Robust and non-robust SNHT tests for changepoint detection.

Joshua M. Browning

Abstract

This vignette provides an example on how to use the **snht** package for changepoint detection.

Keywords: SNHT, robust, time series, climate data, temperature data.

1. Methodology

The Standard Normal Homogeneity Test (SNHT) is an algorithm for detecting "changepoints" in a time-series. The word "changepoints" can take several different meanings, and so we should clarify that in this context, a changepoint is where there is a shift in the mean value of a time-series.

The SNHT test works as follows. For each observation, two means are computed: one for the N days prior to observation i, $\bar{X}_{L,i}$, and one for the N days following, $\bar{X}_{R,i}$. Then, the test statistic

$$T_i = \frac{N}{s_i^2}\left((\bar{X}_{L,i} - \bar{X}_i)^2 + (\bar{X}_{R,i} - \bar{X}_i)^2\right), \tag{1}$$

is computed where $\bar{X}_i$ is the mean of $\bar{X}_{L,i}$ and $\bar{X}_{R,i}$, and s_i is the estimated standard deviation over the N days prior and N days following observation i. If there are not N observations both before and after the current observation, no test is performed. If the largest T_i exceeds some threshold at time $i = i^*$, we conclude that a change point occurred at time i^*, and we adjust all observations after time i^* by $\bar{X}_{L,i^*} - \bar{X}_{R,i^*}$. Homogenization now proceeds iteratively. T_i is recomputed for all i that are sufficiently far away from the current change points, $i \in \{1, \ldots, n\} \setminus \{i^* - k, \ldots, i^* + k\}$, and the test is performed again until no T_i exceed the threshold, and we use $k = N$. Note that in practice, it is generally preferable to homogenize to the most recent data, as that data is considered to be more reliable, and some follow this convention Domonkos (2013).

This statistic can be problematic in the presence of outliers, as it is well known that means and standard deviation estimates can be very poor in this case. Thus, in Browning and Hering (2015) we introduce a robust variant of the SNHT statistic that replaces the above estimates of means and standard deviations with the Huber M-estimator of the mean and standard deviation Huber (2011).

2. Usage of the SNHT

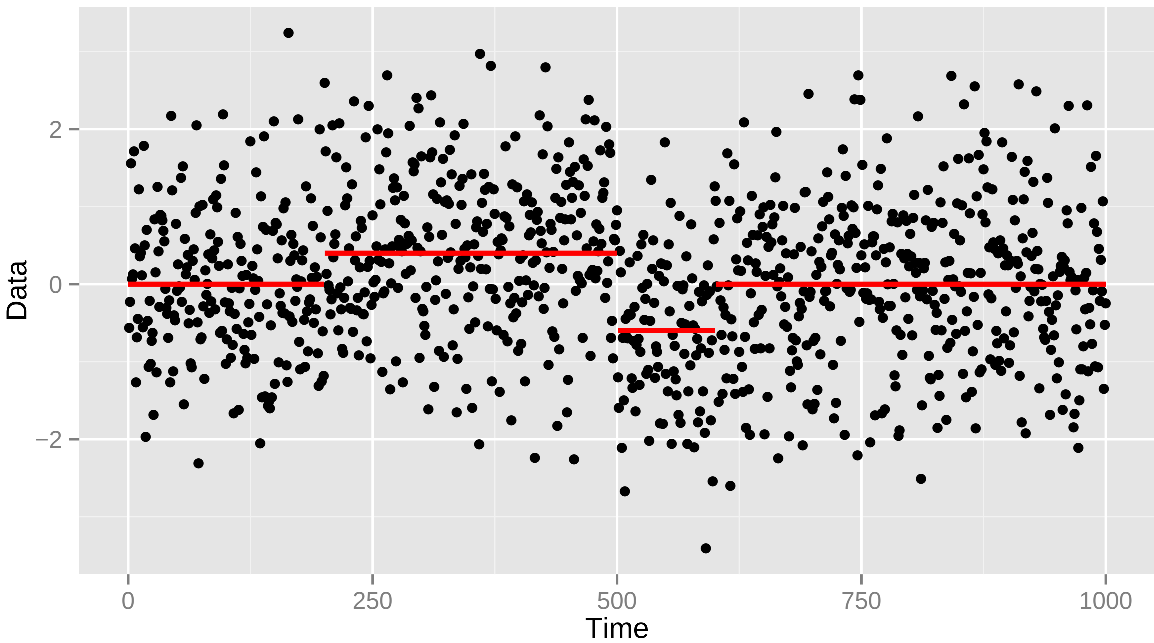
This section is intended to show several examples of how to use this package. We'll consider several different scenarios to show how the SNHT works in each different scenario.

2.1. Example 1: No seasonal trends, no outliers, equal spacing in time

For the first example, let's assume we have normal random errors with no seasonal trends, no outliers, and equal spacing in time. This is obviously a very simple/unrealistic scenario, but it shows the basics of this function.

```
set.seed(123)
baseData = rnorm(1000)
baseData[201:500] = baseData[201:500] + .4
baseData[501:600] = baseData[501:600] - .6
```

And here's a plot depicting this data:



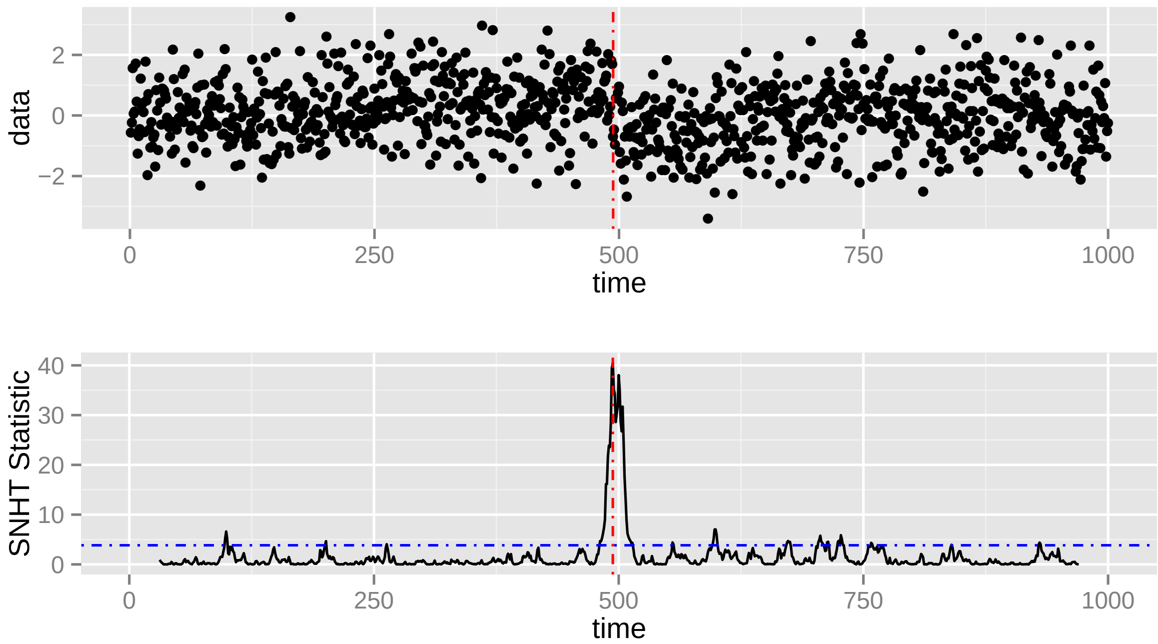
To generate the test statistics, we just use the snht function and pass in the time-series data. Since this dataset is relatively simple, we don't need to worry much about the additional arguments. We arbitrarily chose a period of 30; this specifies how many observations should be used to the left and right of a data point when computing the statistic. A larger period, P, should give a better estimate of the statistic, but the statistic cannot be computed for the first and last P observations.

```
library(snht)
snhtStatistic30 = snht(data = baseData, period = 30)
summary(snhtStatistic30)

##      score            leftMean           rightMean
##  Min.   : 0.0000   Min.   :-0.90009   Min.   :-0.90009
##  1st Qu.: 0.1111   1st Qu.:-0.10870   1st Qu.:-0.14207
##  Median : 0.5283   Median : 0.10943   Median : 0.10357
##  Mean   : 1.6375   Mean   : 0.08714   Mean   : 0.07938
##  3rd Qu.: 1.5264   3rd Qu.: 0.34558   3rd Qu.: 0.34558
##  Max.   :40.4829   Max.   : 0.81457   Max.   : 0.81457
##  NA's   :60        NA's   :60         NA's   :60
```

```
snhtStatistic60 = snht(data = baseData, period = 60)
summary(snhtStatistic60)

##      score            leftMean           rightMean
##  Min.   : 0.0000   Min.   :-0.72696   Min.   :-0.72696
##  1st Qu.: 0.4049   1st Qu.:-0.08610   1st Qu.:-0.10620
##  Median : 1.5558   Median : 0.11384   Median : 0.11249
##  Mean   : 3.3498   Mean   : 0.09017   Mean   : 0.08557
##  3rd Qu.: 3.2328   3rd Qu.: 0.32401   3rd Qu.: 0.32401
##  Max.   :46.4891   Max.   : 0.62958   Max.   : 0.62958
##  NA's   :120       NA's   :120        NA's   :120
```

And, here's a plot of the original data with the SNHT statistics computed above:

```
plotSNHT(data = baseData, stat = snhtStatistic30, alpha = .05)
```



The red dashed vertical line represents the maximum SNHT statistic computed on the data. If you are using the SNHT to homogenize data, you would shift the observations before this time by the difference in the two means at this point in time. That difference is available from the test statistic object:

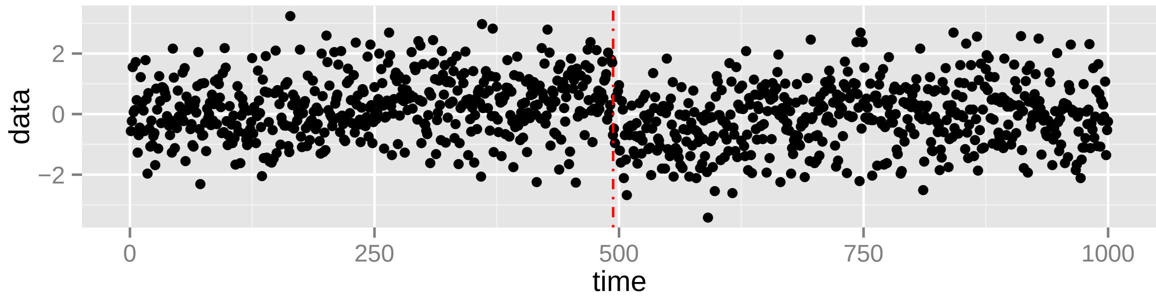
```
largestStatTime = which.max(snhtStatistic60$score)
snhtStatistic60[largestStatTime, ]

##        score  leftMean  rightMean
## 500 46.48912 0.5896035 -0.5740699
```

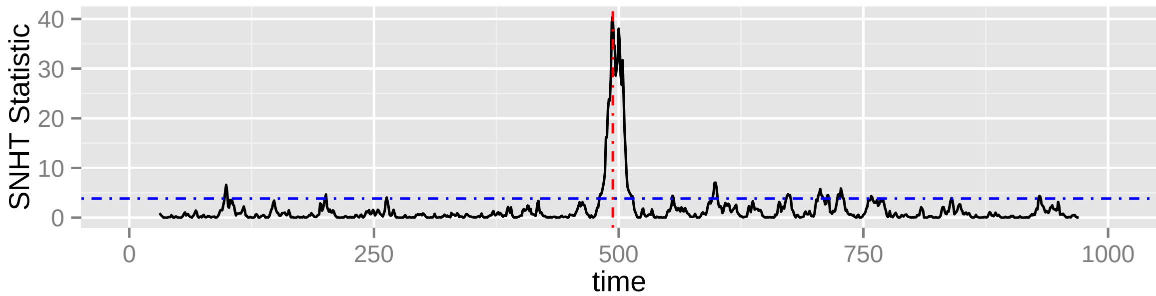
So, the observations to the left of 494 should be shifted by -0.574 - 0.59 = -1.164. Then, the SNHT statistic would be recomputed on this new dataset, and the process would repeat until no new statistic exceeds the threshold. Note that we managed to find the largest break exactly, and our correction is close to the simulated error.

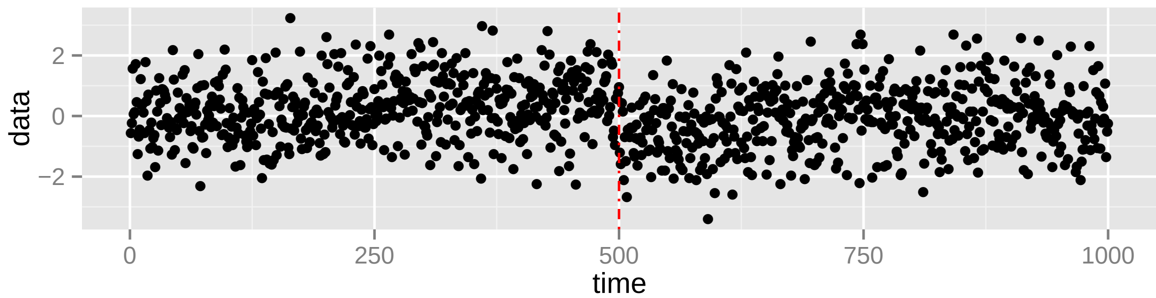
Under the null hypothesis of no changepoints and normal random errors, the test statistic approximately follows a chi-squared distribution with one degree of freedom (the approximation is that we divide by an estimate of the variance rather than the variance, and so actually have the square of a Student's-*t* distribution). However, caution should be used if applying that threshold to all computed statistics simultaneously, as you then have the problem of multiple testing (and non-independent tests). The blue dashed line in the plot above gives this threshold, but should be used with care.

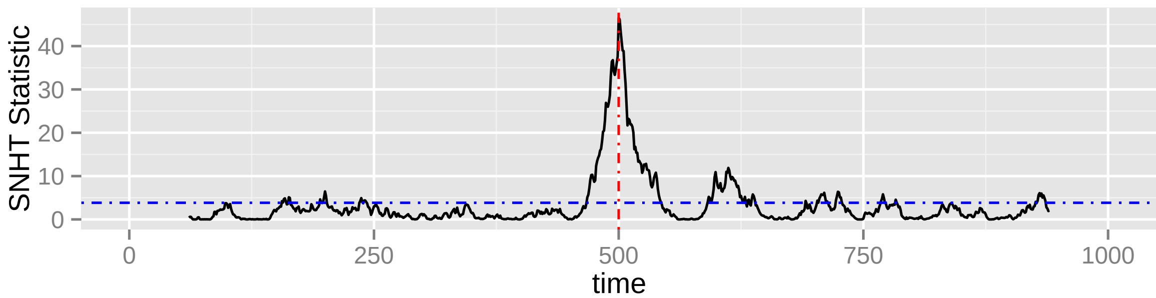
We may also wish to compare the performance of the test with a longer period:

```
plotSNHT(data = baseData, stat = snhtStatistic30, alpha = 0.05)
plotSNHT(data = baseData, stat = snhtStatistic60, alpha = 0.05)
```





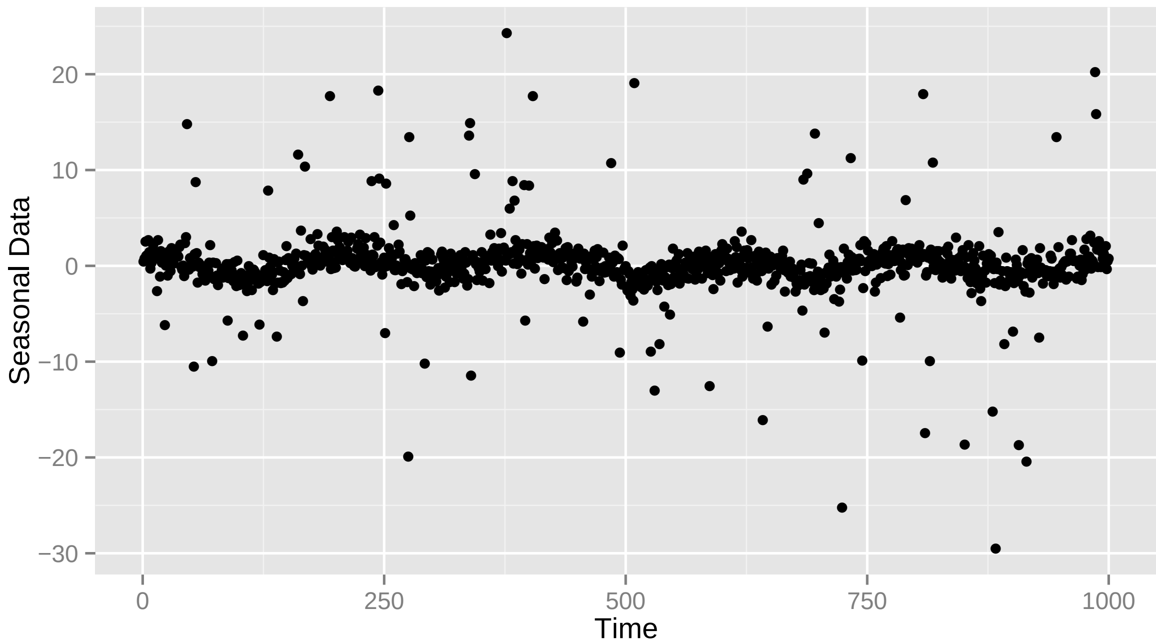




2.2. Example 2: Seasonal trends, outliers, equal spacing in time

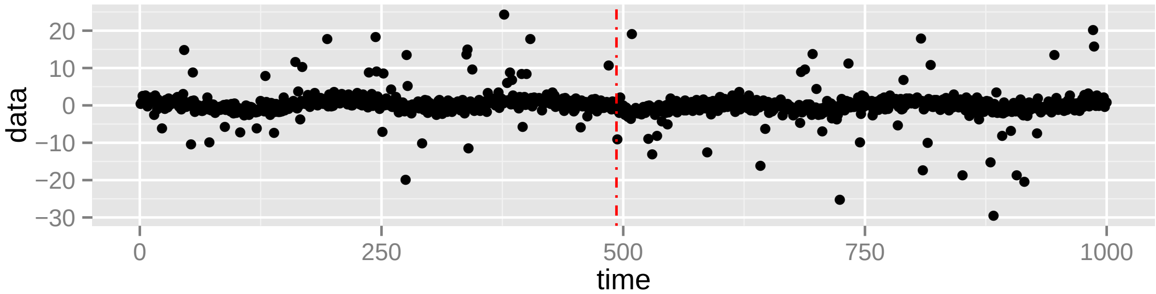
Now, let's suppose that this data has a seasonal trend to it, as well as some outliers:

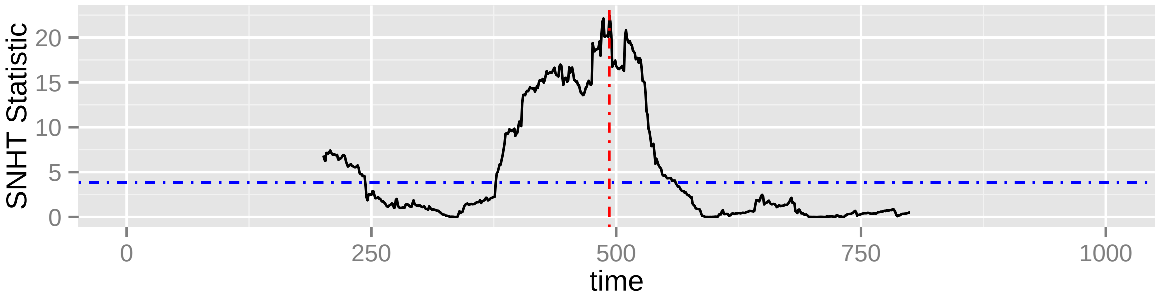
```
seasonalData = baseData + cos(1:200 * 2 * pi / 200)
seasonalData = seasonalData +
    rbinom(1000, p = .1, size = 1) * rnorm(1000, sd = 10)
qplot(1:1000, seasonalData) + labs(x = "Time", y = "Seasonal Data")
```



Now, the assumptions of the SNHT are invalid: we no longer have normal random errors about some mean. We should use a period of 200 now, as we wish to average out the seasonal effects.

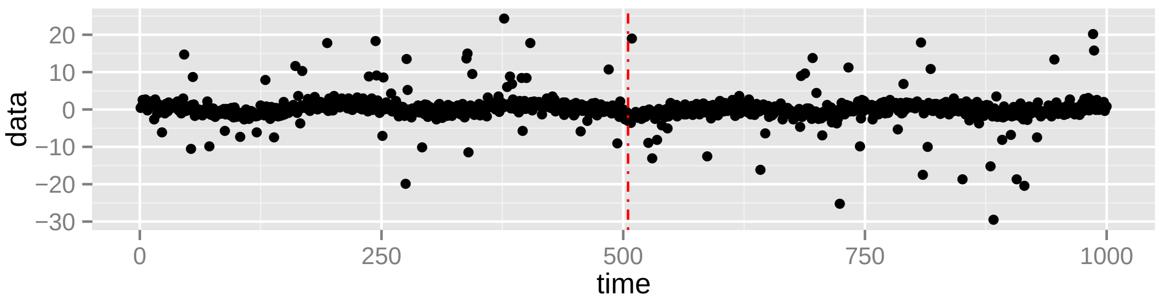
```
snhtStatistic = snht(data = seasonalData, period = 200)
plotSNHT(data = seasonalData, stat = snhtStatistic, alpha = 0.05)
```

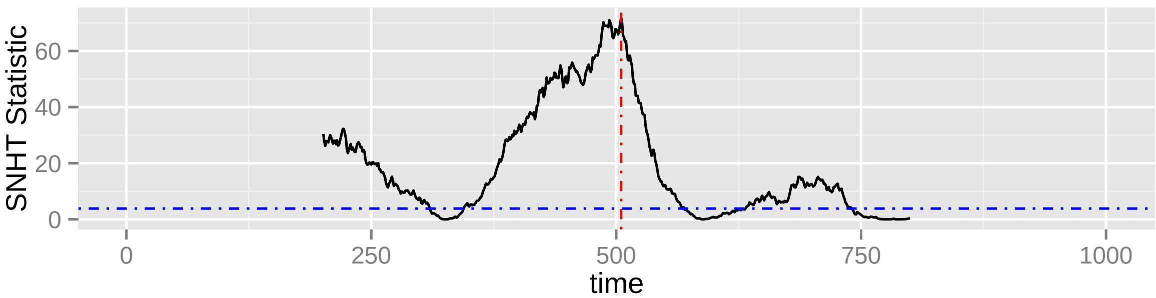




This statistic looks ok, but it completely fails to detect the third changepoint. If we instead use the robust statistic, we see a smoother graph:

```
snhtStatistic = snht(data = seasonalData, period = 200, robust = TRUE,
                     rmSeasonalPeriod = 200)
plotSNHT(data = seasonalData, stat = snhtStatistic, alpha = .05)
```





Now, each of the three changepoints exceeds the naive chi-squared threshold.

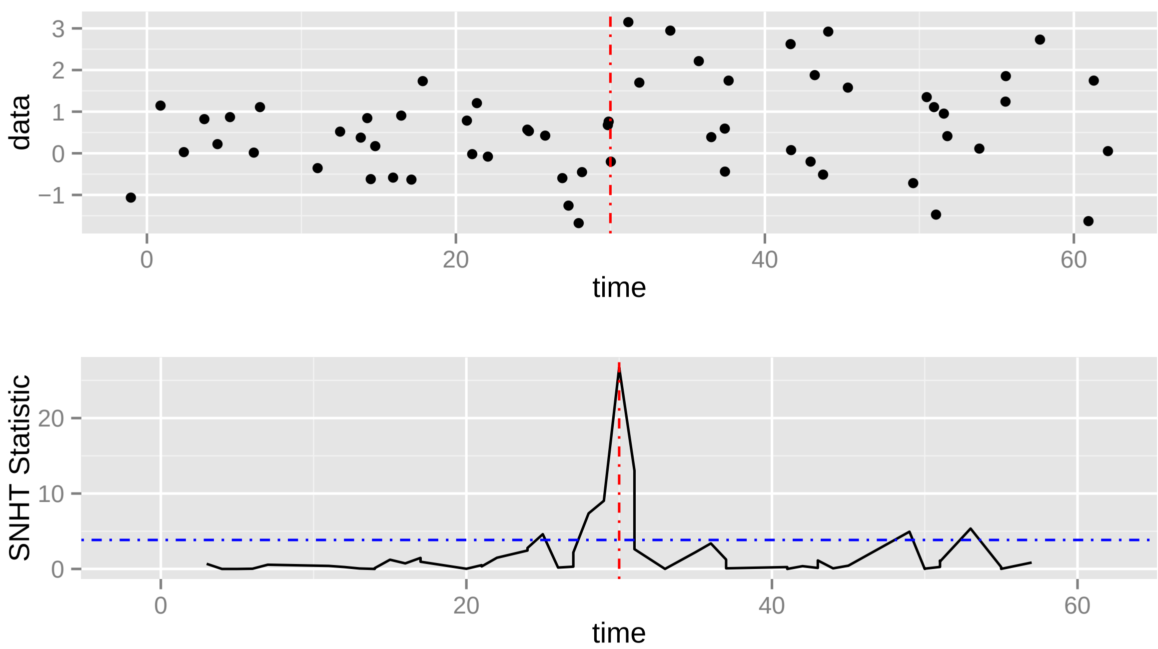
2.3. Example 3: No seasonal trends, no outliers, unequal spacing in time

Lastly, we may have observations that are unequally spaced in time. The snht algorithm will handle these appropriately as long as we specify what times observations occurred at:

```
times = 1:60 + rnorm(60, sd = 3)
times = sort(times)
data = rnorm(60) + c(rep(0, 30), rep(1, 30))
snhtStatistic = snht(data = data, period = 5, time = times)
summary(snhtStatistic)

##      score              leftMean         rightMean           time
##  Min.   : 0.000015   Min.   :-0.3922   Min.   :-0.3922   Min.   :-2.00
##  1st Qu.: 0.089284   1st Qu.: 0.1609   1st Qu.: 0.3252   1st Qu.:15.75
##  Median : 0.468109   Median : 0.4328   Median : 0.5672   Median :29.00
##  Mean   : 2.084822   Mean   : 0.5425   Mean   : 0.7338   Mean   :30.03
##  3rd Qu.: 2.006104   3rd Qu.: 0.7547   3rd Qu.: 1.0407   3rd Qu.:43.25
##  Max.   :26.705928   Max.   : 2.5045   Max.   : 2.5045   Max.   :62.00
##  NA's   :6           NA's   :6         NA's   :6
```

Now, note that an additional column has been added to the snhtStatistic data.frame: time. These times don't correspond exactly to the input times (as only one statistic is computed for each time step, for computational reasons). We can again plot this new statistic:

```
plotSNHT(data = data, stat = snhtStatistic, time = times, alpha = .05)
```



References

Browning J, Hering A (2015). "Simultaneous Treatment of Random and Systematic Errors in the Historical Radiosonde Temperature Archive." *(to be submitted)*.

Domonkos P (2013). "Measuring performances of homogenization methods." *Quarterly Journal of the Hungarian Meteorological Service*, **117**(1), 91–112.

Huber PJ (2011). *Robust Statistics*. Springer.

Affiliation:

Joshua M. Browning
Department of Applied Mathematics and Statistics, Colorado School of Mines
Golden, CO 80401, USA.
E-mail: jbrownin@mines.edu