TEXTE
107/2025

Evaluation and implementation of statistical methods to assess effects in count data

by:

Christian Dietrich, Magnus Wang
WSC Scientific GmbH, Heidelberg

publisher:
German Environment Agency

German Environment Agency

Umwelt Bundesamt

TEXTE 107/2025

Project No. 173522
FB001755/ENG

Evaluation and implementation of statistical methods to assess effects in count data

by

Christian Dietrich, Magnus Wang
WSC Scientific GmbH, Heidelberg

On behalf of the German Environment Agency

Imprint

Publisher
Umweltbundesamt
Wörlitzer Platz 1
06844 Dessau-Roßlau
Tel: +49 340-2103-0
Fax: +49 340-2103-2285
buergerservice@uba.de
Internet: www.umweltbundesamt.de

Report performed by:
WSC Scientific GmbH
Dossenheimer Landstr. 9/1
69121 Heidelberg
Germany

Report completed in:
October 2024

Edited by:
Section IV 2; IV 1.1.; IV 2.4., IV 1.3.1
Benjamin Daniels, Susanne Walter-Rohde, Thomas Gräff, Pia Kotschik (Fachbegleitung)

DOI:
https://doi.org/10.60810/openumwelt-7762

ISSN 1862-4804

Dessau-Roßlau, September 2025

Abstract: Evaluation and implementation of statistical methods to assess effects in count data

The Federal Environment Agency (UBA) is currently reviewing the OECD Test Guideline No. 54, which outlines the statistical methods used in the analysis of ecotoxicological data. As part of this evaluation, the UBA identified a need to assess alternative statistical methods not covered in the guideline, particularly for Poisson-distributed data, common in mesocosm and field studies (e.g. non-target arthropod species). Additionally, the issue of multiple testing, which can lead to a reduction in test power, was identified as a concern.

The main objective of this project was to develop an R package to enhance the analysis of Poisson-distributed data using methods such as 'CPCAT'. Initially, errors in the existing CPCAT R script were corrected, including removing one-sided test options and addressing issues with p-values for control treatments with zero values. The package was further expanded implementing a similar test for binomial data (CPFISH) and a GLM-based Dunnett test (Dunnett.GLM) applicable for overdispersed count data.

Power calculations for CPCAT, CPFISH, and Dunnett.GLM were performed using simulated effect sizes to reflect typical reproduction and field test designs. Results showed that CPCAT and CPFISH exhibit sensitivity to the number of treatment groups without effects, potentially influencing p-values of affected groups. Therefore, caution is advised when using these methods in regulatory contexts. The project concluded with the successful development of a publicly available R package on CRAN and GitHub, providing statistical tools for the analysis of count data, which will support future revisions of OECD Test Guideline No. 54.

Kurzbeschreibung: Bewertung und Implementierung statistischer Methoden zur Beurteilung von Effekten in Zähldaten

Das Umweltbundesamt (UBA) überarbeitet derzeit die OECD-Prüfrichtlinie Nr. 54, in der statistische Methoden für die Analyse ökotoxikologischer Daten beschrieben sind. Im Rahmen dieser Bewertung stellte das UBA fest, dass alternative statistische Methoden, die nicht in der Richtlinie enthalten sind, bewertet werden müssen, insbesondere Methoden für Poisson-verteilte Daten, die in Mesokosmen- und Feldstudien häufig vorkommen (z. B. bei non-target Arthropodenarten). Darüber hinaus wurde die Frage der gängigen Mehrfachtests, die zu einer Verringerung der Teststärke führen können, als Problem erkannt.

Das Hauptziel dieses Projekts war die Entwicklung eines R-Pakets zur Verbesserung der Analyse von Poisson-verteilten Daten mit Methoden wie ‚CPCAT'. Zunächst wurden Fehler im bestehenden CPCAT-R-Skript korrigiert, einschließlich der Entfernung einseitiger Testoptionen und der Lösung von Problemen mit p-Werten für Kontrollgruppen mit Nullwerten. Das Paket wurde durch die Implementierung eines ähnlichen Tests für Binomialdaten (CPFISH) und eines GLM-basierten Dunnett-Tests (Dunnett.GLM) erweitert, der auch für überdispergierte Zähldaten (Varianz > Mittelwert) geeignet ist.

Power-Berechnungen für CPCAT, CPFISH und Dunnett.GLM wurden unter Verwendung simulierter Effektgrößen durchgeführt, um typische Reproduktions- und Feldtestdesigns widerzuspiegeln. Die Ergebnisse haben gezeigt, dass CPCAT und CPFISH sensitiv auf die Anzahl der Treatment-Gruppen ohne Effekte reagieren, was die p-Werte der betroffenen Gruppen beeinflussen kann. Daher ist bei der Anwendung dieser Methoden im regulatorischen Kontext Vorsicht geboten. Das Projekt wurde mit der erfolgreichen Entwicklung eines öffentlich zugänglichen R-Pakets auf CRAN und GitHub abgeschlossen, das statistische Tools für die Analyse von Zähldaten bereitstellt und künftige Überarbeitungen der OECD-Prüfrichtlinie Nr. 54 unterstützen wird.

Table of content

List of figures

List of tables

List of abbreviations

Abbreviation	Explanation
bMDD	Bootstrap Minimum Detectable Difference
CP	Closure Principle
CPCAT	Closure Principle Computational Approach Test
CPFISH	Closure Principle Fisher test
CRAN	Comprehensive R Archive Network
GLM	Generalized Linear Model
LOEC	Lowest Observed Effect Concentration
MDD	Minimum Detectable Difference
NOEC	No Observed Effect Concentration
TG	Test Guideline
UBA	Umweltbundesamt

Summary

The Federal Environment Agency (UBA) is currently reviewing the recommendations of the OECD Test Guideline No. 54 according to the current state of the knowledge. In the context of this project, it was considered necessary to evaluate alternative methods – not mentioned in this guideline – for the statistical analysis of Poisson-distributed data, which are typical endpoints of mesocosm studies or field studies (e.g. non-target arthropod species). Another issue was considered to be multiple testing (e.g. for several dose groups), for which correction methods are used, which can result in the reduction of alpha. The aim of the present project was to develop an R package, containing functions to statistically analyse Poisson-distributed data, namely the 'CPCAT' method. Initially, errors in the existing R script were corrected, including the removal of one-sided test procedures and issues with p-values for control treatments with all-zero values. The package was then expanded to include additional features, such as a measure for overdispersed and underdispersed data, the integration of the 'CPFISH' test, which is a similar test as CPCAT, but is used for binomial data, and the development of a 'Dunnett.GLM' test function. The 'Dunnett.GLM' test is a test based on a Dunnett test for the model parameters of a GLM (Generalized Linear Model) which can model overdispersed count data. Furthermore, power calculations for the three implemented methods were performed based on simulated effect sizes for typical reproduction and field test designs. It was found that the Closure Principle, specifically CPCAT and CPFISH, seem to have a high sensitivity to the number of treatment groups without effects, i.e. that p-values of treatment groups with effects might strongly be affected by adding treatment groups without effects. It is therefore recommended to use CPCAT and CPFISH for the evaluation of field studies with more than one treatment group without any effect with caution as effects might be overseen. The final R package has been made publicly available on CRAN and GitHub.

Zusammenfassung

Das Umweltbundesamt (UBA) überprüft derzeit die Inhalte der OECD-Prüfrichtlinie Nr. 54 im Hinblick auf den aktuellen Stand von Wissenschaft und Technik. Im Rahmen dieses Projektes wurde es als notwendig erachtet, alternative Methoden für die statistische Analyse von Poisson-verteilten Daten zu evaluieren. Ziel des vorliegenden Projekts war die Entwicklung eines R-Pakets, das Funktionen zur statistischen Analyse von Poisson-verteilten Daten enthält, namentlich die Methode ‚CPCAT'. Zunächst wurden Fehler im bestehenden R-Skript korrigiert, einschließlich der Entfernung von einseitigen Testverfahren und Problemen mit p-Werten für Kontrollbehandlungen mit Nullwerten. Das Paket wurde um zusätzliche Funktionen erweitert, wie z. B. ein Maß für eine Über- und Unterdispersion der Daten, die Integration des ‚CPFISH'-Tests (ähnlich CPCAT, aber für binomial-verteilte Daten) und die Entwicklung einer ‚Dunnett.GLM'-Test-Funktion. Der 'Dunnett.GLM'-Test ist ein Test auf der Grundlage eines Dunnett Tests für die Modellparameter eines GLM (Generalized Linear Model), das auch Zähldaten mit Überdispersion modellieren kann. Darüber hinaus wurden Power-Berechnungen für die drei Methoden auf der Grundlage simulierter Effektgrößen für typische Reproduktions- und Feldtestdesigns durchgeführt. Es wurde festgestellt, dass das Closure Principle, insbesondere CPCAT und CPFISH, eine hohe Sensitivität gegenüber der Anzahl der Treatment-Gruppen ohne Effekte zu haben scheint. Die p-Werte von Behandlungsgruppen mit Effekten durch das Hinzufügen von Behandlungsgruppen ohne Effekte stark beeinflusst werden können. Es wird daher empfohlen, CPCAT und CPFISH unter Vorliegen von mehreren Behandlungsgruppen ohne Effekte mit Vorsicht zu verwenden. Das R-Paket wurde auf CRAN und GitHub veröffentlicht.

1 Background

1.1 Introduction

The Federal Environment Agency's mandate regarding chemical safety includes testing the effects of chemicals on ecosystems and groundwater, in line with European regulations. These regulations include data requirements for risk assessments based on standardized ecotoxicological laboratory and field tests, which are described in OECD Test Guidelines (TGs). The primary reference for statistical evaluation is the OECD TG No. 54, '*Current approaches in the statistical analysis of ecotoxicity data: a guidance to application*' (2006). However, some methods mentioned in TG No. 54 need revision. A particularly important point, which is very relevant for the risk assessment of chemicals, is the statistical analysis of non-normally distributed data, in particular of count data, which are typically Poisson distributed. Such data occurs frequently in aquatic mesocosms and field studies on soil organisms and arthropods. Poisson-distributed data, which tends toward a normal distribution for large sample sizes, are often analysed using methods that assume (usually incorrectly) a normal distribution. Alternatively, non-parametric methods are sometimes used, which may have a poorer statistical performance (Daniels et al., 2021).

Since the publication of the OECD TG No. 54, a new method for evaluating such Poisson distributed data, named 'CPCAT', has been published (Lehmann et al., 2016). This method has specifically been developed for the analysis of count data, and it also addresses another issue, which is currently not yet covered sufficiently by OECD TG No. 54: The decrease of test power when conducting multiple tests (die to so called 'α-inflation'). Similar methods have also been developed for binomial data expressed as discrete values (e.g. '12 out of 20 animals affected'), like the Closure Principle and Fisher-Freeman-Halton test (CPFISH) or Cochran-Armitage test. These methods are possible candidates for a revised OECD TG No. 54.

Currently, no update to OECD Document No. 54 is planned in the OECD Test Guidelines Programme. Since December 2023, the Federal Environment Agency has been working on updating Document No. 54 through an in-house research project. This project involves a comprehensive literature review to identify available statistical methods for laboratory and field test data evaluation. The identified methods are currently being assessed for their applicability and suitability within the chemical risk assessment. The present project has been initiated within this context. In particular, the evaluation of the methods CPCAT, CPAFISH and Dunnett.GLM were evaluated and an R-package was developed to make the methods more accessible to stakeholders.

1.2 Description of CPCAT/CPFISH

When conducting statistical tests with multiple treatments, such as a control group and increasing concentrations of a test substance, ANOVA and parametric post-hoc tests (e.g. Dunnett's test) are commonly used. However, these tests require the assumptions of homogeneous variances and normally distributed data. For count data (e.g. counts of animals), these assumptions are typically violated, as the data are usually Poisson-distributed. Additionally, multiple testing using post-hoc tests can lead to α-inflation. To address these issues, CPCAT was proposed by Lehmann et al. (2016).

CPCAT has two components. The first is the Closure Principle (CP), developed by Bretz et al. (2010), which aims to eliminate α-inflation. CP applies a stepwise approach to identify at which

concentration effects begin to occur. For example, in a study with a control group and three test concentrations, the following hypotheses are tested to determine if effects exist between the control mean μ0 and the lowest concentration group mean μ1:

- H1;2;3: $\mu 0 = \mu 1 = \mu 2 = \mu 3$
- H1;3: $\mu 0 = \mu 1 = \mu 3$
- H1;2: $\mu 0 = \mu 1 = \mu 2$
- H1: $\mu 0 = \mu 1$

An effect is only considered statistically significant for a given concentration, if all the possible sub-hypotheses including the evaluated concentration are rejected.

The second part of CPCAT is the actual significance test, CAT (Computational Approach Test; introduced by Chang et al., 2010), which uses a test based on the Poisson distribution rather than a parametric test based on normal distribution assumptions. The CAT basically consists of the following steps:

1. Estimate individual Poisson parameter for each group
2. Calculate the test statistic for the input data ('distance' between the Poisson parameters of the control and the Poisson parameters of the treatment groups)
3. Estimate overall Poisson parameter from all groups (including the control)
4. Generate artificial datasets using parametric bootstrapping with the overall Poisson parameter
5. Calculate the test statistic for each artificial dataset
6. Calculate the p-value from the proportion of test statistics of the artificial datasets greater than the initially calculated test statistic for the input data

CPCAT is designed for the evaluation of Poisson distributed data (mean = variance). Therefore, the prerequisite for its application is to check the data distribution for Poisson distribution. Deviations from these assumptions, explicitly over- and underdispersion of data, should be within an acceptable range. Lehmann et al (2018a) suggested the 'Hampel Identifier' as a tool for detecting deviations from the standard assumption. However, the definition of a threshold for mild violations from the standard assumption needs to be discussed.

For quantal data (e.g. survival data, '14 out of 20 animals died') e.g. in the form of a contingency table, a similar method "CPFISH" was proposed by Lehmann et al. (2018b). Like CPCAT, CPFISH is based on the Closure Principle, but instead of a bootstrapping approach, a Fisher test is performed for all sub-hypotheses to be analyzed.

2 Overview about working tasks

The general aim of this project was to provide technical support for the Federal Environment Agency's current in-house research project to review OECD Document No. 54. Specifically, the aim of the project was to develop an R package that includes a function for the statistical analysis of Poisson-distributed data using the 'CPCAT' method (Lehmann et al., 2016). The source code had been documented in the original publication of Lehmann et al. (2016). A citable R package needed to be created within this project, documented and made available to the public on the CRAN repository (for access and maintenance reasons, it was agreed that the UBA would upload the data to the platforms). In addition, the source code had to be uploaded on the GitHub https://opencode.de/de.

The following steps and sub-steps were formulated before the start of the project and were partially reformulated during the course of the project in consultation with the Federal Environment Agency. Changes to the steps are explained in more detail in the results section.

In a first step, errors already identified in the existing R source code for CPCAT should to be corrected, including:

- Removing the source code for possible use as a one-sided test procedure. Only the two-sided test shall be implemented.
- Error correction for the analysis of data sets with only zero values in all replicates of the control treatment: Currently, for treatments without response (also all replicates =0), a p-value=0 is output in these cases (instead of p~1). A simple workaround for error correction exists and must be implemented.

In a second step, the R package to be developed shall be expanded to include the following functions and properties in addition to the CPCAT function:

- For the CPCAT test, a measure of the degree of overdispersion shall be output in the results display for all treatments in the data set (for example, 'Hampel Identifier', see Lehmann et al, 2018a).
- An additional R function shall be implemented for the statistical test CPFISH (Lehmann et al., 2018b).
- For the above-mentioned tests (CPCAT, CPFISH), an additional argument should be integrated within the functions to allow confidence intervals to be output for the calculated toxicity thresholds according to the concept of Mair et al (2020). Bootstrap methods can be used for this purpose.
- A further R function shall be generated that performs a GLM-Dunnett test with a quasi-Poisson link function (Hothorn et al., 2020) and outputs p-values with corresponding NOEC/LOEC derivations. The method can access the R package multcomp.
- The implemented functions are to be compared with results from the literature and thus validated.
- The functions to be developed for the above-mentioned test methods (CPCAT, CPFISH) should output all p-values of the respective sub-hypotheses and the corresponding maximum p-values of all treatments, including a derivation of the corresponding NOEC/LOEC value, as a summarised result.

In a third step, power calculations of the three implemented methods are performed as a function of simulated effect sizes for typical test designs of reproduction and field tests and reported.

3 Results

3.1 Step 1: Error correction

3.1.1 One-sided test procedure

In the original publication by Lehmann et al. (2016), the option of performing the CPCAT method as a one- or two-sided test was implemented. This option was removed from the function.

3.1.2 Erroneous p-values for all-zero data

With the original R code from Lehmann et al. (2016), it was not possible to generate correct results in every case. If all values in the control group were zero and all values in a treatment group were also zero, the method as described in Lehmann et al. (2016) may return an incorrect (too small) p-value or even a p-value of 0. However, a p-value of 1 should actually be returned, as the groups are identical. For this reason, the code was modified to address such scenarios.

Specifically, an additional check was added to the code to determine whether all values of the control and all values of the considered treatments are zero. In this case, a p-value of 1 is now returned directly. The problem emerged from the calculation of the p-values when comparing the test statistic of the measured data with those of the data artificially generated using bootstrapping. If all input data is zero, the parametric bootstrapping is conducted with a Poisson parameter of 0 producing always the same results. Therefore, the artificial data can never differ from the input data and subsequently also the test statistics are all the same and identical to the test statistic of the input data. As Lehmann et al. (2016) defined the p-value to be the proportion of test statistics from artificial data greater than the test statistic of the measured data, this proportion was always zero (test statistics were always identical).

3.2 Step 2: Development of R functions

3.2.1 CPCAT

The basic algorithm of CPCAT was provided by the Federal Environment Agency. The code was thoroughly revised, particularly with regard to names of variables and comments in the code, to enhance readability. Furthermore, checks were introduced in the code to prevent errors (e.g. incorrect formatting of input data) and new functionality was added (see table below).

Table 1: CPCAT function parameters introduced to the source code

Parameter (see code)	Description	Default value
groups	Vector containing group information (e.g. dose or concentration); numbers or character strings allowed	No default, must be specified
counts	Vector containing count data; only non-negative numbers allowed	No default, must be specified

Parameter (see code)	Description	Default value
control.name	Character string specifying the control group name; if not specified, the first group in the groups vector is considered the control	NULL
bootstrap.runs	Number of bootstrap runs	10000
hampel.threshold	Threshold for Hampel identifier (measure for over- and under-dispersed data)	5
use.fixed.random.seed	Use or don't use fixed random seed for the bootstrapping procedure to enable reproducible results	TRUE
get.contrasts.and.p.values	Get each row of the contrast matrices evaluated	FALSE
show.output	Print out results in the console (results will always be returned as R object when calling the function)	TRUE

Further details can be found in the published source code (provided in the Appendix, section A.2).

3.2.2 Measure of degree of overdispersion and underdispersion

By definition, the variance of Poisson-distributed data is equal to the mean of the data. If the variance is greater than the mean, there is overdispersion in the data. If the variance is less than the mean, there is underdispersion in the data. To quantify the over- and underdispersion, the 'Hampel identifier' was implemented as a measure, as used in Daniels et al. (2021). The data is considered underdispersed, if the difference between the mean and variance is greater than a certain threshold (default threshold of 5 can be adjusted by changing a function parameter). The analogue applies to overdispersion.

If data is considered to be over- or underdispersed, a notification text is printed when the CPCAT function is called. Further details can be found in the published source code.

3.2.3 CPFISH

The basic algorithm of CPFISH was provided by the Federal Environment Agency. The code was thoroughly revised, particularly with regard to names of variables and comments in the code, to enhance readability. Furthermore, checks were introduced in the code to prevent errors (e.g. incorrect formatting of input data) and new functionality was added (see table below).

Table 2: CPFISH function parameters introduced to the source code

Parameter (see code)	Description	Default value
contingency.table	The contingency table is a matrix with observed data (e.g. survival counts, whereas survival must be in the first row)	No default, must be specified
control.name	Character string specifying the control group name; if not specified, the first column in the contingency table is considered the control	NULL
simulate.p.value	Use simulated p-values in implemented Fisher test or not (not to use simulated p-values may lead to errors for higher sample sizes)	TRUE
use.fixed.random.seed	Use or don't use fixed random seed for simulating p-values in the Fisher test to enable reproducible results	TRUE
show.output	Print out results in the console (results will always be returned as R object when calling the function)	TRUE

Further details can be found in the published source code (provided in the Appendix, section A.4).

3.2.4 Confidence limits for toxicity thresholds

The calculation of confidence intervals for toxicity thresholds was originally intended to be implemented in the main functions of the tests (CPCAT, CPFISH and Dunnett.GLM) and based on the concept of Mair et al. (2020). However, it was found that this was not reasonable as the data was transformed in Mair et al. (2020) to obtain approximated normally distributed data. However, this procedure is not appropriate in the context of CPCAT, as an explicit advantage of the method is that the data are not transformed and continue to be Poisson-distributed.

An appropriate alternative for implementing power analyses and MDD (Minimum Detectable Difference) calculation is the use of the bootstrapping method (bMDD = bootstrap MDD). However, this means that the bootstrapping-based concept of CPCAT has to be wrapped by another bootstrapping iteration. The resulting time needed for the calculations is not suitable for integrating the functionality into the main functions of the tests. Instead, the calculation of the bMDD was outsourced and implemented in a joint function due to the similarity in content to the power calculation. Further details can be found in the chapter on power calculation.

3.2.5 Dunnett.GLM test

Another way to analyse count data is to fit a GLM and then apply a Dunnett test to the model parameters. By using a quasi-Poisson distribution and a logarithmic link function, even

overdispersed data can be modelled. The approach is also suitable for analysing count data as it can be used for non-normally distributed data (which is common for count data) and at the same time the Dunnett test controls the type I error rate when comparing multiple experimental groups with a control group. Therefore, the Dunnett.GLM function was implemented as an alternative for the CPCAT approach.

The basic approach of the method was taken from the publication by Hothorn and Kluxen (2020). The code has been thoroughly reviewed, revised and extended, particularly with regard to error handling and commenting of the code.

Since the method from Hothorn and Kluxen (2020) provides a quasi-Poisson link function for the GLM and the data are transformed accordingly, there is a methodological difficulty with all-zero treatment groups. Although no results should actually emerge, the GLM returns an estimate with an extremely high standard error, which leads to a very high p-value for the all-zero treatment group (i.e. the deviation of this treatment group from the control is assessed as not statistically significant, even if the effect is very pronounced). To deal with this methodological shortcoming, two options were implemented:

- The 'identity.link' option: the 'identity' link is used in the GLM instead of the 'log' link, i.e. the data are no longer transformed. Note that this means a methodological deviation from Hothorn and Kluxen (2020), which may distort the results.
- The 'log(x+1)' option: The 'log' link is retained and 1 is added to each count value at the start of the procedure so that the subsequent log-transformation can be carried out without any problems. Note that the preceding data transformation may distort the results.

Both options can only be used if the data contains groups that only contain zeros. A notification text is provided in the results if one of the options was actually used.

A complete list of implemented function parameters is shown below.

Table 3: Dunnett.GLM function parameters introduced to the source code

Parameter (see code)	Description	Default value
groups	Vector containing group information (e.g. dose or concentration); numbers or character strings allowed	No default, must be specified
counts	Vector containing count data; only non-negative numbers allowed	No default, must be specified
control.name	Character string specifying the control group name; if not specified, the first group in the groups vector is considered the control	NULL
zero.treatment.action	Character string specifying the method for dealing with treatments only containing zeros; either use "identity.link" or "log(x+1)"	"identity.link"

Parameter (see code)	Description	Default value
show.output	Print out results in the console (results will always be returned as R object when calling the function)	TRUE

Further details can be found in the published source code (provided in the Appendix, section A.3).

3.2.6 Verification of the code by comparison with results from the literature

To verify the implementation of the functions CPCAT, CPFISH, and Dunnett.GLM, results from CPCAT were compared to results provided in Lehmann et al. (2016), results from CPFISH were compared to results provided in Lehmann et al. (2018b) and results from Dunnett.GLM were compared to results provided in Hothorn and Kluxen (2020). The comparison of the results refers to the data specified in the respective publication (simulation runs with artificial data and parameter variation excluded).

CPCAT results from Lehmann et al. (2016) were identical to those from the implemented CPCAT function (see following table). Results for dataset 4 were not included in the table as the raw data were not provided in the original publication.

Table 4: CPCAT results from Lehmann et al. (2016) compared with results from the implementation provided in the present document.

Dataset	Dose group	Indicated as statistically significantly different from control in Lehmann et al. (2016)	Indicated as statistically significantly different from control by present CPCAT implementation
1	1.06	No	No
1	1.59	No	No
1	2.38	No	No
1	3.53	Yes	Yes
1	5.29	Yes	Yes
1	7.93	Yes	Yes
2	1.06	Yes	Yes
2	1.59	No	No
2	2.38	Yes	Yes
2	3.53	Yes	Yes
2	5.29	Yes	Yes

Dataset	Dose group	Indicated as statistically significantly different from control in Lehmann et al. (2016)	Indicated as statistically significantly different from control by present CPCAT implementation
2	7.93	Yes	Yes
3	1.06	Yes	Yes
3	1.59	Yes	Yes
3	2.38	Yes	Yes
3	3.53	Yes	Yes
3	5.29	Yes	Yes
3	7.93	Yes	Yes
5	0.2	Yes	Yes
5	1	No	No
5	5	Yes	Yes
5	25	Yes	Yes
6	T1	Yes	Yes
6	T2	Yes	Yes
6	T3	No	No
6	T4	Yes	Yes
6	T5	Yes	Yes
7	T1	No	No
7	T2	No	No
7	T3	No	No
7	T4	Yes	Yes
7	T5	Yes	Yes

CPFISH results from Lehmann et al. (2018b) were identical to those from the implemented CPFISH function (see following table).

Table 5: CPFISH results from Lehmann et al. (2018b) compared with results from the implementation provided in the present document.

Dataset	Reference point according to Lehmann et al. (2018b)	Reference point according to present CPFISH implementation
1	NOEL at 3 mg/L (treatment 2)	NOEL at 3 mg/L (treatment 2)
2	LOEL at 3 mg/L (treatment 3)	LOEL at 3 mg/L (treatment 3)

In order to compare the Dunnett.GLM calculations with each other, the settings in the function call of Hothorn and Kluxen (2020) had to be adjusted (instead of a one-sided test, a two-sided test was used). In the implementation of Hothorn and Kluxen (2020), the zero values in the highest dose group cause that this group is no longer considered statistically significantly different from the control. In the present implementation of the Dunnett.GLM function, the method is adapted accordingly so that dose groups with all-zero values can also be considered. Consequently, the results differ with regard to the highest dose group (see following table).

Table 6: Dunnett.GLM results based on source code from Hothorn and Kluxen (2020) compared with results from the implementation in the present document ('daphnia' data from Hothorn and Kluxen, 2020, used for comparison).

Dose group	p-values based on implementation provided in Hothorn and Kluxen (2020)	p-values from present implementation of the Dunnett.GLM function (log(x+1) transformation used)	p-values from present implementation of the Dunnett.GLM function (identity link used)
1.56	0.401	0.393	0.342
3.12	<0.001	<0.001	<0.001
6.25	0.006	0.006	0.006
12.5	<0.001	<0.001	<0.001
25	1.000	<0.001	<0.001

3.2.7 Output format

The standard output of the R functions for CPCAT, CPFISH and Dunnett.GLM includes an R list object with a results variable containing hypothesis-specific p-values and levels of significance and an information variable containing additional information (e.g. Hampel identifier info for overdispersion if detected, information about used settings such as using simulated p-values for the Fisher test in CPFISH, or derivations for the NOEC and LOEC).

An example of the output is provided in the Appendix (section C.1).

3.3 Sensitivity and test power

In order to interpret the results of a statistical test, it is important to find out which effect size can be detected with which power. Otherwise, a test could show statistically insignificant differences simply because the difference between control and treatment was too small to be detected, e.g. due to a small sample size. The difference between the values of a control and those of a treatment group that can just be shown to be statistically significant with a given power (set to 80 %) is called Minimum Detectable Difference, MDD (Duquesne et al., 2020; Zar, 2013). While MDD calculations are frequently used for parametric tests, such as the t-test, they can also be calculated for non-parametric tests using bootstrapping.

The idea behind the bootstrap MDD (bMDD) calculation is to initially assume an effect size of 0 for each treatment individually in a bootstrapping procedure and to gradually increase the effect size (the effect sizes of the other treatments remain unchanged). In each step, N data sets are generated by parametric sampling (N = number of bootstrap runs). The corresponding statistical test is carried out with each data set generated in this way. The proportion of statistically significant differences between the control and the considered treatment group corresponds to the power of the test for the given effect size. Conversely, the effect size can be interpreted as bMDD corresponding to the associated power. A similar method for bMDD determination has already been published by van der Hoeven (2007) involving a gradual shift in the data to determine an bMDD for the non-parametric Mann-Whitney-U test.

More specifically, bMDD calculations for CPCAT and the Dunnett.GLM test were implemented as follows:

1. Estimate Poisson parameter from the control group
2. Conduct number of bootstrap iterations for each treatment group:
   a. Sample data from a Poisson distribution for the currently evaluated treatment group starting with the Poisson parameters of the control (data of other groups remain unchanged)
   b. Conduct the statistical test (CPCAT or Dunnett.GLM) and check for significance
3. Stepwise shift the Poisson parameter and repeat the bootstrapping until all tests of step 2b showed significant differences
4. The bMDD is considered the lowest shift value for which at least 80 % of the bootstrap runs showed significant differences (higher shift values are not allowed to result in less than 80 % significant differences)

For CPFISH, contingency tables (quantal data) are used, i.e. the effect size must be controlled via the gradual change in the probability of a binomial distribution. Therefore, the bMDD calculation for CPFISH must be modified accordingly:

1. Estimate probability of ‘success’ from the control group
2. Conduct number of bootstrap iterations for each treatment group:
   a. Sample data from a binomial distribution for the currently evaluated treatment group starting with the ‘success’ probability of the control (data of other groups remain unchanged)
   b. Conduct CPFISH and check for significance
3. Stepwise shift the probability and repeat the bootstrapping until all tests of step 2b showed significant differences
4. The bMDD is considered the lowest shift value for which at least 80 % of the bootstrap runs showed significant differences (higher shift values are not allowed to result in less than 80 % significant differences)

Calculating the power for a given data set is straightforward: parametric bootstrapping is used to generate artificial data for each treatment group individually, which is then used to perform the statistical test. The proportion of significant results from the bootstrapping describes the power of the test for the corresponding treatment group.

Further details can be found in the published source code. Exemplary power and bMDD calculations for the implemented methods were conducted as a function of simulated effect sizes for typical test designs of reproduction and field tests and reported in the appendix to the R package documentation.

Finally, in order to test the influence of the effect size on the test power of CPCAT, Dunnett.GLM and CPFISH, artificial data sets were analysed. Data sets with different numbers of groups and replicates, different mean control counts and different dispersion factors were tested (dispersion factor DF=variance/mean; R package ZIGP used for sampling).

Similar to the procedure for the bMDD calculation, the effect size was gradually increased for the highest concentration in relation to the control and the power was determined using parametric bootstrapping. For CPCAT and Dunnett.GLM two scenarios were used for four groups, representing one control and three treatments: In a first scenario, a deviation from the control was assumed in the highest dose group, while no deviations from the control were assumed in the other two dose groups. In the second scenario, a monotonically increasing deviation compared to the control was assumed across all three treatment groups (the first treatment showed 1/3 deviation of the deviation in the third treatment and the second treatment had 2/3 of the deviation of the third treatment). An example of the selected deviations is provided in the following table.

Table 7: Example for deviation implementation with mean control count of 25 and 20 % effect in the highest treatment group. The mean counts correspond to the lambda parameter of the Poisson distribution used for sampling.

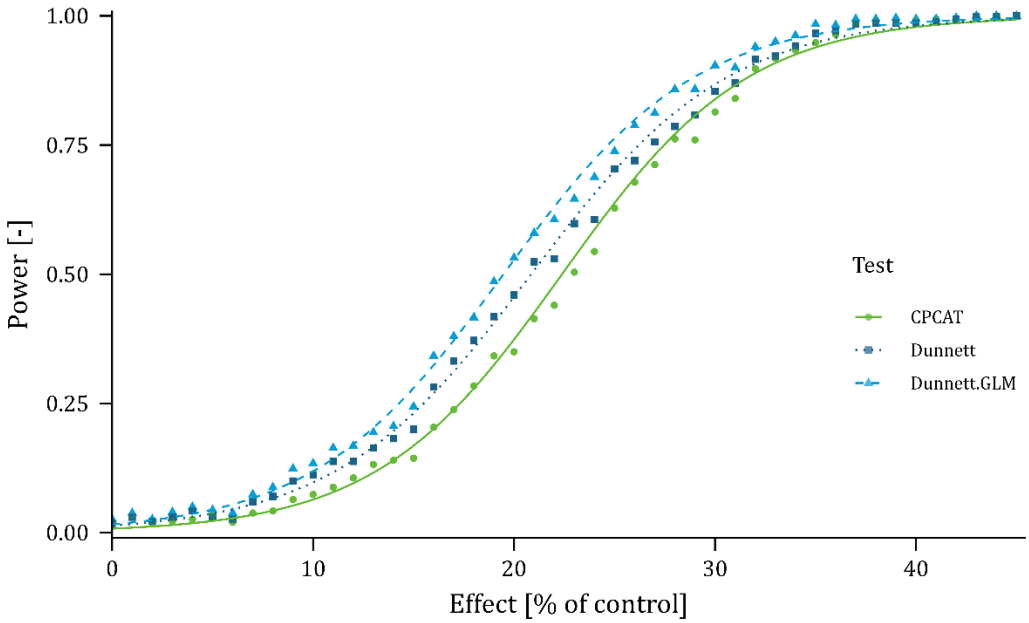
Group	Mean counts for effects only in highest treatment (% of control / % effect)	Mean counts for effects monotonically increasing (% of control / % effect)
Control	25.00 (100.0 % / 0.0 %)	25.00 (100.0 % / 0.0 %)
T1	25.00 (100.0 % / 0.0 %)	23.33 (93.3 % / 6.7 %)
T2	25.00 (100.0 % / 0.0 %)	21.67 (86.7 % / 13.3 %)
T3	20.00 (80.0 % / 20.0 %)	20.00 (80.0 % / 20.0 %)

As expected, the relationship between effect size and power was described by a sigmoid curve (see figures in following sections). However, it was noticeable that the curve shifted depending on the total number of control and treatment groups. The same power is only achieved for data

with more groups with a more pronounced effect size. This phenomenon was analysed in more detail in the following chapter in order to determine the characteristics of the test procedures.

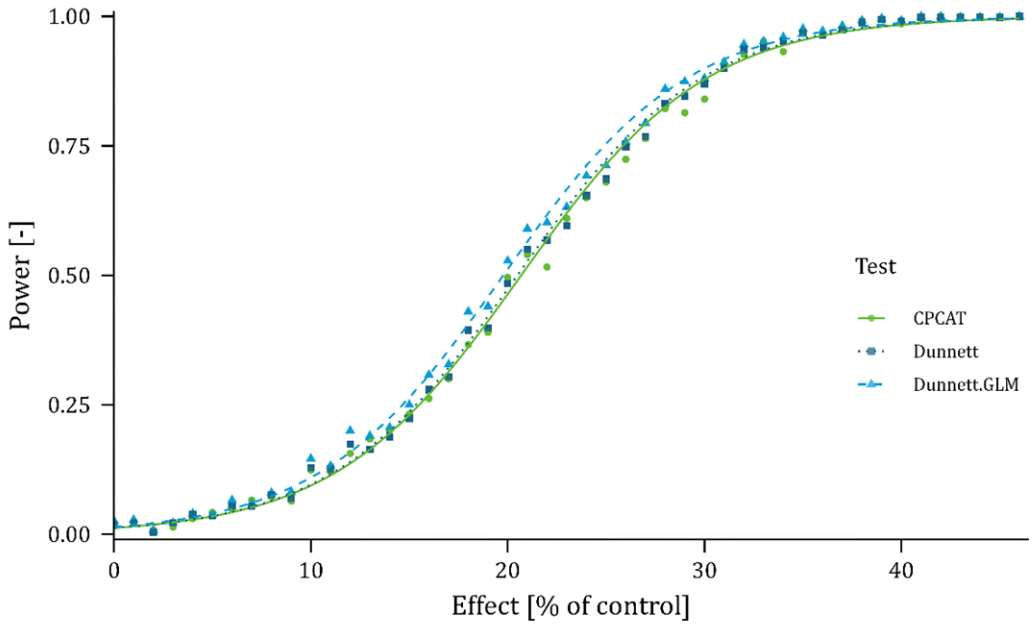
As a first step the relationship between power and effect size was compared between CPCAT and Dunnett.GLM for a data set with four groups (including one control and three treatments), ten replicates per group and a mean control count of 25. For additional validation of the Dunnett.GLM approach, a standard Dunnett test (ANOVA post-hoc test) was also included. The Dunnett.GLM approach showed the best performance for scenarios with effects only in the highest treatment group (see Figure 1). Differences between the three methods were more pronounced for the scenario with deviations in the highest treatment group and no deviations in the two lower treatments (see Figure 1). The differences narrowed, when monotonic effects across all treatment groups were assumed (see Figure 2).

Figure 1: Relationship between effect size and power of CPCAT, Dunnett.GLM and a standard Dunnett test for four groups (one control and three treatments with effects only in highest treatment).



Source: own illustration, WSC Scientific GmbH.

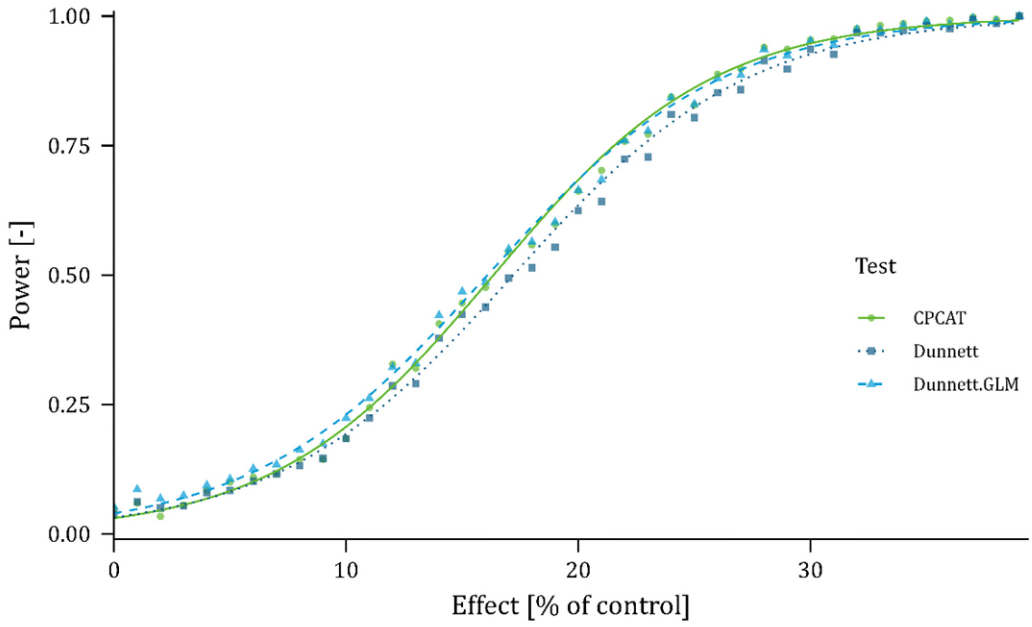
Figure 2: **Relationship between effect size and power of CPCAT, Dunnett.GLM and a standard Dunnett test for four groups (one control and three treatments with monotonically increasing effects).**



Source: own illustration, WSC Scientific GmbH.

When considering only two groups (one control and one treatment with effects), the differences between CPCAT and Dunnett.GLM vanished and both methods performed slightly better than the standard Dunnett test due to the more appropriate distributional assumptions (see Figure 3). It should be noted, that the complexity of the tests is reduced when only two groups are considered, i.e. the Closure Principle of CPCAT is not applied and the Dunnett test is reduced to a t-test.

Figure 3: Relationship between effect size and power of CPCAT, Dunnett.GLM and a standard Dunnett test for two groups (one control and one treatment with effects).



Source: own illustration, WSC Scientific GmbH.

3.3.1 CPCAT and Dunnett.GLM power analysis

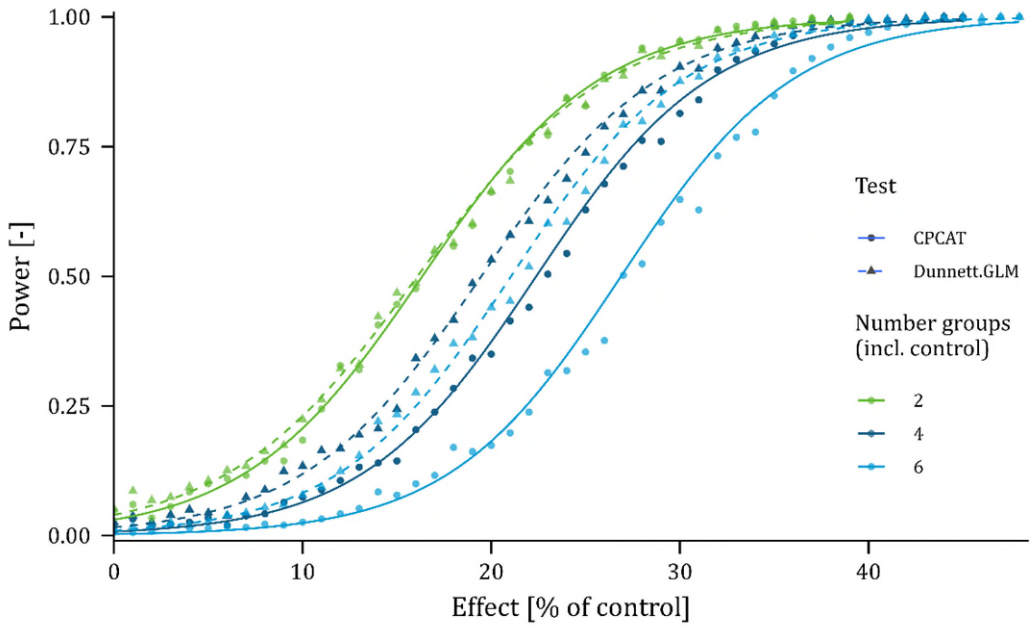
The following sections and figures describe the relationship between power and effect size for CPCAT and Dunnett.GLM considering various characteristics a dataset might have (different numbers of groups and replicates, different mean control counts and different dispersion factors). The default setup included four dose/concentration groups (including one control and three treatments), ten replicates per group, a mean control count of 25 and a dispersion factor of one (i.e. variance = mean). One of these parameters was varied while the others were fixed in order to evaluate the influence of the parameter. Each analysis was performed either with effects only in the highest treatment group or with monotonically increasing effects across all treatment groups.

3.3.1.1 Varying the number of treatment groups

To evaluate the sensitivity of CPCAT and Dunnett.GLM to the number of tested treatment groups, the tests were performed with two, four and six groups (including one control group).

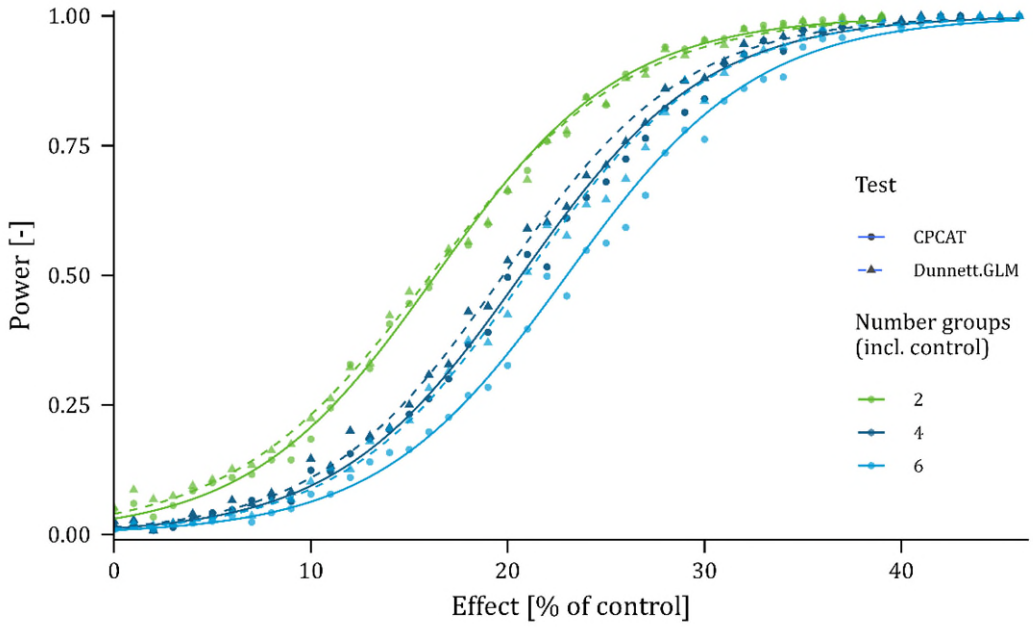
The graphs showing power vs. effect size for both CPCAT and Dunnett.GLM shifted with an increasing number of treatment groups. For two groups (one control and one treatment), CPCAT and Dunnett.GLM delivered almost identical results. For four and six treatment groups, Dunnett-GLM had a higher power than CPCAT, whereby the difference was greater if only effects in the highest treatment group were assumed (see Figure 4). Differences between both methods became smaller, if monotonically increasing effects were present across all treatment groups (see Figure 5). The reduction in the differences was mainly due to the change in the power of CPCAT; the implementation of the effects (only in the highest treatment group or monotonically increasing) had no significant influence on Dunnett.GLM.

Figure 4: **Relationship between effect size and power of CPCAT and Dunnett.GLM for different numbers of groups (one control and one to five treatments with effects only in the highest treatment).**



Source: own illustration, WSC Scientific GmbH.

Figure 5: **Relationship between effect size and power of CPCAT and Dunnett.GLM for different numbers of groups (one control and one to five treatments with monotonically increasing effects).**

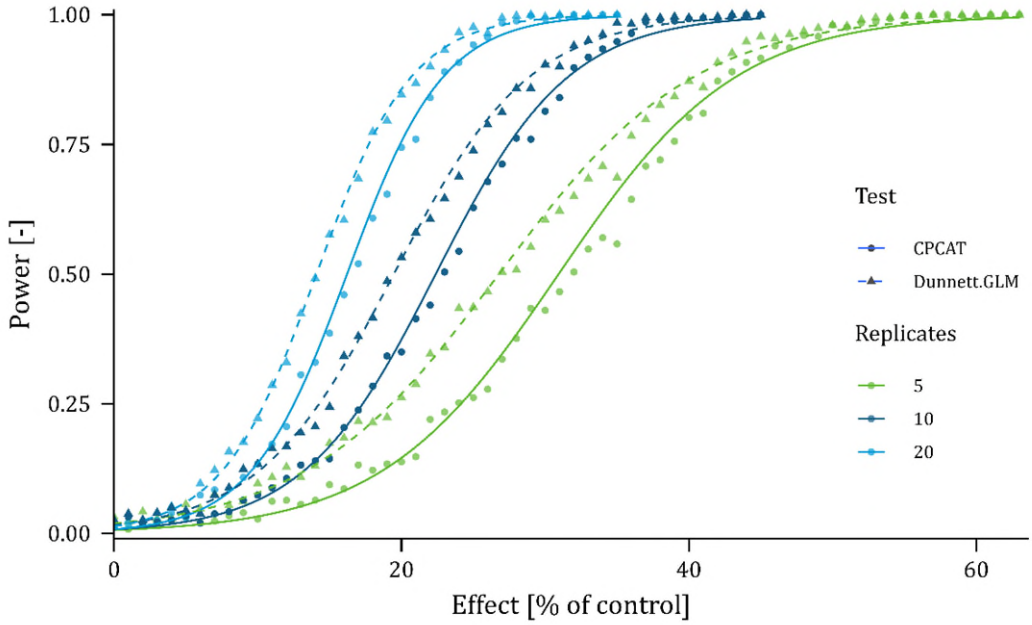


Source: own illustration, WSC Scientific GmbH.

3.3.1.2 Varying the number of replicates

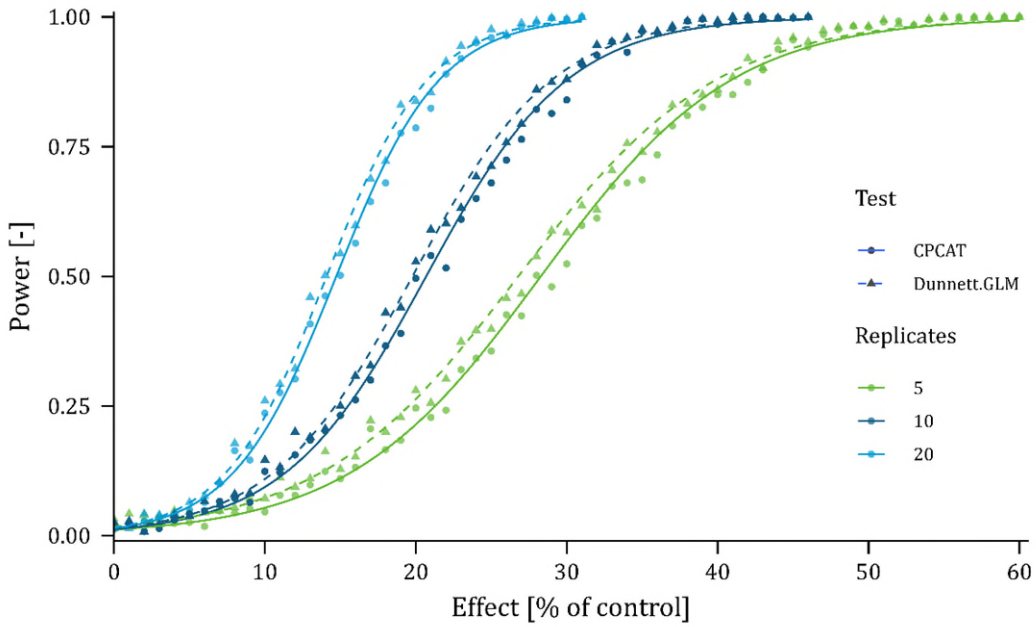
The graphs, showing power vs. effect size for both CPCAT and Dunnett.GLM, shifted for raising numbers of replicates. Considering four groups (one control and three treatments), Dunnett.GLM showed a higher power for all numbers of replicates tested, compared to CPCAT. The power difference between the two methods was smaller for monotonically increasing effects over the treatment groups (see Figure 7) and larger for the scenario with effects only in the highest treatment group (see Figure 6). The power of CPCAT changed slightly more than the power of Dunnett.GLM.

Figure 6: Relationship between effect size and power of CPCAT and Dunnett.GLM for different numbers of replicates (one control and three treatments with effects only in the highest treatment).



Source: own illustration, WSC Scientific GmbH.

Figure 7: Relationship between effect size and power of CPCAT and Dunnett.GLM for different numbers of replicates (one control and three treatments with monotonically increasing effects).



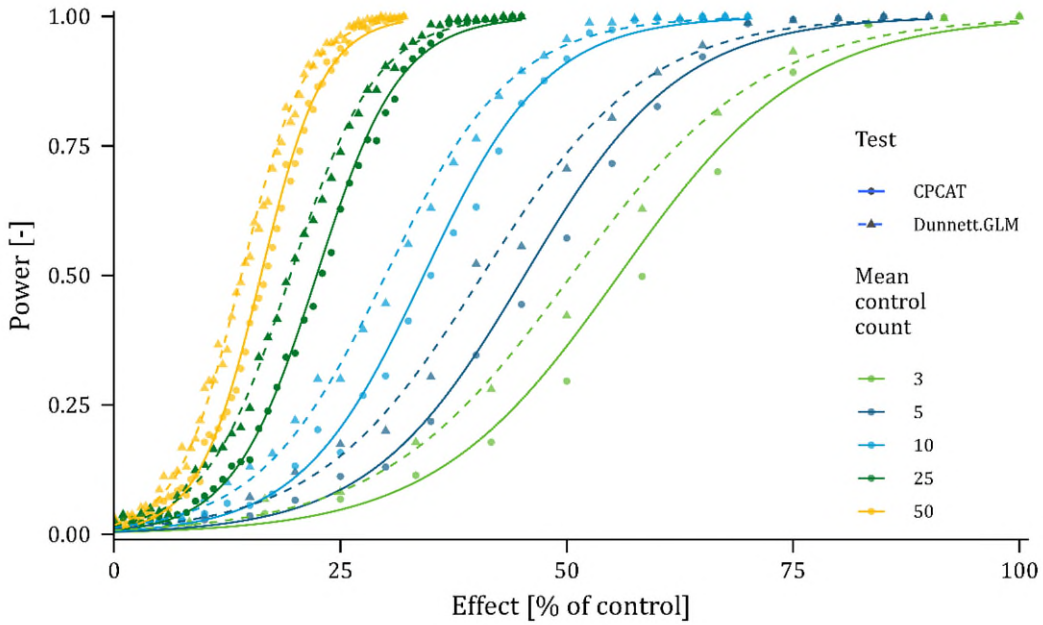
Source: own illustration, WSC Scientific GmbH.

3.3.1.3 Varying the mean control count

Additional calculations were conducted in order to test if the abundance has an effect on the performance of the tests, e.g. if CPCAT or Dunnett.GLM is more or less sensitive for higher or lower counts.

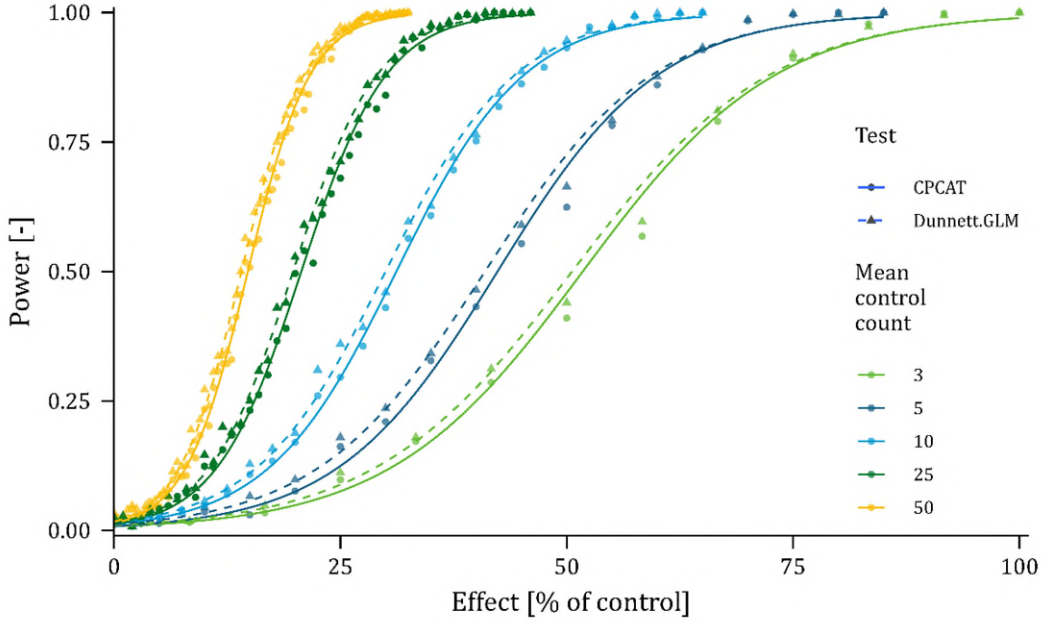
The graphs, showing power vs. effect size for both CPCAT and Dunnett.GLM, shifted for different mean control counts. Considering four groups (one control and three treatments), Dunnett.GLM showed a slightly higher power for all mean control counts tested compared to CPCAT. The power difference between the two methods was smaller for monotonically increasing effects over the treatment groups (see Figure 9) and larger for effects only in the highest treatment group (see Figure 8).

Figure 8: Relationship between effect size and power of CPCAT and Dunnett.GLM for different mean control counts (one control and three treatments with effects only in the highest treatment).



Source: own illustration, WSC Scientific GmbH.

Figure 9: Relationship between effect size and power of CPCAT and Dunnett.GLM for different mean control counts (one control and three treatments with monotonically increasing effects).



Source: own illustration, WSC Scientific GmbH.

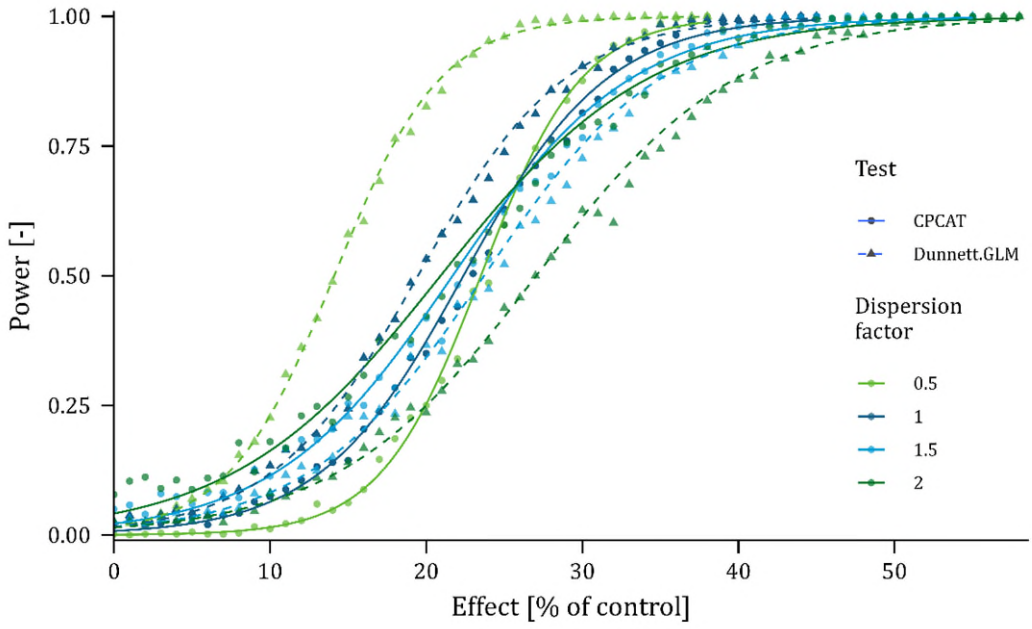
3.3.1.4 Varying the dispersion factor

Since the dispersion of measured data may vary considerably, in particular regarding field trials, it was also analysed how the test power of the CPCAT and Dunnett.GLM changes for differently

dispersed data. A dispersion factor of 1 means that mean=variance. A dispersion factor < 1 represents underdispersion, a dispersion factor > 1 represents overdispersion (e.g. a dispersion factor of 2 means that the variance is twice as large as the mean value).

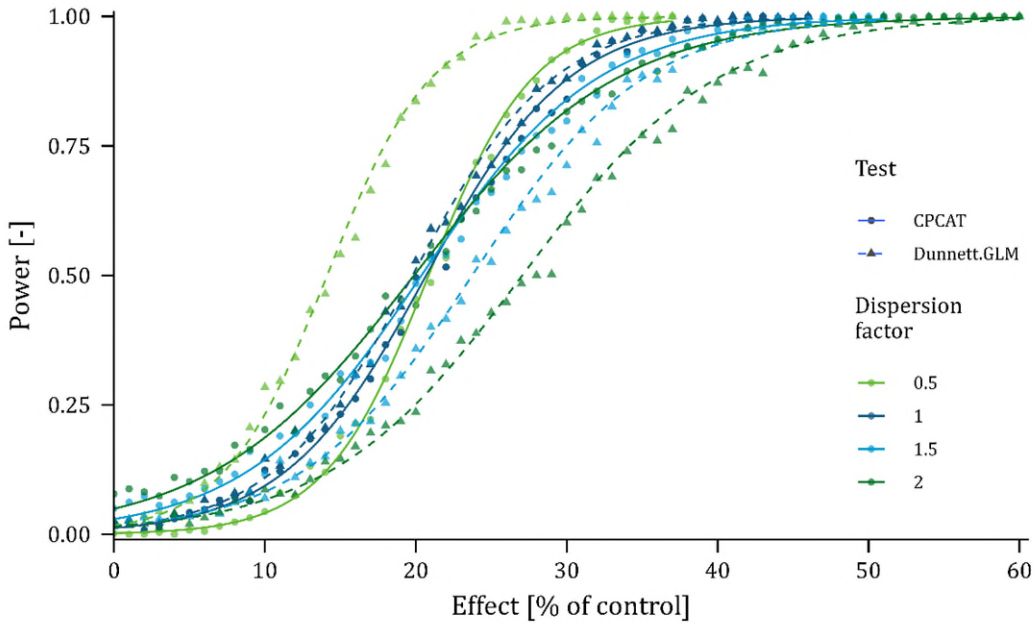
The graphs, showing power vs. effect size for Dunnett.GLM, shifted for different dispersion factors. The curves for CPCAT varied around an inflection point and were not shifted with the effect size (see Figure 10 and Figure 11). This was expected, as CPCAT only takes the group mean values into account for the data evaluation (the within-group variability is ignored in the calculations). Considering four groups (one control and three treatments), Dunnett.GLM showed a higher power for underdispersed data (dispersion factor < 1) and lower power for overdispersed data (dispersion factor > 1), compared to CPCAT. However, CPCAT showed a higher probability for false positive detection of significant effects, when evaluating overdispersed data. This can be explained, as CPCAT does not consider the within-group variability > or < mean. When comparing the power results of CPCAT, the power was generally higher for monotonically increasing effects over three treatment groups compared to effects only in the highest treatment group and no deviations from the control in two lower treatments.

Figure 10: Relationship between effect size and power of CPCAT and Dunnett.GLM for different dispersion factors (one control and three treatments with effects only in the highest treatment).



Source: own illustration, WSC Scientific GmbH.

Figure 11: Relationship between effect size and power of CPCAT and Dunnett.GLM for different dispersion factors (one control and three treatments with monotonically increasing effects).



Source: own illustration, WSC Scientific GmbH.

3.3.2 CPFISH power analysis

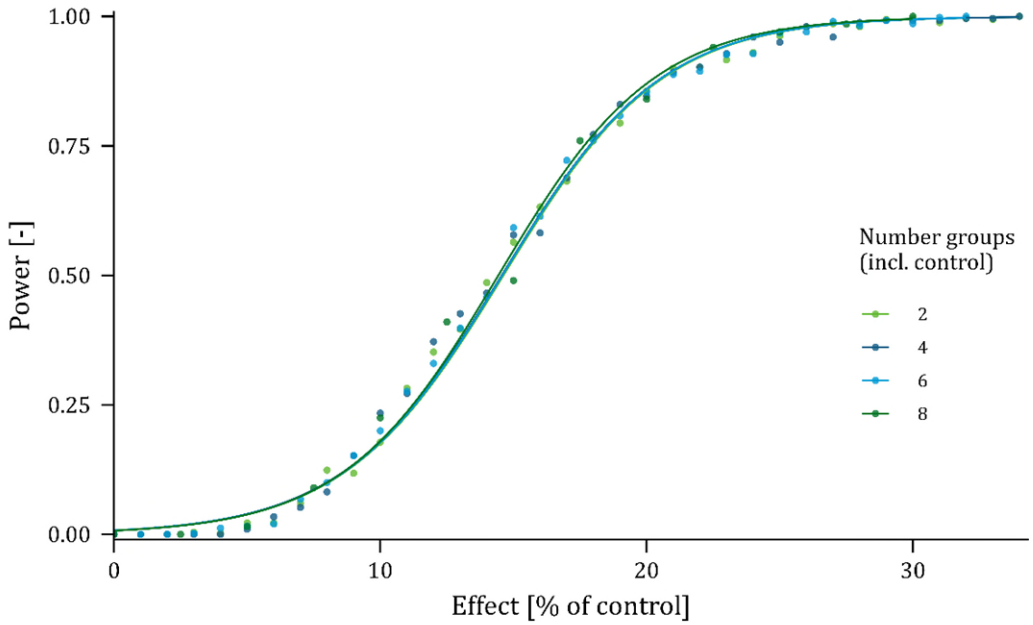
The following figures describe the relationship between power and effect size for CPFISH considering various characteristics a data set might have (different numbers of groups and replicates). The default setup included four dose/concentration groups, four replicates per group and 10 introduced individuals per group. The number of groups and replicates were varied while the other parameters were fixed.

3.3.2.1 Varying the number of treatment groups

To evaluate the sensitivity of CPFISH to the number of tested treatment groups, the tests were performed with two, four, six and eight groups (including control group).

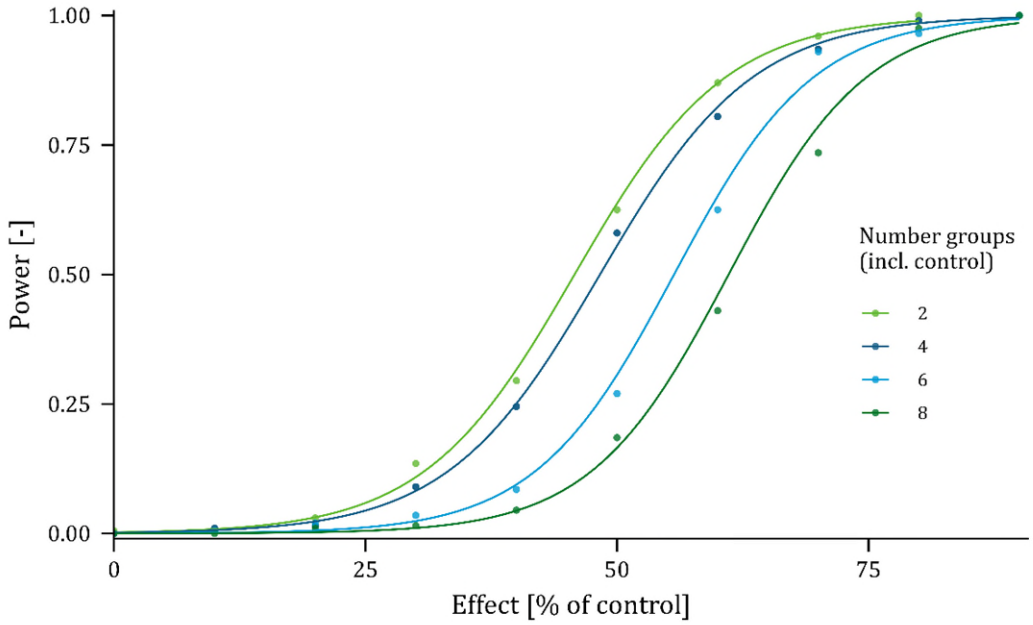
The number of treatment groups was first varied assuming 100 % survival in the control. As the number of groups seemed to have no impact on the power (see Figure 12), another parameter (the background mortality / control survival) was varied and it could be shown that the number of groups affects the power, only if the survival in the control is lower than 100 % (see Figure 13).

Figure 12: Relationship between effect size and power of CPFISH for different numbers of groups (one control and one to seven treatments with effects only in the highest treatment) assuming 100 % survival in the control.



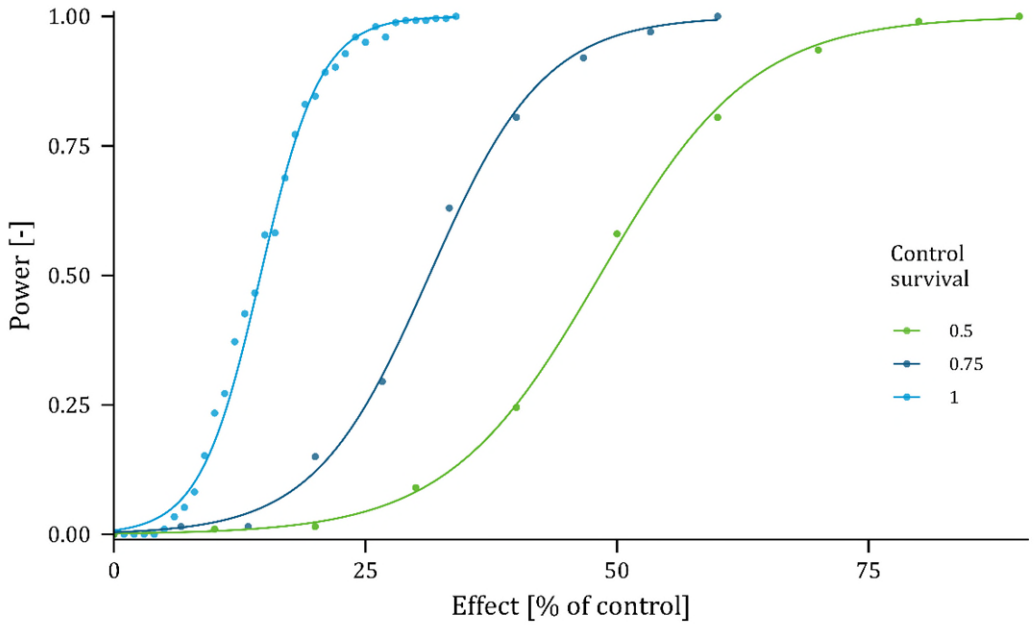
Source: own illustration, WSC Scientific GmbH.

Figure 13: Relationship between effect size and power of CPFISH for different numbers of groups (one control and one to seven treatments with effects only in the highest treatment) assuming 50 % survival in the control.



Source: own illustration, WSC Scientific GmbH.

Figure 14: Relationship between effect size and power of CPFISH for different survival levels in the control (one control and three treatments with effects only in the highest treatment).

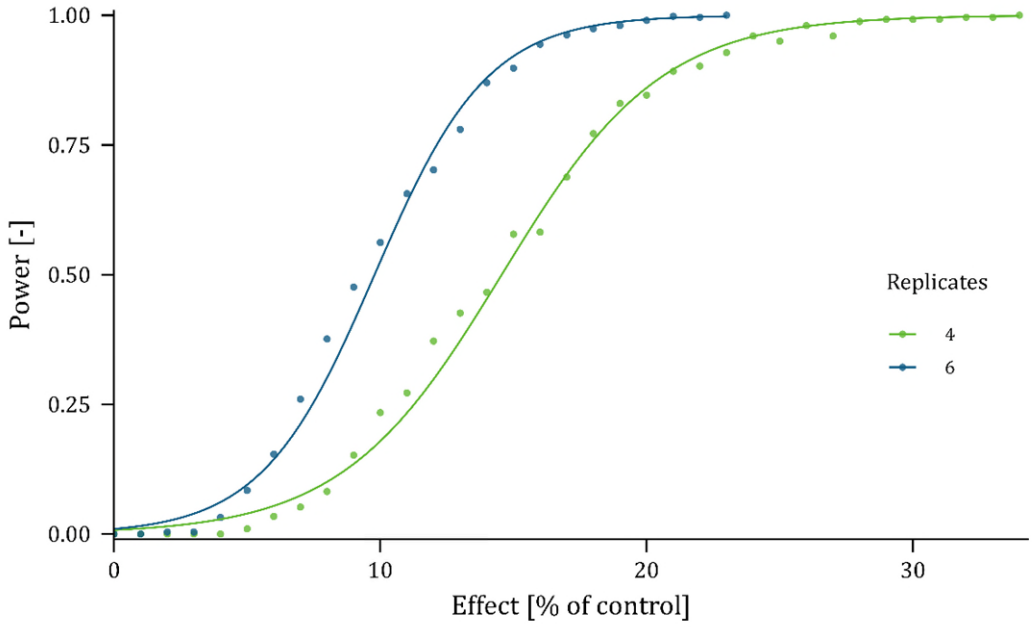


Source: own illustration, WSC Scientific GmbH.

3.3.2.2 Varying the number of replicates

The graphs showing power vs. effect size for CPFISH shifted significantly for different numbers of replicates. Considering four groups (one control and three treatments), CPFISH showed a higher power for a higher number of replicates (see Figure 15).

Figure 15. Relationship between effect size and power of CPFISH for different numbers of replicates (one control and three treatments with effects only in the highest treatment).



Source: own illustration, WSC Scientific GmbH.

3.4 Influence of increasing treatment levels

3.4.1 The sensitivity of p-values to the number of treatments without effects

During the testing of CPCAT and CPFISH it was observed that the test power was particularly low when there were several treatment levels without an effect. Therefore, it was decided to analyse systematically, how the sensitivity of the test methods change depending on the number of treatment groups without effects.

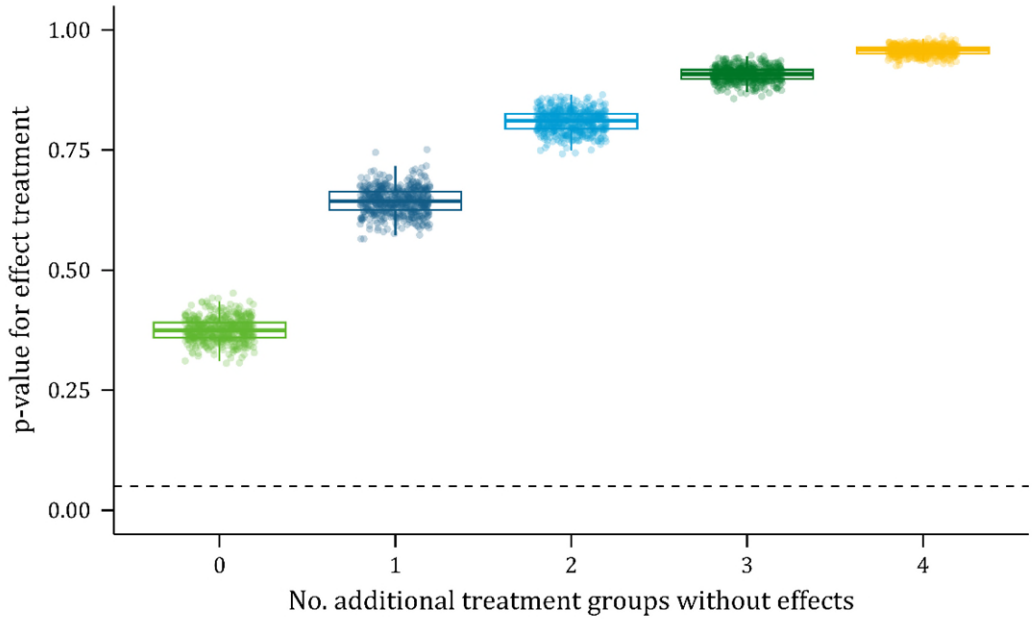
To find out how sensitively the p-values react to different numbers of treatment groups without effects, the following configuration was set up:

- Each group had a sample size of 5.
- For the control and treatment groups without effects one set of Poisson-distributed data was sampled with a Poisson parameter of 30 (count of 30 individuals on average).
- For the treatment group with effects (e.g. the highest concentration) effects of 10 %, 35 % and 65 % were assumed. To generate this data, the control data was multiplied with a factor of 0.9, 0.65 or 0.35.
- The procedure started with the control group and the effect group (2 groups in total, no additional treatment groups without effects). Then treatment groups without effects were added. For each group constellation the test was conducted 500 times. The p-values for the effect group were saved and compared with each other at the end.

It was shown that the p-values from CPCAT for data sets with many treatment groups without effects were significantly higher than for data sets with fewer groups. The calculations showed that CPCAT is particularly sensitive to the number of treatment groups without effects.

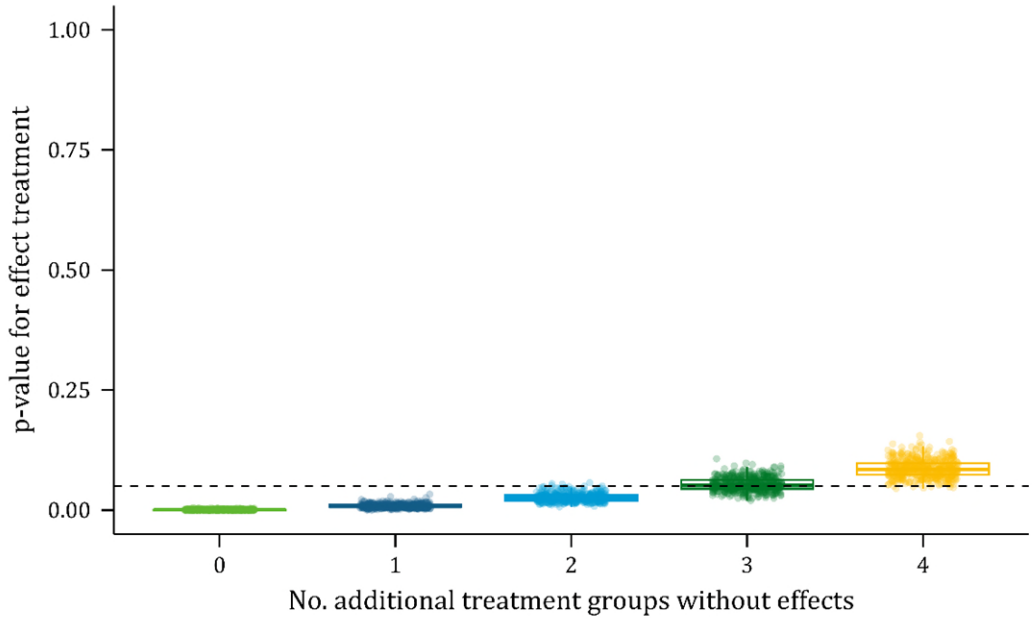
The following graphs illustrates how quickly significant differences between groups with effects and the control are shown to be non-significant when treatment groups without effects are gradually added to the data set.

Figure 16: P-values for a treatment with 10 % effect when evaluating additional treatment groups without effects using CPCAT.



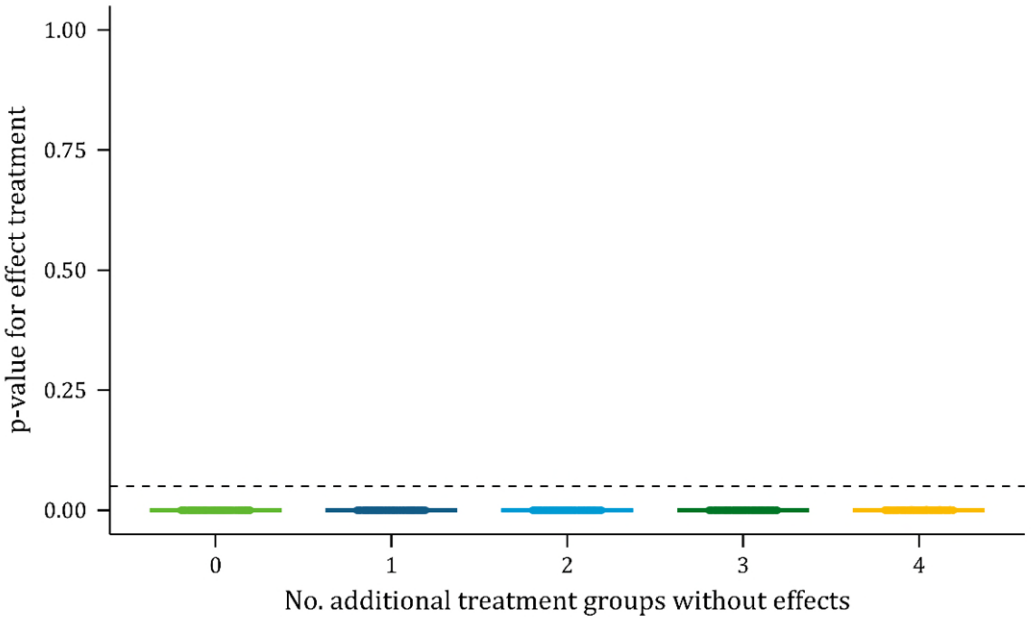
Source: own illustration, WSC Scientific GmbH.

Figure 17: P-values for a treatment with 35 % effect when evaluating additional treatment groups without effects using CPCAT.



Source: own illustration, WSC Scientific GmbH.

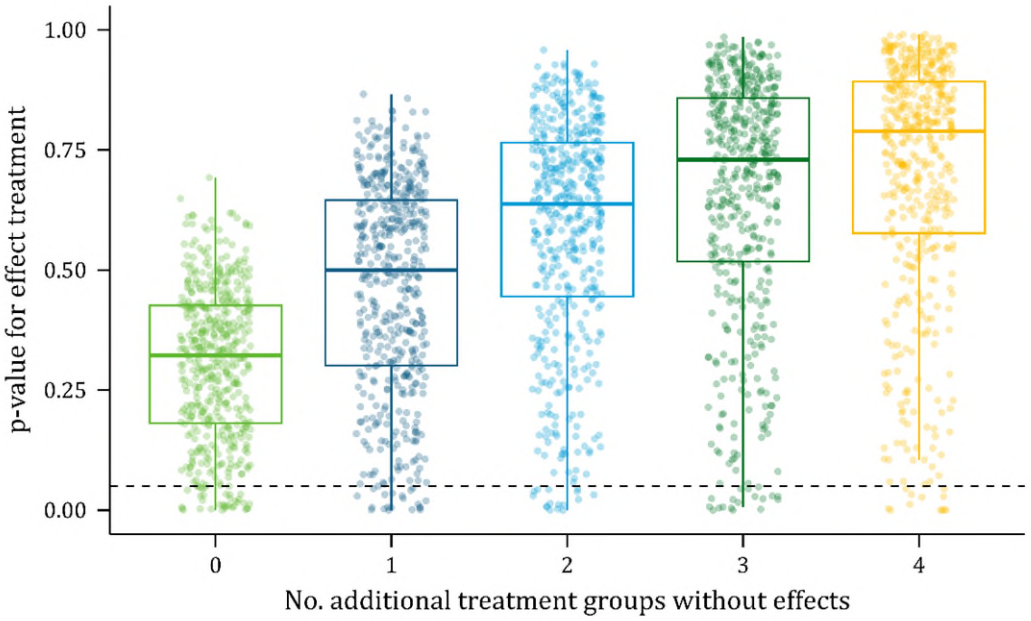
Figure 18: P-values for a treatment with 65 % effect when evaluating additional treatment groups without effects using CPCAT.



Source: own illustration, WSC Scientific GmbH.

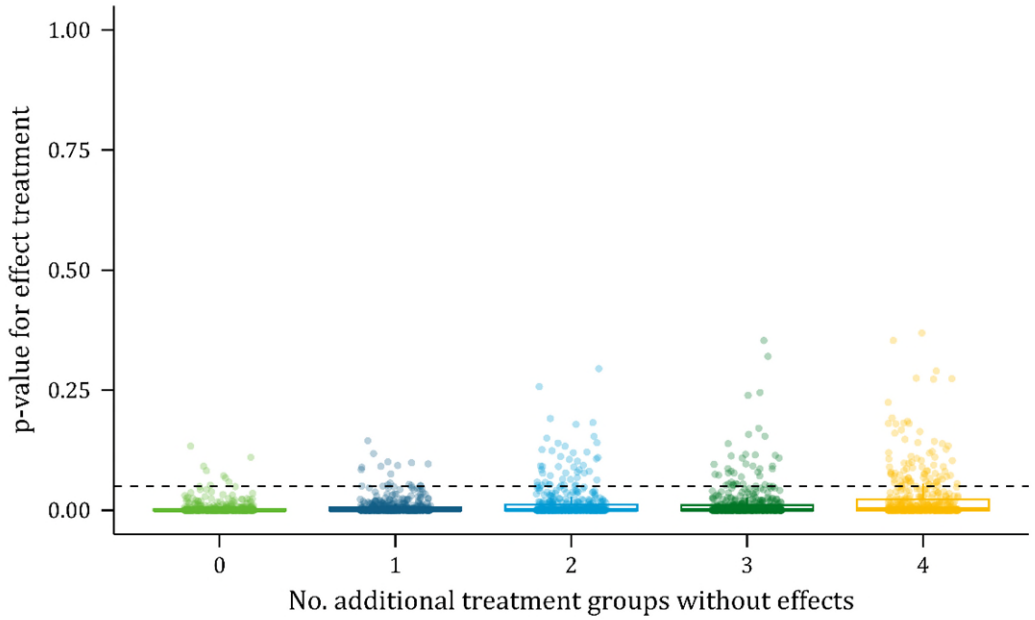
The same procedure was performed for the implemented Dunnett.GLM test. It was not shown to be particularly susceptible to this phenomenon and at the same time generally showed a higher sensitivity than CPCAT. It can also be seen, that there was a greater variance in the p-values from Dunnett.GLM compared to results from CPCAT. However, median p-values for Dunnet.GLM were consistently lower than the p-values from CPCAT.

Figure 19: P-values for a treatment with 10 % effect when evaluating additional treatment groups without effects using Dunnett.GLM.



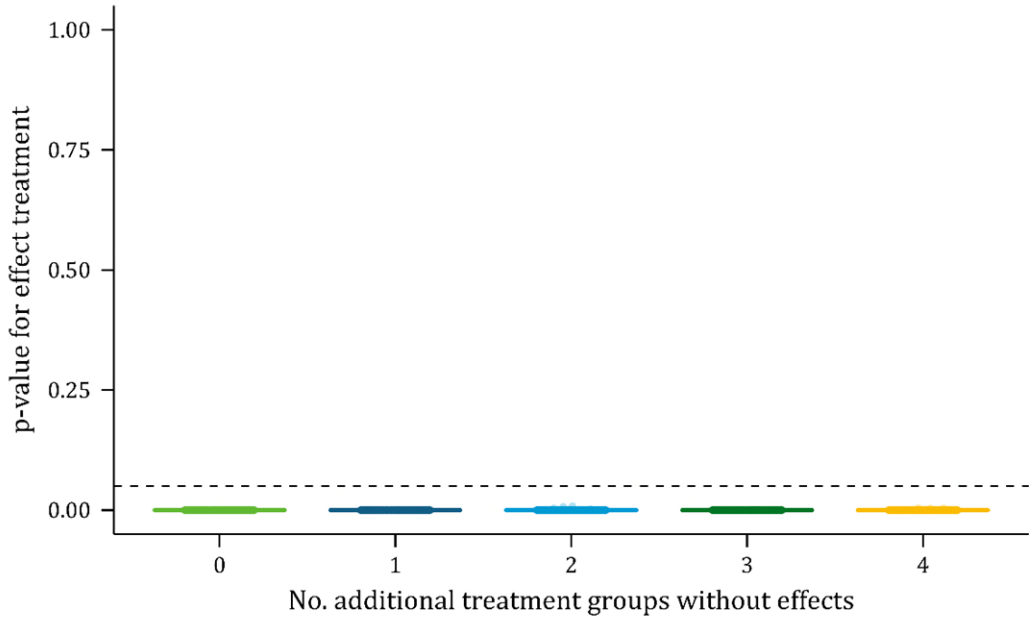
Source: own illustration, WSC Scientific GmbH.

Figure 20: P-values for a treatment with 35 % effect when evaluating additional treatment groups without effects using Dunnett.GLM.



Source: own illustration, WSC Scientific GmbH.

Figure 21: P-values for a treatment with 65 % effect when evaluating additional treatment groups without effects using Dunnett.GLM.



Source: own illustration, WSC Scientific GmbH.

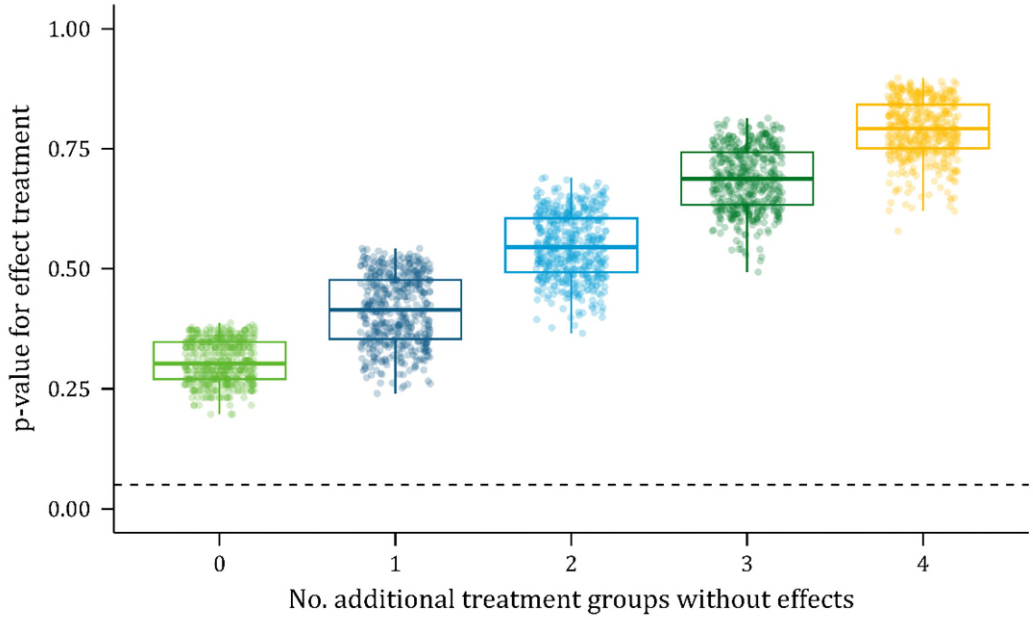
Sensitivity to added treatment groups without effects was also assessed for CPFISH. However, since binomial data are processed in CPFISH, the setup had to be adapted for the evaluation:

- Each group had a sample size of 100 (higher number of individuals needed for a finer resolution of p-values as binomial data is always discrete).

- For the control and treatment groups without effects one set of binomial-distributed data was sampled with a probability of 75 % ('success event' for 75 out of 100 individuals on average).
- For the treatment group with effects (e.g. the highest concentration) a reduction of 'success events' of 10 %, 15 %, 20 %, 35 % and 65 % were assumed (15 % and 20 % were added as there were already no differences in scenarios using 35 % and 65 %).
- The procedure started with the control group and the effect group (2 groups in total, no additional treatment groups without effects). Then treatment groups without effects were added. For each group constellation the test was conducted 500 times. The p-values for the effect group were saved and compared with each other at the end.

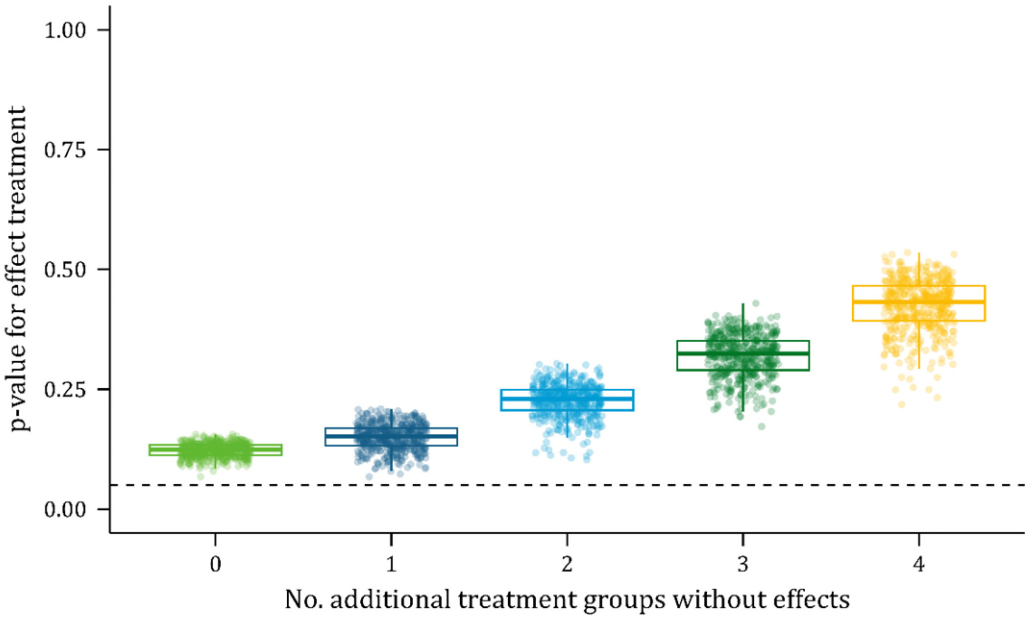
The results were similar to the results from CPCAT. It was shown that the p-values from CPFISH for data sets with many treatment groups without effects were significantly higher than for data sets with fewer groups. The calculations showed that CPFISH is particularly sensitive to the number of treatment groups without effects for effect sizes up to 20 %. For effect sizes of 35 % or higher, there was no difference observed in the scenario used (all p-values were zero). However, for other scenarios (e.g. using a different number of individuals introduced per group or using another level of background mortality), a difference might occur.

Figure 22: P-values for a treatment with 10 % effect when evaluating additional treatment groups without effects using CPFISH.



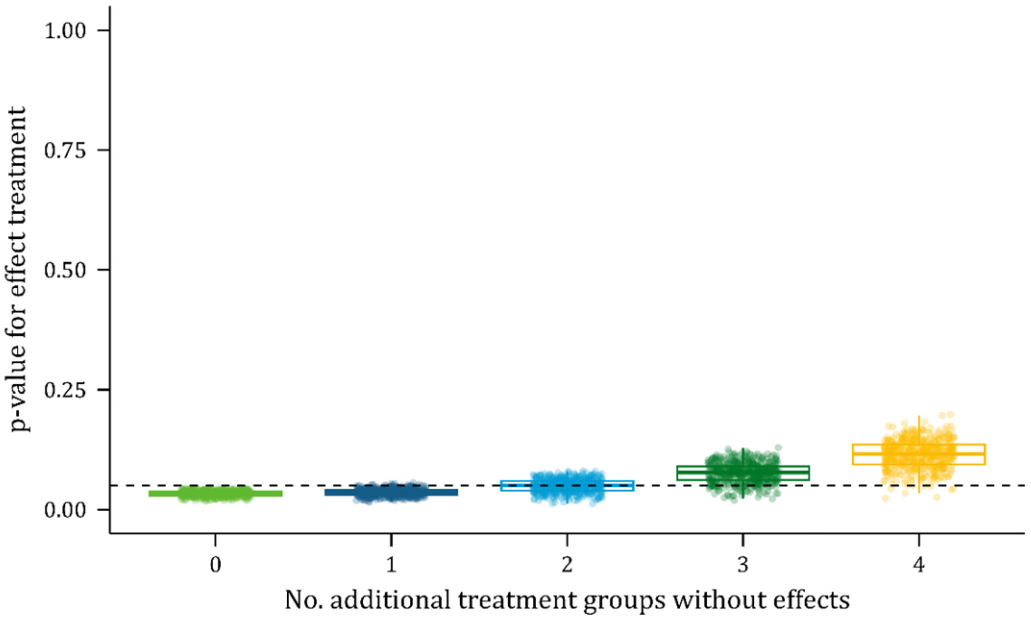
Source: own illustration, WSC Scientific GmbH.

Figure 23: P-values for a treatment with 15 % effect when evaluating additional treatment groups without effects using CPFISH.



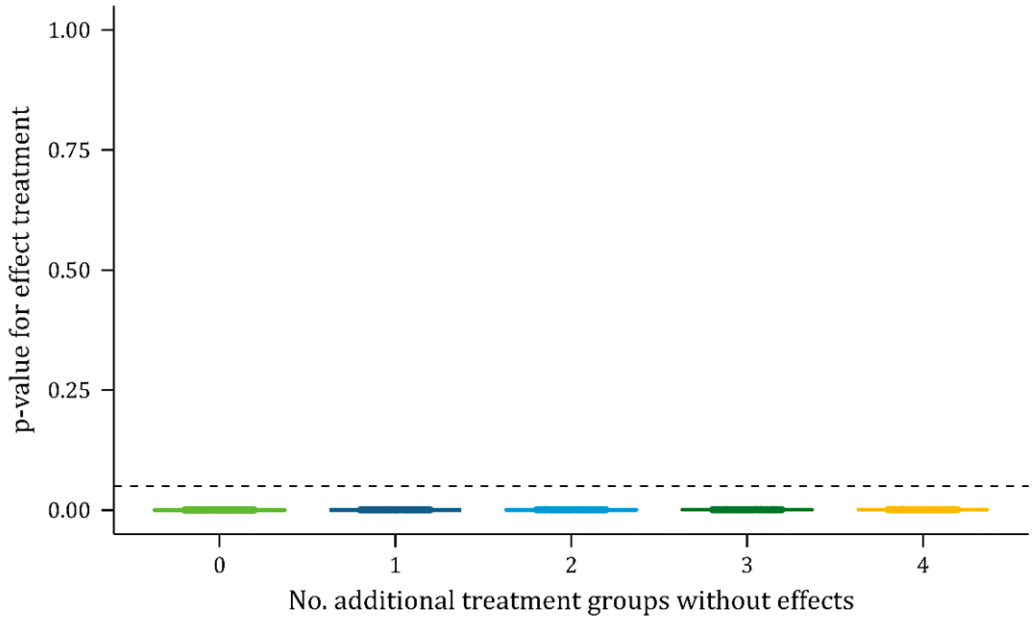
Source: own illustration, WSC Scientific GmbH.

Figure 24: P-values for a treatment with 20 % effect when evaluating additional treatment groups without effects using CPFISH.



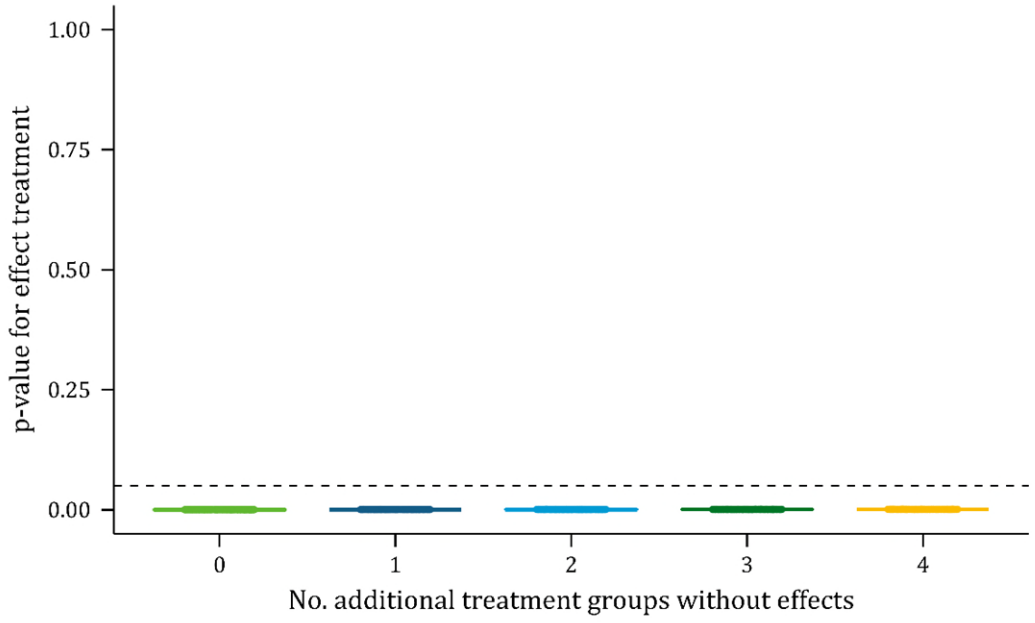
Source: own illustration, WSC Scientific GmbH.

Figure 25: Sensitivity of p-values for a treatment with 35 % effect when evaluating additional treatment groups without effects using CPFISH.



Source: own illustration, WSC Scientific GmbH.

Figure 26: Sensitivity of p-values for a treatment with 65 % effect when evaluating additional treatment groups without effects using CPFISH.



Source: own illustration, WSC Scientific GmbH.

4 Conclusions

For most of the tested scenarios, CPCAT and Dunnett.GLM showed only slight differences in power and performance. Both methods showed variations in power, if more than one treatment was tested. CPCAT performed comparably well to Dunnett.GLM, when only two groups (one control and one treatment) were considered, delivering reliable results due to its appropriate distributional assumptions.

For treatments with monotonous increasing effects, which are expected in ecotoxicology, both methods performed well. Please note that within this report no comparisons to other standard hypothesis tests (e.g. Jonkheere-Terpstra, U-test, etc.) were conducted.

CPCAT and CPFISH demonstrated a high sensitivity to the number of treatment groups without effects, making their power and p-values particularly susceptible to study designs with multiple non-effect groups. This sensitivity can result in non-significant findings for pronounced deviations in treatments compared to the control at high concentrations in studies with many treatment groups. Similar effects might be detected in test designs with fewer dose groups. Those mechanisms are well-known from statistical analyses with alternative multiple test methods (especially Bonferroni-Holm methods but also Williams and Dunnett test).

For overdispersed data, CPCAT shows higher probability for false positive detection of significant effects compared to the evaluation of the same data with Dunnett.GLM. This limitation may restrict the use of CPCAT in datasets with high variability. In future, a threshold for mild violations from the standard assumption (mean = variance) needs to be formulated and implemented as prerequisite for the use of CPCAT. In contrast, Dunnett.GLM adjusts for overdispersion and demonstrated more consistent power across the tested scenarios.

For scenarios with effects only in the highest treatment group, Dunnett.GLM showed slightly higher power compared to CPCAT. Moreover, the sensitivity of Dunnett.GLM is less affected by the number of non-effect groups, compared to CPCAT.

The finding of our evaluations emphasize, that CPCAT and CPFISH should be used with caution in study designs involving numerous non-effect groups or overdispersed data. Adjusting the design to minimize non-effect groups or variability may enhance the reliability of results, as CPCAT is still considered a robust choice for simpler test designs with fewer groups and less variability.

5 List of references

Bretz, F.; Hothorn, T.; Westfall, P. (2010): Multiple comparisons using R. 1st Edition, Chapman and Hall/CRC, New York

Chang, C.-H.; Pal, N.; Lin, J.-J. (2010): A Note on Comparing Several Poisson Means. In: Commun. Stat. Simul. Comput., 2010, 39(8), p. 1605 – 1627, https://doi.org/10.1080/03610918.2010.508860

Daniels, B.; Roß-Nickoll, M.; Jänsch, S.; Pieper, S.; Römbke, J.; Scholz-Starke, B.; Ottermanns, R. (2021): Application of the Closure Principle Computational Approach Test to Assess Ecotoxicological Field Studies: Comparative Analysis Using Earthworm Field Test Abundance Data. Environ Toxicol Chem, 2011, 40, p. 1750 – 1760, https://doi.org/10.1002/etc.5015

Duquesne, S.; Alalouni, U.; Gräff, T.; Frische, T.; Pieper, S.; Egerer, S.; Gergs, R.; Wogram, J. (2020): Better define beta-optimizing MDD (minimum detectable difference) when interpreting treatment-related effects of pesticides in semi-field and field studies. InEnviron Sci Pollut Res Int, 2020, 27(8), p. 8814 – 8821, https://doi.org/10.1007/s11356-020-07761-0

Hothorn, L.; Kluxen, F. (2020): Statistical analysis of no observed effect concentrations or levels in eco-toxicological assays with overdispersed count endpoints. In: bioRxiv, 2020, https://doi.org/10.1101/2020.01.15.907881

Lehmann, R.; Bachmann, J.; Maletzki, D.; Polleichtner, C.; Ratte, H.; Ratte, M. (2016): A new approach to overcome shortcomings with multiple testing of reproduction data in ecotoxicology. In: Stochastic Environmental Research and Risk Assessment, 2016, 30(3), p. 871 – 882, https://doi.org/10.1007/s00477-015-1079-4

Lehmann, R.; Bachmann, J.; Karaoglan, B.; Lacker, J; Lurman, G.; Polleichtner, C.; Ratte, H.T.; Ratte, M. (2018a): The CPCAT as a novel tool to overcome the shortcomings of NOEC/LOEC statistics in ecotoxicology: a simulation study to evaluate the statistical power. In: Environmental Science Europe, 2018, 30(55), https://doi.org/10.1186/s12302-018-0178-5

Lehmann, R.; Bachmann, J.; Karaoglan, B.; Lacker, J.; Polleichtner, C.; Ratte, H.; Ratte, M. (2018b): An alternative approach to overcome shortcomings with multiple testing of binary data in ecotoxicology. In: Stochastic Environmental Research and Risk Assessment, 2018, 32(1), p. 213 – 222, https://doi.org/10.1007/s00477-017-1392-1

Mair, M.; Kattwinkel, M.; Jakoby, O.; Hartig, F. (2020): The Minimum Detectable Difference (MDD) Concept for Establishing Trust in Nonsignificant Results: A Critical Review. In: Environmental Toxicology and Chemistry, 2020, 39(11), p. 2109 – 2123, https://doi.org/10.1002/etc.4847

Van der Hoeven, N. (2007): Calculation of the minimum significant difference at the NOEC using a non-parametric test. In: Ecotoxicology and Environmental Safety 70, p. 61 – 66, https://doi.org/10.1016/j.ecoenv.2007.06.010

A Appendix – Source code

A.1 Source code needed for the Closure Principle

```
# Function to generate hypotheses for the Closure Principle concept using 0/1
contrast matrices
CP.hypotheses = function(n, treatment.names = NULL) {
  combinations_list = list()

  # Generate all possible combinations of treatments
  for (subset_size in 1:n) {
    combinations_list[[subset_size]] = combn(1:n, subset_size)
  }

  hypothesis_matrix_list = list()

  # Create a hypothesis matrix for each treatment
  for (treatment in 1:n) {
    # Initialize a matrix with zeros
    hypothesis_matrix = matrix(0, ncol = n, nrow = (2^(n - 1)))

    # Set column names for the matrix
    if (!is.null(treatment.names) & length(treatment.names) == n) {
      colnames(hypothesis_matrix) = treatment.names
    } else {
      colnames(hypothesis_matrix) = paste("treatment", 1:n)
    }

    row_index = 1  # Start at the first row

    # Fill the matrix based on the combinations
    for (subset_size in 1:length(combinations_list)) {
      for (col in 1:ncol(combinations_list[[subset_size]])) {
        # Check if the current treatment is in the combination
        if (any(combinations_list[[subset_size]][, col] == treatment)) {
          # Mark the corresponding columns in the matrix
```

```
          hypothesis_matrix[row_index, combinations_list[[subset_size]][, col]] = 1
          row_index = row_index + 1  # Move to the next row
        }
      }
    }

    # Store the matrix in the list
    hypothesis_matrix_list[[treatment]] = hypothesis_matrix
  }

  # Name each hypothesis matrix according to its treatment
  names(hypothesis_matrix_list) = paste("H0: mu_0 = mu_", 1:n, sep = "")

  return(hypothesis_matrix_list)
}
```

A.2 Source code for CPCAT

```
# Helper function for CPCAT
CPCAT.Poisson.sub.test = function(dat,              # data to be evaluated
                  contrast,           # contrast matrix
                  bootstrap.runs = 10000) {    # number of bootstrap runs
  # Exclude the treatment groups that are not indicated by the contrast information
  datsheets = c(1, which(contrast == 1) + 1)    # Index to control and CONSIDERED
treatments
  dat2 = list()                                 # Control and considered treatment
data as a list

  # Populate dat2 with data from the specified treatments
  for (l in 1:length(datsheets)) {
    dat2[[l]] = dat[[datsheets[l]]]
  }

  # Initialize vectors for statistics
  musML = ns = xs = rep(0, length(dat2))
```

```
  # Calculate sample sizes, total abundance, and mean abundance for each group
  for (i in 1:length(musML)) {
    ns[i] = nrow(dat2[[i]])        # Sample size per group
    xs[i] = sum(dat2[[i]][, 1])    # Total abundance per group
    musML[i] = xs[i] / ns[i]       # Mean abundance per group
  }

  # Calculate 'total distance' etaML between control and all considered treatments
  etaML = sum((sqrt(musML[-1]) - sqrt(musML[1]))^2)
  n = sum(ns)                          # Total sample size over control and
considered treatments
  x = sum(xs)                          # Total abundance over control and considered
treatments
  mu0RML = x / n                       # Mean abundance over control and considered
treatments

  # Return a p-value of 1 if control and all (considered) treatments are zero
(i.e., no difference)
  if (mu0RML == 0) {
    return(1)
  }

  # Create artificial data (Poisson distributed)
  artificialdata = pseudomus = list()
  for (j in 1:length(dat2)) {
    artificialdata[[j]] = rpois(bootstrap.runs, ns[j] * mu0RML)  # Artificial total
abundance per group
    pseudomus[[j]] = artificialdata[[j]] / ns[j]    # Artificial mean abundance per
group
  }

  # Calculate pseudo etas for the artificial data
  pseudoetasML = rep(0, bootstrap.runs)
  for (l in 1:bootstrap.runs) {
    pseudomushelp = vector()
    for (i in 1:length(pseudomus)) {
      pseudomushelp[i] = pseudomus[[i]][l]
```

```
    }
    pseudoetasML[l] = sum((sqrt(pseudomushelp[-1]) - sqrt(pseudomushelp[1]))^2)
  }

  # Calculate p-value from the number of artificial datasets causing a higher
distance
  pvalue = length(which(pseudoetasML > etaML)) / bootstrap.runs

  return(pvalue)
}

# Helper function for CPCAT
CPCAT.Poisson.test = function(dat, contrastmatrix, bootstrap.runs = 10000) {
  # Re-structure the input data as a list object
  dat2 = list()
  treatments = levels(dat$Groups)

  # Organize data by treatment
  for (j in 1:length(treatments)) {
    index = which(dat$Groups == treatments[j])
    dat2[[j]] = dat[index, ]
  }

  # Initialize p-values vector
  pvalues = rep(0, nrow(contrastmatrix))

  # Perform Poisson test for each hypothesis (each row in contrastmatrix)
  for (j in 1:nrow(contrastmatrix)) {
    pvalues[j] = CPCAT.Poisson.sub.test(dat = dat2,
                      contrast = contrastmatrix[j, ],
                      bootstrap.runs = bootstrap.runs)
  }

  # For each treatment, provide the maximum p-value from all treatment-related
hypotheses
```

```
  return(list(contrastmatrix_pvalues = cbind(contrastmatrix, pvalues), maxpvalue =
max(pvalues)))
}

# CPCAT main function
CPCAT = function(groups,                    # group vector
         counts,                   # vector with count data
         control.name = NULL,             # character string with control group name
         bootstrap.runs = 10000,           # number of bootstrap runs
         hampel.threshold = 5,             # default threshold for Hampel identifier
(measure for over-/underdispersion)
         use.fixed.random.seed = TRUE,         # use fixed seed for reproducible
results
         get.contrasts.and.p.values = FALSE,     # get each row of the contrast
matrices evaluated
         show.output = T) {               # show/hide output

  # check if there is count data for each replicate (length of count and group
vectors) - groups[i] is one replicate
  if (length(groups) != length(counts)) {
    stop("Lengths of groups and counts don't match!")
  }
  # check format of input data
  if (!is.numeric(counts) | min(counts < 0)) {    #  | !all(counts ==
floor(counts))
    stop("Counts must be non-negative numeric values!")
  }

  # setup information to be stored
  info = data.frame(matrix(nrow = 0, ncol = 1))
  info = rbind(info, "Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’
1")

  # Re-structure the input to a data frame
  dat = data.frame(Counts = counts, Groups = groups)

  # Assign new order of levels if control.name was specified
```

```
  if (!is.null(control.name)) {
    if (!is.character(control.name)) {
      stop("Specified control must be provided as a character string!")
    }
    if (!is.element(control.name, unique(dat$Groups))) {
      stop("Specified control cannot be found!")
    }

    # Put desired control in the first place
    dat.temp.1 = dat[dat$Groups == control.name,]
    dat.temp.2 = dat[dat$Groups != control.name,]
    dat = rbind(dat.temp.1, dat.temp.2)
  }

  # Convert groups column to a factor, specifying the desired order of levels
  dat$Groups = factor(dat$Groups, levels = unique(dat$Groups))

  # Use treatments vector for convenience
  treatments = levels(dat$Groups)

  # Exit if not enough data left
  if (dim(na.omit(dat))[1] < 2) {
    stop("Too few valid data!")
  }
  if (dim(dat)[1] != dim(na.omit(dat))[1]) {
    info = rbind(info, paste0(dim(dat)[1] != dim(na.omit(dat))[1], " rows with NA
values were excluded!"))
  }
  dat = na.omit(dat)

  # Check for over- and under-dispersion using the Hampel identifier with a default
cut-off value of 5
  mean.dat = aggregate(dat$Counts, by=list(dat$Groups), mean)$x
  var.dat = aggregate(dat$Counts, by=list(dat$Groups), var)$x
  hampel.value = var.dat - mean.dat
```

```
  if (min(hampel.value) < -hampel.threshold) {
    info = rbind(info, paste0("There was under-dispersed data identified in
treatment(s) ",
                  paste0(paste0(treatments, " (HI: ", round(hampel.value,
digits=1), ")")[which(hampel.value < -hampel.threshold)], collapse = ", "),
                  ". HI = Hampel Identifier."))
  }
  if (max(hampel.value) > hampel.threshold) {
    info = rbind(info, paste0("There was over-dispersed data identified in
treatment(s) ",
                  paste0(paste0(treatments, " (HI: ", round(hampel.value,
digits=1), ")")[which(hampel.value > hampel.threshold)], collapse = ", "),
                  ". HI = Hampel Identifier."))
  }

  # All hypotheses to be tested
  n = length(levels(dat$Groups))
  allhypotheses = CP.hypotheses(n = n - 1, treatment.names = treatments)

  # Transform list to table data.frame
  allhypothesescompact = numeric()
  for (l in 1:length(allhypotheses)) {
    allhypothesescompact = rbind(allhypothesescompact, allhypotheses[[l]])
  }

  # Only unique rows are selected
  allhypothesescompact = unique(allhypothesescompact)

  results = list()
  # Flag all hypotheses which have already been tested by assigning a p-value,
  # else p-value = -9999
  flagpvalues = matrix(-9999, nrow = nrow(allhypothesescompact), ncol =
ncol(allhypothesescompact))
  pvalsCPCAT = rep(1, n - 1)

  # Fix seed for random numbers if desired (enables to reproduce results)
```

```
  if (use.fixed.random.seed) {
    set.seed(123)
  }

  for (j in 1:(n - 1)) {
    # Identify contrasts a p-value != -9999 has been assigned to
    # These contrasts must not be tested again
    contrasts = CP.hypotheses(n = n - 1, treatment.names = treatments)[[j]]
    matchingrows = numeric()
    for (i in 1:nrow(contrasts)) {
      matchingrows = c(matchingrows, which(apply(allhypothesescompact, 1,
identical, contrasts[i, ])))
    }
    alreadyflaggedindex = which(flagpvalues[matchingrows, j] != -9999)

    # Shorten contrasts to be tested by elimination of already tested contrasts
    if (length(alreadyflaggedindex) > 0) {
      contrasts = contrasts[-alreadyflaggedindex, ]
    }

    # In the last step the contrast matrix reduces to a vector
    # Make it a matrix consisting of nrow = 1
    if (is.matrix(contrasts) == FALSE) {
      contrasts = matrix(contrasts, nrow = 1)
    }
    notflaggedindex = which(flagpvalues[matchingrows, j] == -9999)
    # Flag p-values which are still -9999
    # After CPCAT corresponding p-values will be != -9999
    tobeflagged = matchingrows[notflaggedindex]

    results[[j]] = CPCAT.Poisson.test(dat = dat,
                      contrastmatrix = contrasts,
                      bootstrap.runs = bootstrap.runs)[[1]]

    if (j == 1) {
```

```
      contrasts.and.p.values = results[[j]]
    } else {
      contrasts.and.p.values = rbind(contrasts.and.p.values, results[[j]])
    }

    # Write obtained p-values into column j of flagpvalues and find max p-value
    pvalshelp = results[[j]][, ncol(results[[j]])]
    flagpvalues[tobeflagged, j] = pvalshelp

    # Put together new p-values of reduced contrast matrix and relevant p-values in
flagpvalues[, j]
    if (j > 1) {  # In step j = 1 all flagpvalues equal -9999
      pvalshelp2 = c(pvalshelp, flagpvalues[matchingrows[-notflaggedindex], j])
    } else {
      pvalshelp2 = pvalshelp
    }
    pvalsCPCAT[j] = max(pvalshelp2)

    # Copy p-values obtained so far to the next column of flagpvalues
    if (j < (n - 1)) {
      flagpvalues[, j + 1] = flagpvalues[, j]
    }
  }

  # Assign significance levels based on p-values
  significances = rep(NA, n - 1)
  for (j in 1:(n - 1)) {
    if (pvalsCPCAT[j] < 0.05) {
      if (pvalsCPCAT[j] < 0.01) {
        if (pvalsCPCAT[j] < 0.001) {
          significances[j] = "***"
        } else {
          significances[j] = "**"
        }
      } else {
```

```
        significances[j] = "*"
      }
    } else {
      significances[j] = "."
    }
  }

  # Get NOEC and LOEC
  NOEC = treatments[1]
  LOEC = treatments[2]
  for (j in 1:(n - 1)) {
    if (pvalsCPCAT[j] < 0.05) {
      break
    }
    NOEC = treatments[j]
    if (j == (n - 1)) {
      LOEC = NA
      break
    }
    LOEC = treatments[j+1]
  }
  info = rbind(info, paste0("NOEC: ", NOEC, ", LOEC: ", ifelse(is.na(LOEC),
"outside tested dose/concentration", LOEC),
               ". Assuming that any effects are adverse. Otherwise, NOEC and LOEC
should be reconsidered."))

  # Compile results into a data.frame
  results = data.frame(Hypothesis = paste0("H0: ", treatments[1], " <-> ",
treatments[2:n]), p.values = pvalsCPCAT, Signif. = significances)

  # Set header for information object
  colnames(info) = "Information and warnings:"

  # Show output if desired
  if (show.output) {
    if (get.contrasts.and.p.values) {
```

```
      print(structure(list(Contrasts=data.frame(contrasts.and.p.values),
Results=results, Info=info)), row.names = F, quote = F, right = F)
    } else {
      print(structure(list(Results=results, Info=info)), row.names = F, quote = F,
right = F)
    }
  }

  # Provide output as object even if not shown
  if (get.contrasts.and.p.values) {
    invisible(structure(list(Contrasts=data.frame(contrasts.and.p.values),
Results=results, Info=info)))
  } else {
    invisible(structure(list(Results=results, Info=info)))
  }
}
```

A.3 Source code for Dunnett.GLM

```
# Dunnett GLM
Dunnett.GLM = function(  groups,                    # group vector
            counts,                    # vector with count data
            control.name = NULL,            # character string with control group
name
            zero.treatment.action = "identity.link",  # method for dealing with
treatments only containing zeros (alternative: "log(x+1)")
            show.output = T){              # show/hide output

  # do some error handling
  if (length(groups) != length(counts)) {
    stop("Lengths of groups and counts don't match!")
  }

  if (!is.numeric(counts) | min(counts < 0)) {   # | !all(counts == floor(counts))
    stop("Counts must be non-negative numeric values!")
  }
```

```
  if (zero.treatment.action != "identity.link" & zero.treatment.action !=
"log(x+1)") {
    stop("Parameter zero.treatment.action must be either \"identity.link\" or
\"log(x+1)\"!")
  }

  # setup information to be stored
  info = data.frame(matrix(nrow = 0, ncol = 1))

  # Re-structure the input to a data frame
  dat = data.frame(Counts = counts, Groups = groups)

  # Assign new order of levels if control.name was specified
  if (!is.null(control.name)) {
    if (!is.character(control.name)) {
      stop("Specified control must be provided as a character string!")
    }
    if (!is.element(control.name, unique(dat$Groups))) {
      stop("Specified control cannot be found!")
    }

    # Put desired control in the first place
    dat.temp.1 = dat[dat$Groups == control.name,]
    dat.temp.2 = dat[dat$Groups != control.name,]
    dat = rbind(dat.temp.1, dat.temp.2)
  }

  # Convert groups column to a factor, specifying the desired order of levels
  dat$Groups = factor(dat$Groups, levels = unique(dat$Groups))

  # Use treatments vector for convenience
  treatments = levels(dat$Groups)

  # Exit if not enough data left
  if (dim(na.omit(dat))[1] < 2) {
```

```
    stop("Too few valid data!")
  }
  if (dim(dat)[1] != dim(na.omit(dat))[1]) {
    info = rbind(info, paste0(dim(dat)[1] != dim(na.omit(dat))[1], " rows with NA
values were excluded!"))
  }
  dat = na.omit(dat)



  agg = aggregate(dat$Counts, by=list(dat$Groups), mean)


  if (min(agg$x) > 0) {
    # Dunnett GLM with quasi-Poisson link-function
    mod = glm(Counts~Groups, data=dat, family=quasipoisson(link = "log"))
  } else {
    info = rbind(info, paste0("A treatment contained only zeros, hence, the
zero.treatment.action \"", zero.treatment.action, "\" was applied."))


    # Use either the identity link or a data transformation if one treatment mean
is zero (standard log link would fail)
    if (zero.treatment.action == "identity.link") {
      # use another link function (link="identity", i.e. E(Y) = b0 + b1 * X)
      mod = try(glm(Counts~Groups, data=dat, family=quasipoisson(link =
"identity")), silent = T)
      if ("try-error" %in% class(mod)){
        stop("Error in GLM calculation. Consider to change argument
zero.treatment.action to \"log(x+1)\".")
      }
    } else {
      # modify data to always get positive estimates using log link (link="log",
i.e. log(E(Y)) = b0 + b1 * X)
      dat$Counts = dat$Counts + 1
      mod = try(glm(dat$Counts~Groups, data=dat, family=quasipoisson(link =
"log")), silent = T)
      if ("try-error" %in% class(mod)){
        stop("Error in GLM calculation. Consider to change argument
zero.treatment.action to \"identity.link\".")
```

```
      }
    }
  }

  results = summary(glht(mod, linfct = mcp(Groups = "Dunnett"),
alternative="two.sided"))

  # Set header for information object
  colnames(info) = "Information and warnings:"

  # Provide output
  if (show.output) {
    print(structure(list('Results' = results, Info=info)), row.names = F, quote =
F, right = F)
  } else {
    invisible(structure(list('Results' = results, Info=info)))
  }
}
```

A.4 Source code for CPFISH

```
# CPFISH function for testing hypotheses using Fisher's exact test
CPFISH = function(contingency.table,          # contingency.table is a matrix with
observed data (e.g. survival counts, survival must be in first row)
          control.name = NULL,          # character string with control group name
          simulate.p.value = TRUE,        # use simulated p-values or not
          use.fixed.random.seed = TRUE,      # use fixed seed for reproducible
results
          show.output = T){            # show/hide output

  # setup information to be stored
  info = data.frame(matrix(nrow = 0, ncol = 1))
  info = rbind(info, "Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '
1")

  # Assign new order of levels if control.name was specified
  if (!is.null(control.name)) {
```

```
    if (!is.character(control.name)) {
      stop("Specified control must be provided as a character string!")
    }
    if (!is.element(control.name, unique(colnames(contingency.table)))) {
      stop("Specified control cannot be found!")
    }

    # Put desired control in the first place
    dat.temp.1 = data.frame(contingency.table[, which(colnames(contingency.table)
== control.name)])
    colnames(dat.temp.1) = control.name
    dat.temp.2 = contingency.table[, which(colnames(contingency.table) !=
control.name)]
    contingency.table = cbind(dat.temp.1, dat.temp.2)
  }

  treatment_names = colnames(contingency.table)
  num_treatments = ncol(contingency.table) - 1
  if (is.null(treatment_names)) {
    treatment_names = as.character(1:(num_treatments+1))
  }

  # Vector to store p-values for each treatment
  pvalues = rep(1, num_treatments)

  # Generate all possible hypotheses
  all_hypotheses = CP.hypotheses(n = length(treatment_names) - 1, treatment.names =
treatment_names)
  compact_hypotheses = do.call(rbind, all_hypotheses)
  compact_hypotheses = unique(compact_hypotheses)

  # Matrix to track tested hypotheses and their p-values
  pvalue_flags = matrix(-9999, nrow = nrow(compact_hypotheses), ncol =
ncol(compact_hypotheses))

  # Fix seed for random numbers if desired (enables to reproduce results)
```

```
  if (use.fixed.random.seed) {
    set.seed(123)
  }

  for (j in 1:num_treatments) {
    # Retrieve contrast matrix for the current treatment hypothesis
    contrasts = CP.hypotheses(n = num_treatments, treatment.names =
treatment_names)[[j]]
    matching_rows = numeric()
    for (i in 1:nrow(contrasts)) {
      matching_rows = c(matching_rows, which(apply(compact_hypotheses, 1,
identical, contrasts[i, ])))
    }
    already_tested = which(pvalue_flags[matching_rows, j] != -9999)

    # Remove already tested contrasts from the matrix to be tested
    if (length(already_tested) > 0) {
      contrasts = contrasts[-already_tested, ]
    }

    # Ensure contrasts matrix is still a matrix after subsetting
    if (!is.matrix(contrasts)) {
      contrasts = matrix(contrasts, nrow = 1)
    }

    not_tested = which(pvalue_flags[matching_rows, j] == -9999)
    to_be_tested = matching_rows[not_tested]

    # Calculate p-values for the current contrasts
    pvals = rep(0, nrow(contrasts))
    for (l in 1:nrow(contrasts)) {
      test_data = contingency.table[, c(1, (which(contrasts[l, ] == 1) + 1))]
      if (all(rowSums(test_data) != c(0, 0))) {
        pvals[l] = fisher.test(test_data, alternative = "two.sided",
simulate.p.value = simulate.p.value)[[1]]
      } else {
```

```
      pvals[l] = 1
    }
  }

  # Update p-value flags and track maximum p-value
  pvalue_flags[to_be_tested, j] = pvals
  if (j > 1) {
    pvals_combined = c(pvals, pvalue_flags[matching_rows[-not_tested], j])
  } else {
    pvals_combined = pvals
  }
  pvalues[j] = max(pvals_combined)

  # Propagate the p-values to the next step
  if (j < (length(treatment_names) - 1)) {
    pvalue_flags[, j + 1] = pvalue_flags[, j]
  }
}

n = length(treatment_names)
significances = rep(NA, n - 1)

# Assign significance levels based on p-values
for (j in 1:(n - 1)) {
  if (pvalues[j] < 0.05) {
    if (pvalues[j] < 0.01) {
      if (pvalues[j] < 0.001) {
        significances[j] = "***"
      } else {
        significances[j] = "**"
      }
    } else {
      significances[j] = "*"
    }
  } else {
```

```
      significances[j] = "."
    }
  }

  # Get NOEC and LOEC
  NOEC = treatment_names[1]
  LOEC = treatment_names[2]
  for (j in 1:(n - 1)) {
    if (pvalues[j] < 0.05) {
      break
    }
    NOEC = treatment_names[j+1]
    if (j == (n - 1)) {
      LOEC = NA
      break
    }
    LOEC = treatment_names[j+2]
  }
  info = rbind(info, paste0("NOEC: ", NOEC, ", LOEC: ", ifelse(is.na(LOEC),
"outside tested dose/concentration", LOEC),
                ". Assuming that any effects are adverse. Otherwise, NOEC and LOEC
should be reconsidered."))

  results = data.frame(Treatment = treatment_names[-1], p.values = pvalues, Signif.
= significances)

  # Add information about the use of simulated p-values
  if (simulate.p.value) {
    info = rbind(info, "Simulated p-values used.")
  }

  # Set header for information object
  colnames(info) = "Information and warnings:"

  # Provide output
  if (show.output) {
```

```
    print(structure(list('Results' = results, Info=info)), row.names = F, quote =
F, right = F)
  } else {
    invisible(structure(list('Results' = results, Info=info)))
  }
}
```

B Appendix – Data from the literature

B.1 Count data from Lehmann et al. (2016)

Table 8: Example count data from Lehmann et al. (2016) used for validation.

Dataset	Groups	Counts
1	Control	23
1	Control	22
1	Control	24
1	Control	23
1	Control	21
1	Control	21
1	1.06	18
1	1.06	22
1	1.06	22
1	1.59	23
1	1.59	23
1	1.59	21
1	2.38	20
1	2.38	19
1	2.38	21
1	3.53	10
1	3.53	8
1	3.53	9
1	5.29	6
1	5.29	4
1	5.29	6
1	7.93	2
1	7.93	2
1	7.93	0
2	Control	67

Dataset	Groups	Counts
2	Control	59
2	Control	64
2	Control	71
2	Control	63
2	Control	58
2	1.06	47
2	1.06	48
2	1.06	58
2	1.59	55
2	1.59	66
2	1.59	56
2	2.38	39
2	2.38	48
2	2.38	50
2	3.53	12
2	3.53	9
2	3.53	10
2	5.29	6
2	5.29	4
2	5.29	6
2	7.93	2
2	7.93	2
2	7.93	0
3	Control	154
3	Control	130
3	Control	134
3	Control	155
3	Control	142

Dataset	Groups	Counts
3	Control	139
3	1.06	115
3	1.06	104
3	1.06	116
3	1.59	118
3	1.59	120
3	1.59	120
3	2.38	71
3	2.38	76
3	2.38	83
3	3.53	12
3	3.53	10
3	3.53	10
3	5.29	6
3	5.29	4
3	5.29	6
3	7.93	2
3	7.93	2
3	7.93	0
4	Control	139
4	Control	131
4	Control	138
4	Control	116
4	0.2	94
4	0.2	140
4	0.2	100
4	0.2	89
4	1	147

Dataset	Groups	Counts
4	1	109
4	1	98
4	1	139
4	5	84
4	5	105
4	5	79
4	5	73
4	25	33
4	25	42
4	25	39
4	25	46
6	Control	12
6	Control	14
6	Control	15
6	Control	14
6	Control	13
6	Control	16
6	T1	10
6	T1	9
6	T1	11
6	T1	10
6	T1	8
6	T1	8
6	T2	9
6	T2	8
6	T2	8
6	T2	9
6	T2	7

Dataset	Groups	Counts
6	T2	8
6	T3	16
6	T3	18
6	T3	15
6	T3	19
6	T3	17
6	T3	20
6	T4	31
6	T4	35
6	T4	33
6	T4	39
6	T4	41
6	T4	42
6	T5	61
6	T5	53
6	T5	64
6	T5	67
6	T5	59
6	T5	65
7	Control	25
7	Control	22
7	Control	24
7	T1	27
7	T1	29
7	T1	30
7	T2	19
7	T2	18
7	T2	19

Dataset	Groups	Counts
7	T3	34
7	T3	35
7	T3	30
7	T4	15
7	T4	12
7	T4	13
7	T5	13
7	T5	10
7	T5	9

B.2 Quantal data from Lehmann et al. (2018b)

Table 9: Example data 1 from Lehmann et al. (2018b) used for validation.

Replicate	Survived / Dead	Control	T 1.5	T 3	T 6.25	T 12.5	T 25	T 50	T 100
1	Survived	14	14	15	8	13	11	15	7
1	Dead	1	1	0	7	2	4	0	8
2	Survived	13	13	13	14	9	11	9	6
2	Dead	2	2	2	1	6	4	6	9
3	Survived	15	12	10	14	10	11	6	6
3	Dead	0	3	5	1	5	4	9	9
4	Survived	15	15	12	10	14	15	9	9
4	Dead	0	0	3	5	1	0	6	6
5	Survived	14	-	-	-	-	-	-	-
5	Dead	1	-	-	-	-	-	-	-

Replicate	Survived / Dead	Control	T 1.5	T 3	T 6.25	T 12.5	T 25	T 50	T 100
Combined	Survived	71	54	50	46	46	48	39	28
Combined	Dead	4	6	10	14	14	12	21	32

Table 10: Example data 2 from Lehmann et al. (2018b) used for validation.

Replicate	Dead / Survived	Control	T 1	T 2	T 3	T 4	T 5	T 6	T 7
1	Dead	1	1	1	1	1	1	1	1
1	Survived	0	0	0	0	0	0	0	0
2	Dead	0	1	1	1	1	1	1	1
2	Survived	1	0	0	0	0	0	0	0
3	Dead	0	0	0	1	1	1	1	1
3	Survived	1	1	1	0	0	0	0	0
4	Dead	0	0	0	1	1	1	1	1
4	Survived	1	1	1	0	0	0	0	0
5	Dead	0	0	0	1	1	1	1	1
5	Survived	1	1	1	0	0	0	0	0
6	Dead	0	0	0	1	1	1	1	1
6	Survived	1	1	1	0	0	0	0	0
7	Dead	0	0	0	1	1	1	1	1
7	Survived	1	1	1	0	0	0	0	0
8	Dead	0	0	0	0	1	1	1	1
8	Survived	1	1	1	1	0	0	0	0

Replicate	Dead / Survived	Control	T 1	T 2	T 3	T 4	T 5	T 6	T 7
9	Dead	0	0	0	0	1	1	1	1
9	Survived	1	1	1	1	0	0	0	0
10	Dead	0	0	0	0	1	1	1	1
10	Survived	1	1	1	1	0	0	0	0
11	Dead	0	0	0	0	1	1	1	1
11	Survived	1	1	1	1	0	0	0	0
12	Dead	0	0	0	0	0	1	1	1
12	Survived	1	1	1	1	1	0	0	0
13	Dead	0	0	0	0	0	0	1	1
13	Survived	1	1	1	1	1	1	0	0
14	Dead	0	0	0	0	0	0	1	1
14	Survived	1	1	1	1	1	1	0	0
15	Dead	0	0	0	0	0	0	1	1
15	Survived	1	1	1	1	1	1	0	0
16	Dead	0	0	0	0	0	0	1	1
16	Survived	1	1	1	1	1	1	0	0
17	Dead	0	0	0	0	0	0	1	1
17	Survived	1	1	1	1	1	1	0	0
18	Dead	0	0	0	0	0	0	1	1
18	Survived	1	1	1	1	1	1	0	0
19	Dead	0	0	0	0	0	0	1	0

Replicate	Dead / Survived	Control	T 1	T 2	T 3	T 4	T 5	T 6	T 7
19	Survived	1	1	1	1	1	1	0	1
20	Dead	0	0	0	0	0	0	1	0
20	Survived	1	1	1	1	1	1	0	1
21	Dead	0	-	-	-	-	-	-	-
21	Survived	1	-	-	-	-	-	-	-
22	Dead	0	-	-	-	-	-	-	-
22	Survived	1	-	-	-	-	-	-	-
23	Dead	0	-	-	-	-	-	-	-
23	Survived	1	-	-	-	-	-	-	-
24	Dead	0	-	-	-	-	-	-	-
24	Survived	1	-	-	-	-	-	-	-
Combined	Dead	1	2	2	7	11	12	20	18
Combined	Survived	23	18	18	13	9	8	0	2

B.3 Count data from Hothorn and Kluxen (2020)

In the following table, data from a daphnia dataset (data taken from Hothorn and Kluxen, 2020; see table below) was used as input data for CPCAT to illustrate the format of the output. It was also used for verification of the Dunnett.GLM implementation.

Table 11: Input data for CPCAT example output.

Concentration	Young_daphnia
0	27
0	30
0	29
0	31
0	16
0	15
0	18

Concentration	Young_daphnia
0	17
0	14
0	27
1.56	32
1.56	35
1.56	32
1.56	26
1.56	18
1.56	29
1.56	27
1.56	16
1.56	35
1.56	13
3.12	39
3.12	30
3.12	33
3.12	33
3.12	36
3.12	33
3.12	33
3.12	27
3.12	38
3.12	44
6.25	27
6.25	34
6.25	36
6.25	34
6.25	31
6.25	27
6.25	33
6.25	21
6.25	33
6.25	31
12.5	10
12.5	13
12.5	7
12.5	7
12.5	7
12.5	10
12.5	10
12.5	16
12.5	12
12.5	2
25	0
25	0
25	0
25	0
25	0
25	0
25	0
25	0

Concentration	Young_daphnia
25	0
25	0

C Appendix – Example for CPCAT output format

C.1 Example for CPCAT input and output

In the following example, data from a daphnia dataset (data from Hothorn and Kluxen, 2020; see Table 11) was used as input data for CPCAT to illustrate the format of the output.

The CPCAT function call looks as follows (default setting were not changed):

```
CPCAT(groups=Concentration, counts=Young_daphnia)
```

The result is as follows:

```
$Results

 Hypothesis      p.values Signif.

 H0: 0 <-> 1.56 0.0748    .

 H0: 0 <-> 3.12 0.0001    ***

 H0: 0 <-> 6.25 0.0020    **

 H0: 0 <-> 12.5 0.0000    ***

 H0: 0 <-> 25   0.0000    ***


$Info

 Information and warnings:

 Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

 There was under-dispersed data identified in treatment(s) 3.12 (HI: -
11.2), 6.25 (HI: -10.5). HI = Hampel Identifier.

 There was over-dispersed data identified in treatment(s) 0 (HI: 25.6),
1.56 (HI: 37.7), 12.5 (HI: 5.8). HI = Hampel Identifier.

 NOEC: 0, LOEC: 1.56. Assuming that any effects are adverse. Otherwise,
NOEC and LOEC should be reconsidered.
```