



UNIVERSITÉ DE NANTES
FACULTÉ DES SCIENCES
ET DES TECHNIQUES



UNIVERSITÉ DE NANTES
FACULTÉ DES SCIENCES
ET DES TECHNIQUES

UNIVERSITÉ DE NANTES
UFR Mathématiques
Master 2 Mathématiques appliquées, Ingénierie Statistique

RAPPORT DE STAGE

Effectué à l'Agence nationale de sécurité sanitaire de
l'alimentation, de l'environnement et du travail (ANSES)

Développement d'un outil d'aide à la décision
pour le choix des tests statistiques appropriés aux études utilisées
en évaluation de risque réglementaire
volet environnement/écotoxicologie



Boutrin Hervé

Maître de Stage : Dr.Emilie Perrat (Evaluatrice Scientifique et technique ; **Tutrice**) &
Dr.Patrice Carpentier (Chef de projet veille et développement scientifiques ; Co-tuteur)
Tuteur académique : Dr.Aymeric Stamm (Ingénieurs de Recherche)

Année universitaire : 2023 / 2024
Stage effectué du 3 Avril 2024 au 30 Septembre 2024

Résumé

Ce rapport présente le développement d'un outil d'aide à la décision en ligne pour le choix des tests statistiques appropriés aux études utilisées en évaluation de risque réglementaire en écotoxicologie. L'objectif est de simplifier et centraliser les informations nécessaires à l'application des tests statistiques, actualisées avec les méthodes récentes, et permettant d'obtenir un indicateur de fiabilité des tests.

Pour cela une analyse détaillée des données d'écotoxicologie comportant des réplicats soumis à plusieurs concentrations de produits chimiques sera effectué. Et nous utilisons une gamme de méthodes statistiques, y compris l'ajustement de modèles dose-réponse, des tests de signification, et des techniques avancées telles que les méthodes bayésiennes, bootstrap et permutation, pour évaluer les effets toxiques et estimer les paramètres clés comme l' EC_{50} et l' EC_{10} .

Ce projet s'inscrit dans un contexte de réglementation stricte et de mise à jour des pratiques statistiques conformément aux directives de l'OCDE et de la réglementation européenne.

Remerciements

Je tiens à remercier toutes les personnes qui ont contribué au bon déroulement de mon stage et m'ont aidé lors de la rédaction de ce rapport.

Tout d'abord, j'adresse mes remerciements au Dr Vanessa Mazerolles, cheffe de l'unité Évaluation, Écotoxicologie, Environnement des Intrants du Végétal (U3EiV), pour m'avoir permis d'effectuer mon stage dans son unité, ainsi qu'aux Drs Émilie Perrat et Patrice Carpentier pour m'avoir confié ce projet passionnant, intéressant et enrichissant.

Je remercie également Mme Perrat pour sa supervision, sa confiance, sa patience et le partage de ses connaissances biologiques.

Enfin je remercie toute l'équipe de l'unité U3EiV pour leur accueil, leur esprit d'équipe, leur bienveillance et le partage de leurs connaissances. Enfin, je tiens à remercier toutes les personnes qui m'ont conseillée et relu lors de la rédaction de ce rapport de stage, en particulier les Drs Émilie Perrat et Patrice Carpentier.

Sommaire

1	Table des illustrations	6
2	Table des abréviations	7
3	Glossaire	8
4	Introduction	9
5	Méthodologie	11
5.1	Description des données des essais d'écotoxicité	11
5.1.1	Réplicats et méthode de randomisation	12
5.1.2	Exposition des organismes	13
5.2	Préparation des données pour l'analyse	13
5.2.1	Inspection Visuelle et Nettoyage des Données	13
5.2.2	Vérification de l'Effet Solvant	14
5.2.3	Contrôle des Hypothèses et Transformations des Données	14
5.3	Application de Modèles Statistiques Appropriés (illustration cf figure 5.1) . .	15
5.4	Formes Mathématiques des Modèles Statistiques Utilisés dans l'application .	16
5.4.1	Modèle de Weibull	16
5.4.2	Modèle log-logistique	16
5.4.3	Modèle logit	16
5.4.4	Modèle probit	16
5.4.5	Modèle de Hill	17
5.4.6	Modèle de Gumbel	17
5.4.7	Modèle de Poisson	17
5.4.8	Modèle binomial négatif	17
5.4.9	Modèle exponentiel (Slob 2002 [3])	17
5.5	Sélection de modèle	18
5.5.1	Vérification des Modèles	24
5.6	Détermination des valeurs de toxicité	25
5.7	Analyse de la significativité des différences observées dans les ECx par la méthode de permutation [23]	26
5.8	Intervalle de confiance des ECx par Bootstrap	26
5.9	Limites et contraintes liées aux données	27
6	Résultats et discussion	28
6.1	Codage et Transformation des Données :	28
6.2	Préparation des données pour l'analyse	29
6.2.1	Visualisation	29
6.2.2	Gestion des outliers	30
6.2.3	Agrégation des Données :	32
6.3	Analyse des données	34
6.3.1	Vérification de la normalité et de l'homoscédasticité des données : . .	34
6.3.2	Analyse descriptive	37
6.4	Détermination des LOEC/D/L et NOEC/D/L	38
7	Calcul de Puissance	39

8	Modélisation Dose-Réponse	41
8.1	Comparaison des Modèles	43
9	Estimation des ECx ou LCx	47
9.0.1	Test de Permutation et Intervalles de Confiance Bootstrap	48
10	Conclusion & Perspectives	50
A	Annexe A : Inventaire des tests	51
B	Annexe :Application interface	52
C	Annexe :Application architecture	54
D	Distributions de Probabilité	201

1 Table des illustrations

Figures

4.1	ANSES	9
5.1	Illustration NOEC, LOEC, ECx sur courbe dose réponse [21]	28
6.1	Diagramme Essai 201	30
6.2	Boxplot Essai 201	31
6.3	Densité cellulaire à 72h en fonction des concentrations : concaténation	33
6.4	QQplot par concentration	35
6.5	normale <i>vs</i> non normale	36
6.6	scatterplots homoscedasticité vs hétéroscedasticité	36
8.1	Trace log-logistique Pente (b)	45
8.2	Trace log-logistique (EC_{50})	45
8.3	Trace log-normal Moyenne logarithmique	45
8.4	Trace log-normal Variance logarithmique	45
8.5	Trace Weibull Paramètre de forme	45
8.6	Trace Weibull Paramètre d'échelle	45
8.7	ACF pour le modèle log-logistique Pente (b)	46
8.8	ACF pour le modèle log-logistique (EC_{50})	46
8.9	ACF pour le modèle log-normal Moyenne logarithmique	46
8.10	ACF pour le modèle log-normal Variance logarithmique	46
8.11	ACF pour le modèle Weibull Paramètre de forme	46
8.12	ACF pour le modèle Weibull Paramètre d'échelle	46
C.1	Architecture générale de l'application	54
C.2	Diagramme essais standardisés	55

Tableaux

1	Résumé des modèles couramment utilisés avec leurs paramètres possibles (1 à 5)	18
2	Outliers détectés par la méthode de Tukey	31
3	Résultats du test de Shapiro-Wilk pour chaque concentration	34
4	Vérification Homoscedasticité	34
5	Statistiques pour chaque concentration de substance testée (Test 201 OCDE)	37
6	Résultats des tests ANOVA pour déterminer l'effet de chaque concentration sur la croissance des algues (Test 201 OCDE)	38
7	Résultats du test de Kruskal-Wallis pour les différentes concentrations (Test Non-Paramétrique)	39
8	Estimation des paramètres pour les modèles log-logistique, log-normal et Weibull.	42
9	Valeurs de l'AIC	43
10	Facteurs de Bayes	43
11	Taux d'acceptation pour chaque modèle.	44
12	Estimations des doses effectives à 10%, 20%, et 50% pour le modèle Weibull.	47
13	Intervalle de confiance bootstrap à 95% pour les doses effectives estimées.	48
14	Forces et Limites des Tests Statistiques	51
15	Critères de Choix des Tests Statistiques	51
16	Tests Paramétriques: Forces, Limites et Critères de Choix	200
17	Tests Non-Paramétriques: Forces, Limites et Critères de Choix	201

2 Table des abréviations

Abréviation	Signification
EC	Effective Concentration (Concentration Efficace)
ED	Effective Dose
LC	Lethal Concentration
LD	Lethal Dose
LOEC	Lowest Observed Effect Concentration
LOEL	Lowest Observed Effect Level
NOEC	No Observed Effect Concentration
NOEL	No Observed Effect Level
OCDE	Organisation de Coopération et de Développement Économiques

3 Glossaire

Biocide: Ensemble de produits destinés à détruire, repousser ou rendre inoffensifs les organismes nuisibles, à en prévenir l'action ou à les combattre, par une action chimique ou biologique.

Biomasse: La biomasse est l'ensemble de la matière organique d'origine végétale ou animale présente dans un espace fini, un biotope par exemple, à un instant t

EC_x (Effective Concentration x%) (Concentration Efficace): La concentration d'une substance/produit qui provoque des effets sublétaux pour $x\%$ des organismes testés.

Effet subléta: Les effets sublétaux peuvent affecter, entre autres, la durée de vie, le développement, la croissance de la population, la fertilité et le comportement.

Essai dose-reponse Employé pour estimer la concentration à laquelle un pourcentage donné d'organismes montre une réponse, comme l' EC_{50} . Cette méthode utilise plusieurs concentrations pour établir une courbe dose-réponse, fournissant une vue plus complète de la toxicité de la substance testée.

Essai limite: Utilisé pour déterminer si une substance est toxique à une concentration donnée. Cette approche se concentre sur une ou quelques concentrations spécifiques, souvent à des niveaux élevés, pour évaluer la présence ou l'absence d'effet toxique significatif.

Intrants du végétal : substances et produits phytopharmaceutiques, matières fertilisantes et supports de culture, macro-organismes non indigènes utiles aux végétaux.

ISO : Organisation internationale de normalisation.

LC_x/LD_x (Lethal Concentration/Dose x%): Concentration ou dose d'une substance/produit qui est mortelle pour $x\%$ de la population étudiée.

LOEC (Lowest Observed Effect Concentration) ou LOEL (Lowest Observed Effect Level): La concentration ou dose testée la plus faible d'un contaminant où un effet significatif est observé sur les organismes testés.

Minimum Detectable Difference (MDD) : Une mesure utilisée pour déterminer la plus petite différence entre des groupes de traitement qui peut être détectée de manière statistiquement significative dans une étude expérimentale. Le MDD dépend de la variance des données, de la taille de l'échantillon et du niveau de signification statistique choisi. En écotoxicologie, le MDD permet de déterminer la sensibilité de l'essai pour détecter les effets des substances chimiques sur les organismes testés.

NOEC (No Observed Effect Concentration) ou NOEL (No Observed Effect Level): La concentration ou dose testée la plus élevée d'une substance/produit où aucun effet significatif n'est observé sur les organismes testés.

OCDE (Organisation de Coopération et de Développement Économiques): Organisation internationale qui publie des documents guides standardisés pour la mise en application de nombreux essais de laboratoire, y compris les tests d'écotoxicité.

Organismes non-cibles : organismes qui ne sont pas visés par un traitement ou une substance chimique, mais qui peuvent en être affectés.

4 Introduction

L'Agence nationale de sécurité sanitaire de l'alimentation, de l'environnement et du travail (Anses) assure des missions de veille, d'expertise, de recherche et de référence sur un large champ couvrant la santé humaine, la santé et le bien-être animal, la santé végétale et la santé de l'environnement. Elle offre une lecture transversale des questions sanitaires et appréhende ainsi, de manière globale, les expositions auxquelles l'Homme peut être soumis à travers ses modes de vie et de consommation ou les caractéristiques de son environnement, y compris professionnel. L'Anses informe les autorités compétentes, répond à leurs demandes d'expertise. L'Agence exerce ses missions en étroite relation avec ses homologues européens. Afin d'assurer ses diverses missions, l'agence est découpée en plusieurs pôles et directions :



Figure 4.1: ANSES

J'ai effectué mon stage dans le pôle **Produits réglementés**, et plus précisément au sein de la **DEPR** (Direction de l'Évaluation des Produits Réglementés (DEPR) dont l'**unité d'Évaluation Écotoxicologie et Environnement des Intrants du Végétal (U3EiV)** fait partie. Cette structure permet de remplir ses missions au travers de ses multiples directions. La DEPR évalue, selon la réglementation en vigueur, les risques liés à l'utilisation des substances actives, des produits phytopharmaceutiques et **biocides**, ainsi que les matières fertilisantes, supports de cultures, substances actives et produits biocides. Cette évaluation est réalisée, avant décision d'autorisations de mise sur le marché. En ce qui concerne l'évaluation des intrants du végétal (substances et produits phytopharmaceutiques, matières fertilisantes et supports de culture, macro-organismes non indigènes utiles aux végétaux), la DEPR met en œuvre une expertise scientifique interne et peut faire appel à ses comités d'experts spécialisés dédiés. Pour répondre à des saisines des ministères, elle peut faire appel, en complément des experts internes, à des, aux groupes de travail constitués auprès des experts de l'Anses et peut travailler en liaison avec les autres directions de l'Agence. Mon stage s'est déroulé au sein de l'U3EiV. Cette unité est chargée de l'évaluation scientifique des parties devenir et comportement dans l'environnement et écotoxicologie des dossiers d'autorisation des produits phytopharmaceutiques, matières fertilisantes, supports de cultures et des substances actives. À ce titre, elle évalue les parties correspondantes des dossiers, assure la cohérence des évaluations réalisées dans le respect des réglementations européennes et nationales. Cette unité est dirigée par Vanessa Mazerolles (Cheffe d'unité (CU)), assistée de ses adjoints Émilie Farama (CU adjointe) et Pierre-François Chaton (adjoint au CU). Mon stage au sein de cette unité m'a offert l'opportunité d'observer et de comprendre les processus d'évaluation. Il m'a également permis d'acquérir une expérience concrète en travaillant aux côtés de professionnels expérimentés, ce qui m'a aidé à mieux appréhender les défis et les exigences liés à un domaine très différent du mien : l'évaluation réglementaire en écotoxicologie. Cette expérience a permis d'apporter ma contribution pour améliorer la méthodologie d'évaluation statistique des évaluateurs de risque en développant un outil d'aide à la décision pour le choix des tests statistiques les plus appropriés à employer en fonction des tests **d'écotoxicité** déjà réalisés pour une substance

ou un produit. L'écotoxicologie est une science qui étudie les effets des substances chimiques/produits sur les organismes vivants dans l'environnement (ref : Green et al., 2006 [3]). Cette science adopte une approche hiérarchique, où les processus observés à un certain niveau (moléculaire) sont expliqués par des mécanismes à un niveau inférieur (par exemple, l'organisme) et peuvent avoir des conséquences à un niveau supérieur (population ou écosystème). Cette structure complexe permet une évaluation plus intégrée et holistique (c'est-à-dire globale) des effets des polluants. Afin de répondre aux requis réglementaires européens concernant l'évaluation des risques des substances actives et produits phytopharmaceutiques (règlement (CE) n°1107/2009 ref : eu2009 [1]), de nombreux tests en laboratoire et en conditions plus réalistes sont nécessaires pour caractériser le comportement de l'intrant dans l'environnement ainsi que ses effets sur les organismes non-cibles. La majorité de ces essais d'écotoxicité sont **standardisés** et suivent des documents guides lignes directrices pour chaque groupe **d'organismes non-cibles**.

- Oiseaux via les documents OCDE 223[40] ou l'OCDE 205[30] par exemple
- Poissons via les documents OCDE 210[32] par exemple
- Invertébrés d'eau douce (e.g. daphnie, chironomes) OCDE 211[33] ou 218[37] par exemple
- Pollinisateurs via les documents OCDE 245[43] par exemple
- Arthropodes, ESCORT II [49] par exemple
- Organismes du sol via les documents OCDE 222[39] par exemple
- Activité microbienne (e.g. nitrification du sol) via les documents OCDE 216[36] par exemple
- Plantes via les documents OCDE 208[46] par exemple

Ces lignes directrices d'essais sont notamment celles définies via les normes OCDE/ISO [2]. Ces tests permettent d'obtenir des valeurs de toxicité pour chaque groupe d'organismes non-cibles. Pour cela, les organismes sont exposés à différentes concentrations, doses ou niveaux d'exposition afin d'établir des courbes doses-réponses établissant concrètement le lien entre concentration/dose en substance/polluant et effets observés (inhibition de croissance, réduction du nombre de jeunes...) selon des protocoles expérimentaux clairement établis. Toutefois, les lignes directrices ne précisent pas nécessairement les tests statistiques qui seraient les plus appropriés pour déterminer ces valeurs de toxicité au vu du protocole expérimental et du type de résultats. Le document guide OCDE 54 (2006)[2], présente de manière très générale des propositions adaptées à l'écotoxicologie en fonction du type d'essai. Depuis la parution de ce document, de nouvelles méthodes statistiques ont été proposées afin d'améliorer l'analyse des données des essais réalisés en écotoxicologie. De fait, les discussions entre évaluateurs de risque incluent les meilleures méthodes statistiques à utiliser pour analyser les données de toxicité issues de ces essais standardisés. Ce stage vise à développer un guide des bonnes pratiques d'analyses statistiques pour les tests écotoxicologiques, sous forme d'une application web via R Shiny. Cette application a pour but de guider les évaluateurs de risque dans le choix des tests statistiques les plus appropriés, en tenant compte des caractéristiques spécifiques des données obtenues lors des tests en laboratoire. Elle intégrera les recommandations des documents guides et les avancées récentes en méthodes statistiques, afin d'améliorer la précision et la reproductibilité des évaluations de risque. L'outil d'appui statistique est développé en anglais et permettra à l'évaluateur

de risque de :

- De disposer d'un outil d'aide à la décision afin de mieux justifier le choix du test statistique le plus pertinent pour l'essai concerné en se basant sur les contraintes et le protocole expérimental défini dans les normes ;
- De réaliser le/les test(s) statistique(s) adéquat(s) tout en mettant en évidence la validité des conditions d'application du test employé et, si possible, sa/leur puissance et sa/leur robustesse ;
- De fournir, sous forme d'un rapport succinct, un résumé des résultats associés les plus pertinents à l'issue de l'utilisation de l'application en ligne.

De plus, ce projet permet également de :

- Simplifier et centraliser les informations sur les tests statistiques disponibles.
- Mettre à jour les méthodes statistiques employées par les évaluateurs de risque avec des techniques récentes.
- Développer un indicateur de fiabilité des tests utilisés (puissance post-hoc).

5 Méthodologie

Ce travail s'appuie sur le **document guide OCDE 54 (2006)**[2], qui définit les protocoles pour la préparation et le traitement des données obtenues dans les études d'écotoxicité à la suite de leur mise en application "biologique" dans les laboratoires d'essai. La première étape du développement de l'outil a consisté en une analyse exhaustive de la littérature existante, en particulier les documents guides pour les essais d'écotoxicité, de **l'OCDE 54 (2006)** et des ouvrages scientifiques récents sur les méthodes statistiques appliquées en écotoxicologie (ref : Ggreen et al., 2018),[3]. Cette revue de la littérature a permis d'identifier les méthodes statistiques couramment utilisées, leurs avantages et leurs limitations, ainsi que les recommandations actuelles pour l'analyse des données d'écotoxicité. En parallèle de l'analyse de la littérature scientifique, j'ai consacré une partie importante de mon temps à l'apprentissage de RShiny grâce à des ressources disponibles en ligne. Cette maîtrise de RShiny a été essentielle pour la visualisation interactive et l'analyse des données écotoxicologiques afin de mieux comprendre les résultats expérimentaux et d'améliorer l'interface utilisateur pour l'outil d'aide à la décision développé.

5.1 Description des données des essais d'écotoxicité

Les tests d'écotoxicité consistent à exposer des organismes non cibles à un traitement en utilisant soit une concentration unique (**essai limite**), soit plusieurs concentrations différentes suivant un gradient géométrique (**essai dose-réponse**), permettant ainsi d'évaluer les effets potentiels du traitement sur ces organismes. Le **design expérimental** en écotoxicologie inclut souvent la **randomisation des traitements** appliqués aux sujets d'étude pour éviter les biais dans la collecte des données. La **réplication**, ou répétition des expériences, est essentielle pour assurer la fiabilité et la validité des résultats obtenus. Les types de données en lien avec ce design expérimental peuvent être classés selon leur nature et la manière dont elles seront analysées. Cette classification aide à déterminer les tests statistiques les plus appropriés et à interpréter les résultats de manière précise.

1. **Données Quantales (ie. quantitatives Binaires):**

Les données binaires sont souvent utilisées pour enregistrer la présence ou l'absence d'un effet, comme la mortalité ou la survie des organismes testés. Par exemple, dans un test de toxicité, 0 pourrait indiquer la survie et 1 la mortalité.

2. **Données Continues:**

Ces données incluent des mesures qui varient sur un continuum (*i.e.* en fonction de la concentration de la substance toxique, appliquée selon un gradient continu), telles que la concentration de substances chimiques dans l'eau ou la **biomasse** des organismes après exposition. Lors de mon stage, ce type de données a été utilisé pour établir des modèles **dose-réponse**.

3. **Données Discrètes:**

Ces données occupent une position intermédiaire entre les données quantales et continues dans les études d'écotoxicité; Elles apparaissent généralement sous forme de dénombrements.

4. Et définies comme des données Discrètes de comptages :

- **Nominales:** Les données nominales, comme le type de tissu affecté par un toxique, ne comportent **aucun ordre implicite**. Elles servent à catégoriser les échantillons sans échelle de mesure.
- **Ordinales:** Ces données impliquent un **ordre**, comme le degré de toxicité observé (faible, moyen, fort), qui peut être utilisé pour des analyses de tendance.
- **Intervalle:** Moins fréquentes, les données d'intervalle comme la température mesurée en degrés Celsius, où le zéro est arbitraire, permettent de réaliser des analyses plus complexes en conservant la signification des différences entre les valeurs.

Les données traitées comprennent diverses mesures, telles que **la survie, la croissance, la reproduction**, et d'autres indicateurs biologiques pertinents pour évaluer les effets écotoxiques. Chaque mesure est enregistrée pour chaque réplicat et chaque concentration nominale, ce qui génère un ensemble de données complexes à exploiter.

5.1.1 Réplicats et méthode de randomisation

Les réplicats sont analysés pour vérifier leur cohérence et sont essentiels pour estimer la variabilité naturelle, fournissant ainsi une base solide aux tests statistiques. Toutefois, toute variabilité excessive ne pourra être prise en compte que lors de l'interprétation des résultats, car les études sont déjà réalisées. La robustesse et la répétabilité des résultats expérimentaux en écotoxicologie reposent sur l'utilisation de réplicats et la mise en œuvre de méthodes de randomisation rigoureuses. En écotoxicologie, les réplicats consistent en des répétitions indépendantes d'une même condition expérimentale, capturant ainsi la variabilité naturelle des organismes et des conditions environnementales. En multipliant les réplicats, il est possible de calculer des mesures de variabilité, telles que l'écart-type ou l'intervalle de confiance, augmentant la précision des estimations des valeurs de toxicité. L'utilisation de réplicats dans différents groupes de traitement soumis à des concentrations variées de substances, souvent qualifiées de toxiques en fonction de leurs effets, garantit que les données obtenues sont à la fois représentatives et fiables. Ces réplicats sont essentiels pour évaluer l'effet

dose-réponse et minimiser la variabilité expérimentale, assurant ainsi des conclusions basées sur des réponses biologiques véritables. Les données écotoxicologiques peuvent être biaisées de diverses manières, et les réplicats jouent un rôle clé dans l'atténuation de ces biais. Par exemple, le biais de sélection, qui survient lorsque les échantillons ne représentent pas la population d'intérêt, peut être réduit en augmentant la diversité des échantillons grâce aux réplicats. Le biais de mesure, lié à l'imprécision des instruments, peut être identifié et corrigé avec des réplicats, améliorant ainsi la précision des données. De même, les biais de confondance, de traitement et d'observateur peuvent être minimisés par une conception expérimentale rigoureuse et l'utilisation de réplicats. La randomisation, quant à elle, est cruciale pour minimiser les biais expérimentaux. En répartissant aléatoirement les traitements et les réplicats, on s'assure que les variations observées sont dues aux traitements appliqués et non à des facteurs confondants, renforçant ainsi la validité interne des résultats.

5.1.2 Exposition des organismes

Les organismes sont exposés à différents types de concentrations (C), doses (D) ou niveaux d'exposition (L). Pour simplifier, nous nous référerons uniquement aux concentrations (C) dans la suite de ce document, bien que les principes s'appliquent également aux doses (D) et niveaux d'exposition (L). Les expériences incluent différentes concentrations des substances testées, disposées selon un gradient géométrique, allant d'un contrôle (sans substance) à plusieurs niveaux de concentration. Le ou les contrôles sont toujours comparés à une ou plusieurs concentrations de la substance testée, et un contrôle solvant est souvent inclus si un solvant est utilisé.

L'organisation des données inclut typiquement les éléments suivants :

- **Contrôle :**
Groupe témoin sans exposition à la substance testée, servant de référence pour comparer les effets des différentes concentrations.
- **Contrôle solvant :**
Groupe témoin exposé uniquement au solvant utilisé pour servir de vecteur à la substance testée, pour évaluer l'effet potentiel du solvant lui-même.
- **Concentrations d'essai :**
Différents niveaux de la substance testée, généralement disposés en gradient géométrique pour couvrir une large gamme de concentrations.

Ces concentrations permettent de déterminer la relation dose-réponse, c'est-à-dire comment les effets varient en fonction de la concentration de la substance.

5.2 Préparation des données pour l'analyse

Conformément aux protocoles détaillés dans le document **OCDE 54**, la préparation des données pour l'analyse statistique dans les études d'écotoxicité est une étape cruciale. Cette phase inclut plusieurs étapes spécifiques qui garantissent l'intégrité et la validité des analyses suivantes, pour chaque type de données, que ce soit pour des essais limite ou dose-réponse.

5.2.1 Inspection Visuelle et Nettoyage des Données

La première étape consiste à réaliser une inspection visuelle des données afin d'identifier les valeurs aberrantes (appelées « outliers ») et d'autres anomalies potentielles. Cette inspection permet de détecter des incohérences, telles que des réplicats qui divergent de manière

significative des autres, ce qui pourrait indiquer des erreurs expérimentales ou des biais systématiques. Cette phase de vérification est essentielle pour éviter que ces anomalies ne biaisent les résultats finaux.

5.2.2 Vérification de l'Effet Solvant

Les solvants ne sont pas systématiquement utilisés dans tous les essais dose-réponse ou les essais limite. Leur usage dépend de la solubilité de la substance testée. Un test statistique non paramétrique, tel que le test de Wilcoxon-Mann-Whitney, est appliqué pour comparer ces deux groupes. Ce test est choisi en raison de sa robustesse aux distributions non normales, souvent observées dans les données écotoxicologiques. Si le test montre qu'il n'y a pas de différence statistiquement significative entre les deux groupes ($p > 0.05$), les données du groupe contrôle solvant et du groupe contrôle peuvent être regroupées (concaténées) en un seul groupe contrôle (qui sera noté "0"). En revanche, si une différence significative est détectée, une comparaison des moyennes est réalisée pour déterminer quel groupe présente la réponse la plus faible :

- Si la moyenne de la réponse dans le groupe contrôle solvant est inférieure à celle du groupe contrôle pour des paramètres comme la reproduction, cela suggère un effet toxique du solvant. Dans ce cas, seules les données du groupe contrôle solvant sont utilisées pour les analyses ultérieures.
- À l'inverse, si la moyenne de la réponse dans le groupe contrôle est inférieure à celle du groupe contrôle solvant pour des paramètres comme la mortalité, cela indique un effet du contrôle. Seules les données du groupe contrôle sont alors retenues pour les analyses.

5.2.3 Contrôle des Hypothèses et Transformations des Données

Avant d'appliquer des méthodes statistiques spécifiques, il est nécessaire de vérifier les hypothèses sous-jacentes des analyses envisagées, telles que la normalité des résidus et l'homogénéité des variances. Si ces hypothèses ne sont pas respectées, des transformations de données pourraient être nécessaires. Par exemple, pour les données continues qui montrent une variance hétérogène, une transformation logarithmique pourrait être appliquée pour stabiliser la variance[3]. Les principales hypothèses à respecter incluent :

1. **Indépendance des observations:** Les observations doivent être indépendantes les unes des autres pour éviter les biais dans l'estimation des paramètres. Cette hypothèse est cruciale pour assurer la fiabilité des conclusions tirées du modèle.
2. **Spécification correcte du modèle:** Le modèle choisi doit correspondre à la relation réelle entre les variables. Une mauvaise spécification peut entraîner des biais importants et des erreurs dans l'interprétation des résultats.
3. **Convergence de l'algorithme d'estimation:** Lors de l'estimation des paramètres des modèles non linéaires, il est essentiel que l'algorithme converge vers une solution optimale globale optimale, et non vers un minimum local. La convergence assure que les estimations des paramètres sont fiables et reproductibles.
4. **Homoscédasticité:** Les résidus du modèle doivent présenter une variance constante (homoscédasticité) à travers toutes les valeurs des variables indépendantes. Si cette condition n'est pas respectée, les estimations des paramètres pourraient être inefficaces, et les tests statistiques pourraient conduire à des conclusions incorrectes.

5. **Normalité des erreurs:** Bien que non indispensable pour toutes les méthodes d'estimation, la normalité des résidus est souvent souhaitée, surtout si l'on souhaite réaliser des tests de significativité ou construire des intervalles de confiance.
6. **Multicolinéarité :** Dans les modèles multi-paramétriques, il est important de s'assurer qu'il n'y a pas de multicolinéarité excessive entre les variables explicatives, car cela pourrait rendre les estimations des paramètres instables.
7. **Sensibilité aux valeurs aberrantes:** Les modèles non linéaires peuvent être sensibles aux valeurs aberrantes. Il est important d'identifier et de traiter ces valeurs avant l'analyse.
8. **Identifiabilité des paramètres:** Tous les paramètres doivent être identifiables, c'est-à-dire qu'ils doivent pouvoir être estimés de manière unique à partir des données disponibles.

Ces hypothèses doivent être systématiquement vérifiées pour garantir que les modèles non linéaires produisent des résultats fiables et interprétables. Pour plus de détails sur ces aspects techniques, on peut se référer à l'ouvrage de Bates et Watts (1988) [53], qui fournit une analyse approfondie de ces considérations dans le contexte des modèles non linéaires.

5.3 Application de Modèles Statistiques Appropriés (illustration cf figure 5.1)

La modélisation statistique en écotoxicologie représente une pierre angulaire de l'analyse des données, permettant de quantifier et de prédire les effets toxiques en fonction des concentrations de substances chimiques. L'application de modèles adaptés[54] est essentielle pour capturer avec précision la relation dose-réponse et garantir des conclusions robustes. Selon les caractéristiques spécifiques des données et les objectifs de l'étude, divers modèles peuvent être employés, chacun offrant des avantages uniques. Les **modèles dose-réponse non linéaires** sont souvent privilégiés pour leur capacité à décrire fidèlement la complexité des interactions biologiques. Ces modèles, tels que les modèles de Hill, Weibull, ou log-logistique, permettent de représenter la relation entre la concentration d'un toxique et l'effet observé de manière plus précise que les modèles linéaires. Le choix du modèle dépend de plusieurs critères, dont la qualité de l'ajustement aux données, la simplicité du modèle, et la nature des données qu'elles soient continues, discrètes ou binaires. Lorsque les données de dénombrement présentent une grande variabilité, elles peuvent être traitées comme des réponses continues après une transformation adéquate, telle qu'une transformation en racine carrée. Cette approche permet de mieux répondre aux exigences des tests statistiques en minimisant l'impact des écarts-types disproportionnés. Dans les cas où les données de dénombrement suivent une distribution de Poisson, l'utilisation d'un **modèle linéaire mixte généralisé (GLMM)**[3] se révèle particulièrement efficace, offrant une approche robuste et flexible pour des données présentant une grande variabilité. Les méthodes statistiques doivent être soigneusement choisies en fonction du type de données à analyser. Pour les données binaires, les tests de proportions sont souvent utilisés, tandis que les données continues ou ordinales nécessitent des modèles plus sophistiqués pour capter la nature complexe des interactions biologiques. Par exemple, le **modèle probit** est couramment employé pour estimer les effets toxiques à faible concentration, alors que le **modèle à seuil** (comme le modèle en bâton de hockey) est idéal pour décrire des situations où l'effet toxique n'apparaît qu'après avoir atteint un certain seuil de concentration. L'application rigoureuse des modèles statistiques ne se limite pas à leur sélection ; il est nécessaire de vérifier les hypothèses sous-jacentes

pour chaque modèle utilisé. Par exemple, le **modèle de Weibull** nécessite une vérification de la distribution asymétrique des résidus, tandis que le **modèle de Hill** doit correspondre à une relation sigmoïde avec une phase de montée rapide suivie d'une saturation. De même, le **modèle log-logistique** est vérifié pour son aptitude à décrire des courbes dose-réponse asymétriques. Enfin, bien que les modèles non linéaires offrent une flexibilité considérable, ils nécessitent une validation attentive pour garantir la fiabilité des résultats. En combinant une sélection judicieuse des modèles avec une vérification rigoureuse des hypothèses, il est possible de garantir que les conclusions écotoxicologiques reposent sur une base statistique solide, offrant ainsi des prédictions précises et utiles pour la gestion des risques.

5.4 Formes Mathématiques des Modèles Statistiques Utilisés dans l'application

Dans cette sous-section, nous présentons les formes mathématiques des modèles statistiques les plus couramment utilisés en écotoxicologie : Weibull, log-logistique, logit, probit, Hill, Gumbel, Poisson, binomial négatif, et exponentiel (Slob).

5.4.1 Modèle de Weibull

Le modèle de Weibull, souvent utilisé pour modéliser des données de survie, est donné par :

$$Y = a \cdot \left(1 - e^{-\left(\frac{C}{b}\right)^c}\right) + \delta \quad (1)$$

où Y est la réponse, C est la concentration.

5.4.2 Modèle log-logistique

Utilisé pour des données continues et quantales avec une relation dose-réponse sigmoïde :

$$Y = \frac{a}{1 + \left(\frac{C}{EC_{50}}\right)^b} + \delta \quad (2)$$

où a est la réponse maximale, C est la concentration.

5.4.3 Modèle logit

Adapté pour des relations dose-réponse sigmoïdes, souvent pour des données quantales ou continues :

$$Y = a + \frac{b - a}{1 + e^{-k(C - EC_{50})}} + \delta \quad (3)$$

où C est la concentration.

5.4.4 Modèle probit

Le modèle probit, utilisé principalement pour les données quantales, est exprimé par :

$$Y = a + b \cdot \Phi^{-1} \left(\frac{C}{EC_{50}} \right) + \delta \quad (4)$$

où Φ^{-1} est la fonction quantile de la distribution normale

5.4.5 Modèle de Hill

Adapté pour des réponses continues avec une courbe sigmoïde, souvent utilisé en biologie :

$$Y = \frac{E_{\max} \cdot C^n}{EC_{50}^n + C^n} + \delta \quad (5)$$

où C est la concentration et n est le coefficient de Hill.

5.4.6 Modèle de Gumbel

Souvent utilisé pour modéliser les données de survie, ce modèle est donné par :

$$Y = a \cdot \left(1 - e^{-e^{\frac{C-b}{c}}} \right) \quad (6)$$

5.4.7 Modèle de Poisson

Utilisé pour des données de comptage, lorsque la variance est égale à la moyenne :

$$P(Y = k) = \frac{\lambda^k e^{-\lambda}}{k!} \quad (7)$$

où λ est le taux de comptage.

5.4.8 Modèle binomial négatif

Utilisé pour des données de comptage avec surdispersion (variance > moyenne) :

$$P(Y = k) = \frac{\Gamma(k + \theta)}{\Gamma(\theta)k!} \left(\frac{\theta}{\theta + \mu} \right)^\theta \left(\frac{\mu}{\theta + \mu} \right)^k \quad (8)$$

où μ est la moyenne, et θ est le paramètre de dispersion.

5.4.9 Modèle exponentiel (Slob 2002 [3])

Utilisé pour modéliser des réponses exponentielles :

$$Y = a \exp \left(-\frac{C}{b} \right) \quad (9)$$

où C est la concentration.

Modèle	Données	Forme	Paramètres
Weibull 1,2,3,4	Continues	Asymétrique Sigmoides si $\alpha > 1$	a (amplitude), b (échelle), c (forme), δ (décalage)
Hill 2,3,4,5	Continues	Sigmoides	E_{\max} (effet max), EC_{50} , n (pente), δ (décalage)
Log-logistique 2,3,4	Continues Quantales	Sigmoides	a (amplitude), EC_{50} , b (pente), δ (décalage)
Probit 2,3	Quantales	Sigmoides	a (intercept), b (pente), δ (décalage)
Logit 2,3,4	Quantales Continues	Sigmoides	a (intercept), b (pente), k (taux de croissance), δ (décalage)
Gumbel	Quantales	Asymétrique	a (amplitude), b (position), c (pente)
Poisson	Comptage	Exponentiel	λ (taux de comptage)
binomial négatif	Comptage	Exponentiel	μ (moyenne), θ (dispersion)
exponentiel 1,2,3	Continues Quantales	Exponentiel	a (amplitude), b (échelle), c (décroissance)

Table 1: Résumé des modèles couramment utilisés avec leurs paramètres possibles (1 à 5)

5.5 Sélection de modèle

Dans le domaine de la sélection de modèles, le **Critère d'Information d'Akaike (AIC)**[8] et le **Critère d'Information Bayésien (BIC)**[7] figurent parmi les outils les plus fréquemment utilisés. Ces critères offrent des approches quantitatives pour comparer différents modèles en intégrant une pénalisation de la complexité pour éviter le **surajustement**. En complément de ces méthodes, le facteur de Bayes se présente également comme une alternative puissante, permettant d'évaluer la probabilité relative des modèles tout en prenant en compte les distributions a priori.

1. Critère d'Information d'Akaike (AIC):

L'AIC est fondé sur l'idée de minimiser la divergence de Kullback-Leibler, qui mesure la perte d'information entre le modèle ajusté et la "vérité" des données. Le critère AIC est défini comme :

$$AIC = -2\log(L) + 2k$$

Où :

- $\log(L)$ est le logarithme de la vraisemblance du modèle.
- k est le nombre de paramètres estimés dans le modèle.

L'objectif est de minimiser l'AIC. Un AIC plus faible indique un modèle qui offre un bon ajustement aux données avec un nombre raisonnable de paramètres. Cependant, l'AIC peut parfois favoriser des modèles plus complexes, car il pénalise faiblement le nombre de paramètres.

2. Critère d'Information Bayésien (BIC) :

Le BIC, également connu sous le nom de critère de Schwarz, introduit une pénalisation plus forte pour la complexité du modèle, en particulier lorsque la taille de l'échantillon est grande. Le BIC est défini comme suit :

$$BIC = -2 \log(L) + k \log(n)$$

Où :

- n est la taille de l'échantillon.
- $\log(L)$ est le logarithme de la vraisemblance du modèle.
- k est le nombre de paramètres estimés.

Comme pour l'AIC, un BIC plus faible est préférable. Cependant, en raison de la pénalisation plus forte liée à la taille de l'échantillon, le BIC tend à favoriser des modèles plus simples, particulièrement lorsque l'échantillon est grand. l'**AIC** privilégie un bon ajustement avec une pénalisation légère de la complexité. Il est souvent utilisé lorsque l'objectif est de faire des prédictions et que l'on préfère un modèle qui capture plus de nuances des données, même au risque de surajuster. Tandis que, le **BIC** : Privilégie des modèles plus parcimonieux avec une pénalisation plus sévère pour la complexité. Il est préféré lorsque l'on cherche à identifier le modèle "vrai" parmi un ensemble de candidats, en particulier dans un contexte bayésien. En pratique, lors de la vérification et de la sélection des modèles, il est recommandé d'examiner à la fois l'AIC et le BIC pour guider la décision. Si les deux critères suggèrent le même modèle, cela renforce la confiance dans le choix du modèle. Cependant, si les critères AIC et BIC favorisent des modèles différents, le choix dépendra de la priorité donnée à l'ajustement par rapport à la simplicité. Dans les situations où le nombre total d'observations n est limité, une correction de l'AIC, notée **AICc**, est utilisée pour ajuster la pénalisation en fonction de la taille de l'échantillon :

$$AICc = -2 \log(L) + 2k \frac{n}{n - k - 1}$$

Cette correction est particulièrement importante lorsque le ratio entre le nombre de paramètres et la taille de l'échantillon est élevé, garantissant ainsi une évaluation plus précise du modèle. Lorsque les valeurs d'AICc de deux modèles sont proches, il peut être utile de calculer la probabilité qu'un modèle soit meilleur qu'un autre. Cette probabilité est donnée par :

$$\text{Probabilité} = \frac{e^{-D/2}}{1 + e^{-D/2}}$$

où D est la différence entre les valeurs AICc des deux modèles. Cette formule permet d'évaluer la supériorité relative d'un modèle par rapport à un autre, renforçant ainsi

le processus décisionnel. Le **principe de parcimonie**, également connu sous le nom de Rasoir d'Occam, recommande de préférer le modèle le plus simple parmi ceux offrant un ajustement adéquat. Le BIC est particulièrement adapté à ce principe, en pénalisant fortement les modèles trop complexes, ce qui encourage la sélection d'un modèle qui explique les données avec le moins de paramètres possible.

3. Approche Bayésiennes [11] :

Cette approche bayésienne, essentiels pour l'analyse des données écotoxicologiques, offre un cadre robuste pour la comparaison des modèles en permettant d'incorporer des informations a priori et d'obtenir des probabilités a posteriori interprétables. Cette méthode se distingue notamment par l'utilisation des **facteurs de Bayes** (Bayes Factors, BF), qui permettent de comparer directement les probabilités a posteriori de différents modèles en tenant compte de la vraisemblance des données sous chacun d'eux.

Pour appliquer la méthode de sélection de modèles bayésienne, il est essentiel de commencer par définir clairement les modèles en compétition. Le modèle nul, souvent considéré comme modèle de référence, est désigné par M_0 , tandis que M_1 représente le modèle alternatif que l'on souhaite comparer au modèle nul. Chacun de ces modèles est associé à un ensemble de paramètres θ_i , pour lequel on doit spécifier une distribution a priori $\pi_i(\theta)$. Cette distribution intègre les connaissances préalables sur les paramètres, par exemple, si des études antérieures suggèrent que la moyenne d'une distribution est centrée autour d'une certaine valeur, cette information sera incluse dans $\pi(\theta)$. Une fois les modèles définis, il faut spécifier les distributions a priori pour les paramètres sous chaque modèle. Pour le modèle M_0 , la distribution a priori $\pi_0(\theta)$ est choisie en fonction des hypothèses formulées pour M_0 . De même, pour le modèle M_1 , une autre distribution a priori $\pi_1(\theta)$ est déterminée. La fonction de vraisemblance $f^{(n)}(\text{données} \mid \theta)$ est ensuite utilisée pour représenter la probabilité d'observer les données pour un ensemble de paramètres donné. Enfin, le facteur de Bayes $B_{0/1}$ est calculé pour comparer la probabilité marginale des données sous le modèle M_0 par rapport à celle sous le modèle M_1 .

$$B_{0/1} = \frac{\text{Vraisemblance marginale de } M_0}{\text{Vraisemblance marginale de } M_1}$$

Ou en notation plus détaillée :

$$B_{0/1} = \frac{\int_{\Theta_0} \pi_0(\theta_0) f^{(n)}(\text{données} \mid \theta_0) d\theta_0}{\int_{\Theta_1} \pi_1(\theta_1) f^{(n)}(\text{données} \mid \theta_1) d\theta_1}$$

Ce facteur est déterminant pour interpréter les résultats : lorsque $B_{0/1}$ est supérieur à 1, les données fournissent un soutien plus fort en faveur du modèle M_0 , suggérant sa préférence. En revanche, si $B_{0/1}$ est inférieur à 1, cela indique que le modèle M_1 est préféré, les données le soutenant davantage. Le facteur de Bayes $B_{0/1}$ pour comparer deux modèles non linéaires $Y = f(X, \theta) + \epsilon$, M_0 et M_1 est donné par :

$$B_{0/1} = \frac{\int_{\Theta_0} f_0(Y \mid X_0, \theta_0) \pi_0(\theta_0) d\theta_0}{\int_{\Theta_1} f_1(Y \mid X_1, \theta_1) \pi_1(\theta_1) d\theta_1}$$

Où:

- $f_0(Y \mid X_0, \theta_0)$: Est la fonction de vraisemblance du modèle M_0 , c'est-à-dire la probabilité des données Y sachant X_0 et les paramètres θ_0 .
- $\pi_0(\theta_0)$: Est la distribution a priori des paramètres θ_0 pour le modèle M_0 .
- Θ_0 : Est l'espace des paramètres pour M_0 .
- $f_1(Y \mid X_1, \theta_1)$ et $\pi_1(\theta_1)$: Sont les équivalents pour le modèle M_1 .
- $d\theta_0$ et $d\theta_1$: Indiquent l'intégration sur tous les paramètres possibles pour chaque modèle.

Par exemple, un $B_{0/1} = 0,5$ signifie que les données sont deux fois plus probables sous M_1 que sous M_0 , et donc on pourrait préférer le modèle sous la forme :

$$Y = f(X_1, \theta_1) + \epsilon$$

Une remarque importante est que $B_{1/0}$ est simplement l'inverse de $B_{0/1}$, soit :

$$B_{1/0} = \frac{1}{B_{0/1}}$$

Après avoir calculé les facteurs de Bayes, les probabilités a posteriori pour les modèles peuvent être obtenues. Si l'on veut estimer la probabilité qu'un modèle soit correct après avoir observé les données, on utilise la formule suivante :

$$P(M_1 \mid \text{données}) = \frac{B_{1/0} \cdot P(M_1)}{P(M_0) + B_{1/0} \cdot P(M_1)}$$

Par exemple, si la probabilité a priori $P(M_1)$ est de 0,7 (indiquant une croyance forte que M_1 est correct avant d'observer les données) et que le facteur de Bayes calculé $B_{1/0}$ est de 5, cela renforcerait considérablement la probabilité a posteriori en faveur de M_1 . Il est possible de comparer plusieurs modèles en calculant les facteurs de Bayes entre chaque paire de modèles, grâce à la transitivité du facteur de Bayes.

Par exemple, pour comparer trois modèles M_0 , M_1 , et M_2 , on peut calculer :

$$B_{0/1} = \frac{M_0(x)}{M_1(x)} \quad \text{et} \quad B_{1/2} = \frac{M_1(x)}{M_2(x)}$$

Puis, on combine les facteurs pour comparer M_0 à M_2 :

$$B_{0/2} = B_{0/1} \times B_{1/2} = \frac{M_0(x)}{M_2(x)}$$

Ce qui nous donnera :

- Si $B_{0/1} > 1$, cela signifie que les données soutiennent davantage M_0 par rapport à M_1 .
- Si $B_{1/2} > 1$, cela signifie que les données soutiennent davantage M_2 par rapport à M_1 .

- On aura donc dans ce cas $B_{0/2} > 1$, c'est à dire que les données soutiennent davantage M_2 .

Des valeurs élevées de $B_{0/1}$ ou $B_{1/0}$ indiquent une preuve plus forte en faveur de l'un ou l'autre modèle.

A priori : [14][15][16][17][18]

Le choix des a priori est un élément central de l'inférence bayésienne et doit être fait avec soin. Il s'agit d'un compromis entre l'incorporation des connaissances préalables et la flexibilité pour laisser les données guider l'analyse. Dans le contexte de l'écotoxicologie, où les décisions sont souvent prises avec des données limitées et incertaines, l'utilisation judicieuse des a priori permet d'améliorer la précision des estimations et la robustesse des conclusions.

1. A priori informatifs

Un a priori informatif est une distribution qui incorpore des connaissances préexistantes ou des croyances fortes sur les paramètres. Ces connaissances peuvent provenir d'études antérieures, d'expertise dans le domaine, ou de résultats d'expériences précédentes. Par exemple, si plusieurs études ont déjà montré que la probabilité de mortalité sous un certain traitement est faible, un a priori informatif pourrait être une distribution bêta très concentrée autour de faibles valeurs. L'avantage de cet a priori est qu'il permet d'intégrer de manière explicite ces connaissances dans l'analyse, ce qui peut conduire à des conclusions plus robustes, surtout lorsque les nouvelles données sont limitées.

2. A priori non informatifs

Les a priori non informatifs, également appelés a priori neutres ou a priori plats, sont des distributions qui n'incorporent aucune information spécifique sur les paramètres. Ils sont utilisés pour minimiser l'influence de l'a priori sur les résultats, permettant aux données de jouer un rôle dominant. Un exemple classique est l'utilisation d'une distribution uniforme ou d'une distribution bêta(1,1). Ce type d'a priori est utile lorsqu'il y a peu ou pas de connaissances préalables sur les paramètres ou lorsque l'on souhaite que les données soient le principal moteur de l'inférence.

3. A priori partiellement informatifs

Les a priori partiellement informatifs sont un compromis entre les a priori informatifs et non informatifs. Ils intègrent une certaine quantité d'information préalable tout en laissant de la place aux données pour influencer l'inférence. Par exemple, dans une situation où l'on a des connaissances approximatives sur les paramètres mais où l'on souhaite rester ouvert à des résultats inattendus, un a priori partiellement informatif peut être une distribution bêta avec des paramètres modérément concentrés. Cela permet de capturer les connaissances préexistantes tout en restant suffisamment flexible pour être mis à jour de manière significative par les nouvelles données.

4. A priori conjugués

Un a priori conjugué est un a priori qui, lorsqu'il est combiné avec une certaine fonction de vraisemblance, produit une distribution a posteriori de la même famille que l'a priori. Par exemple, une distribution bêta est conjuguée à la distribution binomiale, ce qui signifie que si l'a priori est une distribution bêta et que la vraisemblance est binomiale, la distribution a posteriori sera également une distribution bêta. L'utilisation d'a priori conjugués simplifie souvent les calculs et permet de dériver facilement les distributions a posteriori. Cette propriété est particulièrement utile lorsque l'on doit

mettre à jour les croyances de manière itérative avec l'arrivée de nouvelles données.

5. A priori empiriques

Les a priori empiriques sont dérivés directement des données. Au lieu d'être spécifiés indépendamment des données, ils sont construits en utilisant des statistiques résumées ou des estimations provenant des données elles-mêmes. Par exemple, dans un contexte où l'on a une grande quantité de données historiques, on pourrait utiliser la moyenne et la variance des données pour construire une distribution a priori qui reflète ces caractéristiques. Les a priori empiriques permettent d'adapter les analyses aux données spécifiques tout en conservant une base bayésienne.

Une fois que le modèle statistique et le type d'a priori ont été spécifiés, l'étape suivante consiste à examiner les données collectées et à les utiliser pour mettre à jour les croyances via la distribution a posteriori. Cette mise à jour prend en compte à la fois les données et les a priori, et permet de produire des inférences qui sont enrichies par les connaissances antérieures tout en étant guidées par les nouvelles observations.

Facteur de Bayes utilisant Metropolis-Hastings (MH)

Le facteur de Bayes est utilisé pour comparer deux modèles statistiques M_0 et M_1 en termes de probabilité que chaque modèle soit le générateur des données observées. Le facteur de Bayes est défini comme le rapport des vraisemblances marginales des deux modèles :

$$B_{0/1} = \frac{P(D | M_0)}{P(D | M_1)}$$

où $P(D | M_0)$ est la vraisemblance marginale des données D sous le modèle M_0 , calculée comme :

$$P(D | M_0) = \int_{\theta_0} P(D | \theta_0, M_0) P(\theta_0 | M_0) d\theta_0$$

L'intégrale ci-dessus est souvent très complexe analytiquement, ce qui justifie l'utilisation de méthodes numériques telles que l'algorithme de Metropolis-Hastings. L'algorithme de Metropolis-Hastings (MH) est utilisé pour générer une chaîne de Markov dont les états sont distribués selon la densité postérieure $P(\theta | D, M)$. Une fois la chaîne construite, on peut utiliser les échantillons pour estimer la vraisemblance marginale.

Pour un modèle donné M , la densité postérieure non normalisée est donnée par :

$$P(\theta | D, M) \propto P(D | \theta, M) P(\theta | M)$$

En utilisant Metropolis-Hastings, on génère une chaîne $\theta_1, \theta_2, \dots, \theta_n$ qui suit cette distribution postérieure.

La vraisemblance marginale peut ensuite être estimée en prenant la moyenne des densités postérieures non normalisées évaluées aux échantillons générés par MH :

$$P(D | M) \approx \frac{1}{N} \sum_{i=1}^N \frac{P(D | \theta_i, M) P(\theta_i | M)}{q(\theta_i)}$$

où $q(\theta_i)$ est la distribution de proposition utilisée dans MH.

Le facteur de Bayes entre deux modèles M_0 et M_1 est alors estimé comme le ratio des vraisemblances marginales :

$$B_{0/1} \approx \frac{\frac{1}{N_0} \sum_{i=1}^{N_0} \frac{P(D|\theta_{0,i}, M_0)P(\theta_{0,i}|M_0)}{q(\theta_{0,i})}}{\frac{1}{N_1} \sum_{i=1}^{N_1} \frac{P(D|\theta_{1,i}, M_1)P(\theta_{1,i}|M_1)}{q(\theta_{1,i})}}$$

Cela donne une mesure de la force de l'évidence en faveur de l'un des modèles par rapport à l'autre. Une approche simplifiée pour le calcul du facteur de Bayes, basée sur la LogVraisemblance des modèles, a été réalisée afin d'approximer la vraisemblance marginale lorsque les prioris sont plats ou uniformes. Cette méthode revient à comparer les vraisemblances exponentielles des deux modèles en utilisant la relation suivante :

$$B_{0/1} \approx \exp(\ln L(\hat{\theta}_0) - \ln L(\hat{\theta}_1))$$

où $\ln L(\hat{\theta}_0)$ et $\ln L(\hat{\theta}_1)$ représentent les log-Vraisemblances des modèles comparés.

Ce choix simplifie le calcul des vraisemblances marginales, car les intégrations sur les prioris se réduisent à des constantes, qui s'annulent lors du calcul des facteurs de Bayes. Ainsi, l'évaluation relative des modèles devient plus directe et moins constitutionnellement intensive.

Par ailleurs, l'Autocorrelation Function (ACF) étant une mesure statistique qui décrit la corrélation entre les valeurs d'une série temporelle à différents temps décalés, elle a été utilisée pour évaluer l'autocorrélation entre les échantillons successifs de la chaîne. Où, l'ACF pour un décalage (lag) k est définie comme :

$$\text{ACF}(k) = \frac{\text{Cov}(X_t, X_{t+k})}{\text{Var}(X_t)}$$

où X_t est la valeur de la chaîne à l'itération t , $\text{Cov}(X_t, X_{t+k})$ est la covariance entre les valeurs à deux temps décalés par k , et $\text{Var}(X_t)$ est la variance de la chaîne. L'ACF a donc été utilisée pour diagnostiquer la qualité de l'échantillonnage effectué par l'algorithme de Metropolis-Hastings:

- **Faible autocorrélation:** Une faible autocorrélation indique que les échantillons successifs sont peu dépendants, ce qui est souhaitable pour obtenir une bonne exploration de l'espace des paramètres.
- **Autocorrélation élevée:** Si l'ACF montre une forte autocorrélation pour des décalages k faibles, cela signifie que la chaîne met plus de temps à explorer l'espace des paramètres, ce qui peut nécessiter un échantillonnage plus long ou un ajustement de la proposition dans l'algorithme.

Un diagnostic courant est de tracer l'ACF pour différentes valeurs de k afin de vérifier que la chaîne converge rapidement vers zéro. Si ce n'est pas le cas, cela peut indiquer que les échantillons sont trop corrélés, et des modifications de la stratégie d'échantillonnage peuvent être nécessaires.

5.5.1 Vérification des Modèles

Après l'application des modèles statistiques, il est nécessaire de vérifier que les modèles sélectionnés sont correctement ajustés aux données. Cette vérification peut inclure des tests de résidus et des vérifications des hypothèses du modèle. Les histogrammes des résidus, les

graphes de normalité, et les tests de conformité des hypothèses sous-jacentes sont autant d'outils pour s'assurer que le modèle choisi est non seulement statistiquement justifié, mais aussi biologiquement pertinent. Si un modèle ne s'ajuste pas correctement, il peut être nécessaire de le réajuster ou d'explorer des alternatives.

En suivant ces étapes, les données préparées sont prêtes pour une analyse statistique robuste, ce qui permet d'obtenir des résultats fiables et scientifiquement valides pour l'évaluation de la toxicité. Cette rigueur dans la préparation des données est essentielle pour garantir que les conclusions tirées des études soient valides et applicables dans un contexte réglementaire.

5.6 Détermination des valeurs de toxicité

Les valeurs de toxicité, telles que la concentration à laquelle un certain pourcentage d'organismes montre une réponse (par exemple, EC_{50}), sont cruciales pour évaluer les effets des substances chimiques sur les organismes vivants. Ces valeurs peuvent être déterminées à partir de différents types d'essais d'écotoxicité, notamment les essais limite et les essais dose-réponse. **Les essais limite** permettent de déterminer si une substance a un effet observable sur les organismes à une concentration donnée. Si aucun effet n'est observé à cette concentration, la substance est souvent considérée comme non toxique à cette concentration. Par contre, **les essais dose-réponse** fournissent des informations plus détaillées, en permettant de tracer une courbe dose-réponse à partir de laquelle on peut déterminer des valeurs clés telles que la concentration efficace médiane (EC_{50}), la concentration létale médiane (LC_{50}), ainsi que d'autres paramètres critiques.

Outre l' EC_{50} et la LC_{50} , d'autres valeurs de toxicité sont fréquemment utilisées pour une évaluation plus fine :

- **EC_{10} et EC_{20} cf figure 5.1:** Ces valeurs représentent les concentrations à lesquelles respectivement 10% et 20% des individus d'une population exposée montrent une réponse spécifique. Ces valeurs sont particulièrement utiles dans les évaluations de toxicité chronique, où des **effets sublétaux** à faibles concentrations sont étudiés. L' EC_{10} , en particulier, est souvent utilisé pour déterminer la PNEC (Predicted No Effect Concentration), en raison de son rôle dans l'évaluation des effets à faible niveau.
- **NOEC (No Observed Effect Concentration):** C'est la plus haute concentration à laquelle aucun effet nocif n'est observé. Cette valeur est cruciale pour identifier des seuils de toxicité chronique.
- **LOEC (Lowest Observed Effect Concentration):** Il s'agit de la plus faible concentration à laquelle un effet nocif est observé. Elle est souvent utilisée en complément du NOEC pour caractériser les effets sublétaux ou chroniques.
- **PNEC (Predicted No Effect Concentration):** C'est la concentration en dessous de laquelle on s'attend à ce qu'il n'y ait aucun effet adverse sur l'environnement. Cette valeur est utilisée pour évaluer les risques potentiels des substances chimiques pour les écosystèmes aquatiques, terrestres, et autres.

Ces valeurs sont essentielles dans la démarche d'évaluation des risques écotoxicologiques. Par exemple, l' EC_{10} et l' EC_{20} sont fréquemment utilisés pour caractériser les effets à long terme de faibles expositions, tandis que la PNEC est souvent comparée aux concentrations environnementales prédites (PEC) pour évaluer le risque posé par une substance dans un environnement donné. Les essais et les valeurs obtenues sont utilisés pour établir des **valeurs de référence écotoxicologiques** qui permettent de caractériser le risque environnemental des substances chimiques. Dans ce rapport, les EC_x ont été déterminées à partir des modèles

de régression non linéaire employés. Ces modèles permettent d'estimer avec précision la concentration à laquelle un certain pourcentage de la population répond à la substance testée, ce qui est essentiel pour évaluer les risques écotoxicologiques associés au contaminant étudié.

5.7 Analyse de la significativité des différences observées dans les ECx par la méthode de permutation [23]

La théorie de la permutation a été intégrée dans cette méthodologie, renforçant ainsi l'approche non paramétrique employée.

C'est une méthode statistique **non paramétrique** permettant de tester des hypothèses en comparant la statistique observée à une distribution obtenue par **permutation aléatoire** des données sous l'hypothèse nulle. Concrètement, cette méthode consiste à définir une hypothèse nulle (par exemple, H_0 : "les deux groupes sont indépendants") et à permuter aléatoirement les étiquettes des groupes dans les données. Pour chaque permutation, une statistique de test T^* est calculée, comme le **coefficient de corrélation de Pearson** ou le **tau de Kendall**, afin de construire une distribution empirique \hat{F}_{T^*} sous H_0 .

La p-valeur est ensuite obtenue en comparant la statistique de test observée T_{obs} à cette distribution :

p-valeur = $P(T^* \geq T_{\text{obs}})$, ce qui permet de décider si l'hypothèse nulle doit être rejetée ou non. Les tests par permutation offrent une flexibilité importante, car ils ne nécessitent pas d'hypothèses strictes sur la distribution des données, ce qui les rend particulièrement utiles lorsque les hypothèses des tests paramétriques, telles que **la normalité** ou **homoscédasticité**, ne sont pas vérifiées. Cette approche permet également un contrôle rigoureux du **taux d'erreur de type I**, ce qui est essentiel dans le cas de petits échantillons ou lorsque la distribution des données est incertaine.

5.8 Intervalles de confiance des ECx par Bootstrap

L'approche par **bootstrap**[10] est une méthode statistique particulièrement puissante en écotoxicologie, permettant d'estimer la variabilité des paramètres tels que les ECx (par exemple, EC_{10} , EC_{20} , EC_{50}) en **rééchantillonnant** plusieurs fois les données observées. Cette technique de rééchantillonnage est cruciale lorsque les **hypothèses de normalité des résidus ou homoscédasticité sont remises en question**, ou lorsque les **distributions des erreurs sont inconnues ou non normales**, ce qui est souvent le cas dans les modèles non linéaires. Le bootstrap permet ainsi d'estimer la distribution empirique des paramètres sans faire d'hypothèses strictes sur la distribution des erreurs, rendant cette méthode particulièrement robuste face aux valeurs aberrantes et aux distributions de données non standards. En rééchantillonnant **avec remise** à partir de l'échantillon initial, on peut générer un grand nombre d'échantillons bootstrapés, chacun permettant d'ajuster un modèle dose-réponse pour estimer les ECx correspondants. Ces estimations forment ensuite une distribution empirique des ECx, qui peut être utilisée pour calculer des intervalles de confiance, souvent définis par les percentiles de cette distribution bootstrap.

La forme générale d'un intervalle de confiance bootstrap pour un estimateur \hat{EC}_x est donc donnée par :

$$[\hat{EC}_{x\alpha/2}^*, \hat{EC}_{x1-\alpha/2}^*],$$

où $\hat{EC}_{x\alpha/2}^*$ et $\hat{EC}_{x1-\alpha/2}^*$ sont les percentiles $\alpha/2$ et $1 - \alpha/2$ de la distribution bootstrap des \hat{EC}_x^* . De plus, le bootstrap permet également d'estimer le biais et la variance de l'estimateur, fournissant ainsi des informations précieuses sur la précision et la stabilité des estimations obtenues. Un aspect important du bootstrap est l'estimation par **injection**, où l'idée est de remplacer la loi inconnue de la population par sa fonction de répartition empirique, notée \hat{F}_n , qui est construite à partir des données observées. En pratique, les échantillons bootstrap sont générés en tirant avec remise selon \hat{F}_n , ce qui assure que les nouvelles variables X_i^* , conditionnées à l'échantillon initial, suivent une loi proche de la loi réelle. Cette approche permet de simuler la distribution de l'estimateur sans recourir à des hypothèses paramétriques rigides, offrant ainsi une méthode flexible pour évaluer la fiabilité des estimations dans des contextes complexes. En écotoxicologie, cette technique est souvent utilisée pour estimer les propriétés des ECx, tels que le biais, la variance, et les intervalles de confiance, en utilisant des échantillons rééchantillonnés pour simuler la distribution de l'estimateur. Ainsi, le bootstrap constitue un outil essentiel pour réaliser des inférences statistiques robustes là où les méthodes traditionnelles pourraient être inappropriées ou insuffisantes.

Une autre méthode non paramétrique, tout aussi puissante mais appliquée dans un contexte différent, est l'approche par permutation. Alors que le bootstrap se concentre sur l'estimation des intervalles de confiance et des biais en s'appuyant sur des rééchantillonnages avec remplacement, l'approche par permutation est particulièrement judicieuse pour tester des hypothèses en réarrangeant aléatoirement les données, offrant ainsi une solution robuste et flexible pour évaluer la significativité des résultats sans supposer de distribution spécifique.

5.9 Limites et contraintes liées aux données

L'exploitation des données écotoxicologiques comporte plusieurs limites et contraintes. Ces limitations incluent la variabilité biologique, les incertitudes expérimentales et les défis liés à l'extrapolation des résultats de laboratoire aux conditions environnementales réelles. De plus, certaines données peuvent être incomplètes ou présenter des biais, ce qui nécessite des adaptations méthodologiques pour assurer la fiabilité des conclusions tirées. Certains documents guides de tests réglementaires (Essai 201[27] de toxicité aiguë pour les algues, l'Essai 202 cite Test Guideline No 202 de toxicité aiguë pour les *Daphnia magna*, Essai 210[32] de toxicité prolongée pour les poissons et l'Essai 211[33] de reproduction chez les *Daphnia magna*) exigent la détermination d'une NOEC pour une ou plusieurs réponses, il est donc essentiel de comprendre les modèles statistiques disponibles pour cet objectif. Cela implique une compréhension des conceptions expérimentales qui soutiennent ces tests, ainsi que des propriétés telles que la **puissance**, les **sensibilités** et les **limitations** qui déterminent leur utilité.

Il existe plusieurs tests appropriés pour l'analyse des réponses continues, quantales, discrètes et ordinales provenant d'études écotoxicologiques. Pour la détermination des NOEC, ils sont divisés en deux catégories : les tests paramétriques et non paramétriques. Tous ces tests reposent sur des modèles ayant des critères spécifiques, et l'utilisation de données qui ne les respectent pas peut conduire à des conclusions erronées. Il est donc important d'explorer des outils formels et informels pour déterminer si un ensemble de données répond aux exigences de normalité et d'homogénéité de variance, et s'il est compatible avec une relation dose-réponse monotone (*i.e* une relation où la réponse change systématiquement dans une seule direction à mesure que la concentration augmente ou diminue). Par ailleurs, il est important de rappeler que l'unité de base d'analyse est **le réplicat**, pas l'animal ou la

plante individuelle, et que les organismes au sein de chaque réplicat sont interdépendants.

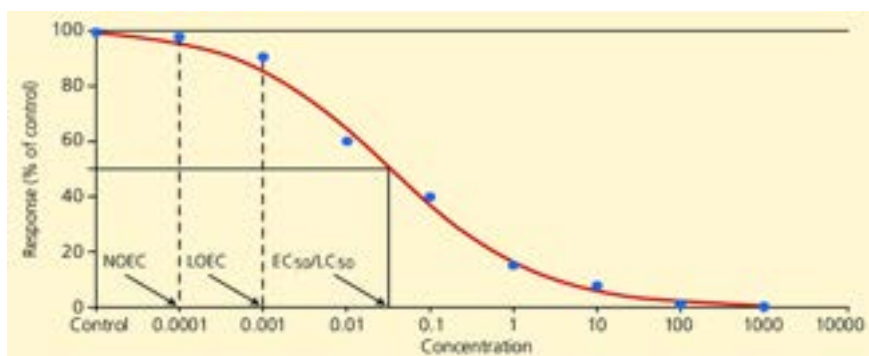


Figure 5.1: Illustration NOEC, LOEC, ECx sur courbe dose réponse [21]

6 Résultats et discussion

Cette étude s'appuie sur l'essai 201 de l'OCDE, qui établit des lignes directrices pour l'évaluation de la toxicité aiguë des substances chimiques sur les algues et les cyanobactéries aquatiques. L'essai 201 est une méthode standardisée utilisée pour déterminer l'impact de différentes concentrations de substances chimiques sur la croissance et la santé des populations algales sur une période de 72 heures. L'objectif principal de cet essai est de déterminer la concentration efficace médiane (EC_{50}), c'est-à-dire la concentration à laquelle la substance testée inhibe la croissance des algues de 50%.

6.1 Codage et Transformation des Données :

Pour harmoniser les données avec l'application Rshiny, les données nettoyées doivent souvent être transformées ou codées. Cela implique de mettre les données en forme pour les adapter aux exigences des analyses statistiques prévues par type de test d'écotoxicité. Par exemple, les catégories de concentration ou de traitement peuvent être codifiées en valeurs numériques pour faciliter l'application des tests statistiques.

La transformation des données en **format long** est essentielle pour simplifier l'analyse des données écotoxicologiques, facilitant ainsi l'analyse statistique, la visualisation, et la gestion des réplicats expérimentaux. Ce format améliore la compatibilité interne avec l'application, permet l'ajout de nouvelles variables et simplifie la comparaison entre différents groupes ou traitements. **Exemple de format long pour le Test OCDE 201 :**

```
Replicat,concentration,response
Replicat1,Control,61.2
Replicat1,Control_Solvent,63.3
Replicat1,0.1,58.2
Replicat1,0.15,48.6
...
```

Dans ce format long, chaque ligne correspond à une observation unique pour un réplicat donné, avec les colonnes représentant le réplicat, la concentration testée, et la réponse mesurée.

Pour la **Ligne directrice OCDE 201** qui concerne l'inhibition de la croissance des algues et des cyanobactéries. Les données doivent être structurées pour permettre une comparaison

claire entre les groupes témoins et les groupes traités, en codifiant par exemple les concentrations de l'agent testée en valeurs numériques.

Et le format d'entrée attendu par l'application est:

```
Replicat,Control,Control_Solvent,0.1,0.15,0.23,0.34,0.51,0.76,1.14
Replicat1,61.2,63.3,58.2,48.6,48.5,44.1,25.5,2.5,1.2
Replicat2,63.1,62.3,60.8,47.5,47.9,46.4,24.3,2.3,1.3
...
```

Pour la **Ligne directrice OCDE 202** qui concerne la toxicité aiguë sur *Daphnia*, les données de mortalité sont souvent présentées sous forme de pourcentages. Ces données doivent être organisées de manière à pouvoir être facilement analysées pour déterminer la concentration létale médiane (CL_{50}).

Et le format d'entrée attendu par l'application est :

```
Replicat,time,concentration,total,immobilized
1,24,0,20,0
2,24,5,20,0
...
```

Lorsqu'on introduit un test qui n'est pas répertorié dans les essais standards de l'application Rshiny, il est nécessaire de mettre en forme le jeu de données sous le format de données compatible, comme le format CSV ou TXT. L'application fournit un exemple du format attendu pour guider l'utilisateur dans la mise en forme des données. Il est important de suivre ces formats pour assurer la compatibilité et la précision des analyses.

Pour un nouveau test, il faut structurer les données brutes en CSV ou txt comme suit :

```
Replicat,concentration,response
Replicat1,0.1,58.2
Replicat1,0.15,48.6
...
```

6.2 Préparation des données pour l'analyse

La préparation des données est une étape cruciale pour assurer la validité des analyses écotoxicologiques. Conformément aux directives de l'OCDE 54, les données collectées ont été préparées minutieusement avant toute analyse statistique (cf méthodologie). Pour illustrer ce processus, nous allons prendre l'exemple d'une étude sur l'effet toxique d'un pesticide sur une espèce d'algues d'eau douce (Essai 201).

6.2.1 Visualisation

Dans l'analyse des données **dose-réponse**, une première étape cruciale consiste à inspecter visuellement les données. Cette étape ne s'applique pas directement aux **essais limite**, qui visent uniquement à déterminer la toxicité à une concentration spécifique. Pour les données **continues**, il est utile de tracer les réponses individuelles ainsi que les moyennes de groupe en fonction de la concentration. Pour les données **quantales**, les fréquences observées de réponse sont tracées en fonction de la dose.

Ces graphiques permettent d'évaluer si les données montrent une relation dose-réponse cohérente et peuvent révéler des valeurs aberrantes, c'est-à-dire des points de données qui s'écartent significativement des attentes ou du schéma général observé.

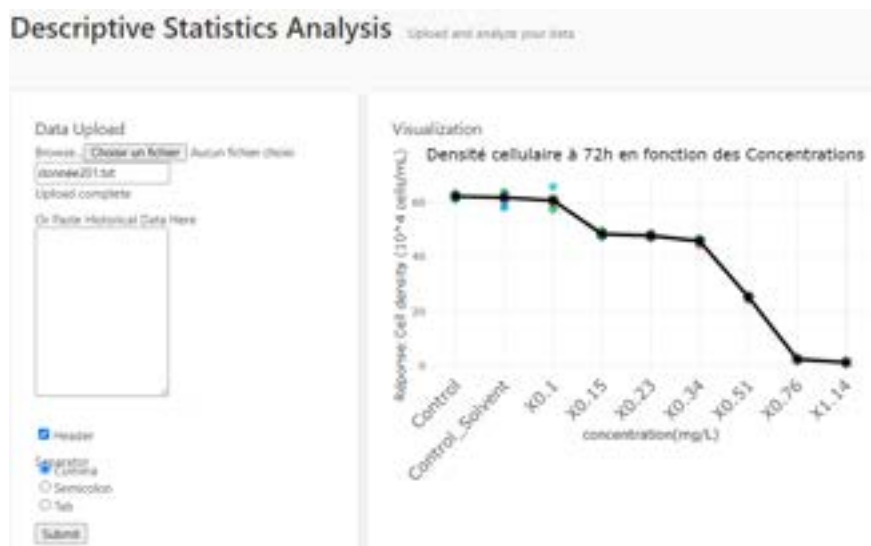


Figure 6.1: Diagramme Essai 201

La Figure 6.1 présente, à travers l'application développée, un diagramme de dispersion illustrant les données de densité cellulaire dans une expérience d'inhibition de la croissance chez les algues (Essai OCDE 201). Les réponses moyennes aux différents traitements sont représentées par des étoiles et reliées par des segments de ligne, ce qui permet de visualiser les tendances entre les traitements et les réponses. Ce type de graphique permet de comparer visuellement la dispersion des données au sein des traitements et de rechercher des preuves de variance inégale. Les moyennes reliées facilitent l'évaluation de la cohérence d'une réponse monotone en fonction de la concentration, ou, à l'inverse, révèlent des réponses non monotones, indiquant des variations plus complexes en fonction de la dose. Par exemple, dans la figure présentée, bien que l'on observe de légers pics aux concentrations de 0,1 et 0,34 mg/L, la tendance générale reste interprétable comme une réponse monotone à la dose, car les fluctuations sont mineures par rapport à la tendance globale. Cependant, si la réponse n'était pas monotone, ce type de visualisation permettrait de détecter ces irrégularités. Cette visualisation initiale permet d'identifier préalablement les tendances et les anomalies potentielles, telles que des valeurs aberrantes qui pourraient biaiser l'analyse ultérieure. La finalité de cette inspection est d'informer une analyse descriptive, qui synthétise les données avant de passer aux phases d'analyse et de traitement des données. Le diagramme de dispersion peut également être utile pour identifier ces valeurs, en montrant des déviations notables par rapport aux tendances attendues, qu'elles soient monotones ou non.

La Figure 6.2 illustre la dispersion des données de l'essai dose-réponse n°201 à l'aide de boxplots pour chaque concentration. Ces boxplots permettent une analyse visuelle de la distribution des résultats, en mettant en évidence la dispersion des données autour des médianes et en identifiant les valeurs aberrantes. Les points situés en dehors des moustaches des boxplots indiquent des outliers potentiels, et cette visualisation offre également la possibilité de détecter un éventuel effet solvant.

6.2.2 Gestion des outliers

À la lumière des observations précédentes, les données ont été vérifiées afin de garantir leur adéquation aux analyses statistiques prévues. Les données initiales ont effectivement révélé la présence d'outliers (voir tableau 1), notamment pour certaines concentrations spécifiques. Mais aucune valeur aberrante identifiée n'a été exclue de l'analyse à ce stade, car aucune

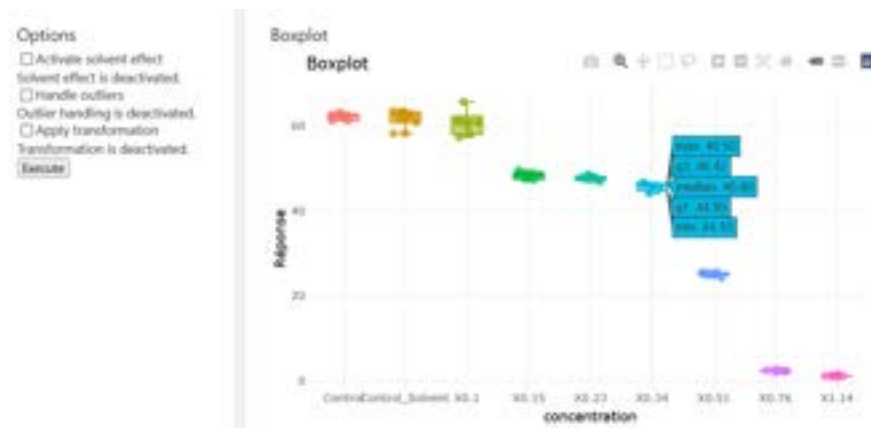


Figure 6.2: Boxplot Essai 201

Concentrations	Outliers
0	58.2
0.1	65.6
0.51	24.3
1.14	1.3

Table 2: Outliers détectés par la méthode de Tukey

erreur expérimentale détectée ne le justifiait. Ce processus vise à maintenir l'intégrité statistique des données tout en assurant que les résultats reflètent fidèlement les effets réels des concentrations testées. Après la visualisation, les données sont donc traitées pour garantir leur adéquation aux analyses statistiques prévues. Le nettoyage des données est une étape cruciale pour assurer leur qualité et fiabilité, en particulier en ce qui concerne la gestion des valeurs aberrantes et des incohérences. Les valeurs aberrantes, définies comme des observations qui dévient notablement des autres dans des conditions similaires (c'est-à-dire qu'elles semblent se situer en dehors de la dispersion habituelle des données au sein d'un groupe de traitement) ([4]), peuvent biaiser les résultats en augmentant la probabilité de faux positifs et faux négatifs ([5]). Ces valeurs sont particulièrement indésirables dans les groupes de concentration car elles peuvent altérer les résultats de manière significative. Pour confirmer statistiquement une valeur aberrante, des tests tels que le test de Tukey et le test de Grubbs peuvent être appliqués. Cependant, pour leur suppression, une justification biologique est nécessaire [24] [25] [26]. Deux cas se présentent alors :

- **Si la valeur aberrante résulte d'une déviation expérimentale naturelle ou inexpliquée**, et que la taille de l'échantillon est petite, il est recommandé de conserver la donnée conformément aux recommandations de l'OCDE 54.
- **Si la valeur aberrante peut être attribuée à une erreur expérimentale identifiable**, elle peut être exclue de l'analyse pour éviter de fausser les résultats, en accord avec l'OCDE 54.

Les valeurs aberrantes dans les données **continues** peuvent biaiser les estimations des effets et augmenter la variance résiduelle, ce qui réduit la puissance des tests statistiques et la précision des paramètres estimés. Par conséquent, si des raisons peuvent être trouvées pour

expliquer les valeurs aberrantes, il est préférable de les supprimer de l'analyse.

Dans les données **quantales**, elles affectent généralement un groupe de traitement entier, ce qui peut indiquer des erreurs systémiques dans la mise en place de l'expérience. Elles peuvent fausser les résultats en entraînant des estimations des effets soit trop faibles, soit trop élevées, augmentant ainsi la variance résiduelle ou le bruit. Cela réduit la puissance des tests statistiques et la précision des paramètres estimés. Les valeurs aberrantes affectent donc davantage l'analyse des données quantales que celle des données continues, car dans les données quantales, les valeurs aberrantes concernent souvent un groupe de traitement entier, puisque la réponse déviante individuelle ne peut être détectée sur une base binaire (oui/non).

Dans les données de **comptage**, les valeurs aberrantes peuvent également avoir un impact significatif, en particulier lorsqu'elles sont présentes en faible nombre. Ces données sont souvent analysées à l'aide de modèles de régression pour données de comptage, tels que les modèles de Poisson ou binomiaux négatifs. Une valeur aberrante dans un jeu de données de comptage peut entraîner des estimations biaisées des paramètres du modèle, ce qui pourrait nécessiter des techniques de correction ou de robustesse, comme l'utilisation de modèles qui tiennent compte de la surdispersion ou des poids ajustés pour réduire l'influence des valeurs extrêmes.

Par ailleurs, les observations influentes (c'est-à-dire une valeur inhabituellement grande ou petite d'une variable explicative), qui peuvent avoir une grande influence sur une analyse, doivent également être prises en compte.

6.2.3 Agrégation des Données :

L'étape suivante a consisté à vérifier l'effet d'un solvant lorsqu'il a été nécessaire d'en employer un pour le test d'écotoxicité. Des tests statistiques ont donc été appliqués pour comparer le groupe témoin sans solvant (contrôle) avec le groupe témoin avec solvant (contrôle + solvant). Pour illustrer cette analyse, nous avons utilisé le test de Wilcoxon (non paramétrique) afin de comparer les médianes de deux groupes distincts. Les résultats de ce test indiquent que l'effet du solvant est statistiquement non significatif. De manière similaire, le test t de Student (paramétrique) a été employé pour comparer les moyennes des mêmes groupes. Les résultats obtenus montrent également qu'il n'y a pas de différence statistiquement significative entre le groupe témoin et le groupe témoin avec solvant. Ces analyses confirment ainsi les observations faites lors de la visualisation des données, à savoir que le solvant utilisé dans cette étude n'a pas d'effet sur la densité cellulaire des algues. Par conséquent, les colonnes "contrôle" et "contrôle + solvant" ont été fusionnées en une seule colonne intitulée "0", qui sera utilisée pour l'interprétation des résultats finaux. Cette approche vise à renforcer la robustesse et la cohérence de l'analyse, tout en évitant la complexité inutile liée à la gestion séparée des données avec et sans solvant.

Pour vérifier en pratique l'existence de cet effet solvant (*i.e* que les deux contrôles sont différents), il est nécessaire de comparer les données issues du groupe témoin à celles du groupe témoin solvant. Cette comparaison peut être réalisée visuellement, afin d'identifier rapidement des différences potentielles dans la distribution des données entre les deux groupes, souvent à l'aide de boxplots, ainsi que statistiquement à l'aide de tests appropriés.

Lorsque les données suivent une distribution normale et que les variances des deux groupes sont égales (homogénéité des variances), un test t de Student (pour échantillons indépendants) est utilisé pour comparer les moyennes des deux groupes. La statistique de test, notée t , est calculée à partir des différences entre les moyennes des groupes, la variance de

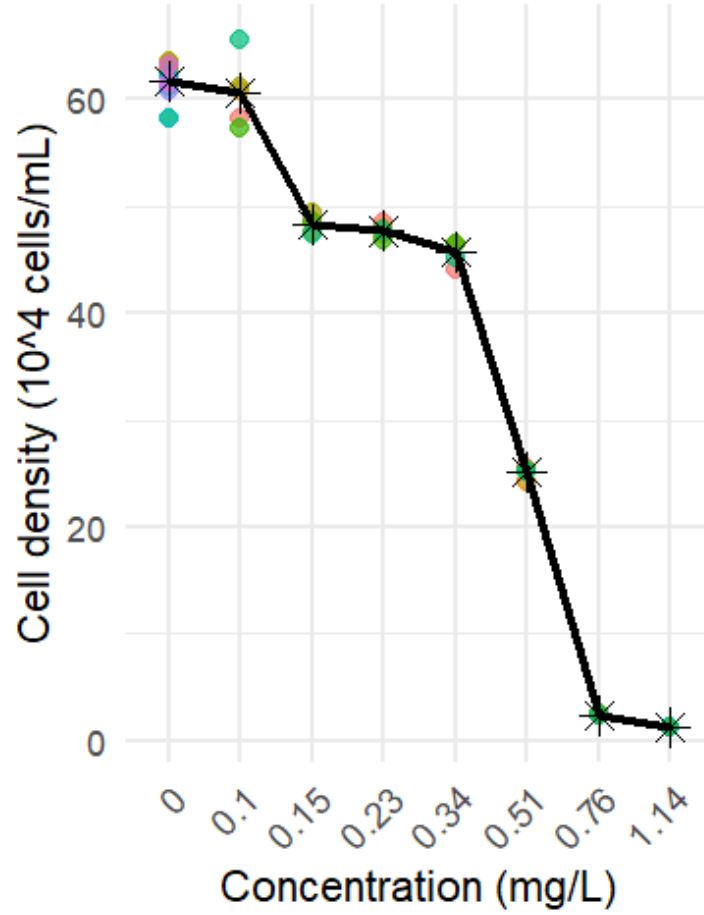


Figure 6.3: Densité cellulaire à 72h en fonction des concentrations : concaténation

chaque groupe, et la taille de chaque échantillon. L'hypothèse nulle (H_0) de ce test est que les moyennes des groupes contrôle et contrôle-solvant sont égales.

$$t = \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}}$$

où \bar{X}_i est la moyenne du groupe i , s_i^2 est la variance, et n_i est la taille de l'échantillon.

Et si les données ne suivent pas une distribution normale, un test non paramétrique, tel que le test de Wilcoxon (pour deux échantillons indépendants), est utilisé. Ce test repose sur les rangs des observations et compare les distributions des deux groupes sans faire d'hypothèses sur la forme de la distribution. L'hypothèse nulle (H_0) de ce test est que les distributions des deux groupes sont identiques. Le Test de Mann-Whitney U, est une autre alternative non paramétrique au test t de Student. Il est utilisé lorsque les données ne suivent pas une distribution normale et teste si les deux groupes proviennent de la même distribution. La statistique de test U est définie par :

$$U = \min(U_1, U_2)$$

où

$$U_1 = n_1 n_2 + \frac{n_1(n_1 + 1)}{2} - R_1$$

et

$$U_2 = n_1 n_2 + \frac{n_2(n_2 + 1)}{2} - R_2$$

avec n_1 et n_2 représentant les tailles des deux échantillons, et R_1 et R_2 les sommes des rangs pour les groupes respectifs.

Pour chacun de ces tests, si la valeur p est supérieure à 0,05, l'hypothèse nulle (H_0) ne sera pas rejetée, ce qui indique qu'il n'y a pas de différence significative entre le contrôle et le contrôle-solvant. Dans ce cas, les données des deux contrôles peuvent être regroupées, en les concaténant sous une même colonne '0'. Si la valeur p est inférieure à 0,05, l'hypothèse nulle (H_0) est rejetée, ce qui indique une différence significative, et il est alors nécessaire de déterminer quel groupe contrôle utiliser pour l'analyse ultérieure.

6.3 Analyse des données

6.3.1 Vérification de la normalité et de l'homoscédasticité des données :

Concentration	p-value
0	0.008540683
0.1	0.5391251
0.15	0.5995957
0.23	0.940293
0.34	0.08510793
0.51	0.05367589
0.76	0.006470001
1.14	0.0001309782

Table 3: Résultats du test de Shapiro-Wilk pour chaque concentration

Test	Statistique	p-value
Bartlett	K-squared = 56.609, df = 7	7.148e-10
Levene	F value = 2.6038, df = 7, 42	0.02514

Table 4: Vérification Homoscédasticité

Les hypothèses statistiques des tests appliqués ont été rigoureusement vérifiées. Comme indiqué à l'étape précédente, aucune différence n'a été observée entre les deux groupes de contrôle. Par conséquent, les données concaténées ont été utilisées pour tester les hypothèses d'homogénéité des variances et de normalité à l'aide des tests de Shapiro-Wilk et de Levene. Les résultats ont révélé des violations de ces hypothèses pour certaines concentrations (voir les tableaux 2 et 3, ainsi que la figure 6.4), ce qui justifie l'utilisation ultérieure de méthodes d'inférence statistique non paramétriques.

Les tests de normalité effectués sur les différentes concentrations ont révélé que, bien que l'effet solvant ait été confirmé comme négligeable, les données ne suivent pas une distribution

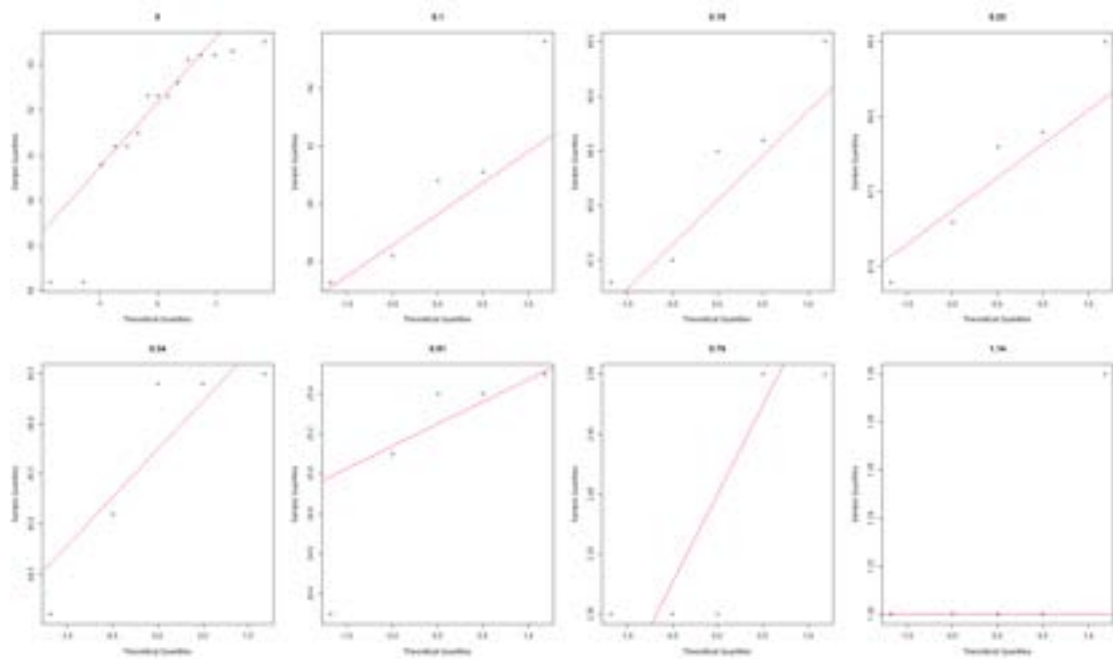


Figure 6.4: QQplot par concentration

normale pour certaines concentrations (cf figure 6.4). Par conséquent, il sera nécessaire d'appliquer des méthodes statistiques non paramétriques dans les analyses ultérieures

En effet, tous les tests et procédures statistiques reposent sur des hypothèses concernant la nature des données à analyser. Parmi celles-ci, deux hypothèses clés sont généralement considérées comme très importantes pour les données continues[3]:

- La distribution des données au sein et entre les groupes de concentration.
Il est important de déterminer si les données de chaque groupe de concentration sont cohérentes avec une distribution normale.
- Et, le cas échéant, si les données des différents groupes de concentration respectent l'hypothèse d'homoscédasticité (entre les concentrations), une hypothèse qui peut être examinée dès la visualisation des boxplots (cf étape de visualisation).

En revanche, pour les données quantales ou de comptage, l'accent est mis sur leur conformité avec une distribution binomiale ou de Poisson et sur la détection éventuelle de surdispersion (c'est-à-dire une variance supérieure à celle attendue). Cependant, étant donné que les tests d'écotoxicité se font souvent sur de petits échantillons, il est préférable d'évaluer la normalité des résidus à l'échelle de l'ensemble de l'étude. Le test de Shapiro-Wilk est couramment utilisé à cette fin, et une transformation logarithmique des données peut être recommandée pour les petits échantillons.[2] L'homoscédasticité est vérifiée à l'aide des tests de Bartlett et de Levene, où une p-value supérieure à 0,05 indique que les variances sont homogènes. Des outils visuels, tels que les histogrammes, les Q-Q plots (cf. figure 6.5) et les scatterplots (nuages de points) (cf. figure 6.6), sont également utilisés pour évaluer la distribution des données continues. Ces outils permettent d'examiner la normalité et l'homoscédasticité, aidant ainsi à déterminer si la dispersion est symétrique et homogène, et à mieux comprendre la distribution des variances:

Pour pallier l'absence de normalité et d'homoscédasticité, la transformation temporaire des données sur une échelle logarithmique (*i.e* $y'_i = \log(y_i)$) est une approche couramment

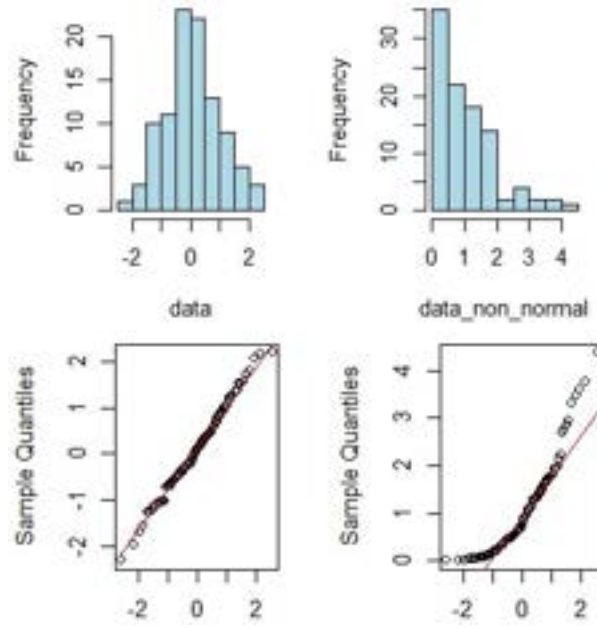


Figure 6.5: normale *vs* non normale

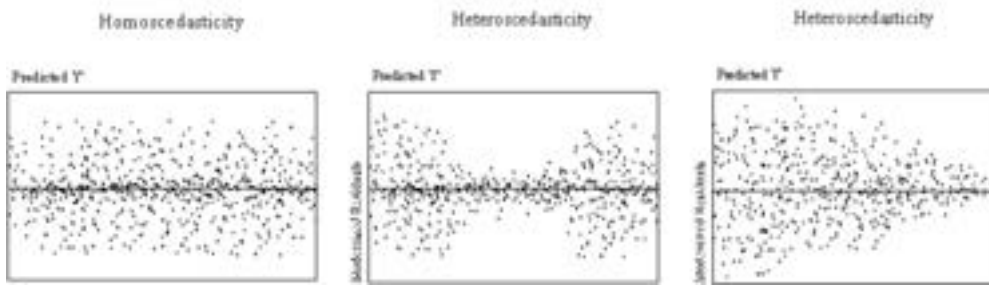


Figure 6.6: scatterplots homoscedasticité vs hétéroscedasticité

utilisée. Cette transformation est particulièrement pertinente lorsque les données présentent une asymétrie marquée ou lorsque la variance des données augmente avec la moyenne (hétéroscedasticité). Car, elle permet d'atténuer ces problèmes en stabilisant la variance et en rapprochant la distribution des données d'une distribution normale[3].

En utilisant par exemple, une transformation Box-Cox définie par l'équation suivante pour les données $y > 0$:

$$y(\lambda) = \begin{cases} \frac{y^\lambda - 1}{\lambda} & \text{si } \lambda \neq 0 \\ \log(y) & \text{si } \lambda = 0 \end{cases}$$

Cette transformation permet d'optimiser la normalisation des données en ajustant le paramètre λ , ce qui peut être particulièrement utile lorsque les écarts par rapport à la normalité et à l'homoscedasticité sont complexes

Pour les **proportions** (données exprimées sous forme de pourcentage ou de fraction d'un total), la distribution sous-jacente est généralement binomiale. Une transformation racine carrée arcsinus ($\arcsin(\sqrt{p})$) est souvent utilisée pour stabiliser la variance, en particulier lorsque les proportions sont très proches de 0 ou de 1, ce qui peut entraîner une variance non constante[3]. Cela permet de réduire la corrélation entre la moyenne et la variance, rendant les données plus adaptées à l'analyse statistique. Pour les **textbf{données de comptage}** (comme par exemple le nombre d'événements ou d'occurrences dans un échantillon), qui suivent souvent une distribution de Poisson, une transformation par la racine carrée peut

être utilisée pour éliminer la relation entre la variance et la moyenne, améliorant ainsi la normalité des résidus et stabilisant la variance. De plus, lorsqu'une surdispersion (variance supérieure à la moyenne) est présente (peut être due à l'hétérogénéité des réponses ou à la présence d'un grand nombre de zéros dans les données), les modèles standard de Poisson peuvent ne pas convenir. Dans ce contexte, des modèles binomiaux négatifs, qui introduisent un paramètre supplémentaire pour modéliser la variance excédentaire, ou les modèles à effets mixtes, qui prennent en compte les variations aléatoires non observées, sont souvent utilisés pour gérer cette surdispersion[3]. Pour les données quantales, la vérification de la normalité et de l'homoscédasticité n'est généralement pas pertinente en raison de la nature binaire de ces données. En effet, les données quantales suivent une distribution binomiale plutôt qu'une distribution normale. Ainsi, plutôt que de chercher à transformer les données quantales pour vérifier la normalité ou l'homoscédasticité, il est plus pertinent d'utiliser directement des modèles et des méthodes statistiques conçus spécifiquement pour les données binaires. Cela permettra de tirer directement des conclusions fiables sur les probabilités d'occurrence des événements d'intérêt. Cependant, l'utilisation de transformations n'est pas sans risque et rend souvent les résultats plus difficiles à interpréter car elles modifient l'échelle d'origine des données, ce qui complique la compréhension des effets biologiques et peut rendre les comparaisons avec d'autres études plus difficiles. De plus, la complexité ajoutée par ces transformations peut conduire à des erreurs d'interprétation si elles ne sont pas bien comprises ou appliquées correctement.[3] Car, les moyennes et intervalles de confiance doivent être retranscrits à l'échelle originale pour faciliter l'interprétation. De plus, la pertinence de rester sous la forme transformée doit être évaluée. Lorsque les hypothèses de normalité et d'homoscédasticité ne sont pas respectées, des méthodes statistiques alternatives peuvent être envisagées. Parmi celles-ci, les **tests non-paramétriques** sont particulièrement utiles, car ils ne reposent pas sur des hypothèses strictes concernant la distribution des données. Contrairement aux tests paramétriques, ces tests n'exigent pas que les données suivent une distribution spécifique, ce qui les rend plus robustes en présence de déviations par rapport aux hypothèses classiques[3]. Ils sont également adaptés lorsque les échantillons sont petits comme c'est majoritairement le cas pour les essais d'écotoxicité[3].

6.3.2 Analyse descriptive

Concentration	Count	Means	Median	STD_DEV	STD_ERR
0	15	61.7933	62.3	1.6875	0.4357
0.1	5	60.6000	60.8	3.2381	1.4481
0.15	5	48.2800	48.5	0.8955	0.4005
0.23	5	47.6800	47.8	0.6099	0.2728
0.34	5	45.7000	46.4	1.0654	0.4765
0.51	5	25.1400	25.4	0.4930	0.2205
0.76	5	2.3800	2.3	0.1095	0.0490
1.14	5	1.2200	1.2	0.0447	0.0200

Table 5: Statistiques pour chaque concentration de substance testée (Test 201 OCDE)

Dans cet exemple d'étude, les algues ont été exposées à plusieurs concentrations du pes-

ticide, avec des réplicats pour chaque concentration. Pour chaque concentration, on peut comparer les réplicats pour vérifier la cohérence des résultats. Par exemple, si les réplicats montrent des taux de croissance similaires, cela indique une bonne répétabilité. Dans ce cas, la moyenne des taux de croissance, l'écart-type et la médiane sont fournis pour chaque concentration (cf table 4). Les résultats montrent une diminution progressive des moyennes et des médianes des taux de croissance des algues avec l'augmentation des concentrations du produit testé, ce qui est un indicateur clé d'une inhibition dose-dépendante. À partir de 0,51 mg/L, une chute drastique de la densité cellulaire indique que cette concentration atteint un seuil critique où l'effet devient significatif. Cette tendance suggère que cela pourrait avoir des implications écologiques importantes, étant donné le rôle fondamental des algues dans l'écosystème aquatique

On constate aussi, une variabilité notable, notamment à la concentration de 0,1 mg/L, où un écart-type plus élevé a été observé. Cette variabilité suggère une réponse plus hétérogène des algues à cette concentration, potentiellement due à des observations influentes ou aberrantes (qui peuvent tirer la moyenne vers le haut ou vers le bas de manière disproportionnée, affectant l'interprétation globale). Toutefois, la suppression des outliers n'a pas été justifiée biologiquement, et ces valeurs ont été maintenues dans l'analyse. Les résultats confirment donc une inhibition marquée de la croissance algale avec l'augmentation de la concentration du produit, malgré une certaine variabilité dans les réponses, particulièrement à 0,1 mg/L.

6.4 Détermination des LOEC/D/L et NOEC/D/L

Les LOEC, NOEC , LOED (Lowest Observed Effect Dose), et NOED (No Observed Effect Dose) sont des valeurs de toxicité en écotoxicologie. Elles permettent de déterminer les concentrations à partir desquelles un effet est observé sur les organismes testés. Dans l'exemple traité, les données représentent les réponses en 10^4 cells/mL d'algues après exposition à plusieurs concentrations d'une substance chimique. Les moyennes et écarts-types pour chaque condition sont donnés dans la table 4. Des tests ANOVA ont été réalisés pour comparer chaque concentration testée par rapport au témoin (*i.e* à "0").

Concentration(mg/L)	F-value	p-value
0.1	0.728	0.418
0.15	165.59	1.26e-06
0.23	194.81	6.73e-07
0.34	220.46	4.17e-07
0.51	1311.48	3.70e-10
0.76	3582.96	6.74e-12
1.14	3730.99	5.74e-12

Table 6: Résultats des tests ANOVA pour déterminer l'effet de chaque concentration sur la croissance des algues (Test 201 OCDE)

Au vu des résultats présentés dans le tableaux 6, la concentration **NOEC** est 0.1mg/L et la concentration **LOEC** est 0.15mg/L, ce qui signifie qu'à partir de 0.15mg/L, la substance commence à avoir un effet statistiquement significatif sur l'organisme étudié par rapport

au témoin, **parce que la p-value est inférieure à 0.05, indiquant que la différence observée n'est probablement pas due au hasard**. Les tests paramétriques peuvent être plus sensibles, mais lorsqu'ils sont appliqués sur des données non conformes aux hypothèses, ils peuvent induire en erreur. Les tests non-paramétriques, bien qu'ils soient moins puissants, sont plus robustes face aux violations d'hypothèses. Dans cette étude, bien que l'application de l'ANOVA ne soit pas pleinement justifiée par la distribution des données, ce test a été utilisé en raison des recommandations des tests OCDE, qui supposent généralement une distribution normale des données. Malgré cela, les résultats obtenus ont été cohérents, soulignant que les données biologiques peuvent parfois produire des résultats fiables même dans des conditions non idéales. Étant donné que les données ne respectaient pas les hypothèses de normalité et d'homogénéité des variances, des tests non-paramétriques ont été appliqués pour analyser les différences entre les groupes de concentrations testées. Le **test de Kruskal-Wallis** a été utilisé comme alternative non-paramétrique à l'ANOVA pour comparer les groupes. Ce test est particulièrement adapté aux données qui ne suivent pas une distribution normale. Les résultats du test de Kruskal-Wallis pour chaque concentration sont les suivants :

Concentration(mg/L)	K-Wallis Stat	p-value
0.1	1.098	0.295
0.15	6.818	0.009
0.23	6.818	0.009
0.34	6.860	0.009
0.51	6.860	0.009
0.76	7.031	0.008
1.14	7.258	0.007

Table 7: Résultats du test de Kruskal-Wallis pour les différentes concentrations (Test Non-Paramétrique)

Les résultats de l'analyse non-paramétrique ont conduit à des valeurs de NOEC et LOEC identiques à celles obtenues avec l'ANOVA. La **NOEC** est donc déterminée à **0.1 mg/L**. Cette concentration n'a pas montré de différence significative par rapport au contrôle ($p > 0.05$). La **LOEC** est déterminée à **0.15 mg/L**, qui est la concentration la plus faible à présenter un effet statistiquement significatif par rapport au contrôle ($p < 0.05$). Ces résultats fournissent une première conclusion essentielle pour les évaluateurs de risques : il est crucial de déterminer la valeur seuil à partir de laquelle l'effet devient significatif pour l'organisme testé. Cette détermination est fondamentale pour établir des seuils de sécurité environnementale et pour orienter les décisions réglementaires concernant l'utilisation des substances chimiques.

7 Calcul de Puissance

La puissance d'un test statistique est un élément fondamental pour évaluer la capacité d'une étude à détecter un effet réel, en particulier dans le contexte de la détermination d'une NOEC (concentration sans effet observé) ou d'autres résultats statistiquement significatifs.

La puissance se définit comme la probabilité de rejeter l'hypothèse nulle (H_0) lorsqu'elle est fausse, minimisant ainsi le risque d'erreurs de type II (faux négatifs). Une étude bien conçue doit avoir une puissance suffisante, généralement fixée à 80% en écotoxicologie, pour garantir que les différences biologiquement pertinentes ne passent pas inaperçues. L'objectif principal du calcul de puissance est d'évaluer la **capacité d'un test à détecter un effet significatif**, ce qui permet de s'assurer que l'étude est suffisamment sensible pour identifier des différences entre les groupes testés.

Ce calcul peut être effectué à deux moments clés :

- **Puissance a priori** : Utilisée pour déterminer la taille d'échantillon nécessaire avant l'expérience, afin de garantir une puissance adéquate. Cette approche est essentielle pour planifier des études qui soient suffisamment robustes dès le départ.
- **Puissance post hoc** : Évaluée après l'obtention des résultats pour estimer la sensibilité des tests réalisés. En évaluation du risque, notamment en écotoxicologie, la puissance post hoc est souvent la seule option disponible, car les tests sont généralement effectués après coup et non planifiés a priori. Cette contrainte impose des défis significatifs pour l'interprétation des résultats, car il est plus difficile de garantir rétrospectivement que l'étude avait une puissance suffisante pour détecter des effets écologiquement pertinents.

La puissance d'un test d'hypothèse dépend de plusieurs paramètres, dont la taille de l'effet, la taille de l'échantillon, et le niveau de signification (α). Par exemple, pour comparer la moyenne d'un groupe témoin avec celle d'un groupe exposé, l'hypothèse nulle (H_0) stipule que les moyennes sont égales ($\mu_0 = \mu_k$). La puissance du test est alors la probabilité de détecter une différence réelle ($H_a : \mu_0 > \mu_k$) si elle existe. Le calcul de la puissance peut être réalisé en utilisant les fonctions de distribution associées aux tests statistiques utilisés, tels que le test de Dunnett ou le test t de Student. Par exemple, pour détecter une diminution de 15% par rapport au contrôle, la puissance peut être visualisée via des courbes de puissance en fonction de la taille de l'échantillon et de la taille de l'effet attendu. Le calcul de puissance pour un test ANOVA se base sur la **taille de l'effet** (f), le nombre de groupes (k), la taille de l'échantillon par groupe (n), et le niveau de signification (α). La taille de l'effet est définie par Cohen comme :

$$f = \sqrt{\frac{\eta^2}{1 - \eta^2}}$$

où η^2 est la proportion de la variance expliquée par les groupes. La puissance est alors déterminée en utilisant la non-centralité λ , donnée par :

$$\lambda = n \times k \times f^2$$

La statistique de test suit une distribution F avec $k - 1$ et $N - k$ degrés de liberté, où N est le nombre total d'observations. La puissance est calculée comme :

$$\text{Puissance} = P(F > F_{\text{crit}}; \lambda, df_1, df_2)$$

où F_{crit} est la valeur critique de la distribution F pour le niveau de signification α , $df_1 = k - 1$, et $df_2 = N - k$.

Pour les données en écotoxicologie, cette analyse de puissance est particulièrement importante, car elle permet de s'assurer que les tests statistiques utilisés sont suffisamment sensibles pour détecter des effets écotoxicologiques significatifs, comme une diminution de la

survie, de la croissance, ou de la reproduction des organismes exposés à des substances chimiques. Si la puissance est faible, il est possible que même des effets biologiquement importants ne soient pas détectés, ce qui pourrait conduire à des conclusions erronées quant à la sécurité d'un produit chimique. Par exemple, en analysant la puissance pour un test ANOVA avec un effet attendu faible (f), si la puissance calculée est inférieure à 0,8, il serait recommandé d'augmenter la taille de l'échantillon ou de revoir le design expérimental. Cela pourrait signifier qu'il y a un risque élevé de ne pas détecter un effet significatif, ce qui est particulièrement critique en écotoxicologie, où la protection des écosystèmes repose sur des tests sensibles et fiables.

Une étude bien conçue doit donc avoir une puissance suffisante et l'objectif principal du calcul de puissance est de s'assurer que l'étude est suffisamment sensible pour identifier des différences entre les groupes testés. Cependant, lorsqu'une relation dose-réponse claire émerge, comme dans notre exemple de l'essai 201, une approche plus approfondie devient nécessaire. En effet, la simple identification des concentrations sans effet observable (NOEC) ou avec effet observable le plus bas (LOEC) peut ne pas suffire à caractériser pleinement l'impact d'un contaminant. Dans ces cas, la modélisation **dose-réponse** est indispensable pour quantifier plus précisément l'effet d'un contaminant en fonction de sa concentration.

8 Modélisation Dose-Réponse

L'analyse des effets d'un contaminant ou d'une substance chimique sur un organisme ou un groupe d'organismes est couramment réalisée *via* des essais dose-réponse. Ces essais permettent de déterminer des valeurs critiques telles que l'EC_x (Concentration Efficace pour x% de la population) ou la LC_x (Concentration Létale pour x% de la population), des paramètres essentiels pour évaluer le risque environnemental d'un polluant (cf méthodologie paragraphe 5.6).

Dans le cadre de ce stage, trois modèles de régression non linéaire ont été utilisés : le modèle log-logistique, le modèle log-normal, et le modèle Weibull. Ces modèles sont couramment employés en écotoxicologie pour décrire la relation dose-réponse et permettent de calculer les EC_x ou LC_x avec précision (E et L dépend du type de test cela doit être défini dans ma méthodologie, mais pour simplifier je ne parlerai que de EC_x. Peux tu donc me faire une phrase ou deux bien écrite à mettre dans ma méthode?).

Les paramètres estimés de ces modèles sont présentés dans le tableau 8. D'après les résultats, on voit que:

1. **Paramètre b (Pente de la courbe dose-réponse):**

Ce paramètre détermine la pente de la courbe dose-réponse. Une valeur plus élevée de b indique une transition plus abrupte entre les faibles et les fortes réponses à la dose, suggérant que les effets de la substance augmentent rapidement sur une courte plage de concentrations. Dans le modèle log-logistique, $b = 3.9007$ ($p = 0.0021$), ce qui indique une réponse rapide et significative à la dose. Le modèle Weibull montre une pente encore plus forte ($b = 4.9421$, $p = 0.00001$), renforçant l'idée que les effets sont fortement liés à une petite variation de la concentration.

2. **Paramètre c (Effet minimum observé ou limite inférieure):**

Ce paramètre représente l'effet minimal observé (l'asymptote inférieure) lorsque la concentration est faible ou nulle. Dans certains cas, c peut être interprété comme la réponse de base ou de fond. Les valeurs de c sont négatives dans tous les modèles, ce

qui peut indiquer des difficultés d'ajustement pour ces paramètres, ou une base d'effet inférieure à zéro, ce qui n'est pas biologiquement plausible dans tous les contextes. Les p-values non significatives suggèrent que c n'est pas un paramètre robuste dans ces modèles.

3. **Paramètre d (Effet maximum observé ou limite supérieure) :**

Ce paramètre décrit l'effet maximal observé (l'asymptote supérieure) à haute concentration. C'est une estimation de l'effet lorsque la concentration atteint un niveau où presque toute la population est affectée. Les trois modèles montrent des valeurs de d très significatives (toutes avec $p < 2.2e-16$), ce qui indique un ajustement robuste pour la réponse maximale à haute dose. Par exemple, $d = 60.2790$ pour le modèle log-logistique montre que la réponse maximale attendue dans cette condition est bien capturée.

4. **Paramètre e (EC_{50} ou concentration inflexion de la courbe) :**

Ce paramètre correspond à la concentration médiane efficace, EC_{50} , c'est-à-dire la concentration à laquelle 50% de l'effet maximal est observé. Il représente le point d'inflexion de la courbe dose-réponse. Le modèle log-logistique donne une estimation de $e = 6.8654$ ($p < 0.0001$), ce qui suggère qu'une concentration relativement basse est nécessaire pour atteindre 50% de l'effet maximal. En revanche, le modèle log-normal présente un $e = 15.5224$ ($p = 0.4005$), beaucoup moins significatif, indiquant que ce modèle pourrait mal ajuster les données dans la zone critique de l' EC_{50} , c'est-à-dire dans la région où la réponse de la population change le plus rapidement. Cette faible significativité pourrait être due à une mauvaise adéquation du modèle log-normal aux données expérimentales, particulièrement dans les zones de transition importantes.

Paramètre	Log-Logistique	Log-Normale	Weibull
b	3.9007 ($p = 0.0021$)	-1.2779 ($p = 0.0472$)	4.9421 ($p = 0.00001$)
c	-36.9821 ($p = 0.2377$)	-262.0235 ($p = 0.6049$)	-2.6619 ($p = 0.4808$)
d	60.2790 ($p < 2.2e-16$)	61.1410 ($p < 2e-16$)	59.1626 ($p < 2.2e-16$)
e	6.8654 ($p < 0.0001$)	15.5224 ($p = 0.4005$)	6.1774 ($p < 2.2e-16$)

Table 8: Estimation des paramètres pour les modèles log-logistique, log-normal et Weibull.

Le choix du modèle adéquat est important pour estimer avec précision les EC_x , car il influence directement la qualité et la fiabilité des résultats obtenus. Sur l'exemple des données obtenues avec l'essai 201[?], le modèle log-normal, bien que souvent utilisé, a montré des limitations dans ce contexte particulier. La faible significativité de certains paramètres critiques, comme le paramètre e , suggère que ce modèle pourrait ne pas capturer adéquatement la relation dose-réponse pour les données étudiées. Cette conclusion est renforcée par la comparaison systématique des modèles, où le modèle Weibull a offert des estimations plus robustes et mieux adaptées. Les limitations du modèle log-normal peuvent ainsi être attribuées à son incapacité à ajuster correctement les variations rapides de réponse autour de l' EC_{50} , ce qui est essentiel pour une modélisation précise des effets à des concentrations spécifiques.

C'est pourquoi la comparaison systématique des modèles est une étape indispensable pour garantir la robustesse des conclusions tirées de l'étude.

8.1 Comparaison des Modèles

Afin de déterminer quel modèle s'ajuste le mieux aux données, nous avons utilisé deux critères de sélection : le critère d'information d'Akaike (AIC) et le facteur de Bayes ($B_{1/0}$) (cf méthodologie). Ces critères permettent de comparer les modèles en termes de qualité d'ajustement tout en tenant compte de la complexité du modèle. Notamment, nous n'avons pas utilisé le critère d'information bayésien (BIC) dans cette analyse, pour capturer avec précision la complexité des données expérimentales, sans sur-pénaliser les modèles qui offrent un meilleur ajustement, même s'ils sont légèrement plus complexes. En effet, le BIC (trop strict) a tendance à pénaliser excessivement des modèles qui, bien qu'un peu plus complexes, capturent mieux les nuances des données.

Modèle	AIC
Log-logistique	307.1229
Log-normal	310.4866
Weibull	301.9307

Table 9: Valeurs de l'AIC

Comparaison des Modèles	$(B_{1/0})$	$(B_{1/0})$ avec MH
Log-logistique vs Log-normal	0.01545374	0.01545374
Log-logistique vs Weibull	0.0006039546	0.0006039546
Log-normal vs Weibull	0.03908144	0.03908144

Table 10: Facteurs de Bayes

Les résultats montrent que le modèle Weibull possède la valeur d'AIC la plus faible (301.9307), indiquant qu'il s'ajuste le mieux aux données tout en restant relativement simple. Le modèle log-logistique suit avec un AIC de 307.1229, tandis que le modèle log-normal présente l'AIC le plus élevé (310.4866), ce qui en fait le modèle le moins performant dans cette comparaison. Les facteurs de Bayes (avec ou sans MH) corroborent les conclusions tirées de l'AIC. Le modèle log-logistique est préféré au modèle log-normal, mais il est moins performant que le modèle Weibull. De plus, la comparaison entre le modèle log-normal et le modèle Weibull montre une nette préférence pour le modèle Weibull. L'analyse bayésienne a utilisé l'algorithme de Metropolis-Hastings (MH) pour échantillonner les distributions postérieures des paramètres, essentiel pour calculer les facteurs de Bayes avec a priori explicite. Deux approches ont été appliquées :

1. **Méthode avec MH (a priori explicite):** Cette méthode utilise explicitement les distributions a priori pour échantillonner les postérieures. Ici, le MH a bien fonctionné pour les approximations, mais avec des défis de convergence pour certains paramètres, comme le montre la variabilité dans les trace plots et les taux d'acceptation. Par exemple, dans le modèle Weibull, un taux d'acceptation de 0.2248 a été observé, suggérant une exploration parfois limitée de l'espace des paramètres.
2. **Méthode sans MH (a priori implicite):** Cette méthode repose sur des a priori implicites, où la distribution postérieure est estimée sans un échantillonnage ex-

plicité via MH. Ici, les résultats ont montré une cohérence avec les méthodes explicites, renforçant la robustesse des conclusions. Cependant, cette approche peut sous-estimer l'incertitude, particulièrement dans des espaces de paramètres complexes, ce qui souligne l'importance d'une évaluation soigneuse des diagnostics de convergence.

Modèle	Taux d'acceptation
Modèle log-logistique	0.5251
Modèle log-normal	0.7885
Modèle Weibull	0.2248

Table 11: Taux d'acceptation pour chaque modèle.

Les graphiques ACF montrent que certains paramètres, notamment la pente du modèle log-logistique et la variance logarithmique du modèle log-normal, présentent une meilleure indépendance entre les échantillons successifs.

En revanche, les paramètres de concentration effective (EC_{50}), moyenne logarithmique, et ceux du modèle Weibull montrent une forte autocorrélation, suggérant un mélange moins efficace et une convergence potentiellement plus lente. Il pourrait être bénéfique de prolonger les chaînes ou d'ajuster les paramètres d'échantillonnage pour ces derniers. Les résultats des différentes approches de modélisation sont remarquablement cohérents. L'approche initiale basée sur l'AIC et la log-vraisemblance a identifié le modèle de Weibull comme le meilleur ajustement pour les données, suivi du modèle log-logistique. Cette conclusion a été renforcée par l'analyse bayésienne utilisant des chaînes MCMC, qui a également favorisé le modèle de Weibull. Cependant, il est important de noter que, bien que les résultats des chaînes MCMC confirment ceux de l'approche initiale, des diagnostics ont montré des lacunes dans la convergence et l'efficacité de l'échantillonnage. Cela suggère qu'une optimisation supplémentaire des chaînes MCMC pourrait encore affiner ces conclusions.

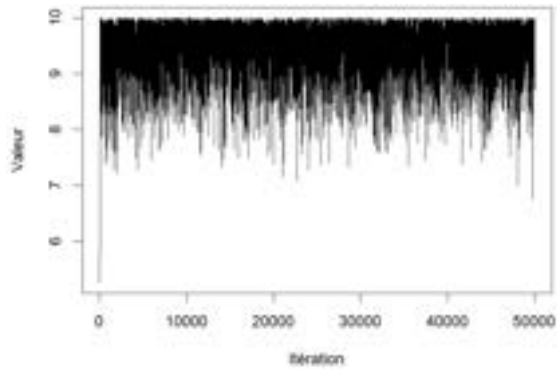


Figure 8.1: Trace log-logistique Pente (b)

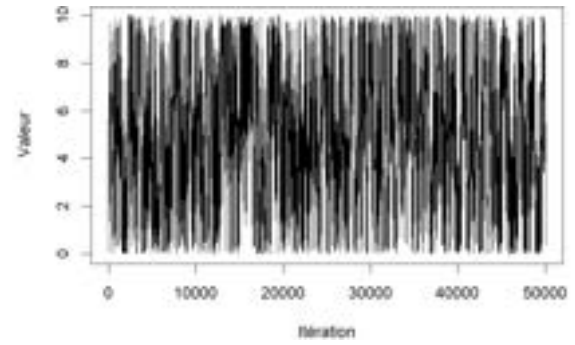


Figure 8.2: Trace log-logistique (EC_{50})

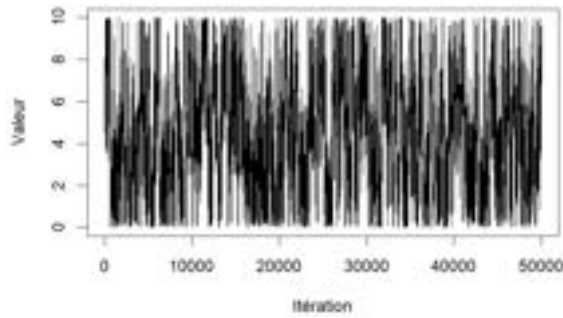


Figure 8.3: Trace log-normal Moyenne logarithmique

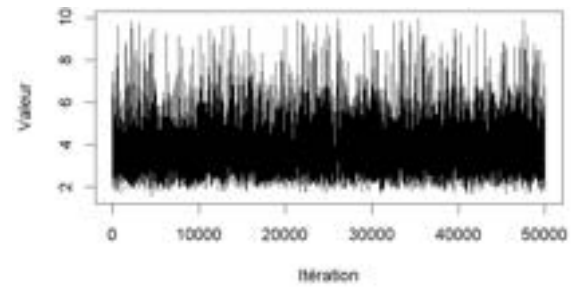


Figure 8.4: Trace log-normal Variance logarithmique

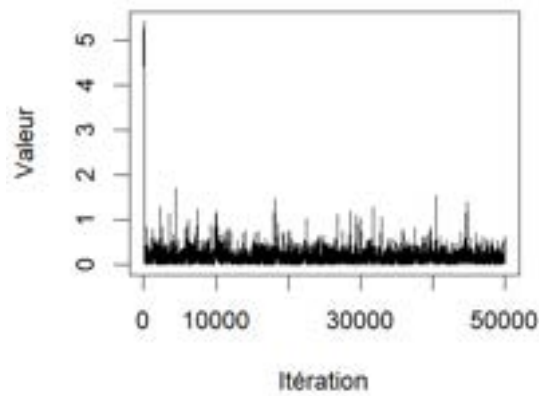


Figure 8.5: Trace Weibull Paramètre de forme

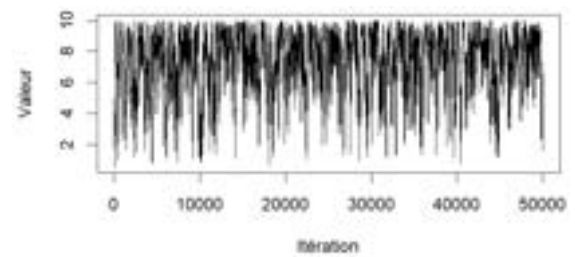


Figure 8.6: Trace Weibull Paramètre d'échelle

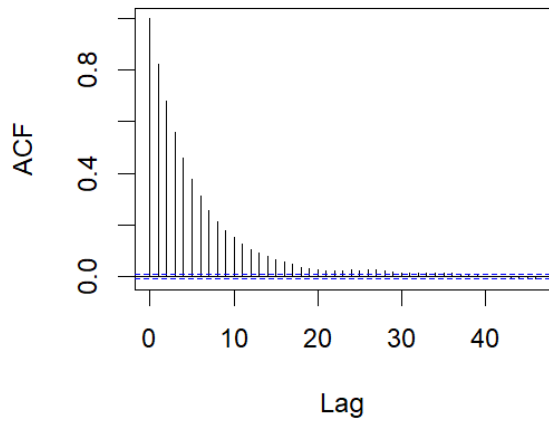


Figure 8.7: ACF pour le modèle log-logistique Pente (b)

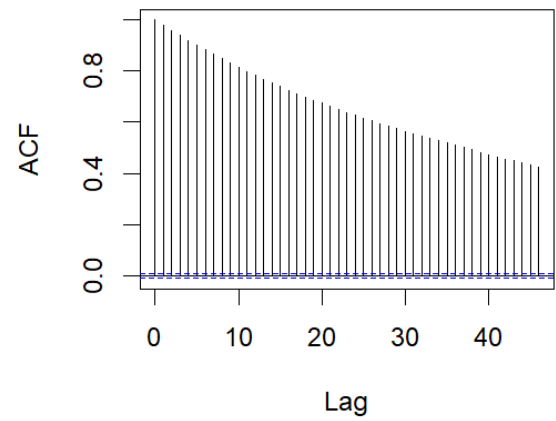


Figure 8.8: ACF pour le modèle log-logistique (EC_{50})

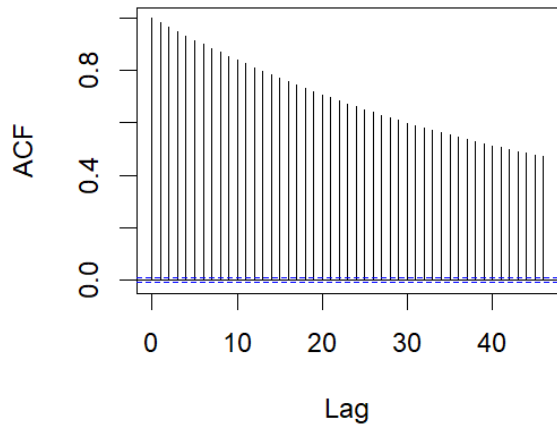


Figure 8.9: ACF pour le modèle log-normal Moyenne logarithmique

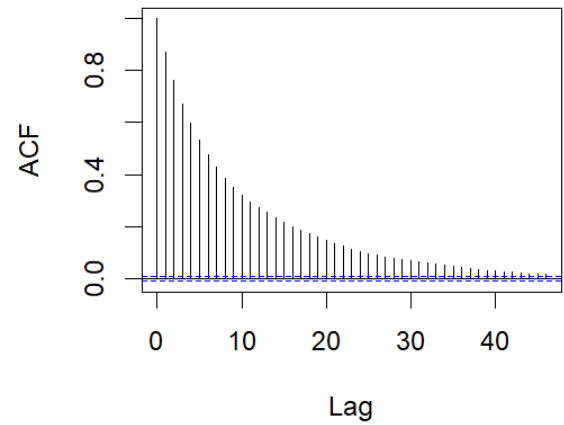


Figure 8.10: ACF pour le modèle log-normal Variance logarithmique

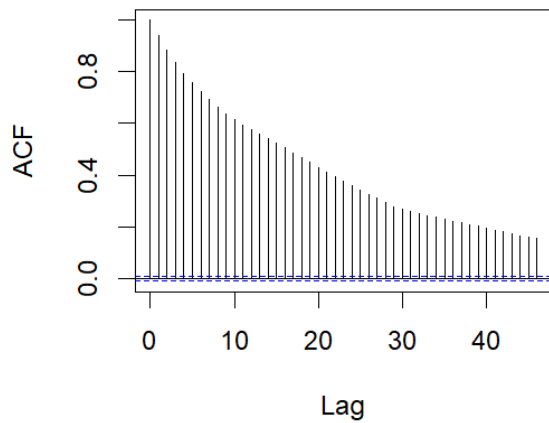


Figure 8.11: ACF pour le modèle Weibull Paramètre de forme

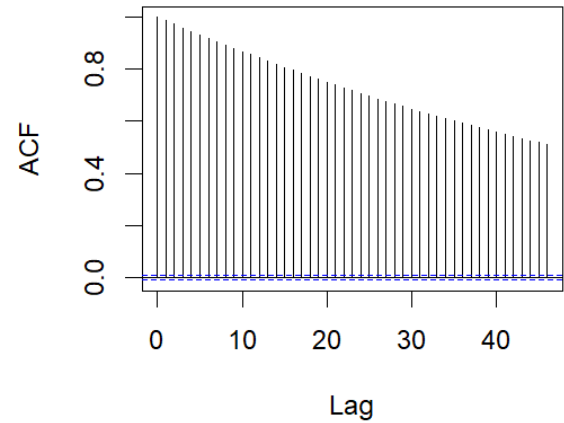


Figure 8.12: ACF pour le modèle Weibull Paramètre d'échelle

9 Estimation des ECx ou LCx

Par exemple, les valeurs de l' EC_{10} , l' EC_{20} , et l' EC_{50} , indiquent les concentrations où 10%, 20%, et 50% de l'effet maximal sont atteints. Ces mesures sont essentielles pour l'évaluation des risques écotoxicologiques, car elles permettent de quantifier la toxicité d'une substance de manière précise et fiable.

Dose Effective	Estimation	Erreur Standard
ERC10 (Weibull)	3.9179	0.3020
ERC20 (Weibull)	4.5603	0.2189
ERC50 (Weibull)	5.7359	0.1224

Table 12: Estimations des doses effectives à 10%, 20%, et 50% pour le modèle Weibull.

Pour le modèle Weibull (cf méthodologie) l' EC_{50} est la concentration à laquelle la réponse est à mi-chemin entre c et d , c'est-à-dire où $f(EC_{50}) = \frac{c+d}{2}$.
Donc pour $f(x) = \frac{c+d}{2}$, on a :

$$\frac{c+d}{2} = d - (d-c) \cdot \exp\left(-\left(\frac{EC_{50}}{b}\right)^a\right)$$

En isolant $\frac{EC_{50}}{b}$, on obtient :

$$\exp\left(-\left(\frac{EC_{50}}{b}\right)^a\right) = \frac{1}{2}$$

Ce qui conduit à :

$$\frac{EC_{50}}{b} = (\log(2))^{\frac{1}{a}} \quad \text{et donc} \quad EC_{50} = b \cdot (\log(2))^{\frac{1}{a}}$$

L' ECx correspond à la concentration x pour laquelle $f(ECx) = c + (d-c) \cdot \frac{x}{100}$.
Donc, pour EC_{10} et EC_{20} on a : $f(EC_{10}) = c + 0.1 \cdot (d-c)$ et $f(EC_{20}) = c + 0.2 \cdot (d-c)$
On peut réécrire ceci en isolant ECx :

$$f(ECx) = d - (d-c) \cdot \exp\left(-\left(\frac{ECx}{b}\right)^a\right)$$

Cela conduit à l'équation suivante :

$$c + \left(\frac{x}{100}\right) \cdot (d-c) = d - (d-c) \cdot \exp\left(-\left(\frac{ECx}{b}\right)^a\right)$$

Isolons ECx :

$$\exp\left(-\left(\frac{ECx}{b}\right)^a\right) = 1 - \frac{x}{100}$$

Cela peut être résolu pour ECx :

$$ECx = b \cdot \left(-\log \left(1 - \frac{x}{100} \right) \right)^{\frac{1}{a}}$$

Pour vérifier les calculs des ECx (EC_{10} , EC_{20} , EC_{50}) en utilisant les paramètres fournis dans la table 8 pour le modèle Weibull.

$d = 59.1626$ (approximé à partir des réponses maximales au contrôle).

$c = -2.6619$ (approximé à partir des réponses minimales aux hautes concentrations).

$b = 4.9421$ mg/L (approximé à partir de l' EC_{50} obtenu avec drc)

$e = a = 6.1774$ (approximé à partir des résultats drc).

Donc, on a:

1.

$$EC_{50} = 4.9421 \cdot (\log(2))^{\frac{1}{6.1774}} = 4.9421 \cdot (0.6931)^{\frac{1}{6.1774}} \approx 4.9421 \cdot 0.9126 \approx \mathbf{4.508} \text{ mg/L}$$

2.

$$EC_{10} = 4.9421 \cdot \left(-\log \left(1 - \frac{10}{100} \right) \right)^{\frac{1}{6.1774}} = 4.9421 \cdot (-\log(0.9))^{\frac{1}{6.1774}}$$

$$EC_{10} = 4.9421 \cdot (0.1054)^{\frac{1}{6.1774}} \approx 4.9421 \cdot 0.7057 \approx \mathbf{3.487} \text{ mg/L}$$

3.

$$EC_{20} = 4.9421 \cdot \left(-\log \left(1 - \frac{20}{100} \right) \right)^{\frac{1}{6.1774}} = 4.9421 \cdot (-\log(0.8))^{\frac{1}{6.1774}}$$

$$EC_{20} = 4.9421 \cdot (0.2231)^{\frac{1}{6.1774}} \approx 4.9421 \cdot 0.8046 \approx \mathbf{3.975} \text{ mg/L}$$

Comparées aux valeurs obtenues avec le package "**drc**", il est notable que les résultats peuvent varier légèrement en fonction de l'ajustement spécifique des modèles et des méthodes utilisées pour l'estimation des paramètres. Les intervalles de confiance (IC) fournissent une mesure de la précision des estimations des ECx , en donnant une indication sur la variabilité autour de ces valeurs. Afin de mieux quantifier cette variabilité et d'obtenir des IC plus robustes, nous appliquons la méthode du bootstrap

9.0.1 Test de Permutation et Intervalles de Confiance Bootstrap

Dose Effective	Intervalle de Confiance à 95% (Percentile)
ERC10 (Weibull)	[3.192, 6.511]
ERC20 (Weibull)	[4.089, 9.127]
ERC50 (Weibull)	[5.609, 15.403]

Table 13: Intervalles de confiance bootstrap à 95% pour les doses efficaces estimées.

Le test de permutation a été utilisé pour évaluer la significativité des résultats obtenus précédemment. Étant une méthode non paramétrique, il permet de vérifier de manière robuste les estimations des concentrations efficaces (EC_{10} , EC_{20} , et EC_{50}) issues du modèle

dose-réponse Weibull à 4 paramètres, sans crainte de biais lié aux hypothèses de distribution. Selon les estimations réelles effectuées, les valeurs d'ECx étaient de 3.91788 mg/L pour l' EC_{10} , 4.56033 mg/L pour l' EC_{20} , et 5.73586 mg/L pour l' EC_{50} . Le test a donc procédé en permutant aléatoirement les données pour générer plusieurs ensembles de données fictives, où les relations entre les concentrations et les réponses biologiques étaient perturbées. Pour chaque ensemble permuté, les estimations des ECx ont été recalculées, créant ainsi une distribution nulle des valeurs d'ECx sous l'hypothèse nulle (H_0), selon laquelle la relation observée initialement est due au hasard. En comparant les estimations réelles à ces distributions permutées, les p-values obtenues pour l' EC_{10} , l' EC_{20} , et l' EC_{50} étaient toutes de 0.989, ce qui est très élevé. Cela suggère qu'il y a une probabilité de 98.9% que les différences observées entre les estimations réelles et celles issues des permutations soient dues au hasard, et non à un effet réel de la concentration sur la réponse biologique. Les intervalles de confiance Bootstrap obtenus (cf tableau 13) sont cruciaux pour évaluer la fiabilité des estimations d'ECx. Ils montrent l'étendue possible des valeurs d'ECx, offrant une plage dans laquelle la vraie valeur d'ECx est susceptible de se situer à 95% de confiance. Les résultats révèlent que l'incertitude des estimations augmente avec le niveau de l'effet (de l' EC_{10} à l' EC_{50}). L' EC_{10} a un intervalle relativement étroit (3.192 à 6.511 mg/L), indiquant une estimation plus précise de cette concentration. Cela suggère que l' EC_{10} est une mesure relativement stable et fiable, avec une moindre sensibilité aux variations dans les données expérimentales.

L' EC_{20} présente un intervalle plus large (4.089 à 9.127 mg/L), ce qui montre une augmentation de l'incertitude. L'estimation est moins précise que celle de l' EC_{10} , reflétant une plus grande variabilité dans cette région de la courbe dose-réponse. Et enfin, l' EC_{50} a l'intervalle de confiance le plus large (5.609 à 15.403 mg/L), soulignant une incertitude encore plus grande. Ce résultat est attendu puisque, dans les modèles dose-réponse, la précision diminue généralement à mesure que l'on s'éloigne de l' EC_{50} en raison de la pente de la courbe qui devient moins raide. Les intervalles de confiance montrent que, malgré les incertitudes croissantes aux doses plus élevées, les estimations d'ECx sont robustes. Le modèle Weibull à 4 paramètres s'avère être une méthode efficace pour capturer la relation dose-réponse, et l'ajout des IC bootstrap renforce la confiance dans ces estimations. Cependant, il est important de noter que l'approche par permutation et l'approche par bootstrap peuvent être considérées comme **complémentaires** plutôt qu'exclusives l'une de l'autre. Le bootstrap, permet entre autre, de construire des intervalles de confiances sans nécessiter l'hypothèse **d'échangeabilité** requise par les tests de permutation. Le choix entre les deux méthodes dépend donc du contexte spécifique de l'analyse et des caractéristiques des données.

En pratique, si les données présentent une structure où l'hypothèse d'échangeabilité est raisonnable, les tests par permutation peuvent fournir des résultats **robustes** pour tester l'indépendance ou la comparaison de distributions entre groupes. Dans les cas où cette hypothèse est difficile à justifier, ou lorsque l'objectif est d'estimer des intervalles de confiance pour des paramètres de modèle, le bootstrap peut être préférable. Ainsi, il peut être judicieux de comparer les résultats obtenus par ces deux approches pour une même analyse, afin d'évaluer leur concordance ou de comprendre les divergences éventuelles. En écotoxicologie, cette complémentarité permet d'aborder l'incertitude des estimations sous différents angles, en renforçant la confiance dans les conclusions tirées, ou en identifiant des aspects des données qui nécessitent une attention particulière.

En somme, les intervalles de confiance Bootstrapés et l'analyse par modèle Weibull offrent une vision claire et robuste des estimations d'ECx, malgré une incertitude croissante avec l'augmentation du niveau d'effet. Cependant, une conclusion importante pour les évalua-

teurs de risque réside dans la détermination de la **valeur seuil à x% d'effet significatif pour l'espèce d'algue testé**, qui constitue une finalité cruciale de cette application. Cette valeur seuil, obtenue en combinant les approches par permutation et bootstrap, permet de fournir une évaluation complète et fiable des risques écotoxicologiques, en tenant compte des incertitudes associées à chaque niveau d'effet.

10 Conclusion & Perspectives

Le développement de l'application a permis de réaliser avec succès le processus d'évaluation en offrant aux évaluateurs un outil pour visualiser graphiquement des endpoints valides et fiables. Grâce à cette application, les évaluateurs pourront suivre un processus guidé, pour identifier les tests statistiques les plus appropriés. Les endpoints sont vérifiés sous les angles de la fiabilité et de la robustesse, ce qui constitue une avancée significative dans l'automatisation et la fiabilisation de l'évaluation. Un exemple concret de l'utilisation de cette application est l'analyse de l'essai 201 de l'OCDE, conçu pour évaluer les effets toxiques d'une substance sur des organismes aquatiques. Cet essai a été méticuleusement analysé au travers de plusieurs étapes statistiques de l'application. Les données, après dépistage des outliers qui n'ont pas été éliminés, ont été utilisées pour estimer les concentrations effectives EC_x à différents niveaux (EC_{10} , EC_{20} , EC_{50}) en utilisant un modèle Weibull, sélectionné par le critère d'information d'Akaike (AIC) et par le facteur Bayes. L'application RShiny que j'ai développée au cours de ce stage est encore à une phase initiale, mais elle a déjà permis de coder plusieurs essais d'écotoxicité. Jusqu'à présent, six essais d'écotoxicité (fonctionnels) ont été entièrement codés, couvrant une gamme variée de besoins identifiés par l'unité U3EiV. De plus, l'application offre des options flexibles pour analyser différents jeux de données, indépendamment du type d'essai standardisé employé. Durant le dernier mois de mon stage, mon objectif est de compléter le codage d'une partie des 15 essais d'écotoxicité qui m'ont été suggérés par l'équipe, afin de répondre de manière plus exhaustive aux exigences spécifiques de l'unité. Je prévois également d'implémenter une fonctionnalité permettant de générer automatiquement des rapports au format PDF ou Word. Cette fonctionnalité offrira aux évaluateurs la possibilité d'extraire un résumé succinct des analyses statistiques effectuées via l'application RShiny. Il est toutefois important de noter que, compte tenu de l'ampleur du projet, le développement de l'application devra se poursuivre après la fin de mon stage pour aboutir à un produit finalisé, utilisable de manière routinière par les évaluateurs de risques. L'application présente un fort potentiel d'utilisation, non seulement dans le domaine de l'écotoxicologie, mais également dans d'autres disciplines, telles que la toxicologie. Elle pourrait également servir de support lors de discussions au niveau des instances européennes et internationales, notamment pour la réévaluation des documents guides statistiques ou pour le développement d'un outil harmonisé à l'échelle européenne. En résumé, le travail accompli durant ce stage représente une première étape essentielle dans un projet de grande envergure. Ce projet pose les bases d'un outil polyvalent, qui pourra être enrichi et utilisé par d'autres, dans un cadre élargi. Mon implication dans ce projet a non seulement permis d'avancer significativement dans le développement de l'application, mais a également ouvert la voie à de nombreuses perspectives de développement et d'amélioration futures.

A Annexe A : Inventaire des tests

Table 14: Forces et Limites des Tests Statistiques

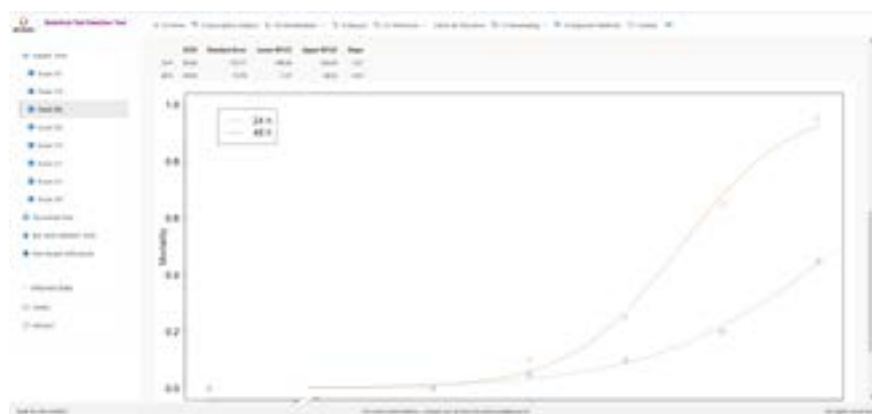
Test	Forces	Limites
T-test	Simple à comprendre et à exécuter; puissant pour comparer les moyennes de deux groupes	Sensible aux écarts de normalité, trompeur avec outliers ou variances inégales
ANOVA	Compare les moyennes de trois groupes ou plus; plus robuste que multiples t-tests	Ne précise pas quels groupes diffèrent; sensible à la non-normalité et hétérogénéité des variances
Régression Linéaire	Relation quantitative entre variables; peut inclure multiples variables indépendantes	Nécessite linéarité, indépendance, normalité des résidus, homoscedasticité; sensible aux valeurs aberrantes et multicollinéarité
Test du Chi-carré	Flexible, utilisable pour comparaisons catégorielles; requiert seulement des données de fréquence	Cases du tableau doivent avoir suffisamment d'observations; ne fournit pas l'ampleur de l'association
Mann-Whitney & Kruskal-Wallis	Convient aux distributions non normales; utilisables avec petits échantillons	ne fournissent pas de mesure de l'ampleur de l'effet Moins puissants que les équivalents paramétriques;

Table 15: Critères de Choix des Tests Statistiques

Test	Critères de Choix
T-test	Données quantitatives continues pour deux groupes; nécessite une distribution normale et des variances similaires
ANOVA	Données continues pour trois groupes ou plus; nécessite une distribution normale et des variances égales parmi les groupes
Régression Linéaire	Pour prédire une variable continue à partir d'une ou plusieurs variables indépendantes; relations entre variables doivent être linéaires
Test du Chi-carré	Comparer distributions de variables catégorielles ou tester l'indépendance entre deux variables catégorielles
Mann-Whitney & Kruskal-Wallis	Données ne respectant pas la normalité et/ou l'homogénéité des variances; Mann-Whitney pour deux groupes, Kruskal-Wallis pour trois ou plus

B Annexe :Application interface





C Annexe :Application architecture

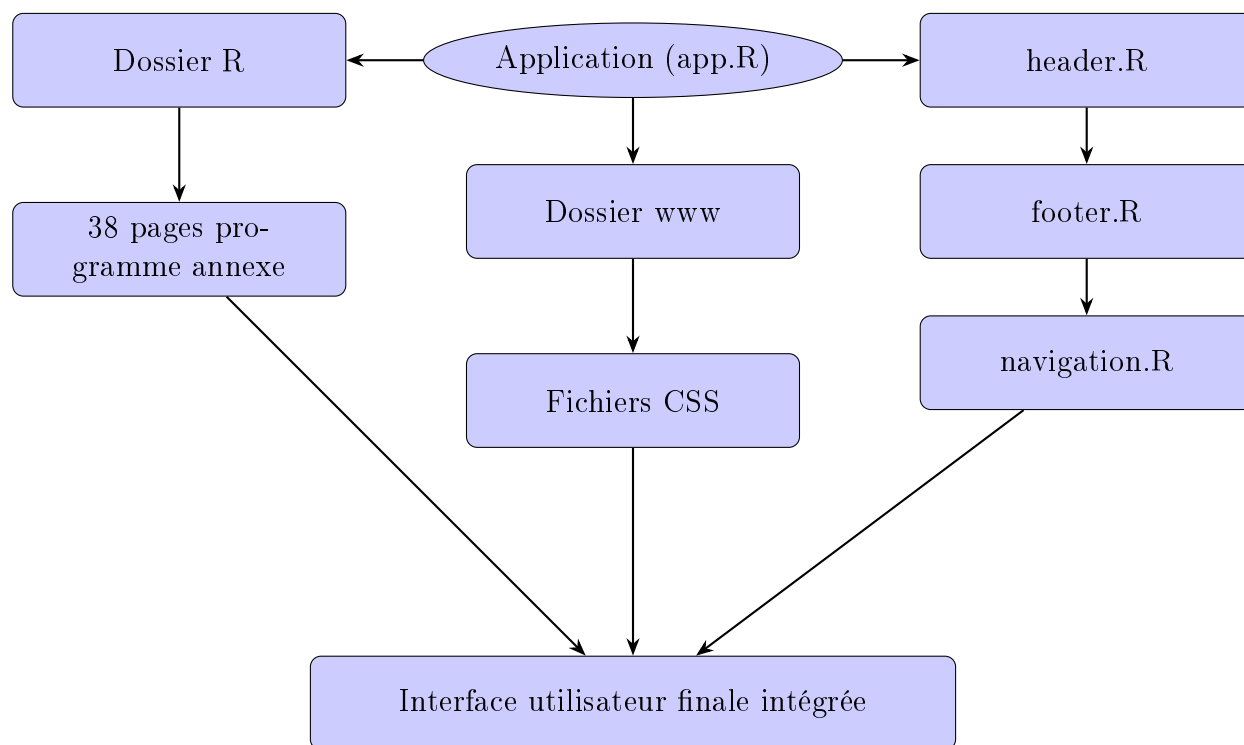


Figure C.1: Architecture générale de l'application

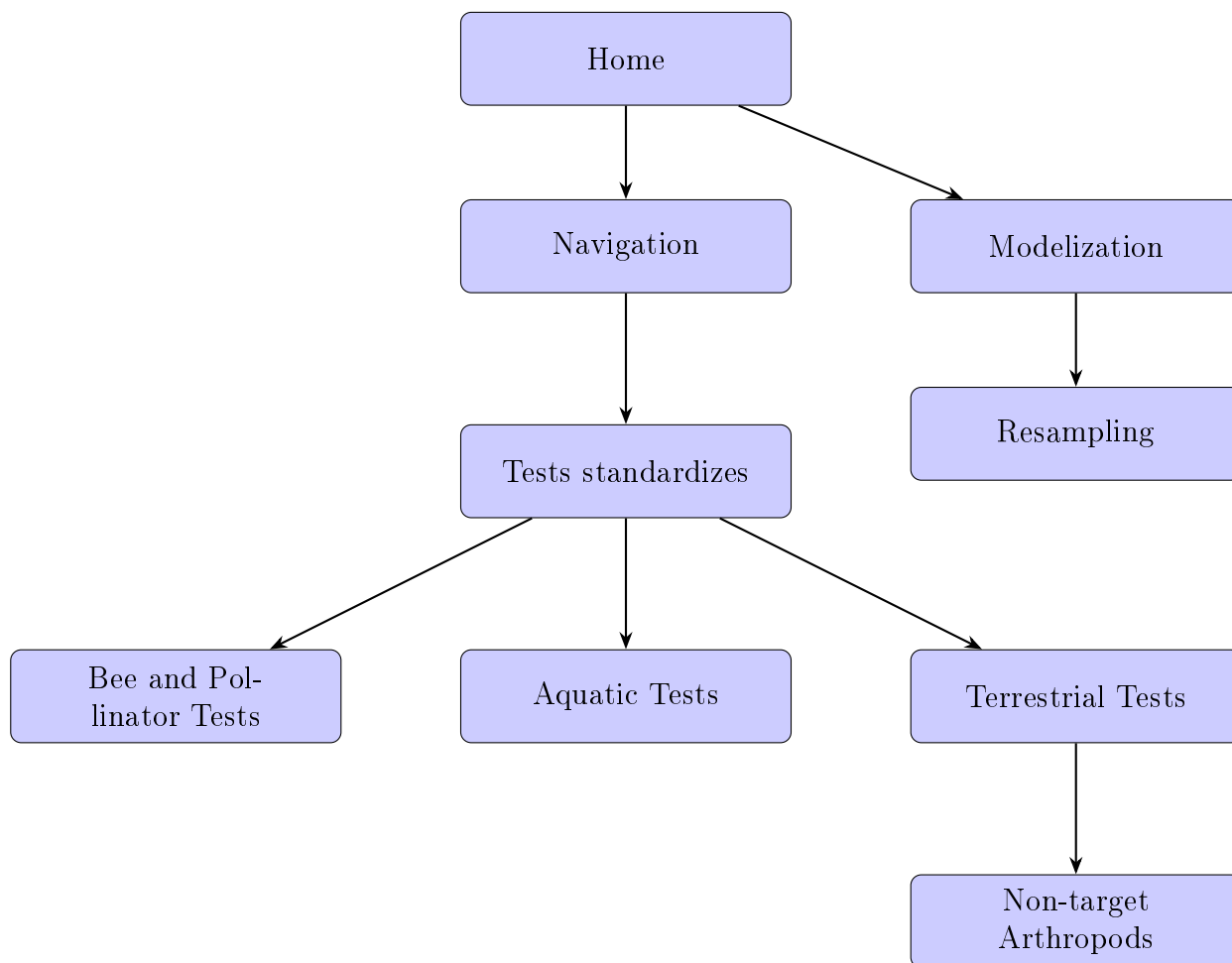


Figure C.2: Diagramme essais standardisés

Annexe : Script application

```

1 library(shiny)
2 library(shiny.fluent)
3 library(shiny.router)
4 library(sass)
5 library(leaflet)
6 library(ggplot2)
7 library(plotly)
8 library(glue)
9 library(dplyr)
10 library(tidyr)
11 library(DT)
12 library(DescTools)
13 library(Publish)
14 library(car)
15 library(multcomp)
16 library(ggpubr)
17 library(reshape2)
18 library(tidyverse)
19 library(drc)
20 library(here)

```

```

21 library(rmarkdown)
22 library(shinyjs)
23 library(shinycssloaders)
24 library(MASS) # Pour la transformation Box-Cox
25 #Pour bayesien
26 library(BayesFactor)
27 library(boot)
28 library(coin)
29 library(plotrix)
30 library(stats)
31 library(bayesmeta)
32 library(pwr)
33 ## Fonctions ####
34 SummaryVisu <- function(df, resp, trt) {
35   df$trt <- as.factor(df[[trt]])
36   df$resp <- df[[resp]]
37   df <- df[complete.cases(df[, c("trt", "resp")]), ]
38   concentration <- levels(df$trt)
39   concentration <- as.vector(concentration)
40   Means <- tapply(df$resp, df$trt, mean, na.rm = TRUE)
41   Means <- round(as.vector(Means), 4)
42   Sample_Size <- tapply(df$resp, df$trt, length)
43   Sample_Size <- as.vector(Sample_Size)
44   Median <- tapply(df$resp, df$trt, median)
45   Median <- round(as.vector(Median), 4)
46   Standard_Deviation <- tapply(df$resp, df$trt, sd)
47   Standard_Deviation <- round(as.vector(Standard_Deviation), 4)
48   Standard_Error <- round(Standard_Deviation / sqrt(Sample_Size), 4)
49   out <- data.frame(concentration, Sample_Size, Means, Median,
50     Standard_Deviation, Standard_Error)
51   return(out)
52 }
53 # Fonction Dunnett
54 analyze_dunnett <- function(data) {
55   data$response <- as.numeric(data$response)
56   data$treatment <- as.factor(data$treatment)
57
58   aov1 <- aov(response ~ treatment, data = data)
59   summary_aov <- summary(aov1)
60   significant <- ifelse(summary_aov[[1]][["Pr(>F)"]][[1]] <= 0.05,
61     "ANOVA results: Difference is statistically significant.",
62     "ANOVA results: Difference is not statistically significant.")
63
64   sw <- shapiro.test(residuals(aov1))
65   normality <- ifelse(sw$p.value <= 0.05, "Normality is rejected", "Normality is accepted")
66
67   b <- bartlett.test(response ~ treatment, data = data)
68   homoscedasticity <- ifelse(b$p.value <= 0.05, "Homoscedasticity is rejected", "Homoscedasticity is accepted")

```



```

68
69 Res_dunnett <- DunnettTest(data$response, data$treatment, control
70   = "0", conf.level = 0.95)
71 summary_dunnett <- summary(glht(aov1, linfct = mcp(treatment = "
72   Dunnett"))))
73
74 return(list(summary_aov = summary_aov,
75   significant = significant,
76   normality = normality,
77   homoscedasticity = homoscedasticity,
78   Res_dunnett = Res_dunnett,
79   summary_dunnett = summary_dunnett))
80 }
81
82 # Fonction ANOVA
83 analyze_anova <- function(data) {
84   data$response <- as.numeric(data$response)
85   data$concentration <- as.factor(data$concentration)
86
87   aov1 <- aov(response ~ concentration, data = data)
88   summary_aov <- summary(aov1)
89   significant <- ifelse(summary_aov[[1]][["Pr(>F)"]][[1]] <= 0.05,
90     "ANOVA results: Difference is statistically
91     significant.",
92     "ANOVA results: Difference is not
93     statistically significant.")
94
95 # Fonction Shapiro/bartlett
96 sw <- shapiro.test(residuals(aov1))
97 normality <- ifelse(sw$p.value <= 0.05, "Normality is rejected", "
98   Normality is accepted")
99
100 b <- bartlett.test(response ~ concentration, data = data)
101 homoscedasticity <- ifelse(b$p.value <= 0.05, "Homoscedasticity is
102   rejected", "Homoscedasticity is accepted")
103
104 return(list(summary_aov = summary_aov,
105   significant = significant,
106   normality = normality,
107   homoscedasticity = homoscedasticity))
108 }
109
110 # Fonction Bootstrap
111 # analyze_bootstrap <- function(data, num_bootstrap = 1000) {
112   # boot_means <- replicate(num_bootstrap, {
113     # sample_data <- sample(data$response, replace = TRUE)
114     # mean(sample_data)
115   # })
116
117   # ci <- quantile(boot_means, probs = c(0.025, 0.975))
118   # mean_val <- mean(boot_means)

```

```

113 # bootstrap_data_frame <- data.frame(Bootstrap = 1:num_bootstrap,
    Mean = boot_means)
114
115 # p <- ggplot(bootstrap_data_frame, aes(x = Mean)) +
116 #   geom_histogram(bins = 30, fill = "blue", alpha = 0.6) +
117 #   geom_vline(aes(xintercept = mean_val), color = "red", linetype
    = "dashed") +
118 #   labs(title = "Distribution des moyennes bootstrap",
119 #         x = "Moyenne bootstrap", y = "Fréquence") +
120 #   theme_minimal()
121 #
122 # return(list(mean = mean_val, ci = ci, plot = p))
123 #}
124 calculate_ec <- function(response, concentration) {
125   tryCatch({
126     #logistic model
127     fit <- nls(response ~ 1 / (1 + exp(-b * (log(concentration) -
        log($EC_{50}$)))),
128               start = list(b = 1, EC50 = median(concentration)),
129               control = nls.control(warnOnly = TRUE))
130     #coefficients
131     b <- coef(fit)["b"]
132     EC50 <- coef(fit)["EC50"]
133
134     #EC values
135     EC10 <- exp((log(10) - b * log(EC50)) / b)
136     EC20 <- exp((log(20) - b * log(EC50)) / b)
137
138     return(c(EC10 = EC10, EC20 = EC20, EC50 = EC50))
139   }, error = function(e) {
140     return(c(EC10 = NA, EC20 = NA, EC50 = NA))
141   })
142 }
143 analyze_bootstrap <- function(data, num_bootstrap = 1000) {
144   data$response <- as.numeric(as.character(data$response))
145   data$concentration <- as.numeric(as.character(data$concentration))
146   data <- data[!is.na(data$response) & !is.na(data$concentration), ]
147
148   if (nrow(data) == 0) {
149     stop("No valid numeric data found in the response or
        concentration columns.")
150   }
151
152   boot_ecs <- replicate(num_bootstrap, {
153     sample_indices <- sample(seq_len(nrow(data)), replace = TRUE)
154     sample_data <- data[sample_indices, ]
155     calculate_ec(sample_data$response, sample_data$concentration)
156   })
157
158   boot_ecs <- t(boot_ecs)
159   colnames(boot_ecs) <- c("EC10", "EC20", "EC50")
160

```

```

161 ec10_ci <- quantile(boot_ecs[, "EC10"], probs = c(0.025, 0.975),
162                   na.rm = TRUE)
163 ec20_ci <- quantile(boot_ecs[, "EC20"], probs = c(0.025, 0.975),
164                   na.rm = TRUE)
165 ec50_ci <- quantile(boot_ecs[, "EC50"], probs = c(0.025, 0.975),
166                   na.rm = TRUE)
167
168 mean_ec10 <- mean(boot_ecs[, "EC10"], na.rm = TRUE)
169 mean_ec20 <- mean(boot_ecs[, "EC20"], na.rm = TRUE)
170 mean_ec50 <- mean(boot_ecs[, "EC50"], na.rm = TRUE)
171
172 bootstrap_data_frame <- data.frame(Bootstrap = 1:num_bootstrap,
173                                     EC10 = boot_ecs[, "EC10"],
174                                     EC20 = boot_ecs[, "EC20"], EC50
175                                     = boot_ecs[, "EC50"])
176
177 p <- ggplot(bootstrap_data_frame, aes(x = EC50)) +
178   geom_histogram(bins = 30, fill = "blue", alpha = 0.6) +
179   geom_vline(aes(xintercept = mean_ec50), color = "red", linetype
180             = "dashed") +
181   labs(title = "Distribution des valeurs EC50 bootstrap",
182        x = "EC50 bootstrap", y = "Fréquence") +
183   theme_minimal()
184
185 return(list(mean_ec10 = mean_ec10, ci_ec10 = ec10_ci,
186            mean_ec20 = mean_ec20, ci_ec20 = ec20_ci,
187            mean_ec50 = mean_ec50, ci_ec50 = ec50_ci, plot = p))
188 }
189
190 #Fonction permutation
191 analyze_permutation <- function(data_long, num_permutations = 1000)
192 {
193   model <- aov(response ~ concentration, data = data_long)
194   obs_stat <- summary(model)[[1]]$'Pr(>F) '[1]
195
196   if (!is.numeric(obs_stat)) {
197     stop("La statistique observée n'est pas numérique.") }
198
199   perm_stats <- numeric(num_permutations)
200
201   for (i in 1:num_permutations) {
202     perm_data <- data_long %>%
203       mutate(response = sample(response, replace = FALSE))#response
204       et non Density
205     perm_model <- aov(response ~ concentration, data = perm_data)
206     perm_stat <- summary(perm_model)[[1]]$'Pr(>F) '[1]
207
208     if (is.numeric(perm_stat)) {
209       perm_stats[i] = perm_stat
210     } else {
211       perm_stats[i] = NA
212     }
213   }
214 }

```

```

205   }
206 }
207
208 perm_stats <- perm_stats[!is.na(perm_stats)]
209 p_value <- mean(perm_stats <= obs_stat)
210 significance_threshold <- 0.05
211 result_message <- ifelse(p_value < significance_threshold, "Test_
    positif_(p<0.05)", "Test_négatif_(p>=0.05)")
212
213 perm_data_frame <- data.frame(Permutation = 1:length(perm_stats),
    Statistic = perm_stats)
214
215 pvalue_plot <- ggplot(perm_data_frame, aes(x = Statistic)) +
216   geom_histogram(bins = 30, fill = "blue", alpha = 0.6) +
217   geom_vline(aes(xintercept = obs_stat), color = "red", linetype =
    "dashed") +
218   labs(title = "Distribution_des_P-values_des_Permutations",
219     x = "P-value_permutée", y = "Fréquence") +
220   theme_minimal()
221
222 return(list(result = data.frame(Test = "Permutation", PValue = p_
    value, Result = result_message),
223     permutations = perm_data_frame,
224     plot = pvalue_plot))
225 }
226
227 # Fonction Bayesien
228 analyze_bayesian <- function(nchain, prop_sd, log_concentration, n_
    animals, n_deaths) {#bioassay
229   chain <- matrix(NA, nchain + 1, 2)
230   colnames(chain) <- paste("beta", 0:1, sep = "")
231   chain[1, ] <- rep(0, 2)
232   acc_rate <- 0
233
234   for (i in 1:nchain) {
235     current <- chain[i, ]
236     prop <- rnorm(2, current, prop_sd)
237     prob_success <- exp(prop[1] + prop[2] * log_concentration) / (1
    + exp(prop[1] + prop[2] * log_concentration))
238     top <- sum(n_deaths * log(prob_success) + (n_animals - n_deaths)
    * log1p(-prob_success))
239     prob_success <- exp(current[1] + current[2] * log_concentration)
    / (1 + exp(current[1] + current[2] * log_concentration))
240     bottom <- sum(n_deaths * log(prob_success) + (n_animals - n_
    deaths) * log1p(-prob_success))
241     acc_prob <- exp(top - bottom)
242
243     if (!is.finite(acc_prob)) acc_prob <- 0
244     if (runif(1) < acc_prob) {
245       chain[i + 1, ] <- prop
246       acc_rate <- acc_rate + 1
247     } else {

```

```

248     chain[i + 1, ] <- current
249   }
250 }
251
252 acc_rate <- acc_rate / nchain
253 return(list(chain = chain, acc_rate = acc_rate))
254 }

```

Interface utilisateur : UI

```

1 #Fichier CSS:
2 sass(sass_file("style.scss"), output = "www/style.css")
3 ui <- fluentPage(
4   tags$head(
5     tags$link(href = "style.css", rel = "stylesheet", type = "text/
6       css")),
7   div(
8     class = "grid-container",
9     div(class = "header", header()),
10    div(class = "sidenav", tagList(navigation()))
11  ),
12  div(
13    class = "main",
14    uiOutput("progressIndicator"),
15    router_ui(
16      route("/", homePage()),
17      route("analysis", analysisPage()),
18      route("permutation", permutationPage()),
19      route("bootstrap", bootstrapPage()),
20      route("bayesian", bayesianPage()),
21      route("Select", BayoyoPage()),
22      route("calcPuiss", PuissancePage()),
23      route("Essai201", Essai201Page()),
24      route("Essai202", Essai202Page()),
25      route("Essai203", Essai203Page()),
26      route("Essai205", Essai205Page()),
27      route("Essai206", Essai206Page()),
28      route("Essai208", Essai208Page()),
29      route("Essai210", Essai210Page()),
30      route("Essai211", Essai211Page()),
31      route("Essai216", Essai216Page()),
32      route("Essai218", Essai218Page()),
33      route("Essai213214", Essai213214Page()),
34      route("Essai221", Essai221Page()),
35      route("Essai222", Essai222Page()),
36      route("Essai223", Essai223Page()),
37      route("Essai225", Essai225Page()),
38      route("Essai226", Essai226Page()),
39      route("Essai227", Essai227Page()),
40      route("Essai232", Essai232Page()),
41      route("Essai239", Essai239Page()),
42      route("Essai245", Essai245Page()),
43      route("Essai246247", Essai246247Page()),

```

```

43     route("ESCORT", ESCORTPage())
44
45   )
46 },
47   div(class = "footer", footer())
48 )
49 )

```

SERVER:

```

1  server <- function(input, output, session) {
2    router_server(root_page = "/")
3    global_data <- reactiveVal()
4    progress <- reactiveVal(0)
5    output$progressIndicator <- renderUI({
6      ProgressIndicator(label = "Processing data...", percentComplete
7        = progress(1))
8    })
9    ## visualisation data/stats desc   ###
10   observeEvent(input$submit, {
11     req(input$submit)
12     df <- reactive({
13       isolate({
14         if (!is.null(input$file_analysis)) {
15           file_data <- read.table(input$file_analysis$datapath,
16             header = input$header_analysis, sep = input$sep_
17             analysis)
18           global_data(file_data)
19           return(file_data)
20         } else if (input$textdata != "") {
21           file_data <- read_delim(text(input$textdata), delim =
22             input$sep_analysis)
23           global_data(file_data)
24           return(file_data)
25         } else {
26           return(NULL)
27         }
28       })
29     })
30   })
31   output$uploaded_data <- renderDT({
32     df_data <- df()
33     if (!is.null(df_data)) {
34       datatable(df_data)
35     } else {
36       NULL
37     }
38   })
39   databb <- reactive({
40     req(input$submit)
41     isolate({
42       if (!is.null(input$file_analysis)) {

```

```

40     file_data <- readLines(input$file_analysis$datapath, header
41       = input$header_analysis, sep = input$sep_analysis)
42     global_data(file_data)
43     return(file_data)
44   } else if (input$textdata != "") {
45     file_data <- read_delim(text(input$textdata), delim = input$
46       sep_analysis)
47     global_data(file_data)
48     return(file_data)
49   } else {
50     return(NULL)
51   }
52 }
53 # Visualisation des données
54 observeEvent(global_data(), {
55   df <- global_data()
56   output$plot_visualization <- renderPlotly({
57     req(df)
58     df_long <- df %>%
59       pivot_longer(cols = -Replicat, names_to = "concentration",
60         values_to = "response") %>%
61       drop_na(response) %>%
62       mutate(concentration = factor(concentration)) %>%
63       mutate(response = as.numeric(response))
64     DataMean <- SummaryVisu(df_long, 'response', 'concentration')
65     visualization <- #ggplot(df_long, aes(x = as.numeric(
66       concentration), y = response)) +
67       #geom_point(size = 2, alpha = 0.7) +
68       #scale_x_log10() +
69       #labs(
70       #   x = paste0("concentration (", input$conc_unit, ")"),
71       #   y = "response"
72       #) +
73       #theme_minimal(base_size = 15) +
74       #theme(axis.text = element_text(size = 14),
75       #   axis.title = element_text(size = 16))
76     ggplot(df_long, aes(x = concentration, y = response, color =
77       Replicat)) +
78     geom_point(size = 2, alpha = 0.7) +
79     labs(title = "Réplicats en fonction des concentrations",
80       x = "concentration",
81       y = "Réponse") +
82     theme_minimal() +
83     theme(axis.text.x = element_text(angle = 45, hjust = 1))
84
85     visualization <- visualization + geom_point(data = DataMean,
86       aes(x = concentration, y = Means), color = "black", size =
87       3, shape = 8) +
88     geom_line(data = DataMean, aes(x = concentration, y = Means,
89       group = 1), color = "black", linewidth = 1)

```

```

84     ggplotly(visualization)
85   })
86 })
87 output$desc_stats <- renderDT({
88   df <- global_data()
89   if (!is.null(df)) {
90     df_long <- df %>%
91       pivot_longer(cols = -Replicat, names_to = "concentration",
92         values_to = "response") %>%
93       drop_na(response) %>%
94       mutate(concentration = factor(concentration)) %>%
95       mutate(response = as.numeric(response))
96     DataMean <- SummaryVisu(df_long, 'response', 'concentration')
97     datatable(DataMean)
98   } else {
99     NULL
100   }
101 })
102 observeEvent(global_data(), {#ou input$file_analysis à la place de
103   global_data()
104   df <- global_data()
105
106   output$plot_box <- renderPlotly({
107     req(df)
108     df_long <- df %>%
109       pivot_longer(cols = -Replicat, names_to = "concentration",
110         values_to = "response") %>%
111       drop_na(response) %>%
112       mutate(concentration = factor(concentration)) %>%
113       mutate(response = as.numeric(response))
114
115     boxP <- ggplot(df_long, aes(x = concentration, y = response,
116       colour = concentration, fill = concentration)) +
117       geom_boxplot(alpha = 0.5, outlier.shape = NA) +
118       geom_jitter(position = position_jitter(0.2), size = 2, alpha
119         = 0.7) +
120       stat_summary(fun = mean, geom = "point", shape = 8, size =
121         3, color = "black", show.legend = FALSE) +
122       labs(title = "Boxplot",
123         x = "concentration",
124         y = "Réponse",
125         caption = "*□:□moyenne") +
126       theme_minimal() +
127       theme(legend.position = "none", plot.caption = element_text(
128         hjust = 0, face = "italic"))
129
130     ggplotly(boxP)
131   })
132 })
133
134 ##### Outliers/Transformation/effet solvant #####
135 observe({

```



```

129   if (!is.null(input$apply_transformation) && input$apply_
130       transformation) {
131     output$apply_transformation_menu <- renderUI({
132       selectInput("transformation_type", "Choose transformation:",
133                   choices = c("Aucune"="Aucune",
134                               "Reduce variance (Box-Cox)" = "
135                               boxcox",
136                               "Normalize distribution (Logarithmic
137                               )" = "log",
138                               "Stabilize variance (Square root)" =
139                               "sqrt"),
140                   selected = "Aucune")
141     })
142   } else {
143     output$apply_transformation_menu <- renderUI({
144       NULL
145     })
146   }
147 }
148
149 transformed_data <- reactive({
150   req(global_data())
151   df2 <- global_data()
152
153   # Gestion des outliers
154   if (!is.null(input$manage_outliers) && input$manage_outliers) {
155     numeric_cols <- sapply(df2, is.numeric)
156     df2_numeric <- df2[, numeric_cols]
157     Q1 <- apply(df2_numeric, 2, quantile, 0.25, na.rm = TRUE)
158     Q3 <- apply(df2_numeric, 2, quantile, 0.75, na.rm = TRUE)
159     IQR <- Q3 - Q1
160     outlier_filter <- apply(df2_numeric, 1, function(row) {
161       any(row < Q1 - 1.5 * IQR | row > Q3 + 1.5 * IQR)
162     })
163
164     # Affichage des indices des lignes identifiées comme outliers
165     outliers_indices <- which(outlier_filter)
166     print(outliers_indices)
167
168     df2_outliers <- df2[outliers_indices, ] # Sélection des
169     outliers
170     df2 <- df2[!outlier_filter, ] # Filtrer les outliers
171   }
172
173   # Application des transformations
174   if (!is.null(input$apply_transformation) && input$apply_
175       transformation) {
176     if (input$transformation_type == "boxcox")
177       {df2 <- as.data.frame(lapply(df2, function(col) {
178         if (is.numeric(col) && all(col > 0, na.rm = TRUE)) { # Pour
179           verifier si les données sont toutes positives
180           boxcox_result <- MASS::boxcox(col ~ 1, lambda = seq(-5,

```

```

5, 0.1))
174   if (length(boxcox_result$y) == length(col)) {
175     return(boxcox_result$y) # envoi la transformation
      uniquement si les longueurs correspondent
176   } else {
177     warning("La transformation de Box-Cox a échoué pour
      une colonne.")
178     return(col)
179   }
180   } else {
181     return(col)
182   }
183   })))
184 }
185 else if (input$transformation_type == "log") {
186   df2 <- as.data.frame(lapply(df2, function(col) {
187     if (is.numeric(col)) {
188       return(log(col + 1))
189     } else {
190       return(col)
191     }
192   })))
193 } else if (input$transformation_type == "sqrt") {
194   df2 <- as.data.frame(lapply(df2, function(col) {
195     if (is.numeric(col)) {
196       return(sqrt(col))
197     } else {
198       return(col)
199     }
200   })))
201 }
202 }
203 # Affichage des outliers dans un tableau (si disponible)
204 if (exists("df2_outliers")) {
205   output$outliers_table <- renderTable({
206     df2_outliers
207   })
208 }
209
210 df2
211 })
212 output$plot_box2 <- renderPlotly({
213   req(transformed_data())
214
215   df_long2 <- transformed_data() %>%
216     pivot_longer(cols = -Replicat, names_to = "concentration",
217       values_to = "response") %>%
218     drop_na(response) %>%
219     mutate(concentration = factor(concentration)) %>%
220     mutate(response = as.numeric(response))
221
222   boxP2 <- ggplot(df_long2, aes(x = concentration, y = response,

```

```

    colour = concentration, fill = concentration)) +
222   geom_boxplot(alpha = 0.5, outlier.shape = NA) +
223   geom_jitter(position = position_jitter(0.2), size = 2, alpha =
    0.7) +
224   stat_summary(fun = mean, geom = "point", shape = 8, size = 3,
    color = "black", show.legend = FALSE) +
225   labs(title = "Boxplot après transformation",
226        x = "concentration",
227        y = "Response",
228        caption = "* : mean") +
229   theme_minimal() +
230   theme(legend.position = "none", plot.caption = element_text(
    hjust = 0, face = "italic"))
231
232   ggplotly(boxP2)
233 })
234
235 output$dataTF <- renderTable({
236   req(transformed_data ())
237   as.data.frame(transformed_data())
238 })
239
240 output$effect_solvantValue <- renderText({
241   if (!is.null(input$effect_solvant) && input$effect_solvant) {
242     "Solvent effect is activated."
243   } else {
244     "Solvent effect is deactivated."
245   }
246 })
247
248 output$manage_outliersValue <- renderText({
249   if (!is.null(input$manage_outliers) && input$manage_outliers) {
250     "Outlier handling is activated."
251   } else {
252     "Outlier handling is deactivated."
253   }
254 })
255
256 output$apply_transformationValue <- renderText({
257   if (!is.null(input$apply_transformation) && input$apply_
    transformation) {
258     "Transformation is applied."
259   } else {
260     "Transformation is deactivated."
261   }
262 })
263
264
265
266 # Application de l'effet solvant
267 # Fonction réactive pour appliquer l'effet solvant
268 effetSov_data <- reactive({

```

```

269 req(input$effect_solvant)
270 req(global_data())
271
272 dfsolv <- global_data()
273
274 # Vérification colonnes
275 if ("controle" %in% names(dfsolv) && "controle_solvant" %in%
    names(dfsolv)) {
276
277   # Test de Student
278   test_result <- t.test(dfsolv$controle, dfsolv$controle_solvant
    )
279
280   # résultat du test
281   output$test_result <- renderPrint({
282     test_result
283   })
284
285   # Si test non significatif, concaténer les colonnes
286   if (test_result$p.value > 0.05)
287
288   {
289     #dfsolv$concentration[df$concentration %in% c("control", "
    control solvent")] <- "0"
290     dfsolv <- dfsolv %>%
291       mutate(concentration = ifelse(concentration %in% c("
    controle", "controle_solvant"), "0", concentration))
292     cat("Effet_solvant_non_significatif, colonnes combinées en
    '0'.\n")
293   } else {
294     cat("Effet_solvant_significatif, ne pas combiner les
    colonnes.\n")
295   }
296
297   #{dfsolv$'0' <- c(dfsolv$controle, dfsolv$controle_solvant)
298   # Supprimer les anciennes colonnes
299   #dfsolv <- dfsolv %>% select(-controle, -controle_solvant)}
300
301   # Mise à jour de la table de données
302   output$dataTable <- renderDT({
303     datatable(dfsolv)
304   })
305
306 } else {
307   output$test_result <- renderText("Effet_solvant_désactivé.")
308   output$dataTable <- renderDT({
309     datatable(dfsolv)
310   })
311 }
312
313 return(dfsolv)
314 })

```

```

315 # Mise à jour du boxplot en fonction des données modifiées
316 observeEvent(effetSov_data(), {
317   df <- effetSov_data()
318
319
320   output$plot_box <- renderPlotly({
321     req(df)
322
323     df_long <- df %>%
324       pivot_longer(cols = -Replicat, names_to = "concentration",
325         values_to = "response") %>%
326       drop_na(response) %>%
327       mutate(concentration = factor(concentration)) %>%
328       mutate(response = as.numeric(response))
329
330     boxP <- ggplot(df_long, aes(x = concentration, y = response,
331       colour = concentration, fill = concentration)) +
332       geom_boxplot(alpha = 0.5, outlier.shape = NA) +
333       geom_jitter(position = position_jitter(0.2), size = 2, alpha
334         = 0.7) +
335       stat_summary(fun = mean, geom = "point", shape = 8, size =
336         3, color = "black", show.legend = FALSE) +
337       labs(title = "Boxplot",
338         x = "Concentration",
339         y = "Réponse",
340         caption = "*□:□moyenne") +
341       theme_minimal() +
342       theme(legend.position = "none", plot.caption = element_text(
343         hjust = 0, face = "italic"))
344
345     ggplotly(boxP)
346   })
347 })
348
349 # Modelisation
350 ## #Methodes Bayésiennes#
351 #####Selection modele(cf method rapport)
352 observeEvent(input$run_analysis, {
353   req(input$file)
354
355   # Chargement des données si on utilise pas les données global
356   # du début(anayle independante)
357   data <- read.csv(input$file$datapath, header = FALSE)
358   colnames(data) <- c("Replicat", "Control", "Control_Solvent",
359     "0.1", "0.15", "0.23", "0.34", "0.51", "0.76", "1.14")
360   data <- data[,-c(2,3)] # Suppression des colonnes Control et
361     Control_Solvent
362   data <- melt(data, id.vars = "Replicat", variable.name = "
363     concentration", value.name = "response")
364   data$concentration <- as.numeric(as.character(data$
365     concentration))

```

```

357
358 # Préparation des données pour l'analyse
359 data <- data[!is.na(data$response), ]
360 data_aggregated <- aggregate(response ~ concentration, data =
    data, FUN = mean)
361
362 # Fonctions de vraisemblance
363 logLikelihood_loglogistic <- function(params, concentration,
    response) {
364   alpha <- params[1]
365   beta <- params[2]
366   predicted <- 1 / (1 + (concentration / beta) ^ (-alpha))
367   ll <- sum(dlogis(response, location = predicted, scale =
    alpha, log = TRUE))
368   return(ll)
369 }
370
371 logLikelihood_lognormal <- function(params, concentration,
    response) {
372   meanlog <- params[1]
373   sdlog <- params[2]
374   predicted <- plnorm(concentration, meanlog, sdlog)
375   ll <- sum(dlnorm(response, meanlog = predicted, sdlog =
    sdlog, log = TRUE))
376   return(ll)
377 }
378
379 logLikelihood_weibull <- function(params, concentration,
    response) {
380   shape <- params[1]
381   scale <- params[2]
382   predicted <- pweibull(concentration, shape = shape, scale =
    scale)
383   ll <- sum(dweibull(response, shape = predicted, scale =
    scale, log = TRUE))
384   return(ll)
385 }
386 prior_uniform <- function(params) {
387   if (all(params > 0 & params < 10)) {
388     return(1)
389   } else {
390     return(0)
391   }
392 }
393 posterior_unnormalized <- function(params, logLikelihoodFunc,
    priorFunc, concentration, response) {
394   likelihood <- exp(logLikelihoodFunc(params, concentration,
    response))
395   prior <- priorFunc(params)
396
397   if (is.na(likelihood) || is.na(prior) || likelihood == 0 ||
    prior == 0) {

```

```

398     return(0)
399   }
400   return(likelihood * prior)
401 }
402 metropolis_hastings <- function(logLikelihoodFunc, priorFunc,
403   concentration, response, start, iterations, proposal_sd) {
404   current_params <- start
405   chain <- matrix(NA, nrow = iterations, ncol = length(start))
406   acceptance <- 0
407   for (i in 1:iterations) {
408     proposed_params <- current_params + rnorm(length(start),
409       mean = 0, sd = proposal_sd)
410     current_posterior <- posterior_unnormalized(current_params,
411       logLikelihoodFunc, priorFunc, concentration, response)
412     proposed_posterior <- posterior_unnormalized(proposed_params,
413       logLikelihoodFunc, priorFunc, concentration, response)
414     acceptance_ratio <- proposed_posterior / current_posterior
415     if (!is.na(acceptance_ratio) && acceptance_ratio > 0 &&
416       runif(1) < acceptance_ratio) {
417       current_params <- proposed_params
418       acceptance <- acceptance + 1
419     }
420     chain[i, ] <- current_params
421   }
422   acceptance_rate <- acceptance / iterations
423   list(chain = chain, acceptance_rate = acceptance_rate)
424 }
425 # Metropolis-Hastings
426 mh_results_loglogistic <- metropolis_hastings(
427   logLikelihoodFunc = logLikelihood_loglogistic,
428   priorFunc = prior_uniform,
429   concentration = data_aggregated$concentration,
430   response = data_aggregated$response,
431   start = c(5, 5),
432   iterations = input$iterations,
433   proposal_sd = c(0.7, 1.0)
434 )
435 mh_results_lognormal <- metropolis_hastings(
436   logLikelihoodFunc = logLikelihood_lognormal,
437   priorFunc = prior_uniform,
438   concentration = data_aggregated$concentration,
439   response = data_aggregated$response,
440   start = c(5, 5),
441   iterations = input$iterations,
442   proposal_sd = c(0.7, 1.0)
443 )
444 mh_results_weibull <- metropolis_hastings(
445   logLikelihoodFunc = logLikelihood_weibull,

```

```

443     priorFunc = prior_uniform,
444     concentration = data_aggregated$concentration,
445     response = data_aggregated$response,
446     start = c(5, 5),
447     iterations = input$iterations,
448     proposal_sd = c(0.7, 1.0)
449   )
450
451   burn_in <- input$burn_in
452
453   post_burn_in_loglogistic <- mh_results_loglogistic$chain[-(1:
     burn_in), ]
454   post_burn_in_lognormal <- mh_results_lognormal$chain[-(1:burn_
     in), ]
455   post_burn_in_weibull <- mh_results_weibull$chain[-(1:burn_in),
     ]
456
457   marginal_likelihood_loglogistic <- mean(apply(post_burn_in_
     loglogistic, 1, function(params) {
458     posterior_unnormalized(params, logLikelihood_loglogistic,
       prior_uniform, data_aggregated$concentration, data_
       aggregated$response)
459   })))
460
461   marginal_likelihood_lognormal <- mean(apply(post_burn_in_
     lognormal, 1, function(params) {
462     posterior_unnormalized(params, logLikelihood_lognormal,
       prior_uniform, data_aggregated$concentration, data_
       aggregated$response)
463   })))
464
465   marginal_likelihood_weibull <- mean(apply(post_burn_in_weibull
     , 1, function(params) {
466     posterior_unnormalized(params, logLikelihood_weibull, prior_
       uniform, data_aggregated$concentration, data_aggregated$
       response)
467   })))
468
469   BF_loglogistic_lognormal <- marginal_likelihood_loglogistic /
     marginal_likelihood_lognormal
470   BF_loglogistic_weibull <- marginal_likelihood_loglogistic /
     marginal_likelihood_weibull
471   BF_lognormal_weibull <- marginal_likelihood_lognormal /
     marginal_likelihood_weibull
472
473   output$bayes_factors <- renderPrint({
474     cat("Bayes Factors:\n")
475     cat("Loglogistic vs Lognormal:", BF_loglogistic_lognormal, "
       \n")
476     cat("Loglogistic vs Weibull:", BF_loglogistic_weibull, "\n")
477     cat("Lognormal vs Weibull:", BF_lognormal_weibull, "\n")
478   })

```



```

479 output$acceptance_rates <- renderPrint({
480   cat("Acceptance_Rates:\n")
481   cat("Loglogistic_Model:", mh_results_loglogistic$acceptance_
482     rate, "\n")
483   cat("Lognormal_Model:", mh_results_lognormal$acceptance_rate
484     , "\n")
485   cat("Weibull_Model:", mh_results_weibull$acceptance_rate, "\
486     n")
487 })
488
489 output$plot_loglogistic_1 <- renderPlot({
490   plot(post_burn_in_loglogistic[,1], type = "l", main = "Trace
491     _plot_for_Loglogistic_Model_-_Parameter_1", ylab = "Value
492     ", xlab = "Iteration")
493 })
494
495 output$plot_loglogistic_2 <- renderPlot({
496   plot(post_burn_in_loglogistic[,2], type = "l", main = "Trace
497     _plot_for_Loglogistic_Model_-_Parameter_2", ylab = "Value
498     ", xlab = "Iteration")
499 })
500
501 output$plot_lognormal_1 <- renderPlot({
502   plot(post_burn_in_lognormal[,1], type = "l", main = "Trace
503     _plot_for_Lognormal_Model_-_Parameter_1", ylab = "Value",
504     xlab = "Iteration")
505 })
506
507 output$plot_lognormal_2 <- renderPlot({
508   plot(post_burn_in_lognormal[,2], type = "l", main = "Trace
509     _plot_for_Lognormal_Model_-_Parameter_2", ylab = "Value",
510     xlab = "Iteration")
511 })
512
513 output$plot_weibull_1 <- renderPlot({
514   plot(post_burn_in_weibull[,1], type = "l", main = "Trace
515     _plot_for>Weibull_Model_-_Parameter_1", ylab = "Value",
516     xlab = "Iteration")
517 })
518
519 output$plot_weibull_2 <- renderPlot({
520   plot(post_burn_in_weibull[,2], type = "l", main = "Trace
521     _plot_for>Weibull_Model_-_Parameter_2", ylab = "Value",
522     xlab = "Iteration")
523 })
524
525 ##### #Test Bayesien#
526
527 observeEvent(global_data(), {

```

```

516 df <- global_data()
517 output$column_select <- renderUI({
518   colnames <- names(df)
519   tagList(
520     selectInput("column1", "Select Column 1:", choices =
521       colnames),
522     selectInput("column2", "Select Column 2:", choices =
523       colnames)
524   )
525 })
526
527 values <- reactiveValues()
528
529 observe({
530   req(input$column1, input$column2)
531   col1 <- input$column1
532   col2 <- input$column2
533   if (!is.numeric(df[[col1]]) || !is.numeric(df[[col2]])) {
534     return(NULL)
535   }
536
537   if (input$design == "single") {
538     t_val <- t.test(df[[col1]], df[[col2]], paired = TRUE)$
539       statistic
540     n1 <- nrow(df)
541     result <- ttest.tstat(t = t_val, n1 = n1)
542   } else {
543     t_val <- t.test(df[[col1]], df[[col2]], paired = FALSE)$
544       statistic
545     n1 <- length(df[[col1]])
546     n2 <- length(df[[col2]])
547     result <- ttest.tstat(t = t_val, n1 = n1, n2 = n2)
548   }
549   values$BF10 <- exp(result[['bf']])
550   values$BF01 <- 1 / exp(result[['bf']])
551   values$pOdds <- input$pH0 / (1 - input$pH0)
552 })
553
554 output$bfText <- renderUI({
555   if (is.null(values$BF10)) {
556     shiny::flushReact()
557     return()
558   }
559
560   if (values$BF01 > 1) {
561     p(withMathJax(sprintf("The Bayes factor for the null is
562       \\(\\text{B}_{0/1}\\) = %.2f", values$BF01)))
563   } else if (values$BF10 > 1) {
564     p(withMathJax(sprintf("The Bayes factor for the alternative
565       is \\(\\text{B}_{1/0}\\) = %.2f", values$BF10)))
566   }
567 })

```

```

562 output$plotOutput <- renderPlot({
563   if (is.null(values$BF10)) return(NULL)
564
565   bf_data <- data.frame(
566     Hypothesis = c("H0", "H1"),
567     BayesFactor = c(values$BF01, values$BF10)
568   )
569
570   ggplot(bf_data, aes(x = Hypothesis, y = BayesFactor, fill =
571     Hypothesis)) +
572     geom_bar(stat = "identity") +
573     theme_minimal() +
574     labs(title = "Bayes_Factor_Comparison", x = "Hypothesis", y
575       = "Bayes_Factor") +
576     scale_fill_manual(values = c("H0" = "skyblue", "H1" = "
577       orange"))
578 })
579
580 output$description <- renderUI({
581   if (is.null(values$BF10)) {
582     shiny:::flushReact()
583     return()
584   }
585
586   if (values$BF01 > 1) {
587     p(withMathJax(sprintf("The_Bayes_factor_for_the_null_is_
588       \\(\\text{B}_{0/1}\\) = %.2f", values$BF01)))
589   } else if (values$BF10 > 1) {
590     p(withMathJax(sprintf("The_Bayes_factor_for_the_alternative_
591       is_\\(\\text{B}_{1/0}\\) = %.2f", values$BF10)))
592   }
593
594   if (values$BF10 > 1) {
595     p(withMathJax(sprintf("This_means_that_the_observed_data_are
596       approximately %.2f times more likely under \\(\\mathcal{H}_1\\)
597       than under \\(\\mathcal{H}_0\\)", values$BF10)))
598   } else {
599     p(withMathJax(sprintf("This_means_that_the_observed_data_are
600       approximately %.2f times more likely under \\(\\mathcal{H}_0\\)
601       than under \\(\\mathcal{H}_1\\)", values$BF01)))
602   }
603 })
604
605 output$postNull <- renderUI({
606   p(withMathJax(sprintf("The_posterior_probability_for_\\(\\mathcal{H}_0\\)
607     is %.4f", (values$BF01 * values$pOdds) / (1
608       + (values$BF01 * values$pOdds)))))
609 })
610
611 output$postAlt <- renderUI({
612   p(withMathJax(sprintf("The_posterior_probability_for_\\(\\mathcal{H}_1\\)

```

```

         $\mathcal{H}_1$  is %.4f", 1 - (values$BF01 * values$pOdds)
        / (1 + (values$BF01 * values$pOdds))))))
    })
  })

output$altText <- renderUI({
  if (input$direction == "none") {
    p(withMathJax(sprintf('\\(\\mathcal{H}_1\\): Un effet détecté')))
  } else if (input$direction == "positive") {
    p(withMathJax(sprintf('\\(\\mathcal{H}_1\\): Un effet positif détecté')))
  } else if (input$direction == "negative") {
    p(withMathJax(sprintf('\\(\\mathcal{H}_1\\): Un effet négatif détecté')))
  }
})
})

#### #Estimation Bayesienne#

# 'values' init en dehors des blocs 'observeEvent'
values <- reactiveValues()

observeEvent(input$run_EstiBayesian, {
  df <- global_data()

  analysisB <- eventReactive(input$run_EstiBayesian, {
    req(input$column1, input$column2)
    col1 <- input$column1
    col2 <- input$column2
    prop_sd <- c(input$sd_beta0, input$sd_beta1)
    log_concentration <- log(df[[col1]])
    n_animals <- df[[col2]]
    n_deaths <- df$n_deaths
    analyze_bayesian(input$nchain, prop_sd, log_concentration, n_animals, n_deaths)
  })

  output$acc_rate <- renderUI({
    HTML(paste("Taux d'acceptation: <span style='color:red;'>",
      round(analysisB()$acc_rate, 3), "</span>"))
  })

  output$chain_plot <- renderPlot({
    req(analysisB())
    chain <- analysisB()$chain
    burnin <- input$burnin
    if (burnin >= nrow(chain)) {
      burnin <- nrow(chain) - 1
    }
    chain <- chain[-(1:burnin), ]
  })

```

```

648 EC10 <- (log(0.1) - chain[, "beta0"]) / chain[, "beta1"]
649 EC20 <- (log(0.2) - chain[, "beta0"]) / chain[, "beta1"]
650 EC50 <- (log(0.5) - chain[, "beta0"]) / chain[, "beta1"]
651 chain <- cbind(chain, EC10 = EC10, EC20 = EC20, EC50 = EC50)
652
653 par(mfrow = c(3, 2))
654 plot(mcmc(chain))
655 })
656
657 output$ec_values <- renderUI({
658   req(analysisB())
659   chain <- analysisB()$chain
660   burnin <- input$burnin
661   if (burnin >= nrow(chain)) {
662     burnin <- nrow(chain) - 1
663   }
664   chain <- chain[-(1:burnin), ]
665   EC10 <- (log(0.1) - chain[, "beta0"]) / chain[, "beta1"]
666   EC20 <- (log(0.2) - chain[, "beta0"]) / chain[, "beta1"]
667   EC50 <- (log(0.5) - chain[, "beta0"]) / chain[, "beta1"]
668
669   # estimations des ECx stoker dans des valeurs réactives(cf
670     code init)
671   values$EC10 <- EC10
672   values$EC20 <- EC20
673   values$EC50 <- EC50
674
675   ec10_mean <- mean(EC10)
676   ec10_sd <- sd(EC10)
677   ec20_mean <- mean(EC20)
678   ec20_sd <- sd(EC20)
679   ec50_mean <- mean(EC50)
680   ec50_sd <- sd(EC50)
681   HTML(paste(
682     "Estimation de l'EC10: 

```

```

693   }
694   chain <- chain[-(1:burnin), ]
695   EC10 <- (log(0.1) - chain[, "beta0"]) / chain[, "beta1"]
696   EC20 <- (log(0.2) - chain[, "beta0"]) / chain[, "beta1"]
697   EC50 <- (log(0.5) - chain[, "beta0"]) / chain[, "beta1"]
698
699   ec10_mean <- mean(EC10)
700   ec10_sd <- sd(EC10)
701   ec20_mean <- mean(EC20)
702   ec20_sd <- sd(EC20)
703   ec50_mean <- mean(EC50)
704   ec50_sd <- sd(EC50)
705
706   interpretation <- paste(
707     "L'analyse a été réalisée avec succès.<br><br>",
708     "Voici un résumé des résultats:<br>",
709     "- Taux d'acceptation des propositions: <span style='color: red;'>round(analysisB()$acc_rate, 3), </span><br>",
710     "- Estimation de l'EC10: <span style='color: red;'>round(ec10_mean, 3), </span> (écart-type: <span style='color: red;'>round(ec10_sd, 3), </span><br>",
711     "- Estimation de l'EC20: <span style='color: red;'>round(ec20_mean, 3), </span> (écart-type: <span style='color: red;'>round(ec20_sd, 3), </span><br>",
712     "- Estimation de l'EC50: <span style='color: red;'>round(ec50_mean, 3), </span> (écart-type: <span style='color: red;'>round(ec50_sd, 3), </span><br><br>",
713     "L'estimation des EC10, EC20 et EC50 (concentrations efficaces pour 10%, 20% et 50% des sujets) signifie que ces doses sont celles auxquelles on s'attend à ce que 10%, 20% et 50% des animaux exposés montrent une réponse spécifique.<br>",
714     "Ces estimations sont accompagnées d'une incertitude représentée par l'écart-type.<br>",
715     "Plus l'écart-type est faible, plus l'estimation est précise."
716   )
717   HTML(interpretation)
718 })
719 })
720
721 # Test d'hypotheses sur ECx et calcul de FB
722 observeEvent(input$test_hypothesis, {
723   req(values$EC10, values$EC20, values$EC50)
724
725   # Define the hypotheses
726   hyp_EC10 <- input$hyp_ec10
727   hyp_EC20 <- input$hyp_ec20
728   hyp_EC50 <- input$hyp_ec50
729
730   # proportion d'ech pour hypotheses
731   bf_EC10 <- mean(values$EC10 > hyp_EC10)

```

```

732   bf_EC20 <- mean(values$EC20 > hyp_EC20)
733   bf_EC50 <- mean(values$EC50 > hyp_EC50)
734
735   # resultats(dans valeurs reactives)
736   values$BF_EC10 <- bf_EC10
737   values$BF_EC20 <- bf_EC20
738   values$BF_EC50 <- bf_EC50
739 })
740
741 # Facteur de Bayes (FB)
742 output$bayes_factors <- renderUI({
743   req(values$BF_EC10, values$BF_EC20, values$BF_EC50)
744   HTML(paste(
745     "Facteur de Bayes pour EC10 > ", input$hyp_ec10, ": <span
       style='color:red;*>", round(values$BF_EC10, 3), "</span><br
       >",
746     "Facteur de Bayes pour EC20 > ", input$hyp_ec20, ": <span
       style='color:red;*>", round(values$BF_EC20, 3), "</span><br
       >",
747     "Facteur de Bayes pour EC50 > ", input$hyp_ec50, ": <span
       style='color:red;*>", round(values$BF_EC50, 3), "</span>"
748   ))
749 })
750
751 output$interpretation <- renderUI({
752   req(values$BF_EC10, values$BF_EC20, values$BF_EC50)
753
754   interpretation <- paste(
755     "Interprétation des résultats:<br>",
756     "- Un facteur de Bayes proche de 1 suggère que les données ne
       favorisent pas particulièrement une hypothèse par rapport à
       l'autre.<br>",
757     "- Un facteur de Bayes > 1 indique que les données sont plus
       probables sous l'hypothèse alternative (par exemple, ECx >
       hypothèse spécifiée).<br>",
758     "- Un facteur de Bayes < 1 indique que les données sont plus
       probables sous l'hypothèse nulle."
759   )
760   HTML(interpretation)
761 })
762
763 ##### #Resampling# #####
764 # Permutation.R(actionButton("run_permutation")+fonction permut
765
766 observeEvent(input$run_permutation, {
767   df <- global_data()
768   if (ncol(df) < 2) {
769     output$perm_result <- renderUI({
770       p("The data must have at least two columns for the
       permutation test.")
771     })
772     return()

```

```

773   }
774
775   df_long <- df %>%
776     pivot_longer(cols = -Replicat, names_to = "concentration",
777                 values_to = "response") %>%
778     drop_na(response) %>%
779     mutate(concentration = factor(concentration))%>%
780     mutate(response <- as.numeric(response))
781
782   #df_long <- df %>%
783   #pivot_longer(cols = -Replicat, names_to = "concentration",
784   #             values_to = "response") %>%
785   #drop_na(response) %>%
786   #mutate(concentration = suppressWarnings(as.numeric(as.
787   #      character(concentration))),
788   #      #response = suppressWarnings(as.numeric(response)))
789
790   analysis_results <- analyze_permutation(df_long)
791   output$results <- renderDT({
792     analysis_results$result
793   })
794
795   output$permutation_data <- renderDT({
796     analysis_results$permutations
797   })
798
799   output$permutation_plot <- renderPlot({
800     print(analysis_results$plot)
801   })
802
803   # Bootstrap.R(actionButton("run_bootstrap")+fonction boots
804   observeEvent(input$run_bootstrap, {
805     df <- global_data()
806     if (ncol(df) < 1) {
807       output$bootstrap_result <- renderUI({
808         p("The data must have at least one column for the bootstrap
809         test.")
810       })
811       return()
812     }
813
814     df_long <- df %>%
815       pivot_longer(cols = -Replicat, names_to = "concentration",
816                   values_to = "response") %>%
817       drop_na(response) %>%
818       mutate(concentration = factor(concentration))%>%
819       mutate(response <- as.numeric(response))
820
821     tryCatch({
822       analysis_results <- analyze_bootstrap(df_long)
823       output$bootstrap_result <- renderPrint({

```



```

820     paste("Mean_EC10:", analysis_results$mean_ec10, "\n95% CI_\n
      EC10:", analysis_results$ci_ec10[1], "-", analysis_
      results$ci_ec10[2],
821       "\nMean_EC20:", analysis_results$mean_ec20, "\n95% CI_\n
      EC20:", analysis_results$ci_ec20[1], "-", analysis_
      results$ci_ec20[2],
822       "\nMean_EC50:", analysis_results$mean_ec50, "\n95% CI_\n
      EC50:", analysis_results$ci_ec50[1], "-", analysis_
      results$ci_ec50[2])
823   })
824
825   output$bootstrap_plot <- renderPlot({
826     print(analysis_results$plot)
827   })
828   }, error = function(e) {
829     output$bootstrap_result <- renderPrint({
830       paste("Error:", e$message)
831     })
832   })
833 })
834
835 #####      #NOEC/LOEC#      ##
836 #      Dunnett      ###
837
838 observeEvent(input$run_dunnett, {
839   df <- global_data()
840   if (ncol(df) < 2) {
841     output$dunnett_result <- renderUI({
842       p("The data must have at least two columns for the Dunnett
      test.")
843     })
844     return()
845   }
846
847   analysis_results <- analyze_dunnett(df)
848   output$dunnett_result <- renderPrint({
849     paste("ANOVA Summary:", analysis_results$summary_aov,
850           "\nSignificance:", analysis_results$significant,
851           "\nNormality:", analysis_results$normality,
852           "\nHomoscedasticity:", analysis_results$homoscedasticity
853           ,
854           "\nDunnett Test:", analysis_results$summary_dunnett)
855   })
856
857   output$dunnett_plot <- renderPlot({
858     plot(analysis_results$Res_dunnett)
859   })
860
861 #####      ANOVA      #####
862
863

```

```

864 observeEvent(input$run_anova, {
865   df <- global_data()
866   if (ncol(df) < 2) {
867     output$anova_result <- renderUI({
868       p("The data must have at least two columns for the ANOVA
869         test.")
870     })
871     return()
872   }
873   analysis_results <- analyze_anova(df)
874   output$anova_result <- renderPrint({
875     paste("ANOVA Summary:", analysis_results$summary_aov,
876           "\nSignificance:", analysis_results$significant,
877           "\nNormality:", analysis_results$normality,
878           "\nHomoscedasticity:", analysis_results$homoscedasticity
879         })
880   })
881
882 #####      #Puissance#      ###
883   observeEvent(global_data(), {
884     df <- global_data()
885     output$column_select2 <- renderUI({
886       colnames <- names(df)
887       tagList(
888         updateSelectInput(session, "groupColumn_t", choices =
889           colnames),
890         updateSelectInput(session, "valueColumn_t", choices =
891           colnames),
892         updateSelectInput(session, "groupColumn_a", choices =
893           colnames),
894         updateSelectInput(session, "valueColumn_a", choices =
895           colnames),
896         updateSelectInput(session, "doseColumn_l", choices =
897           colnames),
898         updateSelectInput(session, "responseColumn_l", choices =
899           colnames)
900       )
901     })
902
903     # Calcul de la puissance et visualisation
904     output$powerPlot <- renderPlot({
905       req(global_data())
906
907       if (input$method == "t-test") {
908         req(input$groupColumn_t, input$valueColumn_t)
909         group_data <- df[, input$groupColumn_t]
910         value_data <- df[, input$valueColumn_t]
911         t_test <- t.test(value_data ~ group_data)
912         effect_size <- t_test$estimate / sd(value_data)

```

```

908     n <- length(value_data)
909     power_result <- pwr.t.test(n = n/2, d = effect_size, sig.
      level = input$alphaLevel_t, power = NULL)
910     plot(power_result)
911   } else if (input$method == "ANOVA") {
912     req(input$groupColumn_a, input$valueColumn_a)
913     group_data <- df[, input$groupColumn_a]
914     value_data <- df[, input$valueColumn_a]
915     aov_model <- aov(value_data ~ group_data)
916     summary_aov <- summary(aov_model)
917     f_value <- summary_aov[[1]][["F_value"]][1]
918     n <- length(value_data) / length(unique(group_data))
919     power_result <- pwr.anova.test(k = length(unique(group_data))
      ), n = n, f = f_value, sig.level = input$alphaLevel_a,
      power = NULL)
920     plot(power_result)
921   } else if (input$method == "Régression Log-Logistique") {
922     req(input$doseColumn_l, input$responseColumn_l)
923     dose_data <- df[, input$doseColumn_l]
924     response_data <- df[, input$responseColumn_l]
925     ll_model <- drm(response_data ~ dose_data, fct = LL.4())
926     plot(ll_model, type = "all")
927   }
928 })
929
930 output$powerResult <- renderPrint({
931   req(global_data())
932
933   if (input$method == "t-test") {
934     req(input$groupColumn_t, input$valueColumn_t)
935     group_data <- df[, input$groupColumn_t]
936     value_data <- df[, input$valueColumn_t]
937     t_test <- t.test(value_data ~ group_data)
938     effect_size <- t_test$estimate / sd(value_data)
939     n <- length(value_data)
940     power_result <- pwr.t.test(n = n/2, d = effect_size, sig.
      level = input$alphaLevel_t, power = NULL)
941     print(power_result)
942   } else if (input$method == "ANOVA") {
943     req(input$groupColumn_a, input$valueColumn_a)
944     group_data <- df[, input$groupColumn_a]
945     value_data <- df[, input$valueColumn_a]
946     aov_model <- aov(value_data ~ group_data)
947     summary_aov <- summary(aov_model)
948     f_value <- summary_aov[[1]][["F_value"]][1]
949     n <- length(value_data) / length(unique(group_data))
950     power_result <- pwr.anova.test(k = length(unique(group_data))
      ), n = n, f = f_value, sig.level = input$alphaLevel_a,
      power = NULL)
951     print(power_result)
952   } else if (input$method == "Régression Log-Logistique") {
953     req(input$doseColumn_l, input$responseColumn_l)

```

```

954     dose_data <- df[, input$doseColumn_1]
955     response_data <- df[, input$responseColumn_1]
956     ll_model <- drm(response_data ~ dose_data, fct = LL.4())
957     summary(ll_model)
958   }
959 })
960 })
961
962 #####          # Essais standardisés #          #####
963 #201#
964
965
966 intest_type201 <- reactive({ 'Essai201' })
967
968 validateFile <- function(filename) {
969   extFile <- tools::file_ext(filename)
970   validate(need(extFile == c("csv","txt"), "Seuls les fichiers CSV
971     sont acceptés. "))
972 }
973
974 filedata201 <- eventReactive(input$buttonRunStep201, {
975   if(intest_type201() == 'Essai201') {
976     req(input$datafile_ESSAI201)
977     validateFile(input$datafile_ESSAI201)
978     ff <- input$datafile_ESSAI201
979     read.csv(file = ff$datapath, header = TRUE)
980   }
981 })
982
983 output$rawdata201 <- DT::renderDataTable({
984   filedata201()
985 })
986
987 fitmodel201 <- eventReactive(input$buttonRunStep201, {
988   data <- filedata201()
989   if(intest_type201() == 'Essai201') {
990     if(input$model_ESSAI201 == '112') {
991       fit <- drm(response ~ as.numeric(concentration), data = data
992         , fct = LL.2())
993     } else if(input$model_ESSAI201 == '113') {
994       fit <- drm(response ~ as.numeric(concentration), data = data
995         , fct = LL.3())
996     } else if(input$model_ESSAI201 == '114') {
997       fit <- drm(response ~ as.numeric(concentration), data = data
998         , fct = LL.4())
999     }
1000     return(fit)
1001   }
1002 })
1003
1004 ECx201 <- eventReactive(input$buttonRunStep201, {
1005   fit <- fitmodel201()

```

```

1002   if(intest_type201() == 'Essai201') {
1003     XX <- as.numeric(input$ecx_ESSAI201)
1004     ec_values <- ED(fit, c(XX, 10, 50), interval = "delta",
1005                     display = FALSE)
1006
1007     drc_df <- data.frame(
1008       EC = c(XX, 10, 50),
1009       Estimate = ec_values[,1],
1010       'Standard Error' = ec_values[,2],
1011       'Lower 95% CI' = ec_values[,3],
1012       'Upper 95% CI' = ec_values[,4]
1013     )
1014
1015     drc_df$Pente <- coefficients(fit)[[1]]
1016
1017     return(drc_df)
1018   }
1019 })
1020
1021 output$drc_result201 <- renderTable({
1022   ECx201()
1023 }, rownames = TRUE)
1024
1025 output$drc_plot201 <- renderPlot({
1026   fit <- fitmodel201()
1027   data <- filedata201()
1028   ec_values <- ECx201()
1029
1030   TestResult201 <- eventReactive(input$buttonRunStep201, {
1031     data <- filedata201()
1032     if(intest_type201() == 'Essai201') {
1033       data$concentration <- as.factor(data$concentration)
1034       fit <- lm(response ~ concentration, data = data)
1035       Res <- summary(gllht(fit, linfct = mcp(concentration = "
1036         Dunnett"), alternative = "less"))
1037       return(Res)
1038     }
1039   })
1040
1041   output$test_result201 <- renderPrint({TestResult201()})
1042
1043   print(ec_values)
1044
1045   if(intest_type201() == 'Essai201') {
1046     plot(fit, log = "x", broken = TRUE, xlab = paste0("
1047       concentration", input$conc_unit201, " "), ylab = "Réponse
1048       ")
1049
1050     ec10 <- ec_values[ec_values$EC == 10,]
1051     ec50 <- ec_values[ec_values$EC == 50,]
1052
1053     print(ec10)

```

```

1050     print(ec50)
1051
1052     if (nrow(ec10) > 0) {
1053         segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)#,
1054                 y1 = ec10$'Upper 95% CI', col = "blue")
1055         segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)#,
1056                 y1 = ec10$'Lower 95% CI', col = "blue")
1057         abline(v = ec10$Estimate, col = "red", lty = 2)
1058     }
1059     if (nrow(ec50) > 0) {
1060         segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)#,
1061                 y1 = ec50$'Upper 95% CI', col = "blue")
1062         segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)#,
1063                 y1 = ec50$'Lower 95% CI', col = "blue")
1064         abline(v = ec50$Estimate, col = "green", lty = 2)
1065     }
1066
1067     test_results <- TestResult201()
1068     if (!is.null(test_results)) {
1069         noec_values <- test_results$test$p.value > 0.05
1070         loec_values <- test_results$test$p.value <= 0.05
1071         print(noec_values) # Vérifiez les valeurs NOEC
1072         print(loec_values) # Vérifiez les valeurs LOEC
1073         if (any(noec_values, na.rm = TRUE)) {
1074             NOEC <- min(test_results$test$p.value[noec_values], na.rm
1075                         = TRUE)
1076             abline(v = NOEC, col = "purple", lty = 2)
1077         }
1078         if (any(loec_values, na.rm = TRUE)) {
1079             LOEC <- max(test_results$test$p.value[loec_values], na.rm
1080                         = TRUE)
1081             abline(v = LOEC, col = "orange", lty = 2)
1082         }
1083     }
1084 }
1085 })
1086
1087 #202
1088
1089 intest_type202 <- reactive({ 'Essai202' })
1090
1091 filedata202 <- eventReactive(input$buttonRunStep202, {
1092     if(intest_type202() == 'Essai202') {
1093         req(input$datafile_ESSAI202)
1094         validateFile(input$datafile_ESSAI202)
1095         ff <- input$datafile_ESSAI202
1096         read.csv(file = ff$datapath, header = TRUE)
1097     }
1098 })

```

```

1096 output$rawdata202 <- DT::renderDataTable({
1097   filedata202()
1098 })
1099
1100 fitmodel202 <- eventReactive(input$buttonRunStep202, {
1101   data <- filedata202()
1102   if(intest_type202() == 'Essai202') {
1103     fit <- switch(input$model_ESSAI202,
1104       '112' = {
1105         fit1 <- drm(immobilized / total ~
1106           concentration, data = filedata202() %>%
1107             dplyr::filter(time == "24"), fct = LL.2(),
1108             type = "binomial")
1109         fit2 <- drm(immobilized / total ~
1110           concentration, data = filedata202() %>%
1111             dplyr::filter(time == "48"), fct = LL.2(),
1112             type = "binomial")
1113         list(fit1 = fit1, fit2 = fit2)
1114       },
1115       '113' = {
1116         fit1 <- drm(immobilized / total ~
1117           concentration, data = filedata202() %>%
1118             dplyr::filter(time == "24"), fct = LL.3(),
1119             type = "binomial")
1120         fit2 <- drm(immobilized / total ~
1121           concentration, data = filedata202() %>%
1122             dplyr::filter(time == "48"), fct = LL.3(),
1123             type = "binomial")
1124         list(fit1 = fit1, fit2 = fit2)
1125       },
1126       '114' = {
1127         fit1 <- drm(immobilized / total ~
1128           concentration, data = filedata202() %>%
1129             dplyr::filter(time == "24"), fct = LL.4(),
1130             type = "binomial")
1131         fit2 <- drm(immobilized / total ~
1132           concentration, data = filedata202() %>%
1133             dplyr::filter(time == "48"), fct = LL.4(),
1134             type = "binomial")
1135         list(fit1 = fit1, fit2 = fit2)
1136       }
1137     )
1138     return(fit)
1139   }
1140 })
1141
1142 ECx202 <- eventReactive(input$buttonRunStep202, {
1143   fit <- fitmodel202()
1144   if (intest_type202() == 'Essai202') {
1145     XX <- as.numeric(input$ecx_ESSAI202)
1146
1147     fit1 <- fit$fit1

```

```

1130     fit2 <- fit$fit2
1131
1132     drc_df1 <- data.frame(ED(fit1, c(XX), interval = "delta",
1133                             display = FALSE), "Slope" = coefficients(fit1)[[1]])
1134     drc_df2 <- data.frame(ED(fit2, c(XX), interval = "delta",
1135                             display = FALSE), "Slope" = coefficients(fit2)[[1]])
1136
1137     colnames(drc_df1) <- c(paste0('EC', XX), 'Standard_Error', '
1138                             Lower_95%CI', 'Upper_95%CI', 'Slope')
1139     colnames(drc_df2) <- c(paste0('EC', XX), 'Standard_Error', '
1140                             Lower_95%CI', 'Upper_95%CI', 'Slope')
1141
1142     drc_df <- rbind(drc_df1, drc_df2)
1143     rownames(drc_df) <- c('24_h', '48_h')
1144
1145     return(drc_df)
1146   }
1147 })
1148
1149 output$drc_result202 <- renderTable({
1150   ECx202()
1151 }, rownames = TRUE)
1152
1153 output$drc_plot202 <- renderPlot({
1154   fit <- fitmodel202()
1155   data <- filedata202()
1156   ec_values <- ECx202()
1157
1158   #Pas de test
1159   print(ec_values)
1160
1161   if(intest_type202() == 'Essai202') {
1162     fit <- fitmodel202()
1163     fit1 <- fit$fit1
1164     fit2 <- fit$fit2
1165
1166     par(mar=c(5,5,2,2))
1167     plot(fit1, log="x", broken=TRUE, xlab=paste0("concentration_(",
1168         input$conc_unit202, ")"), ylab="Mortality",
1169         ylim=c(0,1), lty="dotted", cex=2, cex.axis=2, cex.lab=2)
1170     par(new=TRUE)
1171     plot(fit2, log="x", broken=TRUE, xlab="", ylab="", main="",
1172         ylim=c(0,1), col="#D55E00", cex=2, cex.axis=2, cex.lab=2)
1173     legend("topleft", inset=0.05, legend=c("24_h", "48_h"), col=c("
1174         black", "#D55E00"), lty=c("dotted", "solid"), cex=2)
1175   }
1176 })
1177
1178 #203
1179
1180 intest_type203 <- reactive({ 'Essai203' })

```



```

1176 validateFile <- function(filename) {
1177   extFile <- tools::file_ext(filename)
1178   validate(need(extFile %in% c("csv", "txt"), "Seuls les fichiers
1179     CSV ou TXT sont acceptés."))
1180 }
1181
1182 filedata203 <- eventReactive(input$buttonRunStep203, {
1183   if (intest_type203() == 'Essai203') {
1184     req(input$datafile_ESSAI203)
1185     validateFile(input$datafile_ESSAI203$name)
1186     ff <- input$datafile_ESSAI203
1187     read.csv(file = ff$datapath, header = TRUE) %>%
1188       mutate(time = as.character(time)) # Ensure 'time' is
         treated as a character
1189   }
1190 })
1191
1192 output$rawdata203 <- DT::renderDataTable({
1193   filedata203()
1194 })
1195
1196 fitmodel203 <- eventReactive(input$buttonRunStep203, {
1197   data <- filedata203()
1198   if (intest_type203() == 'Essai203') {
1199     fit_function <- switch(input$model_ESSAI203,
1200       l12 = LL.2(),
1201       l13 = LL.3u(),
1202       l14 = LL.4())
1203     fit_list <- lapply(c("24", "48", "72", "96"), function(t) {
1204       drm(dead/total ~ as.numeric(concentration), data = data %>%
1205         filter(time == t), fct = fit_function, type = "binomial")
1206     })
1207     names(fit_list) <- c("fit1", "fit2", "fit3", "fit4")
1208     return(fit_list)
1209   }
1210 })
1211
1212 ECx203 <- eventReactive(input$buttonRunStep203, {
1213   fit_list <- fitmodel203()
1214   XX <- as.numeric(input$ecx_ESSAI203)
1215
1216   ec_df_list <- lapply(fit_list, function(fit) {
1217     data.frame(ED(fit, c(XX), interval = "delta", display = FALSE)
1218       , "Slope" = coefficients(fit)[[1]])
1219   })
1220
1221   colnames_list <- c(paste0('EC', XX), 'Standard Error', 'Lower
1222     95%CI', 'Upper 95%CI', 'Slope')
1223   ec_df_list <- lapply(ec_df_list, setNames, colnames_list)
1224
1225   drc_df <- do.call(rbind, ec_df_list)

```

```

1223     rownames(drc_df) <- c('24_h', '48_h', '72_h', '96_h')
1224
1225     return(drc_df)
1226 })
1227
1228 output$drc_result203 <- renderTable({
1229     ECx203()
1230 }, rownames = TRUE)
1231
1232 TestResult203 <- eventReactive(input$buttonRunStep203, {
1233     data <- filedata203() %>%
1234         group_by(concentration, time) %>%
1235         summarize(total = sum(total), dead = sum(dead), .groups = '
1236             drop')
1237
1238     total_24 <- data %>% filter(time == "24" & concentration == "0")
1239         %>% pull(total)
1240     total_48 <- data %>% filter(time == "48" & concentration == "0")
1241         %>% pull(total)
1242     total_72 <- data %>% filter(time == "72" & concentration == "0")
1243         %>% pull(total)
1244     total_96 <- data %>% filter(time == "96" & concentration == "0")
1245         %>% pull(total)
1246     dead_24 <- data %>% filter(time == "24" & concentration == "0")
1247         %>% pull(dead)
1248     dead_48 <- data %>% filter(time == "48" & concentration == "0")
1249         %>% pull(dead)
1250     dead_72 <- data %>% filter(time == "72" & concentration == "0")
1251         %>% pull(dead)
1252     dead_96 <- data %>% filter(time == "96" & concentration == "0")
1253         %>% pull(dead)
1254
1255     data_24 <- data %>% mutate(total_ctrl = total_24, dead_ctrl =
1256         dead_24) %>%
1257         filter(concentration != "0" & time == "24")
1258     data_48 <- data %>% mutate(total_ctrl = total_48, dead_ctrl =
1259         dead_48) %>%
1260         filter(concentration != "0" & time == "48")
1261     data_72 <- data %>% mutate(total_ctrl = total_72, dead_ctrl =
1262         dead_72) %>%
1263         filter(concentration != "0" & time == "72")
1264     data_96 <- data %>% mutate(total_ctrl = total_96, dead_ctrl =
1265         dead_96) %>%
1266         filter(concentration != "0" & time == "96")
1267
1268     fisher <- function(a, b, c, d) {
1269         dt <- matrix(c(a, b, c, d), ncol = 2)
1270         c(pvalue = fisher.test(dt)$p.value)
1271     }
1272
1273     Res1 <- data_24 %>%
1274         rowwise() %>%

```

```

1262     mutate(pvalue = fisher(dead, total - dead, dead_ctrl, total_
1263       ctrl - dead_ctrl)) %>% ungroup() %>%
1264     mutate(p_adjusted = p.adjust(pvalue, "holm")) %>%
1265     mutate(Asterisk = ifelse(p_adjusted < 0.05, ifelse(p_adjusted
1266       > 0.01, "*", "***"), ""))
1267 Res2 <- data_48 %>%
1268   rowwise() %>%
1269   mutate(pvalue = fisher(dead, total - dead, dead_ctrl, total_
1270     ctrl - dead_ctrl)) %>% ungroup() %>%
1271   mutate(p_adjusted = p.adjust(pvalue, "holm")) %>%
1272   mutate(Asterisk = ifelse(p_adjusted < 0.05, ifelse(p_adjusted
1273     > 0.01, "*", "***"), ""))
1274 Res3 <- data_72 %>%
1275   rowwise() %>%
1276   mutate(pvalue = fisher(dead, total - dead, dead_ctrl, total_
1277     ctrl - dead_ctrl)) %>% ungroup() %>%
1278   mutate(p_adjusted = p.adjust(pvalue, "holm")) %>%
1279   mutate(Asterisk = ifelse(p_adjusted < 0.05, ifelse(p_adjusted
1280     > 0.01, "*", "***"), ""))
1281 Res4 <- data_96 %>%
1282   rowwise() %>%
1283   mutate(pvalue = fisher(dead, total - dead, dead_ctrl, total_
1284     ctrl - dead_ctrl)) %>% ungroup() %>%
1285   mutate(p_adjusted = p.adjust(pvalue, "holm")) %>%
1286   mutate(Asterisk = ifelse(p_adjusted < 0.05, ifelse(p_adjusted
1287     > 0.01, "*", "***"), ""))
1288
1289 list("24_h" = knitr::kable(Res1), "48_h" = knitr::kable(Res2), "
1290   72_h" = knitr::kable(Res3), "96_h" = knitr::kable(Res4))
1291 })
1292
1293 output$test_result203 <- renderPrint({
1294   TestResult203()
1295 })
1296
1297 output$drc_plot203 <- renderPlot({
1298   fit_list <- fitmodel203()
1299
1300   par(mar = c(5, 5, 2, 2))
1301   colors <- c("black", "#0072B2", "tomato", "black")
1302   linetypes <- c("dotted", "solid", "dotted", "solid")
1303
1304   plot(fit_list[[1]], log = "x", broken = TRUE, xlab = paste0("
1305     concentration_", input$conc_unit203, ""), ylab = "Mortality
1306     ",
1307     ylim = c(0, 1), lty = linetypes[1], cex = 2, cex.axis = 2,
1308     cex.lab = 2, col = colors[1])
1309
1310   for (i in 2:4) {
1311     par(new = TRUE)
1312     plot(fit_list[[i]], log = "x", broken = TRUE, xlab = "", ylab
1313       = "", main = "",

```

```

1301     ylim = c(0, 1), lty = linetypes[i], col = colors[i], cex
1302         = 2, cex.axis = 2, cex.lab = 2, ann = FALSE)
1303   }
1304   legend("topleft", inset = 0.05, legend = c("24_h", "48_h", "72_h",
1305         "96_h"), col = colors, lty = linetypes, cex = 2)
1306 })
1307 #205
1308
1309 intest_type205 <- reactive({ 'Essai205' })
1310
1311 validateFile <- function(filename) {
1312   extFile <- tools::file_ext(filename)
1313   validate(need(extFile == c("csv","txt"), "Seuls les fichiers CSV
1314         sont acceptés."))
1315 }
1316
1317 filedata205 <- eventReactive(input$buttonRunStep205, {
1318   if(intest_type205() == 'Essai205') {
1319     req(input$datafile_ESSAI205)
1320     validateFile(input$datafile_ESSAI205)
1321     ff <- input$datafile_ESSAI205
1322     read.csv(file = ff$datapath, header = TRUE)
1323   }
1324 })
1325
1326 output$rawdata205 <- DT::renderDataTable({
1327   filedata205()
1328 })
1329
1330 fitmodel205 <- eventReactive(input$buttonRunStep205, {
1331   data <- filedata205()
1332   if(intest_type205() == 'Essai205') {
1333     if(input$model_ESSAI205 == 'l12') {
1334       fit <- drm(response ~ as.numeric(dose), data = data, fct =
1335             LL.2())
1336     } else if(input$model_ESSAI205 == 'l13') {
1337       fit <- drm(response ~ as.numeric(dose), data = data, fct =
1338             LL.3())
1339     } else if(input$model_ESSAI205 == 'l14') {
1340       fit <- drm(response ~ as.numeric(dose), data = data, fct =
1341             LL.4())
1342     }
1343     return(fit)
1344   }
1345 })
1346
1347 ECx205 <- eventReactive(input$buttonRunStep205, {
1348   fit <- fitmodel205()
1349   if(intest_type205() == 'Essai205') {
1350     XX <- as.numeric(input$ecx_ESSAI205)

```

```

1347     ec_values <- ED(fit, c(XX, 50), interval = "delta", display =
1348         FALSE)
1349
1350     drc_df <- data.frame(
1351         EC = c(XX, 50),
1352         Estimate = ec_values[,1],
1353         'Standard Error' = ec_values[,2],
1354         'Lower 95% CI' = ec_values[,3],
1355         'Upper 95% CI' = ec_values[,4]
1356     )
1357
1358     drc_df$Pente <- coefficients(fit)[[1]]
1359
1360     return(drc_df)
1361 }
1362 })
1363
1364 output$drc_result205 <- renderTable({
1365     ECx205()
1366 }, rownames = TRUE)
1367
1368 output$drc_plot205 <- renderPlot({
1369     fit <- fitmodel205()
1370     data <- filedata205()
1371     ec_values <- ECx205()
1372
1373     TestResult205 <- eventReactive(input$buttonRunStep205, {
1374         data <- filedata205()
1375         if(intest_type205() == 'Essai205') {
1376             data$dose <- as.factor(data$dose)
1377             fit <- lm(response ~ dose, data = data)
1378             Res <- summary(glht(fit, linfct = mcp(dose = "Dunnett"),
1379                 alternative = "less"))
1380             return(Res)
1381         }
1382     })
1383
1384     output$test_result205 <- renderPrint({TestResult205()})
1385
1386     if(intest_type205() == 'Essai205') {
1387         plot(fit, log = "x", broken = TRUE, xlab = paste0("Dose_",
1388             input$conc_unit205, " "), ylab = "Réponse")
1389
1390         ec10 <- ec_values[ec_values$EC == 10,]
1391         ec50 <- ec_values[ec_values$EC == 50,]
1392
1393         if (nrow(ec10) > 0) {
1394             segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
1395             segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
1396             abline(v = ec10$Estimate, col = "red", lty = 2)
1397         }
1398         if (nrow(ec50) > 0) {

```

```

1396     segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
1397     segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
1398     abline(v = ec50$Estimate, col = "green", lty = 2)
1399   }
1400
1401   test_results <- TestResult205()
1402   if (!is.null(test_results)) {
1403     noec_values <- test_results$test$p.value > 0.05
1404     loec_values <- test_results$test$p.value <= 0.05
1405     if (any(noec_values, na.rm = TRUE)) {
1406       NOEC <- min(test_results$test$p.value[noec_values], na.rm
1407         = TRUE)
1408       abline(v = NOEC, col = "purple", lty = 2)
1409     }
1410     if (any(loec_values, na.rm = TRUE)) {
1411       LOEC <- max(test_results$test$p.value[loec_values], na.rm
1412         = TRUE)
1413       abline(v = LOEC, col = "orange", lty = 2)
1414     }
1415   }
1416 })
1417 #206
1418
1419 intest_type206 <- reactive({ 'Essai206' })
1420
1421 validateFile <- function(filename) {
1422   extFile <- tools::file_ext(filename)
1423   validate(need(extFile == c("csv","txt"), "Seuls les fichiers CSV
1424     sont acceptés."))
1425 }
1426
1427 filedata206 <- eventReactive(input$buttonRunStep206, {
1428   if(intest_type206() == 'Essai206') {
1429     req(input$datafile_ESSAI206)
1430     validateFile(input$datafile_ESSAI206)
1431     ff <- input$datafile_ESSAI206
1432     read.csv(file = ff$datapath, header = TRUE)
1433   }
1434 })
1435
1436 output$rawdata206 <- DT::renderDataTable({
1437   filedata206()
1438 })
1439
1440 fitmodel206 <- eventReactive(input$buttonRunStep206, {
1441   data <- filedata206()
1442   if(intest_type206() == 'Essai206') {
1443     if(input$model_ESSAI206 == 'l12') {
1444       fit <- drm(response ~ as.numeric(dose), data = data, fct =
1445         LL.2())

```

```

1444   } else if(input$model_ESSAI206 == '113') {
1445     fit <- drm(response ~ as.numeric(dose), data = data, fct =
      LL.3())
1446   } else if(input$model_ESSAI206 == '114') {
1447     fit <- drm(response ~ as.numeric(dose), data = data, fct =
      LL.4())
1448   }
1449   return(fit)
1450 }
1451 })
1452
1453 ECx206 <- eventReactive(input$buttonRunStep206, {
1454   fit <- fitmodel206()
1455   if(intest_type206() == 'Essai206') {
1456     XX <- as.numeric(input$ecx_ESSAI206)
1457     ec_values <- ED(fit, c(XX, 50), interval = "delta", display =
      FALSE)
1458
1459     drc_df <- data.frame(
1460       EC = c(XX, 50),
1461       Estimate = ec_values[,1],
1462       'Standard Error' = ec_values[,2],
1463       'Lower 95% CI' = ec_values[,3],
1464       'Upper 95% CI' = ec_values[,4]
1465     )
1466
1467     drc_df$Pente <- coefficients(fit)[[1]]
1468
1469     return(drc_df)
1470   }
1471 })
1472
1473 output$drc_result206 <- renderTable({
1474   ECx206()
1475 }, rownames = TRUE)
1476
1477 output$drc_plot <- renderPlot({
1478   fit <- fitmodel206()
1479   data <- filedata()
1480   ec_values <- ECx206()
1481
1482   TestResult206 <- eventReactive(input$buttonRunStep206, {
1483     data <- filedata()
1484     if(intest_type206() == 'Essai206') {
1485       data$dose <- as.factor(data$dose)
1486       fit <- lm(response ~ dose, data = data)
1487       Res <- summary(glht(fit, linfct = mcp(dose = "Dunnett"),
        alternative = "less"))
1488       return(Res)
1489     }
1490   })
1491

```

```

1492   output$test_result <- renderPrint({TestResult206()})
1493
1494   if(intest_type206() == 'Essai206') {
1495     plot(fit, log = "x", broken = TRUE, xlab = paste0("Dose_",
1496       input$conc_unit206, " "), ylab = "Réponse")
1497
1498     ec10 <- ec_values[ec_values$EC == 10,]
1499     ec50 <- ec_values[ec_values$EC == 50,]
1500
1501     if (nrow(ec10) > 0) {
1502       segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
1503       segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
1504       abline(v = ec10$Estimate, col = "red", lty = 2)
1505     }
1506     if (nrow(ec50) > 0) {
1507       segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
1508       segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
1509       abline(v = ec50$Estimate, col = "green", lty = 2)
1510     }
1511
1512     test_results <- TestResult206()
1513     if (!is.null(test_results)) {
1514       noec_values <- test_results$test$p.value > 0.05
1515       loec_values <- test_results$test$p.value <= 0.05
1516       if (any(noec_values, na.rm = TRUE)) {
1517         NOEC <- min(test_results$test$p.value[noec_values], na.rm
1518           = TRUE)
1519         abline(v = NOEC, col = "purple", lty = 2)
1520       }
1521       if (any(loec_values, na.rm = TRUE)) {
1522         LOEC <- max(test_results$test$p.value[loec_values], na.rm
1523           = TRUE)
1524         abline(v = LOEC, col = "orange", lty = 2)
1525       }
1526     }
1527   })
1528
1529   #208
1530
1531   intest_type208 <- reactive({ 'Essai208' })
1532
1533   validateFile <- function(filename) {
1534     extFile <- tools::file_ext(filename)
1535     validate(need(extFile == c("csv","txt"), "Seuls les fichiers CSV
1536       sont acceptés."))
1537   }
1538
1539   filedata208 <- eventReactive(input$buttonRunStep208, {
1540     if(intest_type208() == 'Essai208') {
1541       req(input$datafile_ESSAI208)
1542       validateFile(input$datafile_ESSAI208)
1543     }
1544   })

```



```

1540     ff <- input$datafile_ESSAI208
1541     read.csv(file = ff$datapath, header = TRUE)
1542   }
1543 })
1544
1545 output$rawdata208 <- DT::renderDataTable({
1546   filedata208()
1547 })
1548
1549 fitmodel208 <- eventReactive(input$buttonRunStep208, {
1550   data <- filedata208()
1551   if(intest_type208() == 'Essai208') {
1552     if(input$model_ESSAI208 == '112') {
1553       fit <- drm(response ~ as.numeric(concentration), data = data
1554         , fct = LL.2())
1555     } else if(input$model_ESSAI208 == '113') {
1556       fit <- drm(response ~ as.numeric(concentration), data = data
1557         , fct = LL.3())
1558     } else if(input$model_ESSAI208 == '114') {
1559       fit <- drm(response ~ as.numeric(concentration), data = data
1560         , fct = LL.4())
1561     }
1562     return(fit)
1563   }
1564 })
1565
1566 ECx208 <- eventReactive(input$buttonRunStep208, {
1567   fit <- fitmodel208()
1568   if(intest_type208() == 'Essai208') {
1569     XX <- as.numeric(input$ecx_ESSAI208)
1570     ec_values <- ED(fit, c(XX, 50), interval = "delta", display =
1571       FALSE)
1572
1573     drc_df <- data.frame(
1574       EC = c(XX, 50),
1575       Estimate = ec_values[,1],
1576       'Standard Error' = ec_values[,2],
1577       'Lower 95% CI' = ec_values[,3],
1578       'Upper 95% CI' = ec_values[,4]
1579     )
1580
1581     drc_df$Pente <- coefficients(fit)[[1]]
1582
1583     return(drc_df)
1584   }
1585 })
1586
1587 output$drc_result208 <- renderTable({
1588   ECx208()
1589 }, rownames = TRUE)
1590
1591 output$drc_plot <- renderPlot({

```

```

1588 fit <- fitmodel208()
1589 data <- filedata208()
1590 ec_values <- ECx208()
1591
1592 TestResult208 <- eventReactive(input$buttonRunStep208, {
1593   data <- filedata208()
1594   if(intest_type208() == 'Essai208') {
1595     data$concentration <- as.factor(data$concentration)
1596     fit <- lm(response ~ concentration, data = data)
1597     Res <- anova(fit)
1598     return(Res)
1599   }
1600 })
1601
1602 output$test_result <- renderPrint({TestResult208()})
1603
1604 if(intest_type208() == 'Essai208') {
1605   plot(fit, log = "x", broken = TRUE, xlab = paste0("
1606     concentration", input$conc_unit208, ""), ylab = "Réponse
1607     ")
1608
1609   ec10 <- ec_values[ec_values$EC == 10,]
1610   ec50 <- ec_values[ec_values$EC == 50,]
1611
1612   if (nrow(ec10) > 0) {
1613     segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
1614     segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
1615     abline(v = ec10$Estimate, col = "red", lty = 2)
1616   }
1617   if (nrow(ec50) > 0) {
1618     segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
1619     segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
1620     abline(v = ec50$Estimate, col = "green", lty = 2)
1621   }
1622
1623   test_results <- TestResult208()
1624   if (!is.null(test_results)) {
1625     noec_values <- test_results$test$p.value > 0.05
1626     loec_values <- test_results$test$p.value <= 0.05
1627     if (any(noec_values, na.rm = TRUE)) {
1628       NOEC <- min(test_results$test$p.value[noec_values], na.rm
1629         = TRUE)
1630       abline(v = NOEC, col = "purple", lty = 2)
1631     }
1632     if (any(loec_values, na.rm = TRUE)) {
1633       LOEC <- max(test_results$test$p.value[loec_values], na.rm
1634         = TRUE)
1635       abline(v = LOEC, col = "orange", lty = 2)
1636     }
1637   }
1638 })

```

```

1636
1637 #210
1638
1639 intest_type210 <- reactive({ 'Essai210' })
1640
1641 validateFile <- function(filename) {
1642   extFile <- tools::file_ext(filename)
1643   validate(need(extFile == c("csv","txt"), "Seuls les fichiers CSV
1644     sont acceptés."))
1645 }
1646
1647 filedata210 <- eventReactive(input$buttonRunStep210, {
1648   if(intest_type210() == 'Essai210') {
1649     req(input$datafile_ESSAI210)
1650     validateFile(input$datafile_ESSAI210)
1651     ff <- input$datafile_ESSAI210
1652     read.csv(file = ff$datapath, header = TRUE)
1653   }
1654 })
1655
1656 output$rawdata210 <- DT::renderDataTable({
1657   filedata210()
1658 })
1659
1660 fitmodel210 <- eventReactive(input$buttonRunStep210, {
1661   data <- filedata210()
1662   if(intest_type210() == 'Essai210') {
1663     if(input$model_ESSAI210 == 'l12') {
1664       fit <- drm(response ~ as.numeric(concentration), data = data
1665         , fct = LL.2())
1666     } else if(input$model_ESSAI210 == 'l13') {
1667       fit <- drm(response ~ as.numeric(concentration), data = data
1668         , fct = LL.3())
1669     } else if(input$model_ESSAI210 == 'l14') {
1670       fit <- drm(response ~ as.numeric(concentration), data = data
1671         , fct = LL.4())
1672     }
1673     return(fit)
1674   }
1675 })
1676
1677 ECx210 <- eventReactive(input$buttonRunStep210, {
1678   fit <- fitmodel210()
1679   if(intest_type210() == 'Essai210') {
1680     XX <- as.numeric(input$ecx_ESSAI210)
1681     ec_values <- ED(fit, c(XX, 10, 50), interval = "delta",
1682       display = FALSE)
1683
1684     drc_df <- data.frame(
1685       EC = c(XX, 10, 50),
1686       Estimate = ec_values[,1],
1687       'Standard Error' = ec_values[,2],

```

```

1683     'Lower 95% CI' = ec_values[,3],
1684     'Upper 95% CI' = ec_values[,4]
1685   )
1686
1687   drc_df$Pente <- coefficients(fit)[[1]]
1688
1689   return(drc_df)
1690 }
1691 })
1692
1693 output$drc_result210 <- renderTable({
1694   ECx210()
1695 }, rownames = TRUE)
1696
1697 output$drc_plot <- renderPlot({
1698   fit <- fitmodel210()
1699   data <- filedata210()
1700   ec_values <- ECx210()
1701
1702   TestResult210 <- eventReactive(input$buttonRunStep210, {
1703     data <- filedata210()
1704     if(intest_type210() == 'Essai210') {
1705       data$concentration <- as.factor(data$concentration)
1706       fit <- lm(response ~ concentration, data = data)
1707       Res <- summary(gllht(fit, linfct = mcp(concentration = "
1708         Dunnett"), alternative = "less"))
1709       return(Res)
1710     }
1711   })
1712
1713   output$test_result <- renderPrint({TestResult210()})
1714
1715   if(intest_type210() == 'Essai210') {
1716     plot(fit, log = "x", broken = TRUE, xlab = paste0("
1717       concentration", input$conc_unit210, ""), ylab = "Réponse
1718       ")
1719
1720     ec10 <- ec_values[ec_values$EC == 10,]
1721     ec50 <- ec_values[ec_values$EC == 50,]
1722
1723     if (nrow(ec10) > 0) {
1724       segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
1725       segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
1726       abline(v = ec10$Estimate, col = "red", lty = 2)
1727     }
1728     if (nrow(ec50) > 0) {
1729       segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
1730       segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
1731       abline(v = ec50$Estimate, col = "green", lty = 2)
1732     }
1733
1734     test_results <- TestResult210()

```

```

1732     if (!is.null(test_results)) {
1733       noec_values <- test_results$test$p.value > 0.05
1734       loec_values <- test_results$test$p.value <= 0.05
1735       if (any(noec_values, na.rm = TRUE)) {
1736         NOEC <- min(test_results$test$p.value[noec_values], na.rm
1737           = TRUE)
1738         abline(v = NOEC, col = "purple", lty = 2)
1739       }
1740       if (any(loec_values, na.rm = TRUE)) {
1741         LOEC <- max(test_results$test$p.value[loec_values], na.rm
1742           = TRUE)
1743         abline(v = LOEC, col = "orange", lty = 2)
1744       }
1745     }
1746   })
1747 #211
1748
1749   intest_type211 <- reactive({ 'Essai211' })
1750
1751   validateFile <- function(filename) {
1752     extFile <- tools::file_ext(filename)
1753     validate(need(extFile == c("csv", "txt"), "Seuls les fichiers CSV
1754       sont acceptés."))
1755   }
1756
1757   filedata211 <- eventReactive(input$buttonRunStep211, {
1758     if(intest_type211() == 'Essai211') {
1759       req(input$datafile_ESSAI211)
1760       validateFile(input$datafile_ESSAI211)
1761       ff <- input$datafile_ESSAI211
1762       read.csv(file = ff$datapath, header = TRUE)
1763     }
1764   })
1765
1766   output$rawdata211 <- DT::renderDataTable({
1767     filedata211()
1768   })
1769
1770   fitmodel211 <- eventReactive(input$buttonRunStep211, {
1771     data <- filedata211()
1772     if(intest_type211() == 'Essai211') {
1773       if(input$model_ESSAI211 == 'lm') {
1774         fit <- lm(response ~ as.numeric(concentration), data = data)
1775       } else if(input$model_ESSAI211 == 'nls') {
1776         fit <- nls(response ~ fct(concentration), data = data)
1777       } else if(input$model_ESSAI211 == 'lme') {
1778         fit <- lme(response ~ as.numeric(concentration), random = ~
1779           1 | group, data = data)
1780       }
1781     }
1782     return(fit)

```

```

1780   }
1781   })
1782
1783   ECx211 <- eventReactive(input$buttonRunStep211, {
1784     fit <- fitmodel211()
1785     if(intest_type211() == 'Essai211') {
1786       XX <- as.numeric(input$ecx_ESSAI211)
1787       ec_values <- ED(fit, c(XX, 10, 50), interval = "delta",
1788         display = FALSE)
1789
1789       drc_df <- data.frame(
1790         EC = c(XX, 10, 50),
1791         Estimate = ec_values[,1],
1792         'Standard Error' = ec_values[,2],
1793         'Lower 95% CI' = ec_values[,3],
1794         'Upper 95% CI' = ec_values[,4]
1795       )
1796
1797       drc_df$Pente <- coefficients(fit)[[1]]
1798
1799       return(drc_df)
1800     }
1801   })
1802
1803
1804   output$drc_result211 <- renderTable({
1805     ECx211()
1806   }, rownames = TRUE)
1807
1808   output$drc_plot <- renderPlot({
1809     fit <- fitmodel211()
1810     data <- filedata211()
1811     ec_values <- ECx211()
1812
1813     TestResult211 <- eventReactive(input$buttonRunStep211, {
1814       data <- filedata211()
1815       if(intest_type211() == 'Essai211') {
1816         data$concentration <- as.factor(data$concentration)
1817         fit <- lm(response ~ concentration, data = data)
1818         Res <- anova(fit)
1819         return(Res)
1820       }
1821     })
1822
1823     output$test_result <- renderPrint({TestResult211()})
1824
1825     if(intest_type211() == 'Essai211') {
1826       plot(fit, log = "x", broken = TRUE, xlab = paste0("
1827         concentration", input$conc_unit211, ""), ylab = "Réponse
1828         ")
1829
1830       ec10 <- ec_values[ec_values$EC == 10,]

```

```

1829     ec50 <- ec_values[ec_values$EC == 50,]
1830
1831     if (nrow(ec10) > 0) {
1832       segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
1833       segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
1834       abline(v = ec10$Estimate, col = "red", lty = 2)
1835     }
1836     if (nrow(ec50) > 0) {
1837       segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
1838       segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
1839       abline(v = ec50$Estimate, col = "green", lty = 2)
1840     }
1841
1842     test_results <- TestResult211()
1843     if (!is.null(test_results)) {
1844       noec_values <- test_results$test$p.value > 0.05
1845       loec_values <- test_results$test$p.value <= 0.05
1846       if (any(noec_values, na.rm = TRUE)) {
1847         NOEC <- min(test_results$test$p.value[noec_values], na.rm
1848           = TRUE)
1849         abline(v = NOEC, col = "purple", lty = 2)
1850       }
1851       if (any(loec_values, na.rm = TRUE)) {
1852         LOEC <- max(test_results$test$p.value[loec_values], na.rm
1853           = TRUE)
1854         abline(v = LOEC, col = "orange", lty = 2)
1855       }
1856     }
1857   })
1858
1859   #213_214
1860
1861   intest_type213214 <- reactive({ 'Essai213214' })
1862
1863   validateFile <- function(filename) {
1864     extFile <- tools::file_ext(filename)
1865     validate(need(extFile == c("csv","txt"), "Seuls les fichiers CSV
1866       sont acceptés."))
1867   }
1868
1869   filedata213214 <- eventReactive(input$buttonRunStep213214, {
1870     if(intest_type213214() == 'Essai213214') {
1871       req(input$datafile_ESSAI213214)
1872       validateFile(input$datafile_ESSAI213214)
1873       ff <- input$datafile_ESSAI213214
1874       read.csv(file = ff$datapath, header = TRUE)
1875     }
1876   })
1877
1878   output$rawdata213214 <- DT::renderDataTable({
1879     filedata213214()

```

```

1878 })
1879
1880 fitmodel213214 <- eventReactive(input$buttonRunStep213214, {
1881   data <- filedata213214()
1882   if(intest_type213214() == 'Essai213214') {
1883     if(input$model_ESSAI213214 == 'probit') {
1884       fit <- drm(response ~ as.numeric(dose), data = data, fct =
1885         PROBIT())
1886     } else if(input$model_ESSAI213214 == 'logit') {
1887       fit <- drm(response ~ as.numeric(dose), data = data, fct =
1888         LOGIT())
1889     }
1890   }
1891   return(fit)
1892 })
1893
1894 ECx213214 <- eventReactive(input$buttonRunStep213214, {
1895   fit <- fitmodel213214()
1896   if(intest_type213214() == 'Essai213214') {
1897     XX <- as.numeric(input$ecx_ESSAI213214)
1898     ec_values <- ED(fit, c(XX, 50), interval = "delta", display =
1899       FALSE)
1900
1901     drc_df <- data.frame(
1902       EC = c(XX, 50),
1903       Estimate = ec_values[,1],
1904       'Standard Error' = ec_values[,2],
1905       'Lower 95% CI' = ec_values[,3],
1906       'Upper 95% CI' = ec_values[,4]
1907     )
1908
1909     drc_df$Pente <- coefficients(fit)[[1]]
1910
1911     return(drc_df)
1912   }
1913 })
1914
1915 output$drc_result213214 <- renderTable({
1916   ECx213214()
1917 }, rownames = TRUE)
1918
1919 output$drc_plot <- renderPlot({
1920   fit <- fitmodel213214()
1921   data <- filedata213214()
1922   ec_values <- ECx213214()
1923
1924   TestResult213214 <- eventReactive(input$buttonRunStep213214, {
1925     data <- filedata213214()
1926     if(intest_type213214() == 'Essai213214') {
1927       data$dose <- as.factor(data$dose)
1928       fit <- lm(response ~ dose, data = data)

```



```

1927     Res <- summary(glht(fit, linfct = mcp(dose = "Dunnett"),
1928       alternative = "less"))
1929     return(Res)
1930   }
1931 })
1932
1933 output$test_result <- renderPrint({TestResult214()})
1934
1935 if(intest_type213214() == 'Essai213214') {
1936   plot(fit, log = "x", broken = TRUE, xlab = paste0("Dose_",
1937     input$conc_unit213214, " "), ylab = "Réponse")
1938
1939   ec10 <- ec_values[ec_values$EC == 10,]
1940   ec50 <- ec_values[ec_values$EC == 50,]
1941
1942   if (nrow(ec10) > 0) {
1943     segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
1944     segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
1945     abline(v = ec10$Estimate, col = "red", lty = 2)
1946   }
1947   if (nrow(ec50) > 0) {
1948     segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
1949     segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
1950     abline(v = ec50$Estimate, col = "green", lty = 2)
1951   }
1952
1953   test_results <- TestResult214()
1954   if (!is.null(test_results)) {
1955     noec_values <- test_results$test$p.value > 0.05
1956     loec_values <- test_results$test$p.value <= 0.05
1957     if (any(noec_values, na.rm = TRUE)) {
1958       NOEC <- min(test_results$test$p.value[noec_values], na.rm
1959         = TRUE)
1960       abline(v = NOEC, col = "purple", lty = 2)
1961     }
1962     if (any(loec_values, na.rm = TRUE)) {
1963       LOEC <- max(test_results$test$p.value[loec_values], na.rm
1964         = TRUE)
1965       abline(v = LOEC, col = "orange", lty = 2)
1966     }
1967   }
1968 })
1969
1970 #216
1971
1972 intest_type216 <- reactive({ 'Essai216' })
1973
1974 validateFile <- function(filename) {
1975   extFile <- tools::file_ext(filename)
1976   validate(need(extFile == c("csv", "txt"), "Seuls les fichiers CSV

```

```

1975     sont_acceptés."))
1976 }
1977
1978 filedata216 <- eventReactive(input$buttonRunStep216, {
1979   if(intest_type216() == 'Essai216') {
1980     req(input$datafile_ESSAI216)
1981     validateFile(input$datafile_ESSAI216)
1982     ff <- input$datafile_ESSAI216
1983     read.csv(file = ff$datapath, header = TRUE)
1984   }
1985 })
1986
1987 output$rawdata216 <- DT::renderDataTable({
1988   filedata216()
1989 })
1990
1991 fitmodel216 <- eventReactive(input$buttonRunStep216, {
1992   data <- filedata216()
1993   if(intest_type216() == 'Essai216') {
1994     if(input$model_ESSAI216 == 'lm') {
1995       fit <- lm(response ~ as.numeric(concentration), data = data)
1996     } else if(input$model_ESSAI216 == 'nls') {
1997       fit <- nls(response ~ fct(concentration), data = data)
1998     } else if(input$model_ESSAI216 == 'lme') {
1999       fit <- lme(response ~ as.numeric(concentration), random = ~
2000         1 | group, data = data)
2001     }
2002     return(fit)
2003   }
2004 })
2005
2006 ECx216 <- eventReactive(input$buttonRunStep216, {
2007   fit <- fitmodel216()
2008   if(intest_type216() == 'Essai216') {
2009     XX <- as.numeric(input$ecx_ESSAI216)
2010     ec_values <- ED(fit, c(XX, 50), interval = "delta", display =
2011       FALSE)
2012
2013     drc_df <- data.frame(
2014       EC = c(XX, 50),
2015       Estimate = ec_values[,1],
2016       'Standard Error' = ec_values[,2],
2017       'Lower 95% CI' = ec_values[,3],
2018       'Upper 95% CI' = ec_values[,4]
2019     )
2020
2021     drc_df$Pente <- coefficients(fit)[[1]]
2022
2023     return(drc_df)
2024   }
2025 })

```

```

2024 output$drc_result216 <- renderTable({
2025   ECx216()
2026 }, rownames = TRUE)
2027
2028 output$drc_plot <- renderPlot({
2029   fit <- fitmodel216()
2030   data <- filedata216()
2031   ec_values <- ECx216()
2032
2033   TestResult216 <- eventReactive(input$buttonRunStep216, {
2034     data <- filedata216()
2035     if(intest_type216() == 'Essai216') {
2036       data$concentration <- as.factor(data$concentration)
2037       fit <- lm(response ~ concentration, data = data)
2038       Res <- anova(fit)
2039       return(Res)
2040     }
2041   })
2042
2043   output$test_result <- renderPrint({TestResult216()})
2044
2045   if(intest_type216() == 'Essai216') {
2046     plot(fit, log = "x", broken = TRUE, xlab = paste0("
      concentration", input$conc_unit216, ""), ylab = "Réponse
      ")
2047
2048     ec10 <- ec_values[ec_values$EC == 10,]
2049     ec50 <- ec_values[ec_values$EC == 50,]
2050
2051     if (nrow(ec10) > 0) {
2052       segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
2053       segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
2054       abline(v = ec10$Estimate, col = "red", lty = 2)
2055     }
2056     if (nrow(ec50) > 0) {
2057       segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
2058       segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
2059       abline(v = ec50$Estimate, col = "green", lty = 2)
2060     }
2061
2062     test_results <- TestResult216()
2063     if (!is.null(test_results)) {
2064       noec_values <- test_results$test$p.value > 0.05
2065       loec_values <- test_results$test$p.value <= 0.05
2066       if (any(noec_values, na.rm = TRUE)) {
2067         NOEC <- min(test_results$test$p.value[noec_values], na.rm
          = TRUE)
2068         abline(v = NOEC, col = "purple", lty = 2)
2069       }
2070       if (any(loec_values, na.rm = TRUE)) {
2071         LOEC <- max(test_results$test$p.value[loec_values], na.rm
          = TRUE)

```

```

2072     abline(v = LOEC, col = "orange", lty = 2)
2073   }
2074 }
2075 }
2076 })
2077
2078
2079 #218
2080
2081 intest_type218 <- reactive({ 'Essai218' })
2082
2083 validateFile <- function(filename) {
2084   extFile <- tools::file_ext(filename)
2085   validate(need(extFile %in% c("csv", "txt"), "Seuls les fichiers
      CSV ou TXT sont acceptés."))
2086 }
2087
2088 filedata218 <- eventReactive(input$buttonRunStep218, {
2089   if (intest_type218() == 'Essai218') {
2090     req(input$datafile_ESSAI218)
2091     validateFile(input$datafile_ESSAI218$name)
2092     ff <- input$datafile_ESSAI218
2093     read.csv(file = ff$datapath, header = TRUE)
2094   }
2095 })
2096
2097 output$rawdata218 <- DT::renderDataTable({
2098   filedata218()
2099 })
2100
2101 fitmodel218 <- eventReactive(input$buttonRunStep218, {
2102   data <- filedata218()
2103   if (intest_type218() == 'Essai218') {
2104     fit_function_mortality <- switch(input$model_E218_mortality,
2105                                     112 = LL.2(),
2106                                     113 = LL.3u(),
2107                                     114 = LL.4())
2108     fit_function_emergence <- switch(input$model_E218_emergence,
2109                                     112 = LL.2(),
2110                                     113 = LL.3(),
2111                                     114 = LL.4())
2112     fit_function_development <- switch(input$model_E218_
      development,
2113                                     113 = LL.3(),
2114                                     114 = LL.4())
2115
2116     fit1 <- drm(dead/total ~ concentration, data = data, fct = fit
      _function_mortality, type = "binomial")
2117     fit2 <- drm(emerged/total ~ concentration, data = data, fct =
      fit_function_emergence, type = "binomial")
2118     fit3 <- drm(development ~ concentration, data = data, fct =
      fit_function_development, type = "continuous")

```

```

2119     fit <- list(fit1 = fit1, fit2 = fit2, fit3 = fit3)
2120     return(fit)
2121   }
2122 }
2123 })
2124
2125 ECx218 <- eventReactive(input$buttonRunStep218, {
2126   fit <- fitmodel218()
2127   fit1 <- fit$fit1
2128   fit2 <- fit$fit2
2129   fit3 <- fit$fit3
2130   if (intest_type218() == 'Essai218') {
2131     XX <- as.numeric(input$ecx_ESSAI218)
2132
2133     drc_df1 <- data.frame(ED(fit1, c(XX), interval = "delta",
2134                             display = FALSE), "Slope" = coefficients(fit1)[[1]])
2135     drc_df2 <- data.frame(ED(fit2, c(XX), interval = "delta",
2136                             display = FALSE), "Slope" = coefficients(fit2)[[1]])
2137     drc_df3 <- data.frame(ED(fit3, c(XX), interval = "delta",
2138                             display = FALSE), "Slope" = coefficients(fit3)[[1]])
2139
2140     colnames_list <- c(paste0('EC', XX), 'Standard_Error', 'Lower_
2141                        95%CI', 'Upper_95%CI', 'Slope')
2142
2143     colnames(drc_df1) <- colnames_list
2144     colnames(drc_df2) <- colnames_list
2145     colnames(drc_df3) <- colnames_list
2146
2147     drc_df <- rbind(drc_df1, drc_df2, drc_df3)
2148     rownames(drc_df) <- c('Mortality', 'Emergence_ratio', "
2149                          Development_rate")
2150
2151     return(drc_df)
2152   }
2153 })
2154
2155 output$drc_result218 <- renderTable({
2156   ECx218()
2157 }, rownames = TRUE)
2158
2159 output$drc_plot218 <- renderPlot({
2160   fit <- fitmodel218()
2161   fit1 <- fit$fit1
2162   fit2 <- fit$fit2
2163   fit3 <- fit$fit3
2164   data <- filedata218()
2165   ec_values <- ECx218()
2166
2167   TestResult218 <- eventReactive(input$buttonRunStep218, {
2168     data <- filedata218()
2169     if (intest_type218() == 'Essai218') {
2170       data$concentration <- as.factor(data$concentration)

```

```

2166 Res_variance <- bartlett.test(development ~ concentration,
2167                               data = data)
2168 if (input$test_method_E218_mortality == "CA") {
2169   data_CA <- data %>% group_by(concentration) %>% summarize(
2170     dead = sum(dead), total = sum(total))
2171   Res1 <- data.frame(concentration = data_CA$concentration,
2172                     total = data_CA$total, dead = data_CA$dead)
2173   LENGTH <- nrow(data_CA)
2174   for (i in 3:LENGTH) {
2175     Res1[i, 4] <- prop.trend.test(data_CA[1:i, ]$dead, data_
2176                                  CA[1:i, ]$total)$statistic
2177     Res1[i, 5] <- prop.trend.test(data_CA[1:i, ]$dead, data_
2178                                  CA[1:i, ]$total)$p.value
2179   }
2180   colnames(Res1) <- c("concentration", "total", "dead", "Chi
2181                       _squared", "pvalue")
2182   Res1 <- Res1 %>%
2183     mutate(Asterisk = ifelse(pvalue < 0.05, ifelse(pvalue >
2184              0.01, "*", "**"), ""))
2185 } else if (input$test_method_E218_mortality == "Fisher") {
2186   data_F <- filedata218() %>% group_by(concentration) %>%
2187     summarize(total = sum(total), dead = sum(dead)) %>%
2188     ungroup()
2189   total <- data_F %>% dplyr::filter(concentration == "0")
2190   %>% dplyr::select(total) %>% as.numeric()
2191   dead_ctrl <- data_F %>% dplyr::filter(concentration == "0"
2192    ) %>% dplyr::select(dead) %>% as.numeric()
2193   data2 <- data_F %>% mutate(total_ctrl = total, dead_ctrl =
2194     dead_ctrl) %>% dplyr::filter(concentration != "0")
2195   fisher <- function(a, b, c, d) {
2196     dt <- matrix(c(a, b, c, d), ncol = 2)
2197     c(pvalue = fisher.test(dt)$p.value)
2198   }
2199   Res1 <- data2 %>%
2200     rowwise() %>%
2201     mutate(pvalue = fisher(dead, total - dead, dead_ctrl,
2202                            total_ctrl - dead_ctrl)) %>% ungroup() %>%
2203     mutate(p_adjusted = p.adjust(pvalue, "holm")) %>%
2204     mutate(Asterisk = ifelse(p_adjusted < 0.05, ifelse(p_
2205              adjusted > 0.01, "*", "**"), ""))
2206 }
2207 if (input$test_method_E218_emergence == "CA") {
2208   data_CA <- data %>% group_by(concentration) %>% summarize(
2209     emerged = sum(emerged), total = sum(total))
2210   Res2 <- data.frame(concentration = data_CA$concentration,
2211                     total = data_CA$total, emerged = data_CA$emerged)
2212   LENGTH <- nrow(data_CA)
2213   for (i in 3:LENGTH) {
2214     Res2[i, 4] <- prop.trend.test(data_CA[1:i, ]$emerged,
2215                                  data_CA[1:i, ]$total)$statistic
2216     Res2[i, 5] <- prop.trend.test(data_CA[1:i, ]$emerged,
2217                                  data_CA[1:i, ]$total)$p.value

```

```

2201 }
2202 colnames(Res2) <- c("concentration", "total", "emerged", "
2203   Chi_squared", "pvalue")
2204 Res2 <- Res2 %>%
2205   mutate(Asterisk = ifelse(pvalue < 0.05, ifelse(pvalue >
2206     0.01, "*", "**"), ""))
2207 } else if (input$test_method_E218_emergence == "Fisher") {
2208   data_F_emer <- filedata218() %>% group_by(concentration)
2209   %>%
2210     summarize(total = sum(total), emerged = sum(emerged))
2211     %>% ungroup()
2212   total <- data_F_emer %>% dplyr::filter(concentration == "0
2213     ") %>% dplyr::select(total) %>% as.numeric()
2214   emerged_ctrl <- data_F_emer %>% dplyr::filter(
2215     concentration == "0") %>% dplyr::select(emerged) %>% as
2216     .numeric()
2217   data2_emer <- data_F_emer %>% mutate(total_ctrl = total,
2218     EMER_ctrl = emerged_ctrl) %>% dplyr::filter(
2219     concentration != "0")
2220   fisher <- function(a, b, c, d) {
2221     dt <- matrix(c(a, b, c, d), ncol = 2)
2222     c(pvalue = fisher.test(dt)$p.value)
2223   }
2224   Res2 <- data2_emer %>%
2225     rowwise() %>%
2226     mutate(pvalue = fisher(emerged, total - emerged, EMER_
2227       ctrl, total_ctrl - EMER_ctrl)) %>% ungroup() %>%
2228     mutate(p_adjusted = p.adjust(pvalue, "holm")) %>%
2229     mutate(Asterisk = ifelse(p_adjusted < 0.05, ifelse(p_
2230       adjusted > 0.01, "*", "**"), ""))
2231 }
2232 if (input$test_method_E218_development == "Dunnnett") {
2233   data_raw <- filedata218()
2234   data_raw$concentration <- as.factor(data_raw$concentration
2235     )
2236   fit <- aov( development ~ concentration, data = data_raw
2237     )
2238   Res3 <- summary (glht (fit, linfct=mcp (concentration="
2239     Dunnnett"), alternative="less"))
2240   list("Bartlett's test for development rate (DR)" = Res_
2241     variance, "Dunnnett's test for DR" = Res3)
2242 } else if ( input$test_method_E218_development == "Steel"){
2243   data_raw=filedata218()
2244   data_raw$concentration <- as.factor(data_raw$concentration
2245     )
2246   Res3 <- steel.test(development ~ concentration, data =
2247     data_raw, control = "0", alternative="less") %>%
2248     mutate(Asterisk = ifelse(p.value<0.05,ifelse(p.value
2249       >0.01,"*","**"),"" ))
2250 }
2251 list("Mortality" = Res1, "Emergence_ratio" = Res2,
2252   "Bartlett's test for development rate (DR)" = Res_

```

```

2235         variance, "Development_rate" = Res3 )
2236     }
2237 )
2238
2239 output$test_result218 <- renderPrint({TestResult218()})
2240
2241 if(intest_type218() == 'Essai218') {par(mar=c(5,5,2,2), mfrow=c
2242     (3,1),mgp=c(2.5, 0.7, 0))
2243     plot(fit1, log="x", broken=TRUE, xlab=paste0("concentration_(",
2244         input$conc_unit218, ")"), ylab="Mortality",
2245         ylim=c(0,1),lty="dotted",cex=2,cex.axis =2, cex.lab=2)
2246     plot(fit2, log="x", broken=TRUE, xlab=paste0("concentration_(",
2247         input$conc_unit218, ")"), ylab="Emergence_ratio",
2248         ylim=c(0,1),lty="dotted",cex=2,cex.axis =2, cex.lab=2)
2249     plot(fit3, log="x", broken=TRUE, xlab=paste0("concentration_(",
2250         input$conc_unit218, ")"), ylab="Development_rate",
2251         lty="dotted",cex=2,cex.axis =2, cex.lab=2)})
2252
2253 })
2254
2255 #221
2256
2257 intest_type221 <- reactive({ 'Essai221' })
2258
2259 validateFile <- function(filename) {
2260     extFile <- tools::file_ext(filename)
2261     validate(need(extFile == c("csv","txt"), "Seuls les fichiers CSV
2262         sont acceptés."))
2263 }
2264
2265 filedata221 <- eventReactive(input$buttonRunStep221, {
2266     if(intest_type221() == 'Essai221') {
2267         req(input$datafile_ESSAI221)
2268         validateFile(input$datafile_ESSAI221)
2269         ff <- input$datafile_ESSAI221
2270         read.csv(file = ff$datapath, header = TRUE)
2271     }
2272 })
2273
2274 output$rawdata221 <- DT::renderDataTable({
2275     filedata221()
2276 })
2277
2278 fitmodel221 <- eventReactive(input$buttonRunStep221, {
2279     data <- filedata221()
2280     if(intest_type221() == 'Essai221') {
2281         if(input$model_ESSAI221 == 'l12') {
2282             fit <- drm(response ~ as.numeric(concentration), data = data
2283                 , fct = LL.2())
2284         } else if(input$model_ESSAI221 == 'l13') {
2285             fit <- drm(response ~ as.numeric(concentration), data = data

```



```

    , fct = LL.3())
2280   } else if(input$model_ESSAI221 == '114') {
2281     fit <- drm(response ~ as.numeric(concentration), data = data
    , fct = LL.4())
2282   }
2283   return(fit)
2284 }
2285 })
2286
2287 ECx221 <- eventReactive(input$buttonRunStep221, {
2288   fit <- fitmodel221()
2289   if(intest_type221() == 'Essai221') {
2290     XX <- as.numeric(input$ecx_ESSAI221)
2291     ec_values <- ED(fit, c(XX, 10, 50), interval = "delta",
        display = FALSE)
2292
2293     drc_df <- data.frame(
2294       EC = c(XX, 10, 50),
2295       Estimate = ec_values[,1],
2296       'Standard Error' = ec_values[,2],
2297       'Lower 95% CI' = ec_values[,3],
2298       'Upper 95% CI' = ec_values[,4]
2299     )
2300
2301     drc_df$Pente <- coefficients(fit)[[1]]
2302
2303     return(drc_df)
2304   }
2305 })
2306
2307
2308 output$drc_result221 <- renderTable({
2309   ECx221()
2310 }, rownames = TRUE)
2311
2312 output$drc_plot <- renderPlot({
2313   fit <- fitmodel221()
2314   data <- filedata221()
2315   ec_values <- ECx221()
2316
2317   TestResult221 <- eventReactive(input$buttonRunStep221, {
2318     data <- filedata221()
2319     if(intest_type221() == 'Essai221') {
2320       data$concentration <- as.factor(data$concentration)
2321       fit <- lm(response ~ concentration, data = data)
2322       Res <- summary(glht(fit, linfct = mcp(concentration = "
        Dunnnett"), alternative = "less"))
2323       return(Res)
2324     }
2325   })
2326
2327   output$test_result221 <- renderPrint({TestResult221()})

```

```

2328
2329   if(intest_type221() == 'Essai221') {
2330     plot(fit, log = "x", broken = TRUE, xlab = paste0("
      concentration", input$conc_unit221, ""), ylab = "Réponse
      ")
2331
2332     ec10 <- ec_values[ec_values$EC == 10,]
2333     ec50 <- ec_values[ec_values$EC == 50,]
2334
2335     if (nrow(ec10) > 0) {
2336       segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
2337       segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
2338       abline(v = ec10$Estimate, col = "red", lty = 2)
2339     }
2340     if (nrow(ec50) > 0) {
2341       segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
2342       segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
2343       abline(v = ec50$Estimate, col = "green", lty = 2)
2344     }
2345
2346     test_results <- TestResult221()
2347     if (!is.null(test_results)) {
2348       noec_values <- test_results$test$p.value > 0.05
2349       loec_values <- test_results$test$p.value <= 0.05
2350       if (any(noec_values, na.rm = TRUE)) {
2351         NOEC <- min(test_results$test$p.value[noec_values], na.rm
          = TRUE)
2352         abline(v = NOEC, col = "purple", lty = 2)
2353       }
2354       if (any(loec_values, na.rm = TRUE)) {
2355         LOEC <- max(test_results$test$p.value[loec_values], na.rm
          = TRUE)
2356         abline(v = LOEC, col = "orange", lty = 2)
2357       }
2358     }
2359   }
2360 })
2361
2362
2363 #222
2364
2365 intest_type222 <- reactive({ 'Essai222' })
2366
2367 validateFile <- function(filename) {
2368   extFile <- tools::file_ext(filename)
2369   validate(need(extFile == c("csv","txt"), "Seuls les fichiers CSV
      sont acceptés."))
2370 }
2371
2372 filedata222 <- eventReactive(input$buttonRunStep222, {
2373   if(intest_type222() == 'Essai222') {
2374     req(input$datafile_ESSAI222)

```

```

2375     validateFile(input$datafile_ESSAI222)
2376     ff <- input$datafile_ESSAI222
2377     read.csv(file = ff$datapath, header = TRUE)
2378   }
2379 })
2380
2381 output$rawdata222 <- DT::renderDataTable({
2382   filedata222()
2383 })
2384
2385 fitmodel222 <- eventReactive(input$buttonRunStep222, {
2386   data <- filedata222()
2387   if(intest_type222() == 'Essai222') {
2388     if(input$model_ESSAI222 == 'lm') {
2389       fit <- lm(response ~ as.numeric(concentration), data = data)
2390     } else if(input$model_ESSAI222 == 'nls') {
2391       fit <- nls(response ~ fct(concentration), data = data)
2392     } else if(input$model_ESSAI222 == 'lme') {
2393       fit <- lme(response ~ as.numeric(concentration), random = ~
2394         1 | group, data = data)
2395     }
2396     return(fit)
2397   }
2398 })
2399
2400 ECx222 <- eventReactive(input$buttonRunStep222, {
2401   fit <- fitmodel222()
2402   if(intest_type222() == 'Essai222') {
2403     XX <- as.numeric(input$ecx_ESSAI222)
2404     ec_values <- ED(fit, c(XX, 50), interval = "delta", display =
2405       FALSE)
2406
2407     drc_df <- data.frame(
2408       EC = c(XX, 50),
2409       Estimate = ec_values[,1],
2410       'Standard Error' = ec_values[,2],
2411       'Lower 95% CI' = ec_values[,3],
2412       'Upper 95% CI' = ec_values[,4]
2413     )
2414
2415     drc_df$Pente <- coefficients(fit)[[1]]
2416
2417     return(drc_df)
2418   }
2419 })
2420
2421 output$drc_result222 <- renderTable({
2422   ECx222()
2423 }, rownames = TRUE)
2424
2425 output$drc_plot <- renderPlot({

```

```

2425 fit <- fitmodel222()
2426 data <- filedata222()
2427 ec_values <- ECx222()
2428
2429 TestResult222 <- eventReactive(input$buttonRunStep222, {
2430   data <- filedata222()
2431   if(intest_type222() == 'Essai222') {
2432     data$concentration <- as.factor(data$concentration)
2433     fit <- lm(response ~ concentration, data = data)
2434     Res <- anova(fit)
2435     return(Res)
2436   }
2437 })
2438
2439 output$test_result <- renderPrint({TestResult222()})
2440
2441 if(intest_type222() == 'Essai222') {
2442   plot(fit, log = "x", broken = TRUE, xlab = paste0("
2443     concentration", input$conc_unit222, " "), ylab = "Réponse
2444     ")
2445
2446   ec10 <- ec_values[ec_values$EC == 10,]
2447   ec50 <- ec_values[ec_values$EC == 50,]
2448
2449   if (nrow(ec10) > 0) {
2450     segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
2451     segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
2452     abline(v = ec10$Estimate, col = "red", lty = 2)
2453   }
2454   if (nrow(ec50) > 0) {
2455     segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
2456     segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
2457     abline(v = ec50$Estimate, col = "green", lty = 2)
2458   }
2459
2460 test_results <- TestResult222()
2461 if (!is.null(test_results)) {
2462   noec_values <- test_results$test$p.value > 0.05
2463   loec_values <- test_results$test$p.value <= 0.05
2464   if (any(noec_values, na.rm = TRUE)) {
2465     NOEC <- min(test_results$test$p.value[noec_values], na.rm
2466       = TRUE)
2467     abline(v = NOEC, col = "purple", lty = 2)
2468   }
2469   if (any(loec_values, na.rm = TRUE)) {
2470     LOEC <- max(test_results$test$p.value[loec_values], na.rm
2471       = TRUE)
2472     abline(v = LOEC, col = "orange", lty = 2)
2473   }
2474 }
2475 })

```

```

2473
2474
2475 #223
2476 intest_type223 <- reactive({ 'Essai223' })
2477
2478 validateFile <- function(filename) {
2479   extFile <- tools::file_ext(filename)
2480   validate(need(extFile == c("csv","txt"), "Seuls les fichiers CSV
      sont acceptés. "))
2481 }
2482
2483 filedata223 <- eventReactive(input$buttonRunStep223, {
2484   if(intest_type223() == 'Essai223') {
2485     req(input$datafile_ESSAI223)
2486     validateFile(input$datafile_ESSAI223)
2487     ff <- input$datafile_ESSAI223
2488     read.csv(file = ff$datapath, header = TRUE)
2489   }
2490 })
2491
2492 output$rawdata223 <- DT::renderDataTable({
2493   filedata223()
2494 })
2495
2496 fitmodel223 <- eventReactive(input$buttonRunStep223, {
2497   data <- filedata223()
2498   if(intest_type223() == 'Essai223') {
2499     if(input$model_ESSAI223 == 'probit') {
2500       fit <- drm(response ~ as.numeric(dose), data = data, fct =
        PROBIT())
2501     } else if(input$model_ESSAI223 == 'logit') {
2502       fit <- drm(response ~ as.numeric(dose), data = data, fct =
        LOGIT())
2503     }
2504     return(fit)
2505   }
2506 })
2507
2508 ECx223 <- eventReactive(input$buttonRunStep223, {
2509   fit <- fitmodel223()
2510   if(intest_type223() == 'Essai223') {
2511     XX <- as.numeric(input$ecx_ESSAI223)
2512     ec_values <- ED(fit, c(XX, 50), interval = "delta", display =
      FALSE)
2513
2514     drc_df <- data.frame(
2515       EC = c(XX, 50),
2516       Estimate = ec_values[,1],
2517       'Standard Error' = ec_values[,2],
2518       'Lower 95% CI' = ec_values[,3],
2519       'Upper 95% CI' = ec_values[,4]
2520     )

```

```

2521     drc_df$Pente <- coefficients(fit)[[1]]
2522
2523     return(drc_df)
2524   }
2525 })
2526
2527
2528
2529 output$drc_result223 <- renderTable({
2530   ECx223()
2531 }, rownames = TRUE)
2532
2533 output$drc_plot <- renderPlot({
2534   fit <- fitmodel223()
2535   data <- filedata223()
2536   ec_values <- ECx223()
2537
2538   TestResult223 <- eventReactive(input$buttonRunStep223, {
2539     data <- filedata223()
2540     if(intest_type223() == 'Essai223') {
2541       data$dose <- as.factor(data$dose)
2542       fit <- lm(response ~ dose, data = data)
2543       Res <- summary(glht(fit, linfct = mcp(dose = "Dunnett"),
2544         alternative = "less"))
2545       return(Res)
2546     }
2547   })
2548
2549   output$test_result <- renderPrint({TestResult223()})
2550
2551   if(intest_type223() == 'Essai223') {
2552     plot(fit, log = "x", broken = TRUE, xlab = paste0("Dose_",
2553       input$conc_unit223, " "), ylab = "Réponse")
2554
2555     ec10 <- ec_values[ec_values$EC == 10,]
2556     ec50 <- ec_values[ec_values$EC == 50,]
2557
2558     if (nrow(ec10) > 0) {
2559       segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
2560       segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
2561       abline(v = ec10$Estimate, col = "red", lty = 2)
2562     }
2563     if (nrow(ec50) > 0) {
2564       segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
2565       segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
2566       abline(v = ec50$Estimate, col = "green", lty = 2)
2567     }
2568
2569     test_results <- TestResult223()
2570     if (!is.null(test_results)) {
2571       noec_values <- test_results$test$p.value > 0.05
2572       loec_values <- test_results$test$p.value <= 0.05

```

```

2571     if (any(noec_values, na.rm = TRUE)) {
2572       NOEC <- min(test_results$test$p.value[noec_values], na.rm
2573         = TRUE)
2574       abline(v = NOEC, col = "purple", lty = 2)
2575     }
2576     if (any(loec_values, na.rm = TRUE)) {
2577       LOEC <- max(test_results$test$p.value[loec_values], na.rm
2578         = TRUE)
2579       abline(v = LOEC, col = "orange", lty = 2)
2580     }
2581   })
2582
2583
2584
2585 #225
2586
2587 intest_type225 <- reactive({ 'Essai225' })
2588
2589 validateFile <- function(filename) {
2590   extFile <- tools::file_ext(filename)
2591   validate(need(extFile == c("csv","txt"), "Seuls les fichiers CSV
2592     sont acceptés."))
2593 }
2594
2595 filedata225 <- eventReactive(input$buttonRunStep225, {
2596   if(intest_type225() == 'Essai225') {
2597     req(input$datafile_ESSAI225)
2598     validateFile(input$datafile_ESSAI225)
2599     ff <- input$datafile_ESSAI225
2600     read.csv(file = ff$datapath, header = TRUE)
2601   }
2602 })
2603
2604 output$rawdata225 <- DT::renderDataTable({
2605   filedata225()
2606 })
2607
2608 fitmodel225 <- eventReactive(input$buttonRunStep225, {
2609   data <- filedata225()
2610   if(intest_type225() == 'Essai225') {
2611     if(input$model_ESSAI225 == 'logit') {
2612       fit <- drm(response ~ as.numeric(concentration), data = data
2613         , fct = LOGIT())
2614     }
2615     return(fit)
2616   }
2617 })
2618
2619 ECx225 <- eventReactive(input$buttonRunStep225, {
2620   fit <- fitmodel225()

```

```

2619   if(intest_type225() == 'Essai225') {
2620     XX <- as.numeric(input$ecx_ESSAI225)
2621     ec_values <- ED(fit, c(XX, 50), interval = "delta", display =
        FALSE)
2622
2623     drc_df <- data.frame(
2624       EC = c(XX, 50),
2625       Estimate = ec_values[,1],
2626       'Standard Error' = ec_values[,2],
2627       'Lower 95% CI' = ec_values[,3],
2628       'Upper 95% CI' = ec_values[,4]
2629     )
2630
2631     drc_df$Pente <- coefficients(fit)[[1]]
2632
2633     return(drc_df)
2634   }
2635 })
2636
2637
2638 output$drc_result225 <- renderTable({
2639   ECx225()
2640 }, rownames = TRUE)
2641
2642 output$drc_plot <- renderPlot({
2643   fit <- fitmodel225()
2644   data <- filedata225()
2645   ec_values <- ECx225()
2646
2647   TestResult225 <- eventReactive(input$buttonRunStep225, {
2648     data <- filedata225()
2649     if(intest_type225() == 'Essai225') {
2650       data$concentration <- as.factor(data$concentration)
2651       fit <- lm(response ~ concentration, data = data)
2652       Res <- summary(glht(fit, linfct = mcp(concentration = "
        Dunnett"), alternative = "less"))
2653       return(Res)
2654     }
2655   })
2656
2657   output$test_result <- renderPrint({TestResult225()})
2658
2659   if(intest_type225() == 'Essai225') {
2660     plot(fit, log = "x", broken = TRUE, xlab = paste0("
        concentration_", input$conc_unit225, " "), ylab = "Réponse
        ")
2661
2662     ec10 <- ec_values[ec_values$EC == 10,]
2663     ec50 <- ec_values[ec_values$EC == 50,]
2664
2665     if (nrow(ec10) > 0) {
2666       segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)

```



```

2667     segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
2668     abline(v = ec10$Estimate, col = "red", lty = 2)
2669   }
2670   if (nrow(ec50) > 0) {
2671     segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
2672     segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
2673     abline(v = ec50$Estimate, col = "green", lty = 2)
2674   }
2675
2676   test_results <- TestResult225()
2677   if (!is.null(test_results)) {
2678     noec_values <- test_results$test$p.value > 0.05
2679     loec_values <- test_results$test$p.value <= 0.05
2680     if (any(noec_values, na.rm = TRUE)) {
2681       NOEC <- min(test_results$test$p.value[noec_values], na.rm
2682         = TRUE)
2683       abline(v = NOEC, col = "purple", lty = 2)
2684     }
2685     if (any(loec_values, na.rm = TRUE)) {
2686       LOEC <- max(test_results$test$p.value[loec_values], na.rm
2687         = TRUE)
2688       abline(v = LOEC, col = "orange", lty = 2)
2689     }
2690   }
2691 })
2692
2693 #226
2694
2695 intest_type226 <- reactive({ 'Essai226' })
2696
2697 validateFile <- function(filename) {
2698   extFile <- tools::file_ext(filename)
2699   validate(need(extFile == c("csv","txt"), "Seuls les fichiers CSV
2700     sont acceptés."))
2701 }
2702
2703 filedata226 <- eventReactive(input$buttonRunStep226, {
2704   if(intest_type226() == 'Essai226') {
2705     req(input$datafile_ESSAI226)
2706     validateFile(input$datafile_ESSAI226)
2707     ff <- input$datafile_ESSAI226
2708     read.csv(file = ff$datapath, header = TRUE)
2709   }
2710 })
2711
2712 output$rawdata226 <- DT::renderDataTable({
2713   filedata226()
2714 })
2715
2716 fitmodel226 <- eventReactive(input$buttonRunStep226, {

```

```

2716 data <- filedata226()
2717 if(intest_type226() == 'Essai226') {
2718   if(input$model_ESSAI226 == 'lm') {
2719     fit <- lm(response ~ as.numeric(concentration), data = data)
2720   } else if(input$model_ESSAI226 == 'nls') {
2721     fit <- nls(response ~ fct(concentration), data = data)
2722   } else if(input$model_ESSAI226 == 'lme') {
2723     fit <- lme(response ~ as.numeric(concentration), random = ~
2724               1 | group, data = data)
2725   }
2726   return(fit)
2727 }
2728 })
2729 ECx226 <- eventReactive(input$buttonRunStep226, {
2730   fit <- fitmodel226()
2731   if(intest_type226() == 'Essai226') {
2732     XX <- as.numeric(input$ecx_ESSAI226)
2733     ec_values <- ED(fit, c(XX, 50), interval = "delta", display =
2734                     FALSE)
2735     drc_df <- data.frame(
2736       EC = c(XX, 50),
2737       Estimate = ec_values[,1],
2738       'Standard Error' = ec_values[,2],
2739       'Lower 95% CI' = ec_values[,3],
2740       'Upper 95% CI' = ec_values[,4]
2741     )
2742     drc_df$Pente <- coefficients(fit)[[1]]
2743     return(drc_df)
2744   }
2745 })
2746 output$drc_result226 <- renderTable({
2747   ECx226()
2748 }, rownames = TRUE)
2749 output$drc_plot <- renderPlot({
2750   fit <- fitmodel226()
2751   data <- filedata226()
2752   ec_values <- ECx226()
2753   TestResult226 <- eventReactive(input$buttonRunStep226, {
2754     data <- filedata226()
2755     if(intest_type226() == 'Essai226') {
2756       data$concentration <- as.factor(data$concentration)
2757       fit <- lm(response ~ concentration, data = data)
2758       Res <- anova(fit)
2759       return(Res)
2760     }
2761   })

```

```

2766 })
2767
2768 output$test_result <- renderPrint({TestResult226()})
2769
2770 if(intest_type226() == 'Essai226') {
2771   plot(fit, log = "x", broken = TRUE, xlab = paste0("
      concentration", input$conc_unit226, " "), ylab = "Réponse
      ")
2772
2773   ec10 <- ec_values[ec_values$EC == 10,]
2774   ec50 <- ec_values[ec_values$EC == 50,]
2775
2776   if (nrow(ec10) > 0) {
2777     segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
2778     segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
2779     abline(v = ec10$Estimate, col = "red", lty = 2)
2780   }
2781   if (nrow(ec50) > 0) {
2782     segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
2783     segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
2784     abline(v = ec50$Estimate, col = "green", lty = 2)
2785   }
2786
2787   test_results <- TestResult226()
2788   if (!is.null(test_results)) {
2789     noec_values <- test_results$test$p.value > 0.05
2790     loec_values <- test_results$test$p.value <= 0.05
2791     if (any(noec_values, na.rm = TRUE)) {
2792       NOEC <- min(test_results$test$p.value[noec_values], na.rm
         = TRUE)
2793       abline(v = NOEC, col = "purple", lty = 2)
2794     }
2795     if (any(loec_values, na.rm = TRUE)) {
2796       LOEC <- max(test_results$test$p.value[loec_values], na.rm
         = TRUE)
2797       abline(v = LOEC, col = "orange", lty = 2)
2798     }
2799   }
2800 }
2801 })

```

#227

```

2805
2806 intest_type227 <- reactive({ 'Essai227' })
2807
2808 validateFile <- function(filename) {
2809   extFile <- tools::file_ext(filename)
2810   validate(need(extFile == c("csv","txt"), "Seuls les fichiers CSV
      sont acceptés. "))
2811 }
2812

```

```

2813 filedata227 <- eventReactive(input$buttonRunStep227, {
2814   if(intest_type227() == 'Essai227') {
2815     req(input$datafile_ESSAI227)
2816     validateFile(input$datafile_ESSAI227)
2817     ff <- input$datafile_ESSAI227
2818     read.csv(file = ff$datapath, header = TRUE)
2819   }
2820 })
2821
2822 output$rawdata227 <- DT::renderDataTable({
2823   filedata227()
2824 })
2825
2826 fitmodel227 <- eventReactive(input$buttonRunStep227, {
2827   data <- filedata227()
2828   if(intest_type227() == 'Essai227') {
2829     if(input$model_ESSAI227 == 'l12') {
2830       fit <- drm(response ~ as.numeric(concentration), data = data
2831         , fct = LL.2())
2832     } else if(input$model_ESSAI227 == 'l13') {
2833       fit <- drm(response ~ as.numeric(concentration), data = data
2834         , fct = LL.3())
2835     } else if(input$model_ESSAI227 == 'l14') {
2836       fit <- drm(response ~ as.numeric(concentration), data = data
2837         , fct = LL.4())
2838     }
2839     return(fit)
2840   }
2841 })
2842
2843 ECx227 <- eventReactive(input$buttonRunStep227, {
2844   fit <- fitmodel227()
2845   if(intest_type227() == 'Essai227') {
2846     XX <- as.numeric(input$ecx_ESSAI227)
2847     ec_values <- ED(fit, c(XX, 50), interval = "delta", display =
2848       FALSE)
2849
2850     drc_df <- data.frame(
2851       EC = c(XX, 50),
2852       Estimate = ec_values[,1],
2853       'Standard Error' = ec_values[,2],
2854       'Lower 95% CI' = ec_values[,3],
2855       'Upper 95% CI' = ec_values[,4]
2856     )
2857
2858     drc_df$Pente <- coefficients(fit)[[1]]
2859
2860     return(drc_df)
2861   }
2862 })
2863
2864 output$drc_result227 <- renderTable({

```

```

2861 ECx227()
2862 }, rownames = TRUE)
2863
2864 output$drc_plot <- renderPlot({
2865   fit <- fitmodel227()
2866   data <- filedata227()
2867   ec_values <- ECx227()
2868
2869   TestResult227 <- eventReactive(input$buttonRunStep227, {
2870     data <- filedata227()
2871     if(intest_type227() == 'Essai227') {
2872       data$concentration <- as.factor(data$concentration)
2873       fit <- lm(response ~ concentration, data = data)
2874       Res <- anova(fit)
2875       return(Res)
2876     }
2877   })
2878
2879   output$test_result <- renderPrint({TestResult227()})
2880
2881   if(intest_type227() == 'Essai227') {
2882     plot(fit, log = "x", broken = TRUE, xlab = paste0("
2883       concentration", input$conc_unit227, " "), ylab = "Réponse
2884
2885       ")
2886
2887     ec10 <- ec_values[ec_values$EC == 10,]
2888     ec50 <- ec_values[ec_values$EC == 50,]
2889
2890     if (nrow(ec10) > 0) {
2891       segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
2892       segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
2893       abline(v = ec10$Estimate, col = "red", lty = 2)
2894     }
2895     if (nrow(ec50) > 0) {
2896       segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
2897       segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
2898       abline(v = ec50$Estimate, col = "green", lty = 2)
2899     }
2900
2901     test_results <- TestResult227()
2902     if (!is.null(test_results)) {
2903       noec_values <- test_results$test$p.value > 0.05
2904       loec_values <- test_results$test$p.value <= 0.05
2905       if (any(noec_values, na.rm = TRUE)) {
2906         NOEC <- min(test_results$test$p.value[noec_values], na.rm
2907           = TRUE)
2908         abline(v = NOEC, col = "purple", lty = 2)
2909       }
2910       if (any(loec_values, na.rm = TRUE)) {
2911         LOEC <- max(test_results$test$p.value[loec_values], na.rm
2912           = TRUE)
2913         abline(v = LOEC, col = "orange", lty = 2)

```

```

2909     }
2910   }
2911 }
2912 })
2913
2914 #232
2915
2916 intest_type232 <- reactive({ 'Essai232' })
2917
2918 validateFile <- function(filename) {
2919   extFile <- tools::file_ext(filename)
2920   validate(need(extFile == c("csv","txt"), "Seuls les fichiers CSV
2921     sont acceptés."))
2922 }
2923
2924 filedata232 <- eventReactive(input$buttonRunStep232, {
2925   if(intest_type232() == 'Essai232') {
2926     req(input$datafile_ESSAI232)
2927     validateFile(input$datafile_ESSAI232)
2928     ff <- input$datafile_ESSAI232
2929     read.csv(file = ff$datapath, header = TRUE)
2930   }
2931 })
2932
2933 output$rawdata232 <- DT::renderDataTable({
2934   filedata232()
2935 })
2936
2937 fitmodel232 <- eventReactive(input$buttonRunStep232, {
2938   data <- filedata232()
2939   if(intest_type232() == 'Essai232') {
2940     if(input$model_ESSAI232 == 'lm') {
2941       fit <- lm(response ~ as.numeric(concentration), data = data)
2942     } else if(input$model_ESSAI232 == 'nls') {
2943       fit <- nls(response ~ fct(concentration), data = data)
2944     } else if(input$model_ESSAI232 == 'lme') {
2945       fit <- lme(response ~ as.numeric(concentration), random = ~
2946         1 | group, data = data)
2947     }
2948     return(fit)
2949   }
2950 })
2951
2952 ECx232 <- eventReactive(input$buttonRunStep232, {
2953   fit <- fitmodel232()
2954   if(intest_type232() == 'Essai232') {
2955     XX <- as.numeric(input$ecx_ESSAI232)
2956     ec_values <- ED(fit, c(XX, 50), interval = "delta", display =
2957       FALSE)
2958
2959     drc_df <- data.frame(
2960       EC = c(XX, 50),

```

```

2958     Estimate = ec_values[,1],
2959     'Standard Error' = ec_values[,2],
2960     'Lower 95% CI' = ec_values[,3],
2961     'Upper 95% CI' = ec_values[,4]
2962   )
2963
2964   drc_df$Pente <- coefficients(fit)[[1]]
2965
2966   return(drc_df)
2967 }
2968 })
2969
2970 output$drc_result232 <- renderTable({
2971   ECx232()
2972 }, rownames = TRUE)
2973
2974 output$drc_plot <- renderPlot({
2975   fit <- fitmodel232()
2976   data <- filedata232()
2977   ec_values <- ECx232()
2978
2979   TestResult232 <- eventReactive(input$buttonRunStep232, {
2980     data <- filedata232()
2981     if(intest_type232() == 'Essai232') {
2982       data$concentration <- as.factor(data$concentration)
2983       fit <- lm(response ~ concentration, data = data)
2984       Res <- anova(fit)
2985       return(Res)
2986     }
2987   })
2988
2989   output$test_result <- renderPrint({TestResult232()})
2990
2991   if(intest_type232() == 'Essai232') {
2992     plot(fit, log = "x", broken = TRUE, xlab = paste0("
2993       concentration", input$conc_unit232, " "), ylab = "Réponse
2994       ")
2995
2996     ec10 <- ec_values[ec_values$EC == 10,]
2997     ec50 <- ec_values[ec_values$EC == 50,]
2998
2999     if (nrow(ec10) > 0) {
3000       segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
3001       segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
3002       abline(v = ec10$Estimate, col = "red", lty = 2)
3003     }
3004     if (nrow(ec50) > 0) {
3005       segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
3006       segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
3007       abline(v = ec50$Estimate, col = "green", lty = 2)
3008     }
3009   }

```

```

3008     test_results <- TestResult232()
3009     if (!is.null(test_results)) {
3010         noec_values <- test_results$test$p.value > 0.05
3011         loec_values <- test_results$test$p.value <= 0.05
3012         if (any(noec_values, na.rm = TRUE)) {
3013             NOEC <- min(test_results$test$p.value[noec_values], na.rm
3014                         = TRUE)
3015             abline(v = NOEC, col = "purple", lty = 2)
3016         }
3017         if (any(loec_values, na.rm = TRUE)) {
3018             LOEC <- max(test_results$test$p.value[loec_values], na.rm
3019                       = TRUE)
3020             abline(v = LOEC, col = "orange", lty = 2)
3021         }
3022     }
3023 }
3024 })
3025
3026 #239
3027
3028 intest_type239 <- reactive({ 'Essai239' })
3029
3030 validateFile <- function(filename) {
3031     extFile <- tools::file_ext(filename)
3032     validate(need(extFile == c("csv","txt"), "Seuls les fichiers CSV
3033              sont acceptés."))
3034 }
3035
3036 filedata239 <- eventReactive(input$buttonRunStep239, {
3037     if(intest_type239() == 'Essai239') {
3038         req(input$datafile_ESSAI239)
3039         validateFile(input$datafile_ESSAI239)
3040         ff <- input$datafile_ESSAI239
3041         read.csv(file = ff$datapath, header = TRUE)
3042     }
3043 })
3044
3045 output$rawdata239 <- DT::renderDataTable({
3046     filedata239()
3047 })
3048
3049 fitmodel239 <- eventReactive(input$buttonRunStep239, {
3050     data <- filedata239()
3051     if(intest_type239() == 'Essai239') {
3052         if(input$model_ESSAI239 == 'probit') {
3053             fit <- drm(response ~ as.numeric(dose), data = data, fct =
3054                     PROBIT())
3055         } else if(input$model_ESSAI239 == 'logit') {
3056             fit <- drm(response ~ as.numeric(dose), data = data, fct =
3057                     LOGIT())
3058         }
3059     }
3060 })

```



```

3055     }
3056     return(fit)
3057   }
3058 })
3059
3060 ECx239 <- eventReactive(input$buttonRunStep239, {
3061   fit <- fitmodel239()
3062   if(intest_type239() == 'Essai239') {
3063     XX <- as.numeric(input$ecx_ESSAI239)
3064     ec_values <- ED(fit, c(XX, 50), interval = "delta", display =
3065       FALSE)
3066
3067     drc_df <- data.frame(
3068       EC = c(XX, 50),
3069       Estimate = ec_values[,1],
3070       'Standard Error' = ec_values[,2],
3071       'Lower 95% CI' = ec_values[,3],
3072       'Upper 95% CI' = ec_values[,4]
3073     )
3074
3075     drc_df$Pente <- coefficients(fit)[[1]]
3076
3077     return(drc_df)
3078   }
3079 })
3080
3081 output$drc_result239 <- renderTable({
3082   ECx239()
3083 }, rownames = TRUE)
3084
3085 output$drc_plot <- renderPlot({
3086   fit <- fitmodel239()
3087   data <- filedata239()
3088   ec_values <- ECx239()
3089
3090   TestResult239 <- eventReactive(input$buttonRunStep239, {
3091     data <- filedata239()
3092     if(intest_type239() == 'Essai239') {
3093       data$dose <- as.factor(data$dose)
3094       fit <- lm(response ~ dose, data = data)
3095       Res <- summary(glht(fit, linfct = mcp(dose = "Dunnett"),
3096         alternative = "less"))
3097       return(Res)
3098     }
3099   })
3100
3101   output$test_result <- renderPrint({TestResult239()})
3102
3103   if(intest_type239() == 'Essai239') {
3104     plot(fit, log = "x", broken = TRUE, xlab = paste0("Dose_",
3105       input$conc_unit239, " "), ylab = "Réponse")

```

```

3104
3105     ec10 <- ec_values[ec_values$EC == 10,]
3106     ec50 <- ec_values[ec_values$EC == 50,]
3107
3108     if (nrow(ec10) > 0) {
3109         segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
3110         segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
3111         abline(v = ec10$Estimate, col = "red", lty = 2)
3112     }
3113     if (nrow(ec50) > 0) {
3114         segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
3115         segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
3116         abline(v = ec50$Estimate, col = "green", lty = 2)
3117     }
3118
3119     test_results <- TestResult239()
3120     if (!is.null(test_results)) {
3121         noec_values <- test_results$test$p.value > 0.05
3122         loec_values <- test_results$test$p.value <= 0.05
3123         if (any(noec_values, na.rm = TRUE)) {
3124             NOEC <- min(test_results$test$p.value[noec_values], na.rm
3125                 = TRUE)
3126             abline(v = NOEC, col = "purple", lty = 2)
3127         }
3128         if (any(loec_values, na.rm = TRUE)) {
3129             LOEC <- max(test_results$test$p.value[loec_values], na.rm
3130                 = TRUE)
3131             abline(v = LOEC, col = "orange", lty = 2)
3132         }
3133     }
3134 })
3135
3136
3137 #245
3138
3139 intest_type245 <- reactive({ 'Essai245' })
3140
3141 validateFile <- function(filename) {
3142     extFile <- tools::file_ext(filename)
3143     validate(need(extFile == c("csv","txt"), "Seuls les fichiers CSV
3144         sont acceptés."))
3145 }
3146
3147 filedata245 <- eventReactive(input$buttonRunStep245, {
3148     if(intest_type245() == 'Essai245') {
3149         req(input$datafile_ESSAI245)
3150         validateFile(input$datafile_ESSAI245)
3151         ff <- input$datafile_ESSAI245
3152         read.csv(file = ff$datapath, header = TRUE)
3153     }
3154 })

```

```

3153 })
3154
3155 output$rawdata245 <- DT::renderDataTable({
3156   filedata245()
3157 })
3158
3159 fitmodel245 <- eventReactive(input$buttonRunStep245, {
3160   data <- filedata245()
3161   if(intest_type245() == 'Essai245') {
3162     fit <- coxph(Surv(time, response) ~ dose, data = data)
3163     return(fit)
3164   }
3165 })
3166
3167 ECx245 <- eventReactive(input$buttonRunStep245, {
3168   fit <- fitmodel245()
3169   if(intest_type245() == 'Essai245') {
3170     XX <- as.numeric(input$ecx_ESSAI245)
3171     ec_values <- ED(fit, c(XX, 50), interval = "delta", display =
3172       FALSE)
3173
3174     drc_df <- data.frame(
3175       EC = c(XX, 50),
3176       Estimate = ec_values[,1],
3177       'Standard Error' = ec_values[,2],
3178       'Lower 95% CI' = ec_values[,3],
3179       'Upper 95% CI' = ec_values[,4]
3180     )
3181
3182     drc_df$Pente <- coefficients(fit)[[1]]
3183
3184     return(drc_df)
3185   }
3186 })
3187
3188 output$drc_result245 <- renderTable({
3189   ECx245()
3190 }, rownames = TRUE)
3191
3192 output$drc_plot <- renderPlot({
3193   fit <- fitmodel245()
3194   data <- filedata245()
3195   ec_values <- ECx245()
3196
3197   TestResult245 <- eventReactive(input$buttonRunStep245, {
3198     data <- filedata245()
3199     if(intest_type245() == 'Essai245') {
3200       data$dose <- as.factor(data$dose)
3201       fit <- lm(response ~ dose, data = data)
3202       Res <- summary(glht(fit, linfct = mcp(dose = "Dunnett"),
3203         alternative = "less"))

```

```

3203     return(Res)
3204   }
3205 })
3206
3207 output$test_result <- renderPrint({TestResult245()})
3208
3209 if(intest_type245() == 'Essai245') {
3210   plot(fit, log = "x", broken = TRUE, xlab = paste0("Dose_",
3211     input$conc_unit245, " "), ylab = "Réponse")
3212
3213   ec10 <- ec_values[ec_values$EC == 10,]
3214   ec50 <- ec_values[ec_values$EC == 50,]
3215
3216   if (nrow(ec10) > 0) {
3217     segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
3218     segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
3219     abline(v = ec10$Estimate, col = "red", lty = 2)
3220   }
3221   if (nrow(ec50) > 0) {
3222     segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
3223     segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
3224     abline(v = ec50$Estimate, col = "green", lty = 2)
3225   }
3226
3227   test_results <- TestResult245()
3228   if (!is.null(test_results)) {
3229     noec_values <- test_results$test$p.value > 0.05
3230     loec_values <- test_results$test$p.value <= 0.05
3231     if (any(noec_values, na.rm = TRUE)) {
3232       NOEC <- min(test_results$test$p.value[noec_values], na.rm
3233         = TRUE)
3234       abline(v = NOEC, col = "purple", lty = 2)
3235     }
3236     if (any(loec_values, na.rm = TRUE)) {
3237       LOEC <- max(test_results$test$p.value[loec_values], na.rm
3238         = TRUE)
3239       abline(v = LOEC, col = "orange", lty = 2)
3240     }
3241   }
3242 }
3243 })
3244
3245 #246_247
3246
3247 intest_type246247 <- reactive({ 'Essai246247' })
3248
3249 validateFile <- function(filename) {
3250   extFile <- tools::file_ext(filename)
3251   validate(need(extFile == c("csv", "txt"), "Seuls les fichiers CSV
    sont acceptés."))

```

```

3251 }
3252
3253 filedata246247 <- eventReactive(input$buttonRunStep246247, {
3254   if(intest_type246247() == 'Essai246247') {
3255     req(input$datafile_ESSAI246247)
3256     validateFile(input$datafile_ESSAI246247)
3257     ff <- input$datafile_ESSAI246247
3258     read.csv(file = ff$datapath, header = TRUE)
3259   }
3260 })
3261
3262 output$rawdata246247 <- DT::renderDataTable({
3263   filedata246247()
3264 })
3265
3266 fitmodel246247 <- eventReactive(input$buttonRunStep246247, {
3267   data <- filedata246247()
3268   if(intest_type246247() == 'Essai246247') {
3269     if(input$model_ESSAI246247 == 'probit') {
3270       fit <- drm(response ~ as.numeric(dose), data = data, fct =
        PROBIT())
3271     } else if(input$model_ESSAI246247 == 'logit') {
3272       fit <- drm(response ~ as.numeric(dose), data = data, fct =
        LOGIT())
3273     }
3274     return(fit)
3275   }
3276 })
3277
3278 ECx246247 <- eventReactive(input$buttonRunStep246247, {
3279   fit <- fitmodel246247()
3280   if(intest_type246247() == 'Essai246247') {
3281     XX <- as.numeric(input$ecx_ESSAI246247)
3282     ec_values <- ED(fit, c(XX, 50), interval = "delta", display =
        FALSE)
3283
3284     drc_df <- data.frame(
3285       EC = c(XX, 50),
3286       Estimate = ec_values[,1],
3287       'Standard Error' = ec_values[,2],
3288       'Lower 95% CI' = ec_values[,3],
3289       'Upper 95% CI' = ec_values[,4]
3290     )
3291
3292     drc_df$Pente <- coefficients(fit)[[1]]
3293
3294     return(drc_df)
3295   }
3296 })
3297
3298
3299 output$drc_result246247 <- renderTable({

```

```

3300 ECx246247()
3301 }, rownames = TRUE)
3302
3303 output$drc_plot <- renderPlot({
3304   fit <- fitmodel246247()
3305   data <- filedata246247()
3306   ec_values <- ECx246247()
3307
3308   TestResult246247 <- eventReactive(input$buttonRunStep246247, {
3309     data <- filedata246247()
3310     if(intest_type246247() == 'Essai246247') {
3311       data$dose <- as.factor(data$dose)
3312       fit <- lm(response ~ dose, data = data)
3313       Res <- summary(glht(fit, linfct = mcp(dose = "Dunnett"),
3314         alternative = "less"))
3315       return(Res)
3316     }
3317   })
3318
3319   output$test_result <- renderPrint({TestResult246247()})
3320
3321   if(intest_type246247() == 'Essai246247') {
3322     plot(fit, log = "x", broken = TRUE, xlab = paste0("Dose_",
3323       input$conc_unit246247, " "), ylab = "Réponse")
3324
3325     ec10 <- ec_values[ec_values$EC == 10,]
3326     ec50 <- ec_values[ec_values$EC == 50,]
3327
3328     if (nrow(ec10) > 0) {
3329       segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
3330       segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
3331       abline(v = ec10$Estimate, col = "red", lty = 2)
3332     }
3333     if (nrow(ec50) > 0) {
3334       segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
3335       segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
3336       abline(v = ec50$Estimate, col = "green", lty = 2)
3337     }
3338
3339     test_results <- TestResult246247()
3340     if (!is.null(test_results)) {
3341       noec_values <- test_results$test$p.value > 0.05
3342       loec_values <- test_results$test$p.value <= 0.05
3343       if (any(noec_values, na.rm = TRUE)) {
3344         NOEC <- min(test_results$test$p.value[noec_values], na.rm
3345           = TRUE)
3346         abline(v = NOEC, col = "purple", lty = 2)
3347       }
3348       if (any(loec_values, na.rm = TRUE)) {
3349         LOEC <- max(test_results$test$p.value[loec_values], na.rm
3350           = TRUE)
3351         abline(v = LOEC, col = "orange", lty = 2)
3352       }
3353     }
3354   }

```

```

3348     }
3349   }
3350 }
3351 })
3352
3353
3354 #246
3355
3356 intest_type246 <- reactive({ 'Essai246' })
3357
3358 validateFile <- function(filename) {
3359   extFile <- tools::file_ext(filename)
3360   validate(need(extFile == c("csv","txt"), "Seuls les fichiers CSV
3361     sont acceptés."))
3362 }
3363
3364 filedata246 <- eventReactive(input$buttonRunStep246, {
3365   if(intest_type246() == 'Essai246') {
3366     req(input$datafile_ESSAI246)
3367     validateFile(input$datafile_ESSAI246)
3368     ff <- input$datafile_ESSAI246
3369     read.csv(file = ff$datapath, header = TRUE)
3370   }
3371 })
3372
3373 output$rawdata246 <- DT::renderDataTable({
3374   filedata246()
3375 })
3376
3377 fitmodel246 <- eventReactive(input$buttonRunStep246, {
3378   data <- filedata246()
3379   if(intest_type246() == 'Essai246') {
3380     if(input$model_ESSAI246 == 'probit') {
3381       fit <- drm(response ~ as.numeric(concentration), data = data
3382         , fct = PROBIT())
3383     } else if(input$model_ESSAI246 == 'logit') {
3384       fit <- drm(response ~ as.numeric(concentration), data = data
3385         , fct = LOGIT())
3386     }
3387     return(fit)
3388   }
3389 })
3390
3391 ECx246 <- eventReactive(input$buttonRunStep246, {
3392   fit <- fitmodel246()
3393   if(intest_type246() == 'Essai246') {
3394     XX <- as.numeric(input$ecx_ESSAI246)
3395     ec_values <- ED(fit, c(XX, 50), interval = "delta", display =
3396       FALSE)
3397
3398     drc_df <- data.frame(
3399       EC = c(XX, 50),

```

```

3396     Estimate = ec_values[,1],
3397     'Standard Error' = ec_values[,2],
3398     'Lower 95% CI' = ec_values[,3],
3399     'Upper 95% CI' = ec_values[,4]
3400   )
3401
3402   drc_df$Pente <- coefficients(fit)[[1]]
3403
3404   return(drc_df)
3405 }
3406 })
3407
3408 output$drc_result246 <- renderTable({
3409   ECx246()
3410 }, rownames = TRUE)
3411
3412 output$drc_plot <- renderPlot({
3413   fit <- fitmodel246()
3414   data <- filedata246()
3415   ec_values <- ECx246()
3416
3417   TestResult246 <- eventReactive(input$buttonRunStep246, {
3418     data <- filedata246()
3419     if(intest_type246() == 'Essai246') {
3420       data$concentration <- as.factor(data$concentration)
3421       fit <- lm(response ~ concentration, data = data)
3422       Res <- summary(glht(fit, linfct = mcp(concentration = "
3423         Dunnett"), alternative = "less"))
3424       return(Res)
3425     }
3426   })
3427
3428   output$test_result <- renderPrint({TestResult246()})
3429
3430   if(intest_type246() == 'Essai246') {
3431     plot(fit, log = "x", broken = TRUE, xlab = paste0("
3432       concentration_", input$conc_unit246, " "), ylab = "Réponse
3433       ")
3434
3435     ec10 <- ec_values[ec_values$EC == 10,]
3436     ec50 <- ec_values[ec_values$EC == 50,]
3437
3438     if (nrow(ec10) > 0) {
3439       segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
3440       segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
3441       abline(v = ec10$Estimate, col = "red", lty = 2)
3442     }
3443     if (nrow(ec50) > 0) {
3444       segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
3445       segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
3446       abline(v = ec50$Estimate, col = "green", lty = 2)
3447     }
3448   }

```



```

3445 test_results <- TestResult246()
3446 if (!is.null(test_results)) {
3447   noec_values <- test_results$test$p.value > 0.05
3448   loec_values <- test_results$test$p.value <= 0.05
3449   if (any(noec_values, na.rm = TRUE)) {
3450     NOEC <- min(test_results$test$p.value[noec_values], na.rm
3451               = TRUE)
3452     abline(v = NOEC, col = "purple", lty = 2)
3453   }
3454   if (any(loec_values, na.rm = TRUE)) {
3455     LOEC <- max(test_results$test$p.value[loec_values], na.rm
3456               = TRUE)
3457     abline(v = LOEC, col = "orange", lty = 2)
3458   }
3459 }
3460 })
3461
3462
3463
3464 #247
3465
3466 intest_type247 <- reactive({ 'Essai247' })
3467
3468 validateFile <- function(filename) {
3469   extFile <- tools::file_ext(filename)
3470   validate(need(extFile == c("csv", "txt"), "Seuls les fichiers CSV
3471             sont acceptés."))
3472 }
3473
3474 filedata247 <- eventReactive(input$buttonRunStep247, {
3475   if(intest_type247() == 'Essai247') {
3476     req(input$datafile_ESSAI247)
3477     validateFile(input$datafile_ESSAI247)
3478     ff <- input$datafile_ESSAI247
3479     read.csv(file = ff$datapath, header = TRUE)
3480   }
3481 })
3482
3483 output$rawdata247 <- DT::renderDataTable({
3484   filedata247()
3485 })
3486
3487 fitmodel247 <- eventReactive(input$buttonRunStep247, {
3488   data <- filedata247()
3489   if(intest_type247() == 'Essai247') {
3490     if(input$model_ESSAI247 == 'probit') {
3491       fit <- drm(response ~ as.numeric(concentration), data = data
3492                 , fct = PROBIT())
3493     } else if(input$model_ESSAI247 == 'logit') {
3494       fit <- drm(response ~ as.numeric(concentration), data = data

```

```

    , fct = LOGIT())
3493   }
3494   return(fit)
3495 }
3496 })
3497
3498 ECx247 <- eventReactive(input$buttonRunStep247, {
3499   fit <- fitmodel247()
3500   if(intest_type247() == 'Essai247') {
3501     XX <- as.numeric(input$ecx_ESSAI247)
3502     ec_values <- ED(fit, c(XX, 50), interval = "delta", display =
        FALSE)
3503
3504     drc_df <- data.frame(
3505       EC = c(XX, 50),
3506       Estimate = ec_values[,1],
3507       'Standard Error' = ec_values[,2],
3508       'Lower 95% CI' = ec_values[,3],
3509       'Upper 95% CI' = ec_values[,4]
3510     )
3511
3512     drc_df$Pente <- coefficients(fit)[[1]]
3513
3514     return(drc_df)
3515   }
3516 })
3517
3518 output$drc_result247 <- renderTable({
3519   ECx247()
3520 }, rownames = TRUE)
3521
3522 output$drc_plot <- renderPlot({
3523   fit <- fitmodel247()
3524   data <- filedata247()
3525   ec_values <- ECx247()
3526
3527   TestResult247 <- eventReactive(input$buttonRunStep247, {
3528     data <- filedata247()
3529     if(intest_type247() == 'Essai247') {
3530       data$concentration <- as.factor(data$concentration)
3531       fit <- lm(response ~ concentration, data = data)
3532       Res <- summary(glht(fit, linfct = mcp(concentration = "
        Dunnett"), alternative = "less"))
3533       return(Res)
3534     }
3535   })
3536
3537   output$test_result <- renderPrint({TestResult247()})
3538
3539   if(intest_type247() == 'Essai247') {
3540     plot(fit, log = "x", broken = TRUE, xlab = paste0("
        concentration", input$conc_unit247, ""), ylab = "Réponse

```

```

    ")
3541
3542   ec10 <- ec_values[ec_values$EC == 10,]
3543   ec50 <- ec_values[ec_values$EC == 50,]
3544
3545   if (nrow(ec10) > 0) {
3546     segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
3547     segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
3548     abline(v = ec10$Estimate, col = "red", lty = 2)
3549   }
3550   if (nrow(ec50) > 0) {
3551     segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
3552     segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
3553     abline(v = ec50$Estimate, col = "green", lty = 2)
3554   }
3555
3556   test_results <- TestResult247()
3557   if (!is.null(test_results)) {
3558     noec_values <- test_results$test$p.value > 0.05
3559     loec_values <- test_results$test$p.value <= 0.05
3560     if (any(noec_values, na.rm = TRUE)) {
3561       NOEC <- min(test_results$test$p.value[noec_values], na.rm
3562         = TRUE)
3563       abline(v = NOEC, col = "purple", lty = 2)
3564     }
3565     if (any(loec_values, na.rm = TRUE)) {
3566       LOEC <- max(test_results$test$p.value[loec_values], na.rm
3567         = TRUE)
3568       abline(v = LOEC, col = "orange", lty = 2)
3569     }
3570   }
3571 }
3572 })
3573
3574 #ESCORT
3575
3576 intest_typeESCORT <- reactive({ 'ESCORT' })
3577
3578 validateFile <- function(filename) {
3579   extFile <- tools::file_ext(filename)
3580   validate(need(extFile == c("csv","txt"), "Seuls les fichiers CSV
3581     sont acceptés."))
3582 }
3583
3584 filedataESCORT <- eventReactive(input$buttonRunStepESCORT, {
3585   if(intest_typeESCORT() == 'ESCORT') {
3586     req(input$datafile_ESCORT)
3587     validateFile(input$datafile_ESCORT)
3588     ff <- input$datafile_ESCORT
3589     read.csv(file = ff$datapath, header = TRUE)
3590   }
3591 })

```

```

3589
3590 output$rawdataESCORT <- DT::renderDataTable({
3591   filedataESCORT()
3592 })
3593
3594 fitmodelESCORT <- eventReactive(input$buttonRunStepESCORT, {
3595   data <- filedataESCORT()
3596   if(intest_typeESCORT() == 'ESCORT') {
3597     if(input$model_ESCORT == 'l12') {
3598       fit <- drm(response ~ as.numeric(concentration), data = data
3599         , fct = LL.2())
3600     } else if(input$model_ESCORT == 'l13') {
3601       fit <- drm(response ~ as.numeric(concentration), data = data
3602         , fct = LL.3())
3603     } else if(input$model_ESCORT == 'l14') {
3604       fit <- drm(response ~ as.numeric(concentration), data = data
3605         , fct = LL.4())
3606     } else if(input$model_ESCORT == 'lm') {
3607       fit <- lm(response ~ as.numeric(concentration), data = data)
3608     } else if(input$model_ESCORT == 'nls') {
3609       fit <- nls(response ~ fct(concentration), data = data)
3610     } else if(input$model_ESCORT == 'lme') {
3611       fit <- lme(response ~ as.numeric(concentration), random = ~
3612         1 | group, data = data)
3613     }
3614     return(fit)
3615   }
3616 })
3617
3618 ECxESCORT <- eventReactive(input$buttonRunStepESCORT, {
3619   fit <- fitmodelESCORT()
3620   if(intest_typeESCORT() == 'ESCORT') {
3621     XX <- as.numeric(input$ecx_ESCORT)
3622     ec_values <- ED(fit, c(XX, 50), interval = "delta", display =
3623       FALSE)
3624
3625     drc_df <- data.frame(
3626       EC = c(XX, 50),
3627       Estimate = ec_values[,1],
3628       'Standard Error' = ec_values[,2],
3629       'Lower 95% CI' = ec_values[,3],
3630       'Upper 95% CI' = ec_values[,4]
3631     )
3632
3633     drc_df$Pente <- coefficients(fit)[[1]]
3634
3635     return(drc_df)
3636   }
3637 })
3638
3639 output$drc_resultESCORT <- renderTable({
3640   ECxESCORT()

```

```

3636 }, rownames = TRUE)
3637
3638 output$drc_plot <- renderPlot({
3639   fit <- fitmodelESCORT()
3640   data <- filedataESCORT()
3641   ec_values <- ECxESCORT()
3642
3643   TestResultESCORT <- eventReactive(input$buttonRunStepESCORT, {
3644     data <- filedataESCORT()
3645     if(intest_typeESCORT() == 'ESCORT') {
3646       data$concentration <- as.factor(data$concentration)
3647       fit <- lm(response ~ concentration, data = data)
3648       Res <- anova(fit)
3649       return(Res)
3650     }
3651   })
3652
3653   output$test_result <- renderPrint({TestResultESCORT()})
3654
3655   if(intest_typeESCORT() == 'ESCORT') {
3656     plot(fit, log = "x", broken = TRUE, xlab = paste0("
       concentration_", input$conc_unitESCORT, ")), ylab = "Ré
       ponse")
3657
3658     ec10 <- ec_values[ec_values$EC == 10,]
3659     ec50 <- ec_values[ec_values$EC == 50,]
3660
3661     if (nrow(ec10) > 0) {
3662       segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
3663       segments(x0 = ec10$Estimate, y0 = 0, x1 = ec10$Estimate)
3664       abline(v = ec10$Estimate, col = "red", lty = 2)
3665     }
3666     if (nrow(ec50) > 0) {
3667       segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
3668       segments(x0 = ec50$Estimate, y0 = 0, x1 = ec50$Estimate)
3669       abline(v = ec50$Estimate, col = "green", lty = 2)
3670     }
3671
3672     test_results <- TestResultESCORT()
3673     if (!is.null(test_results)) {
3674       noec_values <- test_results$test$p.value > 0.05
3675       loec_values <- test_results$test$p.value <= 0.05
3676       if (any(noec_values, na.rm = TRUE)) {
3677         NOEC <- min(test_results$test$p.value[noec_values], na.rm
          = TRUE)
3678         abline(v = NOEC, col = "purple", lty = 2)
3679       }
3680       if (any(loec_values, na.rm = TRUE)) {
3681         LOEC <- max(test_results$test$p.value[loec_values], na.rm
          = TRUE)
3682         abline(v = LOEC, col = "orange", lty = 2)
3683       }

```

```

3684     }
3685   }
3686 })
3687
3688 ### Rapport ##
3689 output$downloadReport <- downloadHandler(
3690   filename = function() {
3691     paste('my-report', sep = '.', switch(
3692       input$format, PDF = 'pdf', HTML = 'html', Word = 'docx'
3693     ))
3694   },
3695   content = function(file) {
3696     src <- normalizePath('report.Rmd')
3697     owd <- setwd(tempdir())
3698     on.exit(setwd(owd))
3699     file.copy(src, 'report.Rmd', overwrite = TRUE)
3700
3701     library(rmarkdown)
3702     out <- render('report.Rmd', switch(
3703       input$format,
3704       PDF = pdf_document(), HTML = html_document(), Word = word_
3705         document()
3706     ))
3707     file.rename(out, file)
3708   }
3709 )
3710
3711 #Sorties#
3712
3713 output$contents <- renderDT({
3714   global_data()
3715 })
3716
3717 output$contents2 <- renderDT({
3718   global_data()
3719 })
3720
3721 output$contents3 <- renderDT({
3722   global_data()
3723 })
3724
3725 output$contents4 <- renderDT({
3726   global_data()
3727 })
3728 }
3729
3730 shinyApp(ui, server)

```

Analysis page:

```

1 library(dplyr)
2 library(ggplot2)

```

```

3 library(glue)
4 library(leaflet)
5 library(plotly)
6 library(sass)
7 library(shiny)
8 library(shiny.fluent)
9 library(shiny.router)
10
11 filters <- function() {
12   Stack(
13     tokens = list(childrenGap = 10),
14     fileInput("file_analysis", "",
15               multiple = FALSE,
16               accept = c("text/csv", "text/comma-separated-values,
17                           text/plain", ".txt/.csv")),
18     textAreaInput("textdata", "Or Paste Historical Data Here", "",
19                   rows = 10),
20     tags$hr(),
21     checkboxInput("header_analysis", "Header", TRUE),
22     radioButtons("sep_analysis", "Separator",
23                  choices = c(Comma = ",", Semicolon = ";", Tab = "\t"),
24                  selected = ","),
25     actionButton("submit", "Submit")
26   )
27 }
28
29 solvant <- function(id) {
30   div(
31     checkboxInput(id, label = "Activate solvent effect", value =
32                     FALSE),
33     textOutput(paste0(id, "Value"))
34   )
35 }
36
37 outlier <- function(id) {
38   div(
39     checkboxInput(id, label = "Handle outliers", value = FALSE),
40     textOutput(paste0(id, "Value"))
41   )
42 }
43
44 transformation <- function(id) {
45   div(
46     checkboxInput(id, label = "Apply transformation", value = FALSE)
47     ,
48     uiOutput(paste0(id, "_menu")),
49     textOutput(paste0(id, "Value"))
50   )
51 }

```

```

50
51 #       Boxplot #####
52 ff <- sidebarLayout(
53   sidebarPanel(
54     solvant("effect_solvant"),
55     outlier("manage_outliers"),
56     transformation("apply_transformation"),
57     actionButton("go", "Execute")
58   ),
59   mainPanel(
60     verbatimTextOutput("result"),
61     plotOutput("plot")
62   )
63 )
64
65 analysisPage <- function() {
66   tagList(
67     makePage(
68       "Descriptive_Statistics_Analysis",
69       "Upload_and_analyze_your_data",
70       div(
71         HorizontalStack(
72           makeCard("Data_Upload", filters(), size = 4, style = "
73             max-height: 520px"),
74           makeCard("Visualization", plotlyOutput("plot_
75             visualization"), size = 8, style = "max-height: 520px
76             ")
77         ),
78         Stack(
79           tokens = list(childrenGap = 10), horizontal = TRUE,
80           makeCard("Options", ff, size = 4, style = "max-height: 520px"),
81           makeCard("Boxplot1", plotlyOutput("plot_box"), size = 9,
82             style = "max-height: 520px"),
83           makeCard("Boxplot2", plotlyOutput("plot_box2"), size =
84             9, style = "max-height: 520px")
85         ),
86         makeCard("Data_Preview", DTOutput("uploaded_data"), style
87           = "margin-top: 100px;"),
88         makeCard("Data_Preview_after_transformation", tableOutput(
89           "dataTF"), style = "margin-top: 100px;"),
90         makeCard("Outliers", tableOutput("outliers_table"), style
91           = "margin-top: 100px;"),
92         makeCard("Descriptive_Statistics", DTOutput("desc_stats"),
93           style = "margin-top: 100px;")
94       )
95     )
96   )
97 }

```

Page ANOVA:

```

1 anovaPage <- function() {

```



```

2   tagList(
3     makePage(
4       "ANOVA_Test",
5       "ANOVA_test_results",
6       div(
7         HorizontalStack(
8           makeCard("File_input",
9             div(
10              actionButton("run_anova", "Run_ANOVA_Test")
11            ),
12            size = 4, style = "max-height: 320px"
13          ),
14          makeCard("Data_Preview", DTOutput("contents4"), size = 8,
15            style = "max-height: 320px")
16        ),
17        div(
18          makeCard("Results", verbatimTextOutput("anova_result"),
19            style = "margin-top: 20px;")
20        )
21      )
22    )
  }

```

Page Tests Bayésien

```

1   library(shiny)
2   library(plotrix)
3   library(ggplot2)
4   library(coda)
5   library(shinyjs)
6
7   bayesianPage <- function() {
8     navbarPage(
9       "",
10
11       # Onglet d'introduction/théorie
12       tabPanel("",
13         fluidPage(
14           h2(""),
15           p(""),
16           h3("Théorie"),
17           h4("Tests Bayésiens"),
18           p("Les tests bayésiens permettent d'incorporer des informations a priori dans l'analyse des données. Le facteur de Bayes (B) compare la plausibilité des données sous deux hypothèses: l'hypothèse nulle ( $H_0$ ) et l'hypothèse alternative ( $H_1$ )."),
19           p("Le facteur de Bayes est calculé comme suit: "),
20           p("$$$B_{1/0} = \frac{P(\text{Données} | H_1)}{P(\text{Données} | H_0)}$$$"),
21           p("Les probabilités a posteriori des hypothèses")

```

```

        nulles_ et_ alternatives_ peuvent_ etre_ calculées_ en_
        utilisant_ le_ facteur_ de_ Bayes_ et_ une_ probabilité_ a_
        priori_ de_ l' hypothèse_ nulle. ")
22     )),
23
24     # Onglet d'analyse bayésienne standard
25     tabPanel("",
26         pageWithSidebar(
27             headerPanel("Test_ effet_ solvant"),
28             sidebarPanel(
29                 uiOutput("column_select"),
30                 radioButtons(inputId = "design",
31                             label = "Design:",
32                             choiceNames = c("Single/paired_ sample(
33                                     s)", "Independent_ samples"),
34                             choiceValues = c("single", "
35                                     independent"),
36                             selected = "single"),
37                 radioButtons(inputId = "direction",
38                             label = "Predicted_ direction:",
39                             choiceNames = c("None", "Positive_
40                                     effect", "Negative_ effect"),
41                             choiceValues = c("none", "positive", "
42                                     negative"),
43                             selected = "none"),
44                 numericInput(inputId = "t",
45                             label = "t-statistic:",
46                             value = 0,
47                             min = -1e5,
48                             max = 1e5,
49                             step=0.1),
50                 numericInput(inputId = "pH0",
51                             label = "Prior_ probability_ of_ null:",
52                             value=0.5,
53                             min=0,
54                             max=1,
55                             step=0.1),
56                 numericInput('nchain', 'Nombre_ d\' itérations',
57                             value_=10000),
58                 ##### numericInput('burnin', 'Période de burn-in', value_=
59                             =500),
60                 ##### numericInput('sd_beta0', 'Ecart-type pour beta0',
61                             value_=1),
62                 ##### numericInput('sd_beta1', 'Ecart-type pour beta1',
63                             value_=2),
64                 ##### actionButton('run_EstiBayesian', 'Lancer l\' analyse
65                             '),
66                 br(),
67                 br(),
68                 p("")
69             ),
70             mainPanel(

```

```

62         div(helpText('Model definitions:'), align = "left",
63             style="font-size:140%"),
64         div(withMathJax(sprintf('\\(\\mathcal{H}_0\\):_\\_
65             Aucun effet détecté')), style="font-size:130%"),
66         div(uiOutput("altText"), style="font-size:130%"),
67         br(),
68         div(helpText('Predictive adequacy:'), align = "left",
69             style="font-size:140%"),
70         plotOutput("plotOutput"),
71         div(helpText('Bayesian Test Results:'), align = "left",
72             style="font-size:140%"),
73         div(uiOutput("bfText"), align = "left", style="font-size:130%;"),
74         div(uiOutput("description"), align = "left", style="font-size:130%;"),
75         br(),
76         div(helpText('Posterior Probabilities:'), style="font-size:140%"),
77         div(uiOutput("postNull"), align = "left", style="font-size:130%;"),
78         div(uiOutput("postAlt"), align = "left", style="font-size:130%;"),
79         br(),
80         htmlOutput('acc_rate'),
81         plotOutput('chain_plot')
82         #htmlOutput('ec_values'), # pour les EC10, EC20 et EC50
83         #htmlOutput('interpretation')
84     )
85 ))),
86
87 # Nouvel onglet pour tester les hypothèses sur les ECx
88 tabPanel("",
89     pageWithSidebar(
90         headerPanel("Tester les hypothèses sur les valeurs ECx"),
91         sidebarPanel(
92             numericInput("hyp_ec10", "Hypothèse pour EC10:",
93                 value = 50, min = 0),
94             numericInput("hyp_ec20", "Hypothèse pour EC20:",
95                 value = 50, min = 0),
96             numericInput("hyp_ec50", "Hypothèse pour EC50:",
97                 value = 50, min = 0),
98             actionButton("test_hypothesis", "Tester l'hypothèse")
99         ),
100         mainPanel(
101             div(helpText('Résultats des tests Bayésiens:'),
102                 align = "left", style="font-size:140%"),
103             div(uiOutput("bayes_factors"), align = "left",
104                 style="font-size:130%;"),
105             br(),

```

```

97         div(helpText('Interprétation des résultats:'),
98             style="font-size:140%"),
99         div(uiOutput("interpretation"), align = "left",
100            style="font-size:130%;")
101     )
102 })
103 }

```

Page selection de modèle (BAYESIEN)

```

1 BayoyoPage <- function() {
2   tagList(
3     makePage(
4       "Analyse de données bayésiennes",
5       "Téléchargez et analysez vos données avec des méthodes bayé-
6         siennes",
7       div(
8         HorizontalStack(
9           makeCard(
10             "Téléchargement des données",
11             fileInput("file", "Choisissez un fichier CSV ou TXT sans
12               nom de colonnes",
13               accept = c("text/csv", "text/comma-separated-
14                 values", "text/plain", ".csv", ".txt")),
15             size = 4,
16             style = "max-height: 520px"
17           ),
18           makeCard(
19             "Options d'analyse",
20             div(
21               numericInput("iterations", "Nombre d'itérations (
22                 Metropolis-Hastings)", value = 50000),
23               numericInput("burn_in", "Burn-in", value = 1000),
24               actionButton("run_analysis", "Lancer l'analyse")
25             ),
26             size = 4,
27             style = "max-height: 520px"
28           )
29         ),
30         Stack(
31           tokens = list(childrenGap = 10), horizontal = TRUE,
32           makeCard("Log-Logistic Distribution",
33             plotOutput("plot_loglogistic_1"),
34             #plotOutput("plot_loglogistic_2"),
35             size = 9, style = "max-height: 520px"),
36           makeCard("Log-Normal Distribution",
37             plotOutput("plot_lognormal_1"),
38             #plotOutput("plot_lognormal_2"),
39             size = 9, style = "max-height: 520px"),
40           makeCard("Weibull Distribution",
41             plotOutput("plot_weibull_1"),
42             #plotOutput("plot_weibull_2"),

```

```

39         size = 9, style = "max-height: 520px")
40     ),
41     makeCard("Facteurs bayésiens", verbatimTextOutput("bayes_
42         factors"), style = "margin-top: 100px;"),
43     makeCard("Taux d'acceptation", verbatimTextOutput("
44         acceptance_rates"), style = "margin-top: 100px;")
45 )
46 }

```

Page BOOTSTRAP

```

1 bootstrapPage <- function() {
2   tagList(
3     makePage(
4       "Bootstrap Test",
5       "Bootstrap test results",
6       div(
7         HorizontalStack(
8           makeCard("File input",
9             div(
10              actionButton("run_bootstrap", "Run Bootstrap
11                Test")
12            ),
13            size = 4, style = "max-height: 320px"
14          ),
15          makeCard("Data Preview", DTOutput("contents2"), size = 8,
16            style = "max-height: 320px")
17        ),
18        div(
19          makeCard("Results", verbatimTextOutput("bootstrap_result")
20            , style = "margin-top: 20px;"),
21          makeCard("Bootstrap Statistics", plotOutput("bootstrap_
22            plot"), style = "margin-top: 20px;")
23        )
24      )
25    )
26  )
27 }

```

Page DUNNETT:

```

1 dunnettPage <- function() {
2   tagList(
3     makePage(
4       "Dunnett Test",
5       "Dunnett test results",
6       div(
7         HorizontalStack(
8           makeCard("File input",
9             div(
10              actionButton("run_dunnett", "Run Dunnett Test")

```

```

11         ),
12         size = 4, style = "max-height: 320px"
13     ),
14     makeCard("Data_Preview", DTOutput("contents3"), size = 8,
15             style = "max-height: 320px")
16 ),
17     div(
18         makeCard("Results", verbatimTextOutput("dunnett_result"),
19                 style = "margin-top: 20px;"),
20         makeCard("Dunnett_Plot", plotOutput("dunnett_plot"), style
21                 = "margin-top: 20px;")
22     )
23 }

```

```

1 footer <- function() {
2   Stack(
3     horizontal = TRUE,
4     horizontalAlign = "space-between",
5     tokens = list(childrenGap = 20),
6     Text(variant = "medium", "Version personnelle pour Université",
7           block = TRUE),
8     Text(variant = "medium", nowrap = FALSE, "For more information,
9           contact me at herve.boutrin@etu.univ-nantes.fr"),
10    Text(variant = "medium", nowrap = FALSE, "All rights reserved.")
11  )
12 }

```

Menu header:

```

1 library(shiny)
2 library(shiny.fluent)
3 library(shiny.router)
4
5 #Eléments du menu secondaire
6 farItems <- list(
7   list(
8     key = "tile",
9     text = "Grid view",
10    ariaLabel = "Grid view",
11    iconOnly = TRUE,
12    iconProps = list(iconName = "Tiles")
13  )
14 )
15
16
17 # Barre de commandes
18 command_bar <- CommandBar(
19   items = list(
20     list(
21       key = "home",

```

```

22     text = "0)␣Home",
23     iconProps = list(iconName = "Home"),
24     href = "#!/\"
25 ),
26 list(
27     key = "analysis",
28     text = "I)␣Descriptive␣Analysis",
29     iconProps = list(iconName = "ExploreData"),
30     href = "#!/analysis\"
31 ),
32 list(
33     key = "modeling",
34     text = "III)␣Modelization",
35     iconProps = list(iconName = "StackedLineChart"),
36     subMenuProps = list(
37         items = list(
38             list(
39                 text = "Continuous␣Data␣Models",
40                 key = "continuousDataModels",
41                 subMenuProps = list(
42                     items = list(
43                         list(text = "Probit␣Model", key = "probitModel",
44                             href = "#!/probitModel"),#ok
45                         list(text = "Logit␣Model", key = "logitModel", href =
46                             "#!/logitModel"),#ok
47                         list(text = "Weibull␣Model", key = "weibullModel",
48                             href = "#!/weibullModel",
49                             subMenuProps = list(
50                                 items = list(
51                                     list(text = "", key = "", href = "#!/\"),
52                                     list(text = "", key = "", href = "#!/\"))
53                                 ),#ok
54                             list(text = "Gompertz␣Model", key = "gompertzModel",
55                                 href = "#!/gompertzModel"),#ok
56                             list(text = "Hill␣Model", key = "hillModel", href =
57                                 "#!/hillModel"),#ok
58                             list(text = "Logistic␣Model", key = "logisticModel",
59                                 href = "#!/logisticModel"),#ok
60                             list(text = "Log-Logistic␣Model", key = "
61                                 logLogisticModel", href = "#!/logLogisticModel"),
62                                 #ok
63                             list(text = "Exponential␣Model", key = "
64                                 exponentialModel", href = "#!/exponentialModel"),
65                                 #ok
66                             list(text = "Beta-Binomial␣Model", key = "
67                                 betaBinomialModel", href = "#!/betaBinomialModel\"
68                                 ),
69                             list(text = "Bruce-Versteeg␣Model", key = "Bruce-
70                                 VersteegModel", href = "#!/Bruce-VersteegModel\"),
71                                 #ok
72                             list(text = "quadratique␣Model", key = \"

```

```

        quadratiqueModel", href = "#!/quadratiqueModel"),
        #ok
60     list(text = "Seuil_Model", key = "SeuilModel", href
        = "#!/SeuilModel",
61         subMenuProps = list(
62             items = list(
63                 list(text = "", key = "", href = "#!/"),
64                 list(text = "", key = "", href = "#!/"),
65                 list(text = "", key = "", href = "#!/"),
66                 list(text = "", key = "", href = "#!/")
67             )
68         )),#ok
69     list(text = "Hormétiques_Model", key = "Hormé
        tiquesmodel", href = "#!/Hormétiquesmodel",
70         subMenuProps = list(
71             items = list(
72                 list(text = "", key = "", href = "#!/"),
73                 list(text = "", key = "", href = "#!/")
74             )
75         )),#ok
76     list(text = "ANCOVA_Model", key = "ANCOVAmode",
        href = "#!/ANCOVAmode")
77 )
78 )
79 ),
80 list(
81     text = "Quantal_Data_Models",
82     key = "quantalDataModels",
83     subMenuProps = list(
84         items = list(
85             list(text = "Binary_Logistic_Regression", key = "
                binaryLogisticRegression", href = "#!/
                binaryLogisticRegression"),
86             list(text = "Probit_Regression", key = "
                probitRegression", href = "#!/probitRegression"),
87             list(text = "Weibull_Model", key = "WeibullModel",
                href = "#!/WeibullModel"),
88             list(text = "Gumbel_Model", key = "GumbelModel",
                href = "#!/GumbelModel")
89         )
90     )
91 ),
92 list(
93     text = "Count_Data_Models",
94     key = "countDataModels",
95     subMenuProps = list(
96         items = list(
97             list(text = "Poisson_Regression", key = "
                poissonRegression", href = "#!/poissonRegression"
                ),
98             list(text = "Negative_Binomial_Regression", key = "
                negativeBinomialRegression", href = "#!/

```



```

99         negativeBinomialRegression")
100     )
101 )
102 )
103 )
104 ),
105 list(
106     key = "Select", # Clé unique pour cette page
107     text = "II) Bayoyo",
108     iconProps = list(iconName = "DataManagementSettings"), #
109         Icône représentative
110     href = "#!/Select" # URL fragment pour accéder à la page
111 ),
112 list(
113     key = "testing",
114     text = "IV) Inferences",
115     iconProps = list(iconName = "ArrangeSendToBack"),
116     subMenuProps = list(
117         items = list(
118             list(
119                 text = "Parametric tests",
120                 key = "parametricInference",
121                 subMenuProps = list(
122                     items = list(
123                         list(text = "t-test", key = "ttest", href = "#!/ttest"),
124                         list(text = "ANOVA ---> post hoc", key = "anova", href = "#!/anova",
125                             subMenuProps = list(
126                                 items = list(
127                                     list(text = "Tukey's HSD", key = "tukeyHSD", href = "#!/tukeyHSD"),
128                                     list(text = "Bonferroni Correction", key = "bonferroni", href = "#!/bonferroni"),
129                                     list(text = "Scheffé's Test", key = "scheffe", href = "#!/scheffe"),
130                                     list(text = "Dunnett's Test", key = "dunnett", href = "#!/dunnett"),
131                                     list(text = "Holm's Sequential Bonferroni Procedure", key = "holmBonferroni", href = "#!/holmBonferroni"),
132                                     list(text = "Fisher's Least Significant Difference (LSD)", key = "lsd", href = "#!/lsd")
133                                 )
134                             )
135                         )
136                     )
137                 )
138             )
139         )
140     ),
141     list(text = "F-test", key = "ftest", href = "#!/ftest")
142 )
143 )

```

```

138     ),
139     list(
140         text = "Non_parametric_tests",
141         key = "nonParametricInference",
142         subMenuProps = list(
143             items = list(
144                 list(
145                     text = "Continuous_Data_Tests",
146                     key = "continuousDataTests",
147                     subMenuProps = list(
148                         items = list(
149                             list(text = "Mann-Whitney_U_Test", key = "
150                                 mannWhitney", href = "#!/mannWhitney"),
151                             list(text = "Kruskal-Wallis_Test --->post_hoc", key =
152                                 "kruskalWallis", href = "#!/kruskalWallis",
153                                 subMenuProps = list(
154                                     items = list(
155                                         list(text = "Dunn's_Test", key = "dunnTest"
156                                             , href = "#!/dunnTest"),
157                                         list(text = "Conover-Inman_Test", key = "
158                                             conoverInmanTest", href = "#!/
159                                             conoverInmanTest"),
160                                         list(text = "Steel-Dwass_Test", key = "
161                                             steelDwassTest", href = "#!/
162                                             steelDwassTest")
163                                     )
164                                 )
165                             ),
166                             list(text = "Wilcoxon_signed-rank_test -->post_hoc",
167                                 key = "wilcoxon", href = "#!/wilcoxon",
168                                 subMenuProps = list(
169                                     items = list(
170                                         list(text = "Pairwise_Wilcoxon_test_with_
171                                             adjustment", key = "pairwiseWilcoxon",
172                                             href = "#!/pairwiseWilcoxon")
173                                     )
174                                 )
175                             ),
176                             list(text = "Friedman_test --->_post_hoc", key = "
177                                 friedman", href = "#!/friedman",
178                                 subMenuProps = list(
179                                     items = list(
180                                         list(text = "Nemenyi_Test", key = "
181                                             nemenyiTest", href = "#!/nemenyiTest")
182                                     )
183                                 )
184                             ),
185                             list(text = "Spearman's_rank_correlation_coefficient
186                                 ", key = "spearman", href = "#!/spearman"),
187                             list(text = "Kendall's_tau_coefficient", key = "
188                                 kendall", href = "#!/kendall"),
189                             list(text = "Kolmogorov-Smirnov_test", key = "

```

```

        kolmogorov", href = "#!/kolmogorov"),
list(text = "Anderson-Darling_test", key = "
    andersonDarling", href = "#!/andersonDarling"),
list(text = "Shapiro-Wilk_test", key = "shapiroWilk"
    , href = "#!/shapiroWilk"),
list(text = "Lilliefors_test", key = "lilliefors",
    href = "#!/lilliefors"),
list(text = "Cramer-vonMises_criterion", key = "
    cramerVonMises", href = "#!/cramerVonMises"),
list(text = "Mood's_median_test", key = "moodMedian"
    , href = "#!/moodMedian"),
list(text = "Savage_score_test", key = "savage",
    href = "#!/savage"),
list(text = "Kolmogorov-Smirnov_goodness-of-fit_test"
    , key = "ksGoodness", href = "#!/ksGoodness"),
list(text = "Mann-Kendall_trend_test", key = "
    mannKendall", href = "#!/mannKendall"),
list(text = "Sen's_slope_estimate", key = "senSlope"
    , href = "#!/senSlope"),
list(text = "Pettitt's_test", key = "pettitt", href =
    "#!/pettitt"),
list(text = "Seasonal_Kendall_test", key = "
    seasonalKendall", href = "#!/seasonalKendall"),
list(text = "Durbin-Watson_test", key = "
    durbinWatson", href = "#!/durbinWatson"),
list(text = "Van_der_Waerden_test", key = "
    vanDerWaerden", href = "#!/vanDerWaerden"),
list(text = "Brown-Forsythe_test", key = "
    brownForsythe", href = "#!/brownForsythe"),
list(text = "Bartlett's_test", key = "bartlett",
    href = "#!/bartlett"),
list(text = "Levene's_test", key = "levene", href =
    "#!/levene"),
list(text = "Durbin_test", key = "durbin", href = "#
    !/durbin")
    )
    )
),
list(
    text = "Quantal_Data_Tests",
    key = "quantalDataTests",
    subMenuProps = list(
        items = list(
            list(text = "Chi-squared_test_for_independence", key =
                "chiSquared", href = "#!/chiSquared"),
            list(text = "Fisher's_exact_test", key = "fisher",
                href = "#!/fisher"),
            list(text = "McNemar's_test", key = "mcnemar", href =
                "#!/mcnemar"),
            list(text = "Cochran's_Q_test", key = "cochran",
                href = "#!/cochran"),
            list(text = "Jonckheere-Terpstra_test", key = "

```

```

        jonckheere", href = "#!/jonckheere"),
list(text = "Sign_test", key = "sign", href = "#!/
sign"),
list(text = "Wald-Wolfowitz_runs_test", key = "
waldWolfowitz", href = "#!/waldWolfowitz"),
list(text = "Median_test", key = "median", href = "#
!/median"),
list(text = "Page's_L_trend_test", key = "pageL",
href = "#!/pageL")
    )
    )
),
list(
    text = "Count_Data_Tests",
    key = "countDataTests",
    subMenuProps = list(
        items = list(
            list(text = "Poisson_Regression", key = "
poissonRegression", href = "#!/poissonRegression"
            ),
            list(text = "Negative_Binomial_Regression", key = "
negativeBinomialRegression", href = "#!/
negativeBinomialRegression"),
            list(text = "Zero-Inflated_Poisson_(ZIP)_Model", key = "
zipModel", href = "#!/zipModel"),
            list(text = "Zero-Inflated_Negative_Binomial_(ZINB)_
Model", key = "zinbModel", href = "#!/zinbModel")
            ,
            list(text = "Hurdle_Models", key = "hurdleModels",
href = "#!/hurdleModels"),
            list(text = "Quasi-Poisson_Regression", key = "
quasiPoissonRegression", href = "#!/
quasiPoissonRegression"),
            list(text = "Autoregressive_Poisson_Models", key = "
autoregressivePoissonModels", href = "#!/
autoregressivePoissonModels"),
            list(text = "Conway-Maxwell_Poisson_Model", key = "
conwayMaxwellPoissonModel", href = "#!/
conwayMaxwellPoissonModel"),
            list(text = "Integer-Valued_ARIMA_Models", key = "
integerValuedARIMAModels", href = "#!/
integerValuedARIMAModels"),
            list(text = "Latent_Class_Poisson_Model", key = "
latentClassPoissonModel", href = "#!/
latentClassPoissonModel"),
            list(text = "Multilevel_Poisson_Regression", key = "
multilevelPoissonRegression", href = "#!/
multilevelPoissonRegression"),
            list(text = "Dirichlet-Multinomial_Model", key = "
dirichletMultinomialModel", href = "#!/
dirichletMultinomialModel"),
            list(text = "Overdispersed_Poisson_Model", key = "

```

```

overdispersedPoissonModel", href = "#!/
overdispersedPoissonModel"),
231     list(text = "Hierarchical_Poisson_Regression", key =
        "hierarchicalPoissonRegression", href = "#!/
        hierarchicalPoissonRegression")
232     )
233     )
234     )
235     )
236     )
237     )
238     )
239     )
240 ),
241 list(
242     key = "calcPuiss",
243     text = "Calcul_de_Puissance",
244     iconProps = list(iconName = ""),
245     href = "#!/calcPuiss"),
246 list(
247     key = "Data_analysis:_Resampling",
248     text = "V)_Resampling",
249     iconProps = list(iconName = "DatabaseSync"),
250     subMenuProps = list(
251         items = list(list(text = "Bootstrap", key = "bootstrap",
252             href = "#!/bootstrap"),
253             list(text = "Permutation_Test", key = "
                permutation", href = "#!/permutation")
254         )
255     ),
256 list(
257     key = "Bayesian_Test",
258     text = "VI)_Bayesian_Methods",
259     iconProps = list(iconName = "ExploreData"),
260     href = "#!/bayesian"
261 ),
262 list(
263     key = "emailMessage",
264     text = "Contact",
265     iconProps = list(iconName = "Website"),
266     href = "mailto:me@example.com"
267 ),
268 ),
269 list(
270     key = "download",
271     text = "Download_Report",
272     iconProps = list(iconName = "Download")#ou CommandBarItem("
        Download_Report","Download",onClick = JS("( ) => alert('
        Download_Report_clicked')"))
273 ),
274 farItems = farItems,

```

```

275   style = list(width = "100%")
276 )
277
278 # En-tete
279 header <- function() {
280   tagList(
281     img(src = "Boubouim.png", class = "logo"),
282     div(Text(variant = "medium", "Statistical Test Selection Tool",
283           style = list(color = "purple", fontWeight = "bold")), class =
284           "title"),
285     command_bar
286   )
287 }

```

Home page:

```

1  homePage <- function() {
2    tagList(
3      makePage(
4        "Decision Support Tool for Statistical Test Selection in
5        Environmental Risk Assessment",
6        "Choose the Right Statistical Method for Your Study",
7        div(
8          makeCard(
9            "Welcome to the Statistical Test Selection Tool!",
10           Stack(
11             Text("This tool helps you choose the appropriate
12                  statistical test for environmental/ecotoxicology
13                  studies."),
14             Text("Use the menu on the left to explore statistical
15                  methods.")
16           )
17         ),
18         makeCard(
19           "Bootstrap Method",
20           Stack(
21             Text("The bootstrap method is a resampling technique
22                  used to estimate statistics on a population by
23                  sampling a dataset with replacement. It helps in
24                  assessing the variability and confidence intervals of
25                  the estimated parameters.")
26           )
27         ),
28         makeCard(
29           "Permutation Tests",
30           Stack(
31             Text("Permutation tests, also known as randomization
32                  tests, involve randomly rearranging the data points
33                  and recalculating the test statistic for each
34                  rearrangement. This method is useful for hypothesis
35                  testing when the distribution of the test statistic
36                  under the null hypothesis is unknown.")
37           )
38         )
39       )
40     )
41   }

```

```

25     ),
26     makeCard(
27       "Bayesian Methods",
28       Stack(
29         Text("Bayesian methods incorporate prior knowledge or
                beliefs into the statistical analysis. They update
                the probability of a hypothesis as more evidence or
                data becomes available. Bayesian inference is
                particularly useful in complex models and when prior
                information is available.")
30       )
31     )
32   )
33 )
34 )
35 }

```

fonction de mise en forme:

```

1  makePage <- function(title, subtitle, contents) {
2    tagList(
3      div(
4        class = "page-title",
5        span(
6          title,
7          class = "ms-fontSize-32 ms-fontWeight-semibold",
8          style = "color: #323130"
9        ),
10       span(
11         subtitle,
12         class = "ms-fontSize-14 ms-fontWeight-regular",
13         style = "color: #605E5C; margin: 14px;"
14       )
15     ),
16     contents
17   )
18 }
19
20
21 makeCard <- function(title, content, size = 12, style = "") {
22   div(
23     class = glue("card ms-depth-8 ms-sm{size} ms-xl{size}"),
24     style = style,
25     Stack(
26       tokens = list(childrenGap = 5),
27       Text(variant = "large", title, block = TRUE),
28       content
29     )
30   )
31 }
32
33
34 HorizontalStack <- function(...) {

```

```

35 Stack(
36   horizontal = TRUE,
37   tokens = list(childrenGap = 10),
38   ...
39 )
40 }

```

Menu navigation:

```

1 library(shiny)
2 library(shiny.fluent)
3 library(shiny.router)
4
5 navigation <- function() {
6   Nav(
7     groups = list(
8       list(
9         links = list(
10          list(
11            name = "Aquatic□Tests",
12            key = "aquaticTests",
13            icon = "HistoricalWeather",
14            links = list(
15              list(
16                name = "Essai□201",
17                url = "#!/Essai201",
18                key = "Essai201",
19                icon = "AdminELogoInverse32"
20              ),
21              list(
22                name = "Essai□218",
23                url = "#!/Essai218",
24                key = "Essai218",
25                icon = "AdminELogoInverse32"
26              ),
27              list(
28                name = "Essai□202",
29                url = "#!/Essai202",
30                key = "Essai202",
31                icon = "AdminELogoInverse32"
32              ),
33              list(
34                name = "Essai□203",
35                url = "#!/Essai203",
36                key = "Essai203",
37                icon = "AdminELogoInverse32"
38              ),
39              list(
40                name = "Essai□210",
41                url = "#!/Essai210",
42                key = "Essai210",
43                icon = "AdminELogoInverse32"
44              ),

```



```

45         list(
46             name = "Essai_211",
47             url = "#!/Essai211",
48             key = "Essai211",
49             icon = "AdminELogoInverse32"
50         ),
51         list(
52             name = "Essai_221",
53             url = "#!/Essai221",
54             key = "Essai221",
55             icon = "AdminELogoInverse32"
56         ),
57         list(
58             name = "Essai_239",
59             url = "#!/Essai239",
60             key = "Essai239",
61             icon = "AdminELogoInverse32"
62         )
63     ),
64 ),
65 list(
66     name = "Terrestrial_Tests",
67     key = "terrestrialTests",
68     icon = "Globe",
69     links = list(
70         list(
71             name = "Essai_205",
72             url = "#!/Essai205",
73             key = "Essai205",
74             icon = "AdminELogoInverse32"
75         ),
76         list(
77             name = "Essai_206",
78             url = "#!/Essai206",
79             key = "Essai206",
80             icon = "AdminELogoInverse32"
81         ),
82         list(
83             name = "Essai_208",
84             url = "#!/Essai208",
85             key = "Essai208",
86             icon = "Flower"
87         ),
88         list(
89             name = "Essai_216",
90             url = "#!/Essai216",
91             key = "Essai216",
92             icon = "AdminELogoInverse32"
93         ),
94         list(
95             name = "Essai_222",
96             url = "#!/Essai222",

```

```

97         key = "Essai222",
98         icon = "AdminELogoInverse32"
99     ),
100     list(
101         name = "Essai_223",
102         url = "#!/Essai223",
103         key = "Essai223",
104         icon = "AdminELogoInverse32"
105     ),
106     list(
107         name = "Essai_226",
108         url = "#!/Essai226",
109         key = "Essai226",
110         icon = "AdminELogoInverse32"
111     ),
112     list(
113         name = "Essai_227",
114         url = "#!/Essai227",
115         key = "Essai227",
116         icon = "Flower"
117     ),
118     list(
119         name = "Essai_232",
120         url = "#!/Essai232",
121         key = "Essai232",
122         icon = "AdminELogoInverse32"
123     )
124 )
125 ),
126 list(
127     name = "Bee_and_Pollinator_Tests",
128     key = "beePollinatorTests",
129     icon = "LadybugSolid",
130     links = list(
131         list(
132             name = "Essai_213",
133             url = "#!/Essai213214",
134             key = "Essai213",
135             icon = "AdminELogoInverse32"
136         ),
137         list(
138             name = "Essai_214",
139             url = "#!/Essai213214",
140             key = "Essai214",
141             icon = "AdminELogoInverse32"
142         ),
143         list(
144             name = "Essai_245",
145             url = "#!/Essai245",
146             key = "Essai245",
147             icon = "AdminELogoInverse32"
148         ),

```

```

149         list(
150             name = "Essai_246",
151             url = "#!/Essai246247",
152             key = "Essai246",
153             icon = "AdminELogoInverse32"
154         ),
155         list(
156             name = "Essai_247",
157             url = "#!/Essai246247",
158             key = "Essai247",
159             icon = "AdminELogoInverse32"
160         )
161     ),
162 ),
163 list(
164     name = "Non-target_arthropods",
165     key = "NTarthrop",
166     icon = "BugSolid",
167     links = list(
168         list(
169             name = "ESCORT",
170             url = "#!/ESCORT",
171             key = "ESCORT",
172             icon = "Bug"
173         )
174     )
175 ),
176 ),
177 ),
178 list(
179     name = "Internet_links",
180     links = list(
181         list(
182             name = "",
183             url = "",
184             key = "",
185             icon = ""
186         ),
187         list(
188             name = "",
189             url = "",
190             key = "",
191             icon = ""
192         )
193     )
194 ),
195 ),
196 initialSelectedKey = "home",
197 styles = list(
198     root = list(
199         height = "100%",
200         boxSizing = "border-box",

```

```

201     border = "1px_solid_#eee",
202     overflowY = "auto"
203   )
204 )
205 )
206 }

```

Menu permutation

```

1 permutationPage <- function() {
2   tagList(
3     makePage(
4       "Permutation_Test",
5       "Permutation_test_results",
6       div(
7         HorizontalStack(
8           makeCard("File_input",
9             div(
10              actionButton("run_permutation", "Run_
11                Permutation_Test")
12            ),
13            size = 4, style = "max-height: 320px"
14          ),
15          makeCard("Data_Preview", DTOutput("contents"), size = 8,
16            style = "max-height: 320px")
17        ),
18        div(
19          makeCard("Results", DTOutput("results"), style = "margin-
20            top: 20px;"),
21          makeCard("Permutation_Statistics", DTOutput("permutation_
22            data"), style = "margin-top: 20px;"),
23          makeCard("Permutation_Plot", plotOutput("permutation_plot"
24            ), style = "margin-top: 20px;")
25        )
26      )
27    )
28  )
29 }

```

Menu puissance

```

1 PuissancePage <- function() {
2   navbarPage(
3     "Calcul_de_Puissance",
4
5     # Onglet d'introduction/théorie
6     tabPanel("Introduction",
7       fluidPage(
8         h2("Introduction_au_Calcul_de_la_Puissance_
9           Statistique"),
10        p("Cette_section_fournit_une_introduction_à_la_
11          puissance_statistique, _ses_concepts_et_son_
12          importance_dans_les_analyses_statistiques."),

```

```

10     h3("Théorie"),
11     p("La puissance statistique est la probabilité de détecter un effet, si cet effet existe réellement. Un test statistique avec une puissance élevée a plus de chances de rejeter une hypothèse nulle fausse."),
12     p("Les paramètres influençant la puissance incluent le niveau de signification ( $\alpha$ ), la taille de l'effet, et la taille de l'échantillon."),
13     p("Il est crucial de calculer la puissance avant de mener une étude pour s'assurer que l'étude est suffisamment sensible pour détecter l'effet d'intérêt.")
14   )
15 ),
16
17 # Onglet de calcul de la puissance
18 tabPanel("Calcul de Puissance",
19   pageWithSidebar(
20     headerPanel("Calcul de la Puissance Statistique"),
21     sidebarPanel(
22       # Sélection de la méthode de calcul de puissance
23       selectInput("method", "Choisir la méthode de calcul de puissance:",
24         choices = c("t-test", "ANOVA", "Régression Log-Logistique")),
25
26       # Inputs pour t-test
27       conditionalPanel(
28         condition = "input.method == 't-test'",
29         h4("Paramètres pour t-test"),
30         selectInput("groupColumn_t", "Choisir la colonne de groupe", choices = NULL),
31         selectInput("valueColumn_t", "Choisir la colonne des valeurs", choices = NULL),
32         sliderInput("alphaLevel_t", "Niveau de signification ( $\alpha$ ):", min = 0.01, max = 0.1, value = 0.05)
33       ),
34
35       # Inputs pour ANOVA
36       conditionalPanel(
37         condition = "input.method == 'ANOVA'",
38         h4("Paramètres pour ANOVA"),
39         selectInput("groupColumn_a", "Choisir la colonne de groupe", choices = NULL),
40         selectInput("valueColumn_a", "Choisir la colonne des valeurs", choices = NULL),
41         sliderInput("alphaLevel_a", "Niveau de signification ( $\alpha$ ):", min = 0.01, max = 0.1, value = 0.05)

```

```

42     ),
43
44     # Inputs pour Régression Log-Logistique
45     conditionalPanel(
46       condition = "input.method_== 'Régression Log-
47         Logistique'",
48       h4("Paramètres pour Régression Log-Logistique"),
49       selectInput("doseColumn_1", "Choisir la colonne
50         des doses", choices = NULL),
51       selectInput("responseColumn_1", "Choisir la
52         colonne des réponses", choices = NULL),
53       sliderInput("alphaLevel_1", "Niveau de
54         signification (alpha):", min = 0.01, max =
55         0.1, value = 0.05)
56     )
57   ),
58   mainPanel(
59     h4("Graphique de Puissance"),
60     plotOutput("powerPlot"),
61     h4("Résultat de la Puissance"),
62     verbatimTextOutput("powerResult"),
63     uiOutput("column_select2")
64   )
65 )
66 }

```

Page essai standardisé ESCORT

```

1 library(shiny)
2 library(DT)
3 library(nlme)
4 library(drc)
5 library(shinycssloaders)
6
7 ESCORTPage <- function() {
8   tagList(
9     fluidPage(
10       titlePanel("ESCORT: Arthropodes non cibles"),
11       fluidRow(
12         column(4,
13           textInput(inputId = "chemical", "Nom du produit
14             chimique testé", value = "Produit chimique"),
15           selectInput("conc_unit", "Unité de Concentration",
16             choices = c("g/L", "mg/L", "microg/L", "ng/L")),
17           fileInput('datafile_ESCORT', 'Sélectionner un fichier
18             ', accept = c('.csv')),
19           radioButtons('model_ESCORT', 'Sélectionner le modèle
20             de régression',
21               choices = c('log-logistic 2 paramètres'
22                 = '112',
23                 'log-logistic 3 paramètres'

```

```

19         = 'l13',
20         'log-logistic_4_paramètres'
21         = 'l14',
22         'Linéaire' = 'lm',
23         'Non-linéaire' = 'nls',
24         'Mixte' = 'lme'),
25         selected = 'l12'),
26         numericInput(inputId = "ecx_ESCORT", "Déterminer la
27             concentration efficace X%", value = 50, min = 0,
28             max = 100),
29         radioButtons('test_method_ESCORT', 'Sélectionner la m
30             éthode de test d\'hypothèse',
31             choices=c('ANOVA', 'anova', 'Ré
32             gression logistique', 'logistic'), selected='anova'),
33         actionButton("buttonRunStep1", "Exécuter", icon=
34             icon("check"), width="100px")
35         ),
36         column(8,
37             mainPanel(
38                 width=12,
39                 DT::dataTableOutput('rawdata'),
40                 br(),
41                 tableOutput('drc_result'),
42                 withSpinner(plotOutput('drc_plot', width="100%",
43                     height="850px"), type=4, color='#9e0093'),
44                 br(),
45                 withSpinner(verbatimTextOutput("test_result")),
46                 br()
47             )
48         ),
49         tabPanel(
50             "Télécharger le rapport",
51             fixedRow(
52                 column(
53                     8,
54                     br(),
55                     HTML("<font face=verdana size=5 color=#9e0093><b>Télé
56                         charger le rapport d'analyse</b></font>"),
57                     br(), br(), br(),
58                     radioButtons("format", "Sélectionner le format du
59                         rapport", c('Word'), inline = TRUE),
60                     downloadButton("DownloadReport", "Télécharger le
61                         rapport", icon = icon("fas fa-download"), style = '
62                         background-color:#9e0093;color:#000000;border-
63                         color:#9d9d9d;')
64                 )
65             )
66         )
67     ))
68 }

```

```

1 library(shiny)
2 library(DT)
3 library(drc)
4 library(multcomp)
5 library(shinycssloaders)
6
7 Essai201Page <- function() {
8   tagList(
9     makePage(
10       "Freshwater_Alga_and_Cyanobacteria_Growth_Inhibition_Test",
11       "(Essai_201)",
12       fluidPage(
13         column(4,
14           textInput(inputId = "chemical", "Nom_du_produit_
15             chimique_testé", value = "Produit_chimique"),
16           selectInput("conc_unit201", "Unité_de_Concentration",
17             choices = c("g/L", "mg/L", "microg/L", "ng/L")),
18           fileInput('datafile_ESSAI201', '', accept = c('.csv')
19             ),
20           tags$br(),
21           selectInput('model_ESSAI201', 'Sélectionner_le_modèle
22             de_régression',
23             choices = c('log-logistic_2_paramètres'
24               = 'l12',
25               'log-logistic_3_paramètres'
26               = 'l13',
27               'log-logistic_4_paramètres'
28               = 'l14'),
29             selected = 'l12'),
30           numericInput(inputId = "ecx_ESSAI201", "Déterminer_la
31             concentration_efficace_X%", value = 50, min = 0,
32             max = 100),
33           selectInput('test_method_ESSAI201', 'Sélectionner_la_
34             méthode_de_test_d\'hypothèse',
35             choices=c('Test de Dunnett'='Dunnett
36               '),selected='Dunnett'),
37           actionButton("buttonRunStep201", "Exécuter", icon=
38             icon("check"),width="100px")
39         ),
40         column(8,
41           mainPanel(
42             width=12,
43             DT::dataTableOutput('rawdata201'),
44             br(),
45             tableOutput('drc_result201'),
46             withSpinner(plotOutput('drc_plot201',width=
47               "100%",height="850px"),type=4,color='#9e0093'),
48             br(),
49             withSpinner(verbatimTextOutput("test_result201")),
50             br()
51           )
52         )
53       )
54     )
55   )
56 }

```



```

39     ),
40     tabPanel(
41       "Télécharger le rapport",
42       fixedRow(
43         column(
44           8,
45           br(),
46           HTML("<font face=verdana size=5 color=#fff><b>Télé
47             charger le rapport d'analyse</b></font>"),
48           br(), br(), br(),
49           radioButtons('format201', 'Document format', c('PDF',
50             'HTML', 'Word'), inline = TRUE),
51           downloadButton("DownloadReport201", "Télécharger le
52             rapport", icon = icon("fas fa-download"))#, style =
53             'background-color:#fff; color:#000000; border-
54             color:#fff;')
55         )
56       )
57     )
58   )
59 }

```

Page essai standardisé 202

```

1 library(shiny)
2 library(DT)
3 library(drc)
4 library(multcomp)
5 library(shinycssloaders)
6
7 Essai202Page <- function() {
8   tagList(
9     makePage(
10       "Daphnia sp. Acute Immobilisation Test",
11       "(Essai 202)",
12       fluidRow(
13         column(4,
14           textInput(inputId = "chemical", "Nom du produit
15             chimique testé", value = "Produit chimique"),
16           selectInput("conc_unit202", "Unité de Concentration",
17             choices = c("g/L", "mg/L", "microg/L", "ng/L")),
18           fileInput('datafile_ESSAI202', '', accept = c('.csv')
19             ),
20           tags$br(),
21           selectInput('model_ESSAI202', 'Sélectionner le modèle
22             de régression',
23             choices = c('log-logistic 2 paramètres'
24               = 'l12',
25               'log-logistic 3 paramètres'
26               = 'l13',
27               'log-logistic 4 paramètres'

```

```

22         = '114'),
23         selected = '112'),
24         numericInput(inputId = "ecx_ESSAI202", "Déterminer la
           concentration efficace X%", value = 50, min = 0,
           max = 100),
25         selectInput('test_method_ESSAI202', 'Sélectionner la
           méthode de test d\'hypothèse',
           choices=c('Test de Dunnett', 'Dunnett
           '), selected='Dunnett'),
26         actionButton("buttonRunStep202", "Exécuter", icon=
           icon("check"), width="100px")
27     ),
28     column(8,
29         mainPanel(
30             width=12,
31             DT::dataTableOutput('rawdata202'),
32             br(),
33             tableOutput('drc_result202'),
34             withSpinner(plotOutput('drc_plot202', width=
           "100%", height="850px"), type="4", color="#9e0093'),
           br()
35         )
36     ),
37     tabPanel(
38         "Télécharger le rapport",
39         fixedRow(
40             column(
41                 8,
42                 br(),
43                 HTML("<font face=verdana size=5 color=#9e0093><b>Télé
44                     charger le rapport d'analyse</b></font>"),
45                 br(), br(), br(),
46                 radioButtons("format", "Sélectionner le format du
47                     rapport", c('Word'), inline = TRUE),
48                 downloadButton("DownloadReport", "Télécharger le
49                     rapport", icon = icon("fas fa-download"), style = '
50                     background-color:#9e0093; color:#000000; border-
51                     color:#9d9d9d;')
52             )
53         )
54     )
55 }

```

Page essai standardisé 203

```

1 library(shiny)
2 library(DT)
3 library(drc)
4 library(multcomp)
5 library(shinycssloaders)
6

```

```

7 Essai203Page <- function() {
8   tagList(
9     makePage(
10       "Fish, Acute Toxicity Test",
11       "(Essai 203)",
12       fluidRow(
13         column(4,
14           textInput(inputId = "chemical", "Nom du produit chimique testé", value = "Produit chimique"),
15           selectInput("conc_unit203", "Unité de Concentration",
16             choices = c("g/L", "mg/L", "microg/L", "ng/L")),
17           fileInput('datafile_ESSAI203', '', accept = c('.csv')),
18           tags$br(),
19           selectInput('model_ESSAI203', 'Sélectionner le modèle de régression',
20             choices = c('log-logistic 2 paramètres' = 'l12',
21               'log-logistic 3 paramètres' = 'l13',
22               'log-logistic 4 paramètres' = 'l14'),
23             selected = 'l12'),
24           numericInput(inputId = "ecx_ESSAI203", "Déterminer la concentration efficace X%", value = 50, min = 0, max = 100),
25           selectInput('test_method_ESSAI203', 'Sélectionner la méthode de test d\'hypothèse',
26             choices = c('Test de Dunnett' = 'Dunnett'),
27             selected = 'Dunnett'),
28           actionButton("buttonRunStep203", "Exécuter", icon = icon("check"), width = "100px")
29         ),
30         column(8,
31           mainPanel(
32             DT::dataTableOutput('rawdata203'),
33             br(),
34             tableOutput('drc_result203'),
35             withSpinner(plotOutput('drc_plot203', width = "100%", height = "850px"), type = "4", color = "#9e0093"),
36             br(),
37             withSpinner(verbatimTextOutput("test_result203")),
38             br()
39           ),
40           tabPanel(
41             "Télécharger le rapport",
42             fixedRow(
43               column(
44                 8,
45                 br(),

```

```

46 HTML("<font_ face=verdana_ size=5_ color=#9e0093><b>Télé
    charger_ le_ rapport_ d'analyse</b></font>"),
47 br(), br(), br(),
48 radioButtons("format", "Sélectionner_ le_ format_ du_
    rapport", c('Word'), inline = TRUE),
49 downloadButton("DownloadReport", "Télécharger_ le_
    rapport", icon = icon("fas_ fa-download"), style = '
    background-color: #9e0093;_ color: #000000;_ border-
    color: #9d9d9d;')
50 )
51 )
52 )
53 )
54 ))
55 }

```

Page essai standardisé 205

```

1 library(shiny)
2 library(DT)
3 library(drc)
4 library(multcomp)
5 library(shinycssloaders)
6
7 Essai205Page <- function() {
8   tagList(
9     fluidPage(
10       titlePanel("Avian_ Dietary_ Toxicity_ Test"),
11       fluidRow(
12         column(4,
13           textInput(inputId = "chemical", "Nom_ du_ produit_
14             chimique_ testé", value = "Produit_ chimique"),
15           selectInput("conc_ unit", "Unité_ de_ Dose", choices = c
16             ("g/kg", "mg/kg", "microg/kg", "ng/kg")),
17           fileInput('datafile_ ESSAI205', 'Sélectionner_ un_
18             fichier', accept = c('.csv')),
19           radioButtons('model_ ESSAI205', 'Sélectionner_ le_ modè
20             le_ de_ régression',
21               choices = c('log-logistic_ 2_ paramètres'
22                 = 'l12',
23                 'log-logistic_ 3_ paramètres'
24                 = 'l13',
25                 'log-logistic_ 4_ paramètres'
26                 = 'l14'),
27               selected = 'l12'),
28           numericInput(inputId = "ecx_ ESSAI205", "Déterminer_ la
29             dose_ efficace_ X%", value = 50, min = 0, max =
30             100),
31           radioButtons('test_ method_ ESSAI205', 'Sélectionner_ la
32             méthode_ de_ test_ d\'hypothèse',
33             choices = c('Test de Dunnett' = 'Dunnett
34               '), selected = 'Dunnett'),
35           actionButton("buttonRunStep1", "Exécuter", icon =

```

```

        icon("check"),_width=_ "100px")
25  ),
26  column(8,
27  mainPanel(
28  width=_12,
29  DT::dataTableOutput('rawdata205'),
30  br(),
31  tableOutput('drc_result205'),
32  withSpinner(plotOutput('drc_plot205',_width=_
    "100%",_height=_ "850px"),_type=_4,_color=_ '#9e0093'),
33  br(),
34  withSpinner(verbatimTextOutput("test_result205")),
35  br()
36  )
37  ),
38  tabPanel(
39  "Télécharger le rapport",
40  fixedRow(
41  column(
42  8,
43  br(),
44  HTML("<font_face=verdana_size=5_color=#9e0093><b>Télé
    charger le rapport d'analyse</b></font>"),
45  br(), br(), br(),
46  radioButtons("format", "Sélectionner le format du
    rapport", c('Word'), inline = TRUE),
47  downloadButton("DownloadReport", "Télécharger le
    rapport", icon = icon("fas_fa-download"), style = '
    background-color:#9e0093;_color:#000000;_border-
    color:#9d9d9d;')
48  )
49  )
50  )
51  )
52  ))
53  }

```

Page essai standardisé 206

```

1  library(shiny)
2  library(DT)
3  library(drc)
4  library(multcomp)
5  library(shinycssloaders)
6
7  Essai206Page <- function() {
8    tagList(
9      fluidPage(
10       titlePanel("Avian_Reproduction"),
11       fluidRow(
12         column(4,
13           textInput(inputId = "chemical", "Nom du produit
             chimique testé", value = "Produit chimique"),

```

```

14     selectInput("conc_unit", "Unité de Dose", choices = c
15       ("g/kg", "mg/kg", "microg/kg", "ng/kg")),
16     fileInput('datafile_ESSAI206', 'Sélectionner un
      fichier', accept = c('.csv')),
17     radioButtons('model_ESSAI206', 'Sélectionner le modèle
      de régression',
18       choices = c('log-logistic_2_paramètres'
19         = 'l12',
20         'log-logistic_3_paramètres'
21         = 'l13',
22         'log-logistic_4_paramètres'
23         = 'l14'),
24     selected = 'l12'),
25     numericInput(inputId = "ecx_ESSAI206", "Déterminer la
      dose efficace X%", value = 50, min = 0, max =
26       100),
27     radioButtons('test_method_ESSAI206', 'Sélectionner la
      méthode de test d\'hypothèse',
28     choices = c('Test de Dunnett' = 'Dunnett
29       '), selected = 'Dunnett'),
30     actionButton("buttonRunStep1", "Exécuter", icon =
      icon("check"), width = "100px")
31   ),
32   column(8,
33     mainPanel(
34       width = 12,
35       DT::dataTableOutput('rawdata206'),
36       br(),
37       tableOutput('drc_result206'),
38       withSpinner(plotOutput('drc_plot206', width =
39         "100%", height = "850px"), type = 4, color = '#9e0093'),
40       br(),
41       withSpinner(verbatimTextOutput("test_result206")),
42       br()
43     )
44   ),
45   tabPanel(
46     "Télécharger le rapport",
47     fixedRow(
48       column(
49         8,
50         br(),
51         HTML("<font face=verdana size=5 color=#9e0093><b>Télé
52           charger le rapport d'analyse</b></font>"),
53         br(), br(), br(),
54         radioButtons("format", "Sélectionner le format du
55           rapport", c('Word'), inline = TRUE),
56         downloadButton("DownloadReport", "Télécharger le
57           rapport", icon = icon("fas fa-download"), style = '
58           background-color:#9e0093; color:#000000; border-
59           color:#9d9d9d;')
60       )
61     )
62   )

```

```

49     )
50   )
51 )
52 ))
53 }

```

Page essai standardisé 208

```

1 library(shiny)
2 library(DT)
3 library(drc)
4 library(shinycssloaders)
5
6 Essai208Page <- function() {
7   tagList(
8     fluidPage(
9       titlePanel("Terrestrial Plant Test: Seedling Emergence and
10         Seedling Growth Test"),
11       fluidRow(
12         column(4,
13           textInput(inputId = "chemical", "Nom du produit
14             chimique testé", value = "Produit chimique"),
15           selectInput("conc_unit", "Unité de Concentration",
16             choices = c("g/L", "mg/L", "microg/L", "ng/L")),
17           fileInput('datafile_ESSAI208', 'Sélectionner un
18             fichier', accept = c('.csv')),
19           radioButtons('model_ESSAI208', 'Sélectionner le modèle
20             de régression',
21             choices = c('log-logistic 2 paramètres'
22               = 'l12',
23               'log-logistic 3 paramètres'
24               = 'l13',
25               'log-logistic 4 paramètres'
26               = 'l14'),
27             selected = 'l12'),
28           numericInput(inputId = "ecx_ESSAI208", "Déterminer la
29             concentration efficace X%", value = 50, min = 0,
30             max = 100),
31           radioButtons('test_method_ESSAI208', 'Sélectionner la
32             méthode de test d\'hypothèse',
33             choices = c('ANOVA' = 'anova'), selected
34             = 'anova'),
35           actionButton("buttonRunStep1", "Exécuter", icon =
36             icon("check"), width = "100px")
37         ),
38       column(8,
39         mainPanel(
40           width = 12,
41           DT::dataTableOutput('rawdata'),
42           br(),
43           tableOutput('drc_result'),
44           withSpinner(plotOutput('drc_plot', width = "100%",
45             height = "850px"), type = 4, color = '#9e0093'),

```

```

32         br(),
33         withSpinner(verbatimTextOutput("test_result")),
34         br()
35     )
36 ),
37 tabPanel(
38     "Télécharger le rapport",
39     fixedRow(
40         column(
41             8,
42             br(),
43             HTML("<font face=verdana size=5 color=#9e0093><b>Télécharger le rapport d'analyse</b></font>"),
44             br(), br(), br(),
45             radioButtons("format", "Sélectionner le format du rapport", c('Word'), inline = TRUE),
46             downloadButton("DownloadReport", "Télécharger le rapport", icon = icon("fas fa-download"), style = 'background-color:#9e0093; color:#000000; border-color:#9d9d9d;')
47         )
48     )
49 )
50 )
51 ))
52 }

```

Page essai standardisé 210

```

1 library(shiny)
2 library(DT)
3 library(drc)
4 library(shinycssloaders)
5
6 Essai210Page <- function() {
7   tagList(
8     makePage(
9       "Fish, Early-life Stage Toxicity Test",
10      "(Essai 210)",
11      fluidRow(
12        column(4,
13          textInput(inputId = "chemical", "Nom du produit chimique testé", value = "Produit chimique"),
14          selectInput("conc_unit210", "Unité de Concentration", choices = c("g/L", "mg/L", "microg/L", "ng/L")),
15          fileInput('datafile_ESSAI210', '', accept = c('.csv'))
16        ),
17        tags$br(),
18        selectInput('model_ESSAI210', 'Sélectionner le modèle de régression',
19          choices = c('log-logistic_2 paramètres' = '112',
20                     'log-logistic_3 paramètres'

```



```

20         = '113',
          'log-logistic_4_paramètres'
          = '114'),
21         selected = '112'),
22         numericInput(inputId = "ecx_ESSAI210", "Déterminer la
          concentration efficace X%", value = 50, min = 0,
          max = 100),
23         selectInput('test_method_ESSAI210', 'Sélectionner la
          méthode de test d\'hypothèse',
24         choices=c('Test de Dunnett', 'Dunnett
          '), selected='Dunnett'),
25         actionButton("buttonRunStep210", "Exécuter", icon=
          icon("check"), width="100px")
26     ),
27     column(8,
28     mainPanel(
29     width=12,
30     DT::dataTableOutput('rawdata'),
31     br(),
32     tableOutput('drc_result'),
33     withSpinner(plotOutput('drc_plot', width="100%",
          height="850px"), type=4, color='#9e0093'),
          br(),
          withSpinner(verbatimTextOutput("test_result")),
          br()
34     ),
35     ),
36     ),
37     ),
38     ),
39     tabPanel(
40     "Télécharger le rapport",
41     fixedRow(
42     column(
43     8,
44     br(),
45     HTML("<font face=verdana size=5 color=#9e0093><b>Télé
          charger le rapport d'analyse</b></font>"),
46     br(), br(), br(),
47     radioButtons("format", "Sélectionner le format du
          rapport", c('Word'), inline = TRUE),
48     downloadButton("DownloadReport", "Télécharger le
          rapport", icon = icon("fas fa-download"), style = '
          background-color:#9e0093; color:#000000; border-
          color:#9d9d9d;')
49     ),
50     ),
51     ),
52     ),
53     ))
54 }

```

Page essai standardisé 211

```

1 library(shiny)
2 library(DT)

```

```

3 library(drc)
4 library(multcomp)
5 library(shinycssloaders)
6
7 Essai211Page <- function() {
8   tagList(
9     makePage(
10       "Daphnia_magna_Reproduction_Test",
11       "(Essai_211)",
12       fluidRow(
13         column(4,
14           textInput(inputId = "chemical", "Nom_du_produit_
15             chimique_testé", value = "Produit_chimique"),
16           selectInput("conc_unit211", "Unité_de_Concentration",
17             choices = c("g/L", "mg/L", "microg/L", "ng/L")),
18           fileInput('datafile_ESSAI211', '', accept = c('.csv')
19             ),
20           tags$br(),
21           selectInput('model_ESSAI211', 'Sélectionner_le_modèle
22             de_régression',
23             choices = c('linéaire' = 'lm',
24               'non-linéaire' = 'nls',
25               'mixte' = 'lme'),
26             selected = 'lm'),
27           numericInput(inputId = "ecx_ESSAI211", "Déterminer_la
28             concentration_efficace_X%", value = 50, min = 0,
29             max = 100),
30           selectInput('test_method_ESSAI211', 'Sélectionner_la
31             méthode_de_test_d\'hypothèse',
32             choices=c('ANOVA'='anova'), selected
33             ='anova'),
34           actionButton("buttonRunStep211", "Exécuter", icon=
35             icon("check"), width="100px")
36         ),
37         column(8,
38           mainPanel(
39             width=12,
40             DT::dataTableOutput('rawdata'),
41             br(),
42             tableOutput('drc_result'),
43             withSpinner(plotOutput('drc_plot', width="100%",
44               height="850px"), type="4", color="#9e0093'),
45             br(),
46             withSpinner(verbatimTextOutput("test_result")),
47             br()
48           )
49         ),
50         tabPanel(
51           "Télécharger_le_rapport",
52           fixedRow(
53             column(
54               8,

```

```

45     br(),
46     HTML("<font_ face=verdana_ size=5_ color=#9e0093><b>Télé
         charger_ le_ rapport_ d'analyse</b></font>"),
47     br(), br(), br(),
48     radioButtons("format", "Sélectionner_ le_ format_ du_
         rapport", c('Word'), inline = TRUE),
49     downloadButton("DownloadReport", "Télécharger_ le_
         rapport", icon = icon("fas_ fa-download"), style = '
         background-color:#9e0093;_ color:#000000;_ border-
         color:#9d9d9d;')
50   )
51 )
52 )
53 )
54 ))
55 }

```

Page essai standardisé 216

```

1 library(shiny)
2 library(DT)
3 library(nlme)
4 library(shinycssloaders)
5
6 Essai216Page <- function() {
7   tagList(
8     fluidPage(
9       titlePanel("Soil_ Microorganisms:_ Nitrogen_ Transformation_ Test_
10         "),
11       fluidRow(
12         column(4,
13           textInput(inputId = "chemical", "Nom_ du_ produit_
14             chimique_ testé", value = "Produit_ chimique"),
15           selectInput("conc_ unit", "Unité_ de_ Concentration",
16             choices = c("g/kg", "mg/kg", "microg/kg", "ng/kg")
17           ),
18           fileInput('datafile_ESSAI216', 'Sélectionner_ un_
19             fichier', accept = c('.csv')),
20           radioButtons('model_ESSAI216', 'Sélectionner_ le_ modè
21             le_ de_ régression',
22             choices = c('Linéaire' = 'lm', 'Non-liné
23               aire' = 'nls', 'Mixte' = 'lme'),
24             selected = 'lm'),
25           numericInput(inputId = "ecx_ESSAI216", "Déterminer_ la
26             _concentration_ efficace_ X%", value = 50, min = 0,
27             max = 100),
28           radioButtons('test_method_ESSAI216', 'Sélectionner_ la
29             _méthode_ de_ test_ d\'hypothèse',
30             choices = c('ANOVA' = 'anova'), selected
31             = 'anova'),
32           actionButton("buttonRunStep1", "Exécuter", icon =
33             icon("check"), width = "100px")
34         ),
35       )
36     )
37   )
38 }

```

```

23 #####column(8,
24 #####mainPanel(
25 #####width=12,
26 #####DT::dataTableOutput('rawdata'),
27 #####br(),
28 #####tableOutput('drc_result'),
29 #####withSpinner(plotOutput('drc_plot',width=100%,
    height=850px),type=4,color='#9e0093'),
30 #####br(),
31 #####withSpinner(verbatimTextOutput("test_result")),
32 #####br()
33 #####)
34 #####),
35 #####tabPanel(
36 #####  "Télécharger le rapport",
37 #####  fixedRow(
38 #####    column(
39 #####      8,
40 #####      br(),
41 #####      HTML("<font face=verdana size=5 color=#9e0093><b>Télé
    charger le rapport d'analyse</b></font>"),
42 #####      br(), br(), br(),
43 #####      radioButton("format", "Sélectionner le format du
    rapport", c('Word'), inline = TRUE),
44 #####      downloadButton("DownloadReport", "Télécharger le
    rapport", icon = icon("fas fa-download"), style = '
    background-color:#9e0093;color:#000000;border-
    color:#9d9d9d;')
45 #####    )
46 #####  )
47 #####)
48 #####)
49 #####))
50 }

```

Page essai standardisé 218

```

1 library(shiny)
2 library(DT)
3 library(nlme)
4 library(shinycssloaders)
5
6 Essai218Page <- function() {
7   tagList(
8     makePage(
9       "Chironomus",
10      "(Essai_218)",
11      fluidRow(
12        column(4,
13          textInput(inputId = "chemical", "Nom du produit
    chimique testé", value = "Produit chimique"),
14          selectInput("conc_unit218", "Unité de Concentration",
    choices = c("g/L", "mg/L", "microg/L", "ng/L")),

```

```

15     fileInput('datafile_ESSAI218', '', accept = c('.csv')
16     ),
17     tags$br(),
18     selectInput('model_E218_mortality',
19                 'Select_fitting_model_for_mortality',
20                 choices = c('log-logistic_2_parameters'
21                             = 'l12',
22                             'log-logistic_3_parameters'
23                             = 'l13',
24                             'log-logistic_4_parameters'
25                             = 'l14'),
26                 selected = 'l12'),
27     selectInput('model_E218_emergence',
28                 'Select_fitting_model_for_emergence',
29                 choices = c('log-logistic_2_parameters'
30                             = 'l12',
31                             'log-logistic_3_parameters'
32                             = 'l13',
33                             'log-logistic_4_parameters'
34                             = 'l14'),
35                 selected = 'l12'),
36     selectInput('model_E218_development',
37                 'Select_fitting_model_for_development_
38                 rate',
39                 choices = c('log-logistic_3_parameters'
40                             = 'l13',
41                             'log-logistic_4_parameters'
42                             = 'l14'),
43                 selected = 'l13'),
44     tags$br(),
45     numericInput(inputId = "ecx_ESSAI218", "Determine_
46     effect_concentration_X%", value = 50, min = 0, max
47     = 100),
48     tags$br(),
49     selectInput('test_method_E218_mortality',
50                 'Select_hypothesis_testing_method_for_
51                 mortality',
52                 choices = c("Cochran-Armitage_test" = '
53                             CA',
54                             "Fisher's_exact_test_with_BH_
55                             correction" = 'Fisher'),
56                 selected = 'CA'),
57     selectInput('test_method_E218_emergence',
58                 'Select_hypothesis_testing_method_for_
59                 emergence_ratio',
60                 choices = c("Cochran-Armitage_test" = '
61                             CA',
62                             "Fisher's_exact_test_with_BH_
63                             correction" = 'Fisher'),
64                 selected = 'CA'),
65     selectInput('test_method_E218_development',
66                 'Select_hypothesis_testing_method_for_

```

```

49         development_rate',
        choices = c("Dunnett's_test" = 'Dunnett',
50                   ,
51                   "Steel's_test" = 'Steel'),
        selected = 'Dunnett'),
52     h5("You can see the bartlett's test result for
        homogeneity of variance of development rate, and
        then select testing method."),
53     actionButton("buttonRunStep218", "Exécuter", icon =
        icon("check"), width = "100px")
54   ),
55   column(8,
56     mainPanel(
57       width = 12,
58       DT::dataTableOutput('rawdata218'),
59       br(),
60       tableOutput('drc_result218'),
61       withSpinner(plotOutput('drc_plot218', width = "100%
        ", height = "850px"), type = 4, color = '#9e0093
        '),
62       br(),
63       withSpinner(verbatimTextOutput("test_result218")),
64       br()
65     )
66   ),
67   tabPanel(
68     "Télécharger le rapport",
69     fixedRow(
70       column(
71         8,
72         br(),
73         HTML("<font face=verdana size=5 color=#9e0093><b>Télé
        charger le rapport d'analyse</b></font>"),
74         br(), br(), br(),
75         radioButtons("format", "Sélectionner le format du
        rapport", c('Word'), inline = TRUE),
76         downloadButton("DownloadReport", "Télécharger le
        rapport", icon = icon("fas fa-download"), style = '
        background-color:#9e0093; color:#000000; border-
        color:#9d9d9d;')
77       )
78     )
79   )
80 )
81 ))
82 }

```

Page essai standardisé 221

```

1 library(shiny)
2 library(DT)
3 library(drc)
4 library(multcomp)

```

```

5 library(shinycssloaders)
6
7 Essai221Page <- function() {
8   tagList(
9     makePage(
10       "Lemna_sp._Growth_Inhibition_Test_",
11       "(Essai_221)",
12       fluidRow(
13         column(4,
14           textInput(inputId = "chemical", "Nom_du_produit_
15             chimique_testé", value = "Produit_chimique"),
16           selectInput("conc_unit221", "Unité_de_Concentration",
17             choices = c("g/L", "mg/L", "microg/L", "ng/L")),
18           fileInput('datafile_ESSAI221', '', accept = c('.csv')
19             ),
20           tags$br(),
21           selectInput('model_ESSAI221', 'Sélectionner_le_modèle
22             de_régression',
23               choices = c('log-logistic_2_paramètres'
24                 = 'l12',
25                 'log-logistic_3_paramètres'
26                 = 'l13',
27                 'log-logistic_4_paramètres'
28                 = 'l14'),
29               selected = 'l12'),
30           numericInput(inputId = "ecx_ESSAI221", "Déterminer_la
31             concentration_efficace_X%", value = 50, min = 0,
32             max = 100),
33           selectInput('test_method_ESSAI221', 'Sélectionner_la
34             méthode_de_test_d\'hypothèse',
35             choices=c('Test de Dunnett'='Dunnett
36               '),selected='Dunnett'),
37           actionButton("buttonRunStep221", "Exécuter", icon=
38             icon("check"),width="100px")
39         ),
40         column(8,
41           mainPanel(
42             width=12,
43             DT::dataTableOutput('rawdata'),
44             br(),
45             tableOutput('drc_result'),
46             withSpinner(plotOutput('drc_plot',width="100%",
47               height="850px"),type=4,color='#9e0093'),
48             br(),
49             withSpinner(verbatimTextOutput("test_result")),
50             br()
51           )
52         ),
53         tabPanel(
54           "Télécharger_le_rapport",
55           fixedRow(
56             column(

```

```

44         8,
45         br(),
46         HTML("<font_ face=verdana_ size=5_ color=#9e0093><b>Télé
47             charger_ le_ rapport_ d'analyse</b></font>"),
48         br(), br(), br(),
49         radioButtons("format", "Sélectionner_ le_ format_ du_
50             rapport", c('Word'), inline = TRUE),
51         downloadButton("DownloadReport", "Télécharger_ le_
52             rapport", icon = icon("fas_ fa-download"), style = '
53             background-color:#9e0093;_ color:#000000;_ border-
54             color:#9d9d9d;')
55     )
56 )
57 )
58 )
59 )
60 }

```

Page essai standardisé 222

```

1  library(shiny)
2  library(DT)
3  library(nlme)
4  library(shinycssloaders)
5
6  Essai222Page <- function() {
7    tagList(
8      fluidPage(
9        titlePanel("Earthworm_ Reproduction_ Test_ (Eisenia_ fetida/
10            Eisenia_ andrei)"),
11        fluidRow(
12          column(4,
13            textInput(inputId = "chemical", "Nom_ du_ produit_
14                chimique_ testé", value = "Produit_ chimique"),
15            selectInput("conc_ unit", "Unité_ de_ Concentration",
16                choices = c("g/kg", "mg/kg", "microg/kg", "ng/kg")
17            ),
18            fileInput('datafile_ESSAI222', 'Sélectionner_ un_
19                fichier', accept = c('.csv')),
20            radioButtons('model_ESSAI222', 'Sélectionner_ le_ modè
21                le_ de_ régression',
22                choices = c('Linéaire' = 'lm', 'Non-liné
23                    aire' = 'nls', 'Mixte' = 'lme'),
24                selected = 'lm'),
25            numericInput(inputId = "ecx_ESSAI222", "Déterminer_ la
26                _concentration_ efficace_ X%", value = 50, min = 0,
27                max = 100),
28            radioButtons('test_method_ESSAI222', 'Sélectionner_ la
29                _méthode_ de_ test_ d\'hypothèse',
30                choices = c('ANOVA' = 'anova'), selected
31                = 'anova'),
32            actionButton("buttonRunStep1", "Exécuter", icon =
33                icon("check"), width = "100px")
34          )
35        )
36      )
37    )
38  }

```



```

22  ),
23  column(8,
24  mainPanel(
25    width=12,
26    DT::dataTableOutput('rawdata'),
27    br(),
28    tableOutput('drc_result'),
29    withSpinner(plotOutput('drc_plot', width="100%",
    height="850px"), type=4, color="#9e0093'),
30    br(),
31    withSpinner(verbatimTextOutput("test_result")),
32    br()
33  )
34  ),
35  tabPanel(
36    "Télécharger le rapport",
37    fixedRow(
38      column(
39        8,
40        br(),
41        HTML("<font face=verdana size=5 color=#9e0093><b>Télé
42          charger le rapport d'analyse</b></font>"),
43        br(), br(), br(),
44        radioButtons("format", "Sélectionner le format du
45          rapport", c('Word'), inline = TRUE),
46        downloadButton("DownloadReport", "Télécharger le
47          rapport", icon = icon("fas fa-download"), style = '
48          background-color:#9e0093;color:#000000;border-
49          color:#9d9d9d;')
50      )
51    )
52  )
53  )
54  )
55  )
56  )
57  )
58  )
59  )
60  )
61  )
62  )
63  )
64  )
65  )
66  )
67  )
68  )
69  )
70  )
71  )
72  )
73  )
74  )
75  )
76  )
77  )
78  )
79  )
80  )
81  )
82  )
83  )
84  )
85  )
86  )
87  )
88  )
89  )
90  )
91  )
92  )
93  )
94  )
95  )
96  )
97  )
98  )
99  )
100 )

```

Page essai standardisé 223

```

1  library(shiny)
2  library(DT)
3  library(drc)
4  library(multcomp)
5  library(shinycssloaders)
6
7  Essai223Page <- function() {
8    tagList(
9      fluidPage(
10       titlePanel("Avian Acute Oral Toxicity Test"),
11       fluidRow(
12         column(4,
13           textInput(inputId = "chemical", "Nom du produit
14             chimique testé", value = "Produit chimique"),
15           selectInput("conc_unit", "Unité de Dose", choices = c

```

```

15         ("g/kg", "mg/kg", "microg/kg", "ng/kg")),
16         fileInput('datafile_ESSAI223', 'Sélectionner_un_
            fichier', accept = c('.csv')),
17         radioButtons('model_ESSAI223', 'Sélectionner_le_modè
            le_de_régression',
18                     choices = c('Probit' = 'probit', 'Logit'
                                = 'logit'),
19                     selected = 'probit'),
20         numericInput(inputId = "ecx_ESSAI223", "Déterminer_la
            dose_efficace_X%", value = 50, min = 0, max =
            100),
21         radioButtons('test_method_ESSAI223', 'Sélectionner_la
            méthode_de_test_d\'hypothèse',
22                     choices=c('Test de Dunnett'='Dunnett
            '), selected='Dunnett'),
23         actionButton("buttonRunStep1", "Exécuter", icon=
            icon("check"), width="100px")
24     ),
25     column(8,
26         mainPanel(
27             DT::dataTableOutput('rawdata'),
28             br(),
29             tableOutput('drc_result'),
30             withSpinner(plotOutput('drc_plot', width="100%",
            height="850px"), type="4", color="#9e0093"),
31             br(),
32             withSpinner(verbatimTextOutput("test_result")),
33             br()
34         )
35     ),
36     tabPanel(
37         "Télécharger_le_rapport",
38         fixedRow(
39             column(
40                 8,
41                 br(),
42                 HTML("<font_face=verdana_size=5_color=#9e0093><b>Télé
                    charger_le_rapport_d\'analyse</b></font>"),
43                 br(), br(), br(),
44                 radioButtons("format", "Sélectionner_le_format_du
                    rapport", c('Word'), inline = TRUE),
45                 downloadButton("DownloadReport", "Télécharger_le
                    rapport", icon = icon("fas_fa-download"), style = '
                    background-color:#9e0093;color:#000000;border-
                    color:#9d9d9d;')
46             )
47         )
48     )
49 )
50 ))
51 }

```

```

1 library(shiny)
2 library(DT)
3 library(drc)
4 library(multcomp)
5 library(shinycssloaders)
6
7 Essai225Page <- function() {
8   tagList(
9     fluidPage(
10       titlePanel("Essai_225"),
11       fluidRow(
12         column(4,
13           textInput(inputId = "chemical", "Nom_du_produit_
14             chimique_testé", value = "Produit_chimique"),
15           selectInput("conc_unit", "Unité_de_Concentration",
16             choices = c("g/kg", "mg/kg", "microg/kg", "ng/kg")
17             ),
18           fileInput('datafile_ESSAI225', 'Sélectionner_un_
19             fichier', accept = c('.csv')),
20           radioButtons('model_ESSAI225', 'Sélectionner_le_modè
21             le_de_régression',
22               choices = c('Logistique' = 'logit'),
23               selected = 'logit'),
24           numericInput(inputId = "ecx_ESSAI225", "Déterminer_la
25             concentration_efficace_X%", value = 50, min = 0,
26             max = 100),
27           radioButtons('test_method_ESSAI225', 'Sélectionner_la
28             méthode_de_test_d\'hypothèse',
29             choices = c('Test de Dunnett' = 'Dunnett
30               '), selected = 'Dunnett'),
31           actionButton("buttonRunStep1", "Exécuter", icon =
32             icon("check"), width = "100px")
33         ),
34         column(8,
35           mainPanel(
36             width = 12,
37             DT::dataTableOutput('rawdata'),
38             br(),
39             tableOutput('drc_result'),
40             withSpinner(plotOutput('drc_plot', width = "100%",
41               height = "850px"), type = 4, color = '#9e0093'),
42             br(),
43             withSpinner(verbatimTextOutput("test_result")),
44             br()
45           )
46         ),
47         tabPanel(
48           "Télécharger_le_rapport",
49           fixedRow(
50             column(
51               8,

```

```

41     br(),
42     HTML("<font_ face=verdana_ size=5_ color=#9e0093><b>Télé
         charger_ le_ rapport_ d'analyse</b></font>"),
43     br(), br(), br(),
44     radioButtons("format", "Sélectionner_ le_ format_ du_
         rapport", c('Word'), inline = TRUE),
45     downloadButton("DownloadReport", "Télécharger_ le_
         rapport", icon = icon("fas_ fa-download"), style = '
         background-color: #9e0093;_ color: #000000;_ border-
         color: #9d9d9d;')
46   )
47 )
48 )
49 )
50 ))
51 }

```

Page essai standardisé 226

```

1 library(shiny)
2 library(DT)
3 library(nlme)
4 library(shinycssloaders)
5
6 Essai226Page <- function() {
7   tagList(
8     fluidPage(
9       titlePanel("Essai_226"),
10      fluidRow(
11        column(4,
12          textInput(inputId = "chemical", "Nom_ du_ produit_
              chimique_ testé", value = "Produit_ chimique"),
13          selectInput("conc_unit", "Unité_ de_ Concentration",
              choices = c("g/kg", "mg/kg", "microg/kg", "ng/kg")
              ),
14          fileInput('datafile_ESSAI226', 'Sélectionner_ un_
              fichier', accept = c('.csv')),
15          radioButtons('model_ESSAI226', 'Sélectionner_ le_ modè
              le_ de_ régression',
16                      choices = c('Linéaire' = 'lm', 'Non-liné
              aire' = 'nls', 'Mixte' = 'lme'),
17                      selected = 'lme'),
18          numericInput(inputId = "ecx_ESSAI226", "Déterminer_ la
              _concentration_ efficace_ X%", value = 50, min = 0,
              max = 100),
19          radioButtons('test_method_ESSAI226', 'Sélectionner_ la
              _méthode_ de_ test_ d\'hypothèse',
20          choices = c('ANOVA' = 'anova'), selected
              = 'anova'),
21          actionButton("buttonRunStep1", "Exécuter", icon =
              icon("check"), width = "100px")
22        ),
23        column(8,

```

```

24 #####mainPanel (
25 #####width=12,
26 #####DT::dataTableOutput('rawdata'),
27 #####br(),
28 #####tableOutput('drc_result'),
29 #####withSpinner(plotOutput('drc_plot',width="100%",
    height="850px"),type=4,color="#9e0093"),
30 #####br(),
31 #####withSpinner(verbatimTextOutput("test_result")),
32 #####br()
33 #####)
34 ),
35 tabPanel(
36   "Télécharger le rapport",
37   fixedRow(
38     column(
39       8,
40       br(),
41       HTML("<font face=verdana size=5 color=#9e0093><b>Télé
    charger le rapport d'analyse</b></font>"),
42       br(), br(), br(),
43       radioButtons("format", "Sélectionner le format du
    rapport", c('Word'), inline = TRUE),
44       downloadButton("DownloadReport", "Télécharger le
    rapport", icon = icon("fas fa-download"), style = '
    background-color:#9e0093;color:#000000;border-
    color:#9d9d9d;')
45     )
46   )
47 )
48 )
49 ))
50 }

```

Page essai standardisé 227

```

1 library(shiny)
2 library(DT)
3 library(drc)
4 library(shinycssloaders)
5
6 Essai227Page <- function() {
7   tagList(
8     fluidPage(
9       titlePanel("Essai 227"),
10      fluidRow(
11        column(4,
12          textInput(inputId = "chemical", "Nom du produit
    chimique testé", value = "Produit chimique"),
13          selectInput("conc_unit", "Unité de Concentration",
    choices = c("g/L", "mg/L", "microg/L", "ng/L")),
14          fileInput('datafile_ESSAI227', 'Sélectionner un
    fichier', accept = c('.csv')),

```

```

15         radioButtons('model_ESSAI227', 'Sélectionner le modèle
16             le de régression',
17                 choices = c('log-logistic_2_paramètres'
18                     = 'l12',
19                     'log-logistic_3_paramètres'
20                     = 'l13',
21                     'log-logistic_4_paramètres'
22                     = 'l14'),
23                 selected = 'l12'),
24         numericInput(inputId = "ecx_ESSAI227", "Déterminer la
25             concentration efficace X%", value = 50, min = 0,
26             max = 100),
27         radioButtons('test_method_ESSAI227', 'Sélectionner la
28             méthode de test d\'hypothèse',
29             choices = c('ANOVA' = 'anova'), selected
30             = 'anova'),
31         actionButton("buttonRunStep1", "Exécuter", icon =
32             icon("check"), width = "100px")
33     ),
34     column(8,
35         mainPanel(
36             width = 12,
37             DT::dataTableOutput('rawdata'),
38             br(),
39             tableOutput('drc_result'),
40             withSpinner(plotOutput('drc_plot', width = "100%",
41                 height = "850px"), type = 4, color = '#9e0093'),
42             br(),
43             withSpinner(verbatimTextOutput("test_result")),
44             br()
45         )
46     ),
47     tabPanel(
48         "Télécharger le rapport",
49         fixedRow(
50             column(
51                 8,
52                 br(),
53                 HTML("<font face=verdana size=5 color=#9e0093><b>Télé
54                     charger le rapport d'analyse</b></font>"),
55                 br(), br(), br(),
56                 radioButtons("format", "Sélectionner le format du
57                     rapport", c('Word'), inline = TRUE),
58                 downloadButton("DownloadReport", "Télécharger le
59                     rapport", icon = icon("fas fa-download"), style = '
60                     background-color:#9e0093; color:#000000; border-
61                     color:#9d9d9d;')
62             )
63         )
64     )
65 )
66 )
67 )
68 )
69 )
70 )
71 )
72 )
73 )
74 )
75 )
76 )
77 )
78 )
79 )
80 )
81 )
82 )
83 )
84 )
85 )
86 )
87 )
88 )
89 )
90 )
91 )
92 )
93 )
94 )
95 )
96 )
97 )
98 )
99 )
100 )
101 )
102 )
103 )
104 )
105 )
106 )
107 )
108 )
109 )
110 )
111 )
112 )
113 )
114 )
115 )
116 )
117 )
118 )
119 )
120 )
121 )
122 )
123 )
124 )
125 )
126 )
127 )
128 )
129 )
130 )
131 )
132 )
133 )
134 )
135 )
136 )
137 )
138 )
139 )
140 )
141 )
142 )
143 )
144 )
145 )
146 )
147 )
148 )
149 )
150 )
151 )
152 )
153 )
154 )
155 )
156 )
157 )
158 )
159 )
160 )
161 )
162 )
163 )
164 )
165 )
166 )
167 )
168 )
169 )
170 )
171 )
172 )
173 )
174 )
175 )
176 )
177 )
178 )
179 )
180 )
181 )
182 )
183 )
184 )
185 )
186 )
187 )
188 )
189 )
190 )
191 )
192 )
193 )
194 )
195 )
196 )
197 )
198 )
199 )
200 )
201 )
202 )
203 )
204 )
205 )
206 )
207 )
208 )
209 )
210 )
211 )
212 )
213 )
214 )
215 )
216 )
217 )
218 )
219 )
220 )
221 )
222 )
223 )
224 )
225 )
226 )
227 )
228 )
229 )
230 )
231 )
232 )
233 )
234 )
235 )
236 )
237 )
238 )
239 )
240 )
241 )
242 )
243 )
244 )
245 )
246 )
247 )
248 )
249 )
250 )
251 )
252 )
253 )
254 )
255 )
256 )
257 )
258 )
259 )
260 )
261 )
262 )
263 )
264 )
265 )
266 )
267 )
268 )
269 )
270 )
271 )
272 )
273 )
274 )
275 )
276 )
277 )
278 )
279 )
280 )
281 )
282 )
283 )
284 )
285 )
286 )
287 )
288 )
289 )
290 )
291 )
292 )
293 )
294 )
295 )
296 )
297 )
298 )
299 )
300 )
301 )
302 )
303 )
304 )
305 )
306 )
307 )
308 )
309 )
310 )
311 )
312 )
313 )
314 )
315 )
316 )
317 )
318 )
319 )
320 )
321 )
322 )
323 )
324 )
325 )
326 )
327 )
328 )
329 )
330 )
331 )
332 )
333 )
334 )
335 )
336 )
337 )
338 )
339 )
340 )
341 )
342 )
343 )
344 )
345 )
346 )
347 )
348 )
349 )
350 )
351 )
352 )
353 )
354 )
355 )
356 )
357 )
358 )
359 )
360 )
361 )
362 )
363 )
364 )
365 )
366 )
367 )
368 )
369 )
370 )
371 )
372 )
373 )
374 )
375 )
376 )
377 )
378 )
379 )
380 )
381 )
382 )
383 )
384 )
385 )
386 )
387 )
388 )
389 )
390 )
391 )
392 )
393 )
394 )
395 )
396 )
397 )
398 )
399 )
400 )
401 )
402 )
403 )
404 )
405 )
406 )
407 )
408 )
409 )
410 )
411 )
412 )
413 )
414 )
415 )
416 )
417 )
418 )
419 )
420 )
421 )
422 )
423 )
424 )
425 )
426 )
427 )
428 )
429 )
430 )
431 )
432 )
433 )
434 )
435 )
436 )
437 )
438 )
439 )
440 )
441 )
442 )
443 )
444 )
445 )
446 )
447 )
448 )
449 )
450 )
451 )
452 )
453 )
454 )
455 )
456 )
457 )
458 )
459 )
460 )
461 )
462 )
463 )
464 )
465 )
466 )
467 )
468 )
469 )
470 )
471 )
472 )
473 )
474 )
475 )
476 )
477 )
478 )
479 )
480 )
481 )
482 )
483 )
484 )
485 )
486 )
487 )
488 )
489 )
490 )
491 )
492 )
493 )
494 )
495 )
496 )
497 )
498 )
499 )
500 )
501 )
502 )
503 )
504 )
505 )
506 )
507 )
508 )
509 )
510 )
511 )
512 )
513 )
514 )
515 )
516 )
517 )
518 )
519 )
520 )
521 )
522 )
523 )
524 )
525 )
526 )
527 )
528 )
529 )
530 )
531 )
532 )
533 )
534 )
535 )
536 )
537 )
538 )
539 )
540 )
541 )
542 )
543 )
544 )
545 )
546 )
547 )
548 )
549 )
550 )
551 )
552 )
553 )
554 )
555 )
556 )
557 )
558 )
559 )
560 )
561 )
562 )
563 )
564 )
565 )
566 )
567 )
568 )
569 )
570 )
571 )
572 )
573 )
574 )
575 )
576 )
577 )
578 )
579 )
580 )
581 )
582 )
583 )
584 )
585 )
586 )
587 )
588 )
589 )
590 )
591 )
592 )
593 )
594 )
595 )
596 )
597 )
598 )
599 )
600 )
601 )
602 )
603 )
604 )
605 )
606 )
607 )
608 )
609 )
610 )
611 )
612 )
613 )
614 )
615 )
616 )
617 )
618 )
619 )
620 )
621 )
622 )
623 )
624 )
625 )
626 )
627 )
628 )
629 )
630 )
631 )
632 )
633 )
634 )
635 )
636 )
637 )
638 )
639 )
640 )
641 )
642 )
643 )
644 )
645 )
646 )
647 )
648 )
649 )
650 )
651 )
652 )
653 )
654 )
655 )
656 )
657 )
658 )
659 )
660 )
661 )
662 )
663 )
664 )
665 )
666 )
667 )
668 )
669 )
670 )
671 )
672 )
673 )
674 )
675 )
676 )
677 )
678 )
679 )
680 )
681 )
682 )
683 )
684 )
685 )
686 )
687 )
688 )
689 )
690 )
691 )
692 )
693 )
694 )
695 )
696 )
697 )
698 )
699 )
700 )
701 )
702 )
703 )
704 )
705 )
706 )
707 )
708 )
709 )
710 )
711 )
712 )
713 )
714 )
715 )
716 )
717 )
718 )
719 )
720 )
721 )
722 )
723 )
724 )
725 )
726 )
727 )
728 )
729 )
730 )
731 )
732 )
733 )
734 )
735 )
736 )
737 )
738 )
739 )
740 )
741 )
742 )
743 )
744 )
745 )
746 )
747 )
748 )
749 )
750 )
751 )
752 )
753 )
754 )
755 )
756 )
757 )
758 )
759 )
760 )
761 )
762 )
763 )
764 )
765 )
766 )
767 )
768 )
769 )
770 )
771 )
772 )
773 )
774 )
775 )
776 )
777 )
778 )
779 )
780 )
781 )
782 )
783 )
784 )
785 )
786 )
787 )
788 )
789 )
790 )
791 )
792 )
793 )
794 )
795 )
796 )
797 )
798 )
799 )
800 )
801 )
802 )
803 )
804 )
805 )
806 )
807 )
808 )
809 )
810 )
811 )
812 )
813 )
814 )
815 )
816 )
817 )
818 )
819 )
820 )
821 )
822 )
823 )
824 )
825 )
826 )
827 )
828 )
829 )
830 )
831 )
832 )
833 )
834 )
835 )
836 )
837 )
838 )
839 )
840 )
841 )
842 )
843 )
844 )
845 )
846 )
847 )
848 )
849 )
850 )
851 )
852 )
853 )
854 )
855 )
856 )
857 )
858 )
859 )
860 )
861 )
862 )
863 )
864 )
865 )
866 )
867 )
868 )
869 )
870 )
871 )
872 )
873 )
874 )
875 )
876 )
877 )
878 )
879 )
880 )
881 )
882 )
883 )
884 )
885 )
886 )
887 )
888 )
889 )
890 )
891 )
892 )
893 )
894 )
895 )
896 )
897 )
898 )
899 )
900 )
901 )
902 )
903 )
904 )
905 )
906 )
907 )
908 )
909 )
910 )
911 )
912 )
913 )
914 )
915 )
916 )
917 )
918 )
919 )
920 )
921 )
922 )
923 )
924 )
925 )
926 )
927 )
928 )
929 )
930 )
931 )
932 )
933 )
934 )
935 )
936 )
937 )
938 )
939 )
940 )
941 )
942 )
943 )
944 )
945 )
946 )
947 )
948 )
949 )
950 )
951 )
952 )
953 )
954 )
955 )
956 )
957 )
958 )
959 )
960 )
961 )
962 )
963 )
964 )
965 )
966 )
967 )
968 )
969 )
970 )
971 )
972 )
973 )
974 )
975 )
976 )
977 )
978 )
979 )
980 )
981 )
982 )
983 )
984 )
985 )
986 )
987 )
988 )
989 )
990 )
991 )
992 )
993 )
994 )
995 )
996 )
997 )
998 )
999 )
1000 )

```

Page essai standardisé 232

```

1 library(shiny)
2 library(DT)
3 library(nlme)
4 library(shinycssloaders)
5
6 Essai232Page <- function() {
7   tagList(
8     fluidPage(
9       titlePanel("Collembolan_Reproduction_Test_in_Soil"),
10      fluidRow(
11        column(4,
12          textInput(inputId = "chemical", "Nom_du_produit_
13            chimique_testé", value = "Produit_chimique"),
14          selectInput("conc_unit", "Unité_de_Concentration",
15            choices = c("g/kg", "mg/kg", "microg/kg", "ng/kg")
16          ),
17          fileInput('datafile_ESSAI232', 'Sélectionner_un_
18            fichier', accept = c('.csv')),
19          radioButtons('model_ESSAI232', 'Sélectionner_le_modè
20            le_de_régression',
21              choices = c('Linéaire' = 'lm', 'Non-liné
22                aire' = 'nls', 'Mixte' = 'lme'),
23              selected = 'lme'),
24          numericInput(inputId = "ecx_ESSAI232", "Déterminer_la
25            concentration_efficace_X%", value = 50, min = 0,
26            max = 100),
27          radioButtons('test_method_ESSAI232', 'Sélectionner_la
28            méthode_de_test_d\'hypothèse',
29            choices=c('ANOVA'='anova'), selected
30              ='anova'),
31          actionButton("buttonRunStep1", "Exécuter", icon=
32            icon("check"), width="100px")
33        ),
34        column(8,
35          mainPanel(
36            width=12,
37            DT::dataTableOutput('rawdata'),
38            br(),
39            tableOutput('drc_result'),
40            withSpinner(plotOutput('drc_plot', width="100%",
41              height="850px"), type="4", color="#9e0093'),
42            br(),
43            withSpinner(verbatimTextOutput("test_result")),
44            br()
45          )
46        ),
47      tabPanel(
48        "Télécharger_le_rapport",
49        fixedRow(

```

```

38     column(
39       8,
40       br(),
41       HTML("<font_ face=verdana_ size=5_ color=#9e0093><b>Télé
42         charger_ le_ rapport_ d'analyse</b></font>"),
43       br(), br(), br(),
44       radioButtons("format", "Sélectionner_ le_ format_ du_
         rapport", c('Word'), inline = TRUE),
45       downloadButton("DownloadReport", "Télécharger_ le_
         rapport", icon = icon("fas_ fa-download"), style = '
         background-color:#9e0093;_ color:#000000;_ border-
         color:#9d9d9d;')
46     )
47   )
48 )
49 ))
50 }

```

Page essai standardisé 239

```

1  library(shiny)
2  library(DT)
3  library(drc)
4  library(multcomp)
5  library(shinycssloaders)
6
7  Essai239Page <- function() {
8    tagList(
9      makePage(
10        "Water-Sediment_Myriophyllum_Spicatum_Toxicity_Test",
11        "(Essai_239)",
12        fluidRow(
13          column(4,
14            textInput(inputId = "chemical", "Nom_ du_ produit_
15              chimique_ testé", value = "Produit_ chimique"),
16            selectInput("conc_unit239", "Unité_ de_ Dose", choices
17              = c("g/L", "mg/L", "microg/L", "ng/L")),
18            fileInput('datafile_ESSAI239', '', accept = c('.csv')
19              ),
20            tags$br(),
21            selectInput('model_ESSAI239', 'Sélectionner_ le_ modèle
22              de_ régression',
23              choices = c('Probit' = 'probit', 'Logit'
24                = 'logit'),
25              selected = 'probit'),
26            numericInput(inputId = "ecx_ESSAI239", "Déterminer_ la
27              dose_ efficace_ X%", value = 50, min = 0, max =
28                100),
29            selectInput('test_method_ESSAI239', 'Sélectionner_ la_
30              méthode_ de_ test_ d\'hypothèse',
31              choices = c('Test de Dunnett' = 'Dunnett
32                '), selected = 'Dunnett'),

```



```

24 #####actionButton("buttonRunStep239", "Exécuter", icon=
      icon("check"), width="100px")
25 #####),
26 #####column(8,
27 #####mainPanel(
28 #####width="12,
29 #####DT::dataTableOutput('rawdata'),
30 #####br(),
31 #####tableOutput('drc_result'),
32 #####withSpinner(plotOutput('drc_plot', width="100%",
      height="850px"), type="4", color="#9e0093'),
33 #####br(),
34 #####withSpinner(verbatimTextOutput("test_result")),
35 #####br()
36 #####)
37 #####),
38 #####tabPanel(
39 #####  "Télécharger le rapport",
40 #####  fixedRow(
41 #####    column(
42 #####      8,
43 #####      br(),
44 #####      HTML("<font face=verdana size=5 color=#9e0093><b>Télé
      charger le rapport d'analyse</b></font>"),
45 #####      br(), br(), br(),
46 #####      radioButtons("format", "Sélectionner le format du
      rapport", c('Word'), inline = TRUE),
47 #####      downloadButton("DownloadReport", "Télécharger le
      rapport", icon = icon("fas fa-download"), style = '
      background-color:#9e0093; color:#000000; border-
      color:#9d9d9d;')
48 #####    )
49 #####  )
50 #####)
51 #####)
52 #####))
53 }

```

Page essai standardisé 245

```

1 library(shiny)
2 library(DT)
3 library(drc)
4 library(multcomp)
5 library(survival)
6 library(shinycssloaders)
7
8 Essai245Page <- function() {
9   tagList(
10     fluidPage(
11       titlePanel("Honey Bee (Apis Mellifera L.), Chronic Oral
          Toxicity Test (10-Day Feeding)"),
12       fluidRow(

```

```

13     column(4,
14         textInput(inputId = "chemical", "Nom_du_produit_
           chimique_testé", value = "Produit_chimique"),
15         selectInput("conc_unit", "Unité_de_Dose", choices = c
           ("g/L", "mg/L", "microg/L", "ng/L")),
16         fileInput('datafile_ESSAI245', 'Sélectionner_un_
           fichier', accept = c('.csv')),
17         radioButtons('model_ESSAI245', 'Sélectionner_le_modè
           le_de_régression',
18             choices = c('Cox' = 'coxph'),
19             selected = 'coxph'),
20         numericInput(inputId = "ecx_ESSAI245", "Déterminer_la
           dose_efficace_X%", value = 50, min = 0, max =
           100),
21         radioButtons('test_method_ESSAI245', 'Sélectionner_la
           méthode_de_test_d\'hypothèse',
22             choices=c('Test de Dunnett'='Dunnett
           '),selected='Dunnett'),
23         actionButton("buttonRunStep1","Exécuter",icon=
           icon("check"),width="100px")
24     ),
25     column(8,
26         mainPanel(
27             width=12,
28             DT::dataTableOutput('rawdata'),
29             br(),
30             tableOutput('drc_result'),
31             withSpinner(plotOutput('drc_plot',width="100%",
           height="850px"),type=4,color='#9e0093'),
32             br(),
33             withSpinner(verbatimTextOutput("test_result")),
34             br()
35         )
36     ),
37     tabPanel(
38         "Télécharger_le_rapport",
39         fixedRow(
40             column(
41                 8,
42                 br(),
43                 HTML("<font_face=verdana_size=5_color=#9e0093><b>Télé
           charger_le_rapport_d'analyse</b></font>"),
44                 br(), br(), br(),
45                 radioButtons("format", "Sélectionner_le_format_du
           rapport", c('Word'), inline = TRUE),
46                 downloadButton("DownloadReport", "Télécharger_le
           rapport", icon = icon("fas_fa-download"), style = '
           background-color:#9e0093;color:#000000;border-
           color:#9d9d9d;')
47             )
48         )
49     )

```

```

50     )
51   ))
52 }

```

Page essai standardisé 246

```

1  library(shiny)
2  library(DT)
3  library(drc)
4  library(shinycssloaders)
5
6  Essai246Page <- function() {
7    tagList(
8      fluidPage(
9        titlePanel("Bumblebee, Acute Contact Toxicity Test"),
10       fluidRow(
11         column(4,
12           textInput(inputId = "chemical", "Nom du produit chimique testé", value = "Produit chimique"),
13           selectInput("conc_unit", "Unité de Concentration",
14             choices = c("g/L", "mg/L", "microg/L", "ng/L")),
15           fileInput('datafile_ESSAI246', 'Sélectionner un fichier', accept = c('.csv')),
16           radioButtons('model_ESSAI246', 'Sélectionner le modèle de régression',
17             choices = c('Probit' = 'probit', 'Logit' = 'logit'),
18             selected = 'probit'),
19           numericInput(inputId = "ecx_ESSAI246", "Déterminer la concentration efficace X%", value = 50, min = 0,
20             max = 100),
21           radioButtons('test_method_ESSAI246', 'Sélectionner la méthode de test d\'hypothèse',
22             choices = c('Test de Dunnett' = 'Dunnett'),
23             selected = 'Dunnett'),
24           actionButton("buttonRunStep1", "Exécuter", icon = icon("check"), width = "100px")
25         ),
26         column(8,
27           mainPanel(
28             width = 12,
29             DT::dataTableOutput('rawdata'),
30             br(),
31             tableOutput('drc_result'),
32             withSpinner(plotOutput('drc_plot', width = "100%", height = "850px"), type = 4, color = '#9e0093'),
33             br(),
34             withSpinner(verbatimTextOutput("test_result")),
35             br()
36           )
37         ),
38       tabPanel(
39         "Télécharger le rapport",

```

```

37     fixedRow(
38       column(
39         8,
40         br(),
41         HTML("<font face=verdana size=5 color=#9e0093><b>Télé
42             charger le rapport d'analyse</b></font>"),
43         br(), br(), br(),
44         radioButtons("format", "Sélectionner le format du
45             rapport", c('Word'), inline = TRUE),
46         downloadButton("DownloadReport", "Télécharger le
47             rapport", icon = icon("fas fa-download"), style = '
48             background-color:#9e0093;color:#000000;border-
49             color:#9d9d9d;')
50       )
51     )
52   )
53 )
54 ))
55 }

```

Page essai standardisé 247

```

1 library(shiny)
2 library(DT)
3 library(drc)
4 library(shinycssloaders)
5
6 Essai247Page <- function() {
7   tagList(
8     fluidPage(
9       titlePanel("Bumblebee, Acute Oral Toxicity Test"),
10      fluidRow(
11        column(4,
12          textInput(inputId = "chemical", "Nom du produit
13              chimique testé", value = "Produit chimique"),
14          selectInput("conc_unit", "Unité de Concentration",
15              choices = c("g/L", "mg/L", "microg/L", "ng/L")),
16          fileInput('datafile_ESSAI247', 'Sélectionner un
17              fichier', accept = c('.csv')),
18          radioButtons('model_ESSAI247', 'Sélectionner le modè
19              le de régression',
20              choices = c('Probit' = 'probit', 'Logit'
21              = 'logit'),
22              selected = 'probit'),
23          numericInput(inputId = "ecx_ESSAI247", "Déterminer la
24              concentration efficace X%", value = 50, min = 0,
25              max = 100),
26          radioButtons('test_method_ESSAI247', 'Sélectionner la
27              méthode de test d\'hypothèse',
28              choices = c('Test de Dunnett' = 'Dunnett
29              '), selected = 'Dunnett'),
30          actionButton("buttonRunStep1", "Exécuter", icon =
31              icon("check"), width = "100px")

```

```

22  ),
23  column(8,
24    mainPanel(
25      width=12,
26      DT::dataTableOutput('rawdata'),
27      br(),
28      tableOutput('drc_result'),
29      withSpinner(plotOutput('drc_plot', width="100%",
    height="850px"), type=4, color="#9e0093'),
30      br(),
31      withSpinner(verbatimTextOutput("test_result")),
32      br()
33    )
34  ),
35  tabPanel(
36    "Télécharger le rapport",
37    fixedRow(
38      column(
39        8,
40        br(),
41        HTML("<font face=verdana size=5 color=#9e0093><b>Télé
    charger le rapport d'analyse</b></font>"),
42        br(), br(), br(),
43        radioButtons("format", "Sélectionner le format du
    rapport", c('Word'), inline = TRUE),
44        downloadButton("DownloadReport", "Télécharger le
    rapport", icon = icon("fas fa-download"), style = '
    background-color:#9e0093;color:#000000;border-
    color:#9d9d9d;')
45      )
46    )
47  )
48 )
49 ))
50 }

```

Page essai standardisé 213_214

```

1  library(shiny)
2  library(DT)
3  library(drc)
4  library(multcomp)
5  library(shinycssloaders)
6
7  Essai213214Page <- function() {
8    tagList(
9      fluidPage(
10        titlePanel("Honeybees, Acute Oral/Contact"),
11        fluidRow(
12          column(4,
13            textInput(inputId = "chemical", "Nom du produit
    chimique testé", value = "Produit chimique"),
14            selectInput("conc_unit", "Unité de Dose", choices = c

```

```

15         ("g/L", "mg/L", "microg/L", "ng/L")),
16         fileInput('datafile_ESSAI213214', 'Sélectionner un
            fichier', accept = c('.csv')),
17         radioButtons('model_ESSAI213214', 'Sélectionner le
            modèle de régression',
18                     choices = c('Probit' = 'probit', 'Logit'
                                   = 'logit'),
19                     selected = 'probit'),
20         numericInput(inputId = "ecx_ESSAI213214", "Déterminer
            la dose efficace X%", value = 50, min = 0, max =
            100),
21         radioButtons('test_method_ESSAI213214', 'Sélectionner
            la méthode de test\ 'hypothèse',
22                     choices=c('Test de Dunnett'='Dunnett
                                   '), selected='Dunnett'),
23         actionButton("buttonRunStep1", "Exécuter", icon=
            icon("check"), width="100px")
24     ),
25     column(8,
26         mainPanel(
27             DT::dataTableOutput('rawdata'),
28             br(),
29             tableOutput('drc_result'),
30             withSpinner(plotOutput('drc_plot', width="100%",
            height="850px"), type="4", color="#9e0093"),
31             br(),
32             withSpinner(verbatimTextOutput("test_result")),
33             br()
34         )
35     ),
36     tabPanel(
37         "Télécharger le rapport",
38         fixedRow(
39             column(
40                 8,
41                 br(),
42                 HTML("<font face=verdana size=5 color=#9e0093><b>Télé
                    charger le rapport d'analyse</b></font>"),
43                 br(), br(), br(),
44                 radioButtons("format", "Sélectionner le format du
                    rapport", c('Word'), inline = TRUE),
45                 downloadButton("DownloadReport", "Télécharger le
                    rapport", icon = icon("fas fa-download"), style = '
                    background-color:#9e0093; color:#000000; border-
                    color:#9d9d9d;')
46             )
47         )
48     )
49 )
50 ))
51 }

```

```

1 library(shiny)
2 library(DT)
3 library(drc)
4 library(multcomp)
5 library(shinycssloaders)
6
7 Essai246247Page <- function() {
8   tagList(
9     fluidPage(
10       titlePanel("Bumblebee, Acute Contact / Oral Toxicity"),
11       fluidRow(
12         column(4,
13           textInput(inputId = "chemical", "Nom du produit chimique testé", value = "Produit chimique"),
14           selectInput("conc_unit", "Unité de Dose", choices = c("g/L", "mg/L", "microg/L", "ng/L")),
15           fileInput('datafile_ESSAI246247', 'Sélectionner un fichier', accept = c('.csv')),
16           radioButtons('model_ESSAI246247', 'Sélectionner le modèle de régression',
17             choices = c('Probit' = 'probit', 'Logit' = 'logit'),
18             selected = 'probit'),
19           numericInput(inputId = "ecx_ESSAI246247", "Déterminer la dose efficace X%", value = 50, min = 0, max = 100),
20           radioButtons('test_method_ESSAI246247', 'Sélectionner la méthode de test de hypothèse',
21             choices = c('Test de Dunnett' = 'Dunnett'),
22             selected = 'Dunnett'),
23           actionButton("buttonRunStep1", "Exécuter", icon = icon("check"), width = "100px")
24         ),
25         column(8,
26           mainPanel(
27             width = 12,
28             DT::dataTableOutput('rawdata'),
29             br(),
30             tableOutput('drc_result'),
31             withSpinner(plotOutput('drc_plot', width = "100%", height = "850px"), type = 4, color = '#9e0093'),
32             br(),
33             withSpinner(verbatimTextOutput("test_result")),
34             br()
35           )
36         ),
37         tabPanel(
38           "Télécharger le rapport",
39           fixedRow(
40             column(

```

```

41         br(),
42         HTML("<font_ face=verdana_ size=5_ color=#9e0093><b>Télé
43             charger_le_rapport_d'analyse</b></font>"),
44         br(), br(), br(),
45         radioButtons("format", "Sélectionner_le_format_du_
46             rapport", c('Word'), inline = TRUE),
47         downloadButton("DownloadReport", "Télécharger_le_
48             rapport", icon = icon("fas_fa-download"), style = '
49             background-color:#9e0093;_color:#000000;_border-
50             color:#9d9d9d;')
51     }

```

Annexe A : Inventaire des tests

Test	Forces	Limites	Critères de Choix
Dunnett's	Contrôle le taux d'erreur de Type I;utile avec un groupe de contrôle	Exige normalité et vari-ances équivalentes	Plusieurs traitements comparés à un groupe de contrôle;données normalement distribuées avec homogénéité des variances
Welch's	Ne nécessite pas l'homogénéité des variances	Moins puissant que l'ANOVA avec variances égales	Deux groupes avec vari-ances inégales pour com-paraison de moyennes
Brown-Forsythe	Robuste aux écarts de normalité;teste l'égalité des médianes	Moins puissant dans cer-tains cas	Comparaison de moyennes quand l'homogénéité des vari-ances est violée

Table 16: Tests Paramétriques: Forces, Limites et Critères de Choix

Test	Forces	Limites	Critères de Choix
Cochran-Armitage Trend	Détecte une tendance dans des proportions sur niveaux ordinaux	Conçu uniquement pour données catégorielles ordinales	Données catégorielles ordinales avec tendance suspectée
Jonckheere-Terpstra	Teste des alternatives ordonnées sans supposer une distribution spécifique	L'ordre des groupes doit être spécifié à l'avance	Détecter des tendances ordonnées parmi les médianes de groupes indépendants
Mantel-Haenszel	Adapte pour l'analyse de données stratifiées; contrôle pour les variables de confusion	Limité aux tableaux de contingence 2x2	Test de l'association entre deux variables binaires en contrôlant pour variables de confusion

Table 17: Tests Non-Paramétriques: Forces, Limites et Critères de Choix

Annexe B : Rappels Théoriques

Définitions

Définition du percentile et du quantile

Pour tout nombre x dans l'intervalle $0 \leq x \leq 100$, le x -ième percentile d'un ensemble de données fini est défini comme un nombre qui est supérieur ou égal à au moins $x\%$ des données et est inférieur ou égal à au moins $(100 - x)\%$ des données.

Règle des valeurs aberrantes de Tukey (nommée d'après John W. Tukey, 1977)

Toute réponse supérieure à 1.5 fois l'intervalle interquartile (IQR) au-dessus du troisième quartile ou en dessous du premier quartile est considérée comme une valeur aberrante selon la règle de Tukey. Cette règle est motivée par les propriétés d'une distribution normale.

D Distributions de Probabilité

La fonction de distribution pour une variable aléatoire X est une fonction réelle non négative F définie par :

$$F(x) = P(X \leq x) \quad \text{pour tous les nombres réels } x$$

où $P(E)$ est la probabilité que E se produise.

References

- [1] European Union, *Regulation (CE) n°1107/2009*, Official Journal of the European Union, 2009.
- [2] Organisation for Economic Co-operation and Development, *OECD Guidelines for the Testing of Chemicals*, Document Guide 54, 2006.
[https://www.google.com/url?sa=t&source=web&rct=j&opi=89978449&url=https://one.oecd.org/document/ENV/JM/MONO\(2006\)18/en/pdf&ved=2ahUKEwi04KCylvyHAXVCRKQEHUpSLTEQFnoECBUQAQ&usg=AOvVaw1IUob7mCm0Ish8Xwl8D85V](https://www.google.com/url?sa=t&source=web&rct=j&opi=89978449&url=https://one.oecd.org/document/ENV/JM/MONO(2006)18/en/pdf&ved=2ahUKEwi04KCylvyHAXVCRKQEHUpSLTEQFnoECBUQAQ&usg=AOvVaw1IUob7mCm0Ish8Xwl8D85V).
- [3] John W. Green, Timothy A. Springer, and Henrik Holbech, *Statistical Analysis of Ecotoxicity Studies*, Publisher Name, 2018.
- [4] Moore, D. S., McCabe, G. P., & Craig, B. A., *Introduction to the Practice of Statistics*, W.H. Freeman, 2016.
- [5] Johnson, R. A., & Wichern, D. W., *Applied Multivariate Statistical Analysis (6th ed.)*, Pearson, 2007.
- [6] Bates, D. M., & Watts, D. G. (1988). *Nonlinear Regression Analysis and Its Applications*. Wiley.
- [7] Burnham, Kenneth P., et David R. Anderson. 2004. *Multimodel Inference: Understanding AIC and BIC in Model Selection*. Sociological Methods & Research 33 (2): 261-304. <https://doi.org/10.1177/0049124104268644>.
- [8] Burnham, Kenneth P., David R. Anderson, et Kathryn P. Huyvaert. 2011. « *AIC Model Selection and Multimodel Inference in Behavioral Ecology: Some Background, Observations, and Comparisons* ». Behavioral Ecology and Sociobiology <https://doi.org/10.1007/s00265-010-1029-6>.
- [9] Philippe, A. *Cours sur le Bootstrap et les Méthodes Bayésiennes*. Université Nantes, Master Ingénierie Statistique, 2023-2024.
- [10] B. Efron. (1979). *Bootstrap Methods: Another Look at the Jackknife*. Annals of Statistics, 7(1), 1-26. <https://doi.org/10.1214/aos/1176344552>
- [11] *Bayesian Data Analysis* Third edition (with errors fixed as of 15 February 2021) by Andrew Gelman, John Carlin, Hal Stern, Donald Rubin, David Dunson, and Aki Vehtari
- [12] *Introducing Monte Carlo Methods with R*. Springer.
- [13] *Bayesian Theory*. Wiley.
- [14] *Choix des a priori en inférence bayésienne*: Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., & Rubin, D. B. (2013). **Bayesian Data Analysis** (3rd ed.). CRC Press.
- [15] *A priori informatifs*: O'Hagan, A., & Forster, J. (2004). Kendall's Advanced Theory of Statistics, Volume 2B: Bayesian Inference (2nd ed.). Arnold.
- [16] *A priori non informatifs*: Bernardo, J. M., & Smith, A. F. M. (2000). Bayesian Theory. Wiley.

- [17] *A priori partiellement informatifs*: Gelman, A. (2006). Prior distributions for variance parameters in hierarchical models. *Bayesian Analysis*, 1(3), 515-533.
- [18] *A priori conjugués*: Raiffa, H., & Schlaifer, R. (1961). *Applied Statistical Decision Theory*. Wiley.
- [19] *A priori empiriques*: Casella, G., & Berger, R. L. (2002). *Statistical Inference* (2nd ed.). Duxbury.
- [20] *Application à l'écotoxicologie*: Clark, J. S. (2005). Why environmental scientists are becoming Bayesians. *Ecology Letters*, 8(1), 2-14.
- [21] *Hazardous and Radioactive Substances in Danish Marine Waters. Status and Temporal Trends*. January 2009 Publisher: National Environmental Research Institute
- [22] Stamm, A. *Cours sur la Théorie des Tests de Permutation*. Université Nantes, Master Ingénierie Statistique, 2023-2024.
- [23] Good, Phillip I. (2005). *Permutation, Parametric and Bootstrap Tests of Hypotheses*, 3rd ed., Springer. ISBN 0-387-98898-X.
- [24] Barnett, V., & Lewis, T. (1994). *Outliers in Statistical Data* (3rd ed.). John Wiley & Sons.
- [25] Grubbs, F. E. (1969). Procedures for Detecting Outlying Observations in Samples. *Technometrics*, 11(1), 1-21.
- [26] Zuur, A. F., Ieno, E. N., & Elphick, C. S. (2010). A protocol for data exploration to avoid common statistical problems. *Methods in Ecology and Evolution*, 1(1), 3-14.
- [27] *OCDE201 : Algues d'eau douce et cyanobactéries*.
https://www.oecd-ilibrary.org/fr/test-no-201-alga-growth-inhibition-test_5lmqcr2k7s5h.pdf?itemId=%2Fcontent%2Fpublication%2F9789264069923-en&mimeType=pdf
- [28] *OCDE202.Daphnia sp., Essai d'immobilisation immédiate*.
https://www.oecd-ilibrary.org/fr/test-no-202-daphnia-sp-acute-immobilisation-test_5lmqcr2k7s28.pdf?itemId=%2Fcontent%2Fpublication%2F9789264069947-en&mimeType=pdf
- [29] *OCDE203.Poisson tox aiguë*.
https://www.oecd-ilibrary.org/fr/test-no-203-fish-acute-toxicity-test_5lmqcr2k7rzt.pdf?itemId=%2Fcontent%2Fpublication%2F9789264069961-en&mimeType=pdf
- [30] *OCDE205.Oiseaux, Essai de Toxicité Liée au Régime*
https://www.oecd-ilibrary.org/fr/test-no-205-avian-dietary-toxicity-test_5lmqcr2k7rs4.pdf?itemId=%2Fcontent%2Fpublication%2F9789264070004-en&mimeType=pdf
- [31] *OCDE206.Oiseaux reproduction*.
https://www.oecd-ilibrary.org/fr/test-no-206-avian-reproduction-test_5lmqcr2k7rq8.pdf?itemId=%2Fcontent%2Fpublication%2F9789264070028-en&mimeType=pdf

- [32] *OCDE210. Poisson, essai de toxicité aux premiers stades de la vie.*
https://www.oecd-ilibrary.org/fr/test-no-210-fish-early-life-stage-toxicity-test_5k43bdxrtps5.pdf?itemId=%2Fcontent%2Fpublication%2F9789264203785-en&imeType=pdf
- [33] *OCDE211. Daphnia magna, essai de reproduction.*
https://www.oecd-ilibrary.org/fr/test-no-211-daphnia-magna-reproduction-test_5k91hspv1jmr.pdf?itemId=%2Fcontent%2Fpublication%2F9789264185203-en&imeType=pdf
- [34] *OCDE213. Abeille domestique, essai de toxicité aiguë par voie orale.*
https://www.oecd-ilibrary.org/fr/test-no-213-honeybees-acute-oral-toxicity-test_5lmqcr2k7r34.pdf?itemId=%2Fcontent%2Fpublication%2F9789264070165-en&imeType=pdf
- [35] *OCDE214. Abeille domestique, essai de toxicité aiguë par contact.*
https://www.oecd-ilibrary.org/fr/test-no-214-honeybees-acute-contact-toxicity-test_5lmqcr2k7r0n.pdf?itemId=%2Fcontent%2Fpublication%2F9789264070189-en&imeType=pdf
- [36] *OCDE216. Micro-organismes du sol essai de transformation de l'azote.*
https://www.oecd-ilibrary.org/fr/test-no-216-soil-microorganisms-nitrogen-transformation-test_5lmqcr2k7qtk.pdf?itemId=%2Fcontent%2Fpublication%2F9789264070226-en&imeType=pdf
- [37] *OCDE218. Essai de toxicité sur les chironomes dans un système eau-sédiment chargé*
https://www.oecd-ilibrary.org/fr/test-no-218-sediment-water-chironomid-toxicity-test_5lmng4d8td9w.pdf?itemId=%2Fcontent%2Fpublication%2F9789264070264-en&imeType=pdf
- [38] *OCDE221. Lemna sp. Essais d'inhibition de la croissance.*
https://www.oecd-ilibrary.org/fr/test-no-221-lemna-sp-growth-inhibition-test_5l9njf5w5bhg.pdf?itemId=%2Fcontent%2Fpublication%2F9789264016194-en&imeType=pdf
- [39] *OCDE222. Reproduction chez le ver de terre.*
<https://www.oecd-ilibrary.org/fr/test-no-222-earthworm-reproduction-test-eisenia-f5j1sh07trg0r.pdf?itemId=%2Fcontent%2Fpublication%2F9789264264496-en&imeType=pdf>
- [40] *OCDE223. Oiseaux, toxicité orale aiguë.*
https://www.oecd-ilibrary.org/fr/test-no-223-avian-acute-oral-toxicity-test_5j1sh05ks9g2.pdf?itemId=%2Fcontent%2Fpublication%2F9789264264519-en&imeType=pdf
- [41] *OCDE226. Reproduction d'un acarien prédateur.*
<https://www.oecd-ilibrary.org/fr/test-no-226-predatory-mite-hypoaspis-geolaelaps-a>

5j1sh04vc8mw.pdf?itemId=%2Fcontent%2Fpublication%2F9789264264557-en& mimeType=pdf

- [42] *OCDE232.Reproduction de collemboles dans le sol.*
<https://www.oecd-ilibrary.org/fr/test-no-232-collembolan-reproduction-test-in-soil-5j1sh03vrzjj.pdf?itemId=%2Fcontent%2Fpublication%2F9789264264601-en&mimeType=pdf>
- [43] *OCDE245.Abeille domestique (Apis mellifera L.), Essai de Toxicité Chronique par Voie Orale (10 jours).*
<https://www.oecd-ilibrary.org/fr/test-no-245-honey-bee-apis-mellifera-l-chronic-on-5jfmclm8xmwg.pdf?itemId=%2Fcontent%2Fpublication%2F9789264284081-en&mimeType=pdf>
- [44] *OCDE246.Bourdon, Essai de Toxicité Aiguë par Contact.*
<https://www.oecd-ilibrary.org/fr/test-no-246-bumblebee-acute-contact-toxicity-test-5jfmclhgtbzn.pdf?itemId=%2Fcontent%2Fpublication%2F9789264284104-en&mimeType=pdf>
- [45] *OCDE247.Bourdon, Essai de Toxicité Aiguë par Voie Orale.*
<https://www.oecd-ilibrary.org/fr/test-no-247-bumblebee-acute-oral-toxicity-test-5jfmcldkg1jc.pdf?itemId=%2Fcontent%2Fpublication%2F9789264284128-en&mimeType=pdf>
- [46] *OCDE208.Plantes.*
<https://www.oecd-ilibrary.org/fr/test-no-208-terrestrial-plant-test-seedling-emerg-5lmqcr2k7rjl.pdf?itemId=%2Fcontent%2Fpublication%2F9789264070066-en&mimeType=pdf>
- [47] *OCDE227.Plantes terrestres vigueur végétative.*
<https://www.oecd-ilibrary.org/fr/test-no-227-terrestrial-plant-test-vegetative-vig-5l9gc7rvtcf5.pdf?itemId=%2Fcontent%2Fpublication%2F9789264067295-en&mimeType=pdf>
- [48] *OCDE239.Plante aquatique système eau-sédiment.*
<https://www.oecd-ilibrary.org/fr/test-no-239-water-sediment-myriophyllum-spicatum-5jxwv1xxcxf1.pdf?itemId=%2Fcontent%2Fpublication%2F9789264224155-en&mimeType=pdf>
- [49] *ESCORT.Escort.*
https://www.pan-europe.info/sites/pan-europe.info/files/public/resources/other/_ESCORT%202%20_non-target%20arthropods.pdf
- [50] Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., & Rubin, D. B., *Bayesian Data Analysis* (3rd ed.), CRC Press, 2013.
- [51] O'Hagan, A., & Forster, J., *Kendall's Advanced Theory of Statistics, Volume 2B: Bayesian Inference* (2nd ed.), Arnold, 2004.
- [52] Bernardo, J. M., & Smith, A. F. M., *Bayesian Theory*, Wiley, 2000.

- [53] *Nonlinear Regression Analysis and Its Applications* Douglas M. Bates, Donald G. Watts
First published:26 August 1988
- [54] *Nonlinear Regression Analysis* Hsin-Hsiung Huang and Qing He <https://arxiv.org/pdf/2402.05342>
- [55] Gelman, A., *Prior distributions for variance parameters in hierarchical models*, Bayesian Analysis, 1(3), 515-533, 2006.
- [56] Raiffa, H., & Schlaifer, R., *Applied Statistical Decision Theory*, Wiley, 1961.
- [57] Casella, G., & Berger, R. L., *Statistical Inference* (2nd ed.), Duxbury, 2002.
- [58] Clark, J. S., *Why environmental scientists are becoming Bayesians*, Ecology Letters, 8(1), 2-14, 2005.