

# The Rice example: illustrating the first five steps for smoothing and extracting traits (SET) using growthPheno

Chris Brien

11 October, 2021

This example is based on the data whose analysis has been published by Al-Tamimi et al. (2016). The five steps of the method for smoothing and extracting traits (SET) described in detail in Brien et al. (2020) is illustrated for this data.

## Initialize

### Step 1: Import, select and derive longitudinal data

#### Step 1(a): Import the data

```
data(RiceRaw.dat)
```

#### Step 1(b): Organize the data

Here the imaging variables are selected and covariates and factors added to produce `longi.dat`.

```
longi.dat <- longitudinalPrime(data=RiceRaw.dat, smarthouse.lev=c("NE","NW"))
```

```
longi.dat <- designFactors(longi.dat, insertName = "xDays",  
                           designfactorMethod="StandardOrder")
```

```
# Particular edits to longi.dat
```

```
longi.dat <- within(longi.dat,  
                   {  
                     Days.after.Salting <- as.numfac(Days) - 29  
                   })
```

```
longi.dat <- with(longi.dat, longi.dat[order(Snapshot.ID.Tag,Days), ])
```

#### Step 1(c): Derive longitudinal traits that result in a value for each observation

```
# Set responses
```

```
responses.image <- c("Area")
```

```
responses.smooth <- paste(responses.image, "smooth", sep=".")
```

```
# Form growth rates for each observation of a subset of responses by differencing
```

```
longi.dat <- splitContGRdiff(longi.dat, responses.image,  
                             INDICES="Snapshot.ID.Tag",  
                             which.rates = c("AGR","RGR"))
```

```

# Form Area.WUI
longi.dat <- within(longi.dat,
  {
    Area.WUI <- WUI(Area.AGR*Days.diffs, Water.Loss)
  })

# Add cumulative responses
longi.dat <- within(longi.dat,
  {
    Water.Loss.Cum <- unlist(by(Water.Loss, Snapshot.ID.Tag,
                               cumulate, exclude.1st=TRUE))
    WUI.cum <- Area / Water.Loss.Cum
  })

# Check longi.dat
head(longi.dat)

```

```

## Snapshot.ID.Tag Days Smarthouse Lane Position Snapshot.Time.Stamp xPosn Reps
## 1 045451-C 28 NE 1 2 2015-02-18 02:14:00 -11 1
## 2 045451-C 30 NE 1 2 2015-02-20 02:14:00 -11 1
## 3 045451-C 31 NE 1 2 2015-02-21 02:14:00 -11 1
## 4 045451-C 32 NE 1 2 2015-02-22 02:14:00 -11 1
## 5 045451-C 33 NE 1 2 2015-02-23 02:14:00 -11 1
## 6 045451-C 34 NE 1 2 2015-02-24 02:14:00 -11 1
## Hour xDays Zones xZones SHZones ZLane ZMainplots Subplots xMainPosn
## 1 2.233333 -7.428571 1 -2.5 1 1 1 1 -10.5
## 2 2.233333 -5.428571 1 -2.5 1 1 1 1 -10.5
## 3 2.233333 -4.428571 1 -2.5 1 1 1 1 -10.5
## 4 2.233333 -3.428571 1 -2.5 1 1 1 1 -10.5
## 5 2.233333 -2.428571 1 -2.5 1 1 1 1 -10.5
## 6 2.233333 -1.428571 1 -2.5 1 1 1 1 -10.5
## Genotype.ID Treatment.1 Weight.Before Weight.After Water.Amount Water.Loss
## 1 121080 Control 4007 4031 28 NA
## 2 121080 Control 4056 4084 32 -25
## 3 121080 Control 4036 4083 52 48
## 4 121080 Control 4027 4085 61 56
## 5 121080 Control 4019 4084 69 66
## 6 121080 Control 4014 4083 74 70
## Area Area.SV1 Area.SV2 Area.TV Boundary.Points.To.Area.Ratio.SV1
## 1 57.446 20.912 11.526 25.008 0.353912
## 2 89.306 29.073 21.495 38.738 0.310735
## 3 100.138 27.751 26.835 45.552 0.354293
## 4 128.323 34.697 32.848 60.778 0.371012
## 5 158.776 46.779 37.871 74.126 0.319823
## 6 182.551 48.849 48.794 84.908 0.328400
## Boundary.Points.To.Area.Ratio.SV2 Boundary.Points.To.Area.Ratio.TV
## 1 0.454104 0.197537
## 2 0.401396 0.172182
## 3 0.332364 0.174175
## 4 0.358469 0.178157
## 5 0.347179 0.172517
## 6 0.290220 0.163153
## Caliper.Length.SV1 Caliper.Length.SV2 Caliper.Length.TV Compactness.SV1
## 1 666.013 668.692 704.189 0.0930821
## 2 632.735 729.044 830.812 0.1327200

```

```

## 3      731.077      931.028      1104.350      0.0925419
## 4      791.760      878.427      1029.300      0.0969068
## 5      830.360      965.221      1197.530      0.1241550
## 6      1103.050      991.259      1408.310      0.0938637
## Compactness.SV2 Compactness.TV Convex.Hull.Area.SV1 Convex.Hull.Area.SV2
## 1      0.0689923      0.1435880      224.662      167.062
## 2      0.0734412      0.1091450      219.055      292.683
## 3      0.0678337      0.0950009      299.875      395.600
## 4      0.0707469      0.1102850      358.045      464.303
## 5      0.0783589      0.1119250      376.780      483.302
## 6      0.1014870      0.0947390      520.425      480.792
## Convex.Hull.Area.TV Center.Of.Mass.Y.SV1 Center.Of.Mass.Y.SV2
## 1      174.165      1841.78      1788.86
## 2      354.921      1837.62      1797.42
## 3      479.490      1826.88      1757.60
## 4      551.097      1798.03      1750.54
## 5      662.283      1796.70      1781.50
## 6      896.231      1809.42      1778.94
## Max.Distance.Above.Horizon.Line.SV1 Max.Distance.Above.Horizon.Line.SV2
## 1      620
## 2      543
## 3      642
## 4      736
## 5      658
## 6      639
## Days.after.Salting Days.diffs Area.AGR Area.RGR Area.WUI WUI.cum
## 1      -1      NA      NA      NA      NA      NA
## 2      1      2      15.930 0.2206116 -1.2744000 -3.5722400
## 3      2      1      10.832 0.1144806 0.2256667 4.3538261
## 4      3      1      28.185 0.2480013 0.5033036 1.6243418
## 5      4      1      30.453 0.2129439 0.4614091 1.0950069
## 6      5      1      23.775 0.1395352 0.3396429 0.8490744
## Water.Loss.Cum
## 1      NA
## 2      -25
## 3      23
## 4      79
## 5      145
## 6      215

```

## Step 2: Exploratory analysis

### Step 2(a): Fit splines to smooth the longitudinal trends in the primary traits and calculate their growth rates

The smoothing.method used is `direct` and `df` is set to 4. The growth rates are calculated by difference, rather than from the spline derivatives.

```

# Smooth responses
for (response in c(responses.image, "Water.Loss"))
  longi.dat <- splitSplines(longi.dat, response, x="xDays", INDICES = "Snapshot.ID.Tag",
                           df = 4, na.x.action="exclude", na.y.action = "exclude")

```

```
## Warning in FUN(X[[i]], ...): Need at least 4 distinct x values to fit a spline -
```



```

## all fitted values set to NA

## Warning in FUN(X[[i]], ...): Need at least 4 distinct x values to fit a spline -
## all fitted values set to NA

## Warning in FUN(X[[i]], ...): Need at least 4 distinct x values to fit a spline -
## all fitted values set to NA

## Warning in FUN(X[[i]], ...): Need at least 4 distinct x values to fit a spline -
## all fitted values set to NA

longi.dat <- with(longi.dat, longi.dat[order(Snapshot.ID.Tag, xDays), ])

# Loop over smoothed responses, forming growth rates by differences
responses.GR <- paste(responses.smooth, "AGR", sep=".")
longi.dat <- splitContGRdiff(longi.dat, responses.smooth,
                             INDICES="Snapshot.ID.Tag",
                             which.rates = c("AGR","RGR"))

# Finalize longi.dat
longi.dat <- with(longi.dat, longi.dat[order(Snapshot.ID.Tag, xDays), ])

```

## Step 2(b): Compare plots of unsmoothed and smoothed longitudinal data

```

responses.longi <- c("Area","Area.AGR","Area.RGR", "Area.WUI")
responses.smooth.plot <- c("Area.smooth","Area.smooth.AGR","Area.smooth.RGR")
titles <- c("Total area (1000 pixels)",
           "Total area AGR (1000 pixels per day)", "Total area RGR (per day)",
           "Total area WUI (1000 pixels per mL)")
titles.smooth<-titles
nresp <- length(responses.longi)
limits <- list(c(0,1000), c(-50,125), c(-0.05,0.40), c(0,30))

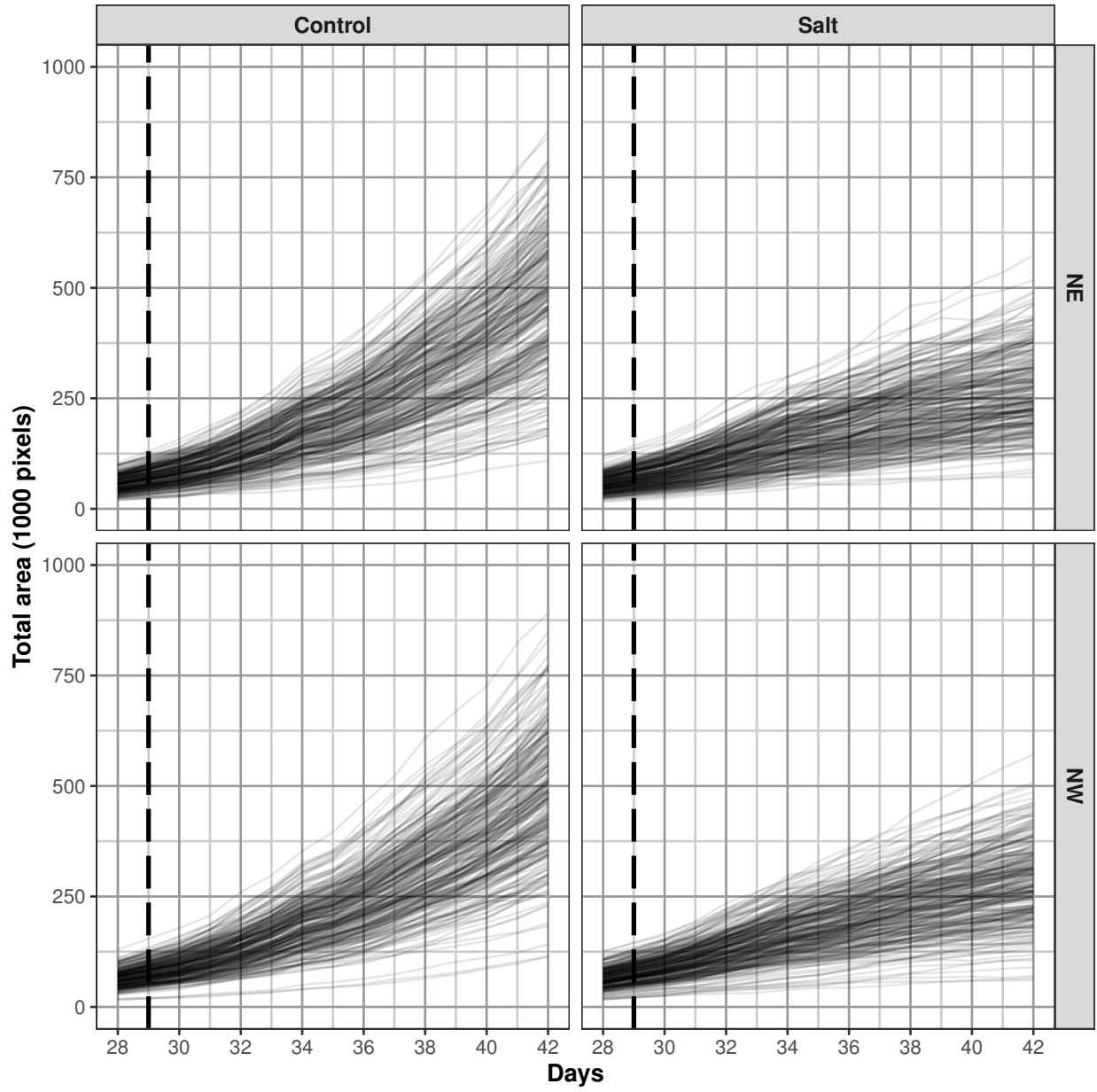
```

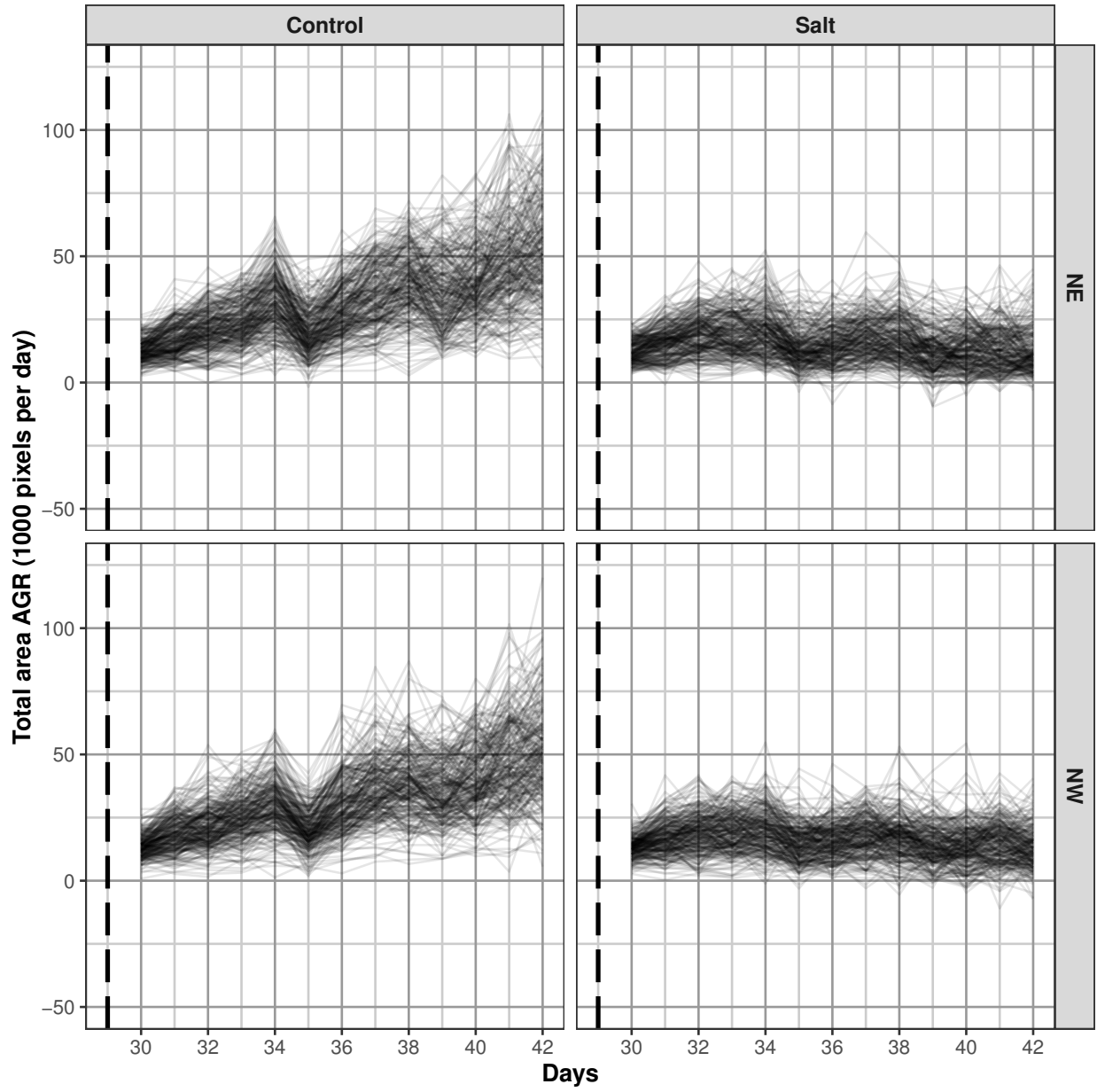
### Plot unsmoothed profiles for all longitudinal responses

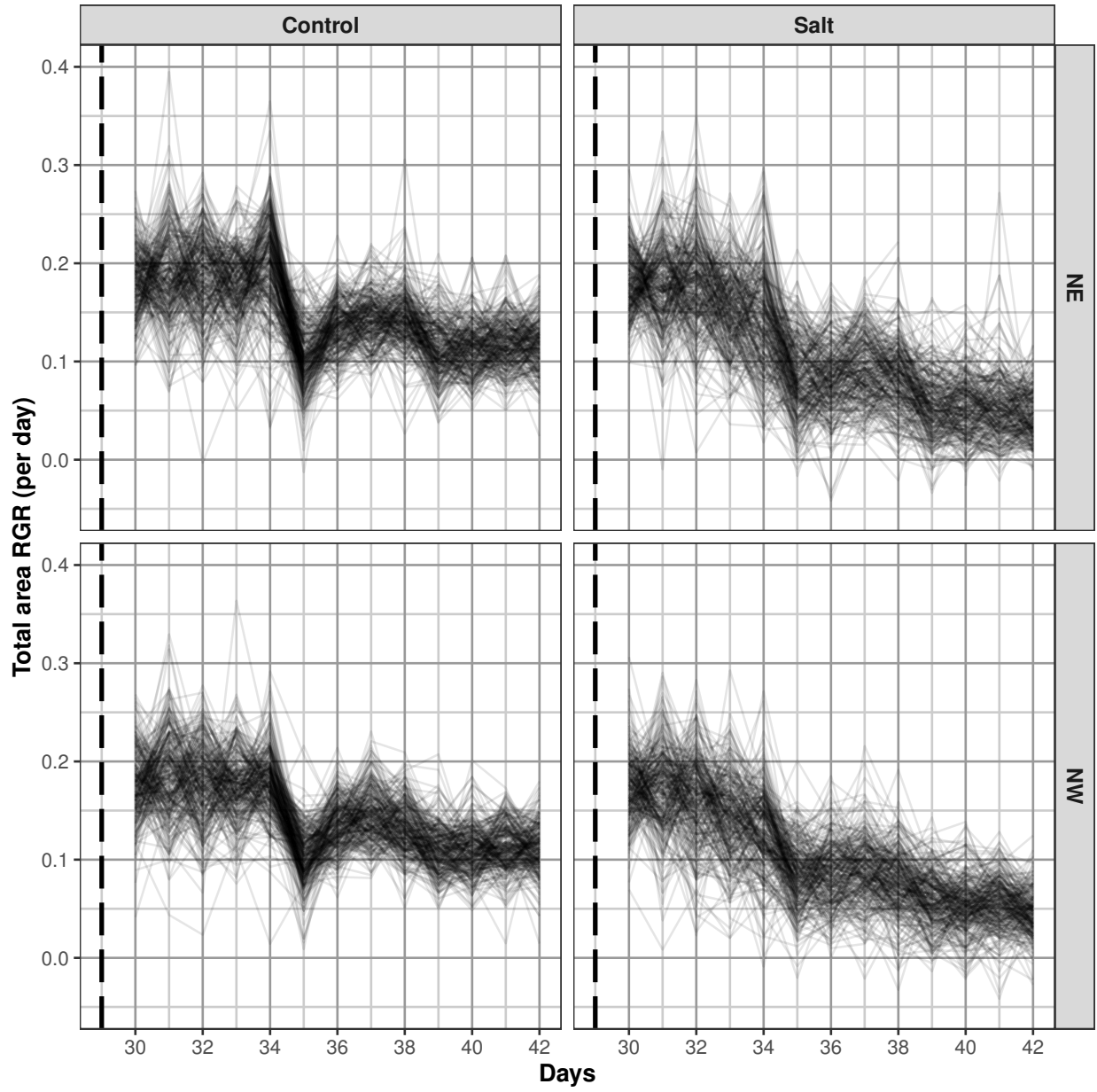
```

klimit <- 0
for (k in 1:nresp)
{
  klimit <- klimit + 1
  plt <- plotLongitudinal(data = longi.dat, response = responses.longi[k],
                          y.title = titles[k], x="xDays+35.42857143", printPlot=FALSE)
  plt <- plt + geom_vline(xintercept=29, linetype="longdash", size=1) +
    scale_x_continuous(breaks=seq(28, 42, by=2)) +
    scale_y_continuous(limits=limits[[klimit]])
  print(plt)
}

```

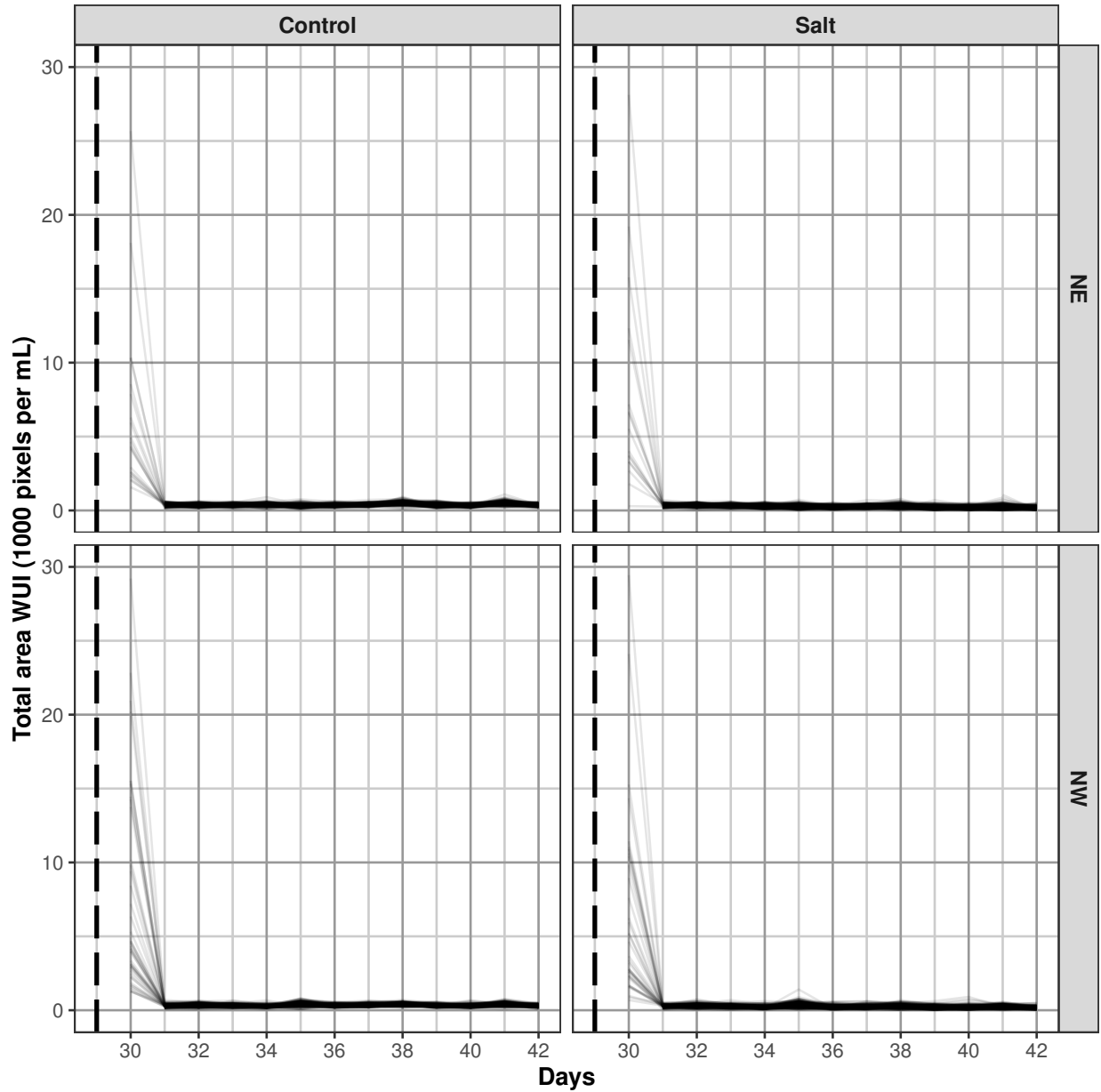






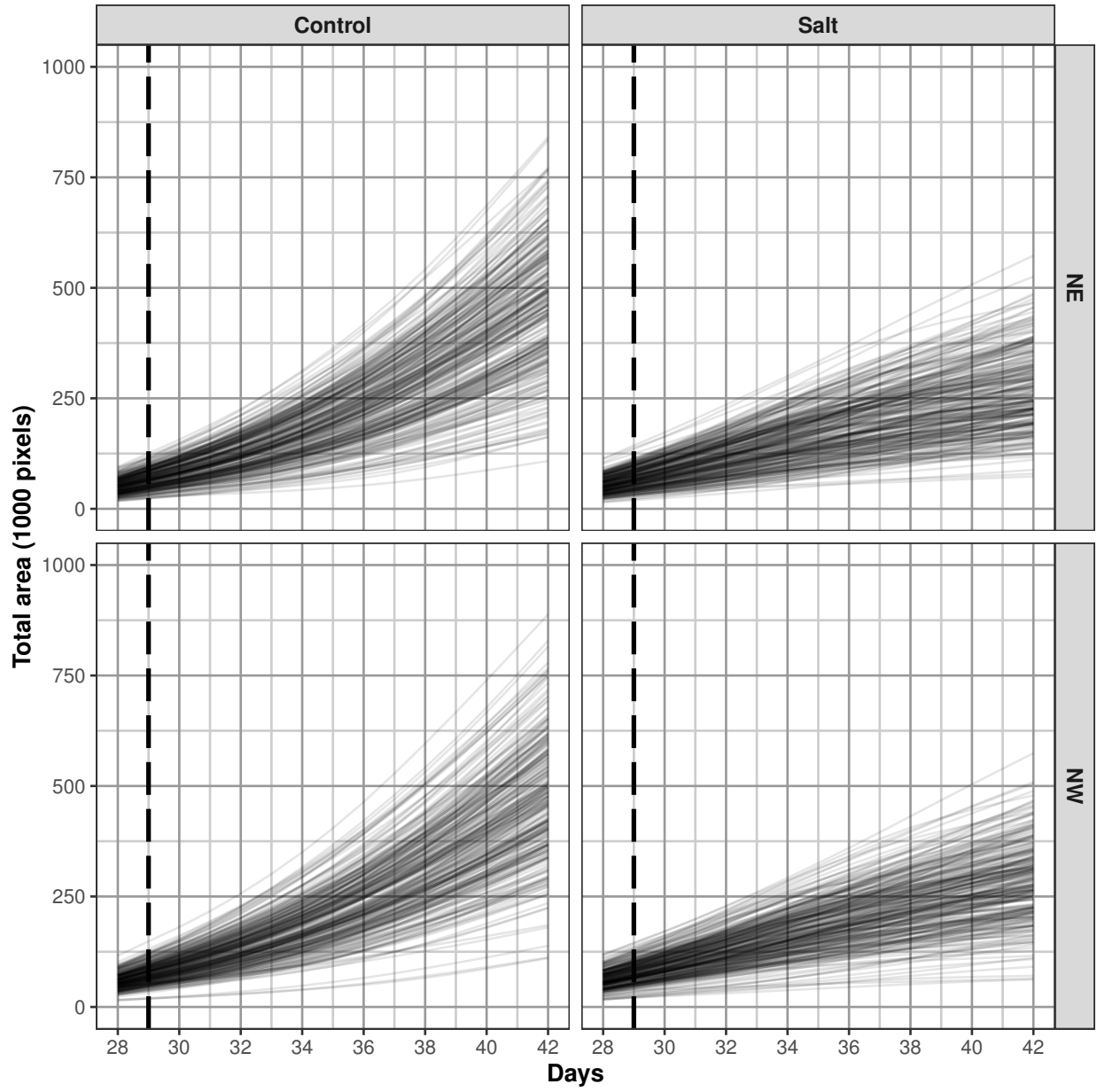
## Warning: Removed 932 row(s) containing missing values (geom\_path).

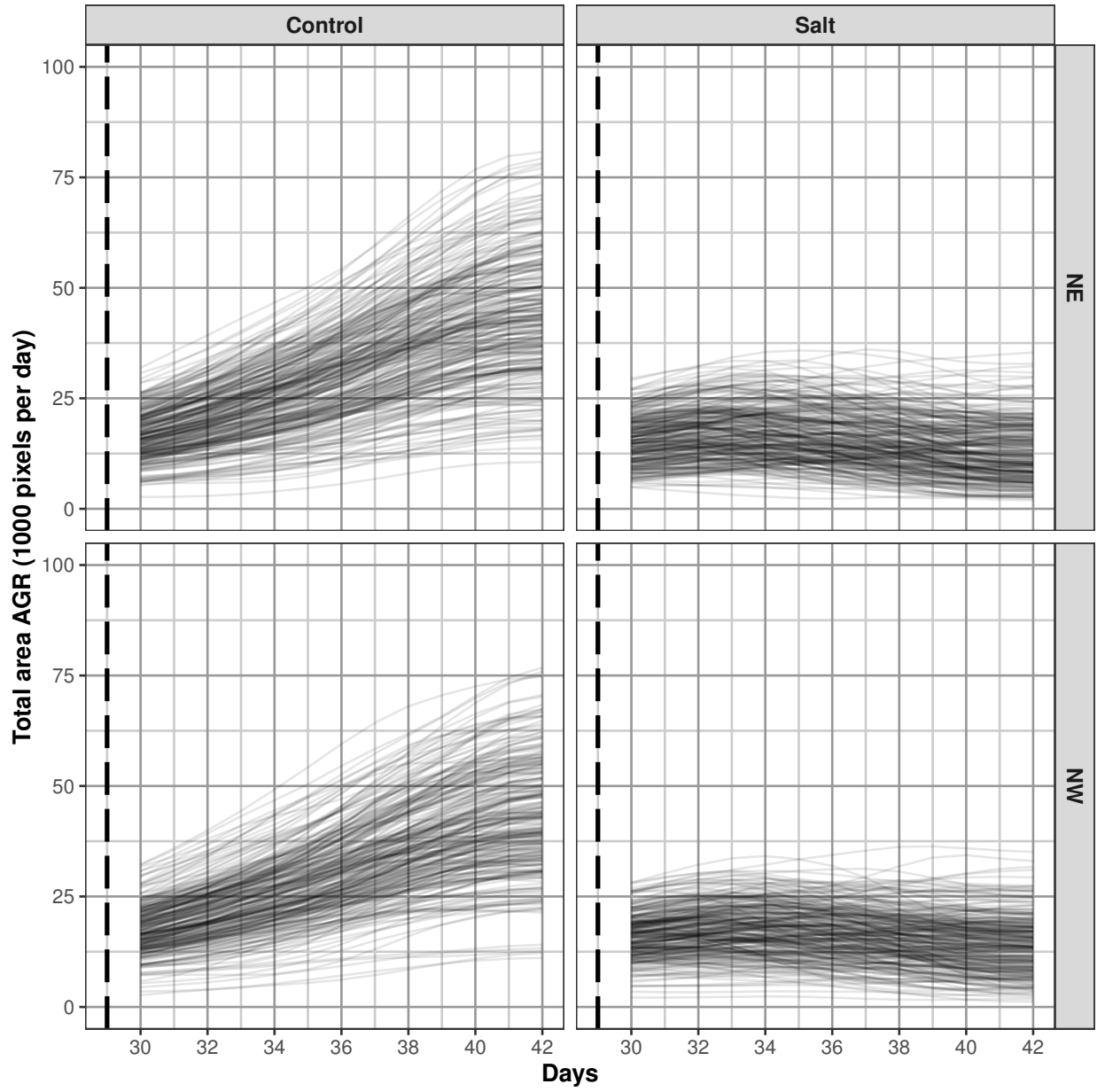


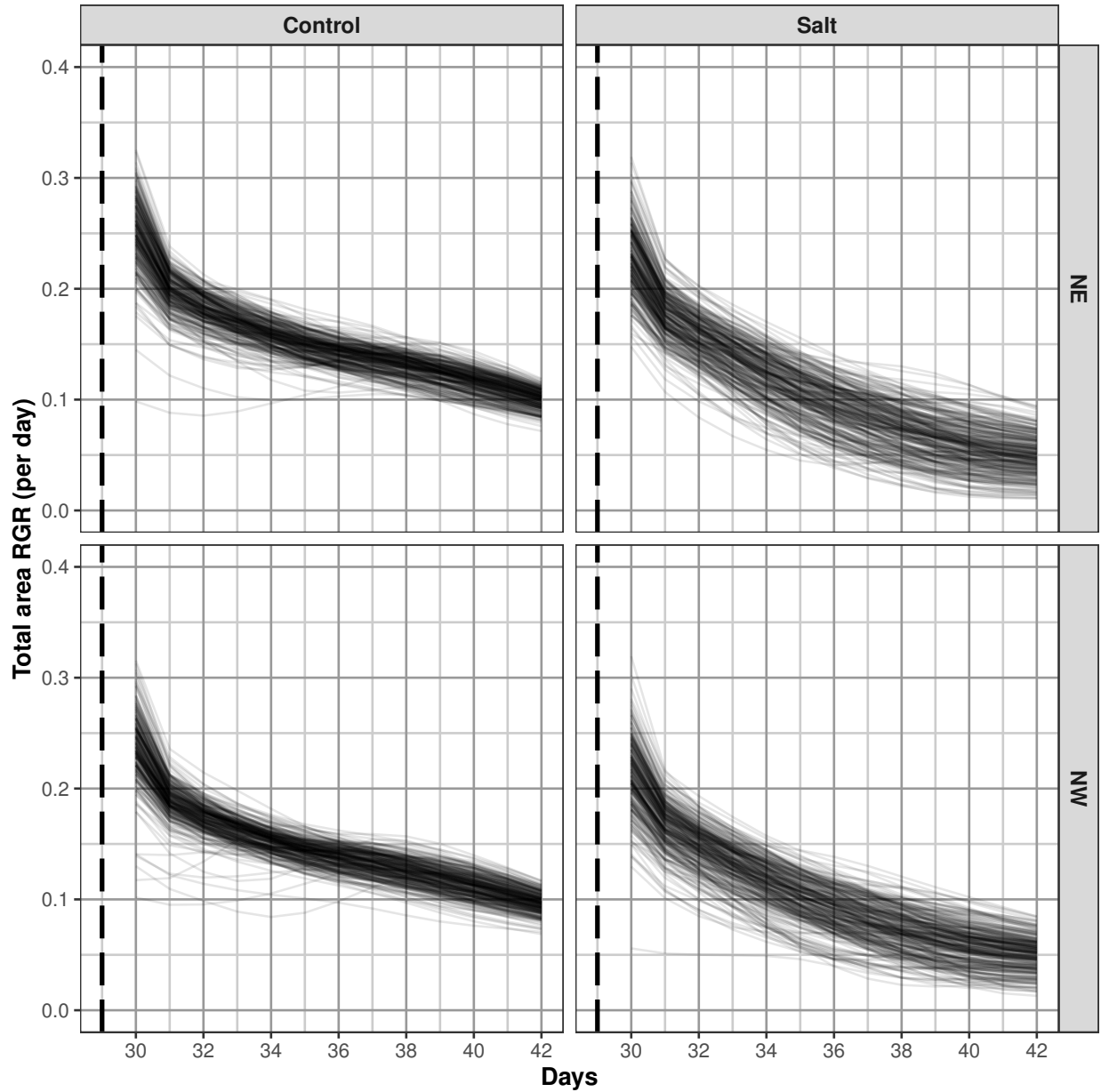


Plot smoothed profiles for all longitudinal responses

```
nresp.smooth <- length(responses.smooth.plot)
limits <- list(c(0,1000), c(0,100), c(0.0,0.40))
for (k in 1:nresp.smooth)
{
  plt <- plotLongitudinal(data = longi.dat, response = responses.smooth.plot[k],
                          y.title = titles.smooth[k], x="xDays+35.42857143", printPlot=FALSE)
  plt <- plt + geom_vline(xintercept=29, linetype="longdash", size=1) +
    scale_x_continuous(breaks=seq(28, 42, by=2)) +
    scale_y_continuous(limits=limits[[k]])
  print(plt)
}
```







### Step 3: Choose the smoothing method and DF

This step has been omitted.

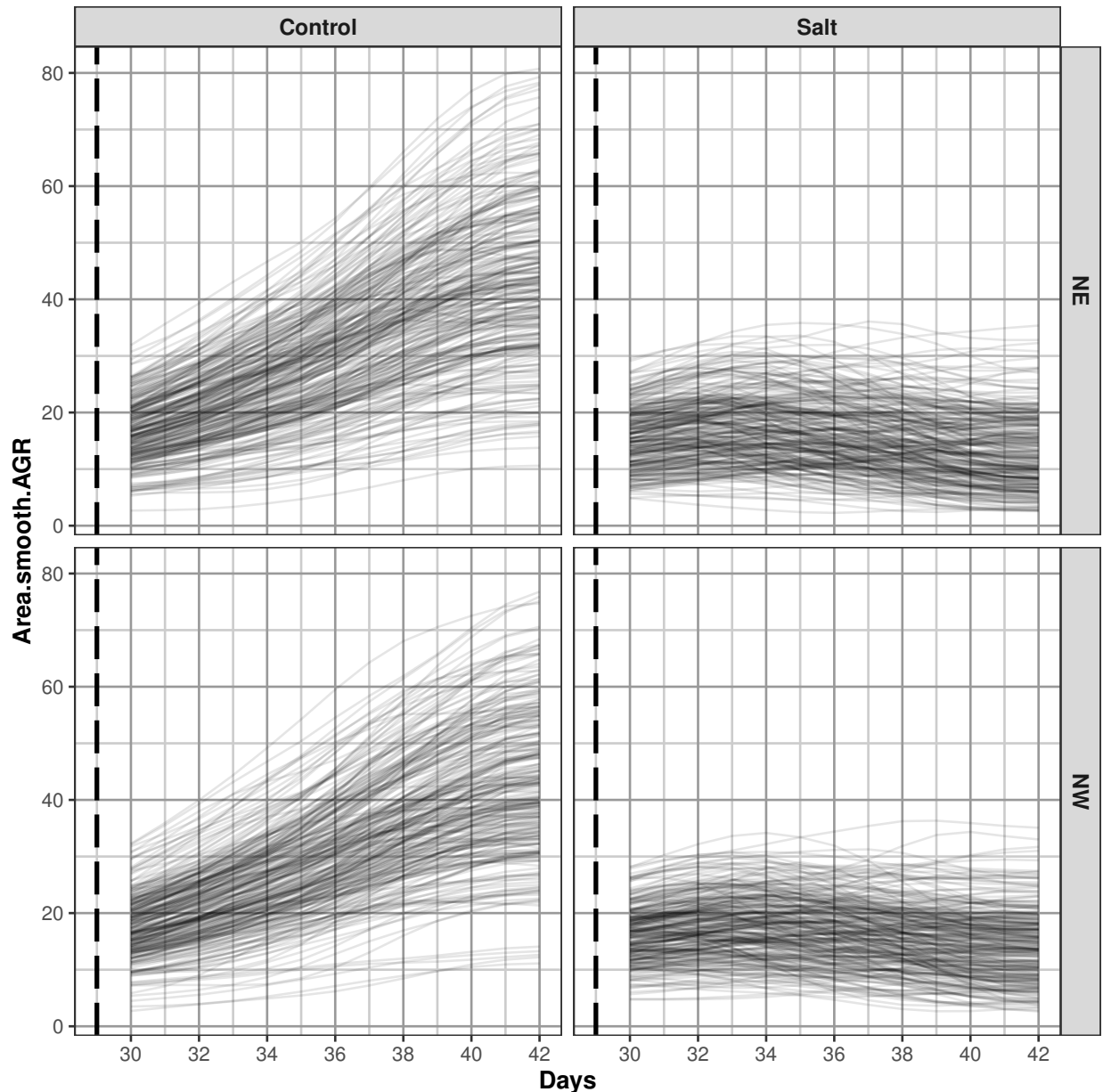
### Step 4: Identify potential outliers and clean the data

It has been decided that plants whose smoothed AGR are less than 2.5 after Day 40 are growing so slowly as to be considered anomalous. These plants are identified using `plotAnom`. Their values on Day 42 are printed. The plants are plotted without the anomalous plants followed by a plot of just the anomalous plants. The images of these anomalous plants were examined and no particular problems were identified with them. They were retained in the data.

```

anom.ID <- vector(mode = "character", length = 0L)
response <- "Area.smooth.AGR"
cols.output <- c("Snapshot.ID.Tag", "Smarthouse", "Lane", "Position",
                "Treatment.1", "Genotype.ID", "Days")
anomalous <- plotAnom(longi.dat, response=response, lower=2.5, start.time=40,
                    x = "xDays+35.42857143", vertical.line=29, breaks=seq(28, 42, by=2),
                    whichPrint=c("innerPlot"), y.title=response)

```



```

subs <- subset(anomalous$data, Area.smooth.AGR.anom & Days==42)
if (nrow(subs) == 0)
{ cat("\n#### No anomalous data here\n\n")
} else
{
  subs <- subs[order(subs["Smarthouse"],subs["Treatment.1"], subs[response]),]
}

```

```

print(subs[c(cols.output, response)])
anom.ID <- unique(c(anom.ID, subs$Snapshot.ID.Tag))
outerPlot <- anomalous$outerPlot + geom_text(data=subs,
                                             aes_string(x = "xDays+35.42857143",
                                                         y = response,
                                                         label="Snapshot.ID.Tag"),
                                             size=3, hjust=0.7, vjust=0.5)

print(outerPlot)
}

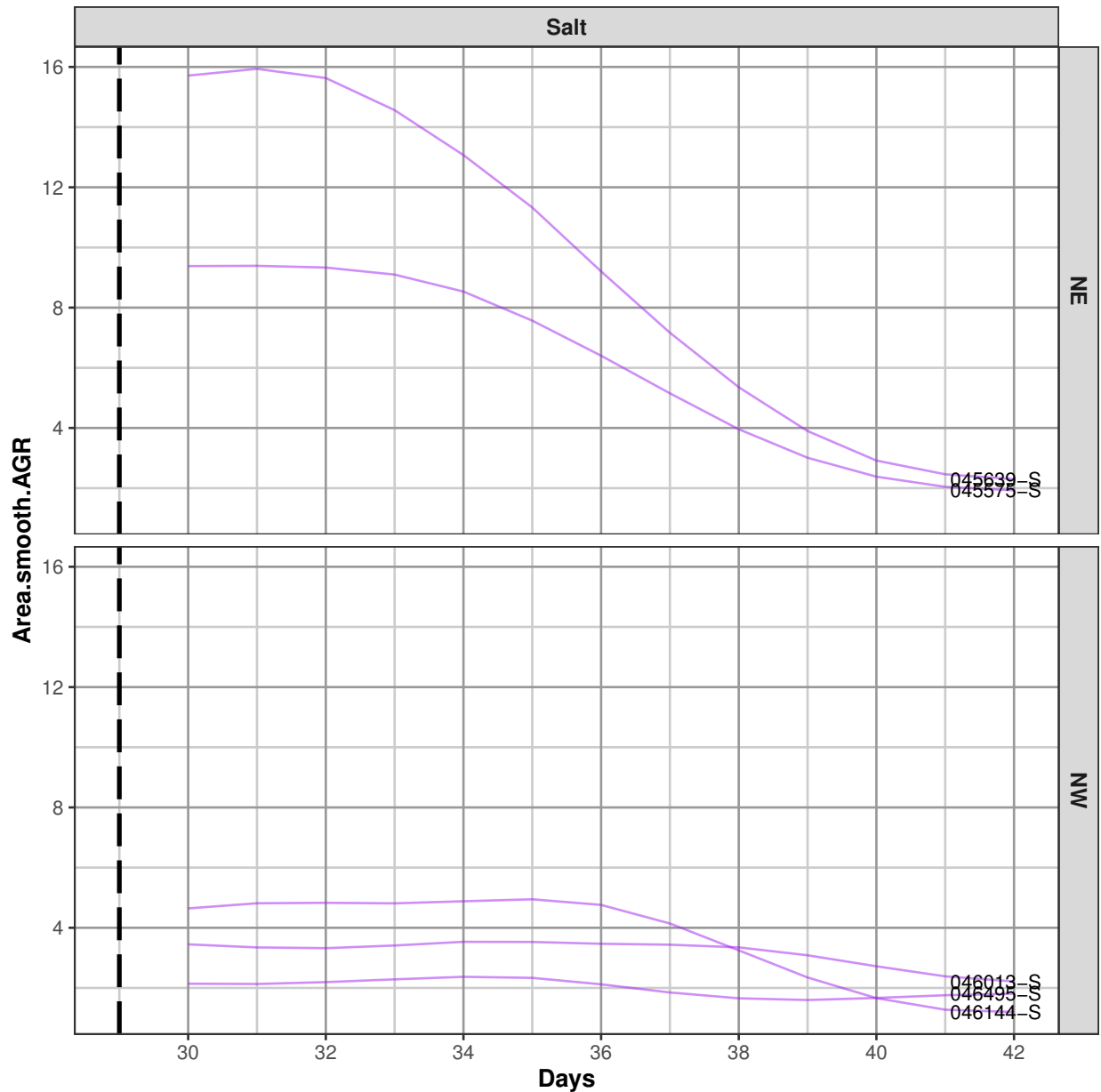
```

```
## Warning in xtfrm.data.frame(x): cannot xtfrm data frames
```

```
## Warning in xtfrm.data.frame(x): cannot xtfrm data frames
```

```
## Warning in xtfrm.data.frame(x): cannot xtfrm data frames
```

```
##      Snapshot.ID.Tag Smarthouse Lane Position Treatment.1 Genotype.ID Days
## 1680      045575-S      NE      6      10      Salt      121701 42
## 2534      045639-S      NE      9      6      Salt      122000 42
## 9282      046144-S      NW      7      5      Salt      121133 42
## 14000     046495-S      NW     22     10      Salt      120952 42
## 7532      046013-S      NW      1     12      Salt      121852 42
##      Area.smooth.AGR
## 1680      1.926575
## 2534      2.297119
## 9282      1.199223
## 14000     1.809133
## 7532      2.216099
```



## Step 5: Extract per-cart traits

A range of single-value plant responses are formed in Snapshot.ID.Tag order.

### Step 5(a): Set up a data frame with factors only

```
cart.dat <- longi.dat[longi.dat$Days == 31,
  c("Smarthouse", "Lane", "Position", "Snapshot.ID.Tag",
    "xPosn", "xMainPosn",
    "Zones", "xZones", "SHZones", "ZLane", "ZMainplots", "Subplots",
    "Genotype.ID", "Treatment.1")]
cart.dat <- cart.dat[do.call(order, cart.dat), ]
```

## Step 5(b): Get responses based on first and last date.

```
# Observation for first and last date
cart.dat <- cbind(cart.dat, getTimesSubset(responses.image, data = longi.dat,
                                          which.times = c(31), suffix = "first"))
cart.dat <- cbind(cart.dat, getTimesSubset(responses.image, data = longi.dat,
                                          which.times = c(42), suffix = "last"))
cart.dat <- cbind(cart.dat, getTimesSubset(c("WUI.cum"),
                                          data = longi.dat,
                                          which.times = c(42), suffix = "last"))
responses.smooth <- paste(responses.image, "smooth", sep=".")
cart.dat <- cbind(cart.dat, getTimesSubset(responses.smooth, data = longi.dat,
                                          which.times = c(31), suffix = "first"))
cart.dat <- cbind(cart.dat, getTimesSubset(responses.smooth, data = longi.dat,
                                          which.times = c(42), suffix = "last"))

# Growth rates over whole period.
tottime <- 42 - 31
cart.dat <- within(cart.dat,
  {
    Area.AGR <- (Area.last - Area.first)/tottime
    Area.RGR <- log(Area.last / Area.first)/tottime
  })

# Calculate water index over whole period
cart.dat <- merge(cart.dat,
  intervalWUI("Area", water.use = "Water.Loss",
             start.times = c(31),
             end.times = c(42),
             suffix = NULL,
             data = longi.dat, include.total.water = TRUE),
  by = c("Snapshot.ID.Tag"))
names(cart.dat)[match(c("Area.WUI", "Water.Loss.Total"), names(cart.dat))] <-
  c("Area.Overall.WUI", "Water.Loss.Overall")
cart.dat$Water.Loss.rate.Overall <- cart.dat$Water.Loss.Overall / (42 - 31)
```

## Step 5(c): Add growth rates and water indices for intervals

```
# Set up intervals
start.days <- list(31,35,31,38)
end.days <- list(35,38,38,42)
suffices <- list("31to35", "35to38", "31to38", "38to42")

# Growth rates for specific intervals from the smoothed data by differencing
for (r in responses.smooth)
{ for (k in 1:length(suffices))
  {
    cart.dat <- merge(cart.dat,
      intervalGRdiff(r,
                    which.rates = c("AGR", "RGR"),
                    start.times = start.days[k][[1]],
                    end.times = end.days[k][[1]],
                    suffix.interval = suffices[k][[1]],
```



```

                                data = longi.dat),
                                by = "Snapshot.ID.Tag")
}
}

# Water indices for specific intervals from the unsmoothed and smoothed data
for (k in 1:length(suffices))
{
  cart.dat <- merge(cart.dat,
                    intervalWUI("Area", water.use = "Water.Loss",
                                start.times = start.days[k][[1]],
                                end.times = end.days[k][[1]],
                                suffix = suffices[k][[1]],
                                data = longi.dat, include.total.water = TRUE),
                    by = "Snapshot.ID.Tag")
  names(cart.dat)[match(paste("Area.WUI", suffices[k][[1]], sep="."),
                        names(cart.dat))] <- paste("Area.WUI", suffices[k][[1]], sep=".")
  cart.dat[paste("Water.Loss.rate", suffices[k][[1]], sep=".")] <-
    cart.dat[[paste("Water.Loss.Total", suffices[k][[1]], sep=".")]] /
      (end.days[k][[1]] - start.days[k][[1]])
}

cart.dat <- with(cart.dat, cart.dat[order(Snapshot.ID.Tag), ])

```

## Form continuous and interval SIITs

This experiment involved the extra step of calculating a measure of shoot ion-independent tolerance (SIIT) of pairs of plants, control and a salt-treated co-located plants.

### Calculate continuous values

```

cols.retained <- c("Snapshot.ID.Tag", "Smarthouse", "Lane", "Position",
                  "Days", "Snapshot.Time.Stamp", "Hour", "xDays",
                  "Zones", "xZones", "SHZones", "ZLane", "ZMainplots",
                  "xMainPosn", "Genotype.ID")
responses.GR <- c("Area.smooth.AGR", "Area.smooth.AGR", "Area.smooth.RGR")
suffices.results <- c("diff", "SIIT", "SIIT")
responses.SIIT <- unlist(Map(paste, responses.GR, suffices.results, sep="."))

longi.SIIT.dat <-
  twoLevelOpcreate(responses.GR, longi.dat, suffices.treatment=c("C", "S"),
                  operations = c("-", "/", "/"), suffices.results = suffices.results,
                  columns.retained = cols.retained,
                  by = c("Smarthouse", "Zones", "ZMainplots", "Days"))
longi.SIIT.dat <- with(longi.SIIT.dat,
                      longi.SIIT.dat[order(Smarthouse, Zones, ZMainplots, Days),])

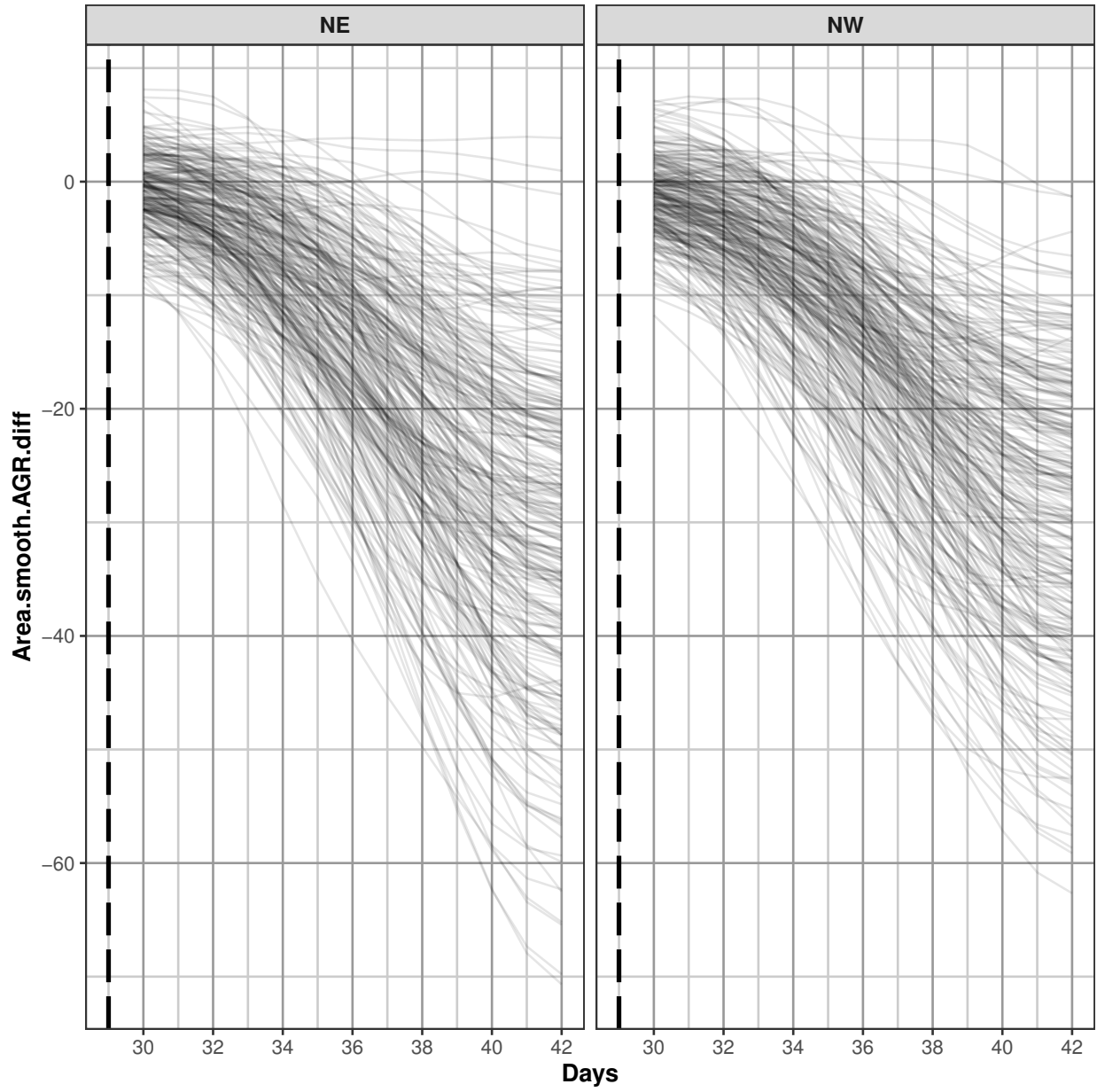
# Plot SIIT profiles
k <- 2
nresp <- length(responses.SIIT)
limits <- with(longi.SIIT.dat, list(c(min(Area.smooth.AGR.diff, na.rm=TRUE),
                                       max(Area.smooth.AGR.diff, na.rm=TRUE)),

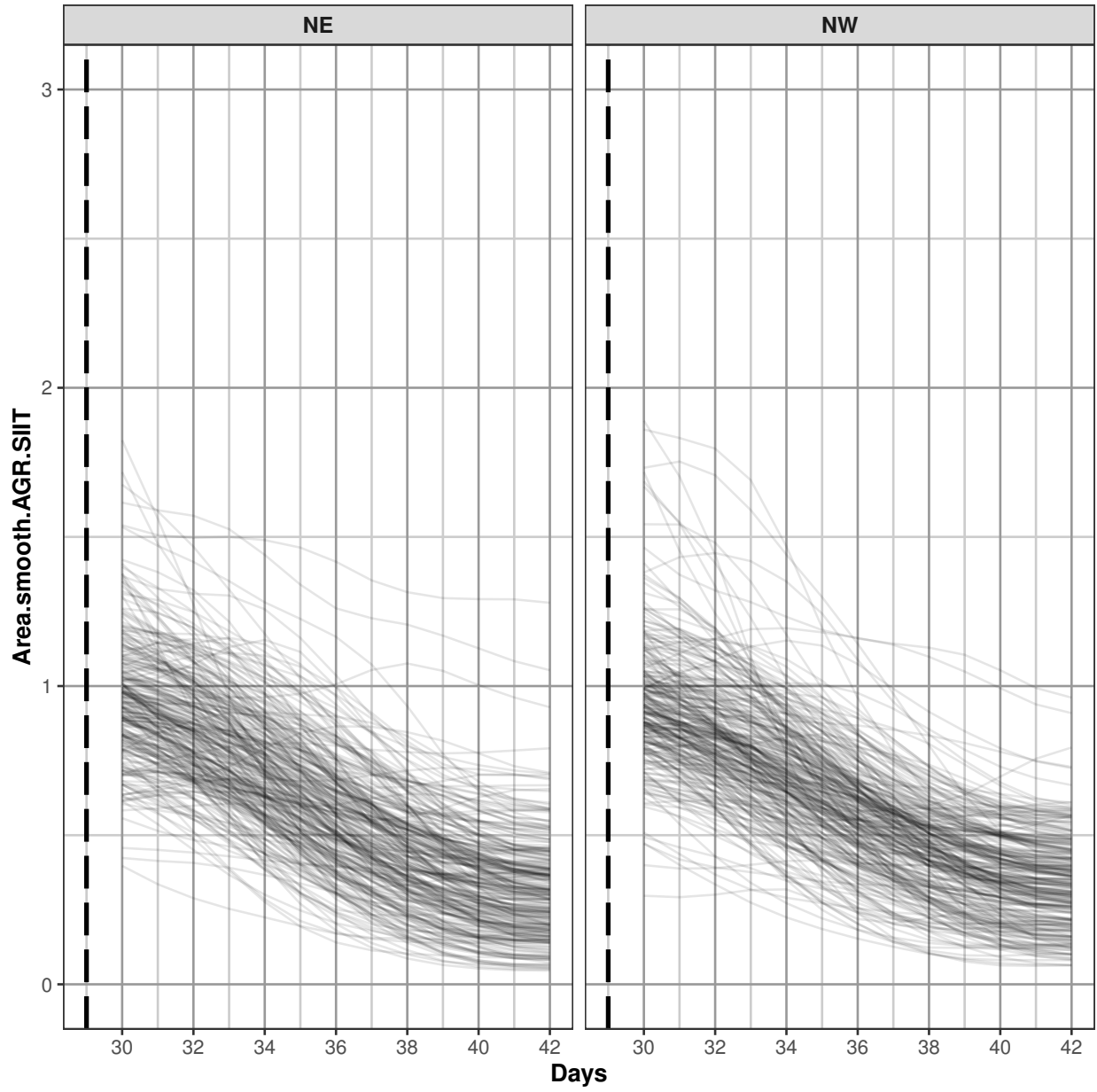
```

```

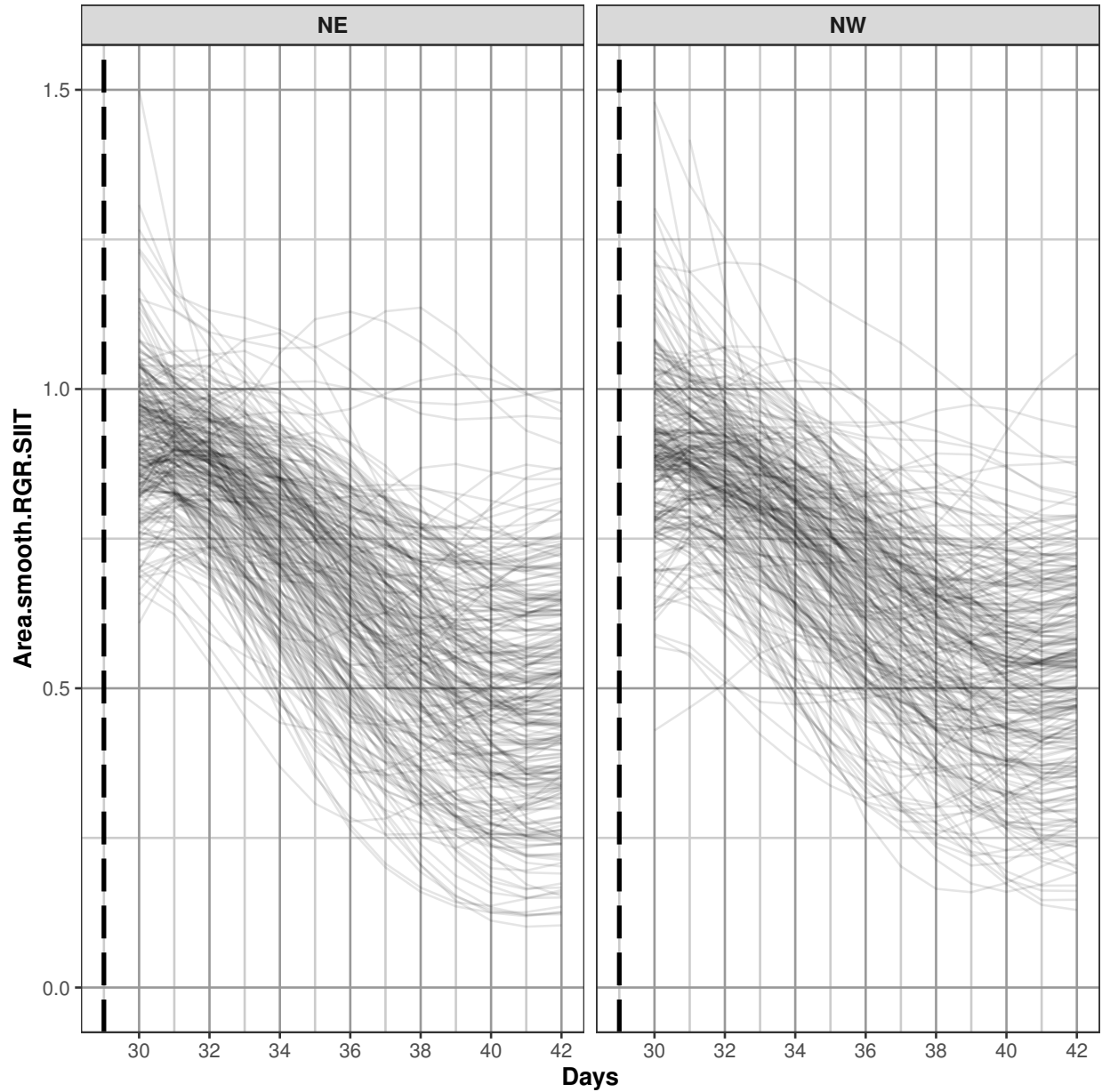
                                c(0,3),
                                c(0,1.5)))
#Plots
for (k in 1:nresp)
{
  plt <- plotLongitudinal(data = longi.SIIT.dat, x="xDays+35.42857143",
                          response = responses.SIIT[k],
                          y.title=responses.SIIT[k],
                          facet.x="Smarthouse", facet.y=".", printPlot=FALSE, )
  plt <- plt + geom_vline(xintercept=29, linetype="longdash", size=1) +
    scale_x_continuous(breaks=seq(28, 42, by=2)) +
    scale_y_continuous(limits=limits[[k]])
  print(plt)
}

```





## Warning: Removed 1 row(s) containing missing values (geom\_path).



Calculate interval SIITs and check for large values for SIIT for Days 31to35

```
suffices <- list("31to35","35to38","31to38","38to42")
response <- "Area.smooth.RGR.31to35"
SIIT <- paste(response, "SIIT", sep=".")
responses.SIITinterval <- as.vector(outer("Area.smooth.RGR", suffices, paste, sep="."))

cart.SIIT.dat <- twoLevel0pcreate(responses.SIITinterval, cart.dat,
                                suffices.treatment=c("C","S"),
                                suffices.results="SIIT",
                                columns.suffixed="Snapshot.ID.Tag")

tmp<-na.omit(cart.SIIT.dat)
print(summary(tmp[SIIT]))
```

```

## Area.smooth.RGR.31to35.SIIT
## Min. :0.4077
## 1st Qu.:0.7160
## Median :0.7999
## Mean :0.7908
## 3rd Qu.:0.8688
## Max. :1.1885

big.SIIT <- with(tmp, tmp[tmp[SIIT] > 1.15, c("Snapshot.ID.Tag.C", "Genotype.ID",
      paste(response, "C", sep="."),
      paste(response, "S", sep="."), SIIT)])

big.SIIT <- big.SIIT[order(big.SIIT[SIIT]),]

## Warning in xtfrm.data.frame(x): cannot xtfrm data frames

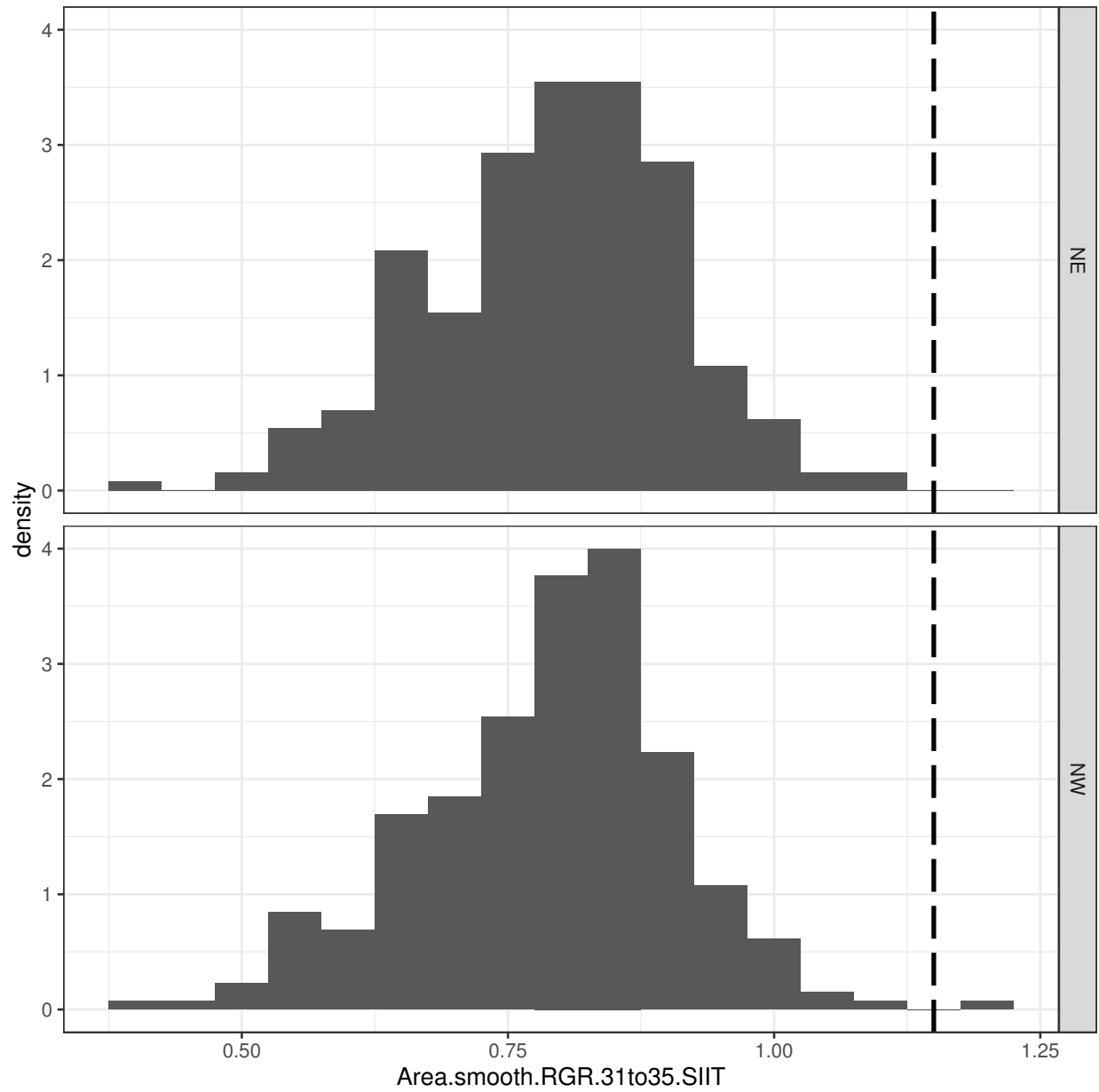
print(big.SIIT)

## Snapshot.ID.Tag.C Genotype.ID Area.smooth.RGR.31to35.C
## 325 046129-C 122090 0.1310631
## Area.smooth.RGR.31to35.S Area.smooth.RGR.31to35.SIIT
## 325 0.1557642 1.188467

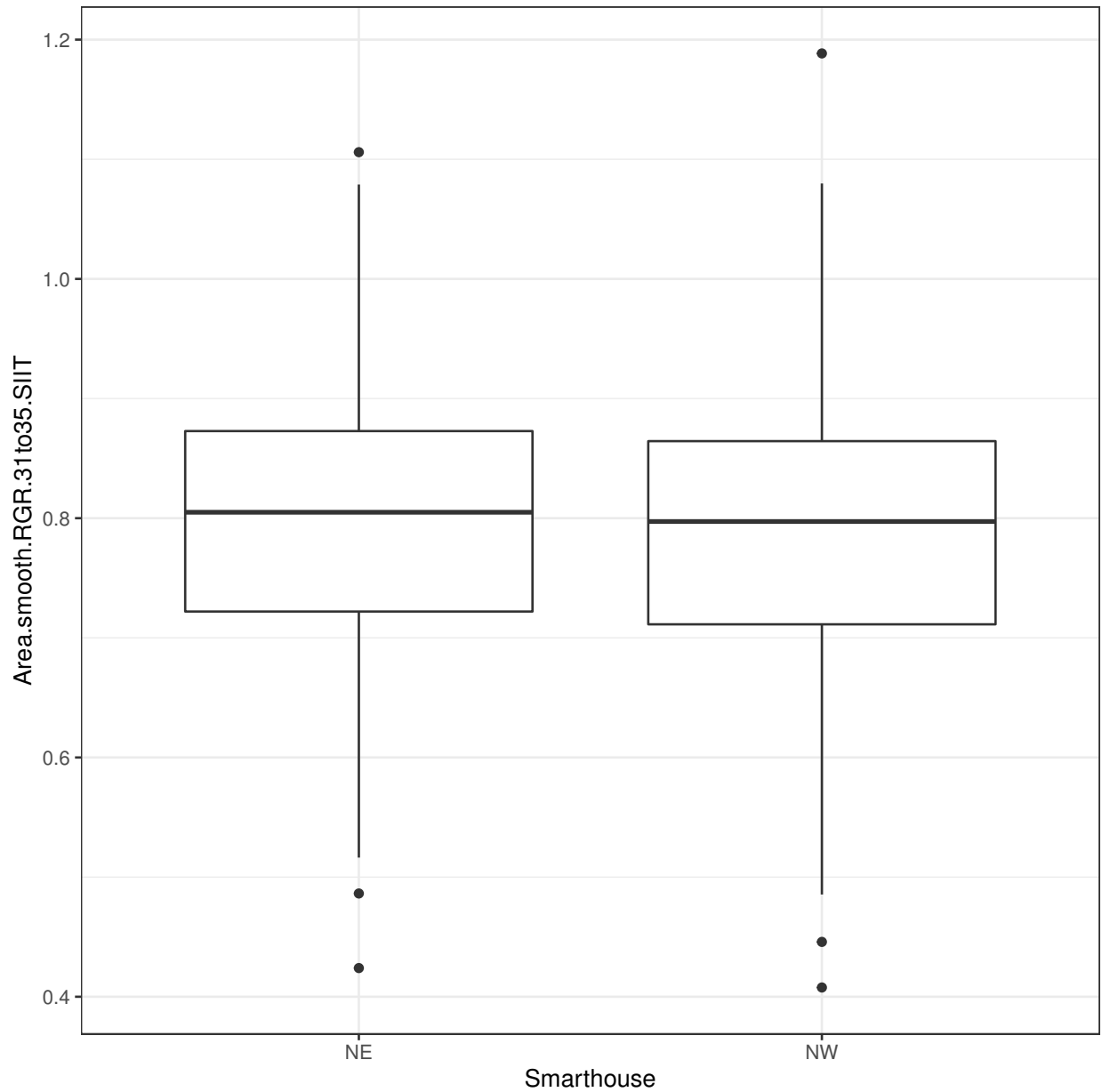
plt <- ggplot(tmp, aes_string(SIIT)) +
  geom_histogram(aes(y = ..density..), binwidth=0.05) +
  geom_vline(xintercept=1.15, linetype="longdash", size=1) +
  theme_bw() + facet_grid(Smarthouse ~.)

print(plt)

```



```
plt <- ggplot(tmp, aes_string(x="Smarthouse", y=SIIT)) +
  geom_boxplot() + theme_bw()
print(plt)
```



```
remove(tmp)
```

## Save image

```
save.image("Rice.RData")
```

## References

Al-Tamimi, N, Brien, C.J., Oakey, H., Berger, B., Saade, S., Ho, Y. S., Schmockel, S. M., Tester, M. and Negrao, S. (2016) New salinity tolerance loci revealed in rice using high-throughput non-invasive phenotyping. *Nature Communications*, **7**, 13342.

Brien, C., Jewell, N., Garnett, T., Watts-Williams, S. J., & Berger, B. (2020). Smoothing and extraction of



traits in the growth analysis of noninvasive phenotypic data. *Plant Methods*, **16**, 36. <http://dx.doi.org/10.1186/s13007-020-00577-6>.