Using IWTomics to detect genomic features that discriminate between different groups of regions.

Marzia A Cremona*†, Alessia Pini*,
Francesca Chiaromonte and Simone Vantini

April 19, 2017

1 Introduction

IWTomics is a package to statistically evaluate differences in genomic features between groups of regions along the genome. Locations (within the regions) and scales at which these differences unfold need not be specified at the outset, and are in fact an output of the procedure. In particular, the package implements an extended version of the Interval-Wise Testing (IWT) for functional data (**?**), specifically designed to work with "Omics" data. IWTomics also includes a set of functions based on the package `GenomicRanges` to import and organize measurements of multiple genomic features in different regions, as well as a set of functions to create graphical representations of the features and of the test results. Importantly, genomic regions can have different length and different features can be measured at different resolutions.

In this vignette we present an example in which IWTomics is used to compare recombination hotspots in the genomic regions surrounding fixed ETns (elements of the Early Transposon family of active Endogenous Retroviruses in mouse) versus control regions. This data is part of a much larger dataset analyzed in Campos-Sánchez et al. (2016). The complete dataset comprises several genomic features, and was used to study integration and fixation preferences of different families of endogenous retroviruses in the mouse and human genomes (the software used in Campos-Sánchez et al. (2016) was a beta version of this package in which we implemented the Interval Testing Procedure of Pini and Vantini (2016) in place of the Interval-Wise Testing employed here).

We also present the complete workflow and various options in IWTomics through some synthetic datasets that are provided within the package.

*These authors contributed equally
†mac78@psu.edu

1.1 Interval-Wise Testing (IWT)

The IWT is an inferential procedure that tests for differences in the distributions of a genomic feature between two sets of regions (e.g. between case and control regions – two sample test), or between a single set of regions and a reference curve (e.g. between case regions and a reference null measurement – one sample test). The feature under study must be measured in windows of fixed size in each region (e.g. at a resolution of 1 bp, or over 1 kb windows), and these contiguous measurements are considered by the IWT as curves. The IWT is a Functional Data Analysis technique, hence it can directly deal with these curves. In particular, the IWT is able to assess whether there are differences in the distributions of the curves globally, and it also investigates local effects, imputing the statistically significant differences to specific locations along the regions (e.g. only in the central part of the regions).

Since the IWT is based on permutation tests (non-parametric tests), it does not require any assumption on the statistical distribution of the data and it can be easily employed to study different types of "Omics" signals, from DNA conformation contents, to transcription data or chromatin modifications. Moreover, it can be used even if the sample sizes differ in the two groups under consideration, or if the sample sizes are small. Sample sizes affect the resolution of the empirical p-value which will not exceed $1/P$, where P the total number of possible permutations leading to distinct values of the test statistics (see Subsection 4.2). For instance, in the two sample test between two independent populations of sizes n_1 and n_2, $P = \binom{n_1+n_2}{n_1}$.

In the extended version implemented in IWTomics, the IWT is not only location-free but also scale-free. Indeed, it is able to investigate a range of different scales (from the finest one provided by the measurement resolution, to the coarsest one given by the region length), making it possible to assess the scale at which a feature displays its effect (see Section 7 for details). Moreover, different test statistics can be employed (e.g. mean difference, variance ratio or quantile difference).

1.2 Installation and loading

IWTomics package is available at bioconductor.org and can be installed via `biocLite`, typing the following commands in the R console (an internet connection is needed)

```
source("http://www.bioconductor.org/biocLite.R")
biocLite("IWTomics",dependencies=TRUE)
```

After the package is installed, it can be loaded into R workspace typing

```
library(IWTomics)
```

1.3 A first example: recombination hotspots around fixed ETn

We illustrate use and output of IWTomics through a real data example from Campos-Sánchez et al. (2016) (the dataset is provided as part of the package). This data

contains two groups of genomic regions, `ETn fixed` and `Control`. In each region we measure one genomic feature, the content of `Recombination hotspots`. This is how to load the data into R:

```
data(ETn_example)
ETn_example

## IWTomicsData object with 2 region datasets with center alignment, and 1 feature:
## Regions:
##  ETn fixed:  1296 regions
##    Control:  1142 regions
## Features:
##  Recombination hotspots content:  1000 bp resolution
## No tests present.
```

The region dataset `ETn fixed` comprises 1296 regions flanking fixed ETn elements (elements of the Early Transposon family elements of active Endogenous Retroviruses in mouse). These regions correspond to 32-kb flanking sequence upstream and 32-kb flanking sequence downstream of each fixed ETn element. The content of `Recombination hotspots` is measured at 1000 bp resolution in each region: the 64-kb length is divided into 64 1-kb and 64 consecutive measurements are produced, expressing the fraction of each 1-kb window covered by recombination hotspots. The goal is to understand whether the presence of fixed ETn elements in the genome is affected by recombination hotspots. To this end, we compare the regions in `ETn fixed` with control regions, defined as 64-kb regions that do not overlap with flanking sequences of ETn

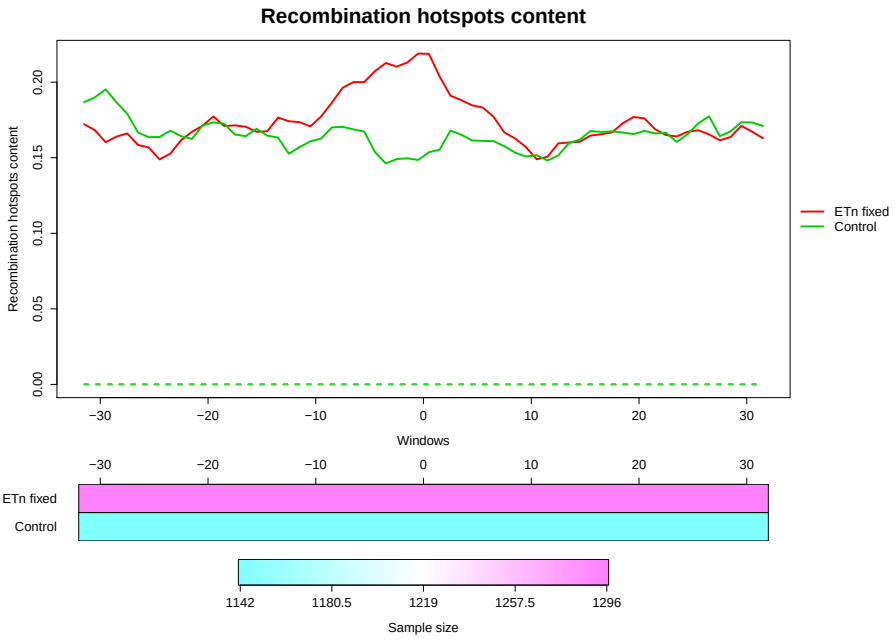


Figure 1: Plot of `Recombination hotspots` in `ETn fixed` regions (red) and `Control` regions (green).

or of other families of Endogenous Retroviruses. The region dataset `Control` contains 1142 of such regions, for each of which we have again 64 consecutive measurements of `Recombination hotspots`.

Figure 1 shows a visual representation of the dataset obtained with the function `plot` in the following code:

```
plot(ETn_example,cex.main=2,cex.axis=1.2,cex.lab=1.2)
```

Figure 1 suggests that the average recombination hotspots content right around the locations of fixed ETns (±5 kb flanks) is higher than that in control regions. The function `IWTomicsTest`, the main function of the package IWTomics, allows us to test differences in a rigorous way, returning a p-value curve (one p-value for each 1-bp window) adjusted considering the whole 64-kb region, as shown in the next chunck of code.

```
ETn_test=IWTomicsTest(ETn_example,
                      id_region1='ETn_fixed',id_region2='Control')

## Performing IWT for 'ETn fixed' vs.  'Control'...
##  Performing IWT for feature 'Recombination hotspots content'...
##    Point-wise tests...
##    Interval-wise tests...

adjusted_pval(ETn_test)

## $test1
## $test1$Recombination_hotspots
##  [1] 0.456 0.457 0.504 0.747 0.864 0.890 0.890 0.890 0.931 0.990
## [11] 0.990 0.992 0.990 0.990 0.990 0.990 0.990 0.990 0.951 0.771
## [21] 0.726 0.726 0.713 0.688 0.575 0.459 0.364 0.197 0.072 0.039
## [31] 0.048 0.085 0.218 0.482 0.768 0.829 0.887 0.946 0.978 0.991
## [41] 0.992 0.996 0.996 0.996 0.996 1.000 1.000 1.000 1.000 1.000
## [51] 0.993 0.993 0.993 0.993 0.993 0.993 0.993 0.993 0.993 0.993
## [61] 0.993 0.993 0.993 0.993
```

Test results can be easily understood using the visual representation provided by the function `plotTest` and shown in Figure 2.

```
plotTest(ETn_test)
```

In this figure, the top panel shows the adjusted p-value heatmap: the x axis represents locations, i.e. the 64 1-kb windows in the 64-kb region, while the y axis represents all possible scales used to adjust the p-value curve (from 1 window, no adjustment; to 64 windows, adjustment based on the entire region). The central panel is a plot of the adjusted p-values at the maximum scale (threshold at 64 windows; more details below and in Subsection 4.1), and the gray area corresponds to significant adjusted p-values (< 0.05). The bottom panel is a visual representation of the feature in the two groups

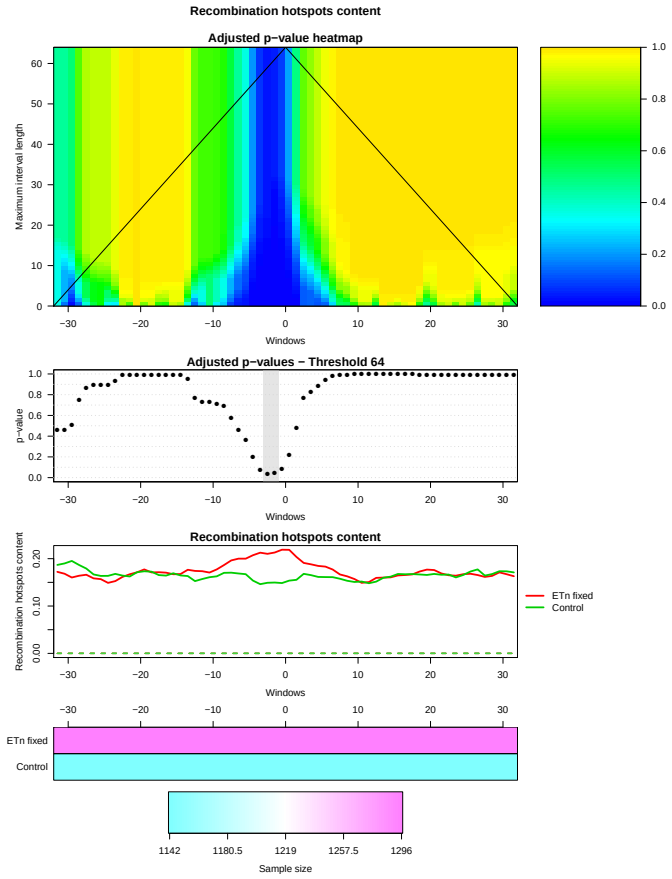


Figure 2: Plots of IWT results for `Recombination hotspots` in the comparisons `Etn fixed` vs `Control`.

of regions. The blue area in the adjusted p-value heatmap (top panel) shows that the difference between `Etn fixed` and `Control` is significant in the central part of the region, i.e. near the ETn's integration site. Notably, the results holds for broader flanks around the integration site at smaller scales - i.e. if we adjust each p-value based on fewer neighboring positions (the lower part of the heatmap shows a larger blue area).

If we focus on a smaller scale (e.g. setting the threshold at 10 windows, i.e 10 kb) and adjust the p-values considering only 10 neighboring positions, the central subregion where the difference is significant (p-value < 0.05) is broader. This is shown in Figure 3, which has a larger the gray area in the central panel. This figure corresponds to Figure 3A in Campos-Sánchez et al. (2016).

```
adjusted_pval(ETn_test,scale_threshold=10)

## $test1
## $test1$Recombination_hotspots
##  [1] 0.358 0.252 0.302 0.565 0.740 0.804 0.824 0.825 0.931 0.990
## [11] 0.990 0.992 0.990 0.990 0.990 0.990 0.990 0.990 0.951 0.737
## [21] 0.634 0.605 0.539 0.472 0.337 0.218 0.134 0.043 0.009 0.002
```

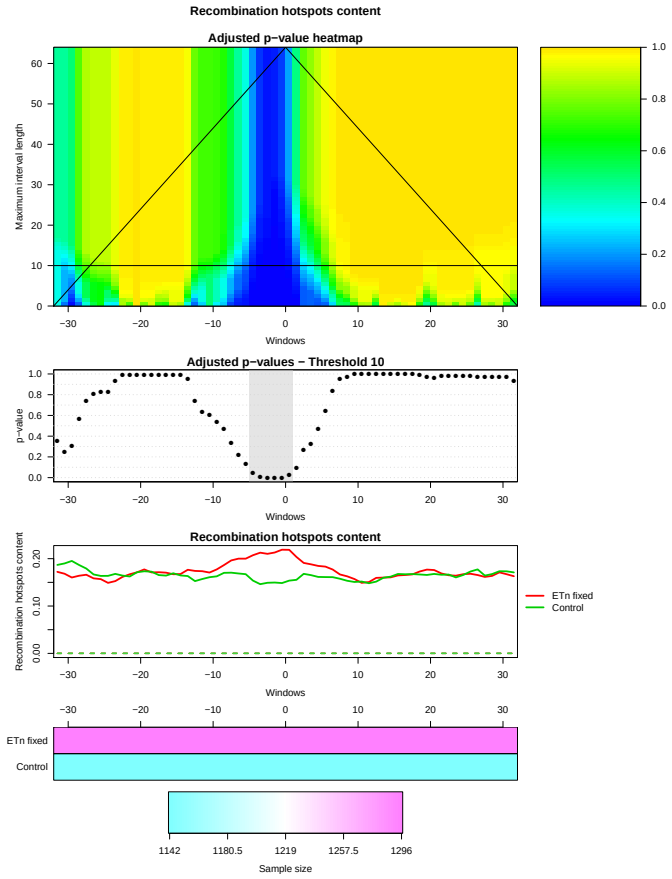


Figure 3: Plots of IWT results for `Recombination hotspots` in the comparisons `Etn fixed` vs `Control`, with scale threshold 10 kb.

```
## [31] 0.002 0.002 0.022 0.097 0.268 0.330 0.472 0.642 0.834 0.954
## [41] 0.976 0.996 0.996 0.996 0.996 1.000 1.000 1.000 1.000 1.000
## [51] 0.991 0.967 0.964 0.983 0.983 0.983 0.983 0.983 0.968 0.968
## [61] 0.968 0.968 0.967 0.932

plotTest(ETn_test,scale_threshold=10)
```

From these results we can conclude that recombination hotspots show significant differences at locations near the integration site of fixed ETn; in particular, they are enriched on average. This effect is stronger at small scales, up to 10-kb.

1.4 Regions, features and measurement resolution

IWTomics can be easily employed to compare several genomic features of different biological nature, in multiple pairs of region groups. For example, in addition to the flanking regions of fixed ETn and control regions in the mouse genome, the complete dataset in Campos-Sánchez et al. (2016) contains flanking regions of mouse polymorphic ETn, fixed IAP (Intracisternal A Particle, another family of active Endogenous

Retroviruses) and polymorphic IAP. Moreover, it also contains three groups of regions in the human genome: the flanking regions of fixed HERV-K (a family of Human Endogenous Retroviruses), those of *in vitro* HERV-K, and control regions. Around 40 genomic features were considered for each region in each group, and compared between different groups. These features reflected DNA conformation, DNA sequence, recombination, replication, gene regulation and expression, and selection. Overall, the dataset allowed us an in-depth investigation of the genomic landscapes characterizing the families of Endogenous Retroviruses, and to separate fixation from integration preferences. The study demostrated the flexibility and broad applicability of IWTomics. Indeed, different types of "Omics" data can be employed as features in the test. However, nature and resolution of measurements should be carefully chosen, because the IWT is designed to work with continuous measurements. "Omics" data that are discrete in nature should be considered at medium-high resolution. For example, consider recombination hotspots, exons or microsatellites: we can measure their content or count in windows of various sizes (e.g., 1-kb or 10-kb), but at very fine resolution (very small windows) these will reduce to presence or absence (1 or 0 measurement). In contrast, "Omics" data such as ChIP-seq signals maintain good ranges also at very fine resolution. Genomic regions can also be defined in different ways, to address different biological questions. For instance, they can be flanking regions of some element of interest as in Campos-Sánchez et al. (2016), regions centered around Transcription Start Sites (TSS) of genes, regions annotated as functional or under selection through genome-wide screens, regions occupied by special DNA structures (e.g. G-quadruplexes), etc. Notably, in some applications the regions will all have the same length and a natural alignment to each other (e.g., the insertion site of an ETn or the TSS). In other applications, lengths may differ and/or an alignment choice selected (see Section 2 for more details).

2 Importing data

The first step consists in importing the datasets (regions and features) that we wish to study in an object of class `"IWTomicsData"`, using the constructor `IWTomicsData`.

Each region dataset can be provided as a BED file or as a table with columns `chr start end` (extra columns present in the input file are ignored). Importantly, IWTomics can deal with regions of different lengths.

```
chr2  49960150  50060150
chr2  55912445  56012445
...
```

Here we consider four different region datasets each containing regions of 50 kb. Three comprise different types of elements (`Elements 1`, `Elements 2` and `Elements 3`) and one comprises control regions (`Control`).

```
examples_path <- system.file("extdata",package="IWTomics")
datasets=read.table(file.path(examples_path,"datasets.txt"),
```

```
                    sep="\t",header=TRUE,stringsAsFactors=FALSE)
datasets

##        id       name            regionFile
## 1   elem1 Elements 1 Elements1_regions.bed
## 2   elem2 Elements 2 Elements2_regions.bed
## 3   elem3 Elements 3 Elements3_regions.bed
## 4 control   Controls  Controls_regions.bed
```

Similarly, we need to provide the feature measurements corresponding to each region dataset. Each feature must be measured in windows of a fixed size inside all the regions (missing values are indicated as `NA`). Feature measurements can be provided as a BED file with four columns `chr start end value` and one row for each window

```
chr2 49960150 49962150 0.942623929894372
chr2 49962150 49964150 0.7816422042578235
...
```

Here we consider two different genomic features (`Feature 1` and `Feature 2`), with measurements at 2 kb resolution.

```
features_datasetsBED=
    read.table(file.path(examples_path,"features_datasetsBED.txt"),
               sep="\t",header=TRUE,stringsAsFactors=FALSE)
features_datasetsBED

##     id      name                  elem1                  elem2
## 1 ftr1 Feature 1 Feature1_elements1.bed Feature1_elements2.bed
## 2 ftr2 Feature 2 Feature2_elements1.bed Feature2_elements2.bed
##                    elem3               control
## 1 Feature1_elements3.bed Feature1_controls.bed
## 2 Feature2_elements3.bed Feature2_controls.bed
```

When importing data, we should specify how the regions should be aligned with respect to each other though the argument `alignment`. Possible choices are `left`, `right` and `center` for aligning the regions on their starting, ending and central positions, respectively. An additional option, `scale`, scales all regions to the same length. If the length is the same for all regions, all choices are equivalent for testing purposes, but subsequent graphical visualizations differ. In this particular example we align the regions on their central position.

When importing the feature measurements from BED files, `IWTomicsData` check for consistency between the region datasets and the feature measurements before aligning the measurements according to the argument `alignment`. This reduces the chances of using mismatched data, but it can be time consuming.

```
startBED=proc.time()
regionsFeatures=IWTomicsData(datasets$regionFile,
    features_datasetsBED[,3:6],alignment='center',
    id_regions=datasets$id,name_regions=datasets$name,
    id_features=features_datasetsBED$id,
    name_features=features_datasetsBED$name,
    path=file.path(examples_path,'files'))

## Reading region dataset 'Elements 1'...
## Reading region dataset 'Elements 2'...
## Reading region dataset 'Elements 3'...
## Reading region dataset 'Controls'...
## Reading feature 'Feature 1'...
##    Region dataset 'Elements 1'...
##    Region dataset 'Elements 2'...
##    Region dataset 'Elements 3'...
##    Region dataset 'Controls'...
## Reading feature 'Feature 2'...
##    Region dataset 'Elements 1'...
##    Region dataset 'Elements 2'...
##    Region dataset 'Elements 3'...
##    Region dataset 'Controls'...

endBED=proc.time()
endBED-startBED

##    user  system elapsed
##   7.128   0.000   7.142
```

Another way to import feature measurements is from a table file with the first three columns `chr start end` corresponding to the different genomic regions, followed on the same row by all the measurements in fixed-size windows.

```
chr2  49960150  50060150  0.942623929894372  0.781642204257823
  0.892165843353036  ...  ...  1.20635198854438
chr2  55912445  56012445  0.871916848756875  0.997520895351788
  1.16194557122965  ...  ...  0.960164147842107
...
```

```
features_datasetsTable=
    read.table(file.path(examples_path,"features_datasetsTable.txt"),
               sep="\t",header=TRUE,stringsAsFactors=FALSE)
features_datasetsTable

##     id      name                  elem1                  elem2
## 1 ftr1 Feature 1 Feature1_elements1.txt Feature1_elements2.txt
## 2 ftr2 Feature 2 Feature2_elements1.txt Feature2_elements2.txt
```

```
##                     elem3               control
## 1 Feature1_elements3.txt Feature1_controls.txt
## 2 Feature2_elements3.txt Feature2_controls.txt
```

When importing the measurements from table files, the constructor `IWTomicsData` needs to perform fewer controls and is therefore faster.

```
startTable=proc.time()
regionsFeatures=IWTomicsData(datasets$regionFile,
    features_datasetsTable[,3:6],alignment='center',
    id_regions=datasets$id,name_regions=datasets$name,
    id_features=features_datasetsBED$id,
    name_features=features_datasetsBED$name,
    path=file.path(examples_path,'files'))

## Reading region dataset 'Elements 1'...
## Reading region dataset 'Elements 2'...
## Reading region dataset 'Elements 3'...
## Reading region dataset 'Controls'...
## Reading feature 'Feature 1'...
##    Region dataset 'Elements 1'...
##    Region dataset 'Elements 2'...
##    Region dataset 'Elements 3'...
##    Region dataset 'Controls'...
## Reading feature 'Feature 2'...
##    Region dataset 'Elements 1'...
##    Region dataset 'Elements 2'...
##    Region dataset 'Elements 3'...
##    Region dataset 'Controls'...

endTable=proc.time()
endTable-startTable

##    user  system elapsed
##    1.36    0.00    1.36
```

`IWtomicsData` returns an object of S4 class `"IWTomicsData"`. This object is a container that stores a collection of aligned genomic region datasets, and their associated feature measurements. In the next chunk of code, we show the content of the object, and how to subset only the measurements of `Feature 1` corresponding to the region datasets `Elements 1` and `Control` using the subsetting method `[`.

```
regionsFeatures

## IWTomicsData object with 4 region datasets with center alignment, and 2 features:
## Regions:
##  Elements 1:  25 regions
```

```
##  Elements 2:  20 regions
##  Elements 3:  28 regions
##    Controls:  35 regions
## Features:
##  Feature 1:  2000 bp resolution
##  Feature 2:  2000 bp resolution
## No tests present.

regionsFeatures_subset=regionsFeatures[c('elem1','control'),'ftr1']
regionsFeatures_subset

## IWTomicsData object with 2 region datasets with center alignment, and 1 feature:
## Regions:
##  Elements 1:  25 regions
##    Controls:  35 regions
## Features:
##  Feature 1:  2000 bp resolution
## No tests present.
```

An alternative way to prepare data for IWTomics is to directly create an object of class `"IWTomicsData"` from genomic region datasets and feature measurements, using the alternative constructor function `IWTomicsData`. The mandatory fields are a `"GRangesList"` object with genomic locations corresponding the different region datasets, their `alignment` and the features measurements, aligned and arranged in matrices. Methods to combine `"IWTomicsData"` objects are also implemented in the package (`c`, `merge`, `rbind` and `cbind`).

3 Visualizing feature measurements

Before proceeding with the test, it can be very useful to visually inspect the data. The `plot` method for `"IWTomicsData"` class provides multiple types of visualizations.

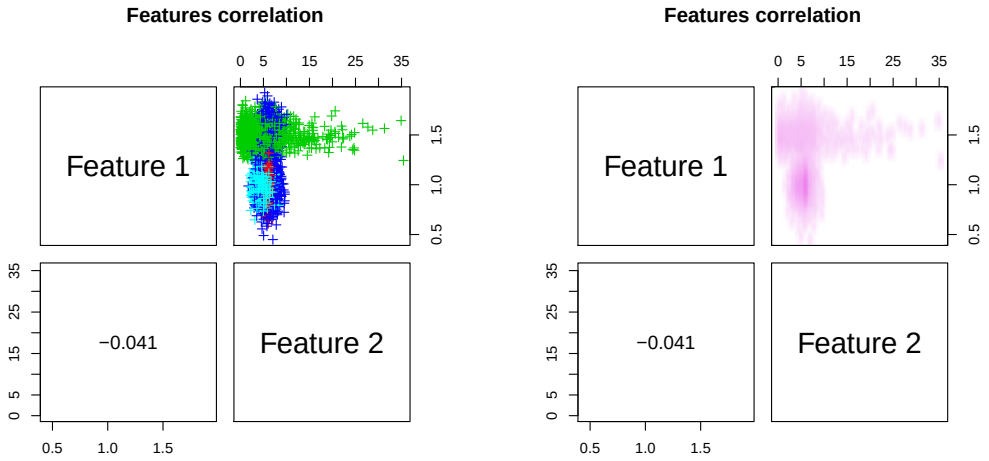
A scatterplot of the measurements corresponding to different features reveals pairwise correlations among them (Figure 4a).

```
plot(regionsFeatures,type='pairs')
```

When datasets are large and many measurements are present, a smoothed scatterplot may represent the data distribution more effectively (Figure 4b).

```
plot(regionsFeatures,type='pairsSmooth',col='violet')
```

The plot function allows the user to plot only a subset of region datasets (argument `id_regions_subset`) and of genomic features (`id_features_subset`). In addition, it is possible to plot only a subsample of `N_regions` regions randomly selected from each region dataset, and the logarithm of the measurements can be plotted instead of the raw values (arguments `log_scale` and `log_shift`).



(a) Scatterplot, with colors indicating different region datasets.

(b) Smoothed scatterplot.

Figure 4: Scatterplot of the measurements for `Feature 1` and `Feature 2`.

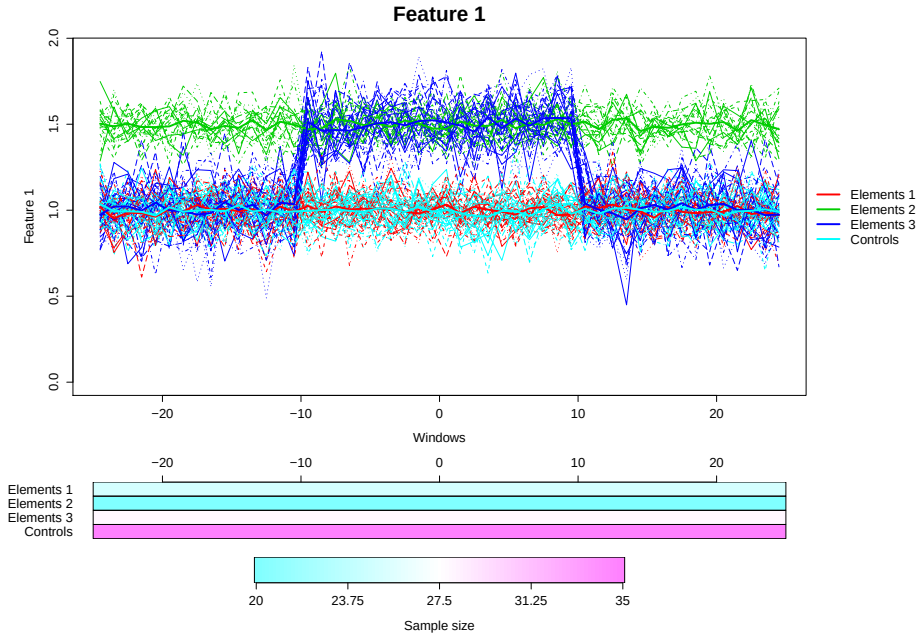


Figure 5: Plot of all the curves for `Feature 1`. The measurements corresponding to different region datasets are represented with different colors, with the mean curves as solid lines. At the bottom of the plot, the heatmap shows the sample size corresponding to each position and each region dataset.

Another useful graphical representation of the data is the plot of the aligned measurement curves. This plot shows the level of roughness in the data and/or in the average and quantile curves, thus suggesting the need for a smoothing step (see Section 6). Also in this case the user can decide which regions and features should be plotted, the number of regions to be shown and whether to plot logarithms of the measurements. As an example, a `curve` plot of `Feature 1` measurements in all the region datasets considered is generated by the next chunk of code and shown in Figure 5. In addition to the measurements, the mean curves can be plotted (`average=TRUE`) and the sample sizes in each position can be shown for each region dataset. When the data under study contain missing measurements (`NA`) or regions of different lengths, information regarding the pointwise sample sizes is essential to evaluate the power of the IWT in different portions of the curves. In the following example individual curves are quite noisy, while the average curves appear to be rather smooth.

```
plot(regionsFeatures,type='curves',
     N_regions=lengthRegions(regionsFeatures),
     id_features_subset='ftr1',cex.main=2,cex.axis=1.2,cex.lab=1.2)
```

Finally, a plot of pointwise quantile curves can suggest which is the most appropriate test statistics to be employed. We refer to this plot as pointwise boxplot, to intuitively indicate that it summarizes distributions in a schematic way, as the classical boxplot

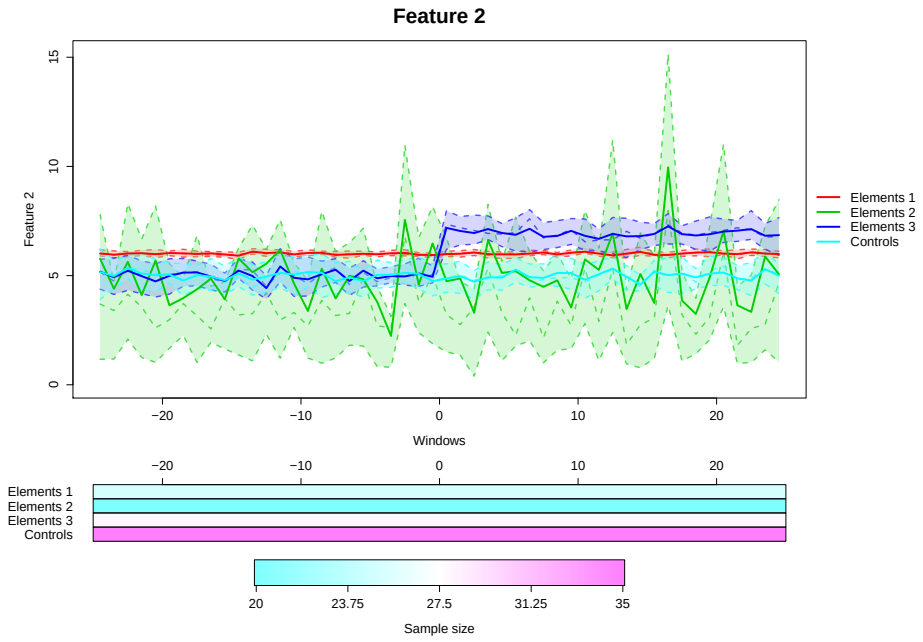


Figure 6: Pointwise boxplot of all the curves for `Feature 2`. The pointwise boxplots corresponding to different region datasets are drawn with different colors, with the mean curves as solid lines and the quartile curves in dashed lines over shaded areas. At the bottom of the plot, the heatmap shows the sample size corresponding to each position and each region dataset.

does. As an example, the next chunk of code produces the plot in Figure 6 for `Feature 2` in all the region datasets under consideration. The default plot shows the mean curves in solid lines and the quartile curves (corresponding to 25%, 50% and 75% of the data in each position) in dashed lines over shaded areas.

```
plot(regionsFeatures,type='boxplot',
     id_features_subset='ftr2',cex.main=2,cex.axis=1.2,cex.lab=1.2)
```

We note that, in the example, a test statistic based on the mean or the median difference effectively captures differences in the distributions of `Feature 2` between `Elements 1` and `Controls`, as well as between `Elements 3` and `Controls`. However, these test statistics are not effective in capturing the difference between `Element 2` and `Controls`. In this case the difference concerns variability instead of the mean values, hence test statistics based on the variance ratio or on multiple quantile curves are probably more appropriate.

4 Testing for differences between curve distributions

The main function of IWTomics package is `IWTomicsTest`, which implements the Interval-Wise Testing for "Omics" data and allows the user to detect genomic features that are relevant in discriminating different groups of regions. The main output of this function is an adjusted p-value curve for each test performed, consisting of an adjusted p-value for each fixed-size window corresponding to the provided resolution (i.e. 2 kb in the example). The correction of each p-value is done considering all the intervals containing the window, with length up to the maximum scale considered. Details about the statistical methodology employed here can be found in `?`.

The function `IWTomicsTest` takes as input an `"IWTomicsData"` object and can handle, in a single call, several tests between different region datasets (both one sample and two samples tests) and multiple genomic features. In particular, the genomic features that we wish to test are provided by the vector `id_features_subset`, while the vectors `id_region1` and `id_region2` contain the identifiers of the region datasets to be compared. To perform a one sample test, the empty string should be inserted in `id_region2`. Another essential argument of function `IWTomicsTest` is the test statistics to be used (`statistics`). Possible test statistics are `mean` (default, based on the difference between the mean curves), `median` (based on the difference between the median curves), `variance` (based on the ratio between the variance curves) and `quantile` (based on the difference between quantile curves). When the `quantile` statistic is selected, the desired probabilities are provided through the `probs` argument. If multiple probabilities are given, the test statistic is the sum of the quantile statistics corresponding to the different probabilities. This option is more time consuming, but it can usually capture subtler differences since it considers the distributions more comprehensively.

In the next chunk of code we illustrate the use of the one sample test. The `mean` statistic is employed to assess whether the center of symmetry of `Feature 1` is equal to 0 and whether the center of symmetry of `Feature 2` is equal to 5 in `Controls`. As

a result of the test we obtain an adjusted p-value curve, i.e. an adjusted p-value for each window in the region dataset, for each test performed. In particular, the function `IWTomicsTest` returns an `"IWTomicsData"` object with the test input and results in the slot `test`. The adjusted p-values of the different tests can be accessed with the method `adjusted_pval`.

```
result1=IWTomicsTest(regionsFeatures,mu=c(0,5),
                     id_region1='control',id_region2='',
                     id_features_subset=c('ftr1','ftr2'))

## Performing IWT for 'Controls'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...
##  Performing IWT for feature 'Feature 2'...
##    Point-wise tests...
##    Interval-wise tests...

result1

## IWTomicsData object with 1 region dataset with center alignment, and 2 features:
## Regions:
##  Controls:  35 regions
## Features:
##  Feature 1:  2000 bp resolution
##  Feature 2:  2000 bp resolution
## 1 test present, with mean statistics, for features Feature 1, Feature 2:
##  One sample test:  Controls

adjusted_pval(result1)

## $test1
## $test1$ftr1
##  [1] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
## [11] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
## [21] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
## [31] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
## [41] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
##
## $test1$ftr2
##  [1] 0.840 0.886 0.840 0.944 0.944 0.944 0.944 0.955 0.955 0.944
## [11] 0.944 0.944 0.944 0.944 0.944 0.944 0.944 0.906 0.906 0.905
## [21] 0.939 0.939 0.905 0.905 0.897 0.897 0.897 0.891 0.891 0.891
## [31] 0.891 0.891 0.891 0.891 0.891 0.891 0.891 0.804 0.804 0.726
## [41] 0.883 0.960 0.960 0.960 0.960 0.936 0.936 0.842 0.755 0.987
```

The results indicate that the null hypothesis that the center of symmetry of `Feature 1` is equal to 0 should be rejected at the usual level of confidence of 5% in the whole

region, while we do not have enough evidence to conclude that the center of symmetry of `Feature 2` is not 5.

A more common problem is to detect differences in feature distributions between two different group of regions, i.e. to perform a two sample test. In the next chunk of code we illustrate the use of the two sample test (`mean` statistic) for differences in the curve distributions of `Feature 1` and `Feature 2` in the comparisons `Elements 1` vs. `Controls`, `Elements 2` vs. `Controls` and `Elements 3` vs. `Controls`.

```
result2_mean=IWTomicsTest(regionsFeatures,
                          id_region1=c('elem1','elem2','elem3'),
                          id_region2=c('control','control','control'))
```

```
## Performing IWT for 'Elements 1' vs.  'Controls'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...
##  Performing IWT for feature 'Feature 2'...
##    Point-wise tests...
##    Interval-wise tests...
## Performing IWT for 'Elements 2' vs.  'Controls'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...
##  Performing IWT for feature 'Feature 2'...
##    Point-wise tests...
##    Interval-wise tests...
## Performing IWT for 'Elements 3' vs.  'Controls'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...
##  Performing IWT for feature 'Feature 2'...
##    Point-wise tests...
##    Interval-wise tests...
```

```
result2_mean
```

```
## IWTomicsData object with 4 region datasets with center alignment, and 2 features:
## Regions:
##  Elements 1:  25 regions
##  Elements 2:  20 regions
##  Elements 3:  28 regions
##    Controls:  35 regions
## Features:
##  Feature 1:  2000 bp resolution
##  Feature 2:  2000 bp resolution
## 3 tests present, with mean statistics, for features Feature 1, Feature 2:
##  Two sample test:  Elements 1  vs  Controls
##  Two sample test:  Elements 2  vs  Controls
##  Two sample test:  Elements 3  vs  Controls
```

```
adjusted_pval(result2_mean)
```

```
## $test1
## $test1$ftr1
##  [1] 0.965 0.965 0.993 0.993 0.993 0.993 0.993 0.983 0.983 0.983
## [11] 0.983 0.983 0.983 0.983 0.989 0.989 0.991 0.991 0.993 0.993
## [21] 0.999 0.999 0.999 0.969 0.969 0.969 0.984 0.902 0.895 0.895
## [31] 0.895 0.854 0.854 0.791 0.864 0.849 0.836 0.836 0.874 0.874
## [41] 0.836 0.836 0.836 0.836 0.836 0.836 0.836 0.836 0.836 0.836
##
## $test1$ftr2
##  [1] 0.002 0.001 0.003 0.001 0.001 0.001 0.001 0.001 0.001 0.001
## [11] 0.001 0.001 0.001 0.001 0.002 0.001 0.003 0.001 0.001 0.001
## [21] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
## [31] 0.003 0.001 0.001 0.001 0.001 0.001 0.001 0.002 0.001 0.001
## [41] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.003 0.001
##
##
## $test2
## $test2$ftr1
##  [1] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
## [11] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
## [21] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
## [31] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
## [41] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
##
## $test2$ftr2
##  [1] 0.850 0.850 0.850 0.850 0.850 0.850 0.874 0.908 0.947 0.908
## [11] 0.908 0.908 0.908 0.908 0.908 0.777 0.893 0.893 0.990 0.990
## [21] 0.684 0.159 0.106 0.418 0.418 0.981 0.981 0.642 0.707 0.997
## [31] 0.997 0.997 0.997 0.997 0.775 0.775 0.858 0.507 0.469 0.784
## [41] 0.307 0.002 0.088 0.067 0.823 0.171 0.366 0.482 0.889 0.945
##
##
## $test3
## $test3$ftr1
##  [1] 0.224 0.610 0.628 0.798 0.457 0.906 0.940 0.906 0.906 0.615
## [11] 0.615 0.541 0.541 0.344 0.333 0.001 0.001 0.001 0.001 0.001
## [21] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
## [31] 0.001 0.001 0.001 0.001 0.001 0.582 0.582 0.582 0.535 0.851
## [41] 0.851 0.921 0.921 0.979 0.914 0.715 0.935 0.935 0.844 0.844
##
## $test3$ftr2
##  [1] 0.985 0.985 0.976 0.976 0.972 0.972 0.972 0.972 0.990 0.983
## [11] 0.972 0.972 0.828 0.828 0.828 0.828 0.830 0.667 0.667 0.666
## [21] 0.883 0.896 0.896 0.970 0.860 0.001 0.001 0.001 0.001 0.001
## [31] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
## [41] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
```

The adjusted p-value curves suggest that we cannot reject the null hypothesis that `Feature 1` has the same distributions in `Elements 1` and `Controls`, and that its distribution in `Elements 2` differs from the one in `Controls`, across the whole region. On the contrary, `Feature 1` in `Elements 3` has a significantly different distribution than in `Controls`, but this difference can be imputed to a specific portion of the region; the 20 windows corresponding to the 40 kb around the center of the region. The adjusted p-values also suggest that `Feature 2` is significant in distinguishing between `Elements 1` and `Controls` and not significant in distinguishing between `Elements 2` and `Controls`. Finally, `Elements 3` appears to differ from `Controls` in terms of `Feature 2` exclusively in the right part of the regions.

The next example shows how to perform the same two sample test on `Feature 1`, using the `quantile` test statistic with multiple probabilities (in particular, using first and third quartiles). The same conclusions can be drawn.

```
result2_quantiles=IWTomicsTest(regionsFeatures,
                               id_region1=c('elem1','elem2','elem3'),
                               id_region2=c('control','control','control'),
                               id_features_subset='ftr1',
                               statistics='quantile',probs=c(0.25,0.75))
```

```
## Performing IWT for 'Elements 1' vs.  'Controls'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...
## Performing IWT for 'Elements 2' vs.  'Controls'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...
## Performing IWT for 'Elements 3' vs.  'Controls'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...
```

```
result2_quantiles
```

```
## IWTomicsData object with 4 region datasets with center alignment, and 1 feature:
## Regions:
##  Elements 1:  25 regions
##  Elements 2:  20 regions
##  Elements 3:  28 regions
##    Controls:  35 regions
## Features:
##  Feature 1:  2000 bp resolution
## 3 tests present, with quantile statistics, for feature Feature 1:
##  Two sample test:  Elements 1  vs  Controls
##  Two sample test:  Elements 2  vs  Controls
##  Two sample test:  Elements 3  vs  Controls
```

```
adjusted_pval(result2_quantiles)
```

```
## $test1
## $test1$ftr1
##  [1] 0.987 0.987 0.987 0.987 0.987 0.987 0.987 0.987 0.987 0.987
## [11] 0.984 0.984 0.984 0.984 0.984 0.984 0.984 0.984 0.984 0.984
## [21] 0.984 0.984 0.984 0.984 0.984 0.977 0.977 0.952 0.945 0.945
## [31] 0.945 0.941 0.941 0.923 0.923 0.923 0.932 0.965 0.972 0.972
## [41] 0.972 0.972 0.972 0.972 0.990 0.918 0.918 0.918 0.902 0.902
##
##
## $test2
## $test2$ftr1
##  [1] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
## [11] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
## [21] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
## [31] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
## [41] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
##
##
## $test3
## $test3$ftr1
##  [1] 0.493 0.739 0.739 0.739 0.095 0.631 0.160 0.547 0.547 0.238
## [11] 0.266 0.397 0.243 0.047 0.201 0.001 0.001 0.001 0.001 0.001
## [21] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
## [31] 0.001 0.001 0.001 0.001 0.001 0.062 0.498 0.339 0.036 0.094
## [41] 0.505 0.777 0.212 0.549 0.456 0.190 0.820 0.621 0.111 0.135
```

4.1 Maximum scale considered

The argument `max_scale` can be used to set the maximum scale at which the Interval-Wise Testing is performed. That is, `max_scale` represents the maximum interval length used to adjust the p-values, i.e. the maximum number of consecutive windows to be employed. As default, `IWTomicsTest` sets the maximum scale to the length of the whole region under consideration. In the examples above, the default `max_scale` is 50 (since all features are measured in 50 consecutive windows in each region), and it can be set to each integer from 1 (no adjustment) to 50. If data comprises regions of different length, the dafault is the length of the union of all the regions, i.e. the length of the largest region on which the test can be performed. The Interval-Wise Testing is performed for all scales ranging from the window length (this is the measurement resolution, hence the smallest scale possible – no p-value correction) to the region length (adjusting the p-values in order to control the interval-wise error rate over all intervals up to the whole region).

4.2 Number of permutations

The desired number of random permutations (the number of iterations of the Monte Carlo algorithm) employed to obtain the empirical p-value curves is provided to the function `IWTomicsTest` through the argument `B`. When `B` is greater than the total number P of permutations leading to different values of the test statistics, exact permutational p-values are computed.

It should be noted that the resolution of the computed p-values depends both on `B` and P. If all permutations are explored, the resolution of the exact p-values is $1/P$. If a Monte Carlo algorithm is employed, an approximated p-value with resolution $1/$`B` is computed. As a consequence, a large number of permutations leads to more accurate results and an approximation of the p-value curves closer to the theoretical ones. However, performing a test with a large number of permutations can be very time consuming, depending on the chosen test statistics and on the sample sizes.

4.3 Reproducibility

Since the function `IWTomicsTest` uses random permutations to compute the empirical p-value curves, it is necessary to use `set.seed` to create a reproducible code that returns exactly the same p-value curves every time it is run.

```
set.seed(16)
result_rep1=IWTomicsTest(regionsFeatures,
                         id_region1='elem1',id_region2='control',
                         id_features_subset='ftr1')

## Performing IWT for 'Elements 1' vs.  'Controls'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...

adjusted_pval(result_rep1)

## $test1
## $test1$ftr1
##  [1] 0.964 0.964 0.998 0.998 0.998 0.998 0.998 0.988 0.988 0.988
## [11] 0.988 0.988 0.988 0.988 0.988 0.988 0.996 0.996 0.996 0.996
## [21] 0.999 0.999 0.999 0.971 0.971 0.971 0.981 0.914 0.897 0.897
## [31] 0.897 0.838 0.838 0.784 0.858 0.846 0.828 0.828 0.855 0.855
## [41] 0.828 0.828 0.828 0.828 0.828 0.828 0.828 0.828 0.828 0.828

set.seed(16)
result_rep2=IWTomicsTest(regionsFeatures,
                         id_region1='elem1',id_region2='control',
                         id_features_subset='ftr1')
```

```
## Performing IWT for 'Elements 1' vs.  'Controls'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...

adjusted_pval(result_rep2)

## $test1
## $test1$ftr1
##  [1] 0.964 0.964 0.998 0.998 0.998 0.998 0.998 0.988 0.988 0.988
## [11] 0.988 0.988 0.988 0.988 0.988 0.988 0.996 0.996 0.996 0.996
## [21] 0.999 0.999 0.999 0.971 0.971 0.971 0.981 0.914 0.897 0.897
## [31] 0.897 0.838 0.838 0.784 0.858 0.846 0.828 0.828 0.855 0.855
## [41] 0.828 0.828 0.828 0.828 0.828 0.828 0.828 0.828 0.828 0.828

identical(result_rep1,result_rep2)

## [1] TRUE
```

4.4 Not fully computable p-value curves

The package IWTomics is designed to work with genomic regions of different length as well as with missing measurements in some of the windows (indicated with `NA`). On one hand, this property allows the user to work directly with genomic annotations of different sizes (aligned according to a landmark or scaled to the same length) without needing to artificially cut them or to insert them in fixed size regions. On the other hand, this can lead to the presence of very few measurements in some locations or portions of the regions under consideration (e.g., at the boundaries of the longest regions). It is important to notice that the IWT has low power in detecting differences at these locations with small sample sizes. In addition, p-value curves may not be fully computable if, in a particular portion of the regions, the number of `NA` measurements exceeds the sample size of one of the two groups. Indeed, in this case there exist some permutations of the observations that show measurements exclusively for one of the two groups, and all `NA` for the other group. The function `IWTomicsTest` computes the p-values corresponding to these locations considering only the permutations that retain measurements in both groups, and generates a warning message.

This issue is demonstrated in the following example, using the synthetic region dataset `regionsFeatures_center` provided with the package. The region dataset `Elements 1` has regions of different lengths, and their maximum length is lower than the maximum length of `Controls` regions. Moreover, `NA` are present in `Feature 1` measurements in the last windows of several `Elements 1` regions.

```
data(regionsFeatures_center)
range(width(regions(regionsFeatures_center)))

##           [,1]   [,2]
```

```
## elem1    84000  98000
## elem2   100000 100000
## elem3   100000 100000
## control  84000 100000
```

```
plot_data=plot(regionsFeatures_center,type='boxplot',
               id_regions_subset=c('elem1','control'),
               id_features_subset='ftr1',size=TRUE)
plot_data$features_position_size
```

```
## $ftr1
##       control elem1
##  [1,]      31     0
##  [2,]      31    12
##  [3,]      33    22
##  [4,]      34    24
##  [5,]      35    25
##  [6,]      34    25
##  [7,]      35    25
##  [8,]      34    25
##  [9,]      34    25
## [10,]      35    24
## [11,]      35    24
## [12,]      35    25
## [13,]      35    25
## [14,]      35    24
## [15,]      34    25
## [16,]      33    25
## [17,]      34    25
## [18,]      33    25
## [19,]      35    23
## [20,]      35    25
## [21,]      35    25
## [22,]      35    25
## [23,]      34    25
## [24,]      35    25
## [25,]      35    25
## [26,]      35    25
## [27,]      35    25
## [28,]      34    25
## [29,]      35    25
## [30,]      34    25
## [31,]      35    25
## [32,]      35    25
## [33,]      35    25
## [34,]      35    25
## [35,]      35    25
## [36,]      35    25
```

```
## [37,]      34    25
## [38,]      35    25
## [39,]      34    25
## [40,]      35    25
## [41,]      35    25
## [42,]      35    25
## [43,]      35    25
## [44,]      35    25
## [45,]      35    25
## [46,]      35    25
## [47,]      34    24
## [48,]      34    23
## [49,]      32    16
## [50,]      31     3
```

Looking at the pointwise sample sizes (Figure 7), we can observe that the IWT cannot be performed in the first window (no measurements present for `Element 1`). In the last window the IWT can be performed but the p-value will be computed using only a subset of the possible permutations, hence a warning will be generated.

```
result_warning=IWTomicsTest(regionsFeatures_center,
                            id_region1='elem1',id_region2='control',
                            id_features_subset='ftr1')

## Performing IWT for 'Elements 1' vs.  'Controls'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...
## Warning:  p-value not fully computable in some points, because of too many NAs
present.

adjusted_pval(result_warning)

## $test1
## $test1$ftr1
##  [1]    NA 0.982 0.989 0.990 0.990 0.990 0.997 0.997 0.997 0.999
## [11] 0.999 0.999 0.999 0.999 0.999 0.999 0.999 0.998 0.998 0.998
## [21] 0.962 0.363 0.032 0.009 0.001 0.001 0.001 0.001 0.217 0.445
## [31] 0.887 0.887 0.839 0.802 0.978 0.978 0.912 0.966 0.966 0.966
## [41] 0.966 0.966 0.966 0.955 0.955 0.955 0.955 0.955 0.736 0.419
```

5 Visualizing test results

The package IWTomics offers two different ways of representing IWT results graphically. The `plotTest` function creates detailed plots of the IWT results. For each test per-

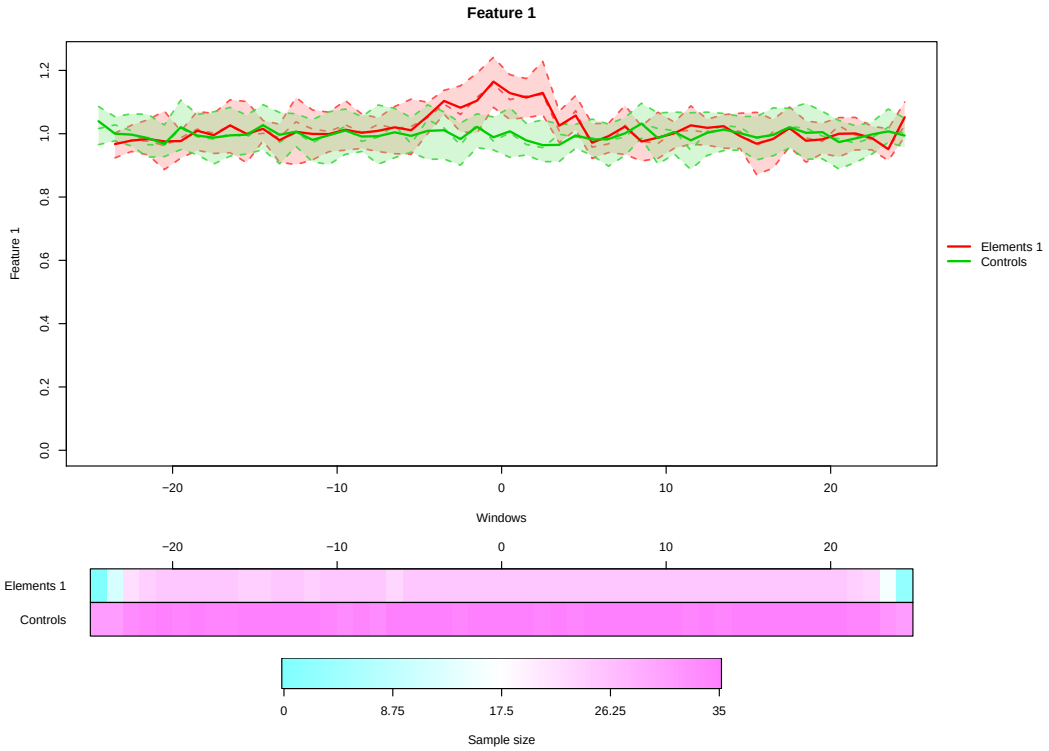


Figure 7: Pointwise boxplot of the curves corresponding to `Element 1` and `Control` for `Feature 1`. The heatmap at the bottom shows that the sample size varies in the different positions.

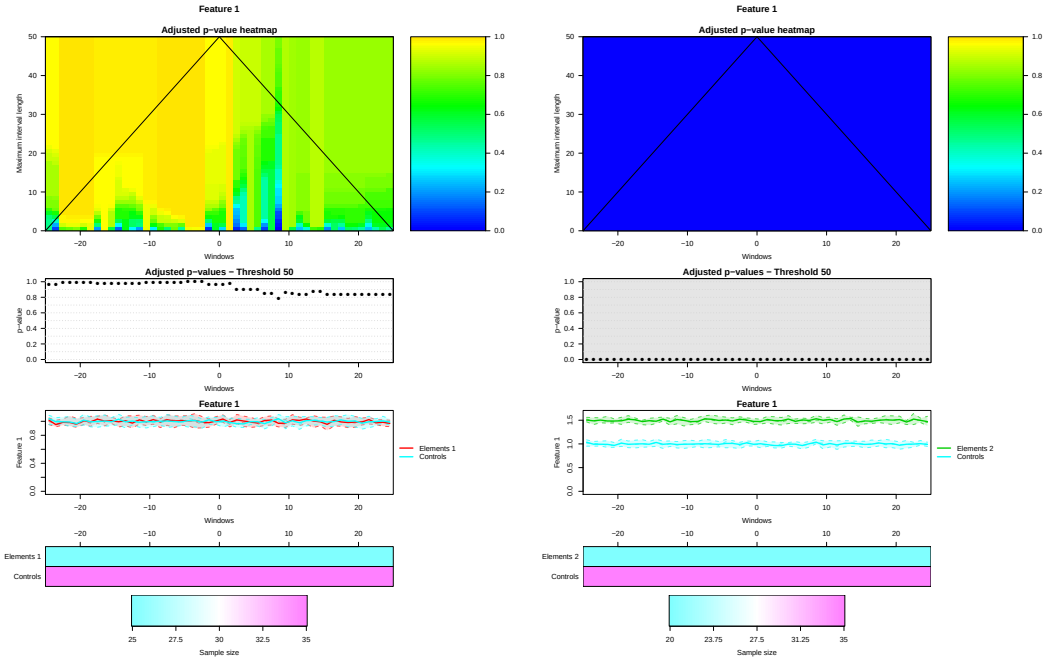
formed, or possibly for a subset of the feature tested (argument `id_features_subset`), it produces:

- a heatmap of the adjusted p-value curves at each scale, from the measurement resolution to the maximum scale considered;
- a plot of the adjusted p-value curve at the chosen scale threshold (provided by `scale_threshold`). Significant windows (corrected p-values $< \alpha$) are highlighted in gray;
- a plot of the feature measurements in the region dataset(s) tested. This can be either a plot of the aligned measurement curves (`type="curves"`) or a pointwise boxplot (`type="boxplot"`, default), with the options provided by the `plot` method of `IWTomicsData` class, for example the possibility to plot the pointwise sample size, or the average curve.

These plots are useful to visualize and interpret IWT results and select the relevant scale (see Section 7). In the next chunk of code we show how to produce plots of IWT results (`mean` statistic) for `Feature 1` in the comparisons `Elements 1` vs. `Controls`, `Elements 2` vs. `Controls` and `Elements 3` vs. `Controls`. The plots are shown in Figure 8.

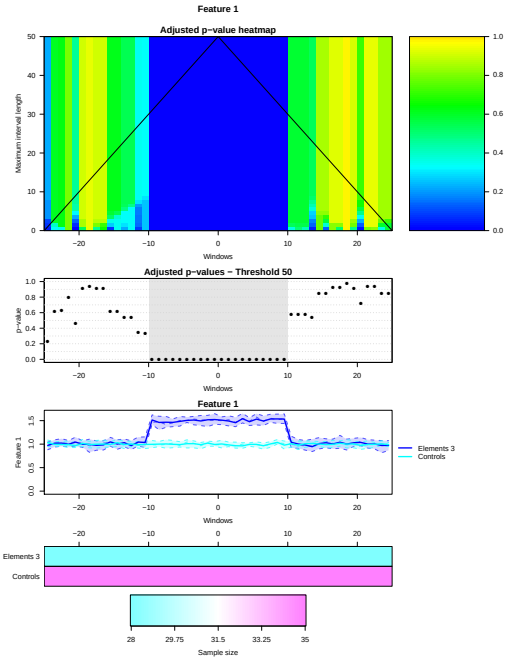
```
plotTest(result2_mean,alpha=0.05,id_features_subset='ftr1')
```

An additional way to visualize results of the IWTomics package is provided by



(a)

(b)



(c)

Figure 8: Plots of IWT results for `Feature 1` in the comparisons: (a) `Elements 1` vs `Controls`, (b) `Elements 2` vs `Controls` and (c) `Elements 3` vs `Controls`.

the function `plotSummary`. This creates a graphical summary of the test results, grouped by the region datasets tested (`groupby="test"`) or by the feature tested (`groupby="feature"`). In particular, it creates a heatmap of the adjusted p-value curves at the chosen thresholds (argument `scale_threshold`) for each group of tests. When the grouping is `"test"` one plot is drawn for each comparison the different raws in the heatmap correspond to the different features. On the contrary, when the tests are grouped by `"feature"` one plot is drawn for each feature and the different raws represent different comparisons. Color intensity is proportional to $-log$(p-value), i.e. more intense colors correspond to lower p-values. Red means that the test statistics is higher in the first dataset tested than in the second one (or that it is positive in one sample test), while blue means that it is lower in the first dataset tested than in the second one (or it is negative in one sample test). Finally, white means that the p-value in that window is not significant, according to the selected threshold α (p-value $\geq \alpha$).

The function `plotSummary` uses a modified version of `pheatmap` package functions, hence it borrows from this package some of the arguments. In particular, `gaps_features` and `gaps_tests` allow the user to insert gaps in the heatmap between features or tests when `groupby` is equal to `"test"` or `"feature"`, respectively, while `cellwidth` and `cellheight` provide the individual cell width and hight, and `filenames` correspond to the file paths where to save the produced plots. In addition, other IWTomics-specific arguments are available, in order to plot only the raw corresponding to significant p-values (`only_significant`) or to plot only a subset of tests or features (arguments `test` and `id_features_subset`).

In the following example we show how to create a summary plot of IWT results (`mean` statistic) in the comparisons `Elements 1` vs. `Controls`, `Elements 2` vs. `Controls` and `Elements 3` vs. `Controls`, grouped by feature (see Figure 9).

```
plotSummary(result2_mean,alpha=0.05,groupby='feature',align_lab='Center')
```

The next chunk of code and Figure 10 show an example of summary plot for one sample IWT results, grouped by feature. In particular, for each feature we are testing whether its center of symmetry is equal to 1, in `Elements 1`, `Elements 2`, `Elements 3` and `Controls`.

```
result1_mu1=IWTomicsTest(regionsFeatures,mu=1,
                         id_region1=c('elem1','elem2','elem3','control'))

## Performing IWT for 'Elements 1'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...
##  Performing IWT for feature 'Feature 2'...
##    Point-wise tests...
##    Interval-wise tests...
## Performing IWT for 'Elements 2'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
```

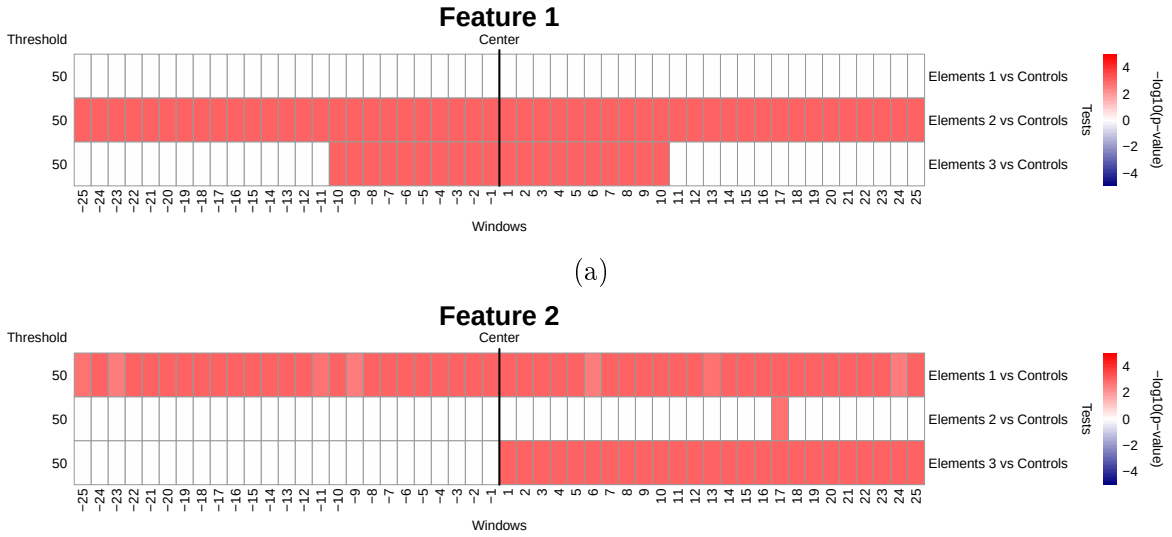


Figure 9: Summary plot of IWT results in the comparisons `Elements 1` vs `Controls`, `Elements 2` vs `Controls` and `Elements 3` vs `Controls`, grouped by (a) `Feature 1` and (b) `Feature 2`.

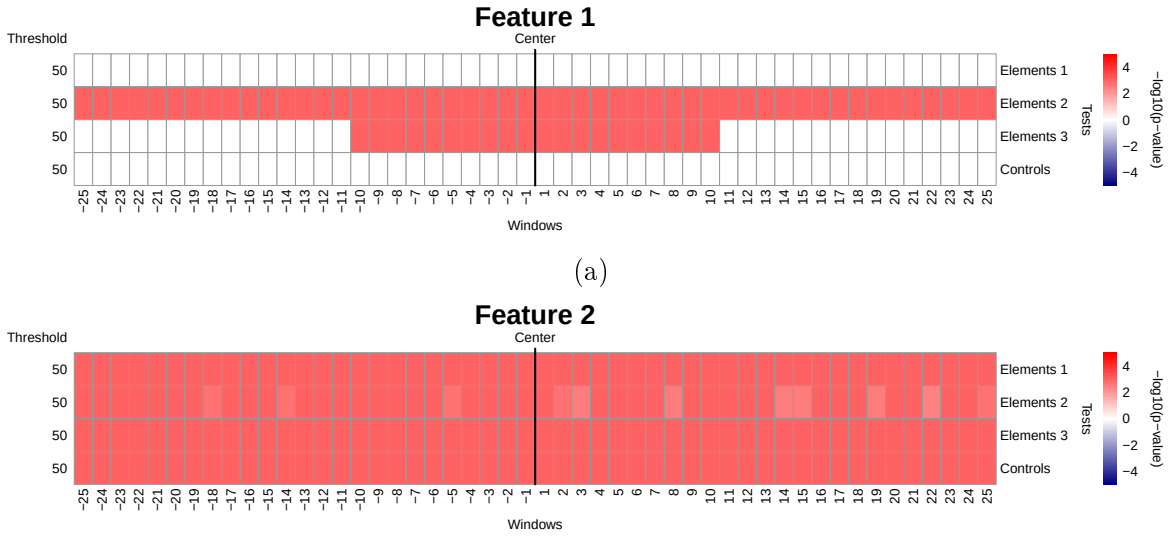


Figure 10: Summary plots of IWT results about the center of symmetry of the distributions of the different region datasets being equal to 1, grouped by (a) `Feature 1` and (b) `Feature 2`.

```
##    Interval-wise tests...
##  Performing IWT for feature 'Feature 2'...
##    Point-wise tests...
##    Interval-wise tests...
## Performing IWT for 'Elements 3'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...
##  Performing IWT for feature 'Feature 2'...
##    Point-wise tests...
##    Interval-wise tests...
## Performing IWT for 'Controls'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...
##  Performing IWT for feature 'Feature 2'...
##    Point-wise tests...
##    Interval-wise tests...
```

```
plotSummary(result1_mu1,alpha=0.05,groupby='feature',align_lab='Center')
```

We can conclude that `Feature 2` has a center of symmetry different from 1 in all region datasets (Figure 10(b)), while `Feature 1` has a center of symmetry different from 1 in `Elements 2` and in the central part of `Elements 1` regions. The corresponding plots grouped by location are shown in Figure 11.

```
plotSummary(result1_mu1,alpha=0.05,groupby='test',align_lab='Center')
```

Finally, the next chunk of code generates a summary plot (Figure 12), grouped by feature, for a more complicated set of 10 tests including both two samples and one sample tests (with `mu=0`) about `Feature 1`. Only significant tests are shown in the heatmap, and gaps between tests are added to separate the different types of tests.

```
result_many=IWTomicsTest(regionsFeatures,
                         id_region1=c('elem1','elem2','elem3',
                                      'elem1','elem1','elem2',
                                      'elem1','elem2','elem3','control'),
                         id_region2=c(rep('control',3),
                                      'elem2','elem3','elem3',
                                      rep('',4)),
                         id_features_subset='ftr1')
```

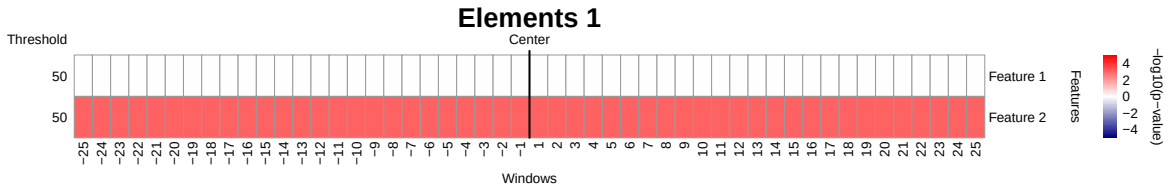
```
## Performing IWT for 'Elements 1' vs.  'Controls'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...
## Performing IWT for 'Elements 2' vs.  'Controls'...
```

```
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...
## Performing IWT for 'Elements 3' vs.  'Controls'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...
## Performing IWT for 'Elements 1' vs.  'Elements 2'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...
## Performing IWT for 'Elements 1' vs.  'Elements 3'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...
## Performing IWT for 'Elements 2' vs.  'Elements 3'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...
## Performing IWT for 'Elements 1'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...
## Performing IWT for 'Elements 2'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...
## Performing IWT for 'Elements 3'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...
## Performing IWT for 'Controls'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...
```

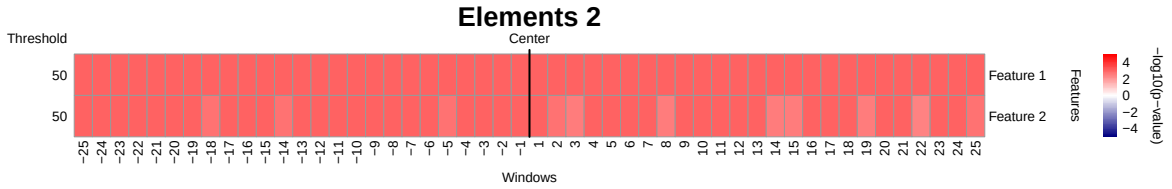
```
plotSummary(result_many,alpha=0.05,groupby='feature',
            align_lab='Center',gaps_tests=c(3,6),
            only_significant=TRUE)
```

6 Scaling regions and smoothing curves

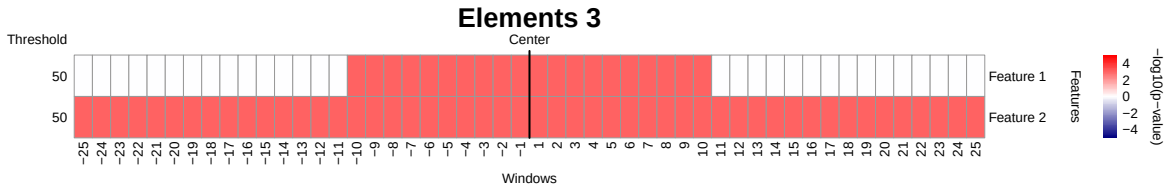
When the genomic regions have different lengths and the `"scale"` alignment is considered, the Interval-Wise Testing cannot be applied directly. Indeed, the function `IWTomicsTest` requires that all the curves considered are evaluated on the same grid (possibly with some `NA`s) and this is not the case for scaled regions of different lengths.



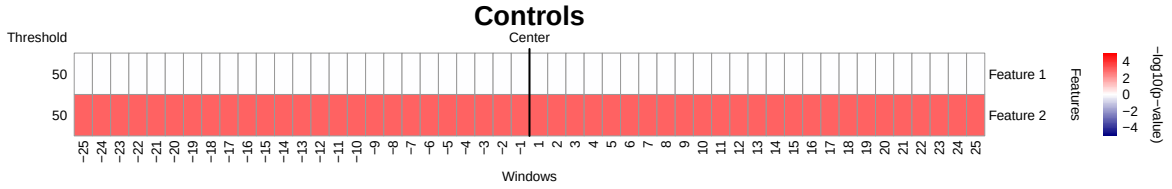
(a)



(b)



(c)



(d)

Figure 11: Summary plots of IWT results about the center of symmetry of the distributions of the different region datasets being equal to 1, grouped by the region datasets (a) `Elements 1`, (b) `Elements 2`, (c) `Elements 3` and (d) `Controls`.

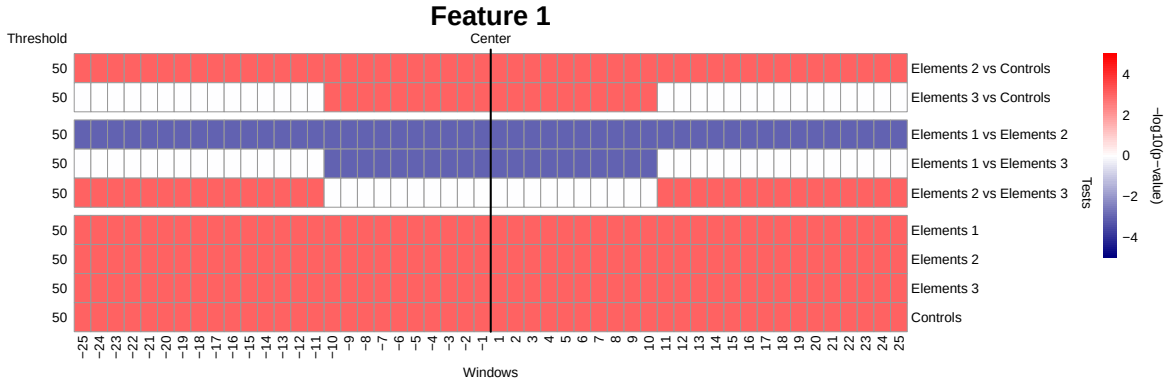


Figure 12: Summary plot of IWT results about a set of 10 tests on `Feature 1`.

The `smooth` function for `"IWTomicsData"` objects provided by IWTomics can be used to scale regions and to obtain measurements on the same grid. This function employs a smoothing technique to construct a functional object (a curve) starting from the discrete measurements, and then evaluates the smoothed curves on a grid (the same grid for all the curves in a dataset). Possible types of smoothing (argument `type`) are local polynomials (`"locpoly"`), Nadaraya-Watson kernel smoothing with Gaussian kernel (`"kernel"`) and regression b-splines (`"splines"`, computational expensive when regions have different length and/or gaps). Smoothing parameters can be chosen through the arguments `bandwidth`, `degree` and `dist_knots`. The number of equally-spaced grid points over which the smoothed curves should be evaluated is supplied by the argument `scale_grid`, with default choice being the maximum number of measurements of the original datasets.

In the following example, we consider the dataset `regionsFeatures_scale` provided with the package, whose regions have different lengths and contain gaps (`NA` measurements). The goal is to perform a two sample test to compare the distribution of `Feature 1` in `Element 1` vs. `Control`, `Element 2` vs. `Control` and `Element 3` vs. `Control`. The `Feature 1` curves are shown in Figure 13. Note that the pointwise sample sizes and the average cuves are not shown in this figure, because of the different grids over which the scaled curves are evaluated.

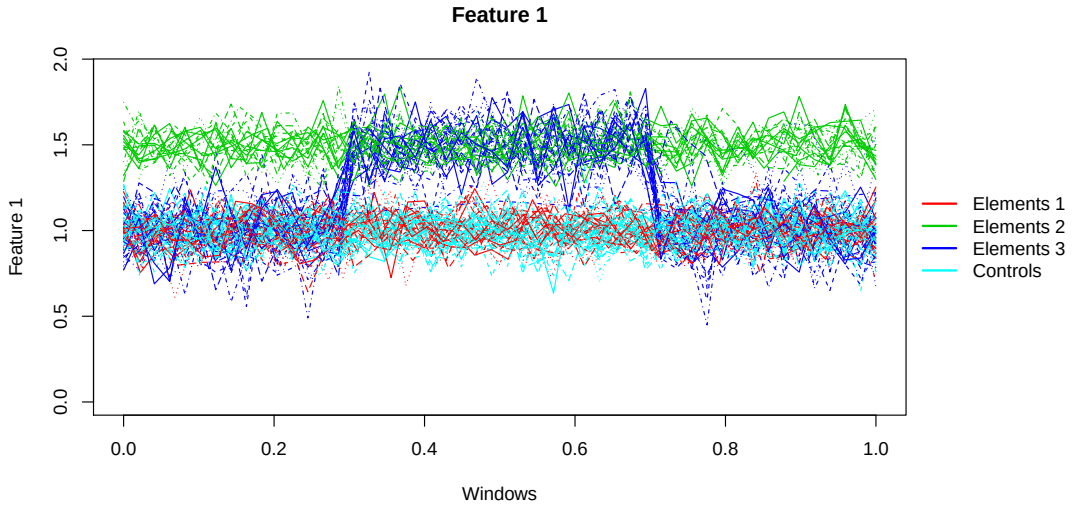


Figure 13: Plot of all the scaled curves for `Feature 1`.

```
data(regionsFeatures_scale)
lengthFeatures(regionsFeatures_scale[,'ftr1'])

## $ftr1
## $ftr1$elem1
##  [1] 40 49 30 50 49 48 49 40 48 48 48 48 46 47 45 45 40 48 45 46
## [21] 45 44 49 25 49
```

```
##
## $ftr1$elem2
##  [1] 50 46 50 50 50 50 50 50 40 35 50 50 50 50 48 50 50 50 50 50
##
## $ftr1$elem3
##  [1] 50 50 50  5 50 50 45 50 50 50 50 50 35 50 50 50 50 50 50 49
## [21] 50 50 50 50 50 50 50 50
##
## $ftr1$control
##  [1] 50 47 50 50 50 50 50 50 40 50 50 50 50 50 50 50 47 50 50 50
## [21] 50 50 45 50 50 50 50 50 50 50 50 50 50 50 50 50
```

```
features(regionsFeatures_scale)[['ftr1']][['elem1']][,1]
```

```
##  [1] 0.9426239 0.7816422 0.8921658 1.1259144 0.7947027 1.1897256
##  [7] 1.0938513 1.1116840 0.7923799 0.9926907 0.9504155 0.9023977
## [13]        NA 1.1034298 0.9398509 1.0845830 1.0296121 1.0654507
## [19] 1.2272291 0.9466287 1.0814647 0.8625043 0.8986767 1.0170905
## [25] 1.1303091 0.9860600 0.9616917 0.9009518 1.0741087 0.7893998
## [31] 0.8776146 0.9386896 1.0238594 1.0895628 0.9127579 1.0833972
## [37] 1.1837634 0.9004450 1.0805632 0.9594636        NA        NA
## [43]        NA        NA        NA        NA        NA        NA
## [49]        NA        NA
```

```
plot(regionsFeatures_scale,type='curves',
     N_regions=lengthRegions(regionsFeatures_scale),
     id_features_subset='ftr1')
```

```
## Warning in .local(x, ...):  average=TRUE is incompatible with 'scale' alignment
and regions of different length.  Setting average=FALSE.
## Warning in .local(x, ...):  size=TRUE is incompatible with 'scale' alignment
and regions of different length.  Setting size=FALSE.
```

If we try to run the IWT through the function `IWTomicsTest` we get an error because the regions have different lengths, hence the scaled curves are evaluated on different grids. To obtain measurements on the same grid we employ the function `smooth` and smooth all the curves with local polynomials, with kernel bandwith of 5 windows and polynomials of degree 3 (default options). The smoothed curves are evaluated on a grid of 50 equally-spaced points (the maximum number of points in the original measurements, default option). Figure 14 shows the smoothed curves. Since the grid is now the same for all regions, the pointwise sample sizes and the mean curves are shown. Note that by applying the function `smooth` we implicitly filled all gaps in the curves.

```
regionsFeatures_scale_smooth=smooth(regionsFeatures_scale,type='locpoly')
lengthFeatures(regionsFeatures_scale_smooth['ftr1'])
```

```
## $ftr1
```

```
## $ftr1$elem1
##  [1] 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50
## [21] 50 50 50 50 50
##
## $ftr1$elem2
##  [1] 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50
##
## $ftr1$elem3
##  [1] 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50
## [21] 50 50 50 50 50 50 50 50
##
## $ftr1$control
##  [1] 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50
## [21] 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50
```

```
features(regionsFeatures_scale_smooth)[['ftr1']][['elem1']][,1]
```

```
##  [1] 0.8535742 0.8935633 0.9319494 0.9644418 0.9888147 1.0045635
##  [7] 1.0123725 1.0138506 1.0111000 1.0062664 1.0013142 0.9976003
## [13] 0.9959682 0.9967295 0.9997855 1.0047444 1.0110318 1.0179481
## [19] 1.0247111 1.0305032 1.0346161 1.0365789 1.0361511 1.0334044
## [25] 1.0286616 1.0223281 1.0148854 1.0066775 0.9980909 0.9894358
```

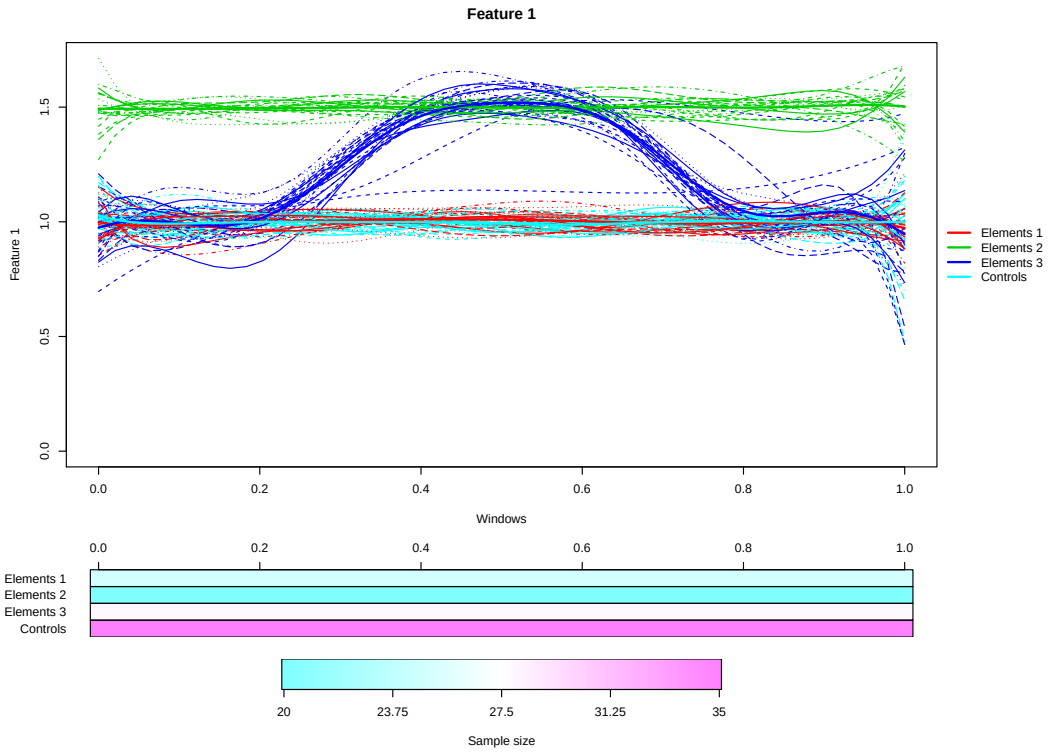


Figure 14: Plot of all the scaled curves for `Feature 1`, after obtaining measurements on the same grid with the function `smooth.IWTomics`.

```
## [31] 0.9809920 0.9729673 0.9657187 0.9595611 0.9549219 0.9523011
## [37] 0.9522055 0.9550985 0.9612534 0.9706898 0.9830838 0.9974872
## [43] 1.0128949 1.0276061 1.0394992 1.0461602 1.0450041 1.0333168
## [49] 1.0087022 0.9690481

plot(regionsFeatures_scale_smooth,type='curves',
    N_regions=lengthRegions(regionsFeatures_scale_smooth),
     id_features_subset='ftr1')
```

On this new dataset we can perform the two sample IWT test with `IWTomicsTest`, and plot the results as illustrated in Section 5 (Figure 15).

```
result=IWTomicsTest(regionsFeatures_scale_smooth,
                    id_region1=c('elem1','elem2','elem3'),
                    id_region2=c('control','control','control'),
                    id_features_subset='ftr1')

## Performing IWT for 'Elements 1' vs.  'Controls'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...
## Performing IWT for 'Elements 2' vs.  'Controls'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...
## Performing IWT for 'Elements 3' vs.  'Controls'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...

adjusted_pval(result)

## $test1
## $test1$ftr1
##  [1] 0.476 0.623 0.803 0.910 0.965 0.994 0.997 0.997 0.997 0.997
## [11] 0.997 0.999 0.997 0.988 0.963 0.933 0.883 0.840 0.777 0.697
## [21] 0.628 0.570 0.517 0.490 0.490 0.492 0.531 0.578 0.637 0.702
## [31] 0.767 0.821 0.835 0.998 0.835 0.835 0.835 0.835 0.840 0.874
## [41] 0.928 0.964 0.979 0.990 0.979 0.979 0.979 0.979 0.905 0.731
##
##
## $test2
## $test2$ftr1
##  [1] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
## [11] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
## [21] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
## [31] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
## [41] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
```

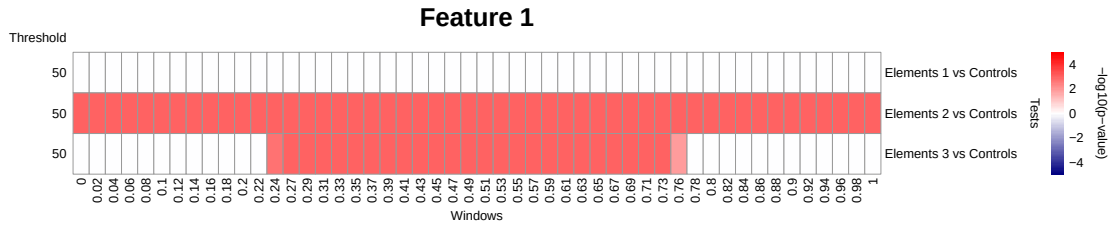


Figure 15: Summary plot of IWT results on `Feature 1`, after obtaining measurements on the same grid with the function `smooth.IWTomics`.

```
##
##
## $test3
## $test3$ftr1
##  [1] 0.247 0.686 0.769 0.729 0.729 0.959 0.729 0.680 0.680 0.886
## [11] 0.584 0.159 0.002 0.001 0.001 0.001 0.001 0.001 0.001 0.001
## [21] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001
## [31] 0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.012 0.067 0.155
## [41] 0.213 0.280 0.247 0.204 0.221 0.265 0.343 0.611 0.789 0.422

plotSummary(result,groupby='feature')
```

The function `smooth` can also be employed when the selected region alignment option is not `"scale"`. In this case it smooths the curves and filters out noise. Also here, there are three possible types of smoothing that can be chosen with the argument `type` (local polynomials, Nadaraya-Watson kernel smoothing with Gaussian kernel and regression b-splines). Smoothing options are set with `bandwidth`, `degree` and `dist_knots`. As default, measurement resolution is mainteined after smoothing, so that smoothed curves are evaluated on the same grid as the original curves. In this default the user can decide whether to fill gaps or to keep `NA`s (argument `fill_gaps`). Different measurement resolutions can be set through the argument `resolution`. If the resolution is set differently from the default, all gaps inside the curves are filled. An example is shown in the next chunk of code, where the resolution is changed from the original 2000 bp to 4000 bp for `Feature 1` and 1000 bp for `Feature 2`.

```
regionsFeatures

## IWTomicsData object with 4 region datasets with center alignment, and 2 features:
## Regions:
##  Elements 1:  25 regions
##  Elements 2:  20 regions
##  Elements 3:  28 regions
##    Controls:  35 regions
## Features:
##  Feature 1:  2000 bp resolution
##  Feature 2:  2000 bp resolution
```

```
## No tests present.

lengthFeatures(regionsFeatures)

## $ftr1
## $ftr1$elem1
##  [1] 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50
## [21] 50 50 50 50 50
##
## $ftr1$elem2
##  [1] 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50
##
## $ftr1$elem3
##  [1] 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50
## [21] 50 50 50 50 50 50 50 50
##
## $ftr1$control
##  [1] 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50
## [21] 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50
##
##
## $ftr2
## $ftr2$elem1
##  [1] 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50
## [21] 50 50 50 50 50
##
## $ftr2$elem2
##  [1] 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50
##
## $ftr2$elem3
##  [1] 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50
## [21] 50 50 50 50 50 50 50 50
##
## $ftr2$control
##  [1] 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50
## [21] 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50

regionsFeatures_smooth=smooth(regionsFeatures,type='locpoly',
                              resolution=c(4000,1000))
regionsFeatures_smooth

## IWTomicsData object with 4 region datasets with center alignment, and 2 features:
## Regions:
##  Elements 1:  25 regions
##  Elements 2:  20 regions
##  Elements 3:  28 regions
##    Controls:  35 regions
## Features:
```

```
##  Feature 1:  4000 bp resolution
##  Feature 2:  1000 bp resolution
## No tests present.

lengthFeatures(regionsFeatures_smooth)

## $ftr1
## $ftr1$elem1
##  [1] 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25
## [21] 25 25 25 25 25
##
## $ftr1$elem2
##  [1] 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25
##
## $ftr1$elem3
##  [1] 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25
## [21] 25 25 25 25 25 25 25 25
##
## $ftr1$control
##  [1] 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25
## [21] 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25
##
##
## $ftr2
## $ftr2$elem1
##  [1] 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100
## [16] 100 100 100 100 100 100 100 100 100 100
##
## $ftr2$elem2
##  [1] 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100
## [16] 100 100 100 100 100
##
## $ftr2$elem3
##  [1] 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100
## [16] 100 100 100 100 100 100 100 100 100 100 100 100
##
## $ftr2$control
##  [1] 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100
## [16] 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100
## [31] 100 100 100 100 100
```

7 Selection of relevant scale

As mentioned in Subsection 1.1, IWT does not require fixing location and scale at the outset; when identifying the features that discriminate two groups of curves, relevant

locations and scales are learned as part of the testing procedure and produced as an output.

The locations harboring a significant effect can be easily visualized by plotting the adjusted p-value curve (see Section 5). Identifying the relevant scale is less direct; it requires a careful investigation of the adjusted p-value heatmap obtained with the function `plotTest`. This heatmap shows the adjusted p-value curves at each of the scales under consideration; the bottom row shows the unadjusted p-value curve obtained by testing each position independently (smallest possible scale), while subsequent rows upward show the p-value curves adjusted up to an increasing maximum interval length (increasing scales). The top row shows the adjusted p-value curve at the maximum scale (the whole region). The heatmap can display different types of behavior. For example, Figure 8(a) shows high p-value curves for all scales and locations, suggesting that the feature does not have any significant effect. On the contrary, Figure 8(b) shows low p-value curves for all scales and locations, suggesting that the feature has a strong significant effect everywhere and acts at all scales. Finally, Figure 8(c) shows a strong effect localized in the central portion of the region and acting at all scales (see blue band around the center). However, sometimes the effect of a feature may display itself only at some particular scales. In order to illustrate this more complex situation, in the next chunk of code we employ again the dataset used in Subsection 4.4 and run the two sample version of IWT to compare `Feature 1` in `Elements 1` vs. `Controls`. Results are shown in Figure 16.

```
result_scale=IWTomicsTest(regionsFeatures_center,
                          id_region1='elem1',id_region2='control',
                          id_features_subset='ftr1')

## Performing IWT for 'Elements 1' vs.  'Controls'...
##  Performing IWT for feature 'Feature 1'...
##    Point-wise tests...
##    Interval-wise tests...
## Warning:  p-value not fully computable in some points, because of too many NAs
present.

plotTest(result_scale,alpha=0.05,scale_threshold=8)
```

Figure 16 indicates that `Feature 1`, in this example, has the ability to distinguish between `Elements 1` and `Controls` in the central part of the region. However, unlike the previous example, the blue band representing low p-values in the adjusted p-value heatmap is not fully separated from the high p-value portion of the region. Indeed, the effect of the feature appears to be more prominent at low scales (the blue portion is larger). The plot of the adjusted p-value curve at a scale threshold of 8 windows (corresponding to 16 kb), shown in the central panel of Figure 16, supports this observation suggesting that the feature acts at a relevant scale of 8 windows (16 kb).

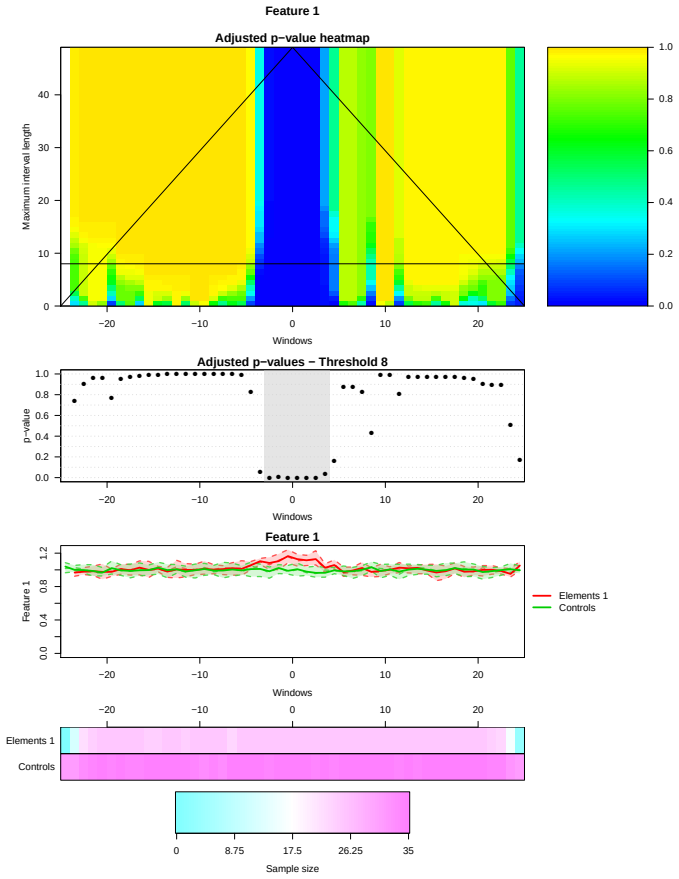


Figure 16: Plots of IWT results for `Feature 1` in the comparisons `Elements 1` vs `Controls`, with adjusted p-value curves at the relevant scale of 8 windows (i.e. 16 kb).

8 Setup

This vignette was built on:

```
sessionInfo()

## R version 3.4.0 (2017-04-21)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 16.04.2 LTS
##
## Matrix products: default
## BLAS: /home/biocbuild/bbs-3.5-bioc/R/lib/libRblas.so
## LAPACK: /home/biocbuild/bbs-3.5-bioc/R/lib/libRlapack.so
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_US.UTF-8        LC_COLLATE=C
##  [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
```

```
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] parallel  stats4    stats     graphics  grDevices utils
## [7] datasets  methods   base
##
## other attached packages:
## [1] IWTomics_1.0.0       GenomicRanges_1.28.0 GenomeInfoDb_1.12.0
## [4] IRanges_2.10.0       S4Vectors_0.14.0     BiocGenerics_0.22.0
## [7] knitr_1.15.1
##
## loaded via a namespace (and not attached):
##  [1] lattice_0.20-35         bitops_1.0-6
##  [3] grid_3.4.0              gtable_0.2.0
##  [5] magrittr_1.5            evaluate_0.10
##  [7] KernSmooth_2.23-15      highr_0.6
##  [9] zlibbioc_1.22.0         stringi_1.1.5
## [11] XVector_0.16.0          Matrix_1.2-9
## [13] splines_3.4.0           tools_3.4.0
## [15] stringr_1.2.0           RCurl_1.95-4.8
## [17] compiler_3.4.0          fda_2.4.4
## [19] GenomeInfoDbData_0.99.0
```

References

Rebeca Campos-Sánchez, Marzia A Cremona, Alessia Pini, Francesca Chiaromonte, and Kateryna D Makova. Integration and fixation preferences of human and mouse endogenous retroviruses uncovered with functional data analysis. *PLoS Computational Biology*, 12(6):1–41, 2016. doi: 10.1371/journal.pcbi.1004956.

Alessia Pini and Simone Vantini. The interval testing procedure: a general framework for inference in functional data analysis. *Biometrics*, 2016. doi: 10.1111/biom.12476.