Zürcher Hochschule
für Angewandte Wissenschaften
Life Sciences und
Facility Management
IUNR Institut für Umwelt und
Natürliche Ressourcen

ZÜRCHER HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN
DEPARTEMENT LIFE SCIENCES UND FACILITY MANAGEMENT
INSTITUT FÜR UMWELT UND NATÜRLICHE RESSOURCEN

Besuchende im Wildnispark Zürich, ZH (S. Woolsey, 2021)

Analyse von Einflussfaktoren auf die Fussgängerzahlen im Wildnispark Zürich

Masterstudium in Umwelt und Natürliche Ressourcen
Fallstudie im Modul Research Methods

M. Grether, M. Maric, R. Winter & S. Woolsey
Fachkorrektur: A. Hochreutener & R. Rupf
Abgabedatum: 21.12.2021

1 Einleitung

Natur- und siedlungsnahe Erholungsgebiete gewinnen im Zuge der wachsenden Urbanisierung an Bedeutung (Ketterer Bonnelame & Siegrist, 2018). Um negative Auswirkungen auf die Lebensräume von Wildtieren und -pflanzen zu minimieren, steigt der Bedarf zur Erfassung und Lenkung von Besucheraufkommen (Cessford & Muhar, 2003; Rupf, 2014; McGinlay et al., 2020).

Die Covid19-Pandemie hat den Nutzungsdruck auf siedlungsnahe Grünräume zusätzlich verstärkt (Derks et al., 2020; Geng et al., 2021). Der Wildnispark Zürich (WPZ) vor den Toren der Stadt Zürich verzeichnete im Jahr 2020 im Vergleich mit dem Vorjahr ein Besucherplus von 70 Prozent. Während des Lockdowns im Frühling 2020 bewegten sich bis zu dreimal so viele Besucher*innen im Sihlwald (Stiftung Wildnispark Zürich, 2021).

Für die Gestaltung von Lenkungsmassnahmen gilt es, neben der Erfassung von Besuchszahlen, Faktoren zu identifizieren, welche die Besucherdynamik beeinflussen. Dabei sind Temperatur, Regen, Wochentage und Ferien als relevante Determinanten von Besucheraufkommen in Pärken erforscht (Hewer et al., 2016; Millhäusler et al., 2016). Als neue Faktoren kommen die Covid19-Pandemie und damit verbundene Lockdowns hinzu. Die vorliegende Fallstudie untersucht mittels multivariater Analyse den Einfluss von Temperatur, Regen, Sonnenscheindauer, Wochentage, Ferien und Pandemiephase auf die Fussgängerzahlen im WPZ.

Der WPZ erhebt die Anzahl Besucherinnen und Besucher mittels automatischen Zählstellen. Insgesamt sind 16 Zählstationen im Perimeter WPZ installiert (Stand 2018). Auswertungen zum Zusammenhang von Besuchszahlen und Wetterparametern resp. Zeitfaktoren werden bisher nicht durchgeführt (Stiftung Wildnispark Zürich, 2018). Ziel der Fallstudie ist es, durch zusätzliche Untersuchungen das Verständnis zum Verhalten der Besuchenden und damit die Grundlage für Managemententscheide zu verbessern.

2 Forschungsfrage

Aufgrund der einleitenden Ausführungen ergibt sich die Forschungsfrage: Wie beeinflussen nachfolgende Faktoren die täglichen Fussgängerzahlen im WPZ?

1. Temperatur, Regen und Sonnenscheindauer
2. Wochentage und Ferien
3. Covid19-Pandemie und damit verbundene Lockdowns

3 Methoden

3.1 Untersuchungsgebiet

Das Untersuchungsgebiet liegt im WPZ (Abb. 1). Der Park ist ein 1'100 ha grosses Naturschutzgebiet und durch die Nähe zur Stadt Zürich für rund 2 Millionen Menschen als Erholungsgebiet von Bedeutung (Roth & Stauffer, 2010).

3.2 Datenherkunft

Die Studie wertet Daten der Zählstelle 502 in der Nähe des Besucherzentrums aus (Abb. 1). Der im Eco-Counter eingebaute Pyrosensor registriert Fussgänger*innen richtungsgetrennt aufgrund ihrer Körpertemperatur (www.eco-counter.com). Die Daten sind Eigentum vom WPZ und wurden nach Validierung, Bereinigung und Kalibrierung der ZHAW zur Verfügung gestellt.

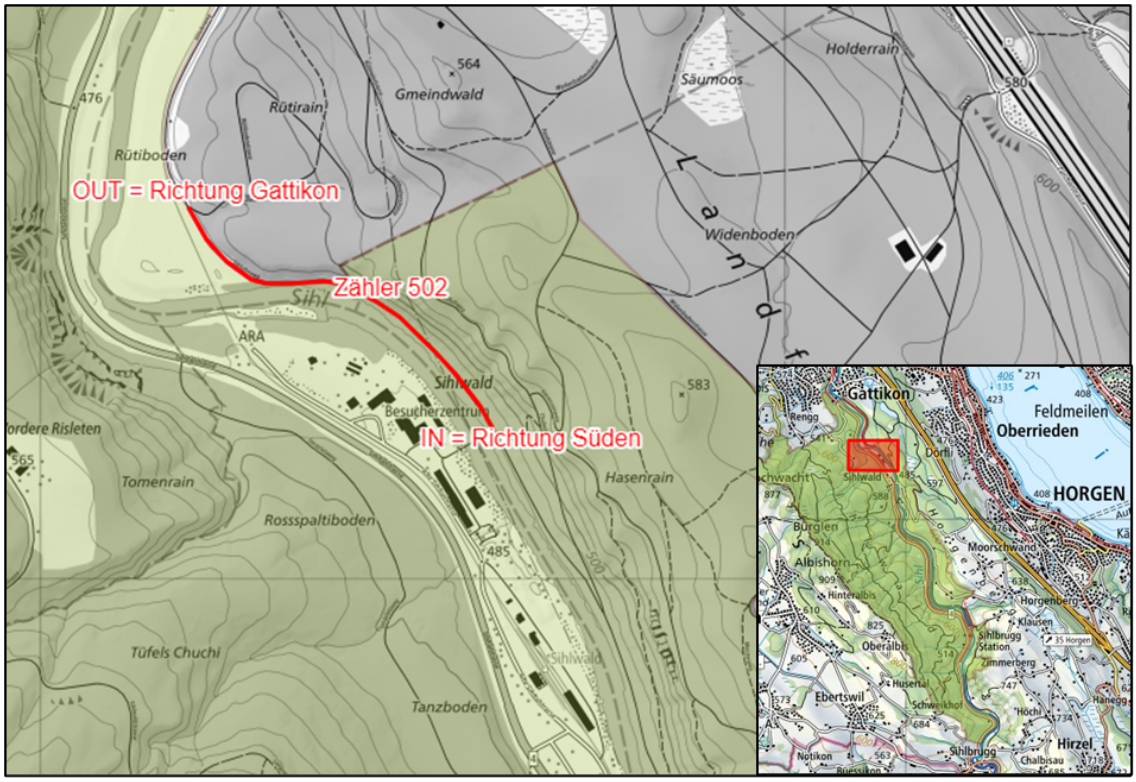

Abb. 1. Standort des WPZ Zählers 502 am Sihluferweg (genauer Standort vertraulich). IN/OUT = Erfassung der Richtung. Die kleine Karte zeigt die Lage und Grenze des WPZ (© swisstopo).

3.3 Variablen

Die stündlichen Fussgängerzahlen zwischen dem 1. Januar 2016 und dem 31. Juli 2021 bilden die abhängige Variable. Zu den unabhängigen Variablen gehören Temperatur, Regen und Sonnenscheindauer, Wochentag, Ferien des Kantons Zürich, Covid19-Pandemiephase (Tabelle 1), Kalenderwoche und Jahr. Die untersuchten Prädiktoren sind in Tabelle 2 zusammengefasst.

Tabelle 1: Daten der vier Covid19-Pandemiephasen.

Covid19-Pandemiephase	Zeitraum
Normal	01.01.2016 - 15.03.2020
Lockdown 1	16.03.2020 - 11.05.2020
Lockdown 2	22.12.2020 - 03.01.2021
Covid (seit Pandemiebeginn, aber ausserhalb der Lockdowns)	12.05.2020 - 21.12.2020 04.01.2021 - 31.07.2021

Tabelle 2: In der Analyse berücksichtigte Prädiktoren der Zahl der Fussgänger*innen im WPZ.

Name	Beschrieb	Einheit	Effekt	Datentyp
Temperatur*	Tagesmaximaltemperatur; Lufttemperatur 2 m über Boden (6 UTC bis 18 UTC)	Grad Celsius	fixed	numerisch
Regen*	Halbtagessumme (6 UTC - 18 UTC, da nächtlicher Niederschlag Tagesbesucher* innen nur indirekt beeinflusst)	Millimeter	fixed	numerisch
Sonnenschein-dauer*	relativ zur absolut möglichen Tagessumme (jahreszeitlich bedingt)	Prozent	fixed	numerisch
Wochentag	Mo, Di, Mi, Do, Fr, Sa, So		fixed	Faktor
Ferien*	ja/nein (Binärcode: 1 = ja/ 0 = nein)		fixed	binärer Faktor
Covid19-Pandemiephase	Pandemiephasen: Normal, Lockdown 1, Lockdown 2, Covid		fixed	Faktor
Kalenderwoche	Kalenderwoche KW 1 bis KW 52 (respektive KW 53)		random	Faktor
Jahr	2016 bis 2021		random	Faktor

* Die Wetterdaten stammen vom Bundesamt für Meteorologie und Klimatologie (2021). Feriendaten wurden von der Seite https://www.schulferien.org/schweiz/ferien/ übernommen (letzte Einsicht, 11.11.2021).

3.4 Statistische Analyse

Alle statistischen Auswertungen erfolgten mit R version 4.1.1 (R Core Team, 2021). Als Signifikanzschwelle wurde $p \leq 0.05$ gewählt. Visualisierungen wurden mit den Packages ggplot2 (Wickham, 2016) und sjPlot (Lüdecke et al., 2021) erstellt.

Sonnenscheindauer war mit Temperatur leicht positiv (r= 0.55) und mit Regen leicht negativ (r= -0.33) korreliert (nach Pearson). Da die Werte unter der kritischen Korrelationsschwelle von 0.6 lagen, wurden alle Wettervariablen für die Modellberechnungen beibehalten. Wegen unterschiedlichen Masseinheiten wurden sie ausserdem skaliert. Die Kalenderwoche und das Jahr wurden als random factors modelliert, um saisonale Schwankungen zu korrigieren. Die Fussgängerdaten wurden anhand des Packages «fitdistrplus» (Delignette-Muller & Dutang, 2015) mit verschiedenen Verteilungen verglichen (normal, log-normal, Poisson, negativ-binomial, exponentiell, Gamma, logistisch, geometrisch und Weibull). Für die passendsten Verteilungen wurden mit dem Package «lme4» (Bates, et al., 2021) Generalised Linear Mixed-Effects Models (GLMMs) sowie Linear Mixed-Effects Models (LMMs) mit und ohne Interaktionen der Prädiktoren konstruiert. Die Modelle wurden mit der Funktion aictab() aus dem Package «AICcmodavg» (Mazerolle, 2020) oder visuell verglichen. Die Modellgüte wurde mit Funktionen aus dem Package «lme4» (Bates, et al., 2021) evaluiert: Normalverteilung mit qqmath(), Varianzhomogenität mit plot(), overdispersion mit dispersion_glmer() und erklärte Varianz mit r.squaredGLMM(). Für die LMMs mit der log-normalen Verteilung wurden die Fussgängerzahlen mit log10 transformiert.

4 Resultate

Während dem Untersuchungszeitraum wurden vom Zähler 502 rund 256’000 Fussgänger*innen erfasst. Die monatlichen Zahlen zeigen saisonale Schwankungen auf (Abb. 2). Im Jahr 2020 zog es deutlich mehr Fussgänger*innen in den WPZ als in vorangehenden Jahren. Der besucherreiche Frühling 2020 fällt in den Zeitraum des ersten Lockdowns. Allerdings waren die Sonnenscheindauer und die Temperaturen in diesem Frühling besonders hoch.

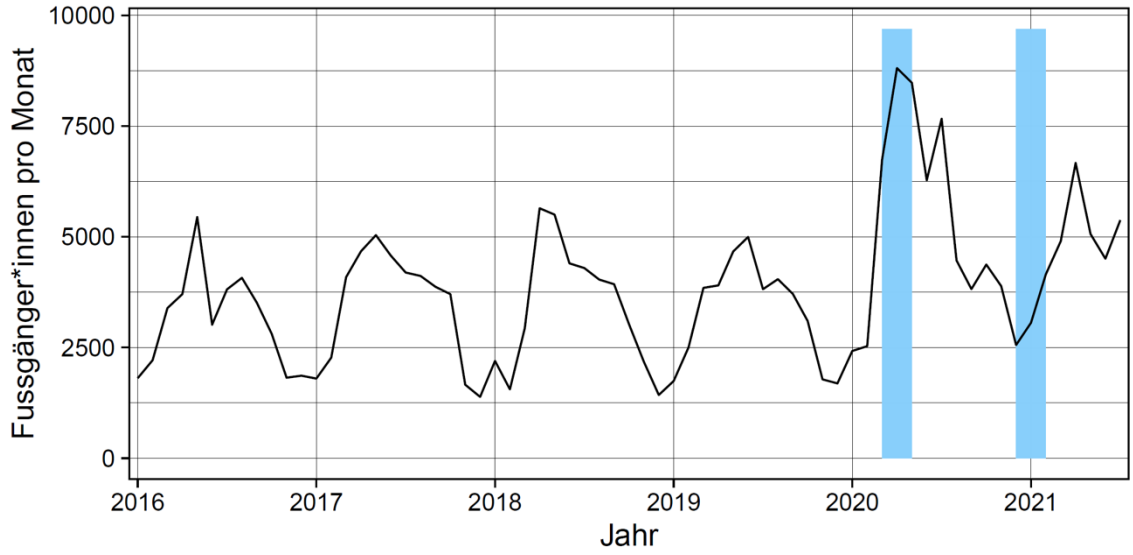

Abb. 2. Monatliche Fussgängerzahlen im WPZ. Zeiträume des 1. und 2. Lockdowns sind blau hinterlegt.

Die stündlichen Fussgängerzahlen sind in allen Phasen jeweils am Samstag und Sonntag am höchsten (Abb. 3). Auffällig ist das erhöhte Besucheraufkommen an Wochentagen während dem Lockdown 1 und der Covid-Phase. Das vergleichsweise tiefe Besucheraufkommen während dem Lockdown 2 lässt sich anhand der kalten Jahreszeit erklären.

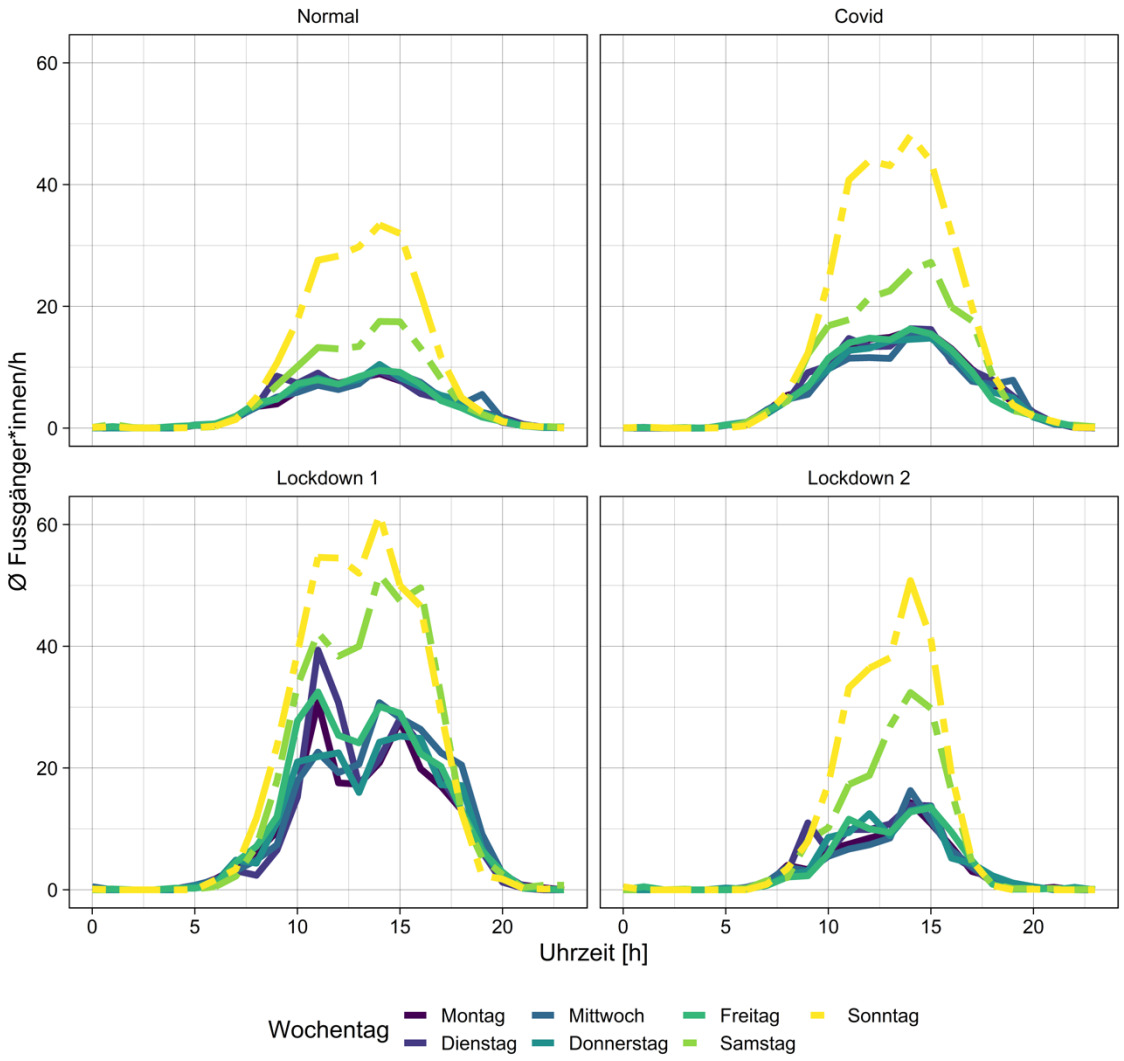

Abb. 3. Durchschnittliche stündliche Fussgängerzahlen im WPZ an verschiedenen Wochentagen in den Covid19-Pandemiephasen Normal, Covid, Lockdown 1 und Lockdown 2.

Als minimal adequates Modell zur Beschreibung der Fussgängerzahlen erwies sich das LMM mit der log-normalen Verteilung mit quadratischem Temperatur-Term und einer Interaktion zwischen Temperatur und Regen sowie Miteinschluss aller Prädiktoren (Tabelle 3). Bezüglich Modellgüte war nur die Normalverteilung trotz log-Transformation nicht gegeben. Dies wurde in Kauf genommen, da LMMs gegenüber Verletzung der Normalitätsannahme als robust gelten (Schielzeth et al., 2020). Das Modell erklärt 70 % der Varianz.

Tabelle 3: Minimal adequates Modell der Fussgängerzahlen im WPZ mit einer Interaktion. Fettgedruckte p-Werte sind signifikant. Für die Wochentage gilt «Montag», für die Covid19-Pandemiephasen gilt die Phase «Normal» und für «Ferien: ja» gilt «Ferien: nein» als Vergleichswert. Zufallsfaktoren: Kalenderwoche (KW) und Jahr. σ^2 = Varianz der Zufallsfaktoren.

Fussgängerzahlen (log10)				
Prädiktoren	**Effektgrössen-schätzung**	**Standard-fehler**	**Konfidenz-intervall**	**p-Wert**
(Achsenabschnitt)	1.82	0.02	1.79 – 1.85	**<0.001**
Tagesmaximaltemperatur	0.07	0.01	0.05 – 0.08	**<0.001**
Regen Halbtagessumme	-0.07	0.00	-0.08 – -0.06	**<0.001**
Sonnenscheindauer	0.10	0.00	0.09 – 0.11	**<0.001**
Tagesmaximaltemperatur quadriert	-0.03	0.00	-0.04 – -0.02	**<0.001**
Wochentag (Dienstag)	0.06	0.01	0.04 – 0.08	**<0.001**
Wochentag (Mittwoch)	0.03	0.01	0.01 – 0.05	**0.013**
Wochentag (Donnerstag)	0.03	0.01	0.01 – 0.05	**0.013**
Wochentag (Freitag)	0.03	0.01	0.01 – 0.06	**0.008**
Wochentag (Samstag)	0.25	0.01	0.23 – 0.28	**<0.001**
Wochentag (Sonntag)	0.49	0.01	0.47 – 0.52	**<0.001**
Ferien (ja)	0.05	0.02	0.02 – 0.08	**0.001**
Phase (Covid)	0.15	0.01	0.14 – 0.17	**<0.001**
Phase (Lockdown 1)	0.28	0.02	0.23 – 0.32	**<0.001**
Phase (Lockdown 2)	0.25	0.02	0.21 – 0.29	**<0.001**
Tagesmaximaltemperatur x Regen Halbtagessumme	0.04	0.00	0.03 – 0.04	**<0.001**
Zufallsfaktoren				
σ^2	0.02			
$\tau_{00\ KW}$	0.01			
$\tau_{00\ Jahr}$	0.00			
ICC	0.22			
N_{KW}	53			
N_{Jahr}	6			
Beobachtungen	2020			
Marginales R^2 / Konditionales R^2	0.709 / 0.772			

Der Wochentag Sonntag hat den stärksten positiven Einfluss auf die Fussgängerzahlen, gefolgt vom Samstag. Die Lockdowns haben einen fast doppelt so grossen positiven Effekt wie die Covid-Phase. Sonnenscheindauer hat einen deutlich positiven, Regen hingegen einen negativen Einfluss. Die Beziehung zur Temperatur ist quadratisch: die Zahlen steigen zunächst mit steigender Temperatur, nehmen aber ab ca. 27°C ab. Es besteht eine signifikante Interaktion zwischen Temperatur und Regen (Abb. 4): Bei höheren Temperaturen hat Regen einen schwach negativen Einfluss. Bei kälteren Temperaturen

ist dieser negative Einfluss viel stärker. Ferien haben einen geringen positiven Einfluss. Abbildungen 5 und 6 zeigen die modellierten Fussgängerzahlen.

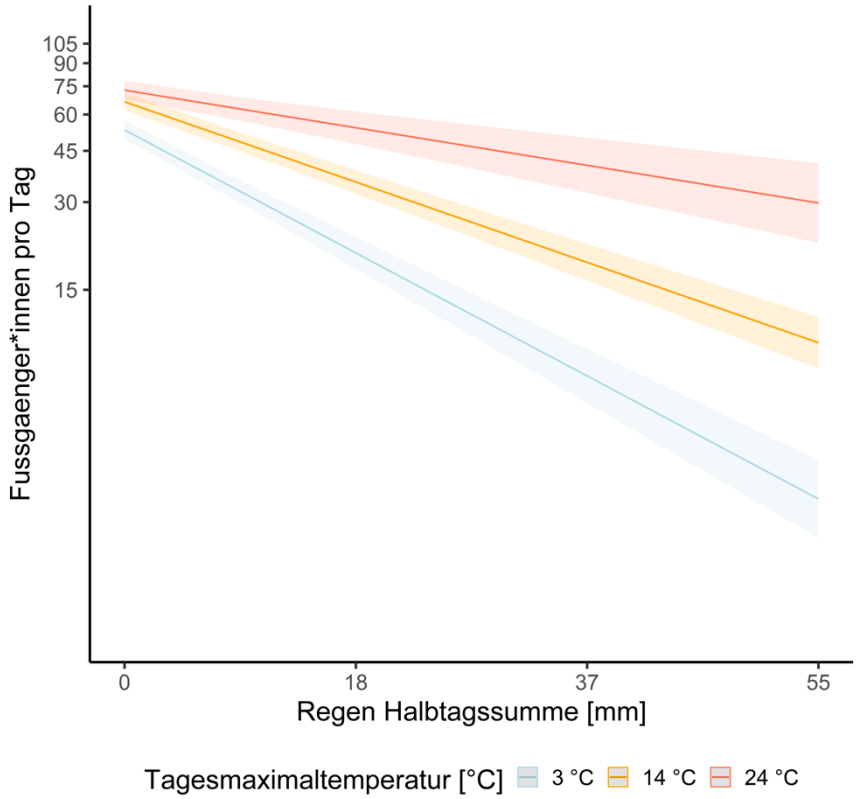

Abb. 4. Interaktion zwischen Temperatur und Regen in Bezug auf die Fussgängerzahlen im WPZ.

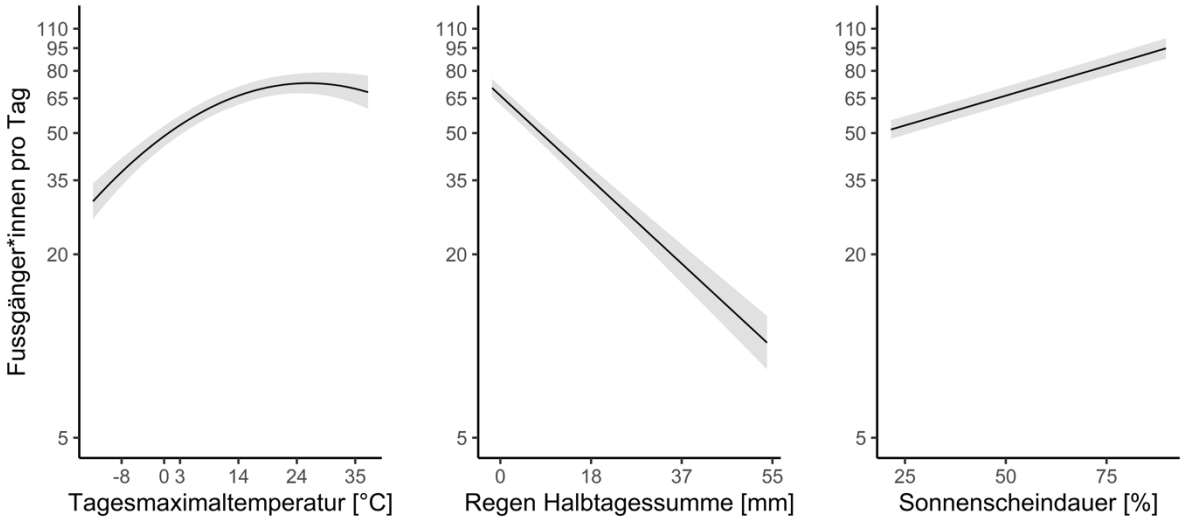

Abb. 5. Modellierte Fussgängerzahlen in Abhängigkeit der Wettervariablen.

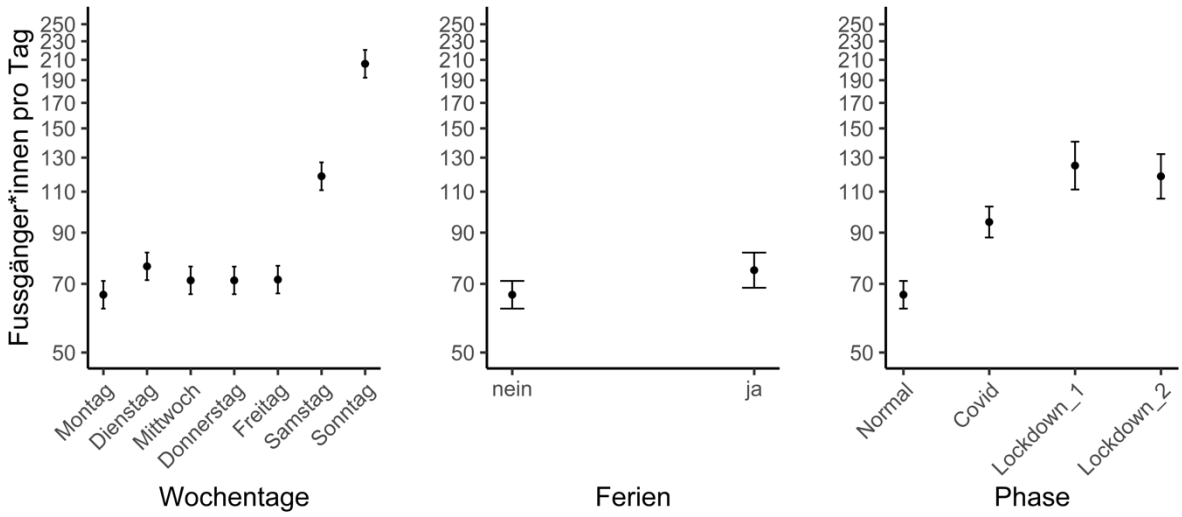

Abb. 6. Modellierte Fussgängerzahlen in Abhängigkeit der Zeitvariablen.

5 Diskussion

Das Wetter hat einen deutlichen Einfluss auf die Fussgängerzahlen am Sihluferweg. Der Besucherrückgang ab 27 Grad kann damit zusammenhängen, dass das Wohlbefinden bei zu hoher Temperatur abnimmt (WHO, 2021). Die Interaktion zwischen Regen und Temperatur lässt sich dadurch erklären, dass Schauer an warmen Tagen als weniger störend wahrgenommen werden (Millhäusler et al., 2016).

Der starke Einfluss des Wochenendes und der schwache Einfluss der Ferien hängen damit zusammen, dass der WPZ ein Naherholungsgebiet ist. Die Naherholung erfolgt im Vergleich zum Tages-, Kurz- oder Ferientourismus stunden- bis halbtagesweise an Werktagen und an Wochenenden (Frick & Buchecker, 2009; Wolf & Appel-Kummer, 2009; Ketterer Bonnelame & Siegrist, 2018).

Der WPZ hat während der Pandemie an Bedeutung gewonnen. Eine repräsentative Befragung der Hochschule Luzern (2021) zeigt, dass die Schweizer Bevölkerung seit Beginn der Covid19-Pandemie mehr Zeit in der Natur verbringt. Auch auf globaler Ebene ist die Nachfrage nach dem Zugang zu Parks während der Pandemie gestiegen (Geng et al., 2021). Das Ergebnis einer Online-Panelstichprobe in der Schweiz zeigt, dass in Summe weniger Menschen den Wald besuchen, dies aber in höherer Frequenz tun. Homeoffice führt dazu, dass die Waldbesuchshäufigkeiten steigen (Wunderlich et al., 2021).

Der Zustrom stellt eine Herausforderung für die Besucherlenkung dar (Derks et al., 2020). Der Fokus muss auf Sensibilisierungskampagnen, Beobachtungsmöglichkeiten und gelenkte Naturerfahrung gelegt werden (Schnabel-Jung & Wipf, 2021).

Als weitere Recherche wäre eine Kombination von qualitativen Besucherbefragungen und Zähldaten denkbar, welche zu einem besseren Verständnis der Besucherdynamik sowie der Bedürfnisse der Besuchenden führen würde (Cessford & Muhar, 2003). In Ergänzung zu den Ferien wäre eine Untersuchung der Auswirkungen von Feiertagen interessant (Frick & Buchecker, 2009; Ketterer Bonnelame & Siegrist, 2018).

6 Literaturverzeichnis

Bates, D., Mächler, M., Bolker, B. & Walker, S. (2021). Linear Mixed-Effects Models using 'Eigen' and S4. Version 1.1-27.1. URL: http://CRAN.R-project.org/package=lme4.

Cessford, G. & Muhar, A. (2003). Monitoring options for visitor numbers in national parks and natural areas. Journal for Nature Conservation, 11(4), 240-250.

Delignette-Muller, M.L. & Dutang, C. (2015). fitdistrplus: An R Package for Fitting Distributions. Journal of Statistical Software, 64, 1-34.

Derks, J., Giessen, L. & Winkel, G. (2020). COVID-19-induced visitor boom reveals the importance of forests as critical infrastructure. Forest Policy and Economics, 118, 102253.

Frick, J. & Buchecker, M. (2009). Ansprüche an die Wohnumgebung im periurbanen Raum. Forschungsbericht. Eidgenössische Forschungsanstalt für Wald, Schnee und Landschaft WSL, Birmensdorf.

Geng, D.C., Innes, J., Wu, W. & Wang, G. (2021). Impacts of COVID-19 pandemic on urban park visitation: a global analysis. Journal of forestry research, 32(2), 553-567.

Hewer, M., Scott, D. & Fenech, A. (2016). Seasonal weather sensitivity, temperature thresholds, and climate change impacts for park visitation. Tourism Geographies, 18(3), 297-321.

Hochschule Luzern HSLU (2021). Corona und Nachhaltigkeit: Ein Jahr danach. Wie «nachhaltig» ist der Corona-Effekt auf das nachhaltige Konsumentenverhalten? 4. Messung zu «Corona und nachhaltiges Konsumentenverhalten» des Instituts für Kommunikation und Marketing (IKM). Luzern.

Ketterer Bonnelame, L. & Siegrist, D. (2018). Naherholungstypen. Leitfaden für die nachfrageorientierte Planung und Gestaltung von naturnahen Naherholungsgebieten. Schriftenreihe des Instituts für Landschaft und Freiraum. HSR Hochschule für Technik Rapperswil, Nr. 15. Rapperswil.

Lüdecke, D (2021). sjPlot: Data Visualization for Statistics in Social Science. R package version 2.8.10. URL: https://CRAN.R-project.org/package=sjPlot.

Mazerolle, M.J. (2020). AICcmodavg: Model selection and multimodel inference based on (Q)AIC(c). R package version 2.3-1. URL: https://cran.r-project.org/package=AICcmodavg.

McGinlay, J., Gkoumas, V., Holtvoeth, J., Fuertes, R.F.A., Bazhenova, E. & Benzoni, A. (2020). The impact of COVID-19 on the management of European protected areas and policy implications. Forests, 11(11), 1214.

Millhäusler, A., Anderwald, P., Haeni, M. & Haller, R. (2016). Publicity, economics and weather - Changes in visitor numbers to a European National Park over 8 years. Journal of Outdoor Recreation and Tourism, 16, 50-57.

R Core Team (2021). R: A language and environment for statistical computing. R Foundation for Statistical Computing. URL: https://www.R-project.org/.

Roth, I. & Stauffer, C. (2010). Charta Wildnispark Zürich Sihlwald 2009-2018. Stiftung Wildnispark Zürich.

Rupf, R. (2014). Choice-Experimente als Grundlage für Agenten-basierte Modelle zur Planung im naturorientierten Outdoorsport. Wandern und Mountainbiking in Tourismus- und Bergregionen sowie Schutzgebieten. Dissertation, University of Natural Resources and Life Sciences Vienna, Austria: University of Natural Resources and Life Sciences.

Schielzeth, H., Dingemanse, N.J., Nakagawa, S., Westneat, D.F., Allegue, H., Teplitsky, C., Réale, D., Dochtermann, N.A., Garamszegi, L.Z. & Araya-Ajoy, Y.G. (2020). Robustness of linear mixed-effects models to violations of distributional assumptions. Methods in Ecology and Evolution, 11(9), 1141-1152.

Schnabel-Jung, U. & Wipf, S. (2021). Extreme Besucherströme bringen neue Herausforderungen für Schutzgebiete. Erholsame Landschaft, 57.

Stiftung Wildnispark Zürich (2018). Managementplan für den Betrieb. Naturerlebnispark Wildnispark Zürich Sihlwald 2020 – 2029. Livebardon Isabelle, Roth Isabelle, Hindenlang Clerc Karin. Sihlwald.

Stiftung Wildnispark Zürich (2021). Jahresbericht 2020. Stiftung Wildnispark Zürich. Sihlwald.

Wickham, H. (2016). ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag, New York.

Wolf, A. & Appel-Kummer, E. (Hrsg.) (2009). Naherholung in Stadt und Land. Norderstedt.

World Health Organization WHO (2021). Heat and health in the WHO European Region: updated evidence for effective prevention. Copenhagen: WHO Regional Office for Europe.

Wunderlich, S., Salak, B., Hegetschweiler, T., Bauer, N. & Hunziker, M. (2021). The woods are calling: Auswirkungen der Corona-Pandemie auf die Schweizer Waldbesuche. WSL Berichte,115, 49–52.

7 Anhang

Das von uns verwendete R-Skript:

```
#.#############################################################################################
# Einfluss von COVID19 auf das Naherholungsverhalten in WPZ ####
# Fallstudie Modul Research Methods, HS21. Melina Grether ####
#.#############################################################################################


# Packages ####
library(tidyverse)              # Data wrangling und piping
library(lubridate)              # Arbeiten mit Datumsformaten
library(data.table)             # schnelles Dateneinlesen
library(ggpubr)                 # to arrange multiple plots in one graph
library(PerformanceAnalytics) # Plotte Korrelationsmatrix
library(MuMIn)                  # Multi-Model Inference
library(AICcmodavg)             # Modellaverageing
library(fitdistrplus)           # Prueft die Verteilung in Daten
library(lme4)                   # Multivariate Modelle
library(blmeco)                 # Bayesian data analysis using linear models
library(sjPlot)                 # Plotten von Modellergebnissen (tab_model)
library(lattice)                # einfaches plotten von Zusammenhaengen zwischen Variablen
library(dplyr)
library(ggplot2)
library(AICcmodavg)
library(patchwork)              # Plots zusammenfuegen


#.#############################################################################################
# METADATA UND DEFINITIONEN ####

#Zeitdefinitionen
depo_start <- as.Date("2016-01-01")
depo_end <- as.Date("2021-07-31")


#lockdown
lock_1_start_2020 <- as.Date("2020-03-16")
lock_1_end_2020 <- as.Date("2020-05-11")

lock_2_start_2021 <- as.Date("2020-12-22")
lock_2_end_2021 <- as.Date("2021-03-01")

# Ferien im Kanton Zuerich
# (https://www.schulferien.org/schweiz/ferien/2020/)

# Als Dataframe abspeichern
# Erster Ferientag: Samstag, Letzter Ferientag: Sonntag
ferien <- data.frame(Start =  1:23, End = 1:23)
row.names(ferien) <- c("W_2015",
                       "F_2016", "S_2016", "H_2016", "W_2016",
                       "F_2017", "S_2017", "H_2017", "W_2017",
                       "F_2018", "S_2018", "H_2018", "W_2018",
                       "F_2019", "S_2019", "H_2019", "W_2019",
                       "F_2020", "S_2020", "H_2020", "W_2020",
                       "F_2021", "S_2021")
ferien$Start <- as.Date(ferien$Start, format = "%Y-%m-%d")
ferien$End <- as.Date(ferien$End, format = "%Y-%m-%d")

ferien["W_2015", "Start"] <- as.Date("2015-12-19")
ferien["W_2015", "End"]   <- as.Date("2016-01-03")

ferien["F_2016", "Start"] <- as.Date("2016-04-23")
ferien["F_2016", "End"]   <- as.Date("2016-05-08")
ferien["S_2016", "Start"] <- as.Date("2016-07-16")
ferien["S_2016", "End"]   <- as.Date("2016-08-21")
ferien["H_2016", "Start"] <- as.Date("2016-10-08")
ferien["H_2016", "End"]   <- as.Date("2016-10-23")
ferien["W_2016", "Start"] <- as.Date("2016-12-24")
ferien["W_2016", "End"]   <- as.Date("2017-01-07")

ferien["F_2017", "Start"] <- as.Date("2017-04-15")
```

```
ferien["F_2017", "End"]   <- as.Date("2017-04-30")
ferien["S_2017", "Start"] <- as.Date("2017-07-15")
ferien["S_2017", "End"]   <- as.Date("2017-08-20")
ferien["H_2017", "Start"] <- as.Date("2017-10-07")
ferien["H_2017", "End"]   <- as.Date("2017-10-22")
ferien["W_2017", "Start"] <- as.Date("2017-12-23")
ferien["W_2017", "End"]   <- as.Date("2018-01-07")

ferien["F_2018", "Start"] <- as.Date("2018-04-21")
ferien["F_2018", "End"]   <- as.Date("2018-05-06")
ferien["S_2018", "Start"] <- as.Date("2018-07-14")
ferien["S_2018", "End"]   <- as.Date("2018-08-19")
ferien["H_2018", "Start"] <- as.Date("2018-10-08")
ferien["H_2018", "End"]   <- as.Date("2018-10-20")
ferien["W_2018", "Start"] <- as.Date("2018-12-24")
ferien["W_2018", "End"]   <- as.Date("2019-01-05")

ferien["F_2019", "Start"] <- as.Date("2019-04-22")
ferien["F_2019", "End"]   <- as.Date("2019-05-04")
ferien["S_2019", "Start"] <- as.Date("2019-07-15")
ferien["S_2019", "End"]   <- as.Date("2019-08-17")
ferien["H_2019", "Start"] <- as.Date("2019-10-07")
ferien["H_2019", "End"]   <- as.Date("2019-10-19")
ferien["W_2019", "Start"] <- as.Date("2019-12-23")
ferien["W_2019", "End"]   <- as.Date("2020-01-05")

ferien["F_2020", "Start"] <- as.Date("2020-04-11")
ferien["F_2020", "End"]   <- as.Date("2020-04-26")
ferien["S_2020", "Start"] <- as.Date("2020-07-11")
ferien["S_2020", "End"]   <- as.Date("2020-08-16")
ferien["H_2020", "Start"] <- as.Date("2020-10-03")
ferien["H_2020", "End"]   <- as.Date("2020-10-18")
ferien["W_2020", "Start"] <- as.Date("2020-12-19")
ferien["W_2020", "End"]   <- as.Date("2021-01-03")

ferien["F_2021", "Start"] <- as.Date("2021-04-24")
ferien["F_2021", "End"]   <- as.Date("2021-05-09")
ferien["S_2021", "Start"] <- as.Date("2021-07-17")
ferien["S_2021", "End"]   <- as.Date("2021-08-22")

#.############################################################################################
# Datenherkunft ####
# Die Zaehldaten der Zaehlstelle 502 gehoeren dem Wildnispark Zuerich
# und wurden nach Bereinigung und Kalibrierung der ZHAW zur Verfuegung gestellt.
# Die Wetterdaten stammen vom Bundesamt fuer Meteorologie und Klimatologie (2021).

#.############################################################################################
# DATENIMPORT UND STRUKTUR ANPASSUNG #####

## Daten einlesen, Struktur anpassen ####
depo <- read.csv("502_sihluferweg_2016_2021.csv", sep = ";")

str(depo) # Struktur anzeigen


depo <- depo %>% # Zeilen auswaehlen, Zeilennamen und Datentypen anpassen
  dplyr::select(-c(Velo_IN, Velo_OUT)) %>% # Spalten auswaehlen
  rename(Fuss_in = Fuss_IN, Fuss_out = Fuss_Out) %>%
  mutate(Datum_Uhrzeit = as.character(DatumUhrzeit)) %>%
  separate(Datum_Uhrzeit, into = c("Datum", "Zeit"), sep = " ") %>% # mit seperate() trennt man 1 Spalte in 2.
  mutate(Datum = as.Date(Datum, format = "%d.%m.%Y")) # hier wird Text zum Datum

depo <- depo %>%  # Besuchszahlen total
  mutate(Fuss_tot = Fuss_in + Fuss_out) %>% # Summe der Zaehlungen in neuer Spalte
  na.omit() # fehlende Werte entfernen

meteo <- read.csv("order_97149_data_2.txt", sep = ";")

str(meteo)

meteo <- meteo %>% # Zeilennamen und Datentypen anpassen
  rename(Stn = stn, Datum = time, Temp_max = tre200jx, Regen = rre150j0, Sonne = sremaxdv) %>%
  transform(Datum = as.Date(as.character(Datum), "%Y%m%d")) %>%
  transform(Temp_max = as.numeric(Temp_max)) %>%
  transform(Regen = as.numeric(Regen))
```

```
meteo <- meteo %>% # ueberfluessige Werte entfernen
  filter(Datum >= depo_start, Datum <= depo_end) %>% # auf Untersuchungsdauer eingrenzen
  na.omit() # fehlende Werte entfernen

sum(is.na(meteo)) # noch fehlende Werte?

## Conveniencevariablen einfuegen, Struktur anpassen ####

depo <- depo %>% # nach Werk- und Wochentage einteilen
  mutate(Wochentag = weekdays(Datum)) %>%
  mutate(Wochentag = base::factor(Wochentag,
                                  levels = c("Montag", "Dienstag", "Mittwoch", "Donnerstag", "Freitag", "Samstag", "Sonntag"))) %>%
  mutate(Wochenende = if_else(Wochentag == "Samstag" | Wochentag == "Sonntag", "Wochenende", "Werktag")) # | ist
"or"

depo <- depo %>% # weitere Conveniencvariablen erstellen
  mutate(KW = week(Datum)) %>% # Kalenderwoche
  mutate(Monat = month(Datum)) %>% # Monat
  mutate(Jahr = year(Datum)) # Jahr


depo <- depo %>% # Covid-Phasen
  mutate(Phase = if_else(Datum >= lock_1_start_2020 & Datum <= lock_1_end_2020,
                         "Lockdown_1",
                         if_else(Datum >= lock_2_start_2021 & Datum <= lock_2_end_2021,
                                 "Lockdown_2",
                                 if_else(Datum < lock_1_start_2020,
                                         "Normal", "Covid"))))

depo <- depo %>% # als Faktor speichern
  transform(Wochenende = as.factor(Wochenende), KW = as.factor(KW))

depo$Phase <- base::factor(depo$Phase, levels = c("Normal", "Covid", "Lockdown_1", "Lockdown_2")) # Levels ordnen

depo <- depo %>% # weitere Conveniencvariable erstellen
  mutate(Stunde = as.numeric(format(as.POSIXct(depo$Zeit,format="%H:%M"),"%H"))) # Stunde

depo <- depo %>% # Zaehldaten als ganze Zahlen speichern
  transform(Fuss_in = round(Fuss_in, digits = 0), Fuss_out = round(Fuss_out, digits = 0), Fuss_tot = round(Fuss_tot, digits
= 0)) %>%
  transform(Fuss_in = as.integer(Fuss_in), Fuss_out = as.integer(Fuss_out), Fuss_tot = as.integer(Fuss_tot))

depo_d <- depo %>% # Studen zu Tagen aggregieren
  group_by(Datum, Wochentag, Wochenende, KW, Monat, Jahr, Phase) %>%
  summarise(Fuss_in = sum(Fuss_in),
            Fuss_out = sum(Fuss_out),
            Fuss_tot = sum(Fuss_in + Fuss_out))

depo_m <- depo %>% # Studen zu Monaten aggregieren
  group_by(Jahr, Monat) %>%
  summarise(Fuss_tot = sum(Fuss_tot))

depo_m <- depo_m %>% # neue Zeile mit Jahr und Monat kombiniert
  mutate(Jahr = as.factor(Jahr)) %>% # Jahr und Monat als Faktoren
  mutate(Monat = as.factor(Monat)) %>%
  mutate(JM = paste(Jahr, Monat)) %>% # neue Spalte, in der Jahr und Monat zusammen
  mutate(JM= factor(JM, levels=unique(JM))) # als Faktor, die Levels sind die einzelnen Eintraege in der Spalte (welche
ja bereits geordnet sind)

#.#############################################################################################

#.#############################################################################################
# DESKRIPTIVE ANALYSE UND VISUALISIERUNG #####

## Besuchszahlen pro Monat total ####
# x-Achse: Jahr und Monat, y-Achse: monatlichen Besuchszahlen
ggplot(data = depo_m, mapping = aes(JM, Fuss_tot, group = 1)) + # group 1 braucht R, dass aus den Einzelpunkten ein
Zusammenhang hergestellt wird
  # Lockdown 1 einzeichnen
  xlab("Jahr") +
  ylab("Fussgänger*innen pro Monat") +
  geom_rect(mapping = aes(xmin="2020 3", xmax="2020 5",
                          ymin =0, ymax=max(Fuss_tot+(Fuss_tot/100*10))), # das Rechteck ist 10% groesser als die maximale
Besuchszahl
            fill = "lightskyblue", alpha = 0.4, colour = NA) + # alpha ist die Transparenz
  # Lockdown 2 einzeichnen
```

```
  geom_rect(mapping = aes(xmin="2020 12", xmax="2021 2",
                          ymin =0, ymax=max(Fuss_tot+(Fuss_tot/100*10))),
            fill = "lightskyblue", alpha = 0.4, colour = NA) +
  # Linie zeichnen
  geom_line() +
  scale_x_discrete(breaks = c("2016 1", "2017 1", "2018 1", "2019 1", "2020 1", "2021 1"),
                   labels = c("2016", "2017", "2018", "2019", "2020", "2021")) +
  theme_linedraw(base_size = 15)

ggsave("Results/Besucher_pro_Monat.pdf", width = 20, height = 10, units = "cm", dpi = 1000)

## Besuchszahlen pro Wochentag pro Phase total ####
tot_phase_wd <- depo_d %>%
  group_by(Phase, Wochentag) %>%
  summarise(sum(Fuss_tot))

write.csv(tot_phase_wd, "results/tot_phase_wd.csv", row.names = FALSE)

ggplot(data = depo_d)+
  geom_boxplot(mapping = aes(x= Wochentag, y = Fuss_tot, fill = Phase)) +
  ylab("Fussgänger*innen pro Tag") +
  theme_classic() +
  scale_fill_brewer(palette="GnBu") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1), legend.title = element_blank())

ggsave("Results/Besucher_pro_Tag.pdf", width = 20, height = 10, units = "cm", dpi = 1000)

## Unterschied Werktage und Wochenende ####
# Sind die Unterschiede zwischen Werktag und Wochenende wirklich signifikant?
# Normal
t.test(depo_d$Fuss_tot[depo_d$Phase == "Normal" & depo_d$Wochenende == "Werktag"],
       depo_d$Fuss_tot[depo_d$Phase == "Normal" & depo_d$Wochenende == "Wochenende"])
# Covid
t.test(depo_d$Fuss_tot[depo_d$Phase == "Covid" & depo_d$Wochenende == "Werktag"],
       depo_d$Fuss_tot[depo_d$Phase == "Covid" & depo_d$Wochenende == "Wochenende"])
# Lockdown 1
t.test(depo_d$Fuss_tot[depo_d$Phase == "Lockdown_1" & depo_d$Wochenende == "Werktag"],
       depo_d$Fuss_tot[depo_d$Phase == "Lockdown_1" & depo_d$Wochenende == "Wochenende"])
# Lockdown 2
t.test(depo_d$Fuss_tot[depo_d$Phase == "Lockdown_2" & depo_d$Wochenende == "Werktag"],
       depo_d$Fuss_tot[depo_d$Phase == "Lockdown_2" & depo_d$Wochenende == "Wochenende"])

## Mittlere Besuchszahlen pro Wochentag pro Stunde pro Phase ####
mean_h <- depo %>%
  group_by(Phase, Wochentag, Stunde) %>%
  summarise(Mittelwert = mean(Fuss_tot))

phase_labels <- as_labeller(c(Covid = "Covid", Lockdown_1 = "Lockdown 1", Lockdown_2 = "Lockdown 2", Normal =
"Normal")) #Labels festlegen, Reihenfolgs unabhaengig

ggplot(subset(mean_h), mapping=aes(x = Stunde, y = Mittelwert, colour = Wochentag, linetype = Wochentag)) +
  geom_line(size = 2) +
  scale_colour_viridis_d() +
  # scale_colour_brewer(palette="GnBu") + # fuer gleiche Farben wie in anderen Plots
  scale_linetype_manual(values = c(rep("solid", 5),  "twodash", "twodash"))+
  labs(x="Uhrzeit [h]", y= "ø Fussgänger*innen/h", title = "")+
  theme_linedraw(base_size = 15)+
  theme(legend.position = "bottom") +
  facet_wrap(Phase~., labeller = phase_labels) + # Phasen als Facets; Name der Facets anpassen
  theme(strip.background = element_blank()) + # strip weiss
  theme(strip.text = element_text(colour = "black"))

ggsave("Results/Tagesgang.png", width=25, height=25, units="cm", dpi=1000)


## Besuchszahlen pro Phase ####
# total
sum_phase_d <- depo %>%
  group_by(Phase, Wochentag) %>%
  summarise(Summe_in = sum(Fuss_in), Summe_out = sum(Fuss_out))

write.csv(sum_phase_d, "Results/sum_phase_d.csv")

# # Mittlere Besuchszahlen pro Phase, nach Richtung getrennt
# mean_phase_d <- depo %>%
#   group_by(Phase, Wochentag) %>%
```

```
#   summarise(Mittelwert_in = mean(Fuss_in), Mittelwert_out = mean(Fuss_out), Mittelwert_tot = mean(Fuss_tot)) %>%
#   mutate(Proz_in = round(100/Mittelwert_tot*Mittelwert_in, 1)) %>% # prozentuale Richtungsverteilung berechnen, auf
eine Nachkommastelle runden
#   mutate(Proz_out = round(100/Mittelwert_tot*Mittelwert_out, 1))
#
# write.csv(mean_phase_d, "Results/mean_phase_d.csv")

# mean_phase_d_abs <- mean_phase_d %>% # Selektion der absoluten Zahlen
#   dplyr::select(-c(Mittelwert_tot:Proz_out)) %>%
#   rename("in" = Mittelwert_in, out = Mittelwert_out)
#
# mean_phase_d_proz <- mean_phase_d %>% # Selektion der relativen Zahlen
#   dplyr::select(-c(Mittelwert_in:Mittelwert_tot)) %>%
#   rename("in" = Proz_in, out = Proz_out)
#
# # Transformation zu long tabel
# mean_phase_d_abs <- pivot_longer(mean_phase_d_abs, cols = c("in","out"),
#                                  names_to = "Gruppe", values_to = "Durchschnitt")
# mean_phase_d_proz <- pivot_longer(mean_phase_d_proz, cols = c("in","out"),
#                                   names_to = "Gruppe", values_to = "Prozent")

# # Visualisierung
# # Visualisierung abs
# abs <- ggplot(data = mean_phase_d_abs, mapping = aes(x = Gruppe, y = Durchschnitt, fill = Phase))+
#   geom_col(position = "dodge", width = 0.8)+
#   scale_fill_brewer(palette="GnBu", name = "Phase") +
#   #scale_fill_manual(values = c("royalblue", "red4", "orangered", "gold2"), name = "Phase")+
#   scale_x_discrete(labels = c("in", "out"))+
#   labs(y = "Durchschnitt [mean]", x= "Bewegungsrichtung")+
#   theme_classic(base_size = 15)+
#   theme(legend.position = "bottom")
#
# # Visualisierung %
# proz <- ggplot(data = mean_phase_d_proz, mapping = aes(x = Gruppe, y = Prozent, fill = Phase))+
#   geom_col(position = "dodge", width = 0.8)+
#   scale_fill_brewer(palette="GnBu", name = "Phase") +
#   scale_x_discrete(labels = c("in", "out"))+
#   labs(y = "Durchschnitt [%]", x= "Bewegungsrichtung")+
#   theme_classic(base_size = 15)+
#   theme(legend.position = "bottom")
#
# # Arrange und Export Verteilung
# ggarrange(abs,                  # plot 1 aufrufen
#           proz,                 # plot 2 aufrufen
#           ncol = 2, nrow = 1,   # definieren, wie die plots angeordnet werden
#           heights = c(1),       # beide sind leich hoch
#           widths = c(1,0.95),   # plot 2 ist aufgrund der fehlenden y-achsenbesch. etwas schmaler
#           labels = c("a) Absolute Verteilung", "b) Relative Verteilung"),
#           label.x = 0,          # wo stehen die labels
#           label.y = 1.0,
#           common.legend = TRUE, legend = "bottom") # wir brauchen nur eine Legende, unten
#
# ggsave("Results/Verteilung.png", width=20, height=15, units="cm", dpi=1000)


#.############################################################################################

#.############################################################################################
# MULTIFAKTORIELLE ANALYSE UND VISUALISIERUNG #####

## Zwei Datensaetze (depo_d, meteo) verbinden ####
# Join: Datensatz mit den taeglichen Zaehldaten und Datumsinformationen angereichert mit Wetterdaten
umwelt <- dplyr::left_join(depo_d, meteo, by = "Datum")

sum(is.na(umwelt)) # Fehlende Daten vorhanden?
umwelt <- na.omit(umwelt)


## Ferienzeitraeume definieren ####
# WENN Ferien waren, DANN = 1, SONST = 0
# Fuer jede Ferien: Datum ueberpruefen, ob in diesen Ferien
# Zutreffend: 1 in die Spalte Ferein zu diesem Datum
for (i in 1:nrow(ferien)){
  umwelt$Ferien[umwelt$Datum >= ferien[i,"Start"] & umwelt$Datum <= ferien[i,"End"]] <- 1
}
umwelt$Ferien[is.na(umwelt$Ferien)] <- 0 # alle Daten ausserhalb der Ferien
```

```
umwelt <- umwelt %>%
  transform(Ferien = as.factor(Ferien), Jahr = as.factor(Jahr)) # Jahr und Ferien als Faktor

umwelt <- umwelt %>% # Skalieren aller Variablen mit Masseinheiten; um den Mittelpunkt gemittelt
  mutate(Temp_max_scaled = scale(Temp_max)) %>%
  mutate(Regen_scaled = scale(Regen)) %>%
  mutate(Sonne_scaled = scale(Sonne))

## Korrelation ####
cor <- cor(umwelt[,16:18]) # Temp_max_scaled:Sonne_scaled
cor <- cor(umwelt[,c("Temp_max_scaled", "Regen_scaled", "Sonne_scaled")]) # alternativ
cor[abs(cor) < 0.6] <-  0  # Alle Werte kleiner 0.6 auf 0
# wir entfernen keine Variablen, da es nur eine schwache Korrelation (<0.6) zwischen Temp_max und Sonne gibt.

chart.Correlation(umwelt[,16:18], histogram=TRUE, pch=19) # Korrelation darstellen

## Daten beschreiben ####
min(umwelt$Temp_max) # niedrigste Hoechsttemperatur
max(umwelt$Temp_max) # maximale Hoechsttemperatur

regen_monat <- umwelt %>% # Regen pro Monat
  group_by(Jahr, Monat) %>%
  summarise(Regen_sum = sum(Regen))

min(umwelt$Sonne) # minimale Sonnenscheindauer
max(umwelt$Sonne) # maximale Sonnenscheindauer
sum(umwelt$Sonne == 0) # wie viele Tage ohne Sonne?
sum(umwelt$Sonne == 100) # wie viele Tage volle Sonne?

sum(depo_d$Fuss_tot) #totale Besuchszahl

## Verteilung der abhaengigen Variable pruefen ####
verteilung <- list()
verteilung[[1]] <-fitdist(umwelt$Fuss_tot,"norm")    # Normalverteilung
verteilung[[2]] <-fitdist(umwelt$Fuss_tot,"lnorm")   # log-Normalvert.
verteilung[[3]] <-fitdist(umwelt$Fuss_tot,"pois")    # Poisson
verteilung[[4]] <-fitdist(umwelt$Fuss_tot,"nbinom")  # negativ binomial
verteilung[[5]] <-fitdist(umwelt$Fuss_tot,"exp")     # exponentiell
verteilung[[6]] <-fitdist(umwelt$Fuss_tot,"gamma")   # gamma
verteilung[[7]] <-fitdist(umwelt$Fuss_tot,"logis")   # logistisch
verteilung[[8]] <-fitdist(umwelt$Fuss_tot,"geom")    # geometrisch
verteilung[[9]] <-fitdist(umwelt$Fuss_tot,"weibull") # Weibull

# Verteilungen vergleichen
gofstat(verteilung,
       fitnames = c("Normalverteilung", "log-Normalverteilung", "Poisson",
                    "negativ binomial","exponentiell","gamma", "logistisch",
                    "geometrisch","weibull"))
# Modell mit kleinster Akaike ist am passensten

# die 4 besten (gemaess Akaike's Information Criterion) als Plot,
plot.legend <- c("log norm", "negativ binomial", "gamma ", "weibull")
# vergleicht mehrere theoretische Verteilungen mit den empirischen Daten
cdfcomp(verteilung[c(2, 4, 6, 9)], legendtext = plot.legend)
# -> Log norm passt am bessten

# Multivariates Modell berechnen ####
# Abhaengige Variable: Totale Besuchszahl
# Erklaerende/Unabhaengige Variablen: Wetterparameter, Wochentag, Ferien, Covid-Phasen
# Random Factors: Saisonalitaet (KW und Jahr)

## Verschiedene Modelle erstellen ####
# Einfaches Modell der Familie poisson
Modell_poisson <- glmer(Fuss_tot ~ Temp_max_scaled + Regen_scaled +
                    Sonne_scaled + Wochentag +
                    Ferien + Phase +
                    (1|KW)+ (1|Jahr),
                  family = poisson, data = umwelt)

# negativ binomial
Modell_nb <- glmer.nb(Fuss_tot ~ Temp_max_scaled + Regen_scaled +
                        Sonne_scaled + Wochentag +
                        Ferien + Phase +
                        (1|KW)+ (1|Jahr), data = umwelt)
```

```
# log norm, nicht mit glmer(, family = gaussian) sondern mit lmer()
Modell_lognorm <- lmer(log10(Fuss_tot) ~ Temp_max_scaled + Regen_scaled + # +1 um log(0) zu vermeiden
                       Sonne_scaled + Wochentag +
                       Ferien + Phase +
                       (1|KW)+ (1|Jahr), data = umwelt)

# ln norm
Modell_lnnorm <- lmer(log(Fuss_tot) ~ Temp_max_scaled + Regen_scaled + # +1 um log(0) zu vermeiden
                      Sonne_scaled + Wochentag +
                      Ferien + Phase +
                      (1|KW)+ (1|Jahr), data = umwelt)

# Gamma
Modell_Gamma <- glmer(Fuss_tot ~ Temp_max_scaled + Regen_scaled + # +1 um log(0) zu vermeiden
                      Sonne_scaled + Wochentag +
                      Ferien + Phase +
                      (1|KW)+ (1|Jahr), family = Gamma(link = "log"), data = umwelt)

## Grundmodelle vergleichen ####
# Aus Package AICcmodavg
cand.models <- list()
cand.models[[1]] <- Modell_nb
cand.models[[2]] <- Modell_Gamma
cand.models[[3]] <- Modell_poisson

Modnames <- c("Neg Binom", "Gamma", "Poisson")
aictab(cand.set=cand.models, modnames=Modnames) # -> Gamma passt am besten

# lognorm und lnnorm lassen sich nur visuell vergleichen, da sie andere Skalen haben

summary(Modell_lognorm)          # Resultat
plot(Modell_lognorm, type = c("p", "smooth")) # Verteilung der Residuen
qqmath(Modell_lognorm)           # Pruefen auf Normalverteilung
dispersion_glmer(Modell_lognorm) # pruefen auf Overdispersion (sollte nicht ueber 1.4 sein)
r.squaredGLMM(Modell_lognorm)    # erklaerte Varianz

summary(Modell_lnnorm)
plot(Modell_lnnorm, type = c("p", "smooth"))
qqmath(Modell_lnnorm)
dispersion_glmer(Modell_lnnorm)
r.squaredGLMM(Modell_lnnorm)

# kaum Unterschiede zwischen log und ln
# -> ich waehle log

# Lognorm und Gamma muessen auch manuell verglichen werden
summary(Modell_lognorm)
plot(Modell_lognorm, type = c("p", "smooth"))
qqmath(Modell_lognorm)
dispersion_glmer(Modell_lognorm)
r.squaredGLMM(Modell_lognorm)

summary(Modell_Gamma)
plot(Modell_Gamma, type = c("p", "smooth"))
qqmath(Modell_Gamma)
dispersion_glmer(Modell_Gamma)
r.squaredGLMM(Modell_Gamma)

# -> lognorm ist das beste Modell

## Finetunig des Grundmodells ####
# Quadratischer Temperatur Term
Modell_lognorm_quad <- lmer(log10(Fuss_tot) ~ Temp_max_scaled + I(Temp_max_scaled^2) + Regen_scaled +
                            Sonne_scaled + Wochentag +
                            Ferien + Phase +
                            (1|KW)+ (1|Jahr), data = umwelt)

# Interaktion von Temperatur und Regen
Modell_lognorm_TempxRegen <- lmer(log10(Fuss_tot) ~ Temp_max_scaled * Regen_scaled +
                            Sonne_scaled + Wochentag +
                            Ferien + Phase +
                            (1|KW)+ (1|Jahr), data = umwelt)

# Quadratischer Temperatur Term, Interaktion von Temperatur und Regen
Modell_lognorm_quad_TempxRegen <- lmer(log10(Fuss_tot) ~ Temp_max_scaled * Regen_scaled +
I(Temp_max_scaled^2) +
```

```
                                Sonne_scaled + Wochentag +
                                Ferien + Phase +
                                (1|KW)+ (1|Jahr), data = umwelt)

# Interaktion von Ferien und Phase
Modell_lognorm_FerienxPhase <- lmer(log10(Fuss_tot) ~ Temp_max_scaled + Regen_scaled +
                                Sonne_scaled + Wochentag +
                                Ferien * Phase +
                                (1|KW)+ (1|Jahr), data = umwelt)

# Quadratischer Temperatur Term, Interaktion von Ferien und Phase
Modell_lognorm_quad_FerienxPhase <- lmer(log10(Fuss_tot) ~ Temp_max_scaled + I(Temp_max_scaled^2) + Regen_scaled +
                                Sonne_scaled + Wochentag +
                                Ferien * Phase +
                                (1|KW)+ (1|Jahr), data = umwelt)

# Quadratischer Temperatur Term, Interaktion von Wochentagen und Phase
Modell_lognorm_quad_WTxPhase <- lmer(log10(Fuss_tot) ~ Temp_max_scaled + I(Temp_max_scaled^2) + Regen_scaled +
                                Sonne_scaled + Wochentag * Phase +
                                Ferien +
                                (1|KW)+ (1|Jahr), data = umwelt)


## Vergleich der Modellguete mittels AICc
cand2.models <- list()
cand2.models[[1]] <- Modell_lognorm
cand2.models[[2]] <- Modell_lognorm_quad
cand2.models[[3]] <- Modell_lognorm_TempxRegen
cand2.models[[4]] <- Modell_lognorm_quad_TempxRegen
cand2.models[[5]] <- Modell_lognorm_FerienxPhase
cand2.models[[6]] <- Modell_lognorm_quad_FerienxPhase
cand2.models[[7]] <- Modell_lognorm_quad_WTxPhase

Modnames2 <- c("Modell_lognorm","Modell_lognorm_quad", "Modell_lognorm_TempxRegen",
          "Modell_lognorm_quad_TempxRegen", "Modell_lognorm_FerienxPhase",
          "Modell_lognorm_quad_FerienxPhase", "Modell_lognorm_quad_WTxPhase")
aictab(cand.set=cand2.models,modnames=Modnames2)

# -> Das Modell Modell_lognorm_quad_TempxRegen ist am besten geeignet

# Modelldiagnostik des besten Modells
summary(Modell_lognorm_quad_TempxRegen)
plot(Modell_lognorm_quad_TempxRegen, type = c("p", "smooth"))
qqmath(Modell_lognorm_quad_TempxRegen) # die Normalverteilung ist nicht gegeben
dispersion_glmer(Modell_lognorm_quad_TempxRegen)
r.squaredGLMM(Modell_lognorm_quad_TempxRegen)

tab_model(Modell_lognorm_quad_TempxRegen, transform = NULL, show.se = TRUE)
# The marginal R squared values are those associated with your fixed effects,
# the conditional ones are those of your fixed effects plus the random effects.
# Usually we will be interested in the marginal effects.

# Environment speichern, damit zu Beginn nicht immer alle Modelle gerechnet werden muessen
# save.image(file='myEnvironment.RData')


## Modellvisualisierung ####
# Environment ohne Modellvisualisierung laden
# load('myEnvironment.RData')
# Limit der y-Achse fuer Meteo-Plots definieren
limit_lower <- 0.7
limit_upper <- 2.05
labels_for_plot <- seq(round(10^limit_lower),round(10^limit_upper), 15)
breaks_for_plot <- log10(labels_for_plot)

# Temperatur
t <- plot_model(Modell_lognorm_quad_TempxRegen, type = "pred",
          terms = "Temp_max_scaled[all]",
          title = "", axis.title = c("Tagesmaximaltemperatur [°C]",
                                "Fussgänger*innen pro Tag"))
# Unskalierte Werte
labels <- round(seq(floor(min(umwelt$Temp_max)), ceiling(max(umwelt$Temp_max)), length.out = 5), 0)
labels_null_value <- c(labels[1], 0,labels[-1]) # 0 Wert einfuegen
```

```
Tempplot <- t + scale_x_continuous(breaks = c(-2,-1-3/11,-1,0,1,2), labels = c(labels_null_value))+
  scale_y_continuous(breaks =breaks_for_plot,
                     labels = labels_for_plot,
                     limits = c(limit_lower, limit_upper)) +
  theme_classic(base_size = 20)
# breaks: transformiert, labels: tatsaechlich

# Exportiere das Resultat
ggsave("Results/Temp.png", width=15, height=15, units="cm", dpi=1000)

# Sonne
s <- plot_model(Modell_lognorm_quad_TempxRegen, type = "pred",
                terms = "Sonne_scaled[all]",
                title = "", axis.title = c("Sonnenscheindauer [%]",
                                           ""))
# Unskalierte Werte
labels <- round(seq(floor(min(umwelt$Sonne)), ceiling(max(umwelt$Sonne)), length.out = 5), 0)
Sonneplot <- s + scale_x_continuous(breaks = c(-2,-1,0,1,2), labels = c(labels))+
  scale_y_continuous(breaks = breaks_for_plot,
                     labels = labels_for_plot,
                     limits = c(limit_lower, limit_upper)) +
  theme_classic(base_size = 20)

ggsave("Results/Sonne.png", width=15, height=15, units="cm", dpi=1000)

# Regen
r <- plot_model(Modell_lognorm_quad_TempxRegen, type = "pred",
                terms = "Regen_scaled[all]",
                title = "", axis.title = c("Regen Halbtagessumme [mm]",
                                           ""))
# Unskalierte Werte
labels <- round(seq(floor(min(umwelt$Regen)), ceiling(max(umwelt$Regen)), length.out = 4), 0)
Regenplot <- r + scale_x_continuous(breaks = c(0,4,8,12), labels = c(labels))+
  scale_y_continuous(breaks =breaks_for_plot,
                     labels = labels_for_plot,
                     limits = c(limit_lower, limit_upper)) +
  theme_classic(base_size = 20)

ggsave("Results/Regen.png", width=15, height=15, units="cm", dpi=1000)

# Plots zusammenfuegen
wrap_plots(Tempplot, Regenplot, Sonneplot)

ggsave("Results/Wetter.png", width=35, height=17, units="cm", dpi=1000)


# Limit der y-Achse fuer Zeit-Plots definieren, ueberschreiben
limit_lower <- 1.7
limit_upper <- 2.4
# labels_for_plot <- seq(round(10^limit_lower),round(10^limit_upper), 10)
labels_for_plot <- seq(round(10^limit_lower),round(10^limit_upper), 20)
breaks_for_plot <- log10(labels_for_plot)

# Wochentage
# Achsenbeschriftung anpassen
wt <- plot_model(Modell_lognorm_quad_TempxRegen, type = "pred",
                 terms = "Wochentag[all]",
                 title = "", axis.title = c("Wochentage",
                                            "Fussgänger*innen pro Tag"))
WTplot <- wt + scale_y_continuous(breaks = breaks_for_plot,
                                  labels = labels_for_plot,
                                  limits = c(limit_lower, limit_upper)) +
  theme_classic(base_size = 22) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1), legend.title = element_blank())

ggsave("Results/Wochentage.png", width=15, height=15, units="cm", dpi=1000)


# Phase
ph <- plot_model(Modell_lognorm_quad_TempxRegen, type = "pred",
                 terms = "Phase[all]",
                 title = "", axis.title = c("Phase",""))
Phaseplot <- ph + scale_y_continuous(breaks = breaks_for_plot,
                                     labels = labels_for_plot,
                                     limits = c(limit_lower, limit_upper)) +
  theme_classic(base_size = 22) +
```

```
  theme(axis.text.x = element_text(angle = 45, hjust = 1), legend.title = element_blank())

ggsave("Results/Phase.png", width=15, height=15, units="cm", dpi=1000)


# Ferien
f <- plot_model(Modell_lognorm_quad_TempxRegen, type = "pred",
                terms = "Ferien[all]",
                title = "", axis.title = c("Ferien",
                                           ""))
Ferienplot <- f + scale_x_continuous(breaks = c(0,1), labels = c("nein","ja")) +
  scale_y_continuous(breaks = breaks_for_plot,
                     labels = labels_for_plot,
                     limits = c(limit_lower, limit_upper)) +
  theme_classic(base_size = 22)

ggsave("Results/Ferien.png", width=15, height=15, units="cm", dpi=1000)

# Plots zusammenfuegen
wrap_plots(WTplot, Ferienplot, Phaseplot)

ggsave("Results/Zeit.png", width=35, height=17, units="cm", dpi=1000)

# Interaktion TempxRegen
labels_temperatur <- round(seq(floor(min(umwelt$Temp_max)), ceiling(max(umwelt$Temp_max)), length.out = 5), 0)
labels_regen <- round(seq(floor(min(umwelt$Regen)), ceiling(max(umwelt$Regen)), length.out = 4), 0)

limit_lower <- -1
limit_upper <- 2.05
labels_for_plot <- seq(round(10^limit_lower),round(10^limit_upper), 15)
breaks_for_plot <- log10(labels_for_plot)
breaks_for_plot[1] = -1

plot_model(Modell_lognorm_quad_TempxRegen, col = 1:3, type = "pred",
           terms = c("Regen_scaled", "Temp_max_scaled [-1,0,1]"), title = "", axis.title = c("Regen Halbtagssumme [mm]",
"Fussgaenger*innen pro Tag")) +
  scale_y_continuous(breaks = breaks_for_plot,
                     labels = labels_for_plot,
                     limits = c(0, limit_upper)) + scale_x_continuous(name = , breaks = c(0,4,8,12), labels = c(labels_regen),
limits = c(0, 12))+
  scale_colour_manual(values = c("lightblue", "orange", "coral1"),name = "Tagesmaximaltemperatur [°C]", labels =
c(paste(labels_temperatur[2], "°C"),paste(labels_temperatur[3], "°C"), paste(labels_temperatur[4], "°C")))+
  theme_classic(base_size = 22)+ scale_fill_manual(values = c("lightblue", "orange", "coral1")) +
  theme(legend.position = "bottom")


ggsave("Results/Interaktion.png", width=25, height=25, units="cm", dpi=1000)

#.############################################################################################
```