pTITAN2

Permutation of Treatment Labels and Threshold Indicator Taxa ANalysis

Stephanie Figary1, Peter E. DeWitt2,3,4, and Naomi Detenbeck5

1(formerly) ORISE participant at U.S. Environmental Protection Agency, Atlantic Coastal Environmental Sciences Division, 27 Tarzwell Drive, Narragansett, RI, 02882, USA

2Section of Informatics and Data Science, Department of Pediatrics, University of Colorado School of Medicine, Aurora Colorado, USA

3National Renewable Energy Laboratory, Golden Colorado, USA

4(formerly) Neptune and Company, Inc. Lakewood Colorado, USA

5U.S. Environmental Protection Agency, Atlantic Coastal Environmental Sciences Division, 27 Tarzwell Drive, Narragansett, RI 02882, USA

**Abstract**

Taxa Indicator Threshold ANalysis (TITAN) was developed to identify thresholds along environmental gradients where rapid changes in taxa frequency and relative abundance are observed. TITAN determines separate change-points for increasing and decreasing taxa in aggregate, as well as change-points for individual taxa, with associated confidence intervals generated using bootstrapping. However, if TITAN is applied to different classes of observations, additional analyses besides using non-overlapping confidence intervals are needed to establish whether change-points differ between treatments or groups because non-overlapping confidence intervals can indicate significant differences but overlapping confidence intervals do not necessarily mean the null hypothesis cannot be rejected. To address this, we present a new R package, pTITAN2, which is an extension to the existing TITAN2 package. The pTITAN2 package was developed to enable comparisons of TITAN output between treatments by permutating the observed data between treatments and rerunning TITAN on the permuted data. The output from pTITAN2 can be used to perform the appropriate statistical tests and determine statistical differences without using overlapping confidence intervals.

**Keywords:** TITAN, permutations, thresholds, community composition

# 1 Introduction

Community ecologists are interested in understanding the structure and interactions of multiple species in a given area or habitat type and many are interested in understanding how communities change in response to changing environmental or anthropogenic gradients. One method a community ecologist can use for understanding to detect changes or ecological thresholds across environmental gradients is Taxa Indicator Threshold ANalysis (TITAN) (Matthew E. Baker and King 2010). TITAN is useful for determining the impacts of environmental or anthropogenic gradients such as upstream impervious cover (IC) in a watershed in community ecology studies because it both analyzes each individual taxa response and the community as a whole in the same analysis. Additionally, unlike other community ecology methods, TITAN separates the taxa that increase across an environmental gradient from those that decrease to provide a more complete picture of the community response to that gradient.

As an overview, TITAN methods use change point (King and Richardson 2003; Qian, King, and Richardson 2003) and indicator species analysis (Dufrêne and Legendre 1997) to determine the point along an environmental gradient, such as percent IC in the upstream watershed area, where individual taxa have the largest change in occurrence frequency and abundance, and then uses the individual taxa results to determine synchronous areas of taxa change at the community level (Matthew E. Baker and King 2010, 2013). Individual taxa change points are determined by calculating the taxa’s indicator (IndVal) score along the environmental gradient and assigning the taxa as either increasing (z+) or declining (z-) in response to an increase in the environmental gradient variable. Permutations of data in TITAN runs are used to calculate the likelihood of random data generating a larger IndVal score than the observed data and to standardize the taxa response to the environmental gradient by calculating the taxa z-scores using permuted distributions. Taxa z-scores are added together for increasing (z+) and declining (z-) taxa along the environmental gradient and the point with the highest sum(z-) and sum(z+) score is defined as the community change point. Next, bootstrapping of observed data is used to calculate percentiles around the community and individual taxa z-scores, and to determine if individual taxon responses are pure and reliable. Purity is a measure of the proportion of bootstrap results that match the taxa’s observed response group as either increasing or declining. Reliability is the proportion of bootstraps with a low probability (p < 0.01) of random data having a higher IndVal score than the bootstrapped observed data. Community change points are also calculated after selecting (filtering) only the taxa that exceeded purity and reliability requirements for the increasing (fsum(z+)) and declining (fsum(z-)) change points. Narrow peaks around the maximum sum(z) or fsum(z) (filtered) scores indicate areas with synchronous change in individual taxa frequency and abundance and may indicate an ecological threshold along the environmental gradient being evaluated. More information on TITAN methods can be found in Matthew E. Baker and King (2013) or in the TITAN R package, TITAN2, (Matthew E. Baker, King, and Kahle 2020).

While TITAN is a powerful tool for community ecologist it requires additional analysis for comparing results from different regions, groups or treatments. Previously, researchers have used non-overlapping confidence intervals for change-points from TITAN output to indicate significant differences between groups (King et al. 2011). 2011). However, although non-overlapping confidence intervals can indicate significant differences, overlapping confidence intervals do not necessarily mean the null hypothesis cannot be rejected (Greenland et al. 2016; Schenker and Gentleman 2001).

We developed a new R package, pTITAN2, as an extension of the existing TITAN2 R package (Matthew E. Baker, King, and Kahle 2020). The goal of pTITAN2 is to enable comparisons of TITAN output between treatments by permuting the observed data between treatments and rerunning TITAN on the permuted data. There are some limitations on the permutations, including (1) a sampling site cannot occur in a category more than once, the same limitation as in the original TITAN runs and (2) the original sample size distribution is maintained. This addresses potential sample size effects and enables comparisons between treatments with different sample sizes more accurately than using non-overlapping confidence intervals. A vignette is provided based on a dataset of macroinvertebrate data from California streams that fall along a gradient of watershed percent impervious cover. We compare change-points among different climate conditions (wet, average, and dry) based on the Palmer Drought Severity Index, which serve as the treatments in this example.

# 2 Methods

## 2.1 Operation:

Like TITAN2, pTITAN2 was developed using the R programming language (R Core Team 2021). pTITAN2 has been testing on Windows, Mac OS, and Ubuntu for the latest R version (4.1.0 at time of writing) along with old release and development versions via github actions. Users should be familiar with the TITAN2 package operations before using pTITAN2 (see Matthew E. Baker, King, and Kahle (2020)).

The basic workflow for pTITAN2 is

1. Prepare and import the environmental gradient dataset into R
2. Prepare and import the taxonomic dataset into R
3. Preprocess raw taxonomic data to determine appropriate taxonomic level of resloution (occurrence function)
4. Select columns for the taxon level retuned by occurrence function
5. Permut the data across treatment labels to generate list of lists
6. Set up cluster for parallele processing (optional)
7. Run TITAN2 series on original and permuted data sets
8. Analyze probability of exceeding observed diffrence in changepoint between treatments based on distribution of paired changepoint differences

## 2.2 Implementation:

The first step of pTITAN2 is to provide the data about the environmental gradient in exactly the format as for TITAN, step 1). This can be either a single file or include in teh taxonomic data file (Matthew E. Baker, King, and Kahle 2020). Like TITAN2, taxonomic information should be provided as counts or density. Unlike TITAN2, pTITAN2 taxonomic data nees to be provide as a code that is 8 characters in length and captures four levels of hierarchical taxonomic classification information.

The pTITAN2 package provides four example data sets, two taxonomic and two environmental gradient (Table 2.1). These data sets are provided as raw csv files and as prepared R datasets.

Table 2.1: Example data sets provided in pTITAN2.

| R Data | csv File | Data Type | Region | Treatment |
| --- | --- | --- | --- | --- |
| C\_IC\_D\_06\_wID | C\_IC\_D\_06\_wID.csv | Environmental Gradient | Chaparral | Dry |
| C\_IC\_N\_06\_wID | C\_IC\_N\_06\_wID.csv | Environmental Gradient | Chaparral | Normal |
| CD\_06\_Mall\_wID | CD\_06\_Mall\_wID.csv | Taxonomic | Chaparral | Dry |
| CN\_06\_Mall\_wID | CN\_06\_Mall\_wID.csv | Taxonomic | Chaparral | Normal |

You can gain access to the csv files via system.file

list.files(system.file("extdata", package = "pTITAN2"))  
## [1] "C\_IC\_D\_06\_wID.csv" "C\_IC\_N\_06\_wID.csv" "CD\_06\_Mall\_wID.csv"  
## [4] "CN\_06\_Mall\_wID.csv"

or get the data sets loaded into your environment via

data(C\_IC\_D\_06\_wID, C\_IC\_N\_06\_wID, CD\_06\_Mall\_wID, CN\_06\_Mall\_wID,  
 package = "pTITAN2")  
  
str(C\_IC\_D\_06\_wID) # Environemntal Gradient, Dry Treatment  
## 'data.frame': 251 obs. of 2 variables:  
## $ StationID: chr "308000" "707063" "707086" "500762" ...  
## $ ImpCover : num 0.0151 0.0907 0.076 3.9944 2.5167 ...  
str(C\_IC\_N\_06\_wID) # Environemntal Gradient, Normal Treatment  
## 'data.frame': 124 obs. of 2 variables:  
## $ StationID: chr "707059" "707076" "707219" "500004" ...  
## $ ImpCover : num 0.05959 0.00695 8.46983 35.8094 12.97293 ...  
dim(CD\_06\_Mall\_wID) # Taxonomic, Dry Treatment  
## [1] 251 501  
dim(CN\_06\_Mall\_wID) # Taxonomic, Normal Treatment  
## [1] 124 501

The CN\_06\_Mall.csv (Chaparral Region, Treatment = Normal) file contains raw macroinvertebrate density data for 500 possible macroinvertebrate codes for each taxonomic level (class, order, family, genus). The occurrences function selects the codes that should be used for the TITAN2::titan run. The goal is to select the macroinvertebrate code with the most taxonomic detail having at least n occurrences. Only one macroinvertebrate code will be associated with the macroinvertebrate counts. For example, if there are at least 5 occurrences at the genus level, the family, order, and class codes would not be used in the TITAN2::titan run.

The names within the data set are expected to have the following structure:

* 8 characters in length
* characters 1 and 2 denote the class
* characters 3 and 4 denote the order
* characters 5 and 6 denote the family
* characters 7 and 8 denote the genus.

If no information at a level exists, use “00” to hold the place. For example: A code that is ‘Bi000000’ is the Bivalvia class, while BiVe0000 is the Bivalvia class, Veneroida order. BiVeSh00 is the Bivalvia class, Veneroida order, Spheriridae family. BiVeSh01 is a genus within that family.

The first new function provided by pTITAN2 is occurrences. Taking the taxonomic data as an input, the return of occurrences is a data.frame with the taxon, the class, order, family, and genus split out into individual columns, and the count of occurrences within the provided taxonomic data set. TITAN2::titan recommends all taxonomic groups have at least five observations (Matthew E. Baker and King 2010). Thus occurances returns only taxons with at least n observations, defaulting to 5. The taxonomic code chosen for analysis should be at the finest possible resolution. For example, if a macroinvertebrate count has at least 5 occurrences in a genus code, the family, order, and class codes associated with these counts should be removed. Further, if there are too few counts at the genus level, but at least 5 counts at the family level- the family code would be retained and the order and class codes would be removed.

The second new function provided by pTITAN2 is the permute function which provides a list of permuted sets of taxa and environmental gradients. This function is used with categorical envrionmental variables (treatments), such as Wet/Dry or Urban/Rural. The funciton permutes the treatment labels across the data such that each station has a non-zero probability of being assigned to each treatment, and the stations are unique within each treatment and replication. There are some limitation on the permutations generated by permute. First, a site cannot occur in a category more than once within a permutation. Second, the original sample size distribution is maintained. These limitations address potential sample size effects in TITAN, where treatments with low sample sizes have wide confidence intervals and variable change points compared to treatments with high sample sizes, and enable comparisons between treatments with different sample sizes.

For example, assume we have sites A, B, C, D, and E with treatments 1 and 2 (Table 2.2). Let Trt0 denote the initial treatment labels for the sites and Trt1, …, Trt4 denote permuted treatment labels. For sites A and C, each permuted set of treatment labels consist of one row for label 1 and one row for label 2. For sites B, D, and E, the initial observations were for treatments 1, 2, and 2 respectively. The balance of these labels is maintained across the permutations.

Table 2.2: Example distribution of sites and perutated treatment labels.

| site | trt0 | trt1 | trt2 | trt3 | trt4 |
| --- | --- | --- | --- | --- | --- |
| A | 1 | 1 | 2 | 1 | 1 |
| A | 2 | 2 | 1 | 2 | 2 |
| B | 1 | 2 | 2 | 2 | 1 |
| C | 1 | 1 | 2 | 2 | 1 |
| C | 2 | 2 | 1 | 1 | 2 |
| D | 2 | 2 | 2 | 1 | 2 |
| E | 2 | 1 | 1 | 2 | 2 |

After permutations, clusters can be used for parallel processing of TITAN::titan() calls. This can be advantageous as TITAN::titan() calls can be time and computationally expensive. Following the needed TITAN::titan() calls the differences between treatment change points in the observed data can be compared to the differences between treatment change points in the permuted stat to determine if the observed treatment differences are statistically significant.

# 3 Example

Here we present an example showing implementation of pTITAN2. We will describe the provided example data sets and how to use the occurrences() and permute() functions.

To reproduce the examples in this vignette you will need to load and attach the pTITAN2 and magrittr namespaces. Other namespaces are used explicitly, loaded (not attached) here.

library(pTITAN2)  
library(magrittr)  
loadNamespace("data.table")  
## <environment: namespace:data.table>  
loadNamespace("dplyr")  
## <environment: namespace:dplyr>  
loadNamespace("tidyr")  
## <environment: namespace:tidyr>

## 3.1 Example data

Example data provided within the pTITAN2 package were based on publicly available stream macroinvertebrate data from California. The data include existing macroinvertegrate abundances from the California Environmental Data Exchange Network (CEDEN, last accessed 30 June 2017), and the Southern California Coastal Water Research Project (SCCWRP) (Fetscher et al. 2014). Samples in the CEDEN dataset were collected between 2000 and 2016, and samples from the SCCWRP dataset were collected between 1997 and 2011. Both data sets were generated using probabilistic sampling designs and are expected to be representative of streams in the region.

For this example, data were extracted for California’s Chapparal Region (Ode et al. 2011). Sample observations were divided into one of three classes based on the precipitation regime for the sampling year using the Palmer Drought Severity Index (PDSI, <https://www7.ncdc.noaa.gov/CDO/CDODivisionalSelect.jsp>). The Palmer Drought Severity Index was determined for each sampling event using monthly PDSI data from the National Oceanic and Atmospheric Administration (NOAA, last accessed 21 December 2016) and climate divisions from the National Climatic Data Center (USGS 2004, last accessed 21 December 2016). All sampling events were classified as dry (less than -2 PDSI), normal (between -2 and 2 PDSI), or wet ( greater than 2 PDSI) and these classifications were used as treatments for the permutations. The environmental gradient of interest was percent impervious cover in the upstream watershed, in this case defined by the National Land Cover Datasets (NLCD, Homer et al. (2007), homer2015completion), with values interpolated between NLCD years of record (2001, 2006, 2011). Impervious area additions beyond 2011 were estimated as 50% of disturbed area for construction sites as documented in the California Stormwater Multiple Application and Report Tracking System (SMARTS dataset, CalEPA) (smarts.waterboards.ca.gov, last accessed 31 July 2017).

For running pTITAN2, the example data sets have a separate csv or pre-built R data sets (2.1), for the environmental variable, in this case percent impervious cover, and macroinvertebrate density data. The data structure that is shown here is not required for pTITAN2 and instead the environmental variables and treatments could be in a single data file and subdivided as desired. Separate data files are provided for each ‘treatment’ that is explored including data from either drought (dry) or normal precipitation years in the Chaparral region of California.

## 3.2 Function occurrences

The taxonomic sets, CD\_06\_Mall\_wID and CN\_06\_Mall\_wID, contains raw macroinvertebrate density data for 500 possible macroinvertebrate codes for each taxonomic level (class, order, family, genus).

dim(CD\_06\_Mall\_wID)  
## [1] 251 501  
  
# top 4 rows and first 10 columns  
CD\_06\_Mall\_wID[1:4, 1:10]  
## StationID Ar000000 Bi000000 BiUn0000 BiUnUi00 BiUnUi01 BiVe0000 BiVeCa00  
## 1 308000 48.484848 0 0 0 0 0 0  
## 2 707063 97.979798 0 0 0 0 0 0  
## 3 707086 1.010101 0 0 0 0 0 0  
## 4 500762 8.080808 0 0 0 0 0 0  
## BiVeCa01 BiVeSh00  
## 1 0 0  
## 2 0 0  
## 3 0 0  
## 4 0 0

The occurrences function selects the codes that should be used for the TITAN2::titan run. The goal is to select the macroinvertebrate code with the most taxonomic detail having at least n occurrences. Only one macroinvertebrate code will be associated with the macroinvertebrate counts. For example, if there are at least 5 occurrences at the genus level, the family, order, and class codes would not be used in the TITAN2::titan run.

The data is parsed within the occurrences call and returns a data.frame with each taxon code split into its components and the frequency of the taxon within the data set. A new function in pTITAN2 is the occurrences function. This is an extension for deciding the taxonomic detail to be included in a TITAN run based on minSplt in TITAN. minSplt is minimum number of occurrences that TITAN is looking for taxa to have across the provided sites. The minSplt default in TITAN is 5 and should never drop below 3. The default for occurrences is minSplt = n = 5.

head(occurrences(CN\_06\_Mall\_wID[, -1]))  
## taxon Class Order Family Genus count  
## 1 Ar000000 Ar 00 00 00 115  
## 2 BiVeCa01 Bi Ve Ca 01 16  
## 3 BiVeSh00 Bi Ve Sh 00 42  
## 4 EnHoTt01 En Ho Tt 01 19  
## 5 GaBaAc01 Ga Ba Ac 01 14  
## 6 GaBaLy02 Ga Ba Ly 02 5

Compare these results to working with the raw data. For example purposes we present the summary of the raw data twice, once using tidyverse syntax and once using data.table syntax.

# Tidyverse  
CN\_06\_Mall\_wID %>%  
 dplyr::select(-StationID) %>%  
 tidyr::pivot\_longer(cols = dplyr::everything(), names\_to = 'taxon', values\_to = 'count') %>%  
 dplyr::mutate(  
 Class = substr(.data$taxon, 1L, 2L),  
 Order = substr(.data$taxon, 3L, 4L),  
 Family = substr(.data$taxon, 5L, 6L),  
 Genus = substr(.data$taxon, 7L, 8L)  
 ) %>%  
 dplyr::group\_by(.data$Class, .data$Order, .data$Family, .data$Genus) %>%  
 dplyr::summarise(  
 taxon = unique(.data$taxon),  
 count = sum(.data$count > 0),  
 .groups = "keep"  
 ) %>%  
 dplyr::ungroup() %>%  
 dplyr::arrange(.data$Class, .data$Order, .data$Family, .data$Genus)  
## # A tibble: 500 × 6  
## Class Order Family Genus taxon count  
## <chr> <chr> <chr> <chr> <chr> <int>  
## 1 Ar 00 00 00 Ar000000 115  
## 2 Bi 00 00 00 Bi000000 46  
## 3 Bi Un 00 00 BiUn0000 0  
## 4 Bi Un Ui 00 BiUnUi00 0  
## 5 Bi Un Ui 01 BiUnUi01 0  
## 6 Bi Ve 00 00 BiVe0000 46  
## 7 Bi Ve Ca 00 BiVeCa00 16  
## 8 Bi Ve Ca 01 BiVeCa01 16  
## 9 Bi Ve Sh 00 BiVeSh00 42  
## 10 Br 00 00 00 Br000000 0  
## # … with 490 more rows

# data.table  
taxon\_count <- data.table::copy(CN\_06\_Mall\_wID)  
data.table::setDT(taxon\_count)  
data.table::set(taxon\_count, j = "StationID", value = NULL)  
for(j in as.integer(which(sapply(taxon\_count, is.integer)))) {  
 data.table::set(taxon\_count, j = j, value = as.numeric(taxon\_count[[j]]))  
}  
  
taxon\_count <- data.table::melt(taxon\_count, variable.factor = FALSE, measure.vars = names(taxon\_count), variable.name = "taxon")  
taxon\_count[, value := sum(value > 0), keyby = .(taxon)]  
taxon\_count <- unique(taxon\_count)  
  
data.table::set(taxon\_count, j = "Class", value = substr(taxon\_count[["taxon"]], 1L, 2L))  
data.table::set(taxon\_count, j = "Order", value = substr(taxon\_count[["taxon"]], 3L, 4L))  
data.table::set(taxon\_count, j = "Family", value = substr(taxon\_count[["taxon"]], 5L, 6L))  
data.table::set(taxon\_count, j = "Genus", value = substr(taxon\_count[["taxon"]], 7L, 8L))  
  
data.table::setkeyv(taxon\_count, c("taxon", "Class", "Order", "Family", "Genus"))  
  
taxon\_count[grepl("^(Ar|Bi).+", taxon)]  
## taxon value Class Order Family Genus  
## 1: Ar000000 115 Ar 00 00 00  
## 2: Bi000000 46 Bi 00 00 00  
## 3: BiUn0000 0 Bi Un 00 00  
## 4: BiUnUi00 0 Bi Un Ui 00  
## 5: BiUnUi01 0 Bi Un Ui 01  
## 6: BiVe0000 46 Bi Ve 00 00  
## 7: BiVeCa00 16 Bi Ve Ca 00  
## 8: BiVeCa01 16 Bi Ve Ca 01  
## 9: BiVeSh00 42 Bi Ve Sh 00

Note that for the Ar class there is only one row with no order, family, or genus level information. Compare to the Bi class where the Un order has no presence counts and is thus not reported in object returned from occurrences. BiVeCa01 has counts and will be reported but BiVeCa00 should not be reported. BiVe0000 and Bi000000 should not be reported as occurrences as preference for the codes with family and genus level information.

## 3.3 Function permute

The function permute is used to generate a list of permuted sets of taxa and environmental gradients. Function parameters include a list of data frames containing taxa for each treatment group, a list of data frames containing the associated environmental gradient variables, and the site ids. Before we can run permute, we need to import the environmental gradients data.

eg <-  
 permute(taxa = list(CD\_06\_Mall\_wID, CN\_06\_Mall\_wID),  
 envs = list(C\_IC\_D\_06\_wID, C\_IC\_N\_06\_wID),  
 sid = "StationID")  
str(eg, max.level = 2)  
## List of 2  
## $ Treatment1:List of 2  
## ..$ env :'data.frame': 251 obs. of 1 variable:  
## ..$ taxa:'data.frame': 251 obs. of 500 variables:  
## $ Treatment2:List of 2  
## ..$ env :'data.frame': 124 obs. of 1 variable:  
## ..$ taxa:'data.frame': 124 obs. of 500 variables:  
## - attr(\*, "minTaxonFreq")= num [1:2] 0 0

The return of permute is list of lists. The first level denotes the treatment, in this example Treatment1 is “dry” and Treatment2 is “normal” – the order of the input data sets. The second level contains the data.frames with environmental and taxonomic data.

## 3.4 Running TITAN2::titan

The most computationally expensive part of this work is calling TITAN2::titan many, many times. A good option is to use the parallel package to send the task of permuting the data and running TITAN2::titan() to individual processing cores. That is system dependent and left to the end user to implement. For an example of generating the permutations with TITAN2::titan() see the example script provided at:

system.file("example-scripts/permutation\_example.R", package = "pTITAN2")  
## [1] "/Users/pdewitt/Library/R/x86\_64/4.1/library/pTITAN2/example-scripts/permutation\_example.R"

That file will generate the provided data set permutation\_example with 10 rows from 10 permutations of the example data set.

permutation\_example  
## trt1cpsumz- trt1cpsumz+ trt1count trt2cpsumz- trt2cpsumz+ trt2count  
## 1 0.24450828 0.9322147 251 0.21052157 7.8379526 124  
## 2 0.20733907 0.9322147 251 0.20307915 0.6447989 124  
## 3 0.17739530 0.6017841 251 0.20532745 1.9463146 124  
## 4 0.43918849 0.9195990 251 0.20949813 4.2981178 124  
## 5 0.20733907 1.0749101 251 0.17238643 0.4298088 124  
## 6 0.20941096 0.6012265 251 0.20675722 1.1443447 124  
## 7 0.06755573 0.6014938 251 0.29043078 2.1666044 124  
## 8 0.14227803 0.6222103 251 0.26105344 0.3336509 124  
## 9 0.22585229 0.6017841 251 0.07078791 10.5727762 124  
## 10 0.24486085 0.6017841 251 0.21857924 2.0596616 124  
## permutation  
## 1 1  
## 2 2  
## 3 3  
## 4 4  
## 5 5  
## 6 6  
## 7 7  
## 8 8  
## 9 9  
## 10 10

The results are the increasing and decreasing taxa sumz values. In this example only four permutations are used and TITAN bootstrapping is limited to five iterations. In an actual analysis these values should be much higher. This process is very computationally intensive and can take hours or days to run depending on the available computing power and the number of bootstraps and permutations used.

If you have three or more treatments and need to permute over them with the condition that no station will be in the same treatment more than once on any particular permutation and that all treatment labels are viable for each station then you can still use the permute function.

## 3.5 Analyzing the results

The output from the above code chunk can then be used to compare the differences in change-point values for treatments from the observed samples versus the permuted samples. A p-value test can be run on these data to test for statistically significant differences between the treatment effects.

# Run TITAN on the observed data  
# limited to 2 cores for CRAN policies, end users can use more cores  
# the following uses the tidyverse dialect, load and attach the magrittr  
# namespace, use dplyr and tidyr namespaces explicitly.  
library(magrittr)  
  
CD\_obs <-  
 TITAN2::titan(  
 env = C\_IC\_D\_06\_wID[["ImpCover"]] # chaparral envgrad dry  
 , txa = subset(CD\_06\_Mall\_wID, select = occurrences(CD\_06\_Mall\_wID[, -1], n = 6L)$taxon)  
 , ncpus = 2  
 , numPerm = 50  
 , nBoot = 50  
 )  
## Screening taxa...  
## taxa frequency screen complete.  
## Partitioning along gradient...  
## Calculating observed IndVal maxima and class values...  
## Calculating IndVals using mean relative abundance...  
## Permuting IndVal scores...  
## Summarizing observed results...  
## Estimating taxa change points using z-score maxima...  
## Bootstrap resampling in parallel using 2 CPUs... no index will be printed to screen.  
## Proportion of pure and reliable taxa = 0.5732  
  
CN\_obs <-  
 TITAN2::titan(  
 env = C\_IC\_N\_06\_wID[["ImpCover"]] # chaparral envgrad dry  
 , txa = subset(CN\_06\_Mall\_wID, select = occurrences(CN\_06\_Mall\_wID[, -1], n = 6L)$taxon)  
 , ncpus = 2  
 , numPerm = 50  
 , nBoot = 50  
 )  
## Screening taxa...  
## taxa frequency screen complete.  
## Partitioning along gradient...  
## Calculating observed IndVal maxima and class values...  
## Calculating IndVals using mean relative abundance...  
## Permuting IndVal scores...  
## Summarizing observed results...  
## Estimating taxa change points using z-score maxima...  
## Bootstrap resampling in parallel using 2 CPUs... no index will be printed to screen.  
## Proportion of pure and reliable taxa = 0.1129

# create a table of the median change point from TITAN for calculating the p-values  
TITAN\_med <-  
 rbind(  
 cbind(as.data.frame(CD\_obs$sumz), "run" = "Tr1\_CD")  
 , cbind(as.data.frame(CN\_obs$sumz), "run" = "Tr2\_CN")  
 )  
TITAN\_med[["sumz"]] <- sub("(.+)(\\d)$", "\\1", rownames(TITAN\_med))  
  
  
TITAN\_med <-  
 TITAN\_med %>%  
 dplyr::select(run, sumz, `0.50`) %>%  
 tidyr::pivot\_wider(names\_from = run, values\_from = "0.50") %>%  
 dplyr::mutate(T1T2\_abs = abs(Tr1\_CD - Tr2\_CN)) %>%  
 print  
## # A tibble: 4 × 4  
## sumz Tr1\_CD Tr2\_CN T1T2\_abs  
## <chr> <dbl> <dbl> <dbl>  
## 1 sumz- 0.259 0.0851 0.174  
## 2 sumz+ 0.782 3.74 2.96   
## 3 fsumz- 0.280 0.0605 0.219  
## 4 fsumz+ 0.602 2.59 1.98

# Create a summary table of the permutation data  
tr1\_filt <-  
 permutation\_example %>%  
 dplyr::select(permutation, `trt1cpsumz-`, `trt1cpsumz+`) %>%  
 tidyr::pivot\_longer(cols = `trt1cpsumz-`:`trt1cpsumz+`,  
 values\_to = "Tr1\_CD",  
 names\_to = "sumz")  
  
tr1\_filt[which(tr1\_filt$sumz == "trt1cpsumz-"), 2] <- "fsumz-"  
tr1\_filt[which(tr1\_filt$sumz == "trt1cpsumz+"), 2] <- "fsumz+"  
  
tr2\_filt <-  
 permutation\_example %>%  
 dplyr::select(permutation, `trt2cpsumz-`, `trt2cpsumz+`) %>%  
 tidyr::pivot\_longer(cols = `trt2cpsumz-`:`trt2cpsumz+`,  
 values\_to = "Tr2\_CN",  
 names\_to = "sumz")  
tr2\_filt[which(tr2\_filt$sumz == "trt2cpsumz-"), 2] <- "fsumz-"  
tr2\_filt[which(tr2\_filt$sumz == "trt2cpsumz+"), 2] <- "fsumz+"  
  
  
# mutate the data and create a summary table for calculating p-values  
out\_perm <-  
 dplyr::left\_join(tr1\_filt, tr2\_filt) %>%  
 dplyr::mutate(T1T2\_abs = abs(Tr1\_CD - Tr2\_CN)) %>%  
 dplyr::select(-permutation)  
## Joining, by = c("permutation", "sumz")  
  
# Calulate the p-values  
dplyr::tibble(treatment = "T1T2\_CDCN",  
 "fsumz-" = (sum((  
 dplyr::filter(out\_perm, sumz == "fsumz-")$T1T2\_abs >  
 (dplyr::filter(TITAN\_med, sumz == "fsumz-")$T1T2\_abs)  
 )) + 1) / 1001, "fsumz+" =  
 (sum((  
 dplyr::filter(out\_perm, sumz == "fsumz+")$T1T2\_abs >  
 (dplyr::filter(TITAN\_med, sumz == "fsumz+")$T1T2\_abs)  
 )) + 1) / 1001)  
## # A tibble: 1 × 3  
## treatment `fsumz-` `fsumz+`  
## <chr> <dbl> <dbl>  
## 1 T1T2\_CDCN 0.00300 0.00400

# 4 Conclusions

TITAN is used in ecological studies to determine individual taxa and community level change points across an environmental gradient for both taxa that increase with the increasing environmental gradient and taxa that decrease with the increasing environmental gradient (Matthew E. Baker and King 2010). pTITAN2 was developed as an extension of TITAN to enable comparing TITAN results between different treatments, including those with variable sample sizes, by permuting the observed data between the treatments and then rerunning TITAN on the permuted dataset. This allows for statistically determining difference between the treatments without using overlapping confidence intervals, which can be problematic and can lead to accepting the null hypothesis more frequently than statistically necessary.

# 5 Data

Data associated with figures in this manuscript will be available via a query available at the url: <https://edg.epa.gov/metadata/catalog/main/home.page> within a few months of publication.

# 6 Software availability

The pTITAN2 R package is available from the Comprehensive R Archive Network (CRAN) at <https://cran.r-project.org/package=pTITAN2>. Source code and example datasets are available at <https://github.com/sfigary/pTITAN2>. This is an open-access article, free of all copyright, and may be freely reproduced, distributed, transmitted, modified, built upon, or otherwise used by anyone for any lawful purpose. The work is made available under the Creative Commons CC0 public domain dedication.

# 7 Author Contributions

* Stephanie Figary: Data curation, Methodology, Software, Formal Analysis, Writing
* Naomi Detenbeck: Conceptualization, Funding Acquisition, Writing, Supervision
* Peter DeWitt: Software development, Writing

# 8 Competing Interests

No competing interests were disclosed

# 9 Grant Information

This work was partially supported by the U.S. Environmental Protection Agency via an interagency agreement with the Department of Energy (DW92429801-9) which provided funding to Stephanie Figary through the ORISE program and through funding on the EPA contract EP-C-13-022 with Neptune, supporting software development.

# 10 Acknowledgments

This is contribution number ORD-041368 of the Atlantic Coastal Environmental Sciences Division, Center for Environmental Measurement and Modeling, Office of Research and Development, U.S. Environmental Protection Agency. Mention of trade names or commercial products does not constitute endorsement or recommendation for use.

# 11 References

Baker, Matthew E., Ryan S. King, and David Kahle. 2020. *Titan2: Threshold Indicator Taxa Analysis*. <https://CRAN.R-project.org/package=TITAN2>.

Baker, Matthew E, and Ryan S King. 2010. “A New Method for Detecting and Interpreting Biodiversity and Ecological Community Thresholds.” *Methods in Ecology and Evolution* 1 (1): 25–37.

———. 2013. “Of TITAN and Straw Men: An Appeal for Greater Understanding of Community Data.” *Freshwater Science* 32 (2): 489–506.

Dufrêne, Marc, and Pierre Legendre. 1997. “Species Assemblages and Indicator Species: The Need for a Flexible Asymmetrical Approach.” *Ecological Monographs* 67 (3): 345–66.

Fetscher, A Elizabeth, Martha Sutula, Ashmita Sengupta, and Naomi Detenbeck. 2014. *Linking Nutrients to Alterations in Aquatic Life in California Wadeable Streams*. US Environmental Protection Agency, Office of Research; Development ….

Greenland, Sander, Stephen J Senn, Kenneth J Rothman, John B Carlin, Charles Poole, Steven N Goodman, and Douglas G Altman. 2016. “Statistical Tests, p Values, Confidence Intervals, and Power: A Guide to Misinterpretations.” *European Journal of Epidemiology* 31 (4): 337–50. https://doi.org/<https://doi.org/10.1007/s10654-016-0149-3>.

Homer, Collin, Jon Dewitz, Joyce Fry, Michael Coan, Nazmul Hossain, Charles Larson, Nate Herold, et al. 2007. “Completion of the 2001 National Land Cover Database for the Counterminous United States.” *Photogrammetric Engineering and Remote Sensing* 73 (4): 337.

King, Ryan S, Matthew E Baker, Paul F Kazyak, and Donald E Weller. 2011. “How Novel Is Too Novel? Stream Community Thresholds at Exceptionally Low Levels of Catchment Urbanization.” *Ecological Applications* 21 (5): 1659–78.

King, Ryan S, and Curtis J Richardson. 2003. “Integrating Bioassessment and Ecological Risk Assessment: An Approach to Developing Numerical Water-Quality Criteria.” *Environmental Management* 31 (6): 795–809.

Ode, PR, TM Kincaid, T Fleming, and AC Rehn. 2011. “Ecological Condition Assessments of California’s Perennial Wadeable Streams, Highlights from the Surface Water Ambient Monitoring Program’s Perennial Streams Assessment (PSA)(2000–2007).” *A Collaboration Between the State Water Resources Control Board’s Non-Point Source Pollution Control Program (NPS Program), Surface Water Ambient Monitoring Program (SWAMP), California Department of Fish and Game Aquatic Bioassessment Laboratory, and the US Environmental Protection Agency*.

Qian, Song S, Ryan S King, and Curtis J Richardson. 2003. “Two Statistical Methods for the Detection of Environmental Thresholds.” *Ecological Modelling* 166 (1-2): 87–97.

R Core Team. 2021. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. <https://www.R-project.org/>.

Schenker, Nathaniel, and Jane F Gentleman. 2001. “On Judging the Significance of Differences by Examining the Overlap Between Confidence Intervals.” *The American Statistician* 55 (3): 182–86.