

# The Tomato example: illustrating the smoothing and extraction of traits (SET) using growthPheno Version 2.x

Chris Brien

10 March, 2023

This vignette illustrates the use of the two `growthPheno` (Brien, 2023) wrapper functions `traitSmooths` and `traitExtractFeatures` that are key to carrying out the smoothing and extracting traits (SET) method described by Brien et al. (2020). The Tomato example, used here, is the example that Brien et al. (2020) used to illustrate the SET method. More details on the rationale for this method are available in Brien et al. (2020, Methods section).

Here, the process has been modified from that described in the paper to take advantage of the new wrapper functions and other new capabilities that have been built into in Version 2.x of `growthPheno`. In particular, both natural cubic smoothing splines (NCSS) and P-splines (PS) are investigated for smoothing not only the Projected Shoot Area (PSA), but also the Water Use (WU). A segmented smooth, as suggested in Brien et al. (2020), is used to allow for a discontinuity in the growth resulting from unintentional, restricted watering for three days following imaging on DAP 39.

Two different approaches are shown for smoothing the two traits:

**PSA:** For this trait, we first use `traitSmooths` to compare several smooths using logarithmic smoothing and then automatically choose a P-spline smooth whose `lambda` value is in the middle of the values for which smooths have been obtained. This is then followed by a comparison of two contending smooths. Finally, the chosen smooth is extracted and added to the data.

**WU:** A more time-efficient approach is taken with this trait. First several direct smooths are compared and stored. Then plots of two contending smooths amongst the stored smooths are compared. Finally the chosen smooth is extracted from the stored smooths.

## Initialize

### Set up characters for variable names and titles

```
# The responses
responses <- c("PSA", paste("PSA", c("AGR", "RGR"), sep = "."))
responses.smooth <- paste0("s", responses)

# Specify time intervals of homogeneous growth dynamics
DAP.endpts <- c(18,22,27,33,39,43,51)
nDAP.endpts <- length(DAP.endpts)
DAP.starts <- DAP.endpts[-nDAP.endpts]
DAP.stops <- DAP.endpts[-1]
DAP.segs <- list(c(DAP.endpts[1]-1, 39),
                c(40, DAP.endpts[nDAP.endpts]))
tune.fac <- c("Method","Type","Tuning")
#Functions to label the plot facets
labelAMF <- as_labeller(function(lev) paste(lev, "AMF"))
```

```
labelZn <- as_labeller(function(lev) paste("Zn:", lev, "mg/kg"))
vline.water <- list(geom_vline(xintercept=39, linetype="longdash",
                             alpha = 0.3, linewidth=1))
vline.DAP.endpts <- list(geom_vline(xintercept=DAP.starts, linetype="longdash",
                                   colour = "blue", alpha = 0.5, linewidth=0.75))
```

## Step I: Import the longitudinal data

In this step, the aim is to produce the data.frame `longi.dat` that contains the imaging variables, covariates and factors for the experiment.

### Load the pre-prepared data

```
data(tomato.dat)
```

### Copy the data to preserve the original data.frame

```
longi.dat <- tomato.dat
```

## Step II: Investigate the smoothing of the PSA and obtain growth rates

The growth rates are the Absolute Growth Rate (AGR) and the Relative Growth Rate (RGR) for the PSA, which must be calculated from the observed data by differencing consecutive observations for a plant. They will also be calculated from the smoothed traits by differencing, although `growthPheno` can also obtain growth rates using the first derivatives of the smooths.

### Fit three-parameter logistic curves to compare with spline curves

We fit a three-parameter logistic curve, using `nlme` (Pinheiro J., Bates D., and R Core Team, 2023), as an alternative to spline smoothing.

*Organize non-missing data into a grouped object*

```
logist.dat <- na.omit(longi.dat)
logist.grp <- nlme::groupedData(PSA ~ cDAP | Snapshot.ID.Tag,
                              data = logist.dat)
```

*Fit logistics to individuals and obtain fitted values*

```
logist.lis <- nlme::nlsList(SSlogis, logist.grp)
logist.dat$sPSA <- fitted(logist.lis)
logist.dat <- cbind(Tuning = factor("Logistic"), logist.dat)
```

### Compute smooths and growth rates of the PSA for a range of smoothing parameters

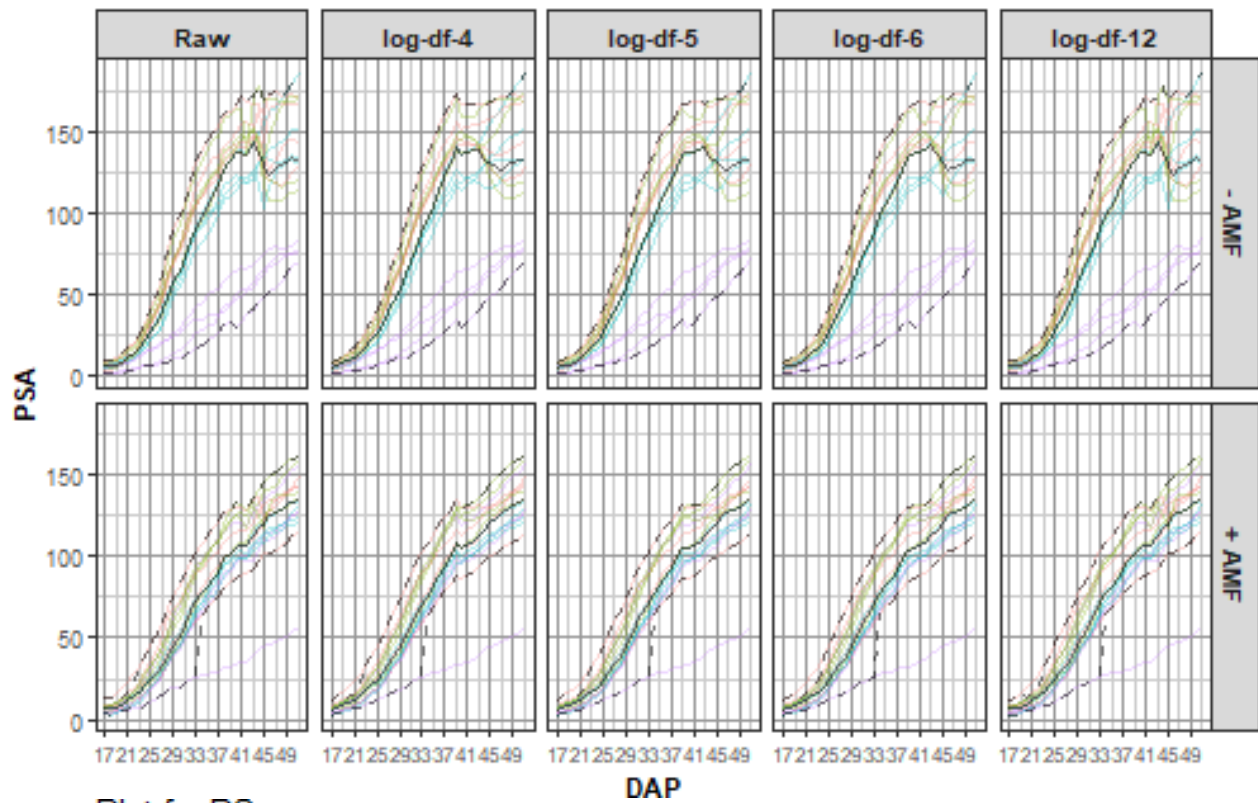
We begin by using the function `traitSmooth` to investigate a set of smooths for the PSA, employing all five `traitSmooth` steps of (i) Smooth, (ii) Profile plots, (iii) Median deviations plots, (iv) Choose a smooth, and (v) Chosen smooth plot. The only changes to the defaults for these five steps are to the `df` values that are investigated and to specify segmented smoothing. This includes allowing `traitSmooth` to choose automatically a single smooth as the chosen smooth. A segmented smooth involving two segments has also been specified,

as suggested by Brien et al. (2020). The breakpoint for the segments is DAP 39, it coinciding with the start of an unintentional, three-day restriction in the watering; thus, the segments consist of DAP 18–39 and DAP 40–51. The growth rates are calculated, by default, from both the unsmoothed trait PSA and the smoothed trait sPSA by difference, rather than from the spline derivatives. Thus, the growth rate calculation for the smoothed data matches that which is obligatory for the observed data. Also, three-parameter logistic curves are fitted to the data using the R package `nlme` and growth rates calculated for it. The default layouts of the three sets of plots produced are modified using the three arguments `profile.plot.args`, `meddevn.plot.args` and `chosen.plot.args`.

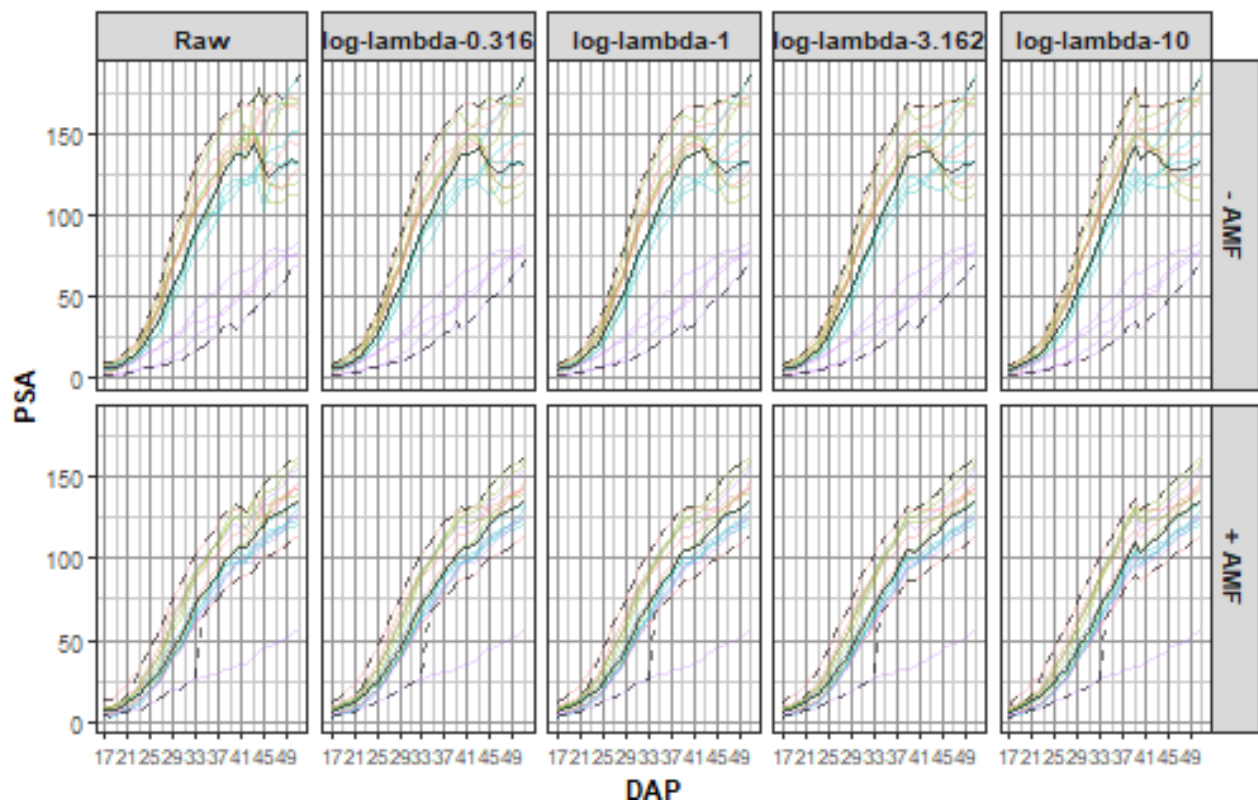
It is noted that the plots that are produced show that the logistic would not be an adequate fit for this data, especially after DAP 42.

```
suppressWarnings(  
  longi.dat <- traitSmooth(data = tomato.dat,  
    response = "PSA", response.smoothed = "sPSA",  
    individuals = "Snapshot.ID.Tag", times = "DAP",  
    keep.columns = c("AMF", "Zn"),  
    smoothing.args = args4smoothing(df = c(4:6, 12),  
      smoothing.segments = DAP.segs,  
      external.smooths = logist.dat),  
    profile.plot.args =  
      args4profile_plot(facet.y = "AMF",  
        colour.column = "Zn",  
        facet.labeller = labeller(AMF = labelAMF)),  
    meddevn.plot.args =  
      args4meddevn_plot(facet.y = "AMF",  
        facet.labeller = labeller(AMF = labelAMF)),  
    chosen.plot.args =  
      args4chosen_plot(facet.y = "AMF",  
        facet.labeller = labeller(AMF = labelAMF),  
        colour.column = "Zn",  
        ggplotFuncs = vline.DAP.endpts),  
    mergedata = tomato.dat))
```

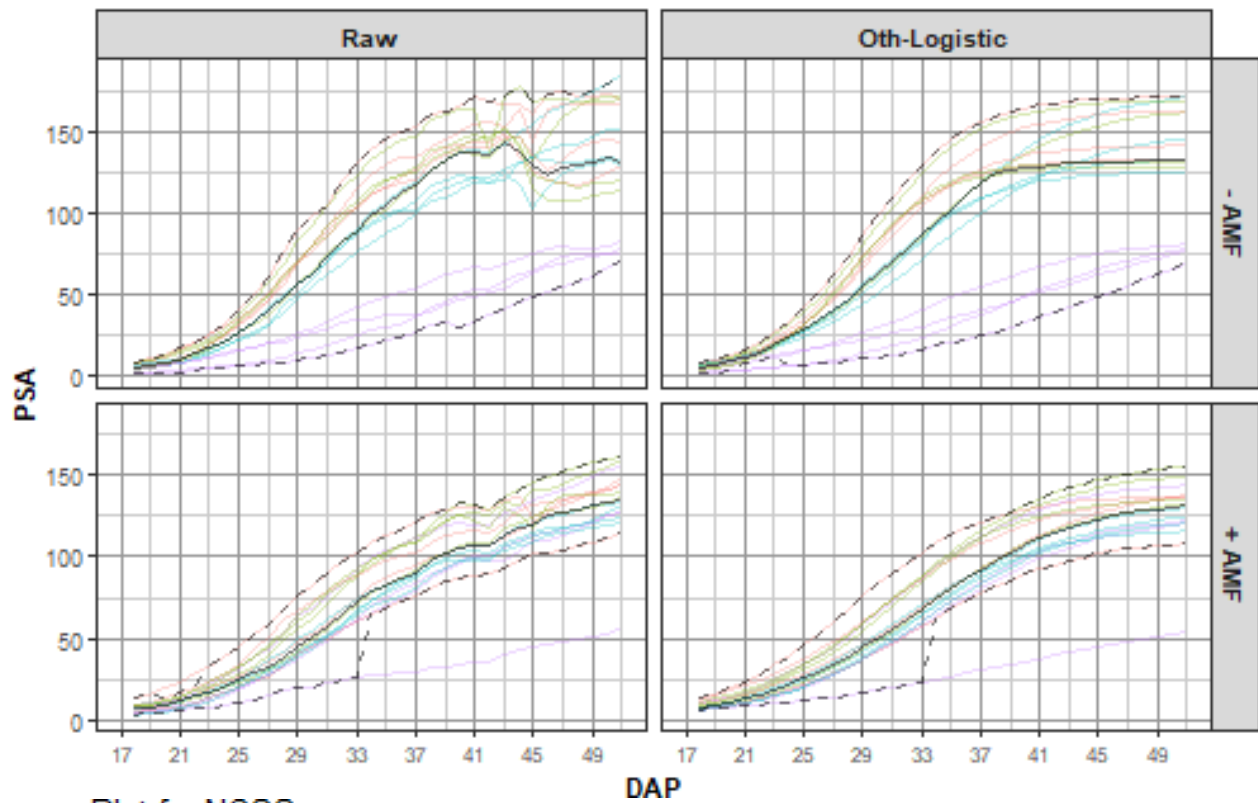
Plot for NCSS



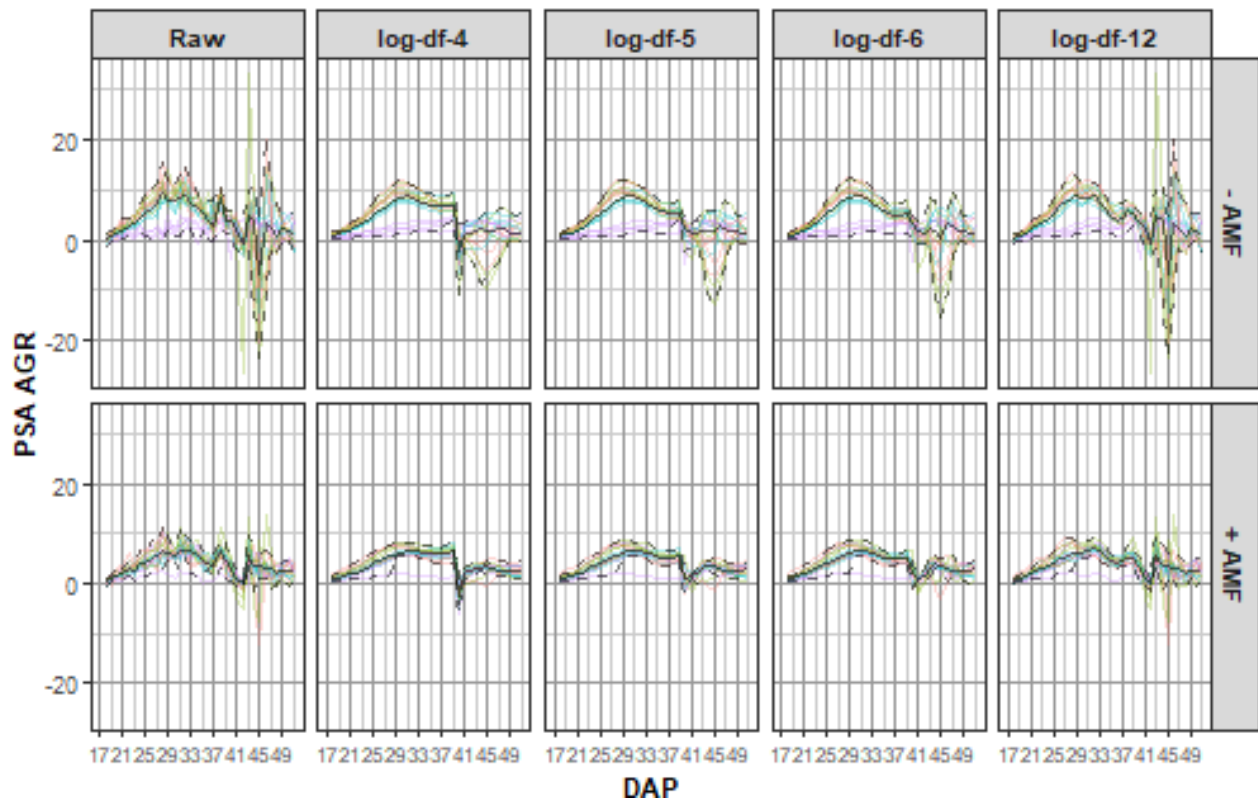
Plot for PS



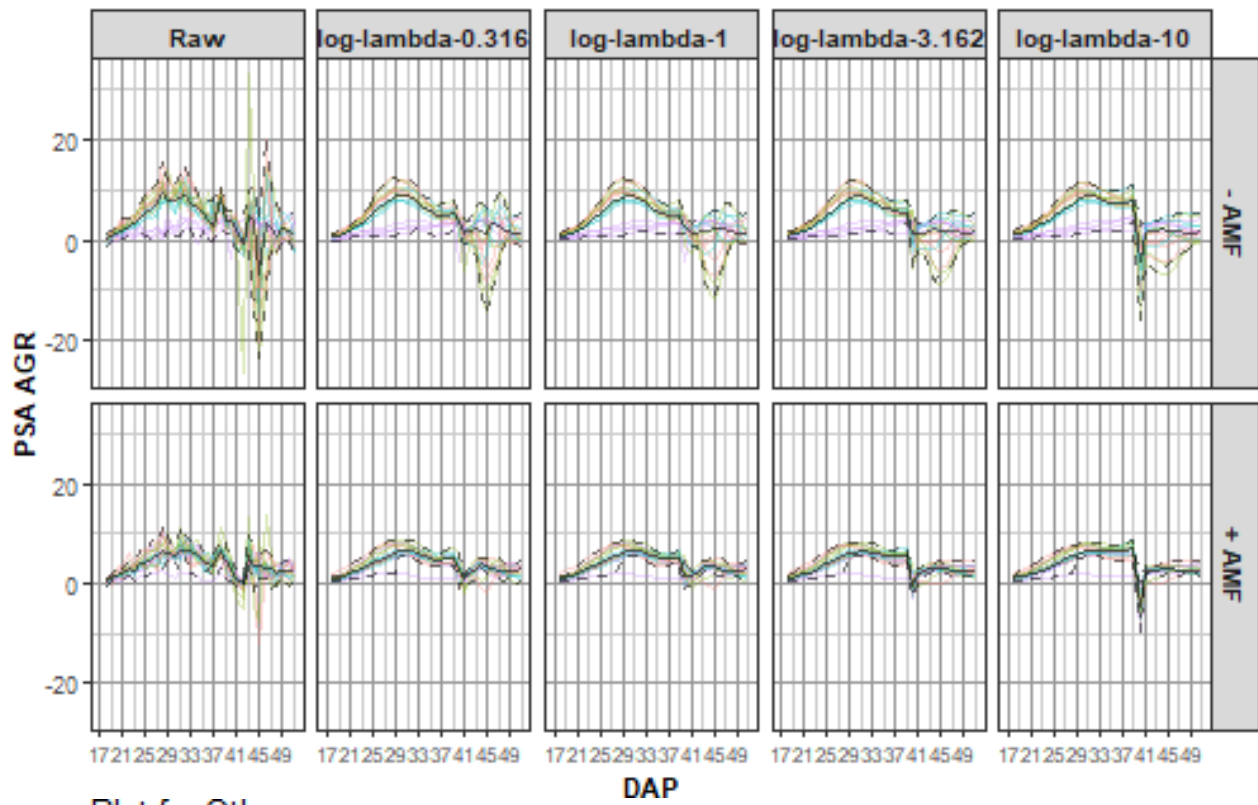
Plot for Other



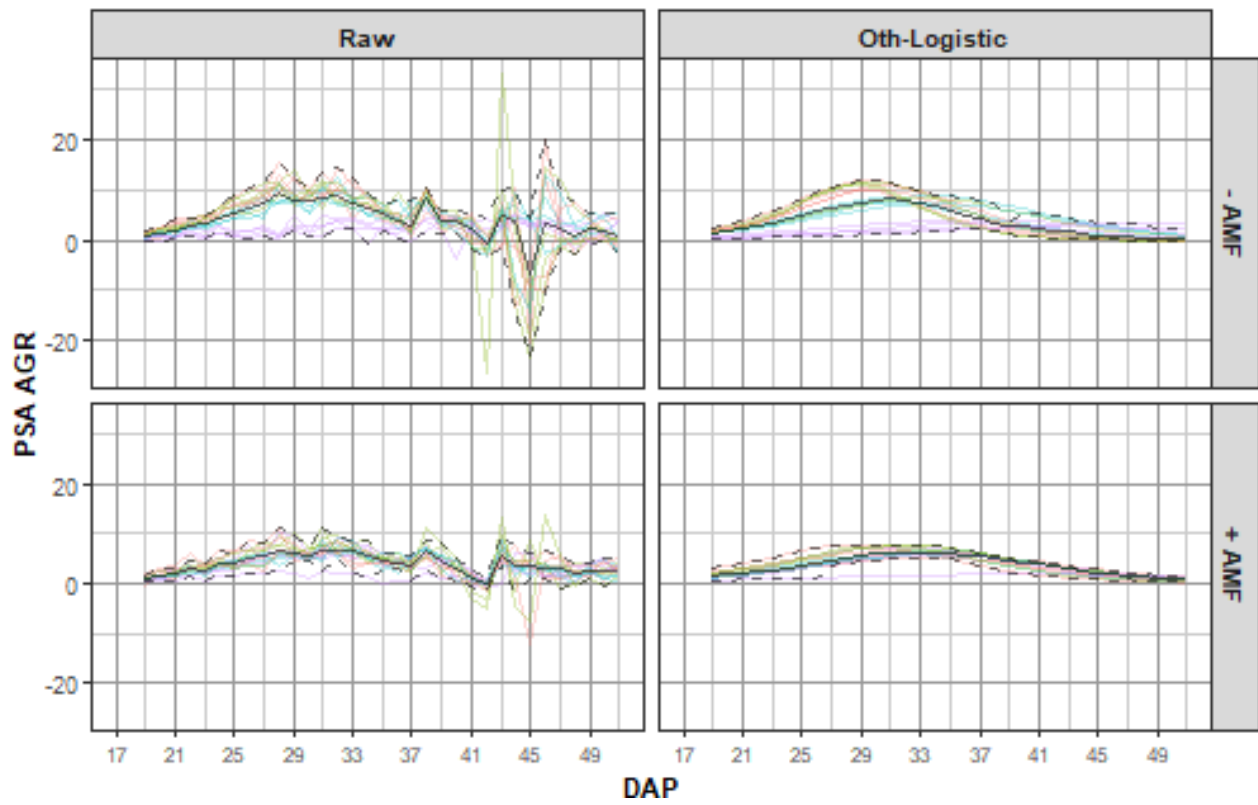
Plot for NCCS



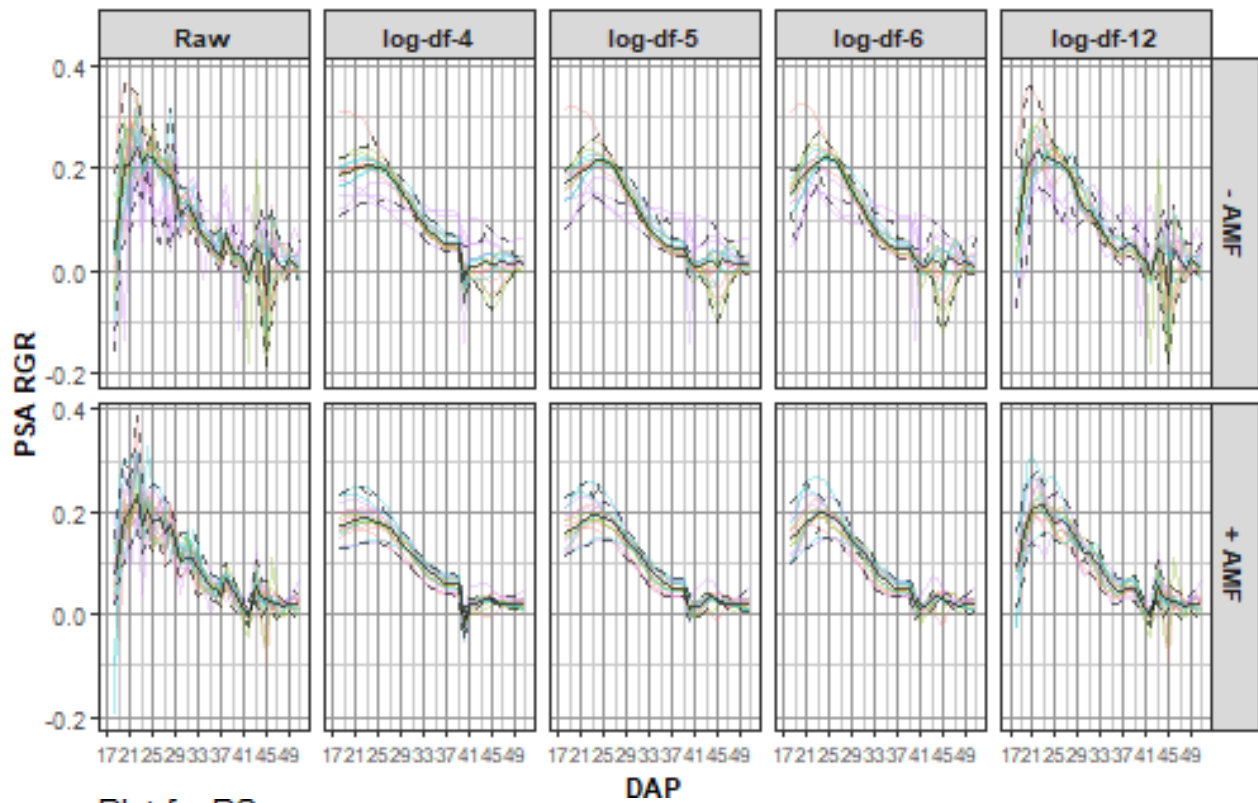
Plot for PS



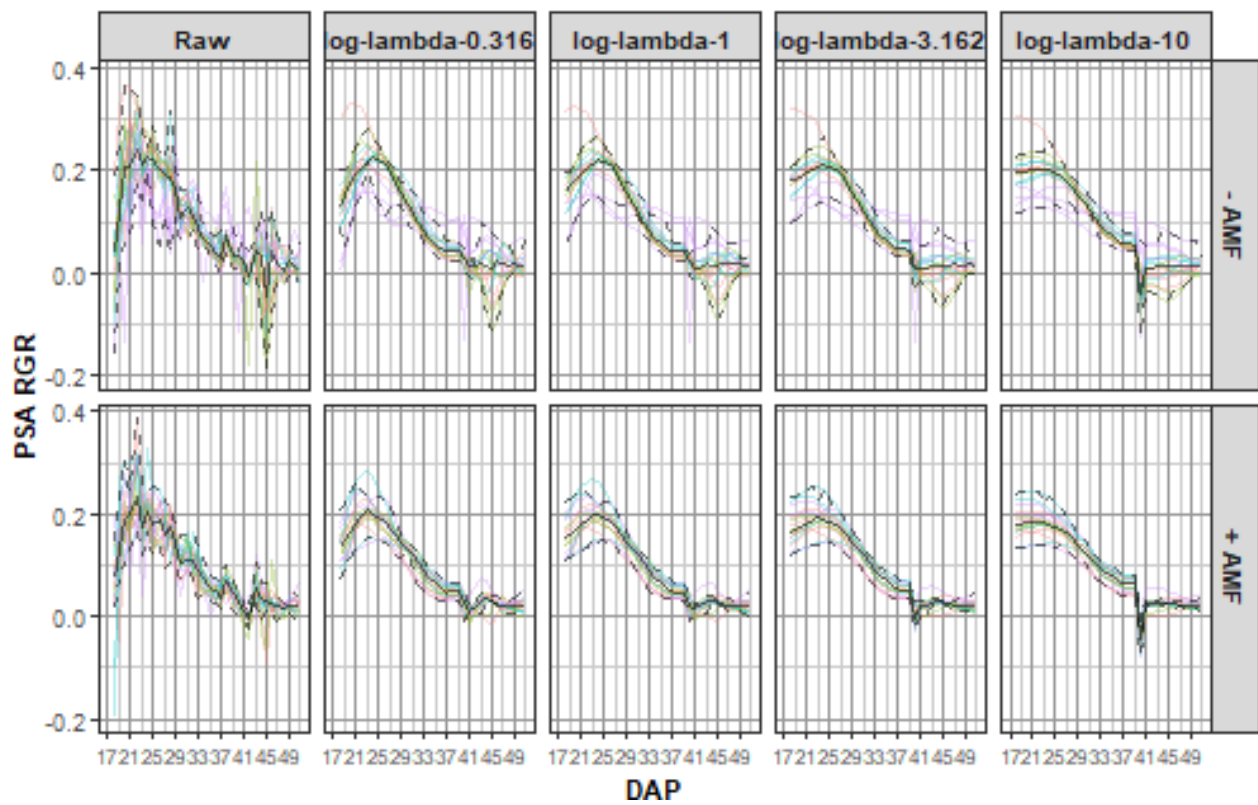
Plot for Other



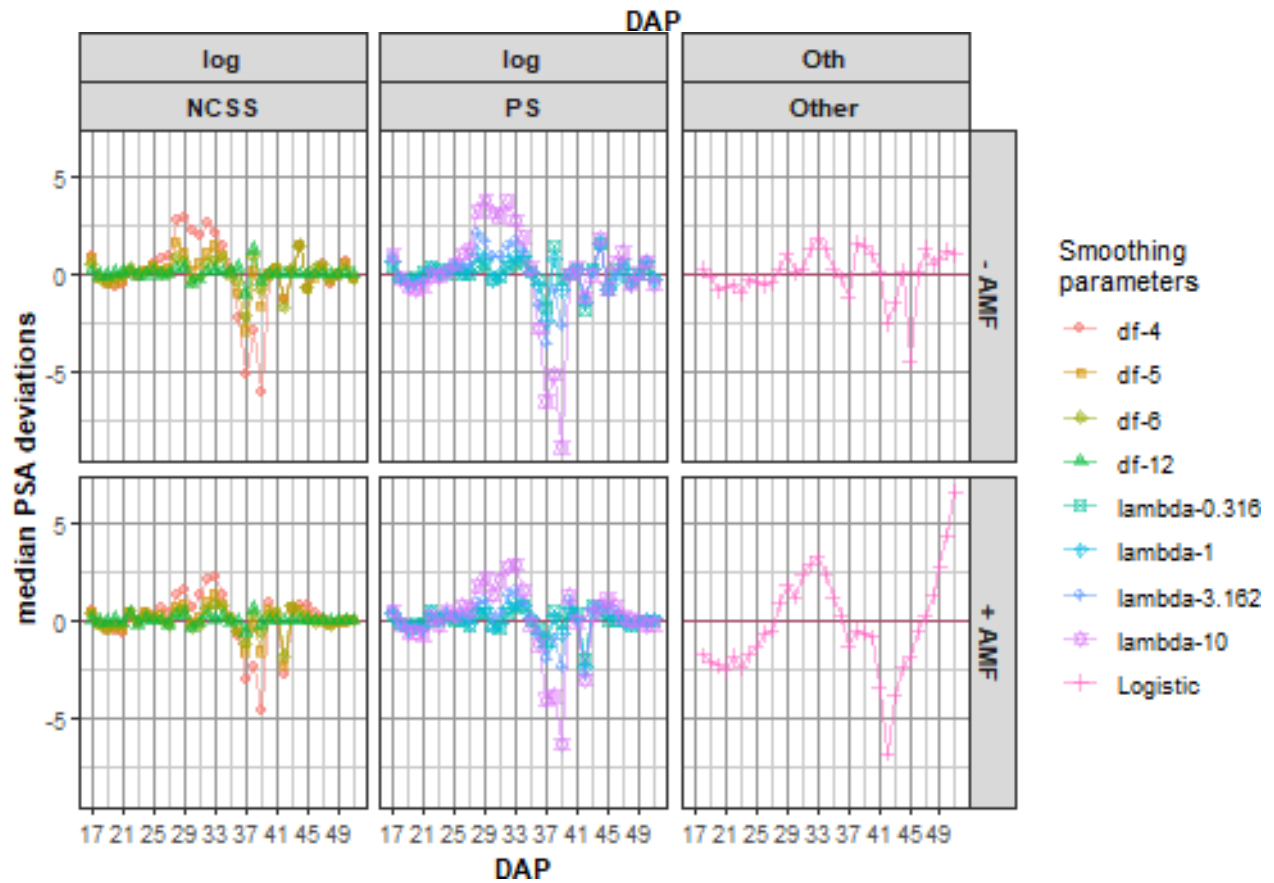
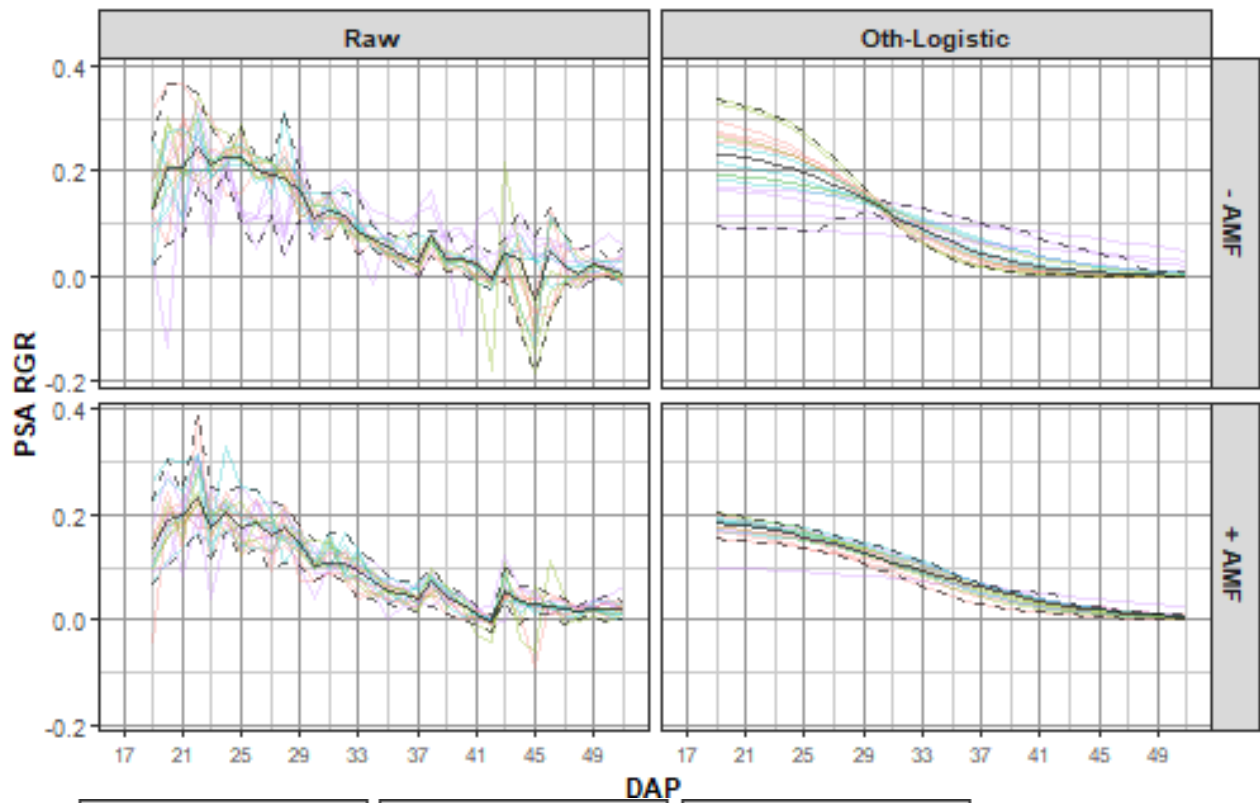
Plot for NCSS



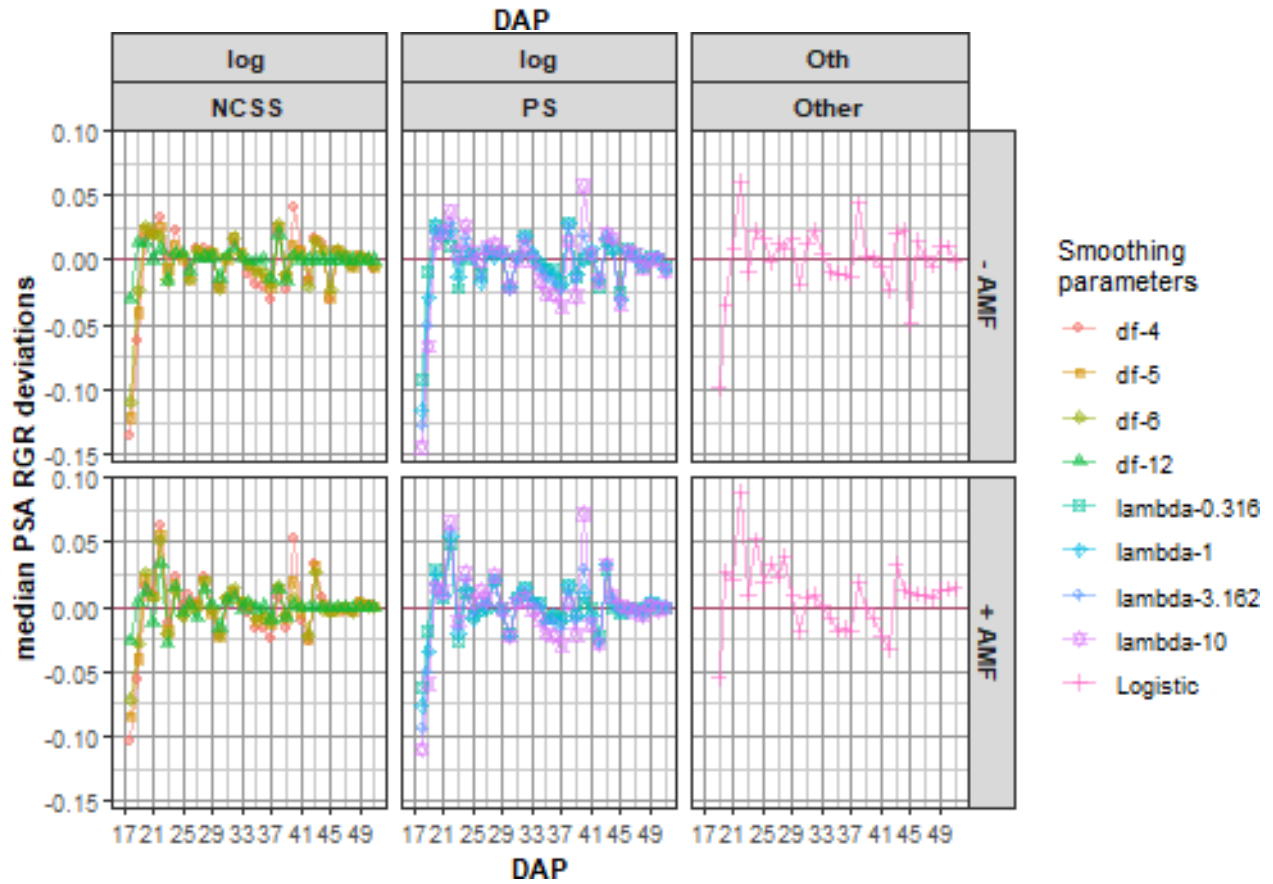
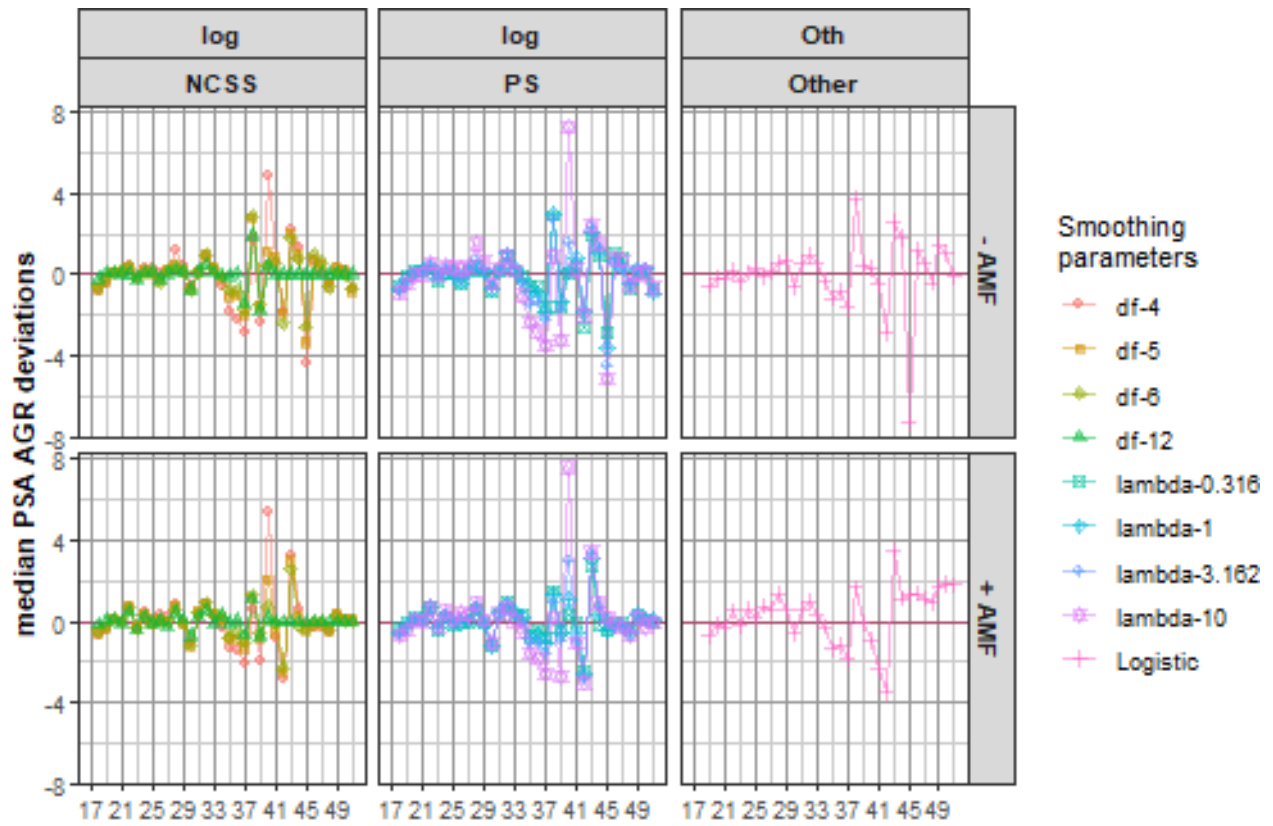
Plot for PS



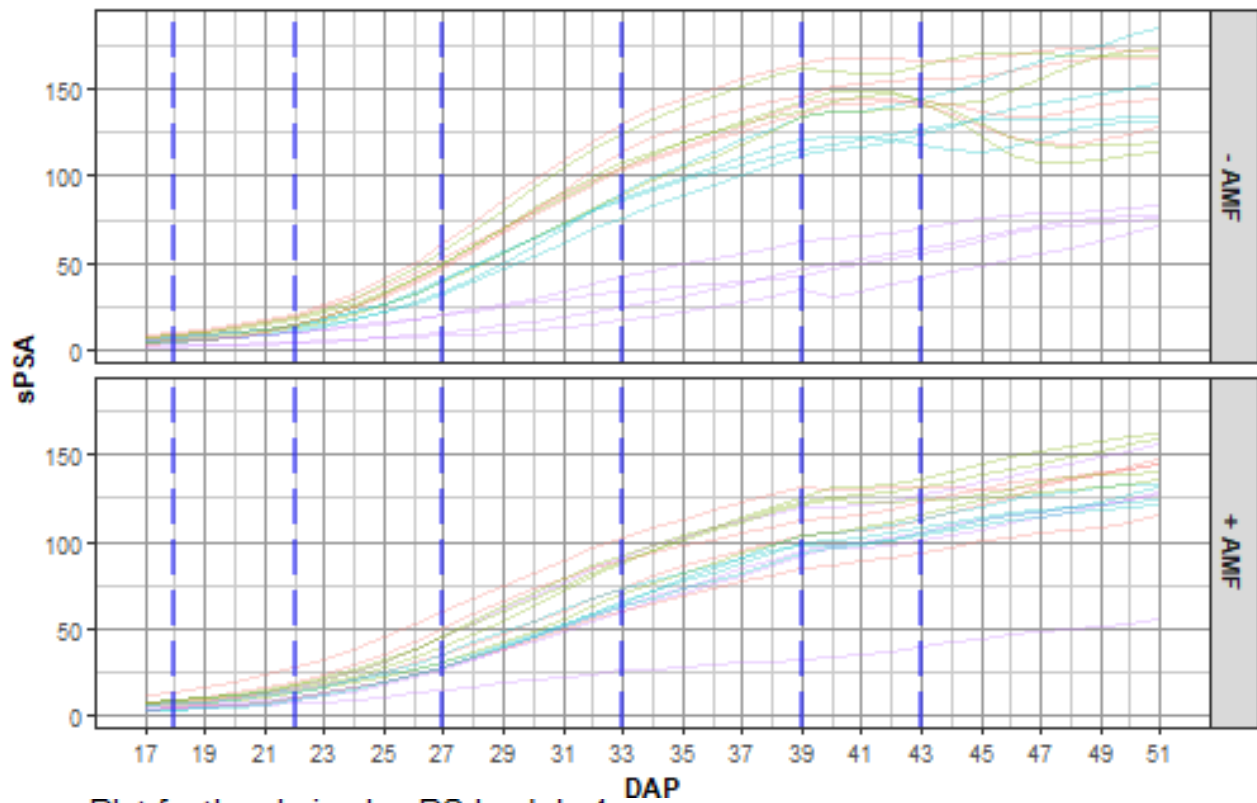
Plot for Other



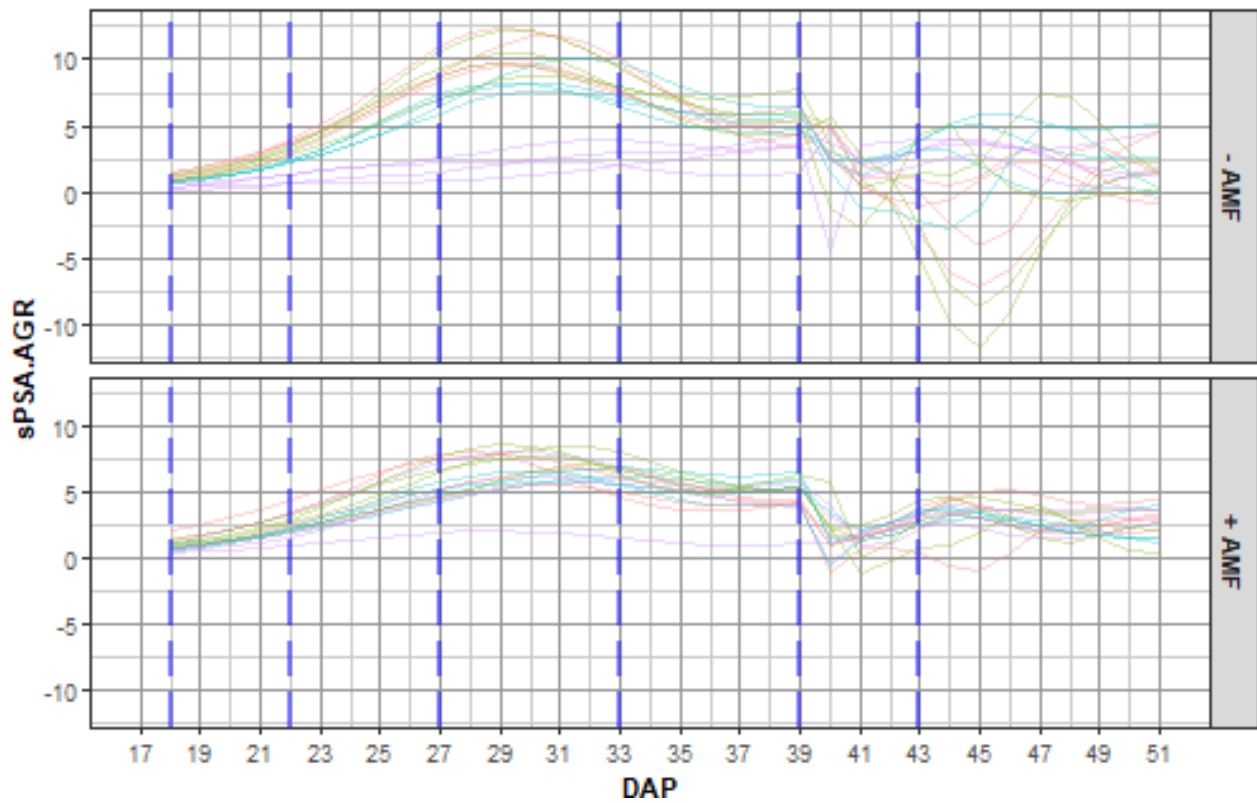


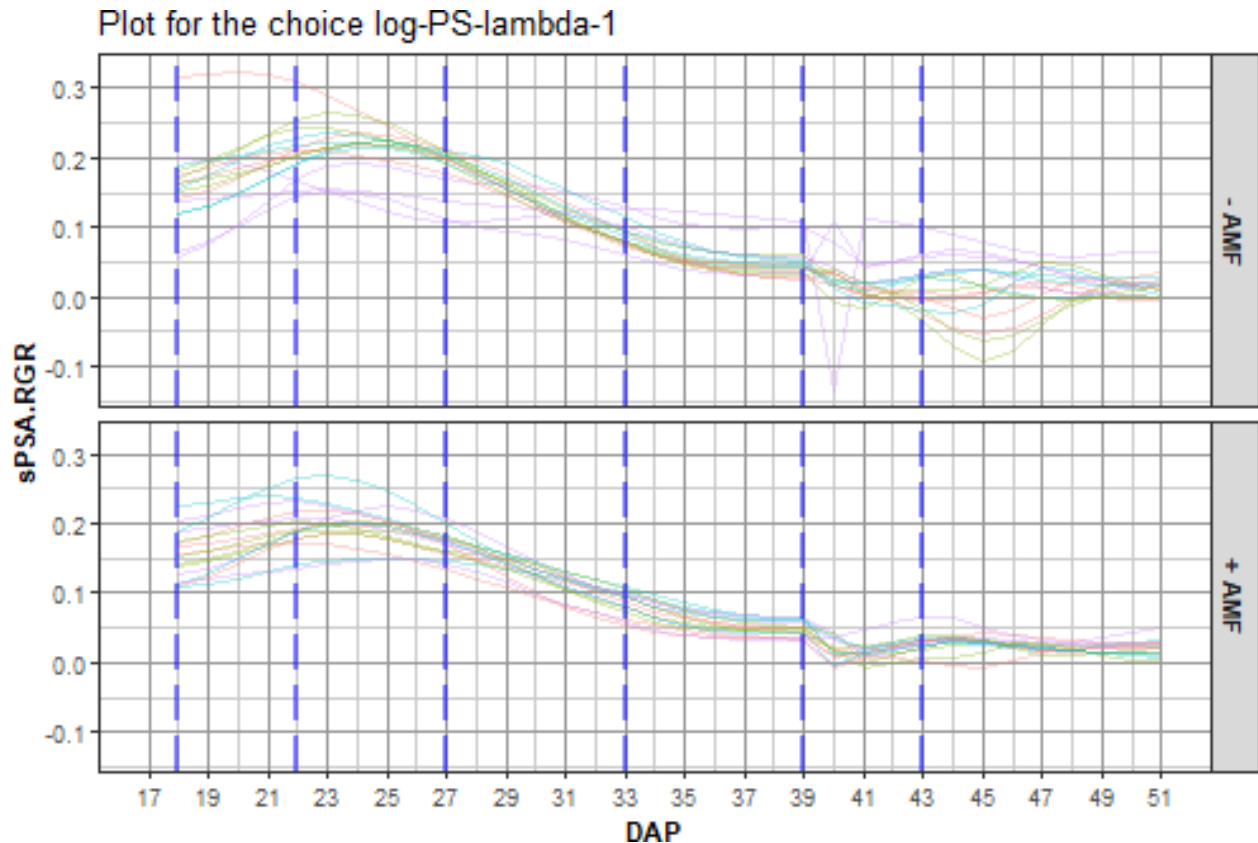


Plot for the choice log-PS-lambda-1



Plot for the choice log-PS-lambda-1





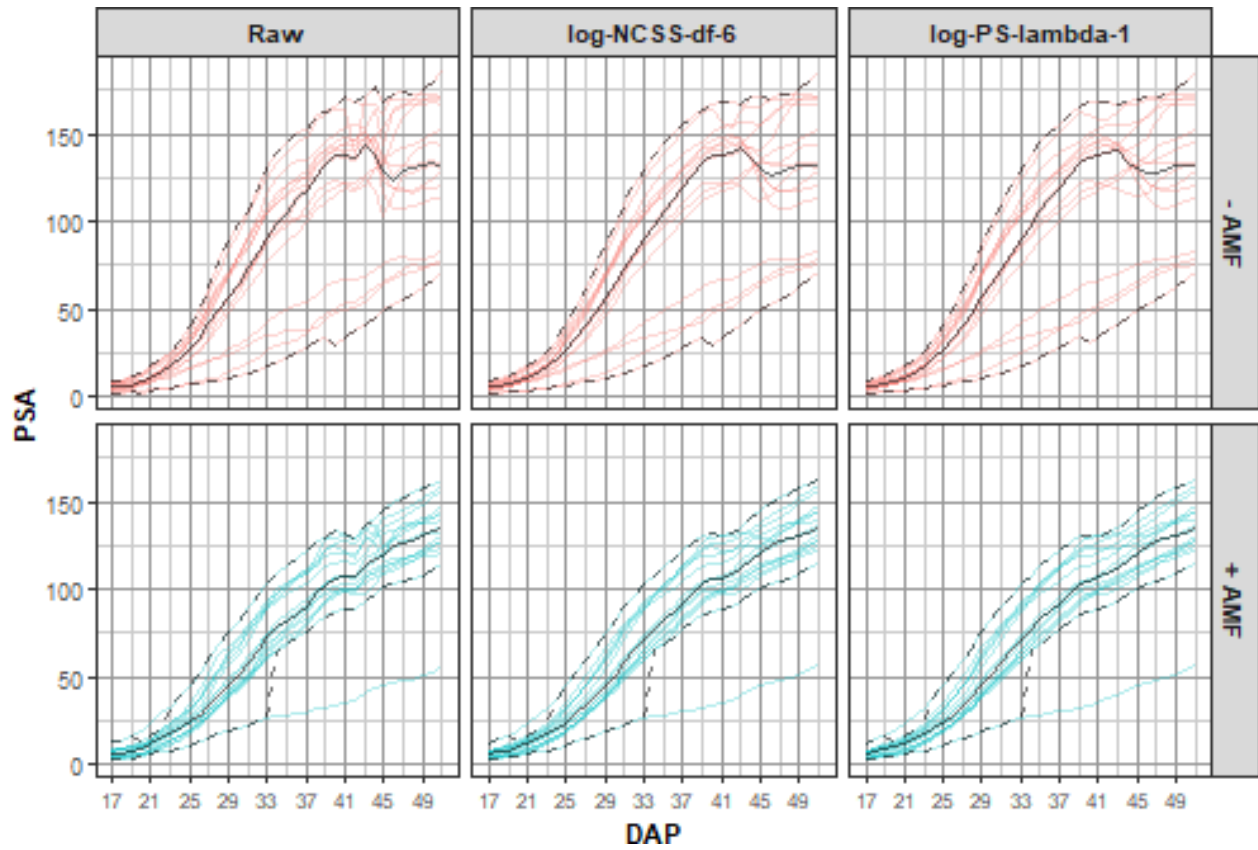
### Compare log smoothing of PSA for NCSS with $DF = 6$ and PS with $\lambda = 1$

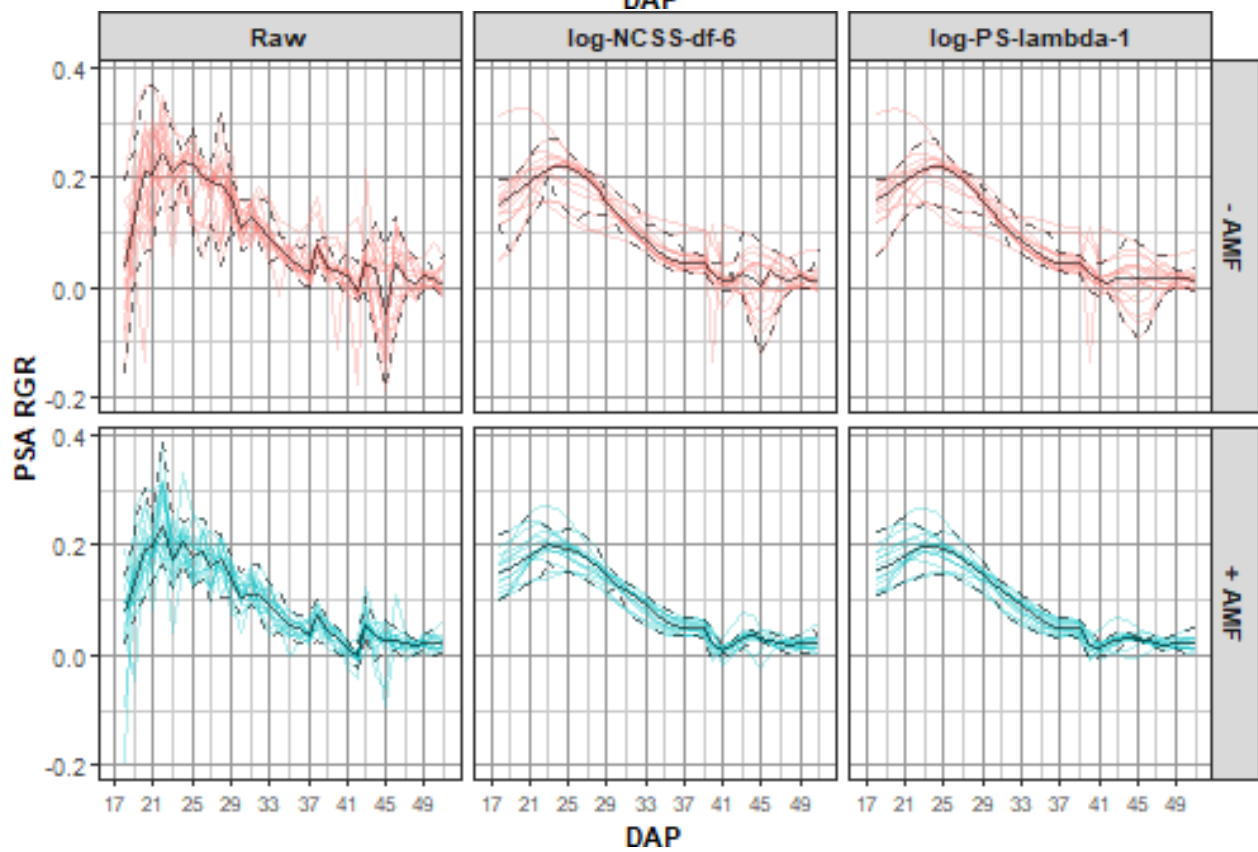
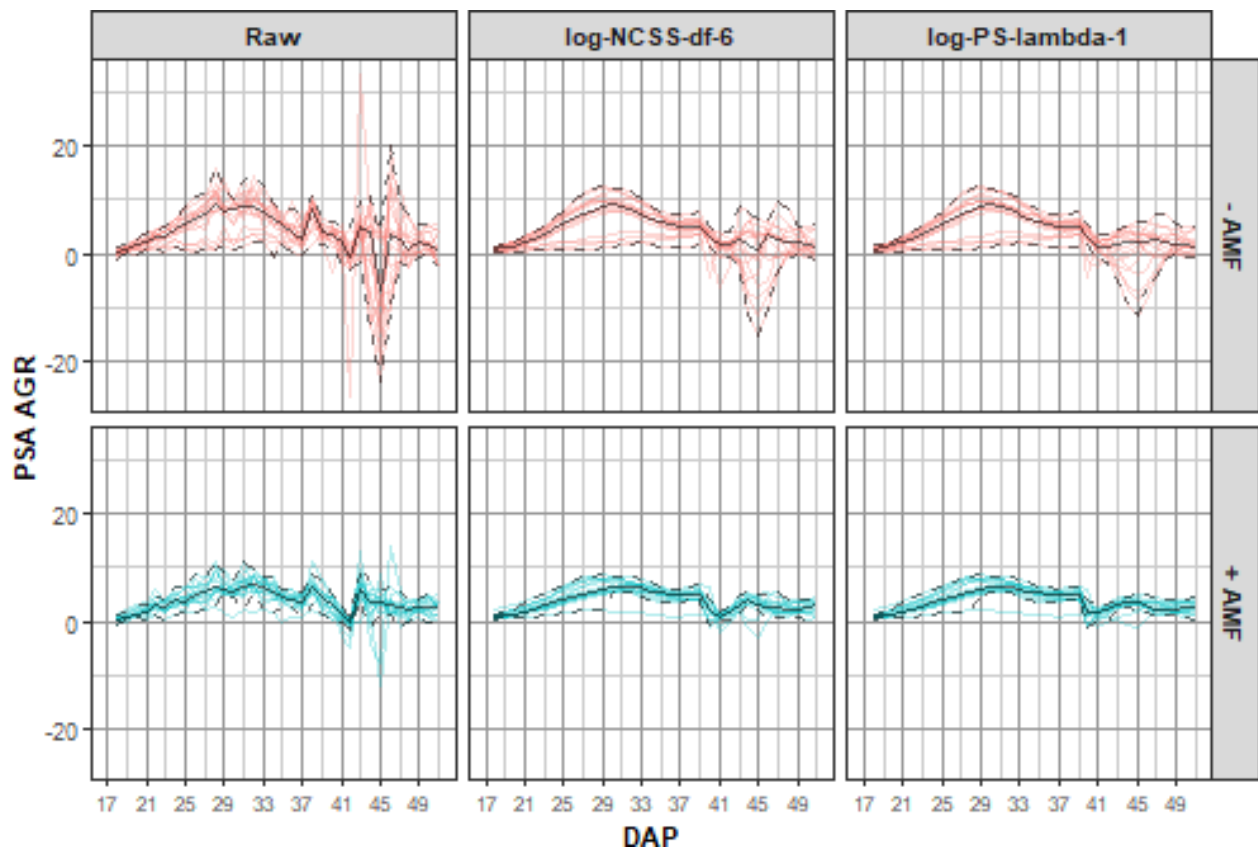
Now compare what appear to be the best smooths for natural cubic smoothing splines (NCSS-df-6) and P-splines (PS-lambda-1) using `traitSmooth`. This is done by supplying `smoothing.args` with a list of parallel vectors, each vector being of length two. The argument `chosen.smooth.args` is set to `NULL` so that one of the smooths is not chosen for output. Again, arguments are included to control the smoothing and the layout of the profile and median-deviations plots.

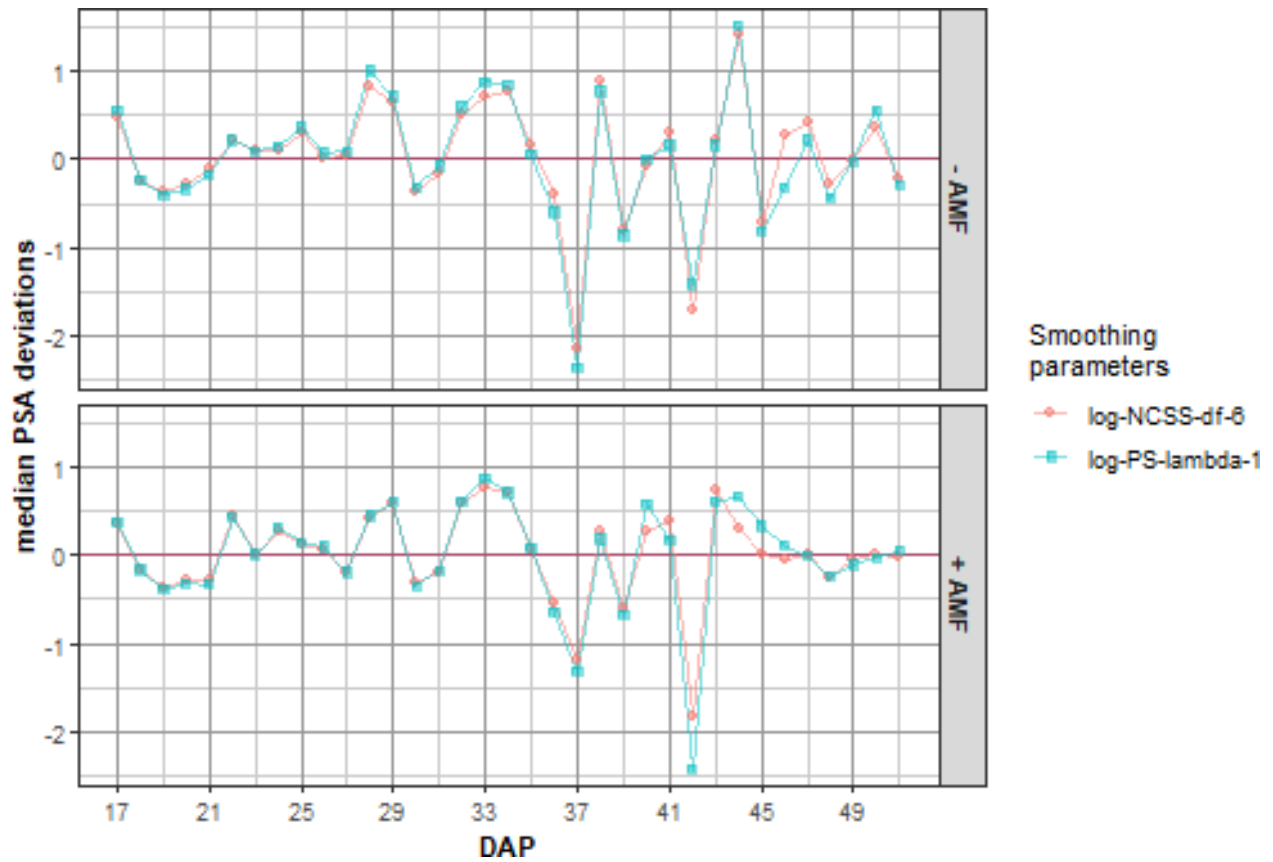
Smoothing based on P-splines is chosen because it tends to smooth somewhat more than that based on NCSS splines, especially after DAP 45. Consequently, there is no need to change the values of the `chosen.splines` argument from the default values.

```
smth.dat <- traitSmooth(data = longi.dat,
  response = "PSA", response.smoothed = "sPSA",
  individuals = "Snapshot.ID.Tag", times = "DAP",
  keep.columns = c("AMF", "Zn"),
  smoothing.args =
    args4smoothing(smoothing.methods = c("log", "log"),
      spline.types = c("N", "P"),
      df = c(6, NA), lambdas = c(NA, 1),
      combinations = "parallel",
      smoothing.segments = DAP.segs),
  chosen.smooth.args = NULL,
  profile.plot.args =
    args4profile_plot(plots.by = NULL,
      facet.x = tune.fac, facet.y = "AMF",
      facet.labeller = labeller(AMF = labelAMF),
      colour.column = "AMF"),
```

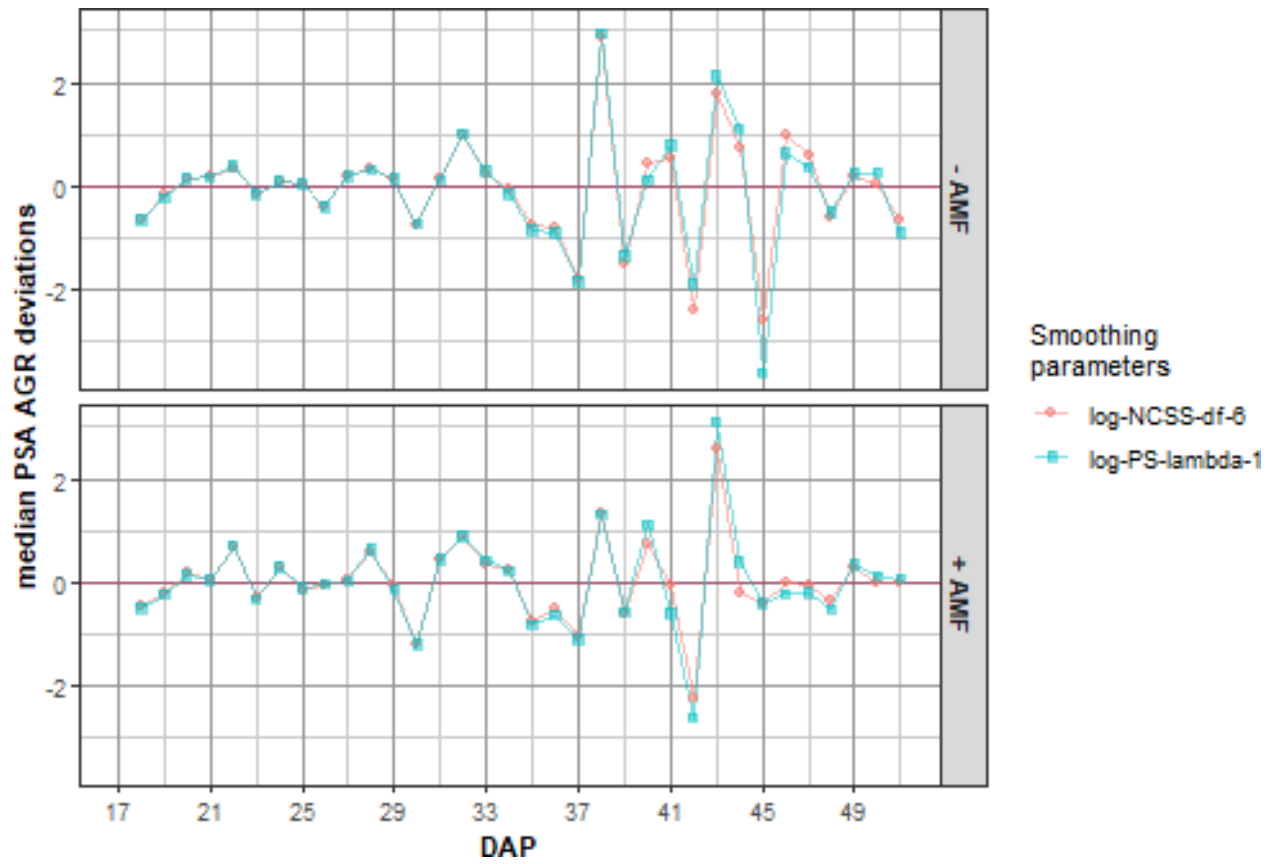
```
meddevn.plot.args =  
  args4meddevn_plot(plots.by = NULL, plots.group = tune.fac,  
                    facet.x = ".", facet.y = "AMF",  
                    facet.labeller = labeller(AMF = labelAMF)))
```



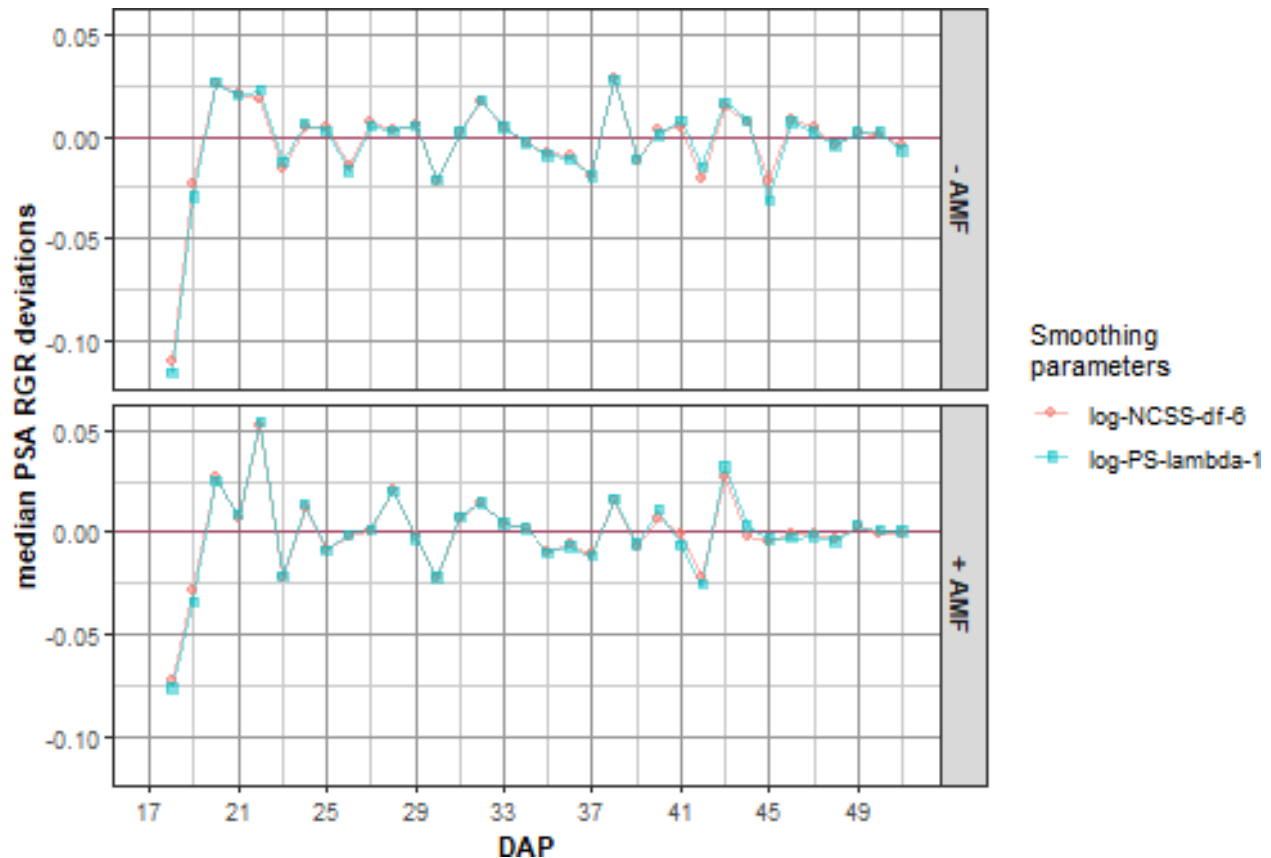




```
## Warning: Removed 2 rows containing missing values (`geom_line()`).
## Warning: Removed 4 rows containing missing values (`geom_point()`).
```



```
## Warning: Removed 2 rows containing missing values (`geom_line()`).
## Removed 4 rows containing missing values (`geom_point()`).
```



Extract the chosen smooth, adding it to longi.dat

```

longi.dat <- traitSmooth(data = smth.dat,
  response = "PSA", response.smoothed = "sPSA",
  individuals = "Snapshot.ID.Tag", times = "DAP",
  keep.columns = c("AMF","Zn"),
  smoothing.args = NULL, which.plots = "none",
  chosen.smooth.args =
    args4chosen_smooth(smoothing.methods = "log",
      spline.types = "PS", lambdas = 1),
  chosen.plot.args =
    args4chosen_plot(facet.y = "AMF",
      facet.labeller = labeller(AMF = labelAMF),
      colour.column = "Zn",
      ggplotFuncs = vline.DAP.endpts),
  mergedata = tomato.dat)

```

### Step III: Investigate the smoothing of the WU

Explore the smooths of WU for a range of smoothing parameters

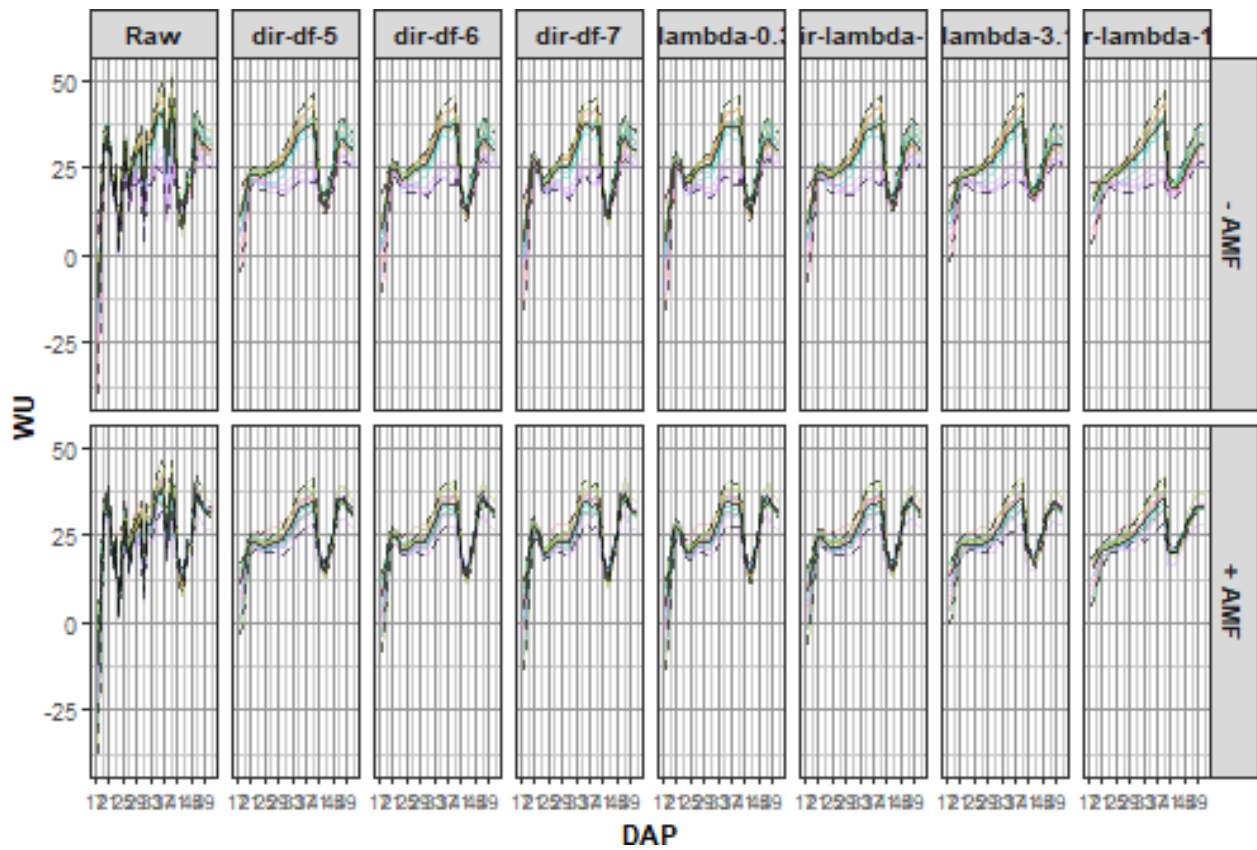
For WU, we take a slightly different approach to that taken with PSA. We first examine the fits for a range of smoothing parameters, setting the `traitSmooth` argument `chosen.smooth.args` to `NULL` so that a single smooth is not chosen for output. We then examine the two smooths that are the main contenders and finally do plots for the smooth chosen from these two. Again, a segmented smooth involving two segments has also

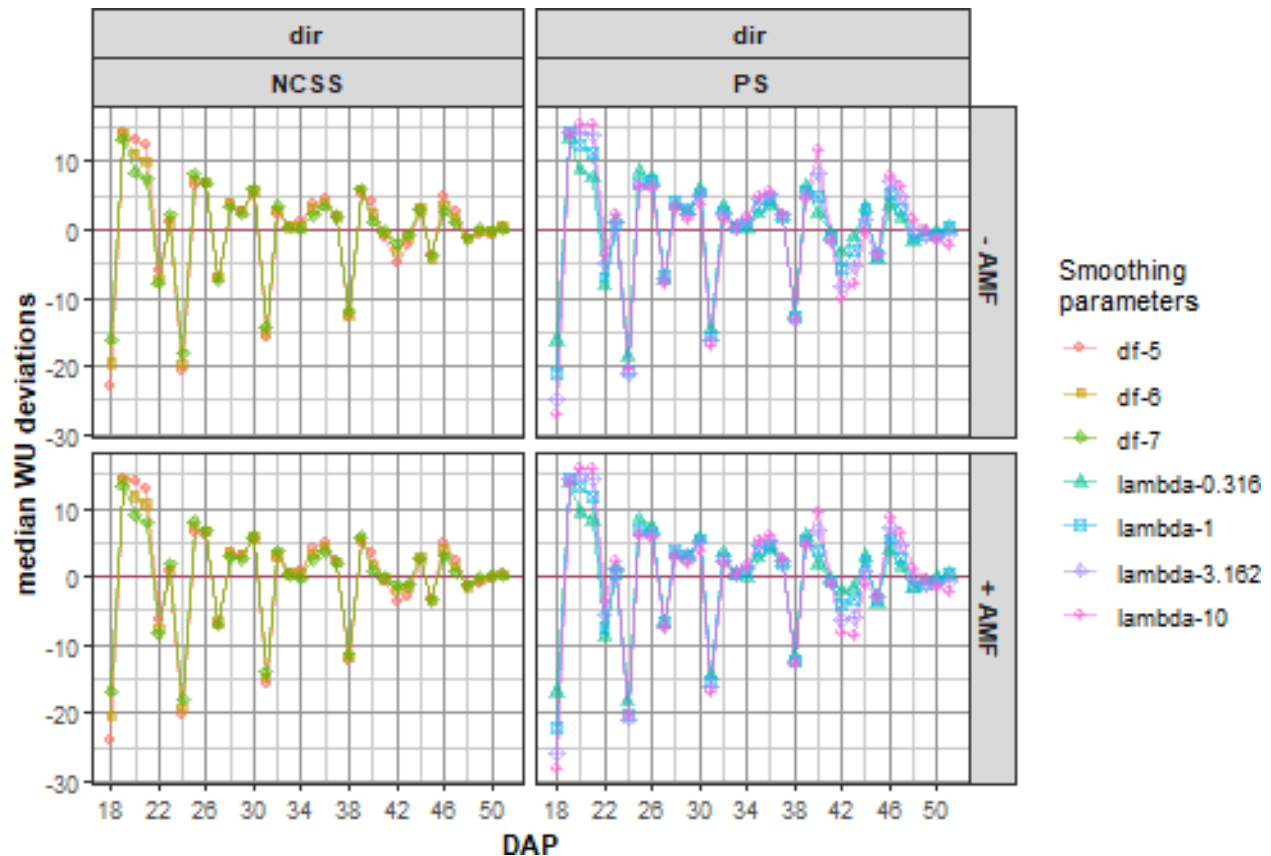


been specified with the breakpoint for the segments being DAP 39.

The function `traitSmooth` is used to produce the smooths. However, because no `chosen.smooth.args` is being specified, the function `probeSmooths` could be called directly instead. In this case, the `get.rates` and `trait.types` arguments from `probeSmooths` are set to `FALSE` and to `"response"` so that only the response is smoothed, without the calculation of growth rates from the smoothed response.

```
suppressWarnings(
  smth.dat <- traitSmooth(data = longi.dat,
    response = "WU", response.smoothed = "sWU",
    individuals = "Snapshot.ID.Tag", times = "DAP",
    keep.columns = c("AMF", "Zn"),
    trait.types = "response",
    smoothing.args =
      args4smoothing(smoothing.methods = "direct",
        smoothing.segments = DAP.segs),
    chosen.smooth.args = NULL,
    profile.plot.args =
      args4profile_plot(plots.by = NULL,
        facet.y = "AMF",
        colour.column = "Zn",
        facet.labeller = labeller(AMF = labelAMF)),
    meddevn.plot.args =
      args4meddevn_plot(plots.by = NULL,
        facet.y = "AMF",
        facet.labeller = labeller(AMF = labelAMF))))
```





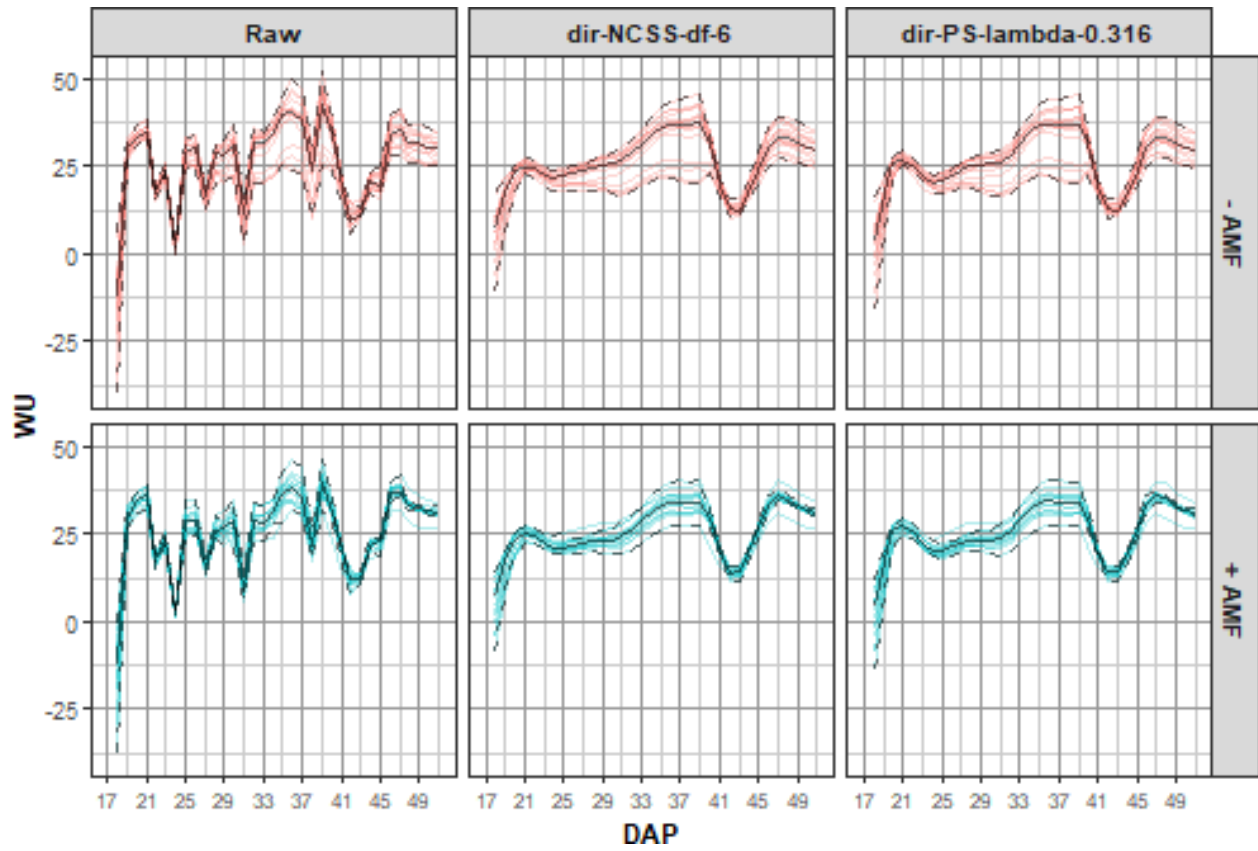
Produce plots comparing direct smoothing of WU for NCSS with  $DF = 6$  and PS with  $\lambda = 0.316$

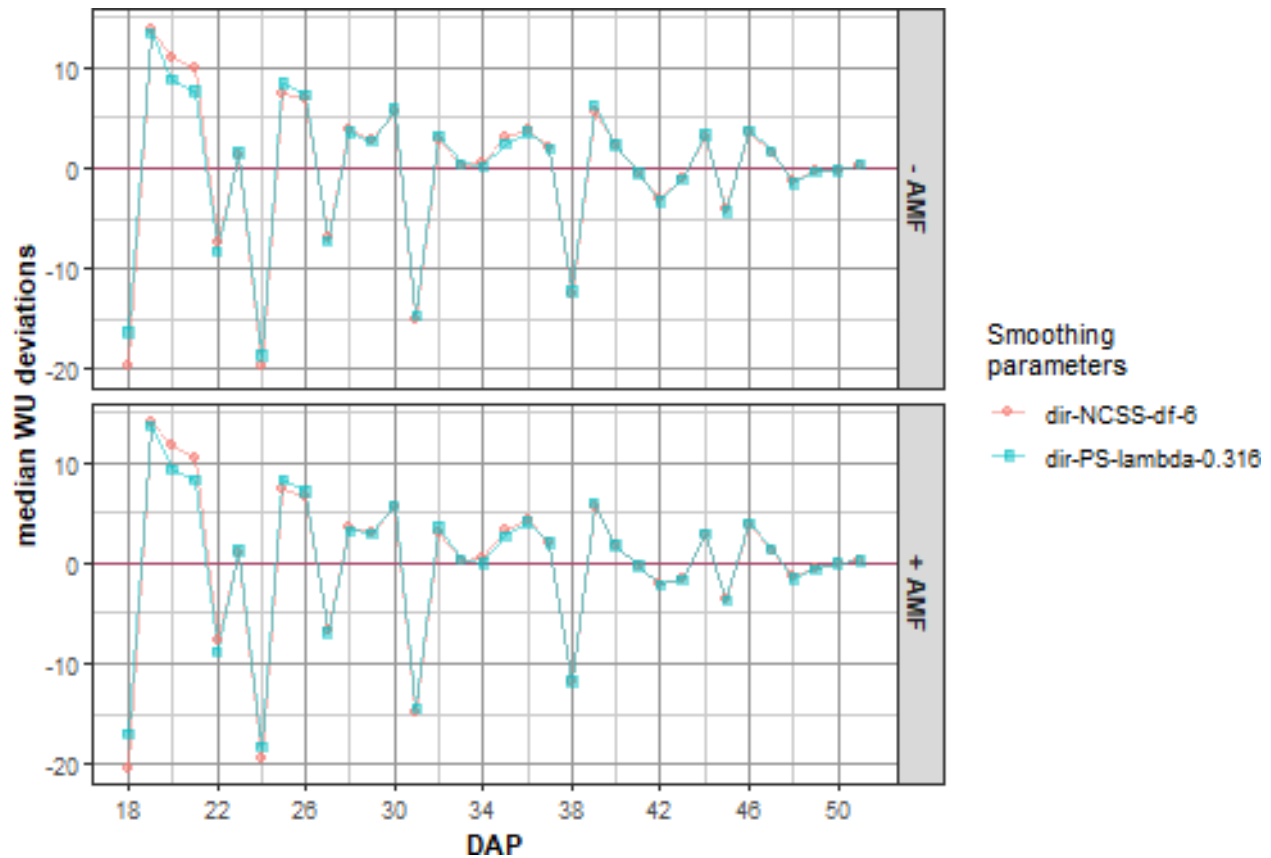
Now compare what appear to be the best smooths for natural cubic smoothing splines (NCSS-df-6) and for P-splines (PS-lambda-0.316). The function `traitSmooth` is used for the comparison, `probeSmooths` could be called directly instead. The PS splines with  $\lambda = 0.316$  are chosen because they tend to smooth a little less than the NCSS splines, especially before DAP 26.

```
suppressWarnings(
  traitSmooth(data = smth.dat,
    response = "WU", response.smoothed = "sWU",
    individuals = "Snapshot.ID.Tag", times = "DAP",
    trait.types = "response",
    smoothing.args = args4smoothing(smoothing.methods = c("dir", "dir"),
      spline.types = c("N", "P"),
      df = c(6, NA), lambdas = c(NA, 0.316),
      smoothing.segments = DAP.segs,
      combinations = "parallel"),

    chosen.smooth.args = NULL,
    profile.plot.args =
      args4profile_plot(plots.by = NULL,
        facet.x = tune.fac, facet.y = "AMF",
        colour.column = "AMF",
        facet.labeller = labeller(AMF = labelAMF)),
    meddevn.plot.args =
      args4meddevn_plot(plots.by = NULL, plots.group = tune.fac,
        facet.x = ".", facet.y = "AMF",
```

```
facet.labeller = labeller(AMF = labelAMF)))
```

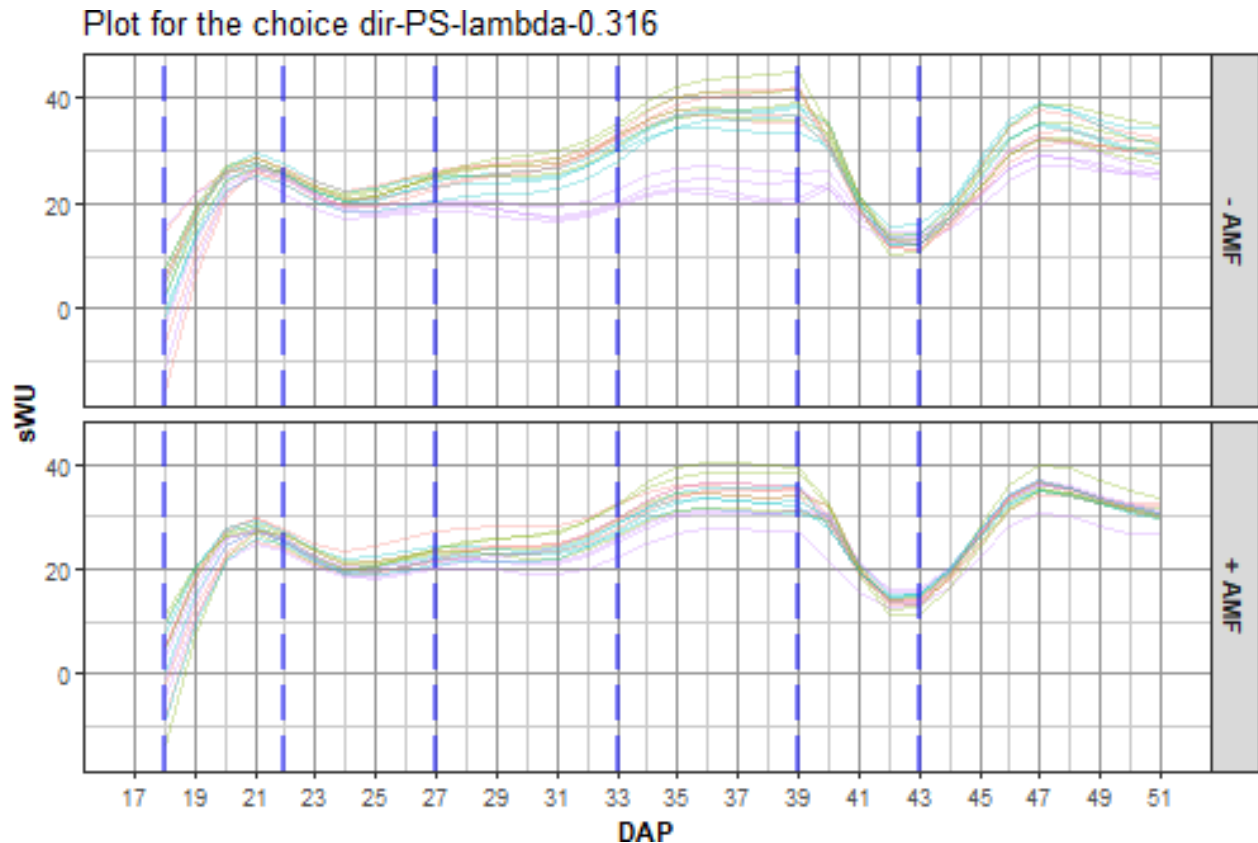




Produce the plots for the chosen smooth and add it to `longi.dat`

Here `traitSmooth` is used to fit the two smooths specified in `spar.schemes` in the previous step and the `chosen.splines` argument is set for the fit using PS splines with  $\lambda = 0.316$ .

```
longi.dat <- traitSmooth(data = smth.dat,
  response = "WU", response.smoothed = "sWU",
  individuals = "Snapshot.ID.Tag", times = "DAP",
  trait.types = "response",
  smoothing.args = NULL, which.plots = NULL,
  chosen.smooth.args =
    args4chosen_smooth(smoothing.method = "direct",
      spline.type = "PS",
      lambdas = 0.316), #tried 1 first
  chosen.plot.args =
    args4chosen_plot(facet.y = "AMF",
      facet.labeller = labeller(AMF = labelAMF),
      colour.column = "Zn",
      ggplotFuncs = vline.DAP.endpts),
  mergedata = longi.dat)
```



## Step IV: Identify potential outliers and remove if justified

A plant was identified as slow growing. Even though its pot had been inoculated with AMF, it had low AMF root colonization and a random mutated shoot phenotype, which could explain why its behaviour was consistent with a plant that was not inoculated with AMF. We omit the it from further analysis.

### Omit responses for the outlier plant

The outlier plant is omitted by setting all of its responses to NA, i.e. the metadata for the plant is retained in `longi.dat`.

```
#Identify the plant
omit <- with(longi.dat, Zn==90 & AMF=="+" & Block ==4)
#Identify the responses columns
NA.cols <- match("Weight.After", names(longi.dat)):length(longi.dat)
responses.all <- names(longi.dat)[NA.cols]
#Set the responses for the plant to NA
longi.dat[responses.all] <- lapply(longi.dat[responses.all],
  function(kcol, omit)
  {
    kcol[omit] <- NA
    return(kcol)
  }, omit = omit)
```

## Step V: Extract single-valued traits for each individual

In this step, traits that have a single-value for each plant (cart) are created from the smoothed PSA (sPSA) and the smoothed WU (sWU), along with the derived traits sPSA AGR, sPSA RGR, sWUR (smoothed Water Use Rate) and sPSA.sWUI (smoothed Water Use Index with sPSA as the numerator). The single-valued traits are based on a set of endpoints for DAP intervals. The DAP endpoints that were chosen, as described by Brien et al. (2020), are 18, 22, 27, 33, 39, 43 and 51. Corresponding to these endpoints are the time intervals DAP 18–22, DAP 22–27, DAP 27–33, DAP 33–39, DAP 39–43 and DAP 43–51. Based on these endpoints and intervals, the following single-valued traits are to be computed:

1. **single-times traits:** sPSA for each DAP
2. **growth rates for a time interval:** sPSA AGR and sPSA RGR for the six intervals.
3. **water use traits for a time interval:** sWU, sWUR and sPSA.sWUI for the six intervals.
4. **total for the overall imaging period:** sWU for DAP 18–51.
5. **maximum for the overall imaging period:** maximum of the sPSA AGR during DAP 18–51 and the DAP on which it occurred.

```

indv.cols <- c("Snapshot.ID.Tag", "Lane", "Position", "Block", "Cart", "AMF", "Zn")
indv.dat <- subset(longi.dat, subset = DAP == DAP.endpts[1],
                  select = indv.cols)
indv.dat <- traitExtractFeatures(data = longi.dat,
                               starts.intvl = DAP.starts, stops.intvl = DAP.stops,
                               responses4singletimes = "sPSA",
                               responses4intvl.rates = "sPSA", growth.rates = c("AGR", "RGR"),
                               water.use4intvl.traits = "sWU", responses4water = "sPSA",
                               responses4overall.total = "sWU",
                               responses4overall.max = "sPSA.AGR",
                               mergedata = indv.dat)

```

## Finalise

```

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

```

```

## Snapshot.ID.Tag   Lane      Position  Block      Cart      AMF      Zn
## Length:32        6:16   5       : 2    1:8   1       :4   -   :16   0   :8
## Class :character  7:16   6       : 2    2:8   2       :4   +   :15  10  :8
## Mode  :character          7       : 2    3:8   3       :4   NA's: 1  40  :8
##                               8       : 2    4:8   4       :4                               90  :7
##                               9       : 2                5       :4                               NA's:1
##                              10      : 2                6       :4
##                               (Other):20      (Other):8
##      sPSA.18      sPSA.22      sPSA.27      sPSA.33
## Min.   : 2.128   Min.   : 4.032   Min.   : 8.37   Min.   : 17.01
## 1st Qu.: 4.789   1st Qu.:10.501   1st Qu.:28.65   1st Qu.: 63.87
## Median : 6.742   Median :14.077   Median :39.35   Median : 86.92
## Mean   : 6.710   Mean   :13.978   Mean   :37.76   Mean   : 79.95
## 3rd Qu.: 8.398   3rd Qu.:16.807   3rd Qu.:47.84   3rd Qu.: 97.53
## Max.   :14.100   Max.   :27.612   Max.   :61.20   Max.   :129.59
## NA's   :1       NA's   :1       NA's   :1       NA's   :1
##      sPSA.39      sPSA.43      sPSA.51      sPSA.AGR.18to22
## Min.   : 34.33   Min.   : 41.16   Min.   : 71.27   Min.   :0.3905
## 1st Qu.: 96.46   1st Qu.:105.27   1st Qu.:122.76   1st Qu.:1.4727
## Median :115.53   Median :123.55   Median :133.45   Median :1.6730
## Mean   :110.98   Mean   :118.08   Mean   :134.50   Mean   :1.8170

```

##	3rd Qu.:133.76	3rd Qu.:140.45	3rd Qu.:154.31	3rd Qu.:2.3631
##	Max. :164.69	Max. :166.76	Max. :185.36	Max. :3.3781
##	NA's :1	NA's :1	NA's :1	NA's :1
##	sPSA.RGR.18to22	sPSA.AGR.22to27	sPSA.RGR.22to27	sPSA.AGR.27to33
##	Min. :0.1131	Min. :0.7833	Min. :0.1262	Min. : 1.441
##	1st Qu.:0.1613	1st Qu.:3.6237	1st Qu.:0.1824	1st Qu.: 5.793
##	Median :0.1827	Median :4.8037	Median :0.2005	Median : 7.266
##	Mean :0.1854	Mean :4.7572	Mean :0.1961	Mean : 7.032
##	3rd Qu.:0.2026	3rd Qu.:6.2821	3rd Qu.:0.2165	3rd Qu.: 8.582
##	Max. :0.3192	Max. :8.0144	Max. :0.2461	Max. :11.397
##	NA's :1	NA's :1	NA's :1	NA's :1
##	sPSA.RGR.27to33	sPSA.AGR.33to39	sPSA.RGR.33to39	sPSA.AGR.39to43
##	Min. :0.08414	Min. :1.434	Min. :0.03775	Min. : -0.7949
##	1st Qu.:0.11848	1st Qu.:4.700	1st Qu.:0.04582	1st Qu.: 1.4347
##	Median :0.12585	Median :5.391	Median :0.05582	Median : 1.9842
##	Mean :0.12554	Mean :5.171	Mean :0.05843	Mean : 1.7757
##	3rd Qu.:0.13267	3rd Qu.:5.862	3rd Qu.:0.06661	3rd Qu.: 2.4714
##	Max. :0.16237	Max. :7.349	Max. :0.11699	Max. : 3.1744
##	NA's :1	NA's :1	NA's :1	NA's :1
##	sPSA.RGR.39to43	sPSA.AGR.43to51	sPSA.RGR.43to51	sWU.18to22
##	Min. : -0.00663	Min. : -3.694	Min. : -0.02885	Min. : 79.80
##	1st Qu.: 0.01199	1st Qu.: 1.539	1st Qu.: 0.01038	1st Qu.: 85.77
##	Median : 0.01797	Median : 2.510	Median : 0.02115	Median : 96.43
##	Mean : 0.01900	Mean : 2.052	Mean : 0.01831	Mean : 93.61
##	3rd Qu.: 0.02424	3rd Qu.: 3.384	3rd Qu.: 0.02619	3rd Qu.:100.05
##	Max. : 0.06542	Max. : 5.224	Max. : 0.06864	Max. :104.25
##	NA's :1	NA's :1	NA's :1	NA's :1
##	sWUR.18to22	sPSA.sWUI.18to22	sWU.22to27	sWUR.22to27
##	Min. :19.95	Min. :0.01654	Min. : 90.13	Min. :18.03
##	1st Qu.:21.44	1st Qu.:0.06260	1st Qu.:102.34	1st Qu.:20.47
##	Median :24.11	Median :0.07068	Median :109.55	Median :21.91
##	Mean :23.40	Mean :0.07817	Mean :107.81	Mean :21.56
##	3rd Qu.:25.01	3rd Qu.:0.10147	3rd Qu.:112.68	3rd Qu.:22.54
##	Max. :26.06	Max. :0.13012	Max. :125.61	Max. :25.12
##	NA's :1	NA's :1	NA's :1	NA's :1
##	sPSA.sWUI.22to27	sWU.27to33	sWUR.27to33	sPSA.sWUI.27to33
##	Min. :0.03858	Min. :106.0	Min. :17.67	Min. :0.07756
##	1st Qu.:0.16720	1st Qu.:140.8	1st Qu.:23.46	1st Qu.:0.24544
##	Median :0.22553	Median :152.7	Median :25.45	Median :0.27223
##	Mean :0.21811	Mean :150.9	Mean :25.15	Mean :0.27200
##	3rd Qu.:0.27152	3rd Qu.:165.4	3rd Qu.:27.56	3rd Qu.:0.31508
##	Max. :0.35963	Max. :182.4	Max. :30.41	Max. :0.40126
##	NA's :1	NA's :1	NA's :1	NA's :1
##	sWU.33to39	sWUR.33to39	sPSA.sWUI.33to39	sWU.39to43
##	Min. :126.7	Min. :21.12	Min. :0.05969	Min. :65.15
##	1st Qu.:190.5	1st Qu.:31.75	1st Qu.:0.13273	1st Qu.:74.32
##	Median :211.3	Median :35.21	Median :0.15037	Median :77.46
##	Mean :204.2	Mean :34.04	Mean :0.15159	Mean :77.00
##	3rd Qu.:223.1	3rd Qu.:37.19	3rd Qu.:0.17207	3rd Qu.:80.52
##	Max. :259.4	Max. :43.24	Max. :0.20415	Max. :83.88
##	NA's :1	NA's :1	NA's :1	NA's :1
##	sWUR.39to43	sPSA.sWUI.39to43	sWU.43to51	sWUR.43to51
##	Min. :16.29	Min. : -0.04207	Min. :190.6	Min. :23.83
##	1st Qu.:18.58	1st Qu.: 0.07150	1st Qu.:230.5	1st Qu.:28.81

```

## Median :19.37 Median : 0.10263 Median :242.5 Median :30.32
## Mean :19.25 Mean : 0.09285 Mean :238.7 Mean :29.84
## 3rd Qu.:20.13 3rd Qu.: 0.13108 3rd Qu.:249.8 3rd Qu.:31.23
## Max. :20.97 Max. : 0.19489 Max. :268.5 Max. :33.56
## NA's :1 NA's :1 NA's :1 NA's :1
## sPSA.sWUI.43to51 sWU sPSA.AGR.max sPSA.AGR.max.DAP
## Min. :-0.13026 Min. :701.0 Min. : 3.963 Min. :12.00
## 1st Qu.: 0.04992 1st Qu.:858.5 1st Qu.: 6.150 1st Qu.:13.00
## Median : 0.08270 Median :884.0 Median : 7.744 Median :14.00
## Mean : 0.06762 Mean :874.0 Mean : 7.791 Mean :15.77
## 3rd Qu.: 0.10781 3rd Qu.:922.0 3rd Qu.: 9.148 3rd Qu.:16.00
## Max. : 0.15907 Max. :988.0 Max. :12.423 Max. :35.00
## NA's :1 NA's :1 NA's :1 NA's :1

```

```
head(indv.dat)
```

```

## Snapshot.ID.Tag Lane Position Block Cart AMF Zn sPSA.18 sPSA.22 sPSA.27
## 1 061472 6 5 1 1 - 0 9.856841 21.132127 61.20433
## 2 061473 6 6 1 2 + 10 8.219937 15.732854 39.75138
## 3 061474 6 7 1 3 - 90 2.469923 4.032111 10.07049
## 4 061475 6 8 1 4 + 40 8.971075 14.864706 31.21562
## 5 061476 6 9 1 5 + 90 4.823554 9.198190 27.09603
## 6 061477 6 10 1 6 - 40 4.998369 11.434154 33.88250
## sPSA.33 sPSA.39 sPSA.43 sPSA.51 sPSA.AGR.18to22 sPSA.RGR.18to22
## 1 129.58879 164.69352 166.75700 171.47291 2.8188215 0.1906572
## 2 87.87222 123.11477 131.05159 159.65092 1.8782293 0.1622972
## 3 24.91082 46.28202 58.39061 77.96569 0.3905471 0.1225258
## 4 65.05030 99.72473 107.67442 131.06986 1.4734077 0.1262460
## 5 62.69652 94.52888 105.67301 127.43397 1.0936589 0.1613739
## 6 89.76055 133.80166 143.57346 185.36485 1.6089464 0.2068733
## sPSA.AGR.22to27 sPSA.RGR.22to27 sPSA.AGR.27to33 sPSA.RGR.27to33
## 1 8.014441 0.2126847 11.397410 0.1250247
## 2 4.803705 0.1853787 8.020140 0.1322065
## 3 1.207676 0.1830638 2.473389 0.1509488
## 4 3.270184 0.1483858 5.639112 0.1223737
## 5 3.579568 0.2160761 5.933415 0.1398198
## 6 4.489670 0.2172588 9.313008 0.1623745
## sPSA.AGR.33to39 sPSA.RGR.33to39 sPSA.AGR.39to43 sPSA.RGR.39to43
## 1 5.850789 0.03995334 0.5158698 0.003112841
## 2 5.873758 0.05620555 1.9842058 0.015618520
## 3 3.561867 0.10324189 3.0271466 0.058100365
## 4 5.779072 0.07120882 1.9874220 0.019174584
## 5 5.305394 0.06843325 2.7860332 0.027861036
## 6 7.340184 0.06653549 2.4429507 0.017622072
## sPSA.AGR.43to51 sPSA.RGR.43to51 sWU.18to22 sWUR.18to22 sPSA.sWUI.18to22
## 1 0.5894883 0.003485951 97.91084 24.47771 0.11515871
## 2 3.5749165 0.024674829 97.85921 24.46480 0.07677272
## 3 2.4468849 0.036139220 94.46701 23.61675 0.01653687
## 4 2.9244298 0.024577301 101.82429 25.45607 0.05788041
## 5 2.7201203 0.023406106 96.41753 24.10438 0.04537179
## 6 5.2239236 0.031934903 98.41988 24.60497 0.06539112
## sWU.22to27 sWUR.22to27 sPSA.sWUI.22to27 sWU.27to33 sWUR.27to33
## 1 111.4264 22.28527 0.35962943 174.3139 29.05232
## 2 105.6890 21.13780 0.22725657 151.6969 25.28282
## 3 90.1329 18.02658 0.06699416 106.0449 17.67415

```



```

## 4  107.0495    21.40991    0.15274160   142.7822    23.79703
## 5  103.1972    20.63943    0.17343342   134.7183    22.45304
## 6  109.6825    21.93651    0.20466657   154.0212    25.67021
##   sPSA.sWUI.27to33 sWU.33to39 sWUR.33to39 sPSA.sWUI.33to39 sWU.39to43
## 1      0.3923063    222.8187    37.13645      0.1575484    80.88604
## 2      0.3172169    203.3876    33.89793      0.1732778    79.70746
## 3      0.1399438    126.7266    21.12110      0.1686403    69.79265
## 4      0.2369671    185.1663    30.86106      0.1872610    77.46181
## 5      0.2642588    183.3993    30.56655      0.1735686    82.71278
## 6      0.3627944    220.4028    36.73380      0.1998210    80.27464
##   sWUR.39to43 sPSA.sWUI.39to43 sWU.43to51 sWUR.43to51 sPSA.sWUI.43to51 sWU
## 1    20.22151    0.02551094    234.1140    29.26424    0.02014364 936
## 2    19.92687    0.09957441    240.2925    30.03657    0.11901881 890
## 3    17.44816    0.17349372    203.2074    25.40092    0.09633057 706
## 4    19.36545    0.10262720    242.5382    30.31727    0.09646084 866
## 5    20.67819    0.13473290    249.2872    31.16090    0.08729273 855
## 6    20.06866    0.12172963    262.7254    32.84067    0.15906873 933
##   sPSA.AGR.max sPSA.AGR.max.DAP
## 1    12.422797      13
## 2     8.415909      15
## 3     4.444479      23
## 4     6.198353      17
## 5     6.100730      14
## 6    10.090972      16

```

## Step VI: Save to files

### Save data files as csv, Excel and rda files

```

save(longi.dat, file="longi.dat.rda")
write.csv(longi.dat, "longi.dat.csv", row.names = F)
WriteXLS("longi.dat", ExcelFileName = "longi.dat.xlsx", SheetNames = "longi.dat",
         row.names = FALSE, BoldHeaderRow = TRUE, AdjWidth = TRUE, FreezeRow = 1)
save(indv.dat, file="indv.dat.rda")
write.csv(indv.dat, "indv.dat.csv", row.names = F)
WriteXLS("indv.dat", ExcelFileName = "indv.dat.xlsx", SheetNames = "indv.dat",
         row.names = FALSE, BoldHeaderRow = TRUE, AdjWidth = TRUE, FreezeRow = 1)

```

### Save the workspace image

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

## Reference

Brien, C. J. (2023) *growthPheno: Functional Analysis of Phenotypic Growth Data to Smooth and Extract Traits*. R package Version 2.1-18. <http://cran.at.r-project.org/package=growthPheno>.

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>.

Pinheiro J., Bates D., and R Core Team (2023). *nlme: Linear and Nonlinear Mixed Effects Models*. R package version 3.1-162, <https://CRAN.R-project.org/package=nlme>.