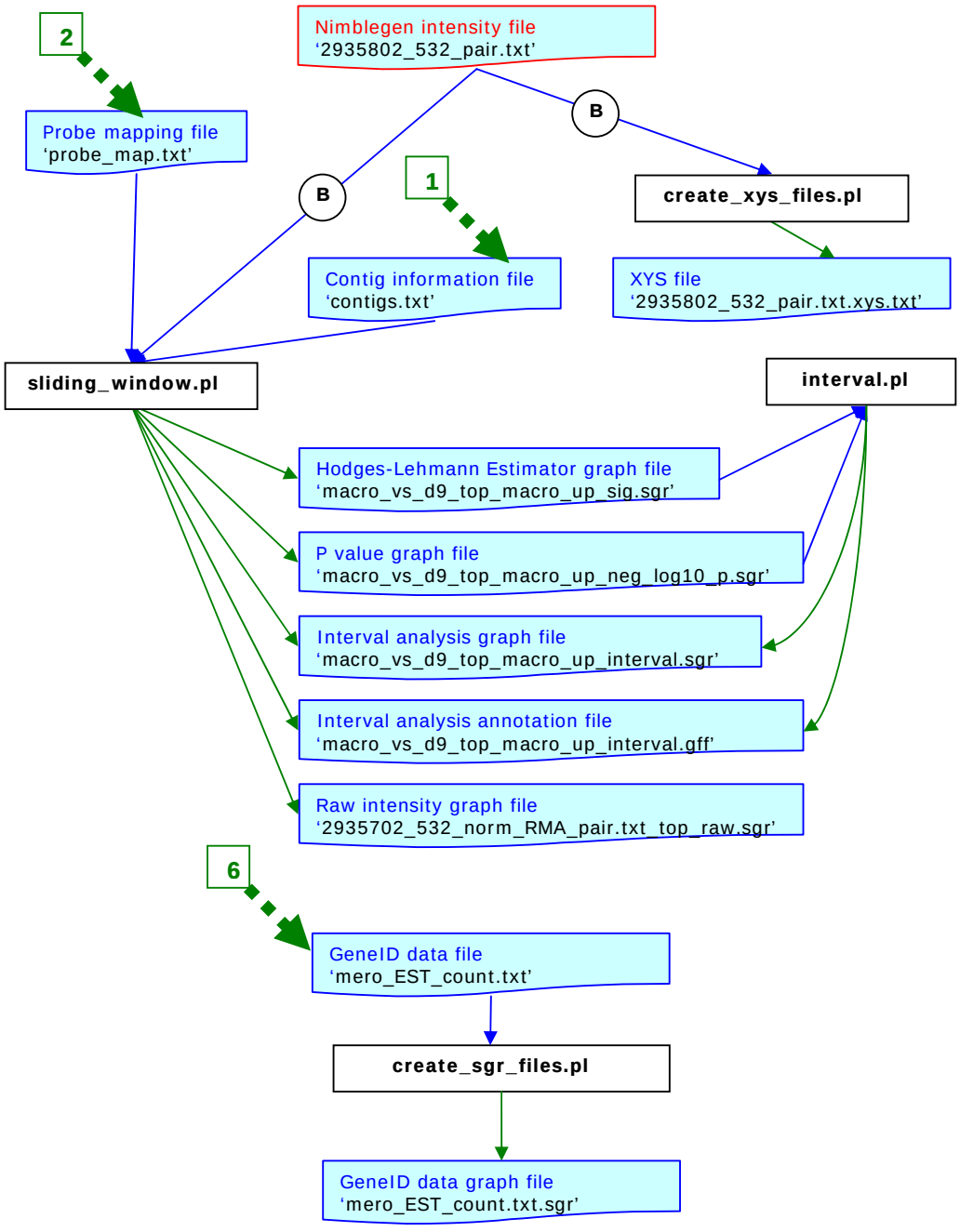


Figure 3. Sliding windows analysis



EMBL file (.embl)

Description

EMBL file format contains annotated chromosomal sequence data and is similar to Genbank format but does not require a series of header lines. A full definition can be found at http://www.ebi.ac.uk/embl/Documentation/User_manual/usrman.html.

File format

Space-delimited text file, with each line beginning with a two letter identifier, e.g. `FT` (feature table) and `RA` (author). In the feature table, data may correspond to `CDS` (coding sequence) with its associated locus information, e.g. `CDS complement(join(2326..3090,3122..3124))` and name / value pairs for feature data, e.g. `locus_tag="Tap370b08.q2ca38.02c"`.

Generated by scripts

- `N/A` (*T. annulata* chromosomal files generated by the Sanger Institute, July 2005)

Used by script

- `process_embl_files.pl`

Examples

- `CR940346.embl`

```
ID   CR940346; SV 1; linear; genomic DNA; HTG; INV; 5905 BP.
XX
AC   CR940346;
XX
PR   Project:153;
XX
DT   15-MAY-2005 (Rel. 83, Created)
DT   31-JUL-2005 (Rel. 84, Last updated, Version 4)
XX
DE   Theileria annulata mitochondrial genome DNA
XX
KW   HTG; HTGS_PHASE2.
XX
OS   Theileria annulata
OC   Eukaryota; Alveolata; Apicomplexa; Aconoidasida; Piroplasmida;
OC   Theileriidae; Theileria.
XX
RN   [1]
RP   1-5905
RX   DOI; 10.1126/science.1110418.
RX   PUBMED; 15994557.
RA   Pain A., Renauld H., Berriman M., Murphy L., Yeats C.A., Weir W.,
RA   Kerhornou A., Aslett M., Bishop R., Bouchier C., Cochet M., Coulson R.M.R.,
RA   Cronin A., de Villiers E.P., Fraser A., Fosker N., Gardner M., Goble A.,
RA   Griffiths-Jones S., Harris D.E., Katzer F., Larke N., Lord A., Maser P.,
RA   McKellar S., Mooney P., Morton F., Nene V., O'Neil S., Price C.,
RA   Quail M.A., Rabbinowitsch E., Rawlings N.D., Rutter S., Saunders D.,
RA   Seeger K., Shah T., Squares R., Squares S., Tivey A., Walker A.R.,
RA   Woodward J., Dobbelaere D.A.E., Langsley G., Rajandream M-A., McKeever D.,
RA   Shiels B., Tait A., Barrell B., Hall N.;
RT   "Genome of the host-cell transforming parasite Theileria annulata compared
RT   with T. parva";
RL   Science 309(5731):131-133(2005).
XX
RN   [2]
RP   1-5905
RA   Pain A., Renauld H., Murphy L., O'Neil S., Harris D.A., Quail M.A.,
RA   Berriman M., Hall N., Barrell B.G.;
RT   ;
RL   Submitted (31-MAR-2005) to the EMBL/GenBank/DDBJ databases.
RL   The Wellcome Trust Sanger Institute Genome Campus, Hinxton, Cambridge CB10
RL   1SA, UK
```

```
XX
FH   Key             Location/Qualifiers
FH
FT   source          1..5905
FT                   /organism="Theileria annulata"
FT                   /strain="Ankara isolate clone C9"
FT                   /mol_type="genomic DNA"
FT                   /db_xref="taxon:5874"
FT   CDS             <1..1516
FT                   /codon_start=2
FT                   /locus_tag="Tap370b08.q2ca38.01"
FT                   /product="cytochrome C oxidase subunit I (COX1 homologue),
FT                   putative"
FT                   /note="SMART pfam:COX1 (PF00115) at aa 38-497,
FT                   E()=5.90e-166"
FT                   /db_xref="GOA:Q4UJ69"
FT                   /db_xref="InterPro:IPR000883"
FT                   /db_xref="UniProtKB/Swiss-Prot:Q4UJ69"
FT                   /protein_id="CAI72674.1"
FT                   /translation="HTQIYVEKFRHIFVLWFLKNFKFYLVFEFVLSLFNSVSGNHKIIG
FT                   ISYLWLAYWFGMIGFYMSVLIRTELGMSGLKIITMDTLEIYNLLFTLHGLIMVFFNIMT
FT                   GLFGGIGNYLYPVLLGSCDVVYPRVNLYSLLLQPIGFVLVVSSVYLEIGSGTGWTLYPP
FT                   LSTSLSNIGIDLIIFGLLAAGIASTLSSINFITTFASIKTIGFVIDRISPAAWSIVLTS
FT                   FLLLLSLPVVTAVFLMVFFDRNHSTMFFESSNSGDPILYQHLFWFFGHPEVYIMILPGF
FT                   GIISLLLSTYTTKEMFGNQTMILAMGSIALLGCLVWGHHMYTSGLEADTRGYFTTVTIL
FT                   IALPTGNKIFNWVTTLQCVESIKSLGLILFAVLFIVNFVIGGTTGVVLGNAGLDVVLHD
FT                   TVYVVGHFHFVLSIGAIISLICFIVYIQRMLFGIILSNRLLSLMAPIFMIAVLFTFLPM
FT                   HFTGFSPLPRRIPDYPDEMWGWNFICTLGATMMLVLKLTVLFIISL"
FT   CDS             complement(join(2326..3090,3122..3124))
FT                   /locus_tag="Tap370b08.q2ca38.02c"
FT                   /product="cytochrome C oxidase subunit III (COX3
FT                   homologue), putative"
FT                   /note="SMART pfam:COX3 (PF00520) at aa 56-264,
FT                   E()=6.00e-05"
FT                   /db_xref="GOA:Q37679"
FT                   /db_xref="InterPro:IPR000298"
FT                   /db_xref="InterPro:IPR013833"
FT                   /db_xref="UniProtKB/Swiss-Prot:Q37679"
FT                   /protein_id="CAI72675.1"
FT                   /translation="MRNSVQSDLKYINIINISETLYLFYSTGLDTLEYIDSTYKNFIIM
FT                   YVNQHLLYGTTLKYLSVGEFFINSLTIFINGIRETMTSSTVIMYAIFGMFIFSEIMVFS
FT                   TFIWGFFHFRLSNPVMIIEVNLEAFLQISDVLNAGSILISLILQRIEERGYFEVDYMLE
FT                   RLILIGFIFLSFQGDEYSLVKSYINNHWVTLYFNVLTGLHSLHVYVGGIFALMQAFASE
FT                   NCGCQKDEDFNAGMYWHFVEIIWIALTMLLFLL"
FT   CDS             complement(4452..5543)
FT                   /locus_tag="Tap370b08.q2ca38.03c"
FT                   /product="cytochrome B, putative"
FT                   /note="SMART pfam:cytochrome_b_N (PF00033) at aa 1-190,
FT                   E()=3.00e-44; pfam:cytochrome_b_C (PF00032) at aa 246-351,
FT                   E()=5.50e-06; 1 transmembrane domain at aa 215-237"
FT                   /db_xref="GOA:Q4UJ67"
FT                   /db_xref="InterPro:IPR005797"
FT                   /db_xref="InterPro:IPR005798"
FT                   /db_xref="UniProtKB/Swiss-Prot:Q4UJ67"
FT                   /protein_id="CAI72676.1"
FT                   /translation="MNLFNSHLLSYMVPKNLNLNWNFGFILGILLVLQIISGLMLSFFY
FT                   VPAKGMAFESTLAVMLNICFGWFVRLYHSFGVSFYFFFMFLHIMKGMWYSSNHLPWSWY
FT                   SGVVIFVLSIATAFVGYVLPDGQMSFWGATVIGGLLKFFGKANVLIFGGQTVGPETLER
FT                   FFSIHVILPVIILLVVIFHLYVLHRDGSSNPLAVIDMLAIFRFHPVVLFSDIRFIVIVI
FT                   LLIGVQSGYGFISIFQADPDNSILSDPLNTPAHIIPEWYLLLFYATLKVFPTKVAGLLA
FT                   MAGMLELLVLLVESRYFKQTVSAMNYHRVWTTSSVPLVPVLFMLGSIGKMVVHVDLIAI
FT                   GTCVVLSVVLFIYKLLDSARVRA"
XX
SQ   Sequence 5905 BP; 2057 A; 883 C; 855 G; 2110 T; 0 other;
     gcatactcaa atatatgtcg aaaaattcag acatatattt gttttgtggt ttttaaaaaa        60
     ttttaagttt tatttagttt ttgagtttgt tttaagcttg tttaattctg tttcaggaaa       120

     ggttaagaac gccgcgaggc agttcgttcc ctatctagta tatttaaatc ctttataact      5820
     tacctagact aaataaaact taaaatttct taaaaaccac aaaacaaata tatgtctgaa      5880
     tttttcgaca tatatttgag tatgc                                           5905
//
```

Exon table file (.txt)

Description

An exon table file contains a list of exons from a single or multiple chromosomes (or contigs). The contig is indicated along with the strand on which the gene/exon lies, i.e. *top* (+) or *bottom* (-), together with the parental gene ID (i.e. locus) and the exon number for that gene. The start and end base positions are also indicated with the base position always with respect to the top strand. The first exon for each gene (i.e. exon number 1) begins with a start codon and the last exon ends with a stop codon. If a gene is encoded on the bottom strand, each exon start position is greater than its end position.

File format

Tab delimited text file with three header lines

```
Contig(tab)Strand(tab)Locus(tab)Exon no(tab)Start(tab)End(eol)
------(tab)------(tab)-----(tab)-------(tab)-----(tab)---(eol)
(eol)
```

Generated by script

- `Process_embl_files.pl`

Used by script

- `map_probes_to_exons.pl`

Example

- `exon_table.txt`

```
Contig    Strand  Locus                Exon no      Start        End
------    ------  -----                -------      -----        ---

CR940346  top     Tap370b08.q2ca38.01  1             1           1516
CR940346  bottom  Tap370b08.q2ca38.02c 1            3124         3122
CR940346  bottom  Tap370b08.q2ca38.02c 2            3090         2326
CR940346  bottom  Tap370b08.q2ca38.03c 1            5543         4452
CR940347  bottom  TA17495              1            704          1
CR940347  bottom  TA17500              1            4289         3054
CR940347  bottom  TA17500              2            2977         2687
```

Generic feature format file (.gff)

Description

Generic feature format files contain genomic annotation data. In the context of this study GFF files represent information defining genes, exons and putative transcripts. A full description of the current version GFF3 can be found at http://song.sourceforge.net/gff3.shtml. GTF (gene transfer format) is a specific subset of GFF and is used for defining exons and their relationship to a parental gene.

File format

Tab delimited text file (with optional header lines beginning `##`):
`contig(tab)source(tab)type(tab)start(end)score(tab)strand(tab)phase(tab)attributes(eol)`
Undefined filed are replaced with the "." character.

Generated by script

- `process_embl_files.pl`
- `interval.pl`

Used by application

- `Integrated Genome Browser (IGB)`

Examples

- `annotation.gtf`

```
CR940346  Genes  CDS   1     1516  .  +  .  locus_tag "Tap370b08.q2ca38.01";product
"cytochrome C oxidase subunit I (COX1 homologue), putative"
CR940346  Exons  EXON  1     1516  .  +  .  gene_id "Tap370b08.q2ca38.01";transcript_id
"Tap370b08.q2ca38.01"
CR940346  Genes  CDS   2326  3124  .  -  .  locus_tag "Tap370b08.q2ca38.02c";product
"cytochrome C oxidase subunit III (COX3 homologue), putative"
CR940346  Exons  EXON  3124  3122  .  -  .  gene_id "Tap370b08.q2ca38.02c";transcript_id
"Tap370b08.q2ca38.02c"
CR940346  Exons  EXON  3090  2326  .  -  .  gene_id "Tap370b08.q2ca38.02c";transcript_id
"Tap370b08.q2ca38.02c"
CR940346  Genes  CDS   4452  5543  .  -  .  locus_tag "Tap370b08.q2ca38.03c";product
"cytochrome B, putative"
CR940346  Exons  EXON  5543  4452  .  -  .  gene_id "Tap370b08.q2ca38.03c";transcript_id
"Tap370b08.q2ca38.03c"
```

- `macro_vs_mero_both_macro_up_interval.gff`

```
## Interval analysis using source files:  macro.txt vs  d9.txt
## i.e. macro vs mero
## Maximum gap: 114
## Minimum run: 190
## p threshold: 13
## Hodges-Lehmann signal threshold: 0
## Bandwidth: 150
## Step size: 38
## Start point: 1
##
CR940351  .  TRANSCRIPT  11780  12844  .  .  .  name "predicted transcript 1"
CR940350  .  TRANSCRIPT  5130   7600   .  .  .  name "predicted transcript 2"
CR940350  .  TRANSCRIPT  7866   9386   .  .  .  name "predicted transcript 3"
```

Contig information file (.txt)

Description

A contig information file contains the names and lengths of each contig to be analysed by a script or application software. This may correspond to the entire genome of an organism with each contig representing a complete chromosome or a fragment of a chromosome.

File format

Tab delimited text file with no header line:
`Contig name(tab)length(eol)`

Generated by script

- `process_embl_files.pl`

Used by script

- `sliding_window.pl`

Used by application

- `Integrated Genome Browser (IGB)`

Example

- `contigs.txt`

```
CR940346     5905
CR940347     2592520
CR940348     1979170
CR940349     15378
CR940350     11011
CR940351     12928
CR940352     1898942
CR940353     1842271
```

Probe design file (.txt)

Description

Eight probe design files were created for the *T. annulata* custom, tiling microarray experiments by NimbleGen in collaboration with Dr A. Ivens at the Sanger Institute, Hinxton. Each file corresponds to a contig from the July 2005 assembly of the genome. Each probe design file contains a list of probe IDs with corresponding nucleotide sequence, a pass / failure flag and if indicated, the reason for the failure. The probe ID identifies the chromosome / contig against which each probe was designed.

File format

Tab delimited text file with single header line:
`PROBE_ID(tab)PROBE_SEQUENCE(tab)TEST_RESULT(tab)FAILURE_REASON(eol)`

Generated by script

- `N/A` (generated by collaborators at NimbleGen / Sanger Institute)

Used by script

- `create_fasta_and_map.pl`

Example

- `CR940346.NimbleGen.txt`

```
PROBE_ID            PROBE_SEQUENCE      TEST_RESULT   FAILURE_REASON
CR940346_0000001    GCATACTCAAATATATGTCGAAAAATTCAGACATATATTTGTTTT  PASS
CR940346_0000046    CTTAAAATTTTTTAAAAACCACAAAACAAATATATGTCTGAATTT  PASS
CR940346_0000091    GTGGTTTTTAAAAAATTTTAAGTTTTATTTAGTTTTTGAGTTTGT  PASS
CR940346_0000136    AACAGAATTAAACAAGCTTAAAACAAACTCAAAAACTAAATAAAA  PASS
CR940346_0000181    TTTAAGCTTGTTTAATTCTGTTTCAGGAAATCATAAAATCATAGG  PASS
CR940346_0000226    TGCCAACCATAAATAAGATATTCCTATGATTTTATGATTTCCTGA  PASS
CR940346_0000271    AATATCTTATTTATGGTTGGCATATTGGTTTGGTATGATAGGATT  PASS
CR940346_0000316    TCTAATCAGCACACTCATATAAAATCCTATCATACCAAACCAATA  PASS
CR940346_0000361    TTATATGAGTGTGCTGATTAGAACTGAATTAGGTATGAGCGGGTT  PASS
CR940346_0000406    AGTATCCATAGTTATTATCTTTAACCCGCTCATACCTAATTCAGT  PASS
```

Probe mapping file (.txt)

Description

A probe mapping file links oligonucleotide probe IDs with genomic loci. The contig, strand, start and end position corresponding to each probe is indicated. If probes correspond to the bottom strand, the end position is greater than the start position. This type of file is used as a resource for mapping probes to exons using `map_probes_to_exons.pl` and for performing sliding windows analysis using `sliding_window.pl`.

File format

Tab delimited text file with single header line:
`ProbeID(tab)Contig(tab)Strand(tab)Start(tab)End(eol)`

Generated by script

- `create_fasta_and_map.pl`

Used by scripts

- `map_probes_to_exons.pl`
- `sliding_window.pl`

Example

- `probe_map.txt`

```
ProbeID             Contig Strand Start  End
CR940346_0000001    CR940346     top          1     45
CR940346_0000046    CR940346     bottom 67    23
CR940346_0000091    CR940346     top          46    90
CR940346_0000136    CR940346     bottom 112   68
CR940346_0000181    CR940346     top          91    135
CR940346_0000226    CR940346     bottom 157   113
```

Probe FASTA file (.txt)

Description

A probe FASTA file contains a list of oligonucleotide probe sequences in FASTA format. The FASTA header may simply contain the probe name (e.g. `bovine_probes_fasta.txt`). For the *T. annulata* probe FASTA file, `probe_fasta.txt`, the contig, start and end positions are also specified. Nucleotide position is always indicated relative to the 'top' strand and if the probe is designed to the bottom strand, then the value of the first position is greater than the second.

File format

Text file containing a list of sequences in FASTA format:

```
>probe name|contig name|start index..end index(eol)
nucleotide sequence(eol)
```

Generated by script

- `create_fasta_and_map.pl (probe_fasta.txt)` for *T. annulata* probes
- `find_bovine_probes.pl (bovine_probes_fasta.txt)`

Used by script

- `blastall.exe (bovine_probes_fasta.txt)`

Examples

- `probe_fasta.txt`

```
>CR940346_0000001|CR940346|1..45
GCATACTCAAATATATGTCGAAAAATTCAGACATATATTTGTTTT
>CR940346_0000046|CR940346|67..23
CTTAAAATTTTTTAAAAACCACAAAACAAATATATGTCTGAATTT
>CR940346_0000091|CR940346|46..90
GTGGTTTTTAAAAAATTTTAAGTTTTATTTAGTTTTTGAGTTTGT
>CR940346_0000136|CR940346|112..68
AACAGAATTAAACAAGCTTAAAACAAACTCAAAAACTAAATAAAA
>CR940346_0000181|CR940346|91..135
TTTAAGCTTGTTTAATTCTGTTTCAGGAAATCATAAAATCATAGG
>CR940346_0000226|CR940346|157..113
```

- `bovine_probes_fasta.txt`

```
>BTAU_0029656
GAGGAGTGAGACTGGACACGGGCTGTGGACTGTAACCTACTTGGG
>BTAU_0060166
GTTTTGATTTAATAAAAGATCCACGATATCACAAAGTAACTTGTT
>BTAU_0104311
CAGCTCCCACTCCCCTGGGGCCCCCTGGGCTCACCAATGGCCTCC
>BTAU_0089326
ACTAAAGAAAAAGGGCTAGACACTCGACGGCCAGAGGAGGCGGAA
>BTAU_0047701
GTCTTTAGCAATTACCCCCTGAGATTTTTGGAGGGGATTTTATTT
```

BLAT output file (.psl)

Description

A `.psl` file describes a series of BLAT-generated alignments in a dense, easily parsed text format. Each line corresponds to an alignment and each query may have one or more alignments associated with it. Further information on this file type can be found at http://genome.ucsc.edu/goldenPath/help/blatSpec.html.

File format

Tab-delimited text file with five header lines:

`matches` *integer*	Number of bases that match that aren't repeats
`misMatches` *integer*	Number of bases that don't match
`repMatches` *integer*	Number of bases that match but are part of repeats
`nCount` *integer*	Number of 'N' bases
`qNumInsert` *integer*	Number of inserts in query
`qBaseInsert` *integer*	Number of bases inserted in query
`tNumInsert` *integer*	Number of inserts in target
`tBaseInsert` *integer*	Number of bases inserted in target
`strand` *string*	+ or - for query strand, optionally followed by + or – for target strand
`qName` *string*	Query sequence name
`qSize` *integer*	Query sequence size
`qStart` *integer*	Alignment start position in query
`qEnd` *integer*	Alignment end position in query
`tName` *string*	Target sequence name
`tSize` *integer*	Target sequence size
`tStart` *integer*	Alignment start position in target
`tEnd` *integer*	Alignment end position in target
`blockCount` *integer*	Number of blocks in alignment. A block contains no gaps.
`blockSizes` *string*	Size of each block in a comma separated list
`qStarts` *string*	Start of each block in query in a comma separated list
`tStarts string`	Start of each block in target in a comma separated list

Generated by scripts

- `BLAT.EXE`
- `filter.pl` (using `BLAT.EXE`)

Used by script

- `filter.pl`

Example

- output.psl

```
psLayout version 3

match  mis-   rep.   N's    Q gap  Q gap  T gap  T gap  strand Q            Q
       Q      Q      T             T      T      T      block  blockSizes
       qStarts       tStarts
       match  match         count  bases  count  bases         name         size
       start  end    name          size   start  end    count
-----------------------------------------------------------------------------
-----------------------------------------------------------------------------
---
44     0      0      0      0      0      0      0      +      CR940346_0000001
       45     1      45     Tap370b08.q2ca38.01 1512   0      44     1      44,
       1,     0,
45     0      0      0      0      0      0      0      -      CR940346_0000046
       45     0      45     Tap370b08.q2ca38.01 1512   21     66     1      45,
       0,     21,
45     0      0      0      0      0      0      0      +      CR940346_0000091
       45     0      45     Tap370b08.q2ca38.01 1512   44     89     1      45,
       0,     44,
45     0      0      0      0      0      0      0      -      CR940346_0000136
       45     0      45     Tap370b08.q2ca38.01 1512   66     111    1      45,
       0,     66,
45     0      0      0      0      0      0      0      +      CR940346_0000181
       45     0      45     Tap370b08.q2ca38.01 1512   89     134    1      45,
       0,     89,
45     0      0      0      0      0      0      0      -      CR940346_0000226
       45     0      45     Tap370b08.q2ca38.01 1512   111    156    1      45,
       0,     111,
```

Filter summary file (.txt)

Description

A filter summary file contains a list of probes which match the genome under investigation. Each probe corresponds to a single line of data, which holds information regarding the probe name, the target gene, the quality of match to the target gene (i.e. Flag of Target), and the size of the probe (in bases). The number of matching bases and the number of aligned bases in the probe are also listed along with the identity to the target sequence and the orientation of the hit (+ is sense, - is antisense). 'BLAT matches' refers to the number of hits for a particular probe derived from a BLAT analysis and 'Filtered matches' is the number of hits follows the initial phase of filtration by `filter.pl`. Next, the flag representing the highest non-target cross-hybridisation candidate is listed along with the orientation of the matches. Overall, this file contains data on all probes which 'hit' the genome sequence, above a threshold value even though no one particular target may be defined.

File format

Tab-delimited text file with optional single header line:

```
ProbeID(tab)Target(tab)Flag of target(tab)Probe size(tab)
Matching bases(tab)Bases aligned(tab)Identity(tab)Strand(tab)
BLAT matches(tab)Filtered matches(tab)Highest cross-h flag(tab)
Strands of cross-h(eol)
```

Generated by script

- `filter.pl`

Used by application

- `MS Excel` (for further analysis)

Examples

- `filter_probe_summary.txt`

```
ProbeID     Target      Flag of target    Probe size  Matching bases
      Bases aligned     Identity    Strand      BLAT matches
      Filtered matches  Highest cross-h flag    Strands of cross-h
CR940346_0000001  Tap370b08.q2ca38.01     2     45    44    44    1
      +     1     1     0     NA
CR940346_0000046  Tap370b08.q2ca38.01     1     45    45    45    1
      -     1     1     0     NA
CR940346_0000091  Tap370b08.q2ca38.01     1     45    45    45    1
      +     1     1     0     NA
CR940346_0000136  Tap370b08.q2ca38.01     1     45    45    45    1
      -     1     1     0     NA
CR940346_0000181  Tap370b08.q2ca38.01     1     45    45    45    1
      +     1     1     0     NA
CR940346_0000226  Tap370b08.q2ca38.01     1     45    45    45    1
      -     1     1     0     NA
CR940346_0000271
```

Object-probe mapping file (.txt)

Description

An object-probe mapping file contains a list of objects together with number of probes and the names of the probes that represent each object. This is a generic file type, and several specific types have been defined corresponding to the following objects:

(A) exons
(B) genes
(C) sets of random probes

File format

Tab delimited text file with no header
`object name(tab)no. matching probes(tab)1st probe name(tab)`
`2nd probe name etc(eol)`

(A) Exon-probe mapping file

Description

An object-probe mapping file where each object represents an exon, named using the convention: `gene_exon number`.

Generated by script

- `map_probes_to_exons.pl`

Used by script

- `map_probes_to_genes.pl`
- `gc_wilcoxon.pl`
- `abstract_intensities.pl`

Example

- `exon_probe_matches.txt`

```
TA09730_3  1  CR940347_0053956
TA09730_4  1  CR940347_0053686
TA09730_5  4  CR940347_0053056  CR940347_0053146  CR940347_0053236  CR940347_0053326
TA09725_1  5  CR940347_0058456  CR940347_0058546  CR940347_0058636  CR940347_0058726
CR940347_0058816
TA09725_2  5  CR940347_0057106  CR940347_0057196  CR940347_0057286  CR940347_0057376
CR940347_0057466
TA09725_3  4  CR940347_0055846  CR940347_0055936  CR940347_0056026  CR940347_0056116
TA09725_4  1  CR940347_0055666
TA09725_5  2  CR940347_0055306  CR940347_0055396
TA09720_1  1  CR940347_0059221
TA09720_2  5  CR940347_0059401  CR940347_0059491  CR940347_0059581  CR940347_0059671
CR940347_0059761
TA09720_3  1  CR940347_0061021
```

(B) Gene-probe mapping file

Description

An object-probe mapping file where each object represents a gene.

Generated by scripts

- `map_probes_to_genes.pl`
- `blastall.exe` (for bovine genes, with additional processing)

Used by scripts

- `gc_wilcoxon.pl`
- `abstract_intensities.pl`

Examples

- `bovine_probe_matches.txt` (bovine genes)
- `gene_probe_matches.txt` (*T. annulata* genes)

```
Tap370b08.q2ca38.01    33      CR940346_0000001       CR940346_0000091
       CR940346_0000181       CR940346_0000271       CR940346_0000361
       CR940346_0000451       CR940346_0000541       CR940346_0000631
       CR940346_0000721       CR940346_0000811       CR940346_0000901
       CR940346_0000991       CR940346_0001081       CR940346_0001171
       CR940346_0001261       CR940346_0001351       CR940346_0001441
       CR940346_0001531       CR940346_0001621       CR940346_0001711
       CR940346_0001801       CR940346_0001891       CR940346_0001981
       CR940346_0002071       CR940346_0002161       CR940346_0002251
       CR940346_0002341       CR940346_0002431       CR940346_0002521
       CR940346_0002611       CR940346_0002701       CR940346_0002791
       CR940346_0002881
Tap370b08.q2ca38.02c   16      CR940346_0004726       CR940346_0004816
       CR940346_0004906       CR940346_0004996       CR940346_0005086
       CR940346_0005176       CR940346_0005266       CR940346_0005356
       CR940346_0005446       CR940346_0005536       CR940346_0005626
       CR940346_0005716       CR940346_0005806       CR940346_0005896
       CR940346_0005986       CR940346_0006076
Tap370b08.q2ca38.03c   23      CR940346_0008956       CR940346_0009046
       CR940346_0009136       CR940346_0009226       CR940346_0009316
       CR940346_0009406       CR940346_0009496       CR940346_0009586
       CR940346_0009676       CR940346_0009766       CR940346_0009856
       CR940346_0009946       CR940346_0010036       CR940346_0010126
       CR940346_0010216       CR940346_0010306       CR940346_0010396
       CR940346_0010486       CR940346_0010576       CR940346_0010666
       CR940346_0010756       CR940346_0010846       CR940346_0010936
TA17495 15      CR940347_0000046       CR940347_0000136       CR940347_0000226
       CR940347_0000316       CR940347_0000406       CR940347_0000496
       CR940347_0000586       CR940347_0000676       CR940347_0000766
       CR940347_0000856       CR940347_0000946       CR940347_0001036
       CR940347_0001126       CR940347_0001216       CR940347_0001306
```

(C) RandomGC-probe mapping file

Description

An object-probe mapping file where each object represents a set random probes matched by GC content. For example, RANDOM10 is the set of probes with a total of 10 G / C bases.

Generated by script

- `group_random_probes.pl`

Used by script

- `abstract_intensities.pl`

Example

- `GC_groups.txt`

```
RANDOM10	1	RANDOM00010601
RANDOM11	1	RANDOM00002168
RANDOM12	5	RANDOM00004605	RANDOM00006020	RANDOM00001495
	RANDOM00000245	RANDOM00004066
RANDOM13	25	RANDOM00000629	RANDOM00003942	RANDOM00014363
	RANDOM00001298	RANDOM00007951	RANDOM00012542
	RANDOM00005008	RANDOM00010929	RANDOM00011945
	RANDOM00002385	RANDOM00014195	RANDOM00003151
	RANDOM00010784	RANDOM00010696	RANDOM00002638
	RANDOM00000080	RANDOM00001866	RANDOM00015277
	RANDOM00011116	RANDOM00014245	RANDOM00003898
	RANDOM00014811	RANDOM00008361	RANDOM00001322
	RANDOM00009250
RANDOM14	25	RANDOM00000611	RANDOM00005967	RANDOM00008957
	RANDOM00005629	RANDOM00003097	RANDOM00002091
	RANDOM00011635	RANDOM00000407	RANDOM00013770
	RANDOM00008089	RANDOM00007639	RANDOM00010471
	RANDOM00007936	RANDOM00010582	RANDOM00001255
	RANDOM00010091	RANDOM00015291	RANDOM00014893
	RANDOM00000146	RANDOM00002373	RANDOM00008214
	RANDOM00014953	RANDOM00003857	RANDOM00007556
	RANDOM00001945
```

NimbleGen design file (.ndf)

Description

A NimbleGen design file contains information about NimbleGen custom microarray designs. Critically, it contains probe ID (column 13), probe sequence (column 6) and X and Y co-ordinates of the probe on the chip (columns 16 and 17). Further information can be found at http://www.nimblegen.com/products/software/nimblescan.html.

File format

Tab delimited text file with single header line:

```
PROBE_DESIGN_ID(tab)CONTAINER(tab)DESIGN_NOTE(tab)SELECTION_CRITERIA(tab)SEQ_I
D
(tab)PROBE_SEQUENCE(tab)MISMATCH(tab)MATCH_INDEX(tab)FEATURE_ID(tab)ROW_NUM(ta
b)
COL_NUM(tab)PROBE_CLASS(tab)PROBE_ID(tab)POSITION(tab)DESIGN_ID(tab)X(tab)Y(eo
l)
```

Generated by script

- `N/A` (provided by NimbleGen)

Used by scripts

- `group_random_probes.pl`
- `find_bovine_probes.pl`

Example

- `2006-10-17_AT_T_annulata.ndf`

```
PROBE_DESIGN_ID  CONTAINER  DESIGN_NOTE  SELECTION_CRITERIA  SEQ_ID
PROBE_SEQUENCE  MISMATCH  MATCH_INDEX  FEATURE_ID  ROW_NUM  COL_NUM
PROBE_CLASS  PROBE_ID  POSITION  DESIGN_ID  X  Y
4785_0255_0635  BLOCK1        AGTGACACAAATATCGAACAGTGGAAAATAAAACGCTTGATTCAA  0
64300252  64300252  635  255    CR940348_0984511  0  4785  255  635
4785_0257_0635  BLOCK1        ATCCATTGATCAATGATATTGTTTGAGATTGGTATGAATATTATT  0
64286327  64286327  635  257    CR940348_0357886  0  4785  257  635
4785_0259_0635  BLOCK1        ATATTTTTTTATATAATAATATTCATCTTTAGTATCGGTGGGTTC  0
64216417  64216417  635  259    CR940347_2396926  0  4785  259  635
4785_0261_0635  BLOCK1        GAGCACTTTGTTTTGGATTAACACCATTTGAAGTGTTAAGAACAC  0
64525271  64525271  635  261    CR940353_3275821  0  4785  261  635
4785_0263_0635  BLOCK1        GATATTGGAGCTGCTGTAGCAAGCACCGTAACGTACCATTACAGT  0
64414115  64414115  635  263    CR940352_2071621  0  4785  263  635
4785_0265_0635  BLOCK1        TCTATATTACTTGTTTTATATTTTACACTATGTTCCAGCATAGAG  0
64175278  64175278  635  265    CR940347_0545671  0  4785  265  635
4785_0267_0635  BLOCK1        ATTCAATGGTACTCTTTTATTTATTTATACTATTCTAACATCTAA  0
64257013  64257013  635  267    CR940347_4223746  0  4785  267  635
4785_0269_0635  BLOCK1        AGTATGGCACTAGTGTATGTCTTAATAGATTCAGGAGCTTCTGGG  0
64447667  64447667  635  269    CR940352_3581461  0  4785  269  635
4785_0271_0635  BLOCK1        CCTTTTCAGGTATTCCAACTTGAGTCAATATGTACTCAAAATCGA  0
64321868  64321868  635  271    CR940348_1957231  0  4785  271  635
4785_0273_0635  BLOCK1        TACATACAATGCTACTCATGTATACAACAATTCGTGACTAAATAG  0
64341895  64341895  635  273    CR940348_2858446  0  4785  273  635
4785_0275_0635  BLOCK1        GCAGAGTAAGTTTATAAATCTTAGTGTGTACAAGGTTTTTCCTTA  0
64333584  64333584  635  275    CR940348_2484451  0  4785  275  635
4785_0277_0635  BLOCK1        TATTTCCTTATGATAAAACGTTCTTGTCGTCCAACTGTTCCGTTT  0
64244784  64244784  635  277    CR940347_3673441  0  4785  277  635
4785_0279_0635  BLOCK1        TATGAGAAGTTATTGATTGGCTTTCTGAATGGTCTAAAATCATTT  0
64210859  64210859  635  279    CR940347_2146816  0  4785  279  635
4785_0281_0635  BLOCK1        GATTAGGATCCAAGTATTCGAAGAGTTTATGGTAACCATACAATT  0
64326756  64326756  635  281    CR940348_2177191  0  4785  281  635
```

Probe information file (.txt)

Description

A probe information file contains a subset of the information in the NimbleGen design file, including probe name, sequence, total number of G and C bases and sequence length.

File format

Tab delimited text file with no header line:
`probe ID(tab)probe sequence(tab)no. GC bases(tab)probe length(eol)`

Generated by scripts

- `create_fasta_and_map.pl (TA_probe_summary.txt)`
- `find_bovine_probes.pl (bovine_probes.txt)`
- `group_random_probes.pl (random_probes.txt)`

Used by scripts

- `blastall.exe (bovine_probes.txt)`
- `gc_wilcoxon.pl (TA_probe_summary.txt)`

Examples

- `TA_probe_summary.txt`

```
CR940346_0000001    GCATACTCAAATATATGTCGAAAAATTCAGACATATATTTGTTTT  11     45
CR940346_0000046    CTTAAAATTTTTTAAAAACCACAAAACAAATATATGTCTGAATTT  8      45
CR940346_0000091    GTGGTTTTTAAAAAATTTTAAGTTTTATTTAGTTTTTGAGTTTGT  8      45
CR940346_0000136    AACAGAATTAAACAAGCTTAAAACAAACTCAAAAACTAAATAAAA  9      45
CR940346_0000181    TTTAAGCTTGTTTAATTCTGTTTCAGGAAATCATAAAATCATAGG  12     45
CR940346_0000226    TGCCAACCATAAATAAGATATTCCTATGATTTTATGATTTCCTGA  13     45
```

- `bovine_probes.txt`

```
BTAU_0029656  GAGGAGTGAGACTGGACACGGGCTGTGGACTGTAACCTACTTGGG  26     45
BTAU_0060166  GTTTTGATTTAATAAAAGATCCACGATATCACAAAGTAACTTGTT  12     45
BTAU_0104311  CAGCTCCCACTCCCCTGGGGCCCCCTGGGCTCACCAATGGCCTCC  33     45
BTAU_0089326  ACTAAAGAAAAAGGGCTAGACACTCGACGGCCAGAGGAGGCGGAA  24     45
BTAU_0047701  GTCTTTAGCAATTACCCCCTGAGATTTTTGGAGGGGATTTTATTT  17     45
BTAU_0026821  GAAGATGTTGCCAAAATAGCTGCTGAAACAATGAACAACTACCTT  17     45
```

- `random_probes.txt`

```
RANDOM00010708  TTATCAACCGAAGACAGAGTTGAGGGTAAAATTTCCGTTTGTGTCCTTAA  19  50
RANDOM00003800  GGAGCGCAGCAAACCGCATGGGATATGACACGATTCAGTATTACAATCAC  24  50
RANDOM00006319  CTACAGCTAAATGCAGTTAGACCAACATTGGGGCTCAAGGCTCACCAGAC  25  50
RANDOM00005362  ACAAAATCTTTACATTCACTATATCGCAGGCCAAATTATATATTGCTTCT  15  50
RANDOM00005050  TTCCGAATGCTCGGGATCGCCGGACAGTTAGTGGGAGTCCGAAGATATCA  27  50
RANDOM00013735  CCCTGTGAATATGCATACAGCAGATATGTCAAATCGATATACGGTTGAAG  20  50
RANDOM00010868  CTCAGTACTTTAGACTTTAAGACCAACGAAATTAACCGAAATTGCGCAAG  19  50
```

NimbleGen intensity 'pair' file (.txt)

Description

A NimbleGen 'pair' file contains the intensity values for a single hybridisation and is generated using NimbleScan software. NimbleScan is a proprietary NimbleGen package designed for automated placement of a design-specific grid on NimbleGen array images, extraction of the corresponding feature intensity raw values, linkage of the raw intensity values with the corresponding probe description parameters, and generation of analysis reports for expression, ChIP-chip and resequencing arrays. Typically, intensity values are not $\log_2$ transformed and the data may or may not have been normalised using RMA (usually indicated in the document name). Further information can be found at http://www.nimblegen.com/products/software/nimblescan.html. Critically, each line of the file contains a probe ID (column 4), x co-ordinate (column 6) and y co-ordinate (column 7) together with a corresponding intensity value (column 10). No mis-matched probes were used in the *T. annulata* microarray.

File format

Tab delimited text file with a single header line:

```
IMAGE_ID(tab)GENE_EXPR_OPTION(tab)SEQ_ID(tab)PROBE_ID(tab)POSITION(tab)
X(tab)Y(tab)MATCH_INDEX(tab)SEQ_URL(tab)PM(tab)MM(eol)
```

Generated by script

- `N/A` (Nimblescan software)

Used by scripts

- `abstract_intensities.pl`
- `create_xys.pl`
- `sliding_window.pl`

Example

- `2936302_532_norm_RMA_pair.txt`

```
IMAGE_ID  GENE_EXPR_OPTION  SEQ_ID  PROBE_ID  POSITION  X  Y  MATCH_INDEX
SEQ_URL  PM  MM
2936302_532  BLOCK1  BTAU_0000001  0  485  787  64160375    63.00  0.00
2936302_532  BLOCK1  BTAU_0000046  0  374  684  64160376    40.57  0.00
2936302_532  BLOCK1  BTAU_0000091  0  676  868  64160377    19.36  0.00
2936302_532  BLOCK1  BTAU_0000136  0  313  207  64160378    19.58  0.00
2936302_532  BLOCK1  BTAU_0000181  0  29   803  64160379    13.38  0.00
2936302_532  BLOCK1  BTAU_0000226  0  108  272  64160380    20.54  0.00
2936302_532  BLOCK1  BTAU_0000271  0  153  491  64160381    25.85  0.00
2936302_532  BLOCK1  BTAU_0000316  0  223  633  64160382    24.07  0.00
2936302_532  BLOCK1  BTAU_0000361  0  139  467  64160383    20.48  0.00
2936302_532  BLOCK1  BTAU_0000406  0  758  456  64160384    12.89  0.00
2936302_532  BLOCK1  BTAU_0000451  0  100  886  64160385    24.90  0.00
2936302_532  BLOCK1  BTAU_0000496  0  532  44   64160386    20.17  0.00
2936302_532  BLOCK1  BTAU_0000541  0  69   807  64160387    19.90  0.00
2936302_532  BLOCK1  BTAU_0000586  0  718  862  64160388    12.72  0.00
2936302_532  BLOCK1  BTAU_0000631  0  657  467  64160389    16.49  0.00
2936302_532  BLOCK1  BTAU_0000676  0  198  708  64160390    25.16  0.00
```

Intensity summary file (.txt)

Description

An intensity summary file holds data for a particular hybridisation directly corresponding to an object-probe mapping file. Each line represents a summarisation of the intensity data for the object together with the individual probe values. It represents a parallel file to the intensity summary file, with the five extra columns with summary statistics, i.e. median, minimum, maximum, upper quartile and lower quartile of the probe intensity values.

File format

Tab delimited text file with no header:

```
object name(tab)no. of matching probes(tab)median value(tab)
minimum value(tab)maximum value(tab)upper quartile(tab)
lower quartile(tab)probe 1 intensity(tab)probe 2 intensity etc.(eol)
```

Generated by scripts

- `abstract_intensities.pl`

Used by script

- `gc_wilcoxon.pl`

Used by applications

- `MS Excel` (for preliminary analysis incl. formatting for rank products)

Examples

- `2936302_532_norm_RMA_pair.txt_gene_summarisation.txt`

```
Tap370b08.q2ca38.01 33     10452.16      190.98 43765.56      6248.55
      16083.03      190.98 624.62 4195.41       6342.89       15411.60
      2302.93       5179.22       10452.16      10666.87      1708.00
      21078.70      15775.34      31571.95      8339.96       43765.56
      18734.48      6248.55       31481.69      6891.58       4628.54
      13023.62      31689.05      33791.85      6895.61       10195.02
      16083.03      36971.62      9479.24       2468.51       29684.11
      9776.27       14908.99      11295.59
Tap370b08.q2ca38.02c       16     4195.035      91.15  17923.74      2429.80
      7760.99       7183.38       14371.07      17923.74      2429.80
      2396.92       4647.30       3619.21       3655.84       10017.63
      12681.54      7760.99       159.86 91.15  1947.90       4086.05
      4304.02
Tap370b08.q2ca38.03c       23     9725.06       1765.72       28368.30
      5179.91       13428.75      5699.01       11962.44      14075.55
      4184.55       8928.03       5996.52       10395.11      13638.41
      3452.65       5179.91       24551.39      1765.72       13264.33
      15792.82      9725.06       16010.98      13428.75      7347.74
      4742.80       28368.30      8700.69       11658.08      3278.04
```

Composite intensity summary table file (.txt)

Description

A composite intensity summary table file contains the median value for all probes corresponding to a particular object (one type of object per file) in a number of array hybridisations. The object may be a gene, exon or a set of random probes with equivalent GC content. These files hold summarisation data for large number of objects (in rows, row 2 onwards) and several hybridisations (in columns, column 2 onwards).

File format

Tab delimited text file with single header line:

```
Object(tab)filename 1(tab)filename 2 etc(eol)
Objectname(tab)median GC score(tab)median GC score etc(eol)
```

Generated by scripts

- `abstract_intensities.pl`

Used by script

- `gc_wilcoxon.pl (all_median_GC_norm_RMA_pair.txt)` as lookup table
- `MS Excel` (for preliminary analysis incl. formatting for rank products)

Examples

- `all_median_gene.txt`

```
Object  2934102_532_norm_RMA_pair.txt   2934802_532_norm_RMA_pair.txt
        2935102_532_norm_RMA_pair.txt   2935602_532_norm_RMA_pair.txt
        2935702_532_norm_RMA_pair.txt
Tap370b08.q2ca38.01     16618.02    11593.38    29385.34    10740.49    11954.58
Tap370b08.q2ca38.02c    9027.725    7039.02     20756.845   6009.48     7153.405
Tap370b08.q2ca38.03c    17499.04    14130.26    35086.76    11973.98    13559.82
TA17495                 7047.5      13657.96    10983.89    24387.64    15751.7
TA17500                 2536.35     6362.98     10025.39    16497.07    6159.52
TA17505                 3591.02     6017.18     2970.2      6538.24     6824.61
TA09760                 11193.67    11686.945   7561.865    7893.12     11885.08
```

- `all_median_GC_norm_RMA_pair.txt`

```
Object  2934102_532_norm_RMA_pair.txt   2934802_532_norm_RMA_pair.txt
        2935102_532_norm_RMA_pair.txt   2935602_532_norm_RMA_pair.txt
        2935702_532_norm_RMA_pair.txt   2935802_532_norm_RMA_pair.txt
        2935902_532_norm_RMA_pair.txt   2936002_532_norm_RMA_pair.txt
        2936102_532_norm_RMA_pair.txt   2936302_532_norm_RMA_pair.txt
RANDOM10        845.80  23.10   20.73   17.83   14.45   1135.29 95.91   45.96   14.97   26.01
RANDOM11        18.18   17.42   18.07   17.16   17.42   128.90  17.50   22.82   42.09   16.76
RANDOM12        16.71   17.83   19.74   16.32   20.28   36.48   20.29   427.18  20.70   14.48
RANDOM13        39.01   17.79   20.77   14.92   17.64   71.73   18.80   35.10   17.61   16.02
RANDOM14        66.93   20.12   22.70   16.69   16.91   150.89  19.70   67.63   21.62   17.43

…

RANDOM31        212.01  20.54   68.40   19.63   21.22   694.86  79.13   406.19  32.37   18.38
RANDOM32        230.83  20.45   61.85   17.15   21.13   584.76  78.04   412.22  28.89   19.42
RANDOM33        195.03  18.635  113.9   20.69   17.89   572.14  114.265 404.585 24.935  21.405
RANDOM34        421.87  22.905  29.5    20.005  26.81   821.625 75.68   761.035 74.985  23.785
RANDOM35        15.68   15.38   36.01   17.67   16.37   93.16   50.40   48.64   15.74   13.80
```

XYS file (.txt)

Description

An XYS file contains the raw intensity data corresponding to the results of a single array hybridisation and is derived from a NimbleGen intensity file. This information may be analysed to detect spatial trends in the dataset.

File format

Tab delimited text file with no header line:
`X co-ordinate(tab)Y co-ordinate(tab)intensity value(eol)`

Generated by script

- `create_xys_file.txt`

Used by script

- `SMIDA / Bioconductor (R)`

Example

- `2934802_532_pair.txt.xys.txt`

```
1	1	2707.78
1	2	0.0001
1	3	3583.56
1	4	0.0001
1	5	5139.11
1	6	0.0001
1	7	95.89
1	8	0.0001
1	9	1073.22
1	10	0.0001
1	11	22759.22
1	12	0.0001
```

GeneID data file (.txt)

Description

A GeneID data file contains a list of genes with a corresponding numerical value. The numerical values may represent a diverse collected of information types, with a single 'type' of information represented within one file, e.g. EST hits, MPSS data or $d_N d_S$ values / nucleotide identity / amino acid identity with respect to orthologous genes in another genome. These files may be converted to .sgr graph files using the script `create_sgr_files.pl`. The script `gc_wilcoxon.pl` generates a GeneID data file with the first and second columns representing gene ID and a Paired-Wilcoxon -10log10(*p value)* with additional columns representing *Wplus*, *Wminus*, *N test pairs* and *p value* with an optional header line present.

File format

Tab delimited text file with optional header line:
`GeneID(tab)numerical value(EOL)`

Generated by scripts

- `gc_wilcoxon.pl`
- *`many files already available from EST / MPSS experiments and comparative genomic analysis`*

Used by script

- `create_sgr_files.pl`

Examples

- `piro_EST_count.txt`

```
TA21355      1
TA21430      1
TA21460      1
TA21500      1
TA02480      2
TA02485      2
TA02685      2
TA02760      2
TA02890      2
```

- `TA_dnds.txt`

```
TA04975      0.8525
TA07165      0.7933
TA11940      0.6721
TA04440      0.6714
TA15095      0.6609
TA07870      0.6339
TA05200      0.6236
TA06815      0.5958
TA05320      0.5754
```

Sequence graph file (.sgr)

Description

Sequence graph files contain scores corresponding to base co-ordinates of a contig or chromosome. These can be viewed using IGB and contain information such as:

- *p value*, -10log10(*p value*) or *Hodges-Lehmann estimator* from sliding windows analysis (for bar / line graph or heat-map display)
- Start and stop base positions of detected regions from interval analysis (for stair-step display)
- Raw intensity data (for bar graph / heat-map display)
- *p value* or -10log10(*p value*) from GC-Wilcoxon analysis (for bar / line graph or heat-map display)
- EST counts, MPSS data etc from geneID data files (for stair-step display)

File format

Tab delimited text file with no header line:

```
Contig name(tab)base position(tab)score value(EOL)
```

Generated by scripts

- `create_sgr_files.pl` (from geneID data files)
- `sliding_window.pl`
- `interval.pl`

Used by script

- `interval.pl`

Used by application

- `Integrated Genome Browser (IGB)`

Examples

- `macro_vs_mero_both_mero_up_neg_log10_p.sgr`

```
CR940351     76           22.1473745540462
CR940351     114          28.5189556181681
CR940351     152          51.2641448276588
CR940351     190          44.3645688436347
CR940351     228          49.4124097051911
CR940351     266          49.4124097051911
```

- `piro_EST_count.sgr`

```
CR940352     111347 1
CR940352     111811 1
CR940352     111346 0
CR940352     111812 0
CR940352     111174 1
CR940352     111275 1
CR940352     111173 0
```

process_embl_files.pl

Description

This script creates an exon table file, a GFF annotation file and a contig information file based on genomic information contained in a single file or collection of files in embl format. The file name of the embl file (with .embl extension) is designated as the contig or chromosome identifier in exon table and GFF files. The script reads coding sequence information, with each CDS uniquely identified using `locus_tag` name / value data. The gene product information is also parsed. Optionally, a separate FASTA files is created with nucleotide sequence data for each contig.

Command line options

`-b, -batch`=*`FILENAME`*

Batch file containing names of embl files

`-em, -emblfile`=*`FILENAME`*

Name of the EMBL file to be processed (if batch file also specified, this file is processed last)

`-ex, -exonfile`=*`FILENAME`*

Name of the exon table file to be generated (default is `exon_table.txt`)

`-g, -gfffile`=*`FILENAME`*

Name of the GFF annotation file to be generated (default is `annotation.gtf`)

`-c, -contigfile`=*`FILENAME`*

Name of the contig information file to be generated (default is `contigs.txt`)

`-f, -fasta`

If specified, a separate FASTA file is generated for each contig

Notes

Either **`-batchfile`** or **`-emblfile`** must be specified.

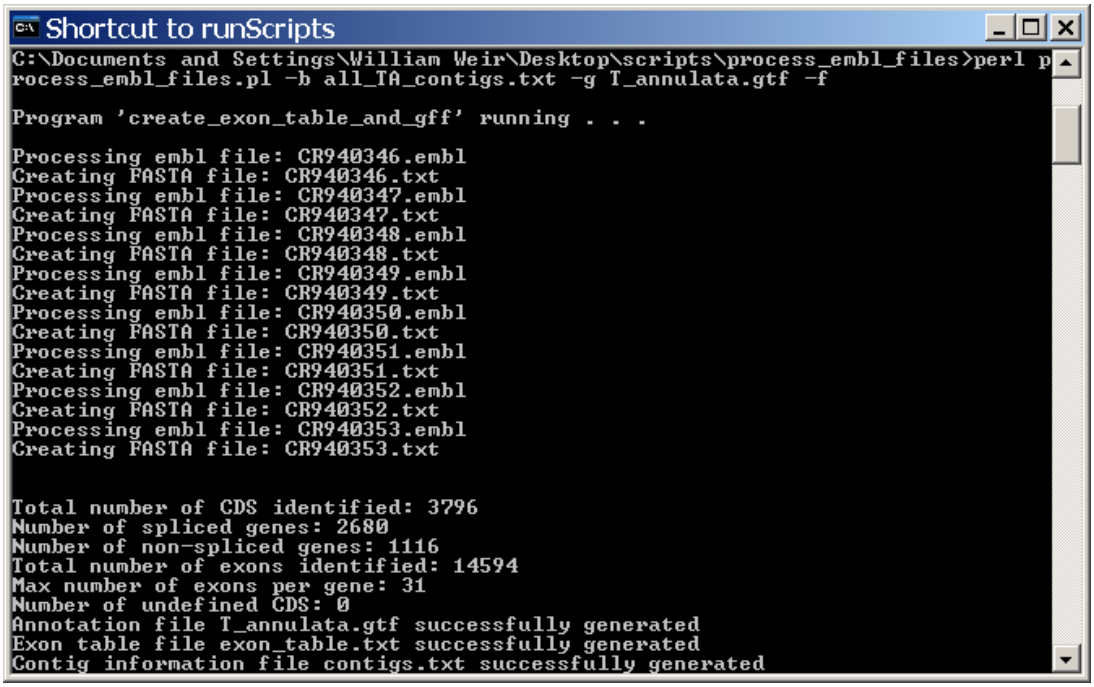
Example

- `perl process_embl_files.pl –b all_TA_contigs.txt -g T_annulata.gtf -f`

  From all the files listed in `all_TA_contigs.txt` the script generates:

  - an exon table file called `exon_table.txt`
  - a GTF file called `T_annulata.gtf`
  - a contig file called `contigs.txt`
  - a series of eight FASTA files, `CR940346.txt` etc.

  Screen-shot on next page


Shortcut to runScripts
C:\Documents and Settings\William Weir\Desktop\scripts\process_embl_files>perl p
rocess_embl_files.pl -b all_TA_contigs.txt -g T_annulata.gtf -f
Program 'create_exon_table_and_gff' running . . .
Processing embl file: CR940346.embl
Creating FASTA file: CR940346.txt
Processing embl file: CR940347.embl
Creating FASTA file: CR940347.txt
Processing embl file: CR940348.embl
Creating FASTA file: CR940348.txt
Processing embl file: CR940349.embl
Creating FASTA file: CR940349.txt
Processing embl file: CR940350.embl
Creating FASTA file: CR940350.txt
Processing embl file: CR940351.embl
Creating FASTA file: CR940351.txt
Processing embl file: CR940352.embl
Creating FASTA file: CR940352.txt
Processing embl file: CR940353.embl
Creating FASTA file: CR940353.txt
Total number of CDS identified: 3796
Number of spliced genes: 2680
Number of non-spliced genes: 1116
Total number of exons identified: 14594
Max number of exons per gene: 31
Number of undefined CDS: 0
Annotation file T_annulata.gtf successfully generated
Exon table file exon_table.txt successfully generated
Contig information file contigs.txt successfully generated

map_probes_to_exons.pl

Description

This script generates an exon-mapping file, linking each exon to a collection of either 'sense' or 'anti-sense' probes. This script integrates data read from a probe-mapping file and an exon table file and only probes which lie completely within an exon are included.

Command line options

`-a, -antisense`

Switch to instruct the script to identify anti-sense probe matches (default is 'sense' matches)

`-exonf, -exonfile`=*`FILENAME`*

Name of the input file (an exon table file) linking exons to chromosomal locations (default is `exon_table.txt`).

`-probef, -probefile`=*`FILENAME`*

Name of the input file (a probe-mapping file) linking probes to chromosomal locations (default is `probe_map.txt`)

`-m, -mapfile`=*`FILENAME`*

Name of the output file (an exon-mapping file) matching probes to exons (default is `exon_probe_matches.txt`)

`-exonh, -exonheader=INTEGER`

Number of header lines in the exon-chromosome location input file (default is 3)

`-probeh, -probeheader=INTEGER`

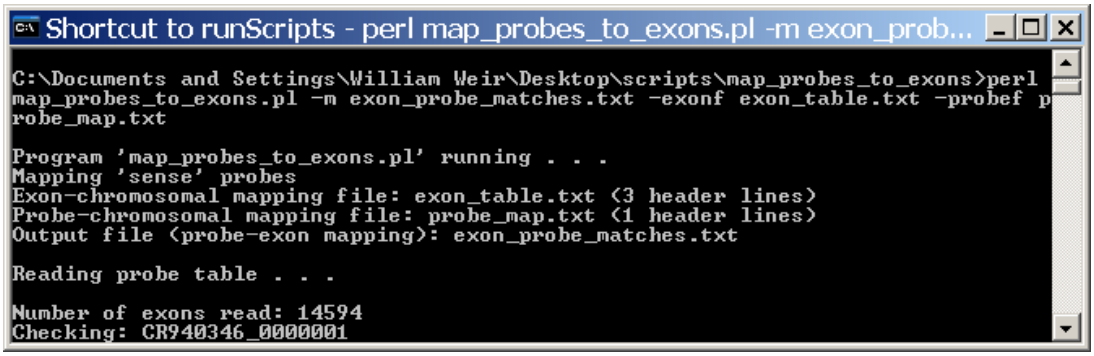
Number of header lines in the probe-chromosome location input file (default is 1)

Examples

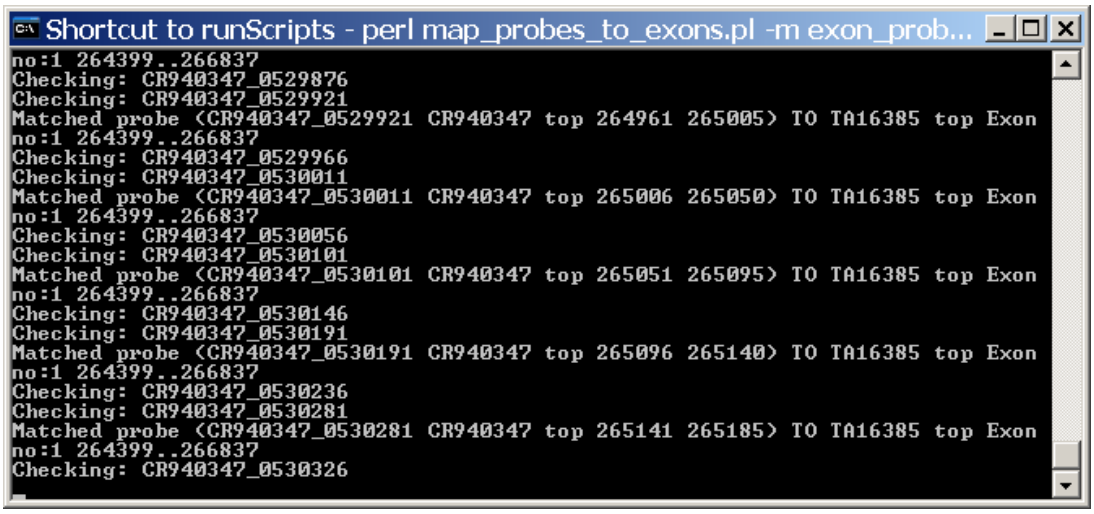
- `perl map_probes_to_exons.pl –m exon_probe_matches.txt –exonf exon_table.txt –probef probe_map.txt`

  From all the exons listed in `exon_table.txt` the script generates an exon-mapping file called `exon_probe_matches.txt` using data from `probe_map.txt`. Only matching sense probes are included in the exon-mapping file.

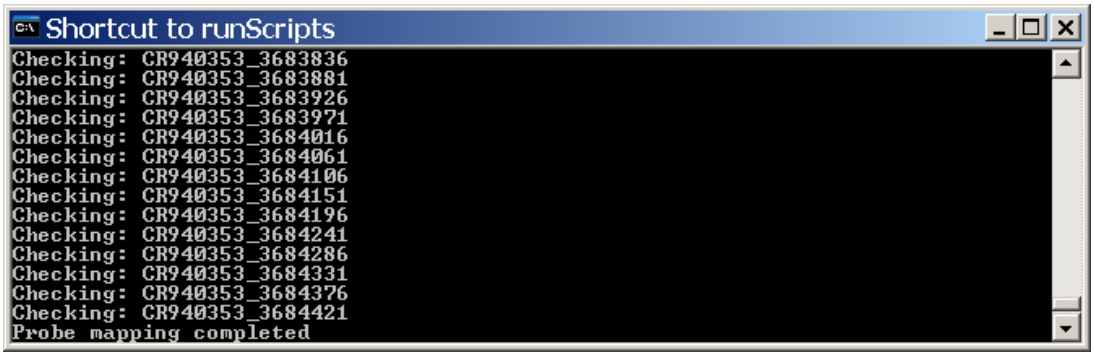
  Screen shots on next page



. . .



. . .



- `perl map_probes_to_exons.pl –a –m AS_exon_probe_matches.txt –exonf exon_table.txt –probef probe_map.txt`

  As above, except antisense probes are mapped to exons and the exon-mapping file is named `AS_exon_probe_matches.txt`.

map_probes_to_genes.pl

Description

This script generates a gene-mapping file, linking each gene to a collection of probes based on data read from an exon-mapping file.

Command line options

`-e, -exonfile`=*`FILENAME`*

Name of the input file (an exon-mapping file) linking probes to exons (default is `exon_probe_matches.txt`)

`-g, -genefile`=*`FILENAME`*

Name of the output file (a gene-mapping file) linking probes to genes (default is `gene_probe_matches.txt`)

`-h, -header`=*`INTEGER`*

Number of header lines in the exon-mapping file (default is 0)

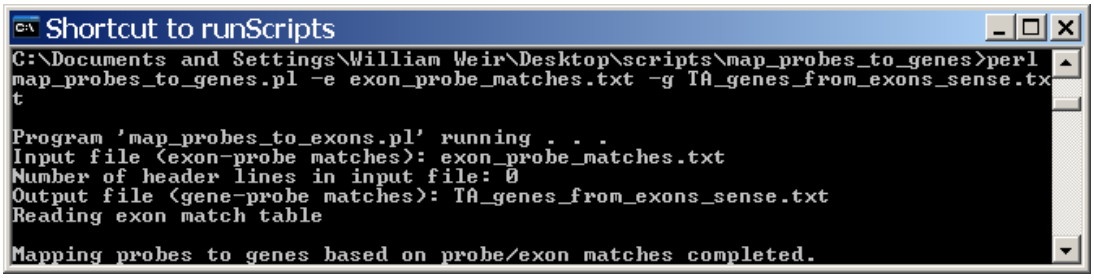
Notes

The contents of the exon-mapping file determine whether the generated gene-mapping file represents sets of ‘sense’ or ‘anti-sense’ probes.

Examples

- `perl map_probes_to_genes.pl –e exon_probe_matches.txt –g TA_genes_from_exons_sense.txt`

  Using all the probe-exon matches listed in `exon_probe_matches.txt`, the script generates a gene-mapping file called `TA_genes_from_exons_sense.txt`.



filter.pl

Description

This script generates an object-probe mapping file based on the output from BLAT analysis (http://www.kentinformatics.com/products.html). BLAT is the 'BLAST-Like Analysis Tool' and is used to rapidly generate DNA alignments. It is more accurate and much faster than existing tools since it keeps an index of the entire genome in memory. Windows executables can be found at http://hgwdev.cse.ucsc.edu/~kent/exe/windows/ and documentation is at http://genome.ucsc.edu/goldenPath/help/blatSpec.html. `filter.pl` was developed using `BLAT.EXE` from blatSuite.33 on Windows XP.

BLAT can be invoked from within `filter.pl`, as long as `BLAT.EXE` is in the same directory as `filter.pl` or if the path to `BLAT.EXE` is already specified in the Environmental variable, PATH. The `.psl` results file generated by BLAT is filtered by this script to produce a probe filter file, which contains a single line of data corresponding to each probe (i.e. BLAT query). The data written to the probe filter file is then summarised in the form of an object-probe mapping file. This file directly maps each gene to a list of probes and excludes non-specific probes which have a high potential to cross-hybridise. If required, the script can trim probes and gene names using | as a delimiter and can identify valid genes based on a string of identifier text in the gene name (the first characters).

A flagging system is used to classify the similarity of a probe with a gene and is based on the system used by ProbeLynx mapping tool, which can be found at http://koch.pathogenomics.ca/probelynx/. Flag values vary between 1 and 5, with 1 representing a perfect match and 5 representing a poor match. Two threshold flag values can be set as criteria for accepting probes based on probe-target similarity and the probe-cross-hybridisation candidate similarity. Probes may be mapping using (A) only 'sense' probes, (B) only 'antisense' probes or (C) 'sense' and 'antisense' probes.

Command line options

`-b, -blat`

If specified, perform a fresh BLAT analysis

`-nop, -nopipe`

If specified, probe names / gene names are not truncated based on the | symbol

`-noh, -noheader`

If specified, a header line is not written to the probe filter file

`-ps, -pslfile`=*`FILENAME`*

Name of the `.psl` file, produced by BLAT analysis (default is `output.psl`)

`-pf, -pffile`=*`FILENAME`*

Name of the probe filter file (default is `filtered_probes.txt`)

`-om, -omfile`=*`FILENAME`*

Name of the object-probe mapping file created

`-genomef, -genomefile`=*`FILENAME`*

Name of the FASTA file containing genomic CDS information

-pr, -probefile=*FILENAME*

Name of the FASTA file containing probe sequence information

-h, -headlinespsl=*INTEGER*

Number of header lines in `.psl` file (default is 5)

-genomei, -genomeid=*STRING*

The identifier for beginning of the name of the genes belonging to a target genome, e.g. '`TA`' for *T. annulata* (genes are named TA12345, TA13810 etc.) (default is no identifier, i.e. include all mapped genes)

-ma, -maxtargetflag=*INTEGER*

Only probes with target genes with this flag number or lower will be included in the mapping process (default is 2)

-mi, -mincrossflag=*INTEGER*

Only probes with target genes with cross-hybridisation candidates with this flag or higher (or no flag) will be included in the mapping process (default is 3)

-or, -orientation=*STRING*

Only probes mapping to this orientation of the target gene will be included in the mapping process. Can be '`sense`', '`antisense`' or '`both`' (default is '`sense`')

Notes

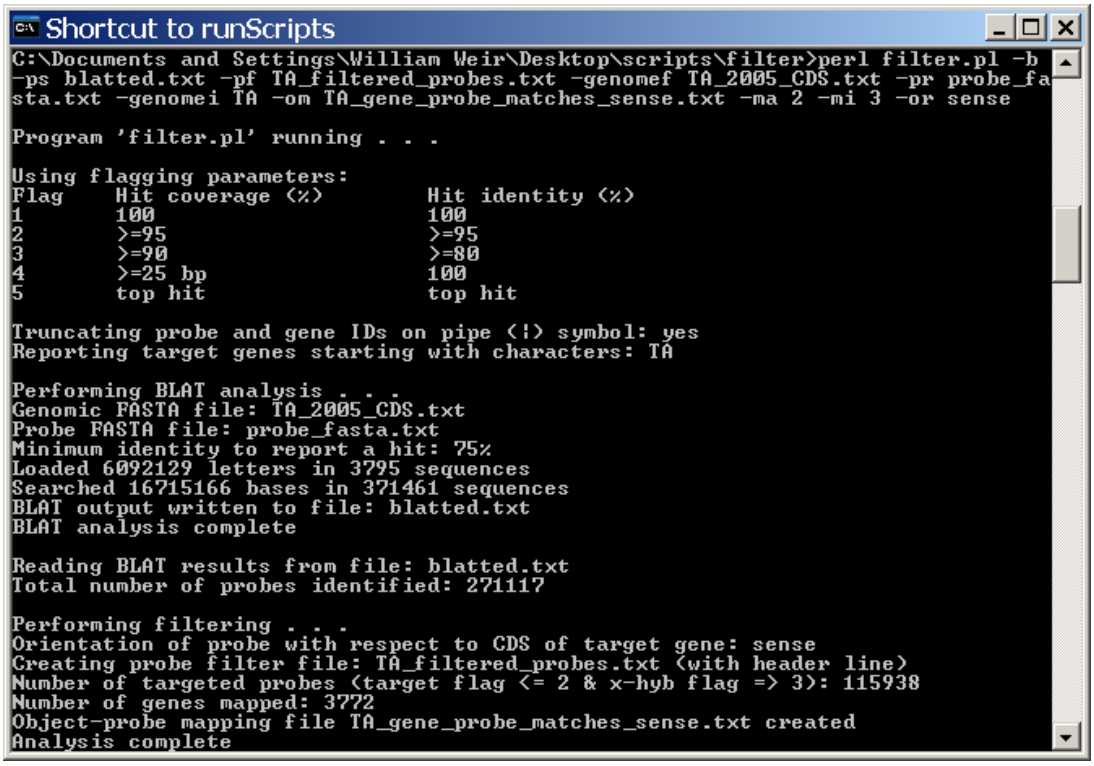
An object-mapping filename must be specified. If BLAT is invoked, then a genome FASTA filename (**`-genomef`**) and probe FASTA filename (**`-pr`**) must be specified. The **`–genomeid`** string is not case sensitive and **`–mincrossflag`** cannot exceed **`-maxtargetflag`**.

Examples

```
perl filter.pl –b –ps blatted.txt –pf TA_filtered_probes.txt
–genomef TA_2005_CDS.txt –pr probe_fasta.txt –genomei TA
-om TA_gene_probe_matches_sense.txt –ma 2 –mi 3 –or sense
```

The list of probes in `probe_fasta.txt` is BLATed against the genome contained in file `TA_CDS.txt` and the output of this process is saved in `blatted.txt`. `blatted.txt` is then filtered and only genes whose name begins with '`TA`', '`Ta`', '`tA`' or '`ta`' are included in the results which are saved in `TA_filtered_probes.txt`. These filtered results are then converted into the object-probe mapping file `TA_gene_probe_matches_sense.txt`. Only probes with target flag 2 or better and cross-hybridisation flag 3 or worse are used.

Screen shot on next page


Shortcut to runScripts
C:\Documents and Settings\William Weir\Desktop\scripts\filter>perl filter.pl -b
-ps blatted.txt -pf TA_filtered_probes.txt -genomef TA_2005_CDS.txt -pr probe_fa
sta.txt -genomei TA -om TA_gene_probe_matches_sense.txt -ma 2 -mi 3 -or sense
Program 'filter.pl' running . . .
Using flagging parameters:
Flag Hit coverage (%) Hit identity (%)
1 100 100
2 >=95 >=95
3 >=90 >=80
4 >=25 bp 100
5 top hit top hit
Truncating probe and gene IDs on pipe (|) symbol: yes
Reporting target genes starting with characters: TA
Performing BLAT analysis . . .
Genomic FASTA file: TA_2005_CDS.txt
Probe FASTA file: probe_fasta.txt
Minimum identity to report a hit: 75%
Loaded 6092129 letters in 3795 sequences
Searched 16715166 bases in 371461 sequences
BLAT output written to file: blatted.txt
BLAT analysis complete
Reading BLAT results from file: blatted.txt
Total number of probes identified: 271117
Performing filtering . . .
Orientation of probe with respect to CDS of target gene: sense
Creating probe filter file: TA_filtered_probes.txt (with header line)
Number of targeted probes (target flag <= 2 & x-hyb flag => 3): 115938
Number of genes mapped: 3772
Object-probe mapping file TA_gene_probe_matches_sense.txt created
Analysis complete

create_fasta_and_map.pl

Description

This script reads a series of probe design files created for the *T. annulata* custom array and uses pre-defined specific parameters to define chromosomal loci (as devised by A. Ivens, Hinxton). The array is a tiling design based upon abutting 45-mer oligos, designed to the top and bottom strands of genomic sequence, staggered by 22 / 23 bases. Three output files are generated –

- a probe information file called `TA_probe_summary.txt`
- a probe-mapping file called `probe_map.txt`
- a probe FASTA file called `probe_fasta.txt`

Command line options

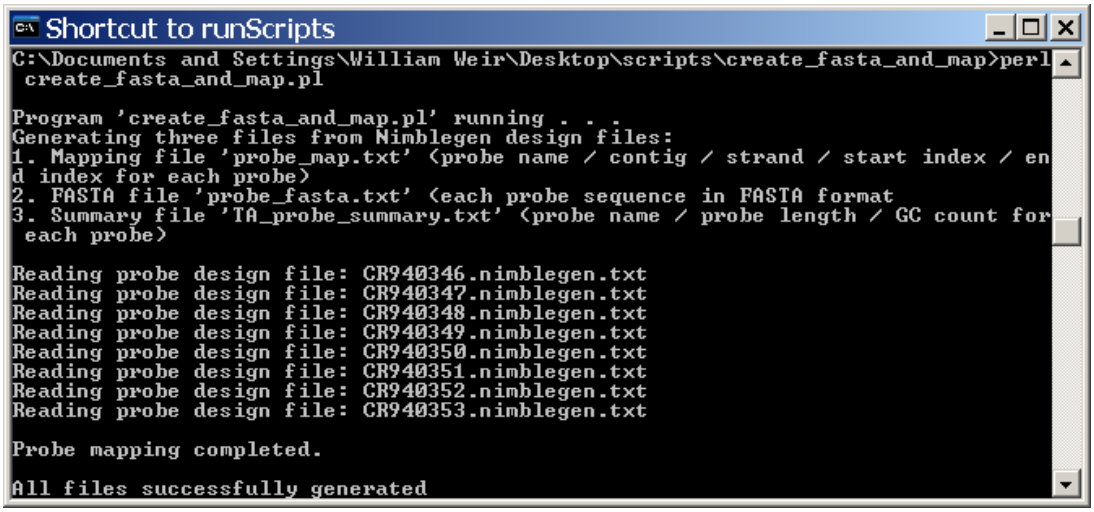
None

Notes

This script was designed solely to process probes designed for the *T. annulata* custom array based on the EMBL genomic annotation files available in July 2005.

Example

- `perl create_fasta_and_map.pl`



create_sgr_files.pl

Description

This script reads a collection of gene ID data files in order to generate a series of sequence graph files for viewing in the Integrated Genome Browser. An exon table file is also read in order to associate the values to be graphed with a chromosomal location. The sequence graph file takes the same name as the corresponding gene ID data file and a `.sgr` suffix is added. The graph values are represented as either a point in the centre of each exon (for a bar graph in IGB) or spanning each exon (for a stair-step graph in IGB, with a value of zero designated at bases flanking the exon). By default, the value in the second column of the gene ID data file will be used to generate the sequence graph file and data in other columns will be ignored. The script can perform $\log_2$ transformations if required.

Command line options

`-b, -batch`=*`FILENAME`*

Name of the batch file containing names of gene ID data files for generating graphs

`-dataf, -datafile`=*`FILENAME`*

Name of a gene ID data file to be processed (if a batch file is also specified, this file is processed last)

`-exonf, -exonfile`=*`FILENAME`*

Name of the exon table file (default is `exon_table.txt`)

`-datac, -datacol`=*`INTEGER`*

Column in the exon table file where the data value is located (default is 2)

`-exonh, -exonheader`=*`INTEGER`*

Number of header lines in the exon table file (default is 3)

`-datah, -dataheader`=*`INTEGER`*

Number of header lines in the gene ID data files (default is 0)

`-l, -log2`

If specified, does $\log_2$ transformation of graphed values (default is no transformation)

`-c, -centremark`

If specified, marks graph data value at the centre of each exon (default is to output stair-step compatible data)

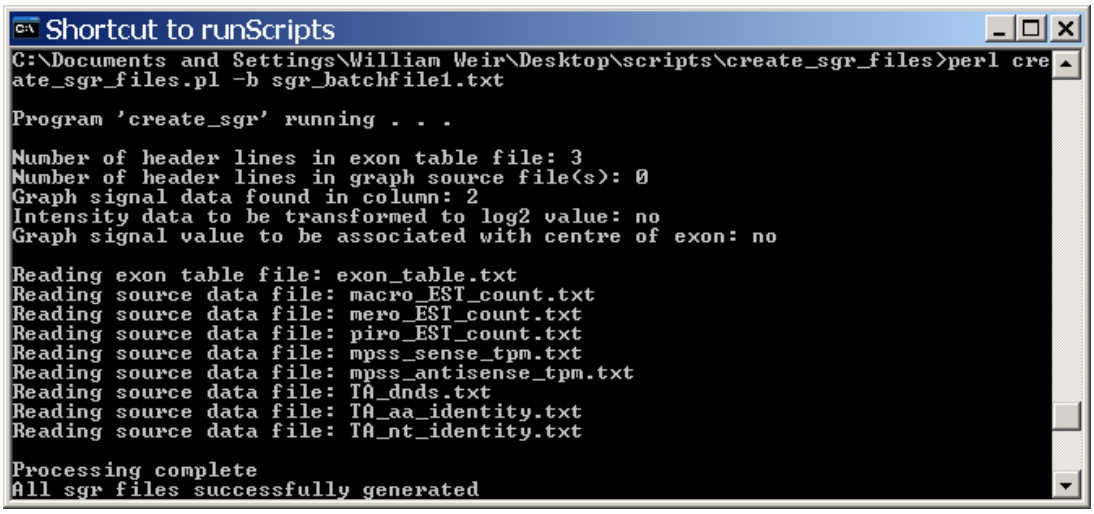
Notes

Either **`-batch`** or **`-datafile`** (or both) must be specified.

Examples

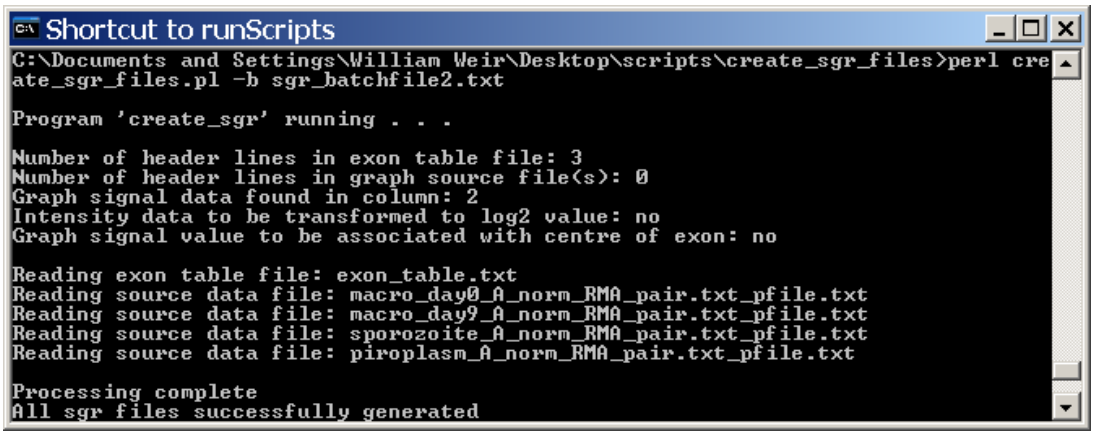
- `perl create_sgr_files.pl –b sgr_batchfile1.txt`

  From all the files specified in `sgr_batchfile2.txt`, a series of `.sgr` files are based on data values from column 2 of each gene ID data file. This example represents EST data, MPSS data and nucleotide identity, protein identity and $d_N d_S$ values from a comparative analysis of *T. annulata* with *T. parva*.



- `perl create_sgr_files.pl –b sgr_batchfile2.txt`

  From all the files specified in `sgr_batchfile2.txt`, a series of `.sgr` files are based on data values from column 2 of each gene ID data file. This example represents the p values (actually, $-10\log_{10}(p\ value)$) produced from gc_wilcoxon analysis



group_random_probes.pl

Description

This script analyses a NimbleGen design file to extract information about random probes of a specific length in order to generate two files. The first is a probe information file containing a list of the random oligos on the array. The second is an object-probe mapping file containing the probe IDs corresponding to each category of GC count. This is analogous to the output from `map_probes_to_exons.pl` and `map_probes_to_genes.pl` scripts. Each object is set of random GC probes with equivalent GC content. Column numbers refer to the current NimbleGen design file format, but may be adjusted using command line options if the format changes in future.

Command line options

`-in, -inputfile`=*`FILENAME`* `(required)`

Name of the NimbleGen design file

`-probef, -probefile`=*`FILENAME`*

Name of the probe information file generated (default is `random_probes.txt`)

`-su, -summaryfile`=*`FILENAME`*

Name of randomGC-mapping file generated (a type of object-mapping file) with probes grouped by their GC count (default is `GC_groups.txt`)

`-id`=*`INTEGER`*

Column in NimbleGen design file where probe ID is located (default is 12)

`-c, -container`=*`INTEGER`*

Column in NimbleGen design file where container information is located (default is 1)

`-se, -sequence`=*`INTEGER`*

Column in NimbleGen design file where probe sequence is located (default is 5)

`-probel, -probelength`=*`INTEGER`*

The size of the random probe oligos on the array (default is 50)

`-l, -lower`=*`INTEGER`*

Lower limit for GC count (default is 10)

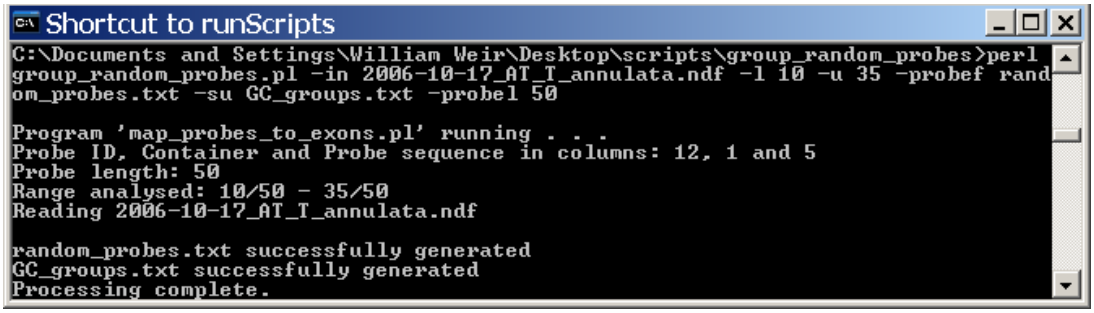
`-u, -upper`=*`INTEGER`*

Upper limit for GC count (default is 35)

Example

```
perl group_random_probes.pl –in 2006-10-17_AT_T_annulata.ndf –l 10 –u 35 –
probef random_probes.txt –su GC_groups.txt –probel 50
```

The NimbleGen design file 2006-10-17_AT_T_annulata.ndf is screened for random 50-mer probes and probe sets containing probes with 21, 22 . . . 40 GC bases are grouped. The probe information file `random_probes.txt` and object-mapping file `GC_groups.txt` are created.


Shortcut to runScripts
C:\Documents and Settings\William Weir\Desktop\scripts\group_random_probes>perl group_random_probes.pl -in 2006-10-17_AT_T_annulata.ndf -l 10 -u 35 -probef random_probes.txt -su GC_groups.txt -probel 50
Program 'map_probes_to_exons.pl' running . . .
Probe ID, Container and Probe sequence in columns: 12, 1 and 5
Probe length: 50
Range analysed: 10/50 - 35/50
Reading 2006-10-17_AT_T_annulata.ndf
random_probes.txt successfully generated
GC_groups.txt successfully generated
Processing complete.

find_bovine_probes.pl

Description

This script analyses a NimbleGen design file to extract information about probes designed to bovine genes. The probes were designed by Dr A. Ivens, Sanger Institute, Hinxton based on a series of 30 bovine control genes supplied by Dr K. Jensen, Roslin Institute, Edinburgh. Bovine probes are identified by the text 'BTAU' in the probe ID field. Two output files are generated –

- a probe information file called `bovine_probes.txt`
- a probe FASTA file called `bovine_probes_fasta.txt`

Command line options

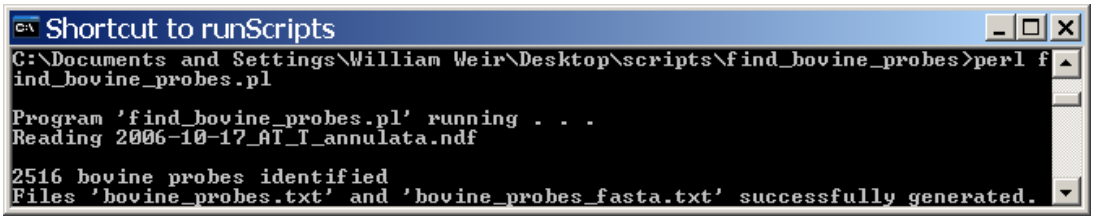
None

Notes

This script was designed solely to identify bovine control probes designed for the *T. annulata* custom array. The source code may be modified to identify probes with alternate identifying text.

Examples

- `perl find_bovine_probes.pl`



create_xys_files.pl

Description

This script generates a series of XYS files for analysis with the R / Biconductor package SMIDA. These files are specifically used for investigating spatial trends on the array. Multiple NimbleGen intensity files may be processed, with a separate XYS file generated for each array hybridisation. The XYS file name is based on the input file name, suffixed with `.xys.txt`. Where cells have no data, the script can either replace the value with the average value across the array or a specific value specified by the user. This value could be zero or, if the data is to be $\log_2$ transformed, close to zero.

Command line options

`-a, -average`

If specified, the average array intensity value is used to replace missing cell data (default is to use a value close to zero)

`-b, -batch`=*`FILENAME`*

Name of the batch file containing names of NimbleGen intensity files to process

`-f, -file`=*`FILENAME`*

Name of the NimbleGen intensity file to be processed (if a batch file is also specified, this file is processed last)

`-h, -header`=*`INTEGER`*

Number of header lines in the NimbleGen intensity file (default is 2)

`-c, -columns`=*`INTEGER`*

Number of columns on the chip (default is 768)

`-r, -rows`=*`INTEGER`*

Number of rows on the chip (default is 1024)

`-x, -xdatacol`=*`INTEGER`*

Column where X data is located in NimbleGen intensity file (default is 6)

`-y, -ydatacol`=*`INTEGER`*

Column where Y data is located in NimbleGen intensity file (default is 7)

`-s, -signalcol`=*`INTEGER`*

Column where signal intensity data is located in NimbleGen intensity file (default is 10)

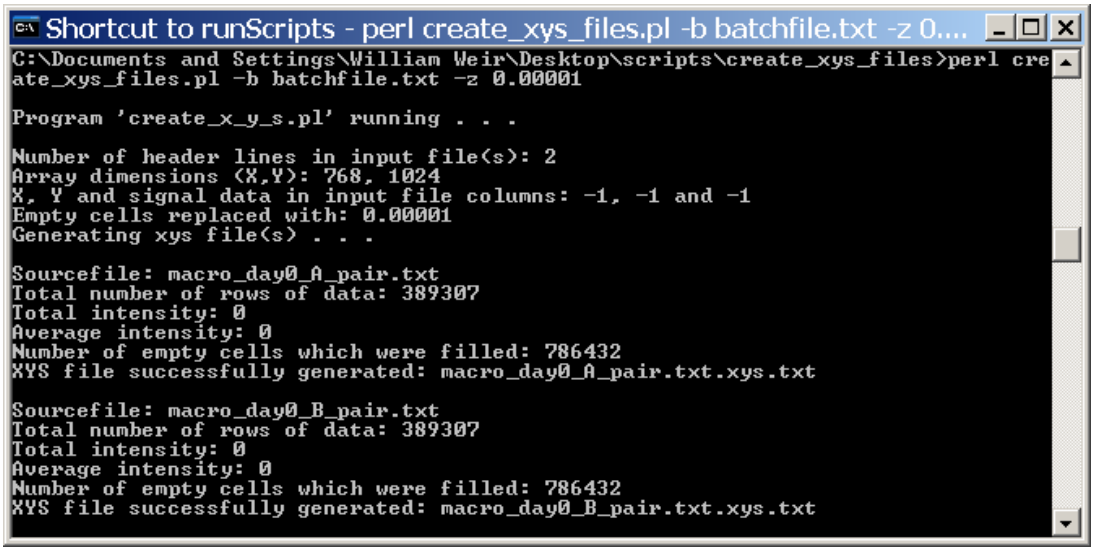
`-z, -zero`=*`FLOATING POINT NUMBER`*

Value close to zero to be entered into empty cells – it not zero to allow $\log_2$ values to be calculated (default is 0.0001)

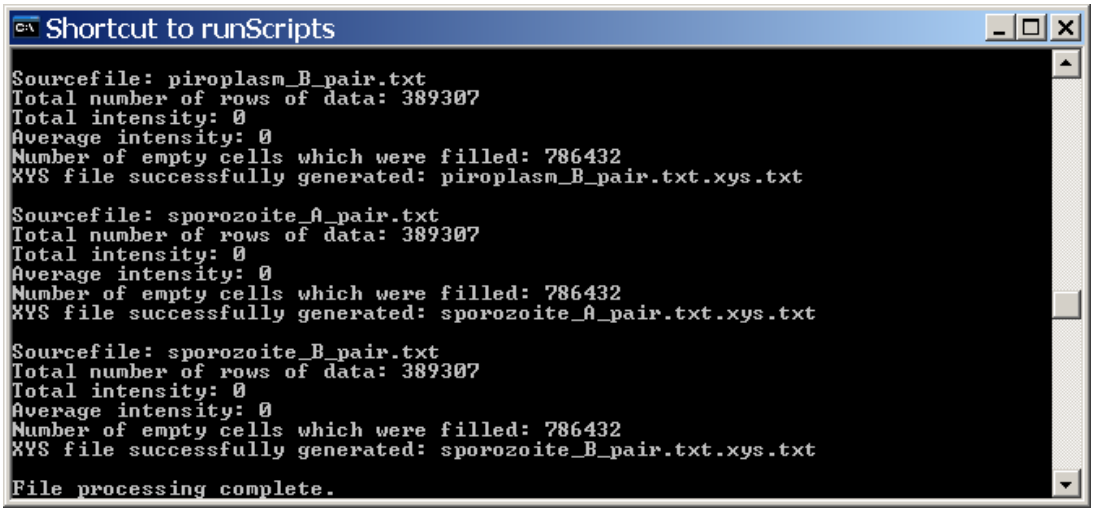
Example

- `perl create_xys_files.pl –b batchfile.txt –z 0.00001`

  The NimbleGen intensity files specified in `batchfile.txt` are processed and a series of XYS files is created, with missing values replaced by 0.00001.


Shortcut to runScripts - perl create_xys_files.pl -b batchfile.txt -z 0....
C:\Documents and Settings\William Weir\Desktop\scripts\create_xys_files>perl cre
ate_xys_files.pl -b batchfile.txt -z 0.00001
Program 'create_x_y_s.pl' running . . .
Number of header lines in input file(s): 2
Array dimensions (X,Y): 768, 1024
X, Y and signal data in input file columns: -1, -1 and -1
Empty cells replaced with: 0.00001
Generating xys file(s) . . .
Sourcefile: macro_day0_A_pair.txt
Total number of rows of data: 389307
Total intensity: 0
Average intensity: 0
Number of empty cells which were filled: 786432
XYS file successfully generated: macro_day0_A_pair.txt.xys.txt
Sourcefile: macro_day0_B_pair.txt
Total number of rows of data: 389307
Total intensity: 0
Average intensity: 0
Number of empty cells which were filled: 786432
XYS file successfully generated: macro_day0_B_pair.txt.xys.txt

. . .


Shortcut to runScripts
Sourcefile: piroplasm_B_pair.txt
Total number of rows of data: 389307
Total intensity: 0
Average intensity: 0
Number of empty cells which were filled: 786432
XYS file successfully generated: piroplasm_B_pair.txt.xys.txt
Sourcefile: sporozoite_A_pair.txt
Total number of rows of data: 389307
Total intensity: 0
Average intensity: 0
Number of empty cells which were filled: 786432
XYS file successfully generated: sporozoite_A_pair.txt.xys.txt
Sourcefile: sporozoite_B_pair.txt
Total number of rows of data: 389307
Total intensity: 0
Average intensity: 0
Number of empty cells which were filled: 786432
XYS file successfully generated: sporozoite_B_pair.txt.xys.txt
File processing complete.

abstract_intensities.pl

Description

This script summarises the intensity values in a NimbleGen intensity file based on object-probe mapping information (gene, exon etc.). Data is read from a single or multiple NimbleGen intensity files to create a series of summarisation files, one for each hybridisation. These files hold median, maximum, minimum, upper quartile and lower quartile values for each object together with individual probe values. An overall composite summarisation file containing just the median values for each object is also created. The script can perform $\log_2$ transformations if required.

Command line options

`-t, -type`=*`STRING`*

Identifier incorporated into output file names, i.e. intensity summary files and composite intensity summary table file (if not specified, mapping file name used)

`-l, -log2`

If specified, does $\log_2$ transformation of intensity values

`-m, -mapfile`=*`FILENAME`* `(required)`

Name of the object-probe mapping file

`-dataf, -datafile`=*`FILENAME`*

Name of the NimbleGen intensity file to be processed (if a batch file is also specified, this file is processed last).

`-b, -batch`=*`FILENAME`*

Name of the batch file containing a list of the names of NimbleGen intensity files

`-h, -header`=*`INTEGER`*

Number of header lines in the NimbleGen intensity files (default is 1)

`-p, -probecol`=*`INTEGER`*

Column where probe ID is located in NimbleGen intensity file (default is 4)

`-datac, -datacol`=*`INTEGER`*

Column where intensity data is located in NimbleGen intensity file (default is 10)

Notes

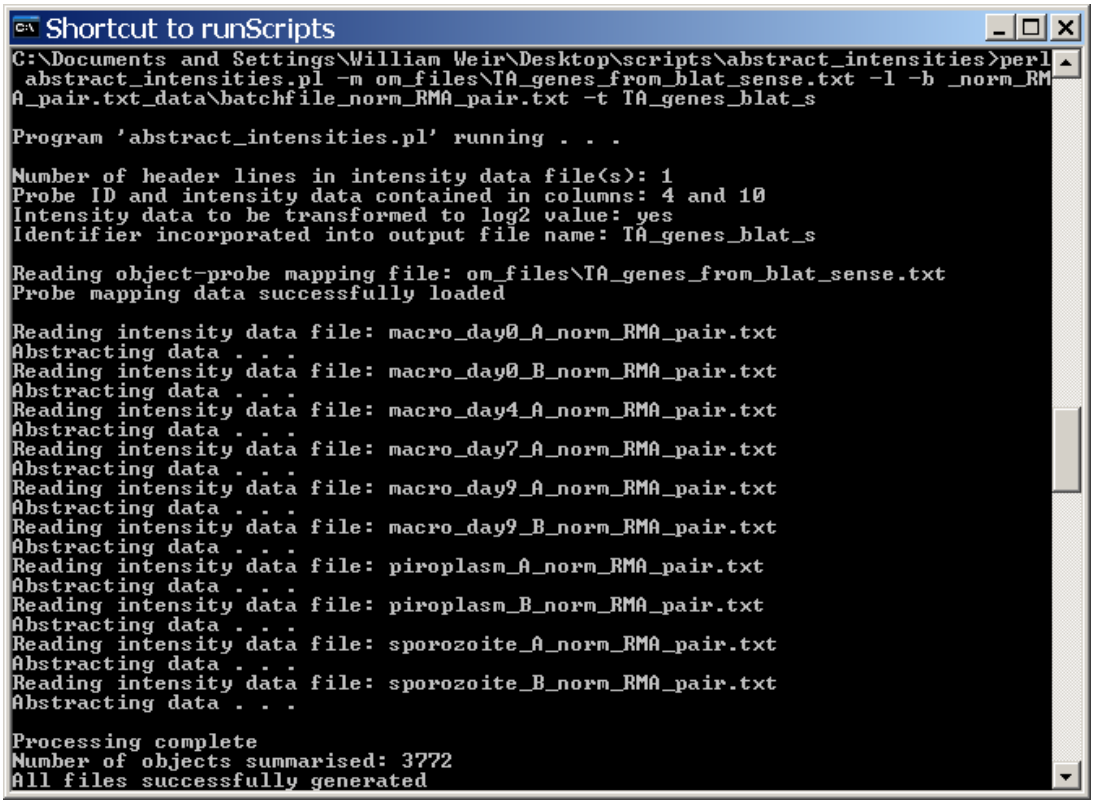
Either `-batch` or `-datafile` (or both) must be specified.

Examples

```
perl abstract_intensities.pl –m om_files\TA_genes_from_blat_sense.txt –l –b
_norm_RMA_pair.txt_data\batchfile_norm_RMA_pair.txt
–t TA_genes_blat_s.txt
```

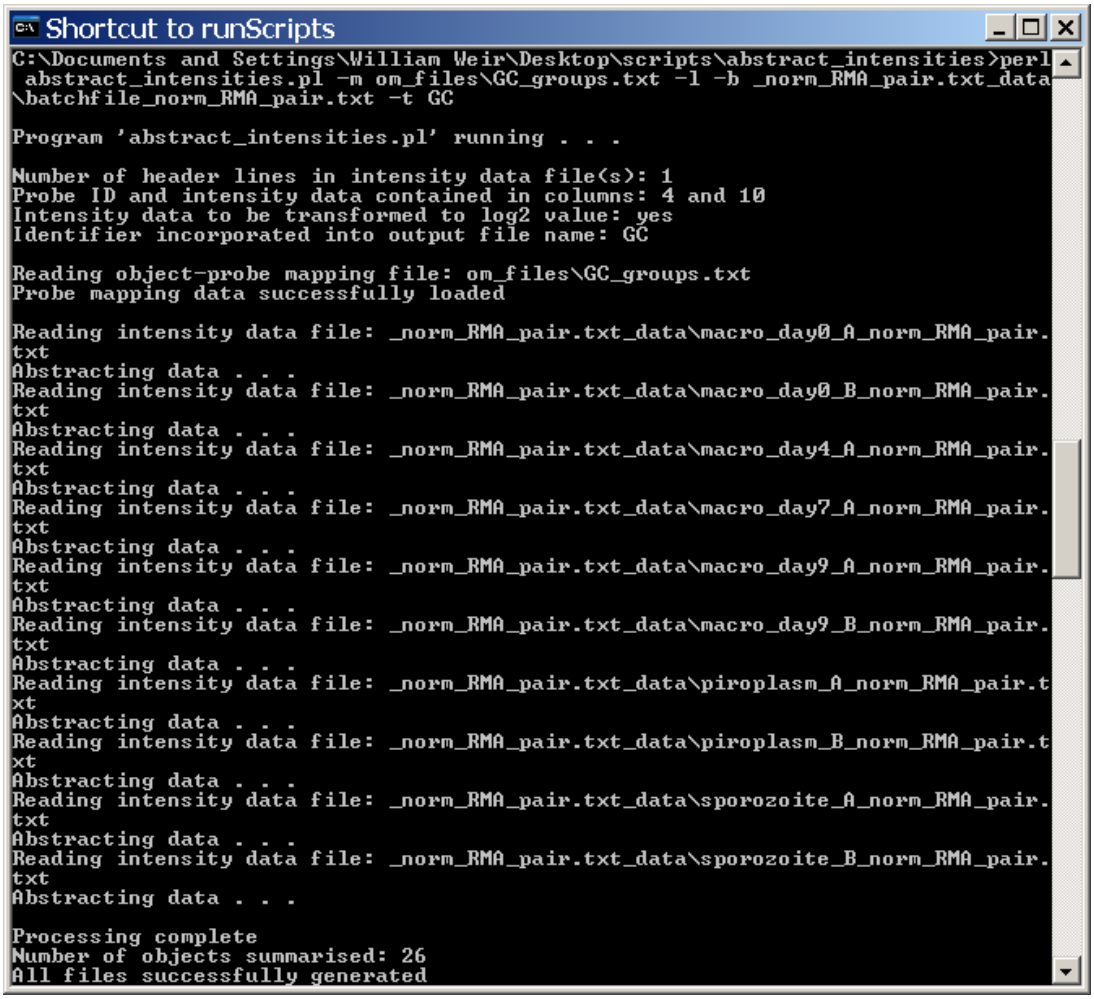
The NimbleGen intensity files specified in `_norm_RMA_pair.txt_data\batchfile_norm_RMA_pair.txt` are summarised as `macro_day0_A_norm_RMA_pair.txt_TA_genes_frblat_s_summarisation.txt` etc. and `all_median_TA_genes_frblat_s.txt`. The object-probe mapping file `TA_genes_from_blat_sense.txt` (in directory `om_files`) is used to specify probe-

sets and probe data is $\log_2$ transformed. These files represent gene-level data for each array hybridisation.



- `perl abstract_intensities.pl –m –om_files\GC_groups.txt –l –b _norm_RMA_pair.txt_data\batchfile_norm_RMA_pair.txt –t GC`

  The NimbleGen intensity files specified in `_norm_RMA_pair.txt_data\batchfile_norm_RMA_pair.txt` are summarised in individual files and `all_median_GC.txt`. The object-probe mapping file `om_files\GC_groups.txt` is used to specify probe-sets. The composite intensity summary table file `all_median_GC.txt` may be used as a 'look-up' file by the `gc_wilcoxon.pl` script. The data is $\log_2$ transformed. *Screen shot on next page*


Shortcut to runScripts
C:\Documents and Settings\William Weir\Desktop\scripts\abstract_intensities>perl abstract_intensities.pl -m om_files\GC_groups.txt -l -b _norm_RMA_pair.txt_data\batchfile_norm_RMA_pair.txt -t GC
Program 'abstract_intensities.pl' running . . .
Number of header lines in intensity data file(s): 1
Probe ID and intensity data contained in columns: 4 and 10
Intensity data to be transformed to log2 value: yes
Identifier incorporated into output file name: GC
Reading object-probe mapping file: om_files\GC_groups.txt
Probe mapping data successfully loaded
Reading intensity data file: _norm_RMA_pair.txt_data\macro_day0_A_norm_RMA_pair.txt
Abstracting data . . .
Reading intensity data file: _norm_RMA_pair.txt_data\macro_day0_B_norm_RMA_pair.txt
Abstracting data . . .
Reading intensity data file: _norm_RMA_pair.txt_data\macro_day4_A_norm_RMA_pair.txt
Abstracting data . . .
Reading intensity data file: _norm_RMA_pair.txt_data\macro_day7_A_norm_RMA_pair.txt
Abstracting data . . .
Reading intensity data file: _norm_RMA_pair.txt_data\macro_day9_A_norm_RMA_pair.txt
Abstracting data . . .
Reading intensity data file: _norm_RMA_pair.txt_data\macro_day9_B_norm_RMA_pair.txt
Abstracting data . . .
Reading intensity data file: _norm_RMA_pair.txt_data\piroplasm_A_norm_RMA_pair.txt
Abstracting data . . .
Reading intensity data file: _norm_RMA_pair.txt_data\piroplasm_B_norm_RMA_pair.txt
Abstracting data . . .
Reading intensity data file: _norm_RMA_pair.txt_data\sporozoite_A_norm_RMA_pair.txt
Abstracting data . . .
Reading intensity data file: _norm_RMA_pair.txt_data\sporozoite_B_norm_RMA_pair.txt
Abstracting data . . .
Processing complete
Number of objects summarised: 26
All files successfully generated

sliding_window.pl

Description

This script performs a similar function to the Affymetrix Tiling Array Software application, TAS (http://www.affymetrix.com/support/developer/tools/affytools.affx). In contrast to TAS, which specifically requires Affymetrix `.cel` files, `sliding_window.pl` uses three generic file types – i.e. intensity files (e.g. NimbleGen intensity files), probe-mapping files and contig information files. Sequence graph files representing the results of sliding windows analysis and interval analysis are generated and may be viewed using the Integrated Genome Browser (IGB). Probes matching to the top strand, bottom strand or either (i.e. both) strands may be used in the analysis. `sliding_window.pl` can also be used to generate raw data intensity sequence graph files from intensity files for viewing in IGB, in order to directly visualise the results of hybridisation experiments.

`sliding_window.pl` can process single files and / or batch files for each of the two conditions being tested. Condition 1 is defined as the 'left side' of the comparison and condition 2 is defined as the 'right side'. That is to say, if comparing 'macroschizont' vs 'merozoite', 'macroschizont' would be condition 1, and 'merozoite' would be condition 2. A two-tailed Ranked Wilcoxon test (unpaired) is used to compare a sliding window of probes in each of two conditions (with or without replicates). For each window, the probability that one condition shows (1) greater, (2) lesser or (3) different expression than the other condition is tested. A *p value* and Hodges-Lehmann estimator is determined for each window and the results are saved in two sequence graph files. As an alternative to using the *p value* of the Rank Wilcoxon test, the default output of `sliding_window.pl` is $-10\log_{10}(p\ value)$, which provides a more easily interpretable graph when viewed in IGB. The Hodges-Lehmann estimator may be used to describe the difference in signal intensity between the two conditions and this corresponds to fold change, if $\log_2$ intensity values have been analysed.

Similar to TAS, `sliding_window.pl` allows the user to perform interval analysis, in order to predict transcripts with differential expression levels between conditions. The results are saved as sequence graph files and GFF annotation files, with predicted transcripts identified as 'predicted transcript 1', 'predicted transcript 2' etc. Valid windows (or bands) may be defined as those having significance better than a threshold value or a Hodges-Lehmann estimator above a threshold value. Similar to TAS, minimum run and maximum gap lengths may be specified together with Hodges-Lehmann estimator and *p value* thresholds to identify detected regions. Interval analysis can also be performed at a later time on Hodges-Lehmann estimator sequence graph files and *p* value sequence graph files using `interval.pl`.

Command line options

`-b, -bandwidth`=*`INTEGER`*

Width of band to use in sliding window analysis (default is 150 nucleotides)

`-ste, -stepsize`=*`INTEGER`*

Size of step to use in sliding window analysis (default is quarter of bandwidth)

`-sta, -startpoint`=*`INTEGER`*

Start nucleotide position to use in first window (default is 1)

`-pr, -probecol`=*`INTEGER`*

Column in NimbleGen intensity data file with probe ID (default is 4)

-datac, -datacol=*INTEGER*

Column in NimbleGen intensity data file with intensity value (default is 10)

-datah, -dataheader=*INTEGER*

Number of header lines in NimbleGen intensity file (default is 1)

-str, -strand=*STRING*

Strand to be analysed, can be `top`, `bottom` or `both` (default is `both`)

-c, -contigfile=*FILENAME*

Name of the contig information file (default is `contigs.txt`)

-mapf, -mapfile=*FILENAME*

Name of the probe-mapping file (default is `probe_map.txt`)

-maph, -mapheader=*INTEGER*

Number of header lines in probe-mapping file (default is 1)

-lefti, -leftid=*STRING*

Short, informative name for condition 1 (default is combination of the filenames specified with `-leftreps` and `–leftdata`)

-righti, -rightid=*STRING*

Short, informative name for condition 2 (default is combination of the filenames specified with `-rightreps` and `–rightdata`)

-pv, -pvalue

If specified, *p value* should be calculated (default option is $-10\log_{10}$(*p value*))

-leftu, -leftup

If specified, generate files for sliding window and interval analysis representing increased expression in condition 1

-rightu, -rightu

If specified, generate files for sliding window and interval analysis representing increased expression in condition 2

-e, -eitherup

If specified, generate files for sliding window and interval analysis representing increased expression in either condition 1 or condition 2

-ra, -rawsgrfile

If specified, generate a raw data intensity sequence graph file (suffixed with `_raw.sgr`) based on the NimbleGen intensity data file

-l, -log2

If specified, $\log_2$ transform intensity values (for both raw intensity sequence graph file and sliding window analysis)

-leftr, -leftreps=*FILENAME*

Name of the batch file containing NimbleGen intensity file names of replicates for condition 1

-leftd, -leftdatafile=*FILENAME*

Name of NimbleGen intensity file containing data for condition 1

-rightr, -rightreps=*FILENAME*

Name of the batch file containing NimbleGen intensity file names of replicates for condition 2

-rightd, -rightdatafile=*FILENAME*

Name of NimbleGen intensity file containing data for condition 2

-i, -interval

If specified, interval analysis will be performed

-maxg, -maxgap=*INTEGER*

During interval analysis, join detected regions with a spacing of equal to or less than this value (default is 3 x step size)

-minr, -minrun=*INTEGER*

During interval analysis, only accept detected regions with a length equal or greater than this value (default is 5 x step size)

-pt, -pthresh=*INTEGER*

During interval analysis, this is the threshold Wilcoxon *p value* (or $-10\log_{10}$(*p value*)) for identifying positive bands (i.e. windows) (default if 0.05 for *p value* and 13 for $-10\log_{10}$(*p value*))

-sig, -sigthresh=*INTEGER*

Threshold absolute Hodges-Lehmann estimator value for identifying positive bands (i.e. windows) (default is 0, i.e. no threshold)

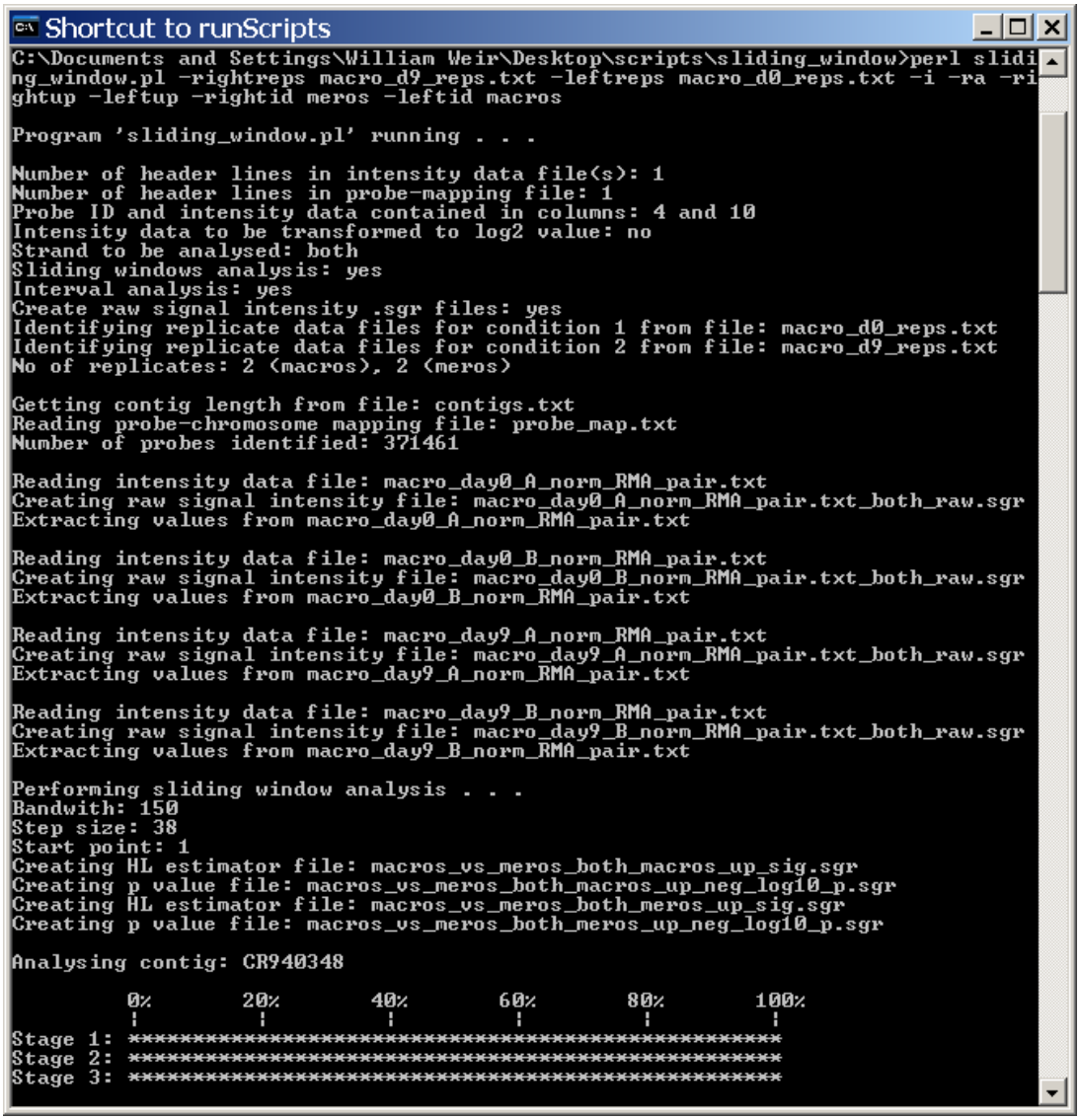
Notes

Either `-leftdatafile` or `–leftreps` (or both) must be specified and either `-rightdatafile` or `–rightreps` (or both) must also be specified for sliding windows analysis. A minimum of one intensity file may specified if the only processing is to convert intensity file data to a sequence graph file. For sliding windows analysis (with or without interval analysis) either `–leftup`, `-rightup` or `-eitherup` (or a combination of these) must be specified.

Examples

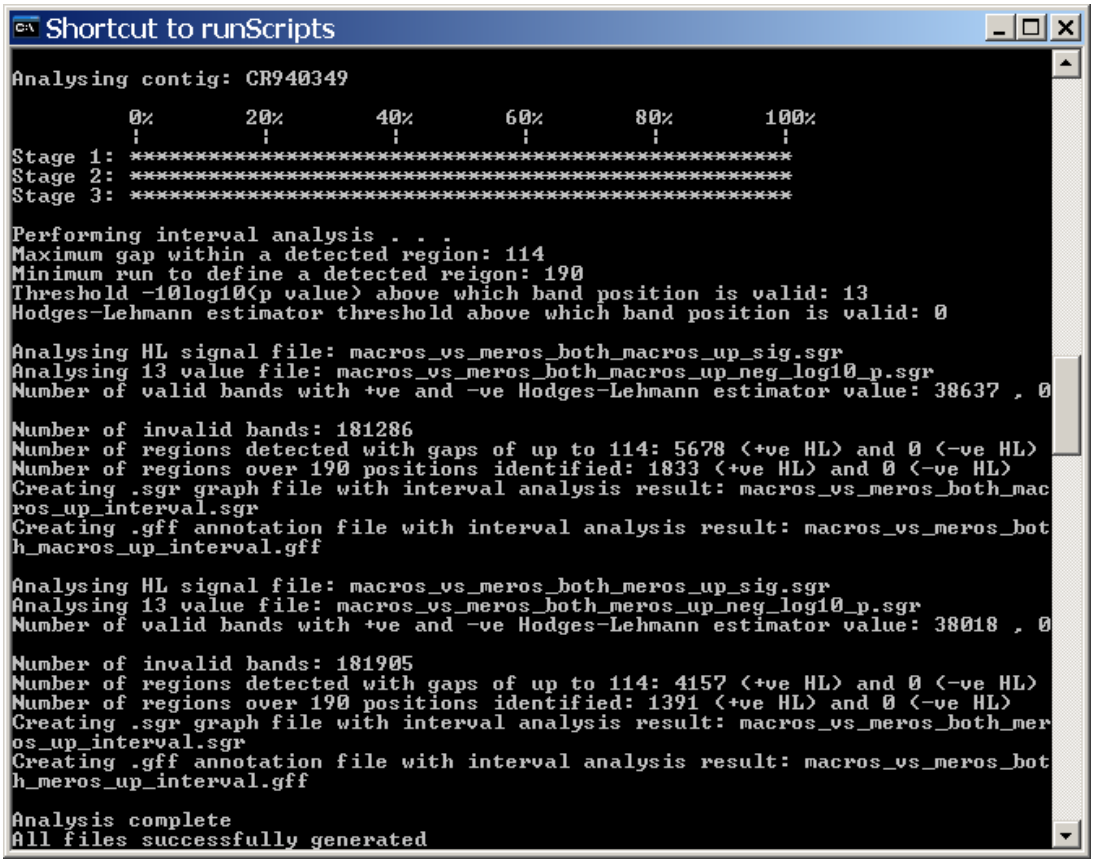
- `perl sliding_window.pl –rightreps macro_d9_reps.txt –leftreps macro_d0_reps.txt –i –ra –rightup –leftup –rightid meros –leftid macros`

Screen shot on next page



. . .

Continued on next page



Sliding windows analysis is performed, comparing replicates of macroschizont hybridisations (listed in `macro_d9_reps.txt`) with replicates of merozoite hybridisations (listed in `macro_d9_reps.txt`). Two analyses are performed, testing for (1) increased expression in macroschizonts (i.e. `-leftup`) and (2) increased expression in merozoites (i.e. `-rightup`). Interval analysis is performed using the default parameters and a raw signal `.sgr` file is produced for every intensity summary file. The -10log$_{10}$(*p values*) are reported from the interval analysis. Separate annotation and sequence graph files are produced for the `-leftup` and `-rightup` analysis.

interval.pl

Description

This script performs interval analysis similar to the Affymetrix Tiling Array Software application, TAS (http://www.affymetrix.com/support/developer/tools/affytools.affx) to predict transcribed genomic regions with differential expression between conditions. In contrast to TAS, which specifically requires Affymetrix `.cel` files, `interval.pl` uses generic sequence graph files representing *p values* and Hodges-Lehmann estimator values created using `sliding_window.pl`. These files represent the results of sliding windows analysis. Although `interval.pl` performs interval analysis in a similar manner to `sliding_window.pl`, `interval.pl` has been made available in order to allow the re-analysis of sliding windows results, which may take considerable processor time to generate, depending on the parameters used.

Similar to interval analysis in `sliding_window.pl`, the results are saved as sequence graph files and GFF annotation files, with predicted transcripts identified as 'predicted transcript 1', 'predicted transcript 2' etc. Valid windows (or bands) may be defined as those having significance better than a threshold value or a Hodges-Lehmann estimator above a threshold value. Similar to `sliding_window.pl`, minimum run and maximum gap lengths may be specified together with Hodges-Lehmann estimator and *p value* thresholds to identify detected regions.

Command line options

`-st, -strand`=*`STRING`*

Strand to be analysed, can be `top`, `bottom` or `both` (default is `both`)

`-h, -hlsigfile`=*`FILENAME`* `(required)`

Name of the Hodges-Lehmann estimator sequence graph file (generated by `sliding_window.pl`)

`-pf, -pfile`=*`FILENAME`* `(required)`

Name of *p value* sequence graph file (generated by `sliding_window.pl`) - can contain *p value* or $-10\log_{10}$(*p value*)

`-o, -outputfile`=*`FILENAME`*

The beginning of the name of the output GFF annotation and sequence graph files (if not specified, the name will be based on the input *p value* file)

`-pv, -pvalue`

If specified, *p value* should be considered rather than the default option of $-10\log_{10}$(*p value*)

`-maxg, -maxgap`=*`INTEGER`*

Join detected regions with a spacing of equal to or less than this value (default is 150)

`-minr, -minrun`=*`INTEGER`*

Only accept detected regions with a length equal to or greater than this value (default 300)

`-pt, -pthresh`=*`INTEGER`*

Threshold Wilcoxon *p value* (or $-10\log_{10}$(*p value*)) for identifying positive bands (i.e. windows) (default if 0.05 for *p value* and 13 for $-10\log_{10}$(*p value*))

-si, -sigthresh=*INTEGER*

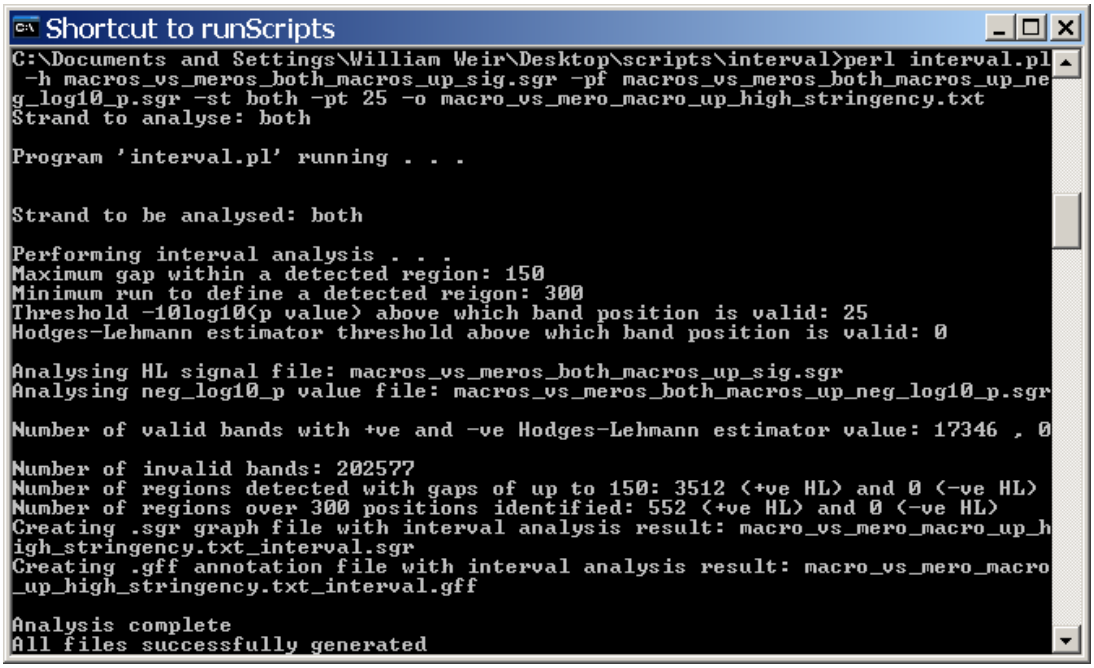
Threshold absolute Hodges-Lehmann estimator value for identifying positive bands (i.e. windows) (default is 0, i.e. no threshold)

Notes

A Hodges-Lehmann estimator sequence graph file and a matching *p value* sequence graph file must be specified, even if threshold values are not set for each of these parameters.

Example

- `perl interval.pl –h macros_vs_meros_both_macros_up_sig.sgr –pf macros_vs_meros_both_macros_up_neg_log10_p.sgr –st both –pt 25 –o macro_vs_mero_macro_up_high_stringency.txt`



Two sequence graph files from a previous sliding window analysis are processed. Hodges-Lehmann Estimator values are read from `macros_vs_meros_both_macros_up_sig.sgr` and -10log$_{10}$(*p values*) are read from `macros_vs_meros_both_macros_up_neg_log10_p.sgr`. A relatively high threshold *p value* of 25 is set to facilitate a more stringent analysis and the results are saved in general feature format and sequence graph format, with filenames based on the text: `macro_vs_mero_macro_up_high_stringency.txt`.