Supplementary material 2: The R code for one step fitting of the viability equation.

Dustin Wolkis, Angelino Carta, Shabnam Rezaei, Fiona R. Hay

This supplementary material demonstrates how to analyze seed longevity in R to fit the viability equation. Original data is from Rezaei et al. (2023), https://doi.org/10.1017/S0960258523000156.

```
# load libraries
library(tidyverse) # useful for all manor of data manipulation and
visualization
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse
2.0.0 ──
## ✓ dplyr     1.1.4     ✓ readr     2.1.5
## ✓ forcats   1.0.0     ✓ stringr   1.5.1
## ✓ ggplot2   3.5.0     ✓ tibble    3.2.1
## ✓ lubridate 1.9.3     ✓ tidyr     1.3.1
## ✓ purrr     1.0.2
## ── Conflicts ────────────────────────────────────────
tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all
conflicts to become errors
```

```
library(MASS) # the dose.p() function used to calculate P50 and P85
```

```
##
## Attaching package: 'MASS'
##
## The following object is masked from 'package:dplyr':
##
##     select
```

```
library(msm) # used for the deltamethod() function
library(gnlm) #"Generalized Nonlinear Regression Models" function bnlr() used
in the moisture models
```

```
## Loading required package: rmutil
##
## Attaching package: 'rmutil'
##
## The following object is masked from 'package:tidyr':
```

```
##
##     nesting
##
## The following object is masked from 'package:stats':
##
##     nobs
##
## The following objects are masked from 'package:base':
##
##     as.data.frame, units
```

```
library(knitr) # used to make tables in R Markdown

# set working directory
#setwd("my/file/path")

# Read in data
# brna short for the study organism, Brassica napus
brna <- read.csv("Rezaei_etal_2023_SSR.csv", header = TRUE)
# NB: MC = %f.wt.
glimpse(brna)
```

```
## Rows: 117
## Columns: 10
## $ Treatment    <chr> "Desorption 170", "Desorption 170", "Desorption 170",
"De…
## $ MC           <dbl> 9.141008, 9.141008, 9.141008, 9.141008, 8.000368,
8.00036…
## $ storage.RH   <dbl> 66.91, 66.91, 66.91, 66.91, 60.95, 60.95, 60.95,
60.95, 5…
## $ period       <int> 8, 17, 24, 27, 13, 24, 31, 41, 20, 37, 55, 90, 50,
59, 65…
## $ sown         <int> 90, 90, 90, 90, 87, 90, 90, 90, 90, 90, 90, 90, 90,
90, 9…
## $ germinated   <int> 89, 23, 3, 0, 83, 28, 10, 0, 84, 73, 14, 0, 73, 57,
51, 3…
## $ germ.percent <dbl> 98.888889, 25.555556, 3.333333, 0.000000, 95.402299,
31.1…
## $ sorption     <chr> "Desorption", "Desorption", "Desorption",
"Desorption", "…
## $ chamber.RH   <int> 70, 70, 70, 70, 60, 60, 60, 60, 50, 50, 50, 50, 40,
40, 4…
## $ cycle        <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, …
```

```
# citation for data set;
# Rezaei, S., Buitink, J. & Hay, F.R. (2023) Contrasting seed moisture
sorption
# behaviour between two species and the implication for seed longevity.
# Seed Science Research, 1–9. https://doi.org/10.1017/ S0960258523000156
```

```r
# Tell R which variables are factors
brna <- brna |>
  mutate(Treatment = as.factor(Treatment),
         sorption = as.factor(sorption),
         chamber.RH = as.factor(chamber.RH),
         cycle = as.factor(cycle))
glimpse(brna)
```

```
## Rows: 117
## Columns: 10
## $ Treatment    <fct> Desorption 170, Desorption 170, Desorption 170,
Desorptio…
## $ MC           <dbl> 9.141008, 9.141008, 9.141008, 9.141008, 8.000368,
8.00036…
## $ storage.RH   <dbl> 66.91, 66.91, 66.91, 66.91, 60.95, 60.95, 60.95,
60.95, 5…
## $ period       <int> 8, 17, 24, 27, 13, 24, 31, 41, 20, 37, 55, 90, 50,
59, 65…
## $ sown         <int> 90, 90, 90, 90, 87, 90, 90, 90, 90, 90, 90, 90, 90,
90, 9…
## $ germinated   <int> 89, 23, 3, 0, 83, 28, 10, 0, 84, 73, 14, 0, 73, 57,
51, 3…
## $ germ.percent <dbl> 98.888889, 25.555556, 3.333333, 0.000000, 95.402299,
31.1…
## $ sorption     <fct> Desorption, Desorption, Desorption, Desorption,
Desorptio…
## $ chamber.RH   <fct> 70, 70, 70, 70, 60, 60, 60, 60, 50, 50, 50, 50, 40,
40, 4…
## $ cycle        <fct> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, …
```

```r
# One stage fit
# fit data to the moisture relations of seed longevity equation (Ellis and
Roberts, 1980)


##### Fist demonstrate model for one treatment (or seed lot, etc)
# In this case we will use the desorption data (i.e.Desorption 170 Desorption
160, Desorption 150, Desorption 140, Desorption 130)
des <- droplevels(subset(brna, sorption == "Desorption" & cycle == "1"))
unique(des$Treatment)
```

```
## [1] Desorption 170 Desorption 160 Desorption 150 Desorption 140 Desorption
130
## 5 Levels: Desorption 130 Desorption 140 Desorption 150 ... Desorption 170
```

```r
# summarize MC grouped by Treatment MCs
des.sum <- des |>
  group_by(Treatment, MC, sorption, chamber.RH) |>
```

```r
  summarise() |>
  ungroup()
```

```
## `summarise()` has grouped output by 'Treatment', 'MC', 'sorption'. You can
## override using the `.groups` argument.
```

```r
attach(des)

### Description of parameters
# Ki = The intercept which is the fitted or theoretical initial viability of
the seed lot in probits.
# Kt = An 'intercept' or theoretical value that is constant for one seed lot
at a given temperature. One of the four "species constants".
# Cw = The effect of change in moisture content on longevity. One of the four
"species constants".

### one treatment (or seed lot, etc)
one.treatment.sv<- c(3.8, 4.1, 3.7) # initial parameters
one.treatment<- bnlr(y= cbind(germinated, sown - germinated), link =
"probit",
                     mu = ~ Ki - (1/10^(Kt-Cw *log10(MC))*period), ###please
see the additional brackets here
                     pmu = one.treatment.sv ,
                     fscale = 1,
                     print.level = 1,
                     typsize = abs(one.treatment.sv), #expected parameters
values at convergence (these may be specified
                     #differently from the starting values if needed)
                     ndigit = 10,
                     gradtol = 1e-05,
                     stepmax = 1 * sqrt(one.treatment.sv %*%
one.treatment.sv),
                     steptol = 1e-05, #this may be changed depending on the
warnings
                     iterlim = 1000
)
```

```
## iteration = 0
## Step:
## [1] 0 0 0
## Parameter:
## [1] 3.8 4.1 3.7
## Function Value
## [1] 881.9697
## Gradient:
## [1]  -300.2797 -2906.4191  2330.1638
##
## iteration = 24
## Parameter:
## [1] 3.797157 4.191536 3.759808
```

```
## Function Value
## [1] 820.9289
## Gradient:
## [1] -5.936673e-07  9.614666e-05 -1.461955e-04
##
## Relative gradient close to zero.
## Current iterate is probably solution.
```

```
# view model outputs including df, AIC, and coefficients (parameter
estimates)
print(one.treatment)
```

```
##
## Call:
## bnlr(y = cbind(germinated, sown - germinated), link = "probit",
##     mu = ~Ki - (1/10^(Kt - Cw * log10(MC)) * period), pmu =
one.treatment.sv,
##     fscale = 1, print.level = 1, typsize = abs(one.treatment.sv),
##     ndigit = 10, gradtol = 1e-05, stepmax = 1 * sqrt(one.treatment.sv %*%
##         one.treatment.sv), steptol = 1e-05, iterlim = 1000)
##
## binomial distribution
##
## Response: cbind(germinated, sown - germinated)
##
## Log likelihood function:
## {
##     m <- pnorm(mu1(p))
##     -sum(wt * (y[, 1] * log(m) + y[, 2] * log(1 - m)))
## }
##
## Location function:
## ~Ki - (1/10^(Kt - Cw * log10(MC)) * period)
##
## -Log likelihood    125.855
## Degrees of freedom 22
## AIC                128.855
## Iterations         24
##
## Location parameters:
##     estimate       se
## Ki     3.797  0.17778
## Kt     4.192  0.06201
## Cw     3.760  0.05985
##
## Correlations:
##         1       2       3
## 1  1.0000 -0.7440 -0.5393
## 2 -0.7440  1.0000  0.9613
## 3 -0.5393  0.9613  1.0000
```

```r
deviance(one.treatment) # view deviance
```

```
## [1] 251.7101
```

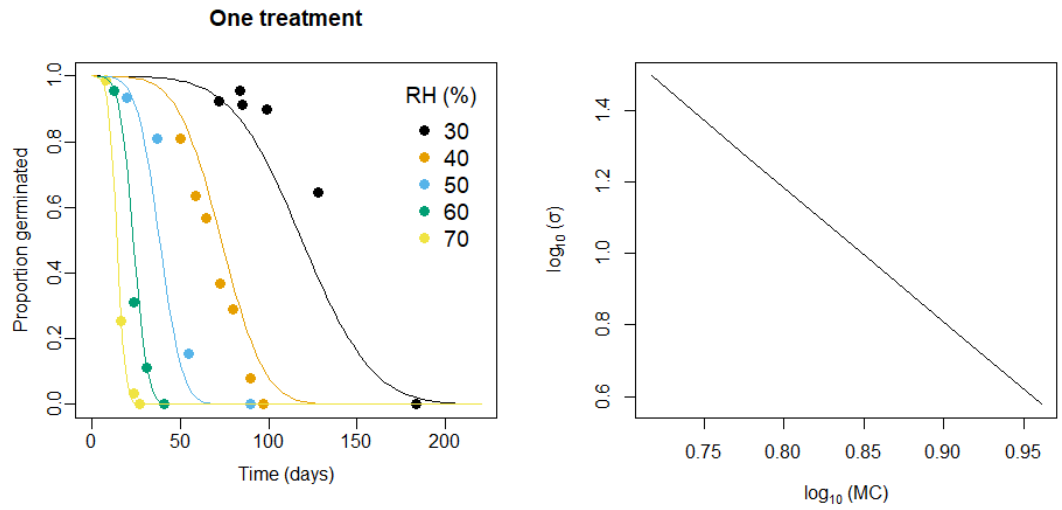
```r
# Estimate Ke based on universal temperature constants
Ke <- one.treatment$coefficients[2] + 45 * 0.0329 + 45 * 45 * 0.000478

# Table of coeficients
ot.coefs<-data.frame(
  Coefficient = c("Ki", "Kt","Cw", "Ke"),
  Estimate = c(one.treatment$coefficients, Ke),
  se = c(one.treatment$se, NA)
)
ot.coefs<- ot.coefs |> mutate(
  Estimate = round(Estimate, 2),
  se = round(se, 2)
)
knitr::kable(ot.coefs, row.names = FALSE)
```

Coefficient	Estimate	se
Ki	3.80	0.18
Kt	4.19	0.06
Cw	3.76	0.06
Ke	6.64	NA

```r
coefficients.for.plotting.M<-cbind(one.treatment$coefficients,
confint(one.treatment, level = 0.95))

vcov_reg <- vcov(one.treatment)
coef_reg <- coef(one.treatment)
coef_reg <- na.omit(coef_reg)
coef_reg <- as.numeric(coef_reg)

# estimate sigma (and CI) for each MC


sigmas.des<-c()
for (i in 1:5) {
  sigmas.des[i] <- 10^(coefficients.for.plotting.M[2,1]-
(coefficients.for.plotting.M[3,1] * log10(des.sum$MC[des.sum$Treatment[i]])))
}



sigmas.des.l <-c()
for (i in 1:5) {
  sigmas.des.l[i] <- 10^(coefficients.for.plotting.M[2,2]-
(coefficients.for.plotting.M[3,2] * log10(des.sum$MC[des.sum$Treatment[i]])))
}
```

```r
sigmas.des.u <-c()
for (i in 1:5) {
  sigmas.des.u[5] <- 10^(coefficients.for.plotting.M[2,3]-
(coefficients.for.plotting.M[3,3] * log10(des.sum$MC[des.sum$Treatment[i]])))
}



sigmas.des.SE<-c()
for (i in 1:5) {
  mcl<- des.sum$MC[des.sum$Treatment[i]]
  sigmas.des.SE[5] <-
    deltamethod(~ 10^(x3-(log10(mcl))), coef_reg, vcov_reg)
}

# put sigmas into one dataframe then join with des.sum
sigma.moist <- data.frame(
  Treatment = des.sum$Treatment,
  sigma = sigmas.des,
  `sigma Std. Error` = sigmas.des.SE,
  `sigma CI.l` = sigmas.des.l,
  `sigma CI.u` = sigmas.des.u,  check.names = FALSE
)

des.sum.0 <- left_join(des.sum, sigma.moist, by = "Treatment")

des.sum.0<- des.sum.0[order(des.sum.0$MC),]

# add p50s
des.sum.0$p50 <- coefficients.for.plotting.M[1,1] * des.sum.0$sigma

# add p50 confidence intervals
des.sum.0$`p50 CI.l`<- coefficients.for.plotting.M[1,2] * des.sum.0$`sigma
CI.l`
des.sum.0$`p50 CI.u`<- coefficients.for.plotting.M[1,3] * des.sum.0$`sigma
CI.u`

# Units are 'days' for sigma and p50, and '%f.wt.' for MC



# Plot
colours = palette.colors(palette = "Okabe-Ito")
plotting.data<- des[order(des$chamber.RH, decreasing = TRUE),]

par(mfrow=c(1,2))
```

```r
plot(plotting.data$period, plotting.data$germinated/plotting.data$sown,
xlab="Time (days)", ylab="Proportion germinated", xlim=c(0,220), ylim=c(0,1),
mgp=c(2,.5,0), pch= 16, cex=1.2, col=colours[1:5][unclass(des$chamber.RH)])

curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.sum.0$sigma [1]*x)),
add=TRUE, lty=1,lwd=1, col = colours[1])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.sum.0$sigma [2]*x)),
add=TRUE, lty=1,lwd=1, col = colours[2])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.sum.0$sigma [3]*x)),
add=TRUE, lty=1,lwd=1, col = colours[3])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.sum.0$sigma [4]*x)),
add=TRUE, lty=1,lwd=1, col = colours[4])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.sum.0$sigma [5]*x)),
add=TRUE, lty=1,lwd=1, col = colours[5])

legend(175, 1, title = "RH (%)", rev(unique(plotting.data$chamber.RH)),
pch=16, col = colours[1:5], bty = "n", cex=1.2)

title("One treatment")

plot(log10(des.sum.0$MC), log10(des.sum.0$sigma), xlab=
expression("log"[10]*" (MC)"), ylab= expression("log"[10]*" (σ)"), type="l")
```




```r
# Round to two decimal places and add units for reporting.
des.sum.0 <- des.sum.0 |>
  mutate(`MC (% f.wt.)` = round(MC, 2),
         `sigma (days)` = round(sigma, 2),
         `sigma Std. Error` = round(`sigma Std. Error`, 2),
         `sigma CI.l` = round(`sigma CI.l`, 2),
         `sigma CI.u` = round(`sigma CI.u`, 2),
         `p50 (days)` = round(p50),
         `p50 CI.l` = round(`p50 CI.l`, 2),
```

```
         `p50 CI.u` = round(`p50 CI.u`, 2)) |>
  dplyr::select(chamber.RH, `MC (% f.wt.)`, `sigma (days)`, `sigma Std.
Error`, `sigma CI.l`,
                `sigma CI.u`, `p50 (days)`, `p50 CI.l`, `p50 CI.u`)

# call the table.
# Units are 'days' for sigma and p50, and '%f.wt.' for MC
# CI.l and CI.u are upper and lower confidence intervals, respectively.
knitr::kable(des.sum.0, row.names = FALSE)
```

chamber.RH	MC (% f.wt.)	sigma (days)	sigma Std. Error	sigma CI.l	sigma CI.u	p50 (days)	p50 CI.l	p50 CI.u
30	5.21	31.31	NA	28.73	NA	119	99.07	NA
40	5.93	19.28	NA	17.96	NA	73	61.93	NA
50	7.03	10.15	NA	9.65	NA	39	33.28	NA
60	8.00	6.25	NA	6.03	NA	24	20.80	NA
70	9.14	3.79	86.72	3.71	3.87	14	12.80	16.02

```
detach(des)

#####################################################
###### Now demonstrate model for two treatments
# Use the full cycle 1 data set (i.e. Desorption 170, Desorption 160,
Desorption 150, Desorption 140, Desorption 130,
#   Adsorption 170, Adsorption 160, Adsorption 150, Adsorption 140,
Adsorption 130)
des.ads <- droplevels(subset(brna,  cycle == "1"))
unique(des.ads$Treatment)

##  [1] Desorption 170 Desorption 160 Desorption 150 Desorption 140
Desorption 130
##  [6] Adsorption 170 Adsorption 160 Adsorption 150 Adsorption 140
Adsorption 130
## 10 Levels: Adsorption 130 Adsorption 140 Adsorption 150 ... Desorption 170

# Add a dummy variable "d1" (0 for desorption, 1 for adsorption)
des.ads$d1 <- as.numeric(ifelse(des.ads$sorption == "Desorption", "0", "1"))

# summarize MC grouped by Treatment MCs
des.ads.sum <- des.ads |>
  group_by(Treatment, chamber.RH, MC, sorption) |>
  summarise() |>
  ungroup()

## `summarise()` has grouped output by 'Treatment', 'chamber.RH', 'MC'. You
can
## override using the `.groups` argument.
```

```r
# Attach the data frame
attach(des.ads)


### Explanation of variables ###
# Ki = The intercept which is the fitted or theoretical initial viability of
the seed lot in probits.
# Kt = An 'intercept' or theoretical value that is constant for one seed lot
at the constant storage temperature, t.
# Cw = The effect of change in moisture content on longevity (sigma).
# 'des' is short for the desorption treatment.
# 'ads' is short for the adsorption treatment.


### moisture models ###

# Ki constrained within each sorption cycle

# specify starting values
sv.i<- c(3.8, -0.5, 4.1, -0.5, 3.7, -0.5) # these are starting values, if you
change these here you should change stepmax following the warnings retunred
by the model

### Independent (independent Kt and Cw)
moisture.independent<- bnlr(y= cbind(germinated, sown - germinated), link =
"probit",
                            mu = ~ Kides + (Kiads*d1) -
(1/10^((Ktdes+(Ktads*d1))-(Cwdes + (Cwads*d1)) *log10(MC))*period), ###please
see the additional brackets here
                            pmu = sv.i,
                            fscale = 1,
                            print.level = 1,
                            typsize = abs(sv.i), #expected parameters values
at convergence (these may be specified
                            #differently from the starting values if needed)
                            ndigit = 10,
                            gradtol = 1e-05,
                            stepmax = 1 * sqrt(sv.i %*% sv.i),
                            steptol = 1e-05, #this may be changed depending
on the warnings
                            iterlim = 1000
)
```

```
## iteration = 0
## Step:
## [1] 0 0 0 0 0 0
## Parameter:
## [1]  3.8 -0.5  4.1 -0.5  3.7 -0.5
## Function Value
```

```
## [1] 3511.276
## Gradient:
## [1]  -1917.993  -1617.729 -21673.872 -18770.050  16733.004  14401.438
##
## iteration = 49
## Parameter:
## [1]  3.82704867 -0.32954365  4.18519736 -0.01457359  3.75605347 -
0.06376494
## Function Value
## [1] 1767.421
## Gradient:
## [1]  0.012310104 -1.296377733 -0.018984645  0.279408260  0.005284098
## [6]  0.186459976
##
## Successive iterates within tolerance.
## Current iterate is probably solution.
```

```
# view model outputs including df, AIC, and coefficients (parameter
estimates)
print(moisture.independent)
```

```
##
## Call:
## bnlr(y = cbind(germinated, sown - germinated), link = "probit",
##     mu = ~Kides + (Kiads * d1) - (1/10^((Ktdes + (Ktads * d1)) -
##         (Cwdes + (Cwads * d1)) * log10(MC)) * period), pmu = sv.i,
##     fscale = 1, print.level = 1, typsize = abs(sv.i), ndigit = 10,
##     gradtol = 1e-05, stepmax = 1 * sqrt(sv.i %*% sv.i), steptol = 1e-05,
##     iterlim = 1000)
##
## binomial distribution
##
## Response: cbind(germinated, sown - germinated)
##
## Log likelihood function:
## {
##     m <- pnorm(mu1(p))
##     -sum(wt * (y[, 1] * log(m) + y[, 2] * log(1 - m)))
## }
##
## Location function:
## ~Kides + (Kiads * d1) - (1/10^((Ktdes + (Ktads * d1)) - (Cwdes +
##     (Cwads * d1)) * log10(MC)) * period)
##
## -Log likelihood    326.3916
## Degrees of freedom 46
## AIC                332.3916
## Iterations         49
##
## Location parameters:
```

```
##        estimate       se
## Kides   3.82705  0.18195
## Kiads  -0.32954  0.20290
## Ktdes   4.18520  0.06395
## Ktads  -0.01457  0.07101
## Cwdes   3.75605  0.06161
## Cwads  -0.06376  0.07438
##
## Correlations:
##         1       2       3       4       5       6
## 1  1.0000 -0.7838 -0.7544  0.6214 -0.5593  0.4429
## 2 -0.7838  1.0000  0.5146 -0.5941  0.3417 -0.3220
## 3 -0.7544  0.5146  1.0000 -0.8179  0.9636 -0.7695
## 4  0.6214 -0.5941 -0.8179  1.0000 -0.7863  0.9491
## 5 -0.5593  0.3417  0.9636 -0.7863  1.0000 -0.8001
## 6  0.4429 -0.3220 -0.7695  0.9491 -0.8001  1.0000
```

```
deviance(moisture.independent) # view deviance
```

```
## [1] 652.7832
```

```
# Estimate Ke based on universal temperature constants
Ke.des <- moisture.independent$coefficients[3] + 45 * 0.0329 + 45 * 45 *
0.000478 # desorption
Ke.ads <- moisture.independent$coefficients[3] +
moisture.independent$coefficients[4] + 45 * 0.0329 + 45 * 45 * 0.000478 #
adsorption

# Table of coeficients
mi.coefs<-data.frame(
  Treatment = c(rep(c("Desorption", "Adsorption"), 4)),
  Coefficient = c(rep("Ki", 2), rep("Kt", 2), rep("Cw", 2), rep("Ke", 2)),
  Estimate = c(moisture.independent$coefficients[1],
moisture.independent$coefficients[1] + moisture.independent$coefficients[2],
moisture.independent$coefficients[3], moisture.independent$coefficients[3] +
moisture.independent$coefficients[4], moisture.independent$coefficients[5],
moisture.independent$coefficients[5] + moisture.independent$coefficients[6],
Ke.des, Ke.ads),
  se = c(moisture.independent$se, NA, NA)
  )
mi.coefs<- mi.coefs |> mutate(
  Estimate = round(Estimate, 2),
  se = round(se, 2)
)
knitr::kable(mi.coefs, row.names = FALSE)
```

Treatment	Coefficient	Estimate	se
Desorption	Ki	3.83	0.18
Adsorption	Ki	3.50	0.20

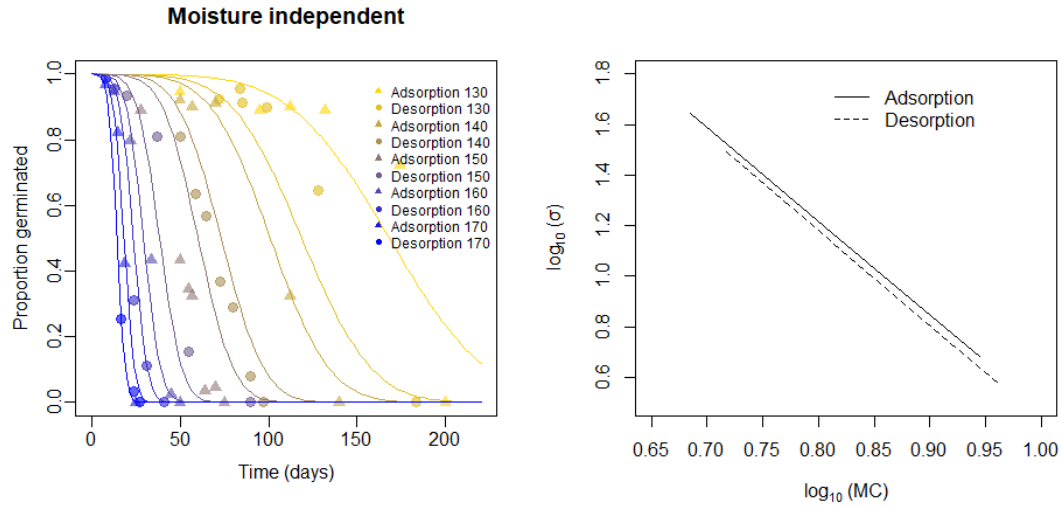
Treatment	Coefficient	Estimate	se
Desorption	Kt	4.19	0.06
Adsorption	Kt	4.17	0.07
Desorption	Cw	3.76	0.06
Adsorption	Cw	3.69	0.07
Desorption	Ke	6.63	NA
Adsorption	Ke	6.62	NA

```
coefficients.for.plotting.M<-cbind(moisture.independent$coefficients,
confint(moisture.independent, level = 0.95))

vcov_reg <- vcov(moisture.independent)
coef_reg <- coef(moisture.independent)
coef_reg <- na.omit(coef_reg)
coef_reg <- as.numeric(coef_reg)

# estimate sigma (and CI) for each MC and sorption cycle
sigmas.ads<-c()
for (i in 1:5) {
  sigmas.ads[i] <- 10^(coefficients.for.plotting.M[3,1] +
coefficients.for.plotting.M[4,1] - ((coefficients.for.plotting.M[5,1]+
coefficients.for.plotting.M[6,1]) *
log10(des.ads.sum$MC[des.ads.sum$Treatment[i]])))
}

sigmas.des<-c()
for (i in 6:10) {
  sigmas.des[i-5] <- 10^(coefficients.for.plotting.M[3,1]-
(coefficients.for.plotting.M[5,1] *
log10(des.ads.sum$MC[des.ads.sum$Treatment[i]])))
}

sigmas.ads.l <-c()
for (i in 1:5) {
  sigmas.ads.l[i] <- 10^(coefficients.for.plotting.M[3,2] +
coefficients.for.plotting.M[4,2] - ((coefficients.for.plotting.M[5,2]+
coefficients.for.plotting.M[6,2]) *
log10(des.ads.sum$MC[des.ads.sum$Treatment[i]])))
}

sigmas.des.l <-c()
for (i in 6:10) {
  sigmas.des.l[i-5] <- 10^(coefficients.for.plotting.M[3,2]-
(coefficients.for.plotting.M[5,2] *
log10(des.ads.sum$MC[des.ads.sum$Treatment[i]])))
}

sigmas.ads.u <-c()
```

```r
for (i in 1:5) {
  sigmas.ads.u[i] <- 10^(coefficients.for.plotting.M[3,3] +
coefficients.for.plotting.M[4,3] - ((coefficients.for.plotting.M[5,3]+
coefficients.for.plotting.M[6,3]) *
log10(des.ads.sum$MC[des.ads.sum$Treatment[i]])))
}

sigmas.des.u <-c()
for (i in 6:10) {
  sigmas.des.u[i-5] <- 10^(coefficients.for.plotting.M[3,3]-
(coefficients.for.plotting.M[5,3] *
log10(des.ads.sum$MC[des.ads.sum$Treatment[i]])))
}

sigmas.ads.SE<-c()
for (i in 1:5) {
  mcl<- des.ads.sum$MC[des.ads.sum$Treatment[i]]
  sigmas.ads.SE[i] <-
    deltamethod(~ 10^(x3-x4-((x5+x6) *log10(mcl))), coef_reg, vcov_reg)
}

sigmas.des.SE<-c()
for (i in 6:10) {
  mcl<- des.ads.sum$MC[des.ads.sum$Treatment[i]]
  sigmas.des.SE[i-5] <-
    deltamethod(~ 10^(x3-( x5 *log10(mcl))), coef_reg, vcov_reg)
}

# put sigmas into one dataframe then join with des.ads.sum
sigma.moist <- data.frame(
  Treatment = des.ads.sum$Treatment,
  sigma = c(sigmas.ads, sigmas.des),
  `sigma Std. Error` = c(sigmas.ads.SE, sigmas.des.SE),
  `sigma CI.l` = c(sigmas.ads.l, sigmas.des.l),
  `sigma CI.u` = c(sigmas.ads.u, sigmas.des.u),  check.names = FALSE
)
des.ads.sum.1 <- left_join(des.ads.sum, sigma.moist, by = "Treatment")
des.ads.sum.1<- as.data.frame(des.ads.sum.1[complete.cases(des.ads.sum.1),])
des.ads.sum.1<- des.ads.sum.1[order(des.ads.sum.1$MC),]

# add p50s
des.ads.sum.1$p50 <- coefficients.for.plotting.M[1,1] * des.ads.sum.1$sigma

# add p50s confidence intervals
des.ads.sum.1$`p50 CI.l`<- coefficients.for.plotting.M[1,2] *
des.ads.sum.1$`sigma CI.l`
des.ads.sum.1$`p50 CI.u`<- coefficients.for.plotting.M[1,2] *
des.ads.sum.1$`sigma CI.u`
```

```r
# The units for sigma and p50 is days. MC = %f.wt.

# Plot
colfunc <- colorRampPalette(c("gold", "blue"))
colours<- colfunc(10)
plotting.data<- des.ads[order(des.ads$MC),]
plotting.data$Treatment <- factor(plotting.data$Treatment,
levels=c('Adsorption 130', 'Desorption 130', 'Adsorption 140', 'Desorption
140', 'Adsorption 150', 'Desorption 150', 'Adsorption 160', 'Desorption 160',
'Adsorption 170', 'Desorption 170' ))

par(mfrow=c(1,2))
plot(plotting.data$period, plotting.data$germinated/plotting.data$sown,
xlab="Time (days)", ylab="Proportion germinated", xlim=c(0,220), ylim=c(0,1),
mgp=c(2,.5,0), pch= c(17, 19)[as.numeric(plotting.data$sorption)], cex=1.2,
col= alpha(colours, 0.6)[plotting.data$Treatment])

curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/
des.ads.sum.1$sigma[1]*x)), add=TRUE, lty=1,lwd=1, col = colours[1])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.1$sigma
[2]*x)), add=TRUE, lty=1,lwd=1, col = colours[2])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.1$sigma
[3]*x)), add=TRUE, lty=1,lwd=1, col = colours[3])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.1$sigma
[4]*x)), add=TRUE, lty=1,lwd=1, col = colours[4])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.1$sigma
[5]*x)), add=TRUE, lty=1,lwd=1, col = colours[5])

curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.1$sigma
[6]*x)), add=TRUE, lty=1,lwd=1, col = colours[6])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.1$sigma
[7]*x)), add=TRUE, lty=1,lwd=1, col = colours[7])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.1$sigma
[8]*x)), add=TRUE, lty=1,lwd=1, col = colours[8])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.1$sigma
[9]*x)), add=TRUE, lty=1,lwd=1, col = colours[9])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.1$sigma
[10]*x)), add=TRUE, lty=1,lwd=1, col = colours[10])

legend(155, 1, unique(plotting.data$Treatment),col=colours, pch=rep(c(17,19),
5), bty = "n", cex=0.75)
title("Moisture independent")

plot(log10(subset(des.ads.sum.1, sorption=="Adsorption")$MC[1:5]),
log10(subset(des.ads.sum.1, sorption=="Adsorption")$sigma[1:5]), type = "l",
lty = 1, xlab= expression("log"[10]*" (MC)"), ylab= expression("log"[10]*"
(σ)"), xlim=c(0.65,1), ylim=c(0.5,1.8))
lines(log10(subset(des.ads.sum.1, sorption=="Desorption")$MC[1:5]),
log10(subset(des.ads.sum.1, sorption=="Desorption")$sigma[1:5]), type = "l",
```

```
lty = 2)
legend(0.8, 1.8, legend = c("Adsorption", "Desorption"),lty = c(1, 2), bty =
"n")
```




```
# Round to two decimal places and add units for reporting.
des.ads.sum.1 <- des.ads.sum.1 |>
  mutate(`MC (% f.wt.)` = round(MC, 2),
         `sigma (days)` = round(sigma, 2),
         `sigma Std. Error` = round(`sigma Std. Error`, 2),
         `sigma CI.l` = round(`sigma CI.l`, 2),
         `sigma CI.u` = round(`sigma CI.u`, 2),
         `p50 (days)` = round(p50),
         `p50 CI.l` = round(`p50 CI.l`, 2),
         `p50 CI.u` = round(`p50 CI.u`, 2)) |>
  dplyr::select(chamber.RH, `MC (% f.wt.)`, sorption, `sigma (days)`, `sigma
Std. Error`, `sigma CI.l`,
                `sigma CI.u`, `p50 (days)`, `p50 CI.l`, `p50 CI.u`)

# call the table.
# Units are 'days' for sigma and p50, and '%f.wt.' for MC.
knitr::kable(des.ads.sum.1, row.names = FALSE)
```

chamber.RH	MC (% f.wt.)	sorption	sigma (days)	sigma Std. Error	sigma CI.l	sigma CI.u	p50 (days)	p50 CI.l	p50 CI.u
30	4.84	Adsorption	43.92	15.09	36.36	53.04	168	126.18	184.08
30	5.21	Desorption	31.05	1.75	28.40	33.95	119	98.56	117.81
40	5.55	Adsorption	26.38	9.14	22.66	30.71	101	78.63	106.58
40	5.93	Desor	19.13	0.97	17.77	20.59	73	61.6	71.4

chamber.RH	MC (% f.wt.)	sorption	sigma (days)	sigma Std. Error	sigma CI.l	sigma CI.u	p50 (days)	p50 CI.l	p50 CI.u
		ption						7	6
50	6.39	Adsorption	15.73	5.50	14.03	17.65	60	48.68	61.24
50	7.03	Desorption	10.08	0.45	9.56	10.63	39	33.17	36.89
60	7.75	Adsorption	7.70	2.73	7.23	8.20	29	25.08	28.46
60	8.00	Desorption	6.21	0.26	5.98	6.45	24	20.76	22.37
70	8.81	Adsorption	4.80	1.72	4.67	4.95	18	16.19	17.17
70	9.14	Desorption	3.76	0.15	3.68	3.85	14	12.79	13.35

```
### Common intercept (Common Kt)
sv.ci<- c(3.8, -0.5, 4.1, 3.7, 0) # initial parameters
moisture.com.inter<- bnlr(y= cbind(germinated, sown - germinated), link =
"probit",
                          mu = ~ Kides + (Kiads*d1) - (1/10^(Kt-(Cwdes +
Cwads*d1) *log10(MC))*period), ###please see the additional brackets here
                          pmu = sv.ci,
                          fscale = 1,
                          print.level = 1,
                          typsize = abs(sv.ci), #expected parameters values
at convergence (these may be specified
                          #differently from the starting values if needed)
                          ndigit = 10,
                          gradtol = 1e-05,
                          stepmax = 1 * sqrt(sv.ci %*% sv.ci),
                          steptol = 1e-05, #this may be changed depending on
the warnings
                          iterlim = 1000
)

## iteration = 0
## Step:
## [1] 0 0 0 0 0
## Parameter:
## [1]  3.8 -0.5  4.1  3.7  0.0
## Function Value
## [1] 2163.953
## Gradient:
## [1] -1050.9432  -750.6783 -9975.1375  8008.7123  5677.8054
##
## iteration = 33
## Parameter:
```

```
## [1]  3.92681909 -0.49486543  4.17432236  3.75586142 -0.06851892
## Function Value
## [1] 1768.037
## Gradient:
## [1]  0.1939986 -6.3966156  0.1586384 -0.1802419 -0.2492885
##
## Last global step failed to locate a point lower than x.
## Either x is an approximate local minimum of the function,
## the function is too non-linear for this algorithm,
## or steptol is too large.
```

```
# view model outputs including df, AIC, and coefficients (parameter
estimates)
print(moisture.com.inter)
```

```
##
## Call:
## bnlr(y = cbind(germinated, sown - germinated), link = "probit",
##     mu = ~Kides + (Kiads * d1) - (1/10^(Kt - (Cwdes + Cwads *
##         d1) * log10(MC)) * period), pmu = sv.ci, fscale = 1,
##     print.level = 1, typsize = abs(sv.ci), ndigit = 10, gradtol = 1e-05,
##     stepmax = 1 * sqrt(sv.ci %*% sv.ci), steptol = 1e-05, iterlim = 1000)
##
## Warning: no convergence - error 3
##
## binomial distribution
##
## Response: cbind(germinated, sown - germinated)
##
## Log likelihood function:
## {
##     m <- pnorm(mu1(p))
##     -sum(wt * (y[, 1] * log(m) + y[, 2] * log(1 - m)))
## }
##
## Location function:
## ~Kides + (Kiads * d1) - (1/10^(Kt - (Cwdes + Cwads * d1) * log10(MC)) *
##     period)
##
## -Log likelihood    327.0077
## Degrees of freedom 47
## AIC                332.0077
## Iterations         33
##
## Location parameters:
##        estimate       se
## Kides   3.92682  0.14631
## Kiads  -0.49487  0.16372
## Kt      4.17432  0.03693
## Cwdes   3.75586  0.03787
```

```
## Cwads  -0.06852  0.02332
##
## Correlations:
##         1       2        3       4        5
## 1  1.0000 -0.6783 -0.56074 -0.1654 -0.59556
## 2 -0.6783  1.0000  0.10181 -0.2173  0.95277
## 3 -0.5607  0.1018  1.00000  0.9026  0.05831
## 4 -0.1654 -0.2173  0.90264  1.0000 -0.25340
## 5 -0.5956  0.9528  0.05831 -0.2534  1.00000
```

```
deviance(moisture.com.inter) # view deviance
```

```
## [1] 654.0153
```

```
# Estimate Ke based on universal temperature constants
Ke <- moisture.com.inter$coefficients[3] + 45 * 0.0329 + 45 * 45 * 0.000478

# Table of coeficients
mci.coefs<-data.frame(
  Treatment = c(rep(c("Desorption", "Adsorption"), 4)),
  Coefficient = c(rep("Ki", 2), rep("Kt", 2), rep("Cw", 2), rep("Ke", 2)),
  Estimate = c(moisture.com.inter$coefficients[1],
moisture.com.inter$coefficients[1] + moisture.com.inter$coefficients[2],
rep(moisture.com.inter$coefficients[3], 2),
moisture.com.inter$coefficients[4], moisture.com.inter$coefficients[4] +
moisture.com.inter$coefficients[5], rep(Ke, 2)),
  se = c(moisture.com.inter$se[1], moisture.com.inter$se[2],
rep(moisture.com.inter$se[3], 2), moisture.com.inter$se[4],
moisture.com.inter$se[5], NA, NA)
)
mci.coefs<- mci.coefs |> mutate(
  Estimate = round(Estimate, 2),
  se = round(se, 2)
)
knitr::kable(mci.coefs, row.names = FALSE)
```

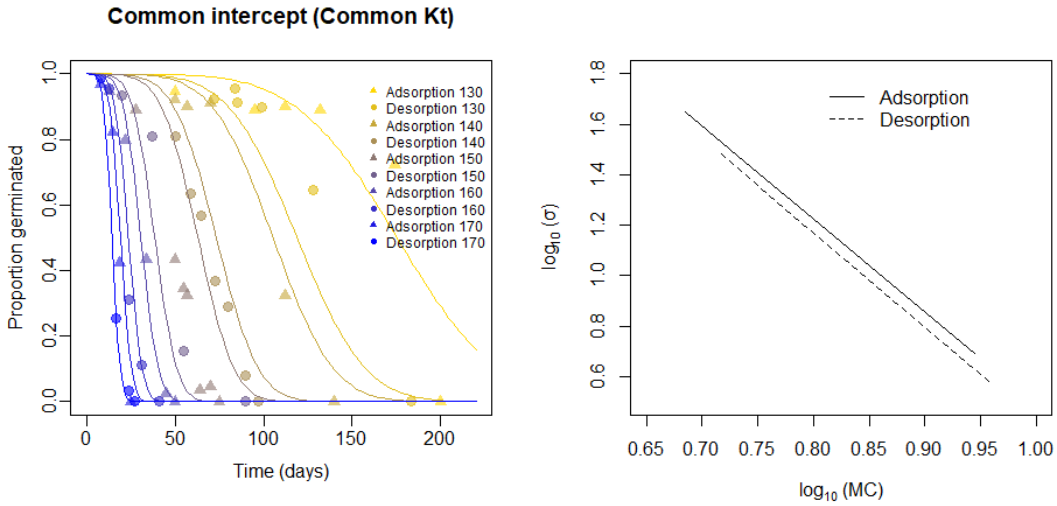
Treatment	Coefficient	Estimate	se
Desorption	Ki	3.93	0.15
Adsorption	Ki	3.43	0.16
Desorption	Kt	4.17	0.04
Adsorption	Kt	4.17	0.04
Desorption	Cw	3.76	0.04
Adsorption	Cw	3.69	0.02
Desorption	Ke	6.62	NA
Adsorption	Ke	6.62	NA

```
coefficients.for.plotting.M<-cbind(moisture.com.inter$coefficients,
confint(moisture.com.inter, level = 0.95))
```

```r
vcov_reg <- vcov(moisture.com.inter)
coef_reg <- coef(moisture.com.inter)
coef_reg <- na.omit(coef_reg)
coef_reg <- as.numeric(coef_reg)

# estimate sigma (and CI) for each MC and sorption cycle
sigmas.ads<-c()
for (i in 1:5) {
  sigmas.ads[i] <- 10^(coefficients.for.plotting.M[3,1] -
((coefficients.for.plotting.M[4,1]+ coefficients.for.plotting.M[5,1]) *
log10(des.ads.sum$MC[des.ads.sum$Treatment[i]])))
}

sigmas.des<-c()
for (i in 6:10) {
  sigmas.des[i-5] <- 10^(coefficients.for.plotting.M[3,1]-
(coefficients.for.plotting.M[4,1] *
log10(des.ads.sum$MC[des.ads.sum$Treatment[i]])))
}

sigmas.ads.l <-c()
for (i in 1:5) {
  sigmas.ads.l[i] <- 10^(coefficients.for.plotting.M[3,2] -
((coefficients.for.plotting.M[4,2]+ coefficients.for.plotting.M[5,2]) *
log10(des.ads.sum$MC[des.ads.sum$Treatment[i]])))
}

sigmas.des.l <-c()
for (i in 6:10) {
  sigmas.des.l[i-5] <- 10^(coefficients.for.plotting.M[3,2]-
(coefficients.for.plotting.M[4,2] *
log10(des.ads.sum$MC[des.ads.sum$Treatment[i]])))
}

sigmas.ads.u <-c()
for (i in 1:5) {
  sigmas.ads.u[i] <- 10^(coefficients.for.plotting.M[3,3] -
((coefficients.for.plotting.M[4,3]+ coefficients.for.plotting.M[5,3]) *
log10(des.ads.sum$MC[des.ads.sum$Treatment[i]])))
}

sigmas.des.u <-c()
for (i in 6:10) {
  sigmas.des.u[i-5] <- 10^(coefficients.for.plotting.M[3,3]-
(coefficients.for.plotting.M[4,3] *
log10(des.ads.sum$MC[des.ads.sum$Treatment[i]])))
}
```

```r
sigmas.ads.SE<-c()
for (i in 1:5) {
  mcl<- des.ads.sum$MC[des.ads.sum$Treatment[i]]
  sigmas.ads.SE[i] <-
    deltamethod(~ 10^(x3-((x4+x5) *log10(mcl))), coef_reg, vcov_reg)
}

sigmas.des.SE<-c()
for (i in 6:10) {
  mcl<- des.ads.sum$MC[des.ads.sum$Treatment[i]]
  sigmas.des.SE[i-5] <-
    deltamethod(~ 10^(x3-( x4 *log10(mcl))), coef_reg, vcov_reg)
}

# put sigmas into one dataframe then join with des.ads.sum
sigma.moist <- data.frame(
  Treatment = des.ads.sum$Treatment,
  sigma = c(sigmas.ads, sigmas.des),
  `sigma Std. Error` = c(sigmas.ads.SE, sigmas.des.SE),
  `sigma CI.l` = c(sigmas.ads.l, sigmas.des.l),
  `sigma CI.u` = c(sigmas.ads.u, sigmas.des.u),  check.names = FALSE
)
des.ads.sum.2 <- left_join(des.ads.sum, sigma.moist, by = "Treatment")
des.ads.sum.2<- as.data.frame(des.ads.sum.2[complete.cases(des.ads.sum.2),])
des.ads.sum.2<- des.ads.sum.2[order(des.ads.sum.2$MC),]

# add p50s
des.ads.sum.2$p50 <- coefficients.for.plotting.M[1,1] * des.ads.sum.2$sigma

# add p50s confidence intervals
des.ads.sum.2$`p50 CI.l`<- coefficients.for.plotting.M[1,2] *
des.ads.sum.2$`sigma CI.l`
des.ads.sum.2$`p50 CI.u`<- coefficients.for.plotting.M[1,3] *
des.ads.sum.2$`sigma CI.u`

# Units are 'days' for sigma and p50, and '%f.wt.' for MC


# Plot
colfunc <- colorRampPalette(c("gold", "blue"))
colours<- colfunc(10)
plotting.data<- des.ads[order(des.ads$MC),]
plotting.data$Treatment <- factor(plotting.data$Treatment,
levels=c('Adsorption 130', 'Desorption 130', 'Adsorption 140', 'Desorption
140', 'Adsorption 150', 'Desorption 150', 'Adsorption 160', 'Desorption 160',
'Adsorption 170', 'Desorption 170' ))

par(mfrow=c(1,2))
plot(plotting.data$period, plotting.data$germinated/plotting.data$sown,
```

```
xlab="Time (days)", ylab="Proportion germinated", xlim=c(0,220), ylim=c(0,1),
mgp=c(2,.5,0), pch= c(17, 19)[as.numeric(plotting.data$sorption)], cex=1.2,
col= alpha(colours, 0.6)[plotting.data$Treatment])

curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/
des.ads.sum.2$sigma[1]*x)), add=TRUE, lty=1,lwd=1, col = colours[1])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.2$sigma
[2]*x)), add=TRUE, lty=1,lwd=1, col = colours[2])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.2$sigma
[3]*x)), add=TRUE, lty=1,lwd=1, col = colours[3])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.2$sigma
[4]*x)), add=TRUE, lty=1,lwd=1, col = colours[4])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.2$sigma
[5]*x)), add=TRUE, lty=1,lwd=1, col = colours[5])

curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.2$sigma
[6]*x)), add=TRUE, lty=1,lwd=1, col = colours[6])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.2$sigma
[7]*x)), add=TRUE, lty=1,lwd=1, col = colours[7])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.2$sigma
[8]*x)), add=TRUE, lty=1,lwd=1, col = colours[8])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.2$sigma
[9]*x)), add=TRUE, lty=1,lwd=1, col = colours[9])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.2$sigma
[10]*x)), add=TRUE, lty=1,lwd=1, col = colours[10])

legend(155, 1, unique(plotting.data$Treatment),col=colours, pch=rep(c(17,19),
5), bty = "n", cex=0.75)
title("Common intercept (Common Kt)")

plot(log10(subset(des.ads.sum.2, sorption=="Adsorption")$MC[1:5]),
log10(subset(des.ads.sum.2, sorption=="Adsorption")$sigma[1:5]), type = "l",
lty = 1, xlab= expression("log"[10]*" (MC)"), ylab= expression("log"[10]*"
(σ)"), xlim=c(0.65,1), ylim=c(0.5,1.8))
lines(log10(subset(des.ads.sum.2, sorption=="Desorption")$MC[1:5]),
log10(subset(des.ads.sum.2, sorption=="Desorption")$sigma[1:5]), type = "l",
lty = 2)
legend(0.8, 1.8, legend = c("Adsorption", "Desorption"),lty = c(1, 2), bty =
"n")
```




```
# Round to two decimal places and add units for reporting.
des.ads.sum.2 <- des.ads.sum.2 |>
  mutate(`MC (% f.wt.)` = round(MC, 2),
         `sigma (days)` = round(sigma, 2),
         `sigma Std. Error` = round(`sigma Std. Error`, 2),
         `sigma CI.l` = round(`sigma CI.l`, 2),
         `sigma CI.u` = round(`sigma CI.u`, 2),
         `p50 (days)` = round(p50),
         `p50 CI.l` = round(`p50 CI.l`, 2),
         `p50 CI.u` = round(`p50 CI.u`, 2)) |>
  dplyr::select(chamber.RH, `MC (% f.wt.)`, sorption, `sigma (days)`, `sigma
Std. Error`, `sigma CI.l`,
                `sigma CI.u`, `p50 (days)`, `p50 CI.l`, `p50 CI.u`)

# call the table.
# Units are 'days' for sigma and p50, and '%f.wt.' for MC
knitr::kable(des.ads.sum.2, row.names = FALSE)
```

chamber.RH	MC (% f.wt.)	sorption	sigma (days)	sigma Std. Error	sigma CI.l	sigma CI.u	p50 (days)	p50 CI.l	p50 CI.u
30	4.84	Adsorption	44.64	1.73	45.65	43.65	175	166.17	183.91
30	5.21	Desorption	30.29	1.19	28.98	31.66	119	105.51	133.39
40	5.55	Adsorption	26.83	0.99	27.90	25.80	105	101.55	108.73
40	5.93	Desorption	18.66	0.70	18.03	19.32	73	65.62	81.39
50	6.39	Adsorption	16.01	0.57	16.93	15.14	63	61.63	63.81

chamber.RH	MC (% f.wt.)	sorption	sigma (days)	sigma Std. Error	sigma CI.l	sigma CI.u	p50 (days)	p50 CI.l	p50 CI.u
50	7.03	Desorption	9.83	0.36	9.62	10.05	39	35.02	42.36
60	7.75	Adsorption	7.84	0.28	8.49	7.25	31	30.89	30.54
60	8.00	Desorption	6.06	0.22	5.98	6.13	24	21.78	25.84
70	8.81	Adsorption	4.90	0.18	5.38	4.46	19	19.59	18.78
70	9.14	Desorption	3.67	0.14	3.66	3.68	14	13.34	15.51

```
### Common slope (common Cw)
sv.cs<- c(3.8, -0.5, 4.1, 0, 3.7) # initial parameters
moisture.com.slope<- bnlr(y= cbind(germinated, sown - germinated), link =
"probit",
                          mu = ~ Kides + (Kiads*d1) -
(1/10^((Ktdes+(Ktads*d1))-Cw *log10(MC))*period), ###please see the
additional brackets here
                          pmu = sv.cs,
                          fscale = 1,
                          print.level = 1,
                          typsize = abs(sv.cs), #expected parameters values
at convergence (these may be specified
                          #differently from the starting values if needed)
                          ndigit = 10,
                          gradtol = 1e-05,
                          stepmax = 1 * sqrt(sv.cs %*% sv.cs),
                          steptol = 1e-05, #this may be changed depending on
the warnings
                          iterlim = 1000
)
```

```
## iteration = 0
## Step:
## [1] 0 0 0 0 0
## Parameter:
## [1]  3.8 -0.5  4.1  0.0  3.7
## Function Value
## [1] 2163.953
## Gradient:
## [1] -1050.9432  -750.6783 -9975.1375 -7070.0278  8008.7123
##
## iteration = 33
## Parameter:
## [1]  3.93214518 -0.49609414  4.14344106  0.05568035  3.71864920
## Function Value
```

```
## [1] 1768.34
## Gradient:
## [1]  0.09567404 -4.85369438  0.01427822  0.12110436 -0.05400259
##
## Last global step failed to locate a point lower than x.
## Either x is an approximate local minimum of the function,
## the function is too non-linear for this algorithm,
## or steptol is too large.
```

```
# view model outputs including df, AIC, and coefficients (parameter
estimates)
print(moisture.com.slope)
```

```
##
## Call:
## bnlr(y = cbind(germinated, sown - germinated), link = "probit",
##     mu = ~Kides + (Kiads * d1) - (1/10^((Ktdes + (Ktads * d1)) -
##         Cw * log10(MC)) * period), pmu = sv.cs, fscale = 1, print.level =
1,
##     typsize = abs(sv.cs), ndigit = 10, gradtol = 1e-05, stepmax = 1 *
##         sqrt(sv.cs %*% sv.cs), steptol = 1e-05, iterlim = 1000)
##
## Warning: no convergence - error 3
##
## binomial distribution
##
## Response: cbind(germinated, sown - germinated)
##
## Log likelihood function:
## {
##     m <- pnorm(mu1(p))
##     -sum(wt * (y[, 1] * log(m) + y[, 2] * log(1 - m)))
## }
##
## Location function:
## ~Kides + (Kiads * d1) - (1/10^((Ktdes + (Ktads * d1)) - Cw *
##     log10(MC)) * period)
##
## -Log likelihood    327.3103
## Degrees of freedom 47
## AIC                332.3103
## Iterations         33
##
## Location parameters:
##        estimate       se
## Kides   3.93215  0.16662
## Kiads  -0.49609  0.19487
## Ktdes   4.14344  0.04060
## Ktads   0.05568  0.02246
## Cw      3.71865  0.03692
```

```
##
## Correlations:
##         1       2       3       4       5
## 1  1.0000 -0.7651 -0.7172  0.7057 -0.3717
## 2 -0.7651  1.0000  0.4384 -0.9668  0.1367
## 3 -0.7172  0.4384  1.0000 -0.4156  0.9073
## 4  0.7057 -0.9668 -0.4156  1.0000 -0.1188
## 5 -0.3717  0.1367  0.9073 -0.1188  1.0000
```

```
deviance(moisture.com.slope) # view deviance
```

```
## [1] 654.6206
```

```
# Estimate Ke based on universal temperature constants
Ke.des <- moisture.com.slope$coefficients[3] + 45 * 0.0329 + 45 * 45 *
0.000478 # desorption
Ke.ads <- moisture.com.slope$coefficients[3] +
moisture.com.slope$coefficients[4] + 45 * 0.0329 + 45 * 45 * 0.000478 #
adsorption

# Table of coeficients
mcs.coefs<-data.frame(
  Treatment = c(rep(c("Desorption", "Adsorption"), 4)),
  Coefficient = c(rep("Ki", 2), rep("Kt", 2), rep("Cw", 2), rep("Ke", 2)),
  Estimate = c(moisture.com.slope$coefficients[1],
moisture.com.slope$coefficients[1] + moisture.com.slope$coefficients[2],
moisture.com.slope$coefficients[3], moisture.com.slope$coefficients[3] +
moisture.com.slope$coefficients[4], rep(moisture.com.slope$coefficients[5],
2), Ke.des, Ke.ads),
  se = c(moisture.com.slope$se[1], moisture.com.slope$se[2],
moisture.com.slope$se[3], moisture.com.slope$se[4],
rep(moisture.com.slope$se[5], 2), NA, NA)
)
mcs.coefs<- mcs.coefs |> mutate(
  Estimate = round(Estimate, 2),
  se = round(se, 2)
)
knitr::kable(mcs.coefs, row.names = FALSE)
```

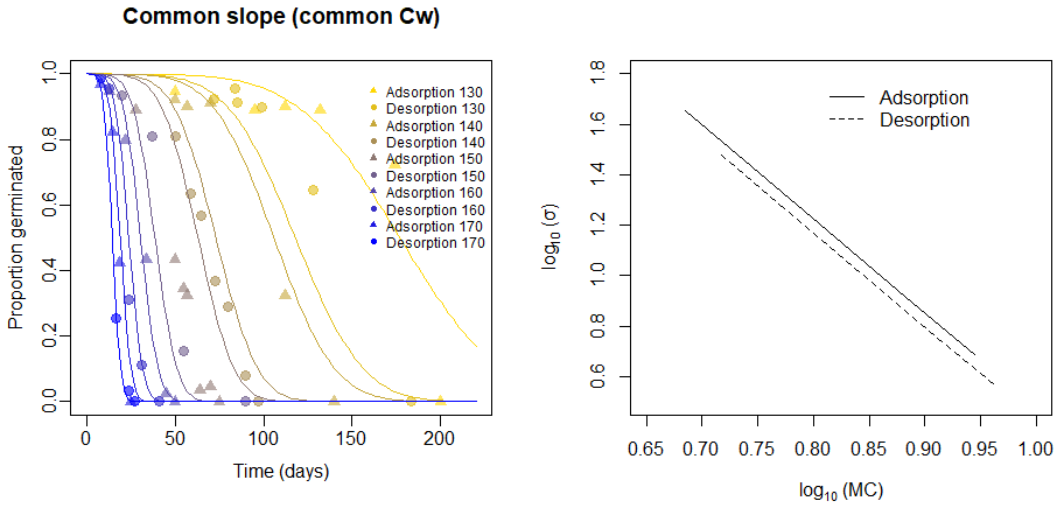
Treatment	Coefficient	Estimate	se
Desorption	Ki	3.93	0.17
Adsorption	Ki	3.44	0.19
Desorption	Kt	4.14	0.04
Adsorption	Kt	4.20	0.02
Desorption	Cw	3.72	0.04
Adsorption	Cw	3.72	0.04
Desorption	Ke	6.59	NA

Treatment	Coefficient	Estimate	se
Adsorption	Ke	6.65	NA

```
coefficients.for.plotting.M<-cbind(moisture.com.slope$coefficients,
confint(moisture.com.slope, level = 0.95))

vcov_reg <- vcov(moisture.com.slope)
coef_reg <- coef(moisture.com.slope)
coef_reg <- na.omit(coef_reg)
coef_reg <- as.numeric(coef_reg)

# estimate sigma (and CI) for each MC and sorption cycle
sigmas.ads<-c()
for (i in 1:5) {
  sigmas.ads[i] <- 10^(coefficients.for.plotting.M[3,1]+
coefficients.for.plotting.M[4,1] - ((coefficients.for.plotting.M[5,1]) *
log10(des.ads.sum$MC[des.ads.sum$Treatment[i]])))
}

sigmas.des<-c()
for (i in 6:10) {
  sigmas.des[i-5] <- 10^(coefficients.for.plotting.M[3,1]-
(coefficients.for.plotting.M[5,1] *
log10(des.ads.sum$MC[des.ads.sum$Treatment[i]])))
}

sigmas.ads.l <-c()
for (i in 1:5) {
  sigmas.ads.l[i] <- 10^(coefficients.for.plotting.M[3,2]+
coefficients.for.plotting.M[4,1]  - ((coefficients.for.plotting.M[5,2]) *
log10(des.ads.sum$MC[des.ads.sum$Treatment[i]])))
}

sigmas.des.l <-c()
for (i in 6:10) {
  sigmas.des.l[i-5] <- 10^(coefficients.for.plotting.M[3,2]-
(coefficients.for.plotting.M[5,2] *
log10(des.ads.sum$MC[des.ads.sum$Treatment[i]])))
}

sigmas.ads.u <-c()
for (i in 1:5) {
  sigmas.ads.u[i] <- 10^(coefficients.for.plotting.M[3,3] +
coefficients.for.plotting.M[4,3] - ((coefficients.for.plotting.M[5,3]) *
log10(des.ads.sum$MC[des.ads.sum$Treatment[i]])))
}

sigmas.des.u <-c()
for (i in 6:10) {
```

```
  sigmas.des.u[i-5] <- 10^(coefficients.for.plotting.M[3,3]-
(coefficients.for.plotting.M[5,3] *
log10(des.ads.sum$MC[des.ads.sum$Treatment[i]])))
}

sigmas.ads.SE<-c()
for (i in 1:5) {
  mcl<- des.ads.sum$MC[des.ads.sum$Treatment[i]]
  sigmas.ads.SE[i] <-
    deltamethod(~ 10^(x3+x4-(x5 *log10(mcl))), coef_reg, vcov_reg)
}

sigmas.des.SE<-c()
for (i in 6:10) {
  mcl<- des.ads.sum$MC[des.ads.sum$Treatment[i]]
  sigmas.des.SE[i-5] <-
    deltamethod(~ 10^(x3-( x5 *log10(mcl))), coef_reg, vcov_reg)
}

# put sigmas into one dataframe then join with des.ads.sum
sigma.moist <- data.frame(
  Treatment = des.ads.sum$Treatment,
  sigma = c(sigmas.ads, sigmas.des),
  `sigma Std. Error` = c(sigmas.ads.SE, sigmas.des.SE),
  `sigma CI.l` = c(sigmas.ads.l, sigmas.des.l),
  `sigma CI.u` = c(sigmas.ads.u, sigmas.des.u),  check.names = FALSE
)
des.ads.sum.3 <- left_join(des.ads.sum, sigma.moist, by = "Treatment")
des.ads.sum.3<- as.data.frame(des.ads.sum.3[complete.cases(des.ads.sum.3),])
des.ads.sum.3<- des.ads.sum.3[order(des.ads.sum.3$MC),]

# add p50s
des.ads.sum.3$p50 <- coefficients.for.plotting.M[1,1] * des.ads.sum.3$sigma

# add p50s confidence intervals
des.ads.sum.3$`p50 CI.l`<- coefficients.for.plotting.M[1,2] *
des.ads.sum.3$`sigma CI.l`
des.ads.sum.3$`p50 CI.u`<- coefficients.for.plotting.M[1,3] *
des.ads.sum.3$`sigma CI.u`

# Units are 'days' for sigma and p50, and '%f.wt.' for MC


# Plot
colfunc <- colorRampPalette(c("gold", "blue"))
colours<- colfunc(10)
plotting.data<- des.ads[order(des.ads$MC),]
plotting.data$Treatment <- factor(plotting.data$Treatment,
levels=c('Adsorption 130', 'Desorption 130', 'Adsorption 140', 'Desorption
```

```
140', 'Adsorption 150', 'Desorption 150', 'Adsorption 160', 'Desorption 160',
'Adsorption 170', 'Desorption 170' ))

par(mfrow=c(1,2))

plot(plotting.data$period, plotting.data$germinated/plotting.data$sown,
xlab="Time (days)", ylab="Proportion germinated", xlim=c(0,220), ylim=c(0,1),
mgp=c(2,.5,0), pch= c(17, 19)[as.numeric(plotting.data$sorption)], cex=1.2,
col= alpha(colours, 0.6)[plotting.data$Treatment])

curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/
des.ads.sum.3$sigma[1]*x)), add=TRUE, lty=1,lwd=1, col = colours[1])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.3$sigma
[2]*x)), add=TRUE, lty=1,lwd=1, col = colours[2])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.3$sigma
[3]*x)), add=TRUE, lty=1,lwd=1, col = colours[3])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.3$sigma
[4]*x)), add=TRUE, lty=1,lwd=1, col = colours[4])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.3$sigma
[5]*x)), add=TRUE, lty=1,lwd=1, col = colours[5])

curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.3$sigma
[6]*x)), add=TRUE, lty=1,lwd=1, col = colours[6])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.3$sigma
[7]*x)), add=TRUE, lty=1,lwd=1, col = colours[7])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.3$sigma
[8]*x)), add=TRUE, lty=1,lwd=1, col = colours[8])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.3$sigma
[9]*x)), add=TRUE, lty=1,lwd=1, col = colours[9])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.3$sigma
[10]*x)), add=TRUE, lty=1,lwd=1, col = colours[10])


legend(155, 1, unique(plotting.data$Treatment),col=colours, pch=rep(c(17,19),
5), bty = "n", cex=0.75)
title("Common slope (common Cw)")

plot(log10(subset(des.ads.sum.3, sorption=="Adsorption")$MC[1:5]),
log10(subset(des.ads.sum.3, sorption=="Adsorption")$sigma[1:5]), type = "l",
lty = 1, xlab= expression("log"[10]*" (MC)"), ylab= expression("log"[10]*"
(σ)"), xlim=c(0.65,1), ylim=c(0.5,1.8))
lines(log10(subset(des.ads.sum.3, sorption=="Desorption")$MC[1:5]),
log10(subset(des.ads.sum.3, sorption=="Desorption")$sigma[1:5]), type = "l",
lty = 2)
legend(0.8, 1.8, legend = c("Adsorption", "Desorption"),lty = c(1, 2), bty =
"n")
```




```
# Round to two decimal places and add units for reporting.
des.ads.sum.3 <- des.ads.sum.3 |>
  mutate(`MC (% f.wt.)` = round(MC, 2),
         `sigma (days)` = round(sigma, 2),
         `sigma Std. Error` = round(`sigma Std. Error`, 2),
         `sigma CI.l` = round(`sigma CI.l`, 2),
         `sigma CI.u` = round(`sigma CI.u`, 2),
         `p50 (days)` = round(p50),
         `p50 CI.l` = round(`p50 CI.l`, 2),
         `p50 CI.u` = round(`p50 CI.u`, 2)) |>
  dplyr::select(chamber.RH, `MC (% f.wt.)`, sorption, `sigma (days)`, `sigma
Std. Error`, `sigma CI.l`,
                `sigma CI.u`, `p50 (days)`, `p50 CI.l`, `p50 CI.u`)

# call the table.
# Units are 'days' for sigma and p50, and '%f.wt.' for MC
knitr::kable(des.ads.sum.3, row.names = FALSE)
```

chamber.RH	MC (% f.wt.)	sorption	sigma (days)	sigma Std. Error	sigma CI.l	sigma CI.u	p50 (days)	p50 CI.l	p50 CI.u
30	4.84	Adsorption	44.98	1.81	41.98	53.35	177	151.36	227.19
30	5.21	Desorption	30.00	1.38	28.15	31.97	118	101.49	136.17
40	5.55	Adsorption	26.92	1.02	25.38	31.61	106	91.49	134.61
40	5.93	Desorption	18.57	0.81	17.59	19.61	73	63.41	83.51
50	6.39	Adsorption	16.00	0.58	15.23	18.59	63	54.92	79.18

chamber.RH	MC (% f.wt.)	sorption	sigma (days)	sigma Std. Error	sigma CI.l	sigma CI.u	p50 (days)	p50 CI.l	p50 CI.u
50	7.03	Desorption	9.85	0.41	9.44	10.27	39	34.05	43.75
60	7.75	Adsorption	7.79	0.27	7.52	8.93	31	27.11	38.01
60	8.00	Desorption	6.10	0.24	5.90	6.30	24	21.27	26.83
70	8.81	Adsorption	4.84	0.17	4.72	5.50	19	17.02	23.43
70	9.14	Desorption	3.71	0.15	3.63	3.80	15	13.09	16.19

```
### one line
sv.ol<- c(3.8, 0, 4.1, 3.7) # initial parameters
moisture.one.line<- bnlr(y= cbind(germinated, sown - germinated), link =
"probit",
                         mu = ~ Kides + (Kiads*d1) - (1/10^(Kt-Cw
*log10(MC))*period), ###please see the additional brackets here
                         pmu = sv.ol,
                         fscale = 1,
                         print.level = 1,
                         typsize = abs(sv.ol), #expected parameters values at
convergence (these may be specified
                         #differently from the starting values if needed)
                         ndigit = 10,
                         gradtol = 1e-05,
                         stepmax = 1 * sqrt(sv.ol %*% sv.ol),
                         steptol = 1e-05, #this may be changed depending on
the warnings
                         iterlim = 1000
)
```

```
## iteration = 0
## Step:
## [1] 0 0 0 0
## Parameter:
## [1] 3.8 0.0 4.1 3.7
## Function Value
## [1] 1899.096
## Gradient:
## [1]  -611.8990  -311.6314 -6169.1249  4967.8151
##
## iteration = 37
## Parameter:
## [1]  3.65643274 -0.02334742  4.17915608  3.72364825
## Function Value
## [1] 1769.247
```

```
## Gradient:
## [1] -0.0002462867 -0.0012111812  0.0015068926 -0.0023931766
##
## Relative gradient close to zero.
## Current iterate is probably solution.
```

```
# view model outputs including df, AIC, and coefficients (parameter
estimates)
print(moisture.one.line)
```

```
##
## Call:
## bnlr(y = cbind(germinated, sown - germinated), link = "probit",
##     mu = ~Kides + (Kiads * d1) - (1/10^(Kt - Cw * log10(MC)) *
##         period), pmu = sv.ol, fscale = 1, print.level = 1, typsize =
abs(sv.ol),
##     ndigit = 10, gradtol = 1e-05, stepmax = 1 * sqrt(sv.ol %*%
##         sv.ol), steptol = 1e-05, iterlim = 1000)
##
## binomial distribution
##
## Response: cbind(germinated, sown - germinated)
##
## Log likelihood function:
## {
##     m <- pnorm(mu1(p))
##     -sum(wt * (y[, 1] * log(m) + y[, 2] * log(1 - m)))
## }
##
## Location function:
## ~Kides + (Kiads * d1) - (1/10^(Kt - Cw * log10(MC)) * period)
##
## -Log likelihood    328.2176
## Degrees of freedom 48
## AIC                332.2176
## Iterations         37
##
## Location parameters:
##        estimate       se
## Kides   3.65643  0.10960
## Kiads  -0.02335  0.04798
## Kt      4.17916  0.03654
## Cw      3.72365  0.03638
##
## Correlations:
##         1        2        3        4
## 1  1.0000 -0.21027 -0.64672 -0.39531
## 2 -0.2103  1.00000 -0.03868 -0.03929
## 3 -0.6467 -0.03868  1.00000  0.94910
## 4 -0.3953 -0.03929  0.94910  1.00000
```

```r
deviance(moisture.one.line) # view deviance

## [1] 656.4351

# Estimate Ke based on universal temperature constants
Ke <- moisture.one.line$coefficients[3] + 45 * 0.0329 + 45 * 45 * 0.000478

# Table of coeficients
mol.coefs<-data.frame(
  Treatment = c(rep(c("Desorption", "Adsorption"), 4)),
  Coefficient = c(rep("Ki", 2), rep("Kt", 2), rep("Cw", 2), rep("Ke", 2)),
  Estimate = c(moisture.one.line$coefficients[1],
moisture.one.line$coefficients[1] + moisture.one.line$coefficients[2],
rep(moisture.one.line$coefficients[3], 2),
rep(moisture.one.line$coefficients[4], 2), rep(Ke, 2)),
  se = c(moisture.one.line$se[1], moisture.one.line$se[2],
rep(moisture.one.line$se[3], 2), rep(moisture.one.line$se[4], 2), NA, NA)
)
mol.coefs<- mol.coefs |> mutate(
  Estimate = round(Estimate, 2),
  se = round(se, 2)
)
knitr::kable(mol.coefs, row.names = FALSE)
```

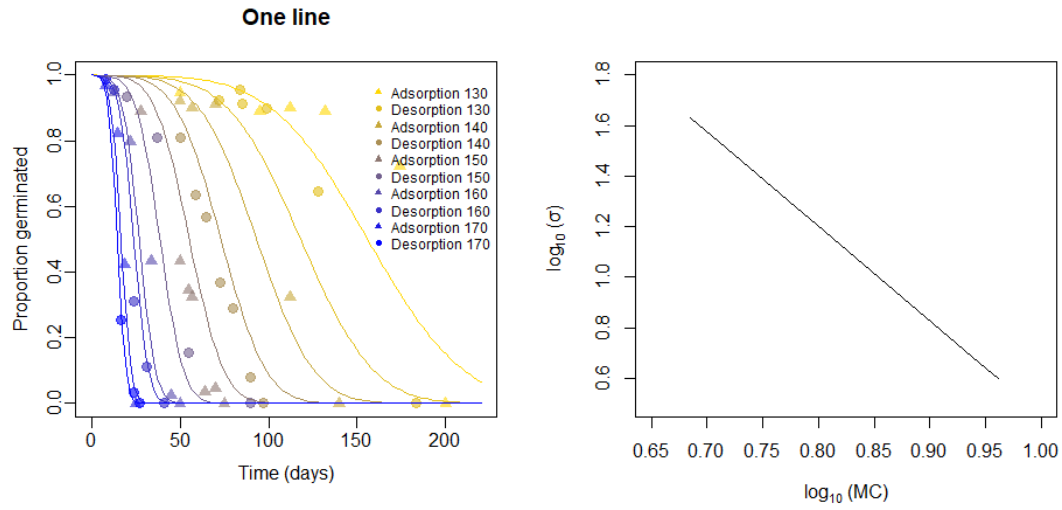
Treatment	Coefficient	Estimate	se
Desorption	Ki	3.66	0.11
Adsorption	Ki	3.63	0.05
Desorption	Kt	4.18	0.04
Adsorption	Kt	4.18	0.04
Desorption	Cw	3.72	0.04
Adsorption	Cw	3.72	0.04
Desorption	Ke	6.63	NA
Adsorption	Ke	6.63	NA

```r
coefficients.for.plotting.M<-cbind(moisture.one.line$coefficients,
confint(moisture.one.line, level = 0.95))

vcov_reg <- vcov(moisture.one.line)
coef_reg <- coef(moisture.one.line)
coef_reg <- na.omit(coef_reg)
coef_reg <- as.numeric(coef_reg)

# estimate sigma (and CI) for each MC and sorption cycle
sigmas.ads<-c()
for (i in 1:5) {
  sigmas.ads[i] <- 10^(coefficients.for.plotting.M[3,1]+ -
((coefficients.for.plotting.M[4,1]) *
log10(des.ads.sum$MC[des.ads.sum$Treatment[i]])))
```

```
}

sigmas.des<-c()
for (i in 6:10) {
  sigmas.des[i-5] <- 10^(coefficients.for.plotting.M[3,1]-
(coefficients.for.plotting.M[4,1] *
log10(des.ads.sum$MC[des.ads.sum$Treatment[i]])))
}

sigmas.ads.l <-c()
for (i in 1:5) {
  sigmas.ads.l[i] <- 10^(coefficients.for.plotting.M[3,2] -
((coefficients.for.plotting.M[4,2]) *
log10(des.ads.sum$MC[des.ads.sum$Treatment[i]])))
}

sigmas.des.l <-c()
for (i in 6:10) {
  sigmas.des.l[i-5] <- 10^(coefficients.for.plotting.M[3,2]-
(coefficients.for.plotting.M[4,2] *
log10(des.ads.sum$MC[des.ads.sum$Treatment[i]])))
}

sigmas.ads.u <-c()
for (i in 1:5) {
  sigmas.ads.u[i] <- 10^(coefficients.for.plotting.M[3,3] -
((coefficients.for.plotting.M[4,3]) *
log10(des.ads.sum$MC[des.ads.sum$Treatment[i]])))
}

sigmas.des.u <-c()
for (i in 6:10) {
  sigmas.des.u[i-5] <- 10^(coefficients.for.plotting.M[3,3]-
(coefficients.for.plotting.M[4,3] *
log10(des.ads.sum$MC[des.ads.sum$Treatment[i]])))
}

sigmas.ads.SE<-c()
for (i in 1:5) {
  mcl<- des.ads.sum$MC[des.ads.sum$Treatment[i]]
  sigmas.ads.SE[i] <-
    deltamethod(~ 10^(x3-(x4 *log10(mcl))), coef_reg, vcov_reg)
}

sigmas.des.SE<-c()
for (i in 6:10) {
  mcl<- des.ads.sum$MC[des.ads.sum$Treatment[i]]
  sigmas.des.SE[i-5] <-
    deltamethod(~ 10^(x3-( x4 *log10(mcl))), coef_reg, vcov_reg)
```

```r
}

# put sigmas into one dataframe then join with des.ads.sum
sigma.moist <- data.frame(
  Treatment = des.ads.sum$Treatment,
  sigma = c(sigmas.ads, sigmas.des),
  `sigma Std. Error` = c(sigmas.ads.SE, sigmas.des.SE),
  `sigma CI.l` = c(sigmas.ads.l, sigmas.des.l),
  `sigma CI.u` = c(sigmas.ads.u, sigmas.des.u),  check.names = FALSE
)
des.ads.sum.4 <- left_join(des.ads.sum, sigma.moist, by = "Treatment")
des.ads.sum.4<- as.data.frame(des.ads.sum.4[complete.cases(des.ads.sum.4),])
des.ads.sum.4<- des.ads.sum.4[order(des.ads.sum.4$MC),]

# add p50s
des.ads.sum.4$p50 <- coefficients.for.plotting.M[1,1] * des.ads.sum.4$sigma

# add p50s confidence intervals
des.ads.sum.4$`p50 CI.l`<- coefficients.for.plotting.M[1,2] *
des.ads.sum.4$`sigma CI.l`
des.ads.sum.4$`p50 CI.u`<- coefficients.for.plotting.M[1,3] *
des.ads.sum.4$`sigma CI.u`

# Units are 'days' for sigma and p50, and '%f.wt.' for MC



# Plot
colfunc <- colorRampPalette(c("gold", "blue"))
colours<- colfunc(10)
plotting.data<- des.ads[order(des.ads$MC),]
plotting.data$Treatment <- factor(plotting.data$Treatment,
levels=c('Adsorption 130', 'Desorption 130', 'Adsorption 140', 'Desorption
140', 'Adsorption 150', 'Desorption 150', 'Adsorption 160', 'Desorption 160',
'Adsorption 170', 'Desorption 170' ))

par(mfrow=c(1,2))
plot(plotting.data$period, plotting.data$germinated/plotting.data$sown,
xlab="Time (days)", ylab="Proportion germinated", xlim=c(0,220), ylim=c(0,1),
mgp=c(2,.5,0), pch= c(17, 19)[as.numeric(plotting.data$sorption)], cex=1.2,
col= alpha(colours, 0.6)[plotting.data$Treatment])

curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/
des.ads.sum.4$sigma[1]*x)), add=TRUE, lty=1,lwd=1, col = colours[1])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.4$sigma
[2]*x)), add=TRUE, lty=1,lwd=1, col = colours[2])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.4$sigma
[3]*x)), add=TRUE, lty=1,lwd=1, col = colours[3])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.4$sigma
[4]*x)), add=TRUE, lty=1,lwd=1, col = colours[4])
```

```
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.4$sigma
[5]*x)), add=TRUE, lty=1,lwd=1, col = colours[5])

curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.4$sigma
[6]*x)), add=TRUE, lty=1,lwd=1, col = colours[6])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.4$sigma
[7]*x)), add=TRUE, lty=1,lwd=1, col = colours[7])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.4$sigma
[8]*x)), add=TRUE, lty=1,lwd=1, col = colours[8])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.4$sigma
[9]*x)), add=TRUE, lty=1,lwd=1, col = colours[9])
curve(pnorm(coefficients.for.plotting.M[1,1] + (-1/ des.ads.sum.4$sigma
[10]*x)), add=TRUE, lty=1,lwd=1, col = colours[10])

legend(155, 1, unique(plotting.data$Treatment),col=colours, pch=rep(c(17,19),
5), bty = "n", cex=0.75)
title("One line")

plot(log10(des.ads.sum.4$MC), log10(des.ads.sum.4$sigma), xlab=
expression("log"[10]*" (MC)"), ylab= expression("log"[10]*" (σ)"),
xlim=c(0.65,1), ylim=c(0.5,1.8), type="l") #, pch= c(17,
19)[as.numeric(des.ads.sum.4$sorption)])
```




```
# Round to two decimal places and add units for reporting.
des.ads.sum.4 <- des.ads.sum.4 |>
  mutate(`MC (% f.wt.)` = round(MC, 2),
         `sigma (days)` = round(sigma, 2),
         `sigma Std. Error` = round(`sigma Std. Error`, 2),
         `sigma CI.l` = round(`sigma CI.l`, 2),
         `sigma CI.u` = round(`sigma CI.u`, 2),
         `p50 (days)` = round(p50),
         `p50 CI.l` = round(`p50 CI.l`, 2),
         `p50 CI.u` = round(`p50 CI.u`, 2)) |>
```

```
  dplyr::select(chamber.RH, `MC (% f.wt.)`, sorption, `sigma (days)`, `sigma
Std. Error`, `sigma CI.l`,
                `sigma CI.u`, `p50 (days)`, `p50 CI.l`, `p50 CI.u`)

# call the table.
# Units are 'days' for sigma and p50, and '%f.wt.' for MC
knitr::kable(des.ads.sum.4, row.names = FALSE)
```

chamber.RH	MC (% f.wt.)	sorption	sigma (days)	sigma Std. Error	sigma CI.l	sigma CI.u	p50 (days)	p50 CI.l	p50 CI.u
30	4.84	Adsorption	42.63	1.48	40.45	44.92	156	139.20	173.91
30	5.21	Desorption	32.30	1.07	30.81	33.86	118	106.05	131.10
40	5.55	Adsorption	25.49	0.81	24.43	26.60	93	84.07	102.99
40	5.93	Desorption	19.98	0.61	19.24	20.76	73	66.21	80.36
50	6.39	Adsorption	15.14	0.44	14.65	15.64	55	50.42	60.55
50	7.03	Desorption	10.59	0.30	10.32	10.87	39	35.52	42.07
60	7.75	Adsorption	7.36	0.20	7.22	7.50	27	24.86	29.04
60	8.00	Desorption	6.55	0.18	6.44	6.66	24	22.17	25.78
70	8.81	Adsorption	4.58	0.12	4.53	4.62	17	15.60	17.89
70	9.14	Desorption	3.99	0.11	3.96	4.02	15	13.63	15.55

```
### Create deviance table
moist.dev.table <- data.frame(
  Model = c("Independent", "Common.intercept", "common.slope", "One.line"),
  Deviance = c(deviance(moisture.independent), deviance(moisture.com.inter),
deviance(moisture.com.slope), deviance(moisture.one.line)),
  df = c(moisture.independent$df, moisture.com.inter$df,
moisture.com.slope$df, moisture.one.line$df),
  AIC = c(moisture.independent$aic, moisture.com.inter$aic,
moisture.com.slope$aic, moisture.one.line$aic)
)

# quickly view of table
knitr::kable(moist.dev.table, row.names = FALSE)
```

Model	Deviance	df	AIC

Model	Deviance	df	AIC
Independent	652.7832	46	332.3916
Common.intercept	654.0153	47	332.0077
common.slope	654.6206	47	332.3103
One.line	656.4351	48	332.2176

```
### F statistics
# Here we demonstrate pairwise comparisons of common slope and common
intercept with independent and one line models.

# Common intercept vs Independent
moist.df.change1<-moisture.com.inter$df - moisture.independent$df
moist.df.change1

## [1] 1

moist.F1<-((deviance(moisture.com.inter)-
deviance(moisture.independent))/moist.df.change1)/(deviance(moisture.independ
ent)/moisture.independent$df)
moist.F1

## [1] 0.08682815

moist.F1prob<-pf(moist.F1, moist.df.change1, moisture.independent$df,
lower.tail = FALSE) # note the lower.tail = False is not default
moist.F1prob<-format(signif(moist.F1prob, 4), nsmall = 4)
moist.F1prob

## [1] "0.7696"

moist.F1sum<-c(moist.df.change1, moist.F1, moist.F1prob)
moist.F1sum

## [1] "1"                 "0.086828145327085" "0.7696"

# Common slope vs Independent
moist.df.change2<-moisture.com.slope$df - moisture.independent$df
moist.df.change2

## [1] 1

moist.F2<-((deviance(moisture.com.slope)-
deviance(moisture.independent))/moist.df.change2)/(deviance(moisture.independ
ent)/moisture.independent$df)
moist.F2

## [1] 0.1294773

moist.F2prob<-pf(moist.F2, moist.df.change2, moisture.independent$df,
lower.tail = FALSE) # note the lower.tail = False is not default
moist.F2prob<-format(signif(moist.F2prob, 4), nsmall = 4)
moist.F2prob
```

```
## [1] "0.7206"
```

```
moist.F2sum<-c(moist.df.change2, moist.F2, moist.F2prob)
moist.F2sum
```

```
## [1] "1"                 "0.129477340034424" "0.7206"
```

```
# One line vs Common intercept
moist.df.change3<-moisture.one.line$df - moisture.com.inter$df
moist.df.change3
```

```
## [1] 1
```

```
moist.F3<-((deviance(moisture.one.line)-
deviance(moisture.com.inter))/moist.df.change3)/(deviance(moisture.com.inter)
/moisture.com.inter$df)
moist.F3
```

```
## [1] 0.173894
```

```
moist.F3prob<-pf(moist.F3, moist.df.change3, moisture.com.inter$df,
lower.tail = FALSE) # note the lower.tail = False is not default
moist.F3prob<-format(signif(moist.F3prob, 4), nsmall = 4)
moist.F3prob
```

```
## [1] "0.6786"
```

```
moist.F3sum<-c(moist.df.change3, moist.F3, moist.F3prob)
moist.F3sum
```

```
## [1] "1"                 "0.173893996601378" "0.6786"
```

```
# One line vs Common Slope
moist.df.change4<-moisture.one.line$df - moisture.com.slope$df
moist.df.change4
```

```
## [1] 1
```

```
moist.F4<-((deviance(moisture.one.line)-
deviance(moisture.com.slope))/moist.df.change4)/(deviance(moisture.com.slope)
/moisture.com.slope$df)
moist.F4
```

```
## [1] 0.1302792
```

```
moist.F4prob<-pf(moist.F4, moist.df.change4, moisture.com.slope$df,
lower.tail = FALSE) # note the lower.tail = False is not default
moist.F4prob<-format(signif(moist.F4prob, 4), nsmall = 4)
moist.F4prob
```

```
## [1] "0.7198"
```

```
moist.F4sum<-c(moist.df.change4, moist.F4, moist.F4prob)
moist.F4sum
```

```
## [1] "1"                 "0.130279182253192" "0.7198"
```

```
# Optional. Build table
moist.F.table1<-data.frame(
  Model.comparison = c("Common intercept vs Independent", "Common slope vs
Independent", "One line vs Common intercept", "One line vs Common slope"),
  DF.change = c(moist.df.change1, moist.df.change2, moist.df.change3,
moist.df.change4),
  F.statistic = c(moist.F1, moist.F2, moist.F3, moist.F4),
  F.probability = c(moist.F1prob, moist.F2prob, moist.F3prob, moist.F4prob)
)


# call the table
knitr::kable(moist.F.table1, row.names = FALSE)
```

Model.comparison	DF.change	F.statistic	F.probability
Common intercept vs Independent	1	0.0868281	0.7696
Common slope vs Independent	1	0.1294773	0.7206
One line vs Common intercept	1	0.1738940	0.6786
One line vs Common slope	1	0.1302792	0.7198