

Connecting Instrument Response to Atmospheric Concentrations for Unknown Organics: Steps Forward in Investigating Secondary Organic Aerosol Formation Processes

Background: Aerosols, or suspended particulate matter, are a crucial area of research in the field of atmospheric chemistry, because they play a significant role in public health and radiative forcing of the climate. Aerosols may be either primary, meaning they are directly emitted from sources, or secondary, meaning that they are formed in the atmosphere from the oxidation of gaseous precursors. In densely vegetated regions such as the tropics, these gasses predominantly originate from trees and other plants- these emissions are termed biogenic volatile organic compounds, or BVOC's. The quantities and characteristics of the aerosols formed from BVOC's are heavily dependent upon the structure-specific identities of the precursor gasses (Yee, 2018). The processes by which these gasses oxidize to form aerosols, and in particular the degree to which human emissions and influences on oxidative conditions, are as yet incompletely understood and must be explored in order to understand current climate phenomena and accurately predict future conditions.

This project seeks to automate and improve upon established methods for quantifying the concentrations of unknown compounds detected in 2D- Gas Chromatography, with the ultimate goal of improving current understandings of how human pollution influences the oxidation of biogenic hydrocarbons.

Methods

Data Source: In 2014, researchers from all over the globe convened at a site in central Amazonia for the Observations and Modeling of the Green Ocean Amazon field campaign (GoAmazon). This campaign was undertaken with the mission of improving scientific understanding of the hydrologic cycle of the Amazon Basin, informing climate models with more accurate representations of atmospheric chemistry in the tropics, predicting future climate scenarios under increasing human influence, and unraveling the mechanisms governing the formation of secondary organic aerosols more generally. The campaign was split into two intensive operating periods, one in the wet season of January through March and one during the dry season of August through October. The selected site is ~70 km from the city of Manaus in an otherwise pristine area of vegetation. Because of its location relative to prevailing wind directions and human activity in the area, the research site in question (T3, as indicated in Figure 1) experienced predominantly clean, biogenically dominated chemical regimes with occasional perturbations from the Manaus plume or local biomass burning from slash and burn. Researchers from the Goldstein Group at UC Berkeley participated in this campaign, and as part of this participation continuously collected aerosol filter samples throughout each of the two intensive operating periods. Samples were collected 4 hour time resolution in the dry season and 12 hour time resolution in the wet season. For a more complete description of the GoAmazon field campaign and filter collection methods, see Yee 2018.

Instrumentation: GoAmazon filter samples were frozen within 24 hours of sampling and shipped to UC Berkeley. In late 2018, these samples were processed using GCxGC-EI-HR-ToF-MS.

This instrument takes in segments of aerosol-coated filter, thermally desorbs the organic compounds in those aerosols, and passes the compounds through a series of two GC columns. The first of these columns separates compounds by volatility, and the second by polarity. The separated compounds are then sent to an electron ionization high resolution time of flight mass spectrometer. For a diagram of TD-GCxGC-EI-HR-ToF-MS, see figure 2. The resulting data is recorded as a 2D image in which each separated compound is represented as a 'blob' in volatility/polarity space, with a mass spectral fingerprint unique to that compound's formula and structure and a volume roughly representative of concentration on the filter (see figure 3).

Preliminary Processing: Based on previous analysis of bulk aerosol characteristics by a collaborator who performed Primary Matrix Factorization (PMF) on results from an Aerosol Mass Spectrometer (AMS), 12 samples were selected as representative of the range of conditions observed throughout the two intensive operating periods (de Sa

2018/de Sa 2019). See Figure 5 bottom half for a representation of the AMS work published in de Sa 2019 and upon which this work is based. Examples of these conditions include urban influenced day, urban influenced night, biomass burning influenced day, aged biogenic night, etc. From these 12 samples, 1,799 unique compounds were identified and compiled into a library. This library will be applied to all of the samples from both intensive operating periods to create timelines of the instrument response volumes of all 1,799 compounds over both operating periods.

Internal Standard: While the volume of a given blob does roughly represent that compound's concentration on the processed filter, there are a number of factors which prevent directly interpreting concentration from blob volume. The first of these is matrix effects- when a sample (aka filter) is more heavily loaded with organic compounds, the evaporation of all compounds is enhanced, causing an artificial increase in the abundance of compounds on heavily loaded samples. Additionally, instrument sensitivity changes over time and with maintenance, and this sensitivity is not always consistent for different classes of compounds. To correct for this, every sample is impregnated with an internal standard before running. The internal standard in a solution of 27 known compounds spanning the entire range of volatilities and polarities encountered in samples. As the same quantity of internal standard is used in every sample run, normalization by the internal standard removes sample to sample inconsistencies caused by matrix effects and instrument changes.

External Standard: Even beyond differences in instrument response depending on instrument condition, sample loading, and compound location in GCxGC space, different compounds respond differently in the GC and mass spec. This means that the concentration of a given compound cannot be directly inferred from the detected volume (or internal standard normalized volume) as reported in GC image. The accepted best practice for precisely quantifying identified compounds in GCxGC is to run a calibration curve for that compound using a dilution series of a pure, synthesized sample of the known compound, known as an External Standard. In a calibration curve, the compound in question is run on the instrument at a series of concentrations covering the range of expected concentrations on the samples in question. Ideally, the ratio between concentration of external standard and internal standard normalized volume of the detected blob in the instrument forms a direct linear relationship with a 0 intercept. The slope of this line, or the calibration factor, can then be used to translate the observed volume of a compound to a concentration. A major challenge of the GoAmazon aerosol analysis is that the majority of the ~2,000 unique compounds from the campaign have no authentic standards available, and most have never been synthesized meaning that they cannot be definitively identified by matches to published spectra.

The method utilized in recent publications involving GCxGC analysis of unknown compounds depends upon a researcher manually identifying the external compound which appears most similar to the unknown compound and assigning the calibration factor of the external standard compound to the unknown compound (Zhang, 2015). This method has a variety drawbacks, the most significant of which is that the manual assignment is highly subjective. Additionally, while this is a manageable task for 10's of unknown compounds and still possible for hundreds, when the number of unknowns reaches into the thousands it becomes highly inefficient from a time perspective. Additionally, there is debate within the field surrounding whether the closest (in volatility/polarity space) or most chemically similar (according to mass spectrum) compound is most appropriate.

For the GoAmazon campaign samples, 135 compounds representing a wide variety of the compound classes expected were utilized as an external standard. The calibration curves were six points each (five above zero), and five calibration curves were run throughout the intensive sample analysis period. A full list of the compounds contained in the external standard can be found in Appendix B.

Mass Spectrum Representation: Given that the intent of the models generated in this project is to accurately predict the calibration factors of unknown compounds, the external standard compound information is formatted in such a way as to reflect the information that will be generated for each unknown compound in each file throughout the time series. This information is limited to location (first dimension retention index indicating x axis position, second dimension time indicating y axis position), instrument response volume, and mass spectrum. The mass spectrum is a key indicator of structural characteristics that control instrument response, but its format does not

lend itself well to direct representation as a single feature. As the samples are run with 70 ev ionization, the organic compounds undergo significant fragmentation. Specific fragments, which are apparent as either peaks in the mass spectrum or mass differences between major peaks, are common to compounds with structural similarities. In order to efficiently capture this information, the m/z of the 10 highest peaks in the mass spectra of each external compound are extracted. These lists are combined, and the 20 most frequently occurring m/z peaks are turned into Boolean features; if a given compound has a significant peak at the commonly occurring m/z that feature is true for that compound, otherwise false. Mass differences are treated similarly; the m/z unit differences between the m/z's of the 5 highest peaks are determined for each compound, and the 20 most common mass differences are made into Boolean factors. This leaves 44 features informing models; 40 boolean features describing the mass spectra, two features describing the location in GCxGC space, the instrument volume, and the run date.

Training and Test Set Formation: the training and test sets are formed by randomly splitting the 135 compounds into a training and test set with a .7 split (.7 in training). The calibration point data corresponding to the compounds in each of the two categories are then separated to create the full training and test sets upon which the models will be trained. It is important to note that some compounds were below detection limits or not detected in some calibration points, meaning that the 70/30 split of compounds is not perfectly reflected in a 70/30 split of calibration point data

Analysis and Results

Direct Concentration Modeling: As the relationship between normalized volume and input concentration is nearly perfectly linear (see figure 4), the first models tested are linear. A first pass linear model treating the mass spectral information as factors, the position data as continuous, and the date information as continuous has very poor performance, with an OSR² of -.53 and MAE of 18.8. The performance of a linear model for factors which do not adhere to a significance threshold of .1 performs only slightly better, with an OSR² of -.39 and MAE of 18.0. A cart model, using a cp value of .001 and minsplits of 5 performs better but is still not an improvement over the baseline model of predicting based on mean training set response; the OSR² is -.17 and MAE is 15.5. The first model to obtain a positive OSR² result is the random forest model; with a mtry parameter of 34 (selected by cross validation), the random forest model has an OSR² of .16 and MAE of 11.7. A boosted model (with an interaction depth of 5) performs similarly, with an OSR² of .17 and MAE of 12.6. None of these models achieve a sufficiently low error in predicting concentrations.

Additionally, the nonlinear models do an extremely poor job of predicting the general shape of the relationship between volume and concentration. Because these models are nonlinear and the calibration curves are run at only the six discrete calibration point concentrations, predictions for concentrations that fall between the calibration curve points bear a very poor resemblance to the direct linear relationship observed in real calibration curve runs (see figure 6). This problem can be rectified by condensing each calibration curve into a single observation and modeling the slope of the volume/concentration linear fit rather than the individual concentrations. This choice is justified by the extremely high r squared of the linear fits of the individual cal curves (see Appendix A, slope modeling).

Calibration Factor (Slope) Modeling: As the primary linear relationship to the dependent variable has been removed, classification methods are first utilized to explore slope prediction. The first model utilized is a CART model with a minsplits of 2 and cp of .001. This results in an OSR² of -1.31 and MAE of 14.4. A boosted model achieves far better performance, with an OSR² of .35 and MAE of 15.6. The best performance is achieved by a random forest model- with a cross validation selected mtry value of 18, this model achieves an OSR² of .42 and MAE of 11.7.

Discussion:

Challenges: In an earlier iteration of this analysis, an incorrect approach to separating training and test sets caused the performance of all of the modeling approaches utilized on this data set to appear significantly more accurate in predicting than they actually were. The mistake was as follows: instead of splitting the external standard compounds into a training and test set, the entire data set containing all of the calibration points was split into a training and test set. This approach meant that the training set upon which the models were built frequently

contained at least a few calibration points of compounds in the test set, meaning that the predictions for the calibration factors in the test set were at least partially informed by data from the same compounds in the training set. Given that the purpose of this analysis is to generate a method for determining the calibration factor of unknown compounds given the set of external standard compounds, this approach does not achieve its purpose of accurately illustrating how well the response factors of unknown compounds can be predicted. The performance of the models trained on incorrectly split data were extremely good, reaching a maximum of an OSR2 of .8 for a boosted model. By comparison, the best performing model trained on the correctly split data achieved an OSR2 of only half that at .4.

Future Steps: Given the relatively poor performance of the models intended to directly predict individual compounds' calibration factors over time, next steps will involve a significant shift towards an approach that more closely mimics current accepted practices for external standard calibration of unknown compounds.

Future Steps: Given the relatively small number of external standard compounds upon which models may be trained and tested, model performance could likely be improved by experimental design. While random test and training set splitting is ideal for large data sets, with a small set such as that used in this analysis performance would be improved if steps were taken to ensure that the different compound classes were appropriately distributed between the training and test sets.

Another option that will be explored is a closer mimic of the manual methods that are currently utilized within the field; instead of directly modeling the calibration factor, the external standard compounds could be split into response factor categories using kmeans. Initial tests indicate that 4 categories of response factor significantly reduce variance within groups. Models will then be trained to assign compounds to one of these four classes based on mass spectral similarities, and the actual calibration factor of a test set compound would then be determined by the calibration factor of the nearest compound from the same factor class (or an average of the factors of the nearest compounds).

Another similar option would be predicting calibration factors by a kmeans approach, averaging the calibration factors of the nearest external standards in multidimensional feature space to estimate test set compound calibration factor.

Impacts: The ability to accurately quantify unknown compounds in the GoAmazon filter samples will create the opportunity to look into the present and future of atmospheric chemistry in the tropics, and in particular probe the following mystery surrounding huge seasonal differences in the Amazonian aerosol burden. Despite the fact that temperatures and other climate conditions remain extremely stable in the Amazon due to its position on the equator, during the dry season, aerosol concentrations are consistently an order of magnitude higher than during the wet season. While prevailing theories have held that this difference is due to a combination of more burning during the dry season and aerosol removal by wet deposition in the wet season, preliminary analyses indicate that there are more complicated processes at play (de Sa 2019, Shrivastava 2019). While bulk property instruments such as aerosol mass spectrometry are powerful tools in indicating the origin of aerosols, an exploration of the individual compounds is critical in establishing if other processes are responsible for the high aerosol loadings. Of particular interest is exploration of the hypothesis that biomass burning and urban pollution, in addition to directly emitting particles, alter the oxidative conditions of the Amazonian boundary layer in such a way as to induce higher aerosol yields from the BVOC's emitted by the rainforest's rich ecosystem.

GoAmazon Acknowledgements:

"We acknowledge the support from the Central Office of the Large Scale Biosphere Atmosphere Experiment in Amazonia (LBA), the Instituto Nacional de Pesquisas da Amazonia (INPA), and the Instituto Nacional de Pesquisas Espaciais (INPE). The work was conducted under 2009/15235-8 of the Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP)."

References

- de Sá, S. S., B. B.Palm, P. Campuzano-Jost, D.A. Day, W. Hu, G. Isaacman-VanWertz, L.D. Yee, J. Brito, S. Carbone, I.O. Ribeiro, G.G. Cirino, Y.J. Liu, R. Thalman, A. Sedlacek, A. Funk, C. Schumacher, J.E. Shilling, J. Schneider, P. Artaxo, A.H. Goldstein, R.A.F. Souza, J. Wang, K.A. McKinney, H. Barbosa, M.L. Alexander, J.L. Jimenez, a
nd S.T. Martin, Urban influence on the concentration and composition of submicron particulate matter in central Amazonia, *Atmos. Chem. Phys.*, 18, 12185-12206, <https://doi.org/10.5194/acp-18-12185-2018>, 2018.
- de Sá, S. S., L.V. Rizzo, B.B. Palm, P. Campuzano-Jost, D.A. Day, L.D. Yee, R. Wernis, G. Isaacman-VanWertz, J. Brito, S. Carbone, Y.J. Liu, A. Sedlacek, S. Springston, A.H. Goldstein, H.M.J. Barbosa, M.L. Alexander, P. Artaxo, J.L. Jimenez, and S.T. Martin, Contributions of biomass-burning, urban, and biogenic emissions to the concentrations and light-absorbing properties of particulate matter in central Amazonia during the dry season, *Atmos. Chem. Phys.*, 19, 7973-8001, <https://doi.org/10.5194/acp-19-7973-2019>, 2019.
- Shrivastava, M., M.O. Andreae, P. Artaxo, H.M.J. Barbosa, L.K. Berg, J. Brito, J. Ching, R.C. Easter, J. Fan, J.D. Fast, Z. Feng, J.D. Fuentes, M. Glasius, A.H. Goldstein, E.G. Alves, H. Gomes, D. Gu, A. Guenther, S.H. Jathar, S. Kim, Y. Liu, S. Lou, S.T. Martin, V.F. McNeill, A. Medeiros, S.S. de Sá, J.E. Shilling, S.R. Springston, R.A.F. Souza, J.A. Thornton, G. Isaacman-VanWertz, L.D. Yee, R. Ynoue, R.A. Zaveri, A. Zelenyuk, and C. Zhao, Urban pollution greatly enhances formation of natural aerosols over the Amazon rainforest, *Nature Communications*, 10, 1046, <https://doi.org/10.1038/s41467-019-08909-4>, 2019.
- Yee, L. D., Isaacman-VanWertz, G., Wernis, R. A., Meng, M., Rivera, V., Kreisberg, N. M., Hering, S. V., Bering, M. S., Glasius, M., Upshur, M. A., Gray Bé, A., Thomson, R. J., Geiger, F. M., Offenberg, J. H., Lewandowski, M., Kourtchev, I., Kalberer, M., de Sá, S., Martin, S. T., Alexander, M. L., Palm, B. B., Hu, W., Campuzano-Jost, P., Day, D. A., Jimenez, J. L., Liu, Y., McKinney, K. A., Artaxo, P., Viegas, J., Manzi, A., Oliveira, M. B., de Souza, R., Machado, L. A. T., Longo, K., and Goldstein, A. H.: Observations of sesquiterpenes and their oxidation products in central Amazonia during the wet and dry seasons, *Atmos. Chem. Phys.*, 18, 10433-10457, <https://doi.org/10.5194/acp-18-10433-2018>, 2018.
- Zhang, H., L.D. Yee, B.H. Lee, M.P. Curtis, D.R. Worton, G. Isaacman-VanWertz, J.H. Offenberg, M. Lewandowski, T.E. Kleindienst, M.R. Beaver, A.L. Holder, W.A. Lonneman, K.S. Docherty, M. Jaoui, H.O.T. Pye, W. Hu, D.A. Day, P. Campuzano-Jost, J.L. Jimenez, H. Guo, R.J. Weber, J. de Gouw, A.R. Koss, E.S. Edgerton, W. Brune, C. Mohr, F.D. Lopez-Hilfiker, A. Lutz, N.M. Kreisberg, S.R. Spielman, S.V. Hering, K.R. Wilson, J.A. Thornton, and A.H. Goldstein, Monoterpenes are the largest source of summertime organic aerosol in the southeastern United States, *Proceedings of the National Academy of Sciences*, 115 (9) 2038-2043, doi:10.1073/pnas.1717513115, 2018.

Figures

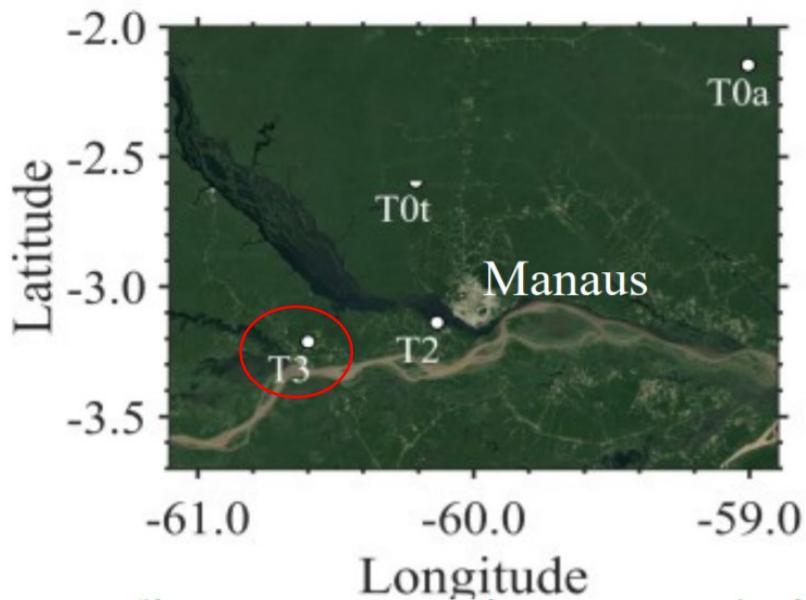


Figure 1. GoAmazon field site location with respect to the city of Manaus. Figure courtesy of de Sa 2019, ACP.

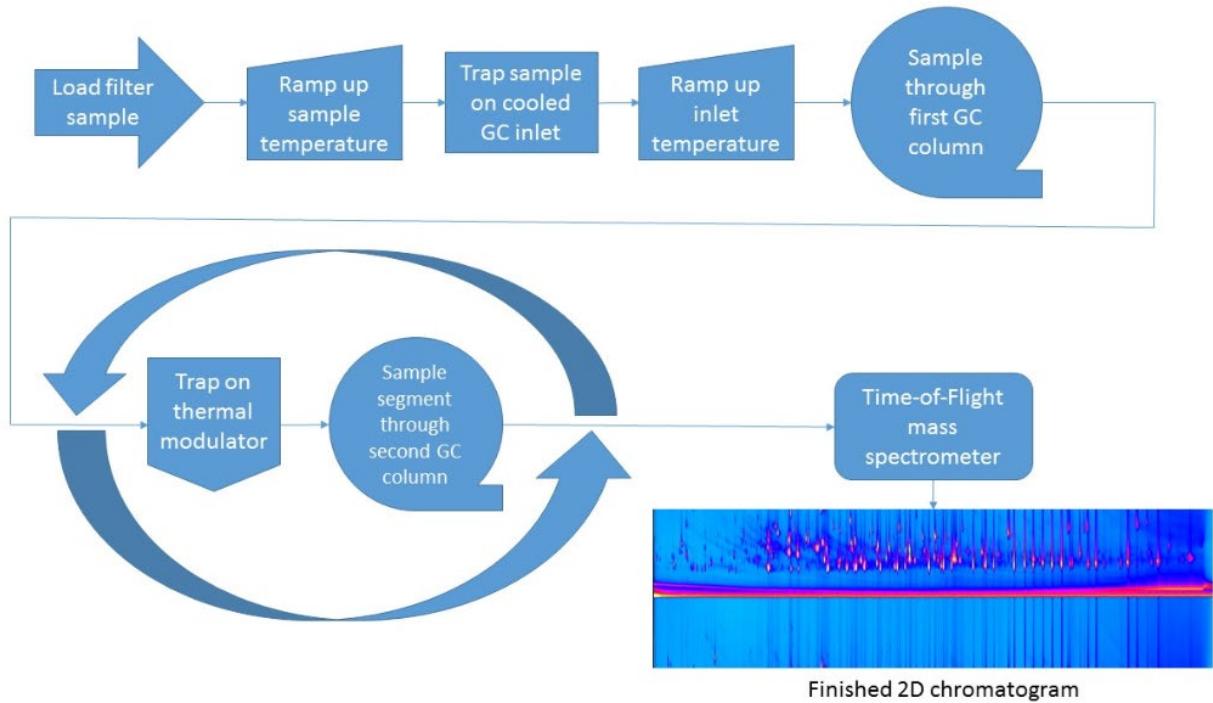


Figure 2: TD-GC_xGC-EI-HR-ToF Mass Spec Diagram. Simplified representation of the Thermal Desorption-2D Gas Chromatography-Electron Ionization-High Resolution-Time of Flight-Mass Spectrometer used to analyze both the Amazonian filter samples and the external standard mixture.

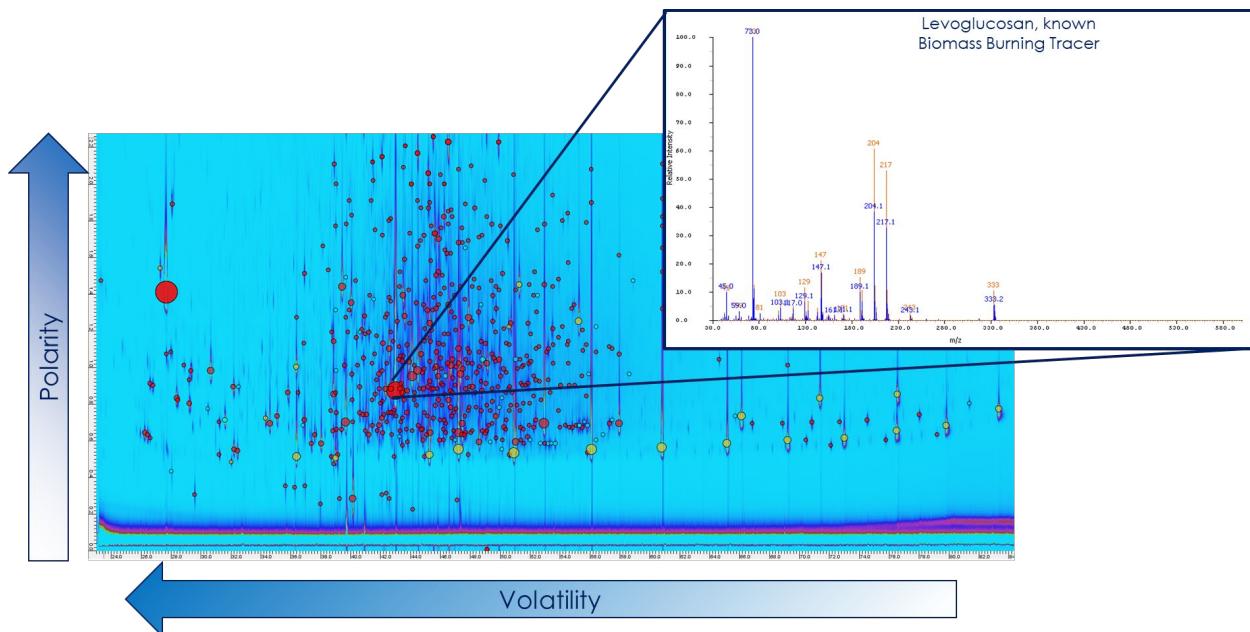


Figure 3: GC-image 2D chromatogram with mass spectra example. Volatility increases right to left, polarity increases vertically, and each circle indicates a compound for which a mass spectrum has been generated. Red circles indicate sample analytes while yellow circles are internal standard compounds.

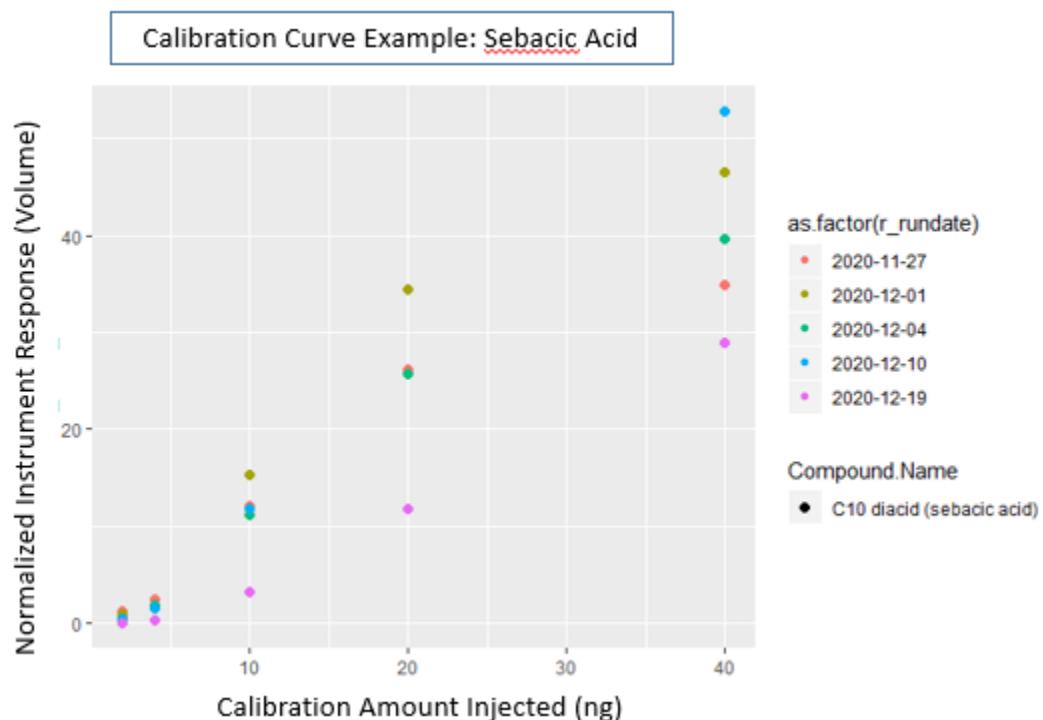


Figure 4: Calibration curve example for sebacic acid

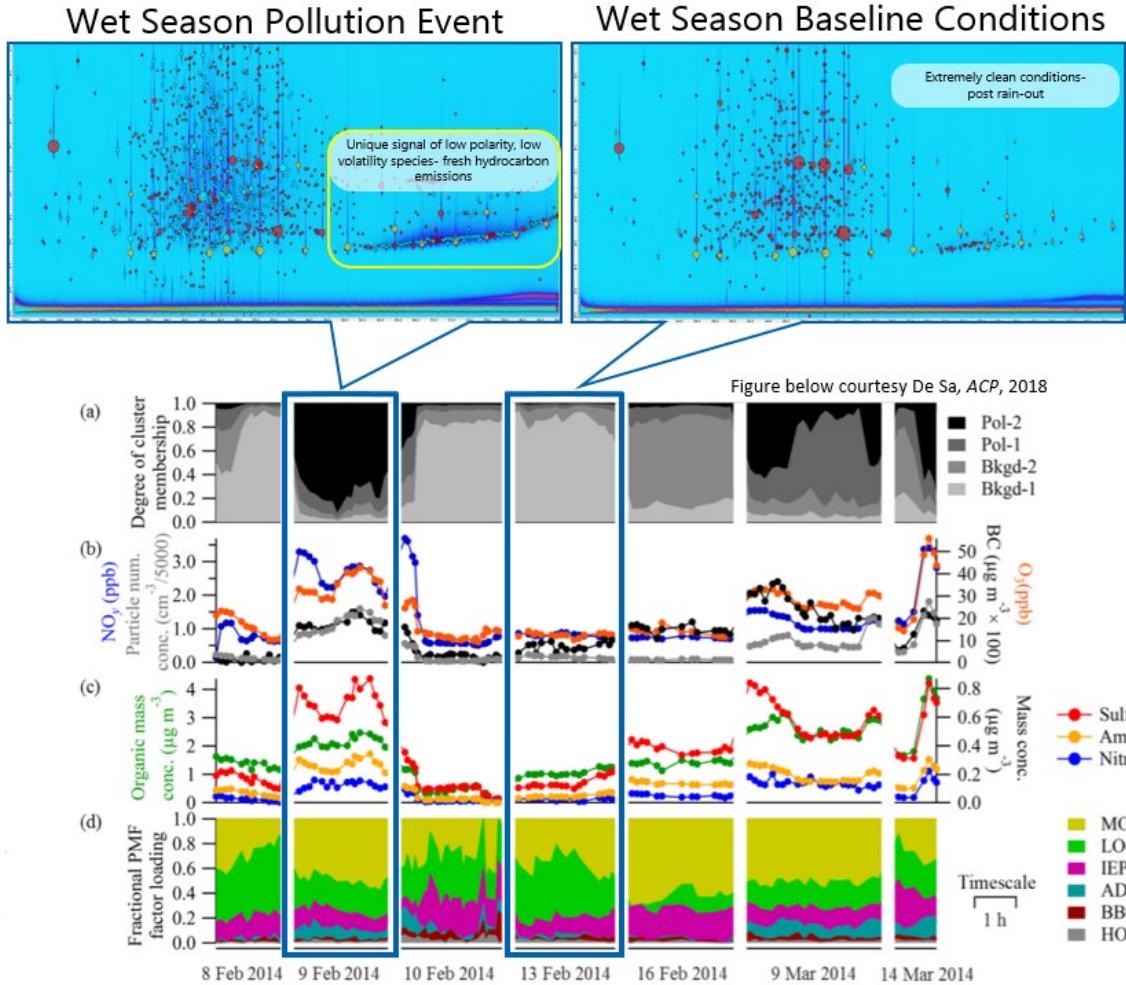


Figure 5. AMS Positive Matrix Factorization example as published in de Sa 2019, coupled with the GCxGC template images selected from the indicated days. PMF factors from AMS illustrated in bottom panel. MO-OOA indicates ‘more oxidized oxidized organic aerosol’, LO-OOA indicates ‘less oxidized oxidized organic aerosol’, IEPOS-SOA indicates isoprene derived secondary organic aerosol, BBOA indicates biomass burning organic aerosol.

Full Predicted Calibration Curve using Boosted Model- Example

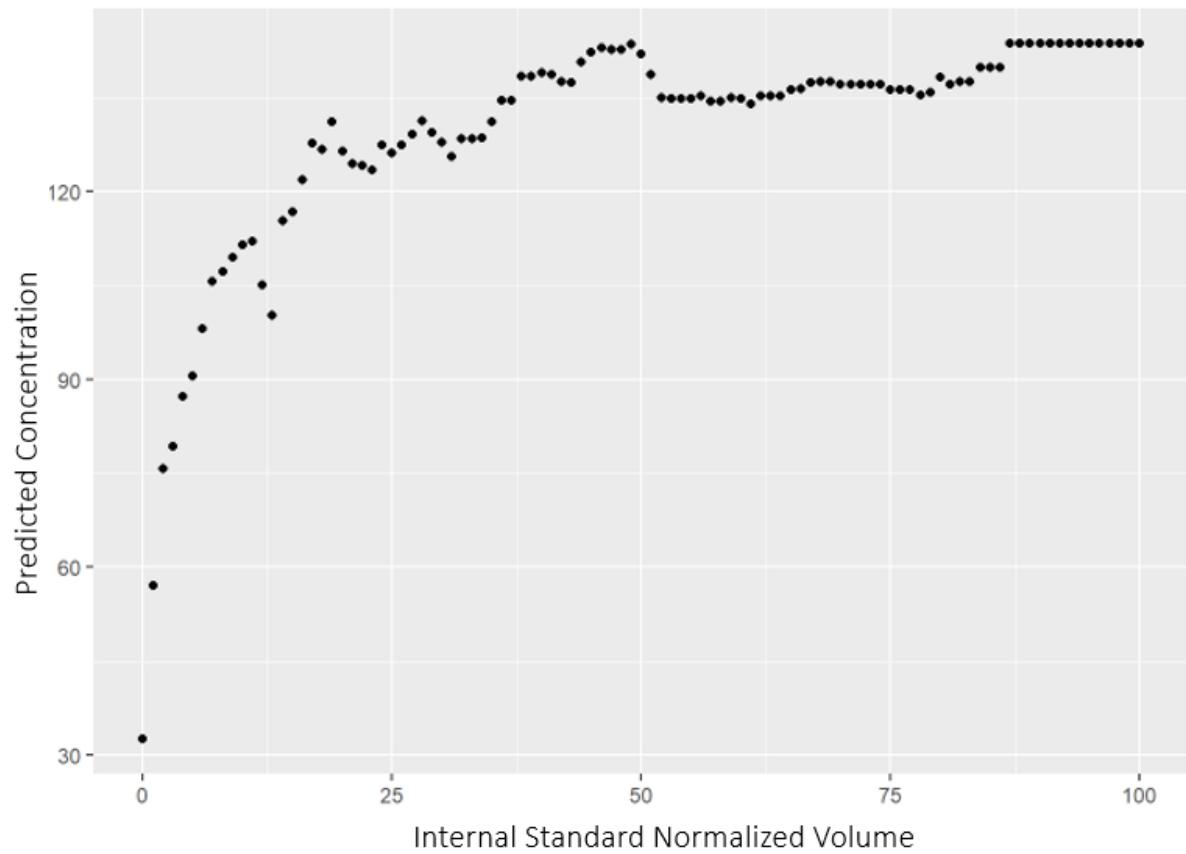


Figure 6. Full predicted calibration curve example using boosted model.

External Standard Modeling Appendix

Emily

10/28/2019

Appendix A: R Code

##Reading In Necessary Libraries

```
## Warning: package 'Rcpp' was built under R version 3.5.3

## Warning: package 'caret' was built under R version 3.5.3

## Loading required package: lattice

## Loading required package: ggplot2

## Warning: package 'randomForest' was built under R version 3.5.3

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

## 
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
## 
##     margin

## Warning: package 'gbm' was built under R version 3.5.3

## Loaded gbm 2.1.5

## Warning: package 'ROCR' was built under R version 3.5.3

## Loading required package: gplots

## Warning: package 'gplots' was built under R version 3.5.3

## 
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
## 
##     lowess

## Warning: package 'caTools' was built under R version 3.5.3

## Warning: package 'dplyr' was built under R version 3.5.3

## 
## Attaching package: 'dplyr'

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

```
## The following object is masked from 'package:randomForest':  
##  
##     combine
```

```
## The following objects are masked from 'package:stats':  
##  
##     filter, lag
```

```
## The following objects are masked from 'package:base':  
##  
##     intersect, setdiff, setequal, union
```

```
## Warning: package 'tidyverse' was built under R version 3.5.3
```

```
## -- Attaching packages -----  
----- tidyverse 1.2.1 --
```

```
## v tibble  2.1.3      v purrr   0.2.5  
## v tidyr   1.0.0      v stringr 1.3.1  
## v readr   1.3.1      vforcats 0.3.0
```

```
## Warning: package 'tibble' was built under R version 3.5.3
```

```
## Warning: package 'tidyr' was built under R version 3.5.3
```

```
## -- Conflicts -----  
---- tidyverse_conflicts() --  
## x dplyr::combine()      masks randomForest::combine()  
## x dplyr::filter()       masks stats::filter()  
## x dplyr::lag()          masks stats::lag()  
## x purrr::lift()         masks caret::lift()  
## x randomForest::margin() masks ggplot2::margin()  
## x dplyr::select()       masks MASS::select()
```

```
##  
## Attaching package: 'scales'
```

```
## The following object is masked from 'package:purrr':  
##  
##     discard
```

```
## The following object is masked from 'package:readr':  
##  
##     col_factor
```

```
## Warning: package 'akima' was built under R version 3.5.3
```

```
## Warning: package 'OrgMassSpecR' was built under R version 3.5.3
```

```
##  
## Attaching package: 'glue'
```

```
## The following object is masked from 'package:dplyr':  
##  
##     collapse
```

```
## Warning: package 'openxlsx' was built under R version 3.5.3
```

```
## Warning: package 'janitor' was built under R version 3.5.3
```

```

## 
## Attaching package: 'janitor'

## The following objects are masked from 'package:stats':
## 
##     chisq.test, fisher.test

## Warning: package 'rpart' was built under R version 3.5.3

## Warning: package 'rpart.plot' was built under R version 3.5.3

```

Parsing Out Mass Spectra

In this section, a CSV file containing the names, locations, and mass spectra of all of the external standard components is read in. The mass spectra are in the form of a semicolon separated string. The principle peaks and the mass differences between key peaks are the key indicators of functional groups that exert the greatest influence on the calibration factor of the compound in question. In this section, the top 10 peaks (by intensity) are identified for each compound. Then, the top 21 peaks (the 21 peaks which appear in the most compounds from the external standard list) are turned into factors, for which each compound indicates 'true' or 'false' depending on whether one of the top 10 peaks for that compound exists on the list.

```

ES_table_o = read_csv(file = "1204_2010_amze_unnamedSESEQ_1_ms.csv")

## Warning: Duplicated column names deduplicated: 'RT2' => 'RT2_1' [24]

## Parsed with column specification:
## cols(
##   .default = col_logical(),
##   data_file = col_character(),
##   BlobID = col_double(),
##   RT1 = col_double(),
##   RT2 = col_double(),
##   Description = col_character(),
##   Name = col_character(),
##   Column_Type = col_character(),
##   Retention_index = col_double(),
##   d_alkane_RTI = col_double(),
##   Contributor = col_character(),
##   `Num Peaks` = col_double(),
##   MS = col_character()
## )

## See spec(...) for full column specifications.

trim.leading <- function (x)  sub("^\\s+", "", x)

ES_table_t = ES_table_o %>%
  mutate(MS = strsplit(as.character(MS), ";")) %>%
  unnest(MS) %>%
  filter(MS != "") %>%
  mutate(MS2 = trim.leading(MS))

ES_table_3 = separate(data = ES_table_t, col = MS2, into = c("mz", "intensity"), sep = " ")

## Warning: Expected 2 pieces. Missing pieces filled with `NA` in 135 rows
## [222, 336, 473, 782, 890, 1046, 1116, 1231, 1539, 1697, 1828, 1915, 2054,
## 2170, 2282, 2492, 2693, 2930, 3077, 3291, ...].

```

```

ES_table_full = ES_table_3 %>%
  mutate(intensity = as.numeric(intensity)) %>%
  mutate(mz = as.numeric(mz)) %>%
  arrange(desc(intensity)) %>%
  arrange(desc(Name)) %>%
  group_by(Name) %>%
  mutate(my_ranks = order(order(intensity, decreasing=TRUE))) %>%
  ungroup()

ES_table_trimmed = ES_table_full %>%
  filter(my_ranks < 11)

mz_tab <- as.data.frame(table(ES_table_trimmed$mz))
mz_tab <- mz_tab %>%
  mutate(mz = as.character(Var1)) %>%
  mutate(mz = as.numeric(mz))
mz_tab <- mz_tab %>%
  arrange(desc(Freq)) %>%
  mutate(freq_ranks = order(order(Freq, decreasing = TRUE)))

ES_table_ranked <- ES_table_trimmed %>%
  left_join(mz_tab) %>%
  filter(freq_ranks < 21)

```

```

## Joining, by = "mz"

```

```

ES_common_mz = ES_table_ranked %>%
  dplyr::select(Name, mz) %>%
  mutate(mz = paste("mz_", mz, sep = "_")) %>%
  mutate(mz_obs = TRUE)

wide_mz = ES_common_mz %>%
  spread(mz, mz_obs, fill = FALSE)

```

Next, the losses are determined. The mass differences between the top 5 peaks for each compound are evaluated and turned into their own list. As in the case of the peak list, the 20 peaks which appear most frequently become true/false factors for inclusion in future analysis

```

# figuring out losses

unique_names = unique(ES_table_trimmed$Name)

x = seq(1, 5, 1)
y = x

d2 <- expand.grid(x = x, y = y, KEEP.OUT.ATTRS = FALSE)
d2 <- d2 %>%
  filter(y > x)

xt = d2$x
yt = d2$y

names_losses = as.character(rep(unique_names, each = length(d2$x)))
losses_vec = rep(999, times = length(names_losses))
loss_num = rep(seq(1, 10, 1), times = length(unique_names))

losses <- data.frame(names_losses, losses_vec, loss_num)
losses = losses %>%
  mutate(names_losses = as.character(names_losses))

for(i in 1:1350) {

  name_t = names_losses[i]
  loss_num_t = losses$loss_num[i]

  upper_index = xt[loss_num_t]
  lower_index = yt[loss_num_t]

  df_t = ES_table_trimmed %>%
    filter(Name == name_t)

  upper_mz = df_t$mz[upper_index]
  lower_mz = df_t$mz[lower_index]

  losses$losses_vec[i] = abs(upper_mz - lower_mz)

}

loss_tab <- as.data.frame(table(losses$losses_vec))
loss_tab <- loss_tab %>%
  arrange(desc(Freq)) %>%
  mutate(freq_ranks = order(order(Freq, decreasing = TRUE))) %>%
  mutate(Var1 = as.character(Var1)) %>%
  mutate(Var1 = as.numeric(Var1))

losses = losses %>%
  left_join(loss_tab, by = c("losses_vec"= "Var1"))

losses_trimmed = losses %>%
  filter(freq_ranks<21)

ES_common_losses = losses_trimmed %>%
  dplyr::select(names_losses, losses_vec) %>%
  mutate(Loss_obs = TRUE)

wide_losses = reshape(ES_common_losses, idvar = "names_losses", timevar = "losses_vec", direction = "wide")

```

```

## Warning in reshapeWide(data, idvar = idvar, timevar = timevar, varying =
## varying, : multiple rows match for losses_vec=28: first taken

```

```

## Warning in reshapeWide(data, idvar = idvar, timevar = timevar, varying =
## varying, : multiple rows match for losses_vec=29: first taken

```

```

## Warning in reshapeWide(data, idvar = idvar, timevar = timevar, varying =
## varying, : multiple rows match for losses_vec=1: first taken

## Warning in reshapeWide(data, idvar = idvar, timevar = timevar, varying =
## varying, : multiple rows match for losses_vec=16: first taken

## Warning in reshapeWide(data, idvar = idvar, timevar = timevar, varying =
## varying, : multiple rows match for losses_vec=44: first taken

## Warning in reshapeWide(data, idvar = idvar, timevar = timevar, varying =
## varying, : multiple rows match for losses_vec=12: first taken

## Warning in reshapeWide(data, idvar = idvar, timevar = timevar, varying =
## varying, : multiple rows match for losses_vec=15: first taken

## Warning in reshapeWide(data, idvar = idvar, timevar = timevar, varying =
## varying, : multiple rows match for losses_vec=30: first taken

## Warning in reshapeWide(data, idvar = idvar, timevar = timevar, varying =
## varying, : multiple rows match for losses_vec=2: first taken

## Warning in reshapeWide(data, idvar = idvar, timevar = timevar, varying =
## varying, : multiple rows match for losses_vec=14: first taken

## Warning in reshapeWide(data, idvar = idvar, timevar = timevar, varying =
## varying, : multiple rows match for losses_vec=42: first taken

## Warning in reshapeWide(data, idvar = idvar, timevar = timevar, varying =
## varying, : multiple rows match for losses_vec=13: first taken

## Warning in reshapeWide(data, idvar = idvar, timevar = timevar, varying =
## varying, : multiple rows match for losses_vec=26: first taken

ES_common_losses = losses_trimmed %>%
  dplyr::select(names_losses, losses_vec) %>%
  mutate(losses_vec = paste("loss_", losses_vec, sep = "_")) %>%
  mutate(Loss_obs = TRUE) %>%
  distinct()

wide_losses = ES_common_losses %>%
  spread(losses_vec, Loss_obs, fill = FALSE)

```

Category Assignment: Future Steps

In the following section, the compounds are classified by functional group and random forest modeling is utilized to predict the category based on mass spectrum. The accuracy of this approach is poor, likely because many compounds have multiple functional groups apparent in their mass spectra so assigning a compound to only one functional group class may be an inappropriate approach. Future investigations will attempt to predict on a true/false basis whether a given compound (meaning its mass spectrum) is indicative of a particular functional group rather than approaching as a classification problem.

```
cat <- read_csv("Categories_ES.csv")
```

```

## Parsed with column specification:
## cols(
##   Name = col_character(),
##   RT1 = col_double(),
##   RT2 = col_double(),
##   Retention_index = col_double(),
##   d_alkane_RTI = col_double(),
##   Func_group_cat = col_character()
## )

```

```

Cat_full <- cat %>%
  left_join(wide_losses, by = c("Name" = "names_losses")) %>%
  left_join(wide_mz) %>%
  mutate(Func_group_cat = as.factor(Func_group_cat))

## Joining, by = "Name"

Cat_full[is.na(Cat_full)] <- FALSE

set.seed(100)
split = sample.split(Cat_full$Func_group_cat, SplitRatio = 0.75)

# what is a split?
Cat.train <- filter(Cat_full, split == TRUE) # is split a variable in loans?
Cat.test <- filter(Cat_full, split == FALSE)

set.seed(144)
train.rf = train(Func_group_cat~, data = Cat.train, method = "rf", tuneGrid = data.frame(mtry=seq(1, 25, 1)),
), trControl = trainControl(method = "cv", number = 5), metric = "Accuracy")

best.rf = train.rf$finalModel
best.rf

rf.plot <- ggplot(train.rf$results, aes(x=mtry, y=Accuracy)) + geom_line(lwd=2) +
  ylab("Accuracy of Predictions")
rf.plot

Cat.mm = as.data.frame(model.matrix(Func_group_cat ~., data = Cat.test))

set.seed(144)
pred.best.rf = predict(best.rf, newdata = Cat.mm, type = "class")

t_rf_all = table(Cat.test$Func_group_cat, pred.best.rf)
t_rf_all

accuracy.rf = (t_rf_all[1,1]+t_rf_all[2,2]+t_rf_all[3,3]+t_rf_all[4,4])/nrow(Cat.test)
accuracy.rf

```

Importing all of the Blob Table Index

Each point of each of the 5 calibration curves is saved as a blob table file containing the abundance of each of the external standard compounds in that sample run (aka cal curve point). In order to read in these files with the necessary general information on what day the cal curve occurred and what point of the cal curve each run corresponds to, an index is read in with the file names and additional information.

```

#reading in the index of the blob tables
bt_sum <- read_csv("Blob_table_summary_ES.csv")

## Parsed with column specification:
## cols(
##   File_num = col_character(),
##   Category = col_character(),
##   Filter_num = col_double(),
##   Filter_punches = col_double(),
##   IOP = col_double(),
##   AMZ_date = col_character(),
##   T_O_D = col_double(),
##   Date_num = col_double(),
##   Internal_load = col_double(),
##   Cal_Point_number = col_double(),
##   Run_date = col_character(),
##   Run_date_num = col_double(),
##   Cal_Point = col_character()
## )

```

```

#adding a date column that R will recognize

bt_sum <- bt_sum %>% mutate(rundate = as.Date(Run_date, format = "%m/%d/%y"))

#excel_numeric_to_date(as.numeric(as.character(Run_date_num)), date_system = "modern")

# making a function that will make the correct file names from the table of samples so that my actual sample
files can be read in
make_file_name_es <- function(date_string) {
  file_start <- "Cal_Curve_blobtables/"
  file_end <- ".h5_cc_Blob_Table.csv"
  full_name <- paste(file_start,date_string,file_end, sep = "")
  return(full_name)
}

# function to read in files
read_bt_files <- function(fi_name_short) {

  fi_name <- make_file_name_es(fi_name_short)
  temp_bt <-> read_csv(file = fi_name)
  temp_bt <- temp_bt %>% mutate(File_num = fi_name_short)
  temp_bt <-> temp_bt
  return(temp_bt)
}

#reading in all of the blob tables and turning them into one big blob table- creating a long table
make_massive_table <- function(summary_table){

  M <- read_bt_files(as.character(summary_table$File_num[1]))

  for(i in 2:length(summary_table$File_num)){
    t <- read_bt_files(as.character(summary_table$File_num[i]))
    Mnew <- rbind(M, t)
    M <- Mnew
    print(i)
  }
  M_t <-> M
}

make_massive_table(bt_sum)

```

```

## Parsed with column specification:
## cols(
##   .default = col_double(),
##   `Compound Name` = col_character(),
##   Description = col_logical(),
##   `Group Name` = col_logical(),
##   `Constellation Name` = col_logical(),
##   Inclusion = col_logical(),
##   `Library Name` = col_character(),
##   `Review Status` = col_logical(),
##   `Quantifier(1)` = col_logical(),
##   `Quantifier(2)` = col_logical()
## )

```

```

## See spec(...) for full column specifications.

```

```
## Parsed with column specification:  
## cols(  
##   .default = col_double(),  
##   `Compound Name` = col_character(),  
##   Description = col_logical(),  
##   `Group Name` = col_logical(),  
##   `Constellation Name` = col_logical(),  
##   Inclusion = col_logical(),  
##   `Library Name` = col_character(),  
##   `Review Status` = col_logical(),  
##   `Quantifier(1)` = col_logical(),  
##   `Quantifier(2)` = col_logical()  
## )
```

```
## See spec(...) for full column specifications.
```

```
## [1] 2
```

```
## Parsed with column specification:  
## cols(  
##   .default = col_double(),  
##   `Compound Name` = col_character(),  
##   Description = col_logical(),  
##   `Group Name` = col_logical(),  
##   `Constellation Name` = col_logical(),  
##   Inclusion = col_logical(),  
##   `Library Name` = col_character(),  
##   `Review Status` = col_logical(),  
##   `Quantifier(1)` = col_logical(),  
##   `Quantifier(2)` = col_logical()  
## )  
## See spec(...) for full column specifications.
```

```
## [1] 3
```

```
## Parsed with column specification:  
## cols(  
##   .default = col_double(),  
##   `Compound Name` = col_character(),  
##   Description = col_logical(),  
##   `Group Name` = col_logical(),  
##   `Constellation Name` = col_logical(),  
##   Inclusion = col_logical(),  
##   `Library Name` = col_character(),  
##   `Review Status` = col_logical(),  
##   `Quantifier(1)` = col_logical(),  
##   `Quantifier(2)` = col_logical()  
## )  
## See spec(...) for full column specifications.
```

```
## [1] 4
```

```
## Parsed with column specification:  
## cols(  
##   .default = col_double(),  
##   `Compound Name` = col_character(),  
##   Description = col_logical(),  
##   `Group Name` = col_logical(),  
##   `Constellation Name` = col_logical(),  
##   Inclusion = col_logical(),  
##   `Library Name` = col_character(),  
##   `Review Status` = col_logical(),  
##   `Quantifier(1)` = col_logical(),  
##   `Quantifier(2)` = col_logical()  
## )  
## See spec(...) for full column specifications.
```

```
## [1] 5
```

```
## Parsed with column specification:  
## cols(  
##   .default = col_double(),  
##   `Compound Name` = col_character(),  
##   Description = col_logical(),  
##   `Group Name` = col_logical(),  
##   `Constellation Name` = col_logical(),  
##   Inclusion = col_logical(),  
##   `Library Name` = col_character(),  
##   `Review Status` = col_logical(),  
##   `Quantifier(1)` = col_logical(),  
##   `Quantifier(2)` = col_logical()  
## )  
## See spec(...) for full column specifications.
```

```
## [1] 6
```

```
## Parsed with column specification:  
## cols(  
##   .default = col_double(),  
##   `Compound Name` = col_character(),  
##   Description = col_logical(),  
##   `Group Name` = col_logical(),  
##   `Constellation Name` = col_logical(),  
##   Inclusion = col_logical(),  
##   `Library Name` = col_character(),  
##   `Review Status` = col_logical(),  
##   `Quantifier(1)` = col_logical(),  
##   `Quantifier(2)` = col_logical()  
## )  
## See spec(...) for full column specifications.
```

```
## [1] 7
```

```
## Parsed with column specification:  
## cols(  
##   .default = col_double(),  
##   `Compound Name` = col_character(),  
##   Description = col_logical(),  
##   `Group Name` = col_logical(),  
##   `Constellation Name` = col_logical(),  
##   Inclusion = col_logical(),  
##   `Library Name` = col_character(),  
##   `Review Status` = col_logical(),  
##   `Quantifier(1)` = col_logical(),  
##   `Quantifier(2)` = col_logical()  
## )  
## See spec(...) for full column specifications.
```

```
## [1] 8
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   `Compound Name` = col_character(),
##   Description = col_logical(),
##   `Group Name` = col_logical(),
##   `Constellation Name` = col_logical(),
##   Inclusion = col_logical(),
##   `Library Name` = col_character(),
##   `Review Status` = col_logical(),
##   `Quantifier(1)` = col_logical(),
##   `Quantifier(2)` = col_logical()
## )
## See spec(...) for full column specifications.
```

```
## [1] 9
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   `Compound Name` = col_character(),
##   Description = col_logical(),
##   `Group Name` = col_logical(),
##   `Constellation Name` = col_logical(),
##   Inclusion = col_logical(),
##   `Library Name` = col_character(),
##   `Review Status` = col_logical(),
##   `Quantifier(1)` = col_logical(),
##   `Quantifier(2)` = col_logical()
## )
## See spec(...) for full column specifications.
```

```
## [1] 10
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   `Compound Name` = col_character(),
##   Description = col_logical(),
##   `Group Name` = col_logical(),
##   `Constellation Name` = col_logical(),
##   Inclusion = col_logical(),
##   `Library Name` = col_character(),
##   `Review Status` = col_logical(),
##   `Quantifier(1)` = col_logical(),
##   `Quantifier(2)` = col_logical()
## )
## See spec(...) for full column specifications.
```

```
## [1] 11
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   `Compound Name` = col_character(),
##   Description = col_logical(),
##   `Group Name` = col_logical(),
##   `Constellation Name` = col_logical(),
##   Inclusion = col_logical(),
##   `Library Name` = col_character(),
##   `Review Status` = col_logical(),
##   `Quantifier(1)` = col_logical(),
##   `Quantifier(2)` = col_logical()
## )
## See spec(...) for full column specifications.
```

```
## [1] 12
```

```
## Parsed with column specification:  
## cols(  
##   .default = col_double(),  
##   `Compound Name` = col_character(),  
##   Description = col_logical(),  
##   `Group Name` = col_logical(),  
##   `Constellation Name` = col_logical(),  
##   Inclusion = col_logical(),  
##   `Library Name` = col_character(),  
##   `Review Status` = col_logical(),  
##   `Quantifier(1)` = col_logical(),  
##   `Quantifier(2)` = col_logical()  
## )  
## See spec(...) for full column specifications.
```

```
## [1] 13
```

```
## Parsed with column specification:  
## cols(  
##   .default = col_double(),  
##   `Compound Name` = col_character(),  
##   Description = col_logical(),  
##   `Group Name` = col_logical(),  
##   `Constellation Name` = col_logical(),  
##   Inclusion = col_logical(),  
##   `Library Name` = col_character(),  
##   `Review Status` = col_logical(),  
##   `Quantifier(1)` = col_logical(),  
##   `Quantifier(2)` = col_logical()  
## )  
## See spec(...) for full column specifications.
```

```
## [1] 14
```

```
## Parsed with column specification:  
## cols(  
##   .default = col_double(),  
##   `Compound Name` = col_character(),  
##   Description = col_logical(),  
##   `Group Name` = col_logical(),  
##   `Constellation Name` = col_logical(),  
##   Inclusion = col_logical(),  
##   `Library Name` = col_character(),  
##   `Review Status` = col_logical(),  
##   `Quantifier(1)` = col_logical(),  
##   `Quantifier(2)` = col_logical()  
## )  
## See spec(...) for full column specifications.
```

```
## [1] 15
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   `Compound Name` = col_character(),
##   Description = col_logical(),
##   `Group Name` = col_logical(),
##   `Constellation Name` = col_logical(),
##   Inclusion = col_logical(),
##   `Library Name` = col_character(),
##   `Review Status` = col_logical(),
##   `Quantifier(1)` = col_logical(),
##   `Quantifier(2)` = col_logical()
## )
## See spec(...) for full column specifications.
```

```
## [1] 16
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   `Compound Name` = col_character(),
##   Description = col_logical(),
##   `Group Name` = col_logical(),
##   `Constellation Name` = col_logical(),
##   Inclusion = col_logical(),
##   `Library Name` = col_character(),
##   `Review Status` = col_logical(),
##   `Quantifier(1)` = col_logical(),
##   `Quantifier(2)` = col_logical()
## )
## See spec(...) for full column specifications.
```

```
## [1] 17
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   `Compound Name` = col_character(),
##   Description = col_logical(),
##   `Group Name` = col_logical(),
##   `Constellation Name` = col_logical(),
##   Inclusion = col_logical(),
##   `Library Name` = col_character(),
##   `Review Status` = col_logical(),
##   `Quantifier(1)` = col_logical(),
##   `Quantifier(2)` = col_logical()
## )
## See spec(...) for full column specifications.
```

```
## [1] 18
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   `Compound Name` = col_character(),
##   Description = col_logical(),
##   `Group Name` = col_logical(),
##   `Constellation Name` = col_logical(),
##   Inclusion = col_logical(),
##   `Library Name` = col_character(),
##   `Review Status` = col_logical(),
##   `Quantifier(1)` = col_logical(),
##   `Quantifier(2)` = col_logical()
## )
## See spec(...) for full column specifications.
```

```
## [1] 19
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   `Compound Name` = col_character(),
##   Description = col_logical(),
##   `Group Name` = col_logical(),
##   `Constellation Name` = col_logical(),
##   Inclusion = col_logical(),
##   `Library Name` = col_character(),
##   `Review Status` = col_logical(),
##   `Quantifier(1)` = col_logical(),
##   `Quantifier(2)` = col_logical()
## )
## See spec(...) for full column specifications.
```

```
## [1] 20
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   `Compound Name` = col_character(),
##   Description = col_logical(),
##   `Group Name` = col_logical(),
##   `Constellation Name` = col_logical(),
##   Inclusion = col_logical(),
##   `Library Name` = col_character(),
##   `Review Status` = col_logical(),
##   `Quantifier(1)` = col_logical(),
##   `Quantifier(2)` = col_logical()
## )
## See spec(...) for full column specifications.
```

```
## [1] 21
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   `Compound Name` = col_character(),
##   Description = col_logical(),
##   `Group Name` = col_logical(),
##   `Constellation Name` = col_logical(),
##   Inclusion = col_logical(),
##   `Library Name` = col_character(),
##   `Review Status` = col_logical(),
##   `Quantifier(1)` = col_logical(),
##   `Quantifier(2)` = col_logical()
## )
## See spec(...) for full column specifications.
```

```
## [1] 22
```

```

## Parsed with column specification:
## cols(
##   .default = col_double(),
##   `Compound Name` = col_character(),
##   Description = col_logical(),
##   `Group Name` = col_logical(),
##   `Constellation Name` = col_logical(),
##   Inclusion = col_logical(),
##   `Library Name` = col_character(),
##   `Review Status` = col_logical(),
##   `Quantifier(1)` = col_logical(),
##   `Quantifier(2)` = col_logical()
## )
## See spec(...) for full column specifications.

```

```
# [1] 23
```

```

## Parsed with column specification:
## cols(
##   .default = col_double(),
##   `Compound Name` = col_character(),
##   Description = col_logical(),
##   `Group Name` = col_logical(),
##   `Constellation Name` = col_logical(),
##   Inclusion = col_logical(),
##   `Library Name` = col_character(),
##   `Review Status` = col_logical(),
##   `Quantifier(1)` = col_logical(),
##   `Quantifier(2)` = col_logical()
## )
## See spec(...) for full column specifications.

```

```
# [1] 24
```

```

#adding in all information from the summary table
M_t_full <- M_t %>% left_join(bt_sum)

```

```
## Joining, by = "File_num"
```

```

# tidying column names
names(M_t_full) <- make.names(names(M_t_full),unique = TRUE)

```

Importing and Cleaning Raw Blob Tables

The raw blob tables may contain a variety of false matches (aka compounds labeled as external standard but that are not actually standard components). In this section mass spectral match factors, location statistics (difference between library retention index and retention index in the given blob table), and size are used to screen out incorrect matches.

```

# filtering out bad LRI matches
Match_factor_floor <- 750
Reverse_match_factor_floor <- 800
LRI_diff_floor <- 10

fb_match_factor_floor <- 600
fb_reverse_match_factor_floor <- 700
IS_lib_name <- "amzi0503_b"
FB_lib_name <- "fb_amz_compiled_t"
ES_lib_name <- "amze_unnamedsesq_1"

rt2_floor <- .5 # note: check on this, but for purposes of indexing retention times in 2d this is important

# creating a column of the differences in linear retention indecies so that poor matches can be screened out
M_t_LRI <- M_t_full %>% mutate(LRI_diff = abs(Library.RI-LRI.I))

# identifying the internal standard
IS_goodmatch <- M_t_LRI %>%
  filter(Library.Name == IS_lib_name) %>%
  filter(LRI_diff < LRI_diff_floor-5) %>%
  filter(Library.Match.Factor > Match_factor_floor)

FB_goodmatch <- M_t_LRI %>%
  filter(Library.Name == FB_lib_name) %>%
  filter(LRI_diff < LRI_diff_floor) %>%
  filter(Library.Match.Factor > fb_match_factor_floor || Library.Reverse.Match.Factor > fb_reverse_match_factor_floor)

M_t_all_analytes <- M_t_LRI %>%
  anti_join(IS_goodmatch) %>%
  anti_join(FB_goodmatch)

M_t_analyte_unique <- M_t_all_analytes %>%
  filter(Library.Name != "amzi0503_b") %>%
  filter(Library.Name != "fb_amz_compiled_t") %>%
  filter(LRI_diff < LRI_diff_floor) %>%
  filter(Library.Match.Factor > Match_factor_floor || Library.Reverse.Match.Factor > Reverse_match_factor_floor) %>%
  arrange(desc(Volume)) %>%
  arrange(Compound.Name) %>%
  arrange(File_num) %>%
  distinct(Compound.Name, File_num, .keep_all = TRUE)

```

The metrics used for screening out bad matches are up for some level of interpretation and as a result it is important to keep track of performance metrics.

```

# tracking the mass of all of the analytes before bad matches are screened out for later analysis of library performance
M_t_all_analytes_volcount <- M_t_all_analytes %>%
  group_by(File_num) %>%
  summarise(rawvol = sum(Volume))

# tracking the number of all analutes in
M_t_all_analytes_ncount <- M_t_all_analytes %>%
  count(File_num) %>%
  rename(total_num_analyte = n)

M_t_match_analytes_volcount <- M_t_analyte_unique %>%
  group_by(File_num) %>%
  summarise(rawvol_match = sum(Volume))

M_t_match_analytes_ncount <- M_t_analyte_unique %>%
  count(File_num) %>%
  rename(total_num_analyte_match = n)

M_t_analyte_summary <- M_t_all_analytes_volcount %>%
  left_join(M_t_all_analytes_ncount) %>%
  left_join(M_t_match_analytes_volcount) %>%
  left_join(M_t_match_analytes_ncount) %>%
  mutate(perct_nfound = (total_num_analyte_match/total_num_analyte)*100) %>%
  mutate(perct_volfound = (rawvol_match/rawvol)*100)

```

```

## Joining, by = "File_num"
## Joining, by = "File_num"
## Joining, by = "File_num"

```

```

compound_pop <- M_t_analyte_unique %>%
  count(Compound.Name) %>%
  rename(comp.n = n) %>%
  mutate(pct_of_samp = (comp.n/max(comp.n)*100))



```

```

## Joining, by = "File_num"

```

```

#
#perct_comps_traced %>%
#  ggplot(aes(y = pct_found)) +
#  geom_boxplot()

summary(perct_comps_traced$pct_found)

```

```
##      Min. 1st Qu. Median     Mean 3rd Qu.    Max.
##  0.1344  0.2109  0.3370  0.3755  0.4911  0.7701
```

```
M_t_analyte_unique <- M_t_analyte_unique %>% left_join(num_analyte_unique)
```

```
## Joining, by = "File_num"
```

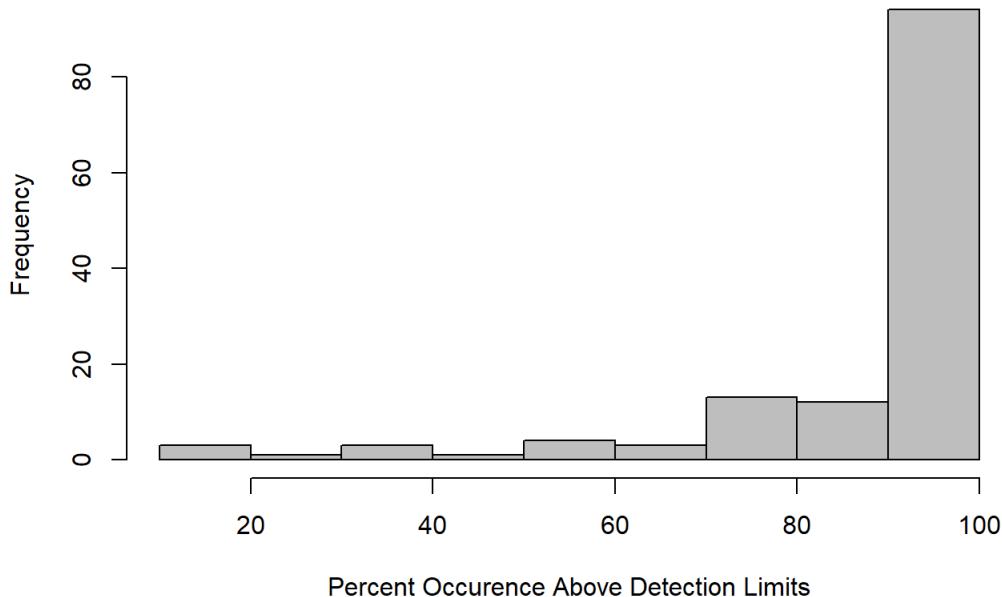
```
compound_pop <- M_t_analyte_unique %>%
  count(Compound.Name) %>%
  rename(comp.n = n) %>%
  mutate(pct_of_samp = (comp.n/max(comp.n)*100))

table(compound_pop$comp.n)
```

```
## 
##   3   4   5   8   9  12  13  14  15  16  17  18  19  20  21  22  23  24
##   2   1   1   2   1   1   3   2   1   4   6   3   7   5   9  30  55
```

```
hist(compound_pop$pct_of_samp,
  main = paste("Histogram of", nrow(compound_pop), "Traced Compounds\nOver 5 Day Test Period"),
  xlab = "Percent Occurrence Above Detection Limits",
  col = "grey")
```

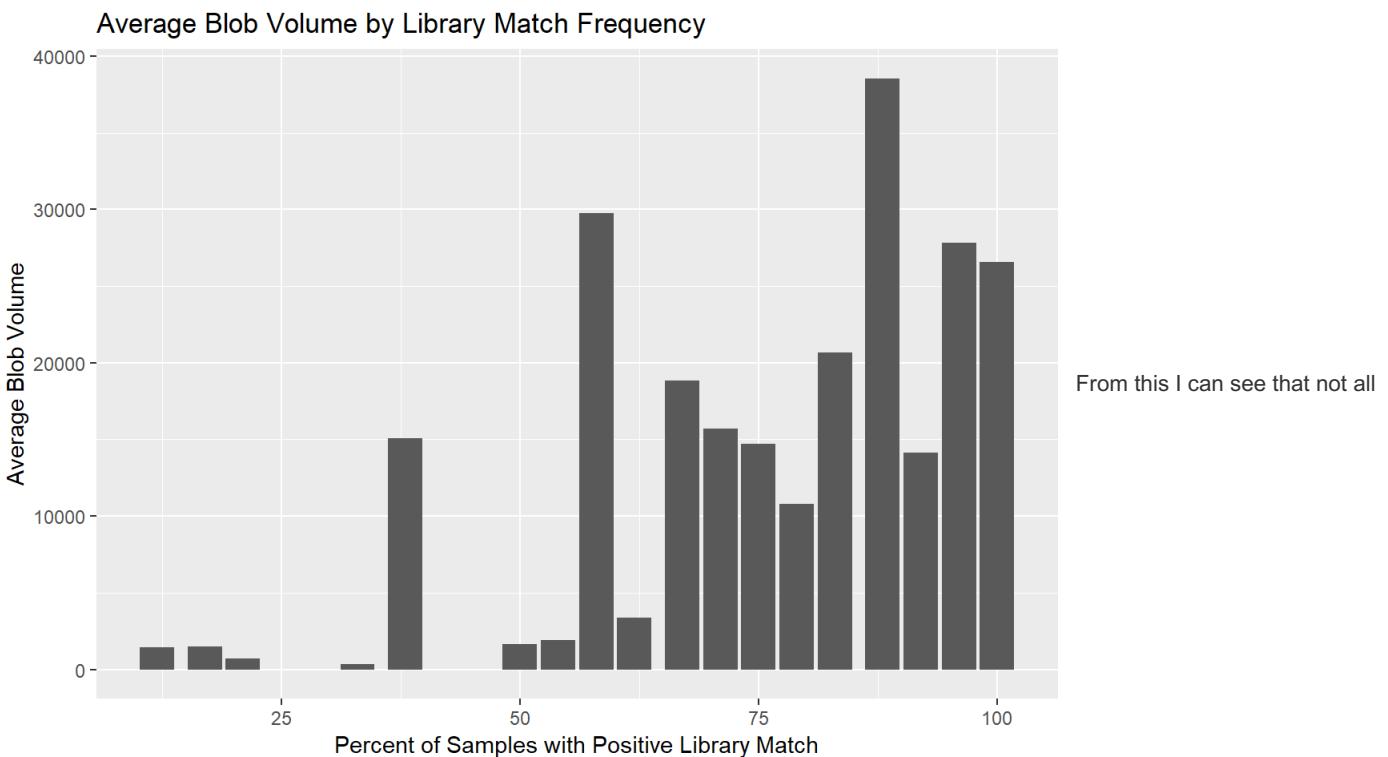
Histogram of 134 Traced Compounds Over 5 Day Test Period



```
M_t_analyte_unique <- M_t_analyte_unique %>%
  left_join(compound_pop)
```

```
## Joining, by = "Compound.Name"
```

```
M_t_analyte_unique %>%
  mutate(char_comp_n = as.integer(round(pct_of_samp, 0))) %>%
  group_by(char_comp_n) %>%
  summarise(avg_vol_byn = mean(Volume)) %>%
  ggplot(aes(x = char_comp_n, y = avg_vol_byn)) +
  geom_col()+
  labs(title = "Average Blob Volume by Library Match Frequency")+
  xlab("Percent of Samples with Positive Library Match")+
  ylab("Average Blob Volume")
```



of the compounds are being traced properly- this may or may not be a problem, and is an area for additional checking.

Internal Standard Mapping

Internal standard assignments in raw blob tables require cleaning before they can be utilized for normalization, similar to the external standard components.

```
# note: unlike sample analytes, internal standard compounds are frequently split vertically into multiple blobs because of the high volume. In order to account for this, I am summing all blobs together that meet the high match criteria and are very close in the first dimension.
IS_unique <- IS_goodmatch %>%
  group_by(Compound.Name, File_num) %>%
  summarise(Volume_tot = sum(Volume))

IS_positions <- IS_goodmatch %>%
  group_by(Compound.Name, File_num) %>%
  summarise(LRI_avg = mean(LRI.I), RI2 = min(Retention.II..sec.-rt2_floor))

IS_unique_pos <- IS_unique %>%
  left_join(IS_positions)

## Joining, by = c("Compound.Name", "File_num")

IS_test1 <- IS_unique_pos %>%
  filter(File_num == "GCxGC_20181201_0136")

IS_1_vol <- IS_test1$Volume_tot
RI1 <- IS_test1$LRI_avg
RI2 <- IS_test1$RI2

# IS_1_pos <- IS_test1 %>%
#   ungroup() %>%
#   dplyr::select("LRI_avg", "RI2")
```

There are a variety of options for normalizing by internal standard volume, and these different options will be explored in greater depth in future. The option utilized in this test is the creation of a generalized additive model which uses the recovery of each internal standard in each of the images to create a model of how compound recovery is affected by the sample in which each compound is measured, the position in terms of retention index (a proxy for volatility), and the position in terms of RI2 (a proxy for polarity). The advantage of this method is that it more easily allows analyte compounds which fall outside of the bounds of the positions of the internal standards to be normalized, and it diminishes the impact of one incorrect assignment of an internal standard species in a single image, as modeled recoveries will be informed by the typical recovery of each compound rather than being informed only by the range of recoveries within a single sample. Below I test a few options for modeling internal standard recovery using a GAM.

```
# trying IS by GAM
library(gam)

## Loading required package: splines

## Loading required package: foreach

## 
## Attaching package: 'foreach'

## The following objects are masked from 'package:purrr':
## 
##     accumulate, when

## Loaded gam 1.16

library(mgcv)

## Loading required package: nlme

## 
## Attaching package: 'nlme'

## The following object is masked from 'package:glue':
## 
##     collapse

## The following object is masked from 'package:dplyr':
## 
##     collapse

## This is mgcv 1.8-26. For overview type 'help("mgcv-package")'.

## 
## Attaching package: 'mgcv'

## The following objects are masked from 'package:gam':
## 
##     gam, gam.control, gam.fit, s

## The following object is masked from 'package:nnet':
## 
##     multinom
```

```

# using a generalized additive model to identify what internal standard recovery would be expected at each point in the GCxGC space and for each file
IS_unique_pos_f <- IS_unique_pos %>%
  mutate(F_num_f = as.factor(File_num))

y <- IS_unique_pos_f$Volume_tot
Fac <- IS_unique_pos_f$F_num_f
x <- IS_unique_pos_f$LRI_avg
z <- IS_unique_pos_f$RI2

# IS_gam <- gam(IS_unique_pos_f$Volume_tot ~ as.factor(IS_unique_pos_f$F_num_f) + s(IS_unique_pos_f$LRI_avg) +
# s(IS_unique_pos_f$RI2))
#
# summary(IS_gam)
# plot(IS_gam, residuals = TRUE)

IS_gam2 <- gam(y ~ Fac + s(x)+s(z), family = quasi())
summary(IS_gam2)

```

```

## 
## Family: quasi
## Link function: identity
##
## Formula:
## y ~ Fac + s(x) + s(z)
##
## Parametric coefficients:
##                               Estimate Std. Error t value Pr(>|t|)    
## (Intercept)            21362.1    2670.8   7.998 7.71e-15 ***
## FacGCxGC_20181127_1834 -2665.1     3737.6  -0.713  0.47612  
## FacGCxGC_20181127_2026  2372.2     3905.8   0.607  0.54387  
## FacGCxGC_20181127_2218 2162.5     4009.0   0.539  0.58983  
## FacGCxGC_20181128_0011 -2333.3     3742.4  -0.623  0.53324  
## FacGCxGC_20181201_1254 1063.6     3905.3   0.272  0.78546  
## FacGCxGC_20181201_1719 -2285.8     3956.3  -0.578  0.56366  
## FacGCxGC_20181201_1910 -10175.1    3904.6  -2.606  0.00941 ** 
## FacGCxGC_20181201_2104 -6293.2     3905.6  -1.611  0.10769  
## FacGCxGC_20181204_1626 -4520.6     3774.4  -1.198  0.23156  
## FacGCxGC_20181204_1819 -8348.3     3736.6  -2.234  0.02588 *  
## FacGCxGC_20181204_2010 -3455.8     3772.5  -0.916  0.36004  
## FacGCxGC_20181204_2201 -6582.6     3775.9  -1.743  0.08185 . 
## FacGCxGC_20181204_2352 -5411.5     3772.7  -1.434  0.15205  
## FacGCxGC_20181210_1223 1400.7      3814.4  0.367  0.71360  
## FacGCxGC_20181210_1606 -494.3      3852.5  -0.128  0.89795  
## FacGCxGC_20181210_1756 -1774.7      3814.4  -0.465  0.64194  
## FacGCxGC_20181210_1946 -1903.3      3772.6  -0.505  0.61410  
## FacGCxGC_20181210_2137 -656.7      3780.0  -0.174  0.86215  
## FacGCxGC_20181218_2025 -4335.0      3809.3  -1.138  0.25562  
## FacGCxGC_20181219_0008 -1719.3      3821.1  -0.450  0.65292  
## FacGCxGC_20181219_0200 -6438.5      3758.8  -1.713  0.08730 . 
## FacGCxGC_20181219_0351 -4261.9      3797.6  -1.122  0.26224  
## FacGCxGC_20181219_0543 -7933.7      3826.2  -2.074  0.03860 * 
## --- 
## Signif. codes:  0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##          edf Ref.df    F p-value    
## s(x)  6.469  7.546 26.92 <2e-16 ***
## s(z)  7.998  8.684 47.72 <2e-16 ***
## --- 
## Signif. codes:  0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) = 0.542  Deviance explained = 57.2%
## GCV = 1.9052e+08  Scale est. = 1.7787e+08 n = 579

```

```

#IS_gam3 <- gam(y ~ Fac + s(x)+s(z), family = poisson())
#summary(IS_gam3)

#vis.gam(IS_gam3)

# testing gam on a single file
test_gam_1 <- IS_unique_pos_f %>%
  filter(File_num == "GCxGC_20181201_0136") %>%
  rename(Volume_tot_inst = Volume_tot) %>%
  mutate(F_num_f = as.factor(F_num_f))

#test_gam_1a <- data.frame(Fac = test_gam_1$F_num_f, x=test_gam_1$LRI_avg, z=test_gam_1$RI2)

#pred_gam <- predict.gam(IS_gam2, data.frame(Fac = as.factor("GCxGC_20181201_0136"), x = test_gam_1a$test_gam_1$LRI_avg, z = test_gam_1a$test_gam_1.RI2))

# visualizing IS_gam2 and how it would predict internal standard volumes in one fil
test_gam_1b <- test_gam_1 %>%
  mutate(Volume_tot_model = predict.gam(IS_gam2, data.frame(Fac = as.factor(F_num_f), x = LRI_avg, z = RI2)))
)

test_gam_1b %>%
  ggplot(aes(x = LRI_avg, y = RI2, color = Volume_tot_inst)) +
  geom_point()

```

RI2

LRI_avg

```

test_gam_1b %>%
  ggplot(aes(x = LRI_avg, y = RI2, color = Volume_tot_model)) +
  geom_point()

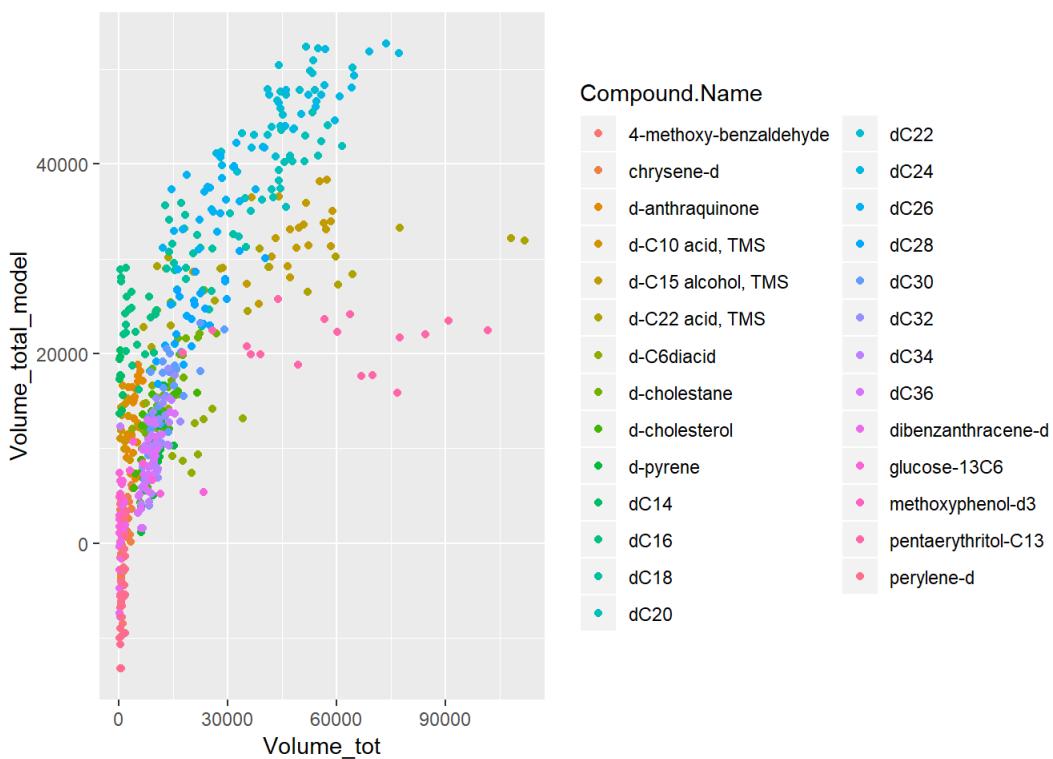
```

RI2

LRI_avg

```
test_gam_all_b <- IS_unique_pos_f %>%
  mutate(Volume_total_model = predict.gam(IS_gam2, data.frame(Fac = as.factor(F_num_f), x = LRI_avg, z = RI2)))

### note that the output of the following plot indicates that the IS_gam2 is not appropriate for my data; it is impossible for signal to be less than zero, meaning that the low concentration compounds are not being modeled well. a solution for this is to try a log transformation to improve modeling of low concentration internal standards
test_gam_all_b %>%
  ggplot(aes(x = Volume_tot, y = Volume_total_model, color = Compound.Name)) +
  geom_point()
```



```

##### gam with a log transform of volumes
#y <- log(IS_unique_pos_f$Volume_tot)
y <- IS_unique_pos_f$Volume_tot
Fac <- IS_unique_pos_f$F_num_f
x <- IS_unique_pos_f$LRI_avg
z <- IS_unique_pos_f$RI2

#IS_gam2b <- gam(y ~ Fac + s(x)+s(z), family = gaussian())
IS_gam2b <- gam(y ~ Fac + s(x)+s(z), family = poisson())
summary(IS_gam2b)

```

```

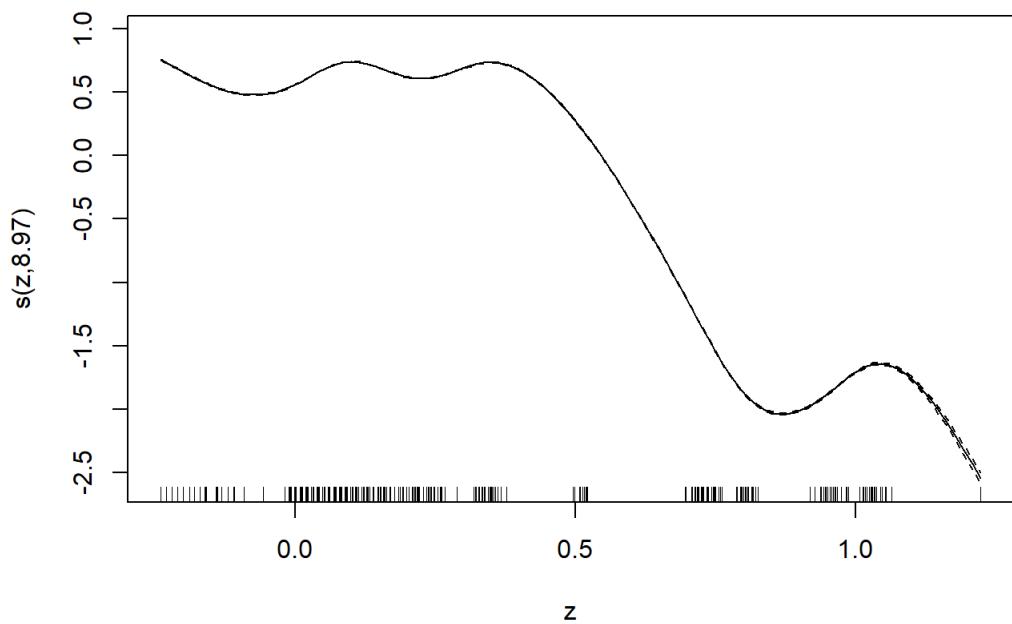
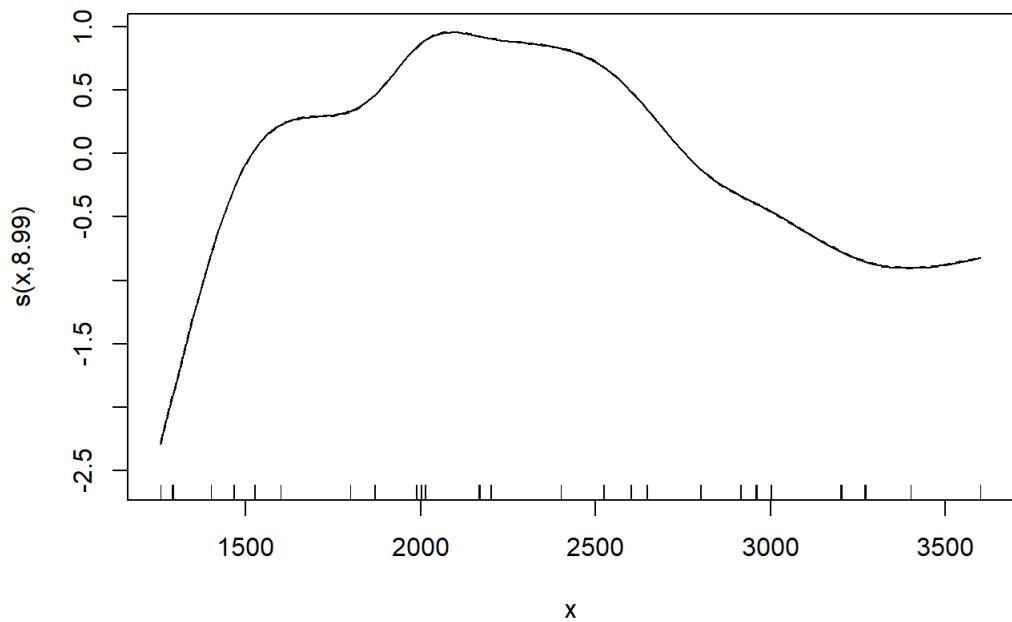
##
## Family: poisson
## Link function: log
##
## Formula:
## y ~ Fac + s(x) + s(z)
##
## Parametric coefficients:
##                               Estimate Std. Error z value Pr(>|z|)
## (Intercept)             9.436116  0.001482 6366.157 < 2e-16 ***
## FacGCxGC_20181127_1834 -0.105037  0.001992 -52.727 < 2e-16 ***
## FacGCxGC_20181127_2026  0.117951  0.001956  60.316 < 2e-16 ***
## FacGCxGC_20181127_2218  0.114877  0.001996  57.559 < 2e-16 ***
## FacGCxGC_20181128_0011 -0.071230  0.001995 -35.707 < 2e-16 ***
## FacGCxGC_20181201_1254  0.063683  0.001984  32.095 < 2e-16 ***
## FacGCxGC_20181201_1719 -0.107063  0.002081 -51.458 < 2e-16 ***
## FacGCxGC_20181201_1910 -0.633357  0.002437 -259.921 < 2e-16 ***
## FacGCxGC_20181201_2104 -0.329350  0.002215 -148.660 < 2e-16 ***
## FacGCxGC_20181204_1626 -0.222698  0.002089 -106.615 < 2e-16 ***
## FacGCxGC_20181204_1819 -0.475316  0.002208 -215.314 < 2e-16 ***
## FacGCxGC_20181204_2010 -0.185066  0.002040 -90.724 < 2e-16 ***
## FacGCxGC_20181204_2201 -0.375739  0.002164 -173.652 < 2e-16 ***
## FacGCxGC_20181204_2352 -0.314836  0.002115 -148.857 < 2e-16 ***
## FacGCxGC_20181210_1223  0.083300  0.001924  43.284 < 2e-16 ***
## FacGCxGC_20181210_1606  0.016637  0.001991  8.357 < 2e-16 ***
## FacGCxGC_20181210_1756 -0.108898  0.002000 -54.452 < 2e-16 ***
## FacGCxGC_20181210_1946 -0.103127  0.001997 -51.648 < 2e-16 ***
## FacGCxGC_20181210_2137 -0.006781  0.001969 -3.444 0.000574 ***
## FacGCxGC_20181218_2025 -0.230223  0.002133 -107.952 < 2e-16 ***
## FacGCxGC_20181219_0008 -0.056442  0.002031 -27.791 < 2e-16 ***
## FacGCxGC_20181219_0200 -0.339413  0.002145 -158.242 < 2e-16 ***
## FacGCxGC_20181219_0351 -0.204332  0.002095 -97.514 < 2e-16 ***
## FacGCxGC_20181219_0543 -0.431959  0.002238 -192.982 < 2e-16 ***
## ---
## Signif. codes: 0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##          edf Ref.df Chi.sq p-value
## s(x)    8.994    9 1709614 <2e-16 ***
## s(z)    8.973    9 2362246 <2e-16 ***
## ---
## Signif. codes: 0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) = 0.58 Deviance explained = 68.9%
## UBRE = 5897.4 Scale est. = 1 n = 579

```

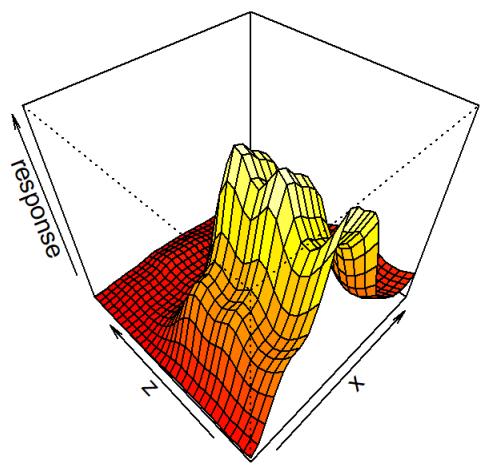
```

plot(IS_gam2b) # first graph illustrating how recovery varies with volatility, second graph illustrating how
recovery varies with polarity. Moral of the story is that recovery is best around the middle and drops off in
all directions

```



```
# visualizing the IS response surface in GCxGC space implied by the internal standards from the 9 test filters
vis.gam(IS_gam2b, view = c("x", "z"), theta = -45, phi = 45, type = "response")
```



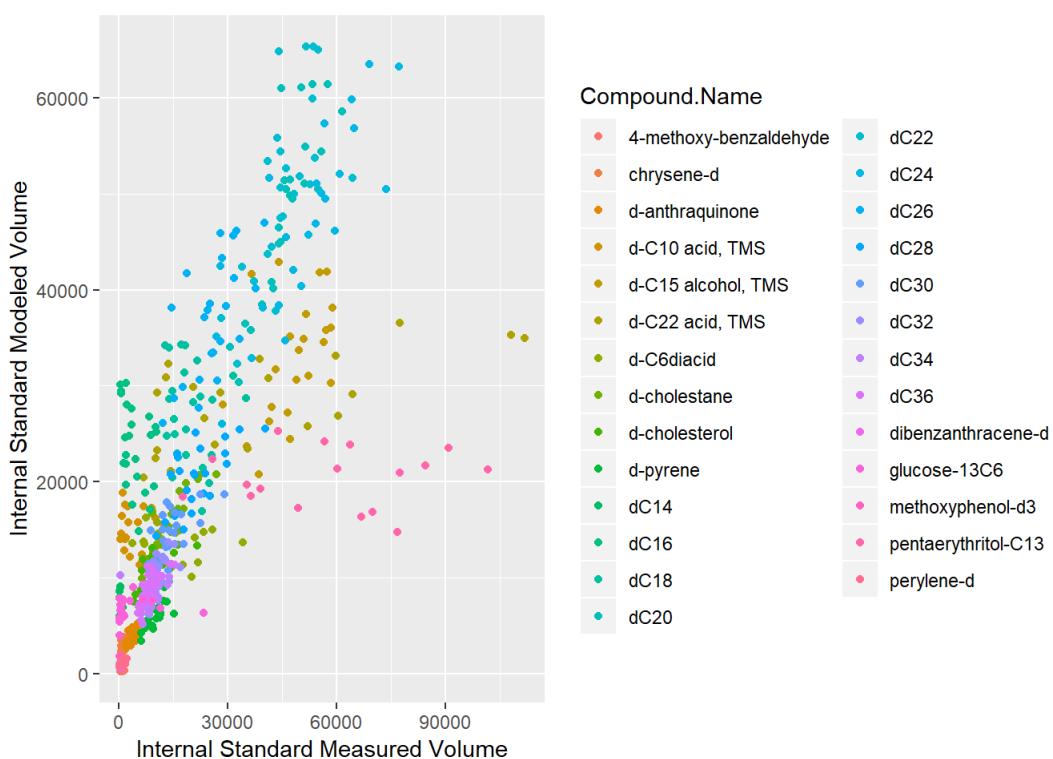
```

test_gam_all <- IS_unique_pos_f %>%
  mutate(Vol_inst_log = log(Volume_tot)) %>%
  #mutate(Volume_total_model_log = predict.gam(IS_gam2b, data.frame(Fac = as.factor(F_num_f), x = LRI_avg,
z = RI2))) %>%
  #mutate(Volume_total_model = exp(Volume_total_model_log))
  mutate(Volume_total_IS_model = predict.gam(IS_gam2b, data.frame(Fac = as.factor(F_num_f), x = LRI_avg, z
= RI2), type = "response"))

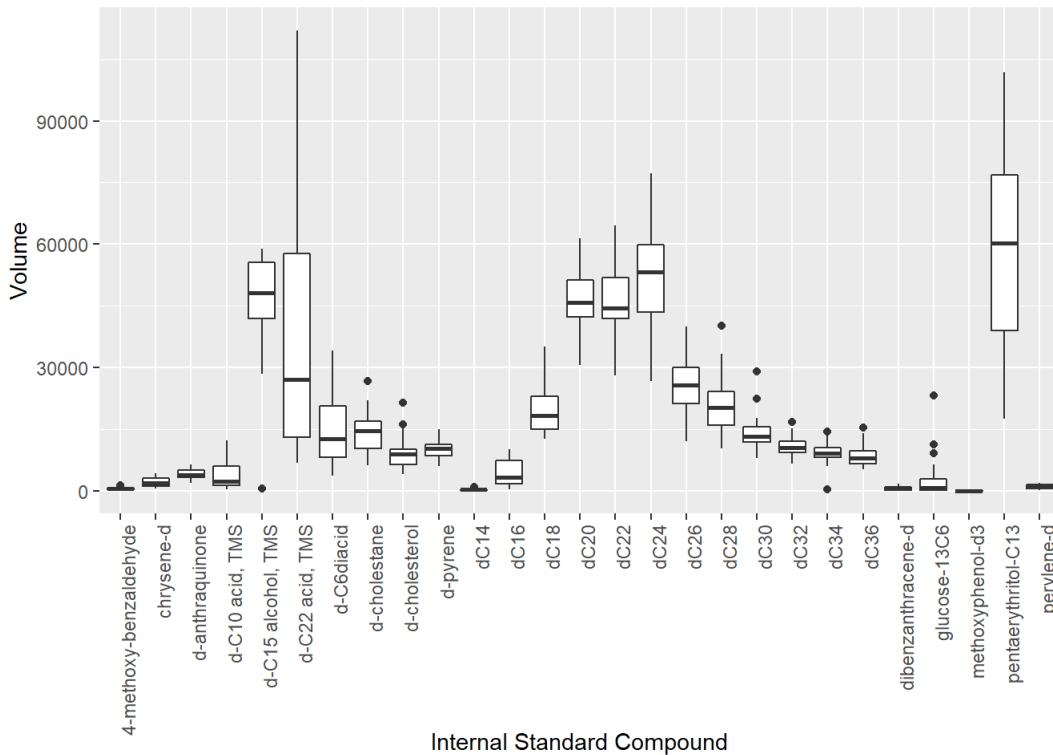
#test_gam_all %>%
#  ggplot(aes(x = Vol_inst_log, y = Volume_total_model_log, color = Compound.Name)) +
#  geom_point()

test_gam_all %>%
  ggplot(aes(x = Volume_tot, y = Volume_total_IS_model, color = Compound.Name)) +
  geom_point() +
  xlab("Internal Standard Measured Volume") +
  ylab("Internal Standard Modeled Volume")

```



```
test_gam_all %>%
  ggplot(aes(x = Compound.Name, y = Volume_tot)) +
  geom_boxplot() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  ylab("Volume") +
  xlab("Internal Standard Compound")
```



```
## add a column to the master list of unique analytes which is the internal standard normalized volume at each point in time
M_t_analyte_unique_IS <- M_t_analyte_unique %>%
  mutate(IS_glm = exp(predict.gam(IS_gam2b, data.frame(Fac = as.factor(File_num), x = LRI.I, z = Retention.II..sec.))) %>%
  mutate(Vol_norm_GLM = Volume/IS_glm)
```

Note- moving forward, before doing the correction between ES and sample, must train a GAM on everything- both the sample and the ES (cant do just one)

Loading External Standard

Next the External standard calibration point information must be added into the processed blob files and reformatted for future modeling. Additionally, cal curves must be plotted to identify problematic points.

```
cal_pts <- read_csv("Cal_Curve_points_ES.csv")
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   `Compound Name` = col_character(),
##   Description = col_character(),
##   `Group Name` = col_character(),
##   `Constellation Name` = col_logical(),
##   Inclusion = col_logical(),
##   `Library Name` = col_character(),
##   `Review Status` = col_logical(),
##   Quantifier_1 = col_logical(),
##   Quantifier_2 = col_logical(),
##   H_Cat = col_character()
## )
```

```
## See spec(...) for full column specifications.
```

```

# tidying column names
names(cal_pts) <- make.names(names(cal_pts), unique = TRUE)

cal_pts <- cal_pts %>%
  dplyr::select(-Description)

cal_pts_long <- cal_pts %>%
  gather(key = "Cal_Point", value = "Cal_amt_ng",
         Cal_pt_1, Cal_pt_2, Cal_pt_3, Cal_pt_4, Cal_pt_5, Cal_pt_6) %>%
  dplyr::select(Compound.Name, Cal_Point, Cal_amt_ng)

n_c_full <- M_t_analyte_unique_IS %>%
  left_join(cal_pts_long)

## Joining, by = c("Compound.Name", "Cal_Point")

for(i in 1:nrow(cal_pts)){
  bid_temp <- cal_pts$Compound.Name[i]

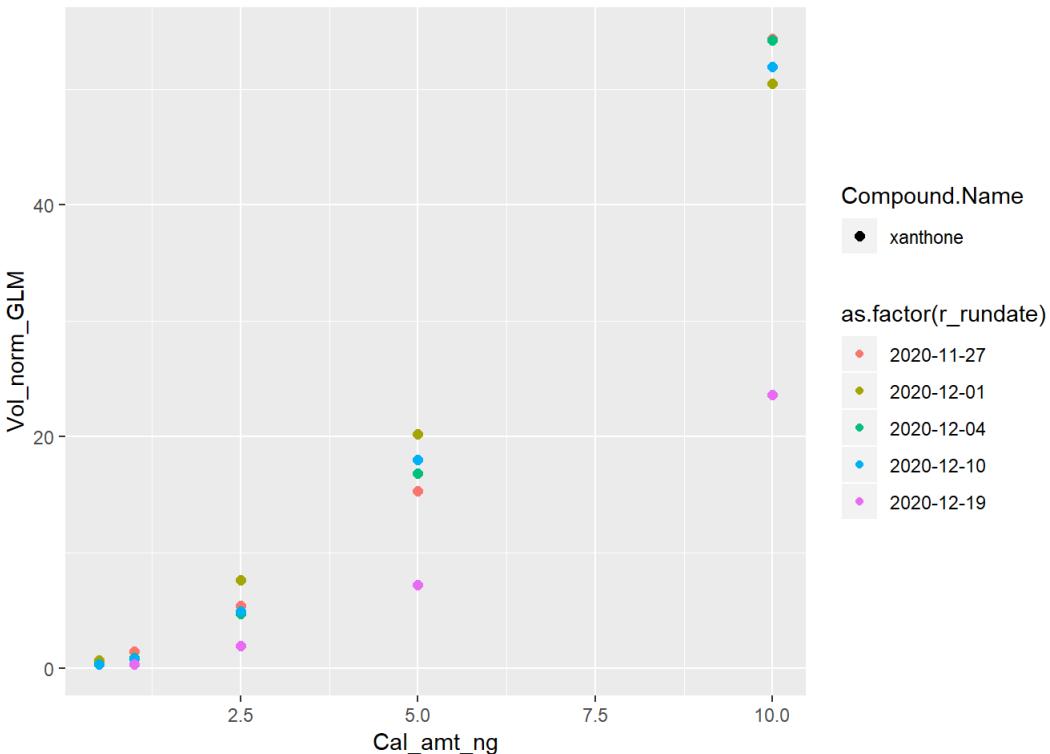
  n_c_full_sub <- n_c_full %>%
    filter(Compound.Name == bid_temp)

  p <- n_c_full_sub %>%
    ggplot(aes(x = Cal_amt_ng, y = Vol_norm_GLM, color = as.factor(r_rundate), size = Compound.Name)) +
    geom_point()

  print(p)
}

## Warning: Using size for a discrete variable is not advised.

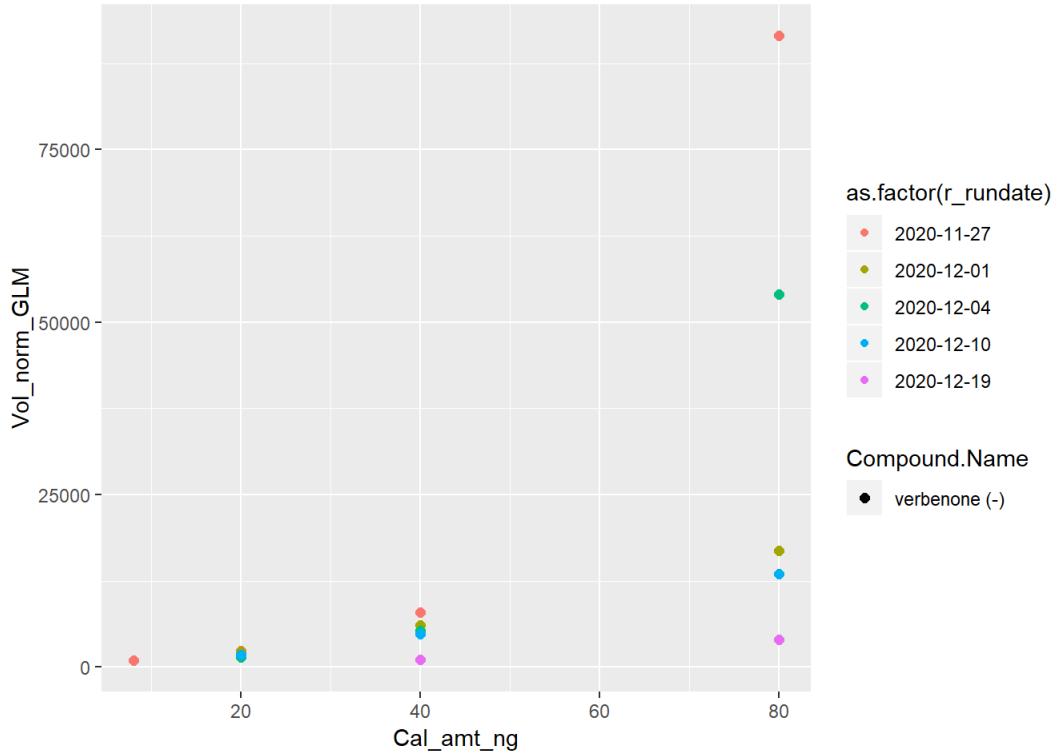
```



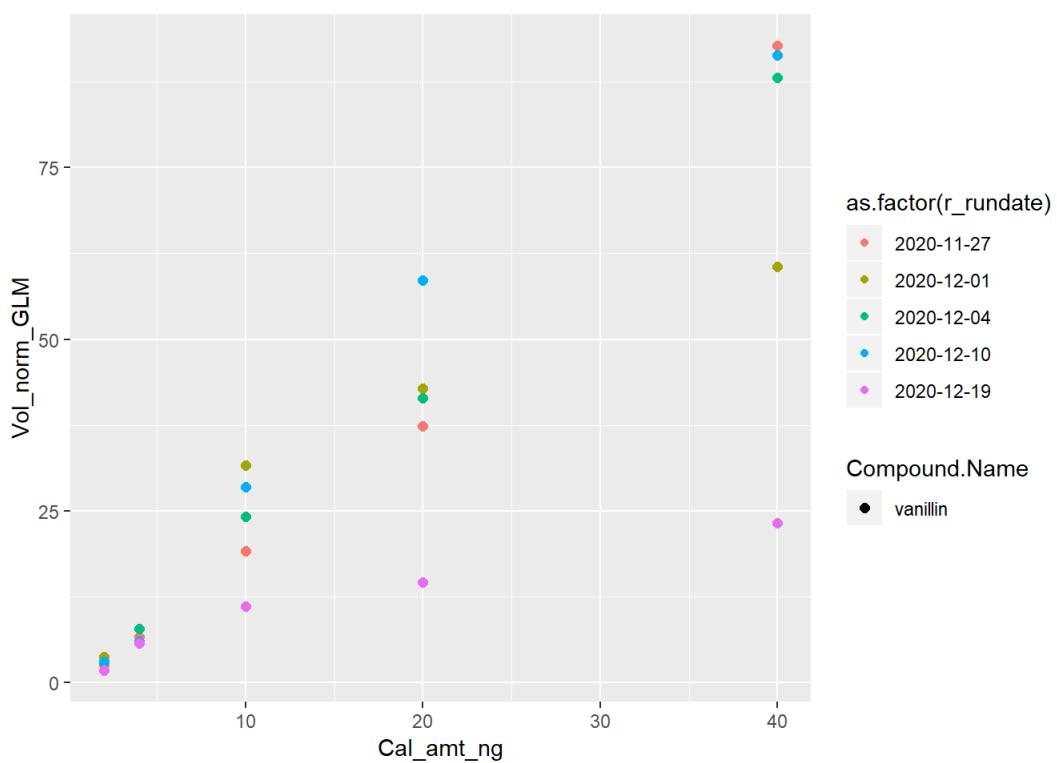
```

## Warning: Using size for a discrete variable is not advised.

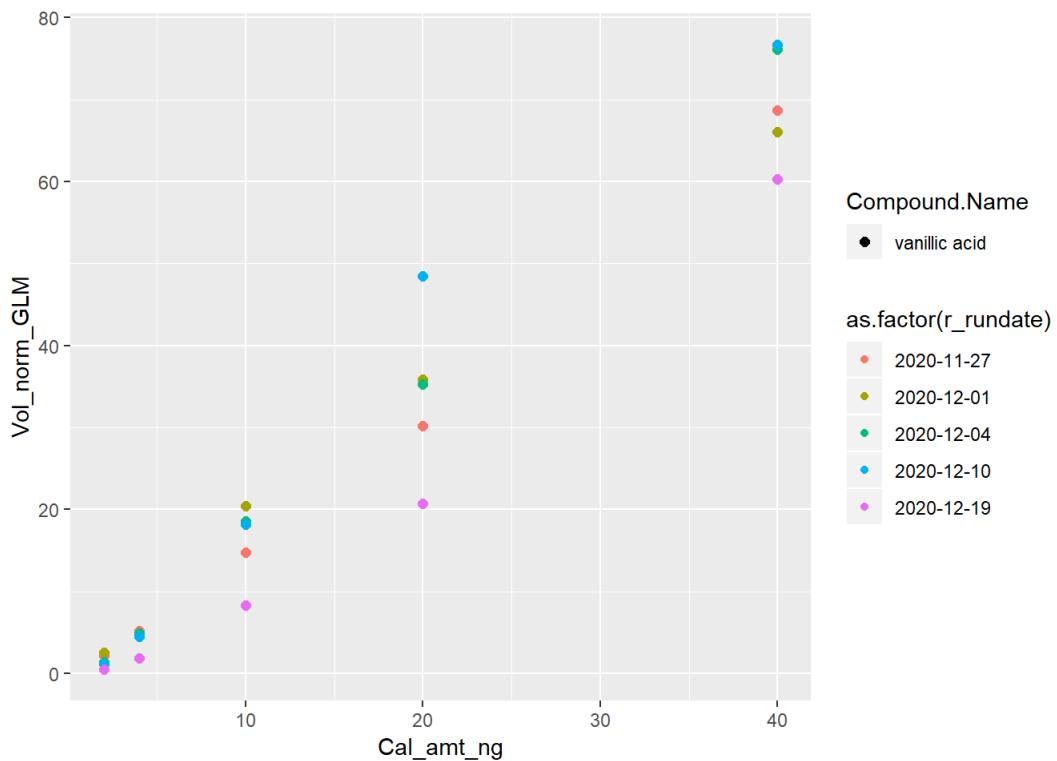
```



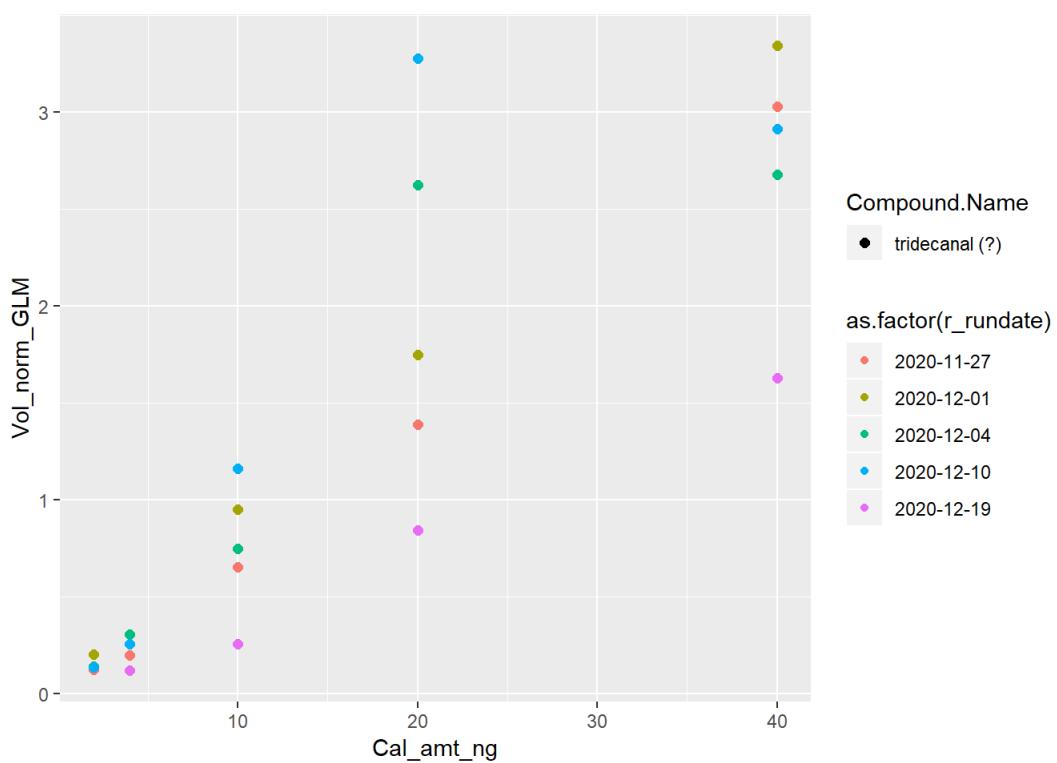
```
## Warning: Using size for a discrete variable is not advised.
```



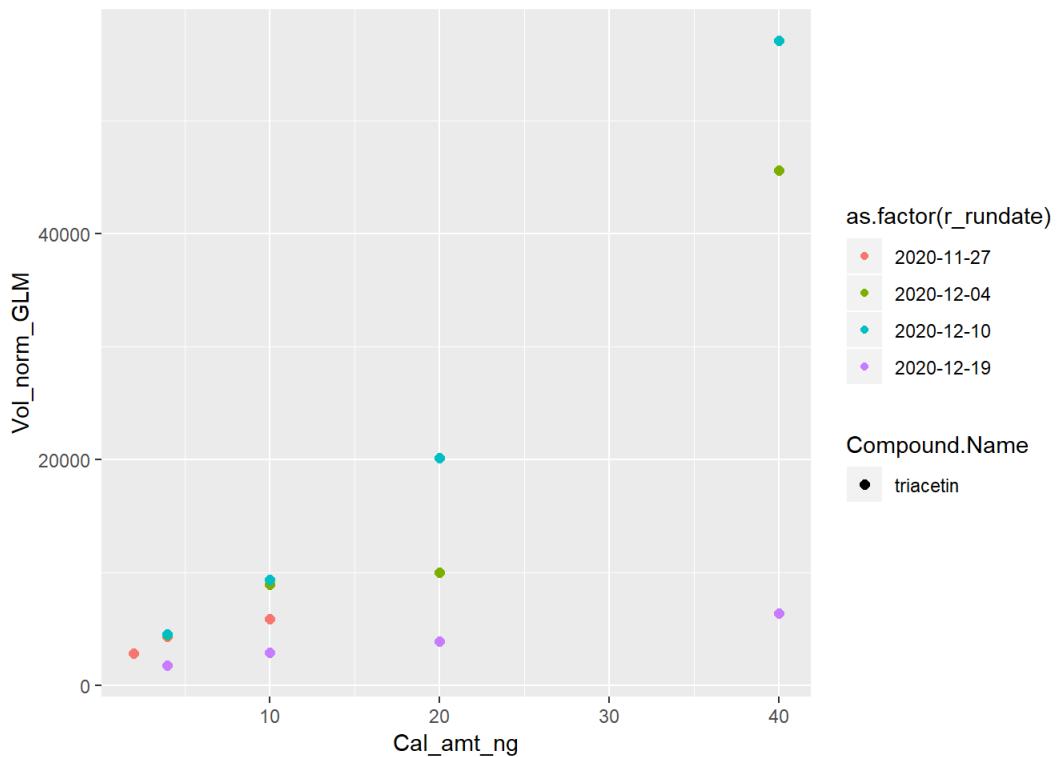
```
## Warning: Using size for a discrete variable is not advised.
```



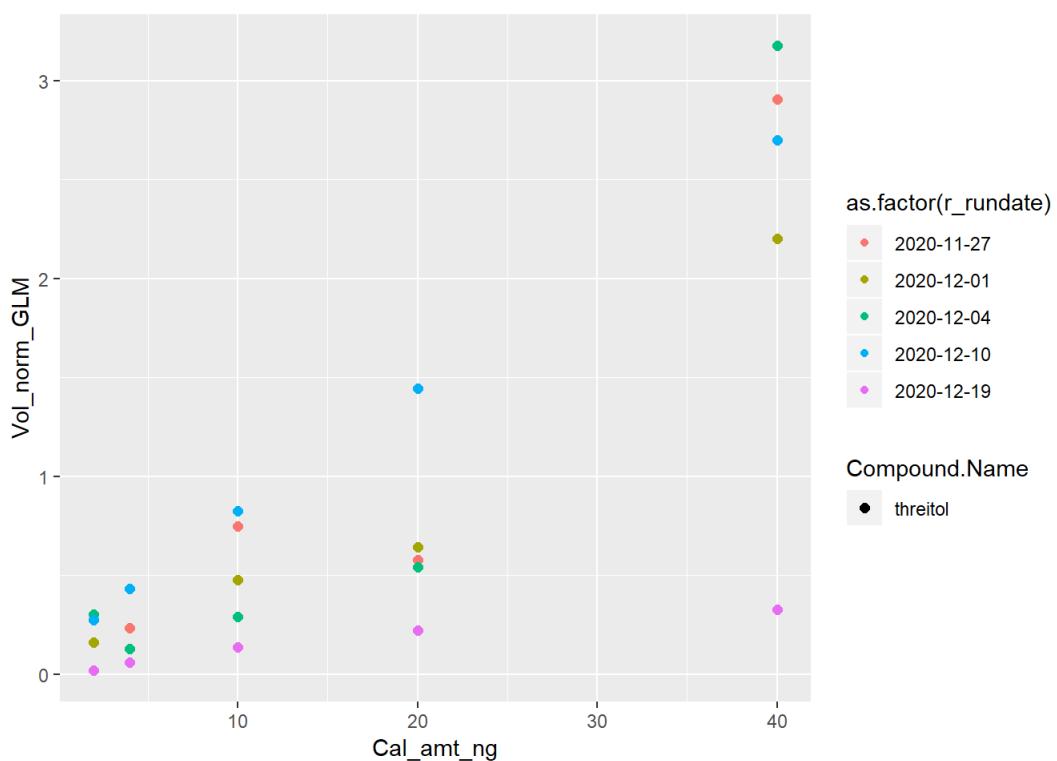
```
## Warning: Using size for a discrete variable is not advised.
```



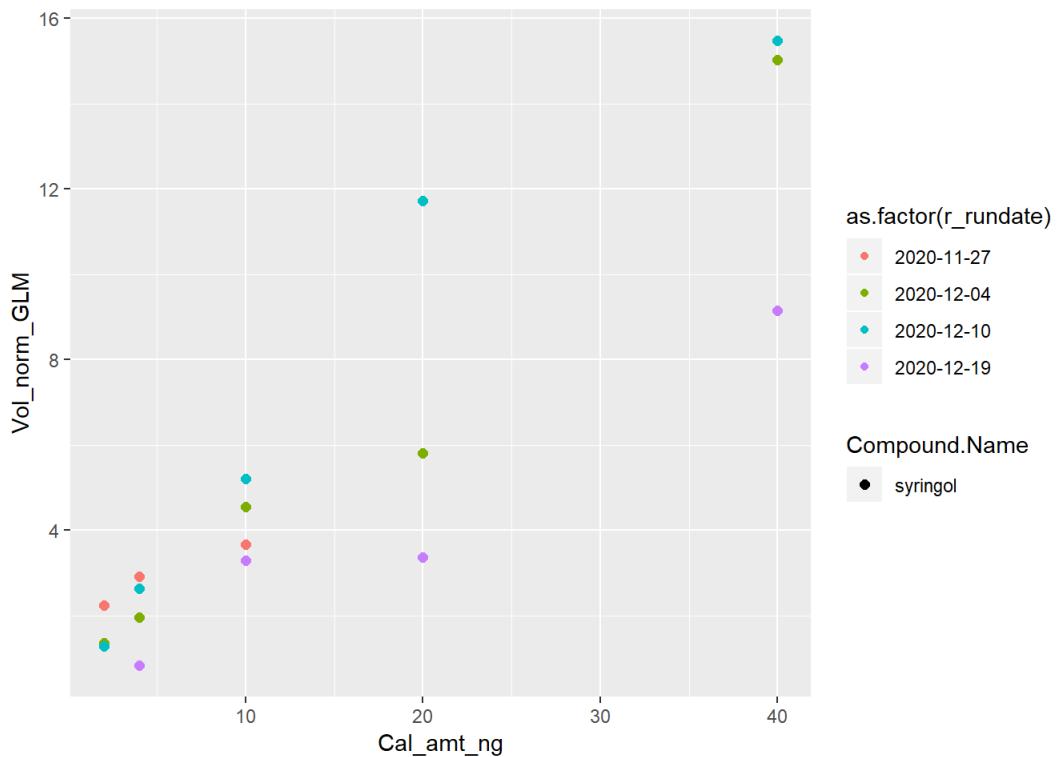
```
## Warning: Using size for a discrete variable is not advised.
```



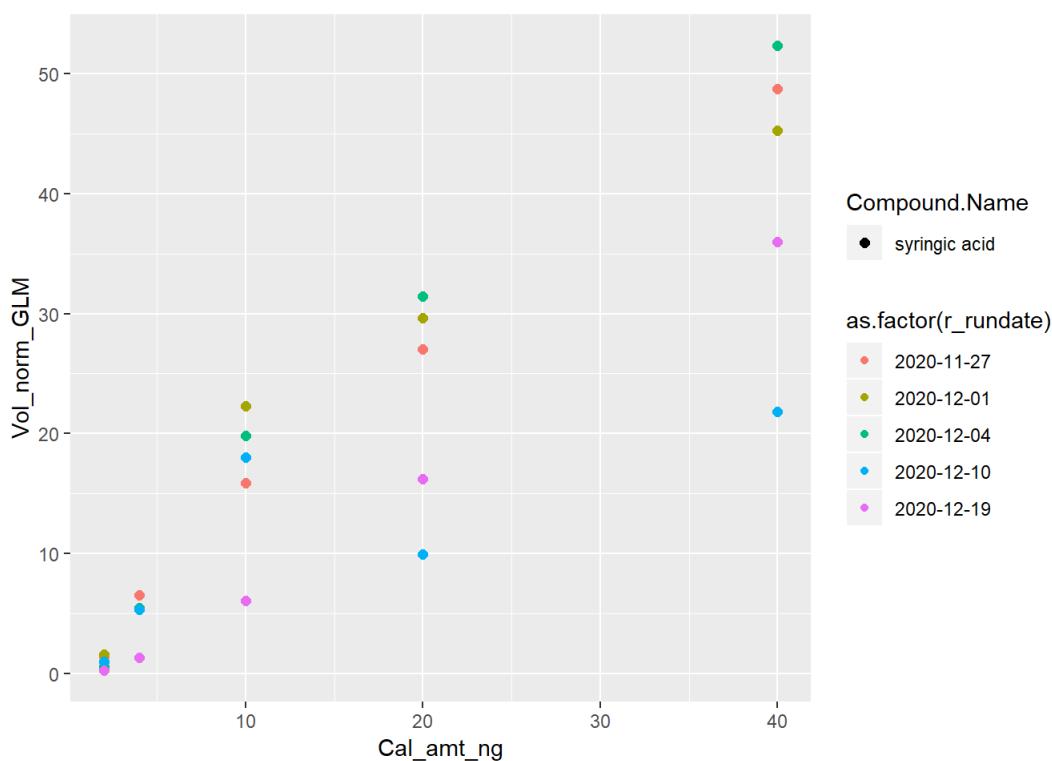
```
## Warning: Using size for a discrete variable is not advised.
```



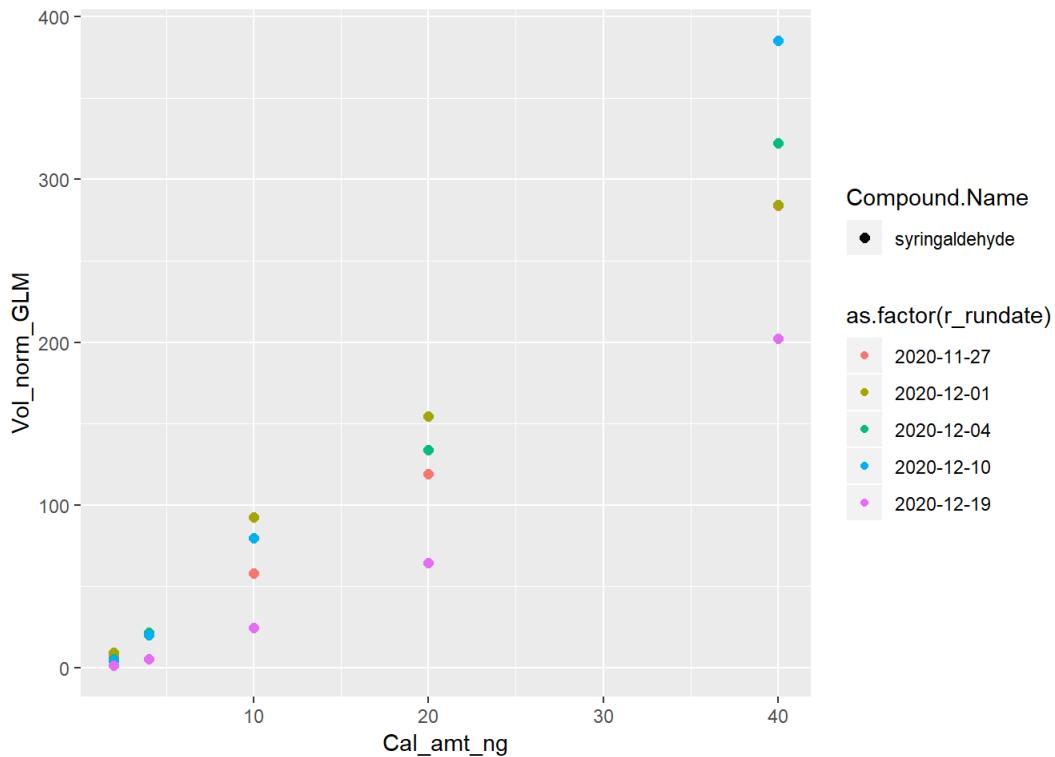
```
## Warning: Using size for a discrete variable is not advised.
```



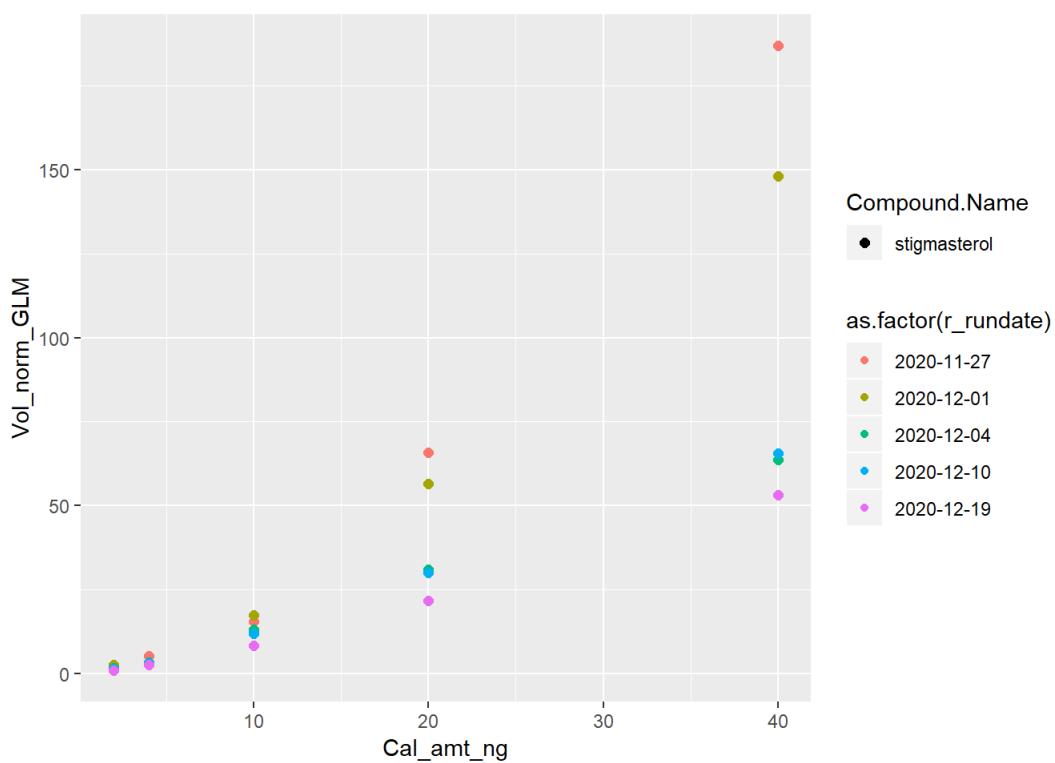
```
## Warning: Using size for a discrete variable is not advised.
```



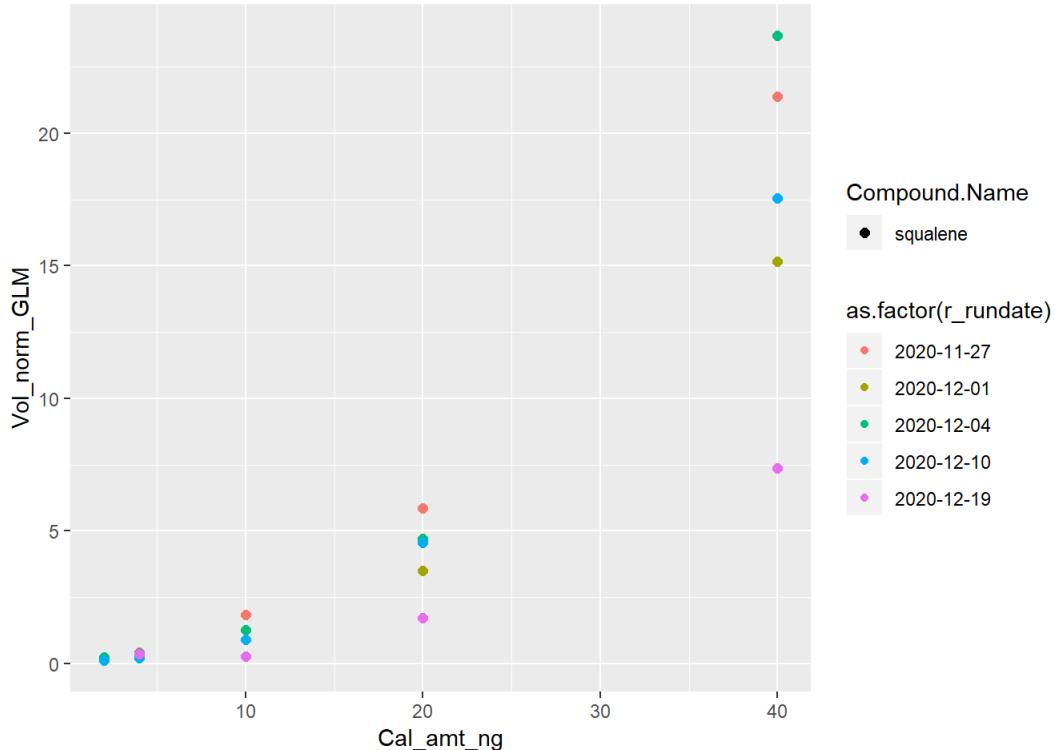
```
## Warning: Using size for a discrete variable is not advised.
```



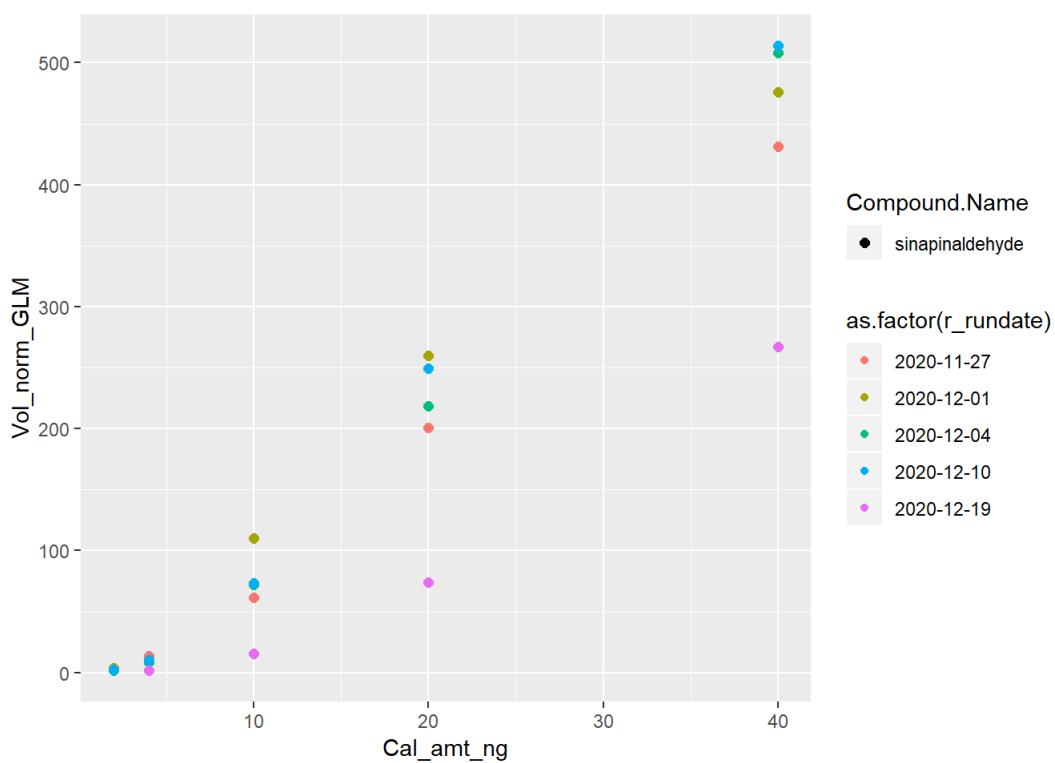
```
## Warning: Using size for a discrete variable is not advised.
```



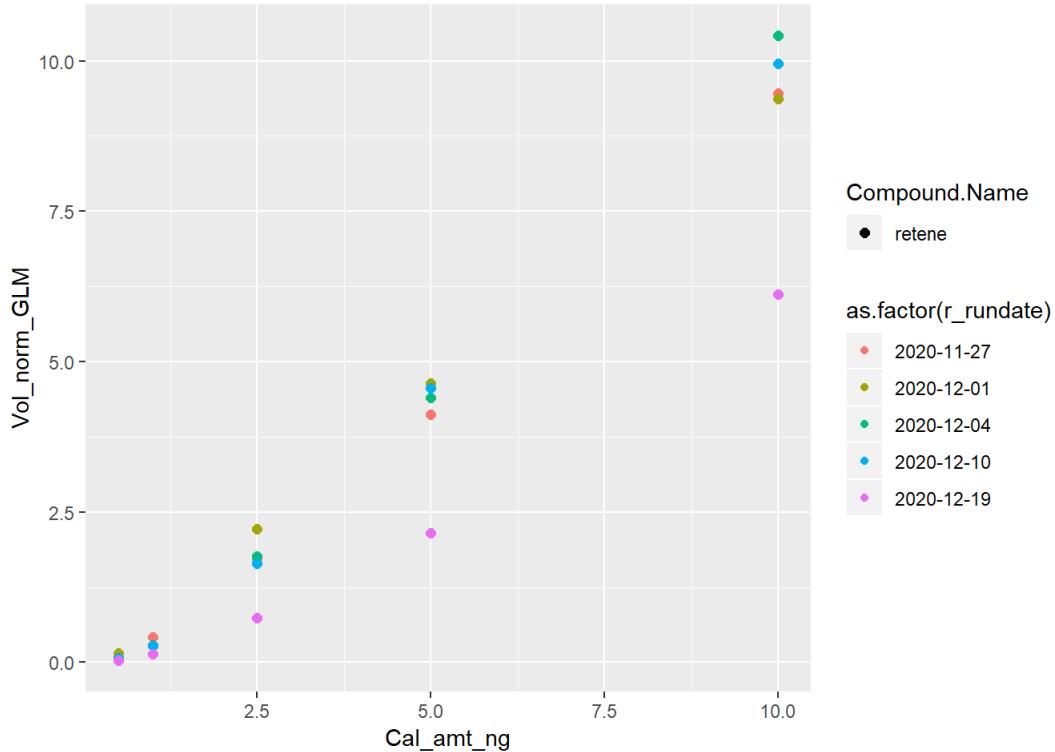
```
## Warning: Using size for a discrete variable is not advised.
```



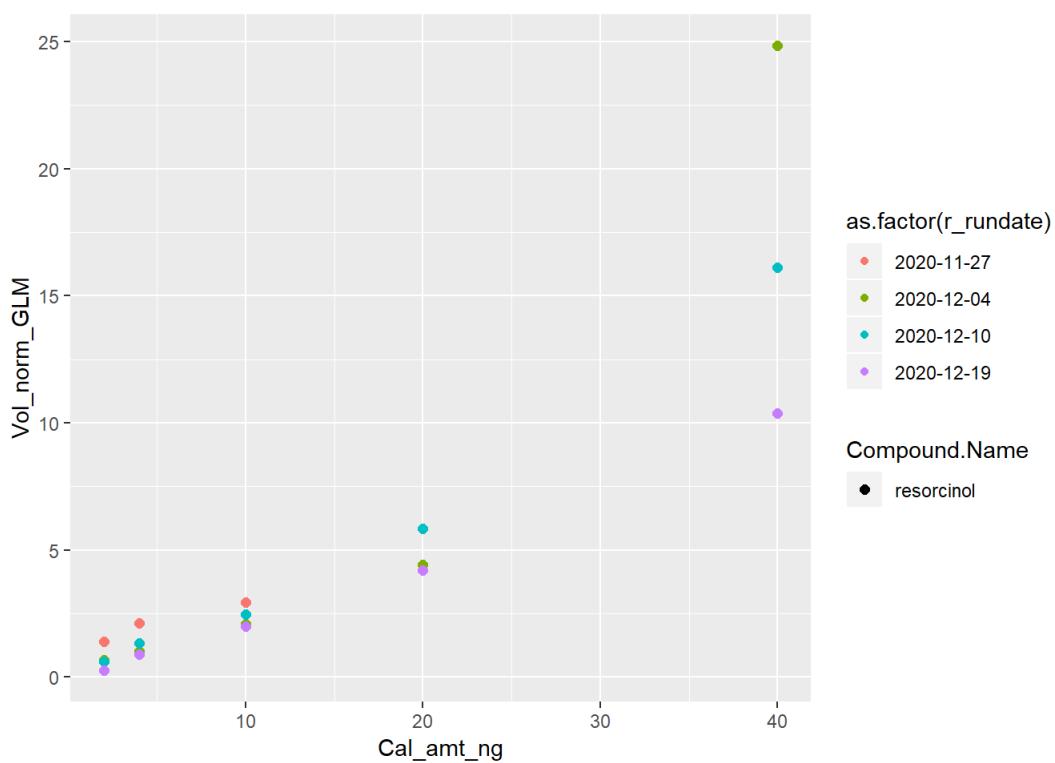
```
## Warning: Using size for a discrete variable is not advised.
```



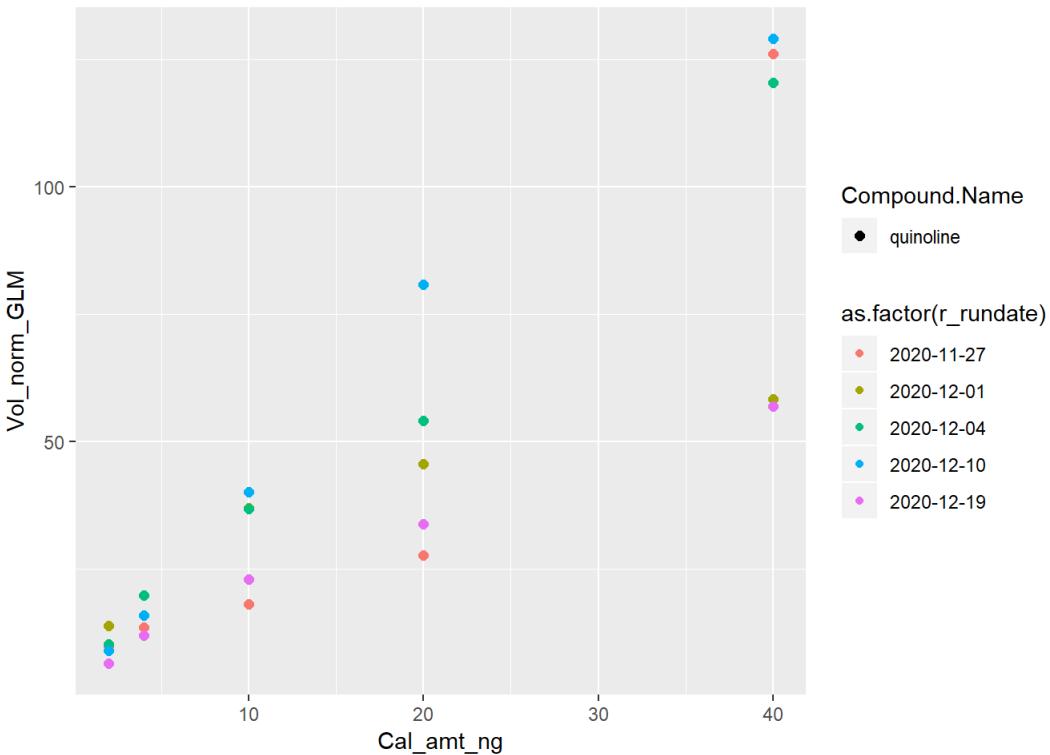
```
## Warning: Using size for a discrete variable is not advised.
```



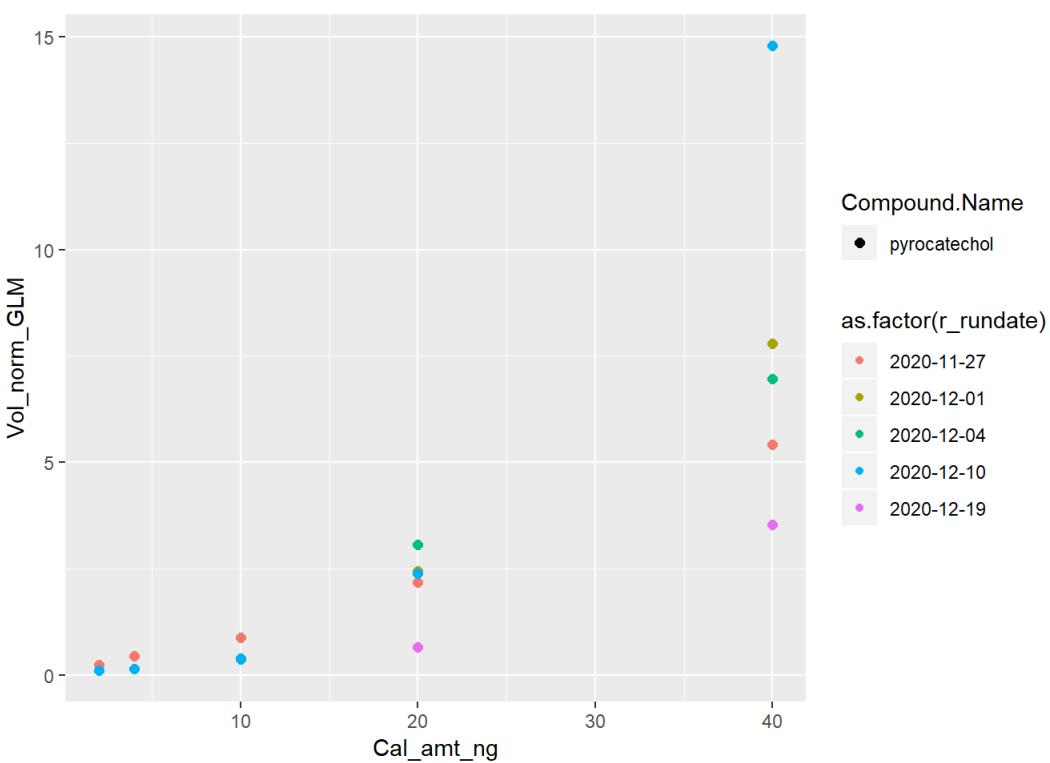
```
## Warning: Using size for a discrete variable is not advised.
```



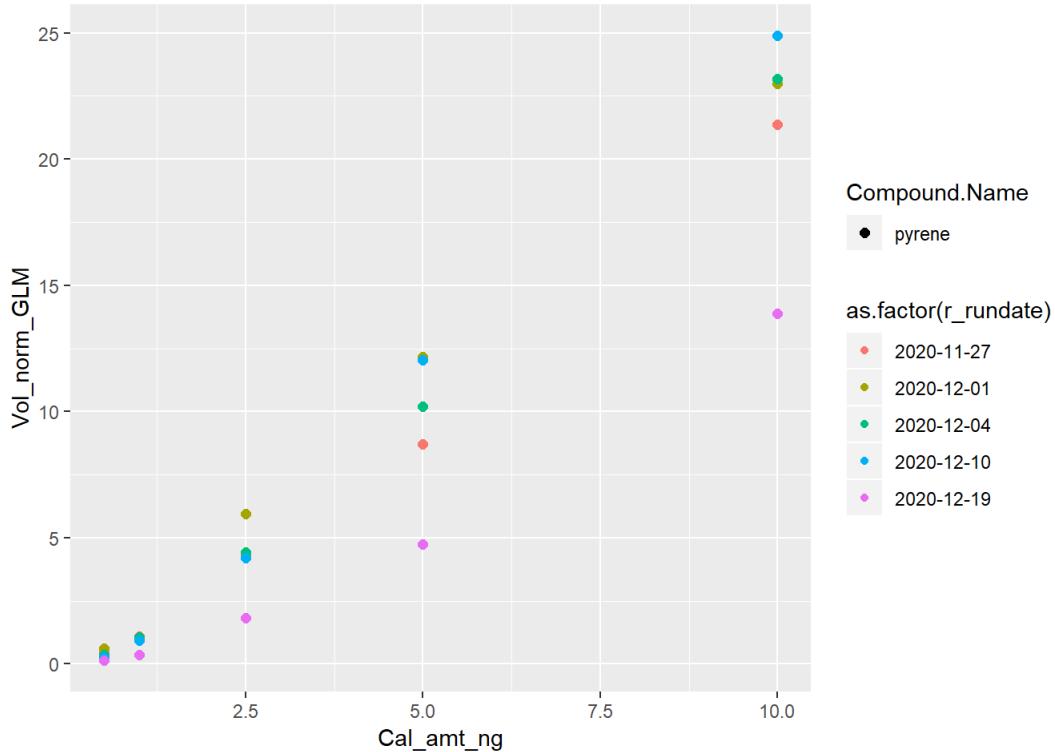
```
## Warning: Using size for a discrete variable is not advised.
```



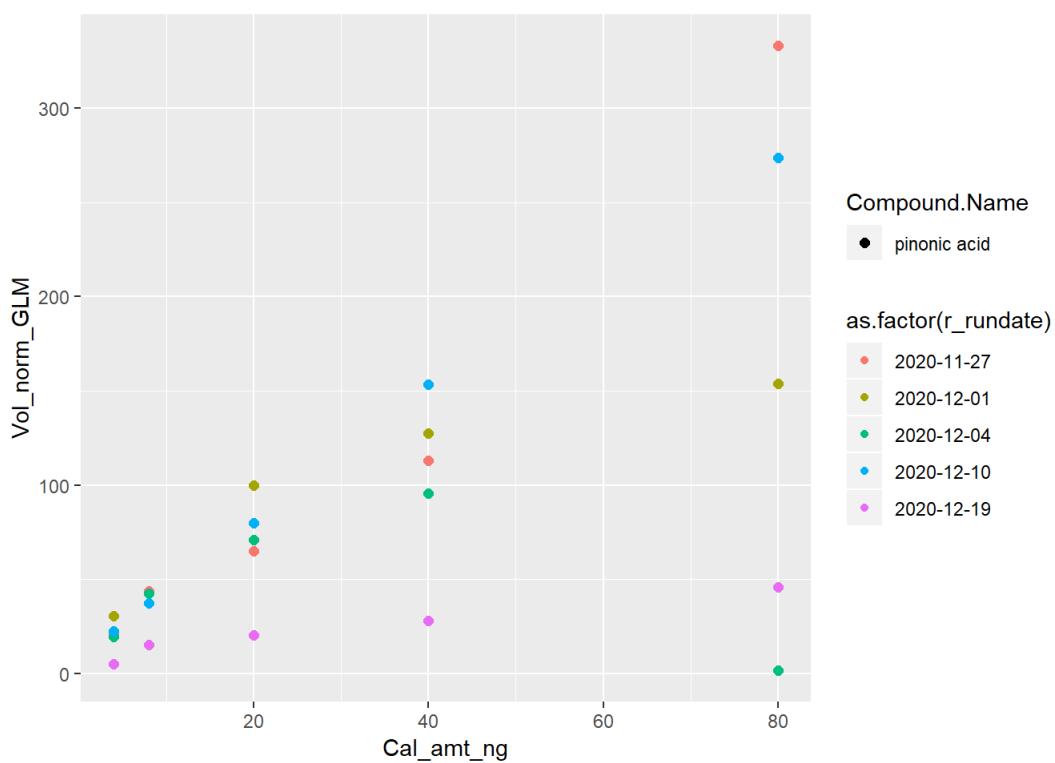
```
## Warning: Using size for a discrete variable is not advised.
```



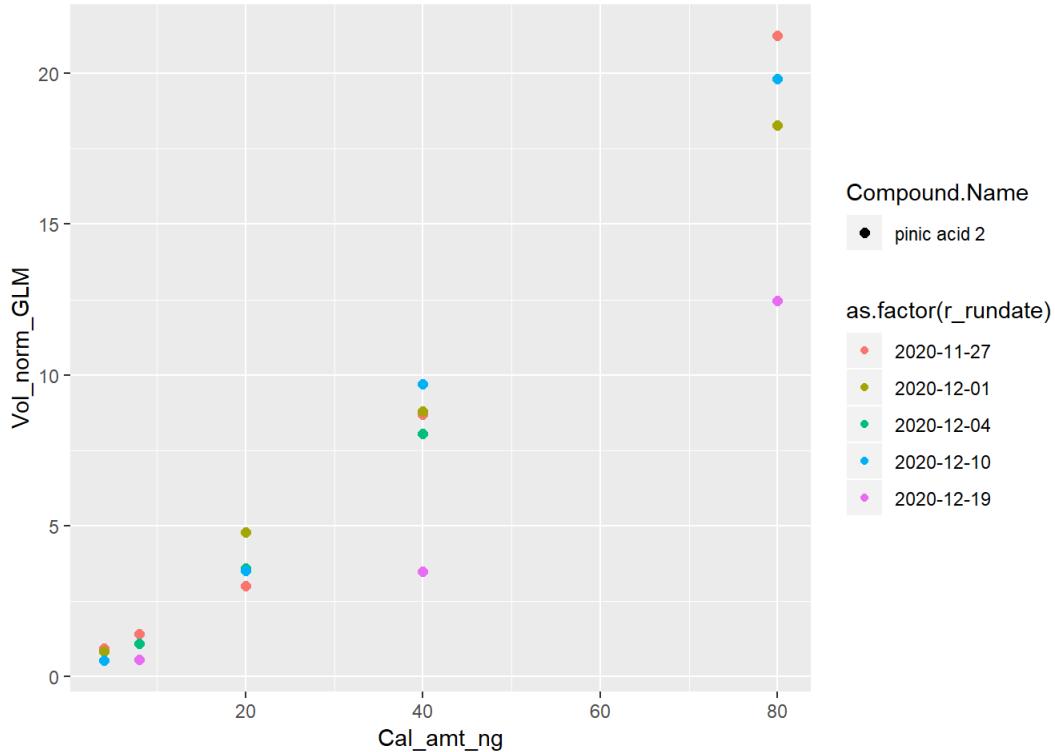
```
## Warning: Using size for a discrete variable is not advised.
```



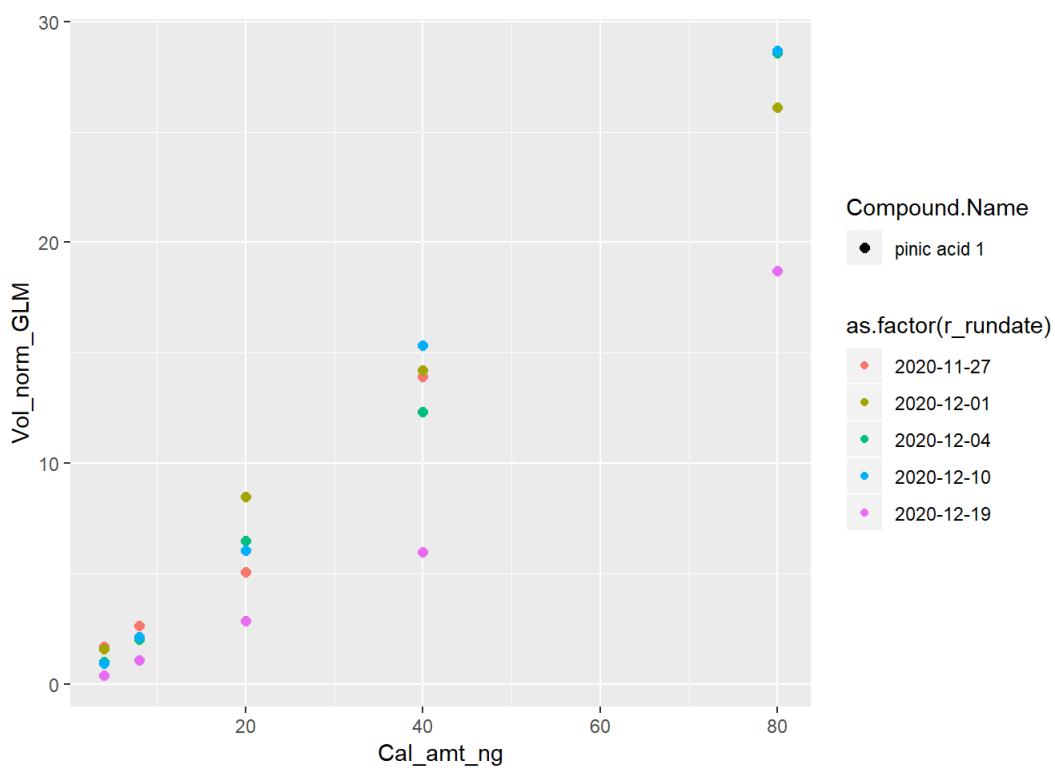
```
## Warning: Using size for a discrete variable is not advised.
```



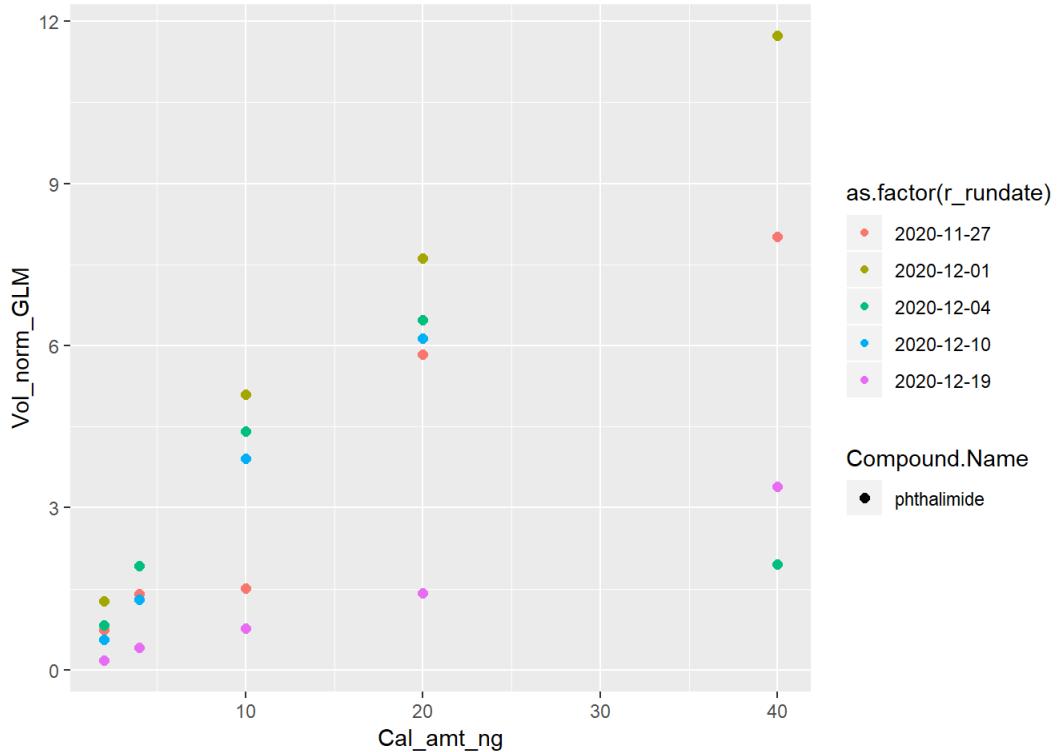
```
## Warning: Using size for a discrete variable is not advised.
```



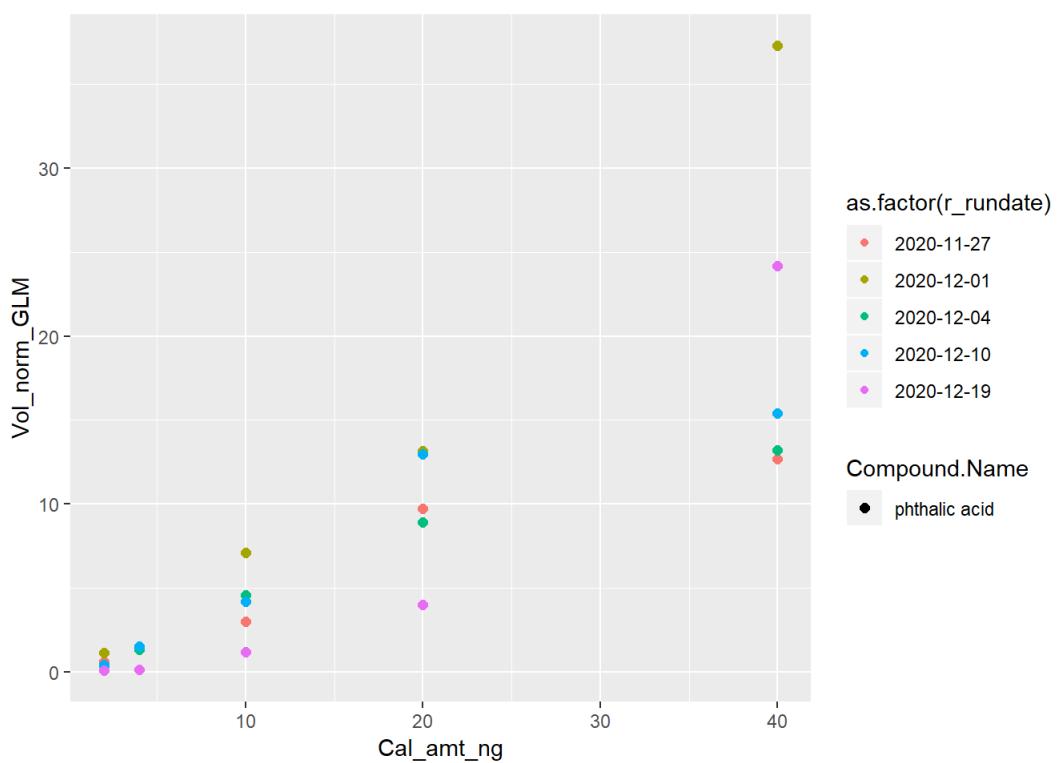
```
## Warning: Using size for a discrete variable is not advised.
```



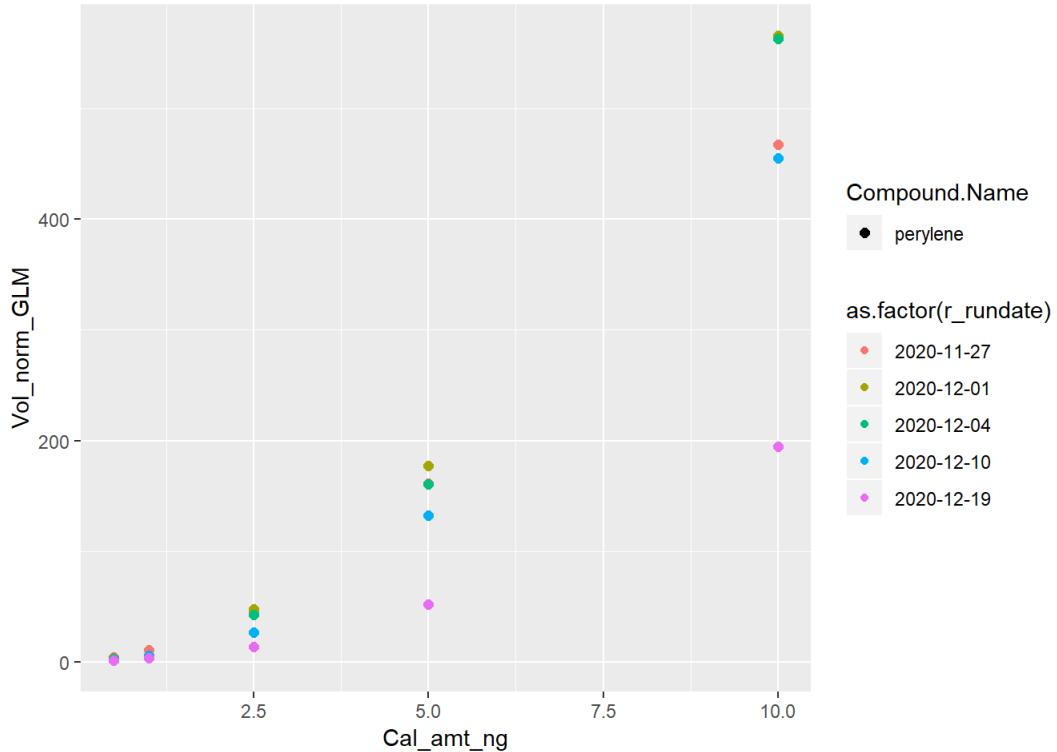
```
## Warning: Using size for a discrete variable is not advised.
```



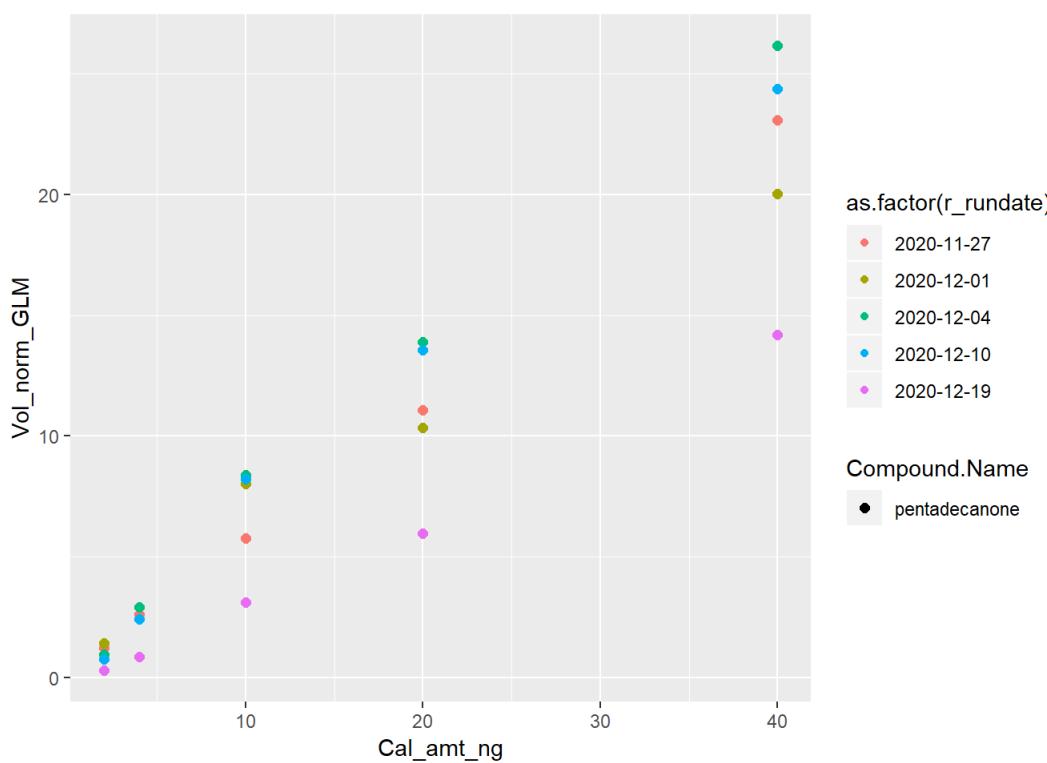
```
## Warning: Using size for a discrete variable is not advised.
```



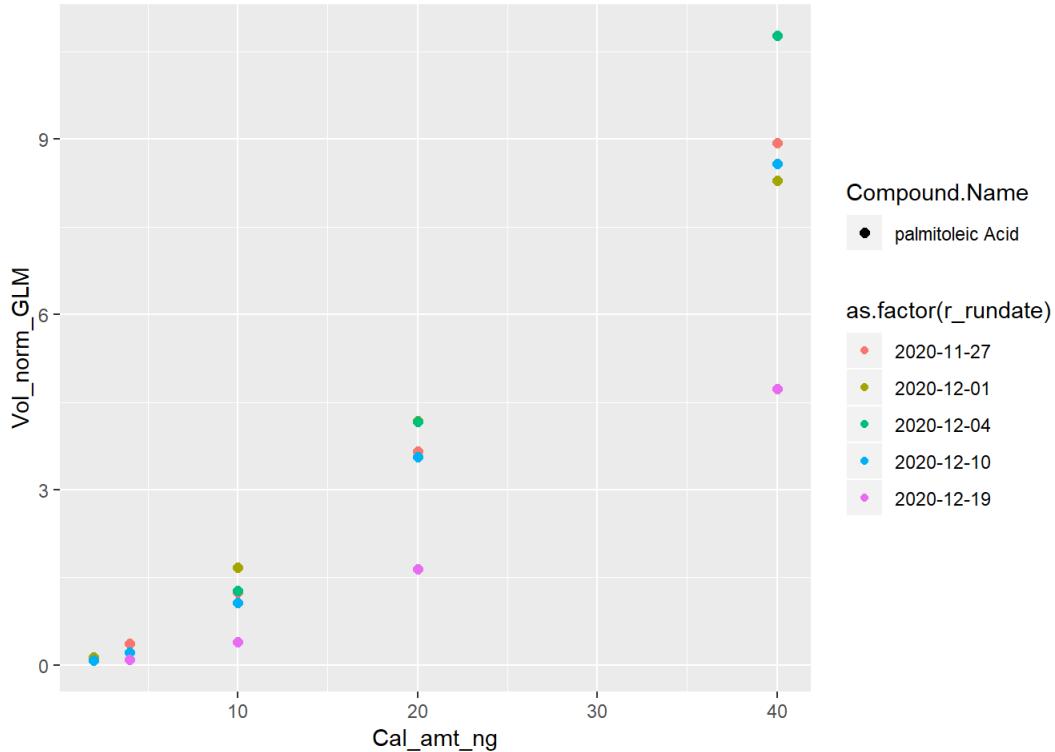
```
## Warning: Using size for a discrete variable is not advised.
```



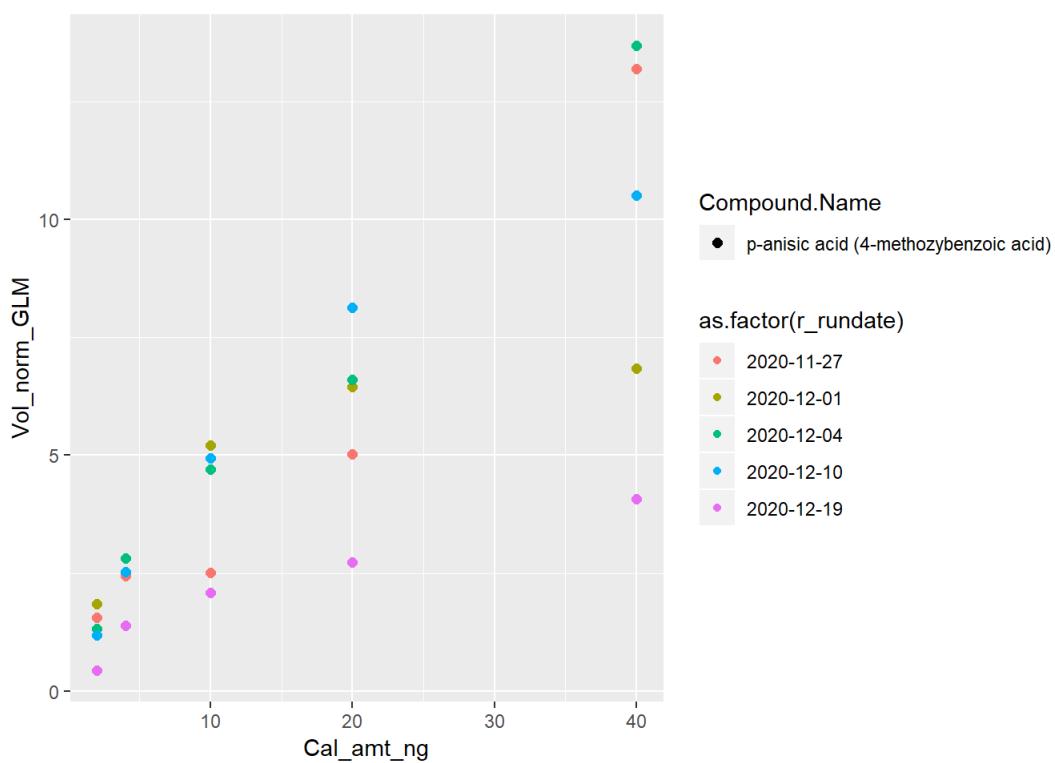
```
## Warning: Using size for a discrete variable is not advised.
```



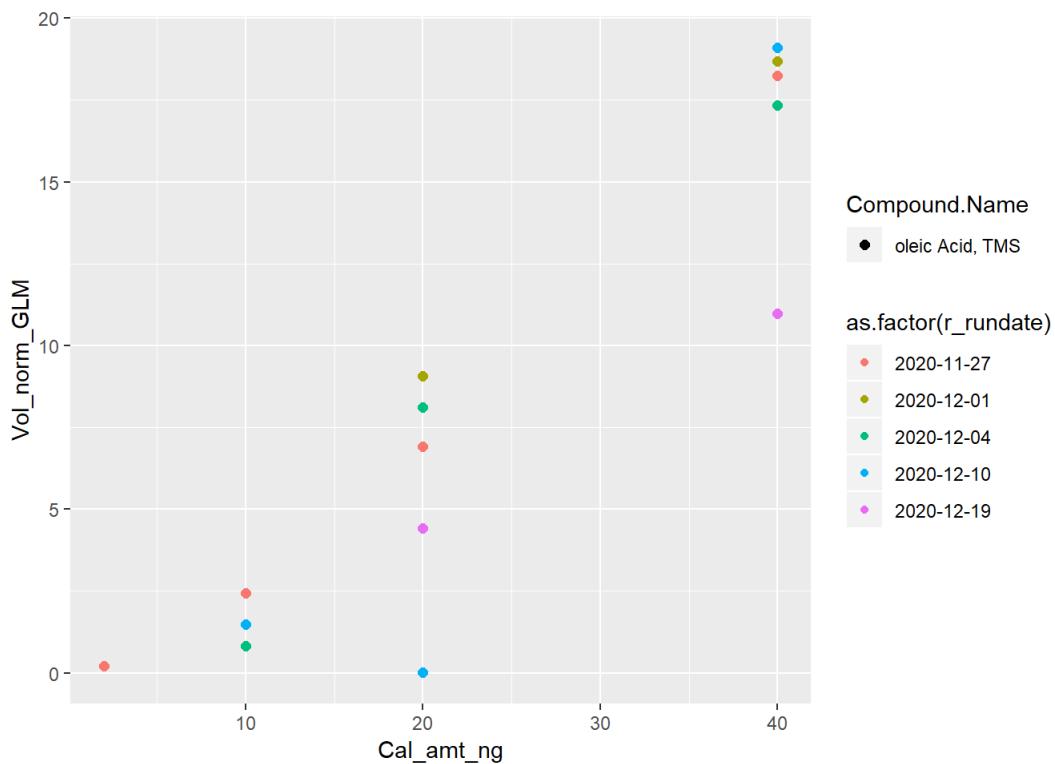
```
## Warning: Using size for a discrete variable is not advised.
```



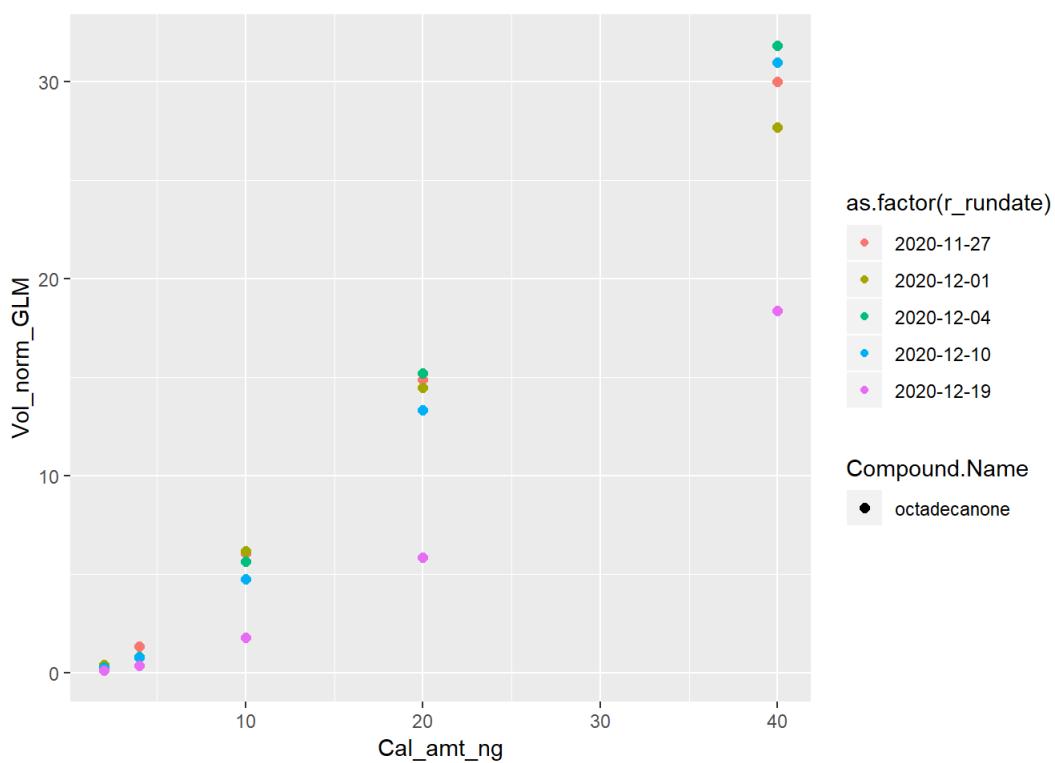
```
## Warning: Using size for a discrete variable is not advised.
```



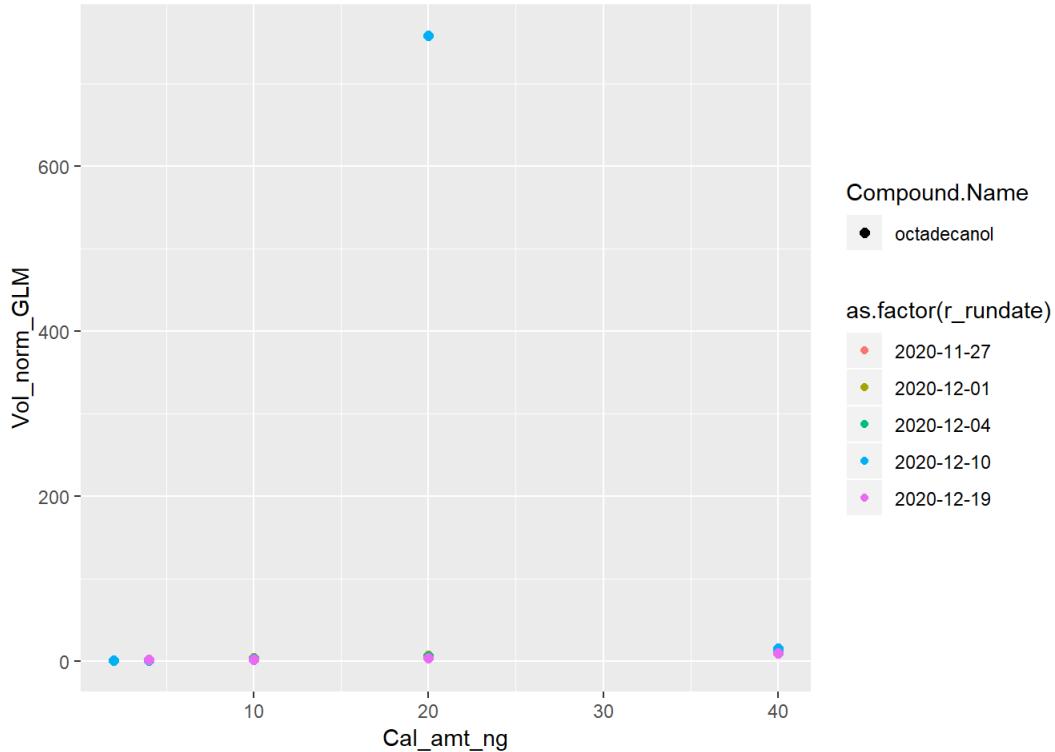
```
## Warning: Using size for a discrete variable is not advised.
```



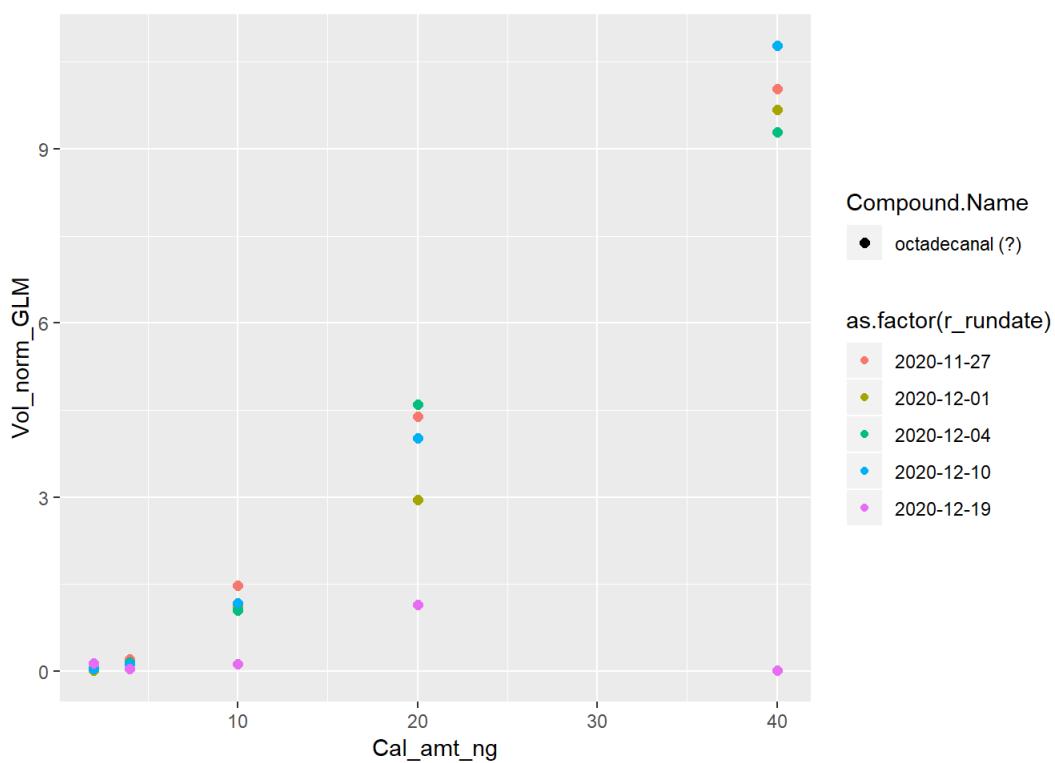
```
## Warning: Using size for a discrete variable is not advised.
```



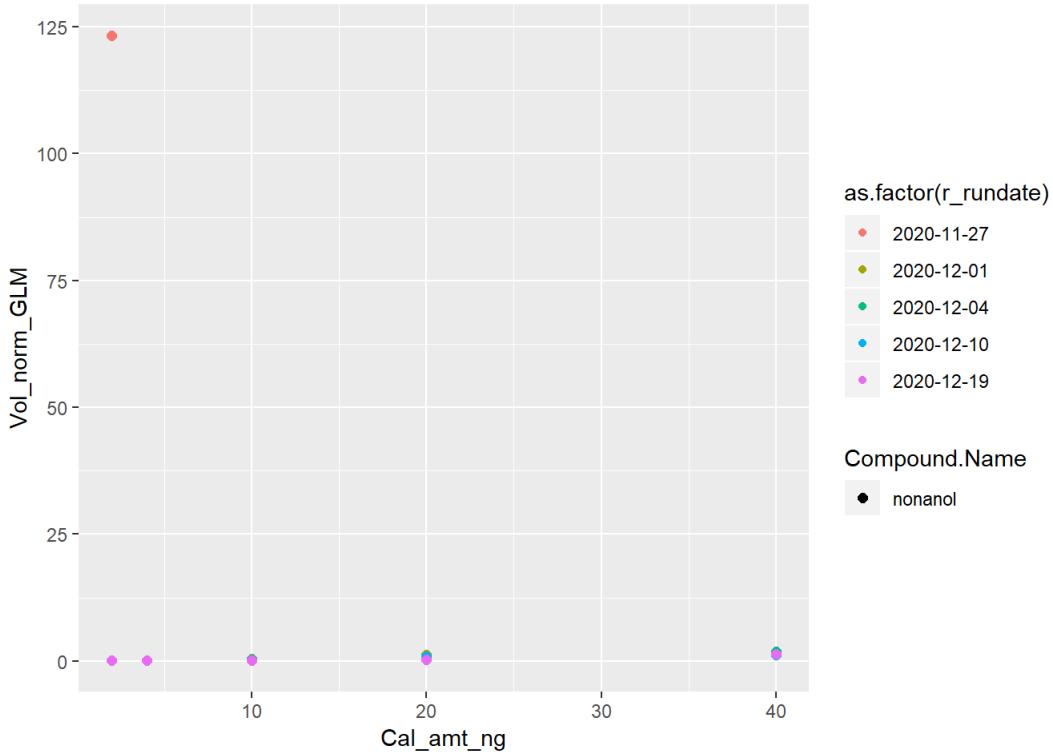
```
## Warning: Using size for a discrete variable is not advised.
```



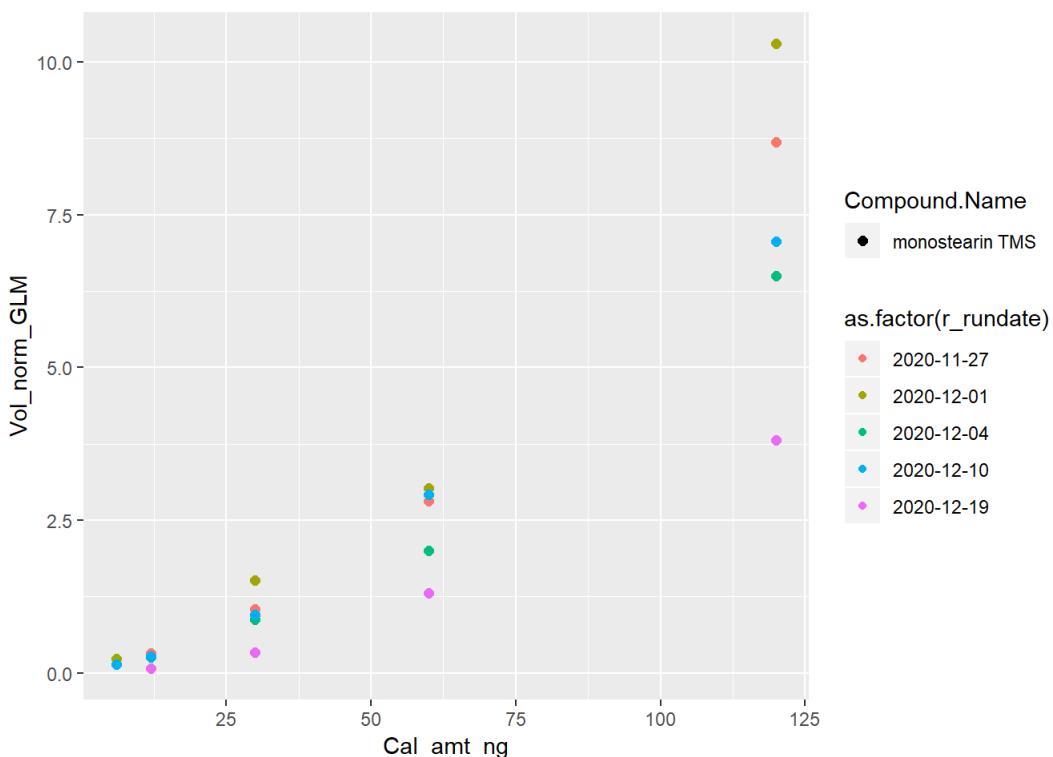
```
## Warning: Using size for a discrete variable is not advised.
```



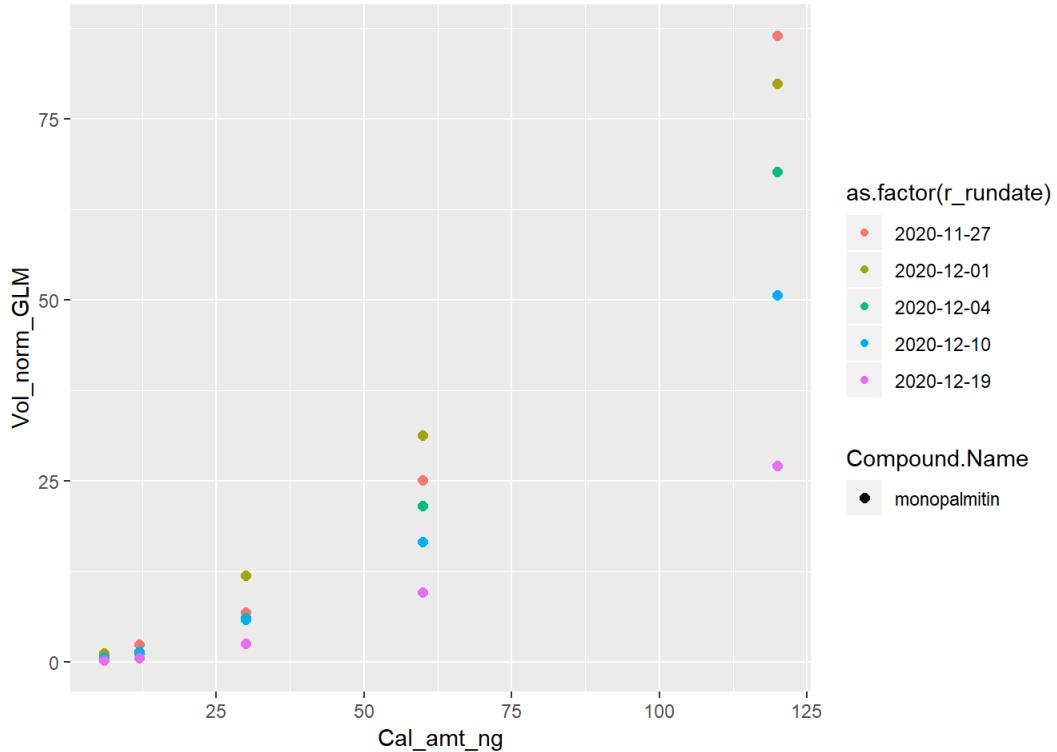
```
## Warning: Using size for a discrete variable is not advised.
```



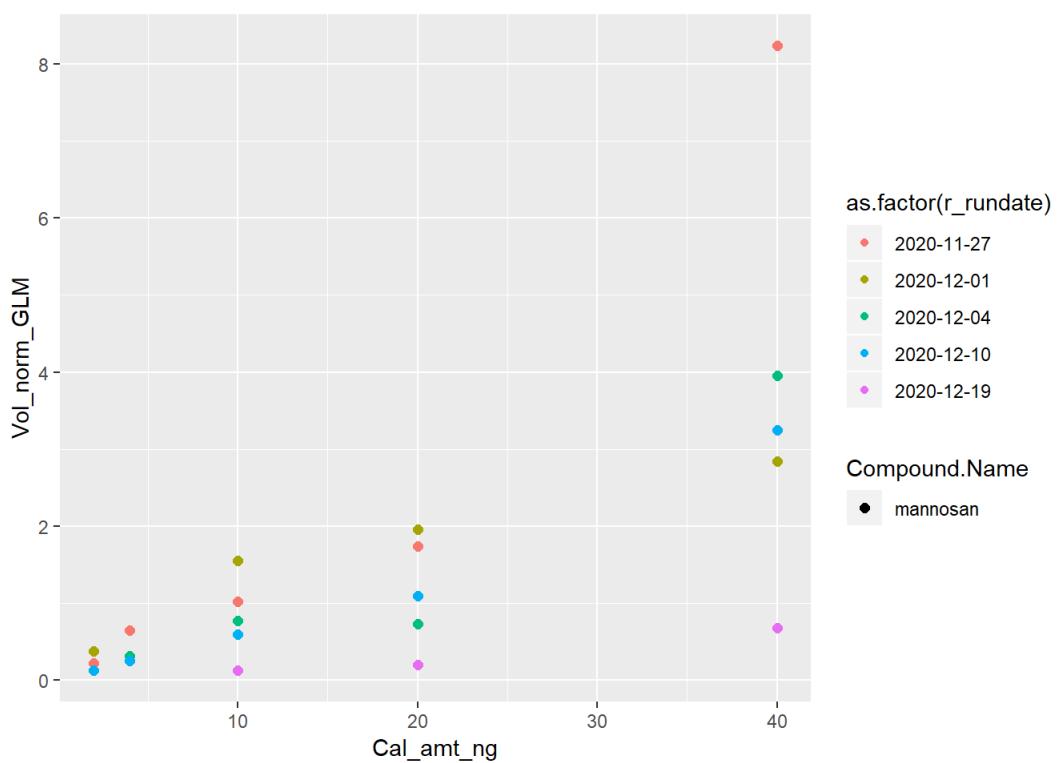
```
## Warning: Using size for a discrete variable is not advised.
```



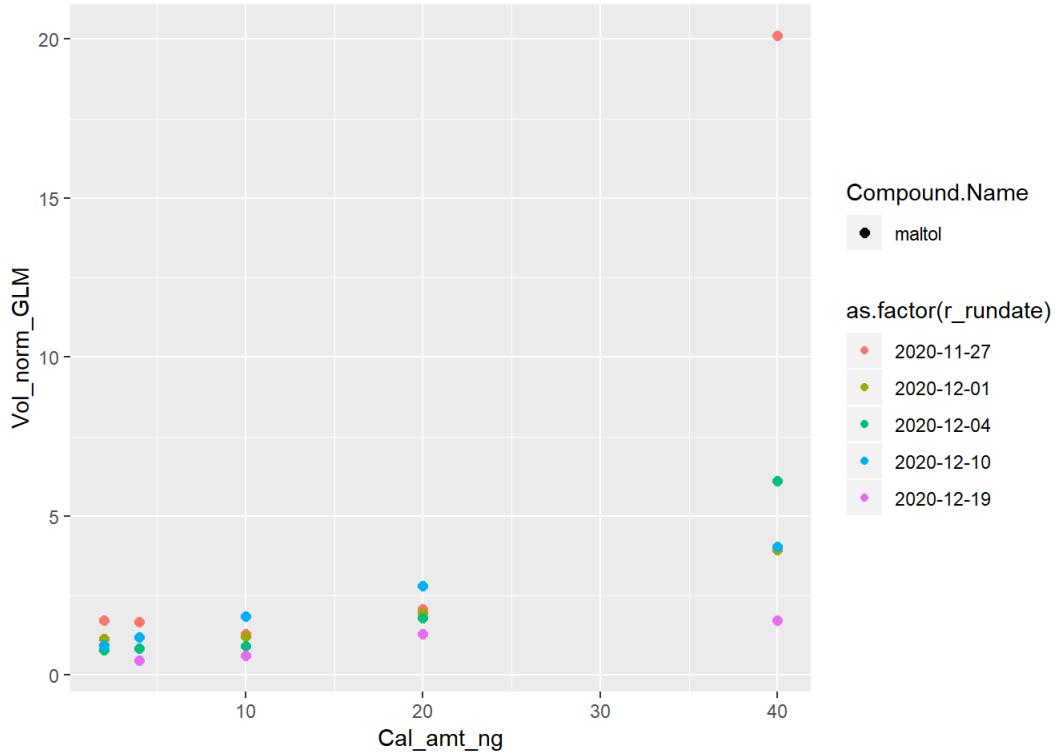
```
## Warning: Using size for a discrete variable is not advised.
```



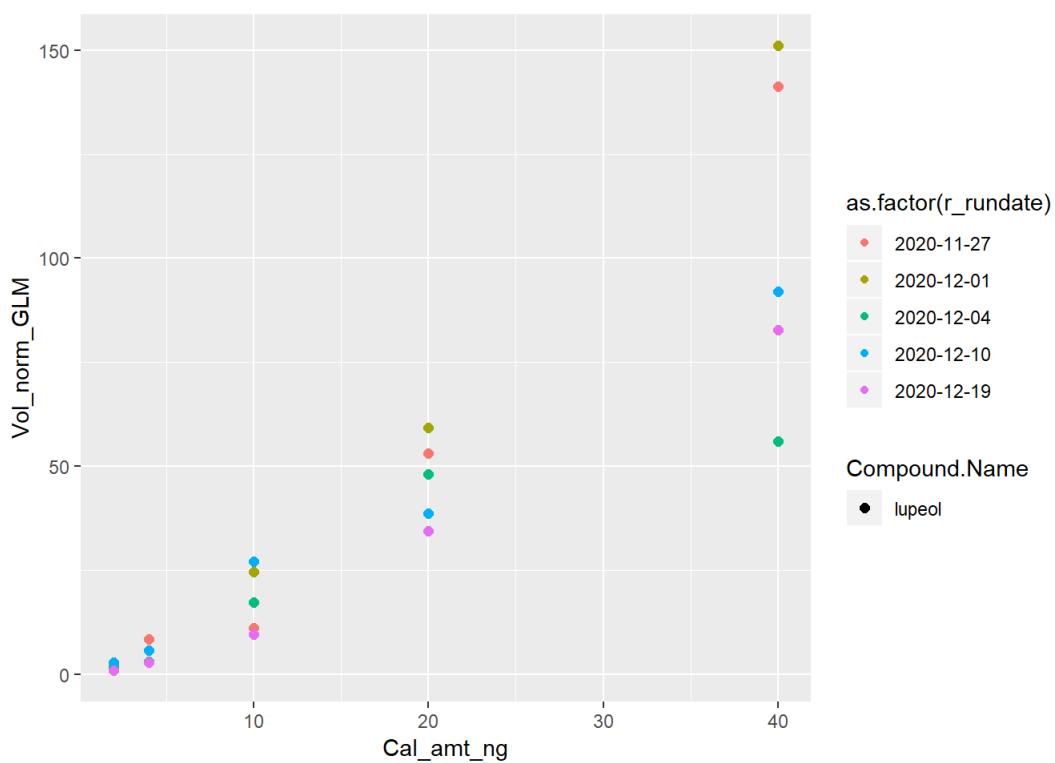
```
## Warning: Using size for a discrete variable is not advised.
```



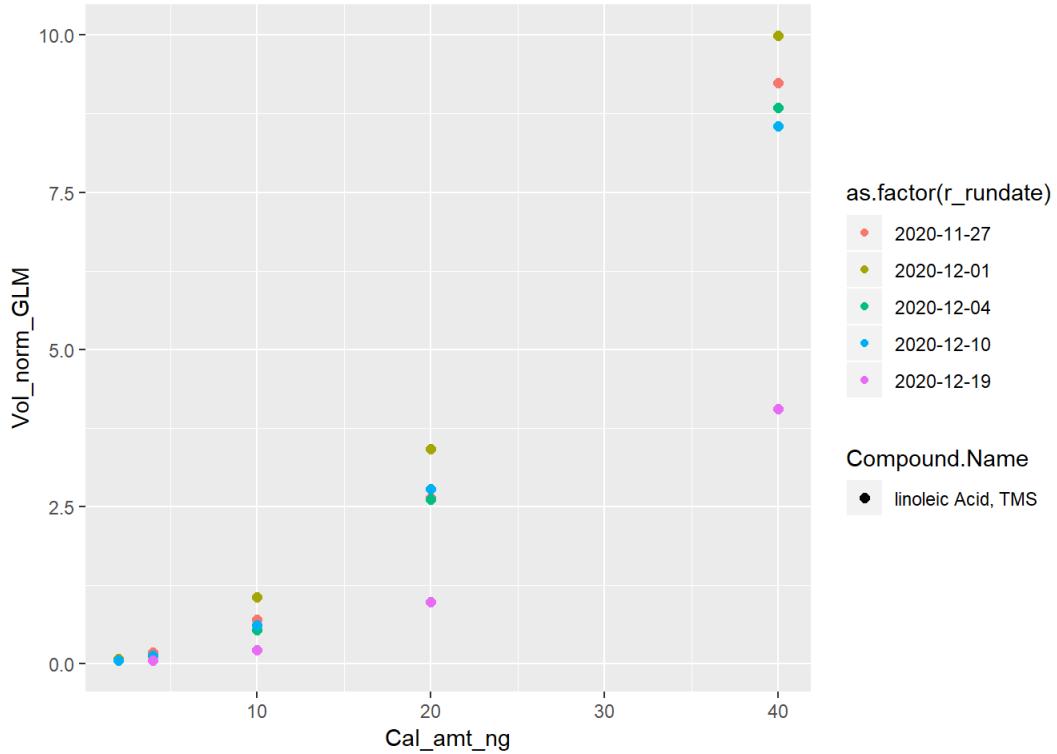
```
## Warning: Using size for a discrete variable is not advised.
```



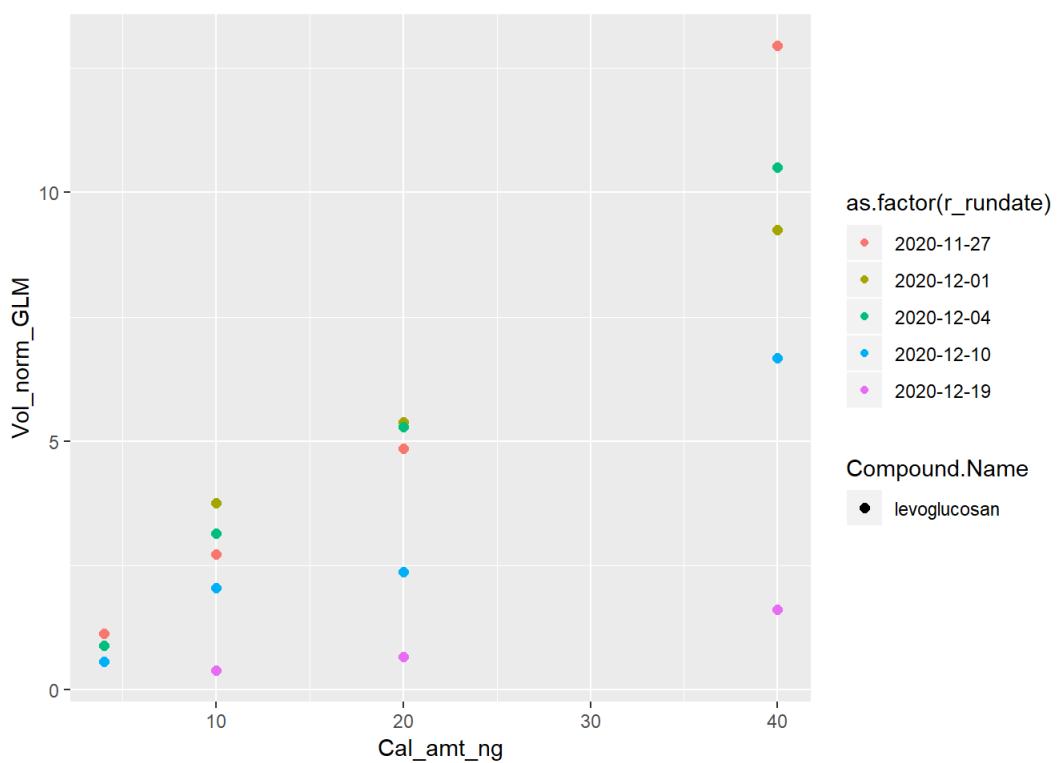
```
## Warning: Using size for a discrete variable is not advised.
```



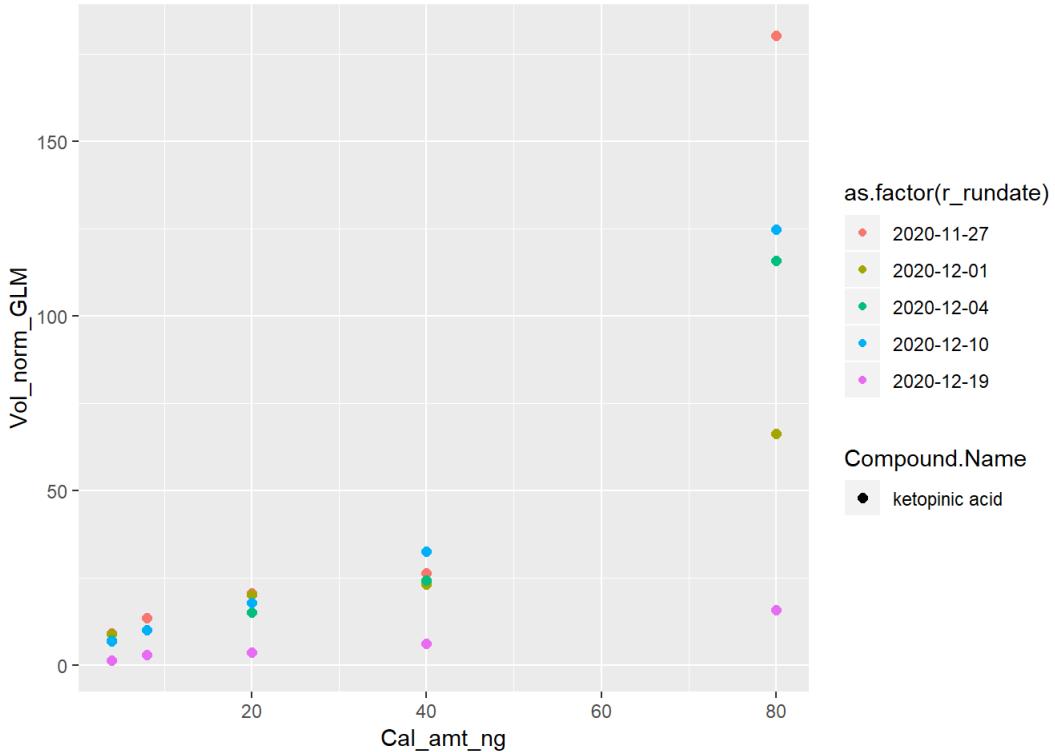
```
## Warning: Using size for a discrete variable is not advised.
```



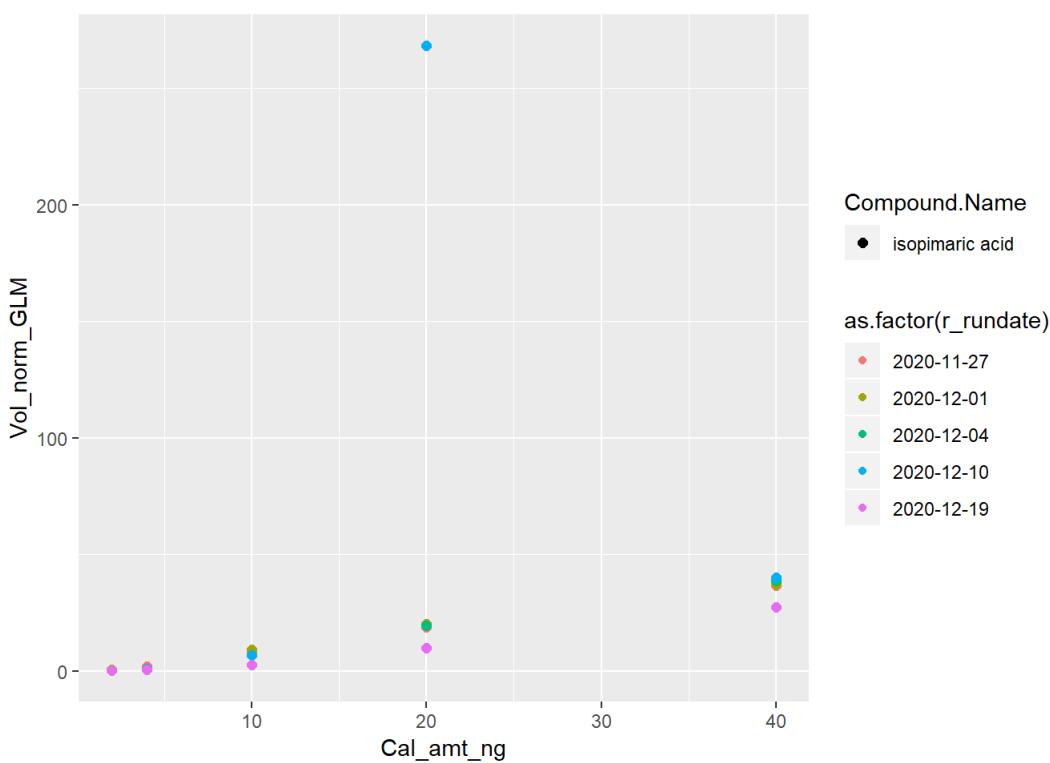
```
## Warning: Using size for a discrete variable is not advised.
```



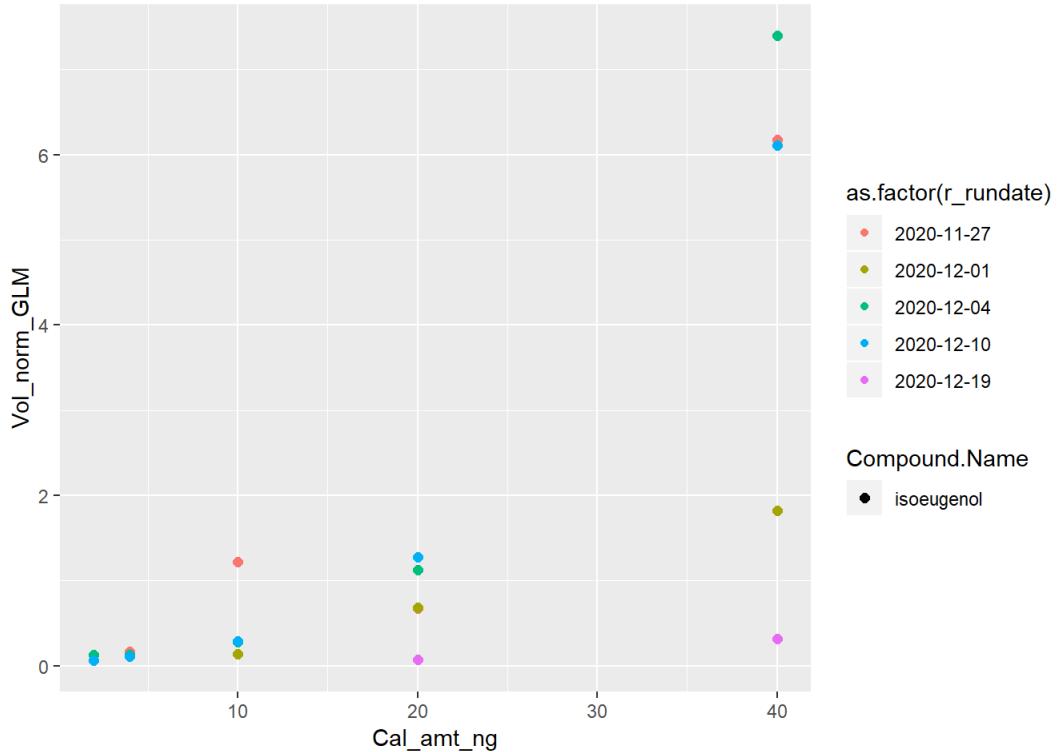
```
## Warning: Using size for a discrete variable is not advised.
```



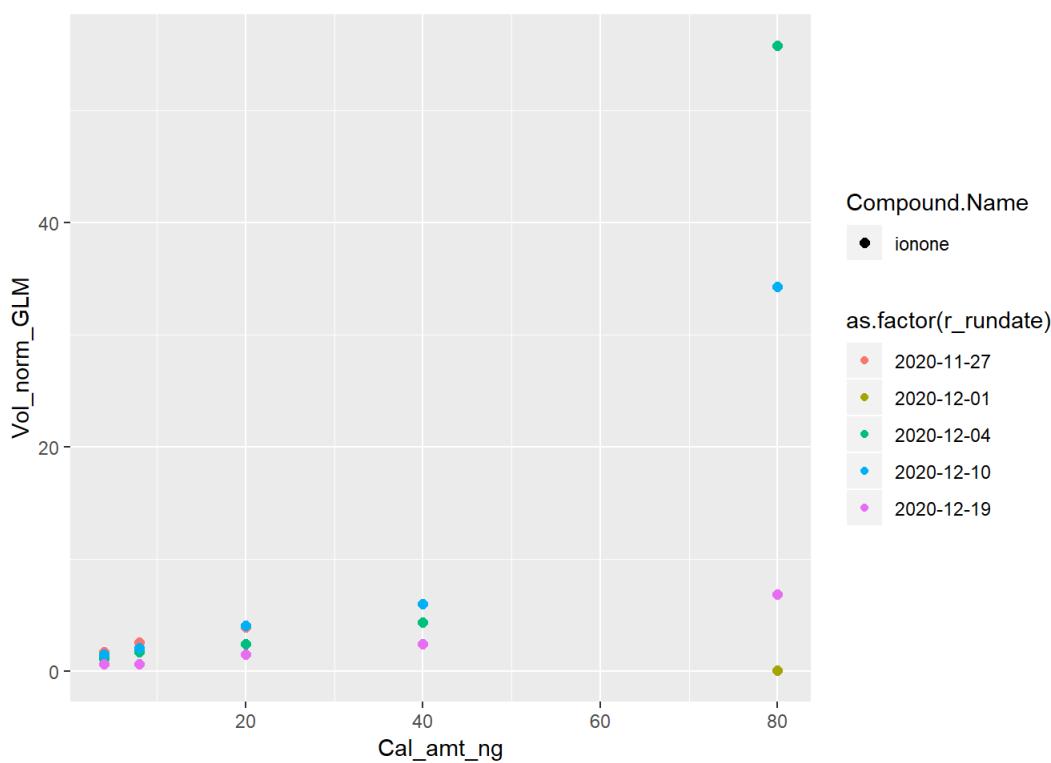
```
## Warning: Using size for a discrete variable is not advised.
```



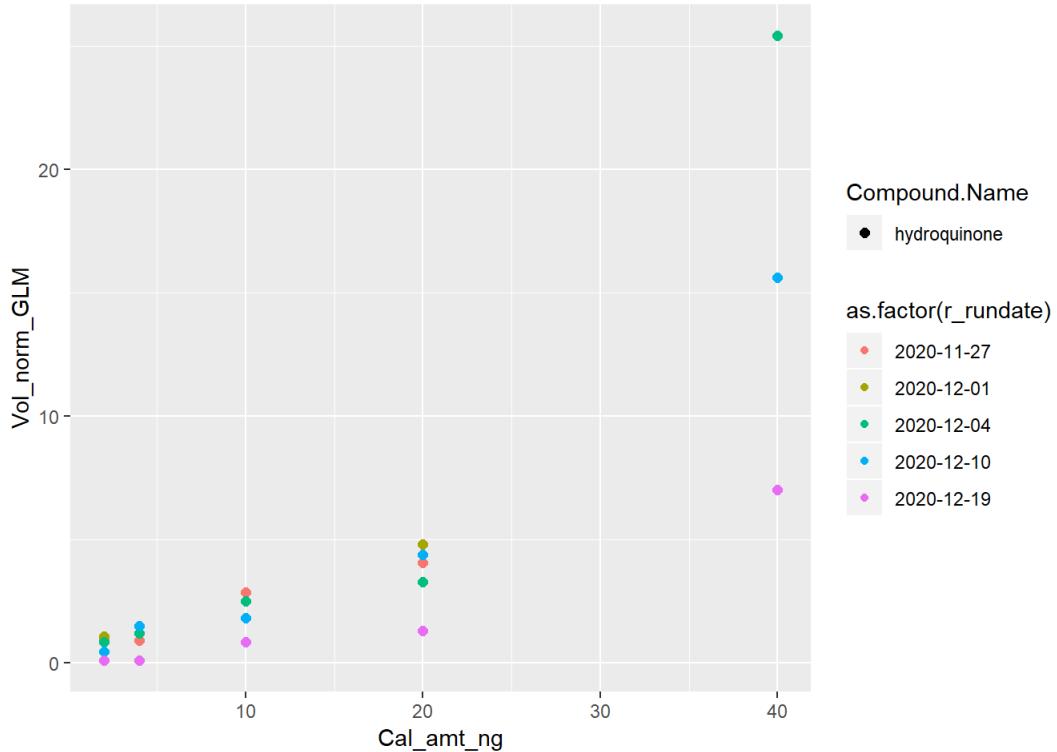
```
## Warning: Using size for a discrete variable is not advised.
```



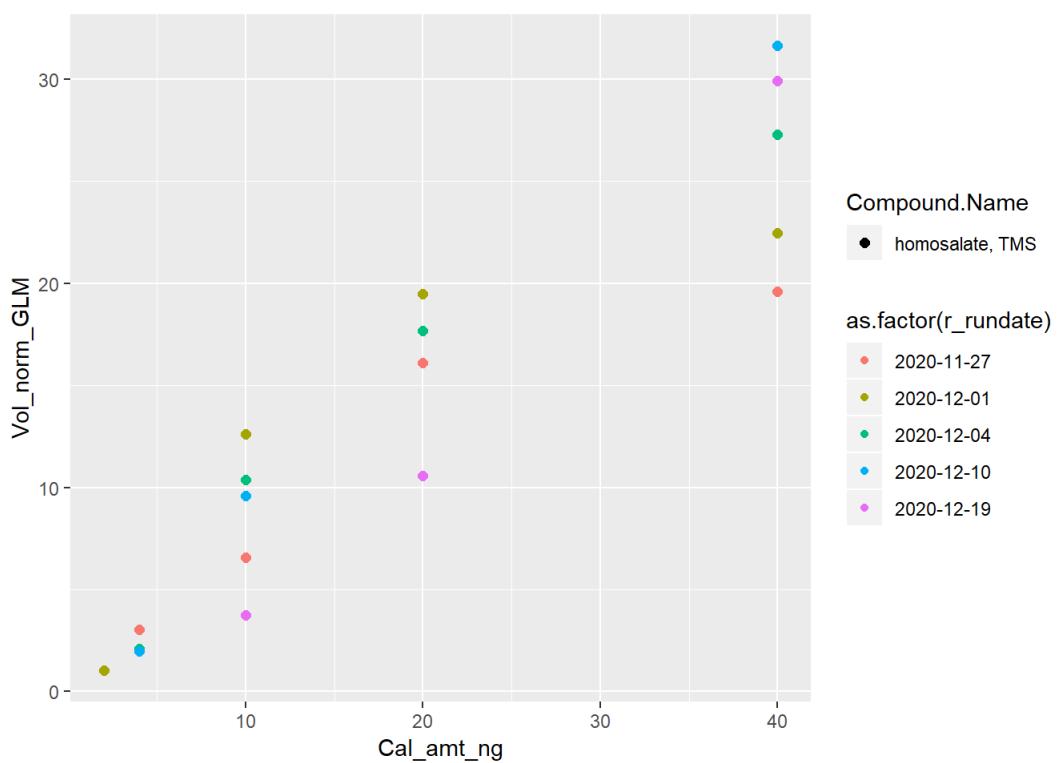
```
## Warning: Using size for a discrete variable is not advised.
```



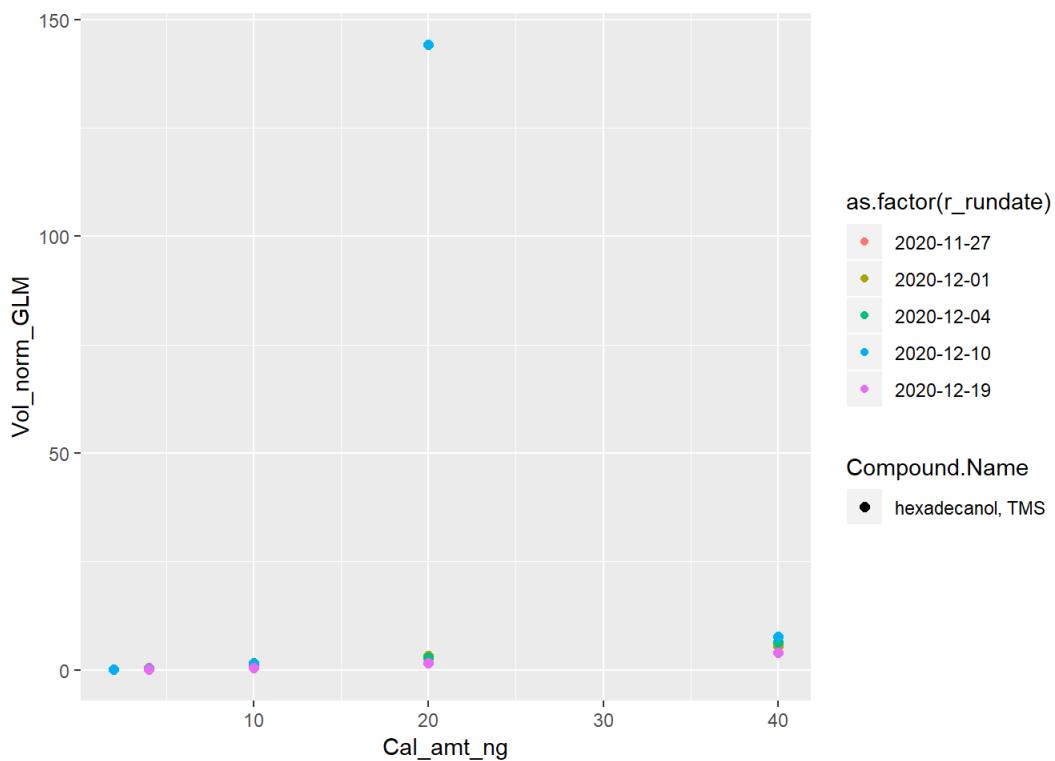
```
## Warning: Using size for a discrete variable is not advised.
```



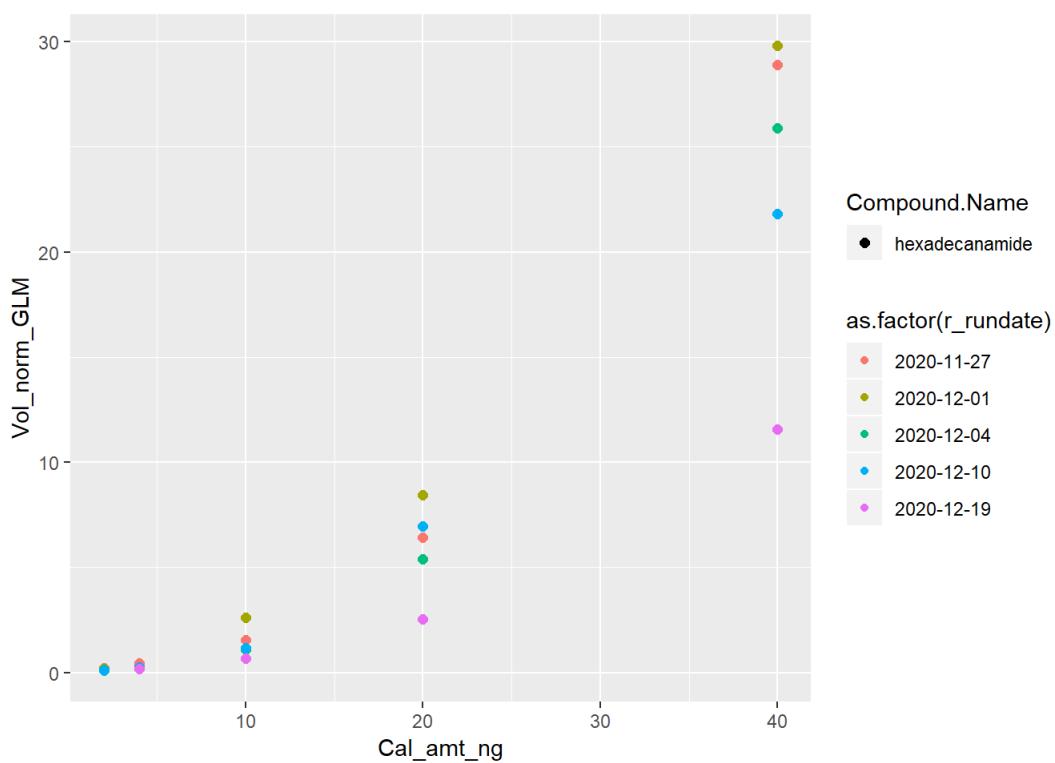
```
## Warning: Using size for a discrete variable is not advised.
```



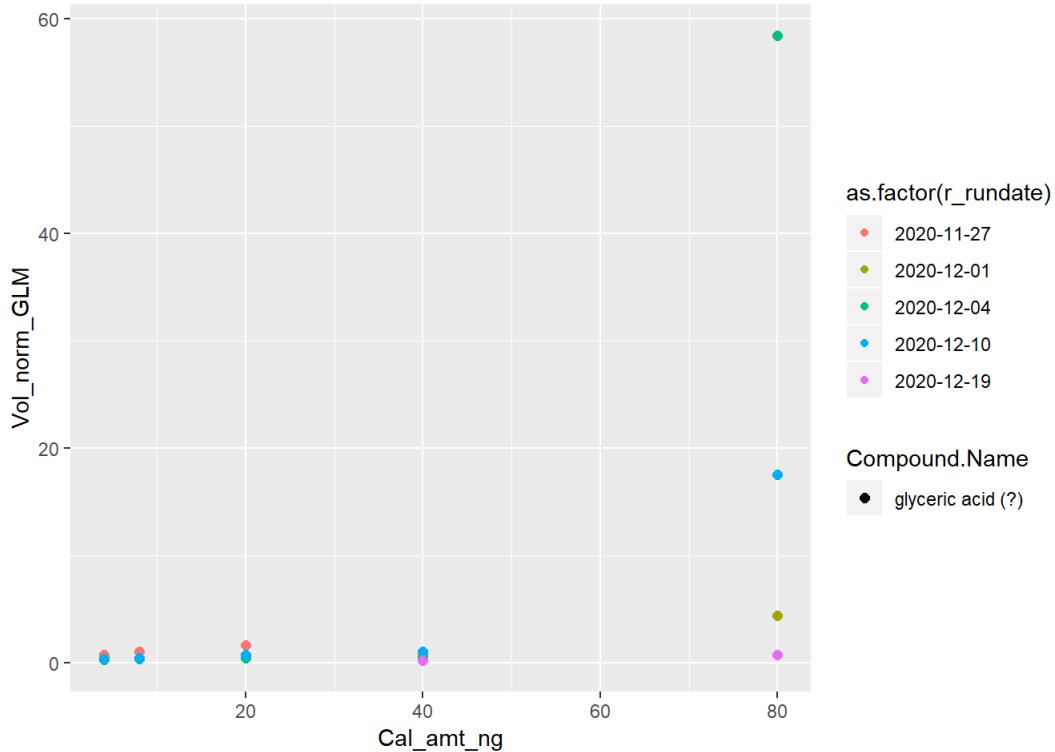
```
## Warning: Using size for a discrete variable is not advised.
```



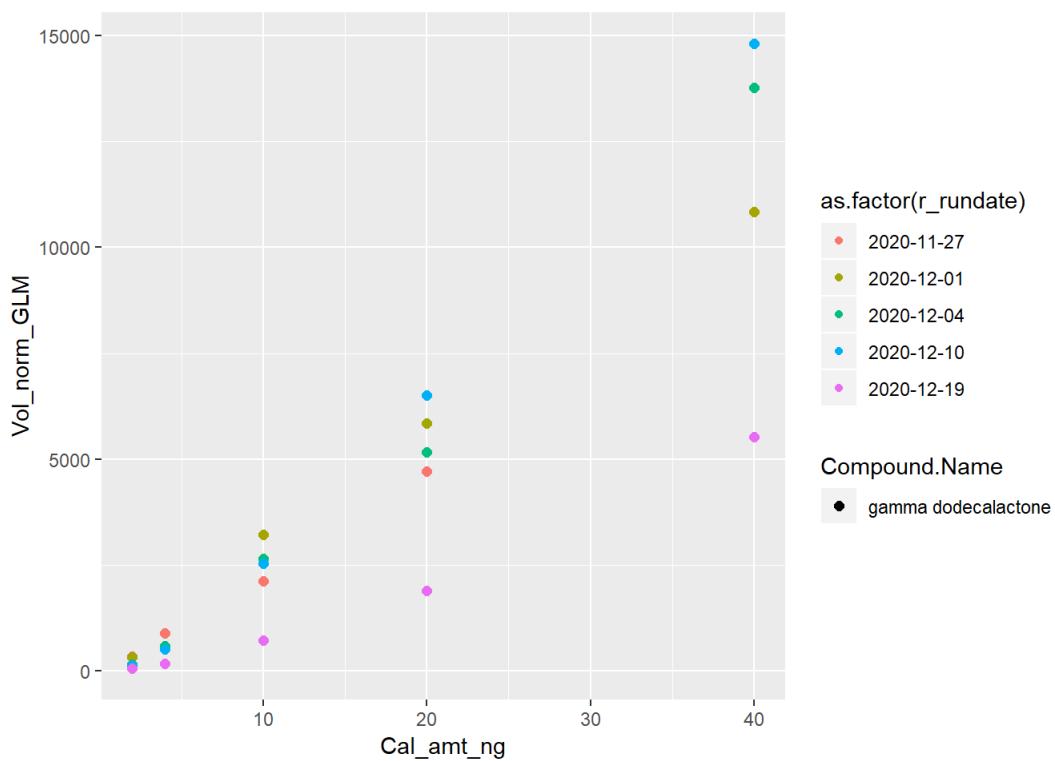
```
## Warning: Using size for a discrete variable is not advised.
```



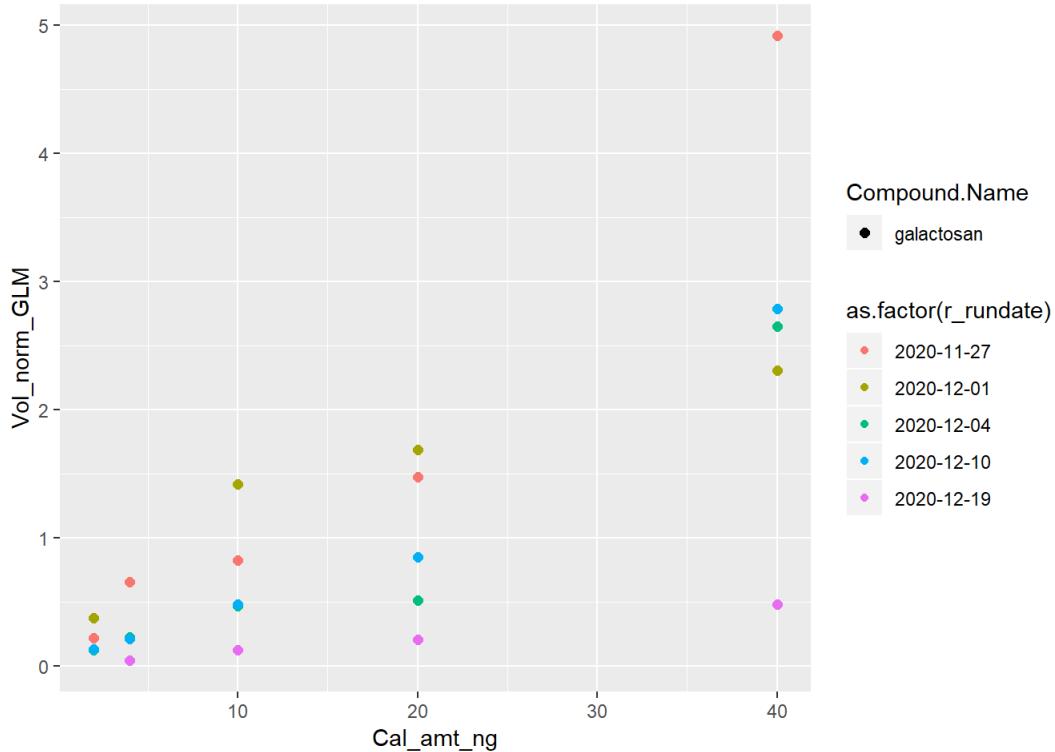
```
## Warning: Using size for a discrete variable is not advised.
```



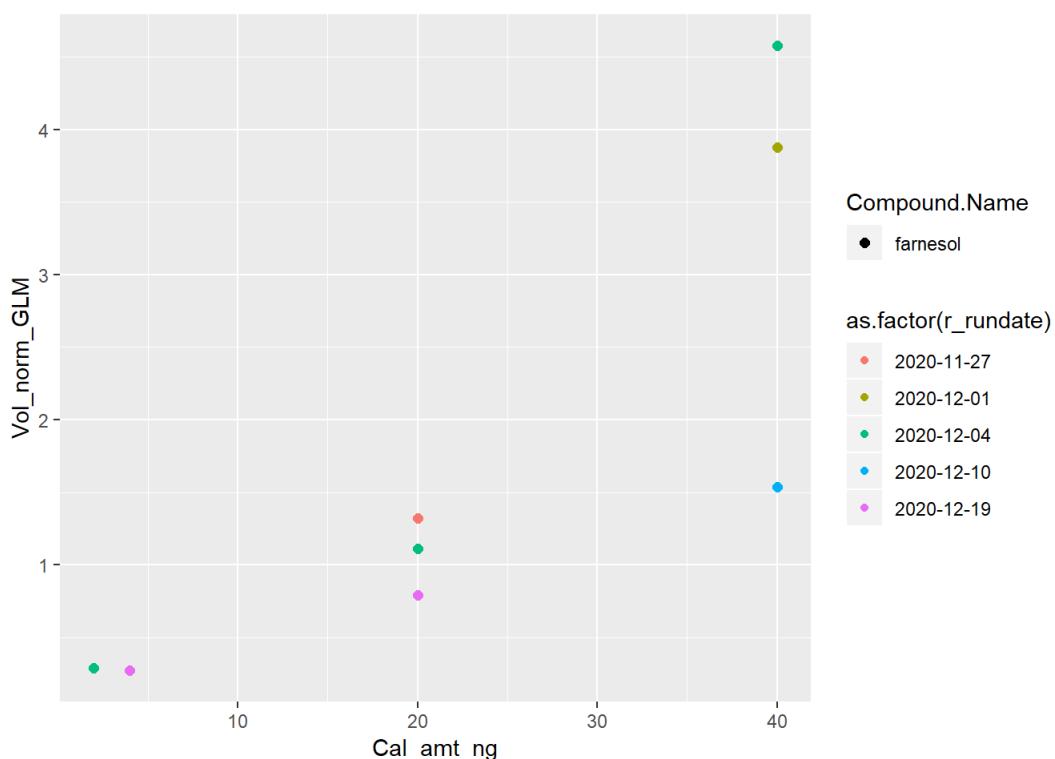
```
## Warning: Using size for a discrete variable is not advised.
```



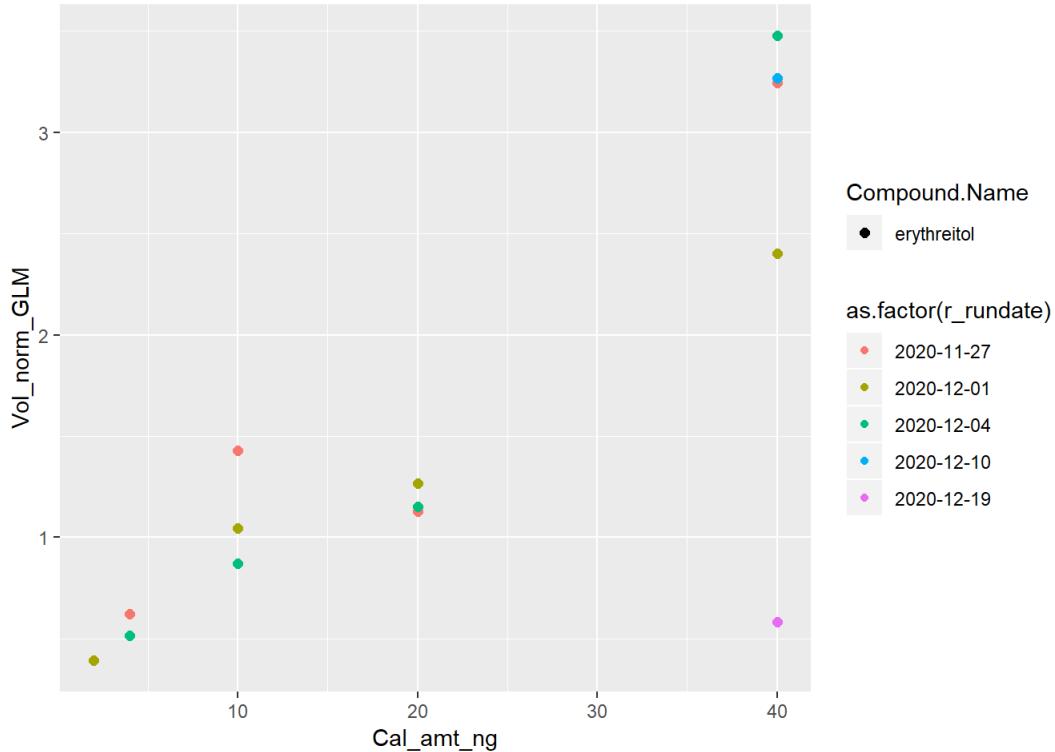
```
## Warning: Using size for a discrete variable is not advised.
```



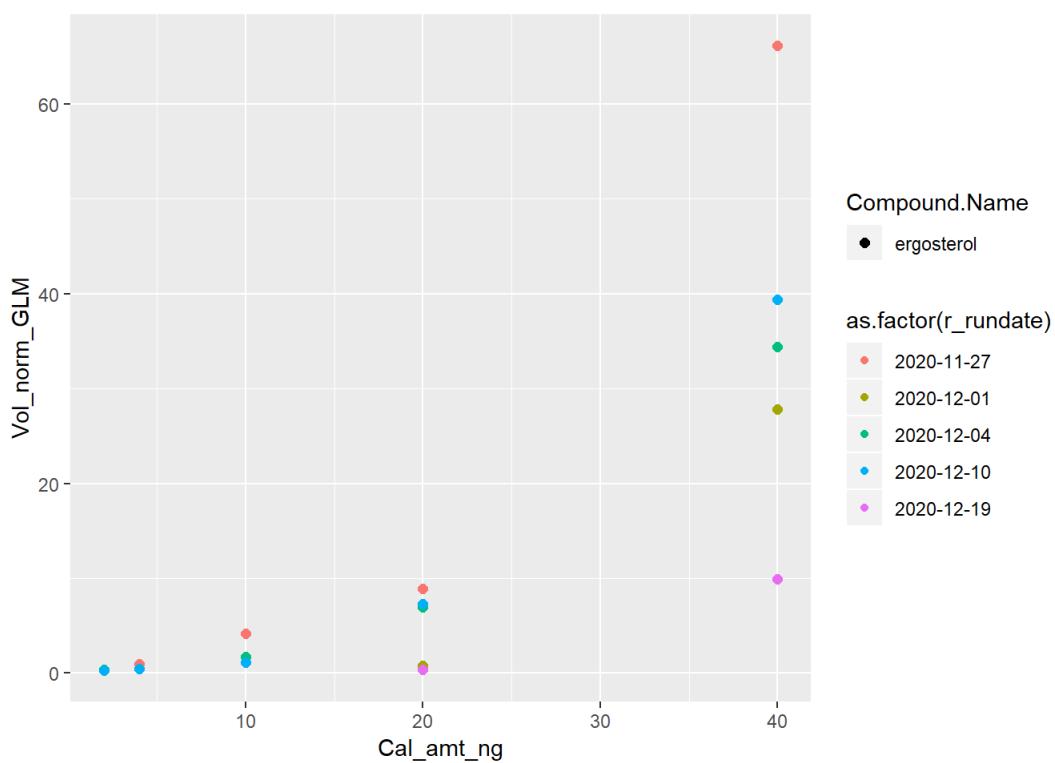
```
## Warning: Using size for a discrete variable is not advised.
```



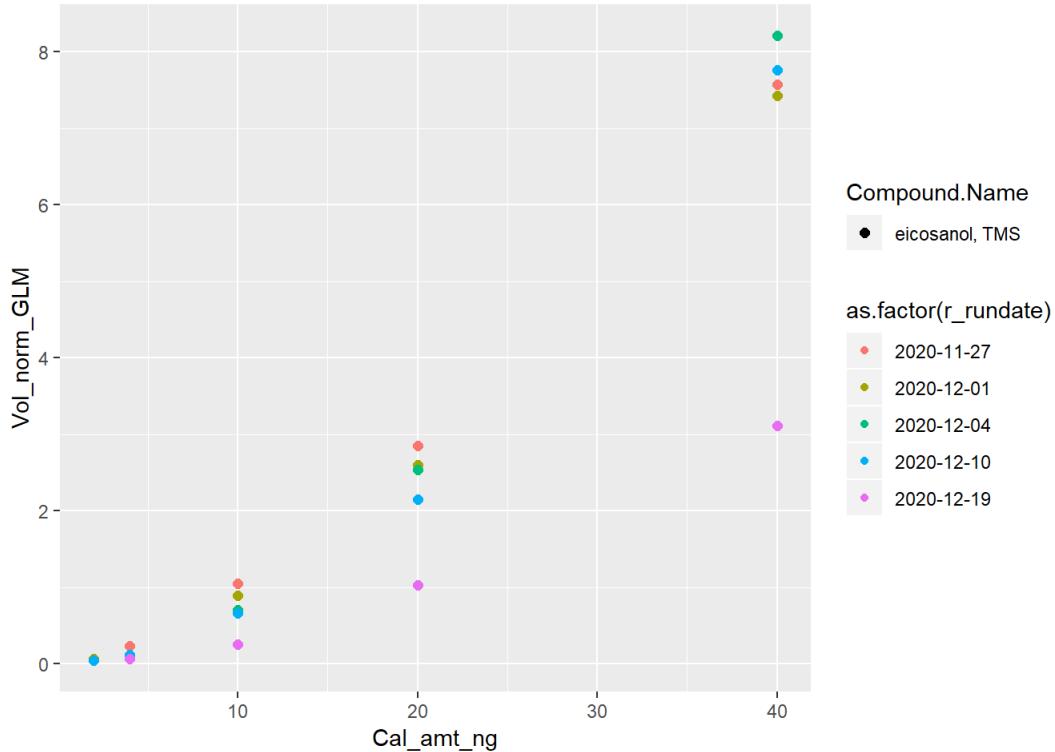
```
## Warning: Using size for a discrete variable is not advised.
```



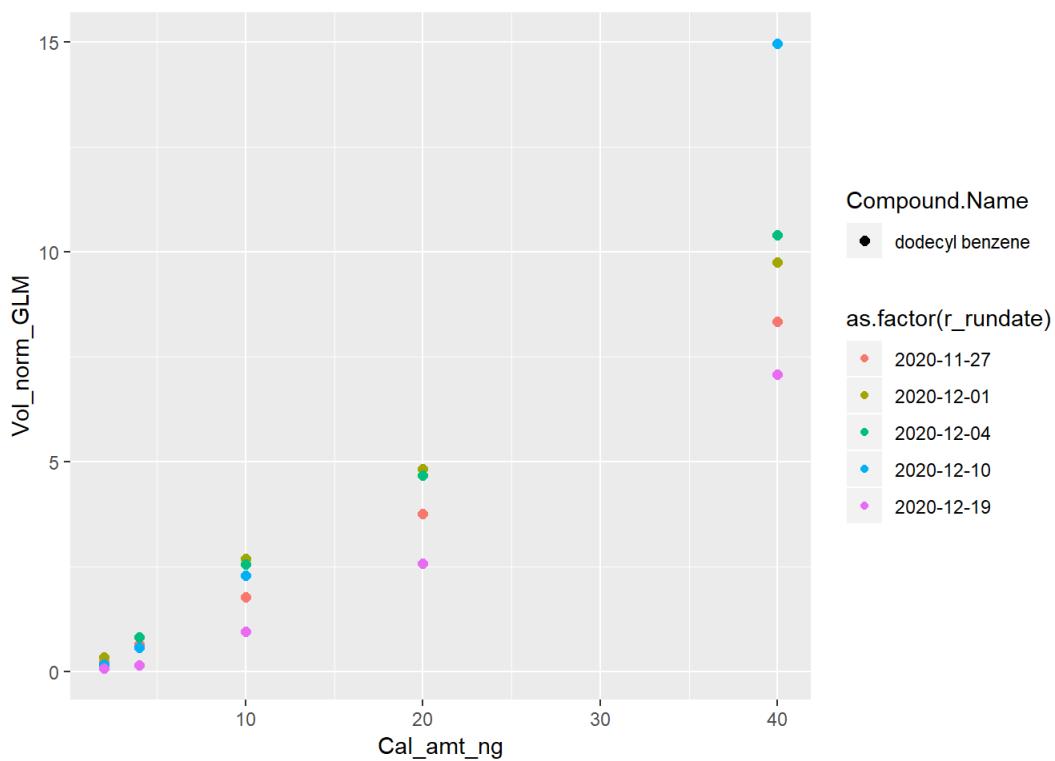
```
## Warning: Using size for a discrete variable is not advised.
```



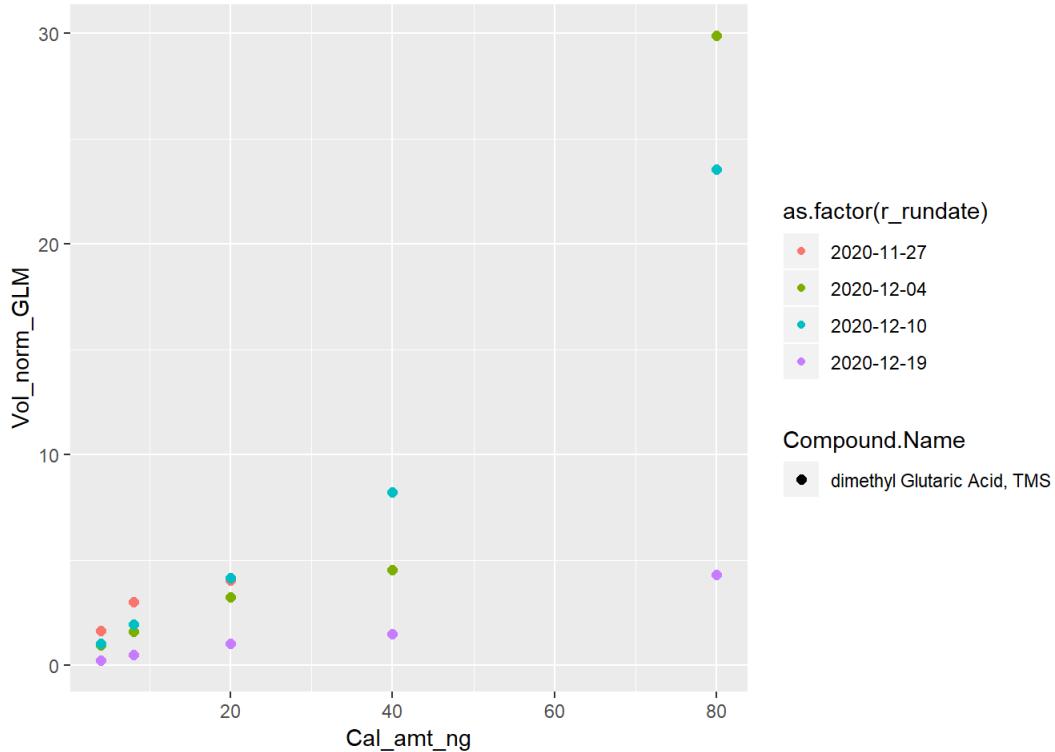
```
## Warning: Using size for a discrete variable is not advised.
```



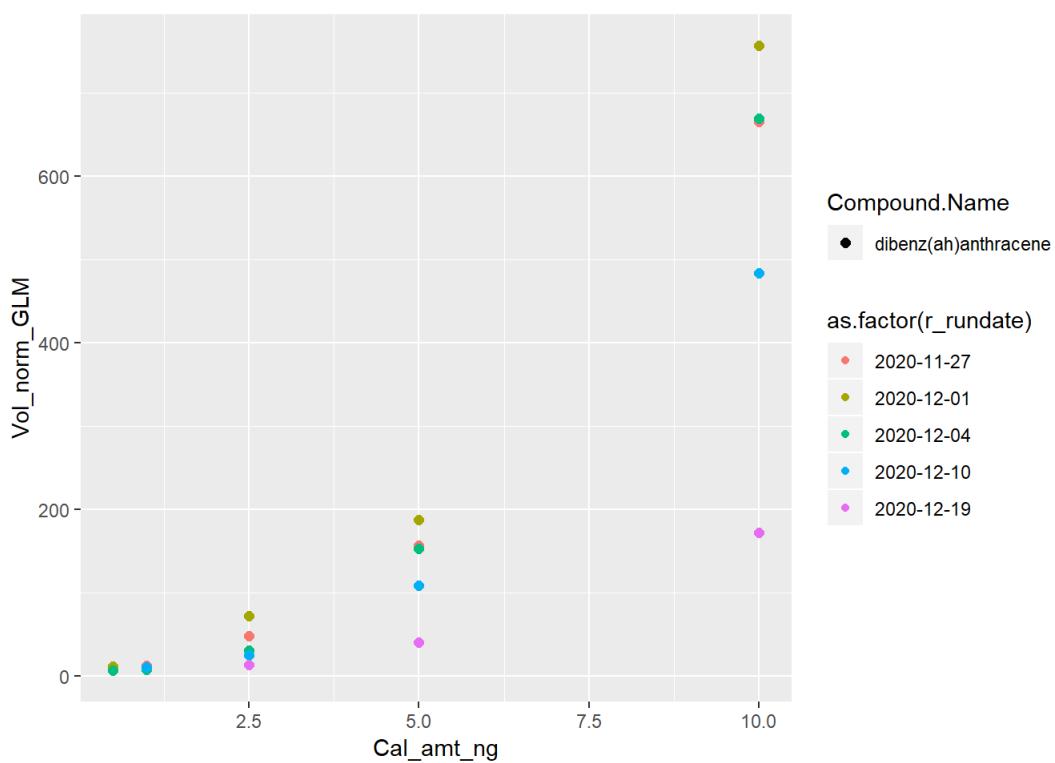
```
## Warning: Using size for a discrete variable is not advised.
```



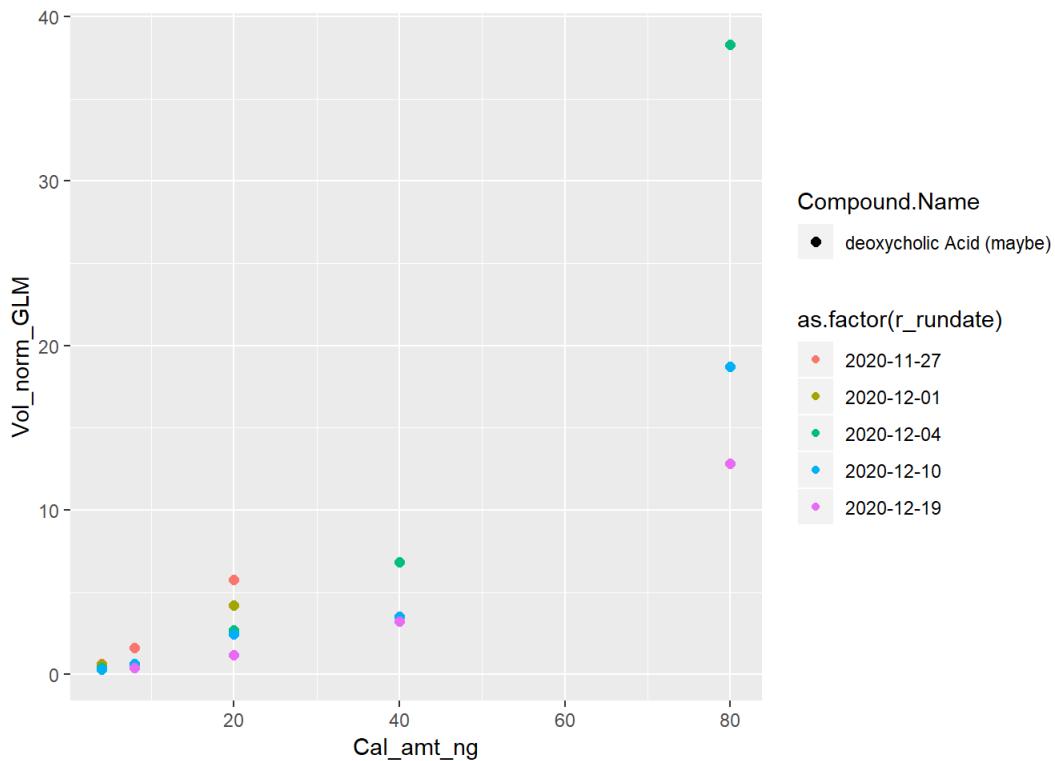
```
## Warning: Using size for a discrete variable is not advised.
```



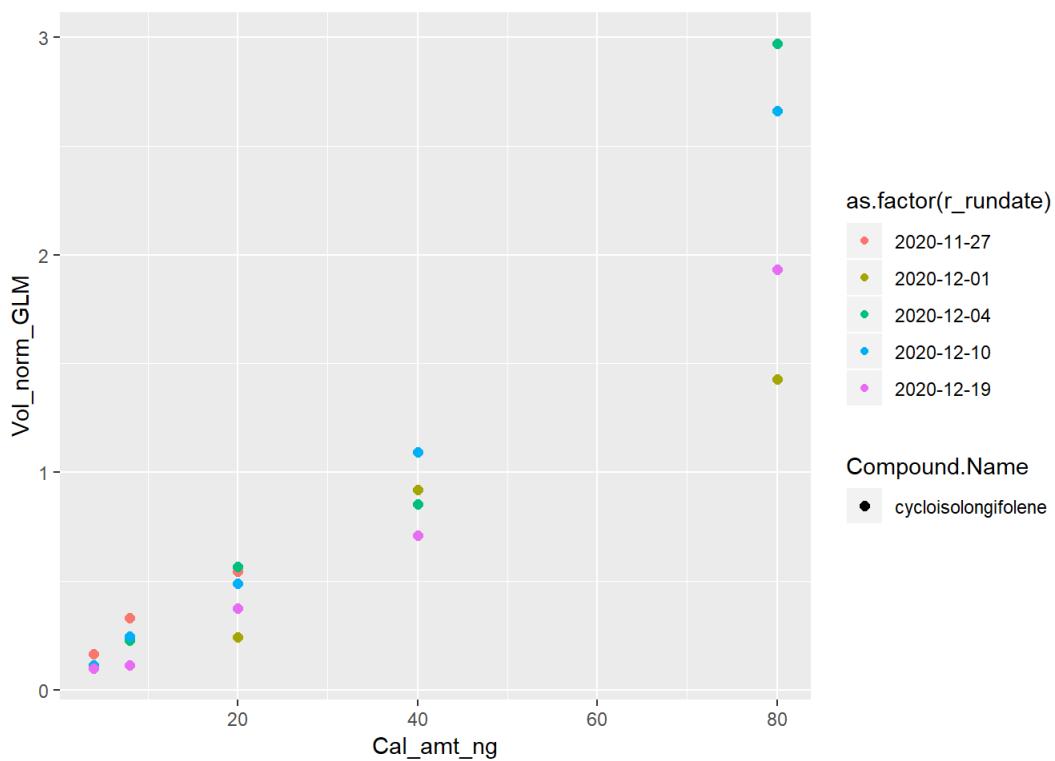
```
## Warning: Using size for a discrete variable is not advised.
```



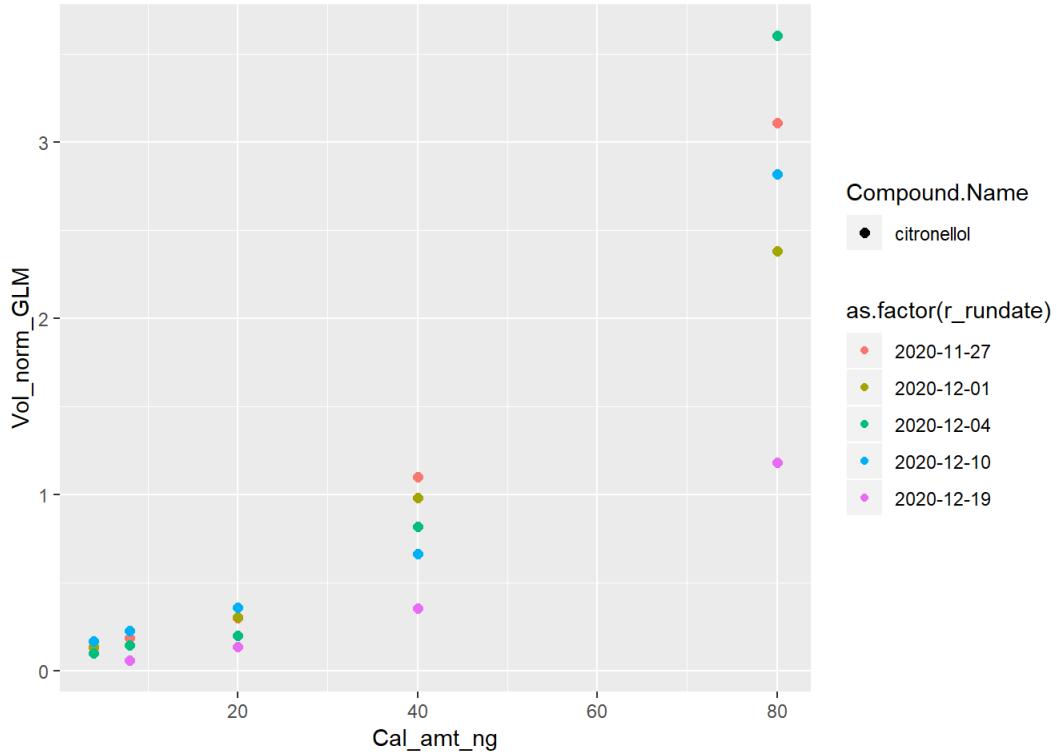
```
## Warning: Using size for a discrete variable is not advised.
```



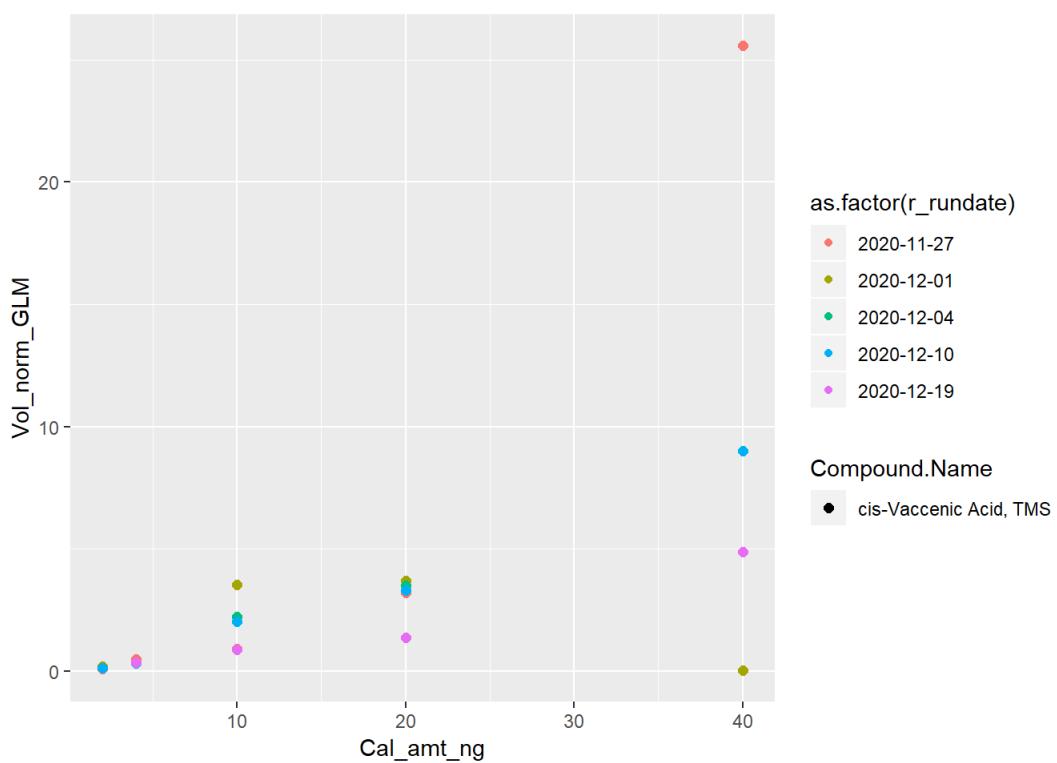
```
## Warning: Using size for a discrete variable is not advised.
```



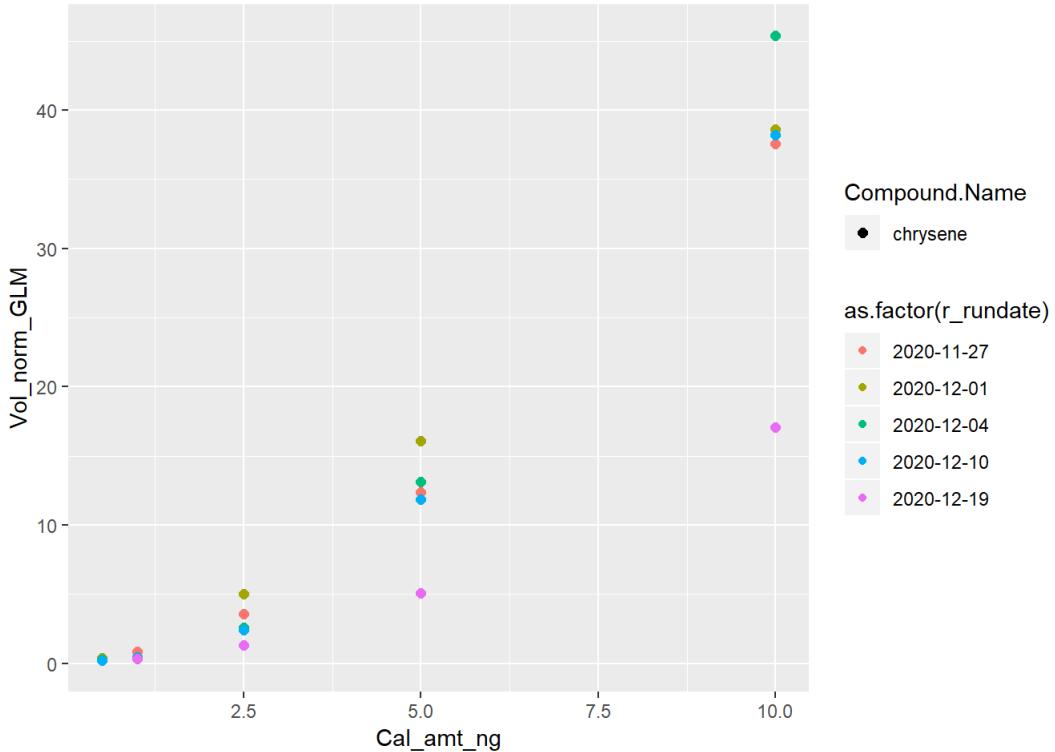
```
## Warning: Using size for a discrete variable is not advised.
```



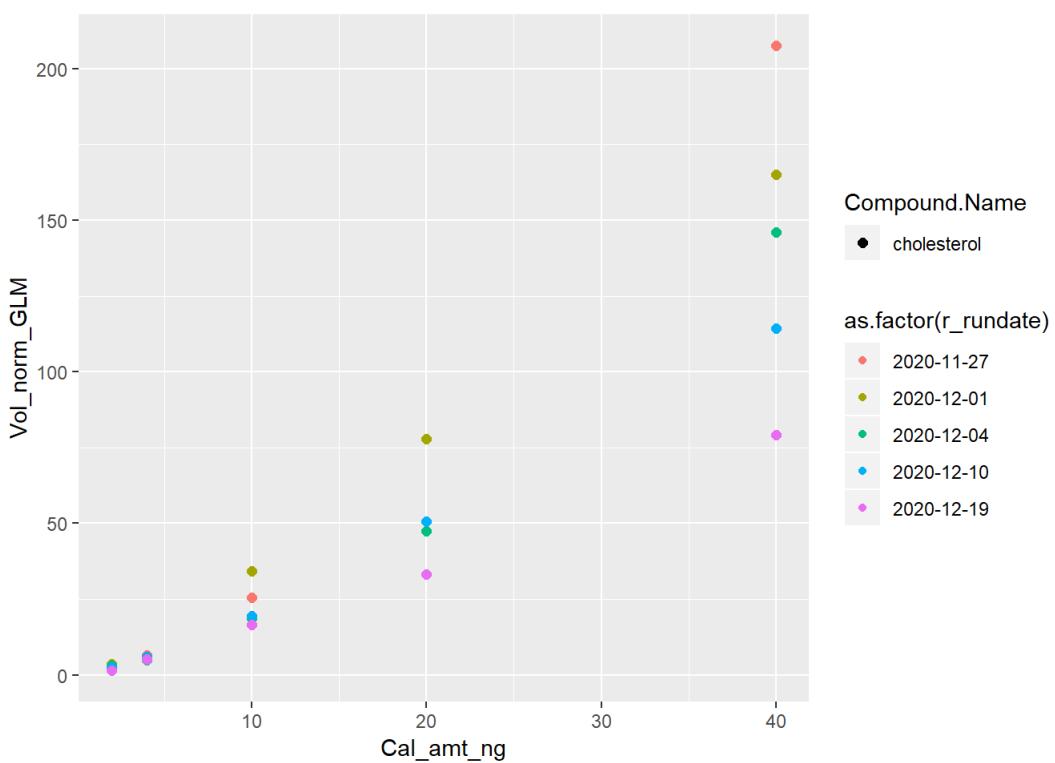
```
## Warning: Using size for a discrete variable is not advised.
```



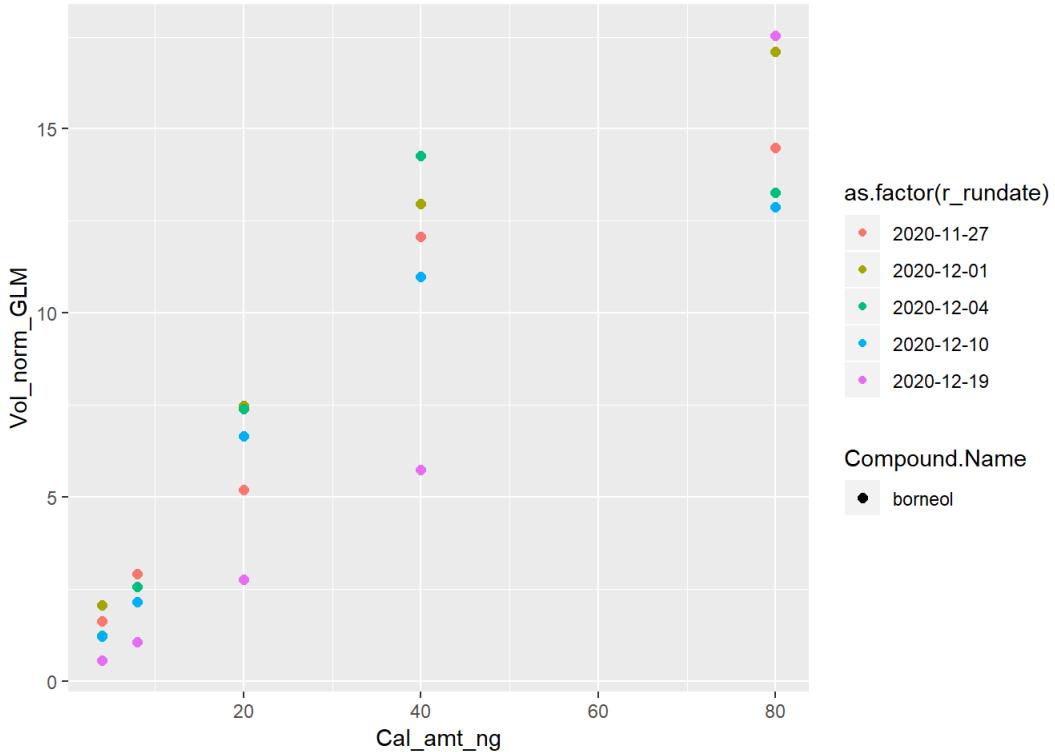
```
## Warning: Using size for a discrete variable is not advised.
```



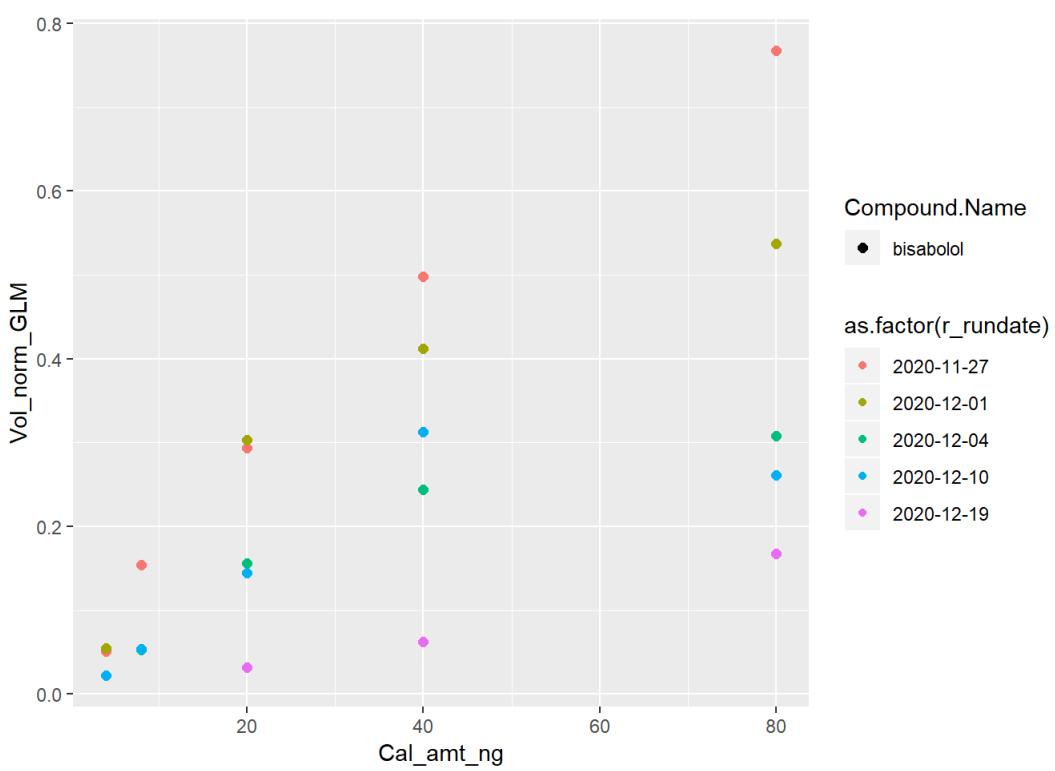
```
## Warning: Using size for a discrete variable is not advised.
```



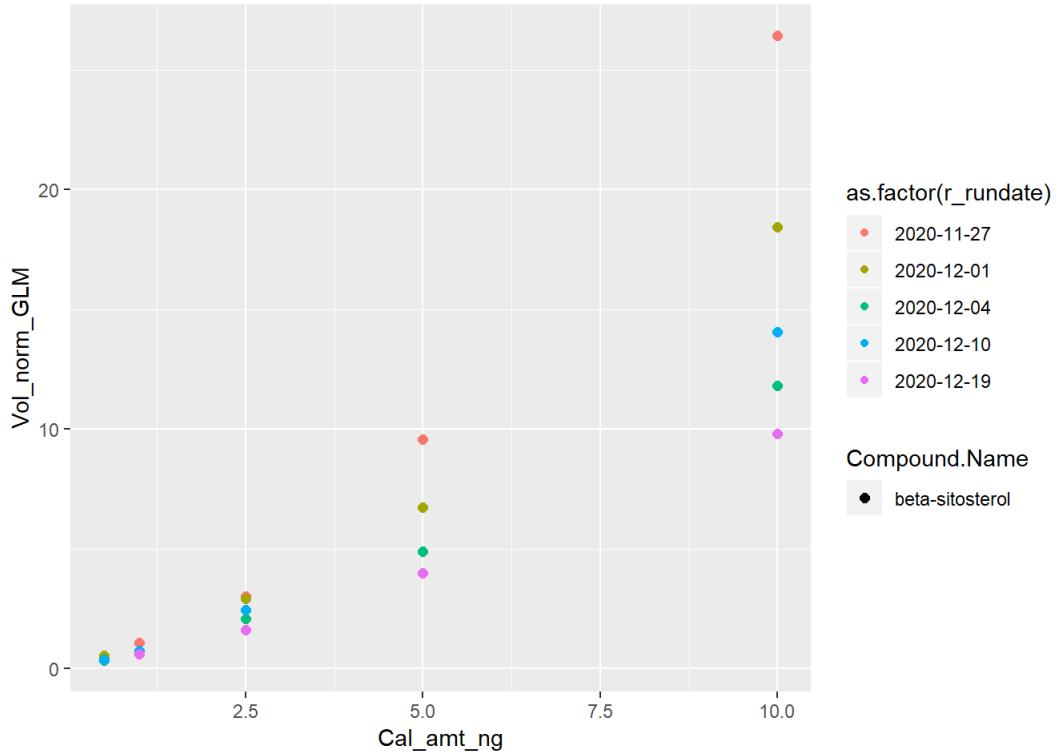
```
## Warning: Using size for a discrete variable is not advised.
```



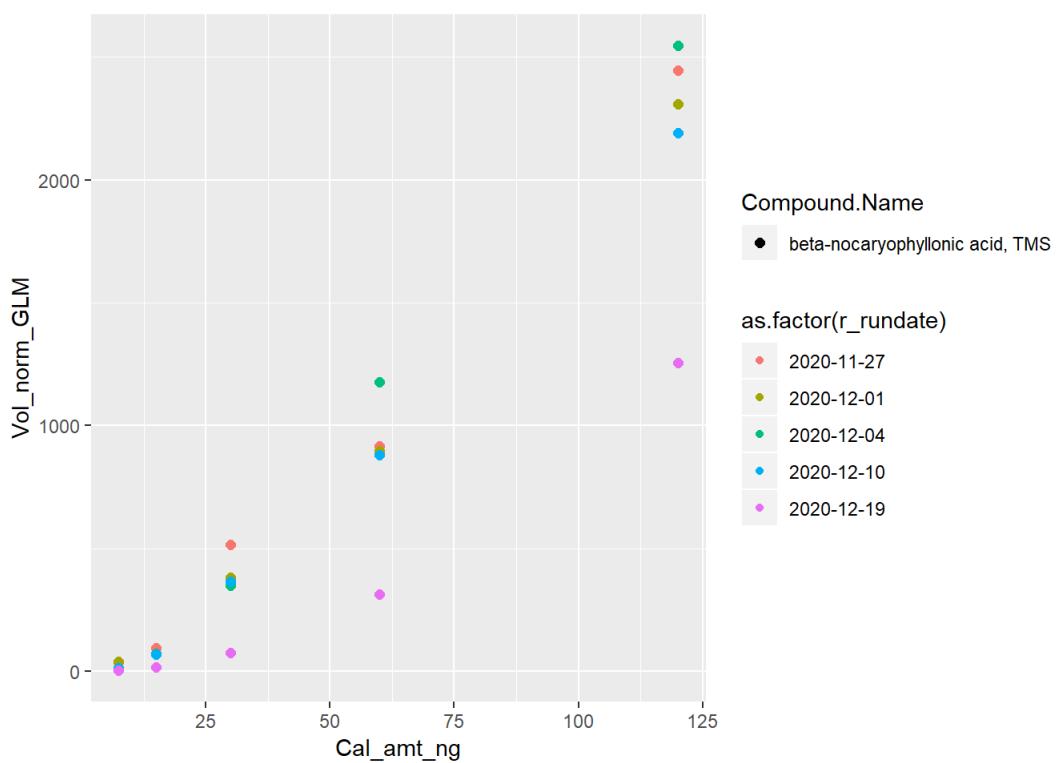
```
## Warning: Using size for a discrete variable is not advised.
```



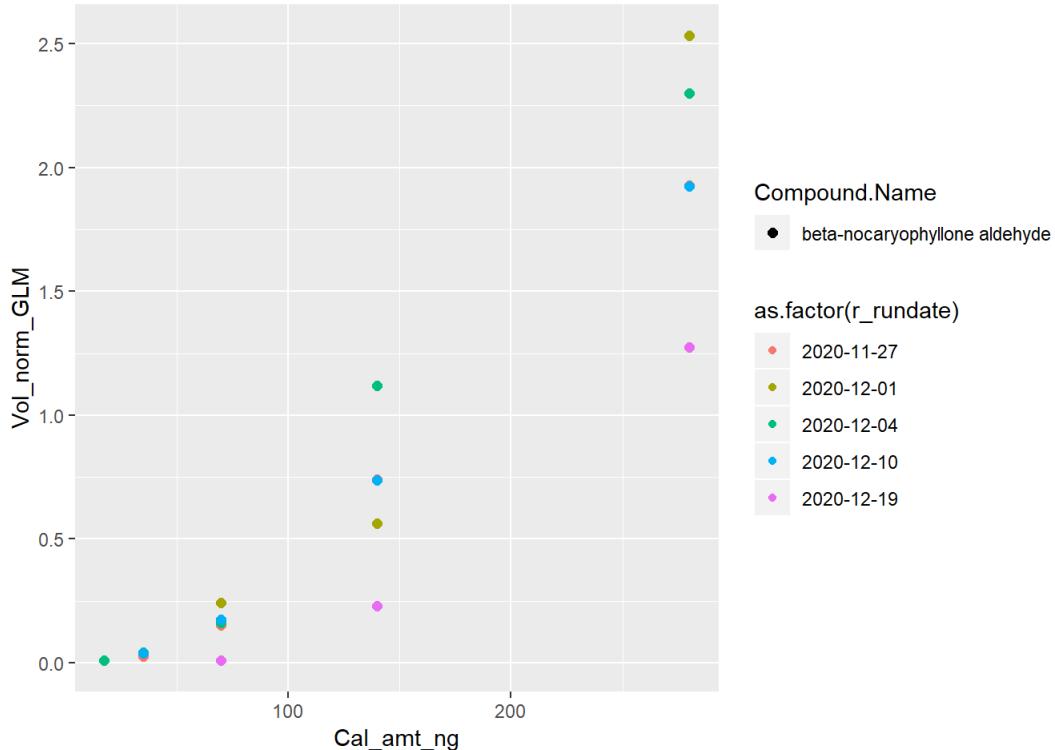
```
## Warning: Using size for a discrete variable is not advised.
```



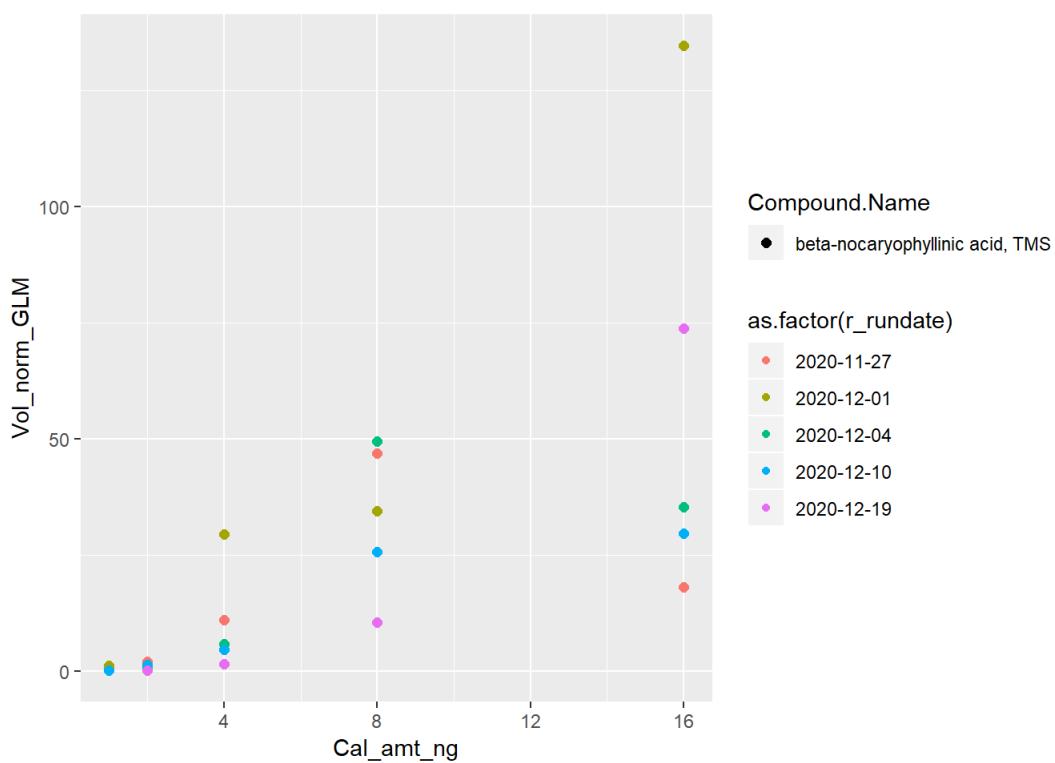
```
## Warning: Using size for a discrete variable is not advised.
```



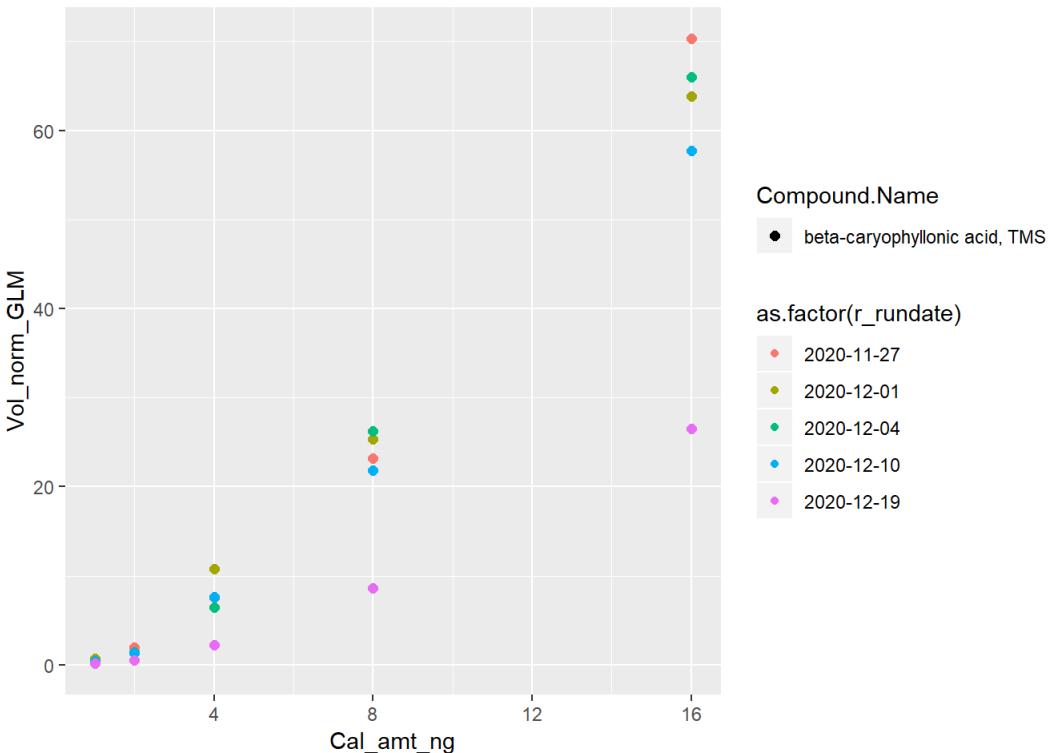
```
## Warning: Using size for a discrete variable is not advised.
```



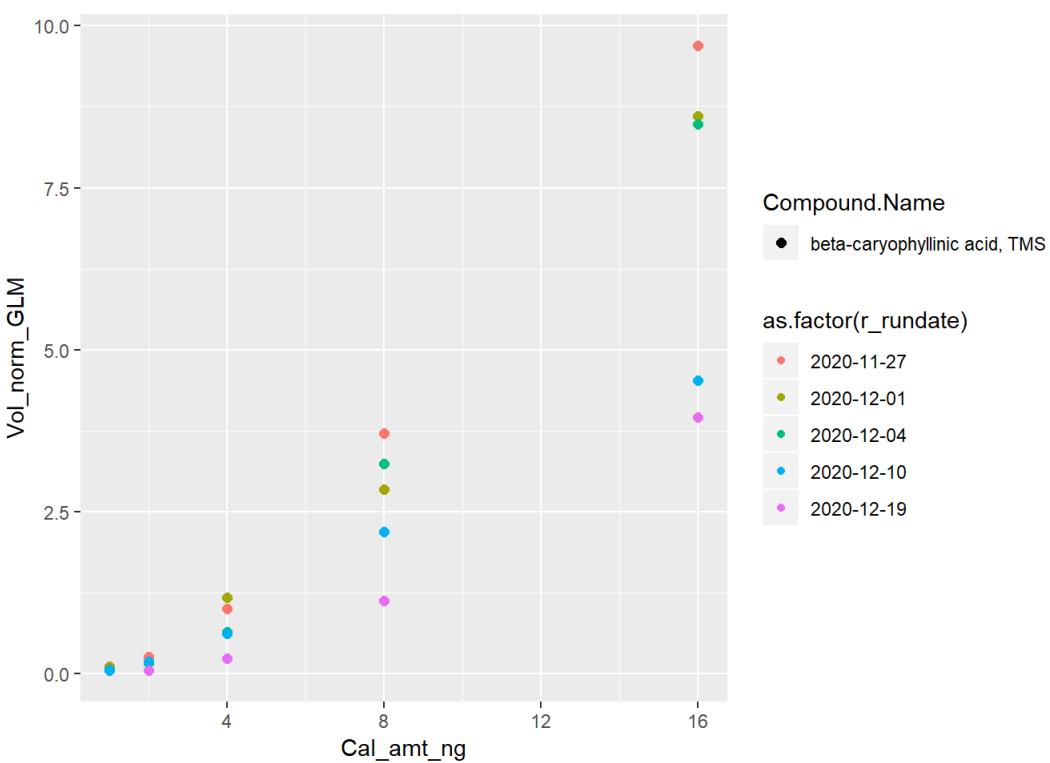
```
## Warning: Using size for a discrete variable is not advised.
```



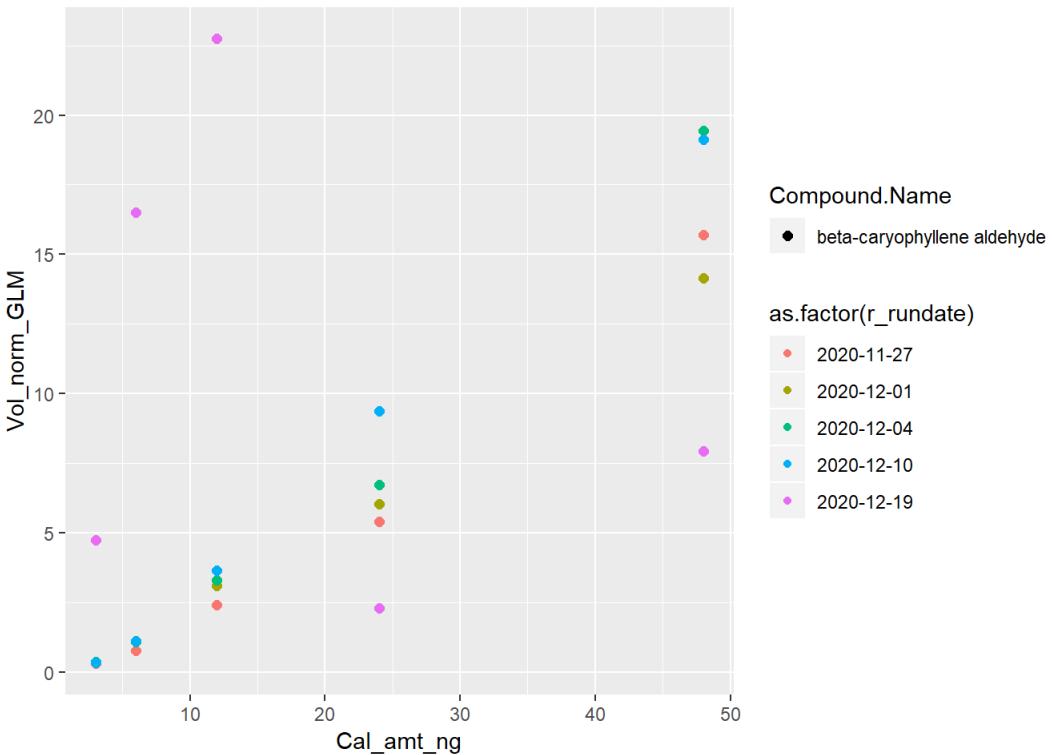
```
## Warning: Using size for a discrete variable is not advised.
```



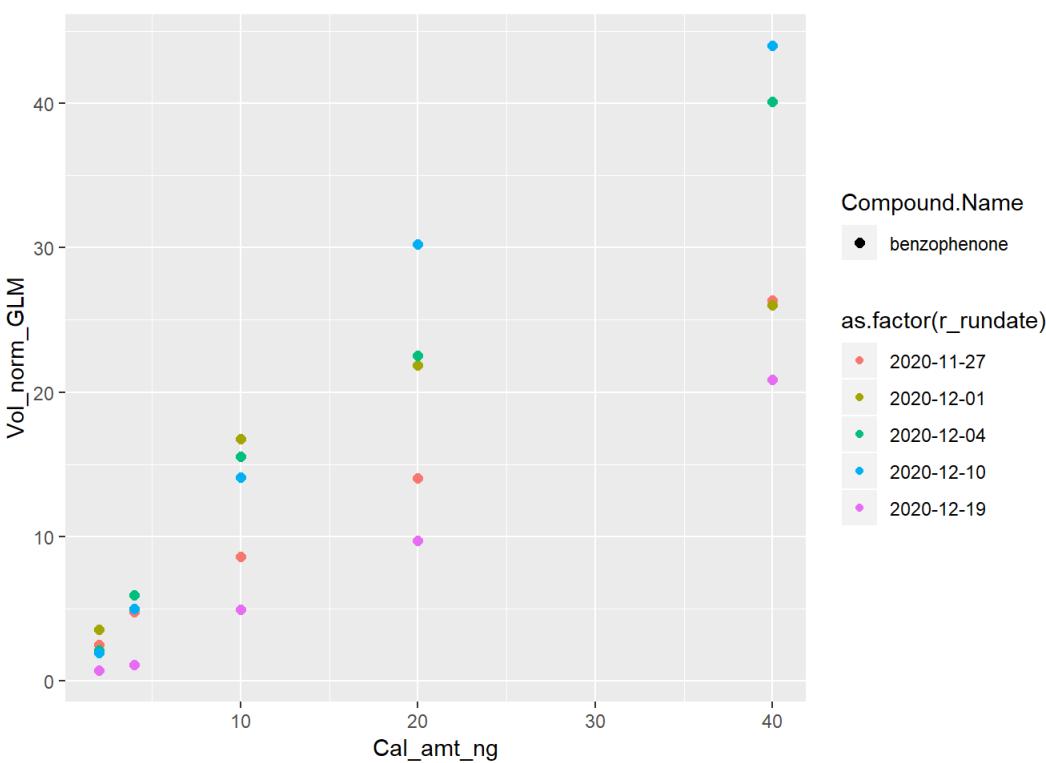
```
## Warning: Using size for a discrete variable is not advised.
```



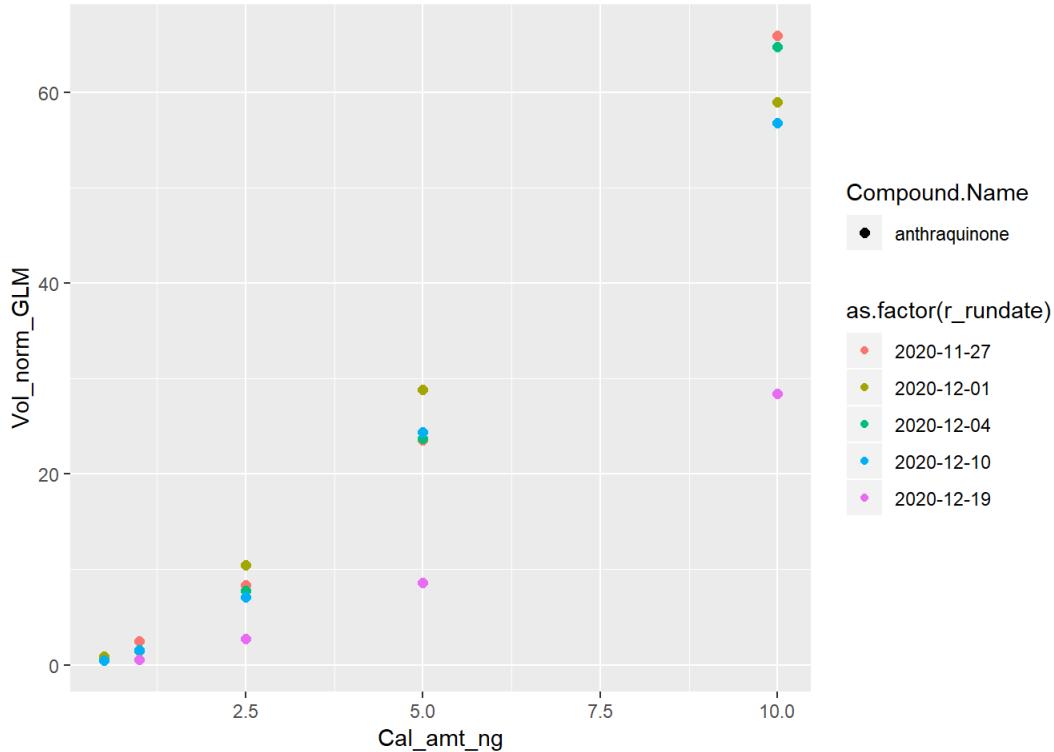
```
## Warning: Using size for a discrete variable is not advised.
```



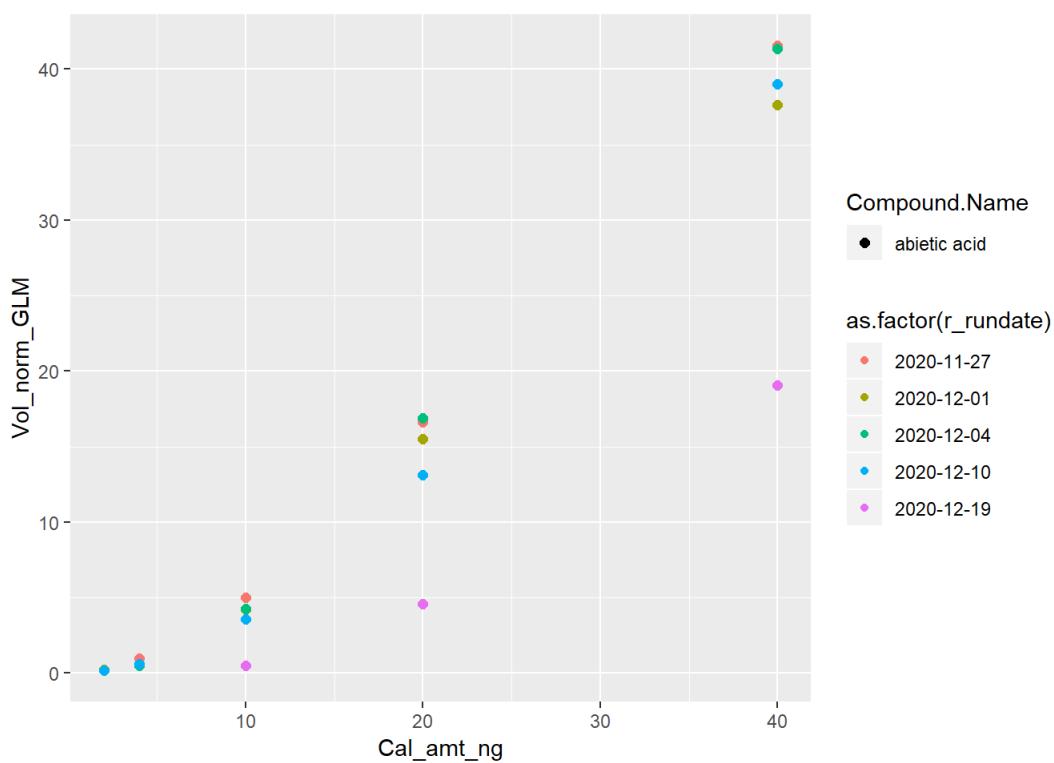
```
## Warning: Using size for a discrete variable is not advised.
```



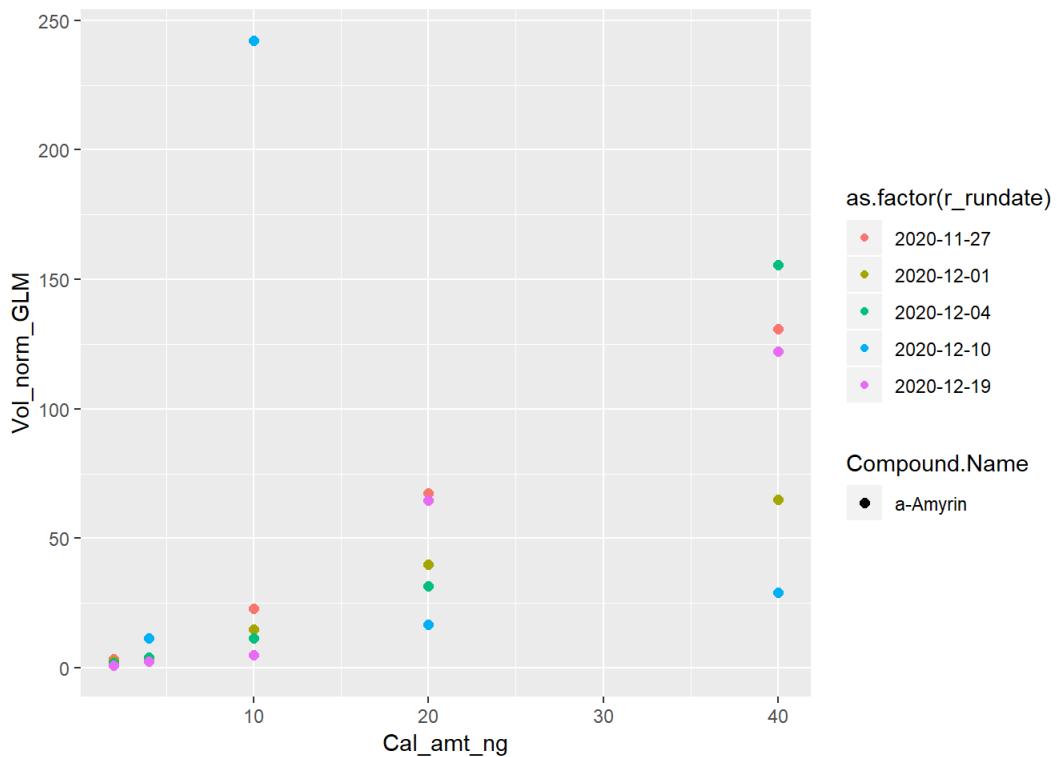
```
## Warning: Using size for a discrete variable is not advised.
```



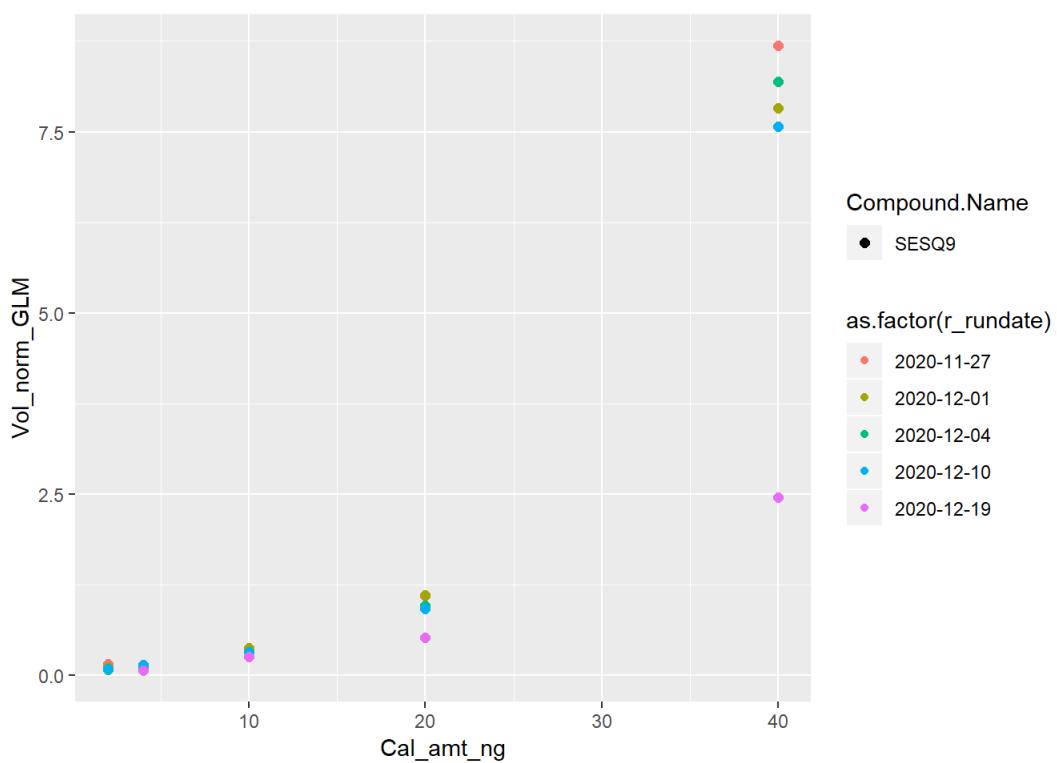
```
## Warning: Using size for a discrete variable is not advised.
```



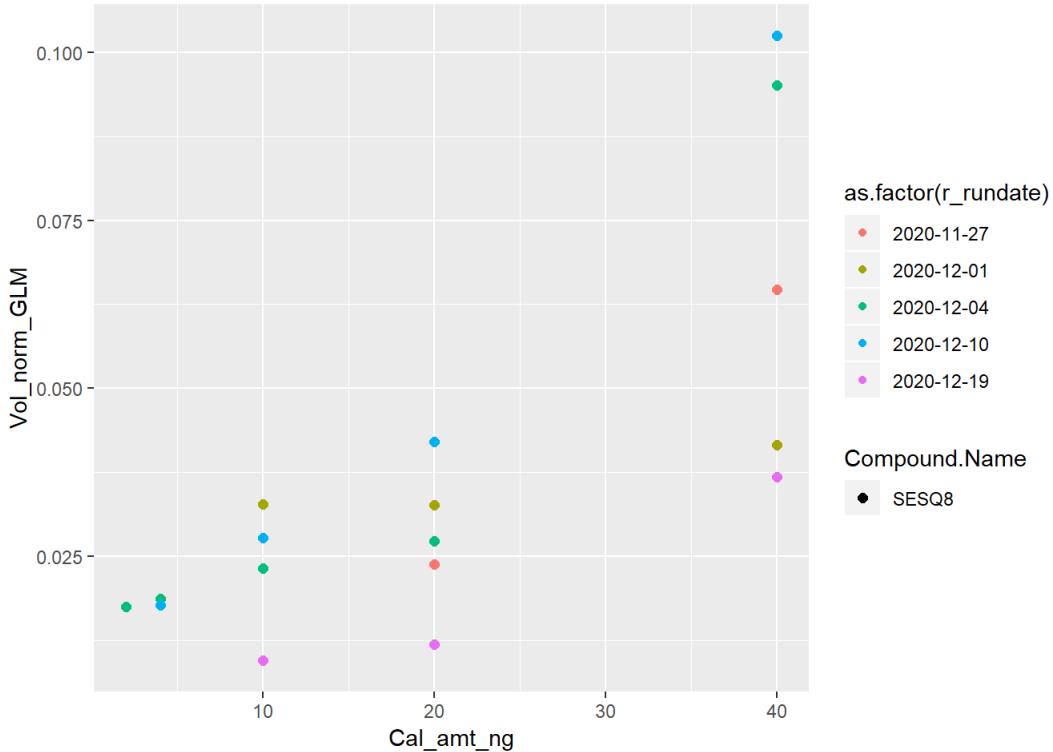
```
## Warning: Using size for a discrete variable is not advised.
```



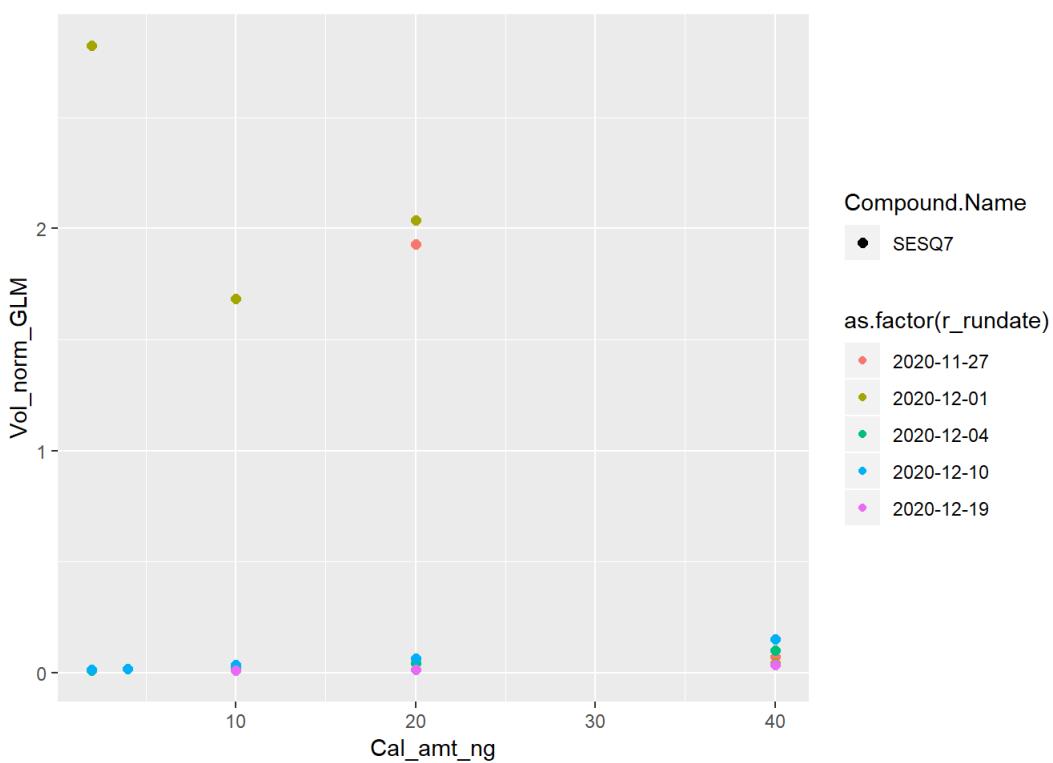
```
## Warning: Using size for a discrete variable is not advised.
```



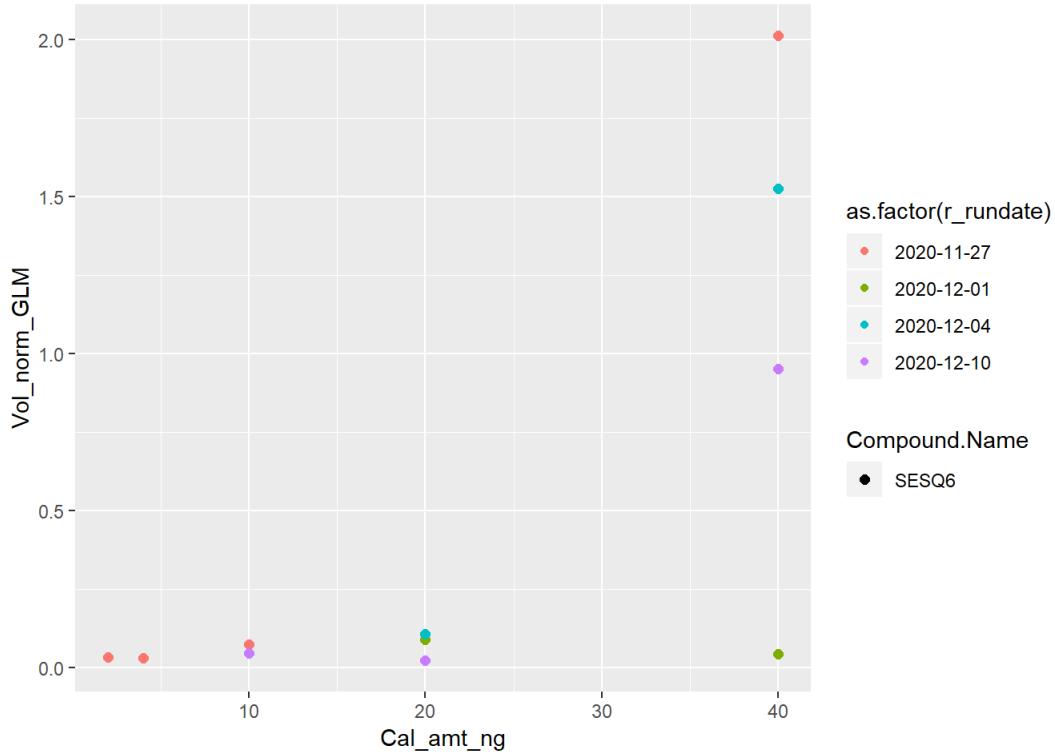
```
## Warning: Using size for a discrete variable is not advised.
```



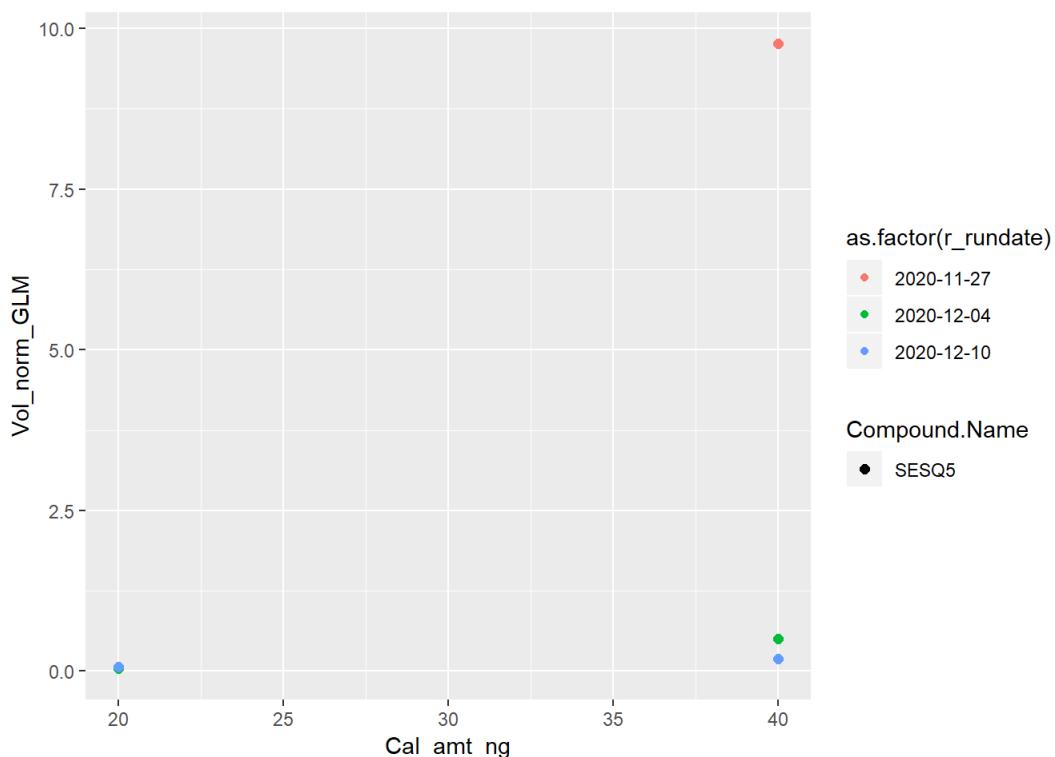
```
## Warning: Using size for a discrete variable is not advised.
```



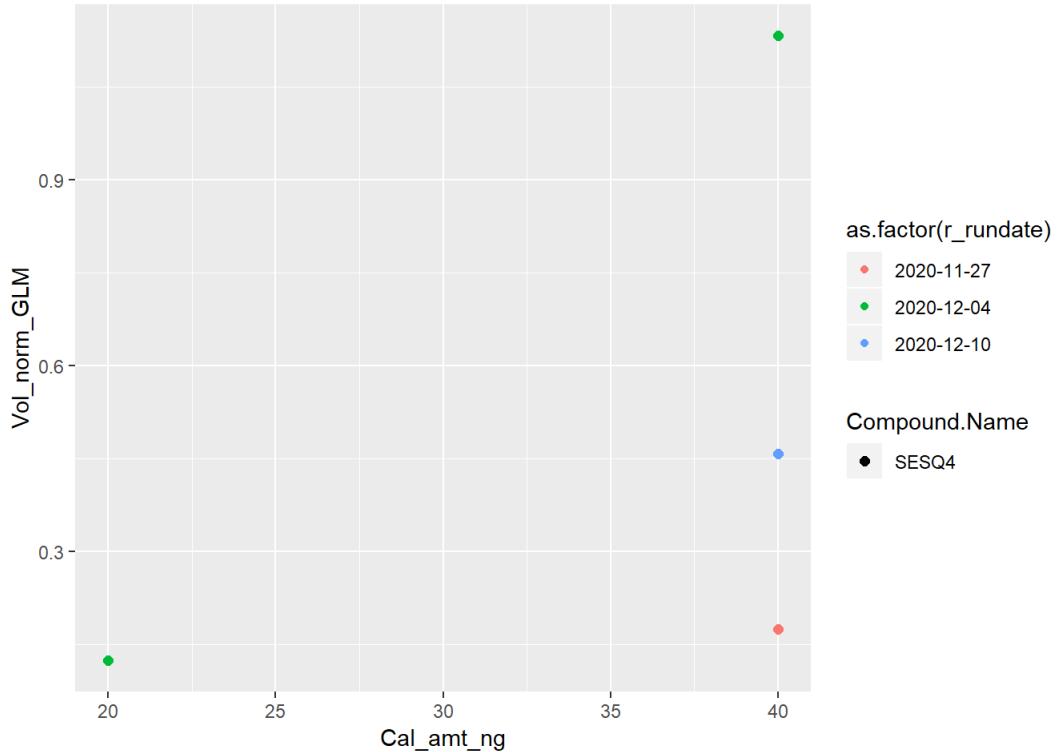
```
## Warning: Using size for a discrete variable is not advised.
```



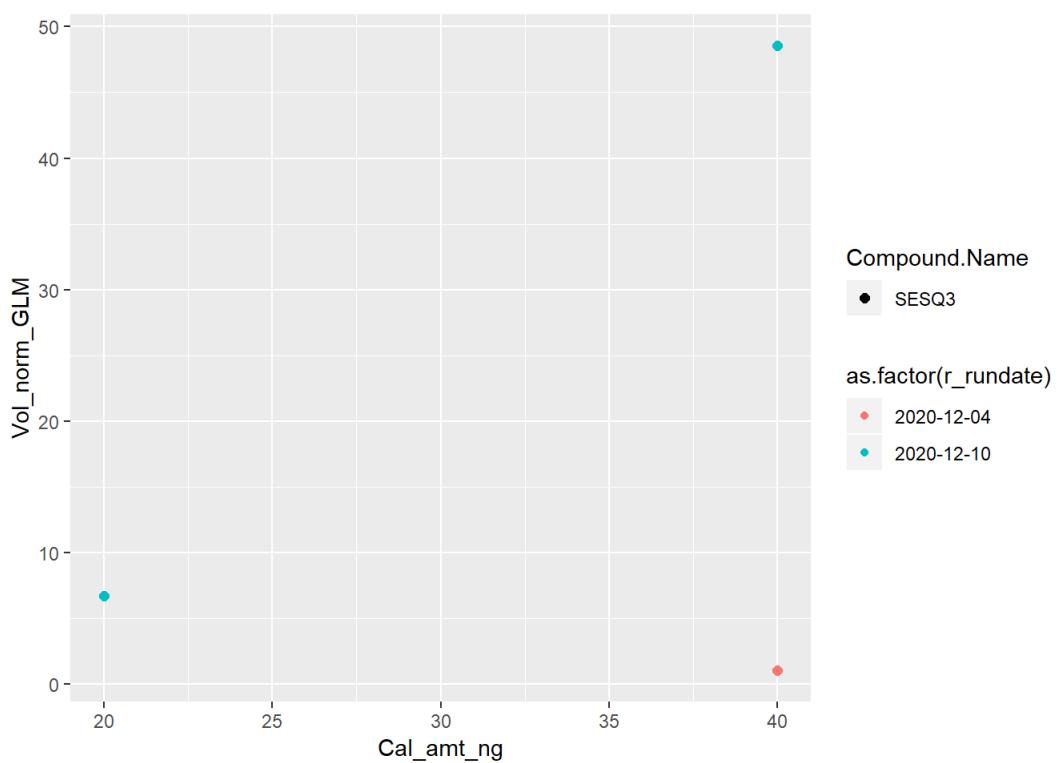
```
## Warning: Using size for a discrete variable is not advised.
```



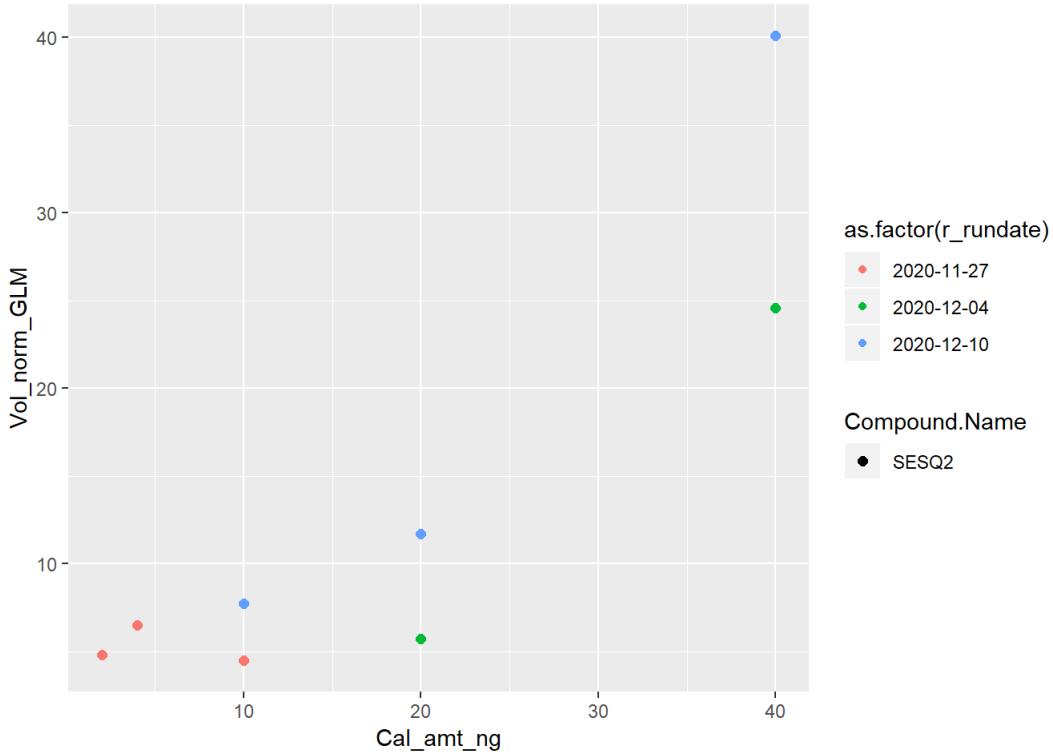
```
## Warning: Using size for a discrete variable is not advised.
```



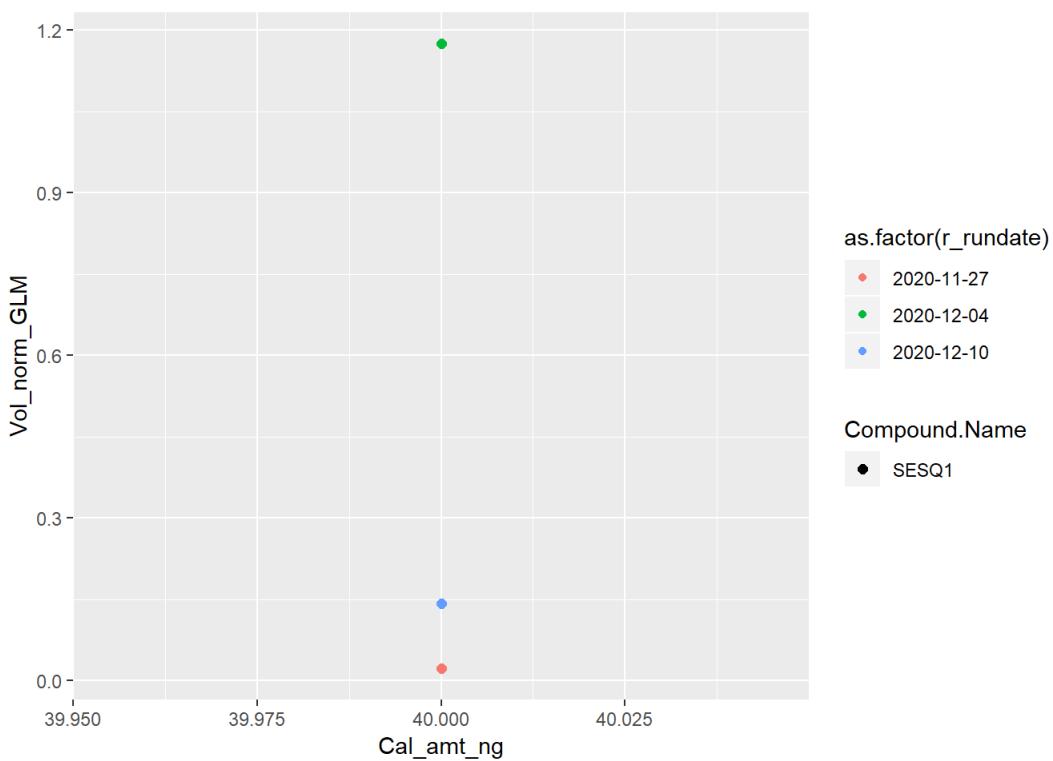
```
## Warning: Using size for a discrete variable is not advised.
```



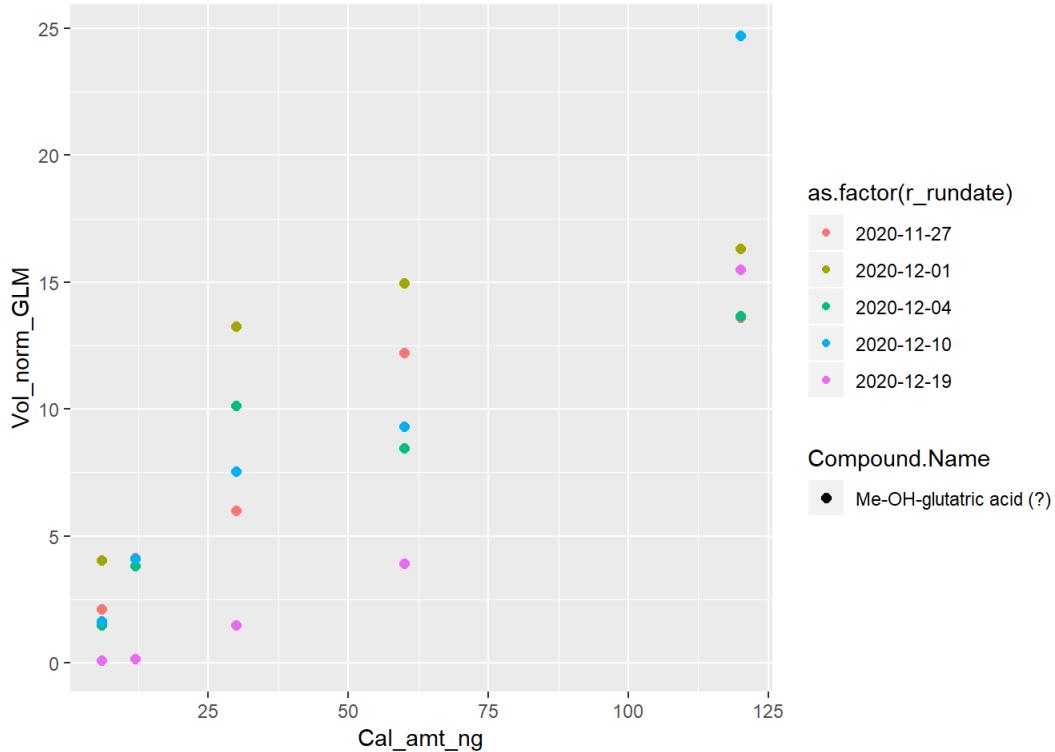
```
## Warning: Using size for a discrete variable is not advised.
```



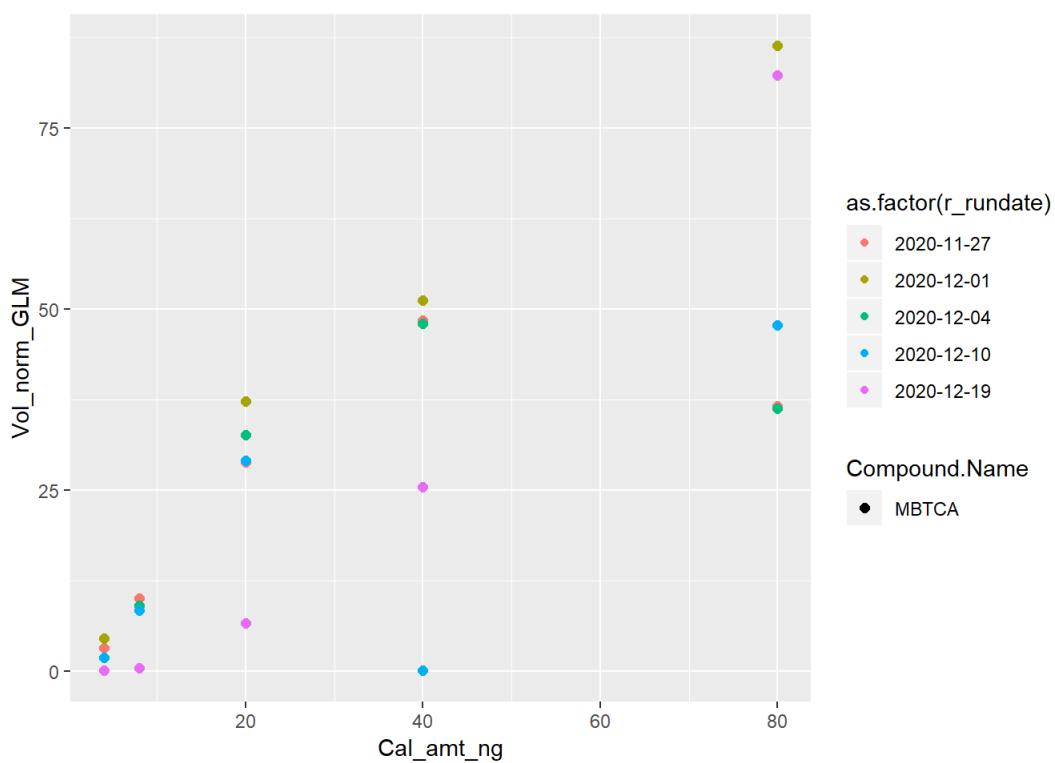
```
## Warning: Using size for a discrete variable is not advised.
```



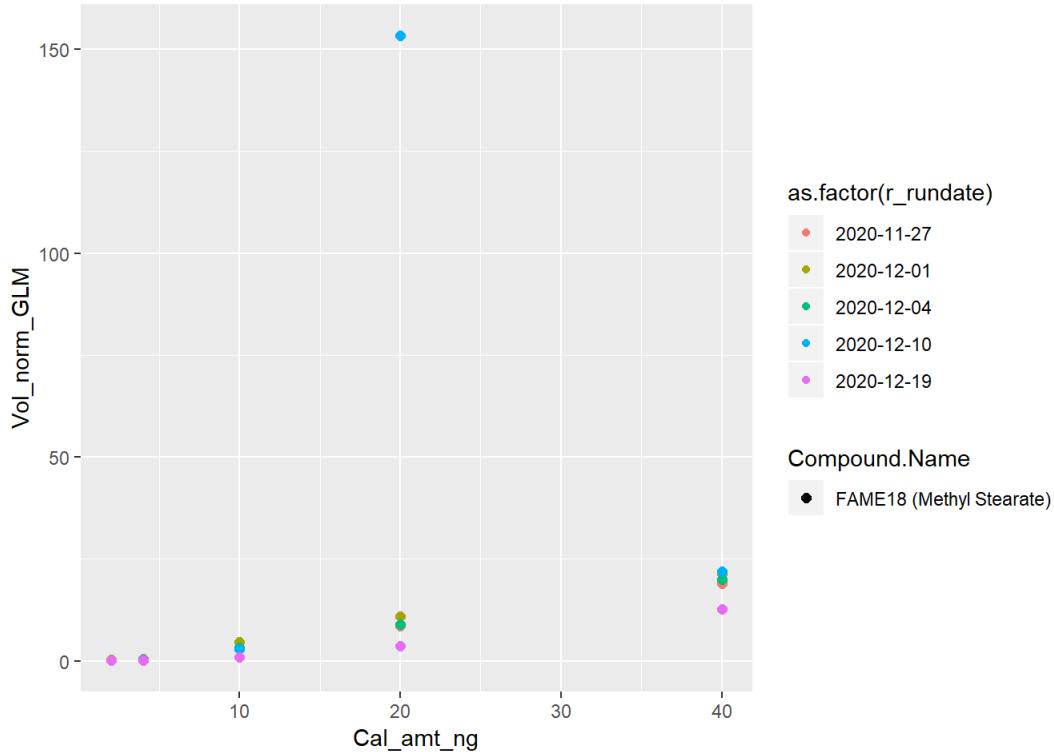
```
## Warning: Using size for a discrete variable is not advised.
```



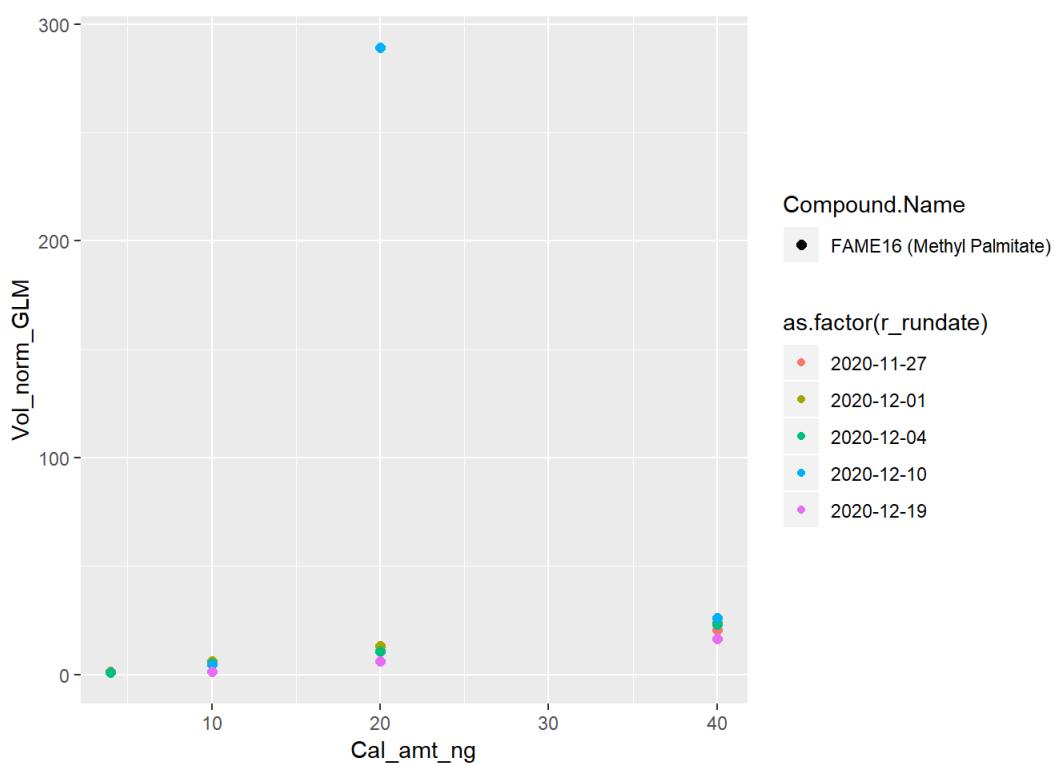
```
## Warning: Using size for a discrete variable is not advised.
```



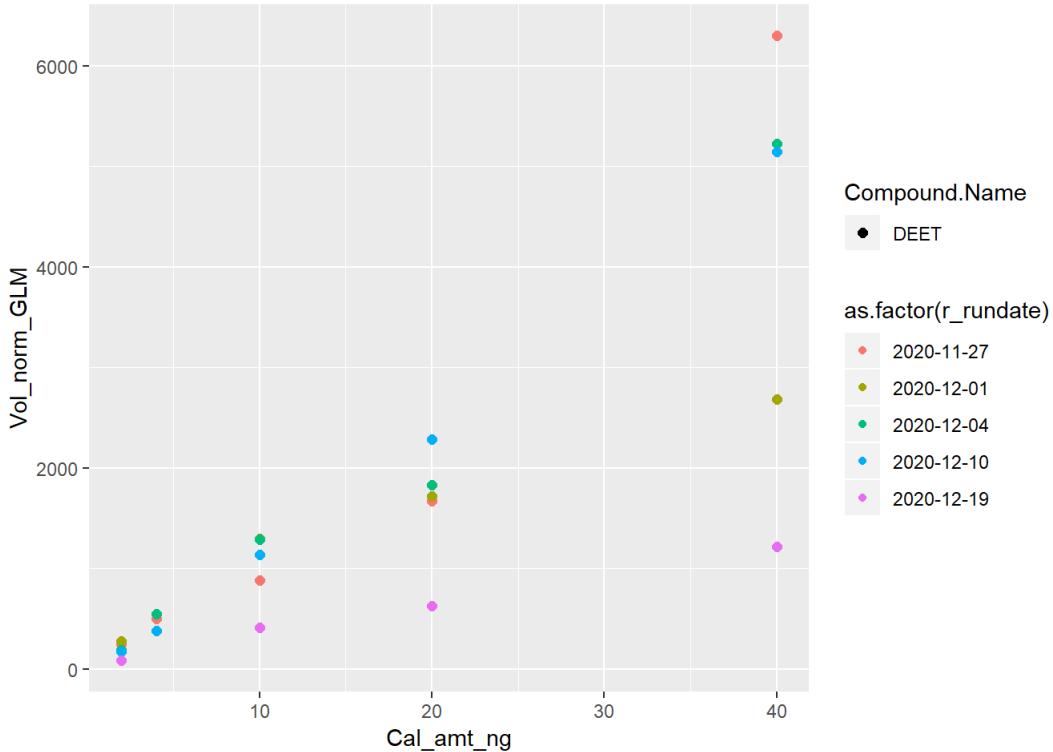
```
## Warning: Using size for a discrete variable is not advised.
```



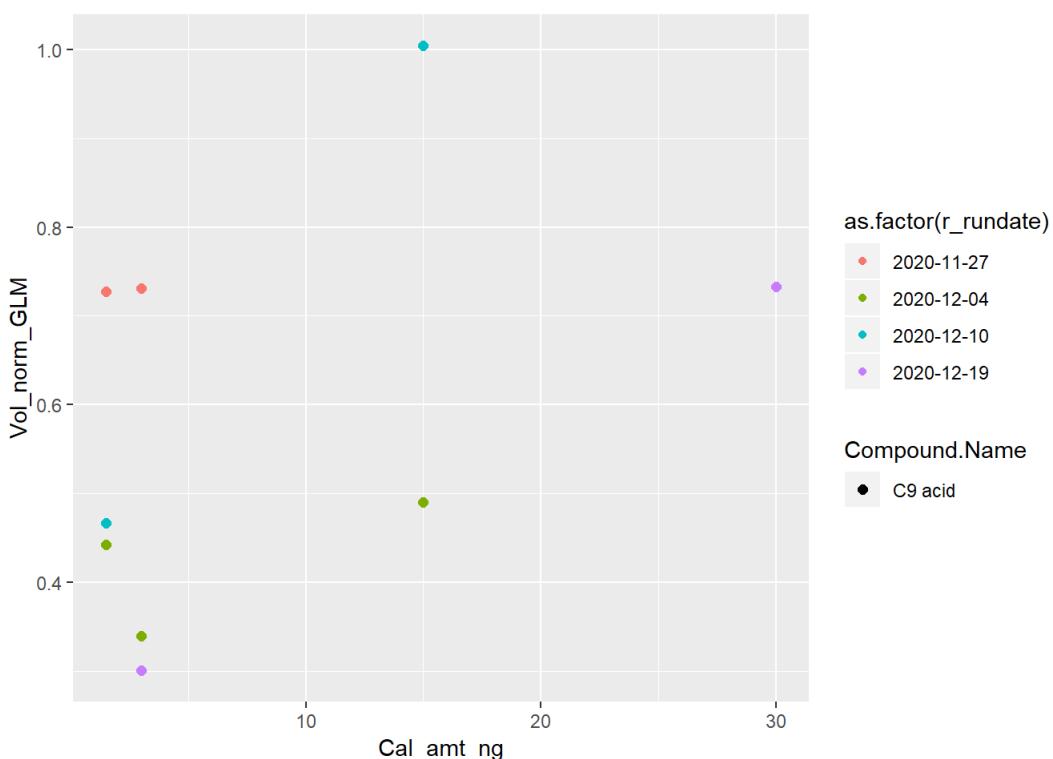
```
## Warning: Using size for a discrete variable is not advised.
```



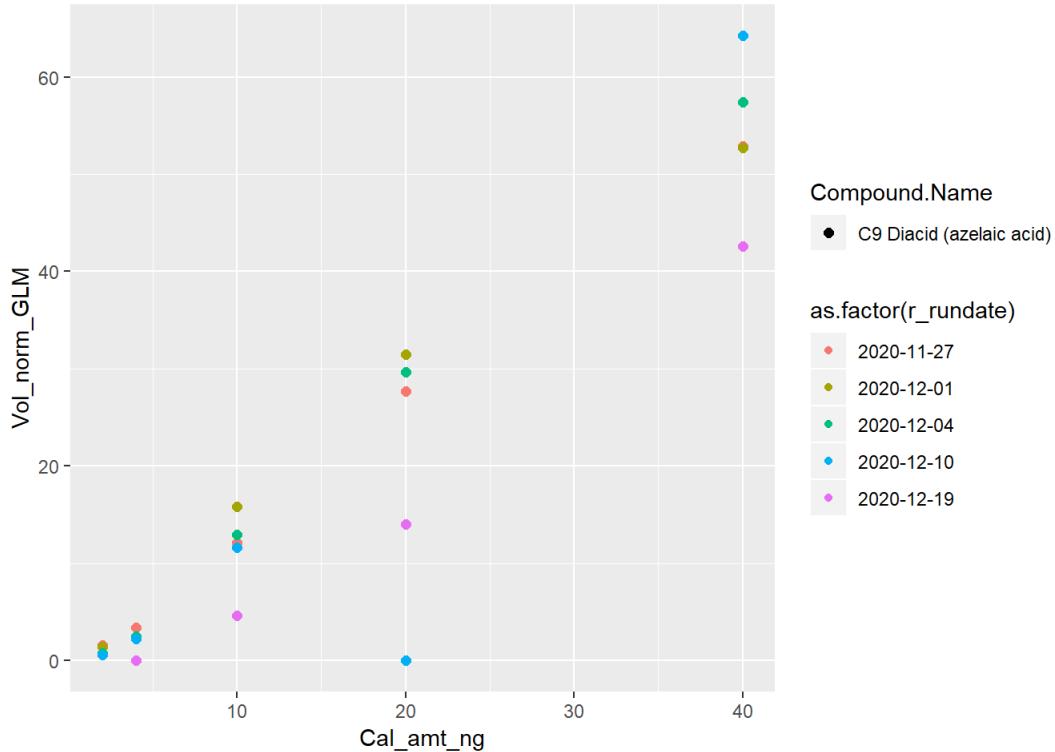
```
## Warning: Using size for a discrete variable is not advised.
```



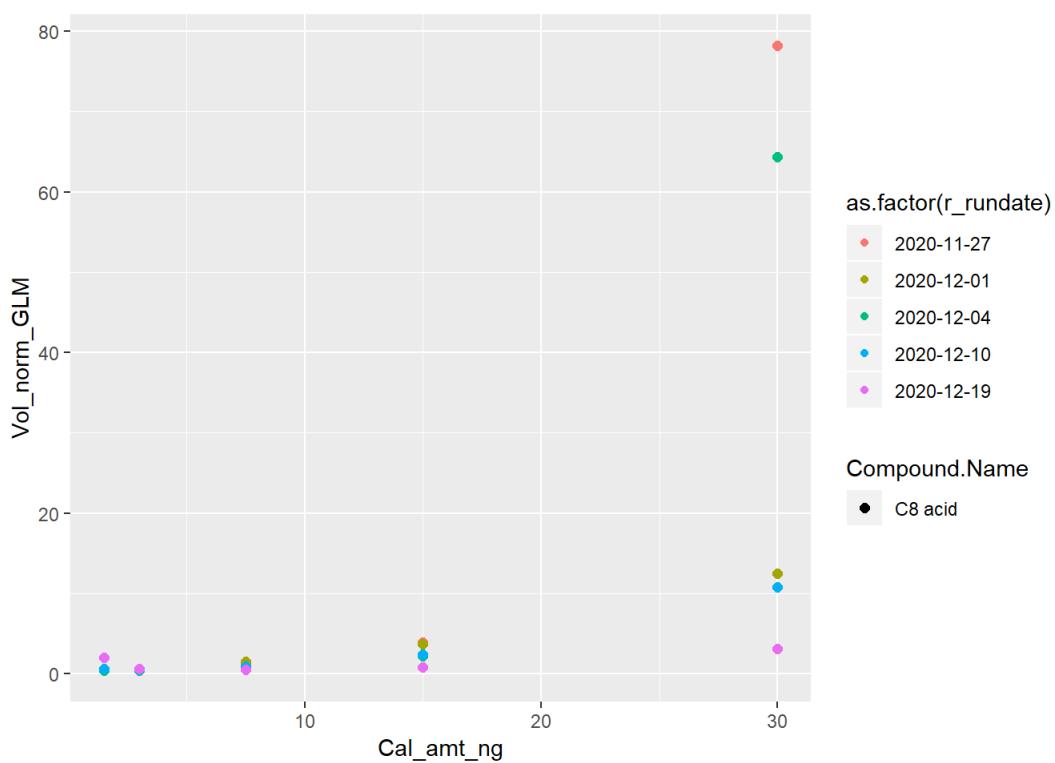
```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```

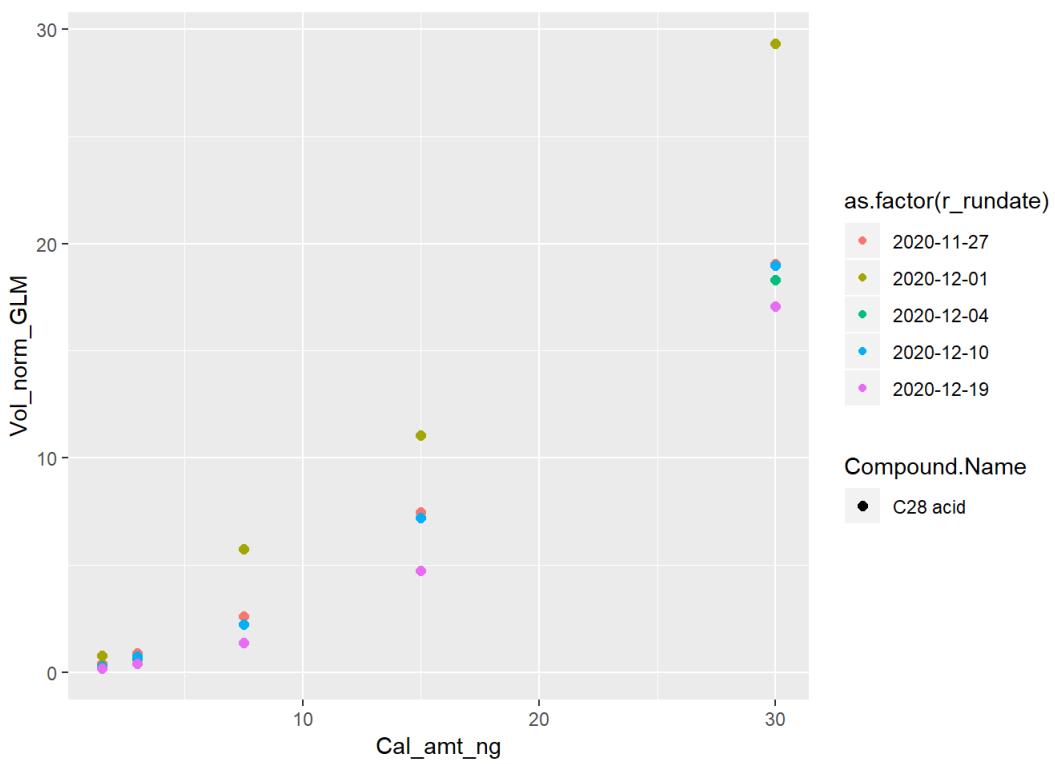


```
## Warning: Using size for a discrete variable is not advised.
```

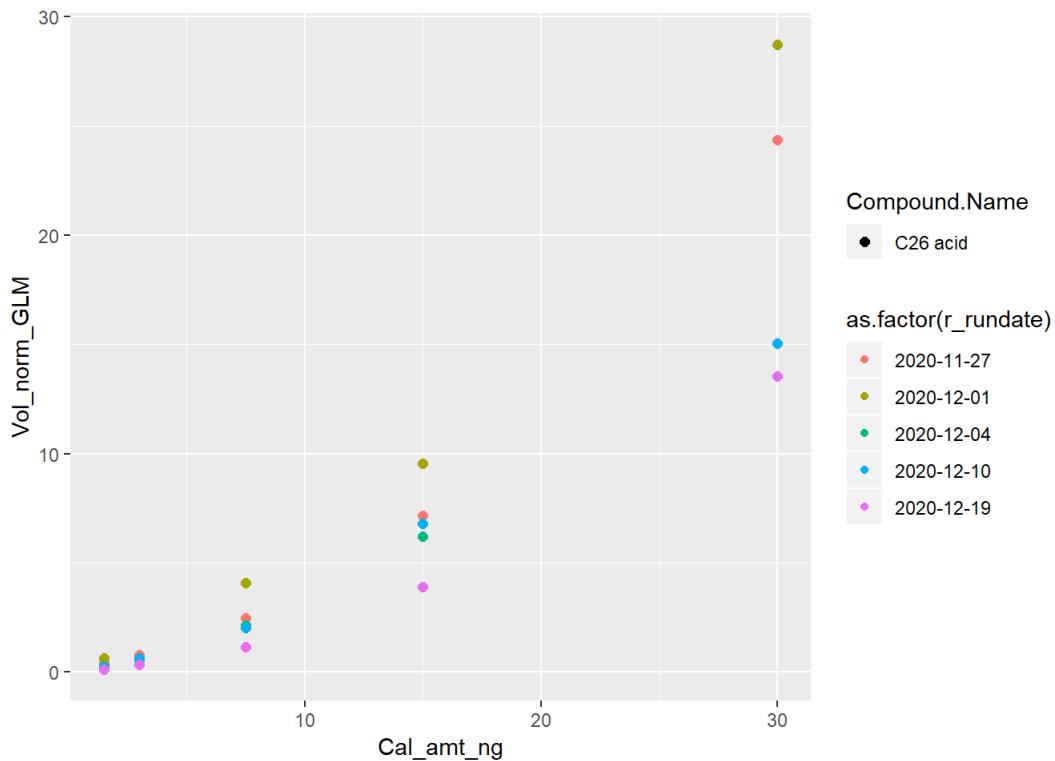
Vol_norm_GLM

Cal_amt_ng

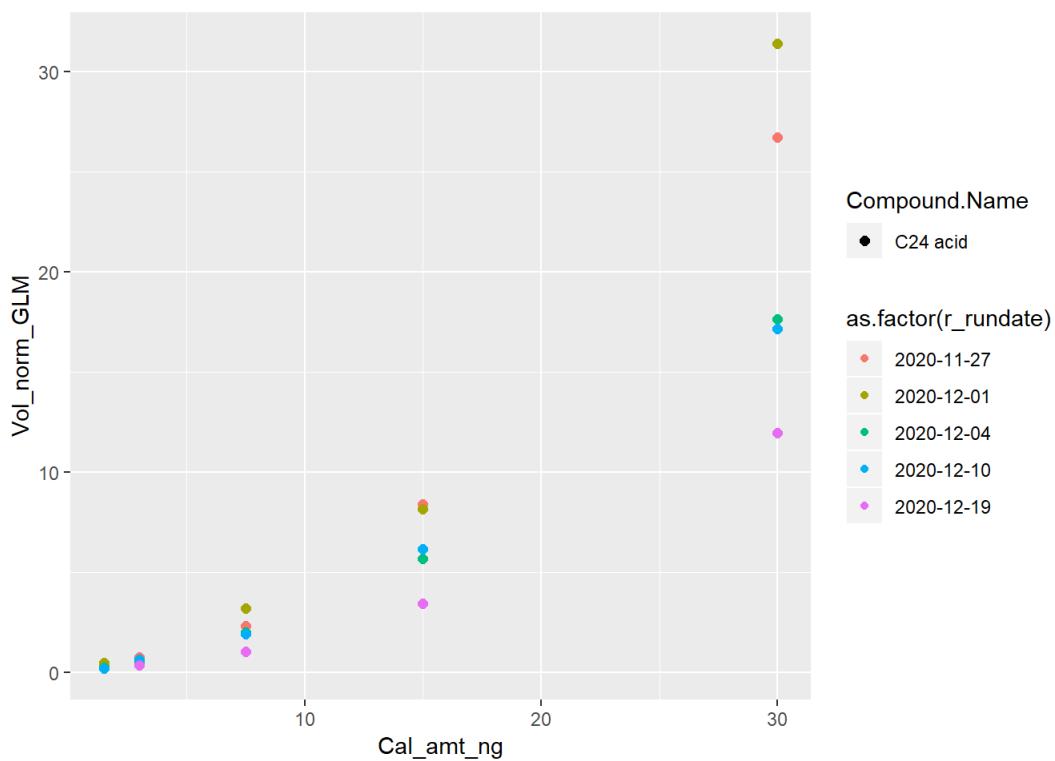
```
## Warning: Using size for a discrete variable is not advised.
```



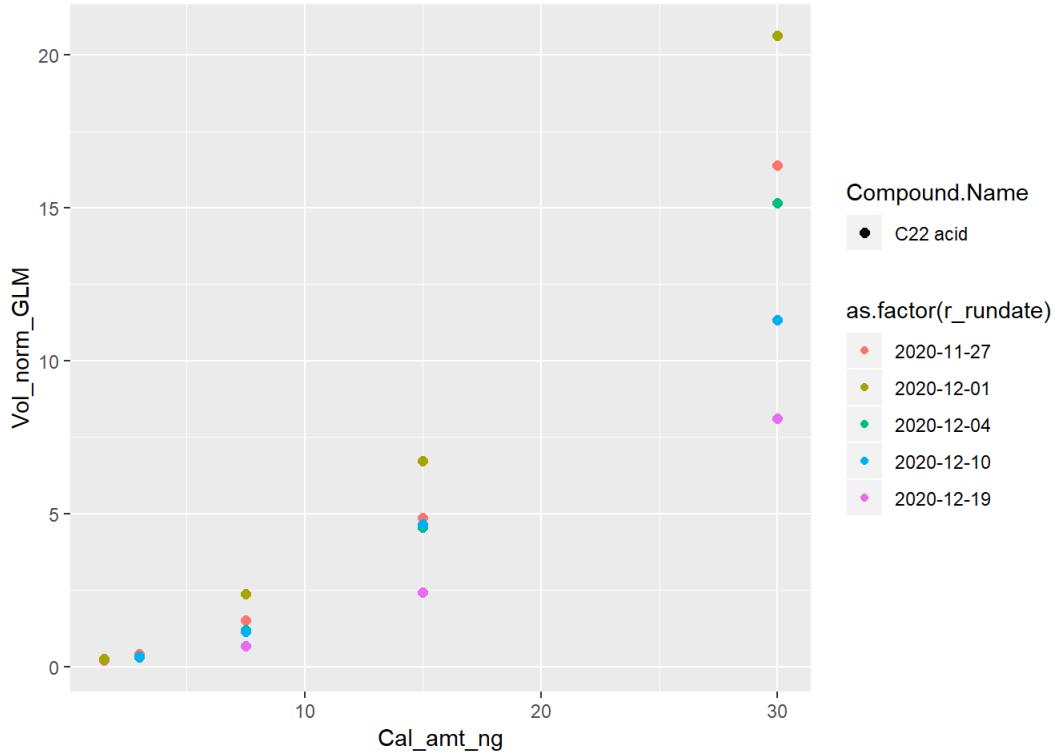
```
## Warning: Using size for a discrete variable is not advised.
```



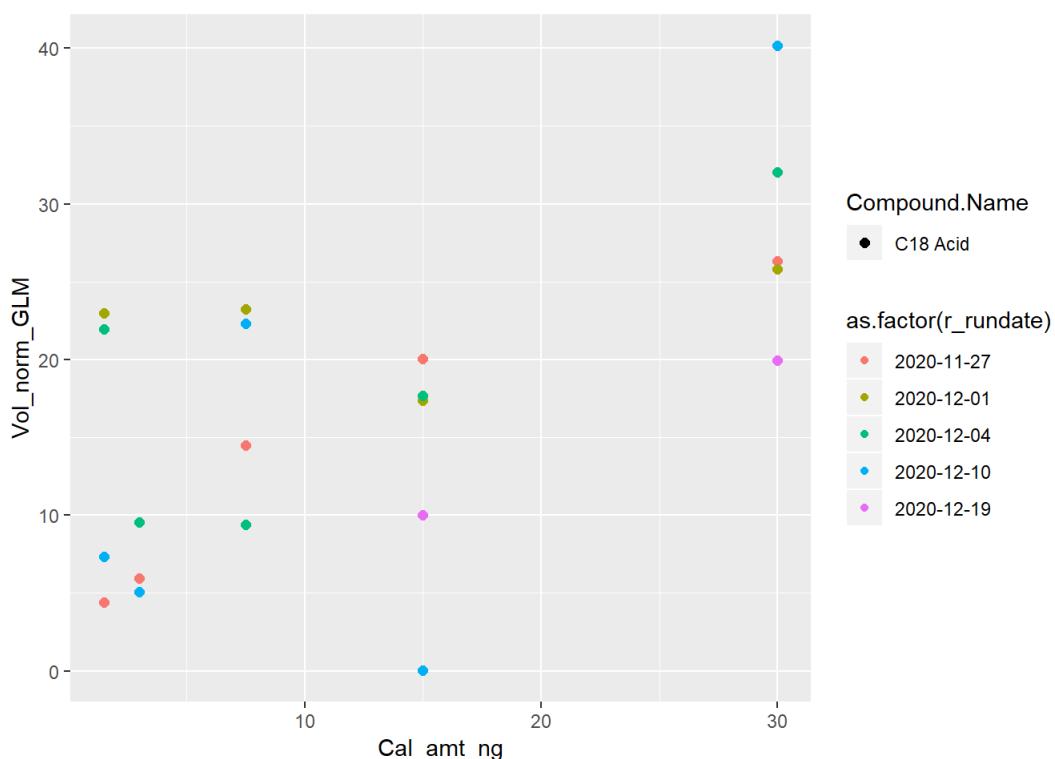
```
## Warning: Using size for a discrete variable is not advised.
```



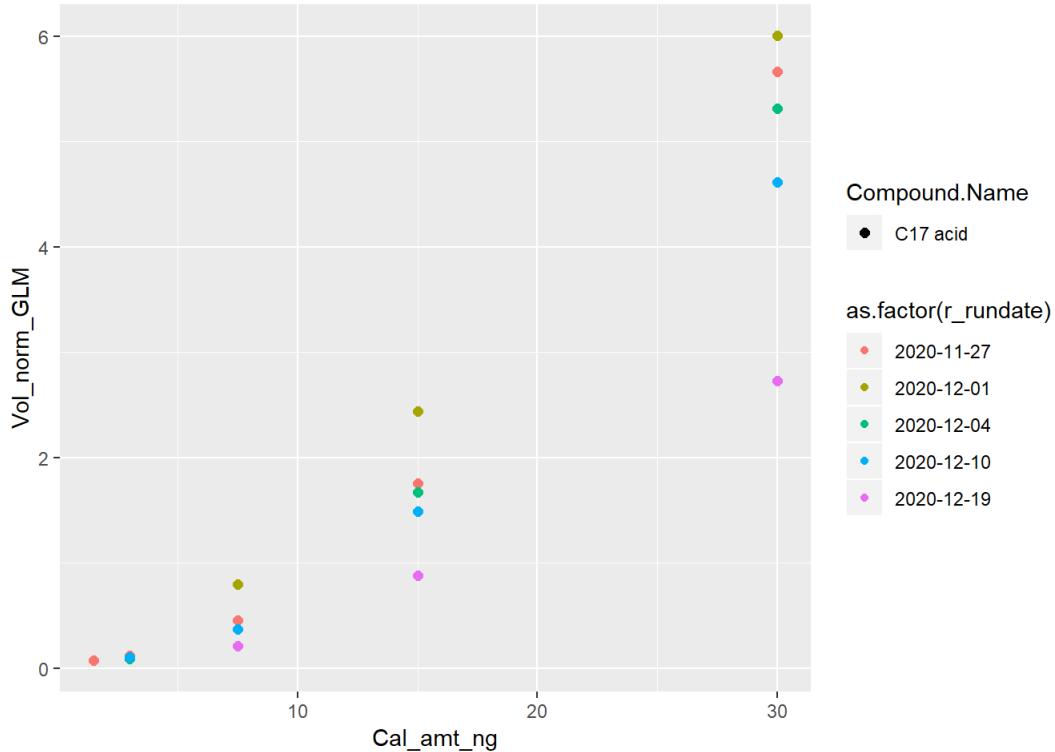
```
## Warning: Using size for a discrete variable is not advised.
```



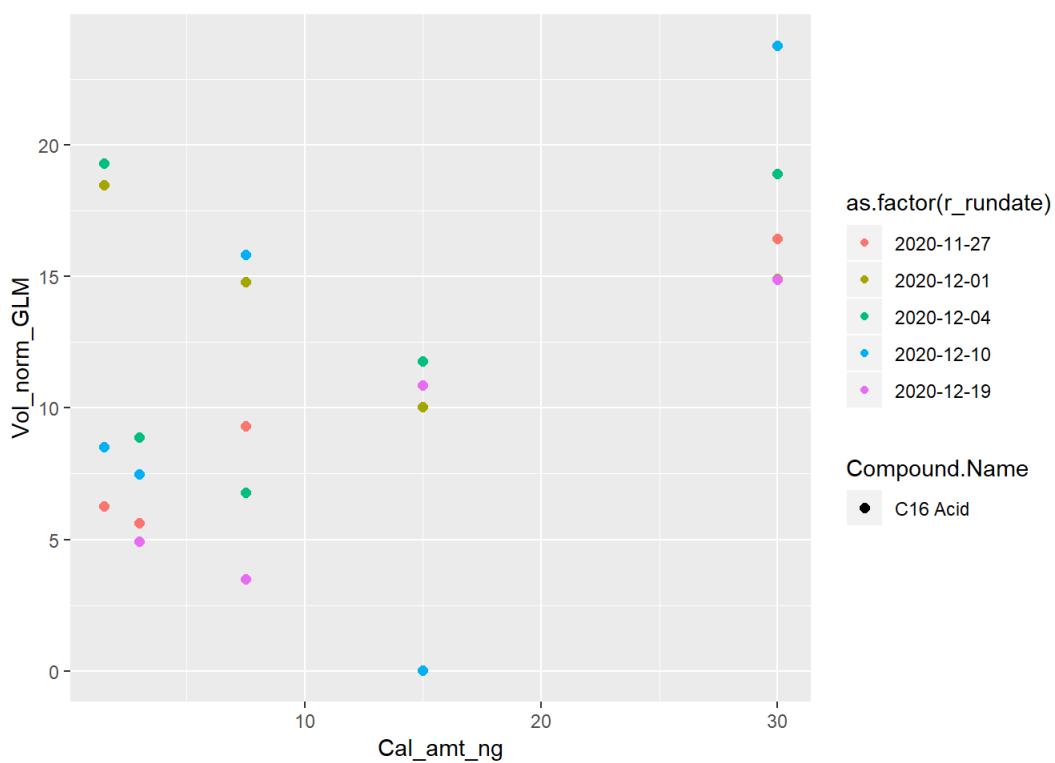
```
## Warning: Using size for a discrete variable is not advised.
```



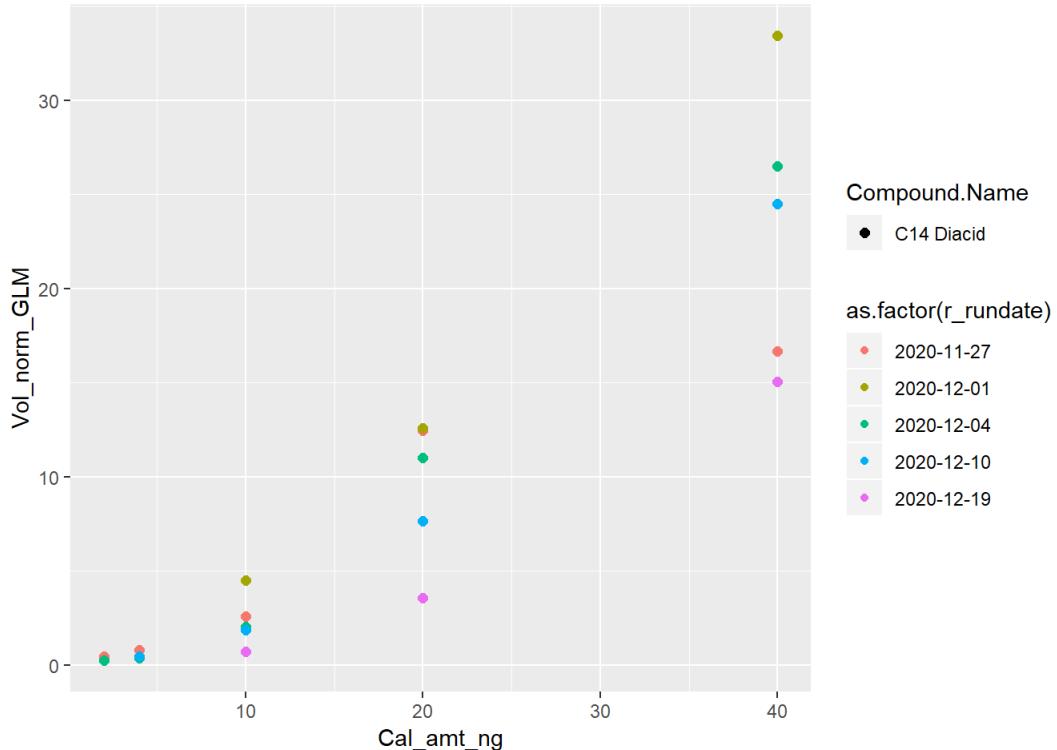
```
## Warning: Using size for a discrete variable is not advised.
```



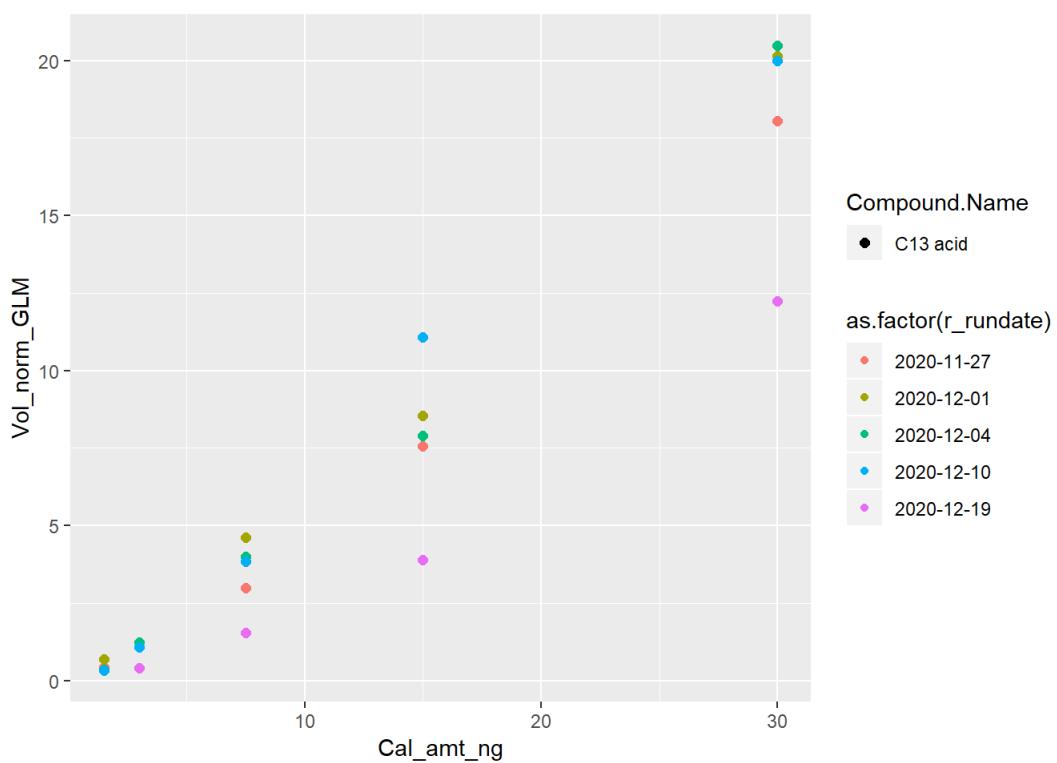
```
## Warning: Using size for a discrete variable is not advised.
```



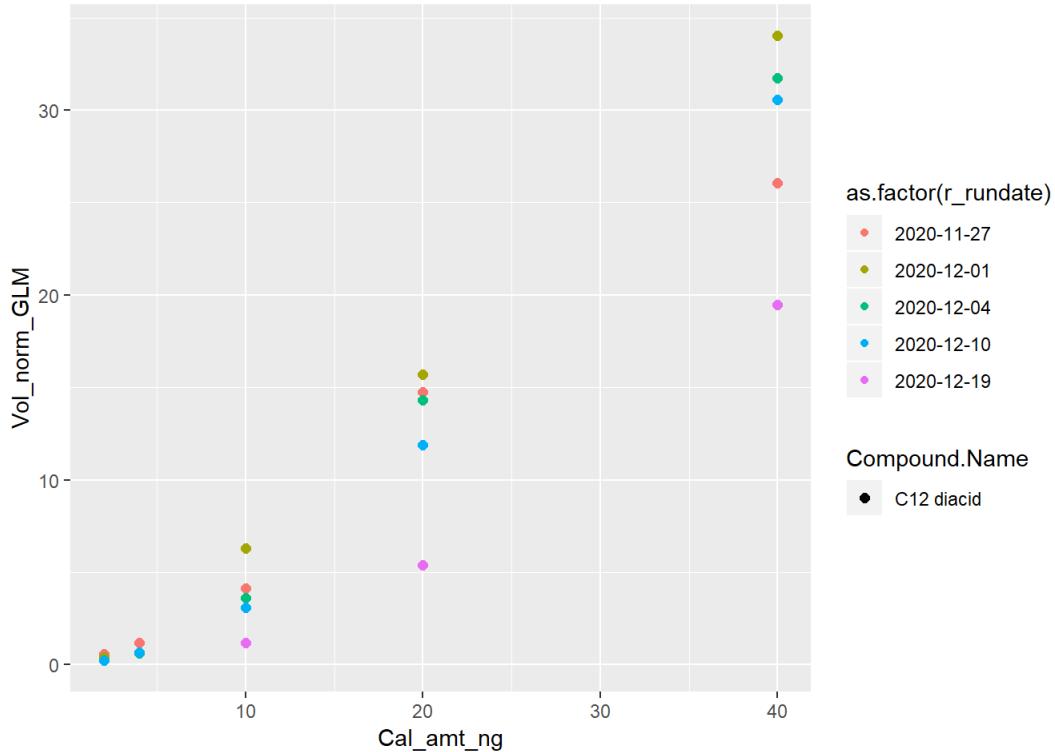
```
## Warning: Using size for a discrete variable is not advised.
```



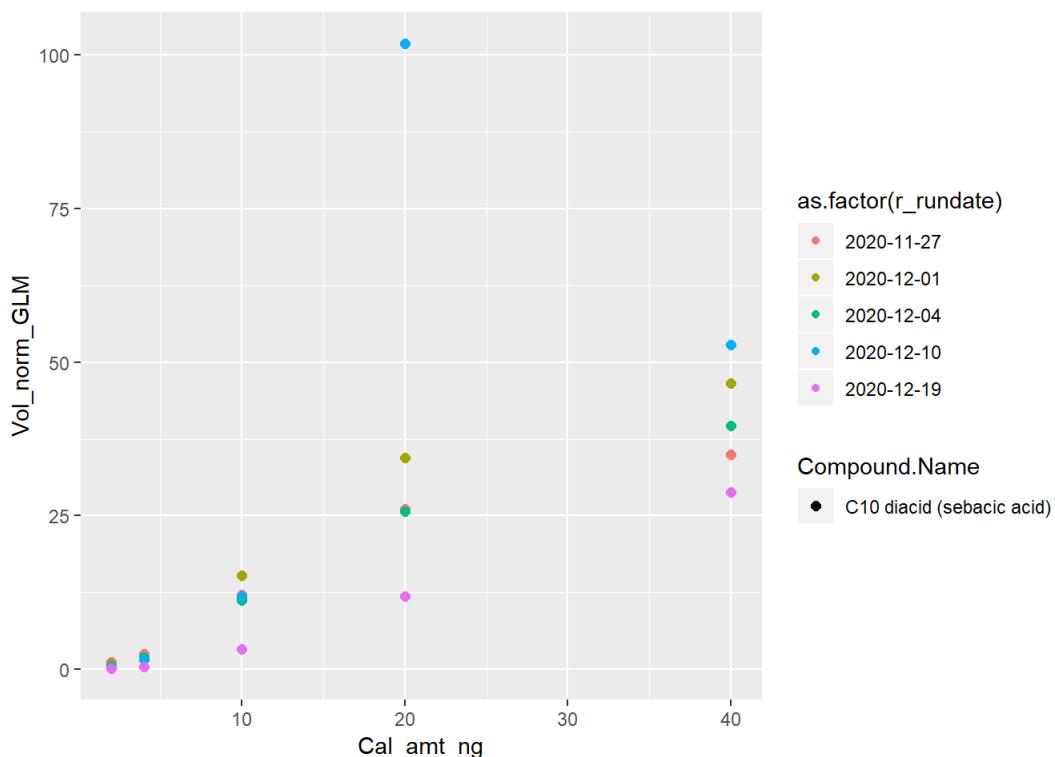
```
## Warning: Using size for a discrete variable is not advised.
```



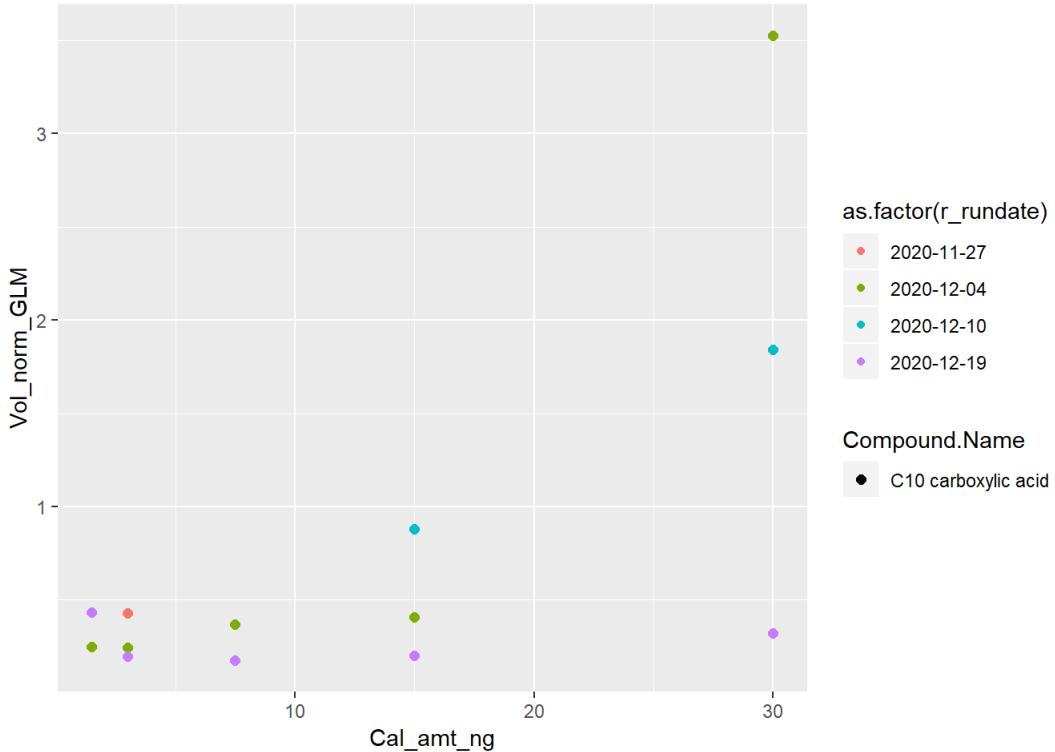
```
## Warning: Using size for a discrete variable is not advised.
```



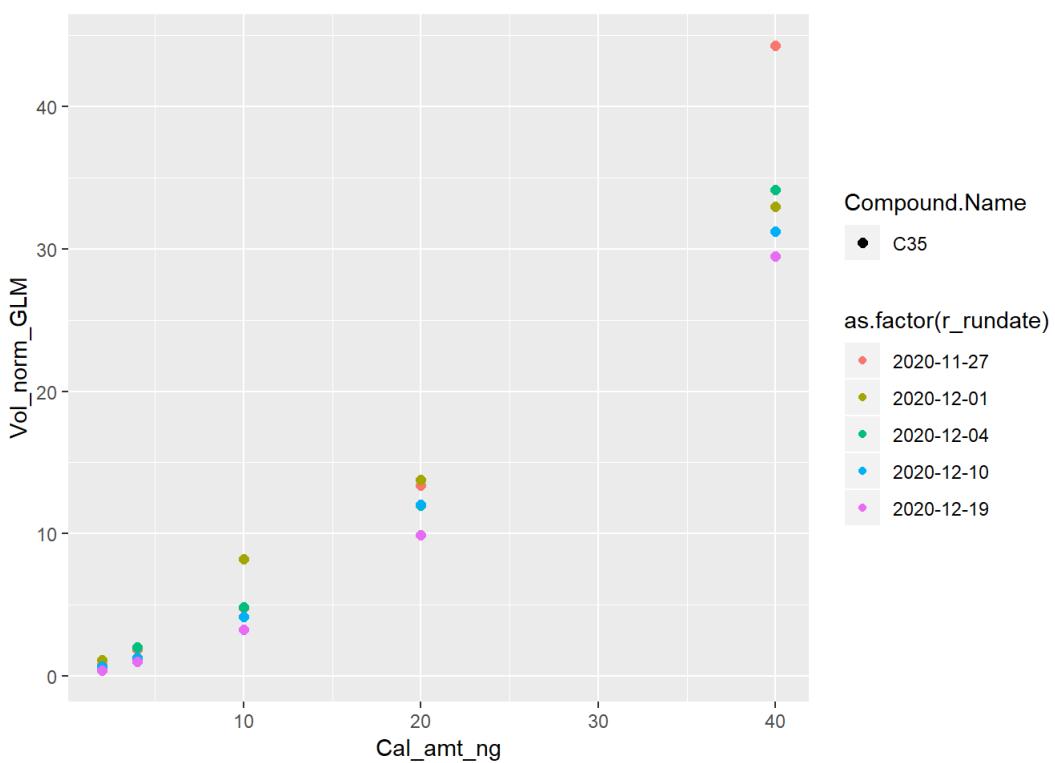
```
## Warning: Using size for a discrete variable is not advised.
```



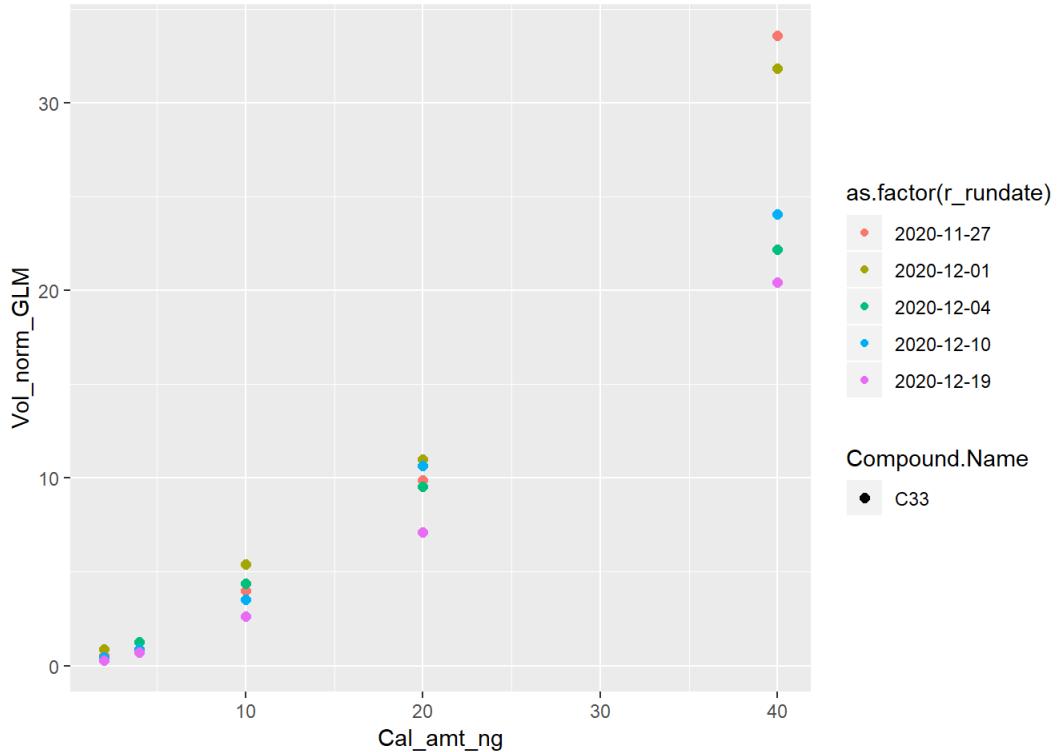
```
## Warning: Using size for a discrete variable is not advised.
```



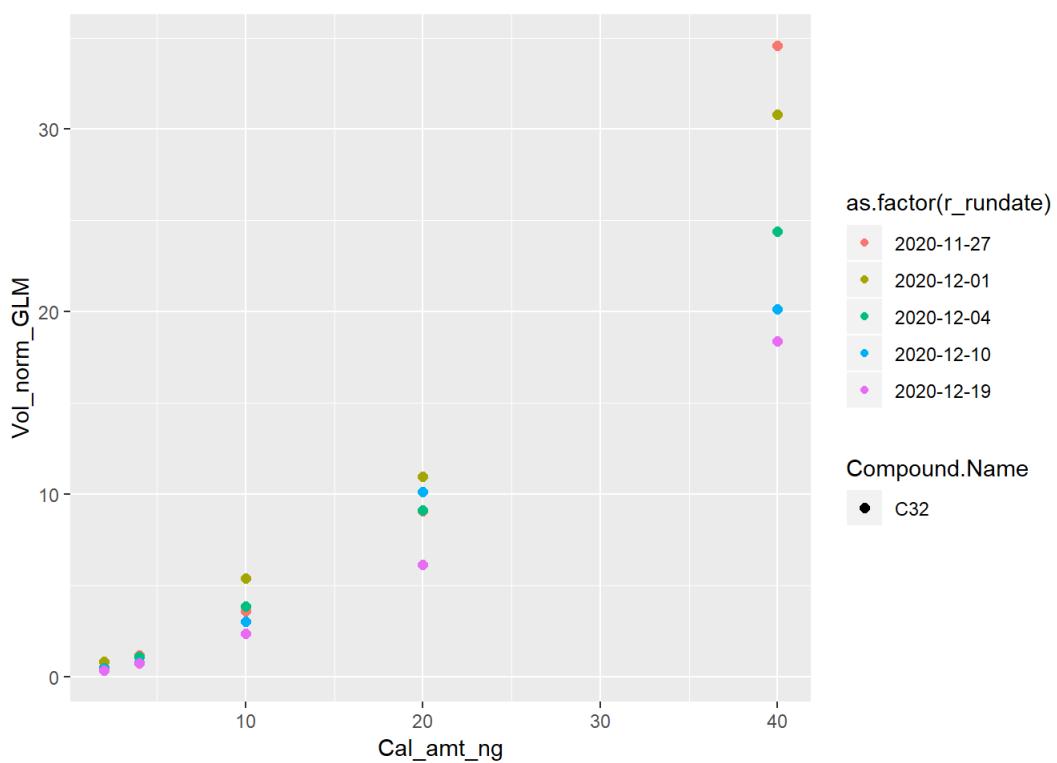
```
## Warning: Using size for a discrete variable is not advised.
```



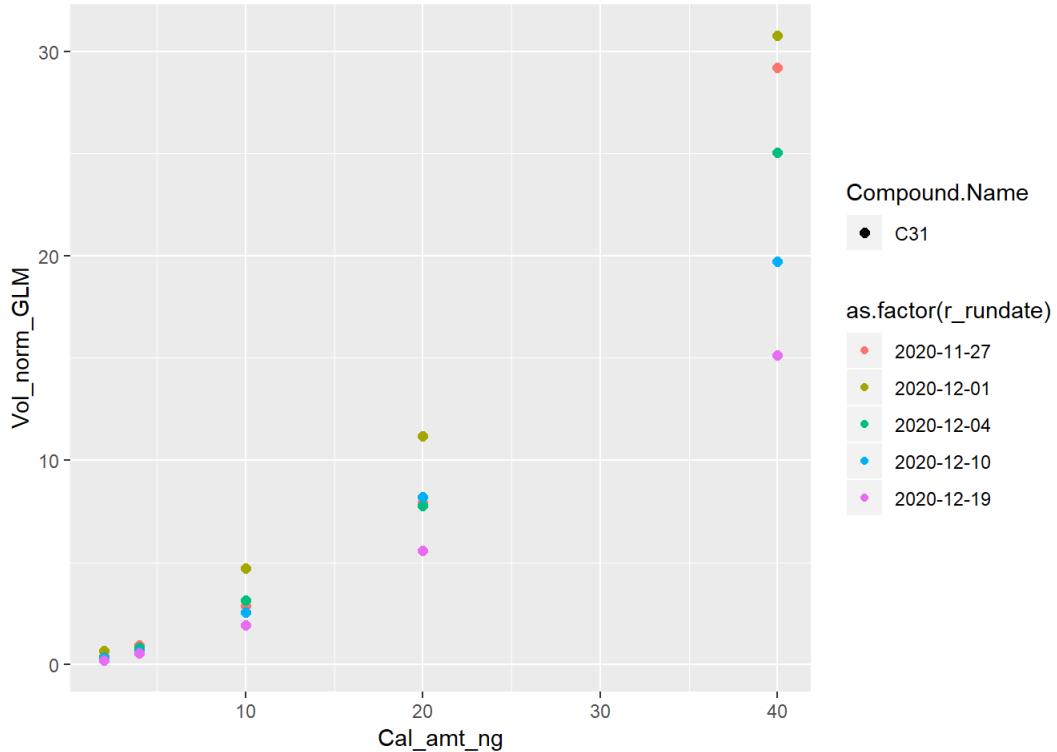
```
## Warning: Using size for a discrete variable is not advised.
```



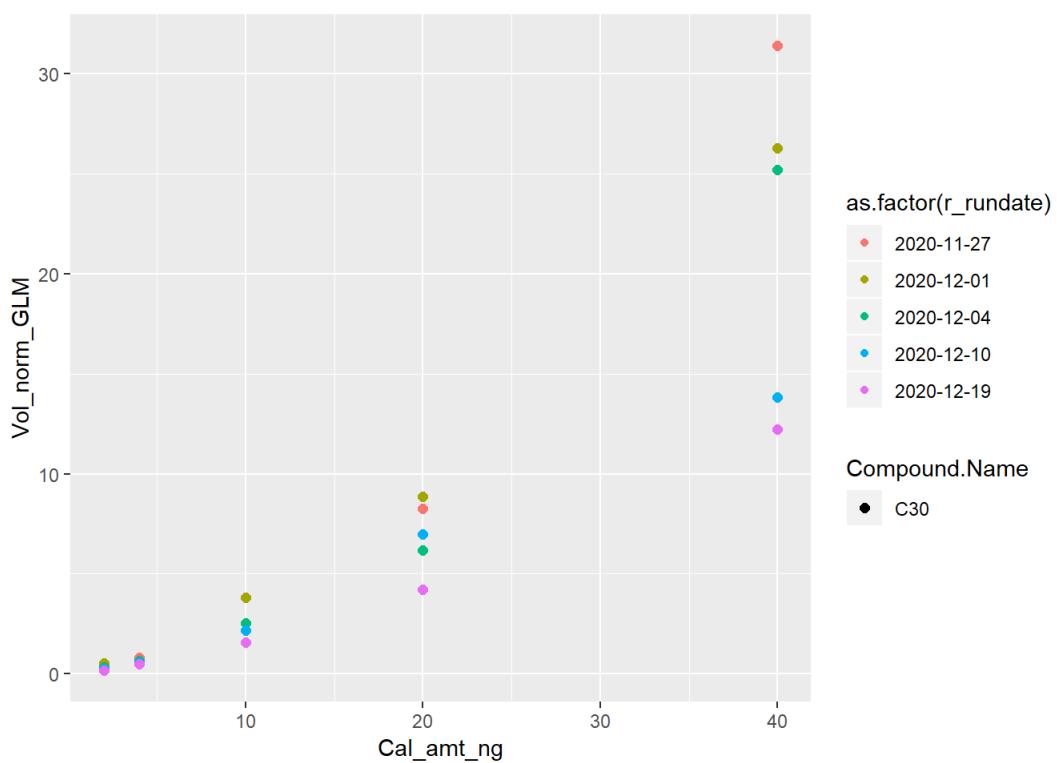
```
## Warning: Using size for a discrete variable is not advised.
```



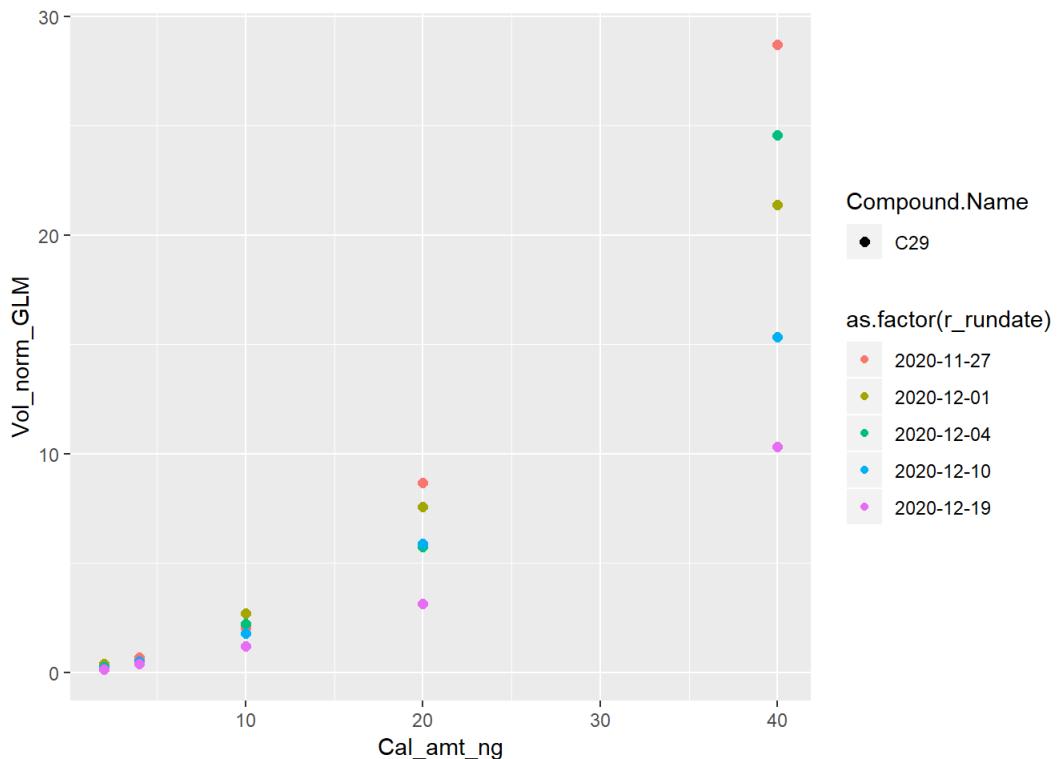
```
## Warning: Using size for a discrete variable is not advised.
```



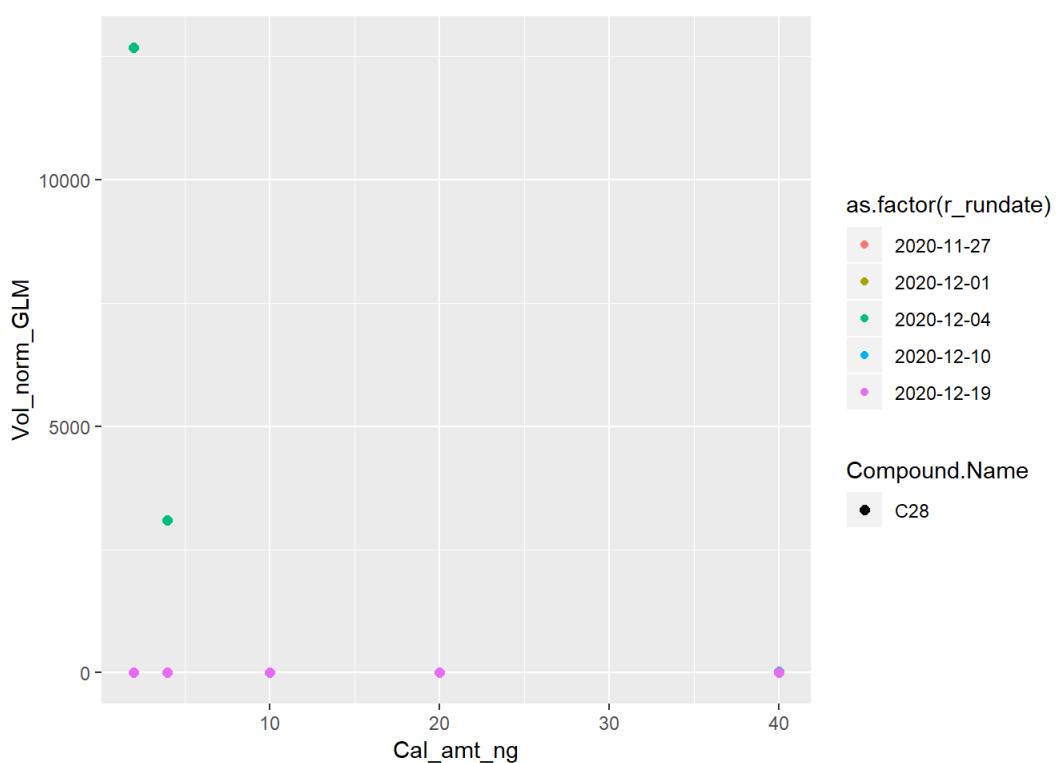
```
## Warning: Using size for a discrete variable is not advised.
```



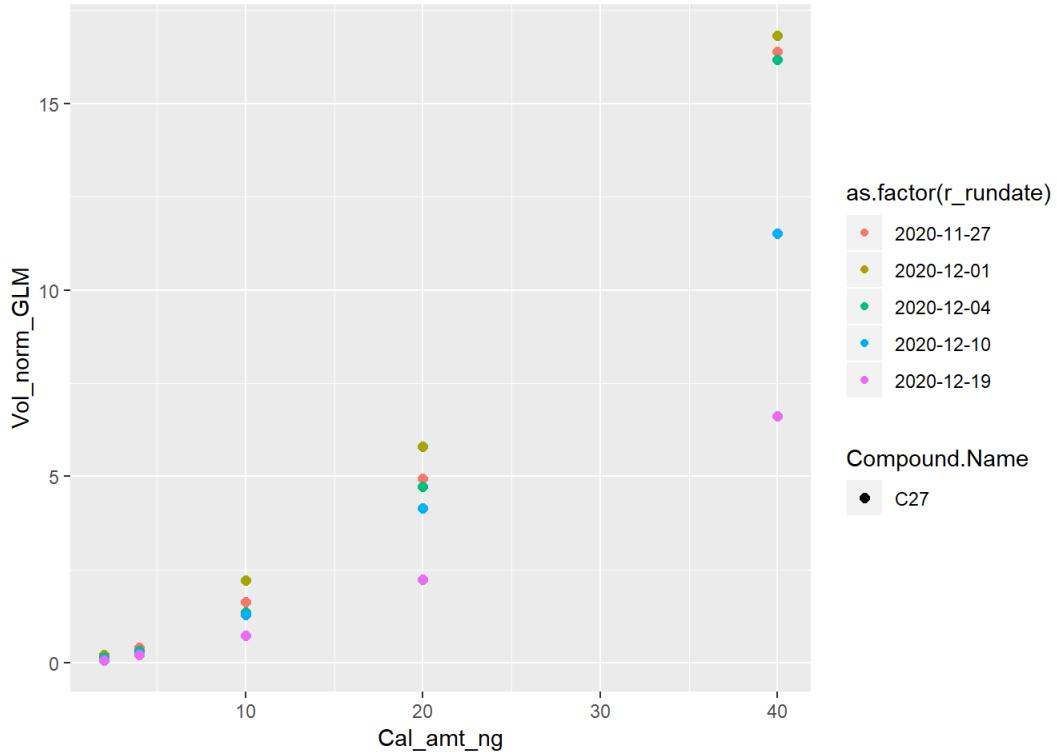
```
## Warning: Using size for a discrete variable is not advised.
```



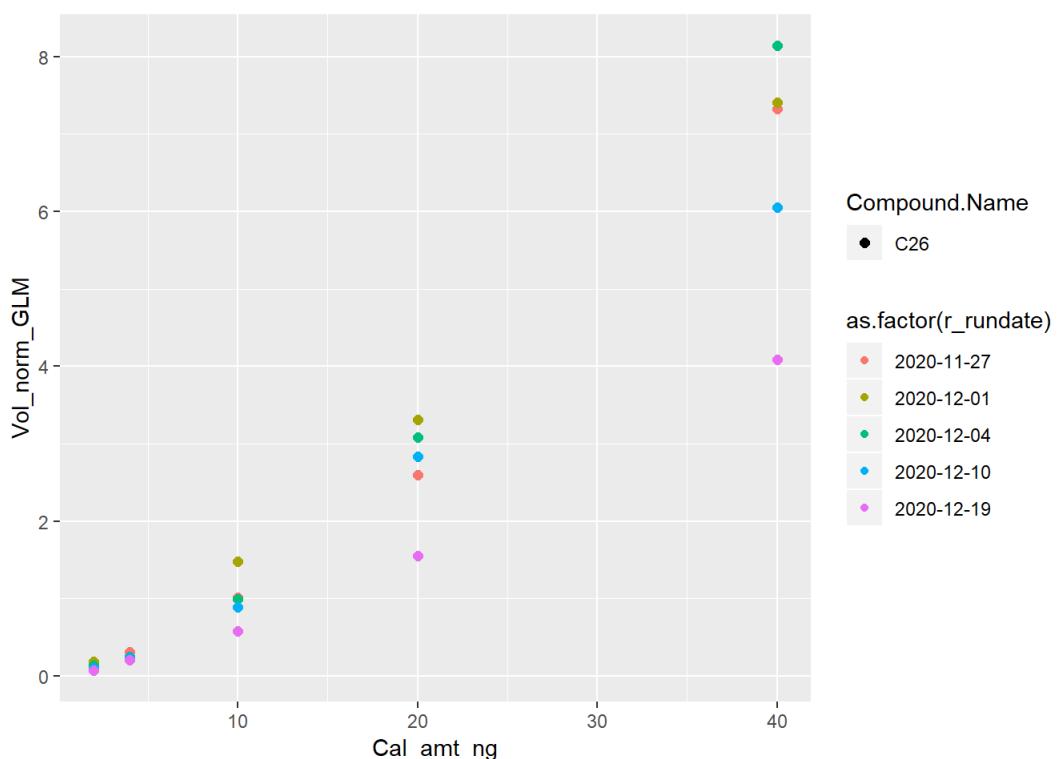
```
## Warning: Using size for a discrete variable is not advised.
```



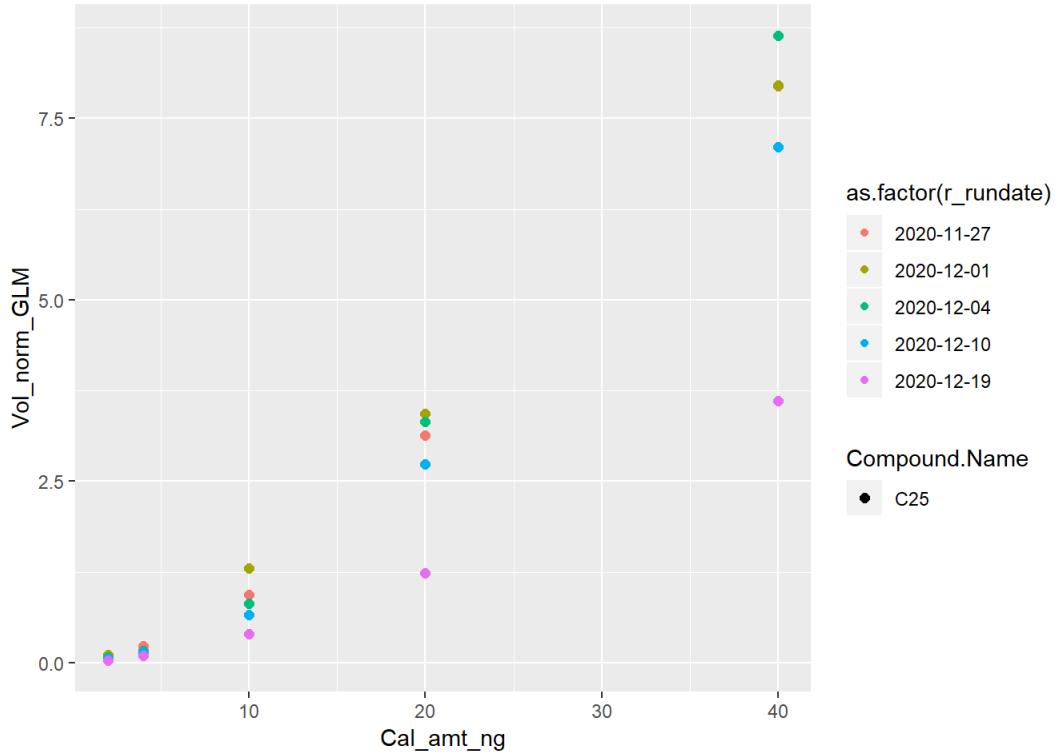
```
## Warning: Using size for a discrete variable is not advised.
```



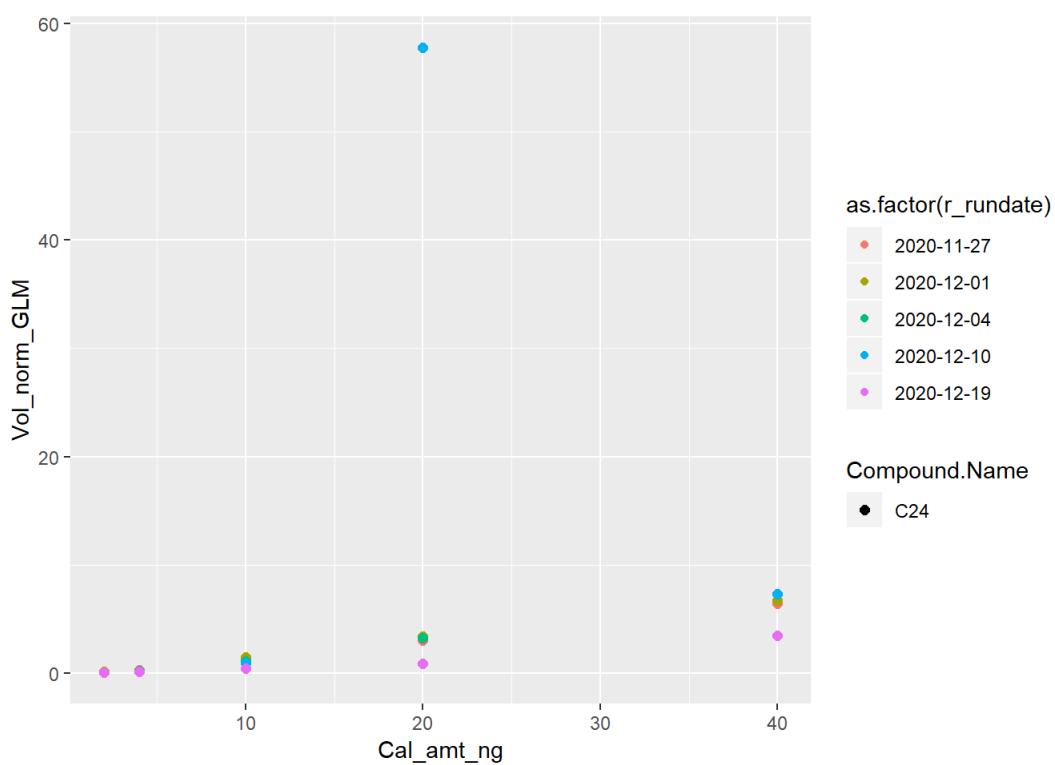
```
## Warning: Using size for a discrete variable is not advised.
```



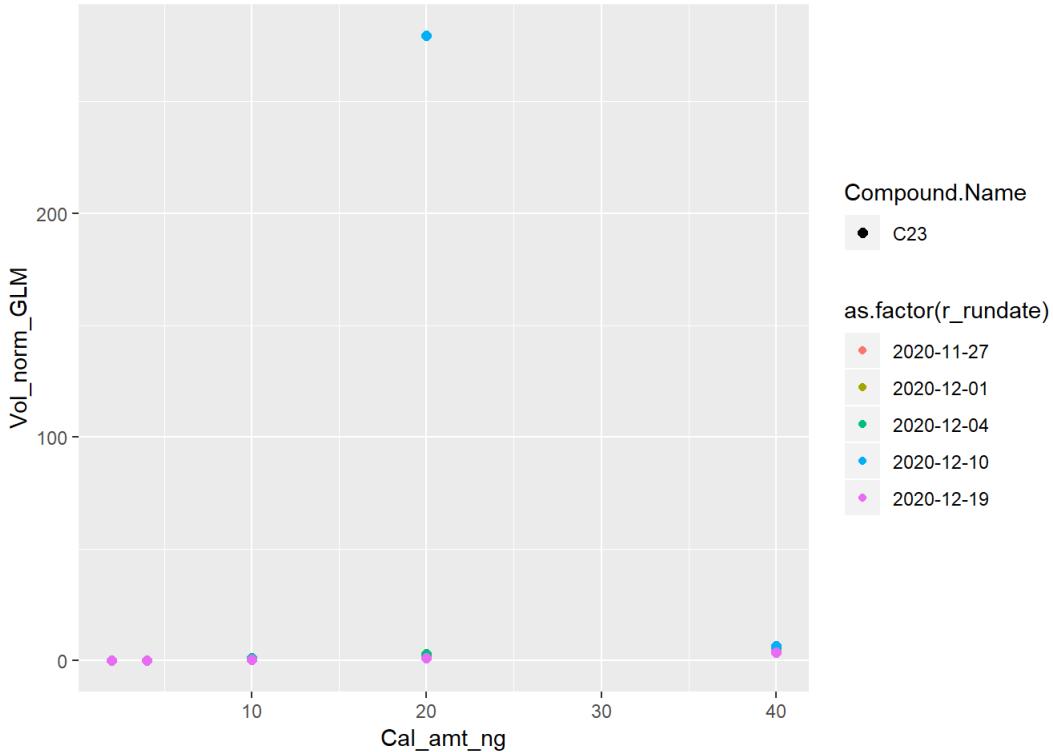
```
## Warning: Using size for a discrete variable is not advised.
```



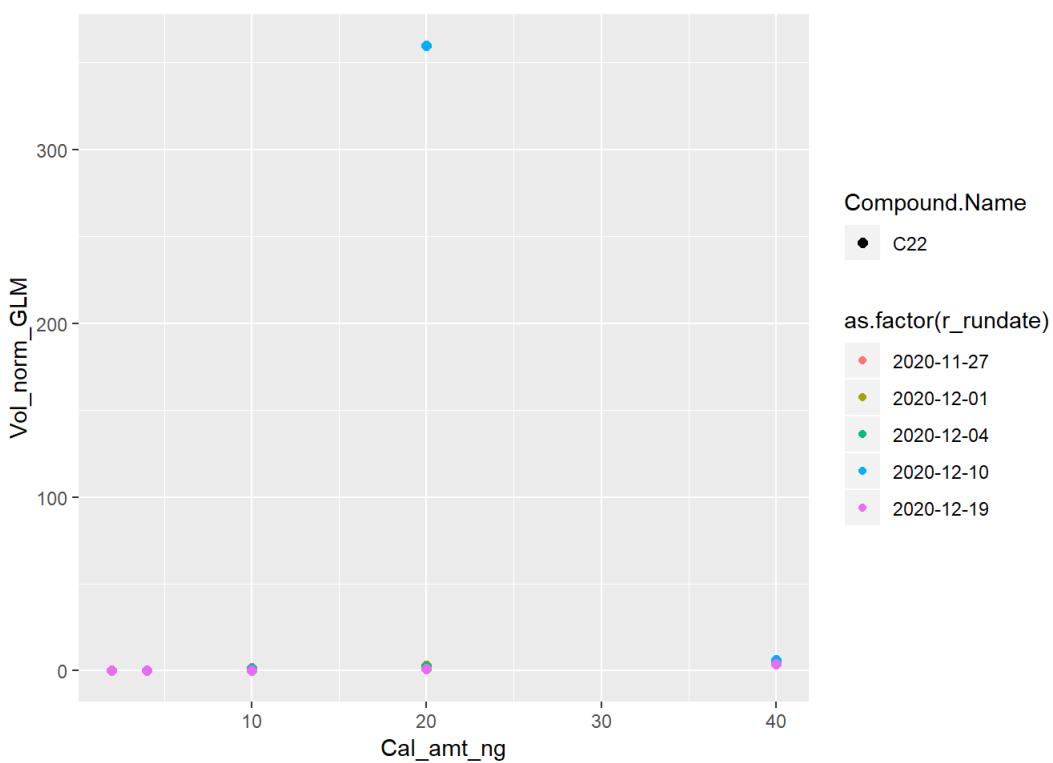
```
## Warning: Using size for a discrete variable is not advised.
```



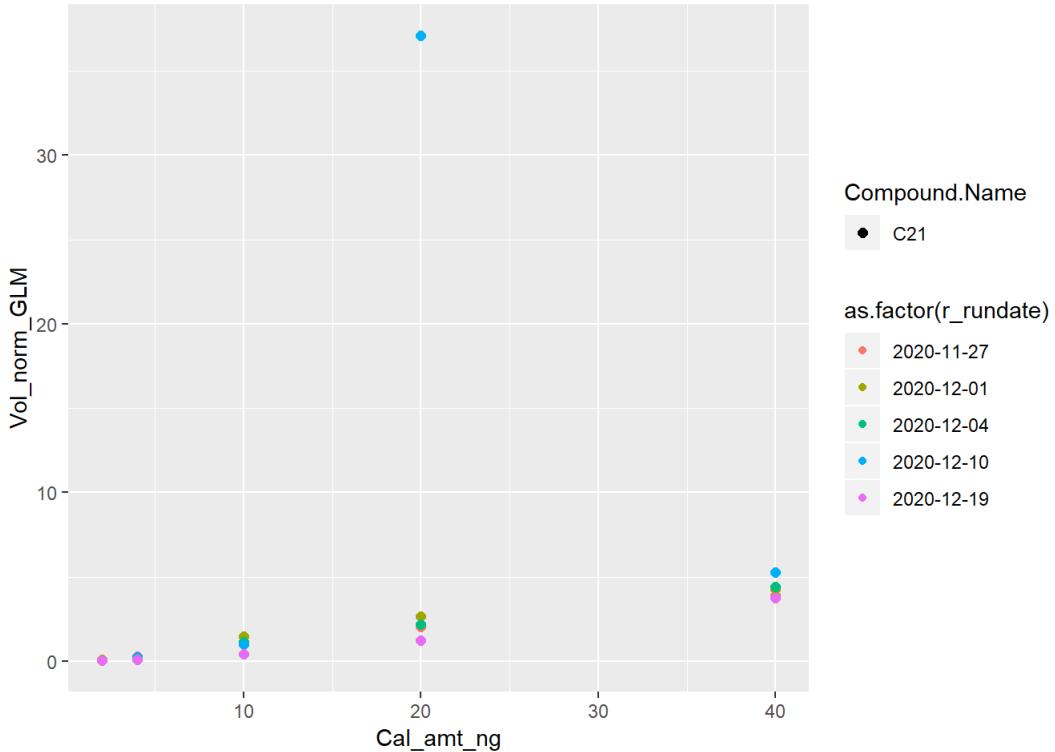
```
## Warning: Using size for a discrete variable is not advised.
```



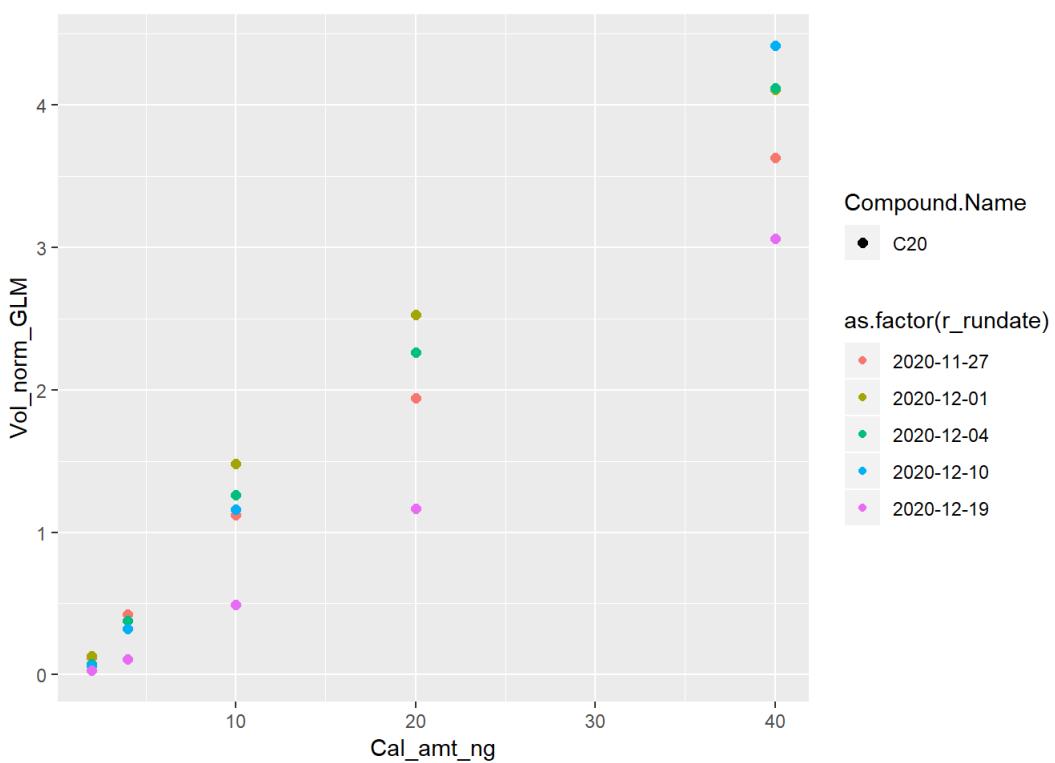
```
## Warning: Using size for a discrete variable is not advised.
```



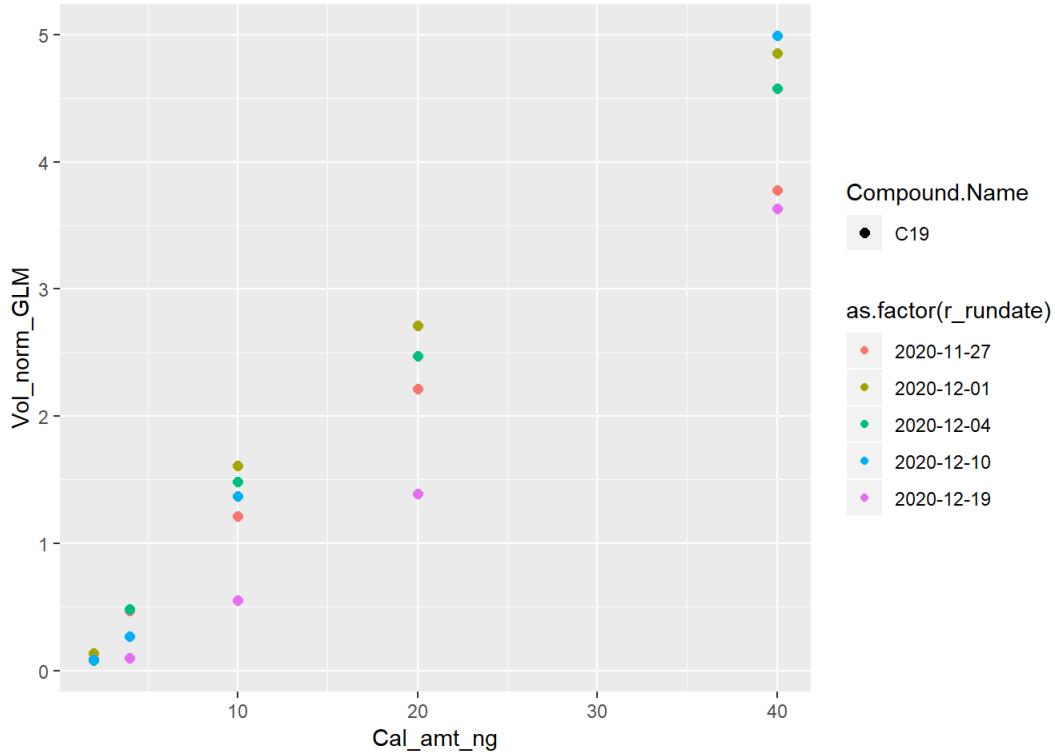
```
## Warning: Using size for a discrete variable is not advised.
```



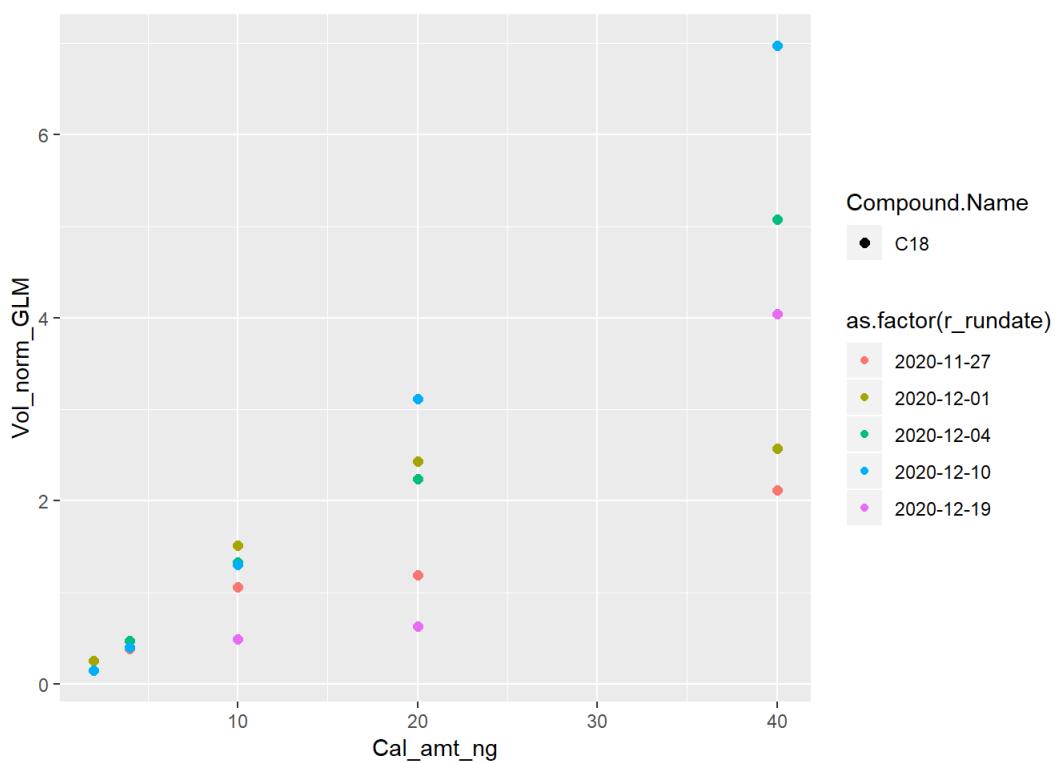
```
## Warning: Using size for a discrete variable is not advised.
```



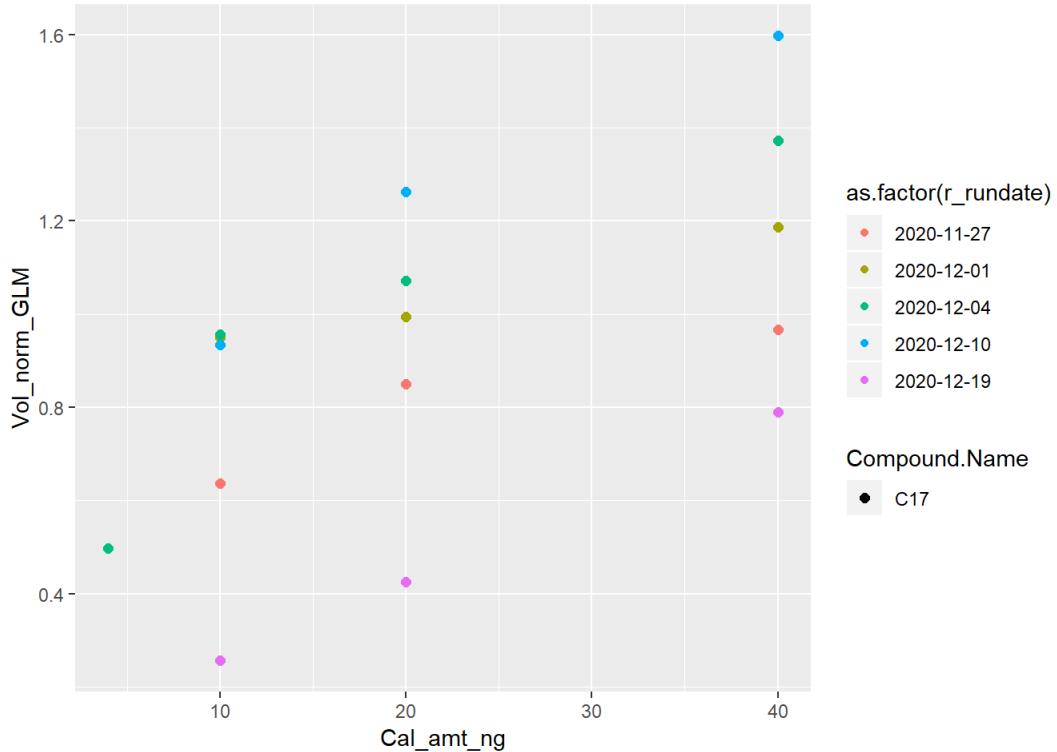
```
## Warning: Using size for a discrete variable is not advised.
```



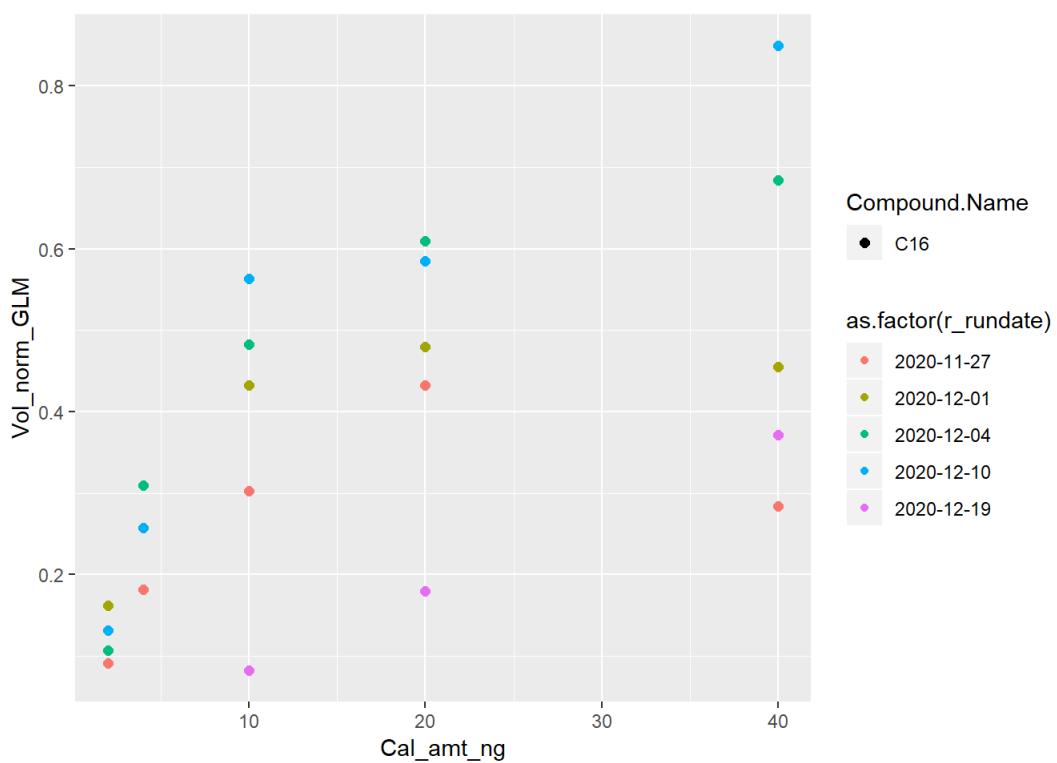
```
## Warning: Using size for a discrete variable is not advised.
```



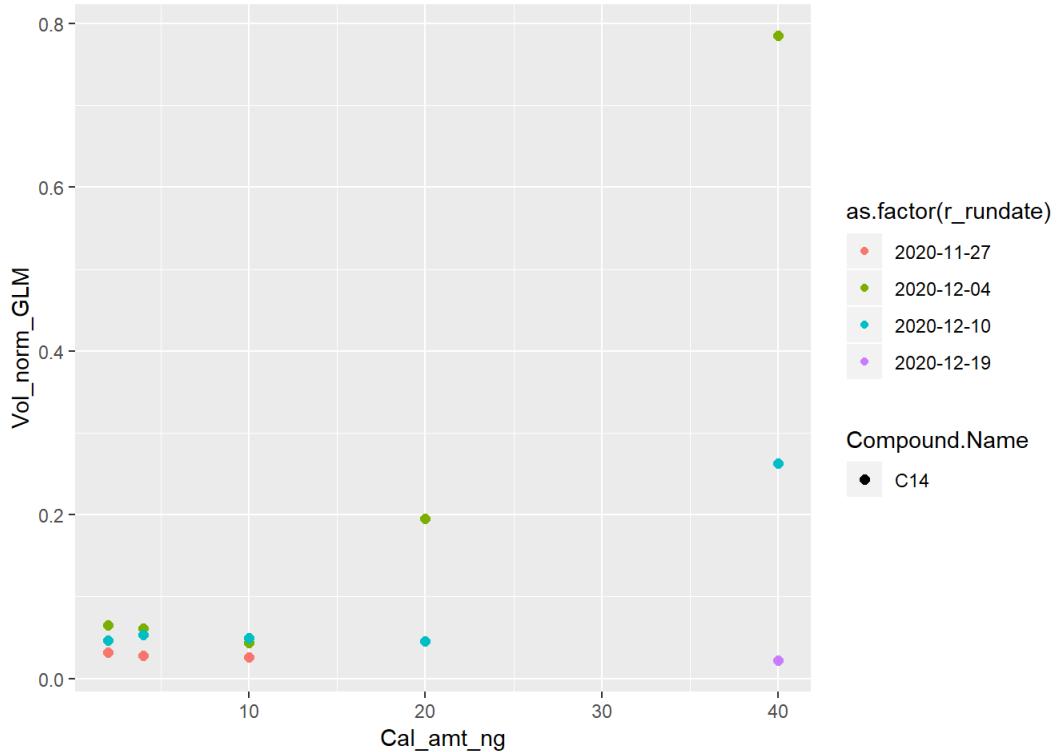
```
## Warning: Using size for a discrete variable is not advised.
```



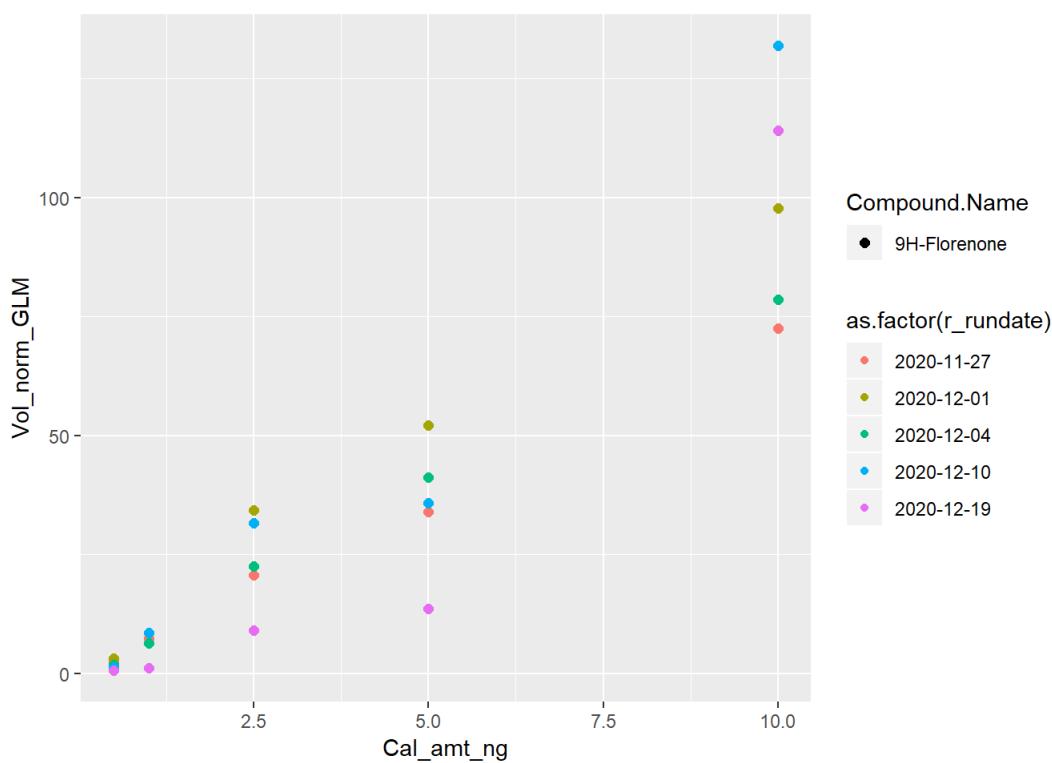
```
## Warning: Using size for a discrete variable is not advised.
```



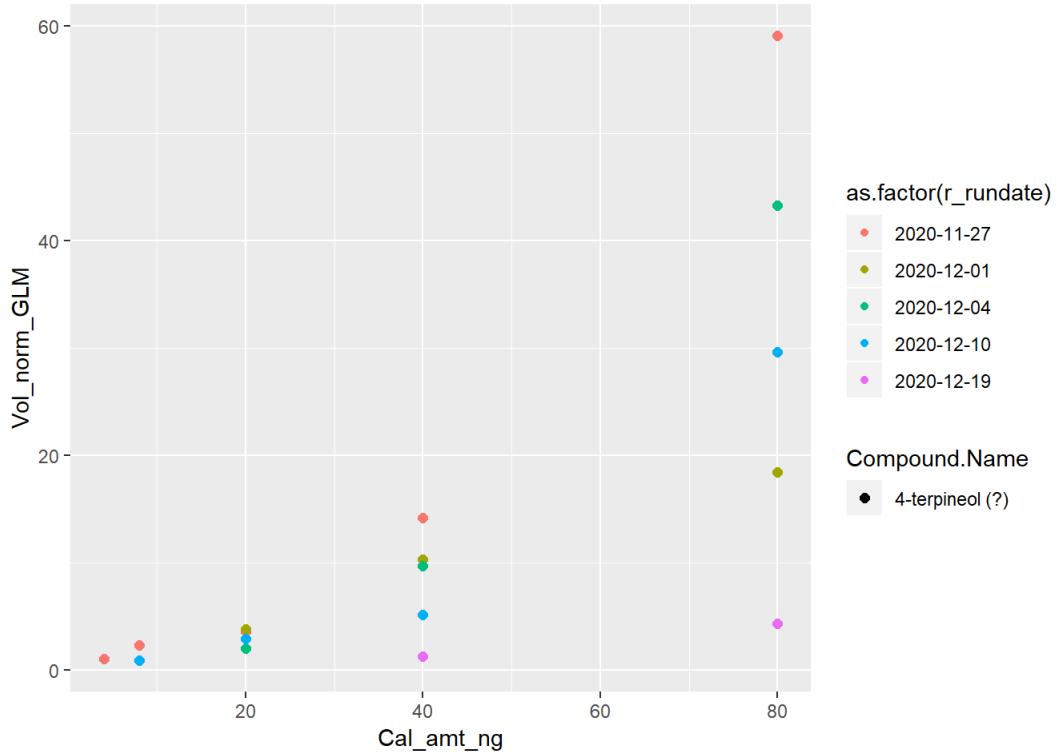
```
## Warning: Using size for a discrete variable is not advised.
```



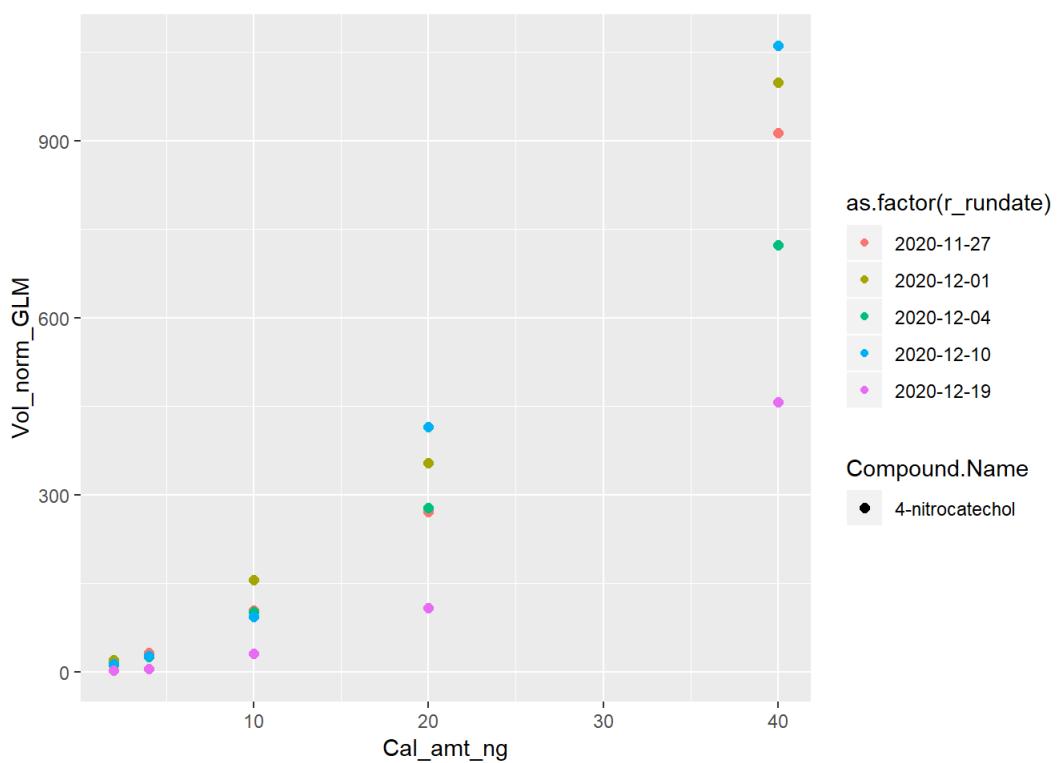
```
## Warning: Using size for a discrete variable is not advised.
```



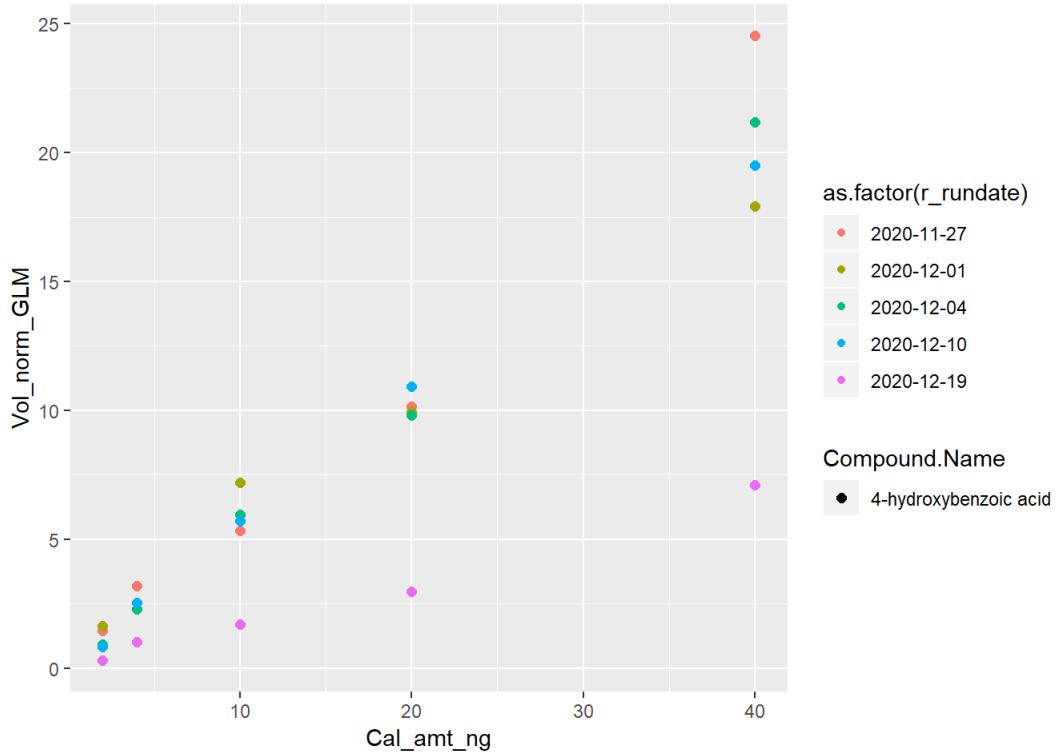
```
## Warning: Using size for a discrete variable is not advised.
```



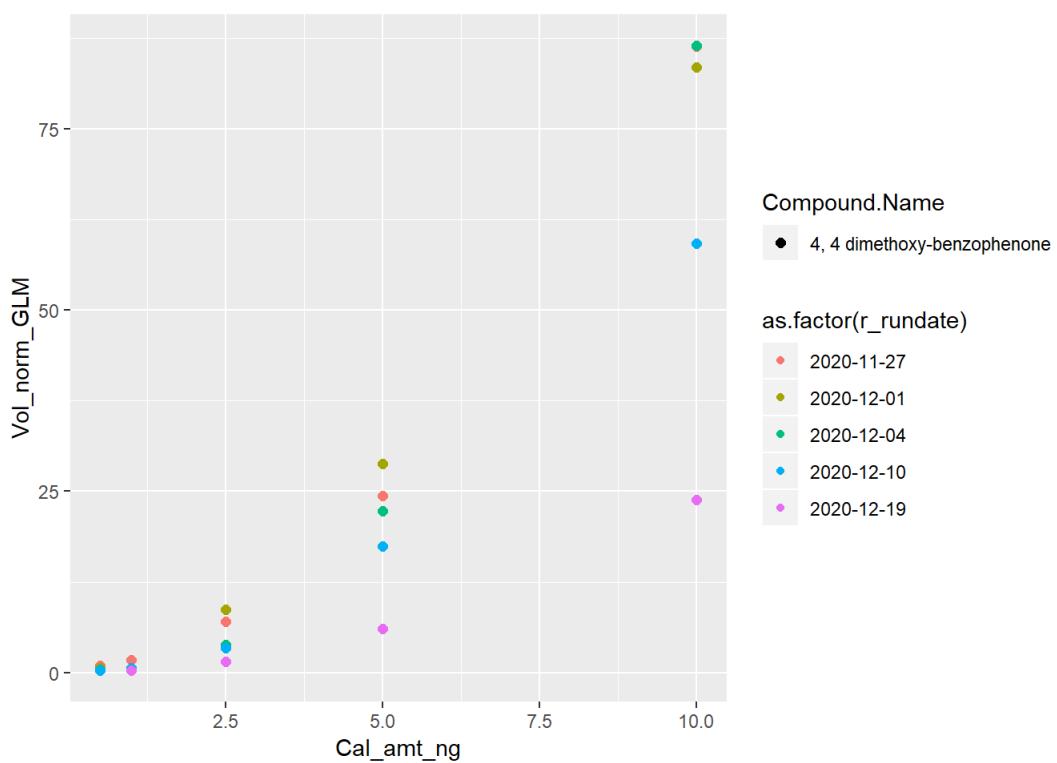
```
## Warning: Using size for a discrete variable is not advised.
```



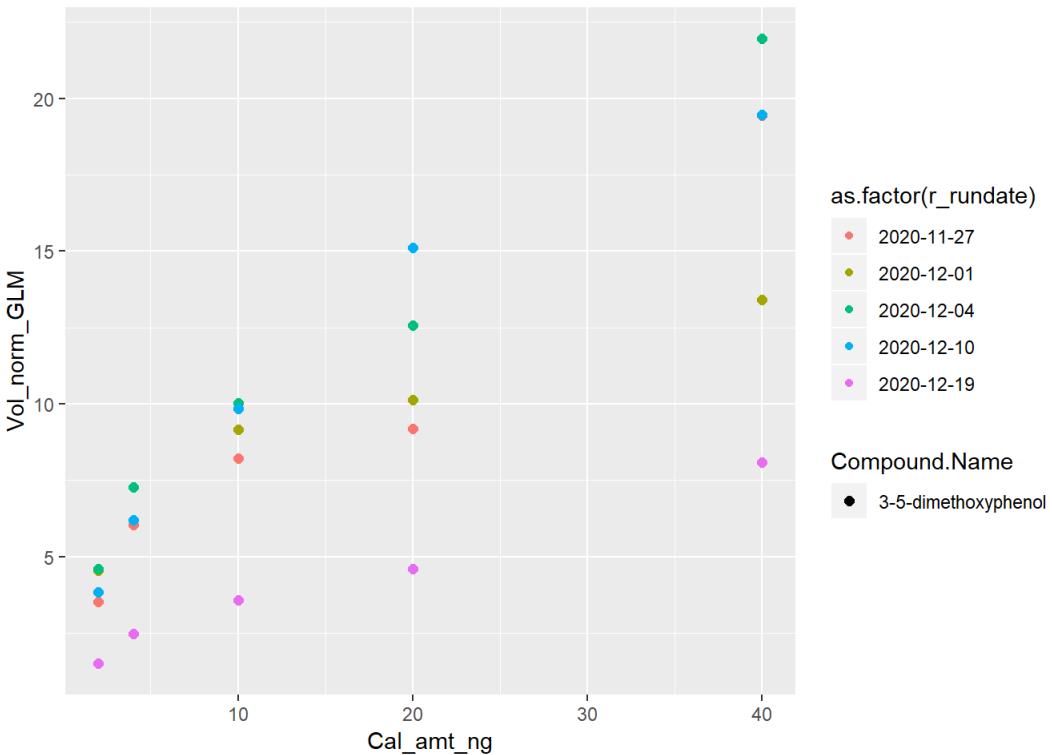
```
## Warning: Using size for a discrete variable is not advised.
```



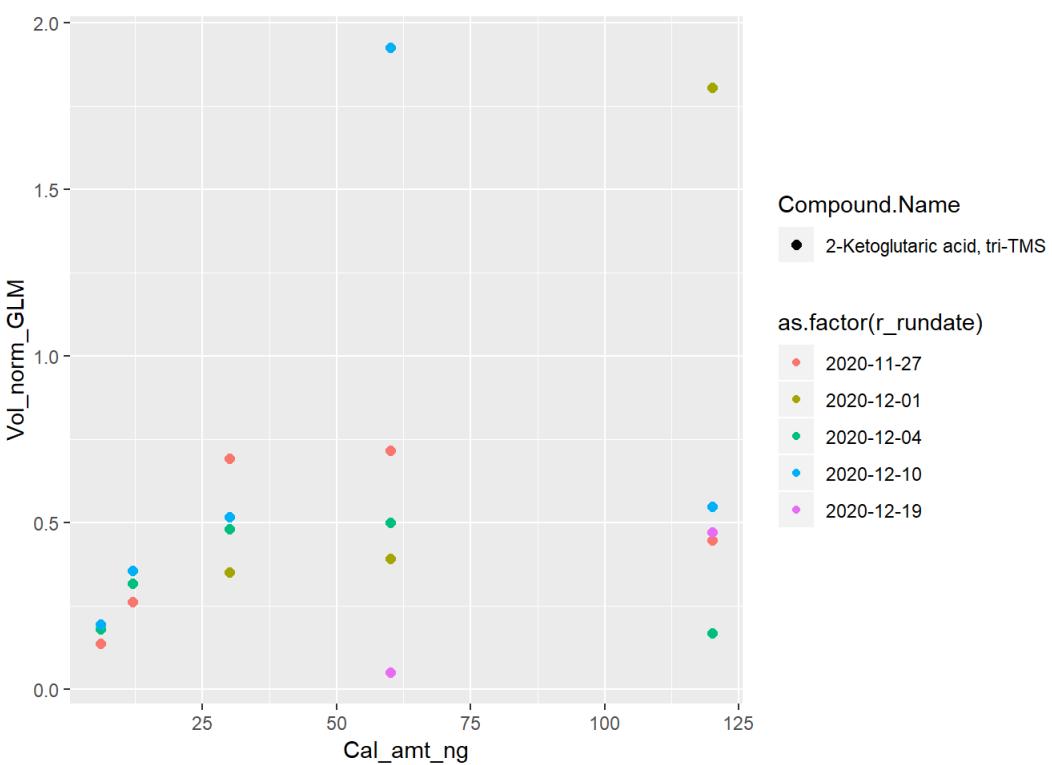
```
## Warning: Using size for a discrete variable is not advised.
```



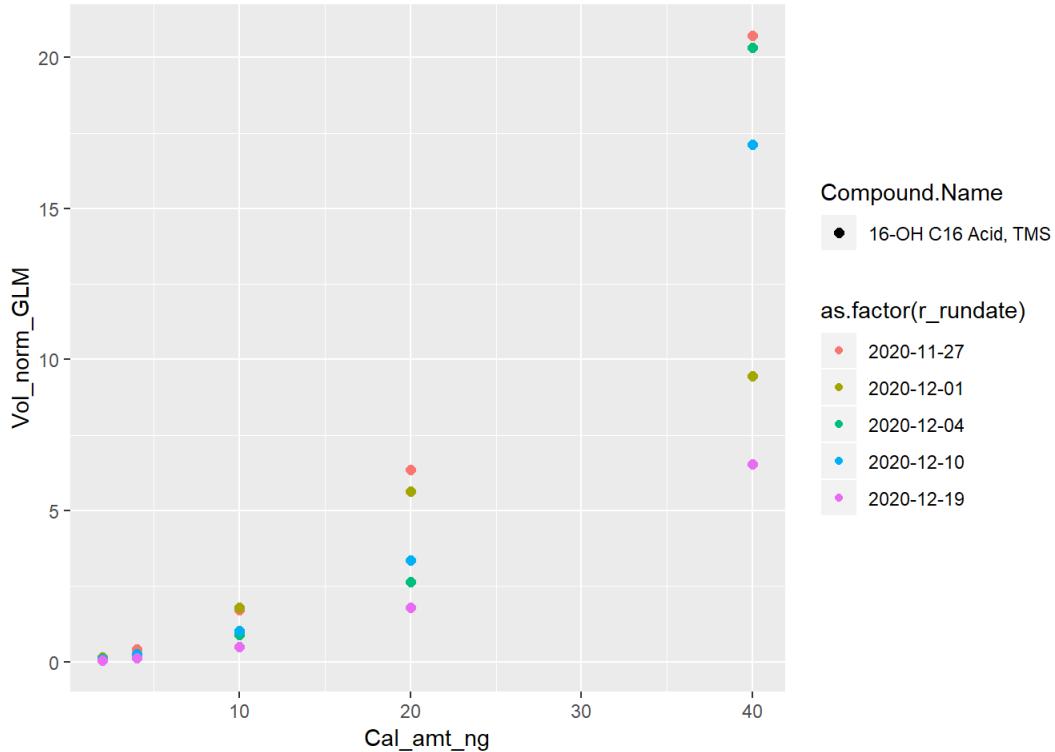
```
## Warning: Using size for a discrete variable is not advised.
```



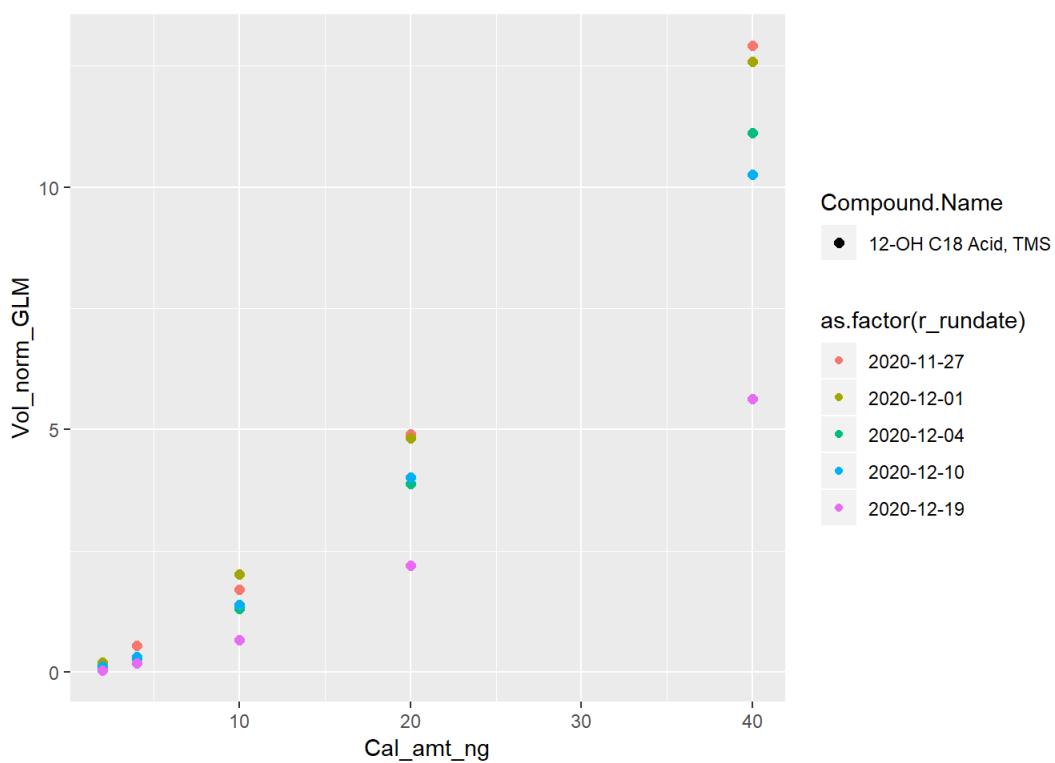
```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```

Vol_norm_GLM

Cal_amt_ng

```
## Warning: Using size for a discrete variable is not advised.
```

Vol_norm_GLM

Cal_amt_ng

```
## Warning: Using size for a discrete variable is not advised.
```

Vol_norm_GLM

Cal_amt_ng

```
## Warning: Using size for a discrete variable is not advised.
```

Vol_norm_GLM

Cal_amt_ng

```
## Warning: Using size for a discrete variable is not advised.
```

Vol_norm_GLM

Cal_amt_ng

```
## Warning: Using size for a discrete variable is not advised.
```

Vol_norm_GLM

Cal_amt_ng

```
## Warning: Using size for a discrete variable is not advised.
```

Vol_norm_GLM

Cal_amt_ng

```
## Warning: Using size for a discrete variable is not advised.
```

Vol_norm_GLM

Cal_amt_ng

```
## Warning: Using size for a discrete variable is not advised.
```

Problematic cal curve points

Cal_amt_ng

must be manually removed- as no single cal point is consistently off, it is probable that rather than a preparation error, there has been an issue with compound identification either in GC image or in the screening tools used prior to remove duplicate identification. As the goal is to quantify unknown compounds and I know that the points should if prepared correctly form a straight line, points which violate this are manually removed.

```

rows_todrop1 <- n_c_full %>%
  filter(Cal_Point == 5)

rows_todrop1 <- n_c_full %>%
  filter((Compound.Name == "MBTCA" & Cal_Point == "Cal_pt_5" & r_rundate == "2020-12-10") |
  (Compound.Name == "oleic Acid, TMS" & Cal_Point == "Cal_pt_5" & r_rundate == "2020-12-10") |
  (Compound.Name == "SESQ7" & r_rundate == "2020-12-01") |
  (Compound.Name == "SESQ7" & r_rundate == "2020-11-27" & Cal_Point == "Cal_pt_5") |
  (Compound.Name == "SESQ5") |
  (Compound.Name == "SESQ6" & r_rundate == "2020-12-01") |
  (Compound.Name == "SESQ4") |
  (Compound.Name == "SESQ3") |
  (Compound.Name == "SESQ1") |
  (Compound.Name == "octadecanol" & Cal_Point == "Cal_pt_5" & r_rundate == "2020-12-10") |
  (Compound.Name == "octadecanal (?)" & Cal_Point == "Cal_pt_6" & r_rundate == "2020-12-19") |
  (Compound.Name == "nonanol" & Cal_Point == "Cal_pt_2" & r_rundate == "2020-11-27") |
  (Compound.Name == "isopimamic acid" & Cal_Point == "Cal_pt_5" & r_rundate == "2020-12-10") |
  (Compound.Name == "hexadecanol, TMS" & Cal_Point == "Cal_pt_5" & r_rundate == "2020-12-10") |
  (Compound.Name == "isopimamic acid" & Cal_Point == "Cal_pt_5" & r_rundate == "2020-12-10") |
  (Compound.Name == "beta-caryophyllene aldehyde" & r_rundate == "2020-12-19") |
  (Compound.Name == "a-Amyrin" & Cal_Point == "Cal_pt_4" & r_rundate == "2020-12-10") |
  (Compound.Name == "FAME18 (Methyl Stearate)" & Cal_Point == "Cal_pt_5" & r_rundate == "2020-12-10") |
  (Compound.Name == "FAME16 (Methyl Palmitate)" & Cal_Point == "Cal_pt_5" & r_rundate == "2020-12-10") |
  (Compound.Name == "FAME18 (Methyl Stearate)" & Cal_Point == "Cal_pt_5" & r_rundate == "2020-12-10") |
  (Compound.Name == "C9 acid") |
  (Compound.Name == "C9 Diacid (azelaic acid)" & Cal_Point == "Cal_pt_5" & r_rundate == "2020-12-10") |
  (Compound.Name == "C18 Acid" & r_rundate == "2020-12-01") |
  (Compound.Name == "C18 Acid" & r_rundate == "2020-12-10") |
  (Compound.Name == "C18 Acid" & r_rundate == "2020-12-01") |
  (Compound.Name == "erythreitol" & r_rundate == "2020-12-19") |
  (Compound.Name == "Cis-Vaccenic Acid, TMS" & Cal_Point == "Cal_pt_6" & r_rundate == "2020-12-01") |
  (Compound.Name == "tridecanal (?)" & Cal_Point == "Cal_pt_5" & r_rundate == "2020-12-10") |
  (Compound.Name == "C18 Acid" & Cal_Point == "Cal_pt_2") |
  (Compound.Name == "C10 diacid (sebacic acid)" & Cal_Point == "Cal_pt_5" & r_rundate == "2020-12-10") |
  (Compound.Name == "C10 carboxylic acid") |
  (Compound.Name == "C28" & r_rundate == "2020-12-04") |
  (Compound.Name == "pinonic acid" & Cal_Point == "Cal_pt_6" & r_rundate == "2020-12-04") |
  (Compound.Name == "phthalimide" & Cal_Point == "Cal_pt_6" & r_rundate == "2020-12-04") |
  (Compound.Name == "C24" & Cal_Point == "Cal_pt_5" & r_rundate == "2020-12-10") |
  (Compound.Name == "C23" & Cal_Point == "Cal_pt_5" & r_rundate == "2020-12-10") |
  (Compound.Name == "C22" & Cal_Point == "Cal_pt_5" & r_rundate == "2020-12-10") |
  (Compound.Name == "C21" & Cal_Point == "Cal_pt_5" & r_rundate == "2020-12-10") |
  (Compound.Name == "glyceric acid (?)" & Cal_Point == "Cal_pt_6" & r_rundate == "2020-12-10") |
  (Compound.Name == "glyceric acid (?)" & Cal_Point == "Cal_pt_6" & r_rundate == "2020-12-04") |
  (Compound.Name == "2-Ketoglutaric acid, tri-TMS" & Cal_Point == "Cal_pt_5" & r_rundate == "2020-12-10"))

ES_cleaned1 <- n_c_full %>%
  anti_join(rows_todrop1)

```

```

## Joining, by = c("BlobID", "Compound.Name", "Description", "Group.Name", "Constellation.Name", "Inclusion",
, "LRI.I", "Library.RI", "Library.ID", "Library.Name", "Library.NIST.ID", "Library.Match.Factor", "Library.R
everse.Match.Factor", "Library.Probability", "Review.Status", "Internal.Standard", "Retention.I..min.", "Ret
ention.II..sec.", "Peak.Value", "Volume", "Quantifier.1.", "Quantifier.2.", "Quantifier.Response.1.", "Quant
ifier.Response.2.", "Volume.Ratio", "File_num", "Category", "Filter_num", "Filter_punches", "IOP", "AMZ_date
", "T_O_D", "Date_num", "Internal_load", "Cal_Point_number", "Run_date", "Run_date_num", "Cal_Point", "r_run
date", "LRI_diff", "unique.analyte", "comp.n", "pct_of_samp", "IS_glm", "Vol_norm_GLM", "Cal_amt_ng")

```

```

for(i in 1:nrow(cal_pts)){
  bid_temp <- cal_pts$Compound.Name[i]

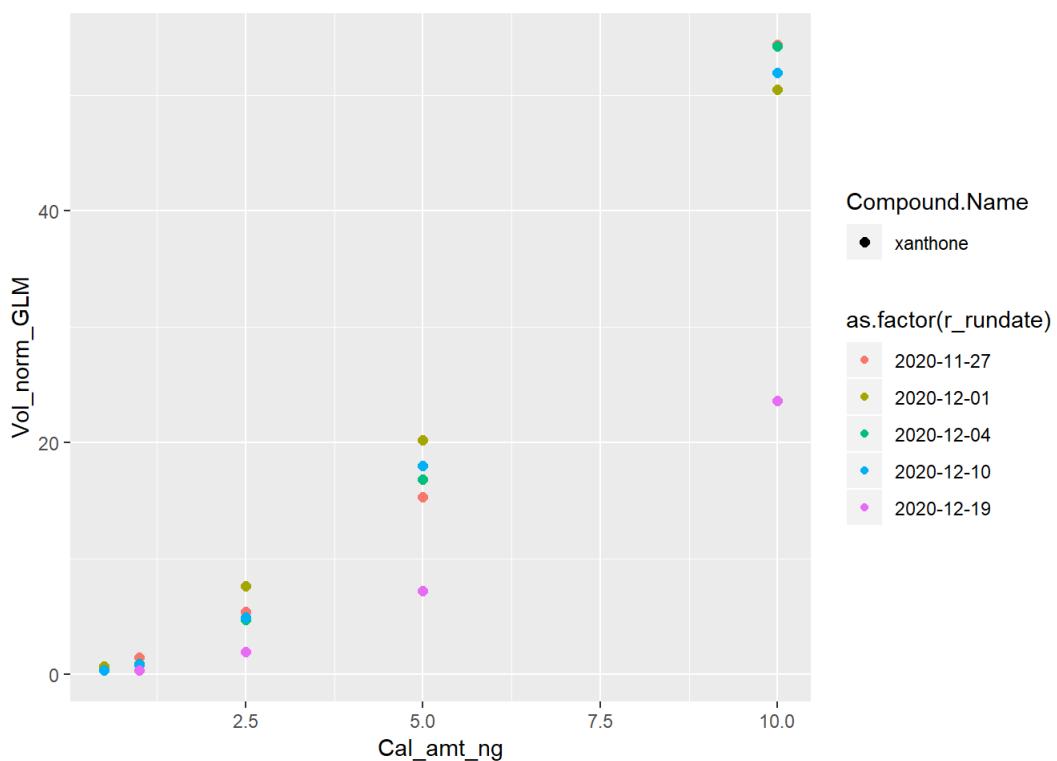
  n_c_full_sub <- ES_cleaned1 %>%
    filter(Compound.Name == bid_temp)

  p <- n_c_full_sub %>%
    ggplot(aes(x = Cal_amt_ng, y = Vol_norm_GLM, color = as.factor(r_rundate), size = Compound.Name)) +
    geom_point()

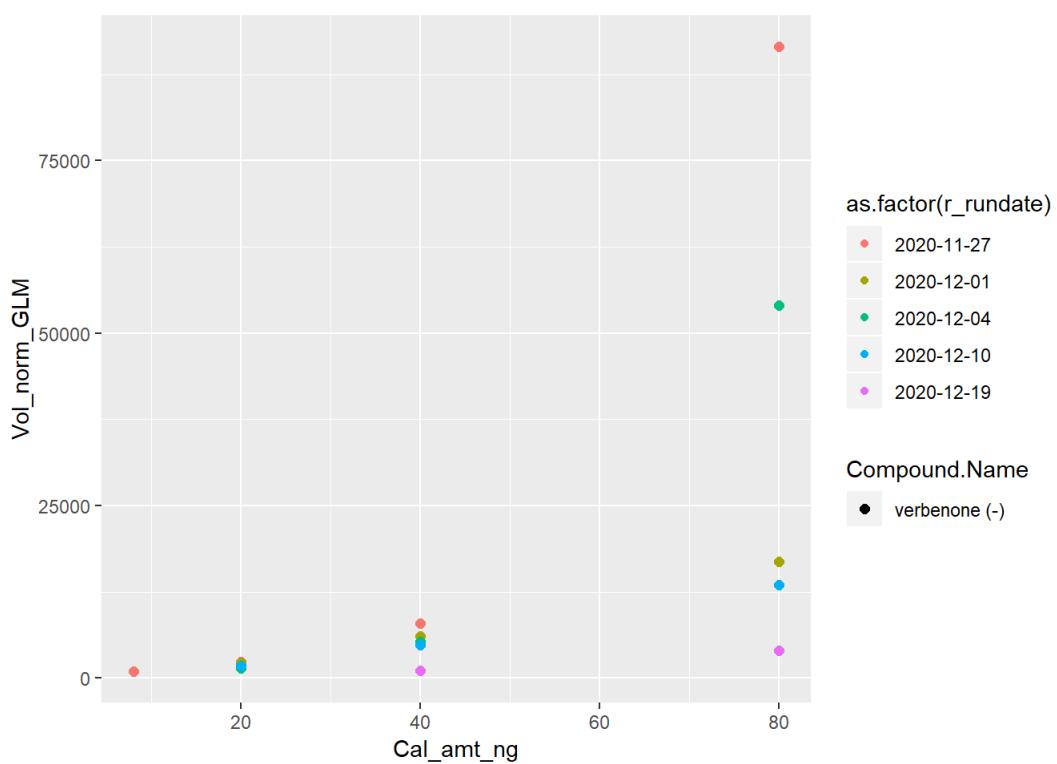
  print(p)
}

```

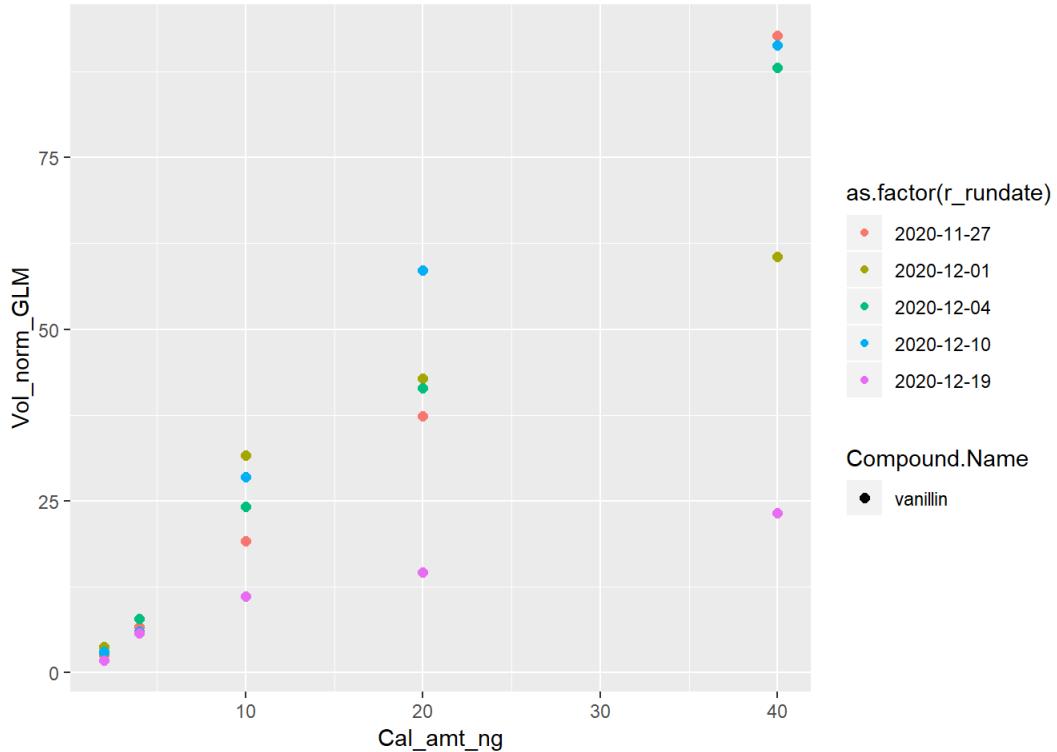
```
## Warning: Using size for a discrete variable is not advised.
```



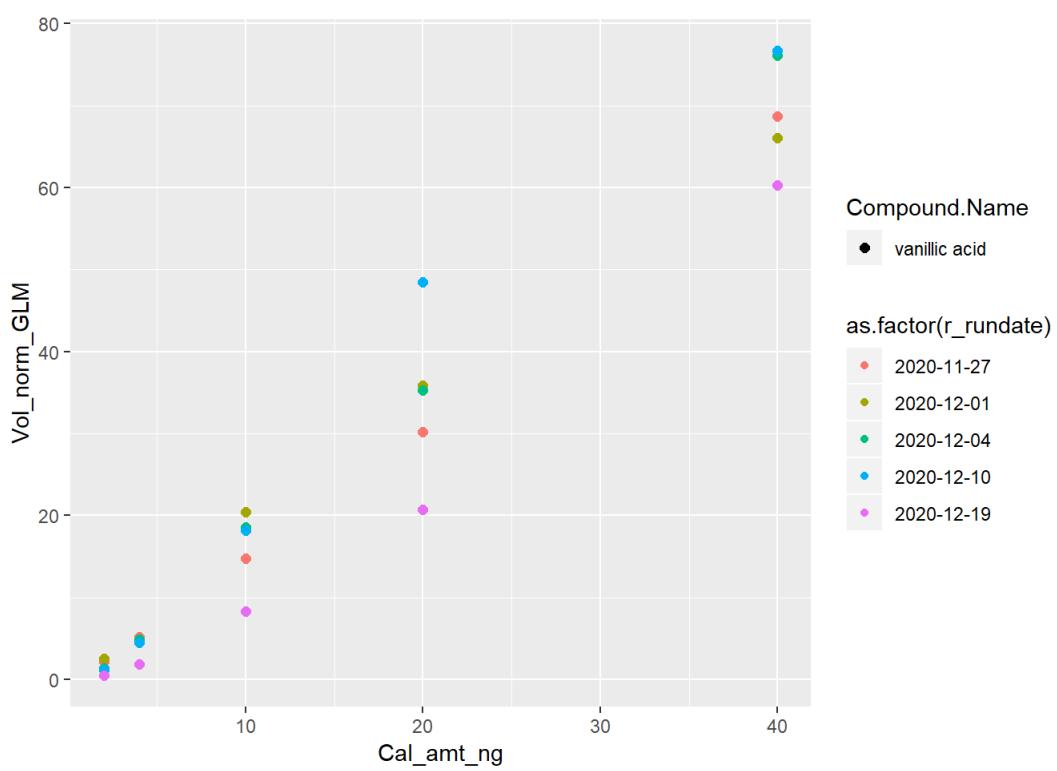
```
## Warning: Using size for a discrete variable is not advised.
```



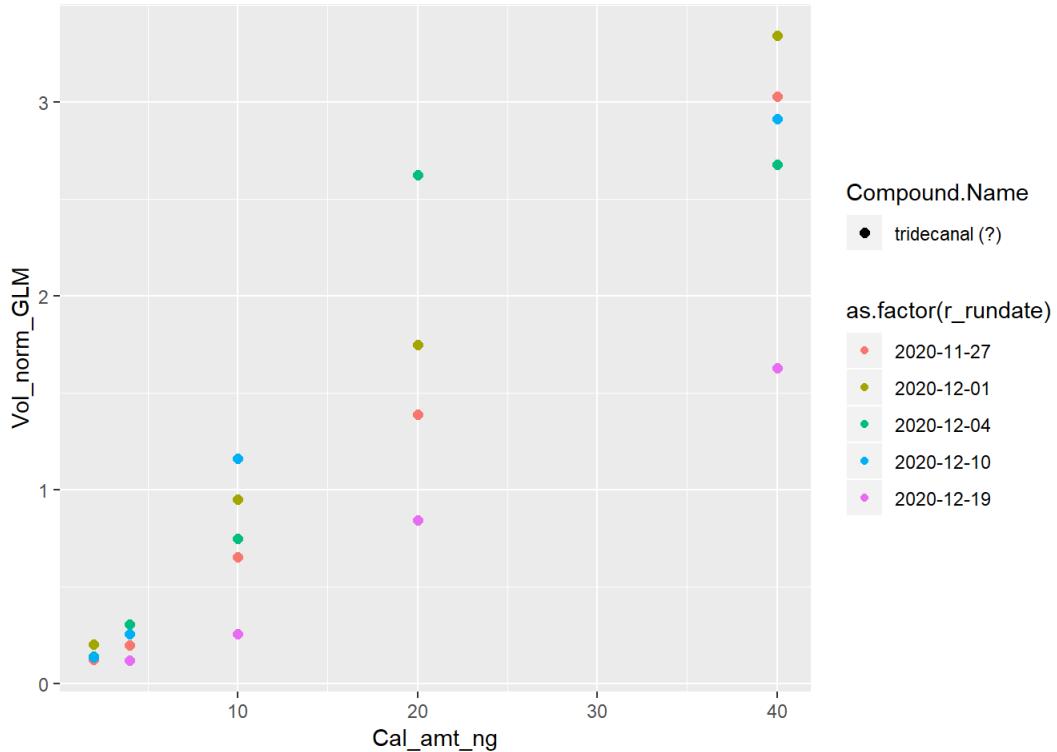
```
## Warning: Using size for a discrete variable is not advised.
```



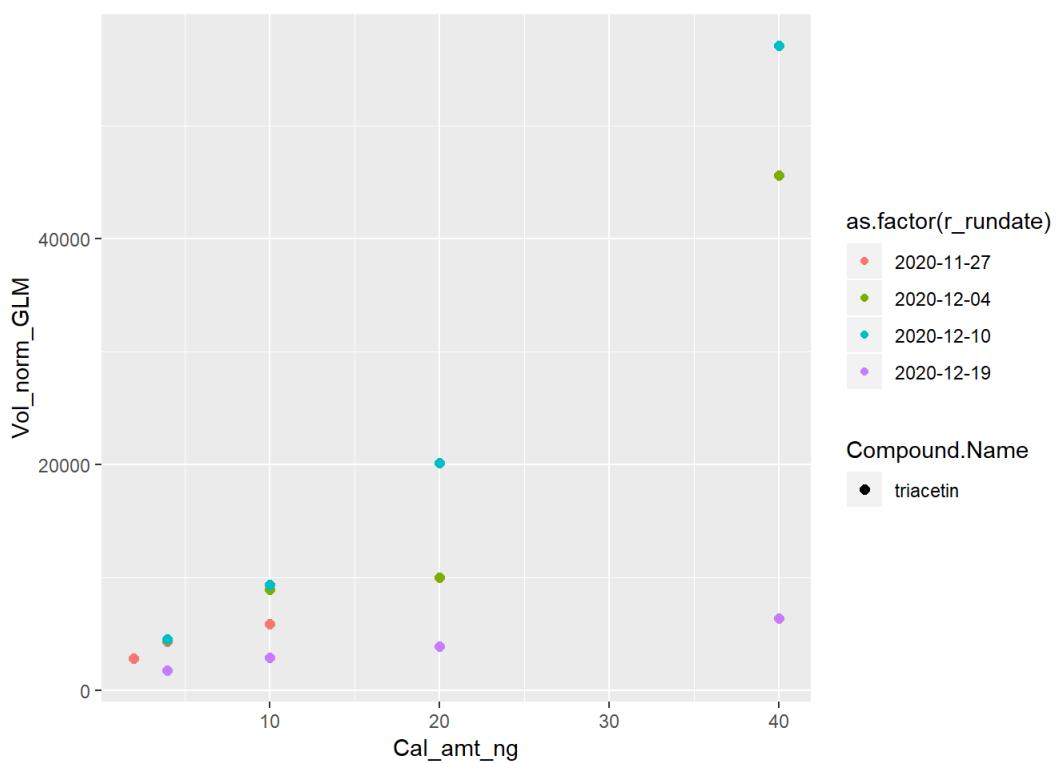
```
## Warning: Using size for a discrete variable is not advised.
```



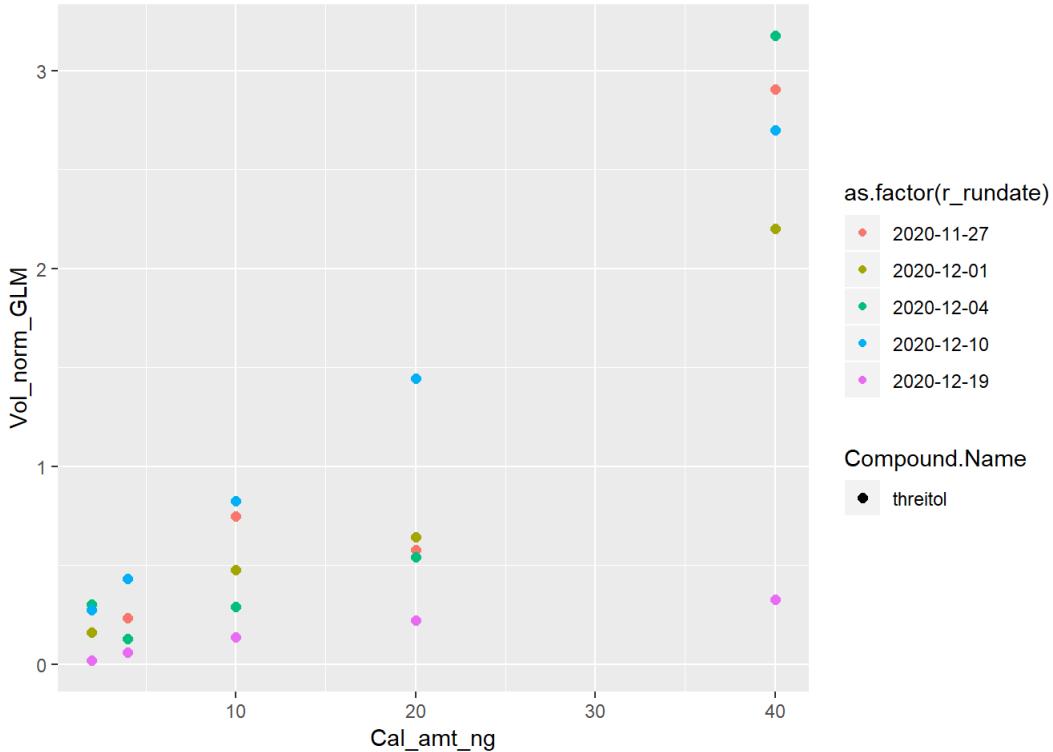
```
## Warning: Using size for a discrete variable is not advised.
```



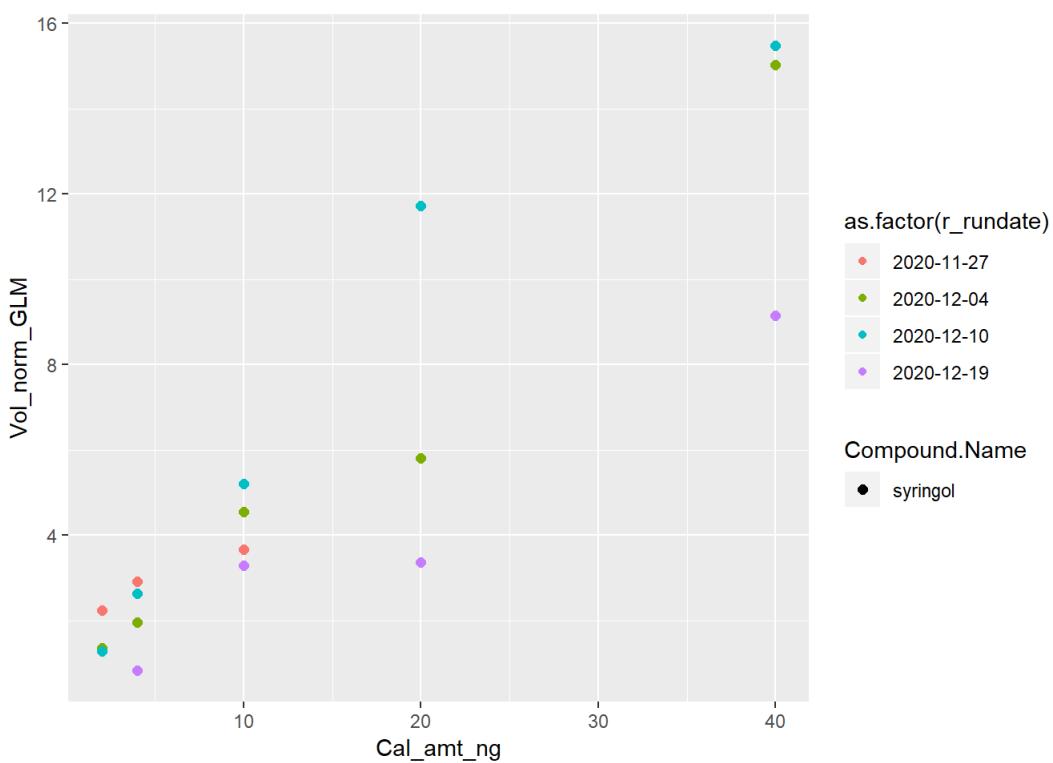
```
## Warning: Using size for a discrete variable is not advised.
```



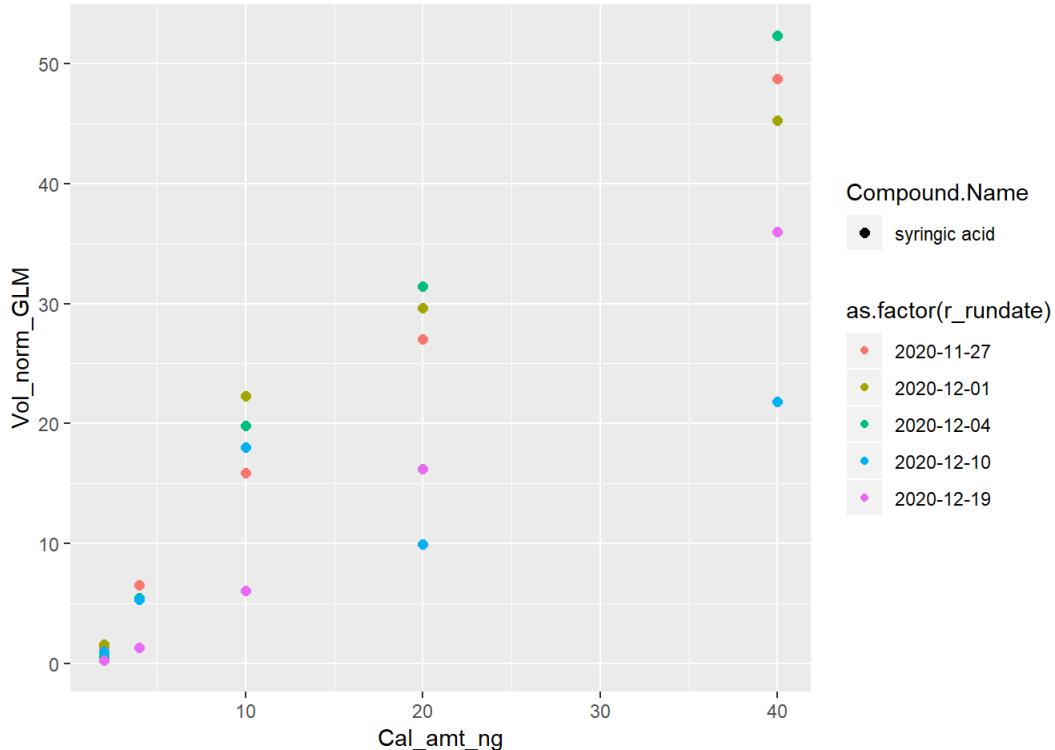
```
## Warning: Using size for a discrete variable is not advised.
```



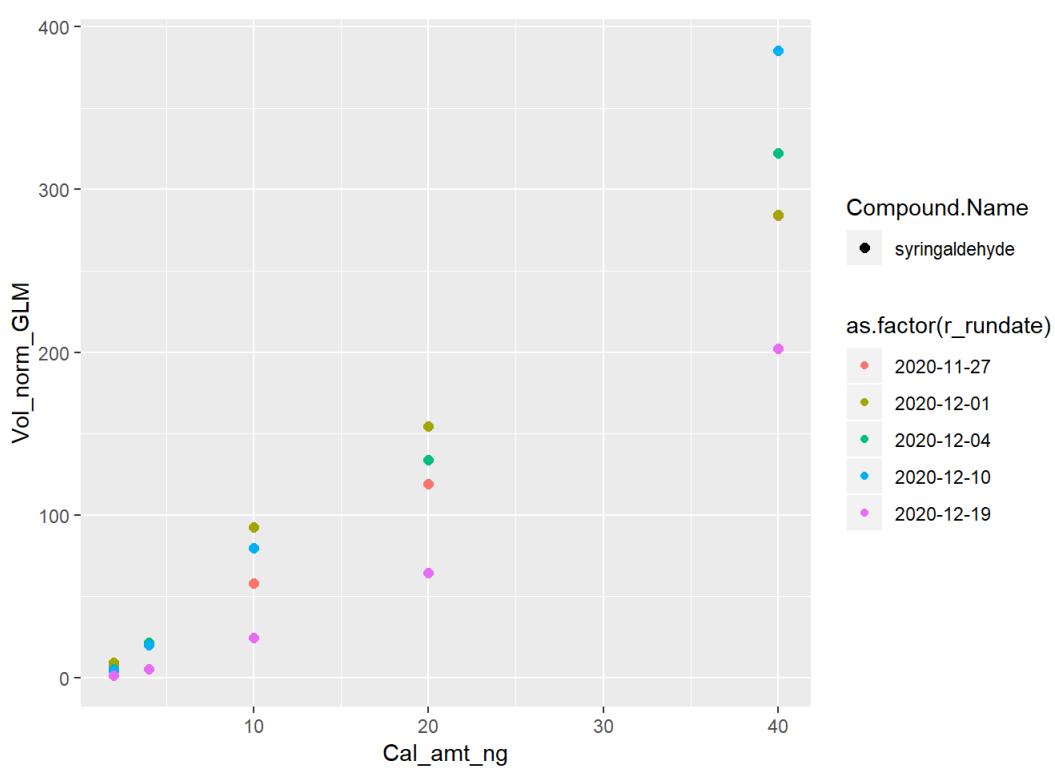
```
## Warning: Using size for a discrete variable is not advised.
```



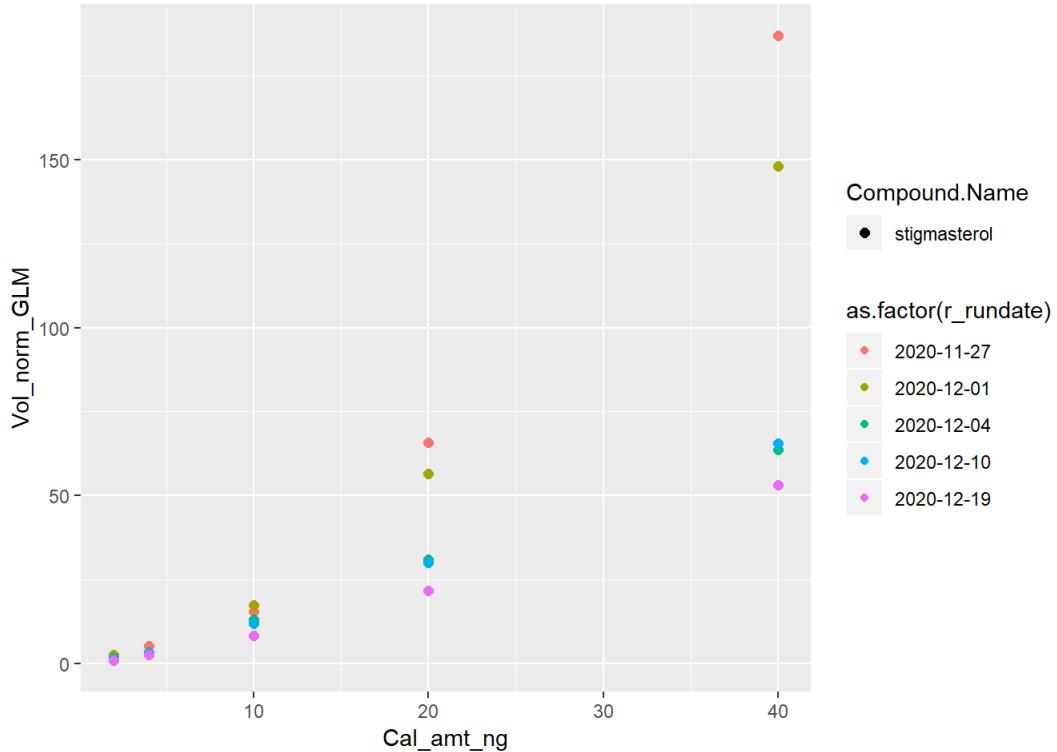
```
## Warning: Using size for a discrete variable is not advised.
```



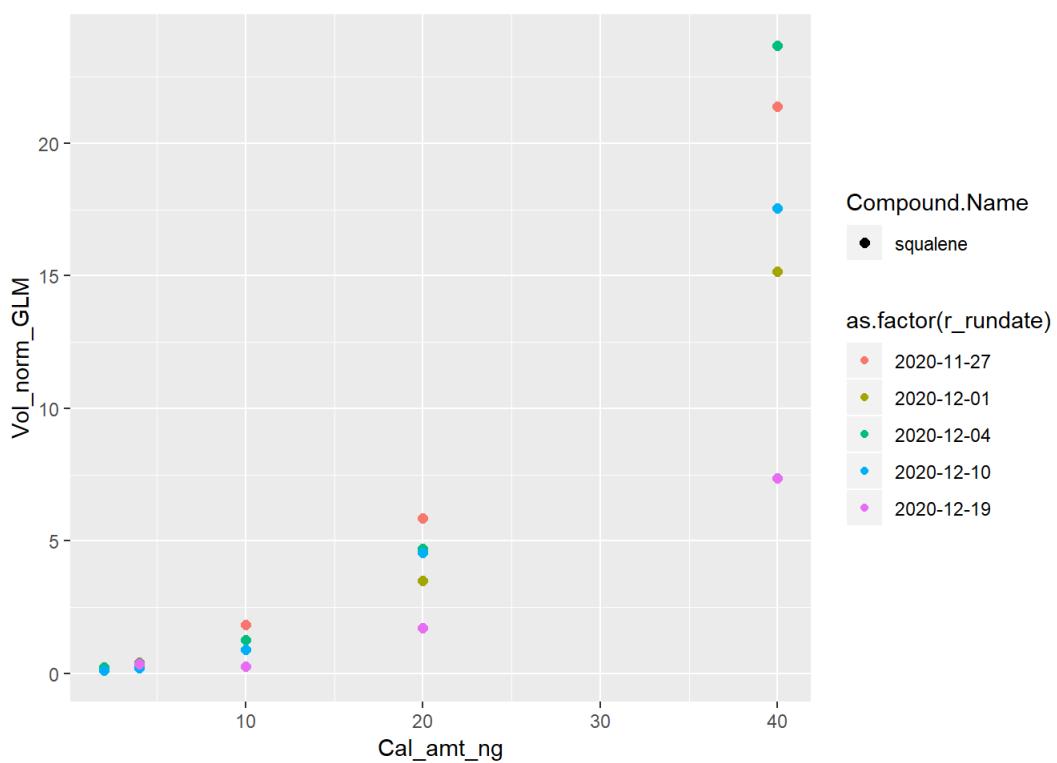
```
## Warning: Using size for a discrete variable is not advised.
```



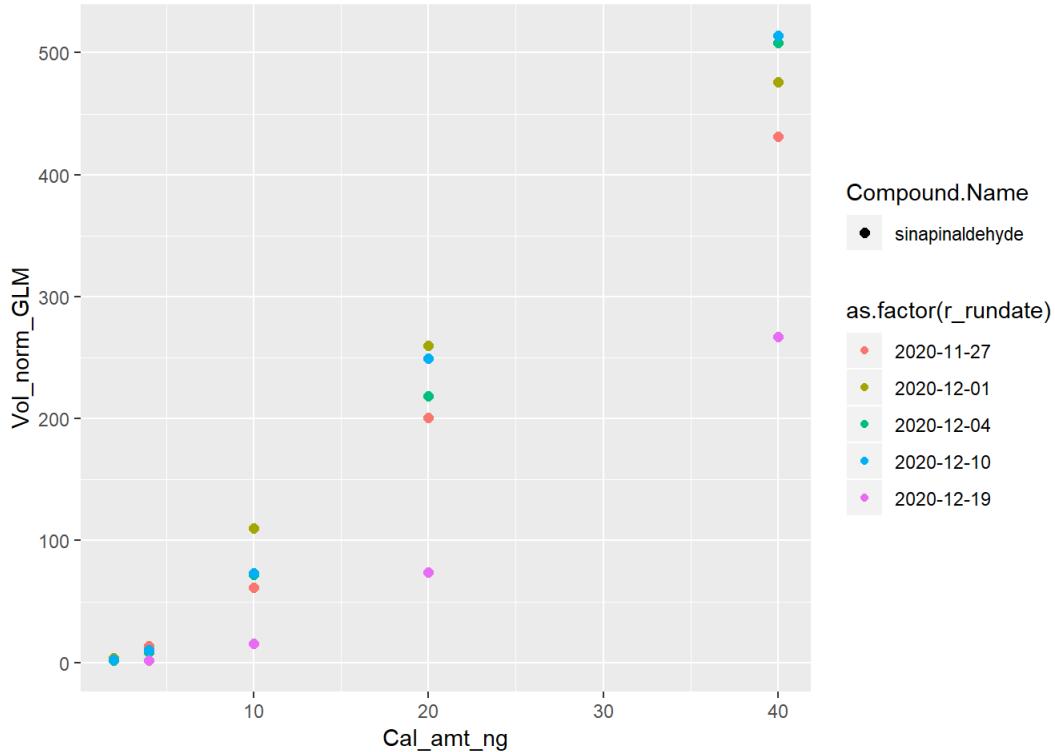
```
## Warning: Using size for a discrete variable is not advised.
```



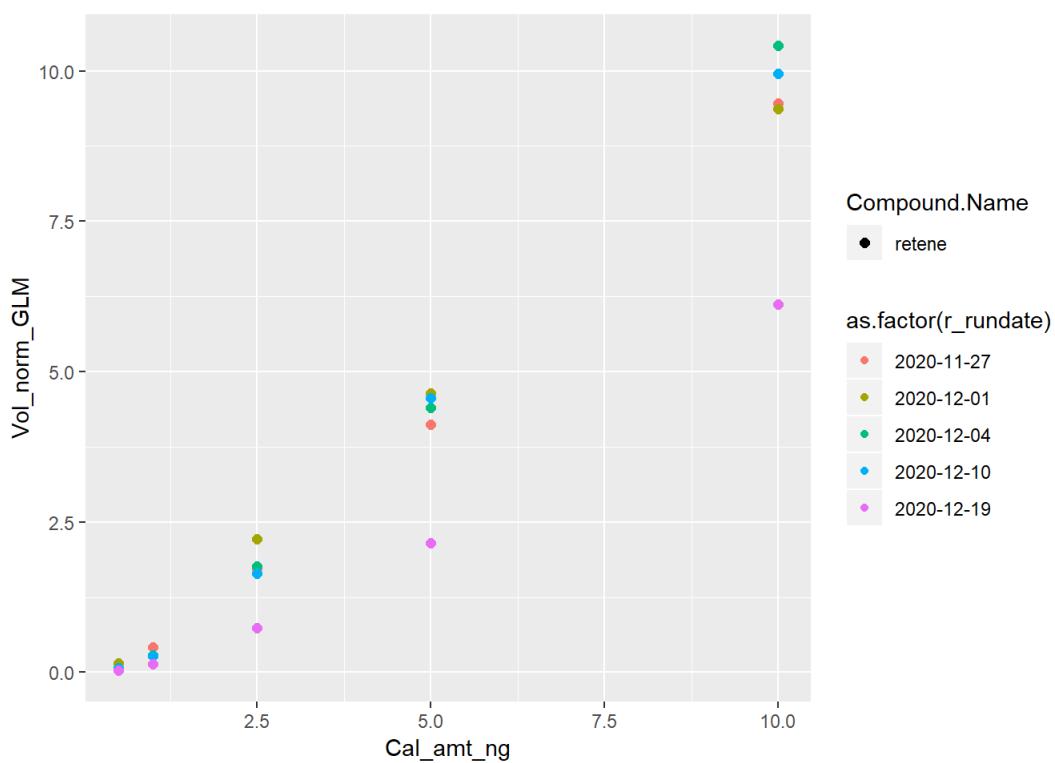
```
## Warning: Using size for a discrete variable is not advised.
```



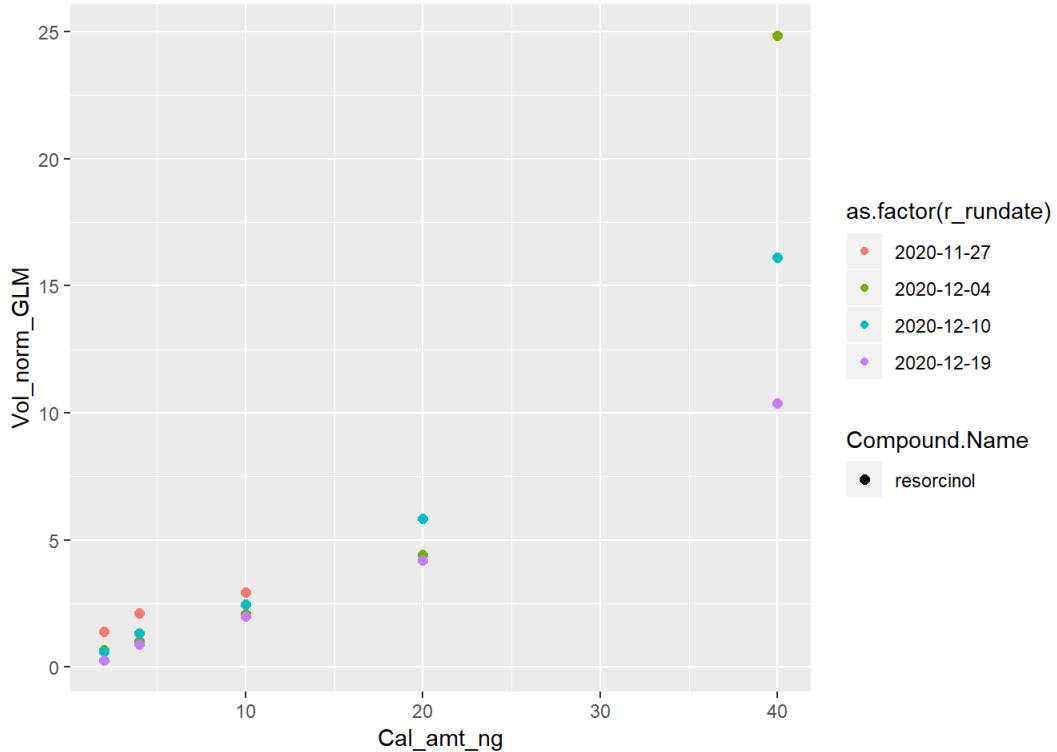
```
## Warning: Using size for a discrete variable is not advised.
```



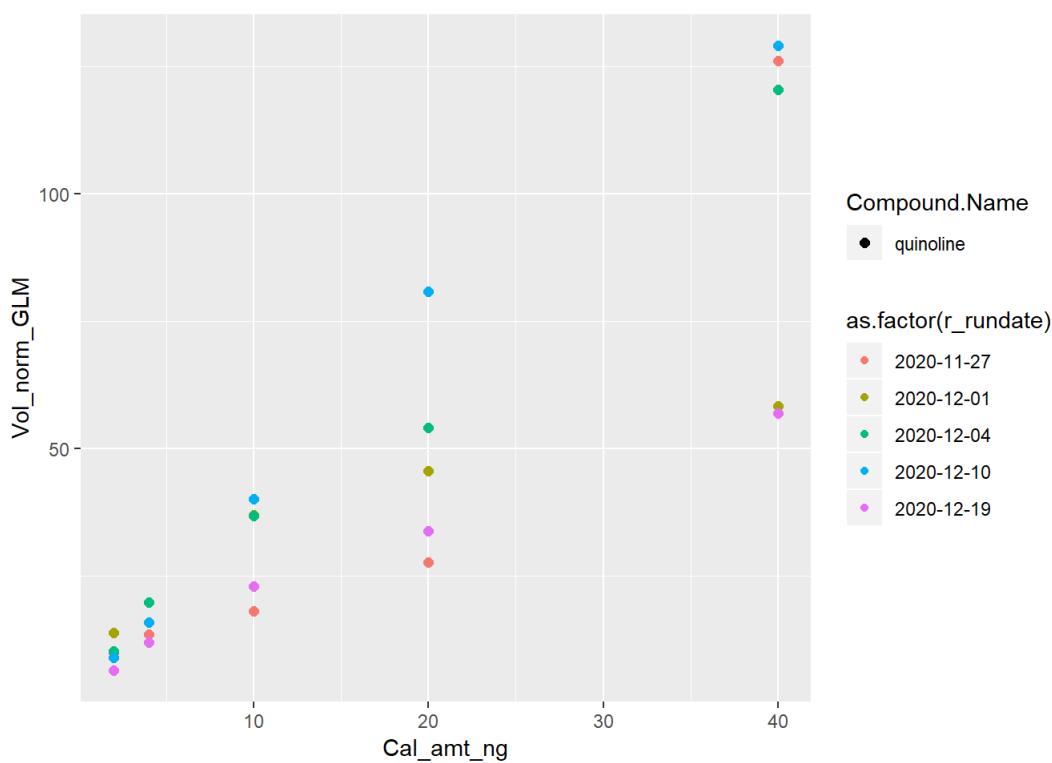
```
## Warning: Using size for a discrete variable is not advised.
```



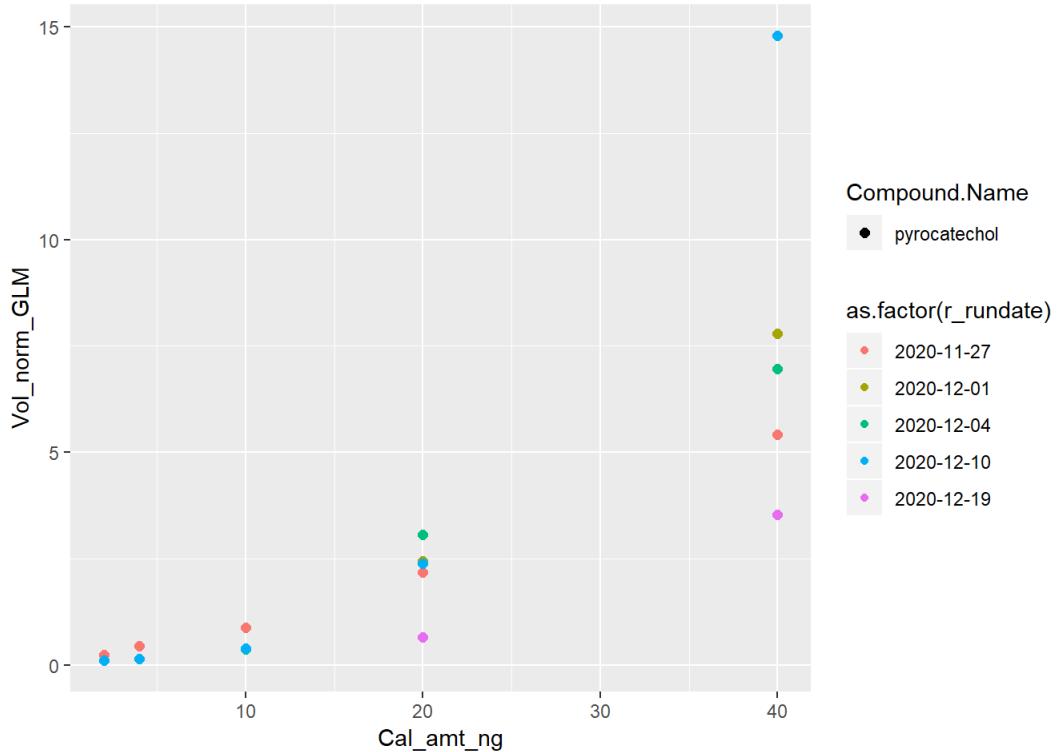
```
## Warning: Using size for a discrete variable is not advised.
```



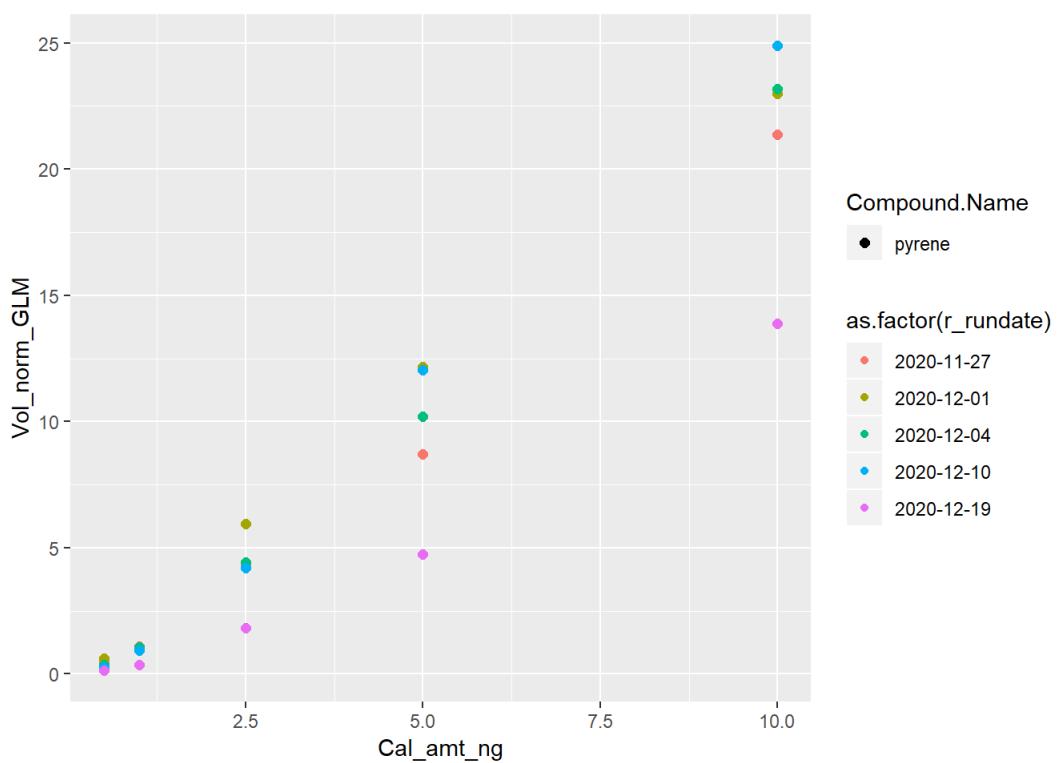
```
## Warning: Using size for a discrete variable is not advised.
```



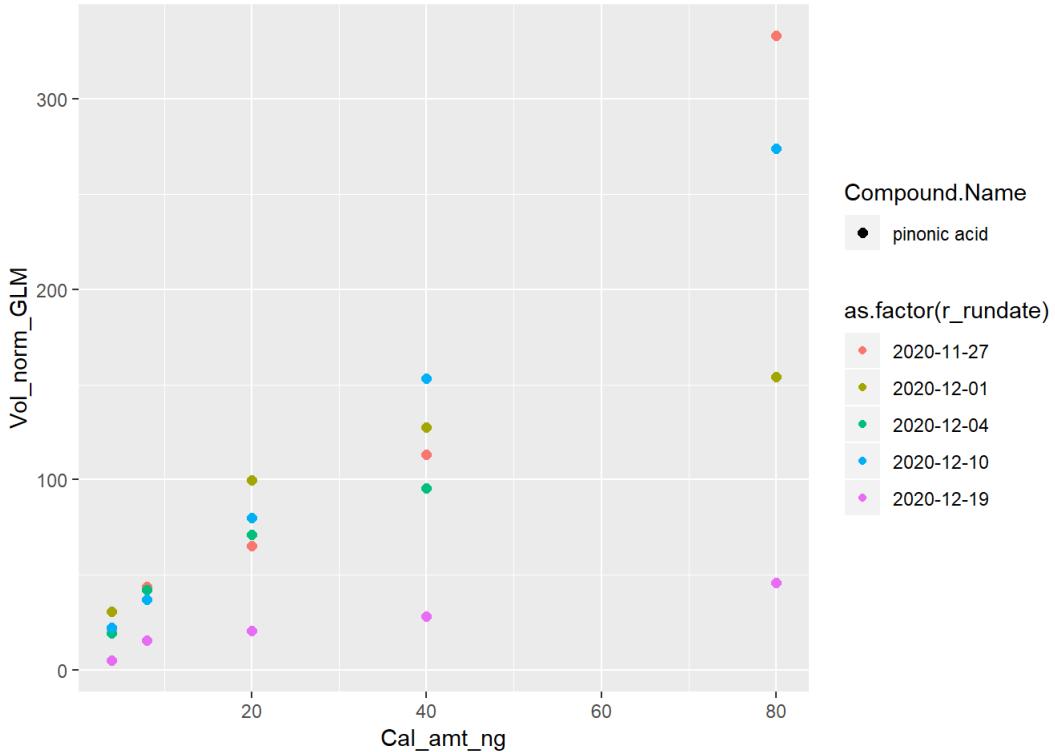
```
## Warning: Using size for a discrete variable is not advised.
```



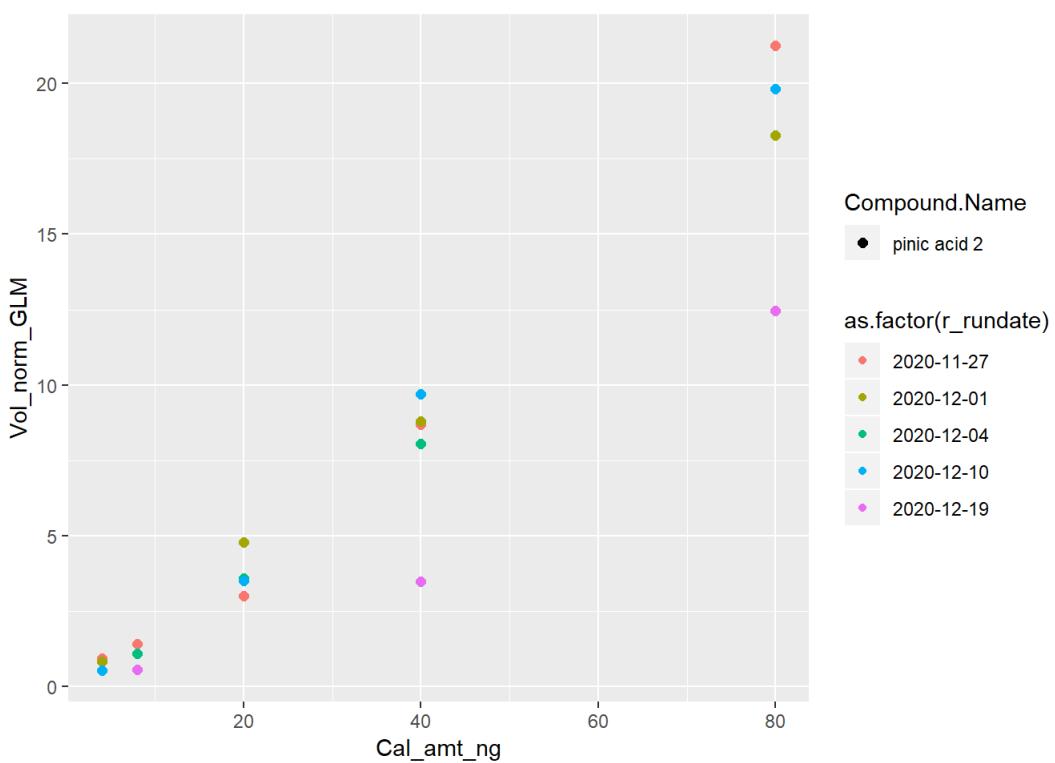
```
## Warning: Using size for a discrete variable is not advised.
```



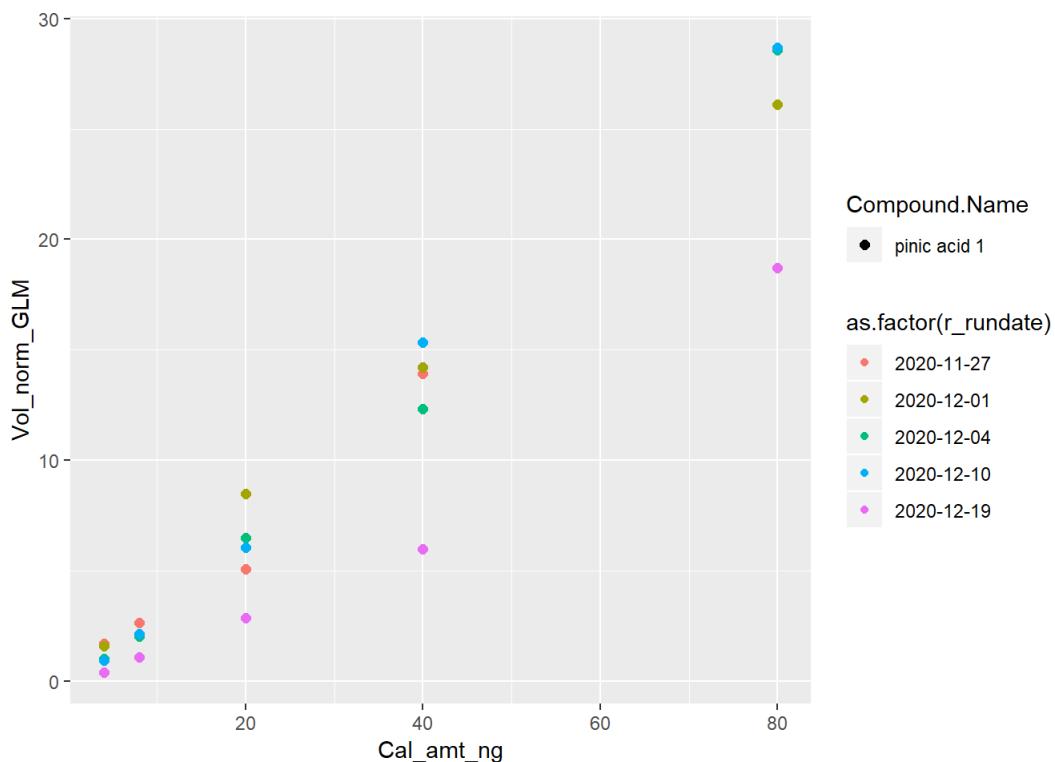
```
## Warning: Using size for a discrete variable is not advised.
```



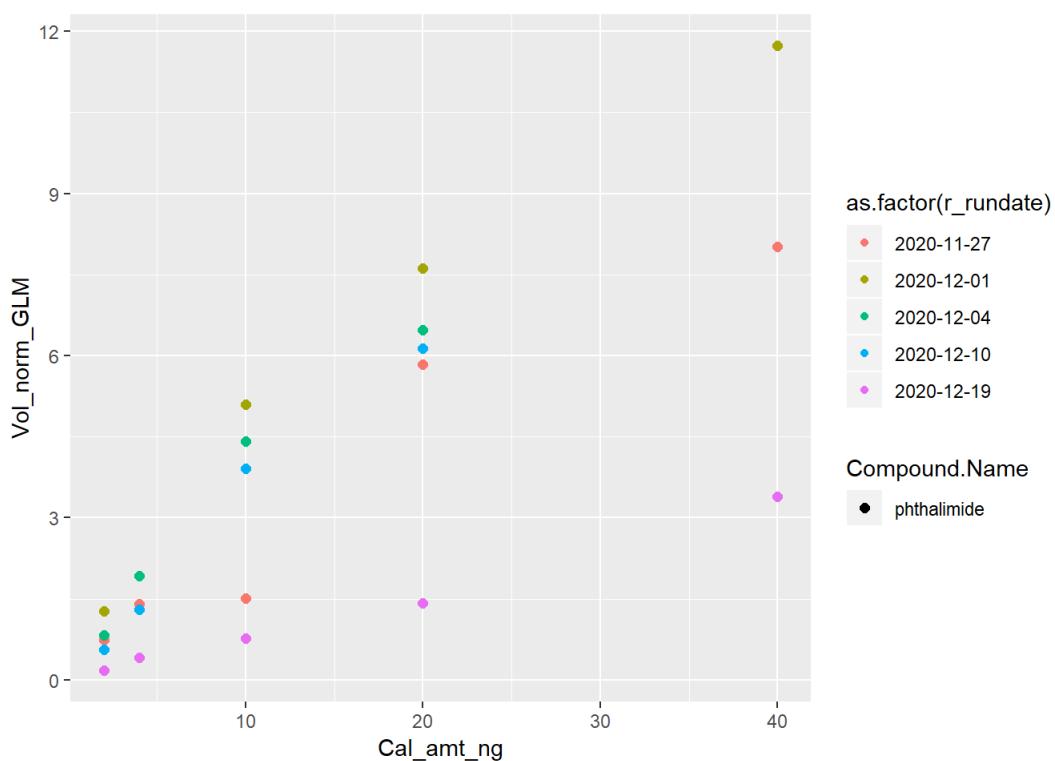
```
## Warning: Using size for a discrete variable is not advised.
```



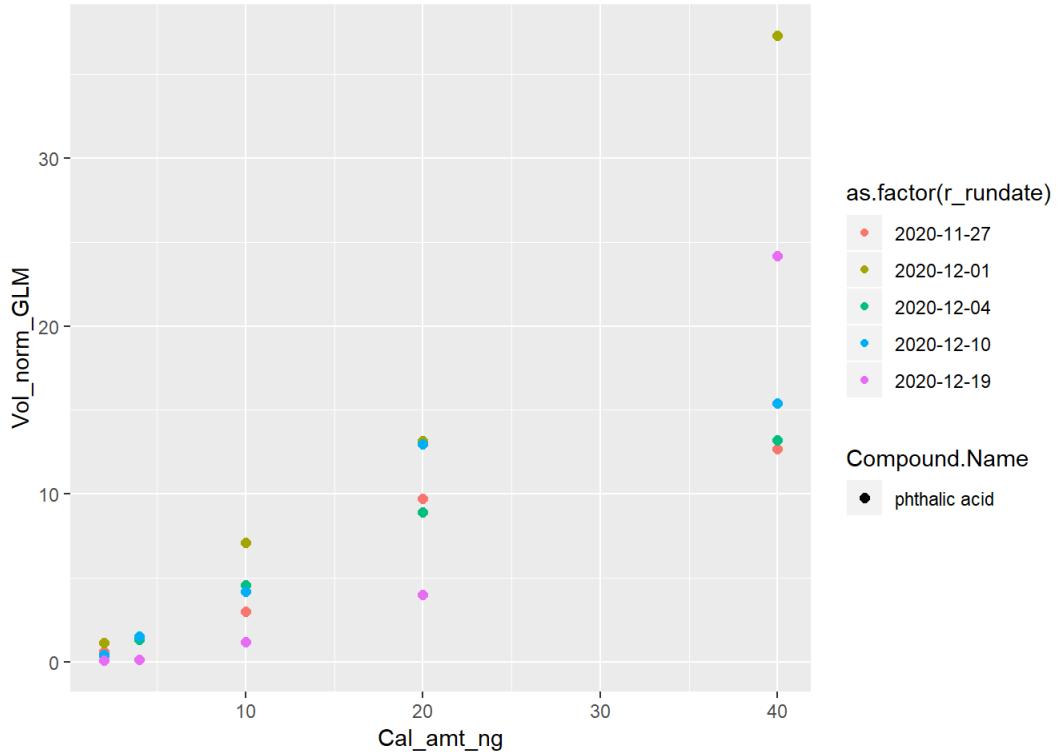
```
## Warning: Using size for a discrete variable is not advised.
```



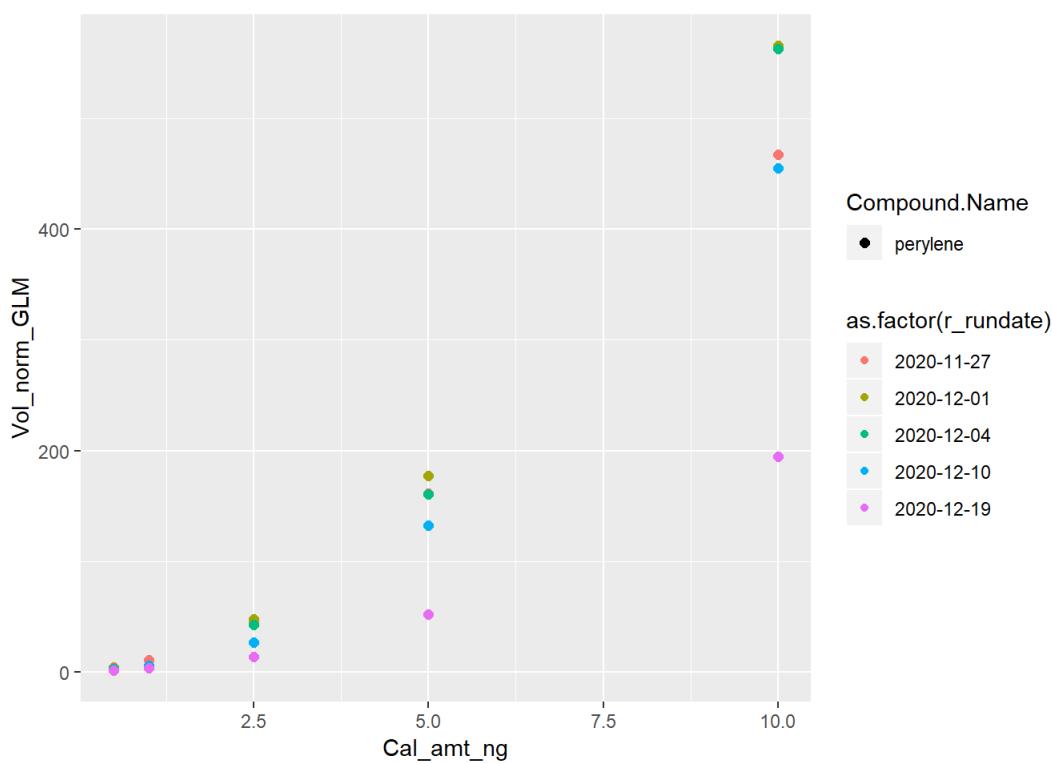
```
## Warning: Using size for a discrete variable is not advised.
```



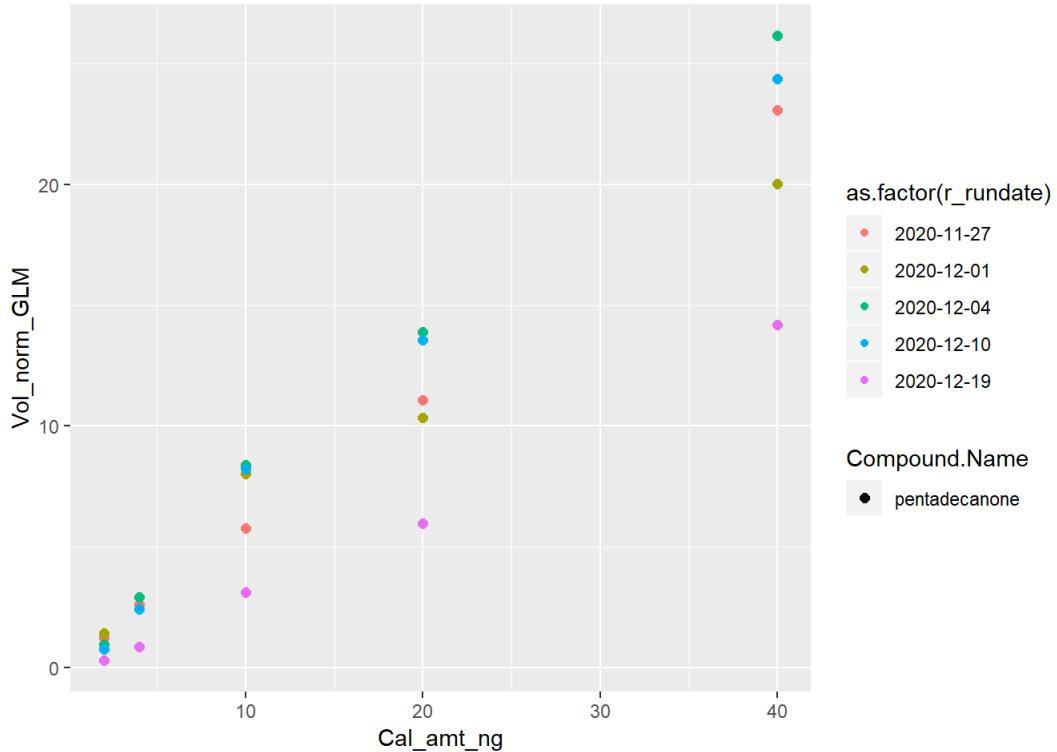
```
## Warning: Using size for a discrete variable is not advised.
```



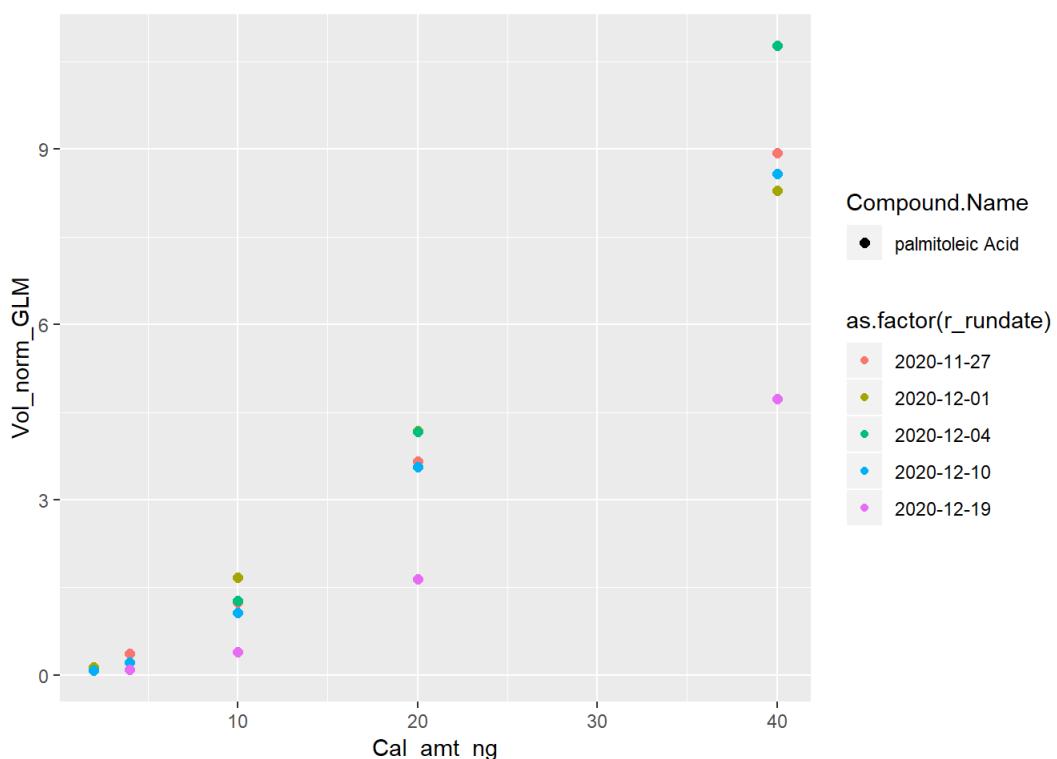
```
## Warning: Using size for a discrete variable is not advised.
```



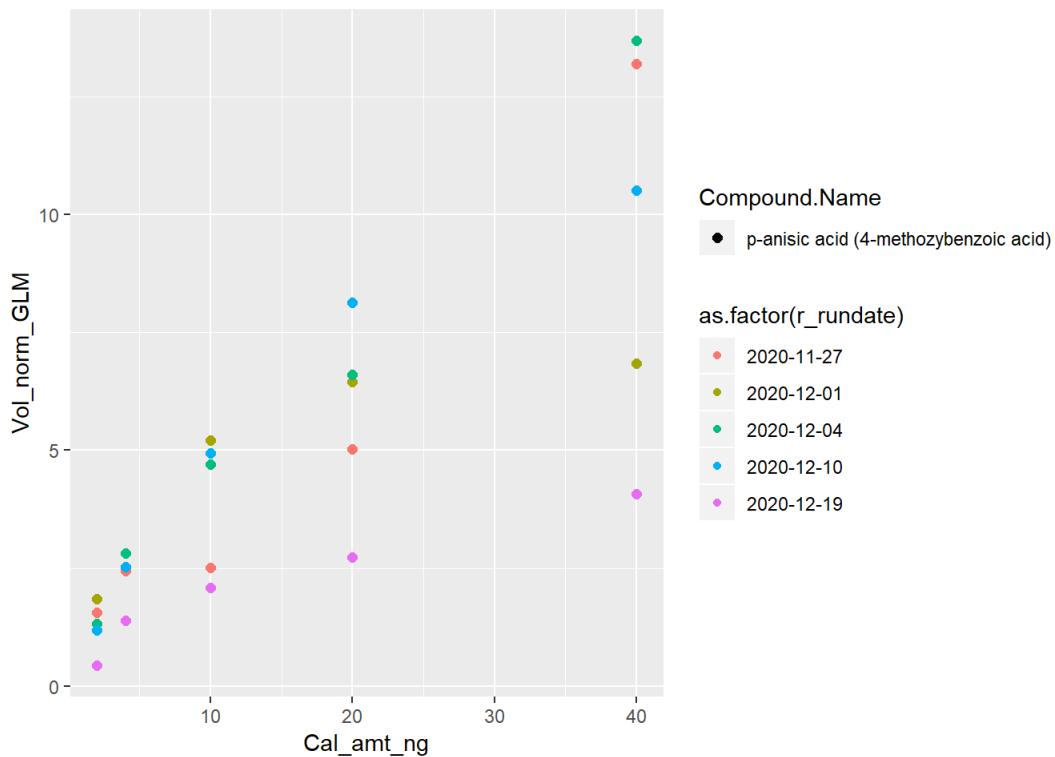
```
## Warning: Using size for a discrete variable is not advised.
```



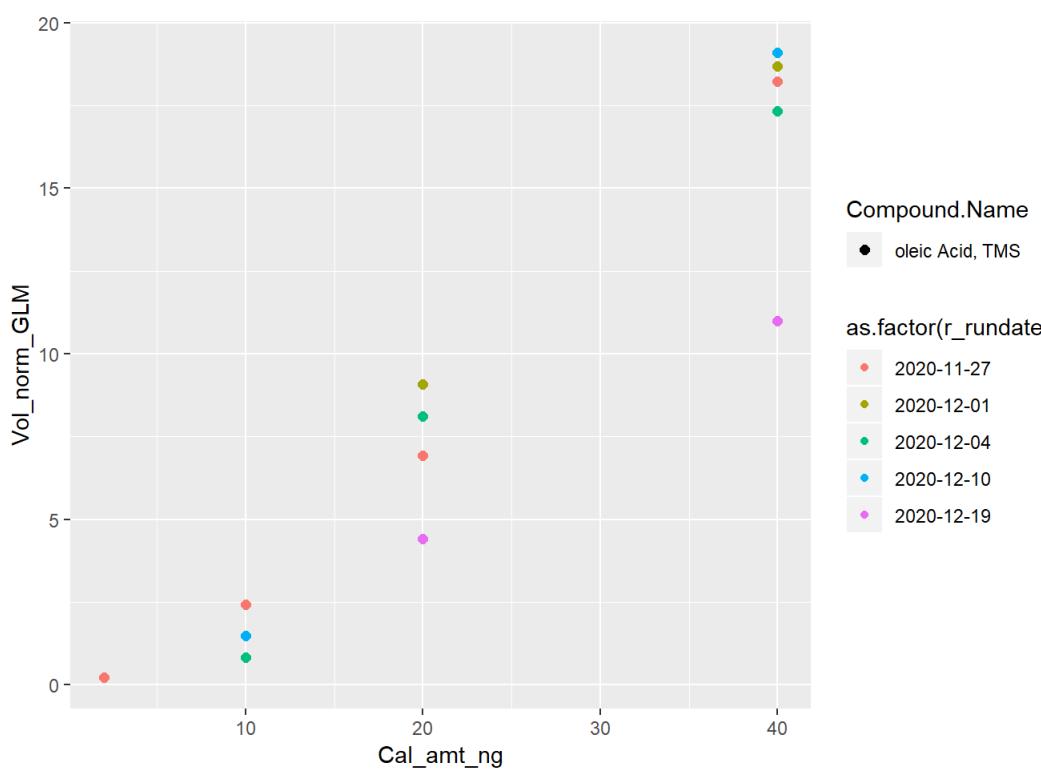
```
## Warning: Using size for a discrete variable is not advised.
```



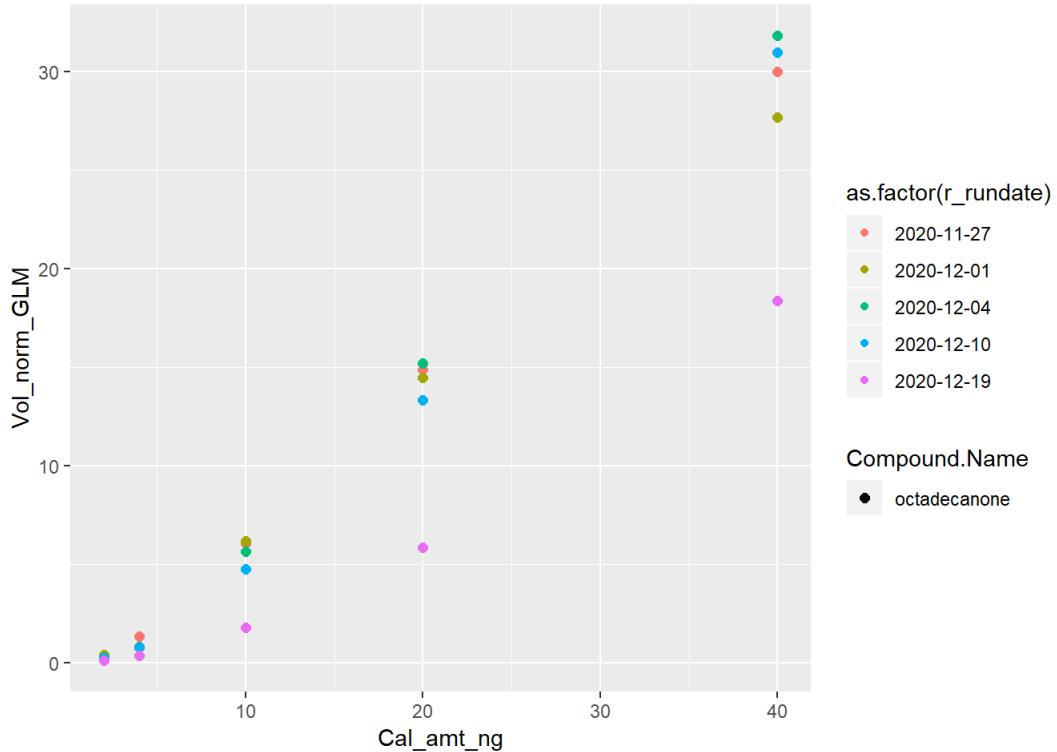
```
## Warning: Using size for a discrete variable is not advised.
```



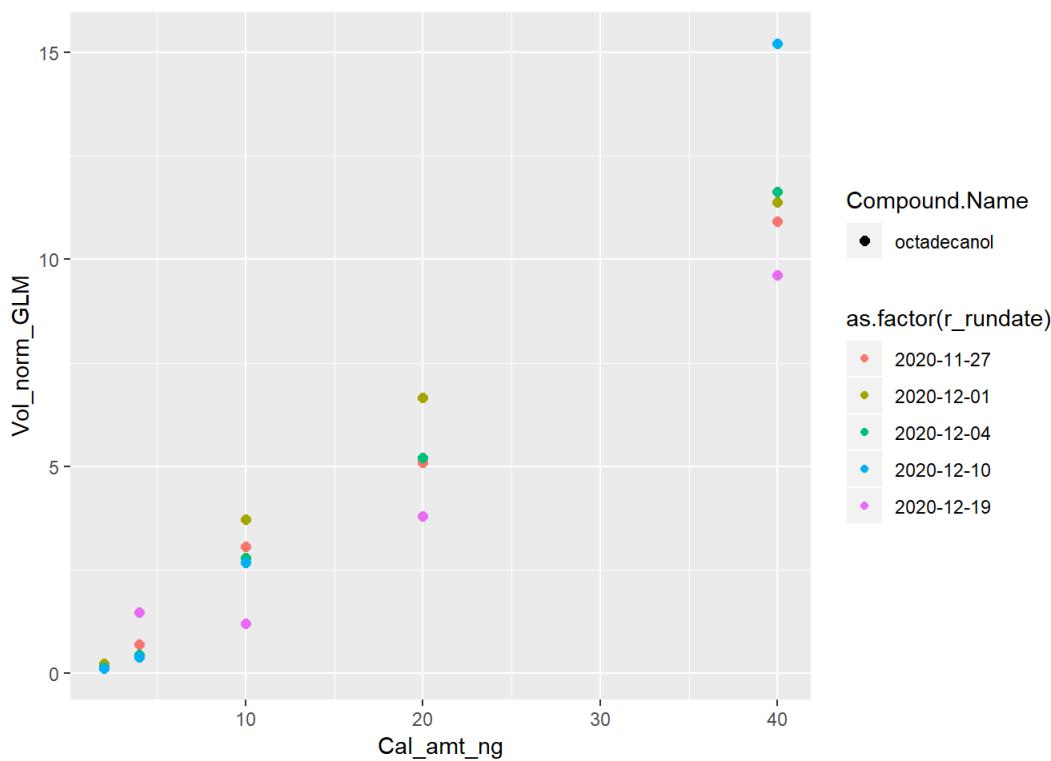
```
## Warning: Using size for a discrete variable is not advised.
```



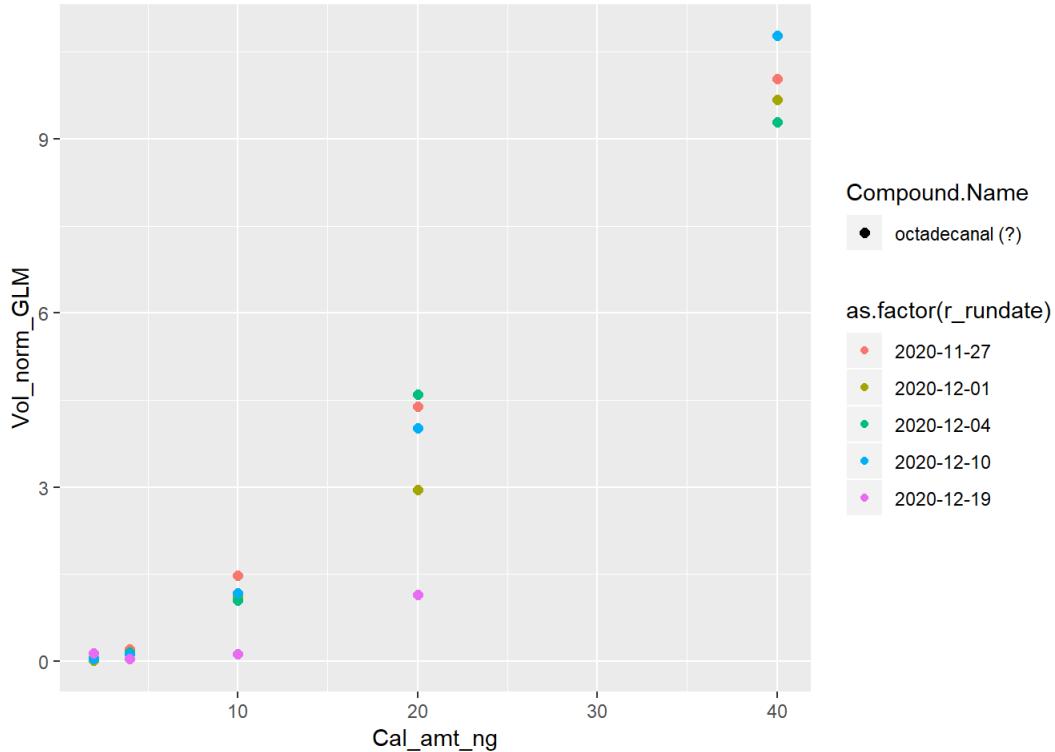
```
## Warning: Using size for a discrete variable is not advised.
```



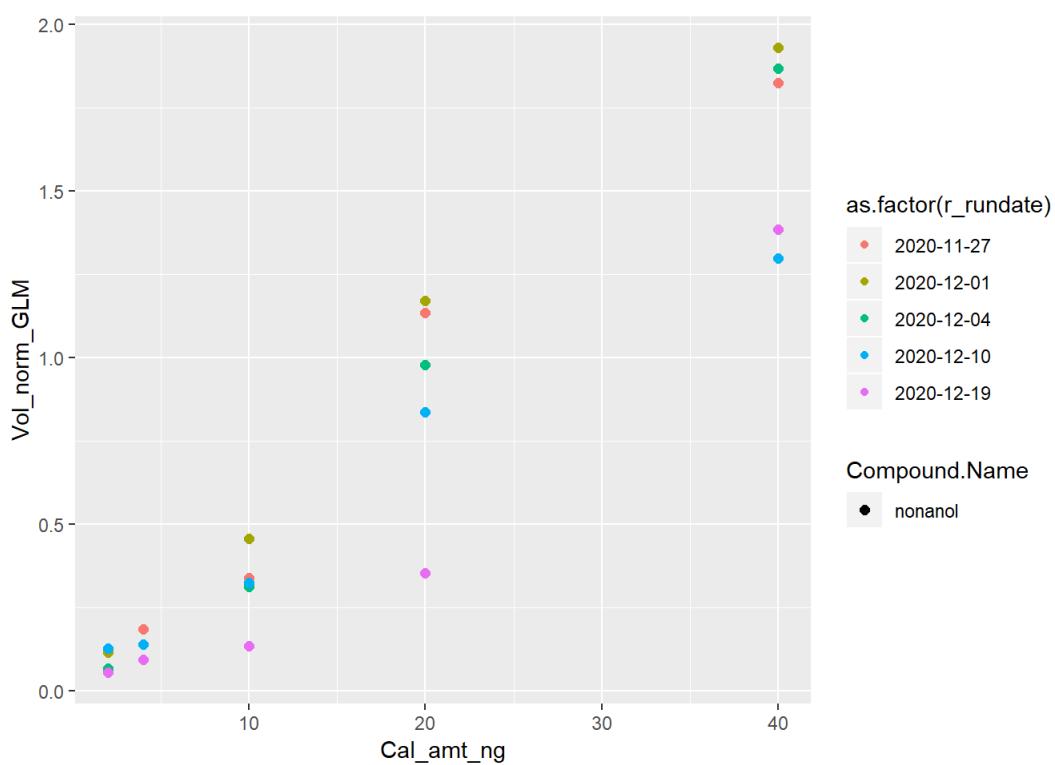
```
## Warning: Using size for a discrete variable is not advised.
```



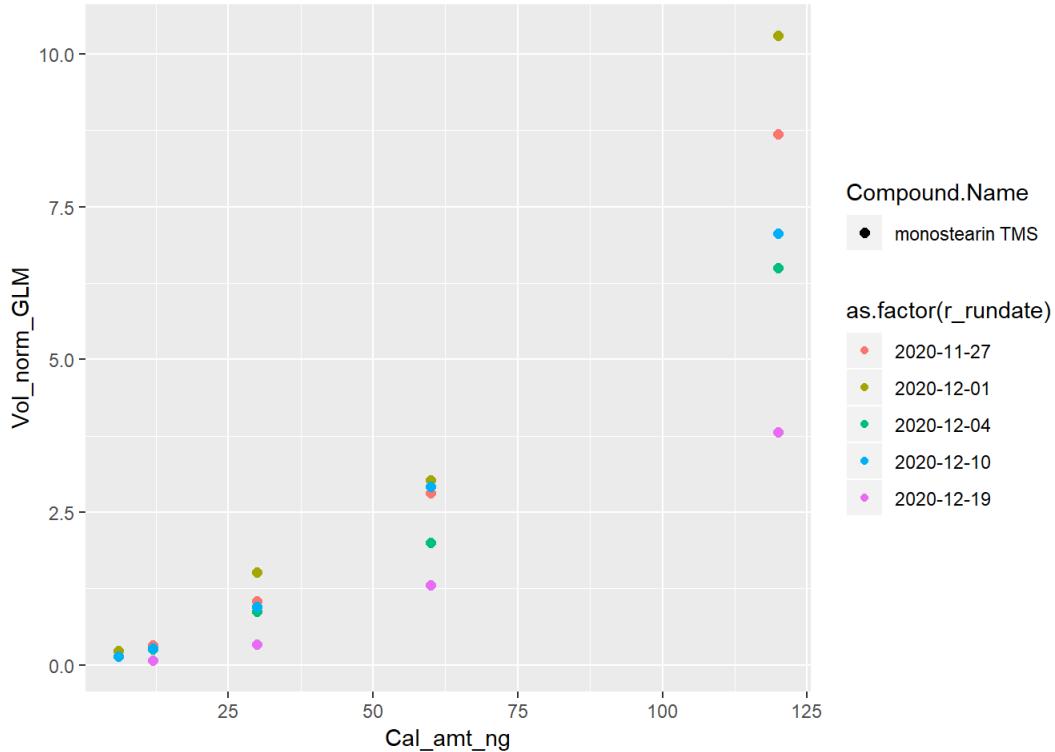
```
## Warning: Using size for a discrete variable is not advised.
```



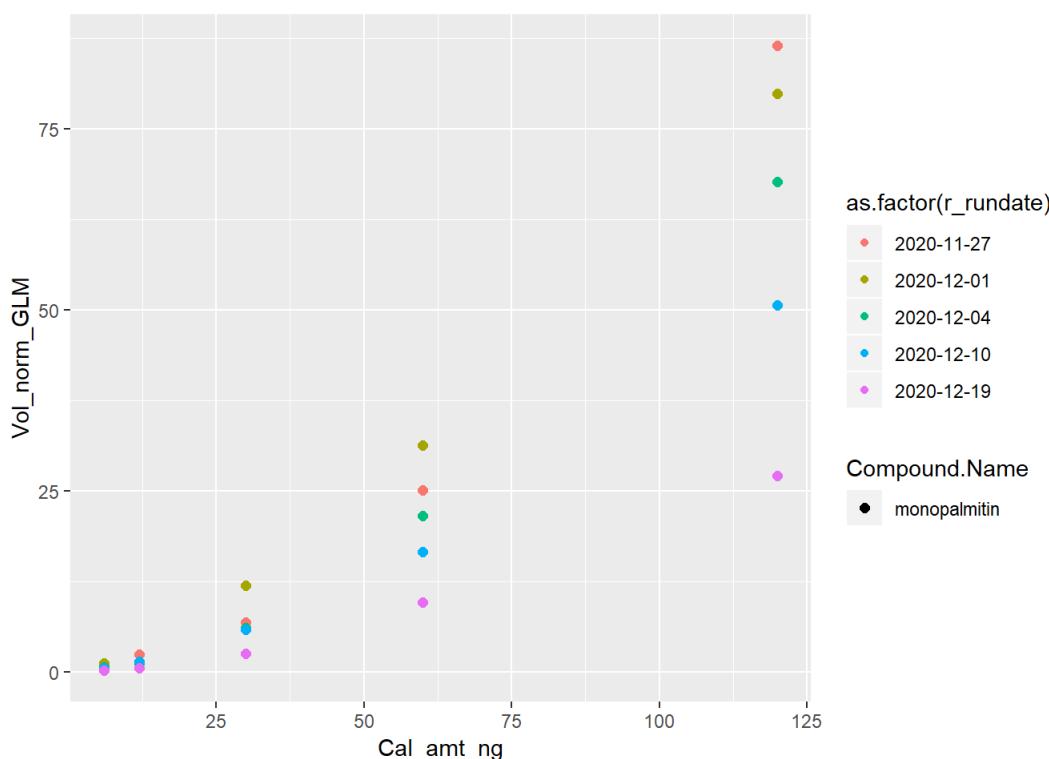
```
## Warning: Using size for a discrete variable is not advised.
```



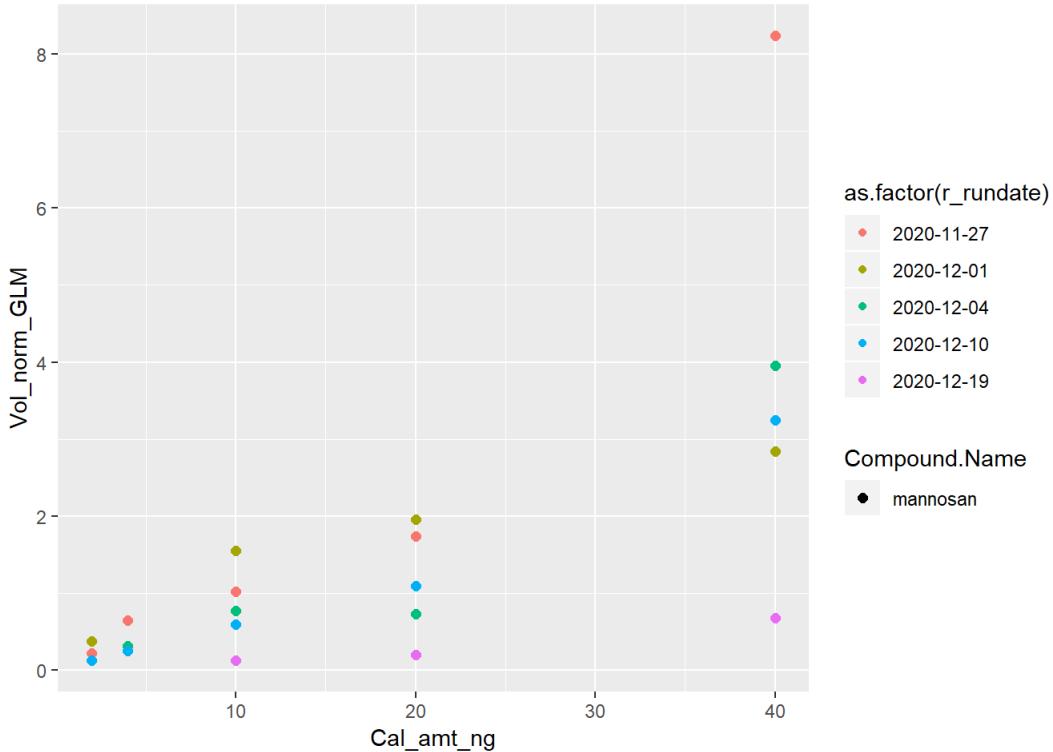
```
## Warning: Using size for a discrete variable is not advised.
```



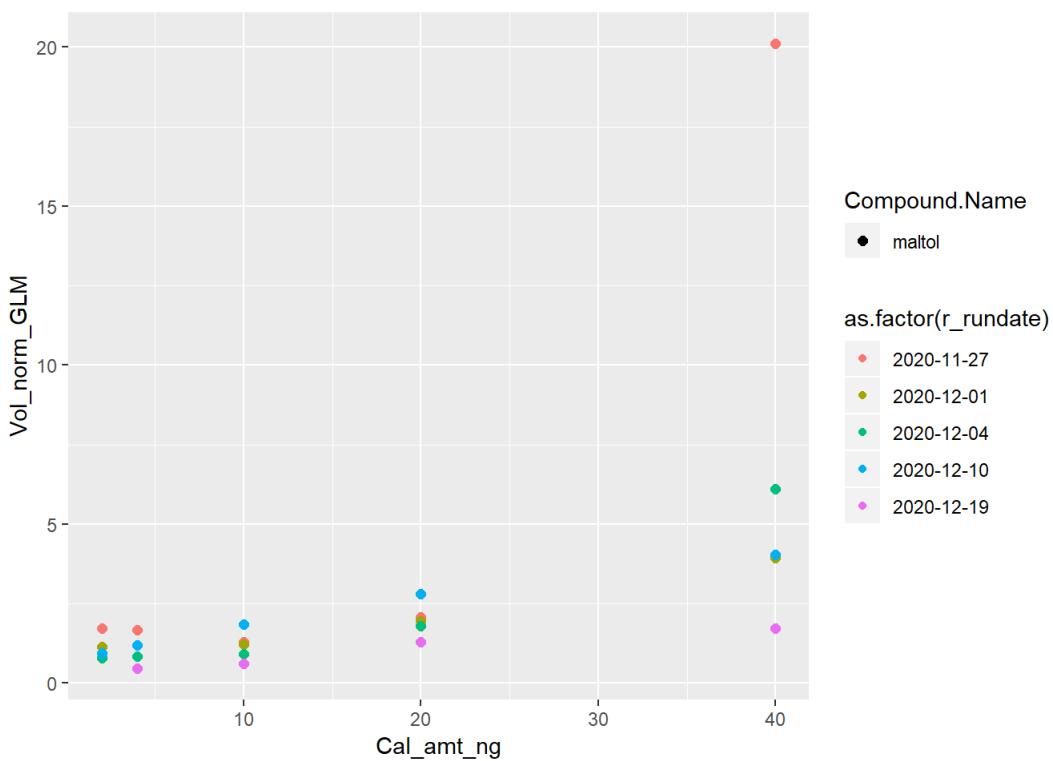
```
## Warning: Using size for a discrete variable is not advised.
```



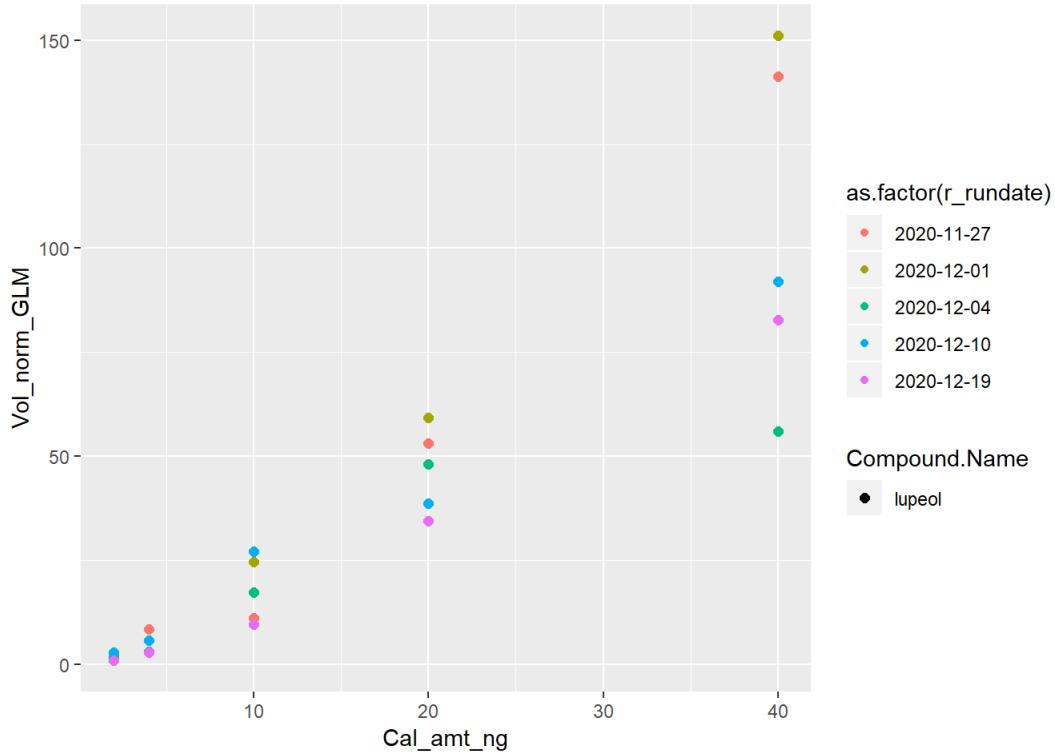
```
## Warning: Using size for a discrete variable is not advised.
```



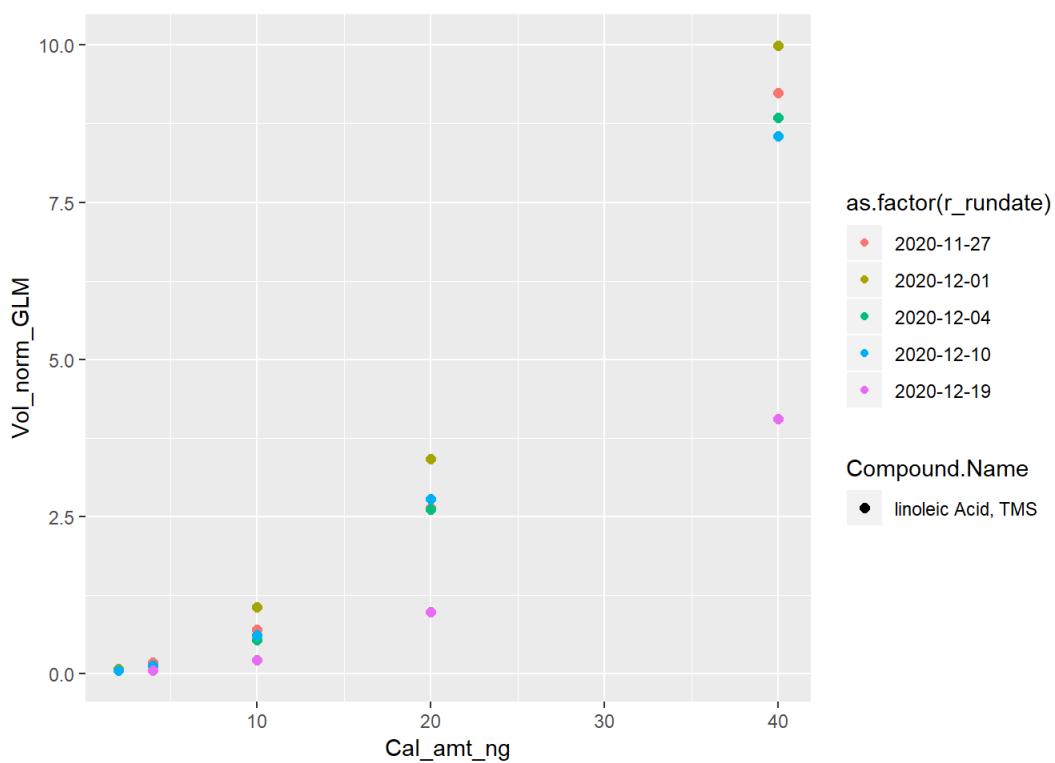
```
## Warning: Using size for a discrete variable is not advised.
```



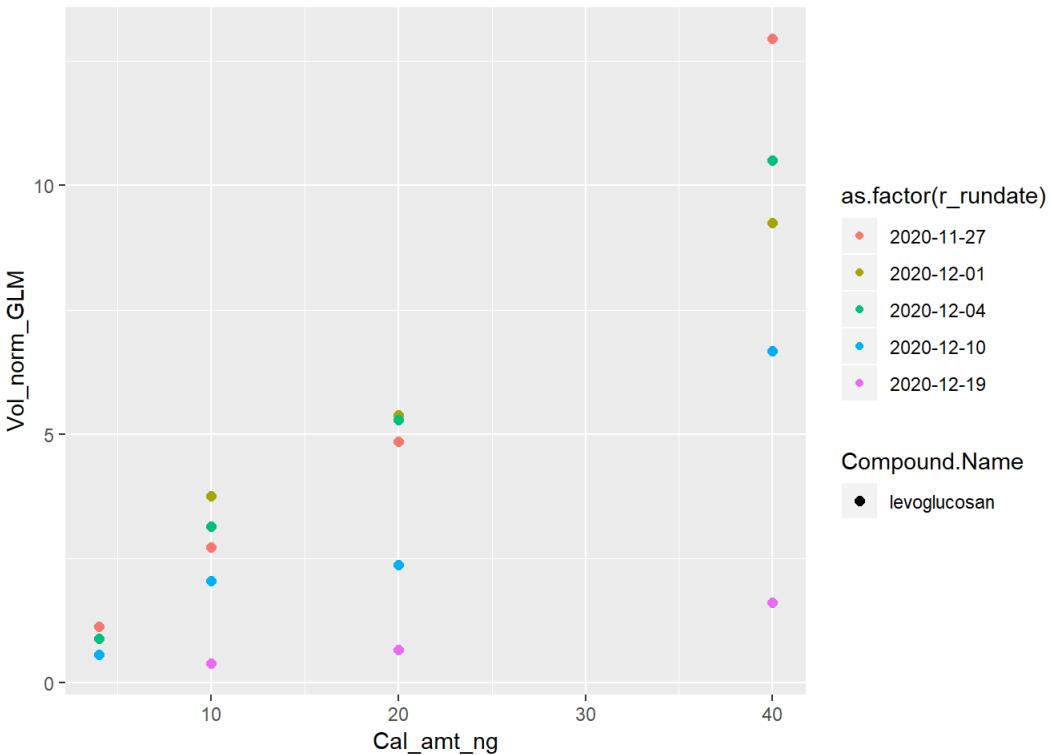
```
## Warning: Using size for a discrete variable is not advised.
```



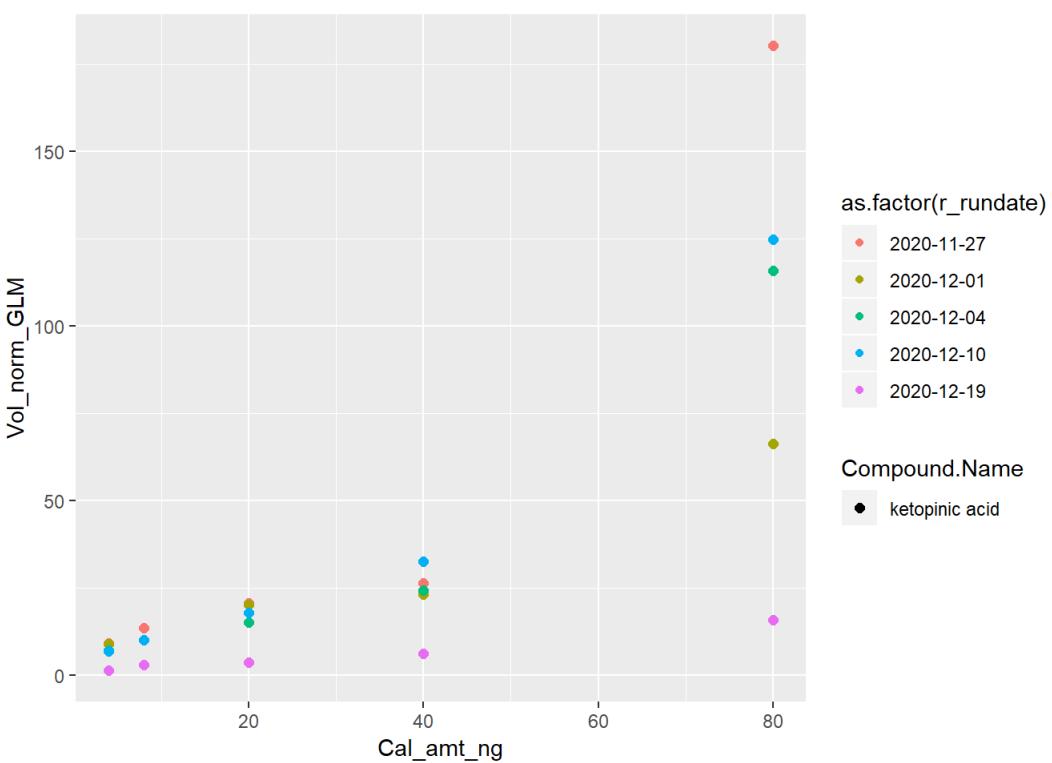
```
## Warning: Using size for a discrete variable is not advised.
```



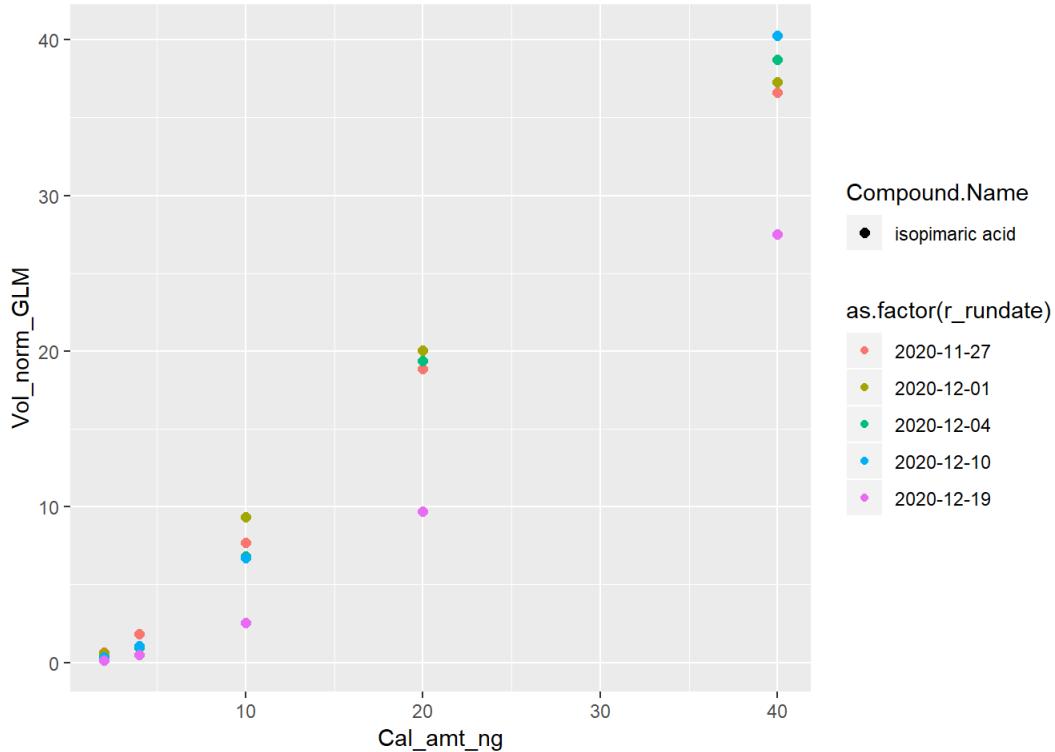
```
## Warning: Using size for a discrete variable is not advised.
```



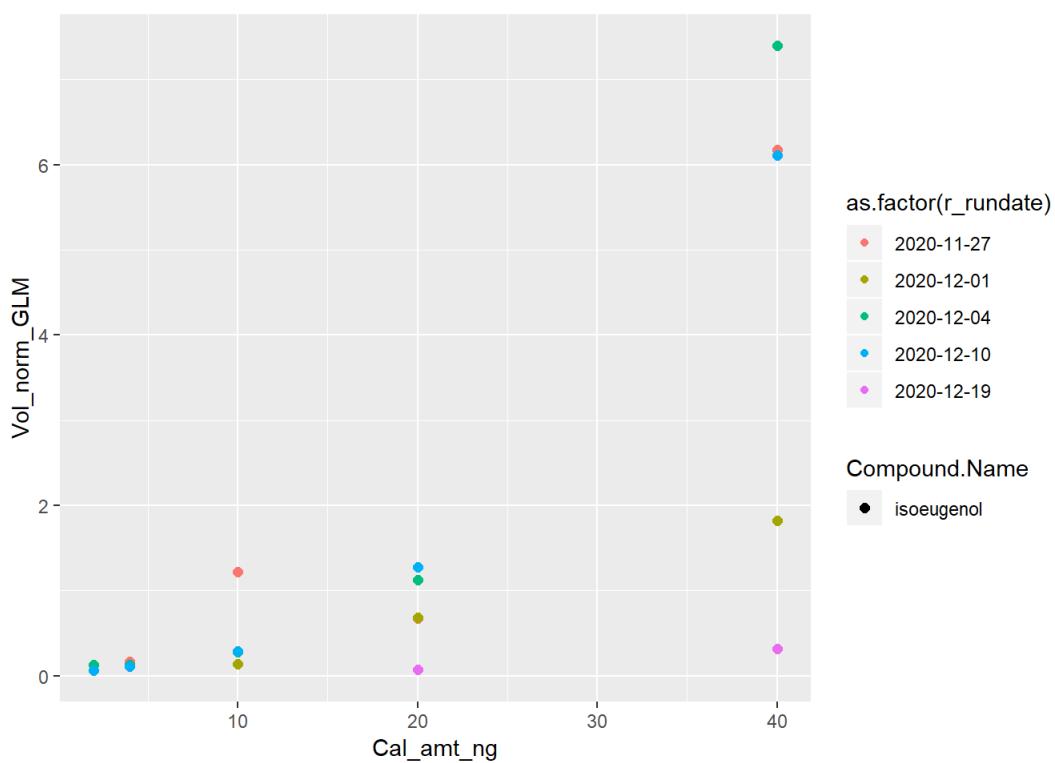
```
## Warning: Using size for a discrete variable is not advised.
```



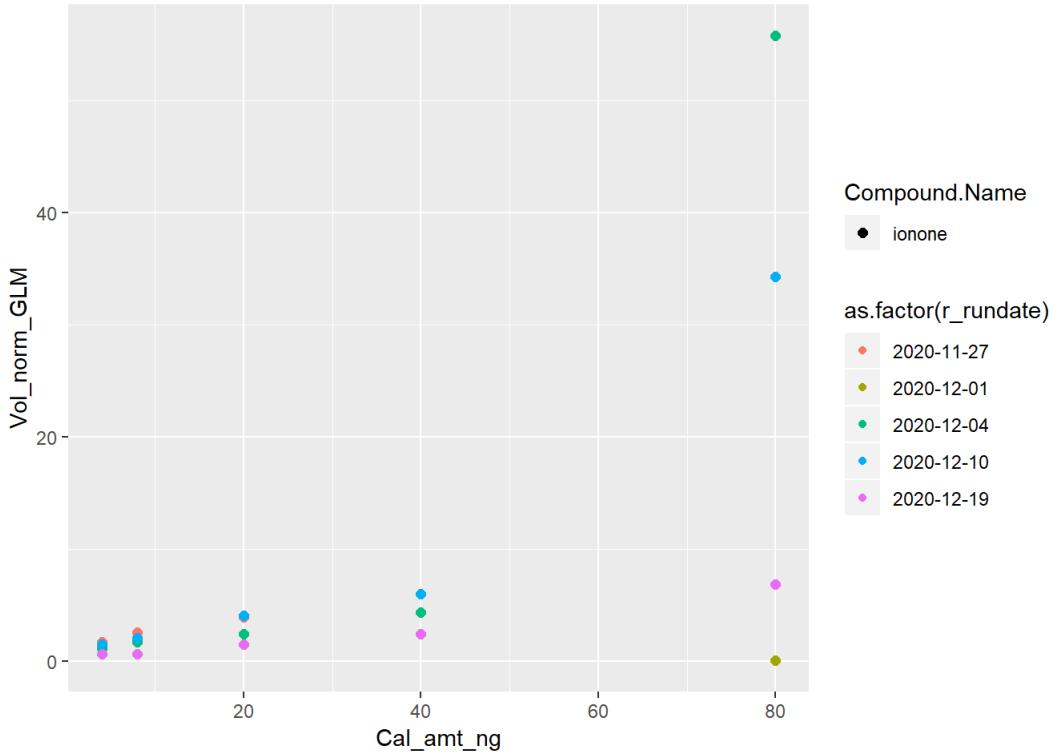
```
## Warning: Using size for a discrete variable is not advised.
```



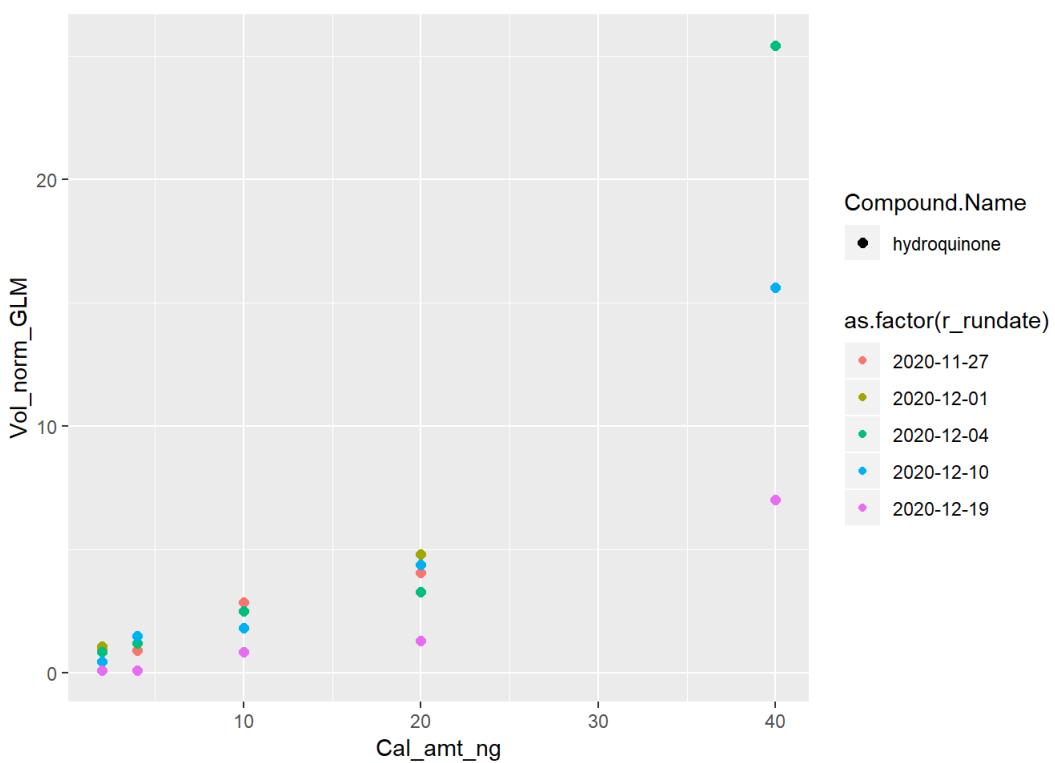
```
## Warning: Using size for a discrete variable is not advised.
```



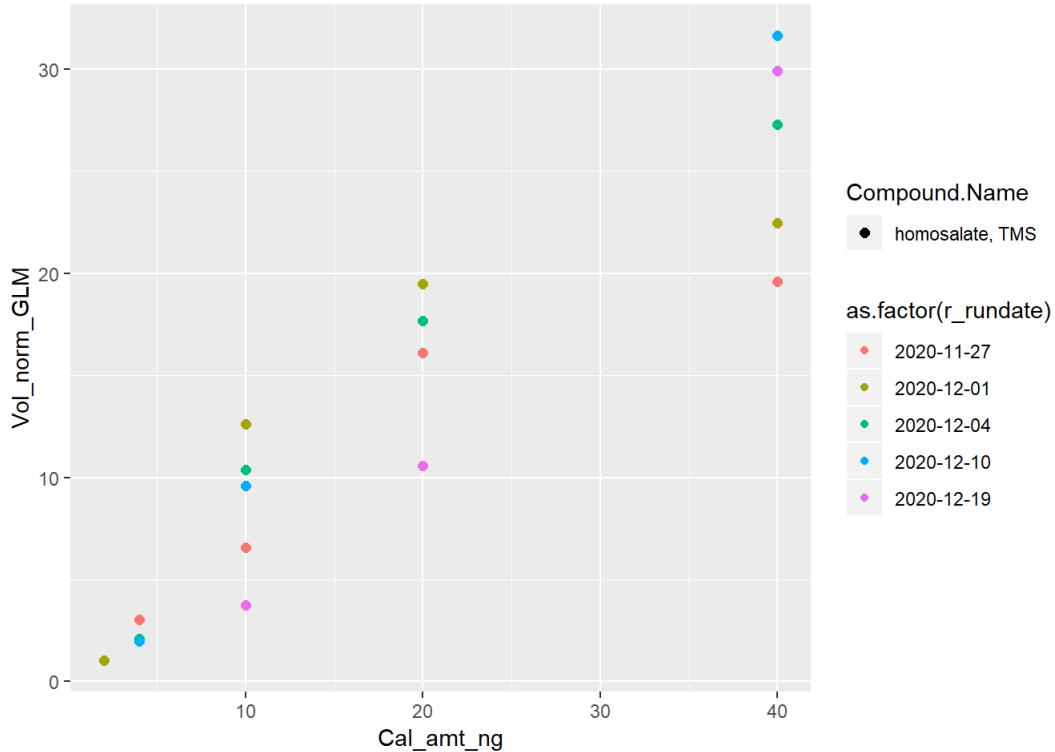
```
## Warning: Using size for a discrete variable is not advised.
```



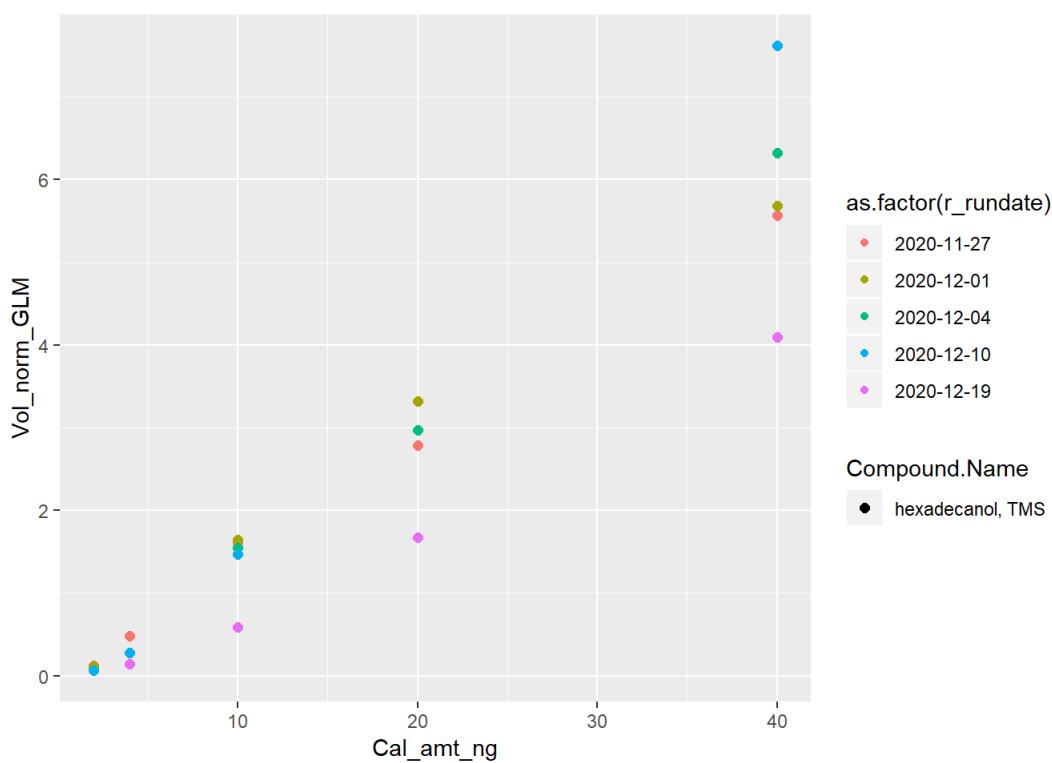
```
## Warning: Using size for a discrete variable is not advised.
```



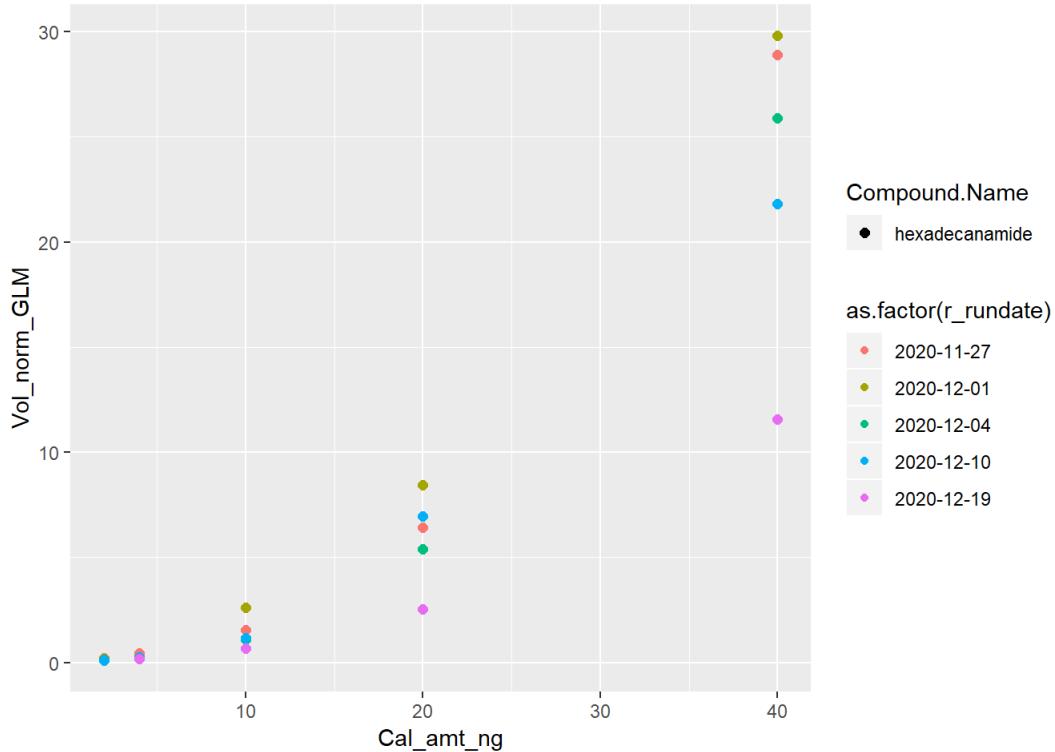
```
## Warning: Using size for a discrete variable is not advised.
```



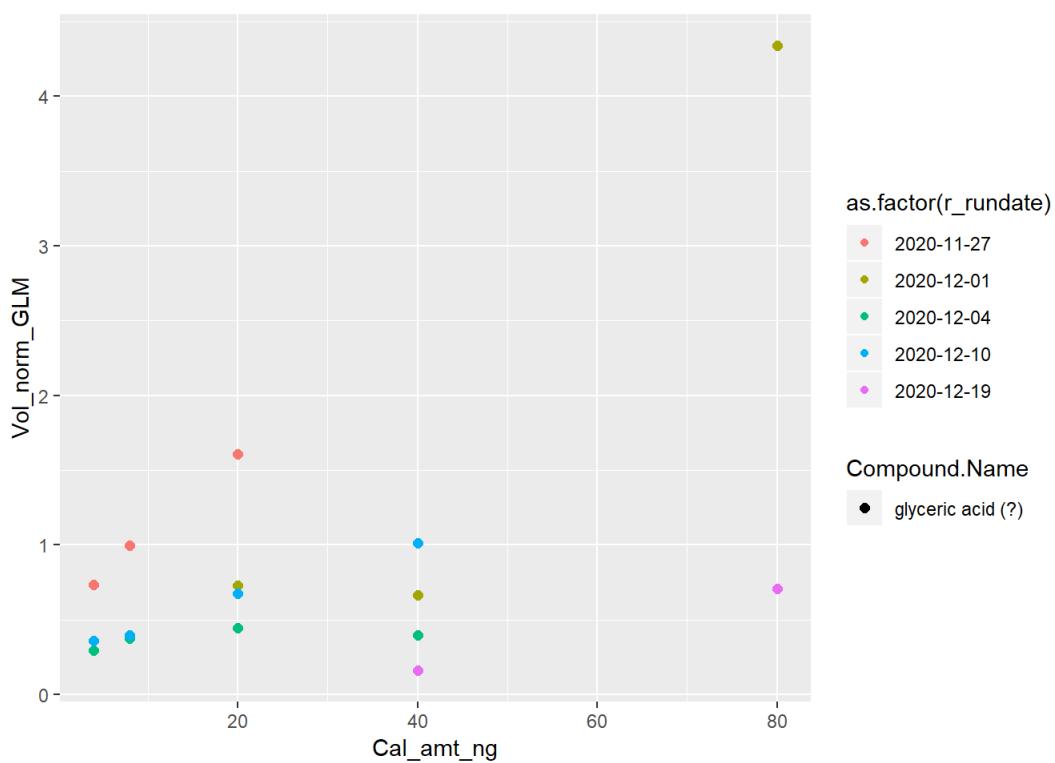
```
## Warning: Using size for a discrete variable is not advised.
```



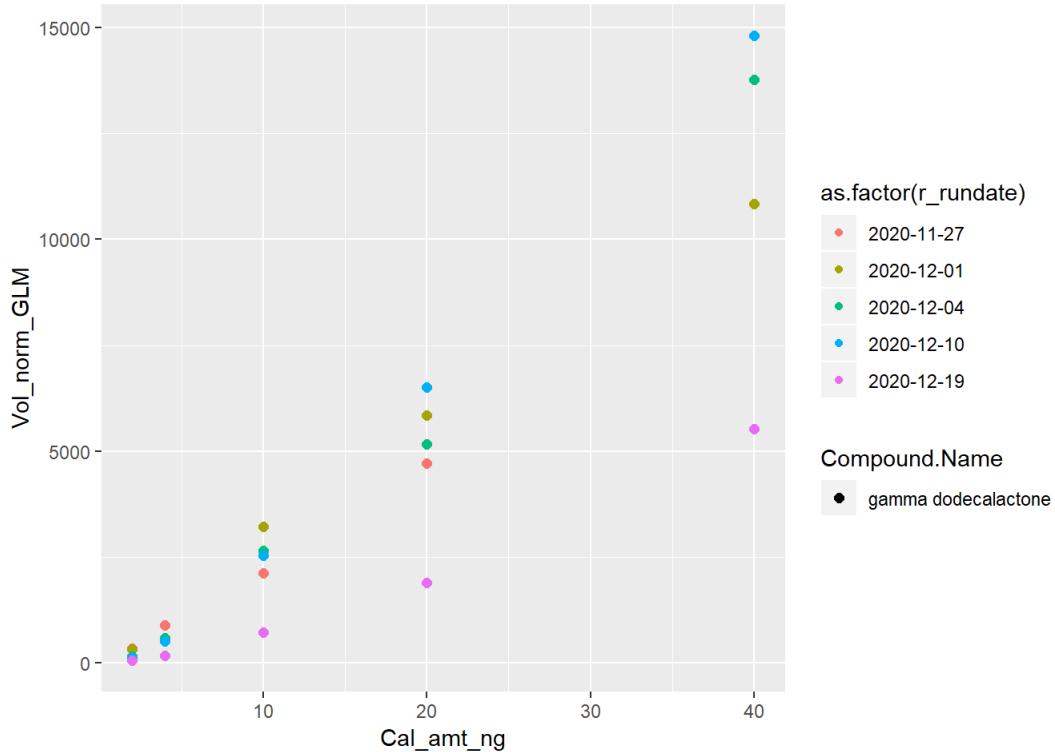
```
## Warning: Using size for a discrete variable is not advised.
```



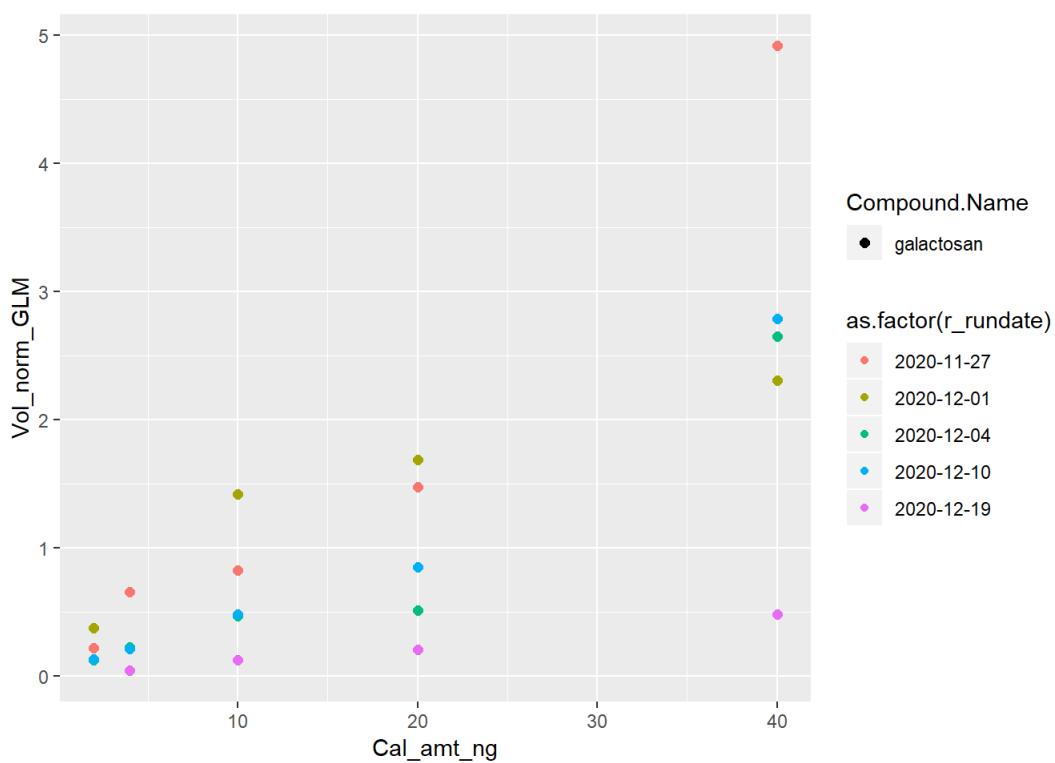
```
## Warning: Using size for a discrete variable is not advised.
```



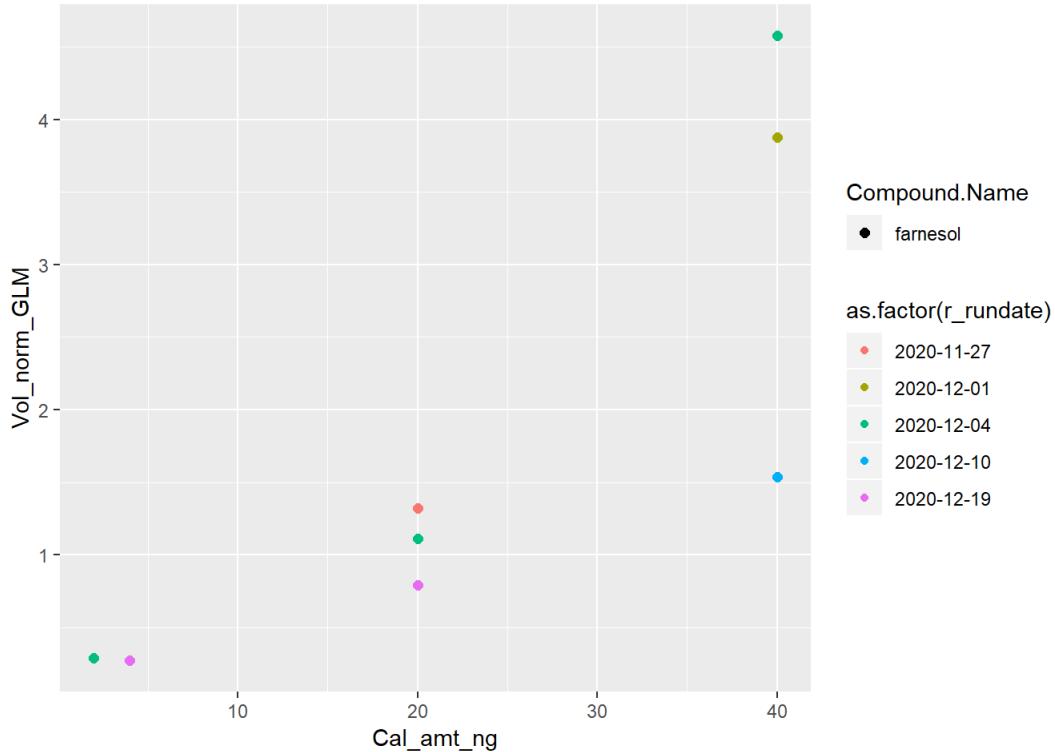
```
## Warning: Using size for a discrete variable is not advised.
```



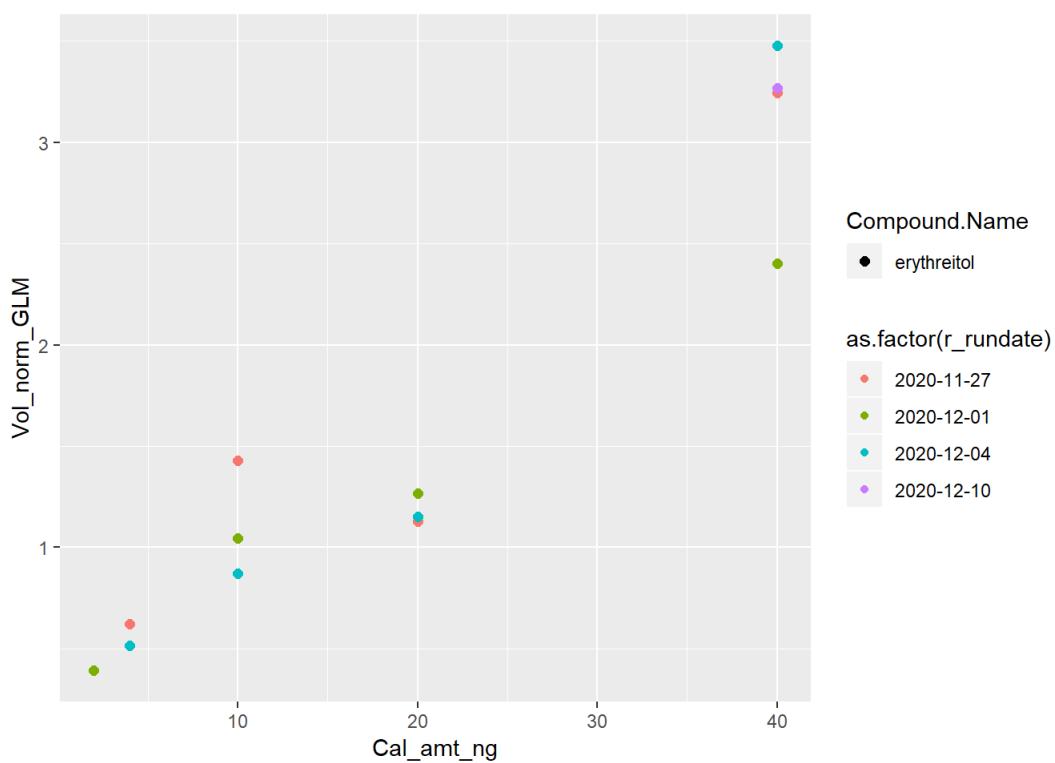
```
## Warning: Using size for a discrete variable is not advised.
```



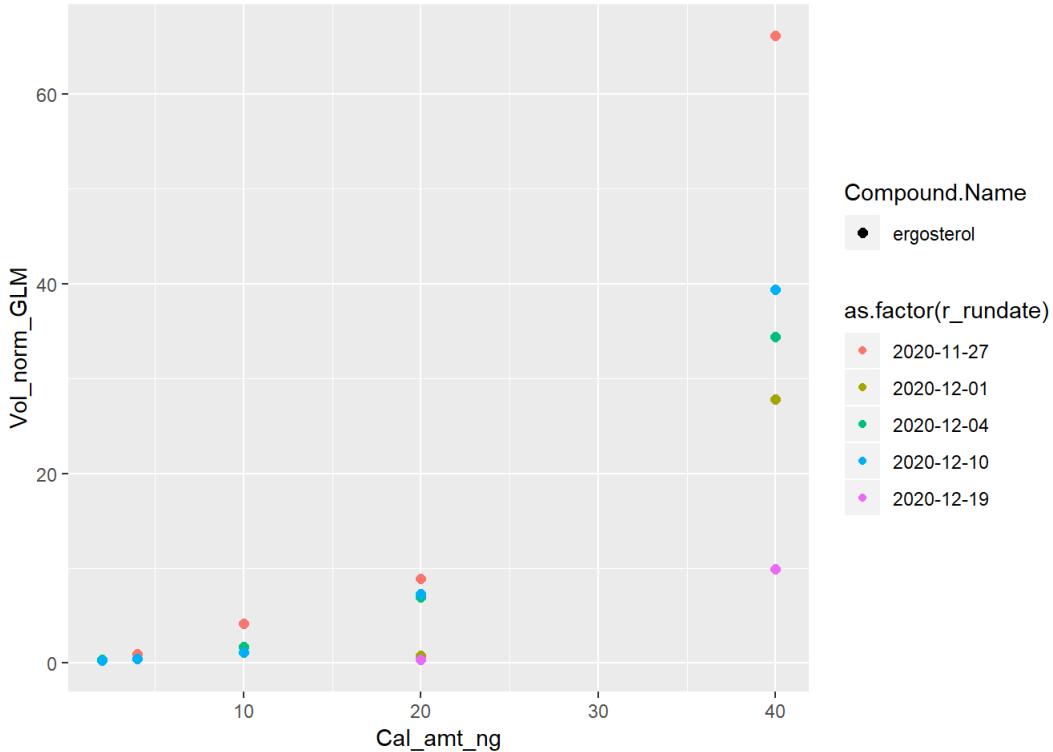
```
## Warning: Using size for a discrete variable is not advised.
```



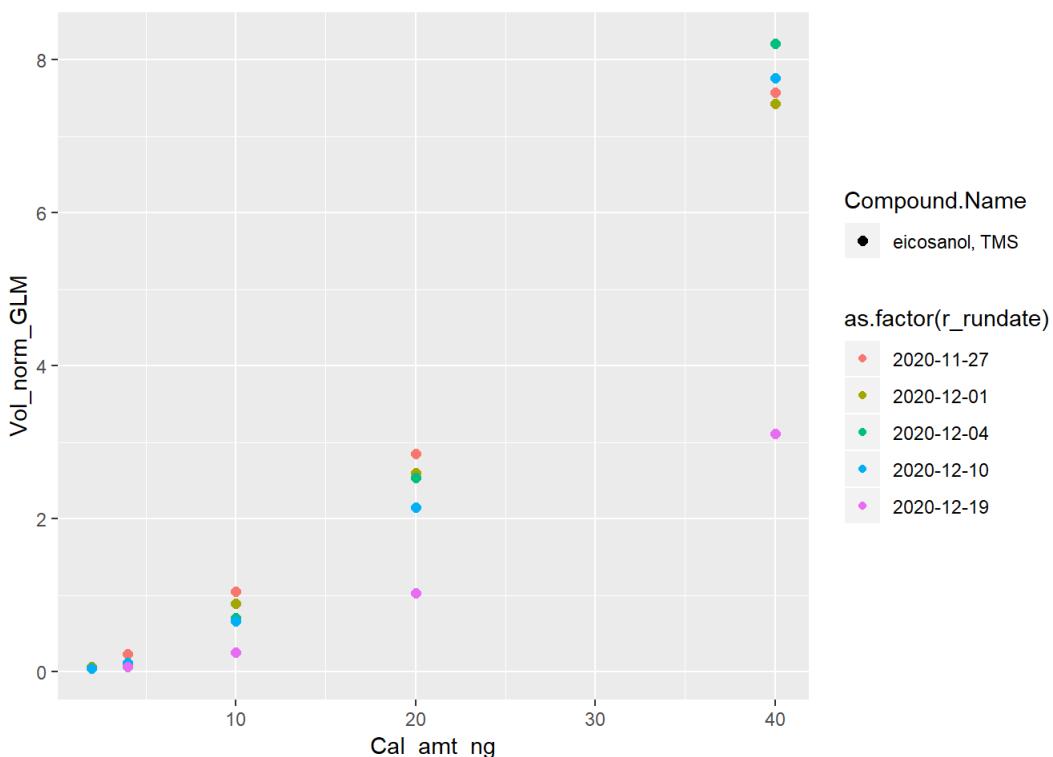
```
## Warning: Using size for a discrete variable is not advised.
```



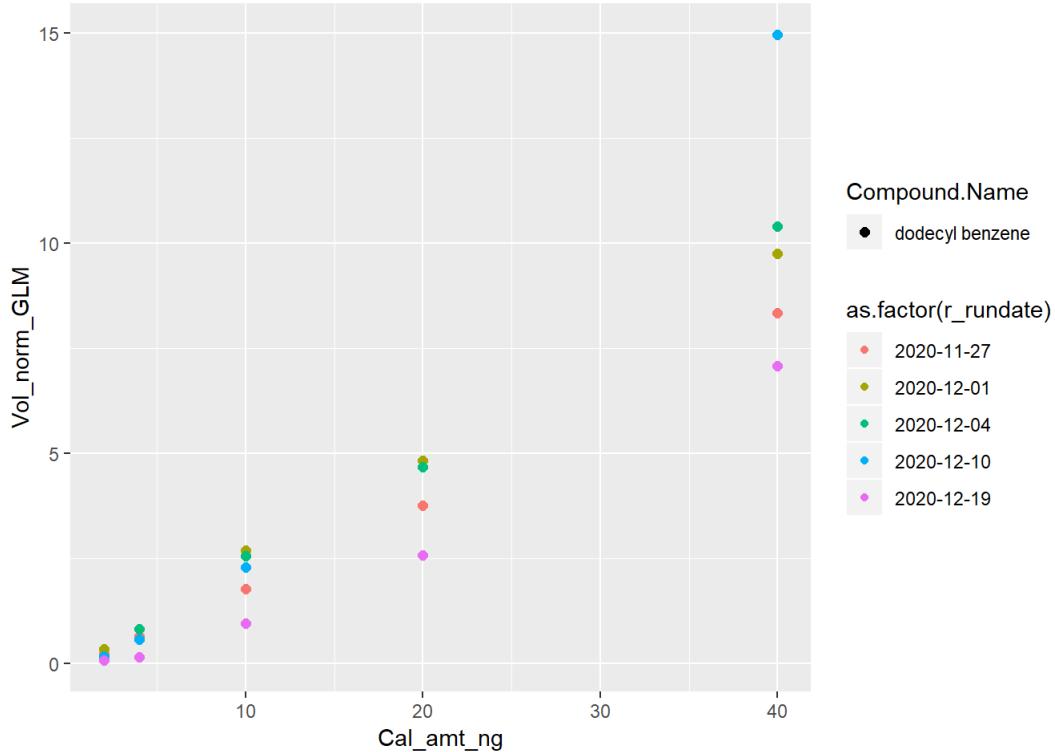
```
## Warning: Using size for a discrete variable is not advised.
```



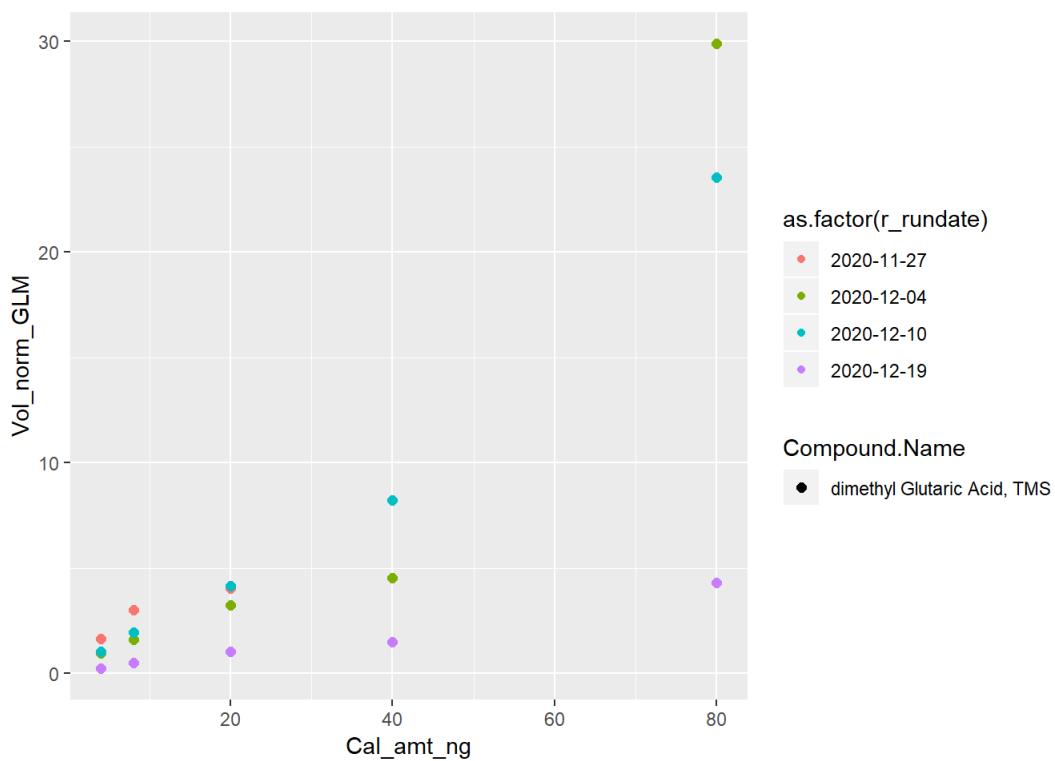
```
## Warning: Using size for a discrete variable is not advised.
```



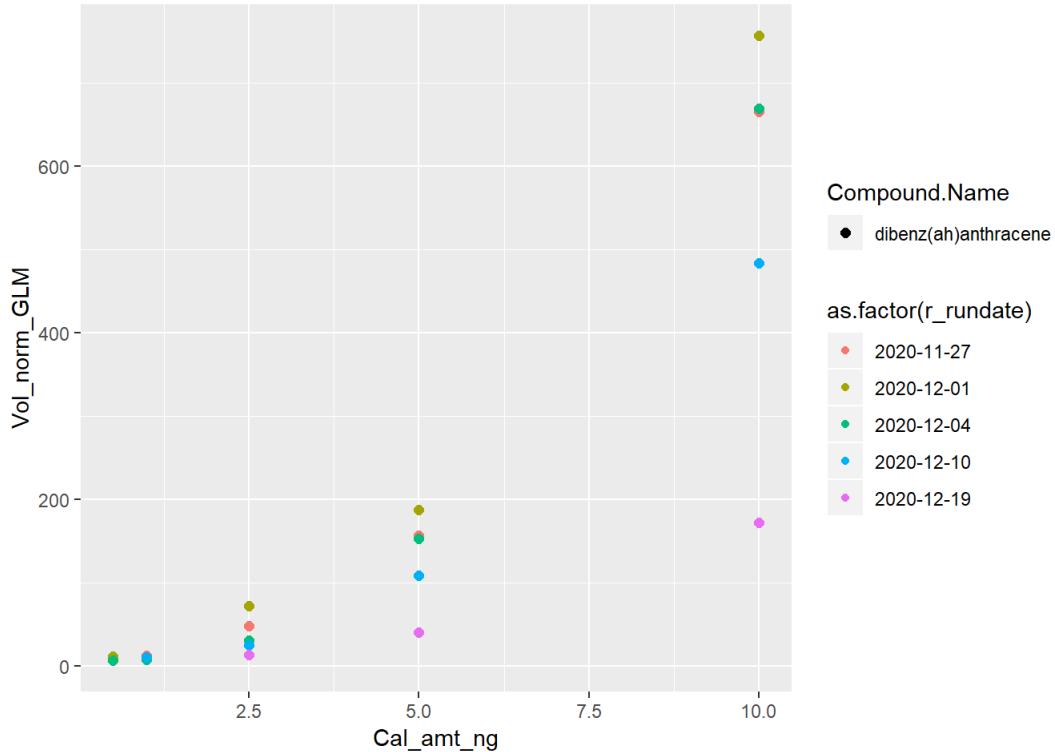
```
## Warning: Using size for a discrete variable is not advised.
```



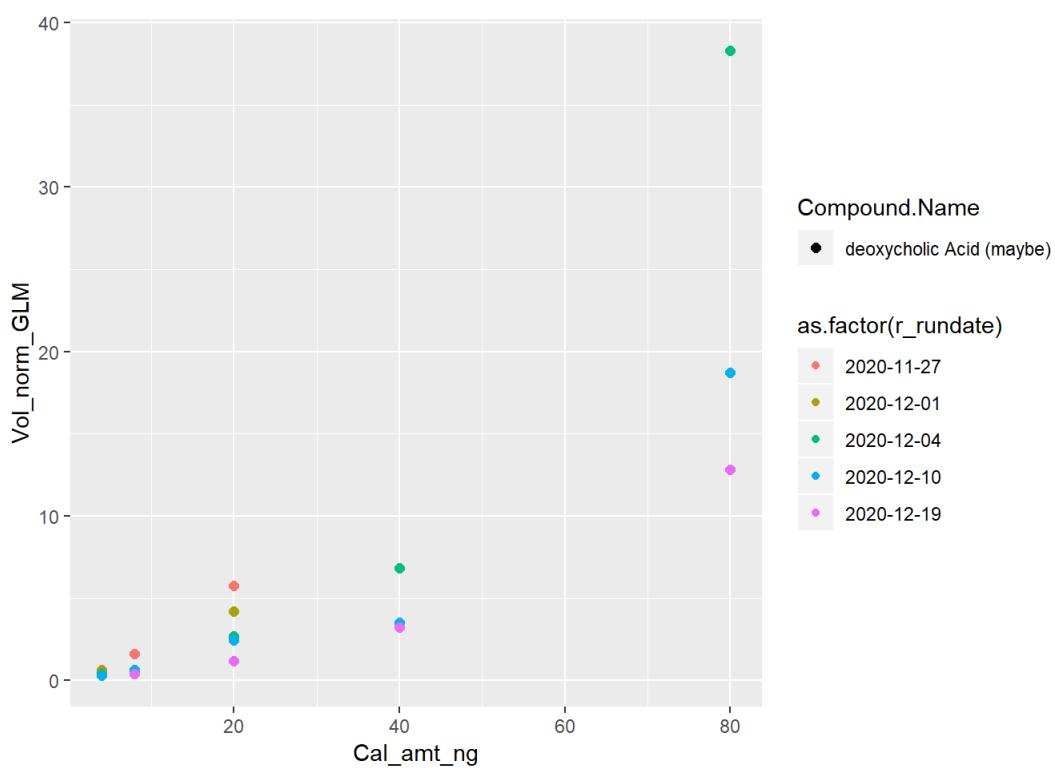
```
## Warning: Using size for a discrete variable is not advised.
```



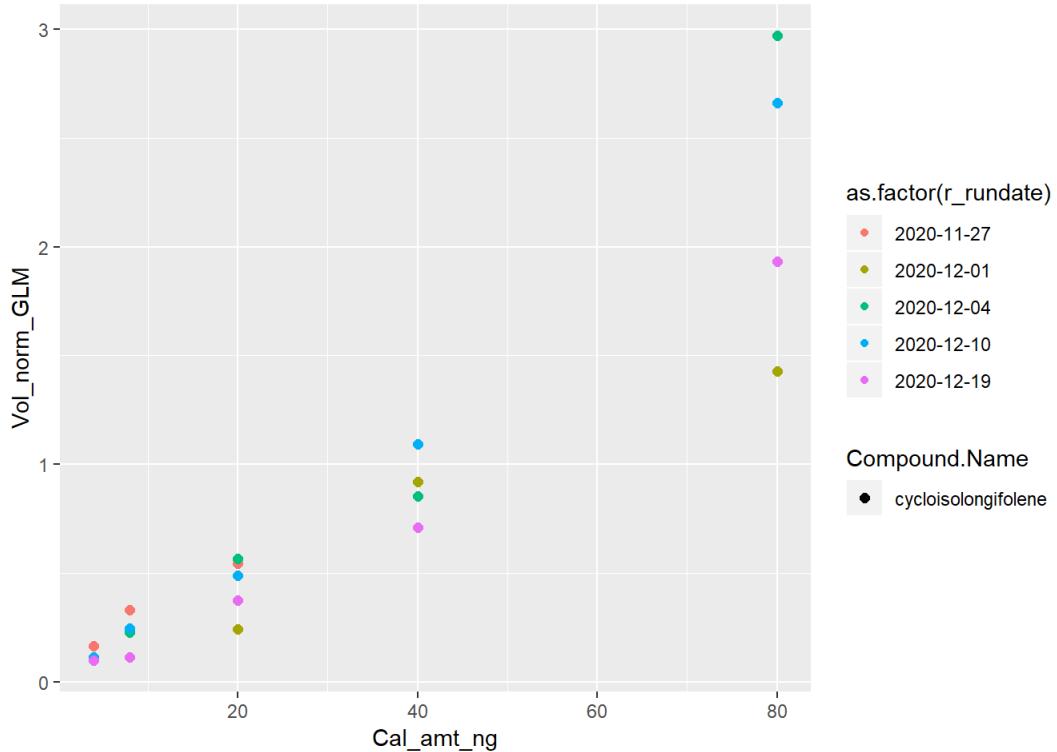
```
## Warning: Using size for a discrete variable is not advised.
```



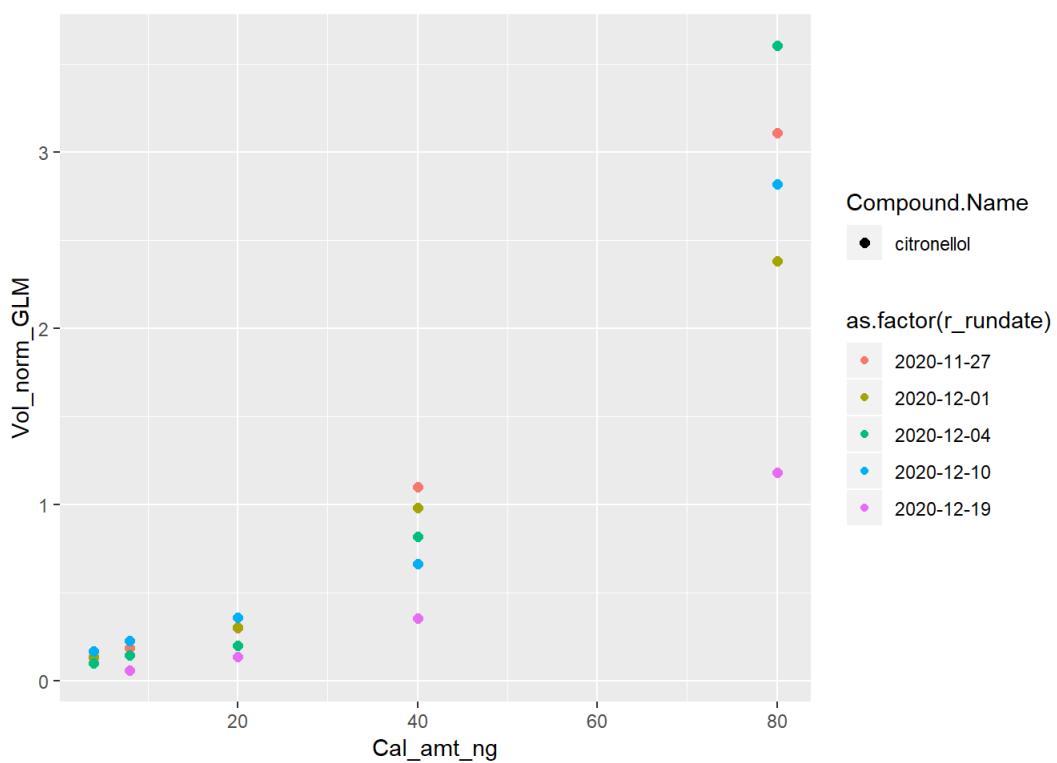
```
## Warning: Using size for a discrete variable is not advised.
```



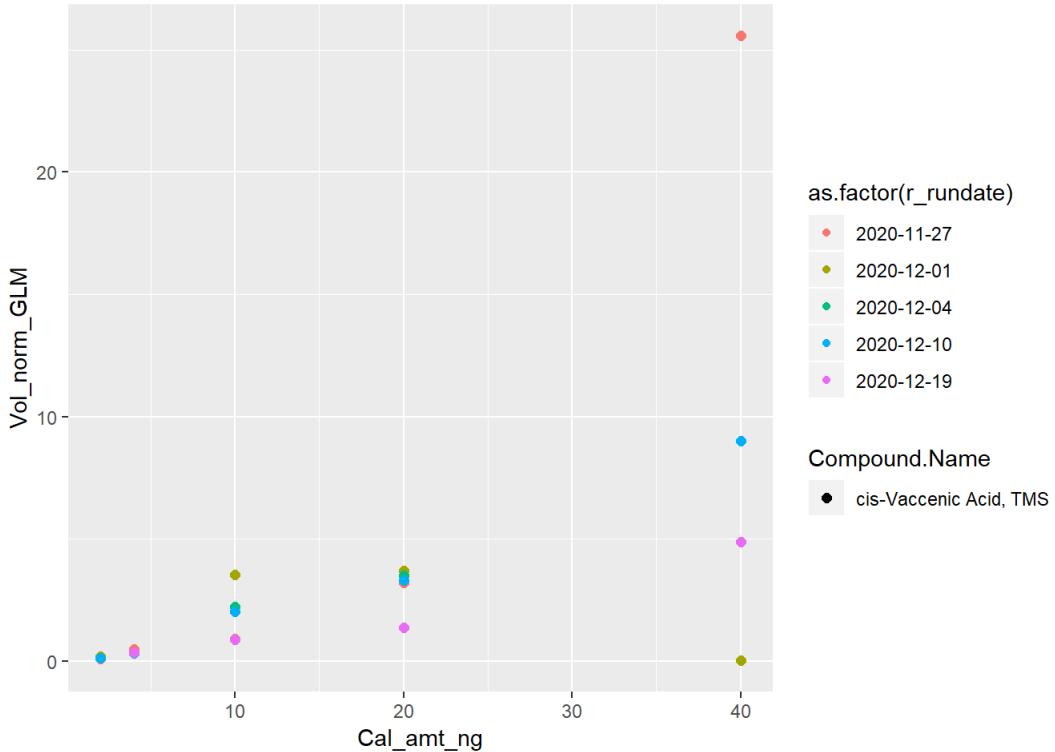
```
## Warning: Using size for a discrete variable is not advised.
```



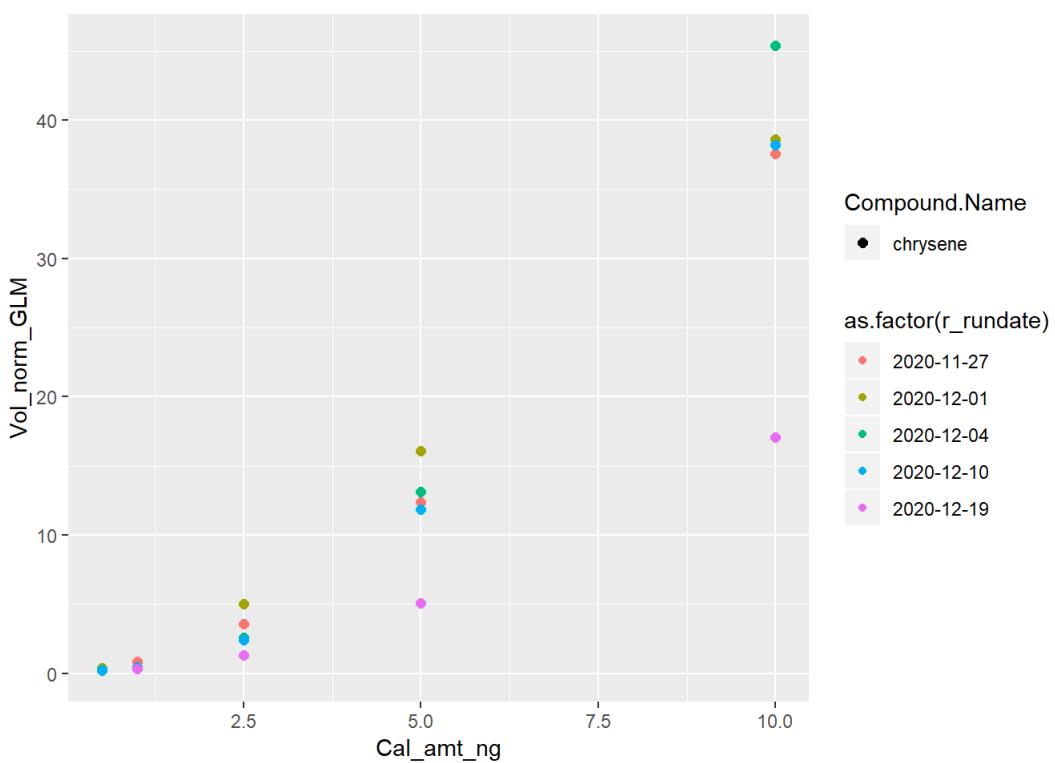
```
## Warning: Using size for a discrete variable is not advised.
```



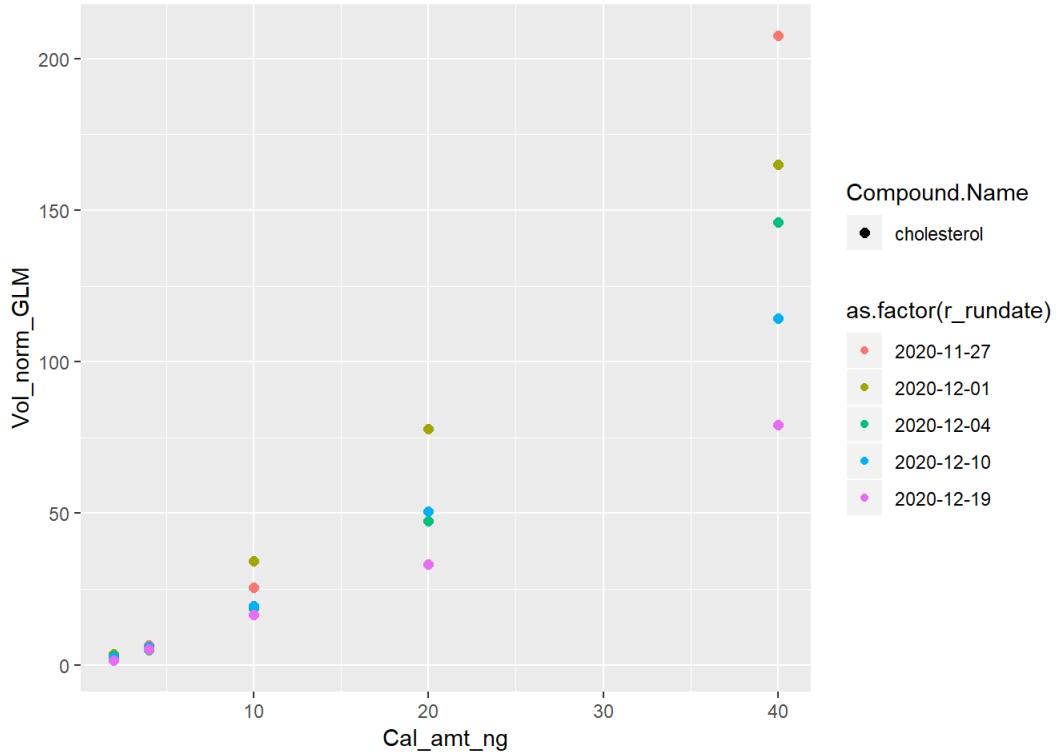
```
## Warning: Using size for a discrete variable is not advised.
```



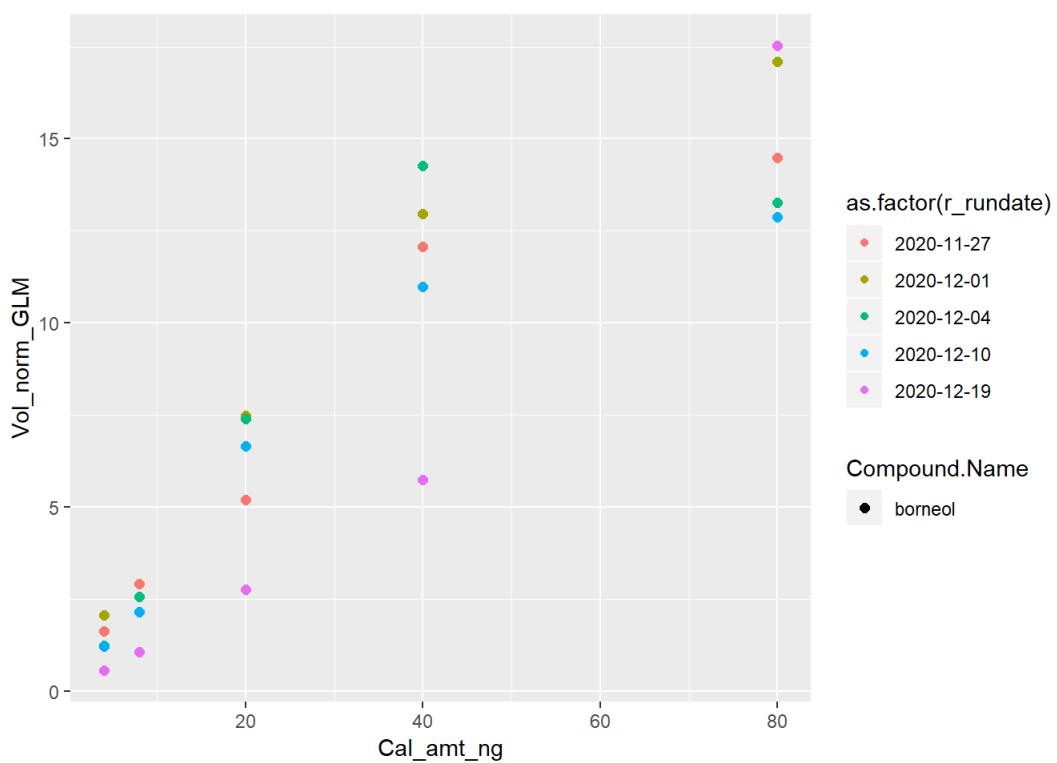
```
## Warning: Using size for a discrete variable is not advised.
```



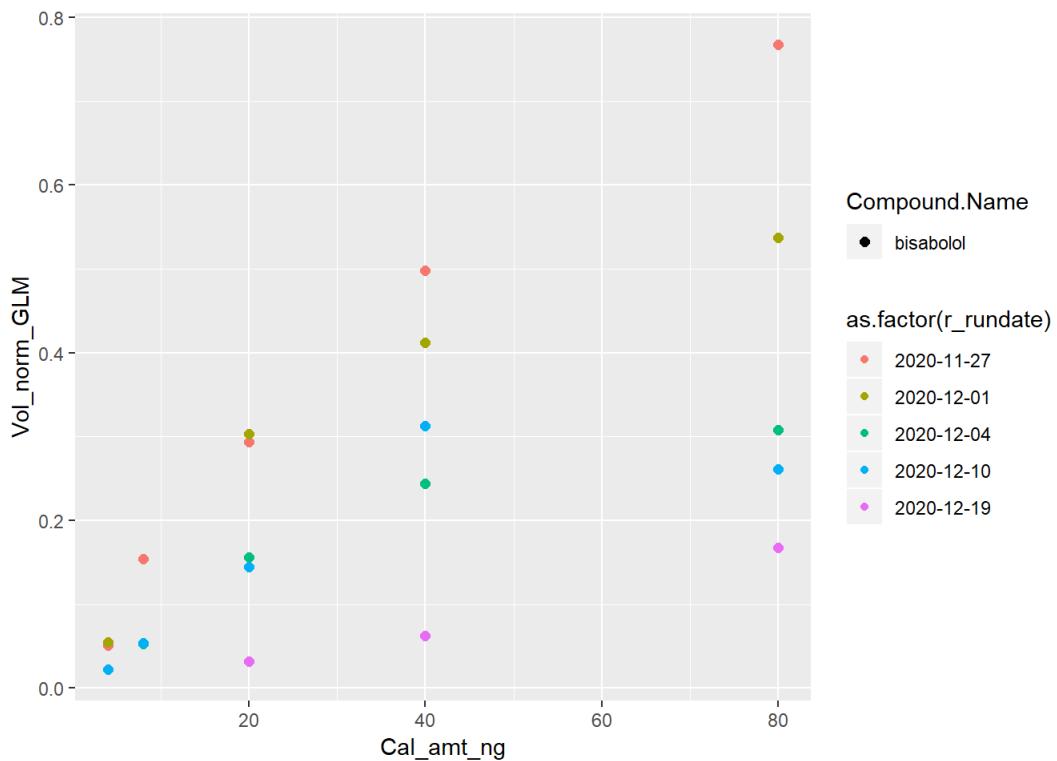
```
## Warning: Using size for a discrete variable is not advised.
```



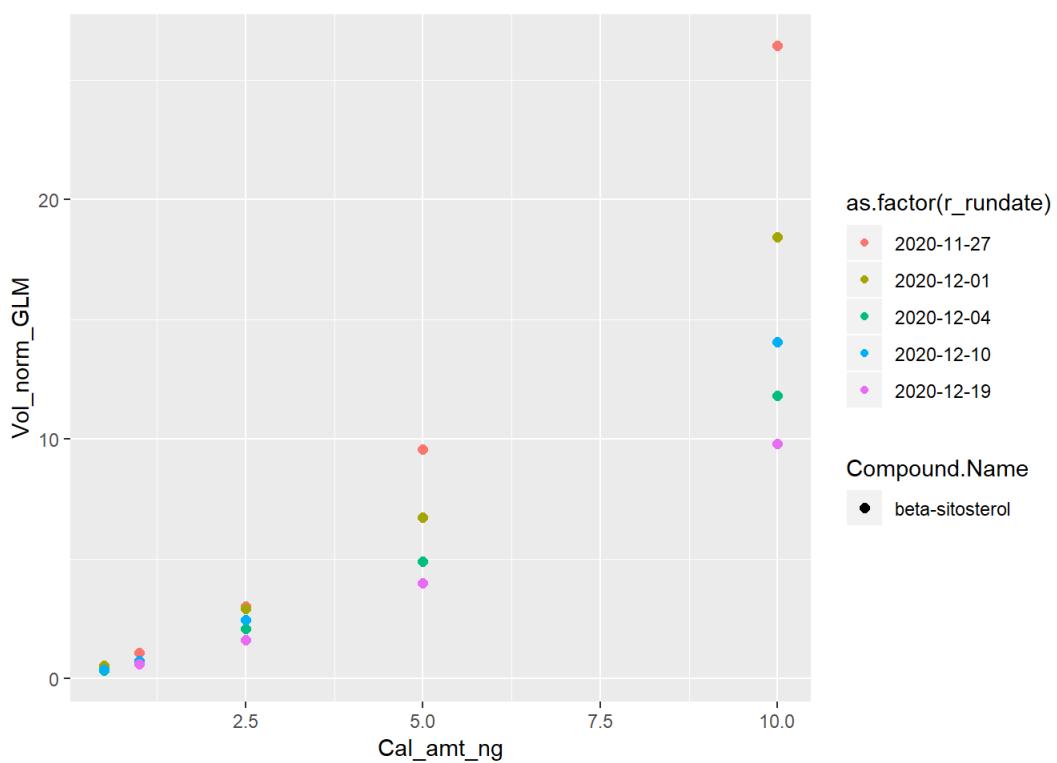
```
## Warning: Using size for a discrete variable is not advised.
```



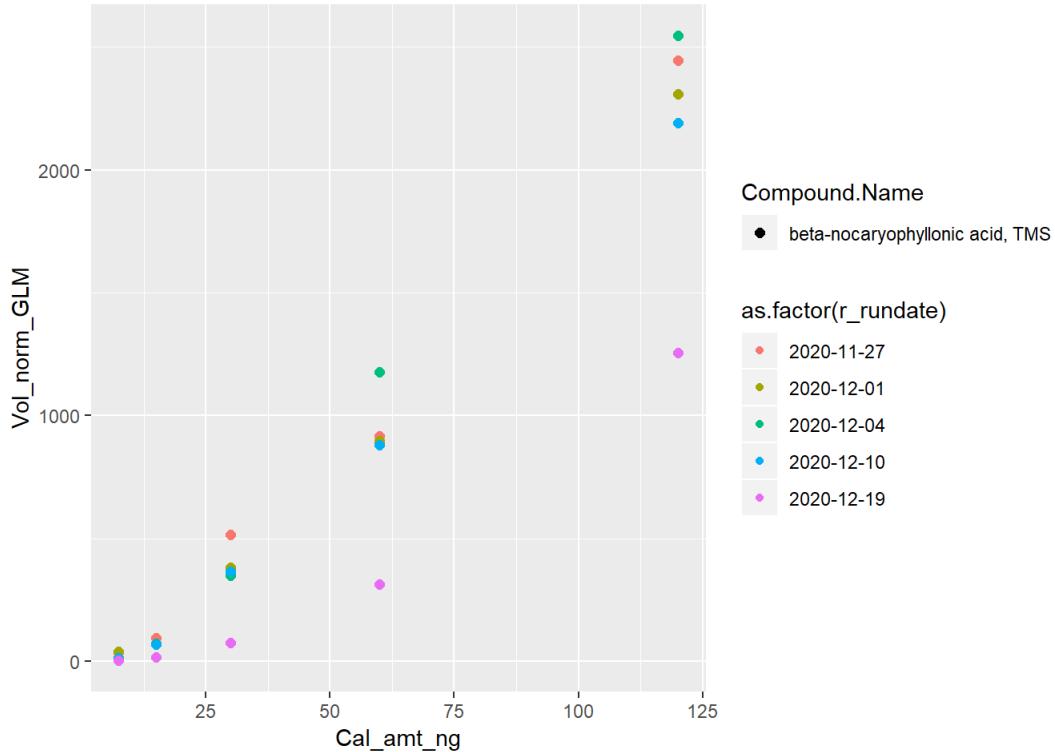
```
## Warning: Using size for a discrete variable is not advised.
```



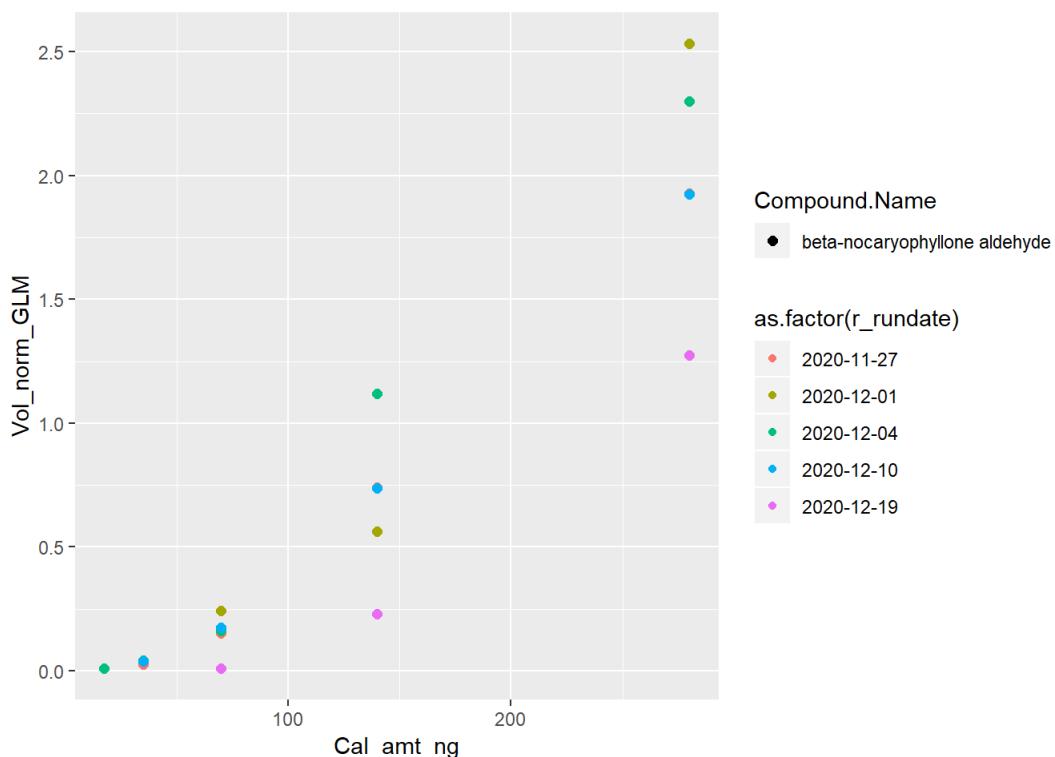
```
## Warning: Using size for a discrete variable is not advised.
```



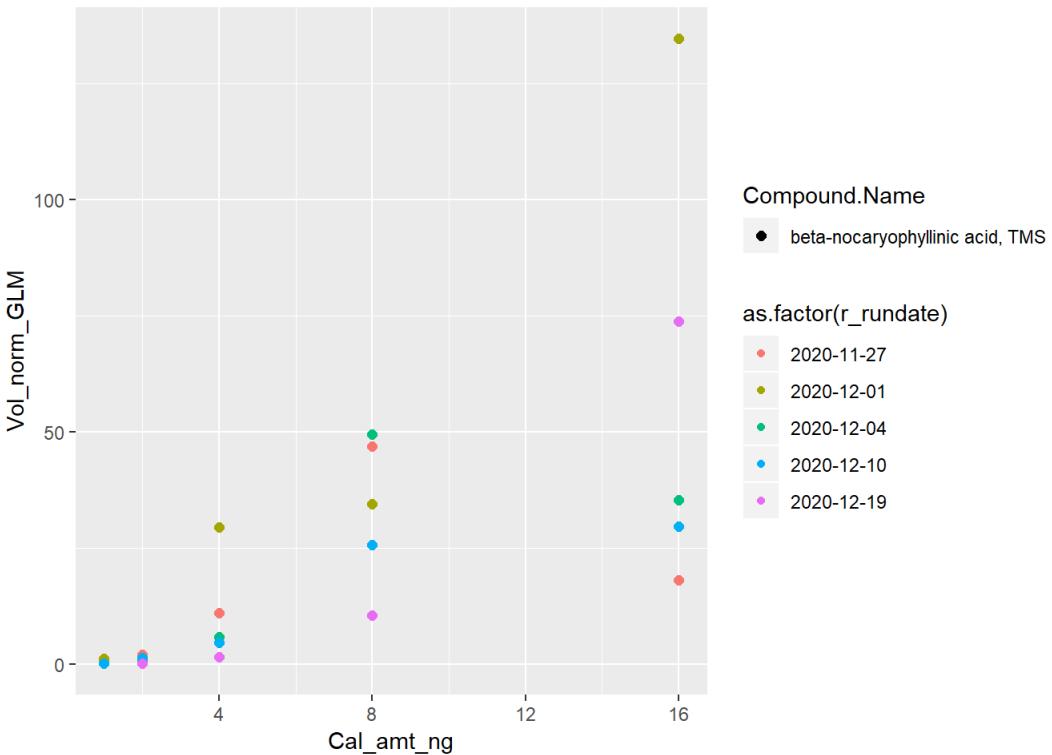
```
## Warning: Using size for a discrete variable is not advised.
```



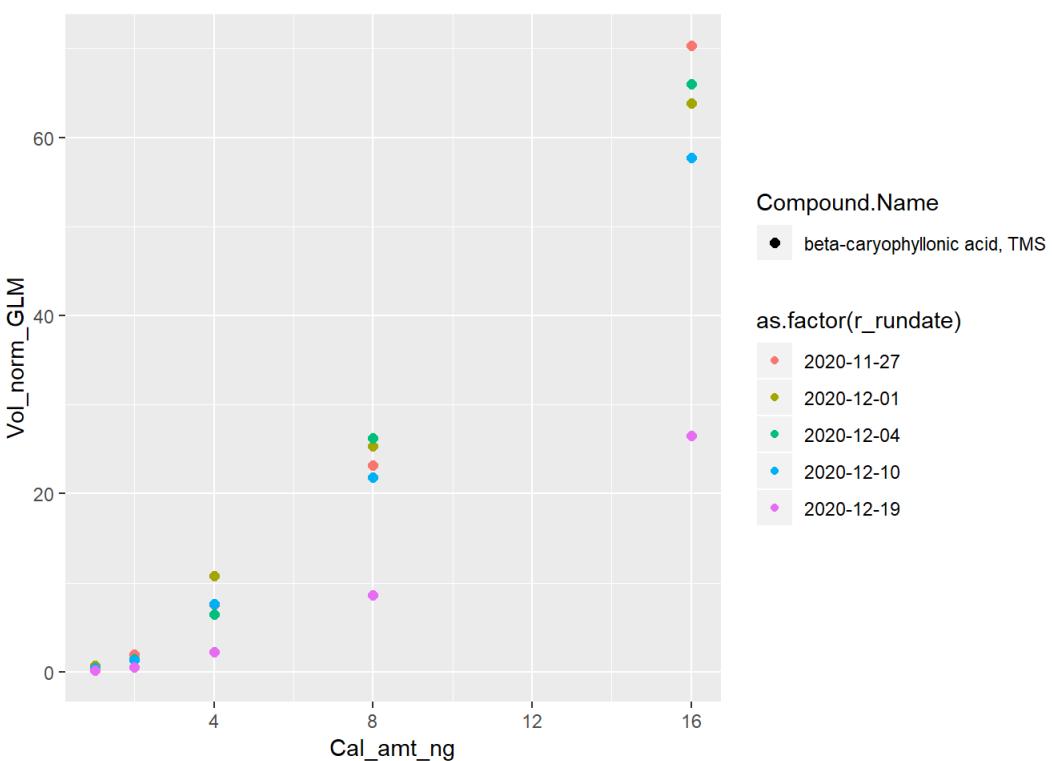
```
## Warning: Using size for a discrete variable is not advised.
```



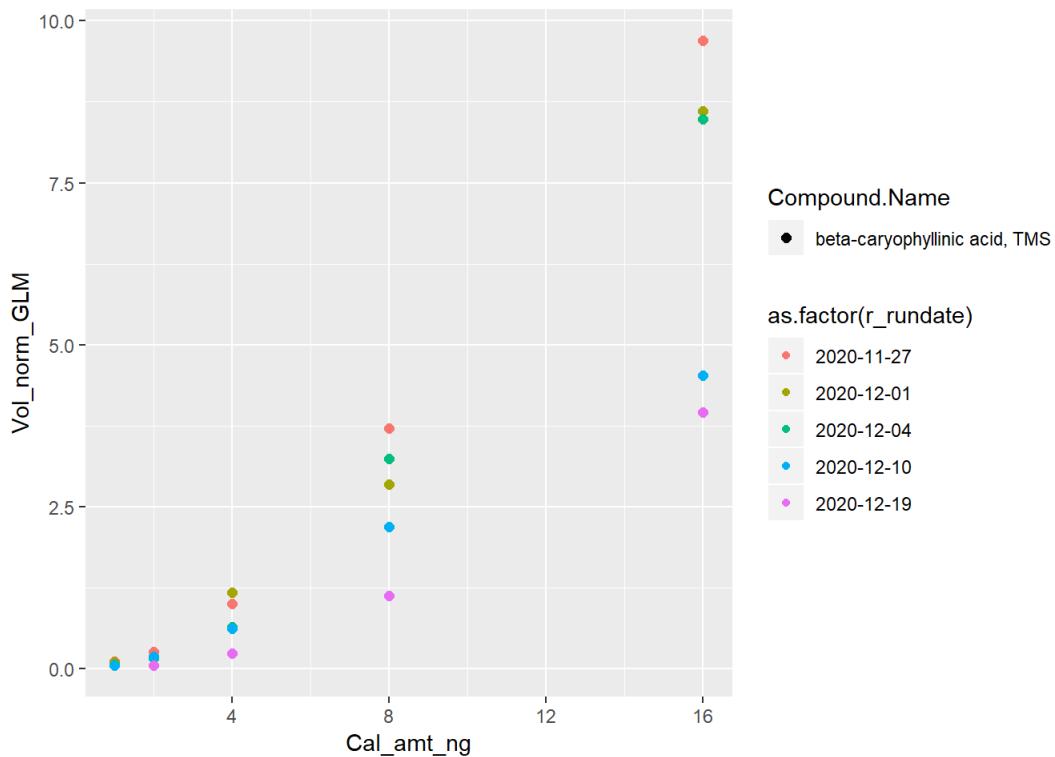
```
## Warning: Using size for a discrete variable is not advised.
```



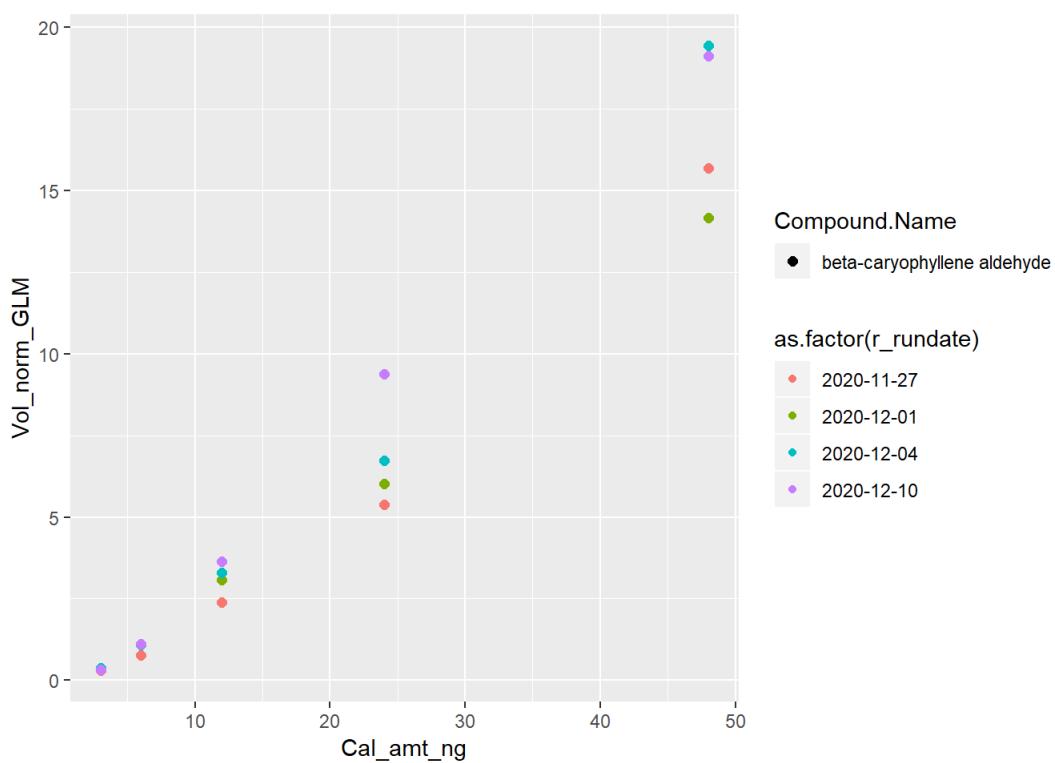
```
## Warning: Using size for a discrete variable is not advised.
```



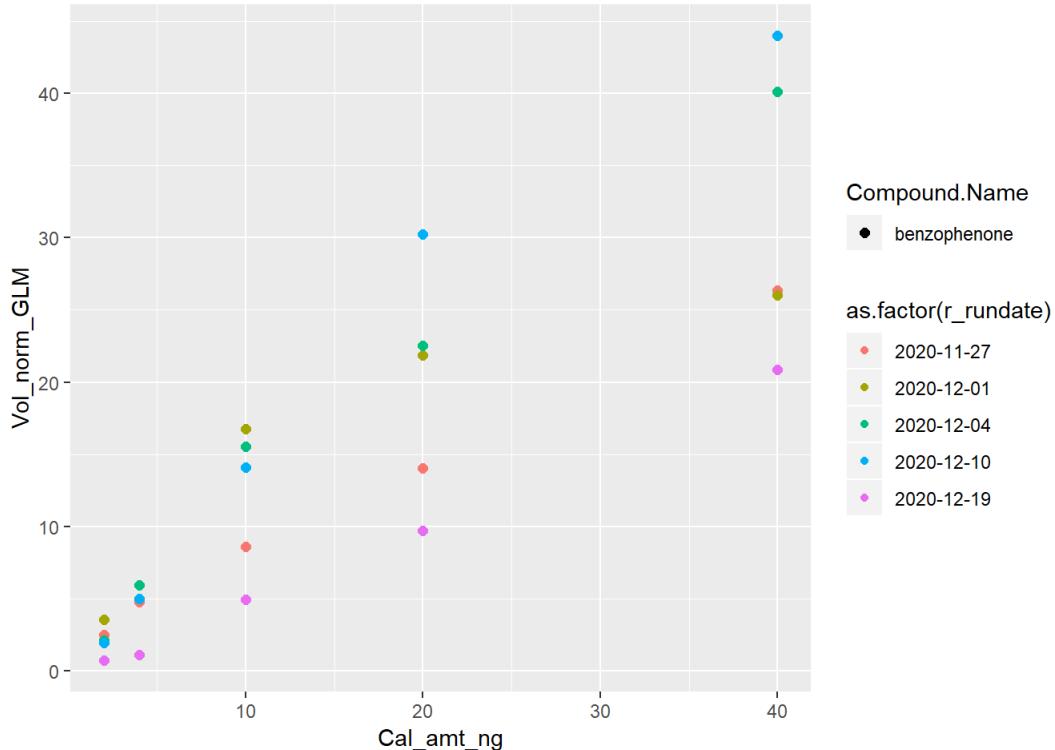
```
## Warning: Using size for a discrete variable is not advised.
```



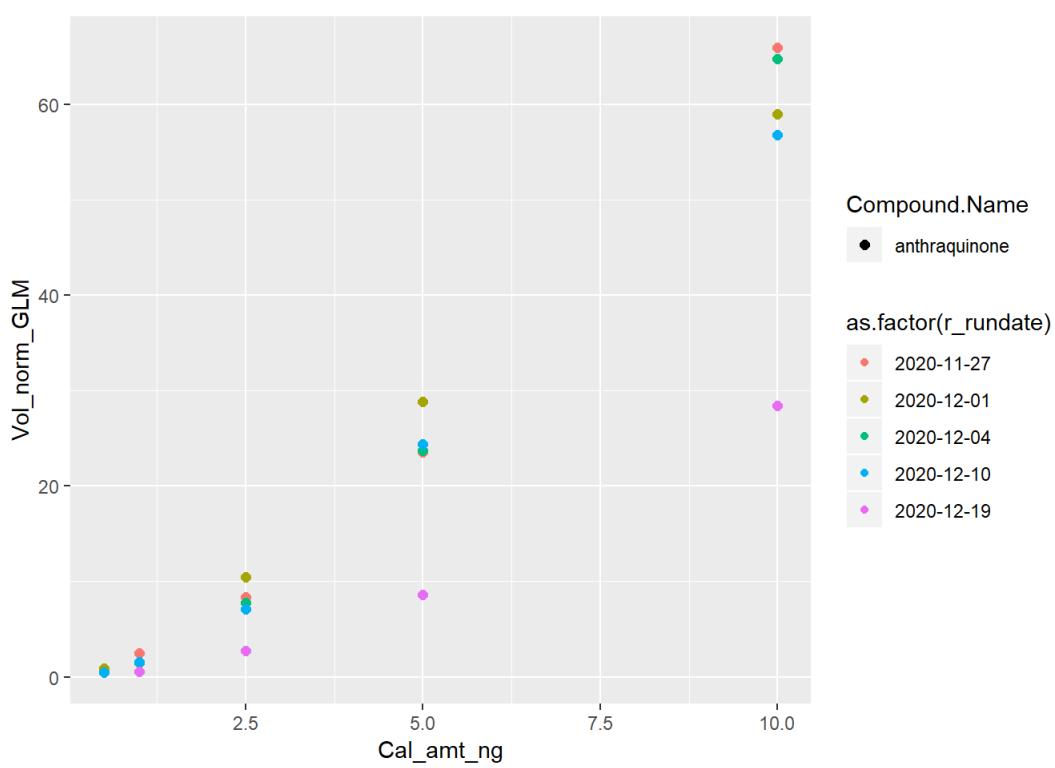
```
## Warning: Using size for a discrete variable is not advised.
```



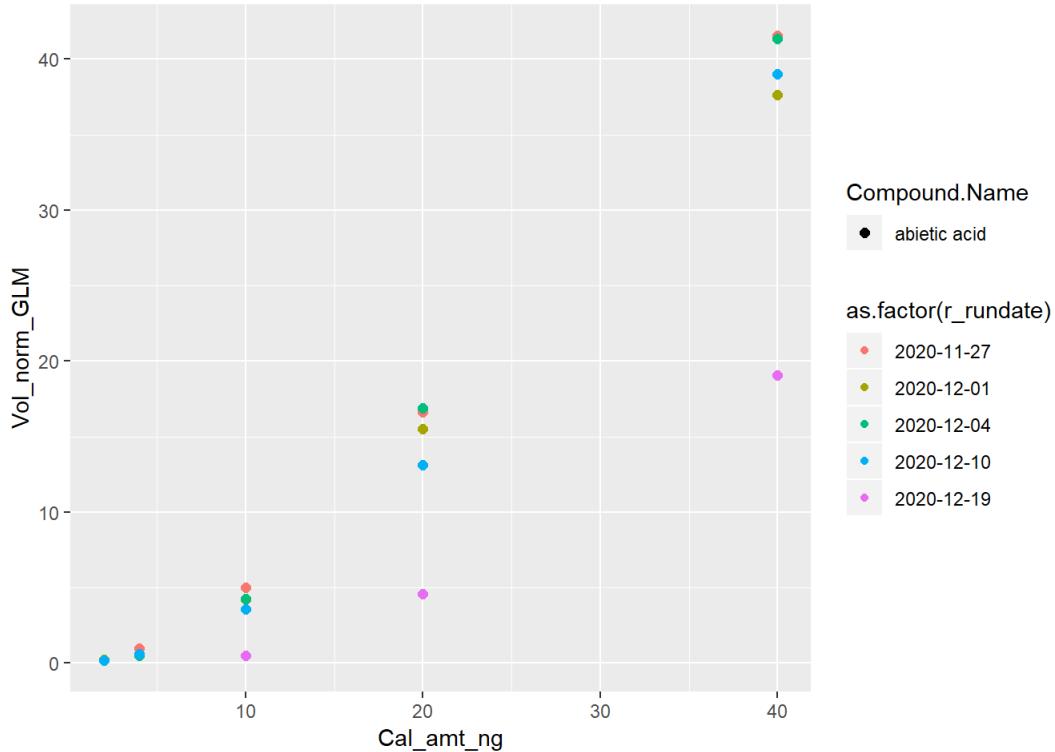
```
## Warning: Using size for a discrete variable is not advised.
```



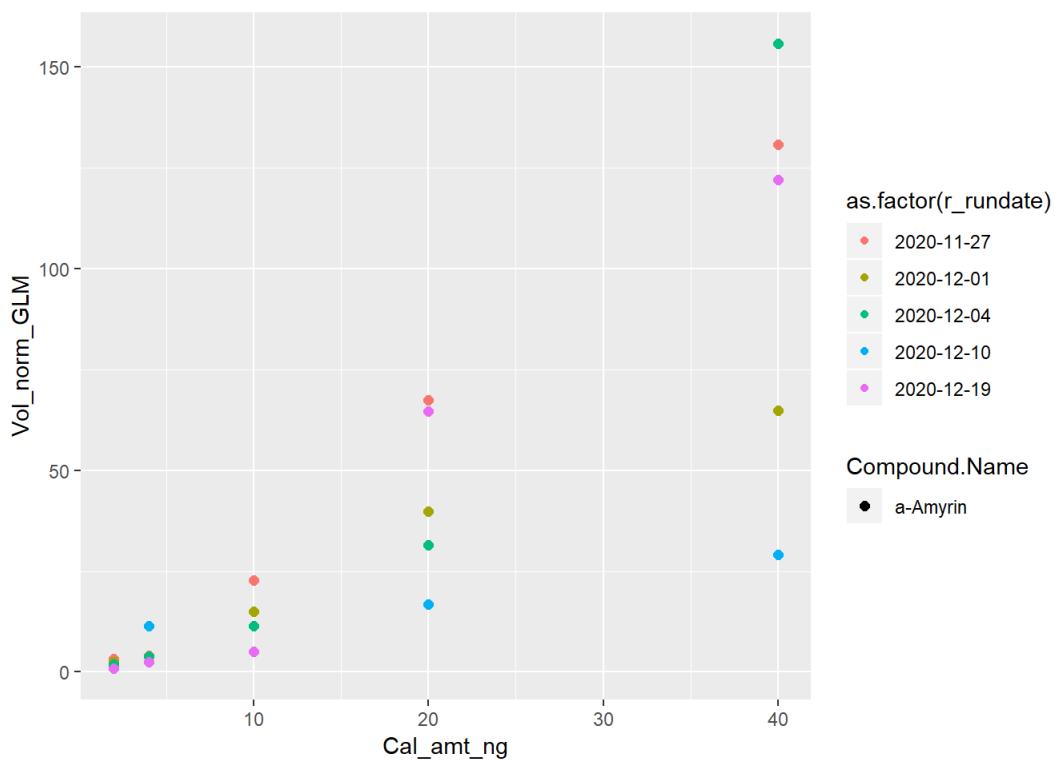
```
## Warning: Using size for a discrete variable is not advised.
```



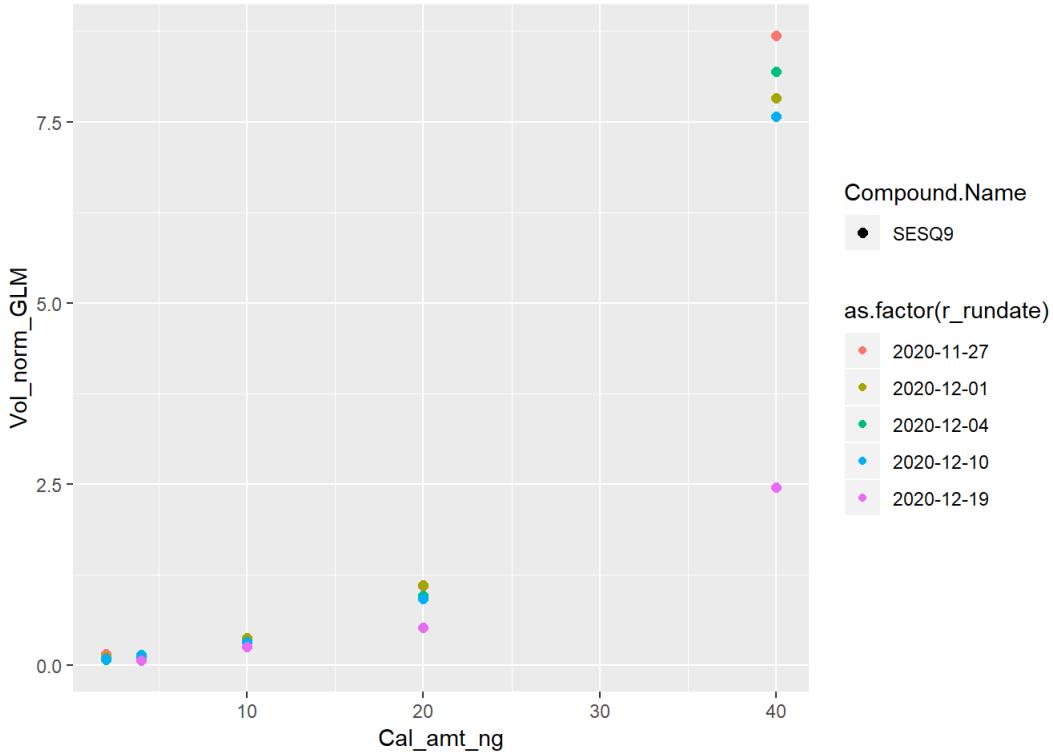
```
## Warning: Using size for a discrete variable is not advised.
```



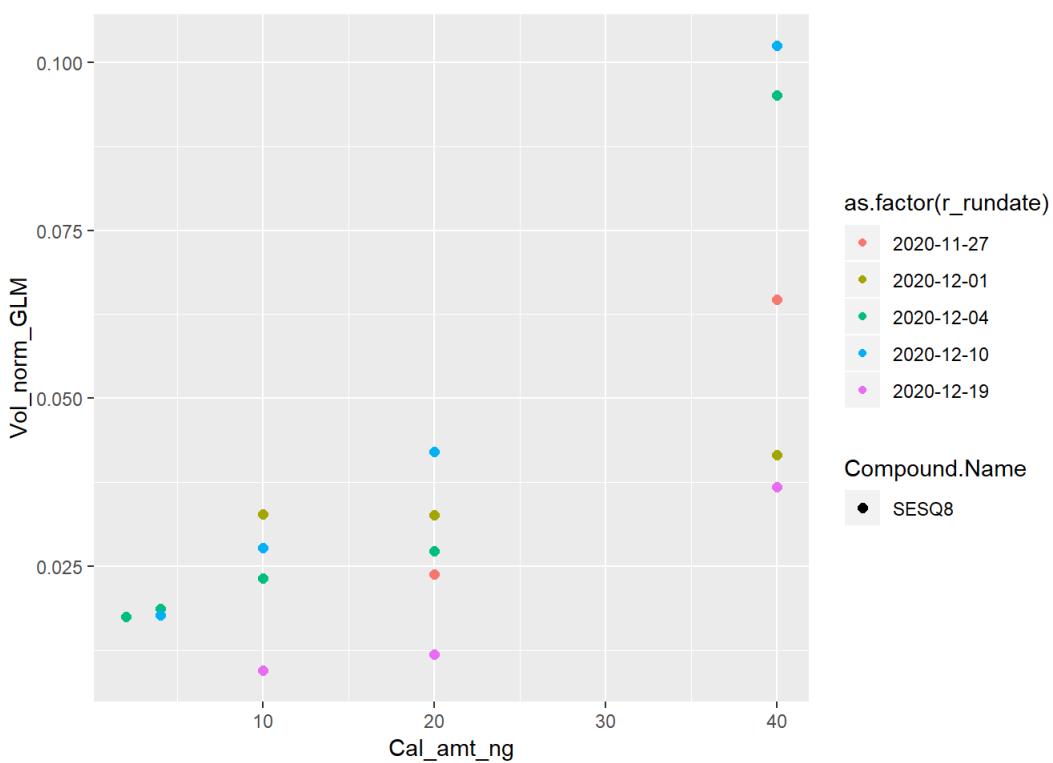
```
## Warning: Using size for a discrete variable is not advised.
```



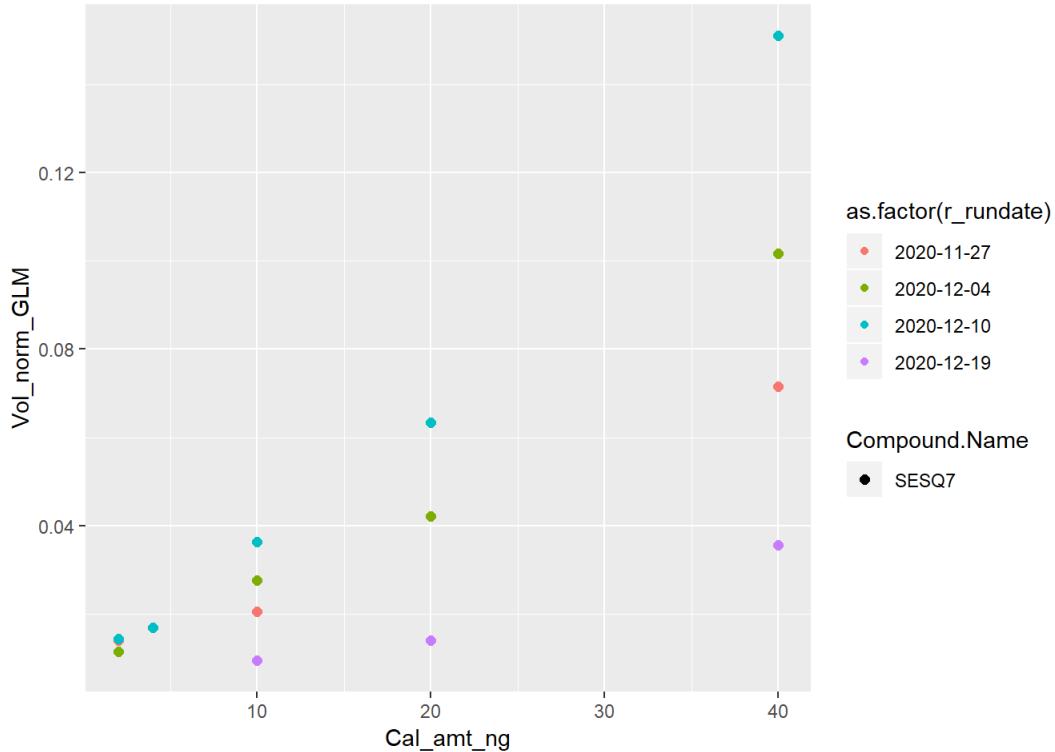
```
## Warning: Using size for a discrete variable is not advised.
```



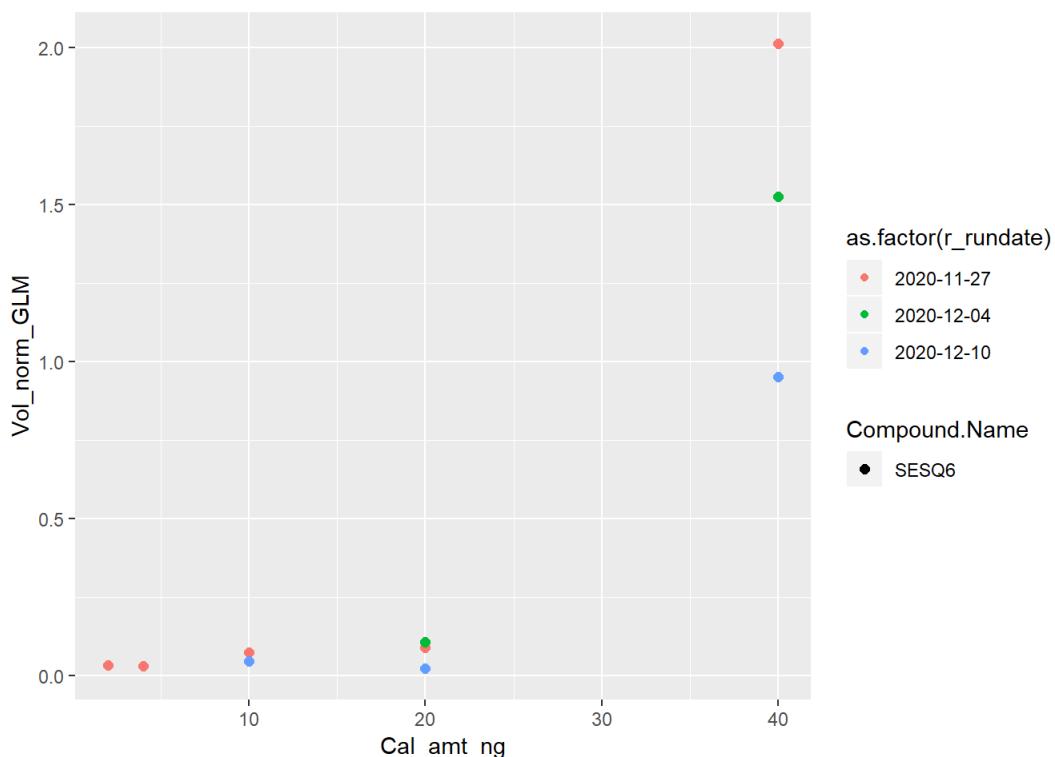
```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```

Vol_norm_GLM

Cal_amt_ng

```
## Warning: Using size for a discrete variable is not advised.
```

Vol_norm_GLM

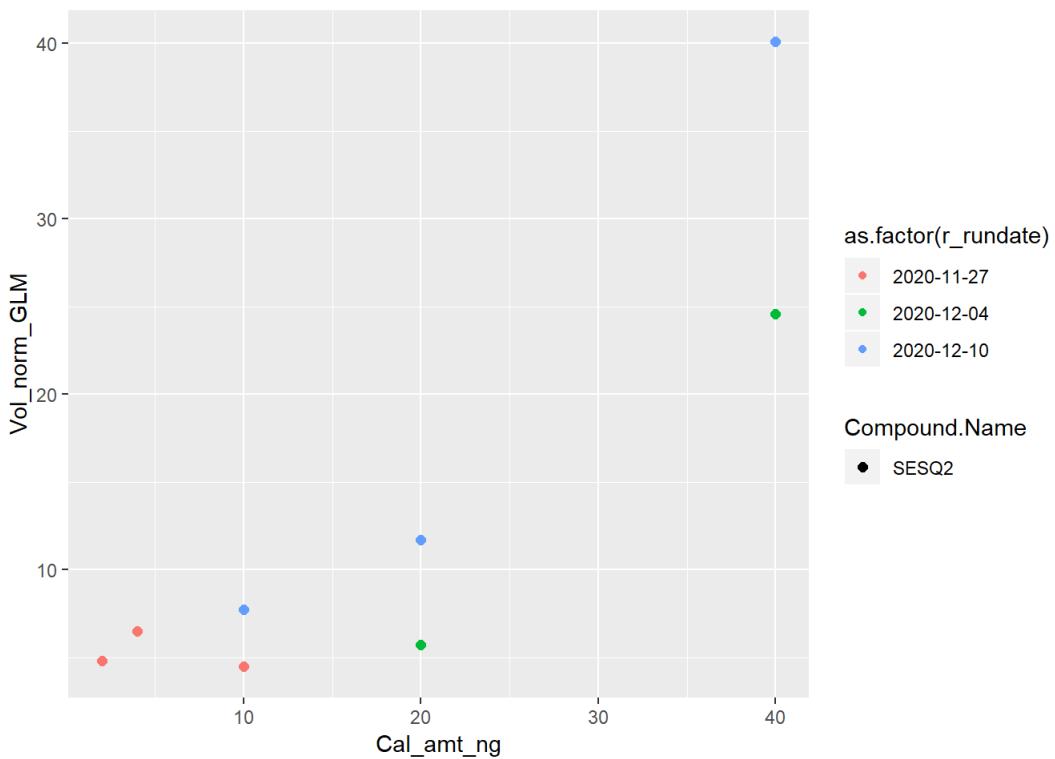
Cal_amt_ng

```
## Warning: Using size for a discrete variable is not advised.
```

Vol_norm_GLM

Cal_amt_ng

```
## Warning: Using size for a discrete variable is not advised.
```

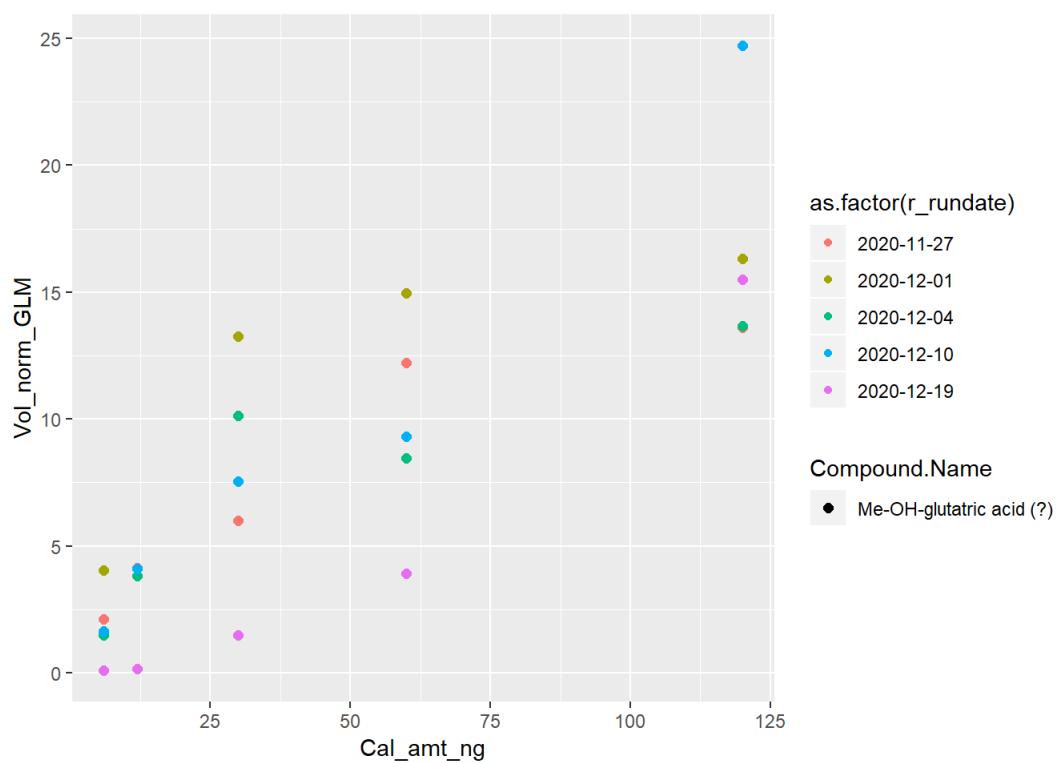


```
## Warning: Using size for a discrete variable is not advised.
```

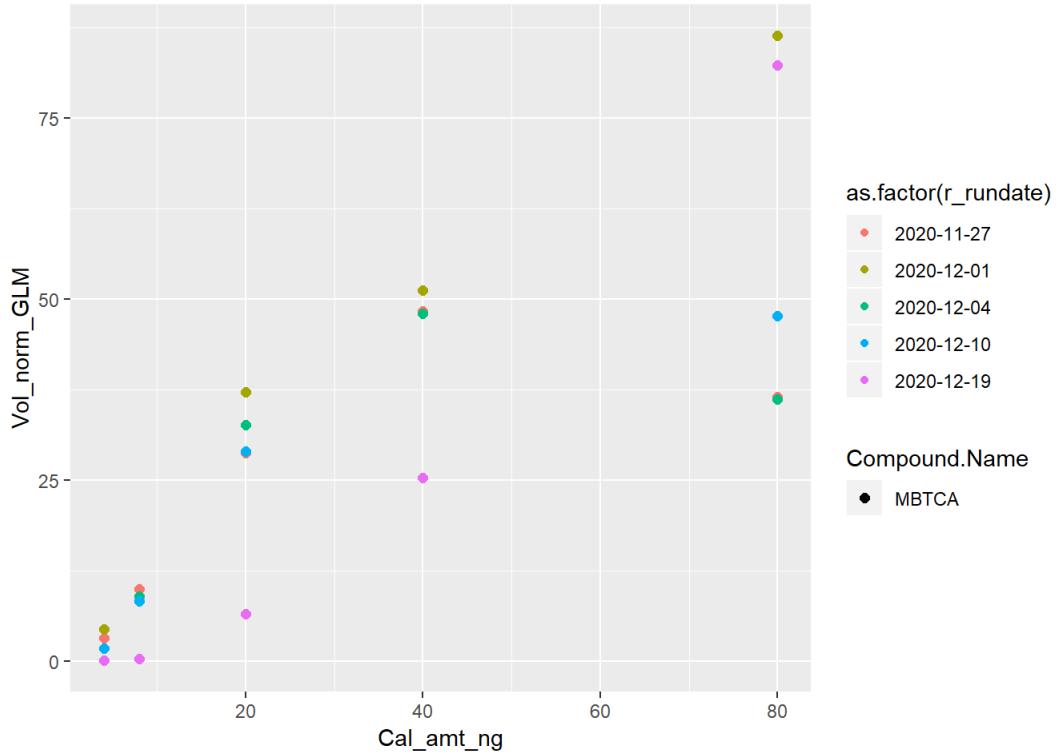
Vol_norm_GLM

Cal_amt_ng

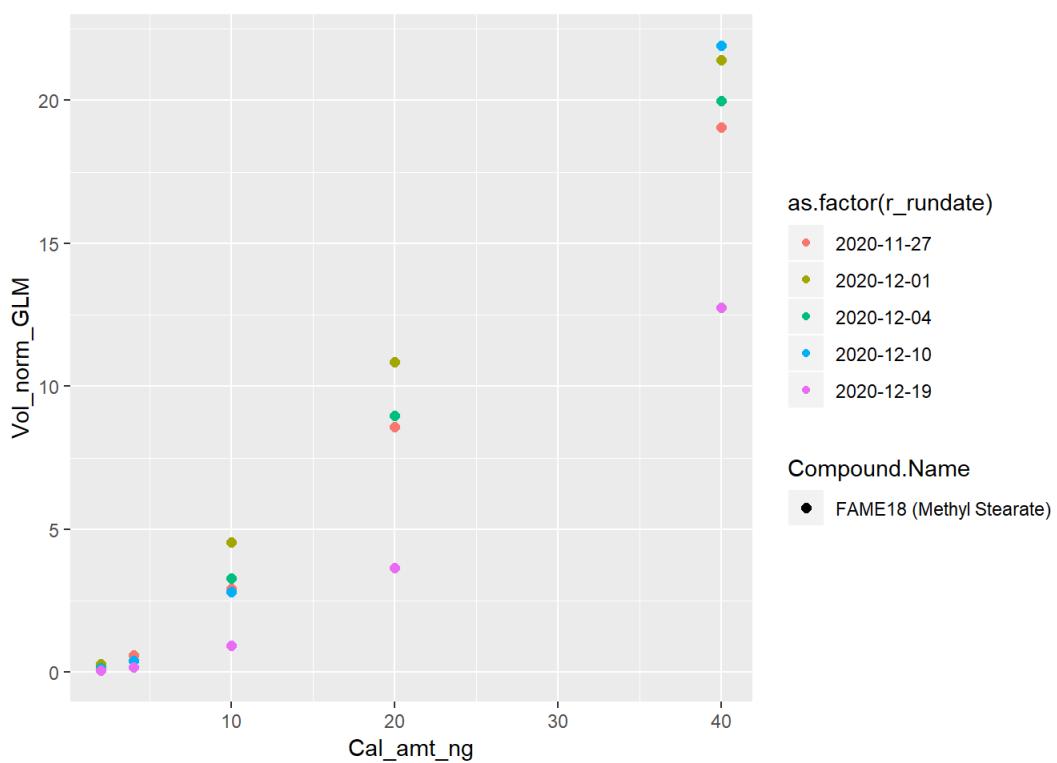
```
## Warning: Using size for a discrete variable is not advised.
```



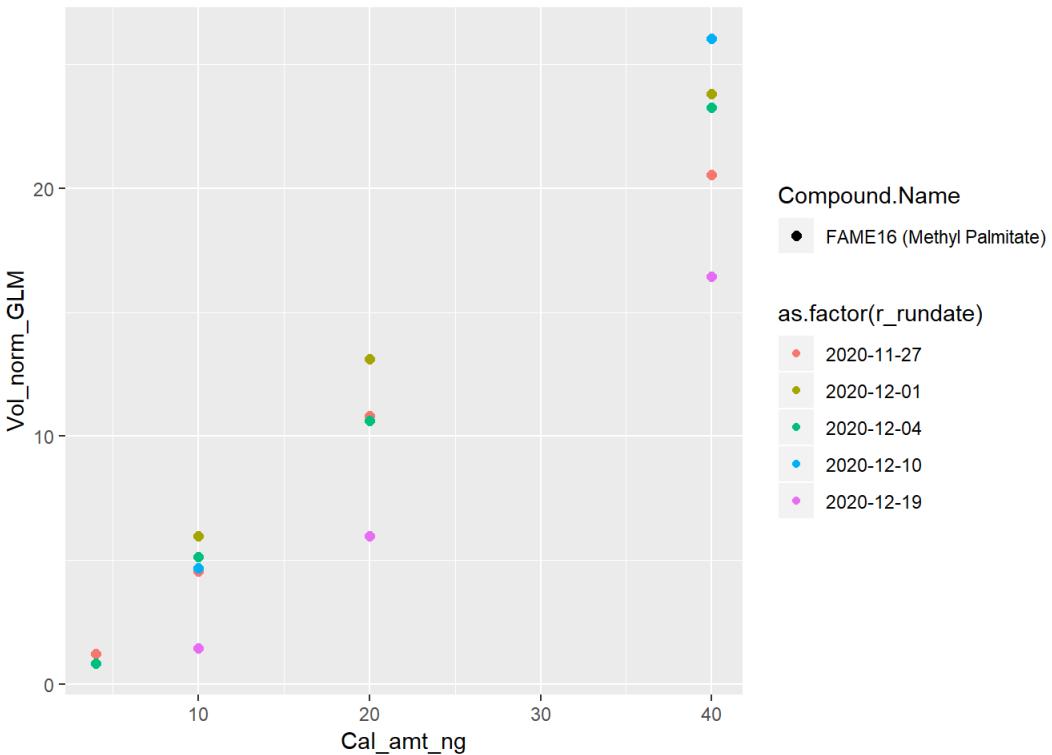
```
## Warning: Using size for a discrete variable is not advised.
```



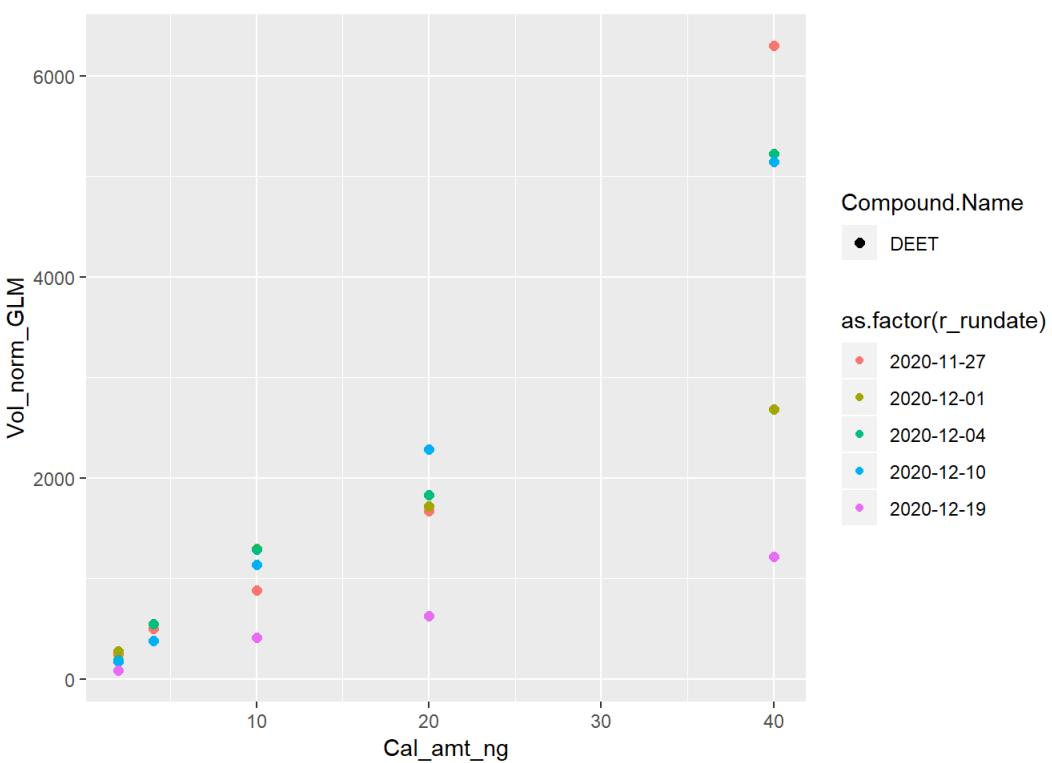
```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```

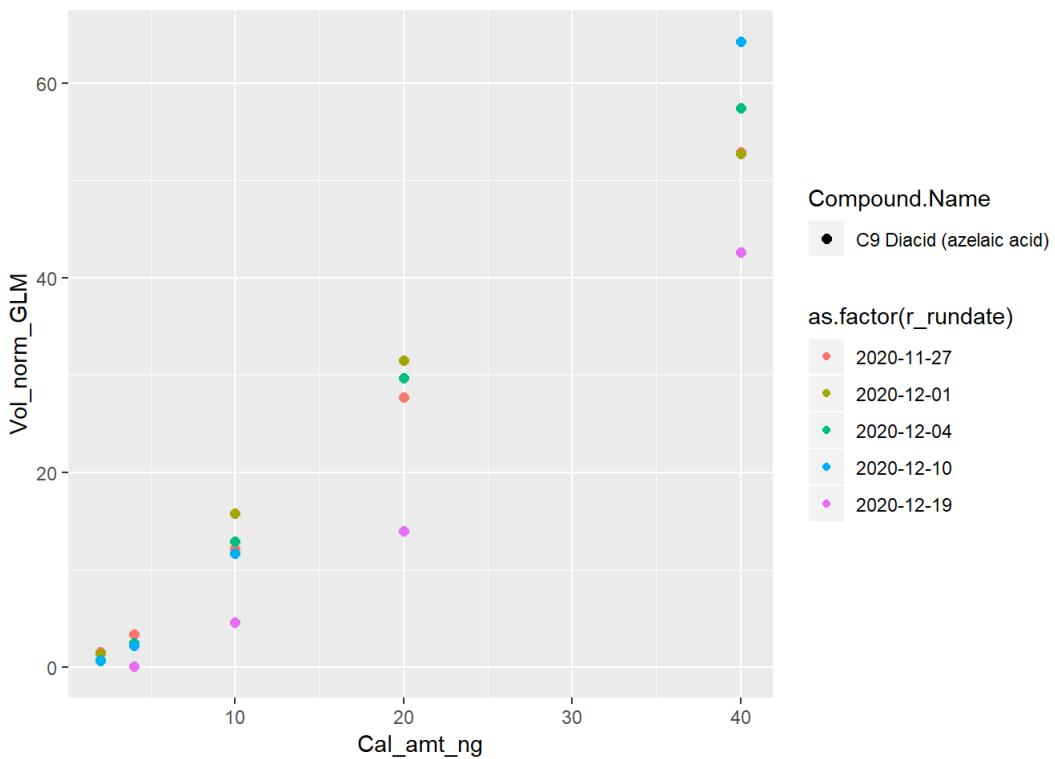


```
## Warning: Using size for a discrete variable is not advised.
```

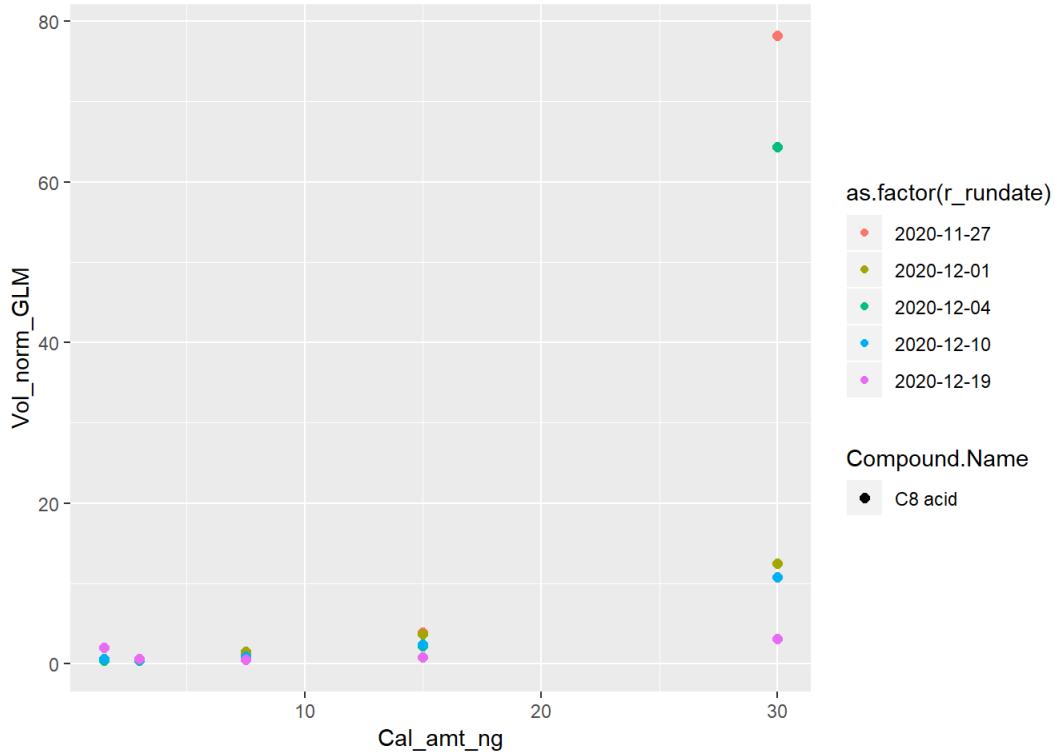
Vol_norm_GLM

Cal_amt_ng

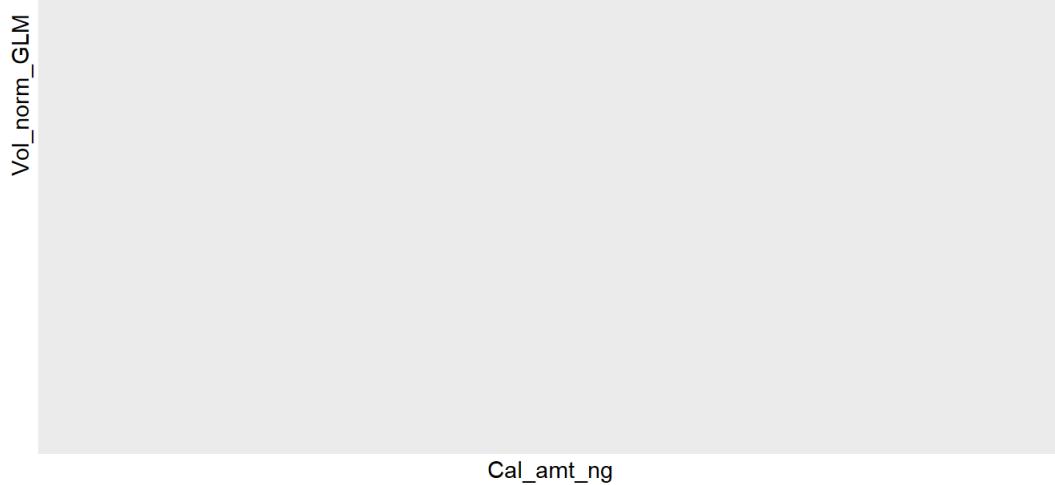
```
## Warning: Using size for a discrete variable is not advised.
```



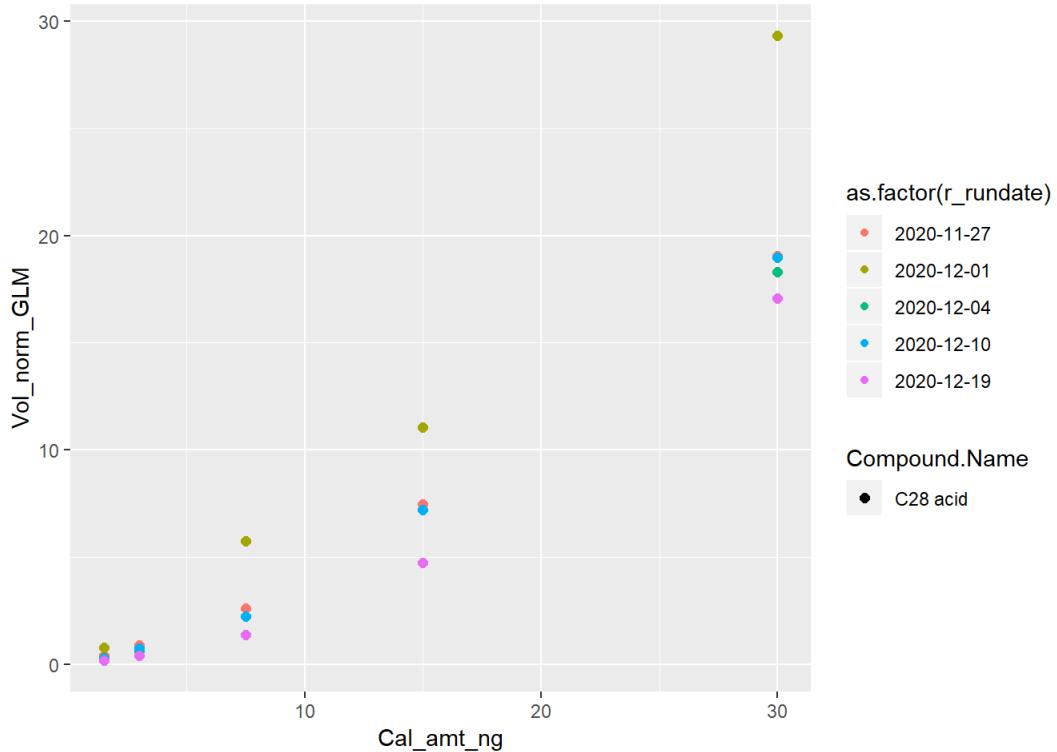
```
## Warning: Using size for a discrete variable is not advised.
```



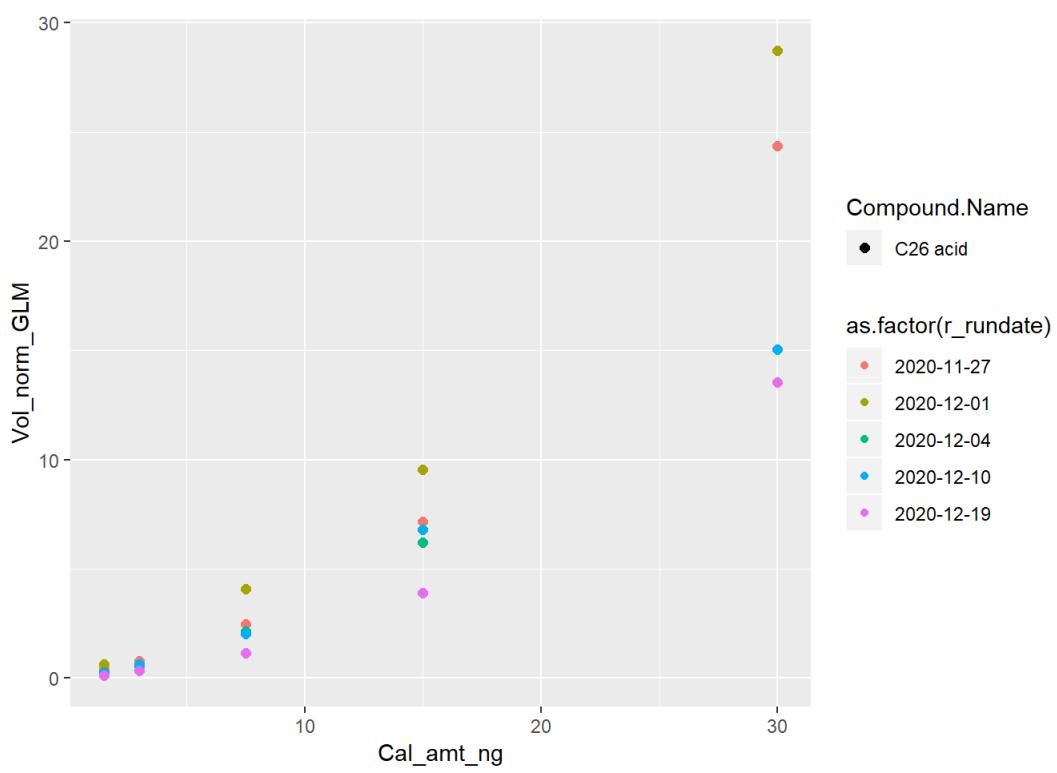
```
## Warning: Using size for a discrete variable is not advised.
```



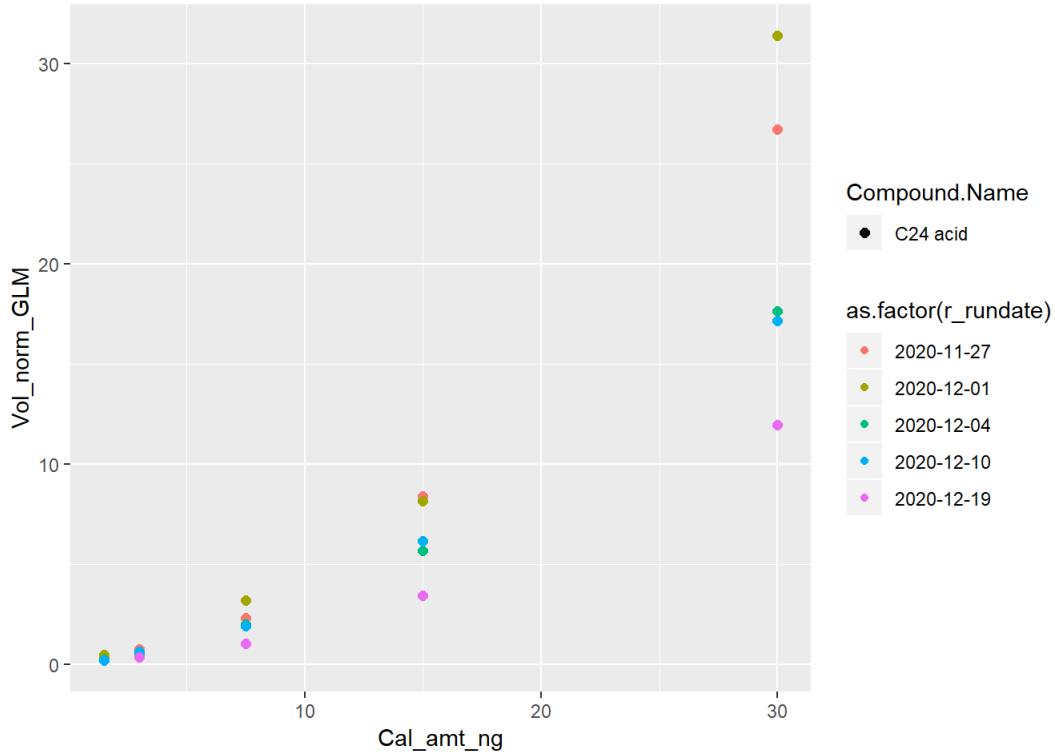
```
## Warning: Using size for a discrete variable is not advised.
```



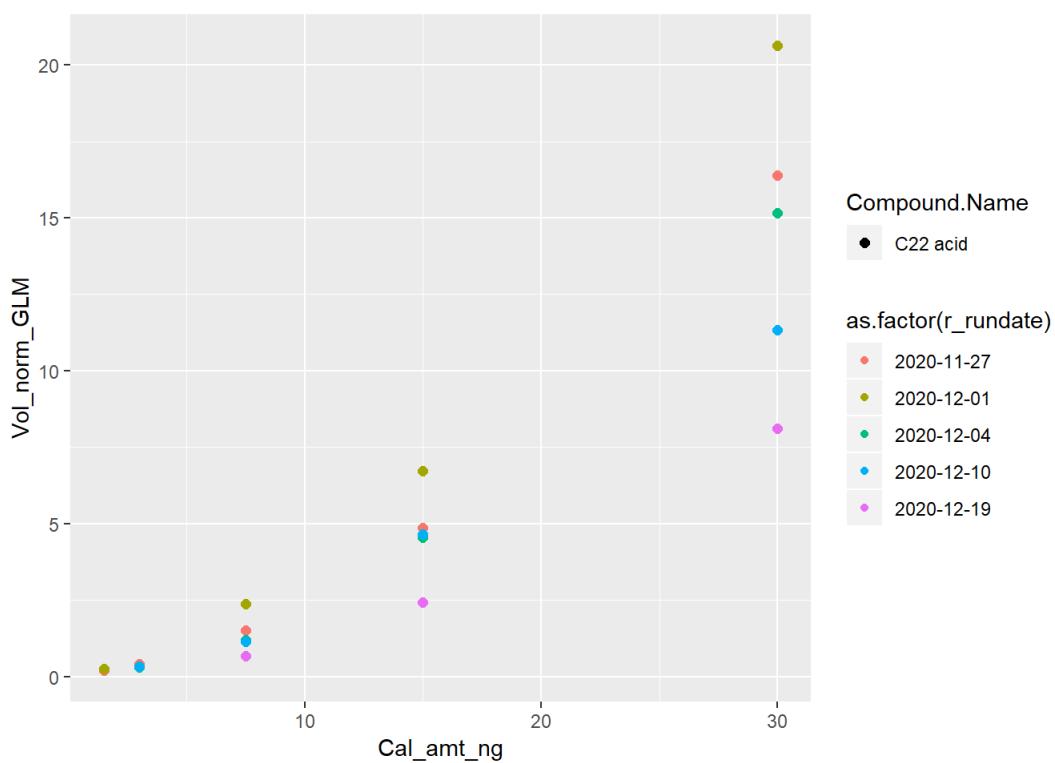
```
## Warning: Using size for a discrete variable is not advised.
```



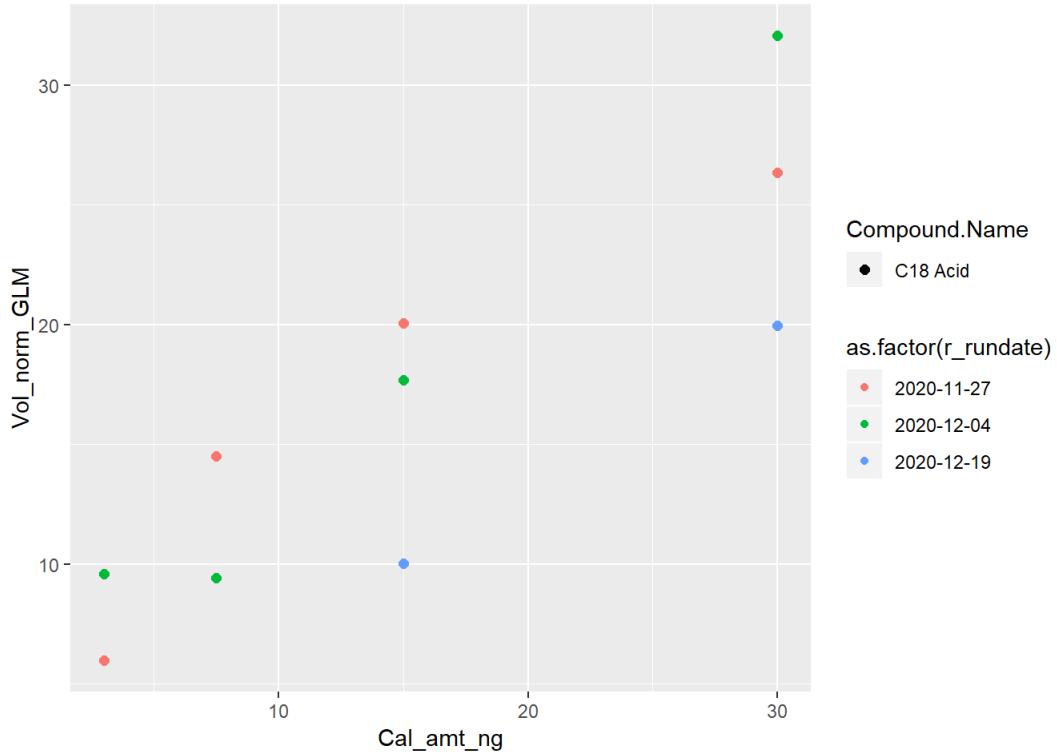
```
## Warning: Using size for a discrete variable is not advised.
```



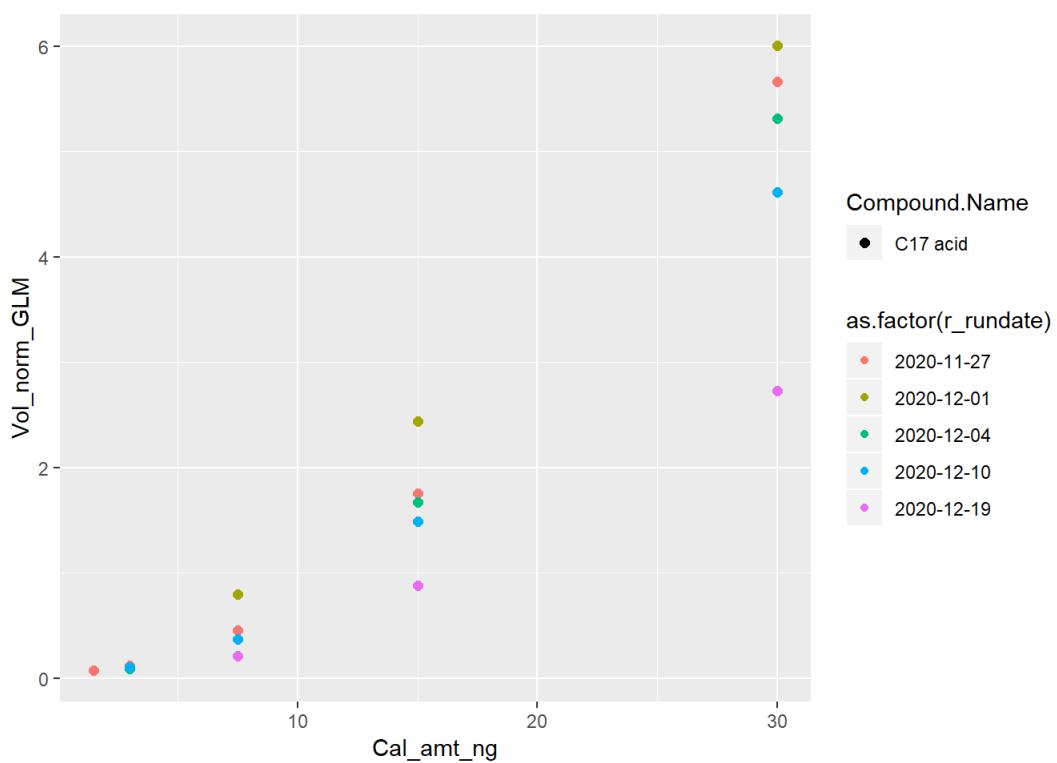
```
## Warning: Using size for a discrete variable is not advised.
```



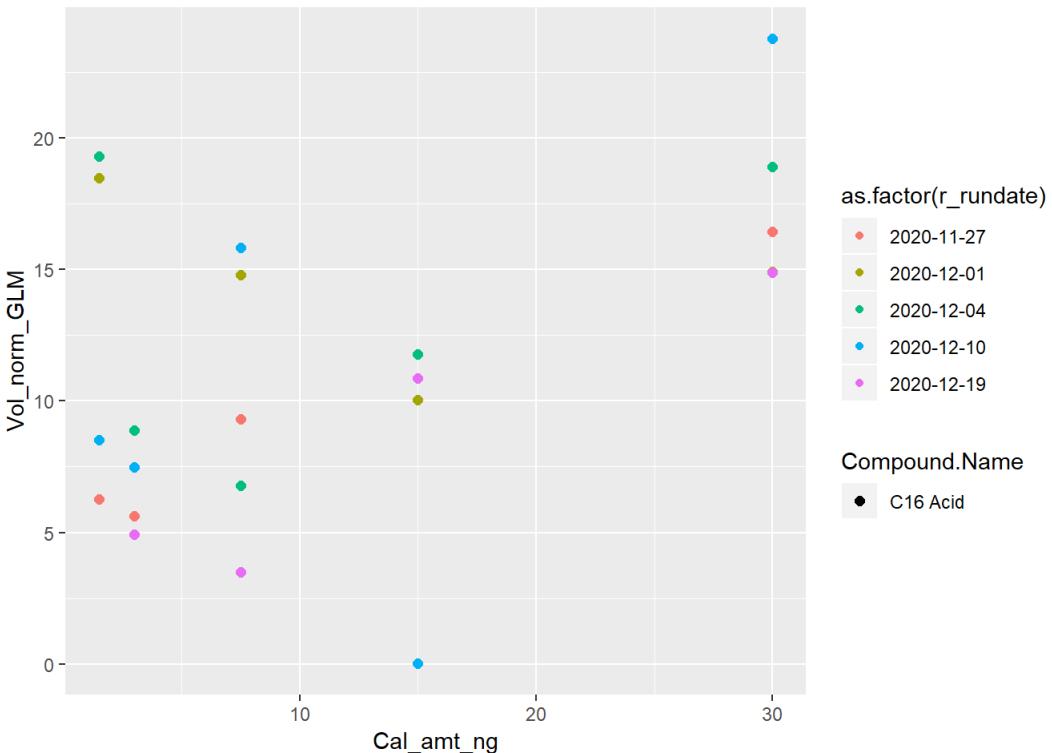
```
## Warning: Using size for a discrete variable is not advised.
```



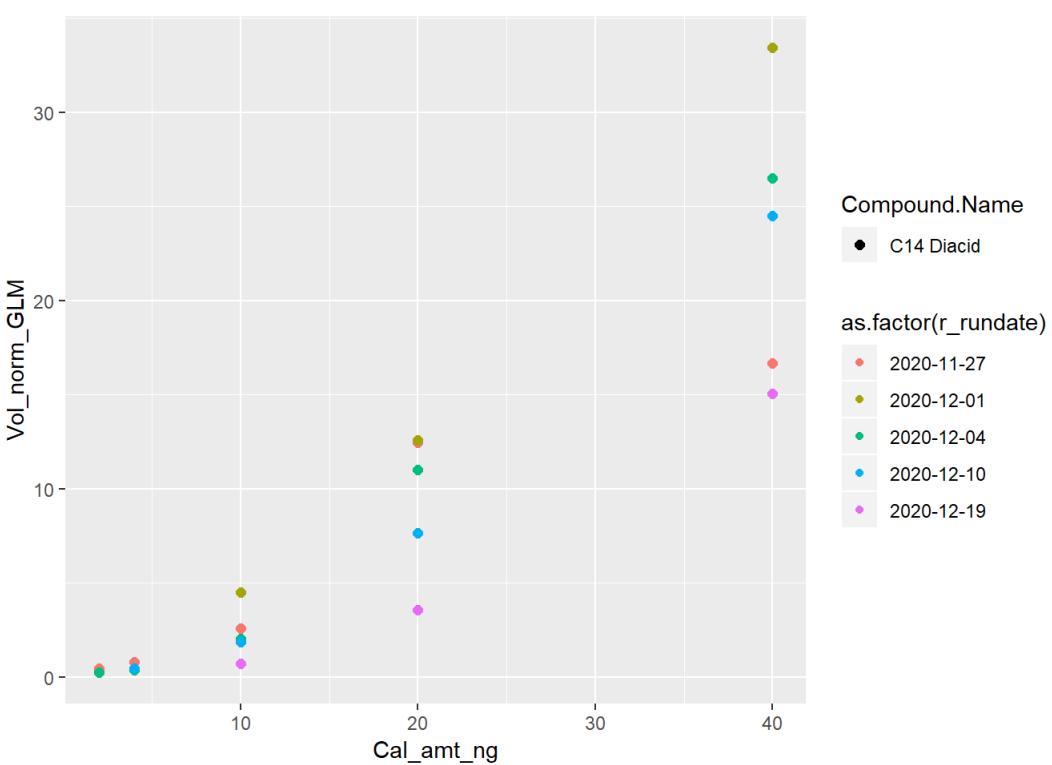
```
## Warning: Using size for a discrete variable is not advised.
```



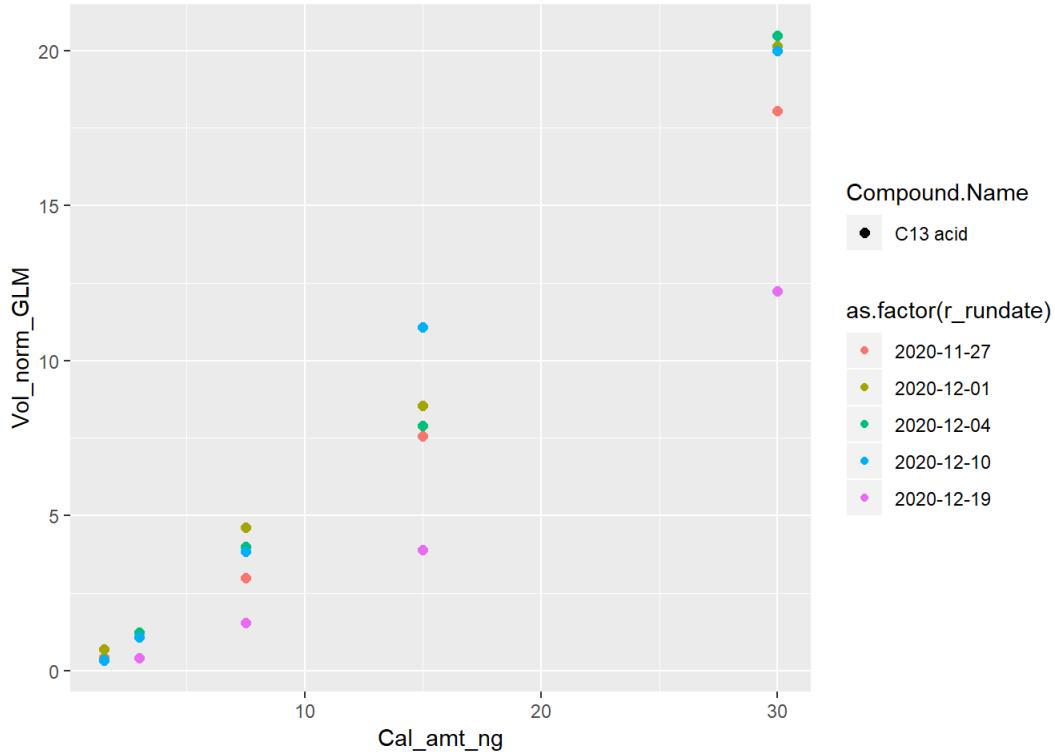
```
## Warning: Using size for a discrete variable is not advised.
```



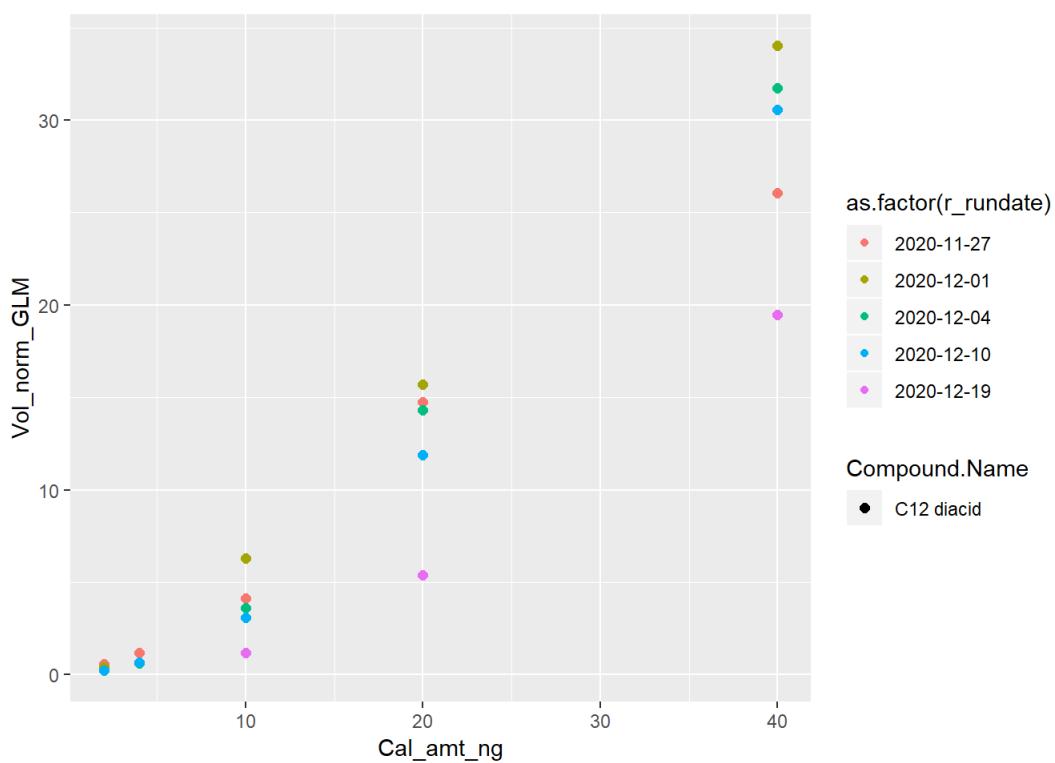
```
## Warning: Using size for a discrete variable is not advised.
```



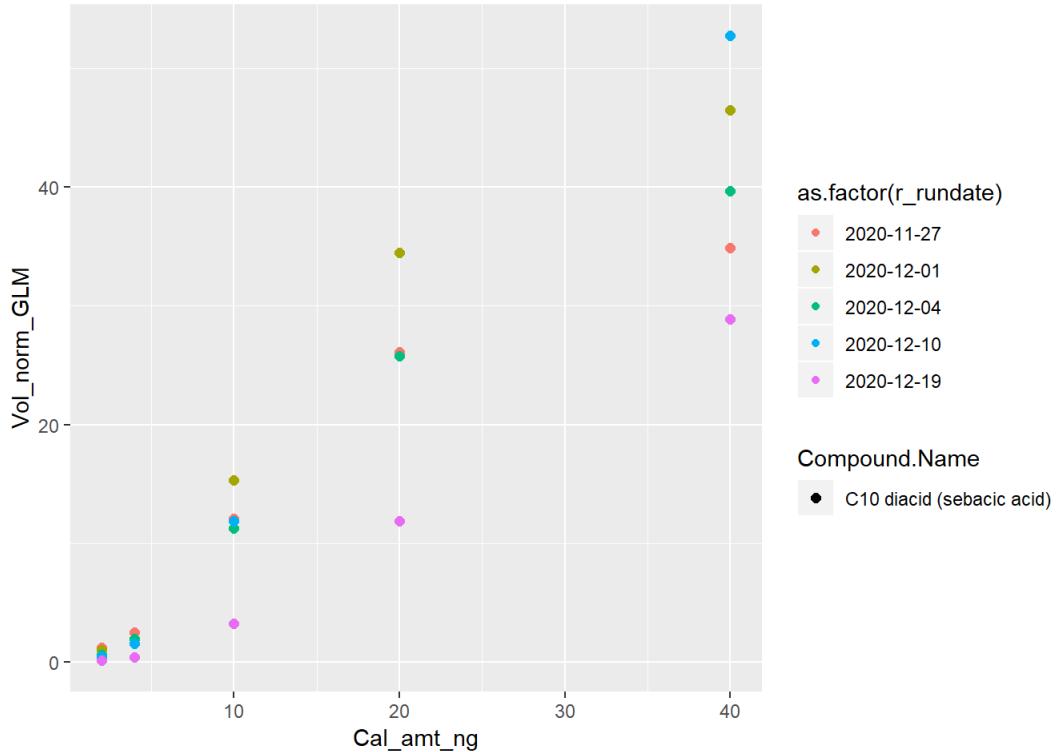
```
## Warning: Using size for a discrete variable is not advised.
```



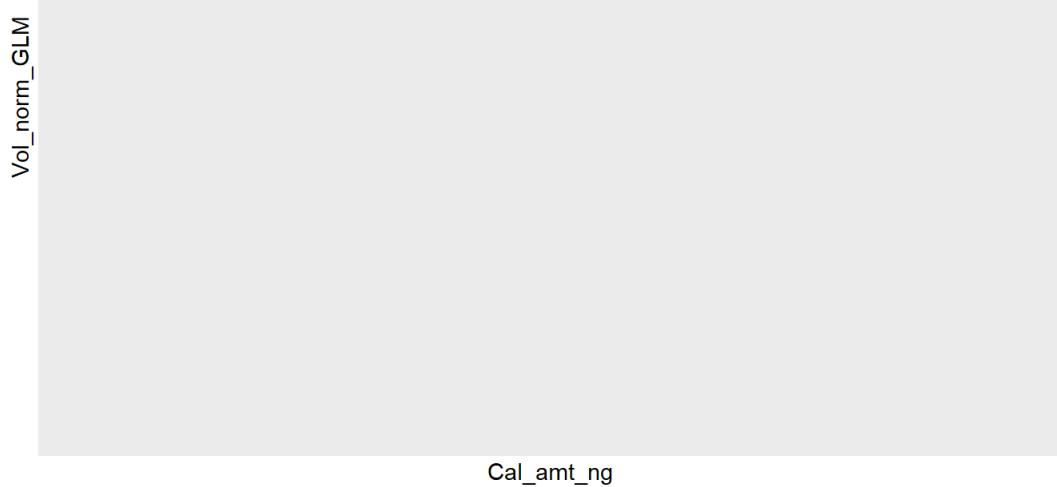
```
## Warning: Using size for a discrete variable is not advised.
```



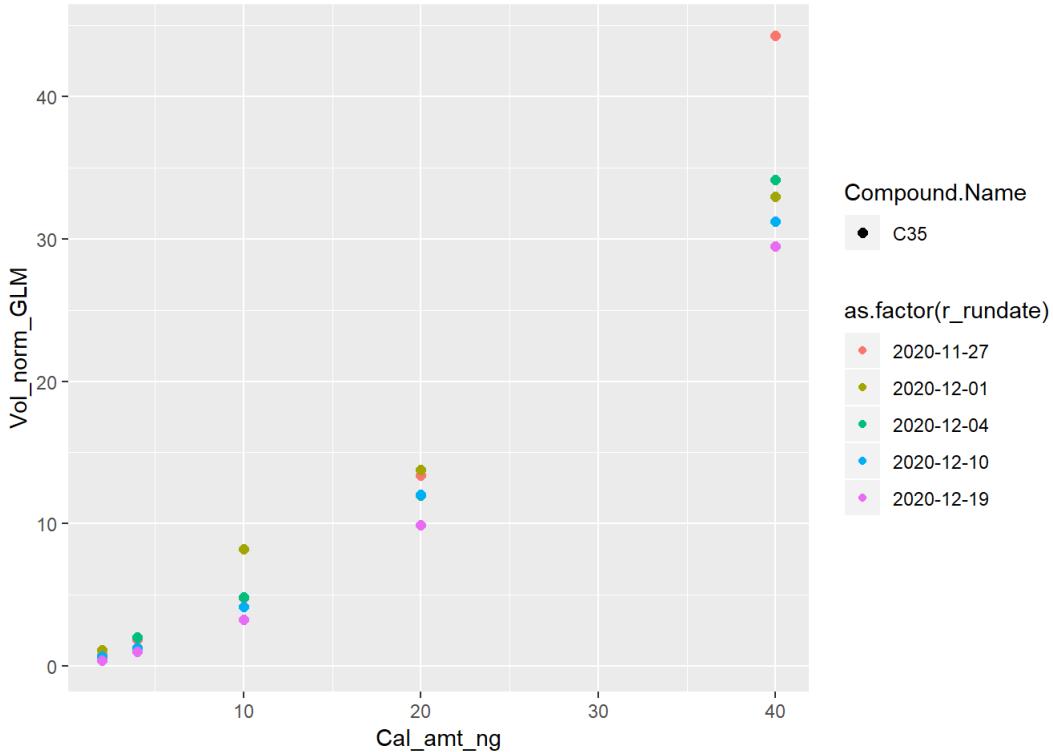
```
## Warning: Using size for a discrete variable is not advised.
```



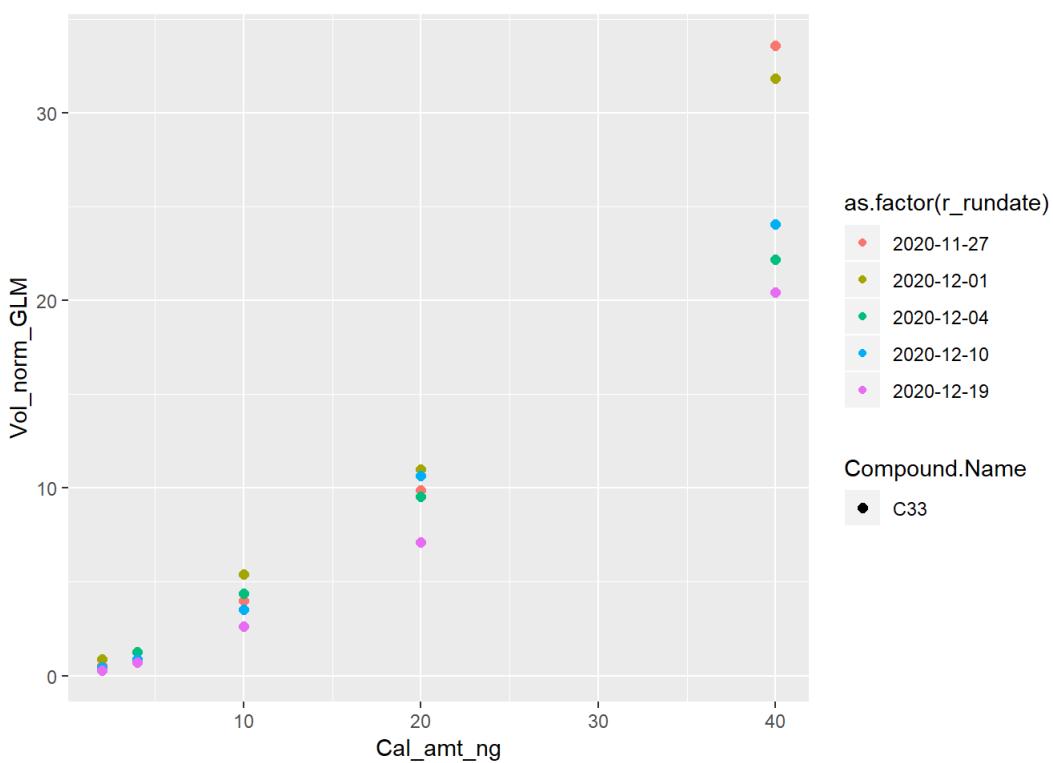
```
## Warning: Using size for a discrete variable is not advised.
```



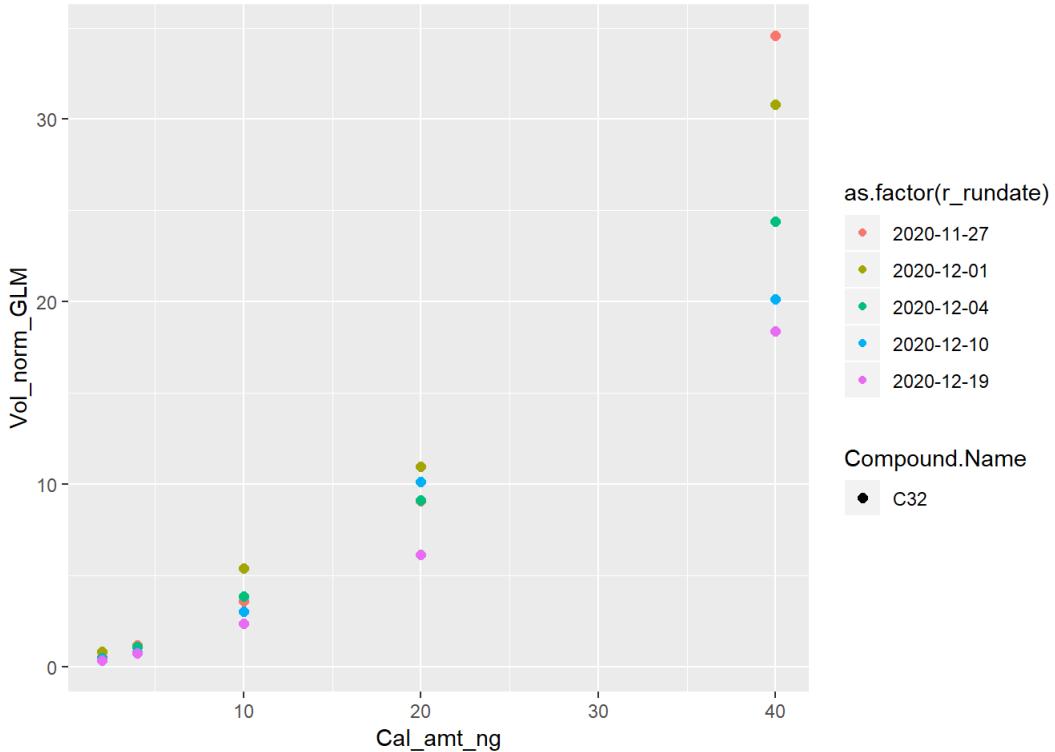
```
## Warning: Using size for a discrete variable is not advised.
```



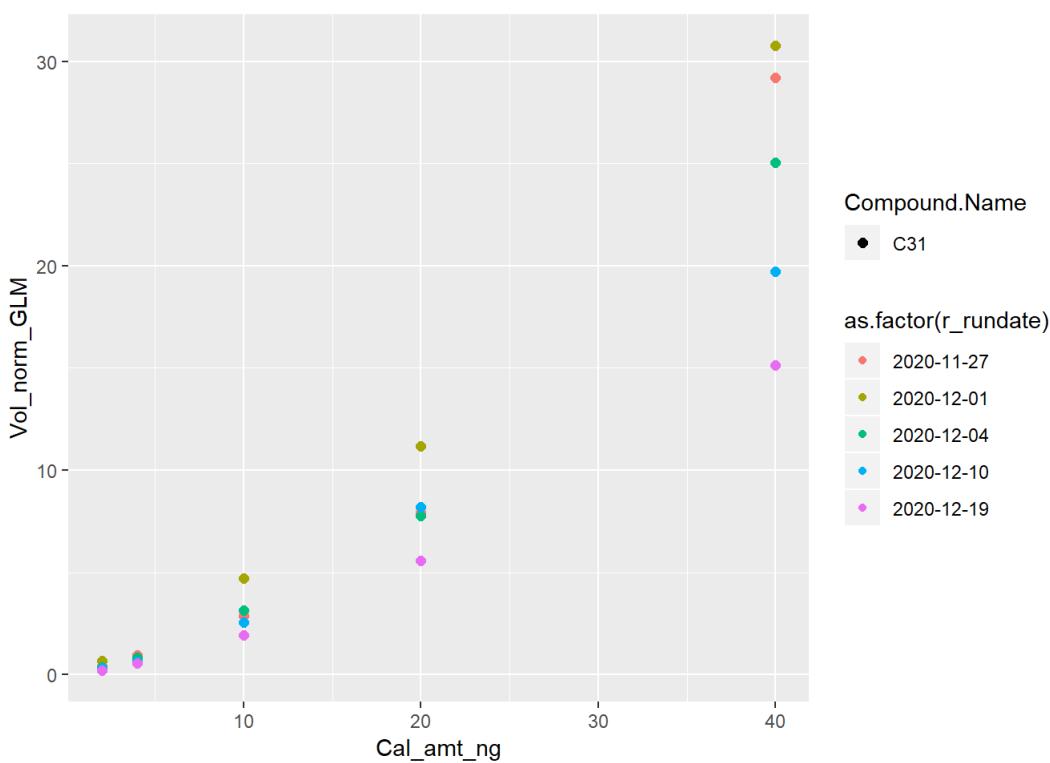
```
## Warning: Using size for a discrete variable is not advised.
```



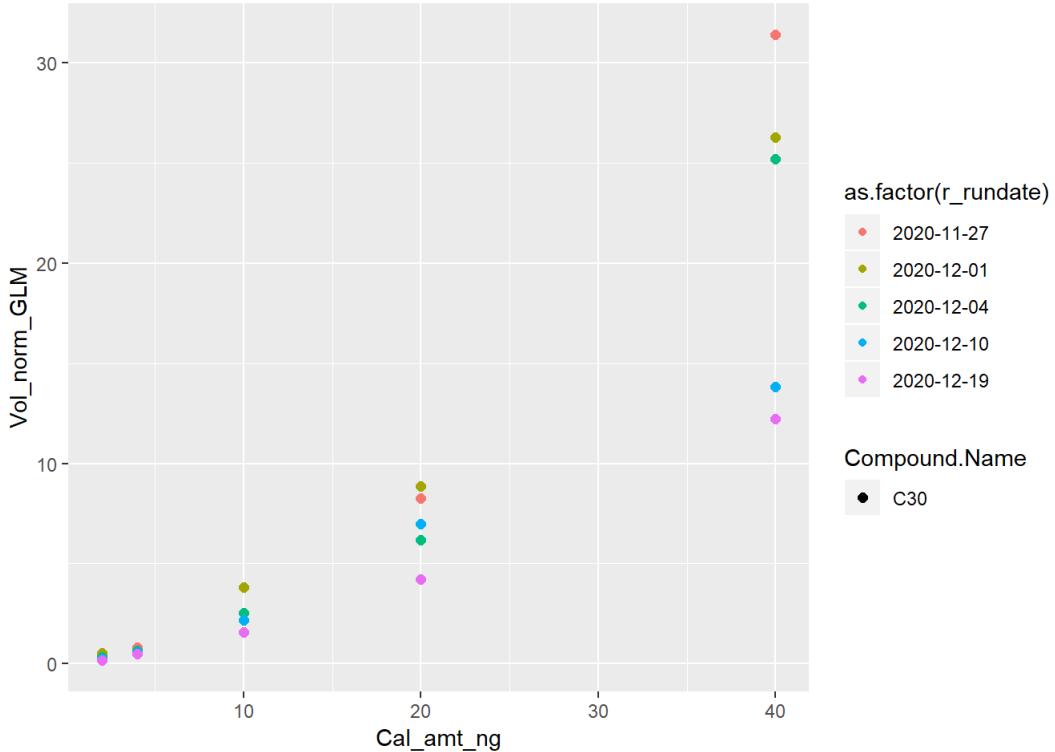
```
## Warning: Using size for a discrete variable is not advised.
```



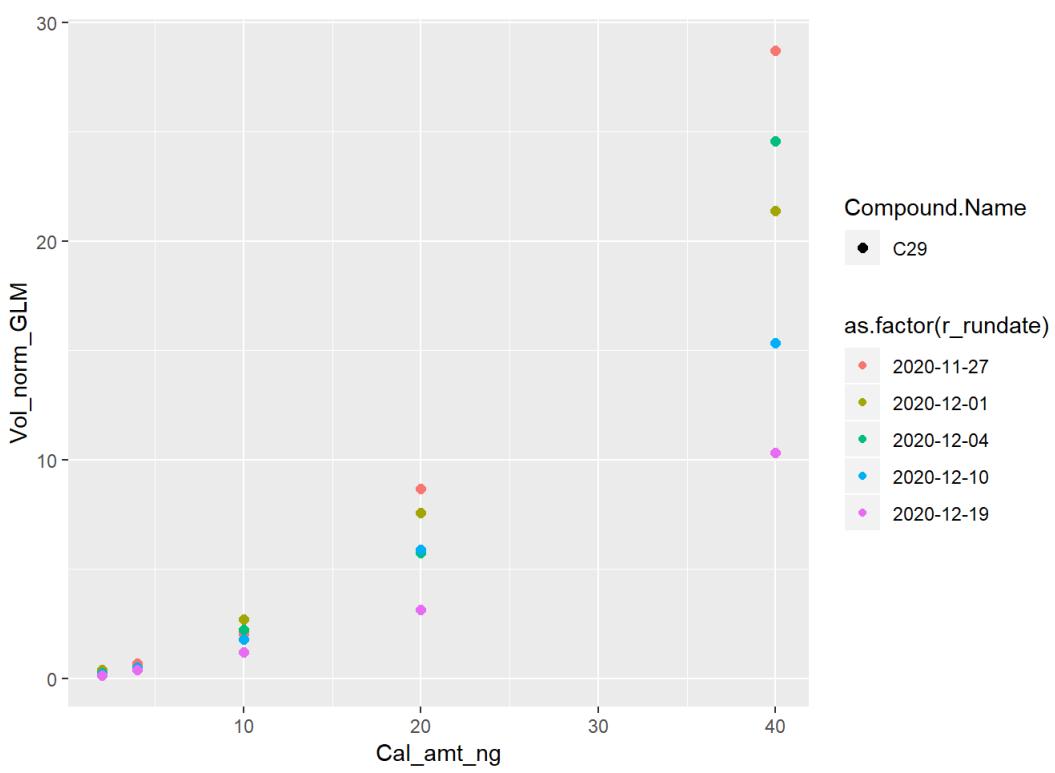
```
## Warning: Using size for a discrete variable is not advised.
```



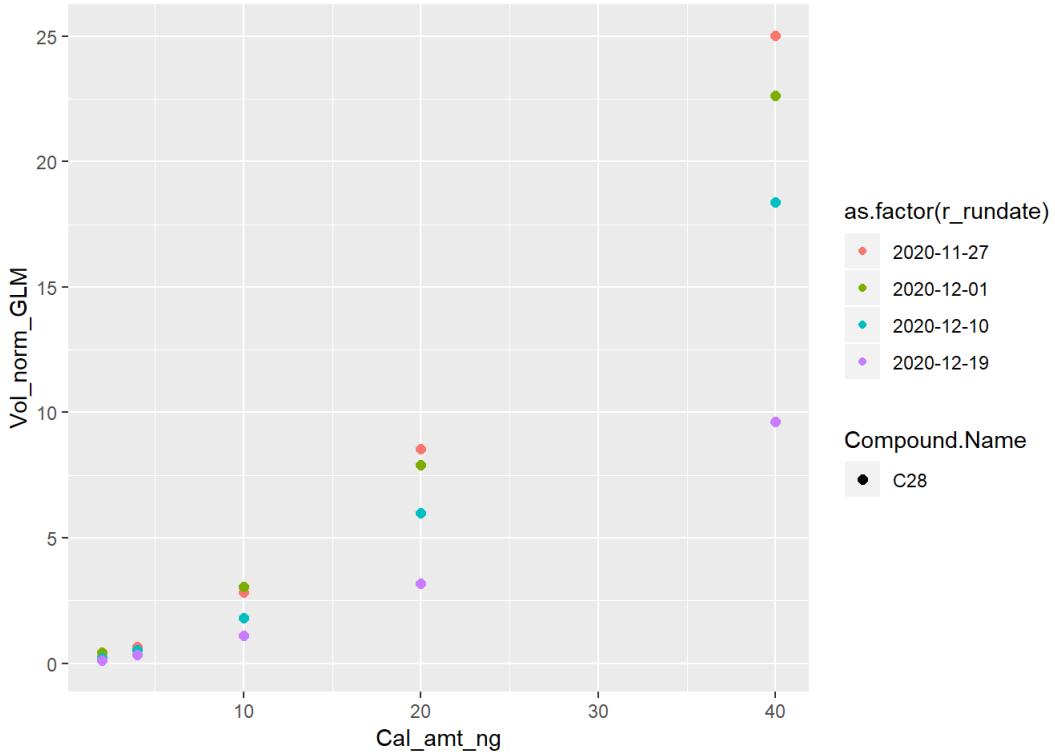
```
## Warning: Using size for a discrete variable is not advised.
```



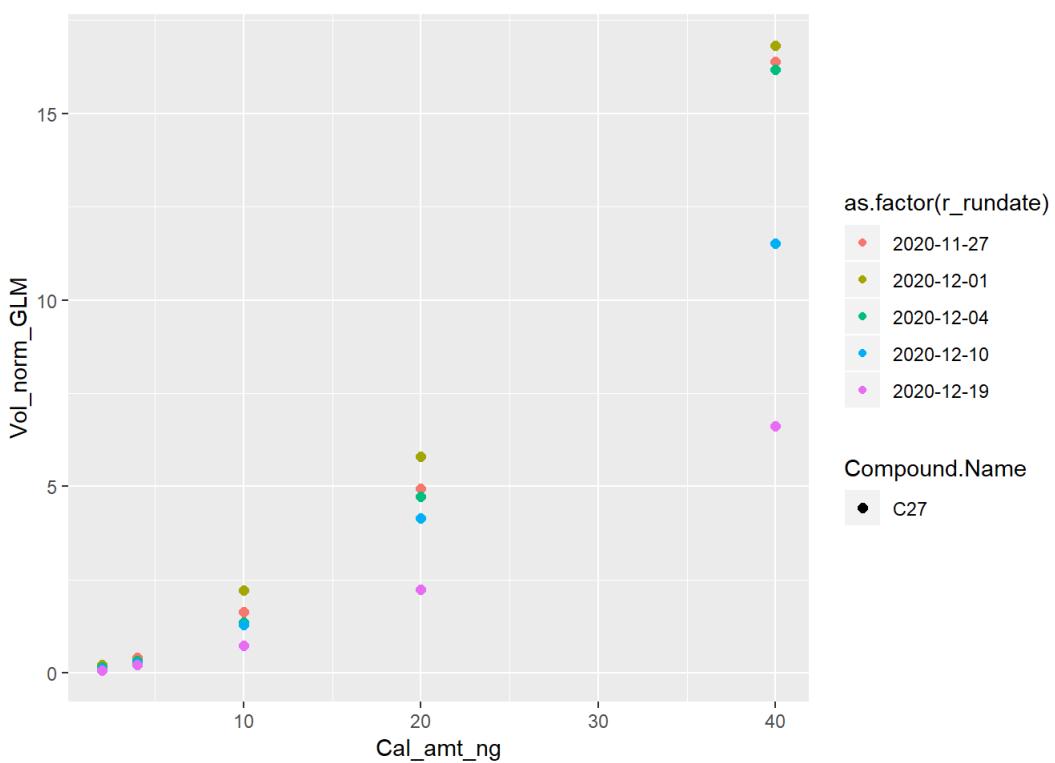
```
## Warning: Using size for a discrete variable is not advised.
```



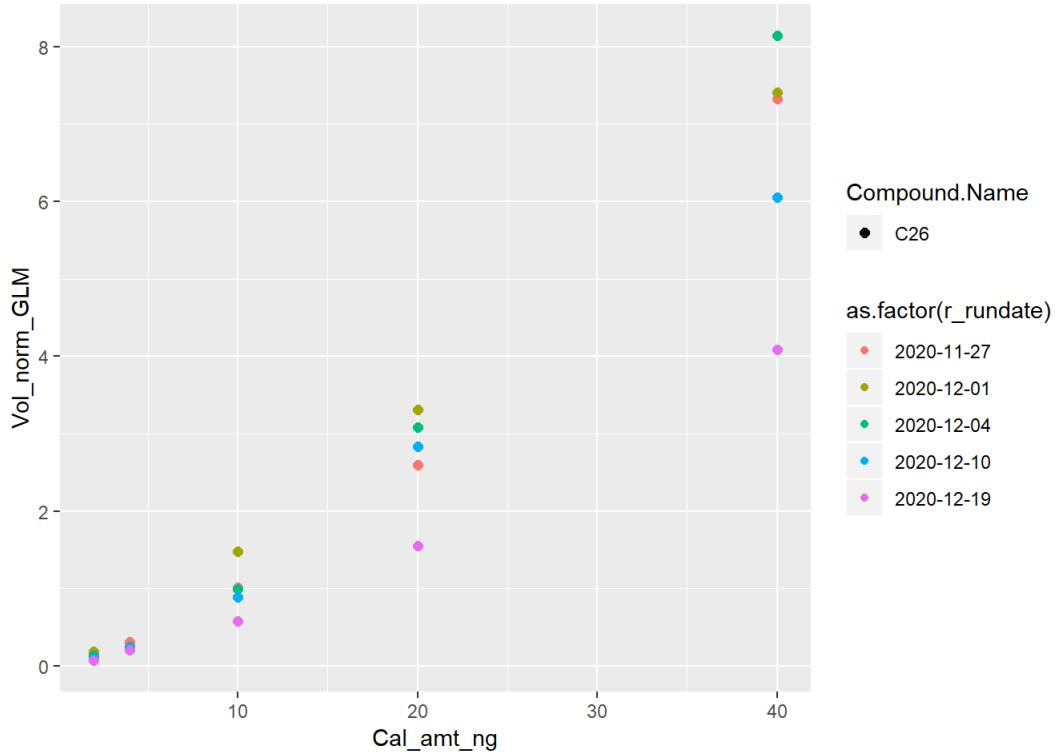
```
## Warning: Using size for a discrete variable is not advised.
```



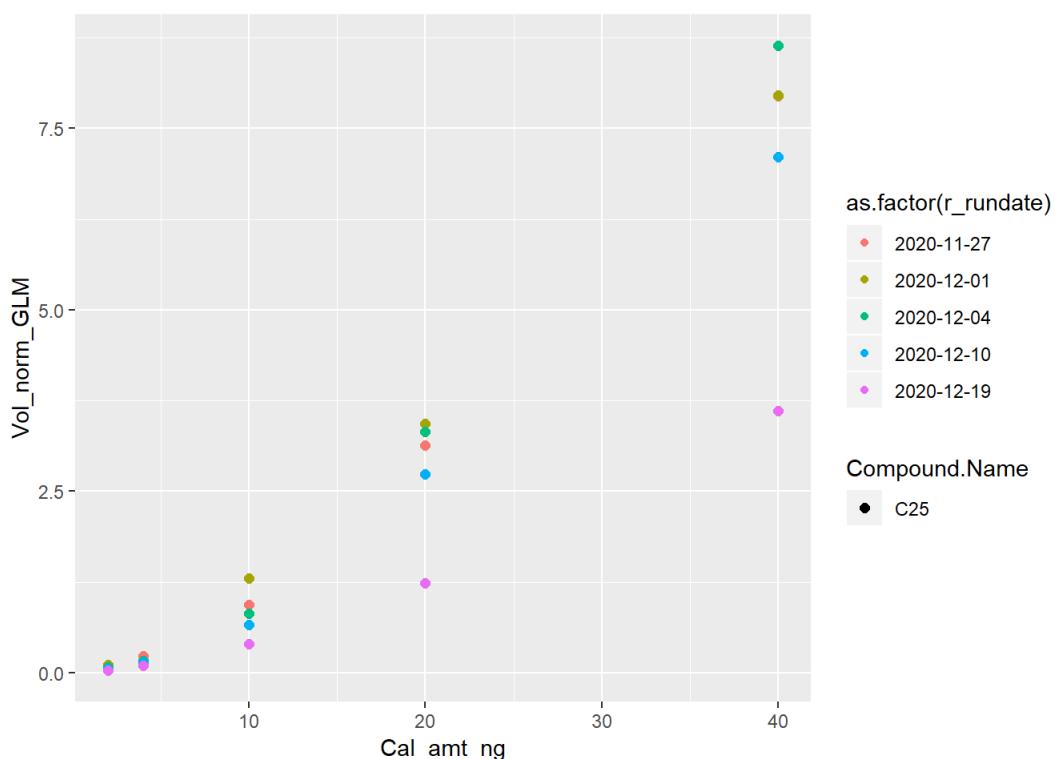
```
## Warning: Using size for a discrete variable is not advised.
```



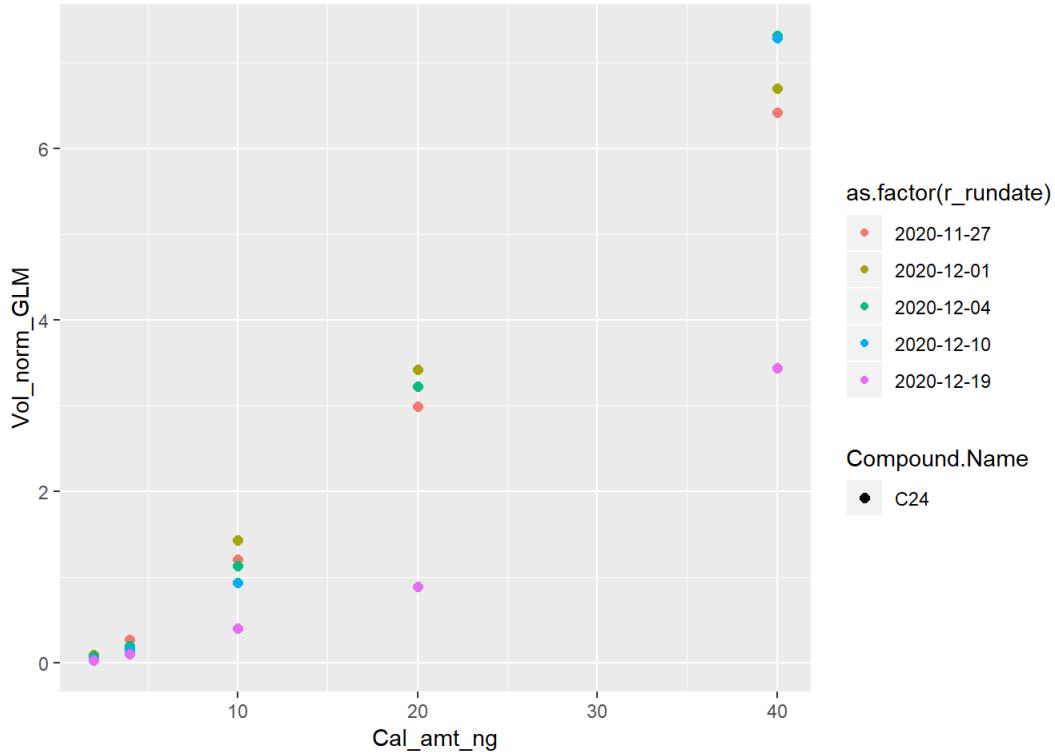
```
## Warning: Using size for a discrete variable is not advised.
```



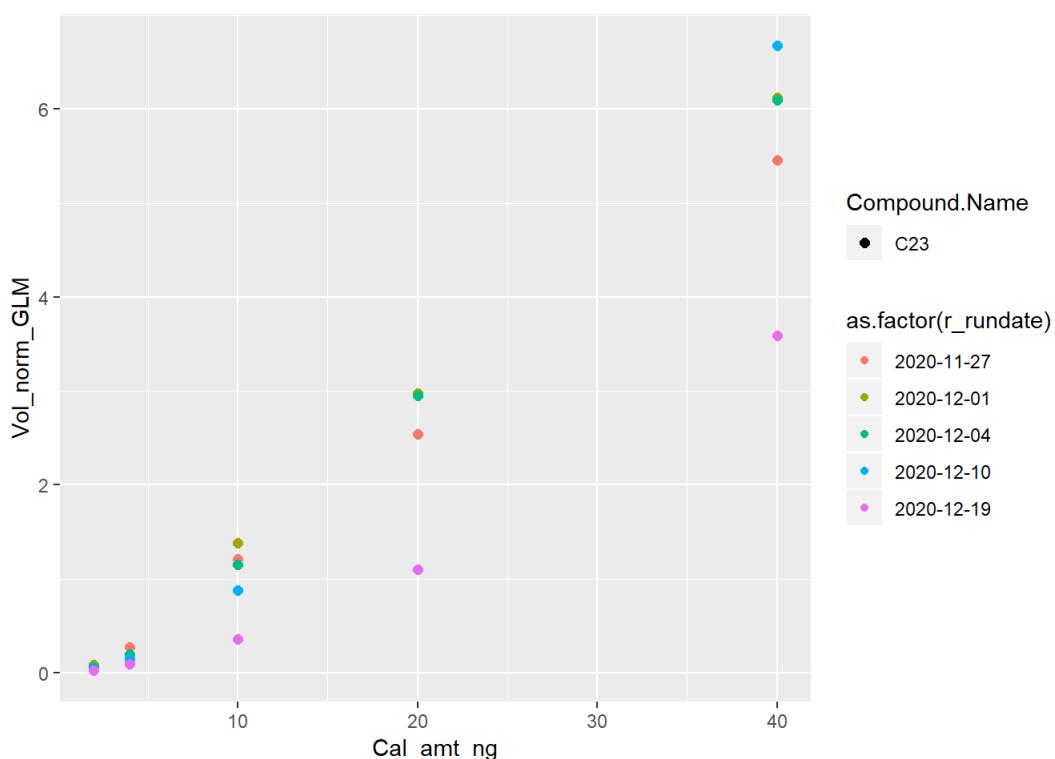
```
## Warning: Using size for a discrete variable is not advised.
```



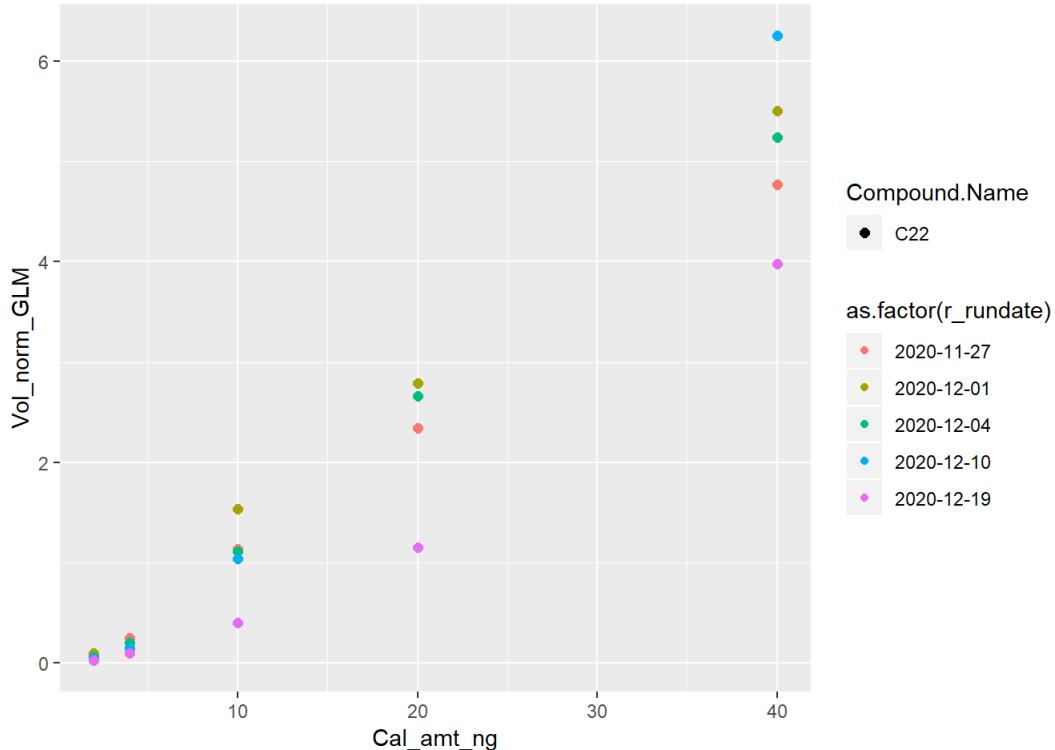
```
## Warning: Using size for a discrete variable is not advised.
```



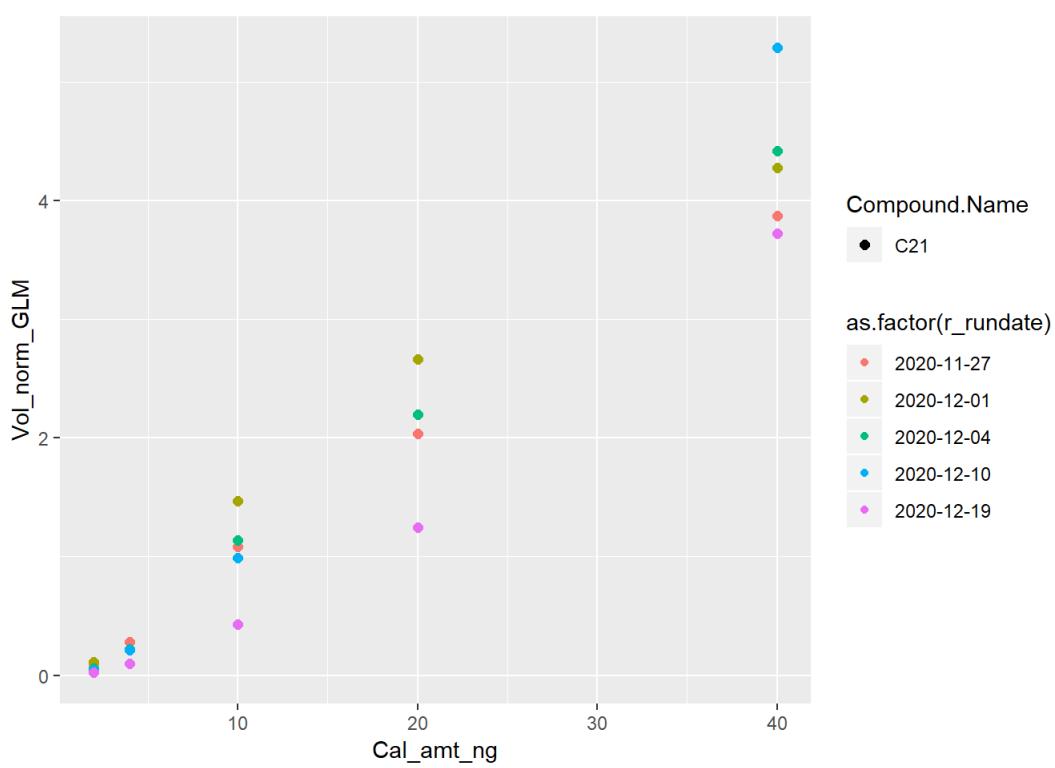
```
## Warning: Using size for a discrete variable is not advised.
```



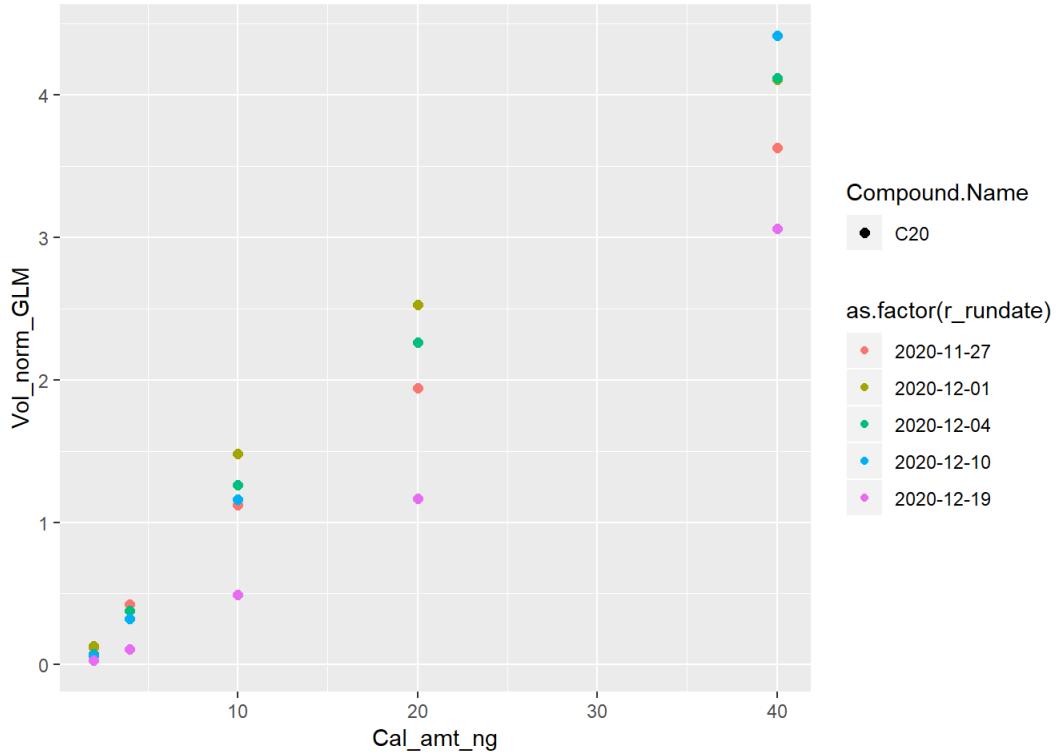
```
## Warning: Using size for a discrete variable is not advised.
```



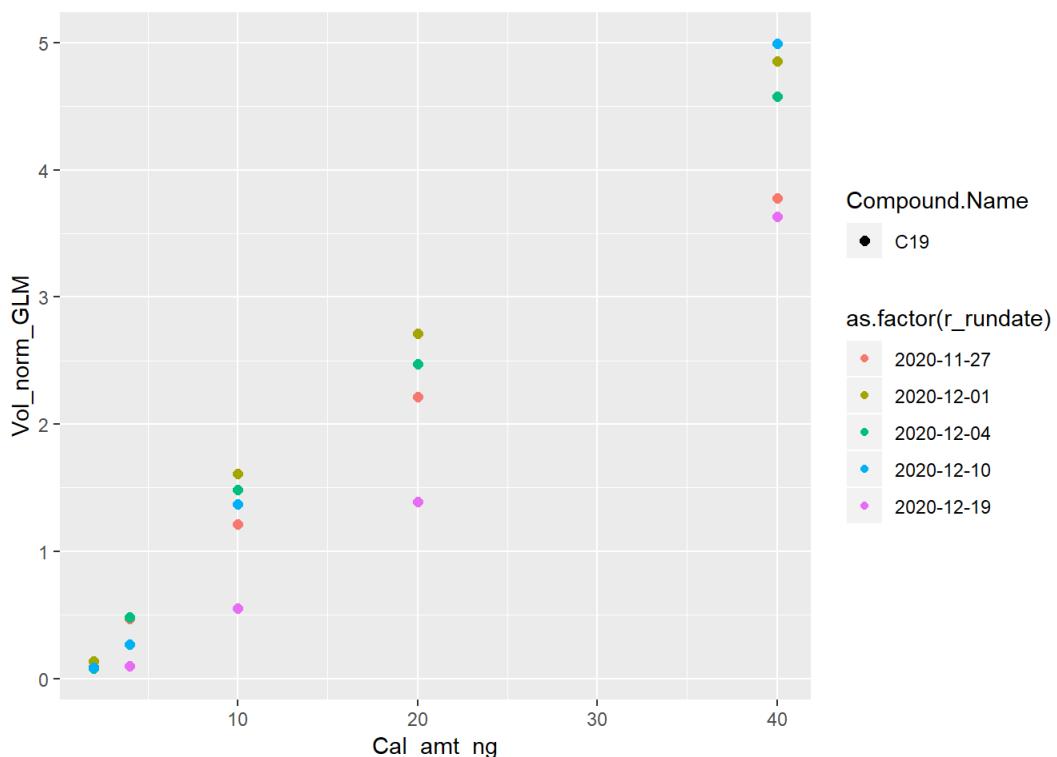
```
## Warning: Using size for a discrete variable is not advised.
```



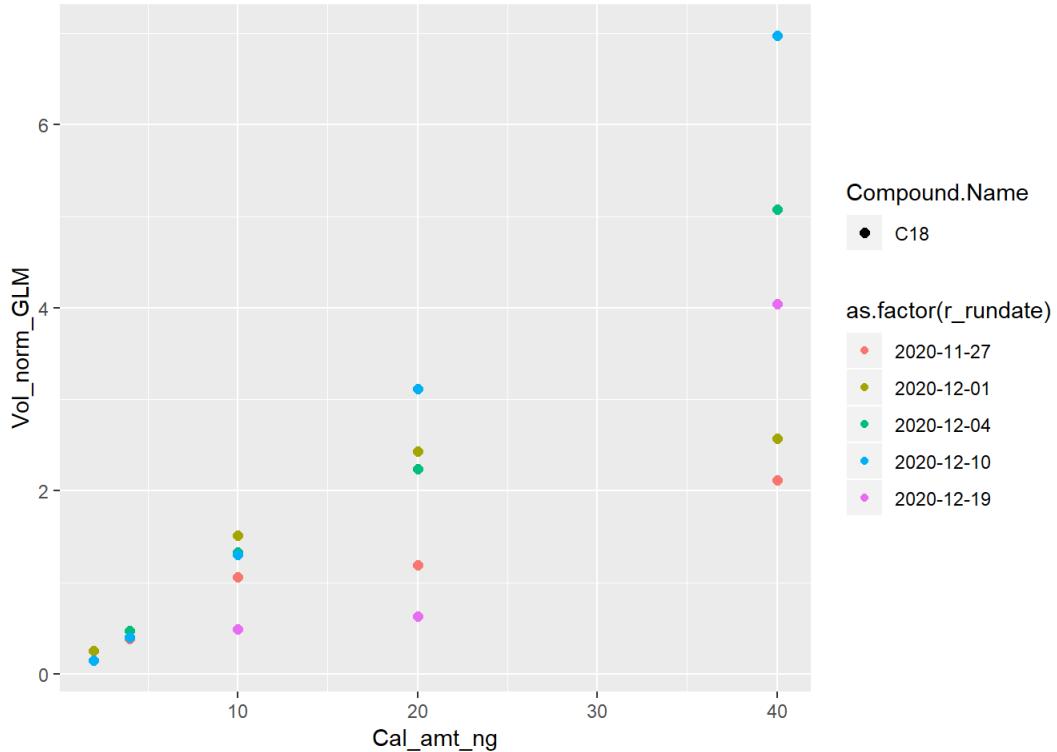
```
## Warning: Using size for a discrete variable is not advised.
```



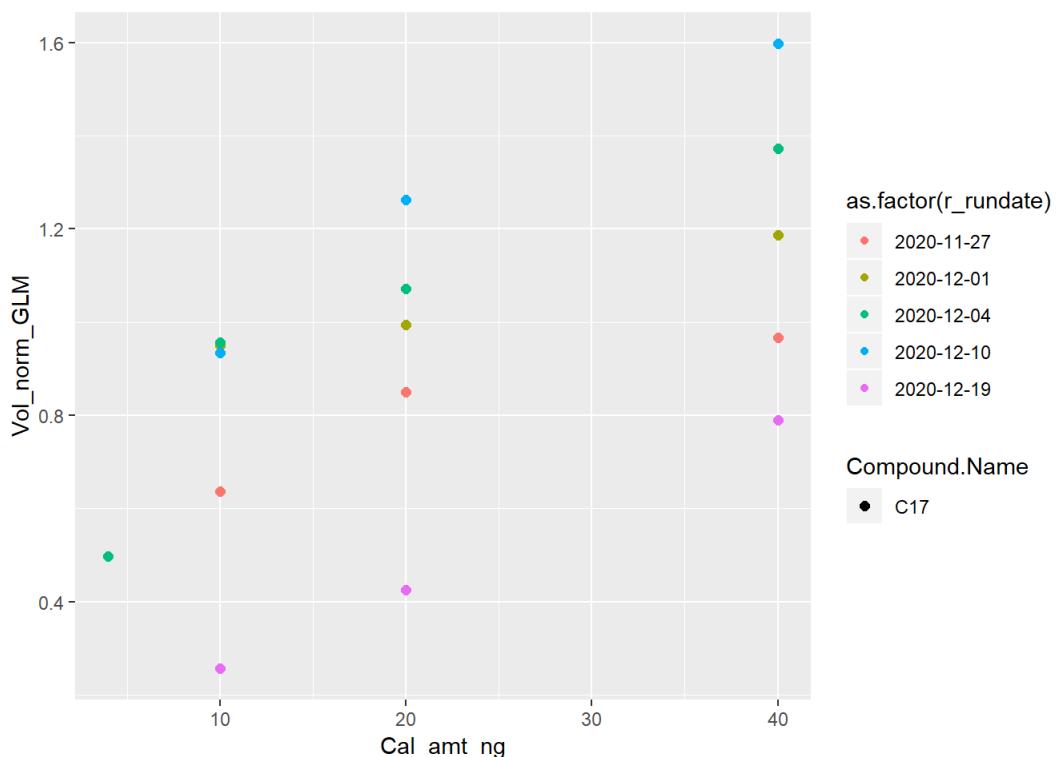
```
## Warning: Using size for a discrete variable is not advised.
```



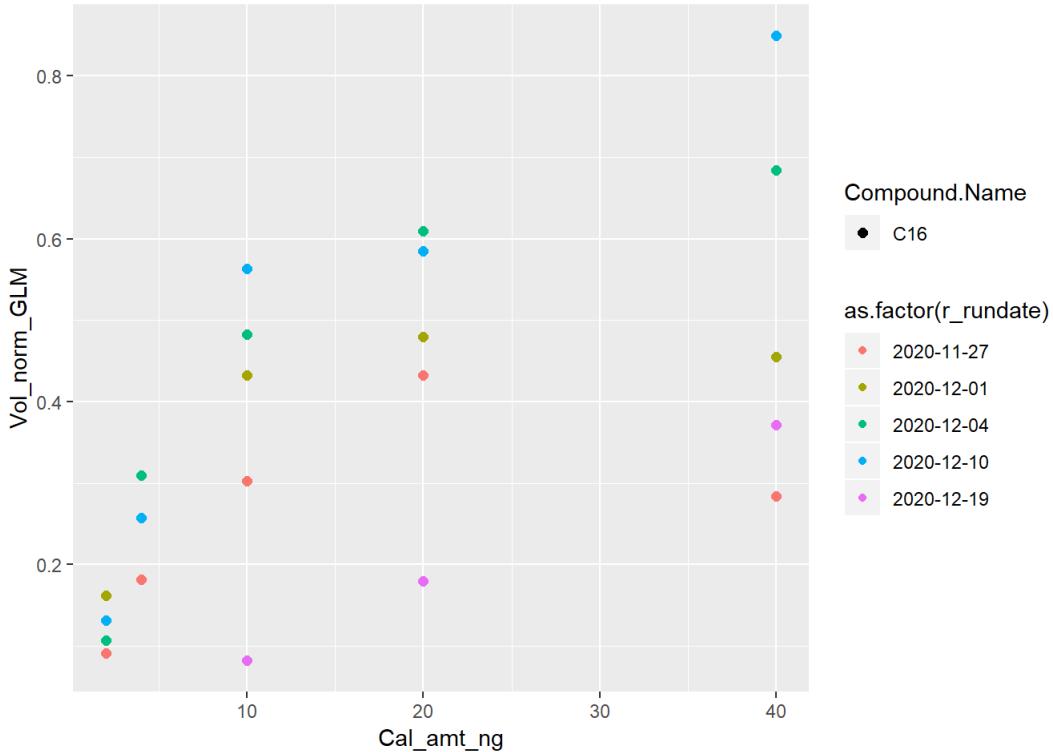
```
## Warning: Using size for a discrete variable is not advised.
```



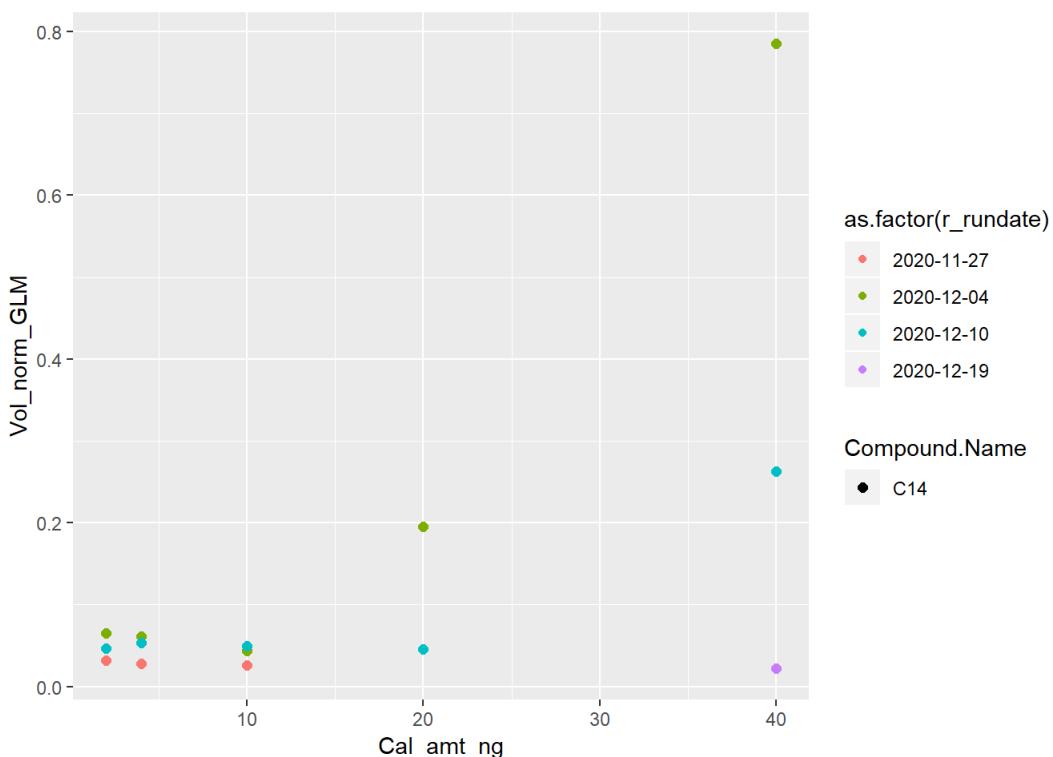
```
## Warning: Using size for a discrete variable is not advised.
```



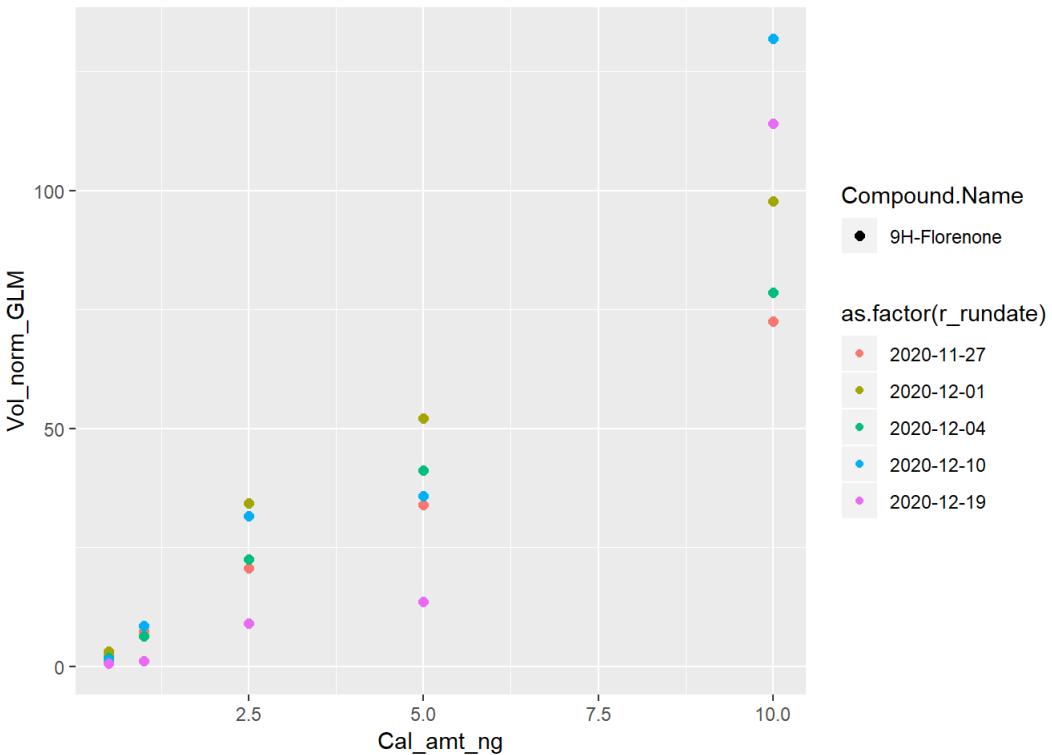
```
## Warning: Using size for a discrete variable is not advised.
```



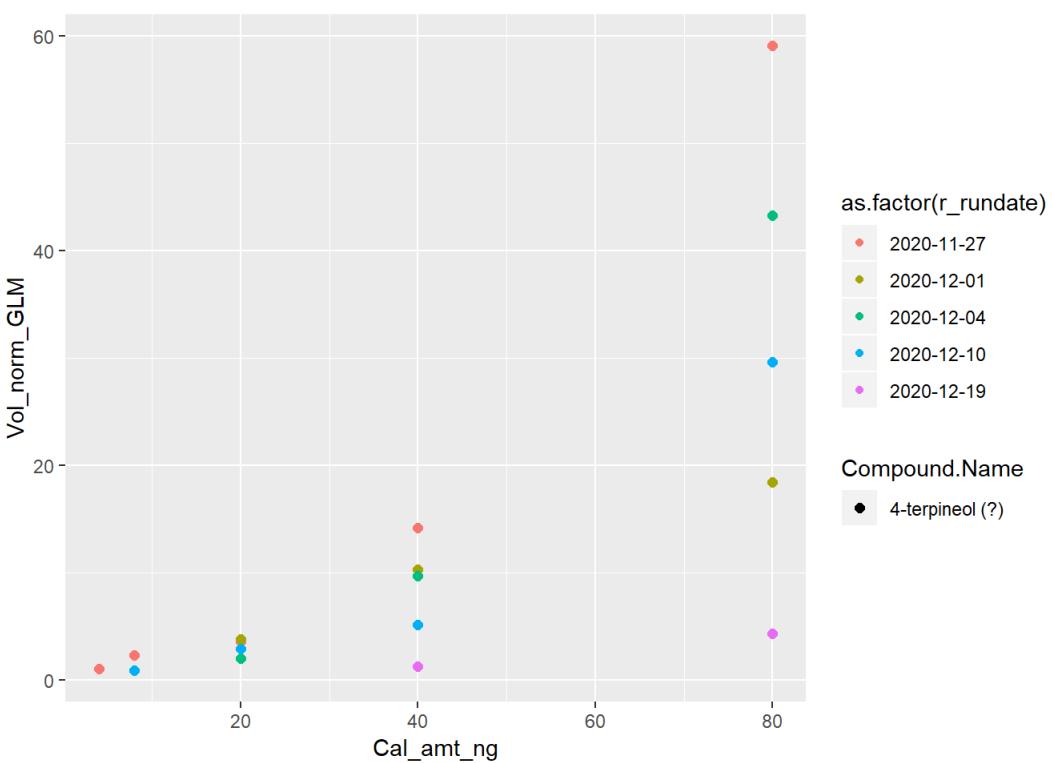
```
## Warning: Using size for a discrete variable is not advised.
```



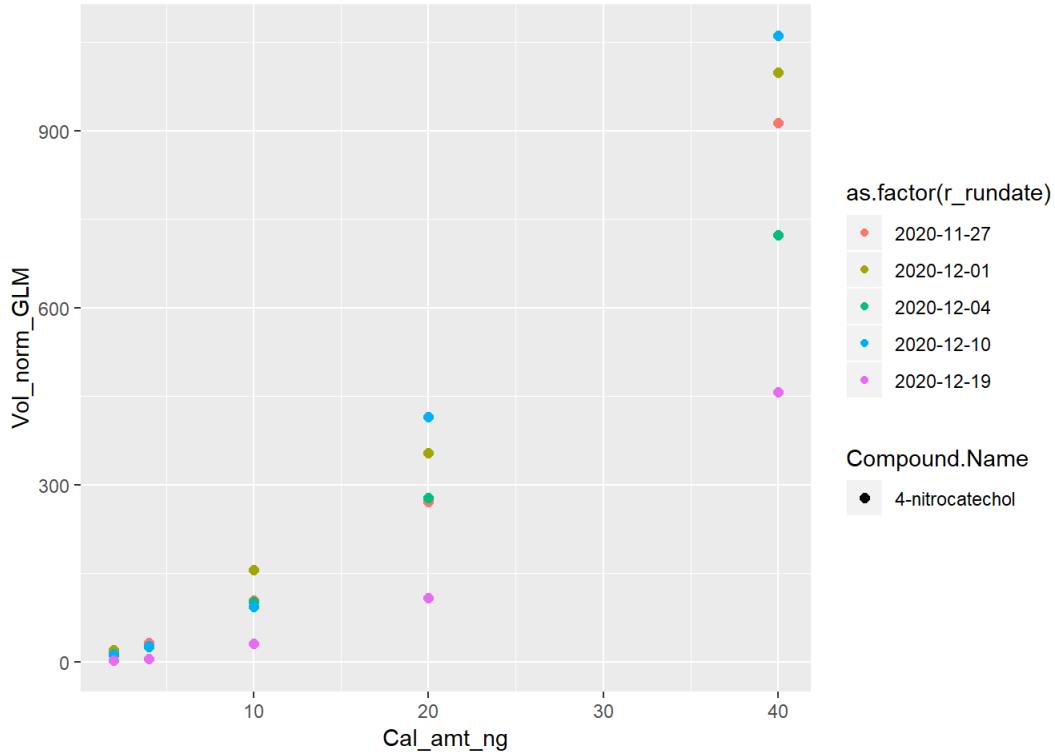
```
## Warning: Using size for a discrete variable is not advised.
```



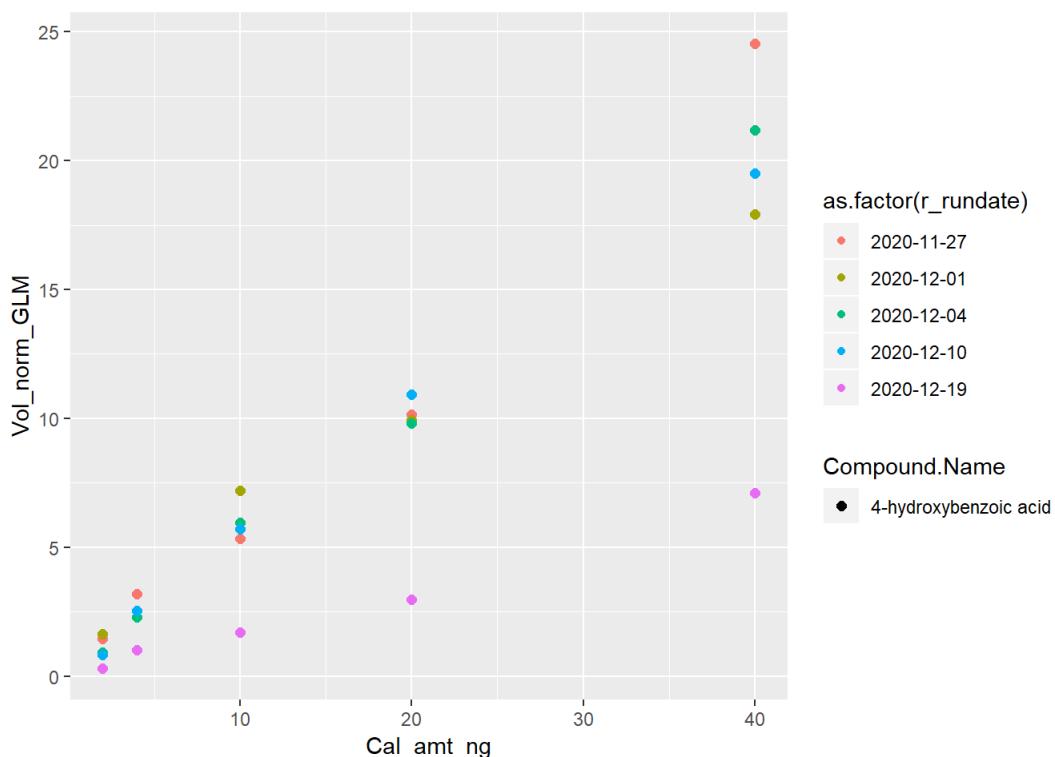
```
## Warning: Using size for a discrete variable is not advised.
```



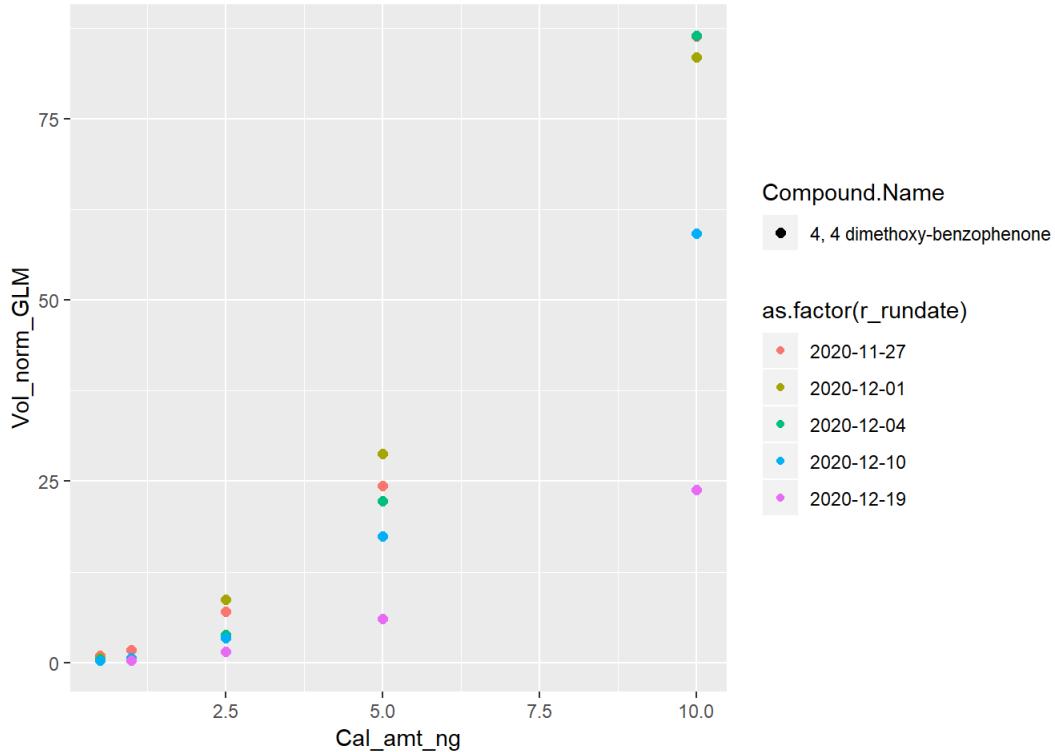
```
## Warning: Using size for a discrete variable is not advised.
```



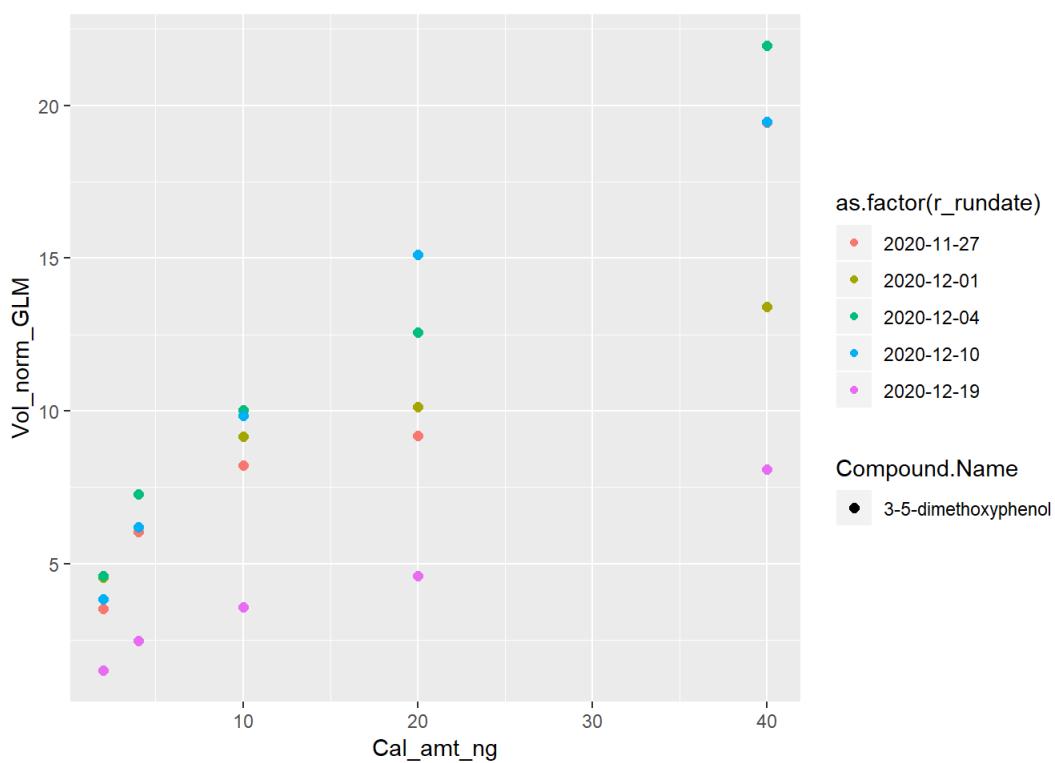
```
## Warning: Using size for a discrete variable is not advised.
```



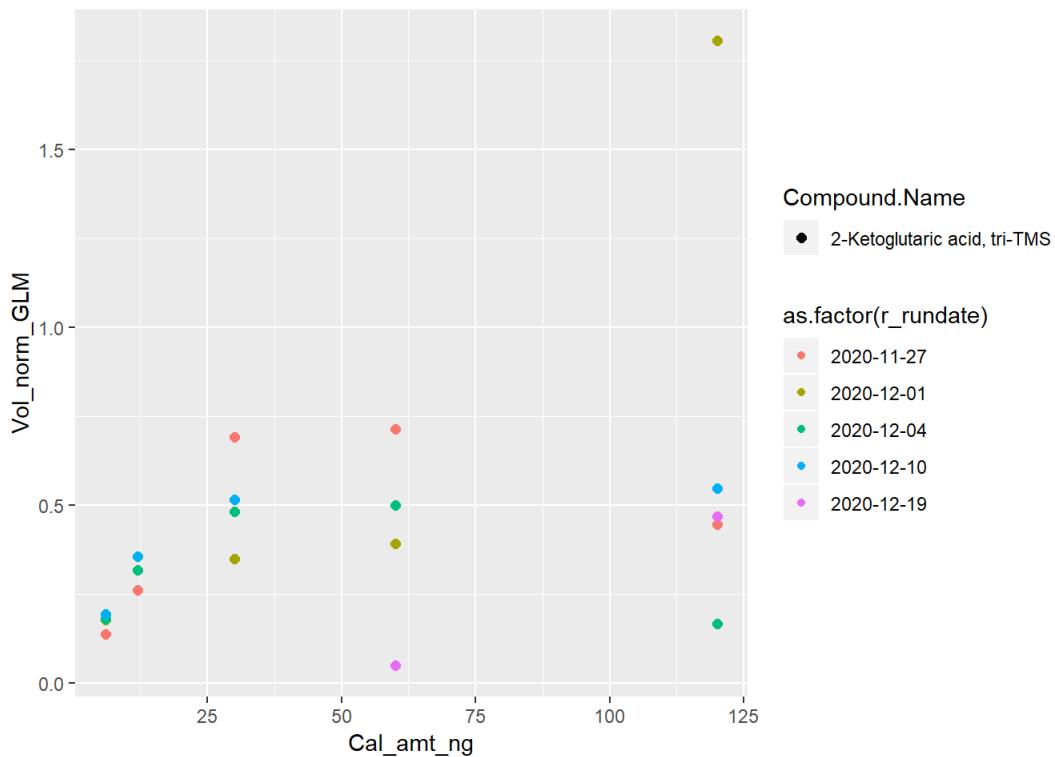
```
## Warning: Using size for a discrete variable is not advised.
```



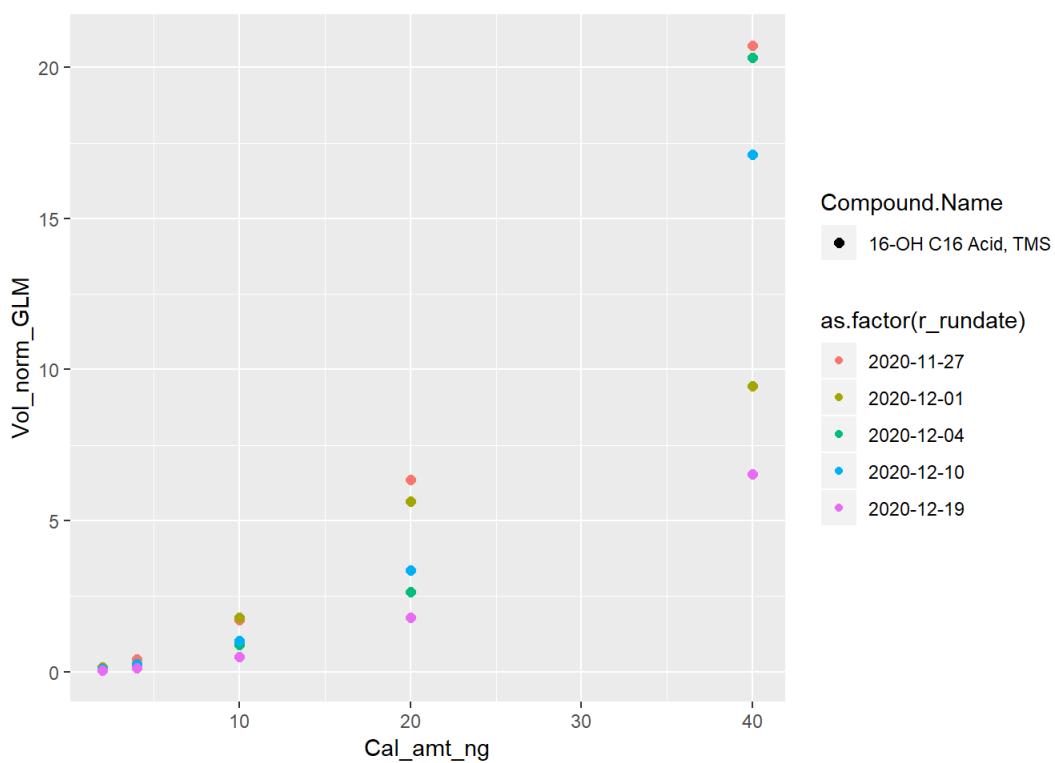
```
## Warning: Using size for a discrete variable is not advised.
```



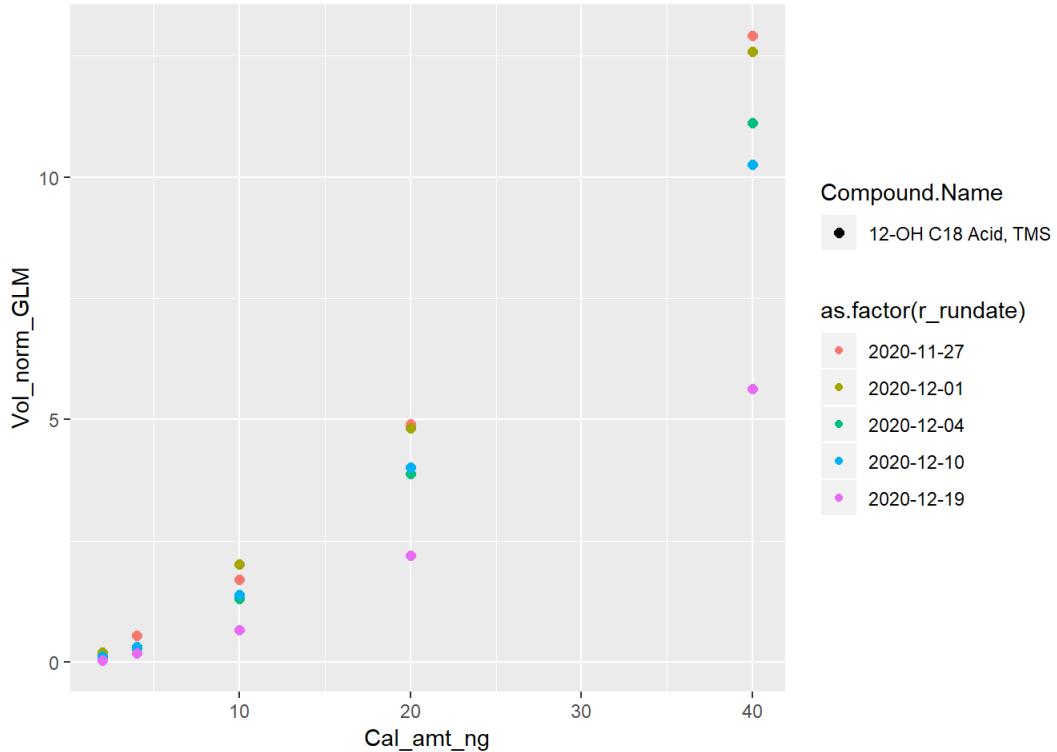
```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```

Vol_norm_GLM

Cal_amt_ng

```
## Warning: Using size for a discrete variable is not advised.
```

Vol_norm_GLM

Cal_amt_ng

```
## Warning: Using size for a discrete variable is not advised.
```

Vol_norm_GLM

Cal_amt_ng

```
## Warning: Using size for a discrete variable is not advised.
```

Vol_norm_GLM

Cal_amt_ng

```
## Warning: Using size for a discrete variable is not advised.
```

Vol_norm_GLM

Cal_amt_ng

```
## Warning: Using size for a discrete variable is not advised.
```

Vol_norm_GLM

Cal_amt_ng

```
## Warning: Using size for a discrete variable is not advised.
```

Vol_norm_GLM

Cal_amt_ng

```
## Warning: Using size for a discrete variable is not advised.
```

Vol_norm_GLM

The cleaned cal curves look

Cal_amt_ng

significantly better.

```
Cat_trimmed <- Cat_full %>%
  dplyr::select(-c(RT1, RT2, Retention_index, d_alkane_RTI))

ES_withcat <- ES_cleaned1 %>%
  left_join(Cat_trimmed, by = c("Compound.Name" = "Name"))
```

```
OSR2 <- function(predictions, train, test) {
  SSE <- sum((test - predictions)^2)
  SST <- sum((test - mean(train))^2)
  r2 <- 1 - SSE/SST
  return(r2)
}
```

Modeling First Steps

```
ES_for_quant <- ES_withcat %>%
  dplyr::select(-c(BlobID, Description, Group.Name, Constellation.Name, Inclusion, LRI.I, Library.ID, Library.NIST.ID, Library.Match.Factor,
                  Library.Reverse.Match.Factor, Library.Probability, Review.Status, Internal.Standard, Retention.I..min.,
                  Peak.Value, Volume, Quantifier.1., Quantifier.2., Quantifier.Response.1., Quantifier.Response.2.,
                  Volume.Ratio, File_num,
                  Category, Filter_num, Filter_punches, IOP, AMZ_date, T_O_D, Date_num, Internal_load, Cal
                  _Point_number, Run_date,
                  Cal_Point, LRI_diff, unique.analyte, comp.n, pct_of_samp, IS_glm, Func_group_cat, Library
                  .Name))

# split into training and test
#train.ids = sample(nrow(ES_for_quant), .7*nrow(ES_for_quant))
#ES.train = ES_for_quant[train.ids,]
#ES.test = ES_for_quant[-train.ids,]

set.seed(186)
train.ids2 = sample(nrow(Cat_trimmed), .7*nrow(Cat_trimmed))
train_names = Cat_trimmed[train.ids2,] %>%
  dplyr::select(Name) %>%
  mutate(is_train = 1)

ES_for_quant2 <- ES_for_quant %>%
  left_join(train_names, by = c("Compound.Name"= "Name"))

ES_for_quant2$is_train[is.na(ES_for_quant2$is_train)] <- 0

ES.train <- ES_for_quant2 %>%
  filter(is_train == 1) %>%
  dplyr::select(-is_train)

ES.test <- ES_for_quant2 %>%
  filter(is_train == 0) %>%
  dplyr::select(-is_train)

ES.train.withname <- ES.train
ES.test.withname <- ES.test

ES.train_lm1 <- ES.train %>%
  dplyr::select(-Compound.Name) %>%
  dplyr::select(-r_rundate)

ES.test_lm1 <- ES.test %>%
  dplyr::select(-Compound.Name) %>%
  dplyr::select(-r_rundate)

linmod1 <- lm(Cal_amt_ng ~., data = ES.train_lm1)

summary(linmod1)
```

```

## 
## Call:
## lm(formula = Cal_amt_ng ~ ., data = ES.train_lm1)
## 
## Residuals:
##    Min      1Q  Median      3Q     Max 
## -39.061 -11.969   -4.669    6.796   84.332 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) -2.656e+03 2.380e+03 -1.116 0.264697  
## Library.RI   -5.942e-03 1.227e-03 -4.842 1.39e-06 *** 
## Retention.II..sec. 5.263e+00 2.833e+00  1.857 0.063418 .  
## Run_date_num  6.146e-02 5.479e-02  1.122 0.262161  
## Vol_norm_GLM  3.842e-03 7.266e-04  5.288 1.38e-07 *** 
## loss_1TRUE    -2.171e+00 1.686e+00 -1.287 0.198141  
## loss_12TRUE   8.446e-02 2.777e+00  0.030 0.975745  
## loss_13TRUE   9.899e-01 2.261e+00  0.438 0.661552  
## loss_14TRUE   -2.620e+00 1.566e+00 -1.673 0.094505 .  
## loss_15TRUE   -2.728e+00 2.031e+00 -1.343 0.179357  
## loss_16TRUE   6.041e+00 1.731e+00  3.489 0.000496 *** 
## loss_18TRUE   -9.282e+00 2.742e+00 -3.385 0.000726 *** 
## loss_2TRUE    -3.964e+00 1.800e+00 -2.202 0.027765 *  
## loss_22TRUE   8.244e+00 2.927e+00  2.817 0.004905 ** 
## loss_26TRUE   1.177e+01 2.803e+00  4.200 2.79e-05 *** 
## loss_28TRUE   -5.561e-01 2.135e+00 -0.261 0.794479  
## loss_29TRUE   8.875e-02 2.269e+00  0.039 0.968799  
## loss_30TRUE   5.747e+00 2.316e+00  2.481 0.013172 *  
## loss_31TRUE   -5.404e+00 2.499e+00 -2.163 0.030685 *  
## loss_42TRUE   5.451e+00 2.235e+00  2.439 0.014834 *  
## loss_44TRUE   -2.609e+00 1.978e+00 -1.319 0.187230  
## loss_54TRUE   -3.163e+00 2.434e+00 -1.300 0.193889  
## loss_56TRUE   5.289e+00 1.951e+00  2.710 0.006779 ** 
## loss_70TRUE   -7.893e+00 2.134e+00 -3.698 0.000224 *** 
## loss_74TRUE   3.121e+00 2.418e+00  1.291 0.196870  
## mz_105TRUE   1.585e+00 2.899e+00  0.547 0.584636  
## mz_113TRUE   1.232e+01 3.006e+00  4.100 4.31e-05 *** 
## mz_117TRUE   -9.954e+00 2.367e+00 -4.205 2.73e-05 *** 
## mz_129TRUE   -3.095e-01 2.338e+00 -0.132 0.894684  
## mz_147TRUE   9.194e+00 2.614e+00  3.517 0.000446 *** 
## mz_41TRUE    3.722e+00 3.320e+00  1.121 0.262397  
## mz_43TRUE    5.568e+00 2.065e+00  2.696 0.007077 ** 
## mz_45TRUE    -2.019e+00 1.977e+00 -1.021 0.307199  
## mz_55TRUE    7.758e+00 2.132e+00  3.639 0.000281 *** 
## mz_57TRUE    -2.694e+00 2.363e+00 -1.140 0.254255  
## mz_69TRUE    8.852e-01 2.223e+00  0.398 0.690496  
## mz_71TRUE    1.969e+01 3.561e+00  5.530 3.64e-08 *** 
## mz_73TRUE    7.862e+00 2.122e+00  3.705 0.000218 *** 
## mz_75TRUE    2.488e+00 2.008e+00  1.240 0.215311  
## mz_81TRUE    -3.682e+00 3.067e+00 -1.201 0.230006  
## mz_83TRUE    2.653e+00 1.943e+00  1.366 0.172123  
## mz_85TRUE    -3.046e+01 6.392e+00 -4.765 2.03e-06 *** 
## mz_91TRUE    -2.802e+00 2.915e+00 -0.961 0.336526  
## mz_93TRUE    5.718e+00 3.103e+00  1.843 0.065552 .  
## mz_99TRUE    -1.053e+01 4.648e+00 -2.266 0.023556 *  
## --- 
## Signif. codes:  0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 
## 
## Residual standard error: 18.37 on 1893 degrees of freedom
## Multiple R-squared:  0.1676, Adjusted R-squared:  0.1483 
## F-statistic: 8.663 on 44 and 1893 DF, p-value: < 2.2e-16

```

```

ES.train.lm <- ES.train_lml %>%
  dplyr::select(-c(Retention.II..sec., loss_22, loss_29, loss_44, mz_105, mz_55))

ES.test.lm <- ES.test_lml %>%
  dplyr::select(-c(Retention.II..sec., loss_22, loss_29, loss_44, mz_105, mz_55))

linmod2 <- lm(Cal_amt_ng ~ ., data = ES.train.lm)

summary(linmod2)

```

```

## 
## Call:
## lm(formula = Cal_amt_ng ~ ., data = ES.train.lm)
## 
## Residuals:
##    Min      1Q  Median      3Q     Max 
## -39.098 -12.130   -5.101    7.048   85.326 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) -2.698e+03 2.387e+03 -1.130 0.258496  
## Library.RI   -7.236e-03 1.136e-03 -6.371 2.36e-10 *** 
## Run_date_num 6.269e-02 5.496e-02  1.141 0.254154  
## Vol_norm_GLM 4.046e-03 6.789e-04  5.961 2.99e-09 *** 
## loss_1TRUE   -3.167e+00 1.626e+00 -1.947 0.051635 .  
## loss_12TRUE   -3.382e+00 2.021e+00 -1.673 0.094427 .  
## loss_13TRUE   2.414e+00 2.103e+00  1.148 0.250986  
## loss_14TRUE   -1.949e+00 1.457e+00 -1.338 0.181039  
## loss_15TRUE   -3.888e+00 1.955e+00 -1.989 0.046892 *  
## loss_16TRUE   5.434e+00 1.662e+00  3.271 0.001093 ** 
## loss_18TRUE   -5.390e+00 2.454e+00 -2.196 0.028213 *  
## loss_2TRUE    -2.728e+00 1.662e+00 -1.642 0.100830  
## loss_26TRUE   6.862e+00 2.368e+00  2.898 0.003802 ** 
## loss_28TRUE   3.607e-01 1.703e+00  0.212 0.832325  
## loss_30TRUE   7.632e-01 1.821e+00  0.419 0.675226  
## loss_31TRUE   -1.719e+00 2.243e+00 -0.766 0.443692  
## loss_42TRUE   3.100e+00 1.870e+00  1.658 0.097513 .  
## loss_54TRUE   -4.215e-01 2.180e+00 -0.193 0.846719  
## loss_56TRUE   4.185e+00 1.707e+00  2.452 0.014315 *  
## loss_70TRUE   -8.410e+00 2.025e+00 -4.154 3.41e-05 *** 
## loss_74TRUE   2.918e+00 2.088e+00  1.398 0.162375  
## mz_113TRUE   1.298e+01 2.928e+00  4.431 9.91e-06 *** 
## mz_117TRUE   -8.625e+00 2.059e+00 -4.190 2.92e-05 *** 
## mz_129TRUE   1.080e+00 2.017e+00  0.536 0.592277  
## mz_147TRUE   6.726e+00 2.135e+00  3.150 0.001661 ** 
## mz_41TRUE    2.021e+00 3.041e+00  0.664 0.506460  
## mz_43TRUE    7.568e+00 1.927e+00  3.927 8.92e-05 *** 
## mz_45TRUE    -3.138e+00 1.950e+00 -1.609 0.107746  
## mz_57TRUE    -2.821e+00 2.061e+00 -1.369 0.171083  
## mz_69TRUE    1.421e+00 1.933e+00  0.735 0.462505  
## mz_71TRUE    1.808e+01 3.433e+00  5.267 1.55e-07 *** 
## mz_73TRUE    8.175e+00 1.900e+00  4.303 1.77e-05 *** 
## mz_75TRUE    2.746e+00 1.711e+00  1.605 0.108682  
## mz_81TRUE    -6.813e-01 2.352e+00 -0.290 0.772082  
## mz_83TRUE    3.942e+00 1.796e+00  2.194 0.028339 *  
## mz_85TRUE    -2.180e+01 5.878e+00 -3.709 0.000214 *** 
## mz_91TRUE    -1.733e+00 2.502e+00 -0.693 0.488623  
## mz_93TRUE    5.239e+00 2.776e+00  1.888 0.059217 .  
## mz_99TRUE    -1.379e+01 4.402e+00 -3.132 0.001761 ** 
## --- 
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 
## 
## Residual standard error: 18.43 on 1899 degrees of freedom 
## Multiple R-squared:  0.1596, Adjusted R-squared:  0.1428 
## F-statistic: 9.489 on 38 and 1899 DF, p-value: < 2.2e-16

```

```

predicted_lm1 <- predict(linmod1, ES.test_lm1)

predicted_lm2 <- predict(linmod2, ES.test_lm1)

osr_lm1 <- OSR2(predicted_lm1, ES.train_lm1$Cal_amt_ng, ES.test_lm1$Cal_amt_ng)

mae_lm1 <- MAE(obs = ES.test_lm1$Cal_amt_ng, pred = predicted_lm1)

osr_lm2 <- OSR2(predicted_lm2, ES.train_lm1$Cal_amt_ng, ES.test_lm1$Cal_amt_ng)

mae_lm2 <- MAE(obs = ES.test_lm1$Cal_amt_ng, pred = predicted_lm2)

```

The OSR2 of the original model is -0.5319696 and the MAE is 18.7784344. With the least significant factors removed, the OSR2 is -0.3919011 and the MAE is 18.0444347. Both are quite terrible- clearly linear modeling is not the best approach for this effort.

CART modeling

```

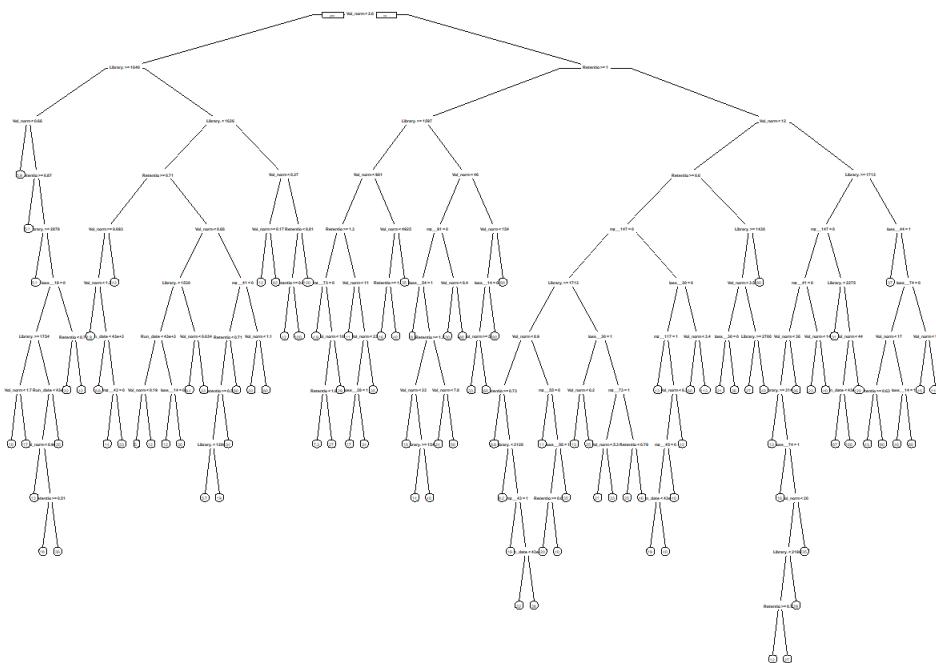
cart.mod <- rpart(Cal_amt_ng ~., data = ES.train_lm1, method = "anova", cp = 0.001, minsplit = 5)

#method anova does regression not classification
## fill in the method yourselves. Might require some online searches!!

prp(cart.mod)

```

```
## Warning: labs do not fit even at cex 0.15, there may be some overplotting
```



```

CartPredictions <- predict(cart.mod, newdata=ES.test_lm1)
SSE = sum((ES.test_lm1$Cal_amt_ng - CartPredictions)^2)
SST = sum((ES.test_lm1$Cal_amt_ng - mean(ES.train_lm1$Cal_amt_ng))^2)
OSR2_cart = 1 - SSE/SST # is this model useful at all?
OSR2_cart

```

```
## [1] -0.1675092
```

```
mae_cart <- MAE(obs = ES.test_lm1$Cal_amt_ng, pred = CartPredictions)
```

While there has been moderate improvement with a OSR2 of -.16 and a MAE of 15.5, the CART model also does a poor job of predicting response.

```

set.seed(144)
mod.rf1 <- randomForest(Cal_amt_ng ~ .-Compound.Name, data = ES.train, mtry = 34, nodesize = 5, ntree = 1000
)

pred.rf <- predict(mod.rf1, newdata = ES.test)

RFPredictions <- predict(mod.rf1, newdata=ES.test)

SSE = sum((ES.test$Cal_amt_ng - RFPredictions)^2)
SST = sum((ES.test$Cal_amt_ng - mean(ES.train$Cal_amt_ng))^2)
OSR2_rf = 1 - SSE/SST # is this model useful at all?
OSR2_rf

```

```
## [1] 0.1569098
```

```
mae_rf <- MAE(obs = ES.test$Cal_amt_ng, pred = RFPredictions)
```

Cross validation has determined that the appropriate interaction depth is 34. With this interaction depth, the random forest model has a MAE of 11.7 and an OSR of .16.

```

set.seed(99)
train.rf <- train(Cal_amt_ng ~ . -Compound.Name -r_rundate,
                   data = ES.train,
                   method = "rf",
                   tuneGrid = data.frame(mtry=28:35),
                   trControl = trainControl(method="cv", number=5, verboseIter = TRUE),
                   metric = "RMSE")
# RMSE or Rsquared doesn't matter actually -- both will be generated for regression problems
train.rf$results
train.rf
best.rf <- train.rf$finalModel
pred.best.rf <- predict(best.rf, newdata = ES.test) # can use same model matrix

SSE = sum((ES.test$Cal_amt_ng - pred.best.rf)^2)
SST = sum((ES.test$Cal_amt_ng - mean(ES.test$Cal_amt_ng))^2)
OSR2_rf_cv = 1 - SSE/SST # is this model useful at all?
OSR2_rf_cv

```

```

ES.train.boost1 <- ES.train%>%
  mutate(loss_1 = as.factor(loss_1)) %>%
  mutate(loss_12 = as.factor(loss_12)) %>%
  mutate(loss_14 = as.factor(loss_14)) %>%
  mutate(loss_13 = as.factor(loss_13)) %>%
  mutate(loss_15 = as.factor(loss_15)) %>%
  mutate(loss_16 = as.factor(loss_16)) %>%
  mutate(loss_18 = as.factor(loss_18)) %>%
  mutate(loss_2 = as.factor(loss_2)) %>%
  mutate(loss_22 = as.factor(loss_22)) %>%
  mutate(loss_26 = as.factor(loss_26)) %>%
  mutate(loss_28 = as.factor(loss_28)) %>%
  mutate(loss_29 = as.factor(loss_29)) %>%
  mutate(loss_30 = as.factor(loss_30)) %>%
  mutate(loss_31 = as.factor(loss_31)) %>%
  mutate(loss_42 = as.factor(loss_42)) %>%
  mutate(loss_44 = as.factor(loss_44)) %>%
  mutate(loss_54 = as.factor(loss_54)) %>%
  mutate(loss_56 = as.factor(loss_56)) %>%
  mutate(loss_70 = as.factor(loss_70)) %>%
  mutate(loss_74 = as.factor(loss_74)) %>%
  mutate(mz_105 = as.factor(mz_105)) %>%
  mutate(mz_113 = as.factor(mz_113)) %>%
  mutate(mz_117 = as.factor(mz_117)) %>%
  mutate(mz_129 = as.factor(mz_129)) %>%
  mutate(mz_147 = as.factor(mz_147)) %>%
  mutate(mz_41 = as.factor(mz_41)) %>%
  mutate(mz_43 = as.factor(mz_43)) %>%
  mutate(mz_45 = as.factor(mz_45)) %>%

```

```

mutate(mz_55 = as.factor(mz_55)) %>%
mutate(mz_57 = as.factor(mz_57)) %>%
  mutate(mz_69 = as.factor(mz_69)) %>%
mutate(mz_71 = as.factor(mz_71)) %>%
mutate(mz_73 = as.factor(mz_73)) %>%
mutate(mz_75 = as.factor(mz_75)) %>%
mutate(mz_85 = as.factor(mz_85)) %>%
  mutate(mz_83 = as.factor(mz_83)) %>%
mutate(mz_81 = as.factor(mz_81)) %>%
mutate(mz_91 = as.factor(mz_91)) %>%
mutate(mz_93 = as.factor(mz_93)) %>%
  mutate(mz_99 = as.factor(mz_99))

ES.test.boost1 <- ES.test%>%
  mutate(loss_1 = as.factor(loss_1)) %>%
  mutate(loss_12 = as.factor(loss_12)) %>%
  mutate(loss_14 = as.factor(loss_14)) %>%
  mutate(loss_13 = as.factor(loss_13)) %>%
mutate(loss_15 = as.factor(loss_15)) %>%
  mutate(loss_16 = as.factor(loss_16)) %>%
  mutate(loss_18 = as.factor(loss_18)) %>%
mutate(loss_2 = as.factor(loss_2)) %>%
  mutate(loss_22 = as.factor(loss_22)) %>%
  mutate(loss_26 = as.factor(loss_26)) %>%
  mutate(loss_28 = as.factor(loss_28)) %>%
  mutate(loss_29 = as.factor(loss_29)) %>%
  mutate(loss_30 = as.factor(loss_30)) %>%
  mutate(loss_31 = as.factor(loss_31)) %>%
  mutate(loss_42 = as.factor(loss_42)) %>%
  mutate(loss_44 = as.factor(loss_44)) %>%
  mutate(loss_54 = as.factor(loss_54)) %>%
  mutate(loss_56 = as.factor(loss_56)) %>%
  mutate(loss_70 = as.factor(loss_70)) %>%
  mutate(loss_74 = as.factor(loss_74)) %>%
  mutate(mz_105 = as.factor(mz_105)) %>%
  mutate(mz_113 = as.factor(mz_113)) %>%
  mutate(mz_117 = as.factor(mz_117)) %>%
  mutate(mz_129 = as.factor(mz_129)) %>%
  mutate(mz_147 = as.factor(mz_147)) %>%
  mutate(mz_41 = as.factor(mz_41)) %>%
  mutate(mz_43 = as.factor(mz_43)) %>%
  mutate(mz_45 = as.factor(mz_45)) %>%
  mutate(mz_55 = as.factor(mz_55)) %>%
  mutate(mz_57 = as.factor(mz_57)) %>%
  mutate(mz_69 = as.factor(mz_69)) %>%
  mutate(mz_71 = as.factor(mz_71)) %>%
  mutate(mz_73 = as.factor(mz_73)) %>%
  mutate(mz_75 = as.factor(mz_75)) %>%
  mutate(mz_85 = as.factor(mz_85)) %>%
  mutate(mz_83 = as.factor(mz_83)) %>%
  mutate(mz_81 = as.factor(mz_81)) %>%
  mutate(mz_91 = as.factor(mz_91)) %>%
  mutate(mz_93 = as.factor(mz_93)) %>%
  mutate(mz_99 = as.factor(mz_99))

set.seed(144)
mod.boost <- gbm(Cal_amt_ng ~ . -rundate -Compound.Name,
                  data = ES.train.boost1,
                  distribution = "gaussian",
                  n.trees = 1000,
                  interaction.depth = 5)

set.seed(144)
pred.boost <- predict(mod.boost, newdata = ES.test.boost1, n.trees=1000)

SSE = sum((ES.test.boost1$Cal_amt_ng - pred.boost)^2)
SST = sum((ES.test.boost1$Cal_amt_ng - mean(ES.train.boost1$Cal_amt_ng))^2)
OSR2_boost = 1 - SSE/SST # is this model useful at all?
#OSR2_boost

```

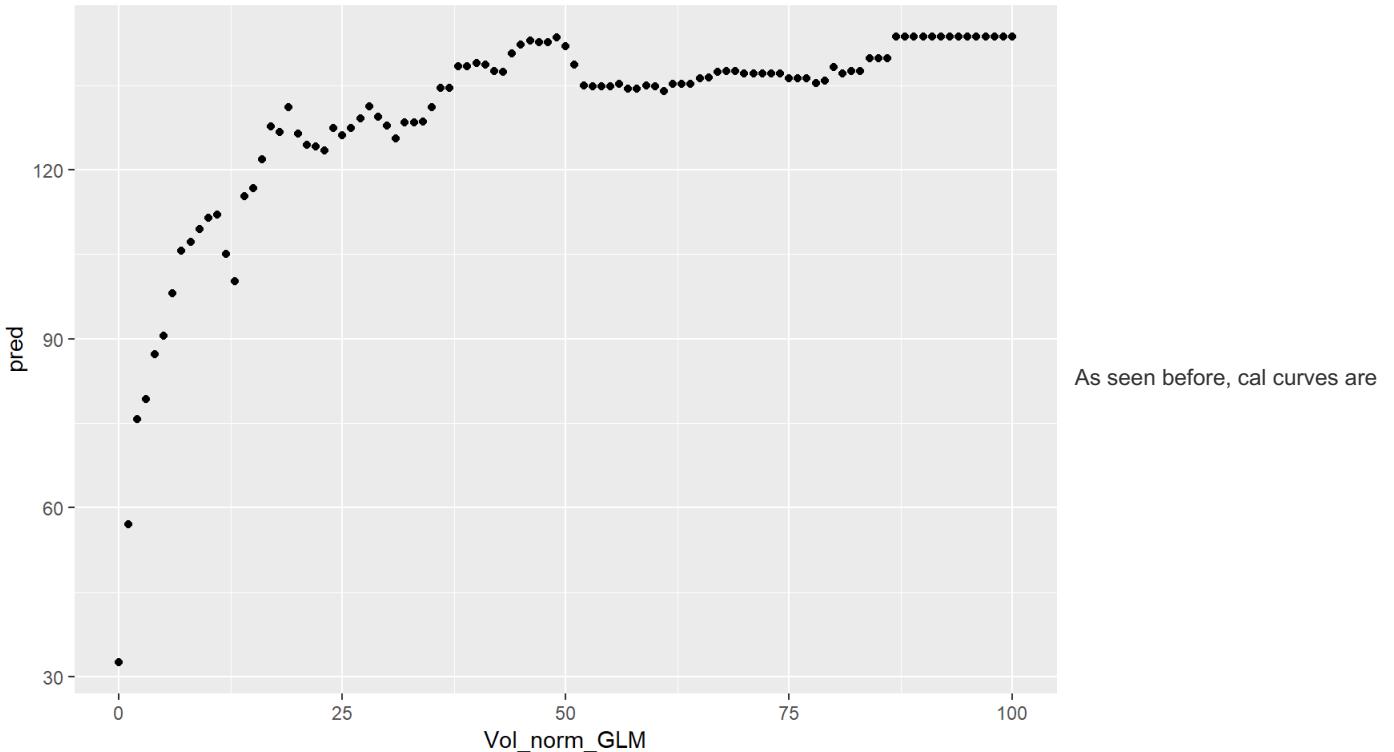
```
mae_boosted <- MAE(obs = ES.test.boost1$Cal_amt_ng, pred = pred.boost)
```

The boosted model performs slightly better than the random forest model with a OSR2 of .17. However the MAE is slightly worse at 12.6

Slope Prediction

Predicting slopes by taking the slopes from predicted cal points

```
rep.row<-function(x,n){  
  matrix(rep(x,each=n),nrow=n)  
}  
  
ind <- 2  
t_vol <- seq(from = 0, to = 100, by = 1)  
t_len <- length(t_vol)  
  
boosted_vis <- ES.train.boost1[ind,]  
  
boosted_vis.1 <- boosted_vis[rep(1, t_len),]  
  
boosted_vis.1$t_vol <- t_vol  
  
boosted_vis.2 <- boosted_vis.1 %>%  
  mutate(Vol_norm_GLM = t_vol)  
  
boosted.vis.pred <- predict(mod.boost, newdata = boosted_vis.2, n.trees=1000)  
  
boosted_vis.2$pred <- boosted.vis.pred  
  
boosted_vis.2 %>%  
  ggplot(aes(x = Vol_norm_GLM, y = pred)) +  
  geom_point()
```



linear and the boosted model predictions are decidedly not linear. This means that interpolation using this method is an unwise solution and a different approach should be attempted.

Now trying something to generate the slope through (and r squared) for each individual cal curve

```

cpnds <- cal_pts$Compound.Name[1:135]
dates <- bt_sum$r_rundate

cal_curves <- expand.grid(cpnds, dates, KEEP.OUT.ATTRS = TRUE, stringsAsFactors = FALSE)

cal_curves <- distinct(cal_curves)

names(cal_curves) <- c("cpnd", "dates")

cal_curves$slope <- rep(-999, nrow(cal_curves))
cal_curves$r2 <- rep(-999, nrow(cal_curves))

for(i in 1:nrow(cal_curves)) {

  compound = cal_curves$cpnd[i]
  day = cal_curves$dates[i]

  subset <- ES_cleaned1 %>%
    filter(Compound.Name == compound) %>%
    filter(r_rundate == day)

  if(nrow(subset)<3) {
    next
  }

  lm_temp <- lm(Cal_amt_ng ~ 0 + Vol_norm_GLM, data = subset)

  #summary(lm_temp)

  slope <- as.numeric(coefficients(lm_temp))

  cal_curves$slope[i] <- slope

  r2 <- summary(lm_temp)$r.squared

  cal_curves$r2[i] <- r2

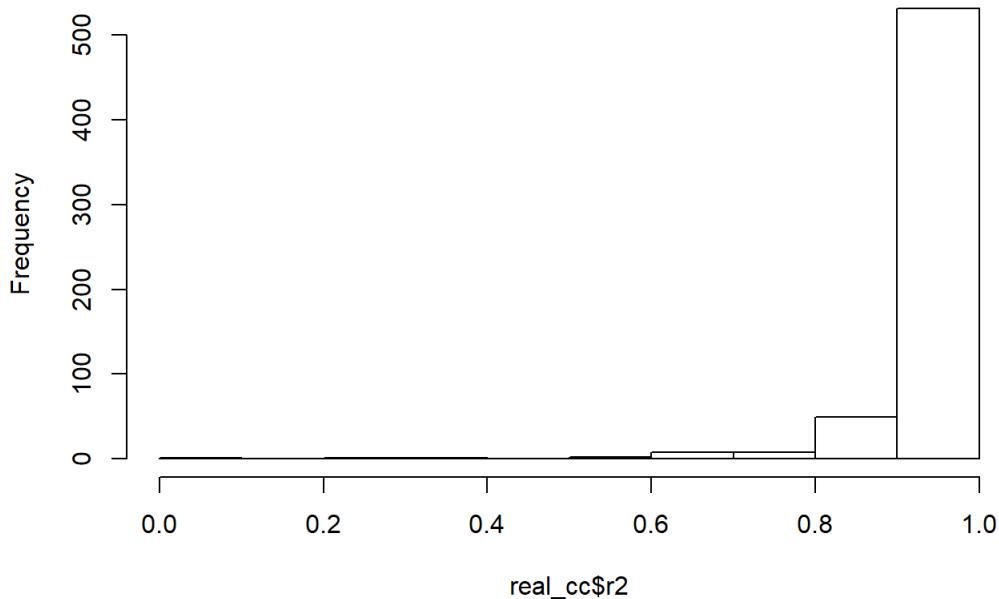
}

real_cc <- cal_curves %>%
  filter(r2 > 0)

hist(real_cc$r2)

```

Histogram of real_cc\$r2



```
# joining on the important ms info
left_joinF <- function(x, y, fill = FALSE){
  z <- left_join(x, y)
  tmp <- setdiff(names(z), names(x))
  z <- replace_na(z, setNames(as.list(rep(fill, length(tmp)))), tmp))
  z
}

colnames(wide_mz)[1] <- "cpnd"
colnames(wide_losses)[1] <- "cpnd"

names_positions <- cal_pts %>%
  dplyr::select(c("Compound.Name", "Library.RI", "Retention.II.sec."))

cc_slopemod <- real_cc %>%
  left_joinF(wide_losses) %>%
  left_joinF(wide_mz) %>%
  left_join(names_positions, by = c("cpnd"= "Compound.Name"))
```

```
## Joining, by = "cpnd"
## Joining, by = "cpnd"
```

```

ES_slopes_boosted <- ES.test.boost1

ES_slopes_boosted$boostpred <- pred.boost

for(i in 1:nrow(cal_curves)) {

  compound = cal_curves$cpnd[i]
  day = cal_curves$dates[i]

  subset <- ES_slopes_boosted %>%
    filter(Compound.Name == compound) %>%
    filter(r_rundate == day)

  if(nrow(subset)<3) {
    cal_curves$slope_fromboostedpred[i] <- -99
    next
  }

  lm_temp <- lm(boostpred ~ 0 + Vol_norm_GLM, data = subset)

  #summary(lm_temp)

  slope <- as.numeric(coefficients(lm_temp))

  cal_curves$slope_fromboostedpred[i] <- slope

  r2 <- summary(lm_temp)$r.squared

  #cal_curves$r2_fromboostedpred[i] <- r2

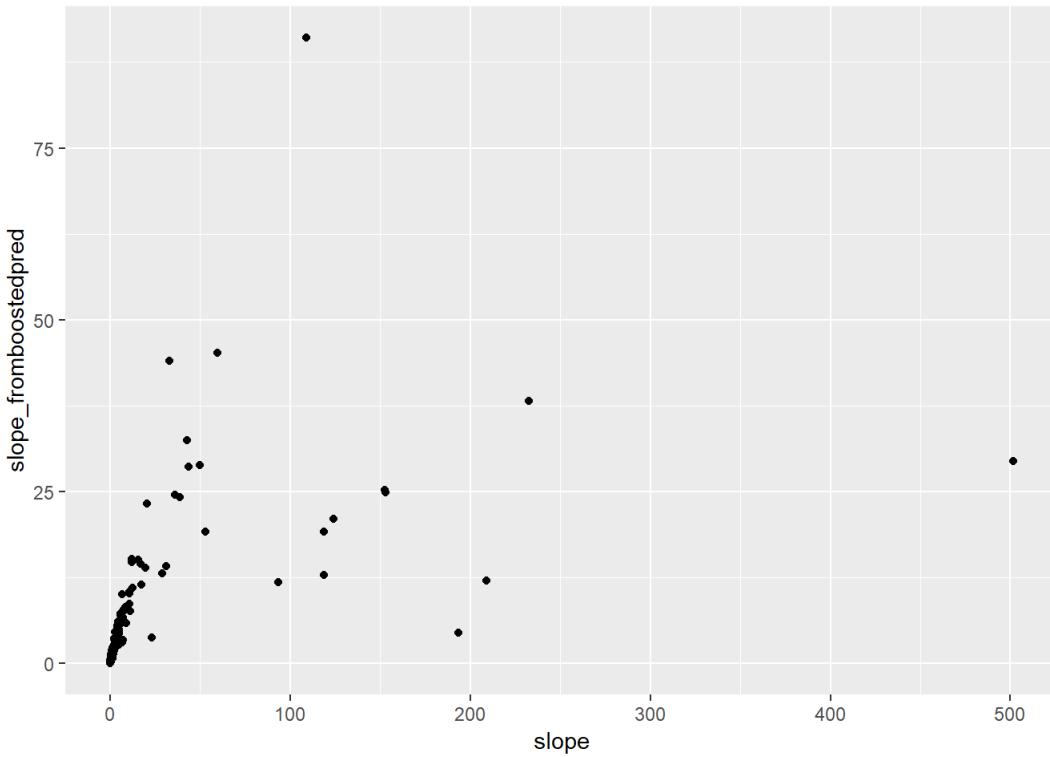
}

boosted_slopes <- cal_curves %>%
  filter(slope_fromboostedpred >0) %>%
  filter(slope > 0)

RMSE_boostedslopes <- RMSE(boosted_slopes$slope_fromboostedpred, boosted_slopes$slope)
OSR_sorta <- OSR2(boosted_slopes$slope_fromboostedpred, cal_curves$slope, boosted_slopes$slope)

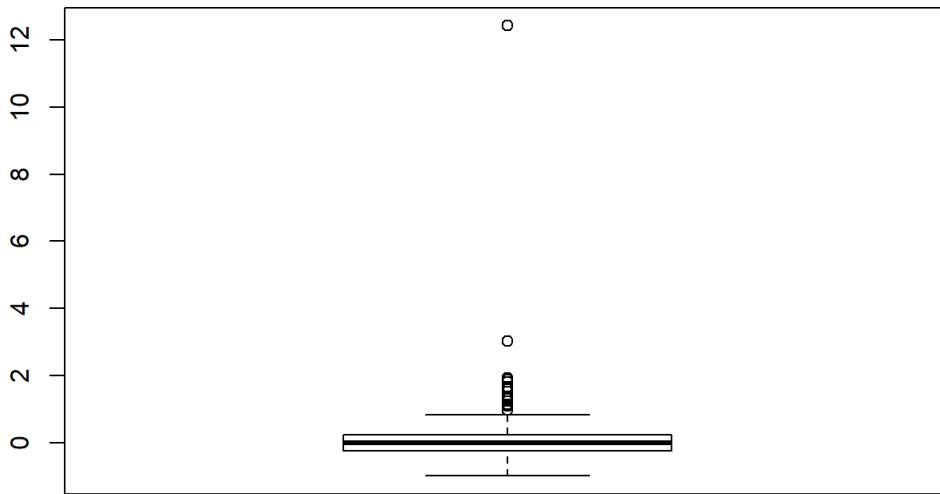
boosted_slopes %>%
  ggplot(aes(x = slope, y = slope_fromboostedpred)) +
  geom_point()

```



```
boosted_slopes <- boosted_slopes %>%
  mutate(slopedif_pct = (slope_fromboostedpred - slope)/slope) %>%
  mutate(as_slopedifpct = abs(slopedif_pct))

boxplot(boosted_slopes$slopedif_pct)
```



```
summary(boosted_slopes$as_slopedifpct)
```

	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
##	0.000718	0.103954	0.243586	0.478559	0.544618	12.411784

This actually gives me some hope- there appear to be a few outliers which are skewing what is otherwise a not terrible approximation.

```

# not working investigate later

set.seed(88)
train.ids = sample(nrow(cc_slopemod), .7*nrow(cc_slopemod))
slopemod_temp <- cc_slopemod
#%>% dplyr::select(-cpnd)

slopemod.train = slopemod_temp[train.ids,]
slopemod.test = slopemod_temp[-train.ids,]

set.seed(99)



```

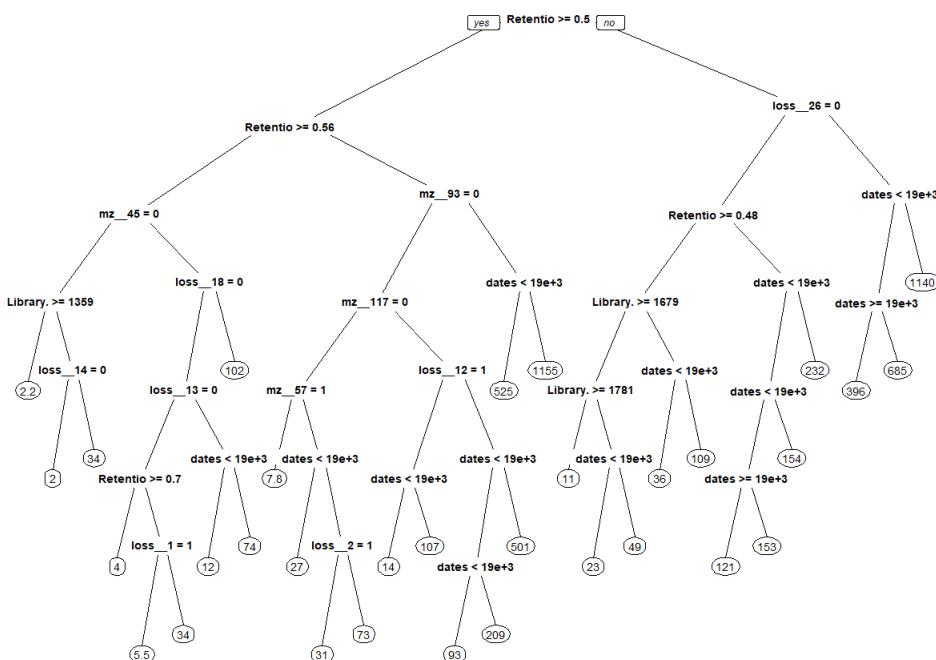
```

cart.mod.s <- rpart(slope ~., data = slopemod.train, method = "anova" , cp = 0.0001, minsplit = 2)

#method anova does regression not classification
## fill in the method yourselves. Might require some online searches!!

prp(cart.mod.s)

```



```

CartPredictions_s <- predict(cart.mod.s, newdata=slopemod.test)
SSE = sum((slopemod.test$slope - CartPredictions_s)^2)
SST = sum((slopemod.test$slope - mean(slopemod.train$slope))^2)
OSR2_cart.s = 1 - SSE/SST # is this model useful at all?
OSR2_cart.s

```

```

## [1] -1.312782

```

```
mae_cart.s <- MAE(pred = CartPredictions_s, obs = slopemod.test$slope)
```

```
train.rf <- train(slope ~ .,
                   data = slopemod.train,
                   method = "rf",
                   tuneGrid = data.frame(mtry=1:20),
                   trControl = trainControl(method="cv", number=5, verboseIter = TRUE),
                   metric = "RMSE")
```

```
## + Fold1: mtry= 1
## - Fold1: mtry= 1
## + Fold1: mtry= 2
## - Fold1: mtry= 2
## + Fold1: mtry= 3
## - Fold1: mtry= 3
## + Fold1: mtry= 4
## - Fold1: mtry= 4
## + Fold1: mtry= 5
## - Fold1: mtry= 5
## + Fold1: mtry= 6
## - Fold1: mtry= 6
## + Fold1: mtry= 7
## - Fold1: mtry= 7
## + Fold1: mtry= 8
## - Fold1: mtry= 8
## + Fold1: mtry= 9
## - Fold1: mtry= 9
## + Fold1: mtry=10
## - Fold1: mtry=10
## + Fold1: mtry=11
## - Fold1: mtry=11
## + Fold1: mtry=12
## - Fold1: mtry=12
## + Fold1: mtry=13
## - Fold1: mtry=13
## + Fold1: mtry=14
## - Fold1: mtry=14
## + Fold1: mtry=15
## - Fold1: mtry=15
## + Fold1: mtry=16
## - Fold1: mtry=16
## + Fold1: mtry=17
## - Fold1: mtry=17
## + Fold1: mtry=18
## - Fold1: mtry=18
## + Fold1: mtry=19
## - Fold1: mtry=19
## + Fold1: mtry=20
## - Fold1: mtry=20
## + Fold2: mtry= 1
## - Fold2: mtry= 1
## + Fold2: mtry= 2
## - Fold2: mtry= 2
## + Fold2: mtry= 3
## - Fold2: mtry= 3
## + Fold2: mtry= 4
## - Fold2: mtry= 4
## + Fold2: mtry= 5
## - Fold2: mtry= 5
## + Fold2: mtry= 6
## - Fold2: mtry= 6
## + Fold2: mtry= 7
## - Fold2: mtry= 7
## + Fold2: mtry= 8
## - Fold2: mtry= 8
## + Fold2: mtry= 9
## - Fold2: mtry= 9
## + Fold2: mtry=10
## - Fold2: mtry=10
## + Fold2: mtry=11
## - Fold2: mtry=11
```

```
-  
## + Fold2: mtry=12  
## - Fold2: mtry=12  
## + Fold2: mtry=13  
## - Fold2: mtry=13  
## + Fold2: mtry=14  
## - Fold2: mtry=14  
## + Fold2: mtry=15  
## - Fold2: mtry=15  
## + Fold2: mtry=16  
## - Fold2: mtry=16  
## + Fold2: mtry=17  
## - Fold2: mtry=17  
## + Fold2: mtry=18  
## - Fold2: mtry=18  
## + Fold2: mtry=19  
## - Fold2: mtry=19  
## + Fold2: mtry=20  
## - Fold2: mtry=20  
## + Fold3: mtry= 1  
## - Fold3: mtry= 1  
## + Fold3: mtry= 2  
## - Fold3: mtry= 2  
## + Fold3: mtry= 3  
## - Fold3: mtry= 3  
## + Fold3: mtry= 4  
## - Fold3: mtry= 4  
## + Fold3: mtry= 5  
## - Fold3: mtry= 5  
## + Fold3: mtry= 6  
## - Fold3: mtry= 6  
## + Fold3: mtry= 7  
## - Fold3: mtry= 7  
## + Fold3: mtry= 8  
## - Fold3: mtry= 8  
## + Fold3: mtry= 9  
## - Fold3: mtry= 9  
## + Fold3: mtry=10  
## - Fold3: mtry=10  
## + Fold3: mtry=11  
## - Fold3: mtry=11  
## + Fold3: mtry=12  
## - Fold3: mtry=12  
## + Fold3: mtry=13  
## - Fold3: mtry=13  
## + Fold3: mtry=14  
## - Fold3: mtry=14  
## + Fold3: mtry=15  
## - Fold3: mtry=15  
## + Fold3: mtry=16  
## - Fold3: mtry=16  
## + Fold3: mtry=17  
## - Fold3: mtry=17  
## + Fold3: mtry=18  
## - Fold3: mtry=18  
## + Fold3: mtry=19  
## - Fold3: mtry=19  
## + Fold3: mtry=20  
## - Fold3: mtry=20  
## + Fold4: mtry= 1  
## - Fold4: mtry= 1  
## + Fold4: mtry= 2  
## - Fold4: mtry= 2  
## + Fold4: mtry= 3  
## - Fold4: mtry= 3  
## + Fold4: mtry= 4  
## - Fold4: mtry= 4  
## + Fold4: mtry= 5  
## - Fold4: mtry= 5  
## + Fold4: mtry= 6  
## - Fold4: mtry= 6  
## + Fold4: mtry= 7  
## - Fold4: mtry= 7  
## + Fold4: mtry= 8
```

```
## + Fold4: mtry= 0
## - Fold4: mtry= 8
## + Fold4: mtry= 9
## - Fold4: mtry= 9
## + Fold4: mtry=10
## - Fold4: mtry=10
## + Fold4: mtry=11
## - Fold4: mtry=11
## + Fold4: mtry=12
## - Fold4: mtry=12
## + Fold4: mtry=13
## - Fold4: mtry=13
## + Fold4: mtry=14
## - Fold4: mtry=14
## + Fold4: mtry=15
## - Fold4: mtry=15
## + Fold4: mtry=16
## - Fold4: mtry=16
## + Fold4: mtry=17
## - Fold4: mtry=17
## + Fold4: mtry=18
## - Fold4: mtry=18
## + Fold4: mtry=19
## - Fold4: mtry=19
## + Fold4: mtry=20
## - Fold4: mtry=20
## + Fold5: mtry= 1
## - Fold5: mtry= 1
## + Fold5: mtry= 2
## - Fold5: mtry= 2
## + Fold5: mtry= 3
## - Fold5: mtry= 3
## + Fold5: mtry= 4
## - Fold5: mtry= 4
## + Fold5: mtry= 5
## - Fold5: mtry= 5
## + Fold5: mtry= 6
## - Fold5: mtry= 6
## + Fold5: mtry= 7
## - Fold5: mtry= 7
## + Fold5: mtry= 8
## - Fold5: mtry= 8
## + Fold5: mtry= 9
## - Fold5: mtry= 9
## + Fold5: mtry=10
## - Fold5: mtry=10
## + Fold5: mtry=11
## - Fold5: mtry=11
## + Fold5: mtry=12
## - Fold5: mtry=12
## + Fold5: mtry=13
## - Fold5: mtry=13
## + Fold5: mtry=13
## - Fold5: mtry=14
## + Fold5: mtry=14
## - Fold5: mtry=14
## + Fold5: mtry=15
## - Fold5: mtry=15
## + Fold5: mtry=15
## - Fold5: mtry=16
## + Fold5: mtry=16
## - Fold5: mtry=16
## + Fold5: mtry=17
## - Fold5: mtry=17
## + Fold5: mtry=17
## - Fold5: mtry=18
## + Fold5: mtry=18
## - Fold5: mtry=18
## + Fold5: mtry=19
## - Fold5: mtry=19
## + Fold5: mtry=19
## - Fold5: mtry=20
## + Fold5: mtry=20
## - Fold5: mtry=20
## Aggregating results
## Selecting tuning parameters
## Fitting mtry = 18 on full training set
```

```
# RMSE or Rsquared doesn't matter actually -- both will be generated for regression problems
train.rf$results
```

```
##   mtry      RMSE  Rsquared      MAE    RMSESD RsquaredSD    MAESD
## 1    1 66.76301 0.5869442 22.91298 54.81006  0.2340349 8.838175
## 2    2 55.11667 0.6734142 17.35421 47.81411  0.1886899 7.805179
## 3    3 49.52507 0.7017018 14.84608 44.11329  0.1786014 7.585761
## 4    4 47.65369 0.7351394 13.55577 42.99463  0.1573161 6.832266
## 5    5 46.12185 0.7309652 13.07311 40.88219  0.1443990 6.937045
## 6    6 44.74455 0.7306419 12.62889 39.90346  0.1640685 6.882133
## 7    7 43.80387 0.7447494 12.13825 39.28737  0.1388545 6.819546
## 8    8 43.95158 0.7431179 12.05859 38.56812  0.1396547 6.668695
## 9    9 43.30996 0.7534858 11.71387 38.69197  0.1353892 6.602108
## 10  10 41.77452 0.7591158 11.38698 37.36489  0.1387055 6.528008
## 11  11 41.57701 0.7597915 11.32502 37.00833  0.1420113 6.306228
## 12  12 42.11185 0.7513599 11.24071 37.38724  0.1321399 6.191077
## 13  13 40.61851 0.7606251 10.94074 36.54779  0.1426015 6.343113
## 14  14 40.88087 0.7579561 10.79775 36.74818  0.1331153 6.037758
## 15  15 39.78693 0.7752660 10.55324 35.15421  0.1178714 5.829800
## 16  16 39.95041 0.7747625 10.62527 36.12246  0.1275236 6.345280
## 17  17 39.99719 0.7754936 10.53594 35.82309  0.1186042 6.147328
## 18  18 39.65325 0.7659599 10.45812 35.47643  0.1230800 5.942407
## 19  19 40.30878 0.7669197 10.54431 35.56743  0.1299860 5.769775
## 20  20 40.45559 0.7547113 10.56816 36.04052  0.1454454 5.832412
```

```
train.rf
```

```
## Random Forest
##
## 420 samples
## 43 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 336, 336, 336, 336, 336
## Resampling results across tuning parameters:
##
##   mtry  RMSE      Rsquared      MAE
## 1    1 66.76301 0.5869442 22.91298
## 2    2 55.11667 0.6734142 17.35421
## 3    3 49.52507 0.7017018 14.84608
## 4    4 47.65369 0.7351394 13.55577
## 5    5 46.12185 0.7309652 13.07311
## 6    6 44.74455 0.7306419 12.62889
## 7    7 43.80387 0.7447494 12.13825
## 8    8 43.95158 0.7431179 12.05859
## 9    9 43.30996 0.7534858 11.71387
## 10  10 41.77452 0.7591158 11.38698
## 11  11 41.57701 0.7597915 11.32502
## 12  12 42.11185 0.7513599 11.24071
## 13  13 40.61851 0.7606251 10.94074
## 14  14 40.88087 0.7579561 10.79775
## 15  15 39.78693 0.7752660 10.55324
## 16  16 39.95041 0.7747625 10.62527
## 17  17 39.99719 0.7754936 10.53594
## 18  18 39.65325 0.7659599 10.45812
## 19  19 40.30878 0.7669197 10.54431
## 20  20 40.45559 0.7547113 10.56816
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was mtry = 18.
```

```
best.rf <- train.rf$finalModel
best.rf
```

```

## 
## Call:
##   randomForest(x = x, y = y, mtry = param$mtry)
##     Type of random forest: regression
##       Number of trees: 500
## No. of variables tried at each split: 18
## 
##       Mean of squared residuals: 2877.38
##       % Var explained: 69.22

```

```

set.seed(99)
rf.slopemod <- randomForest(slope ~ ., data = slopemod.train, mtry = 18, nodesize = 5, ntree = 1000)
pred.best.rf.s <- predict(rf.slopemod, newdata = slopemod.test) # can use same model matrix

SSE = sum((slopemod.test$slope - pred.best.rf.s)^2)
SST = sum((slopemod.test$slope - mean(slopemod.train$slope))^2)
OSR2_rf_s = 1 - SSE/SST # is this model useful at all?
OSR2_rf_s

```

```

## [1] 0.4156719

```

```

mae_rf.s <- MAE(pred = pred.best.rf.s, obs = slopemod.test$slope)

```

The slope prediction is functioning far better than the direct prediction- the random forest model (cross validated, using an mtry value of 18) has an OSR squared of .41 and a mean average error of 11.07.

```

slopemod.train.goodr <- slopemod.train
#%>%
#filter(r2>.7)

slopemod.test.goodr <- slopemod.test
#%>%
#filter(r2>.7)

```

Next, other values of mtry are evaluated for performance for comparison to the mtry indicated by cross validation.

```

set.seed(190)
mod.rf1 <- randomForest(slope ~ ., data = slopemod.train.goodr, mtry = 2, nodesize = 5, ntree = 1000)

pred.rf.s2 <- predict(mod.rf1, newdata = slopemod.test.goodr)

RFPredictions.s2 <- predict(mod.rf1, newdata = slopemod.test.goodr)

#SSE = sum((slopemod.test.goodr$slope - RFPredictions.s2)^2)
#SST = sum((slopemod.test.goodr$slope - mean(slopemod.test.goodr$slope))^2)
#OSR2_rf_s2 = 1 - SSE/SST # is this model useful at all?
OSR2_rf_s2= OSR2(RFPredictions.s2, slopemod.train.goodr$slope, slopemod.test.goodr$slope)
OSR2_rf_s2

```

```

## [1] 0.807931

```

```

mae_rf.s2 <- MAE(pred = RFPredictions.s2, obs = slopemod.test$slope)

```

A low value of mtry creates generates predictions with a higher OSR squared, but a slightly worse mae indicating that performance has not in fact improved as much as the high OSR squared would indicate.

Visualizing random forest slope prediction performance

```

#appending on table of predictions
slopemod.test.withpred <- slopemod.test.o

slopemod.test.withpred$rf_slope <- pred.best.rf.s

slopemod.test.withpred <- slopemod.test.withpred %>%
  mutate(rf_slope_dif = abs(slope-rf_slope))

for(i in 1:nrow(cal_pts)){
  bid_temp <- cal_pts$Compound.Name[i]

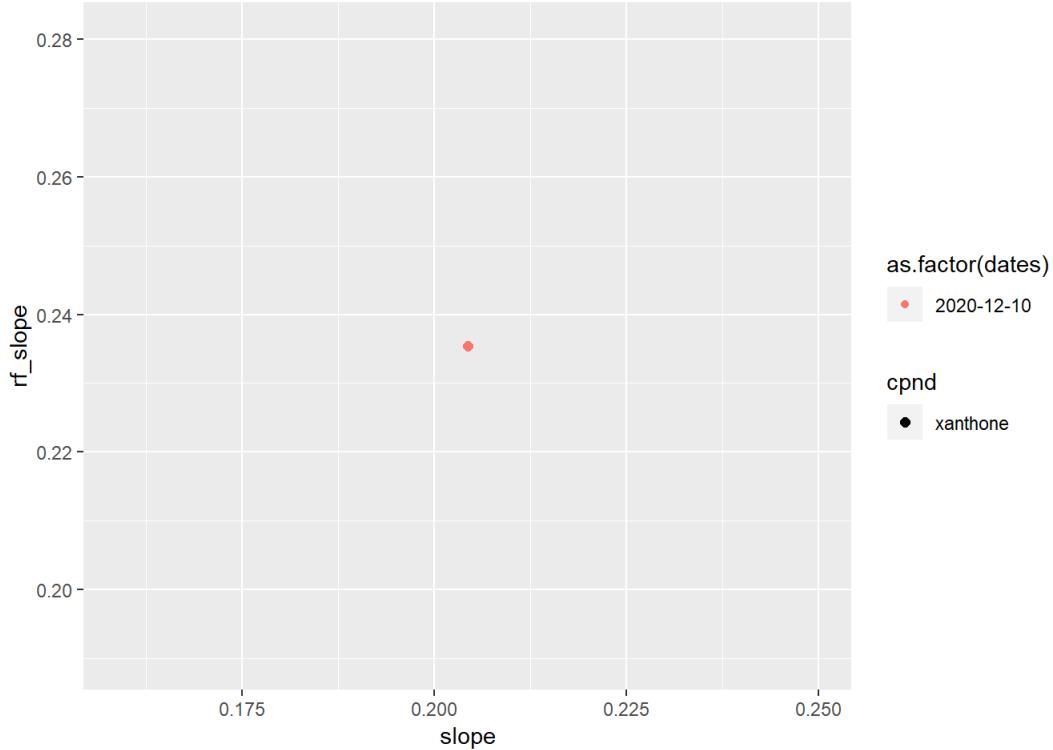
  n_c_full_sub <- slopemod.test.withpred %>%
    filter(cpnd == bid_temp)

  p <- n_c_full_sub %>%
    ggplot(aes(x = slope, y = rf_slope, color = as.factor(dates), size = cpnd)) +
    geom_point()

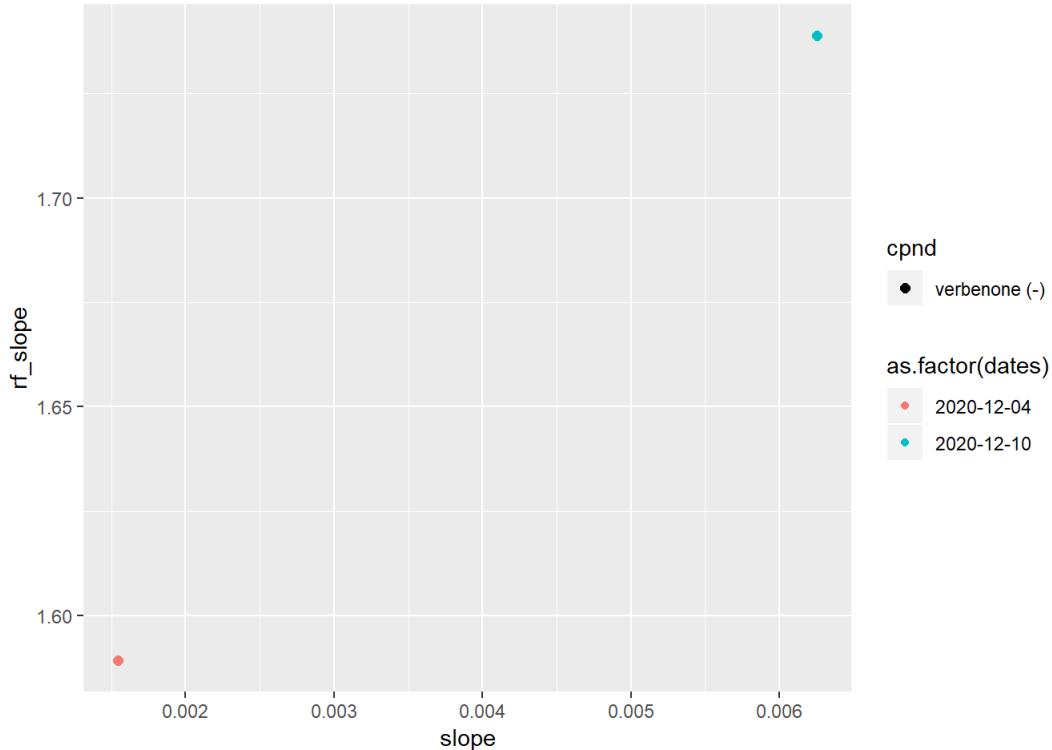
  print(p)
}

```

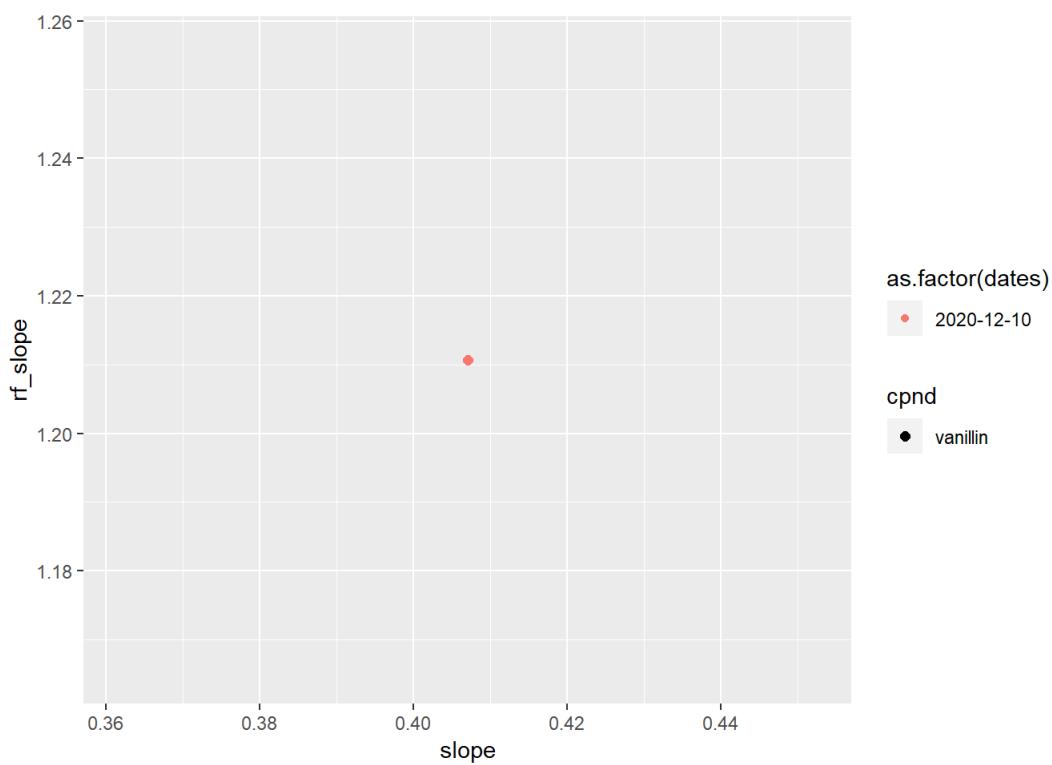
Warning: Using size for a discrete variable is not advised.



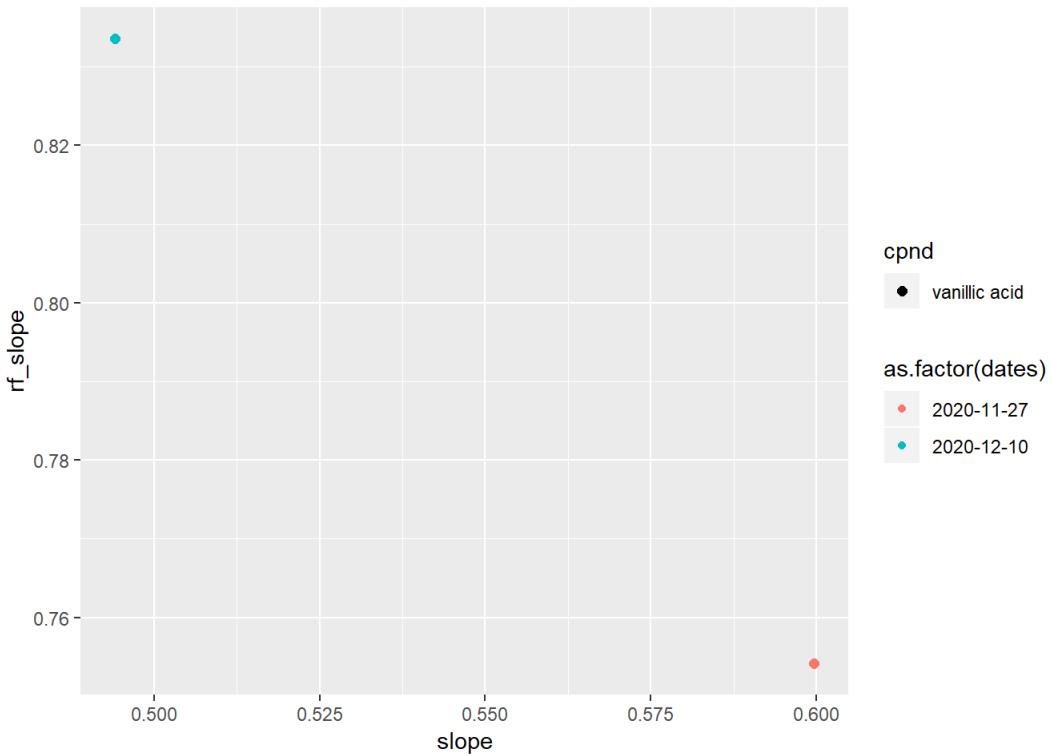
Warning: Using size for a discrete variable is not advised.



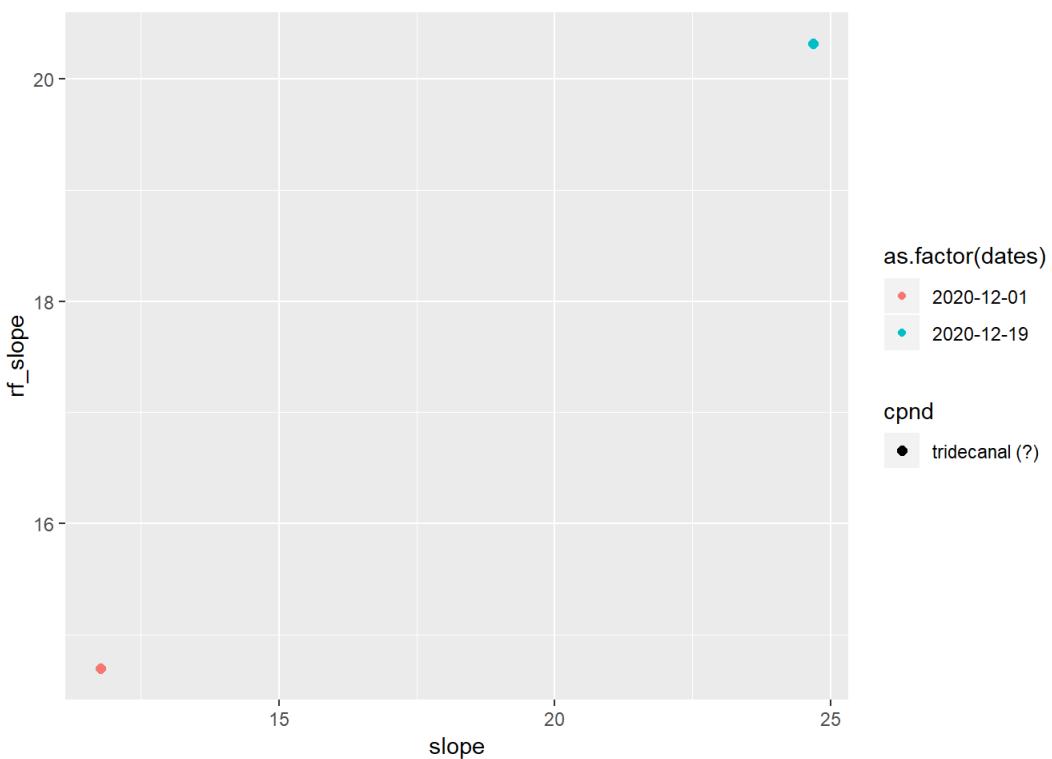
```
## Warning: Using size for a discrete variable is not advised.
```



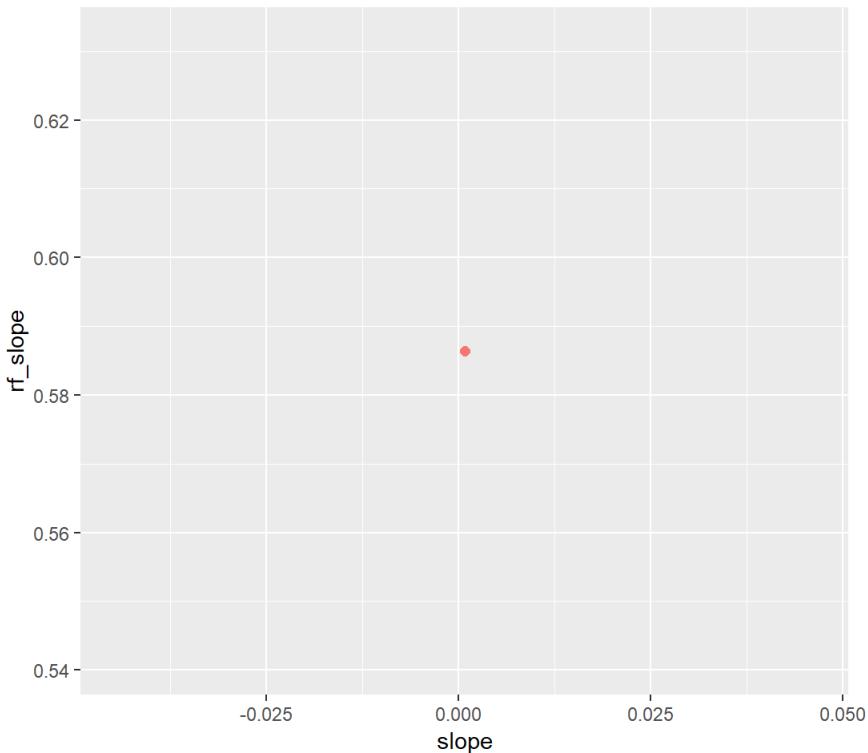
```
## Warning: Using size for a discrete variable is not advised.
```



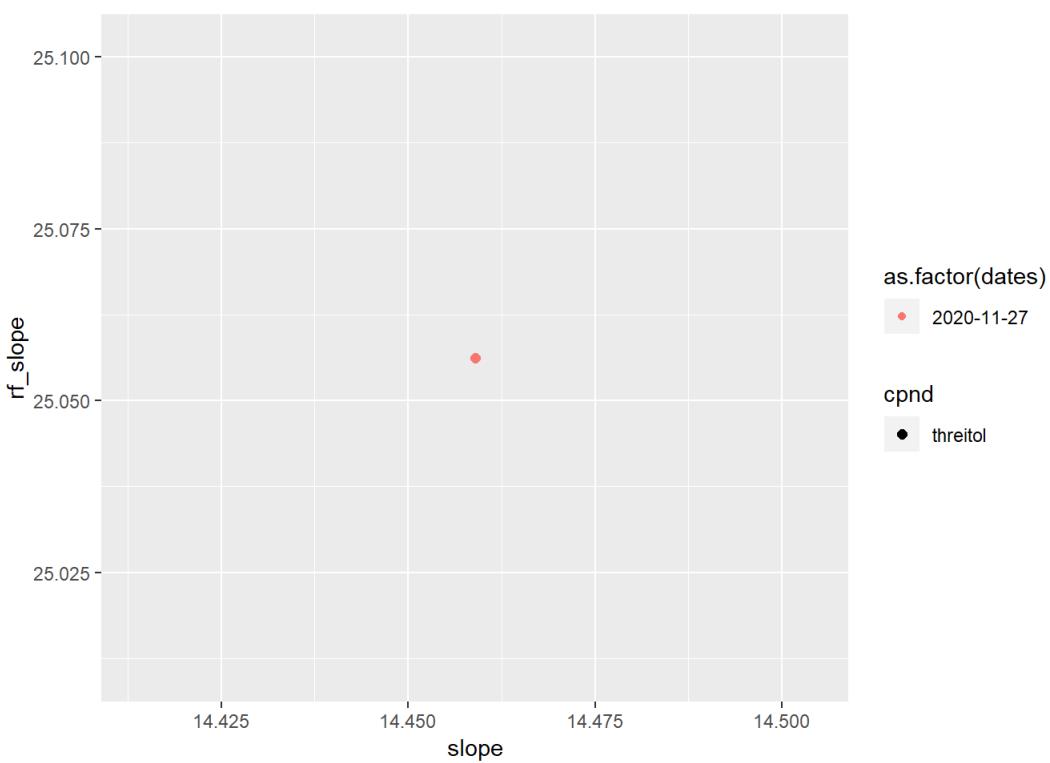
```
## Warning: Using size for a discrete variable is not advised.
```



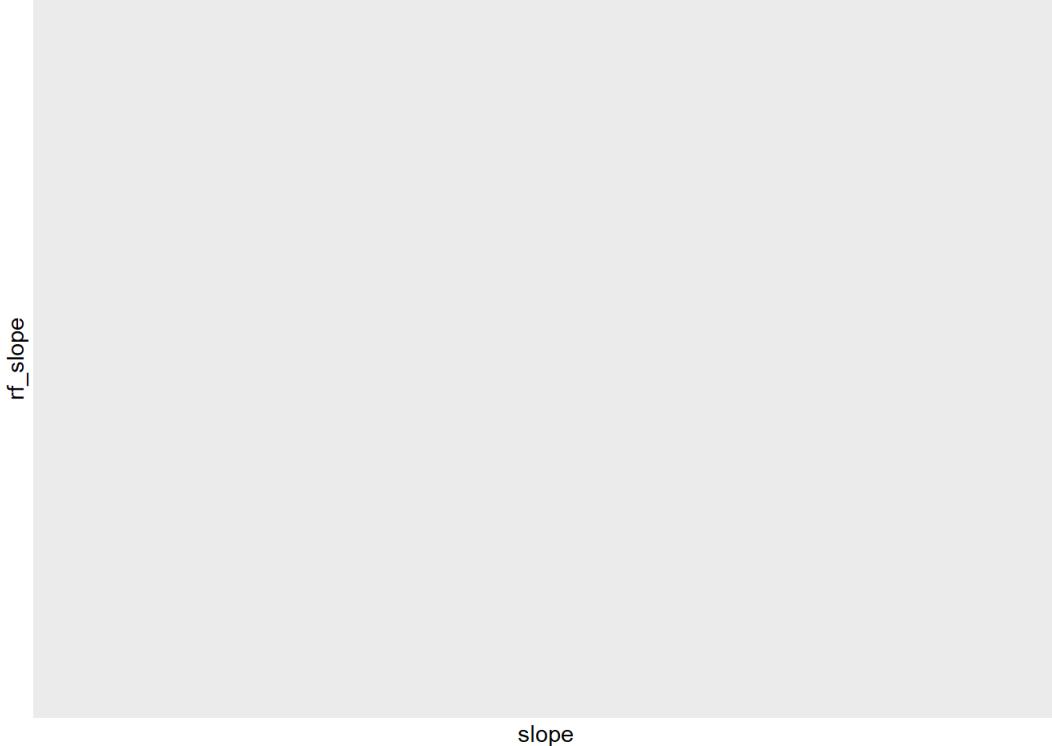
```
## Warning: Using size for a discrete variable is not advised.
```



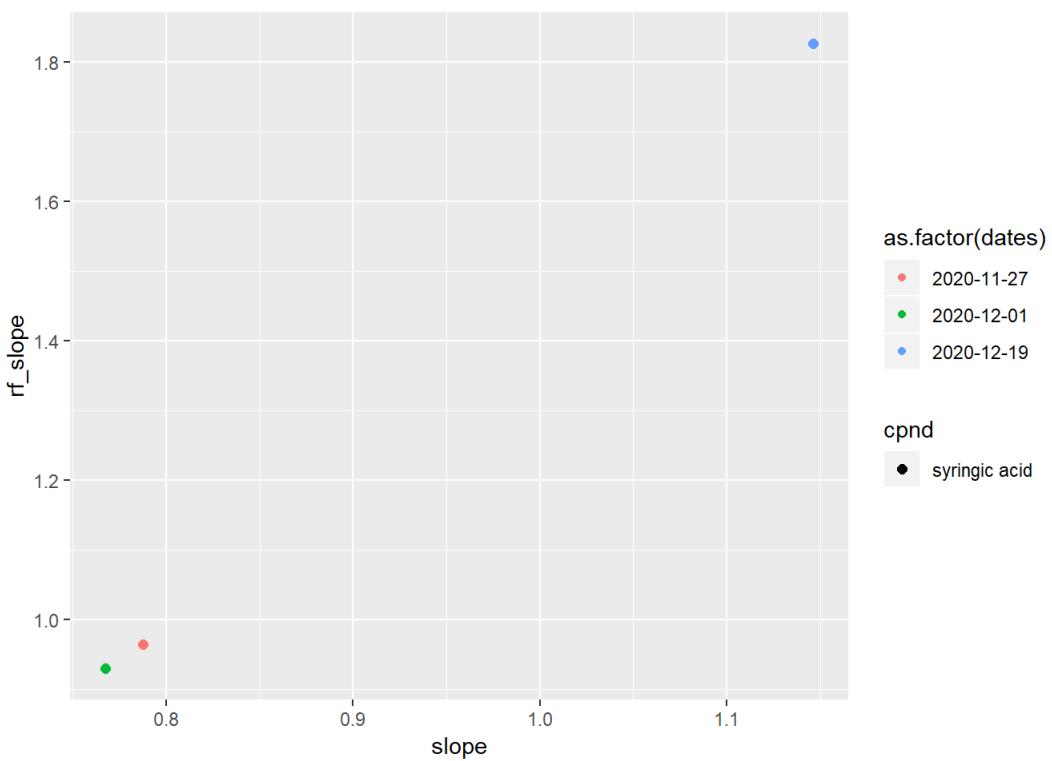
```
## Warning: Using size for a discrete variable is not advised.
```



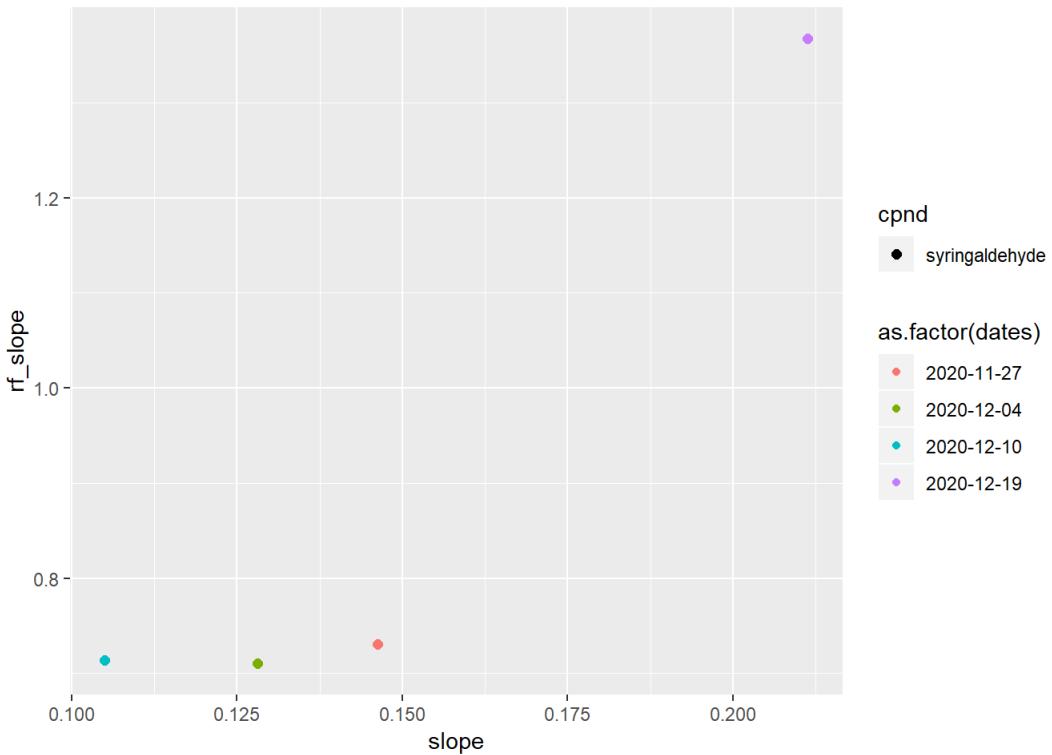
```
## Warning: Using size for a discrete variable is not advised.
```



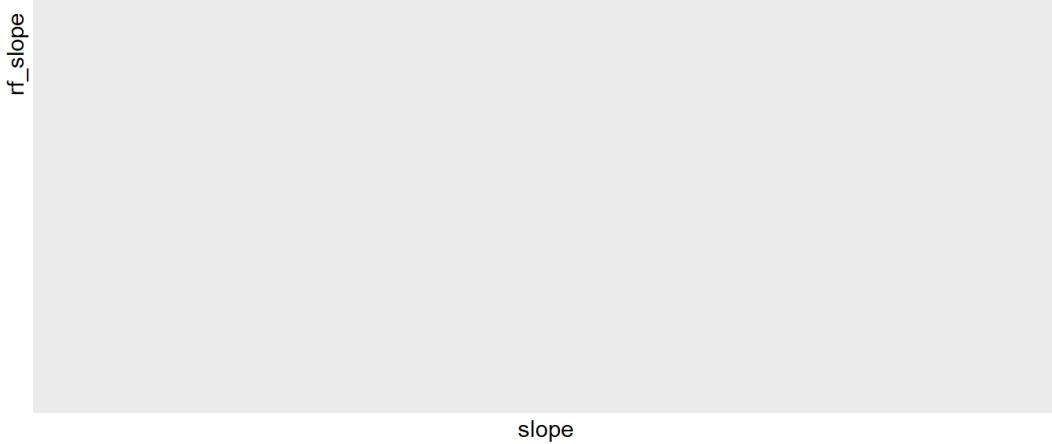
```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```

rf_slope

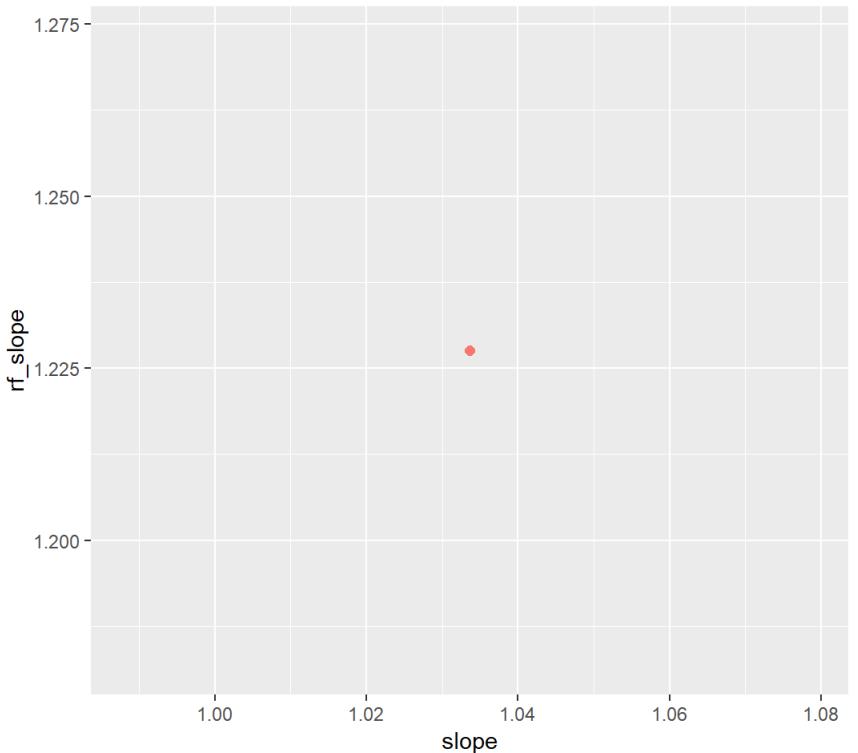
slope

```
## Warning: Using size for a discrete variable is not advised.
```

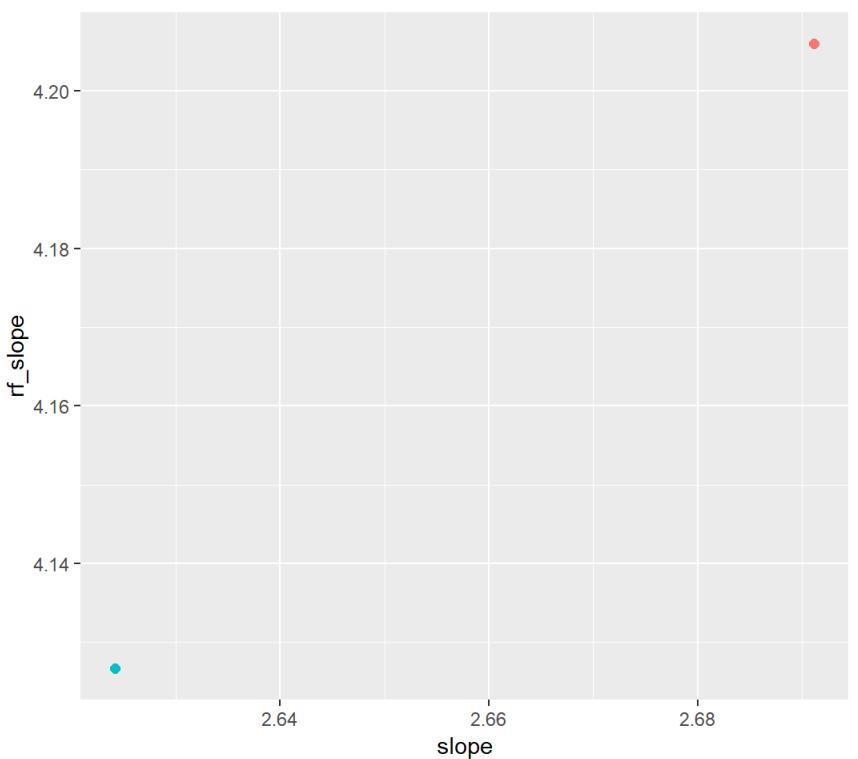
rf_slope

slope

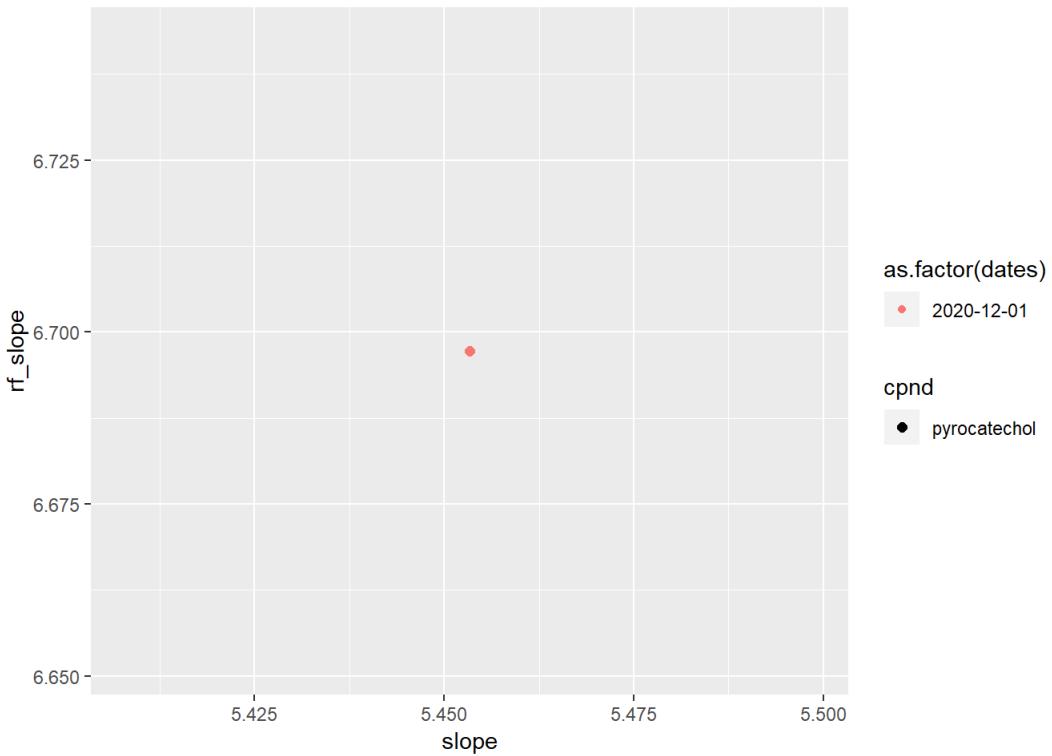
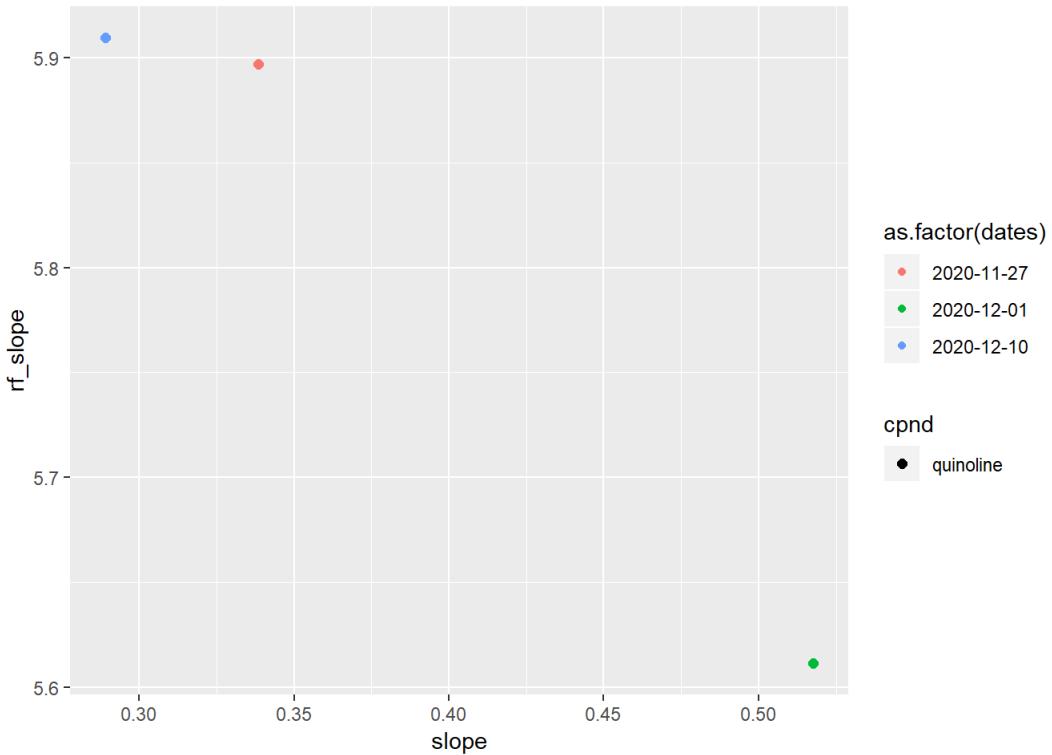
```
## Warning: Using size for a discrete variable is not advised.
```

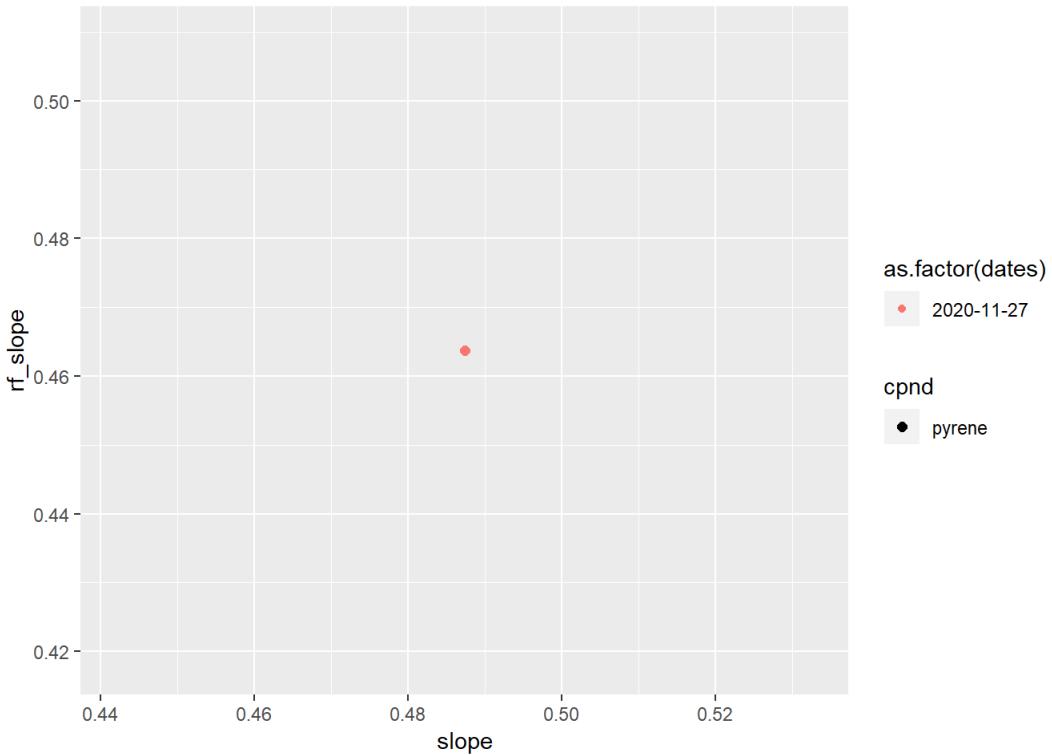


```
## Warning: Using size for a discrete variable is not advised.
```

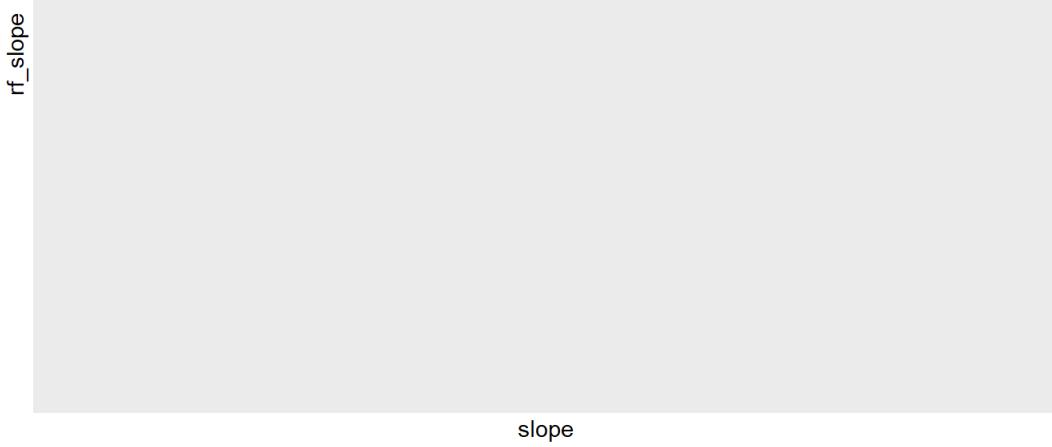


```
## Warning: Using size for a discrete variable is not advised.
```

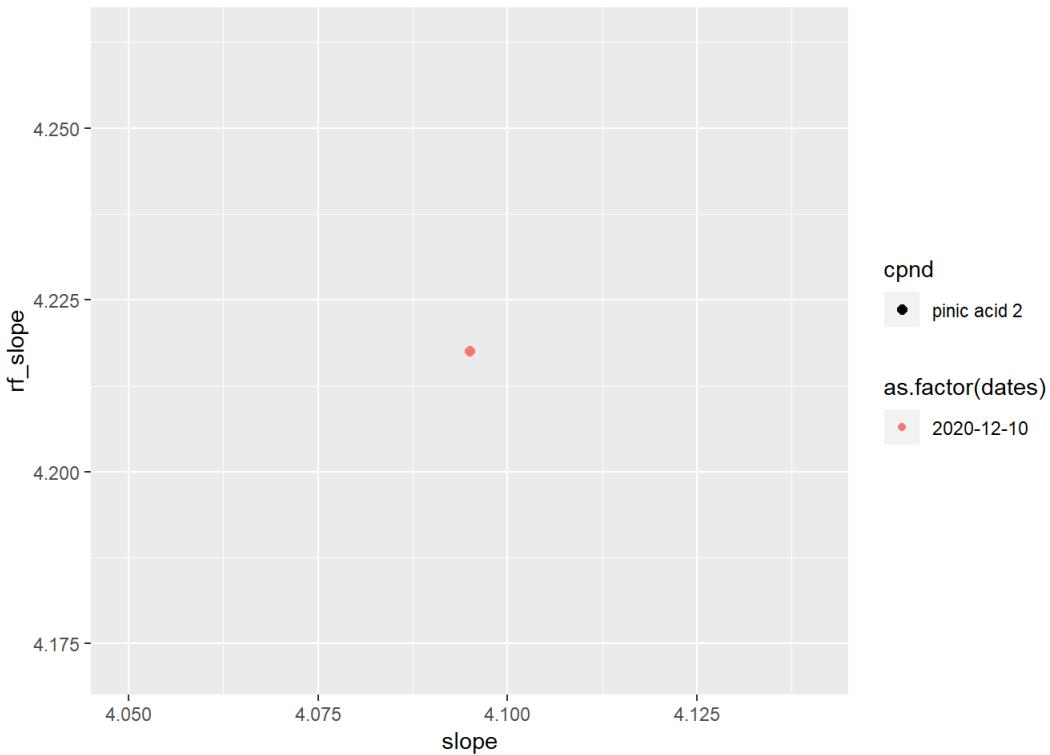




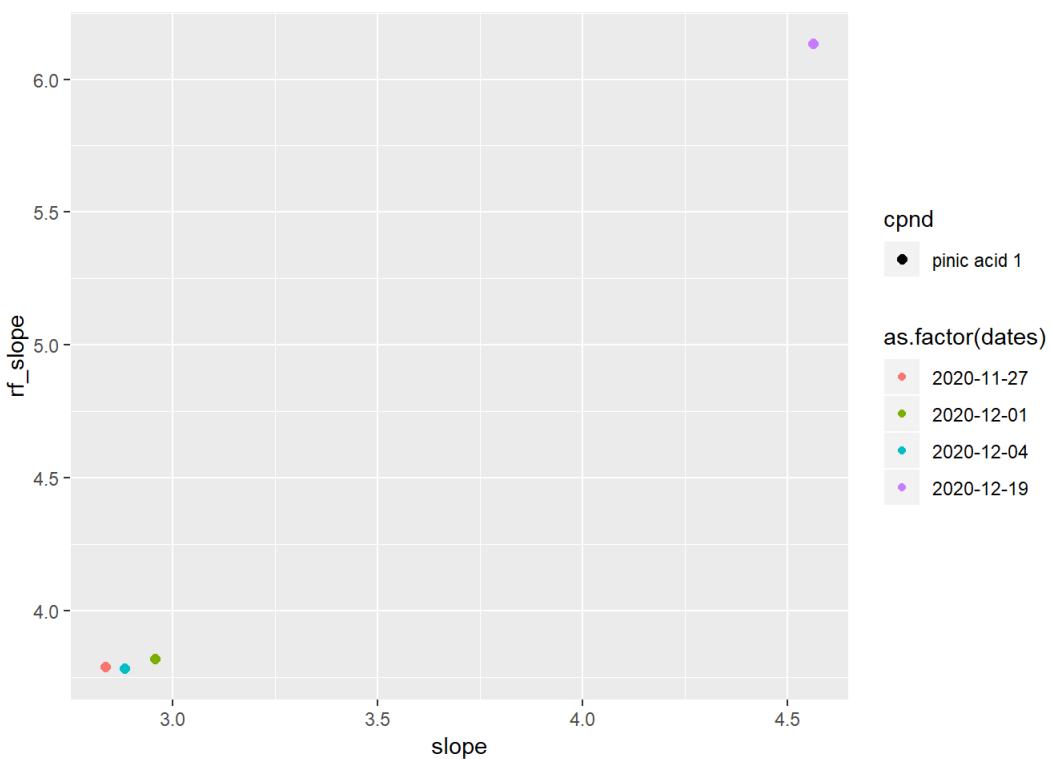
```
## Warning: Using size for a discrete variable is not advised.
```



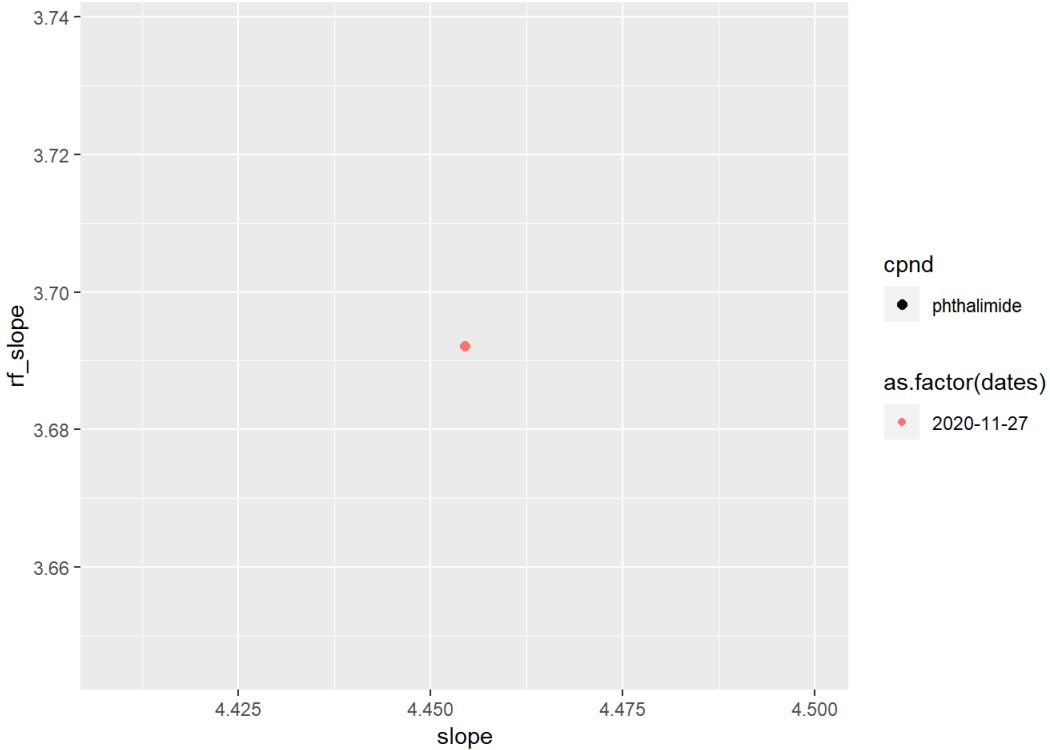
```
## Warning: Using size for a discrete variable is not advised.
```



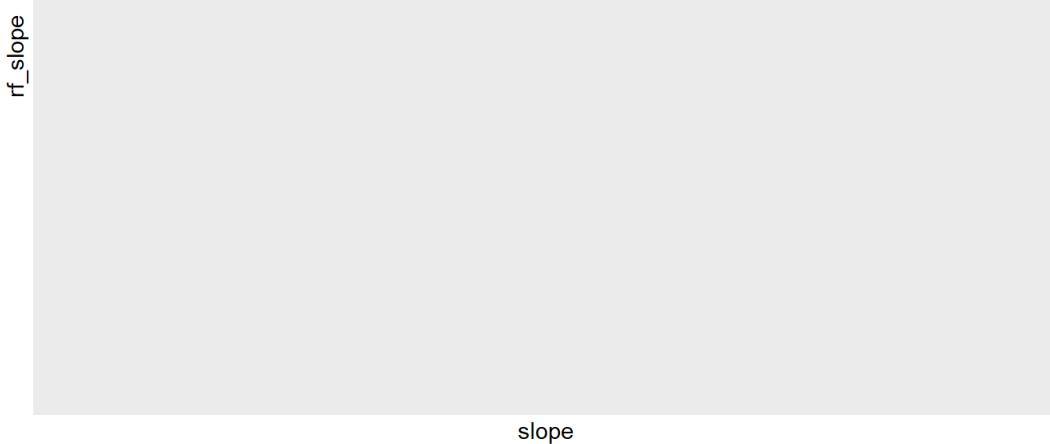
```
## Warning: Using size for a discrete variable is not advised.
```



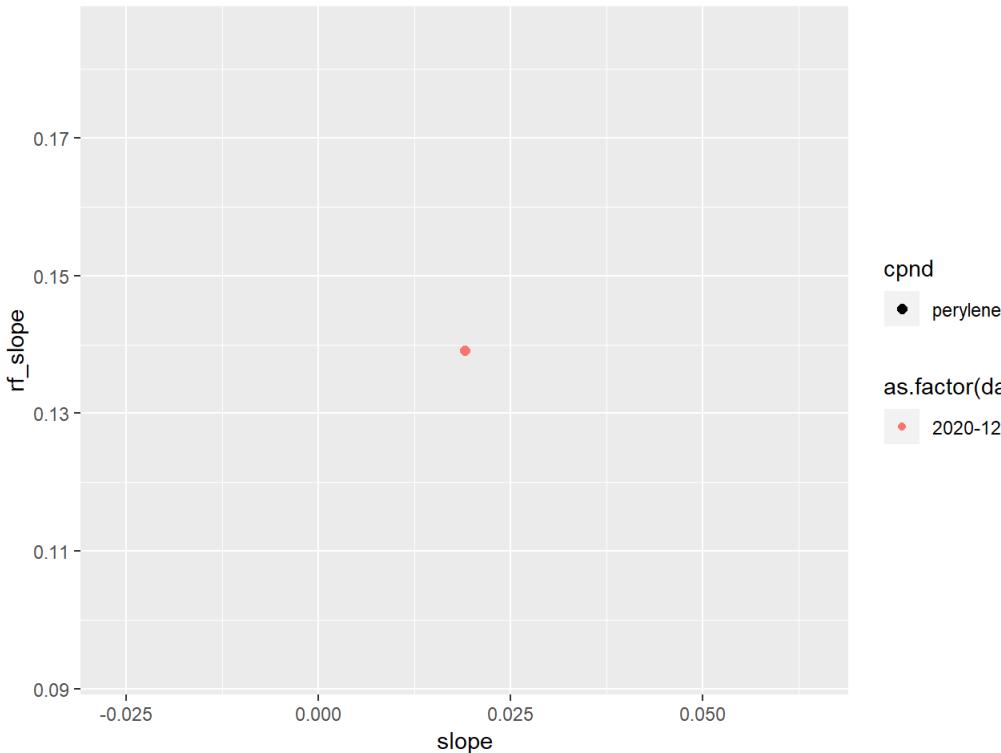
```
## Warning: Using size for a discrete variable is not advised.
```



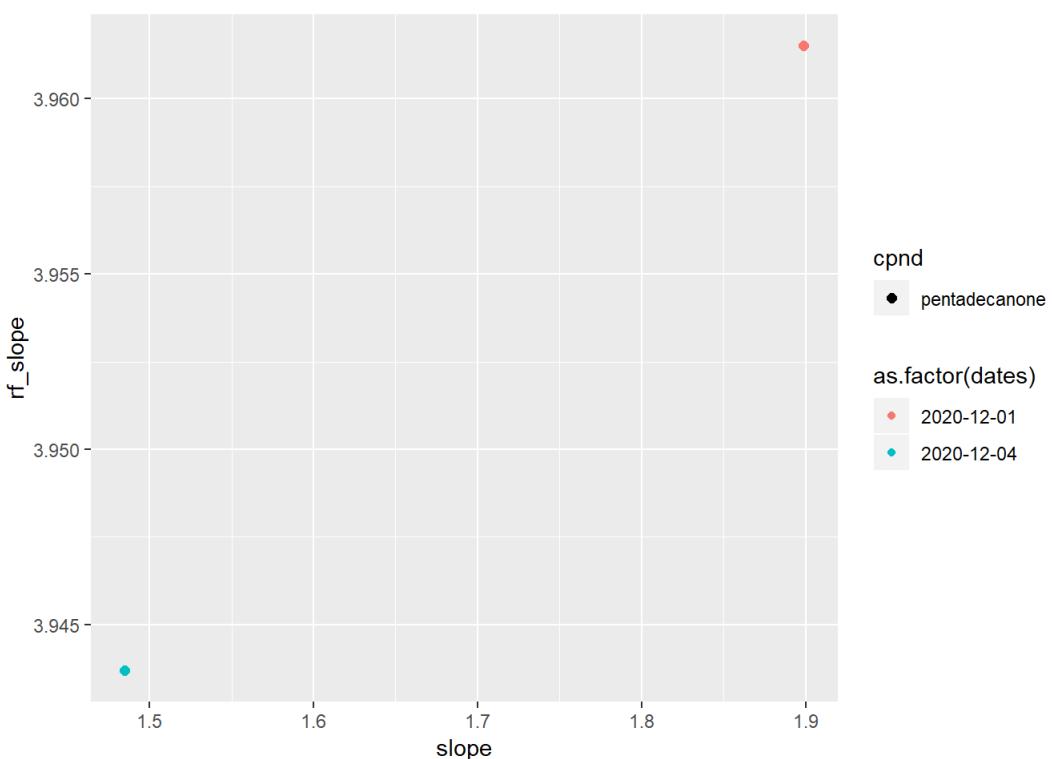
```
## Warning: Using size for a discrete variable is not advised.
```



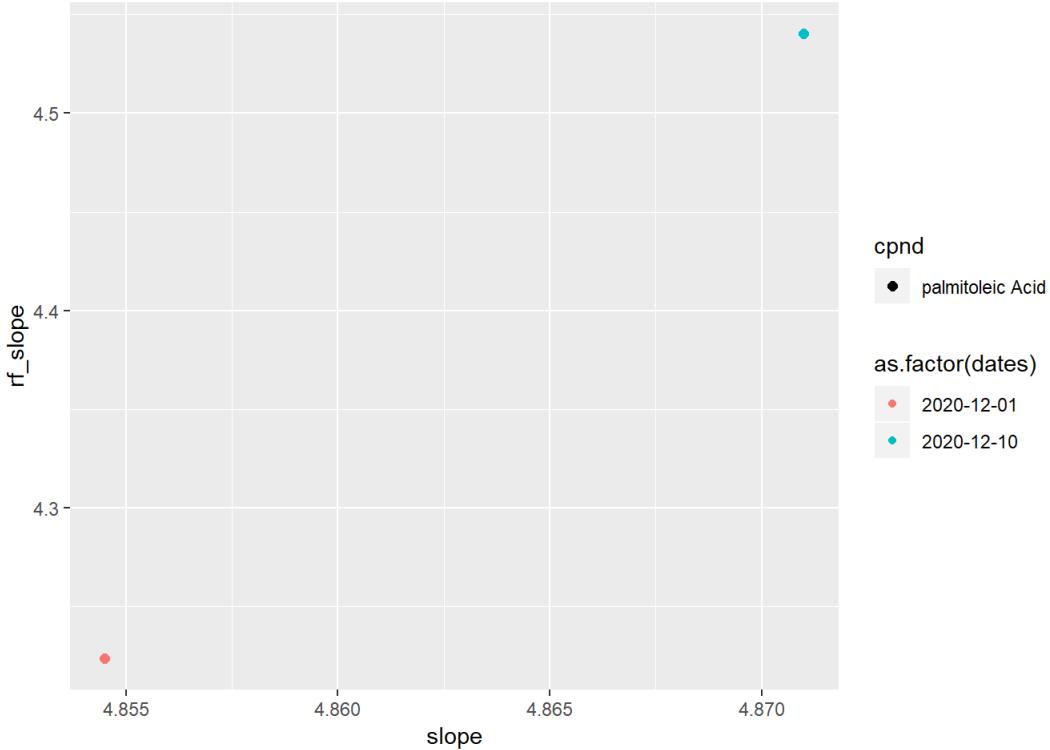
```
## Warning: Using size for a discrete variable is not advised.
```



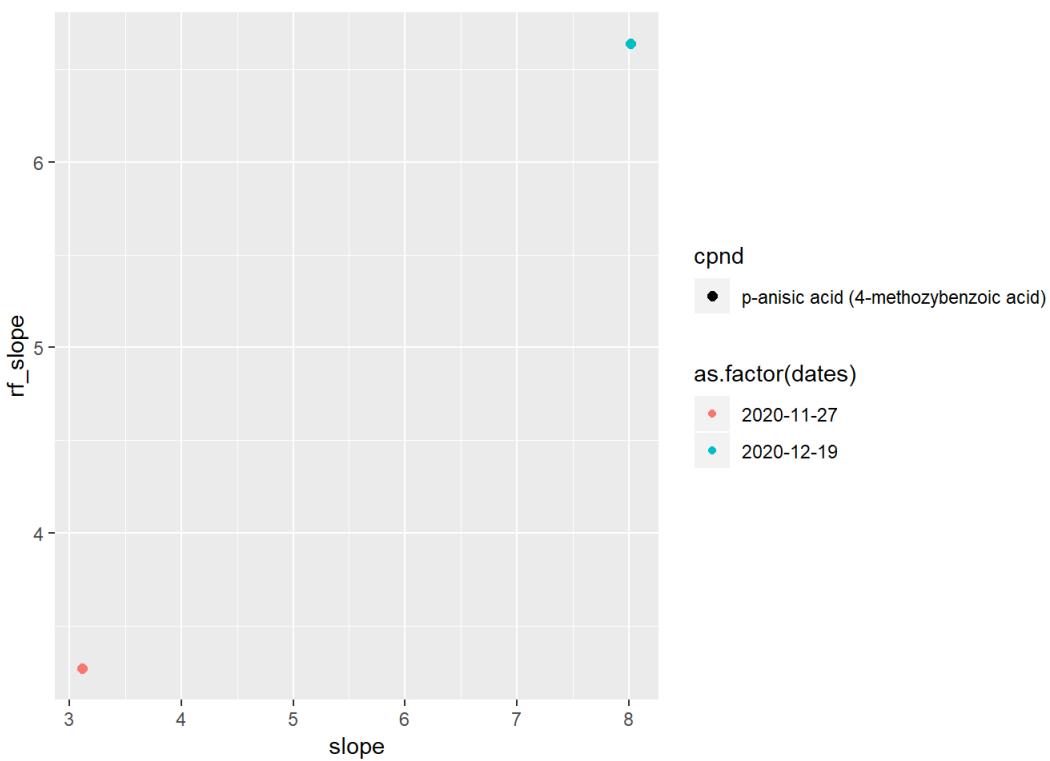
```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```

rf_slope

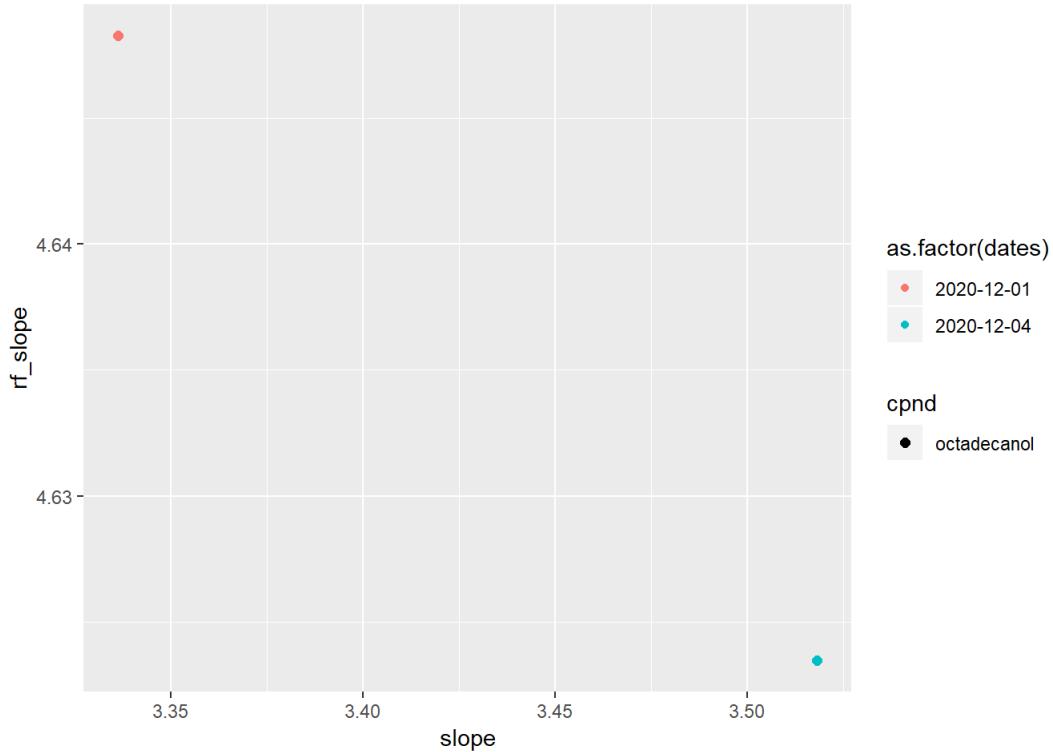
slope

```
## Warning: Using size for a discrete variable is not advised.
```

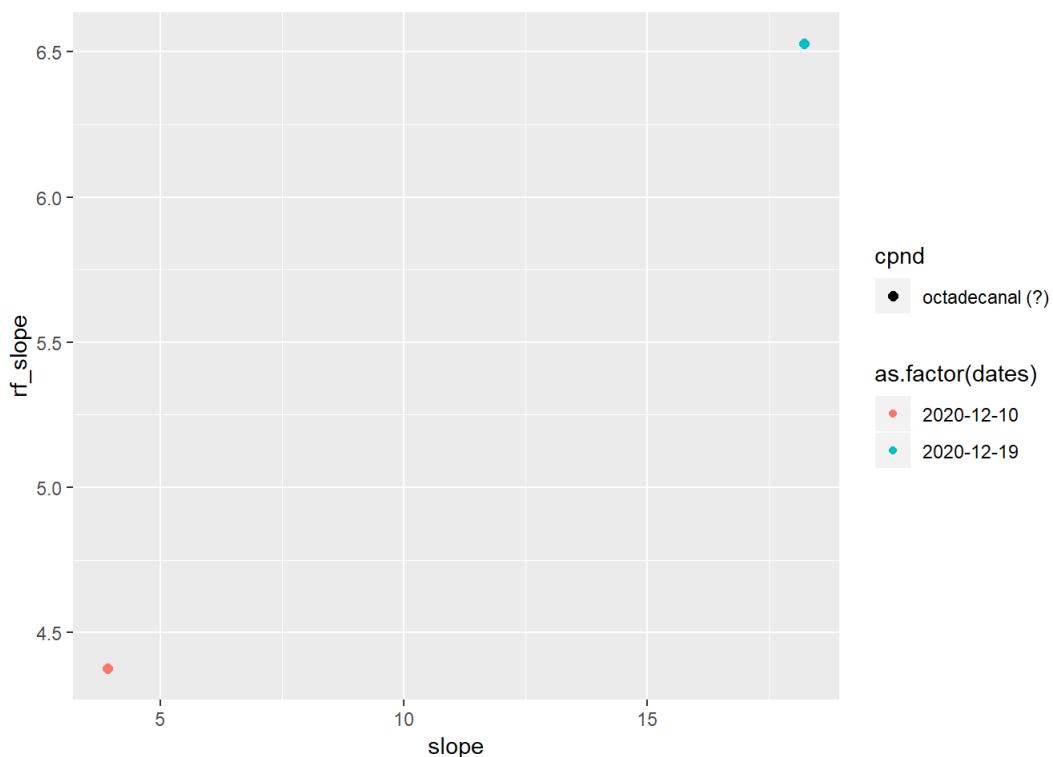
rf_slope

slope

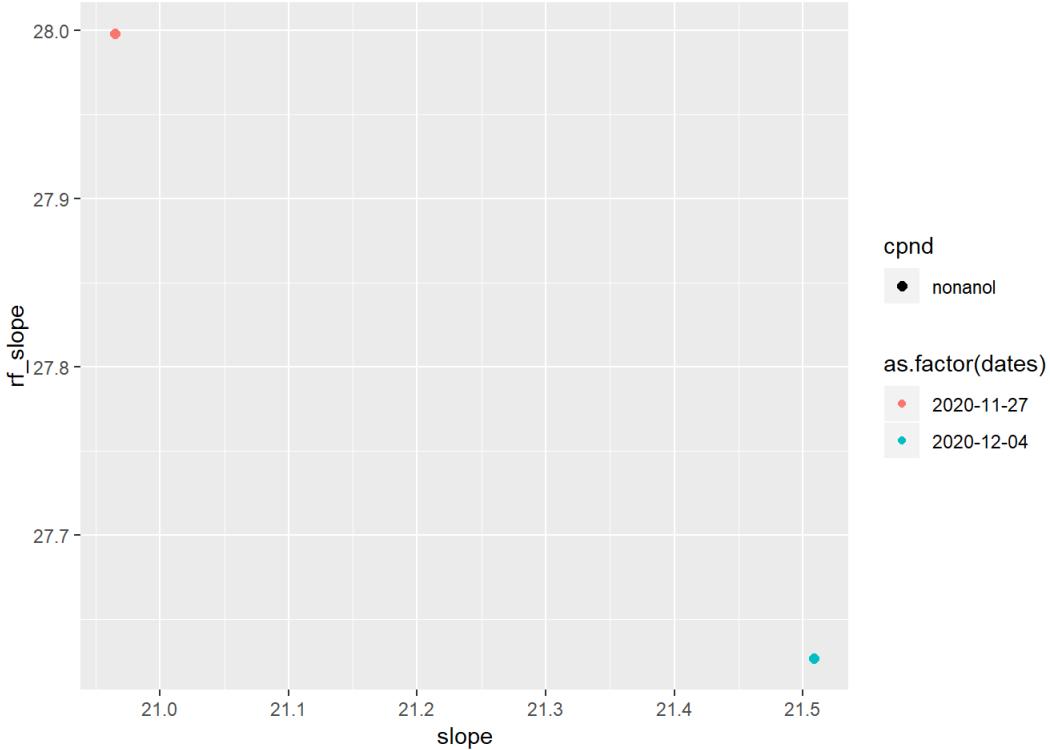
```
## Warning: Using size for a discrete variable is not advised.
```



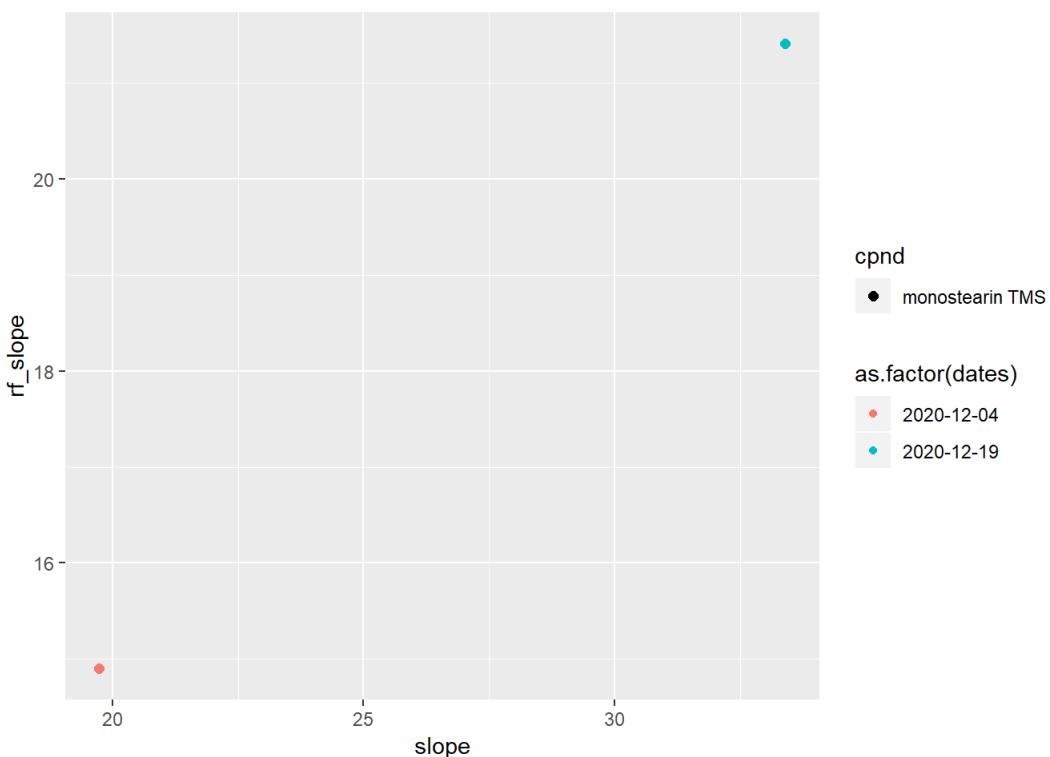
```
## Warning: Using size for a discrete variable is not advised.
```



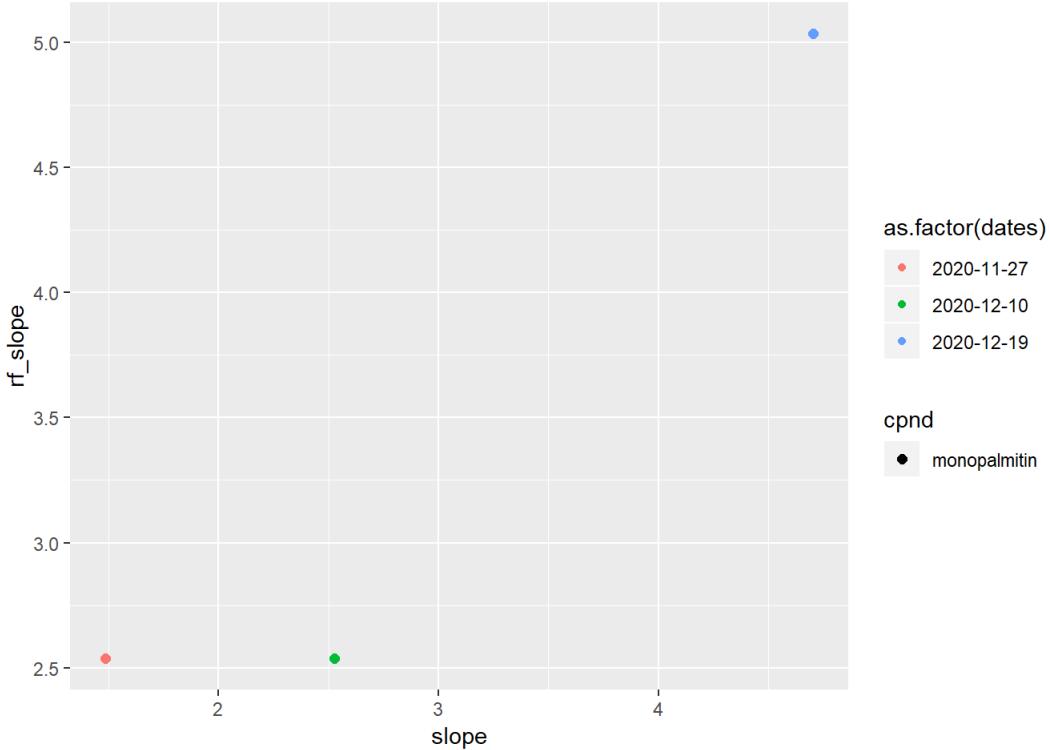
```
## Warning: Using size for a discrete variable is not advised.
```



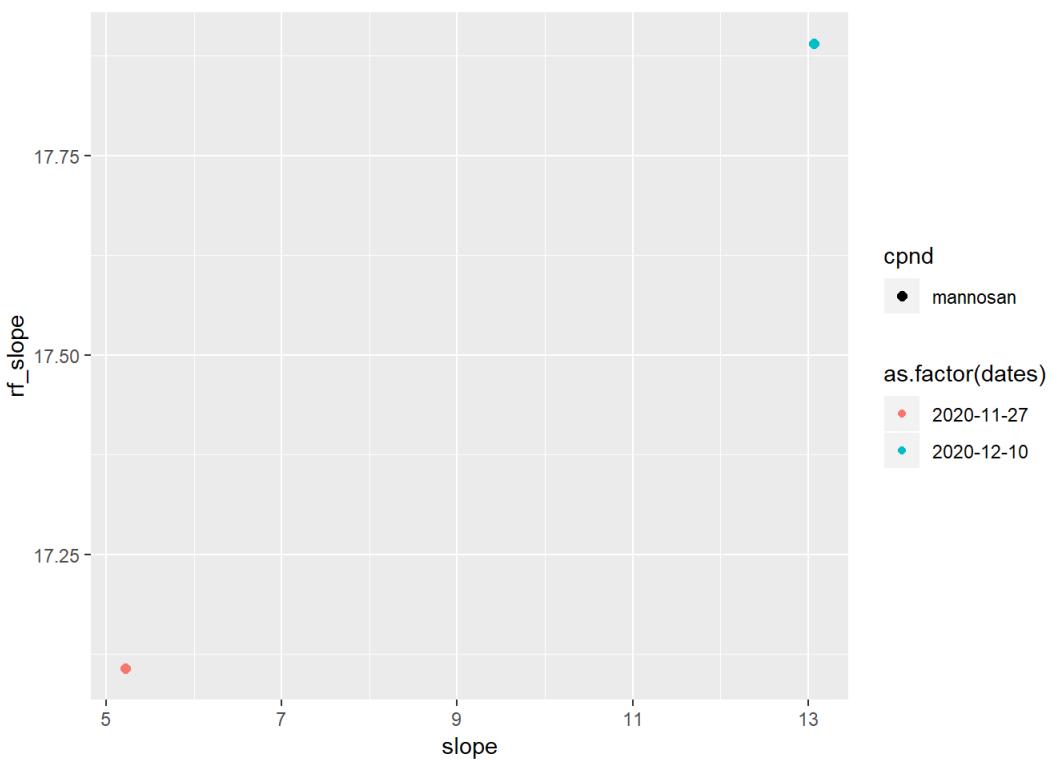
```
## Warning: Using size for a discrete variable is not advised.
```



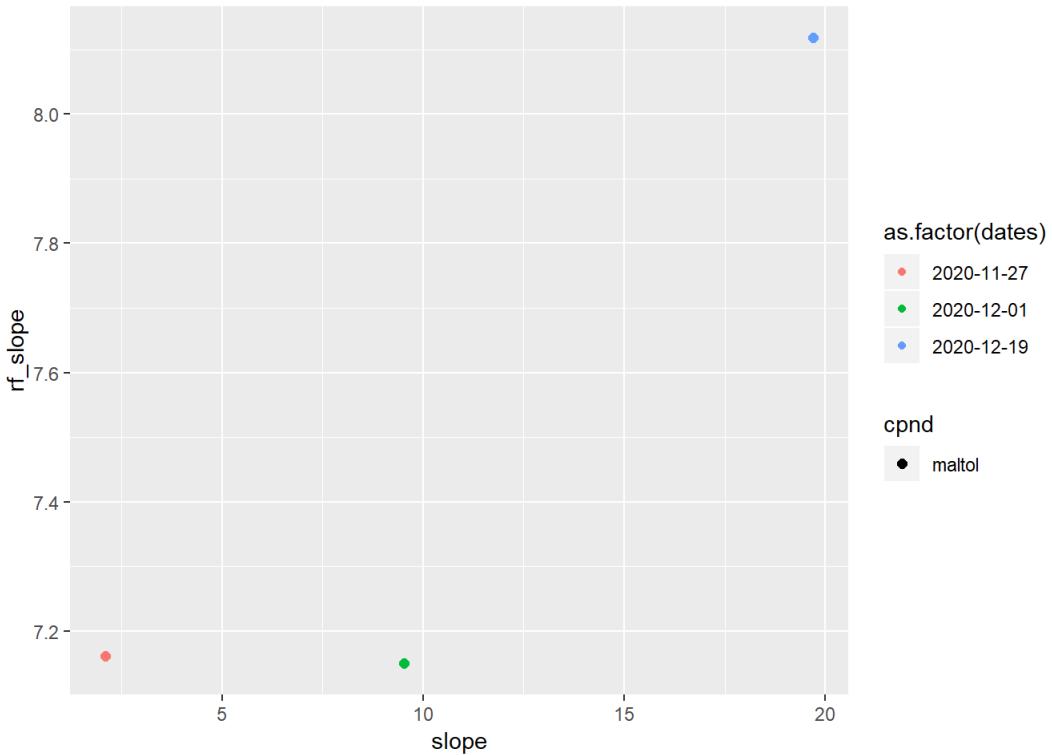
```
## Warning: Using size for a discrete variable is not advised.
```



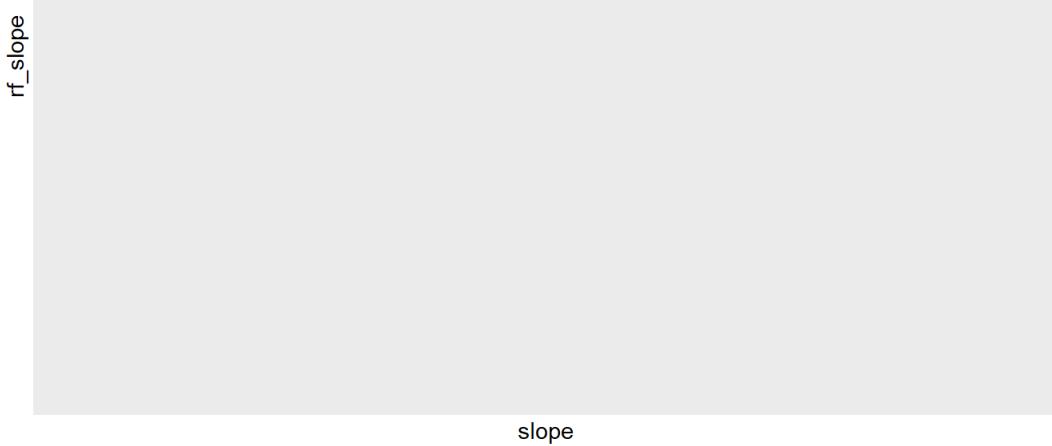
```
## Warning: Using size for a discrete variable is not advised.
```



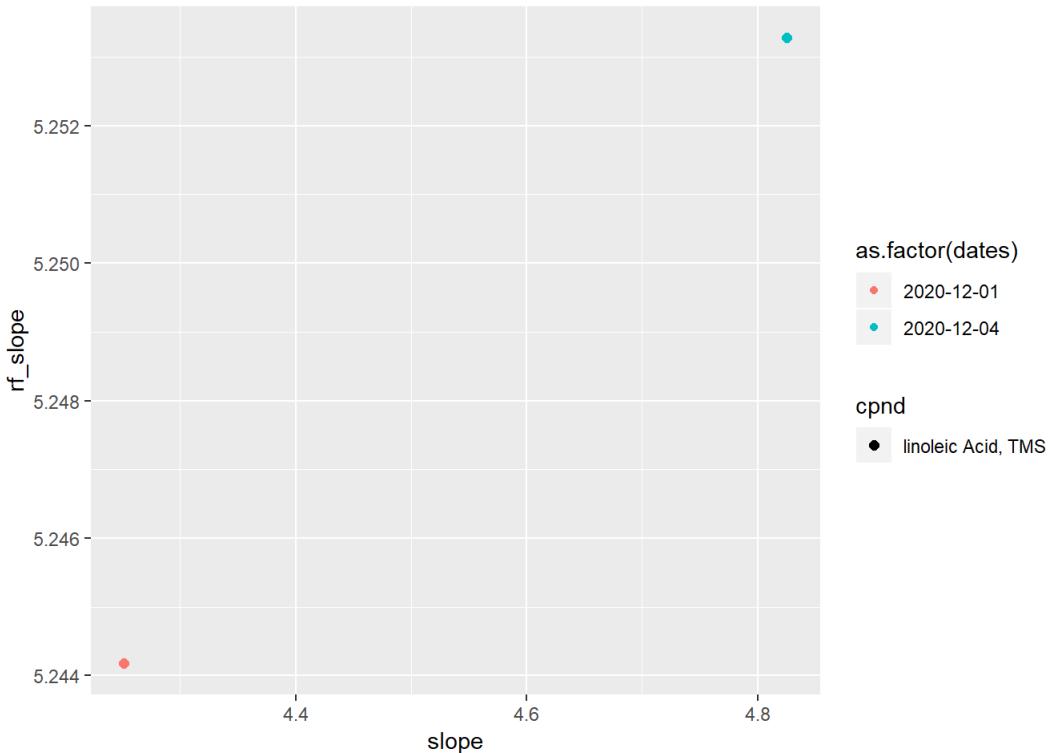
```
## Warning: Using size for a discrete variable is not advised.
```



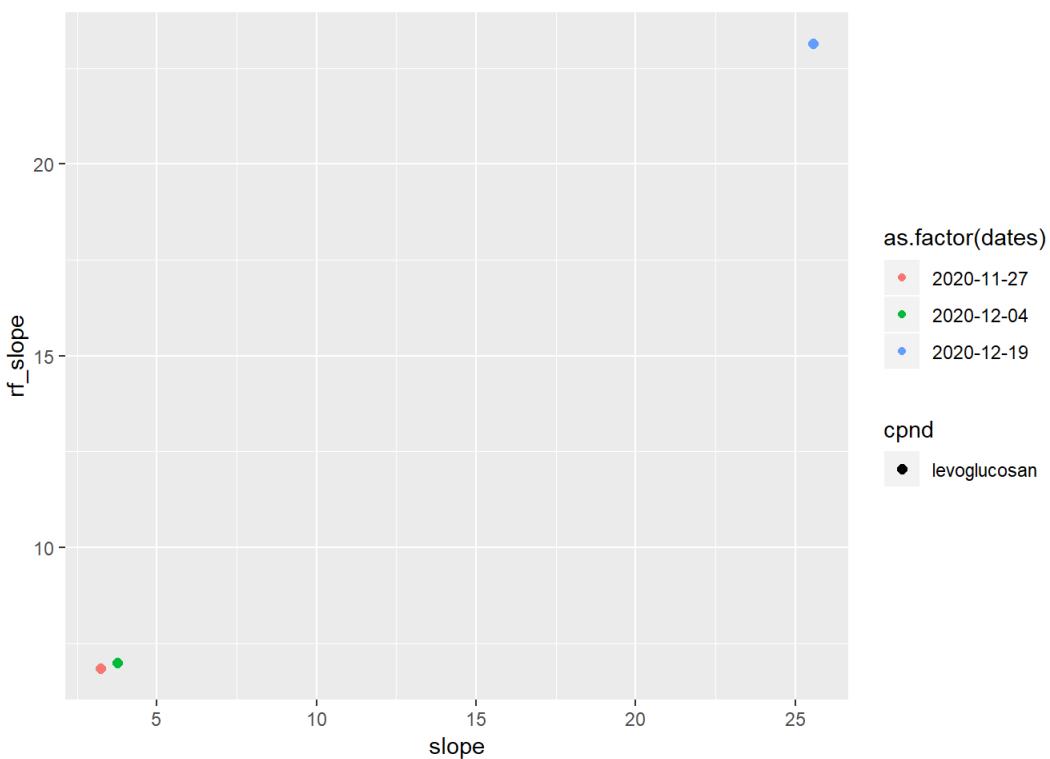
```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```

rf_slope

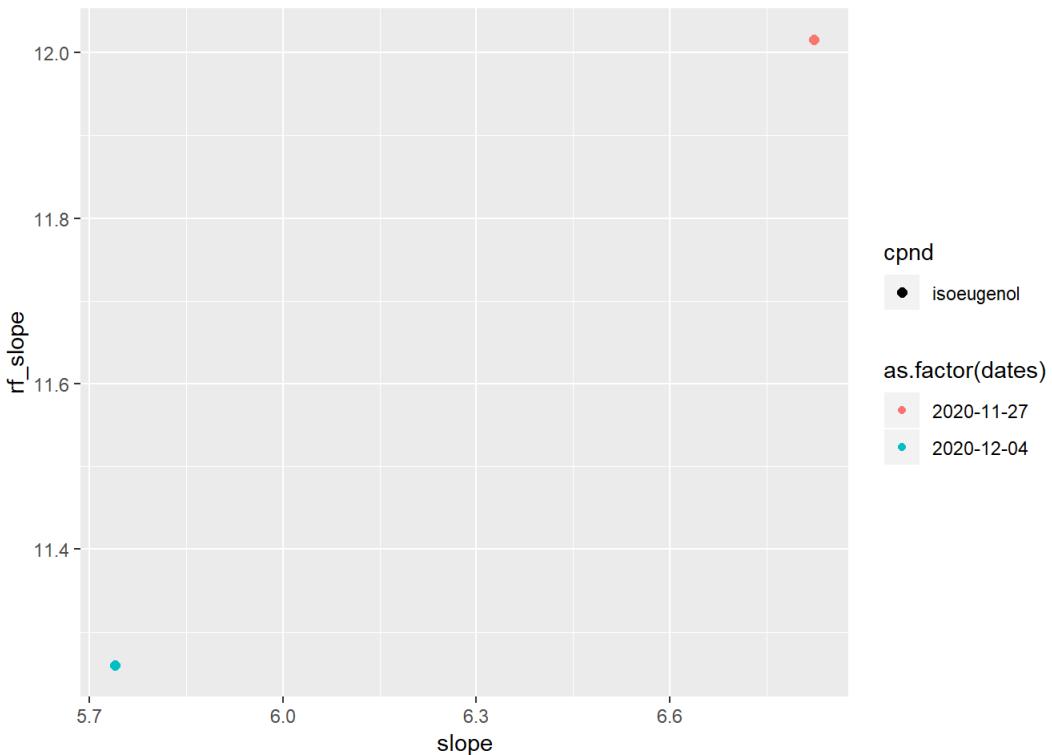
slope

```
## Warning: Using size for a discrete variable is not advised.
```

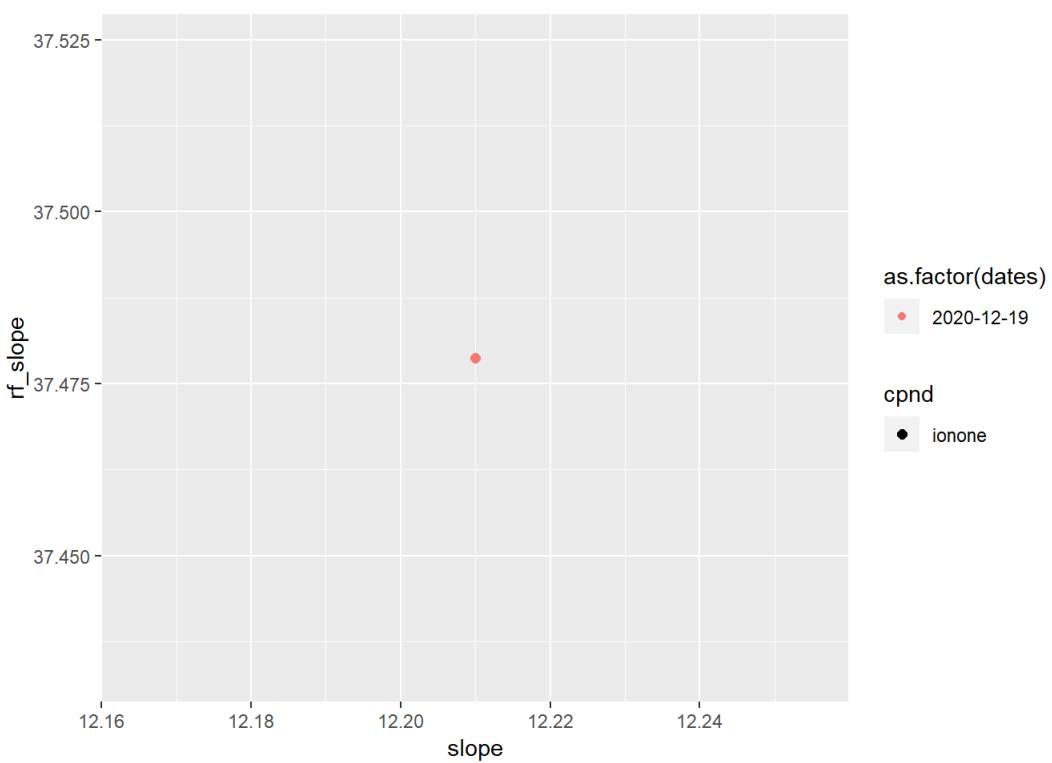
rf_slope

slope

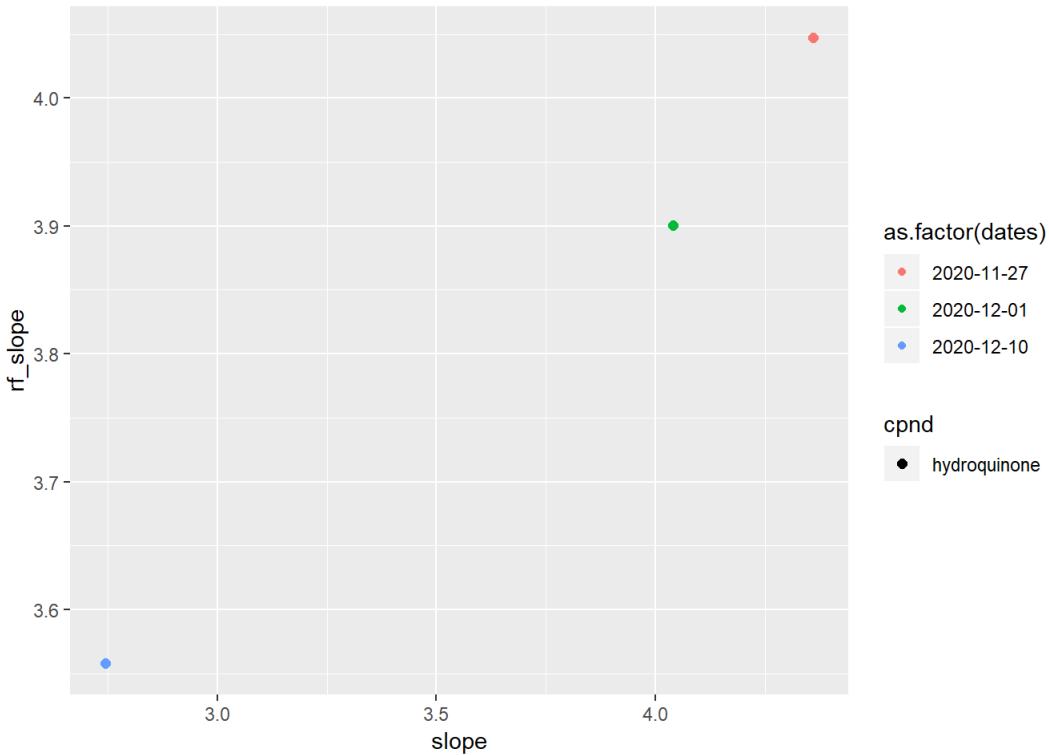
```
## Warning: Using size for a discrete variable is not advised.
```



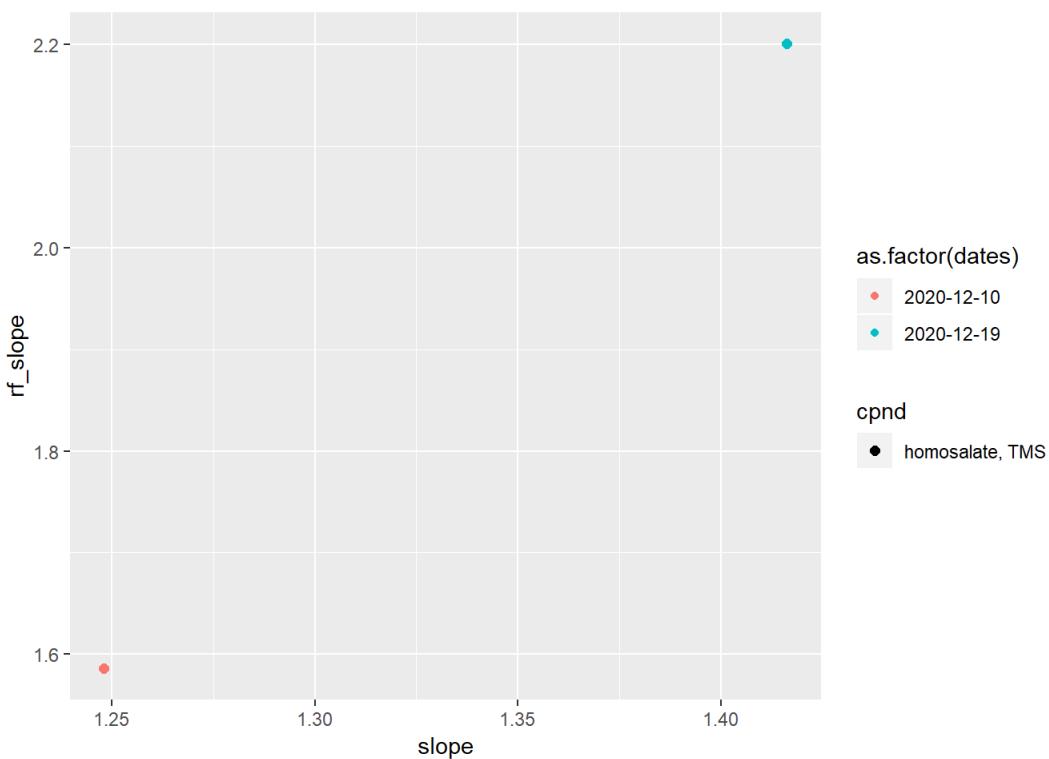
```
## Warning: Using size for a discrete variable is not advised.
```



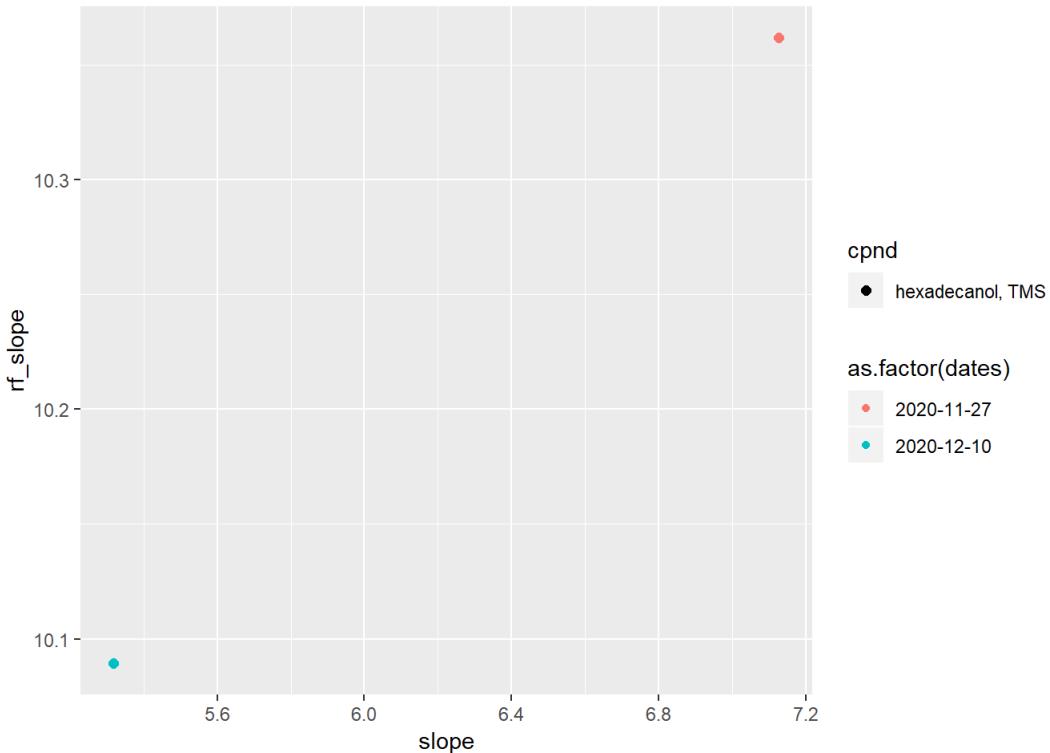
```
## Warning: Using size for a discrete variable is not advised.
```



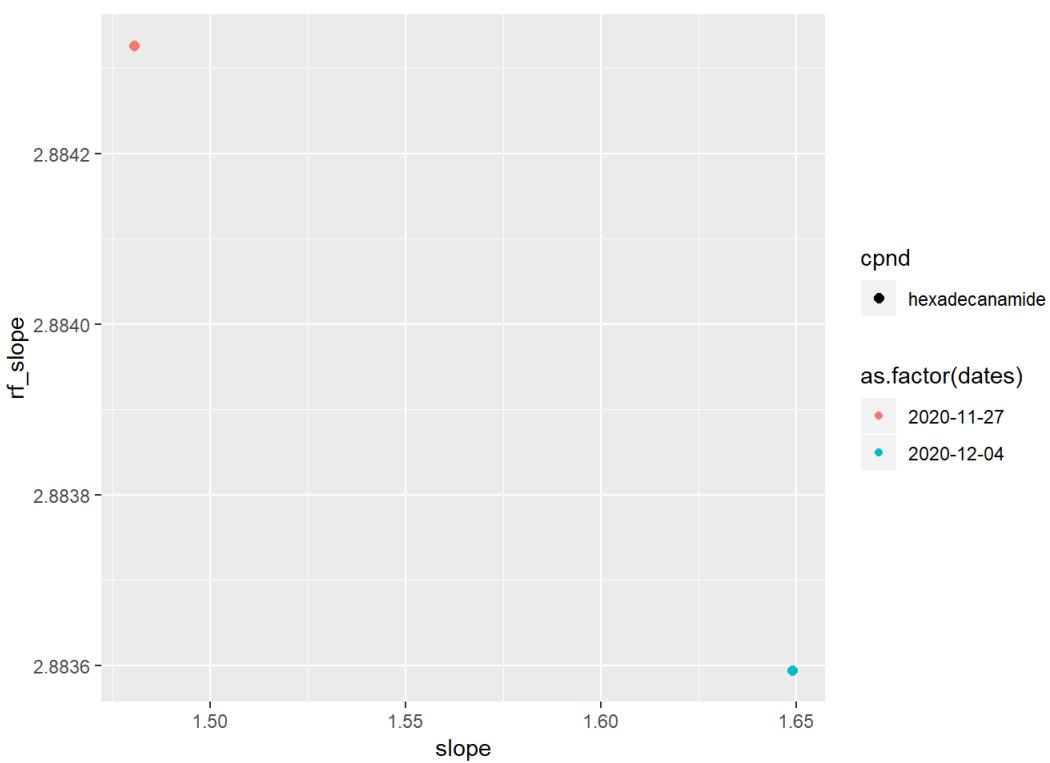
```
## Warning: Using size for a discrete variable is not advised.
```



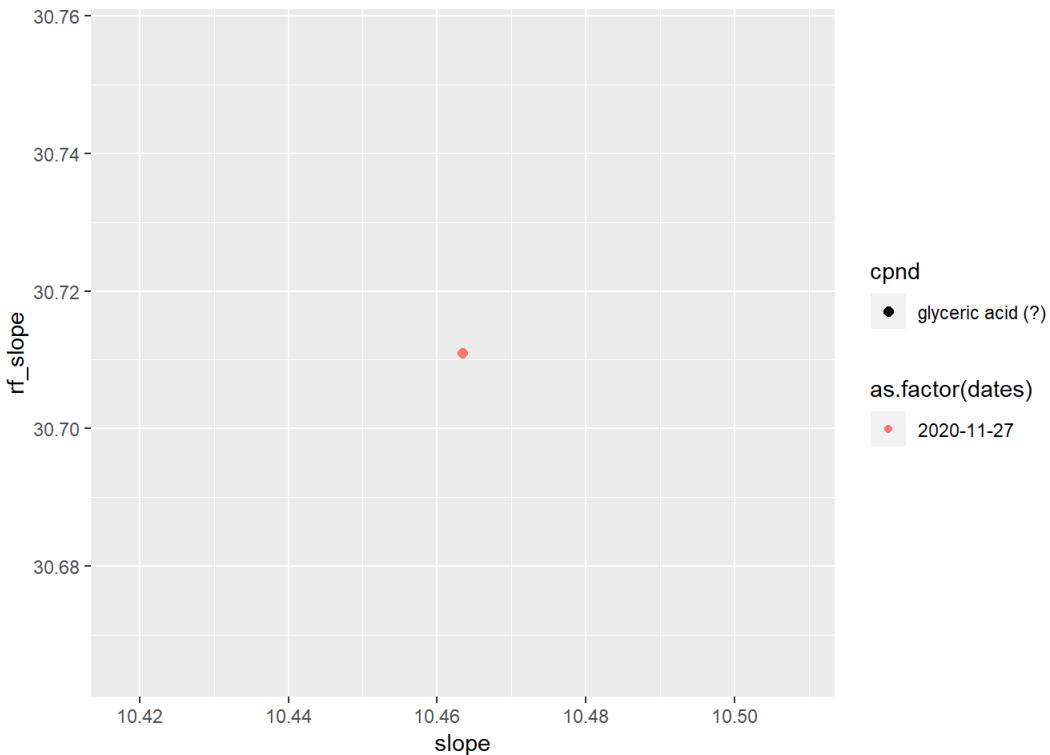
```
## Warning: Using size for a discrete variable is not advised.
```



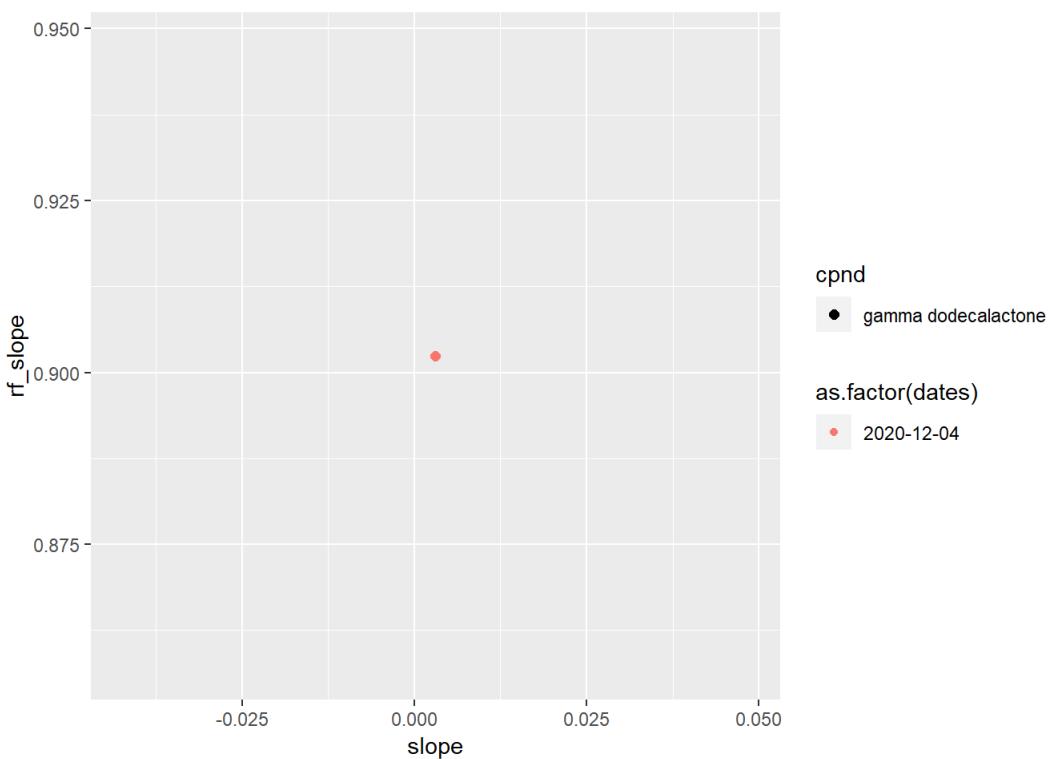
```
## Warning: Using size for a discrete variable is not advised.
```



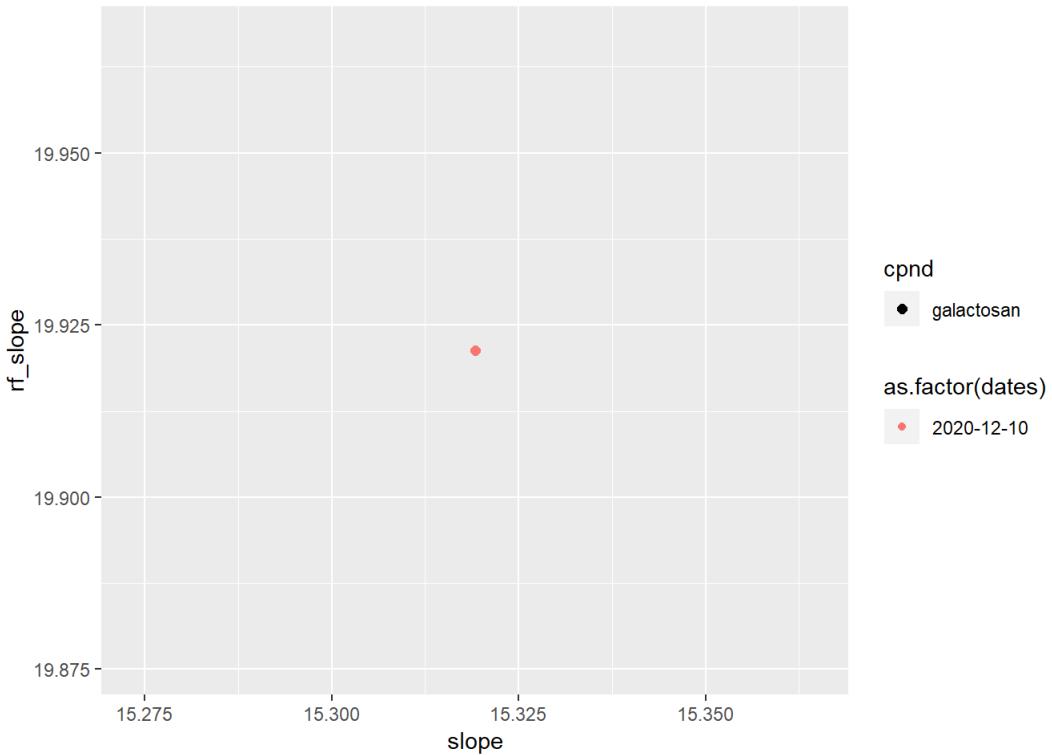
```
## Warning: Using size for a discrete variable is not advised.
```



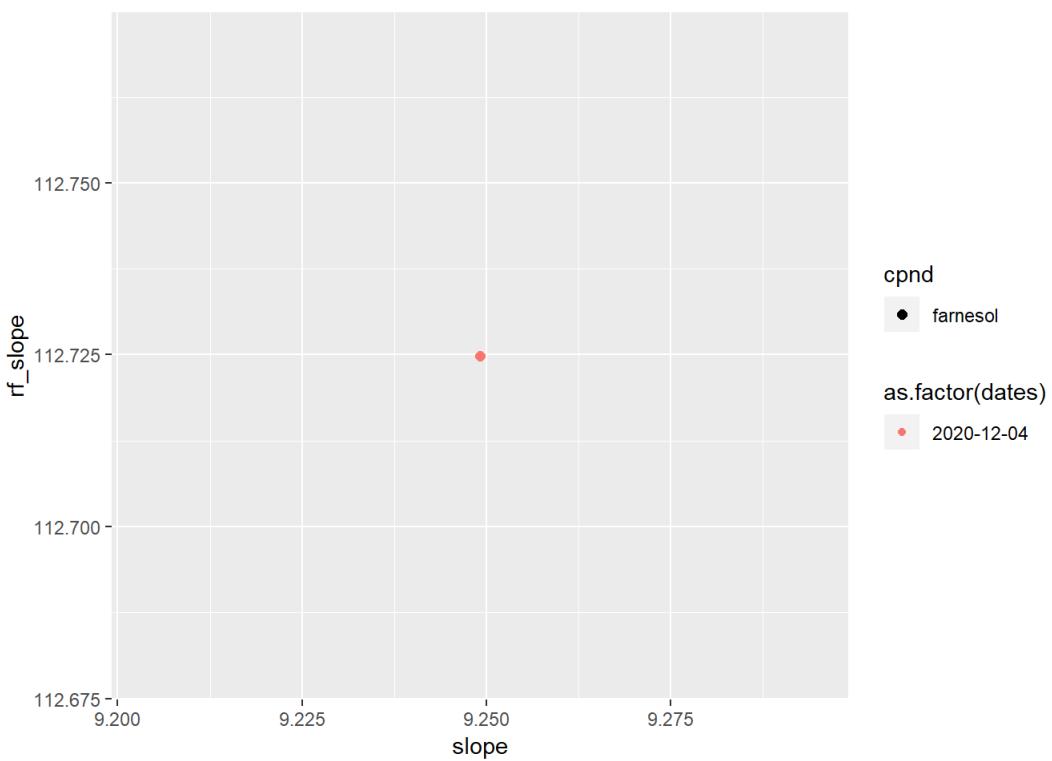
```
## Warning: Using size for a discrete variable is not advised.
```



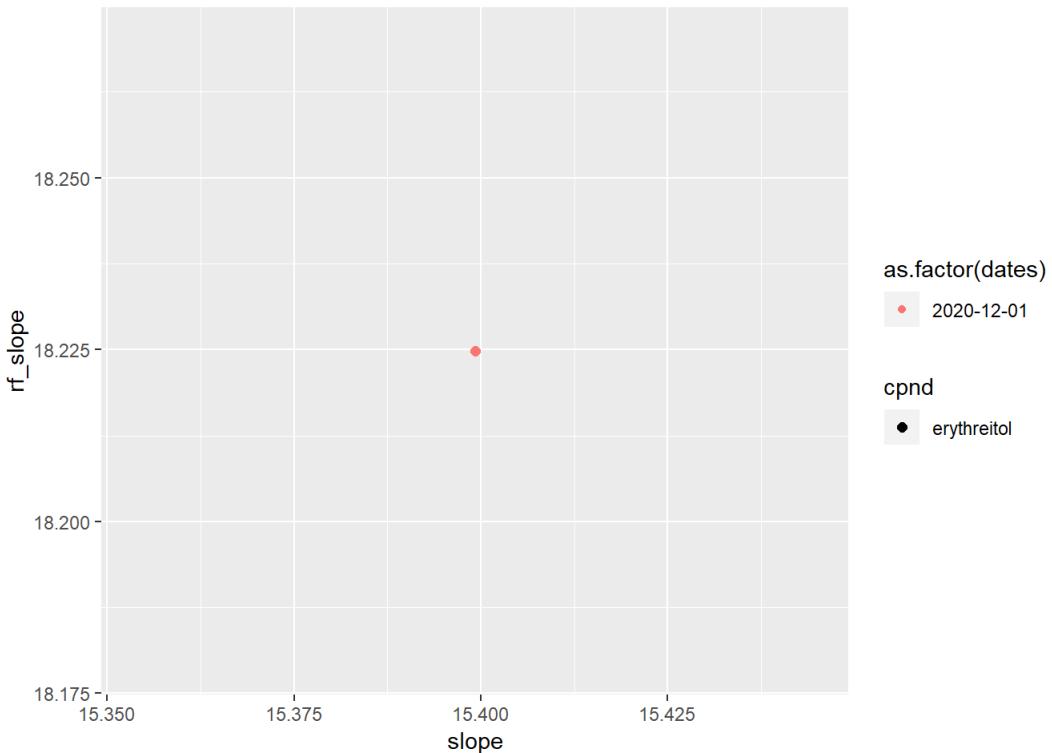
```
## Warning: Using size for a discrete variable is not advised.
```



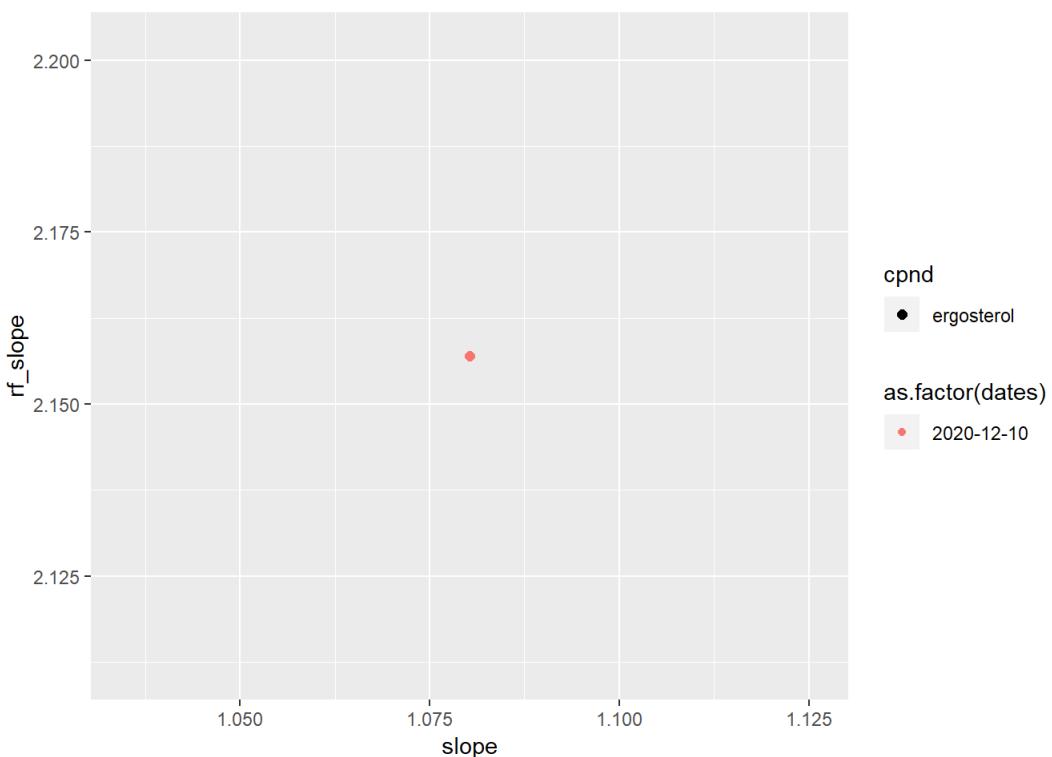
```
## Warning: Using size for a discrete variable is not advised.
```



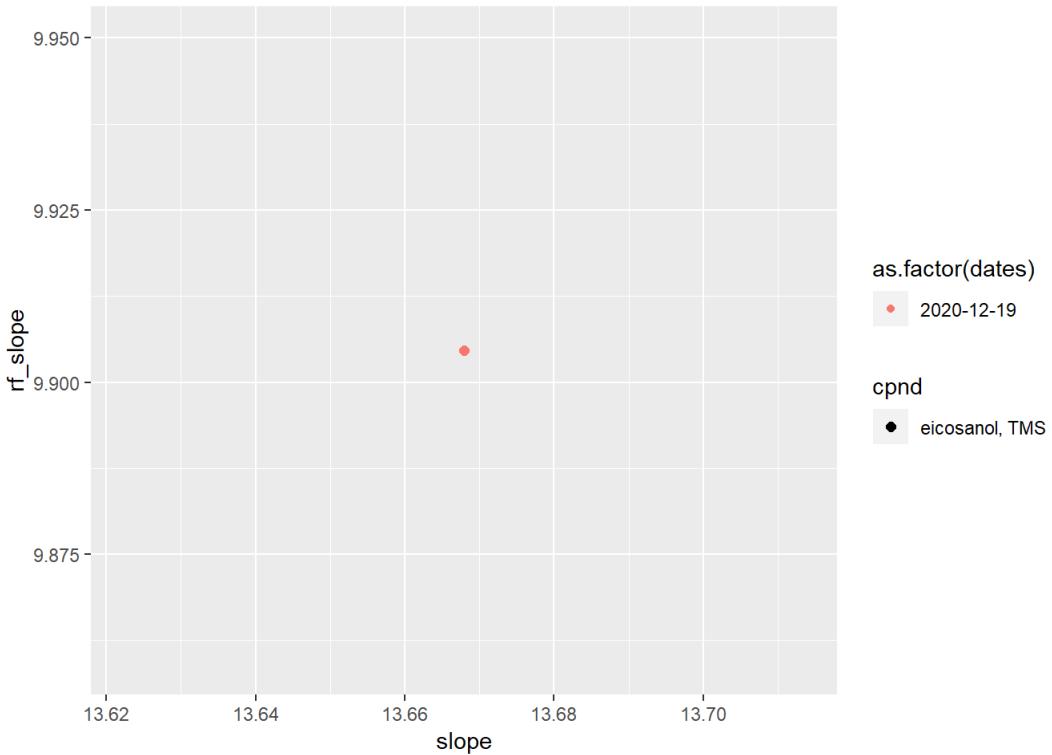
```
## Warning: Using size for a discrete variable is not advised.
```



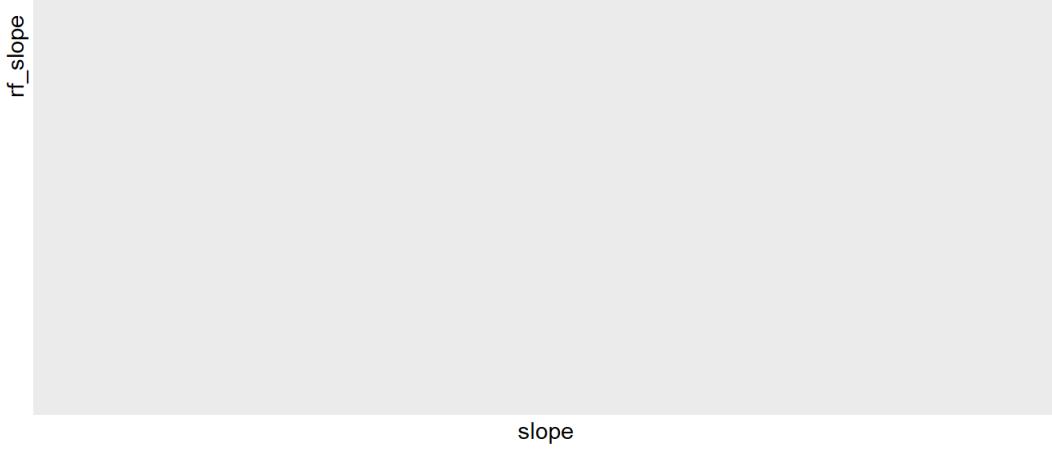
```
## Warning: Using size for a discrete variable is not advised.
```



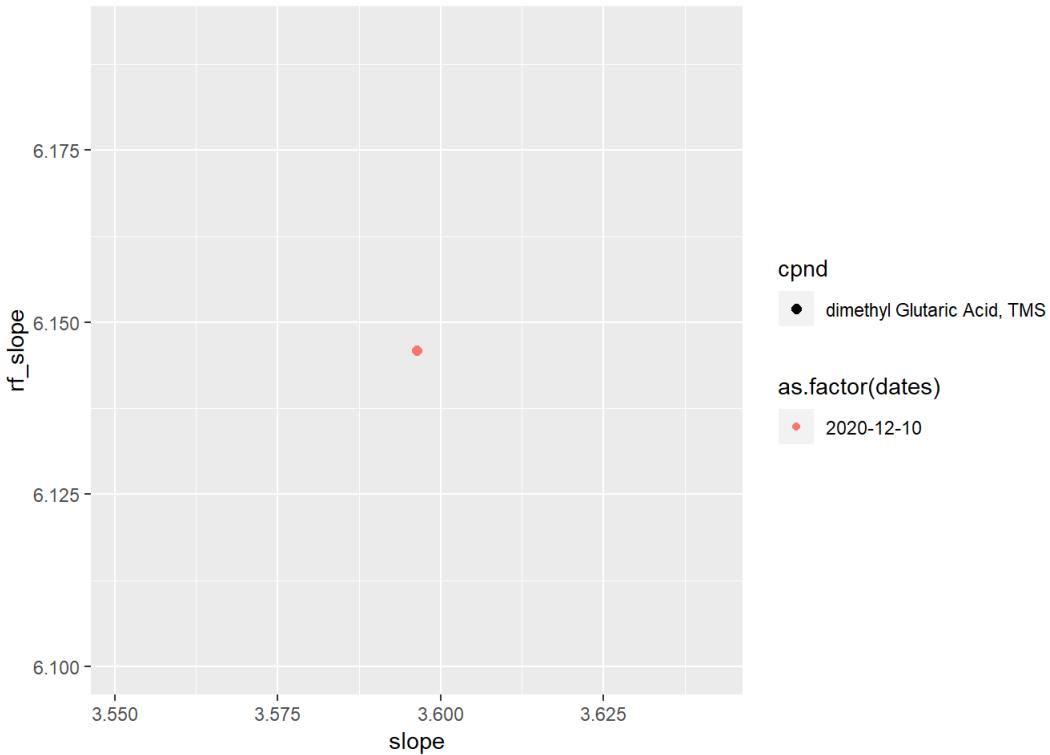
```
## Warning: Using size for a discrete variable is not advised.
```



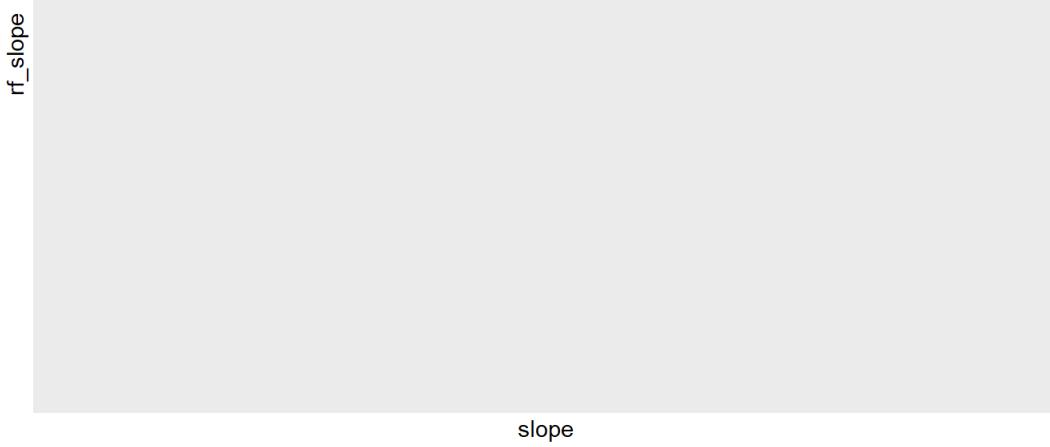
```
## Warning: Using size for a discrete variable is not advised.
```



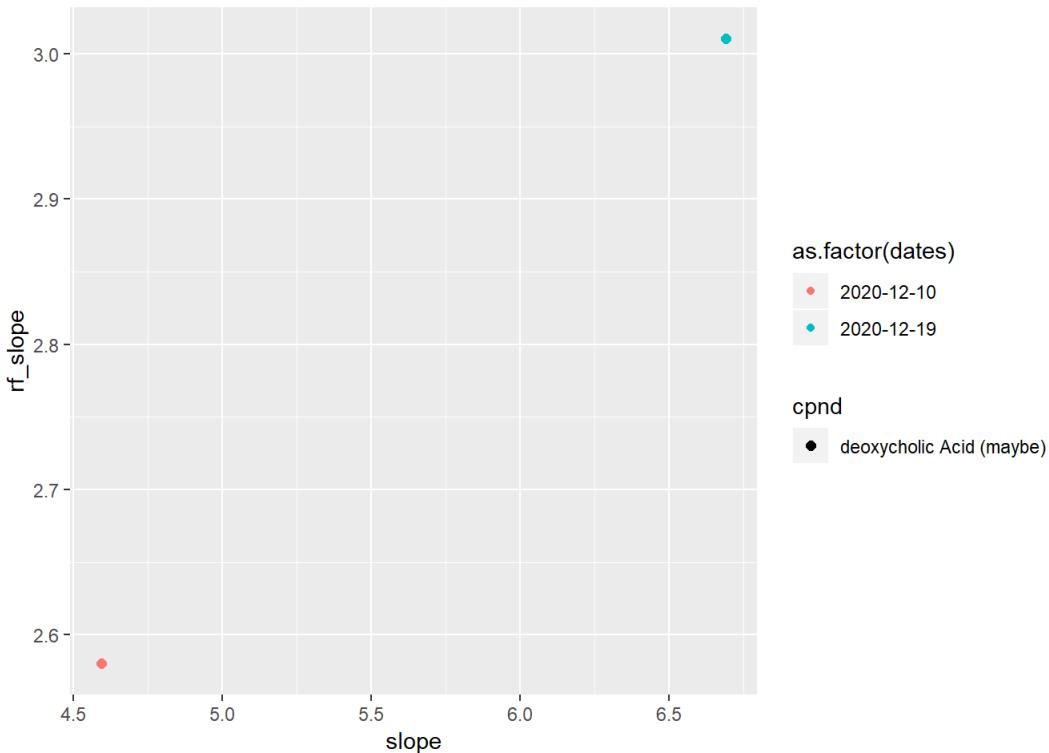
```
## Warning: Using size for a discrete variable is not advised.
```



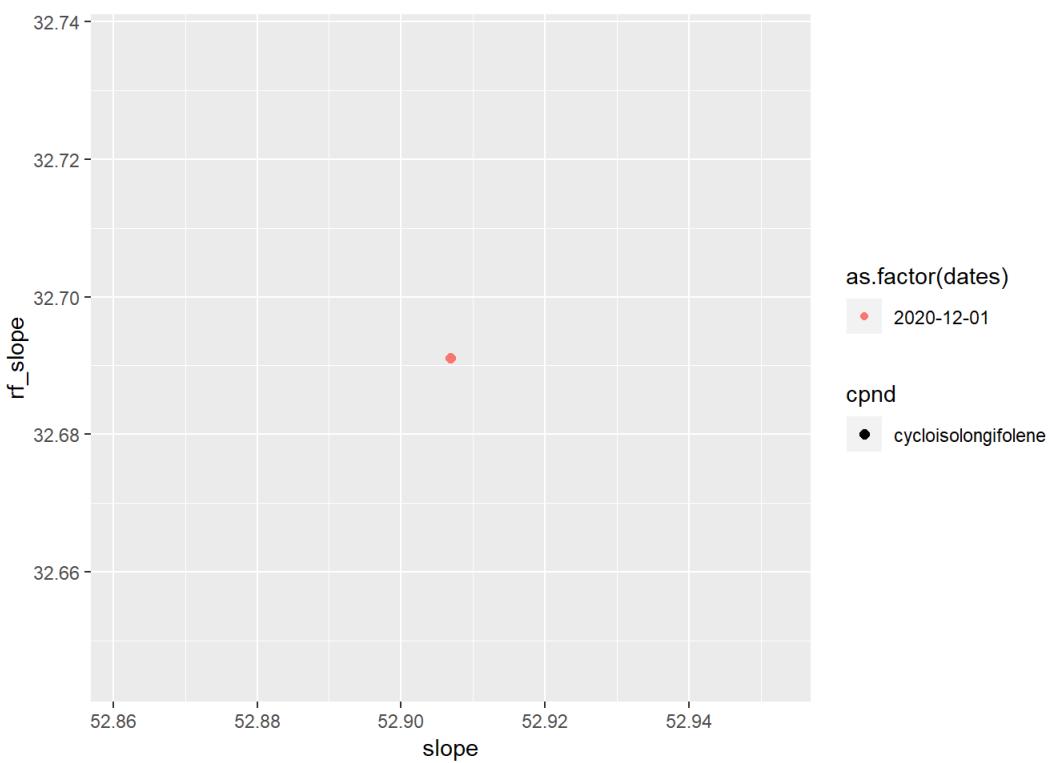
```
## Warning: Using size for a discrete variable is not advised.
```



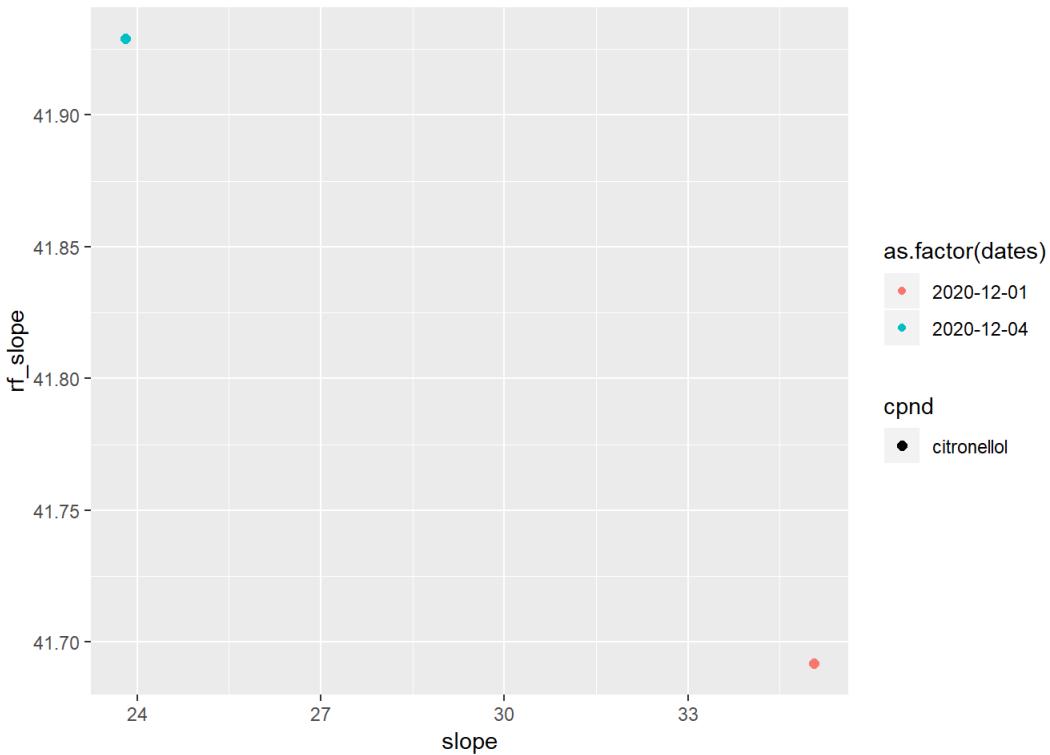
```
## Warning: Using size for a discrete variable is not advised.
```



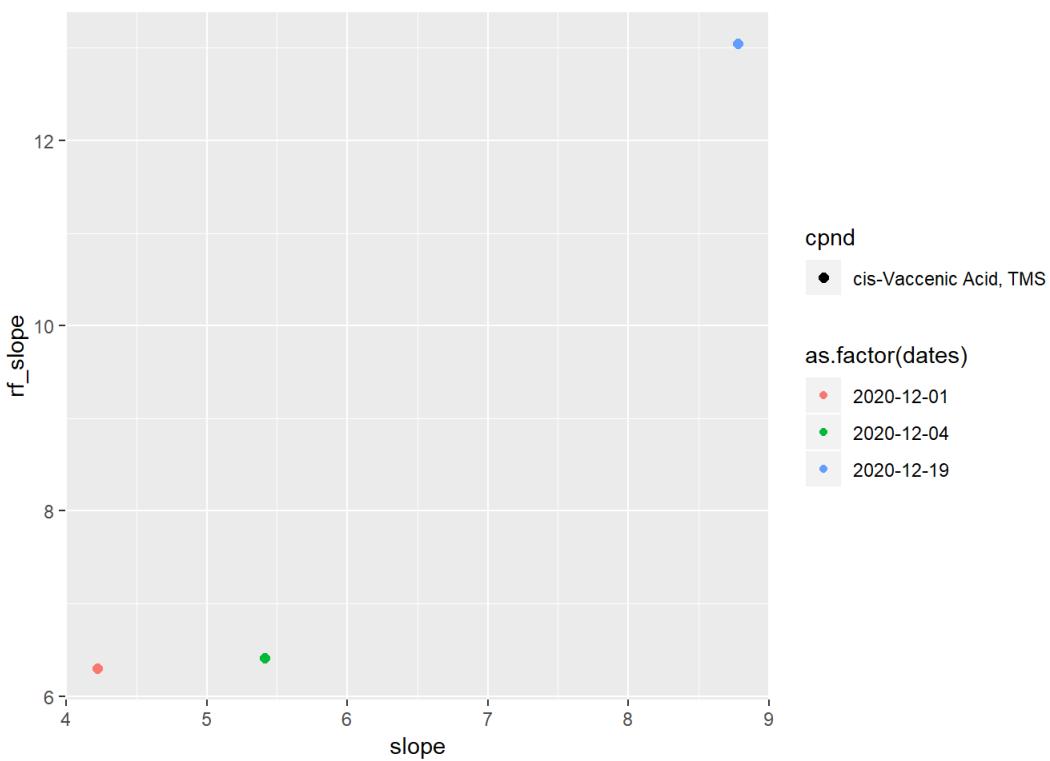
```
## Warning: Using size for a discrete variable is not advised.
```



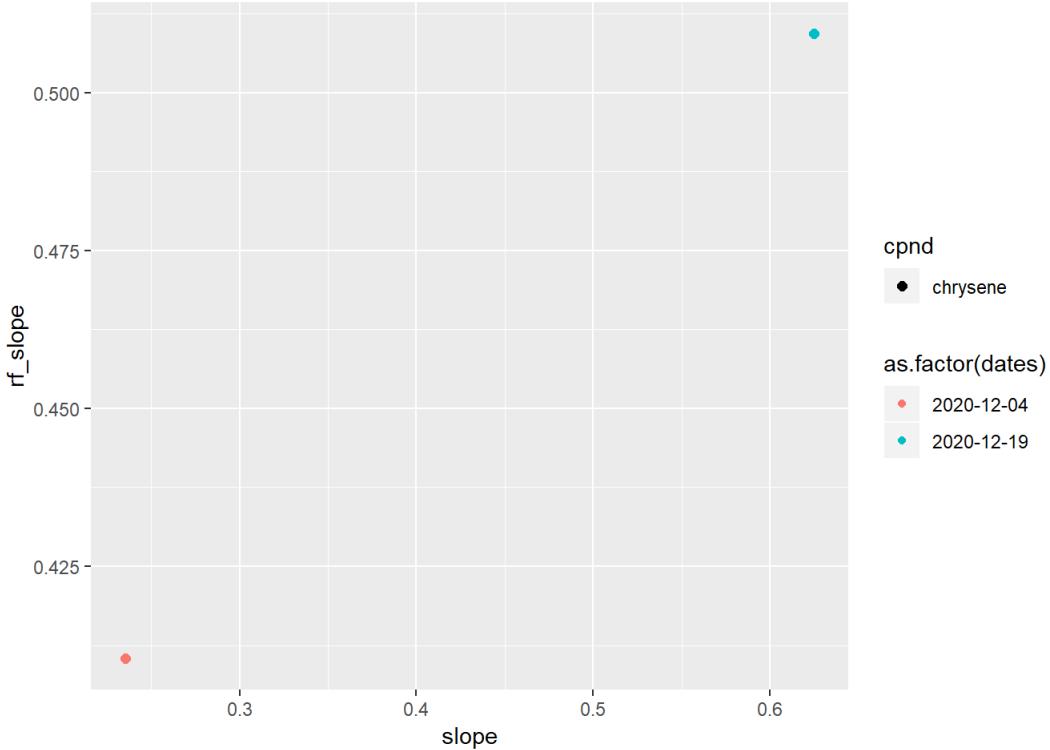
```
## Warning: Using size for a discrete variable is not advised.
```



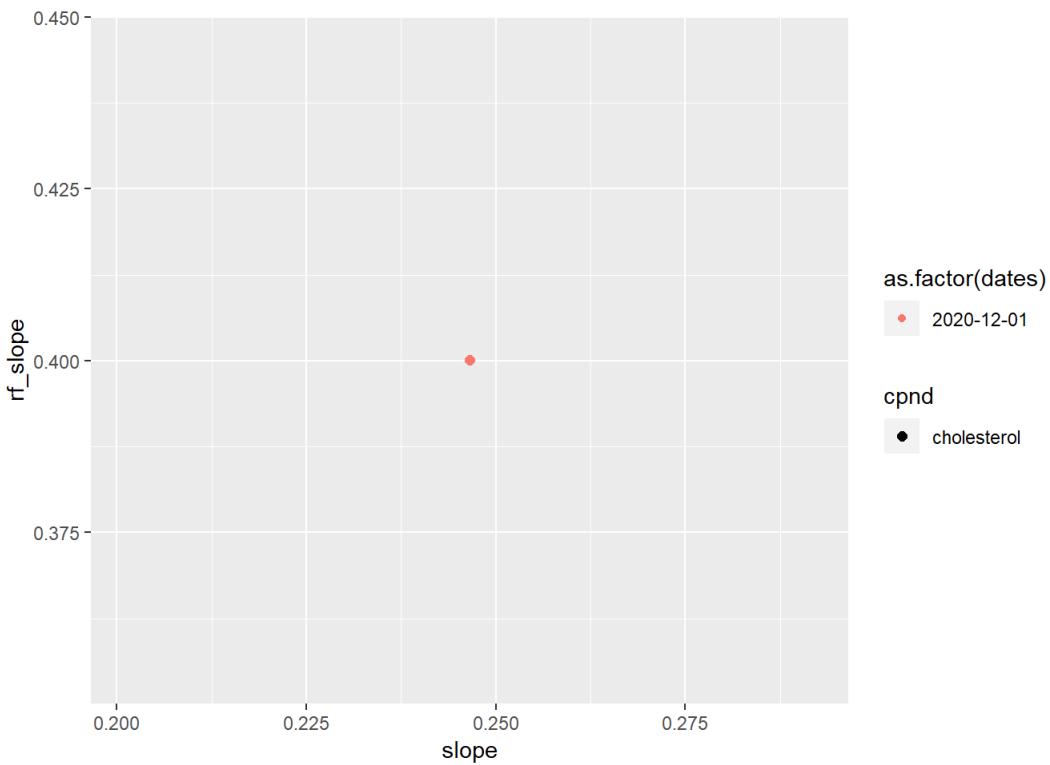
```
## Warning: Using size for a discrete variable is not advised.
```



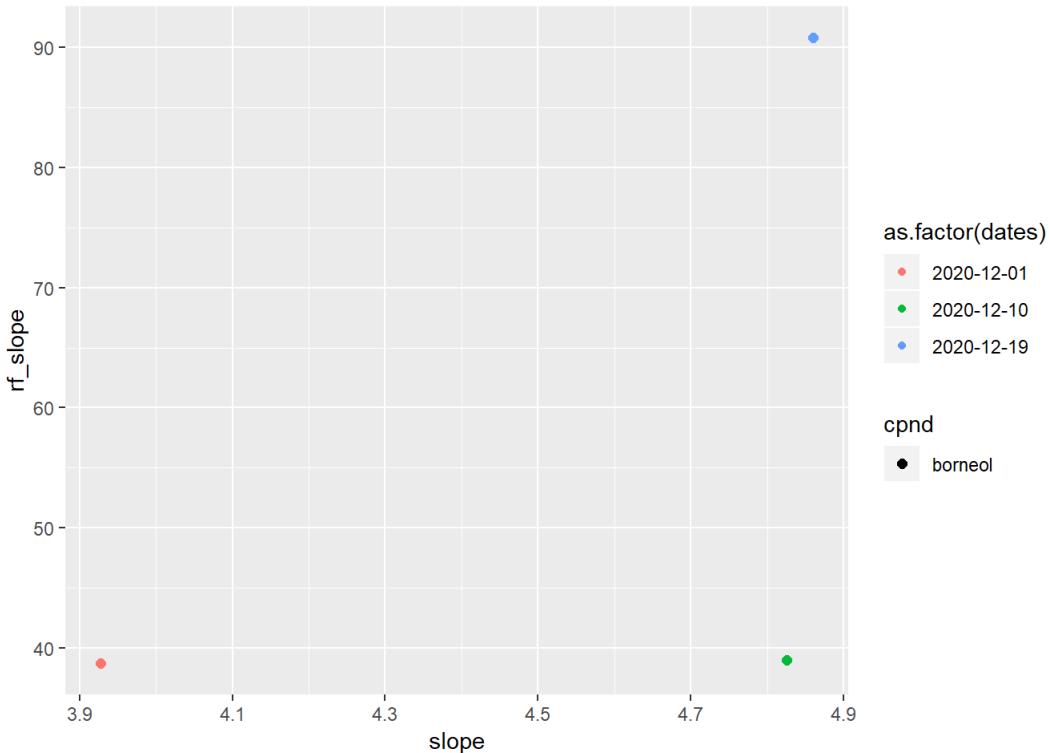
```
## Warning: Using size for a discrete variable is not advised.
```



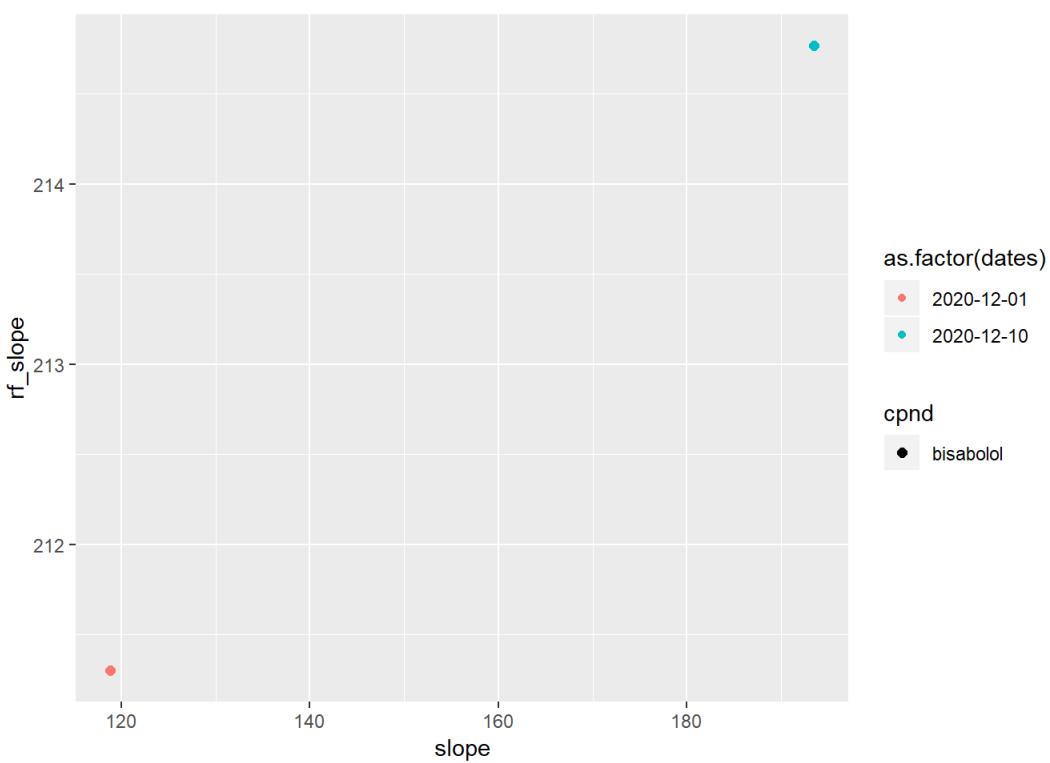
```
## Warning: Using size for a discrete variable is not advised.
```



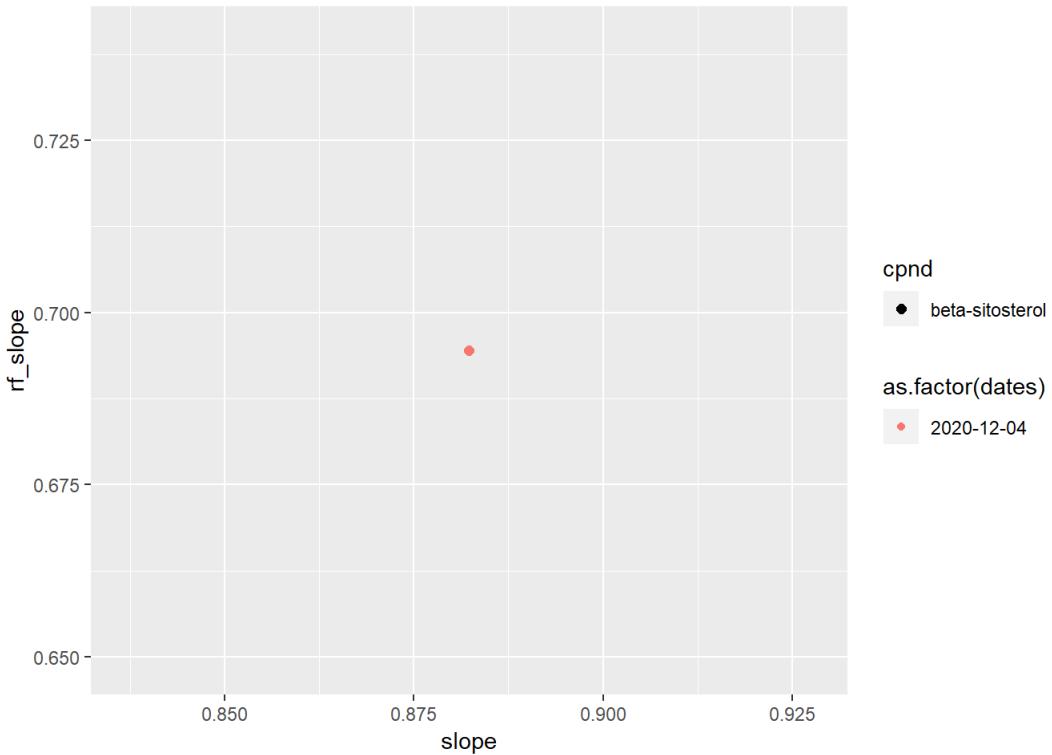
```
## Warning: Using size for a discrete variable is not advised.
```



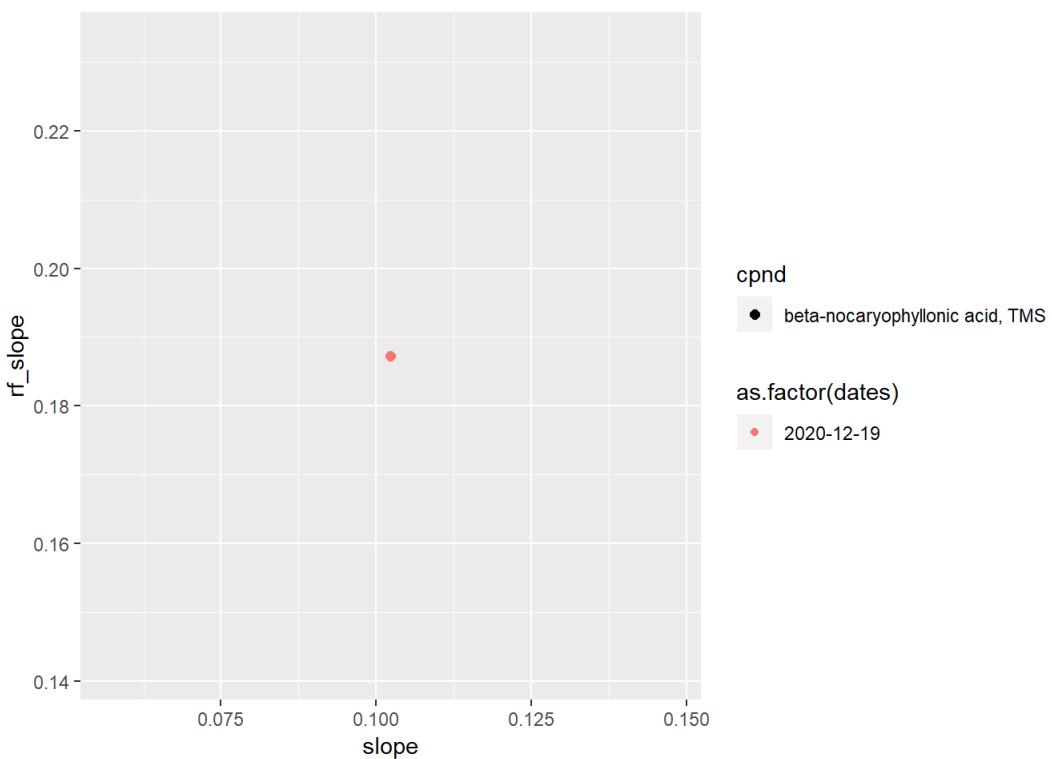
```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



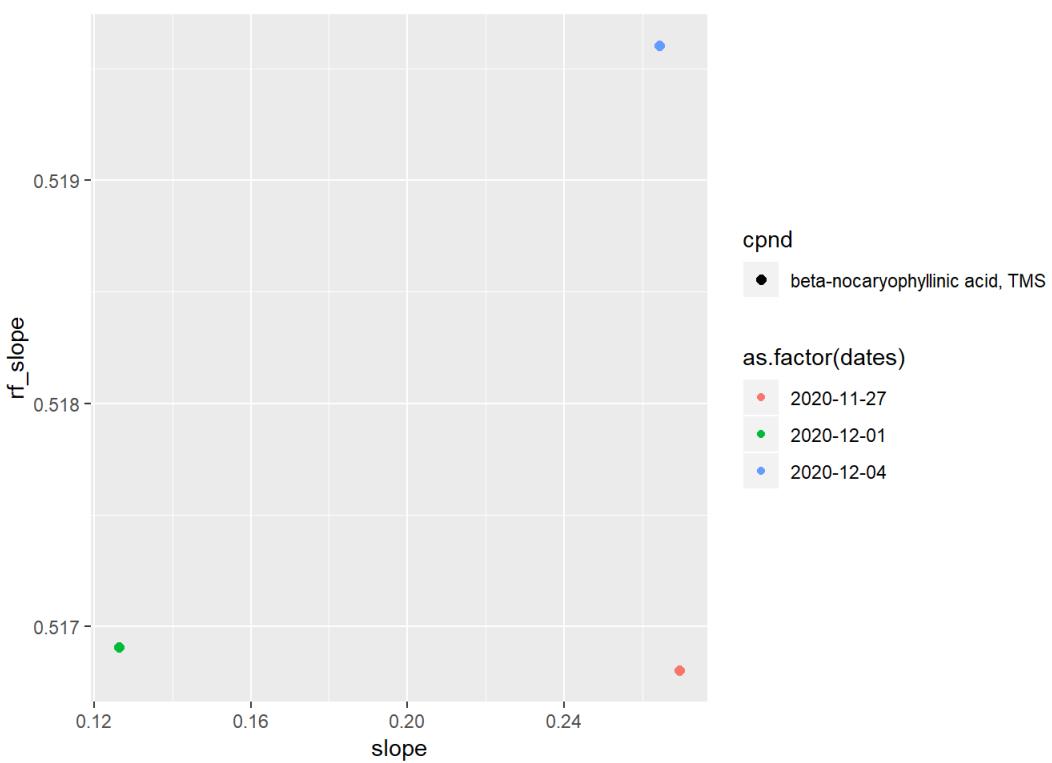
```
## Warning: Using size for a discrete variable is not advised.
```

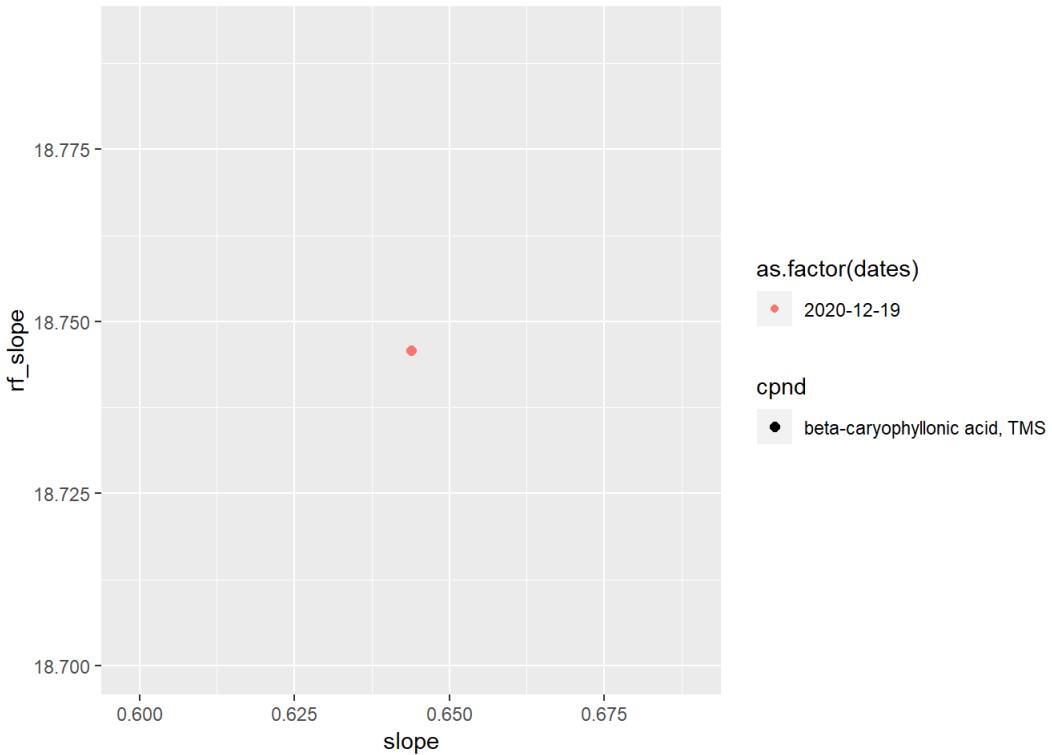


```
## Warning: Using size for a discrete variable is not advised.
```

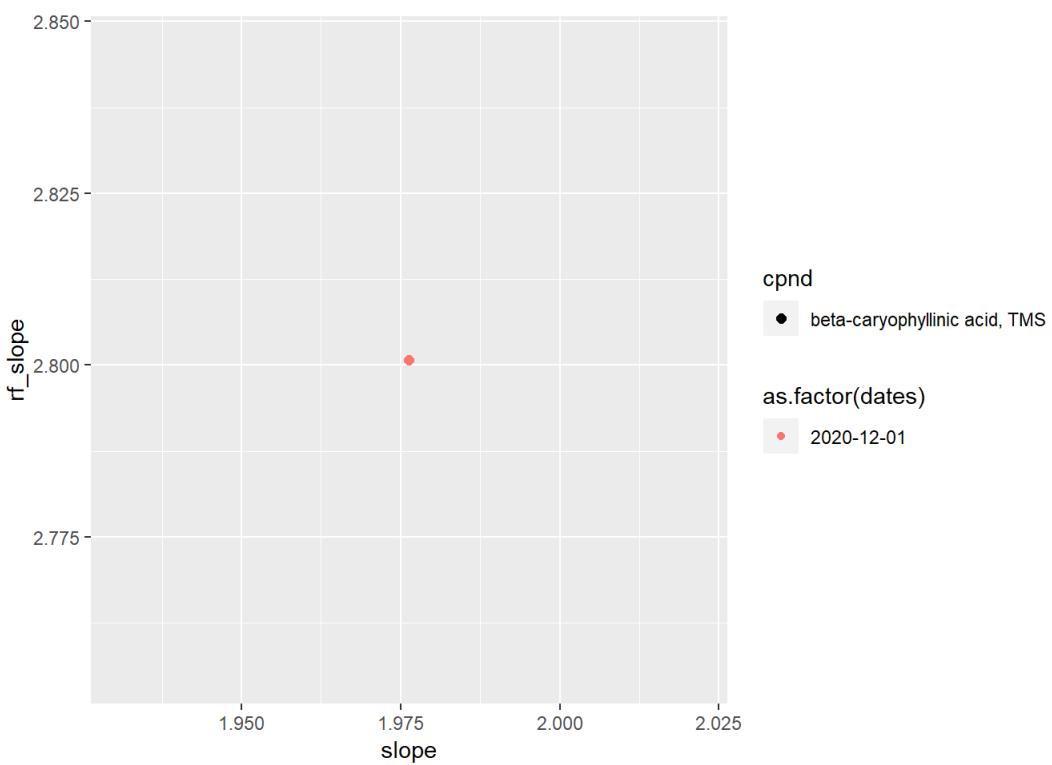


```
## Warning: Using size for a discrete variable is not advised.
```





```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```

rf_slope

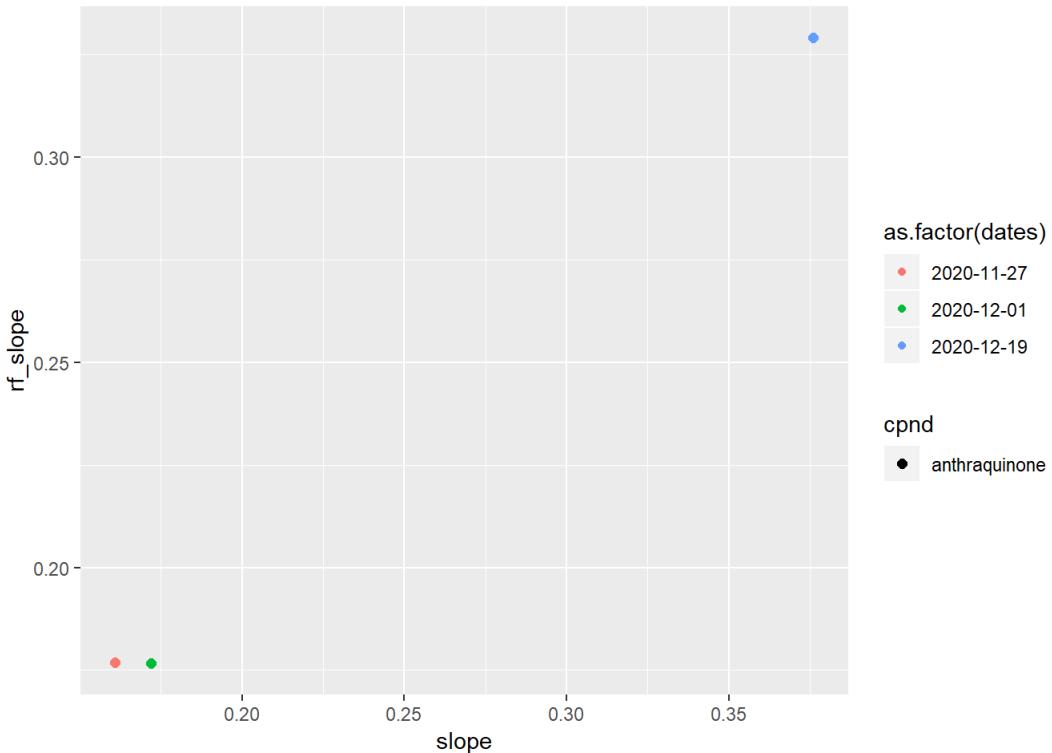
slope

```
## Warning: Using size for a discrete variable is not advised.
```

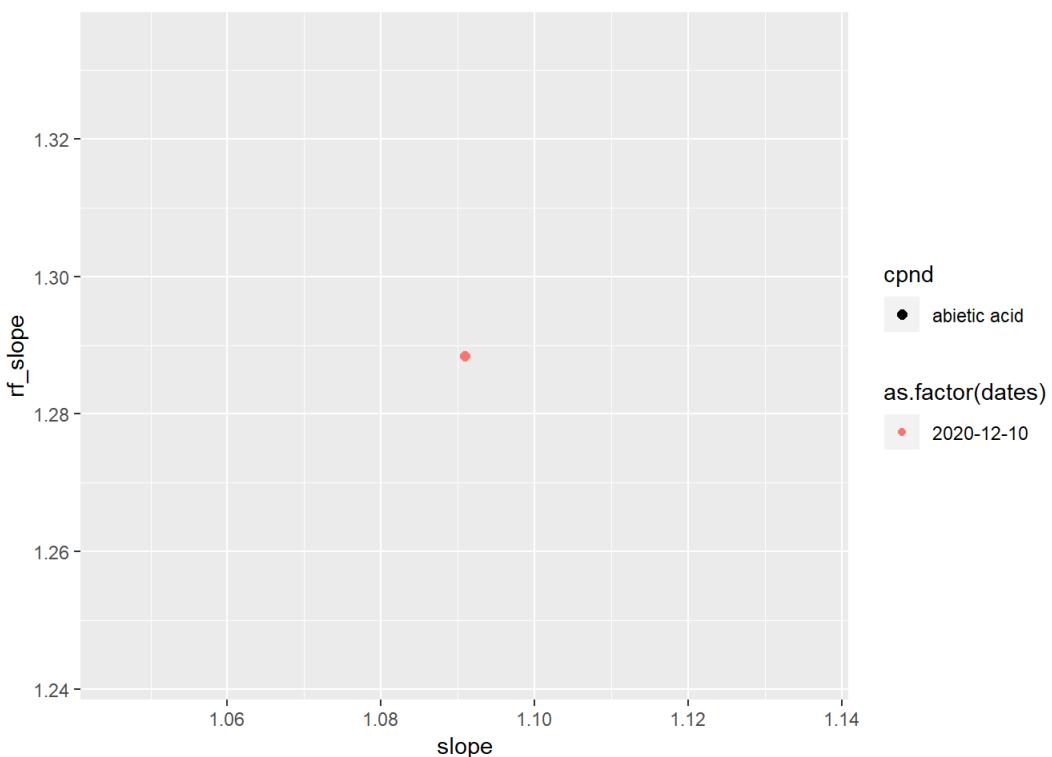
rf_slope

slope

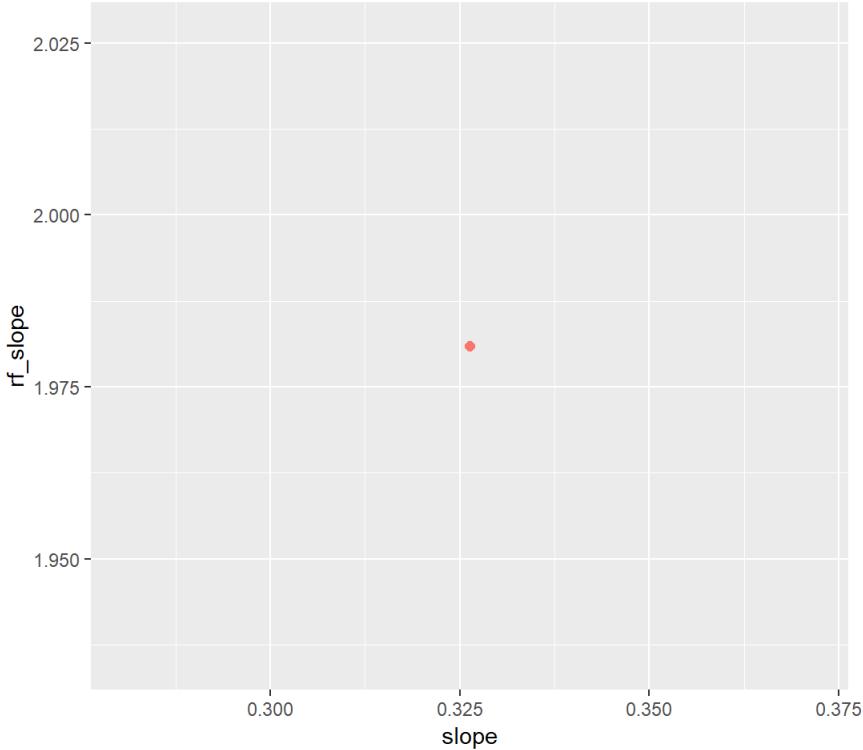
```
## Warning: Using size for a discrete variable is not advised.
```



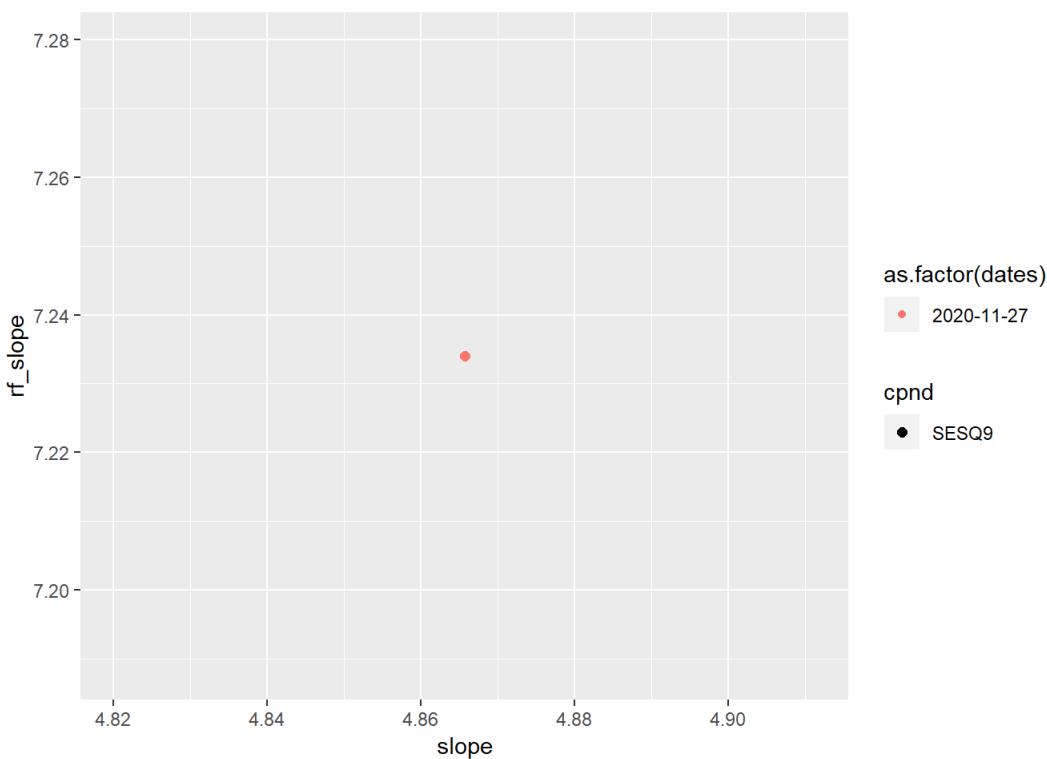
```
## Warning: Using size for a discrete variable is not advised.
```



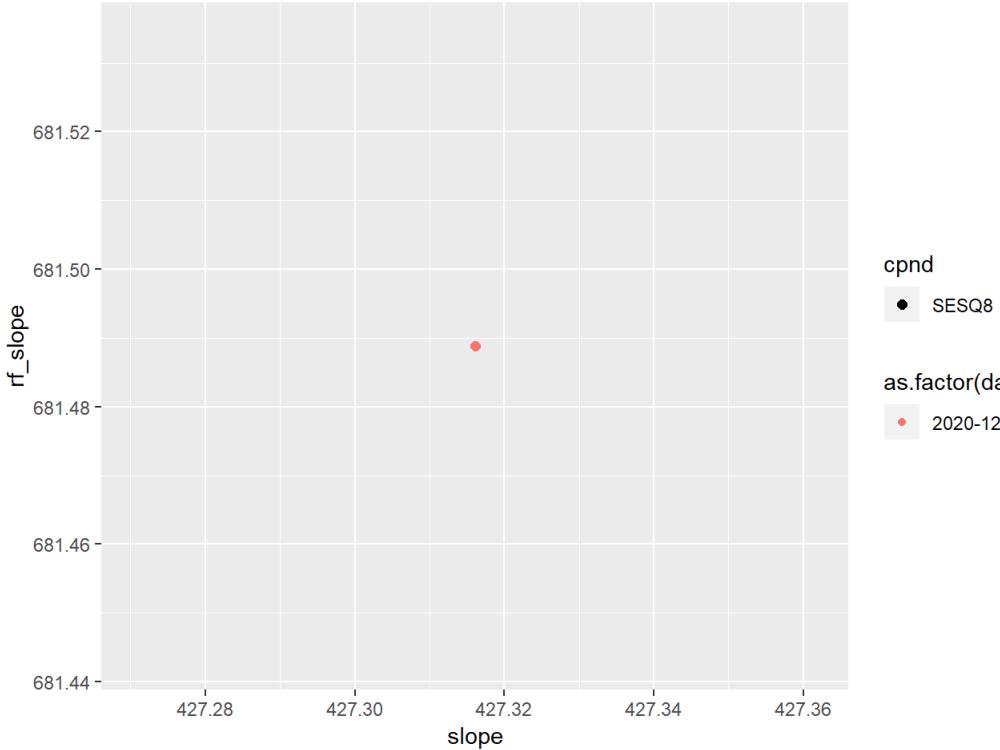
```
## Warning: Using size for a discrete variable is not advised.
```



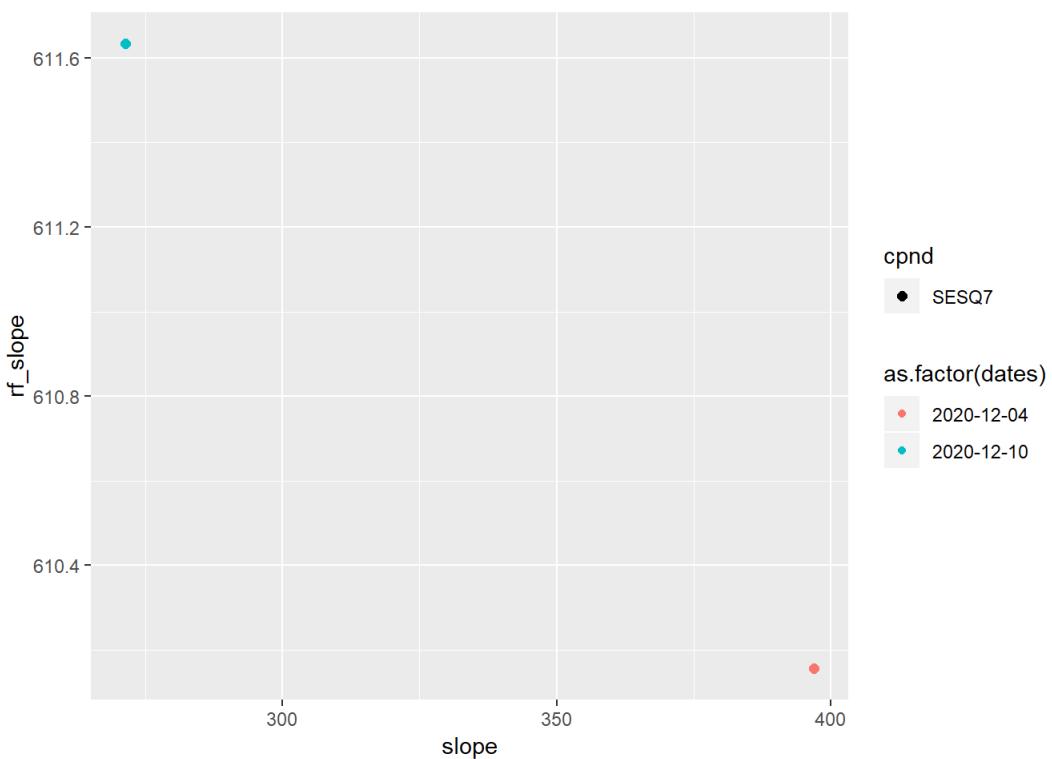
```
## Warning: Using size for a discrete variable is not advised.
```



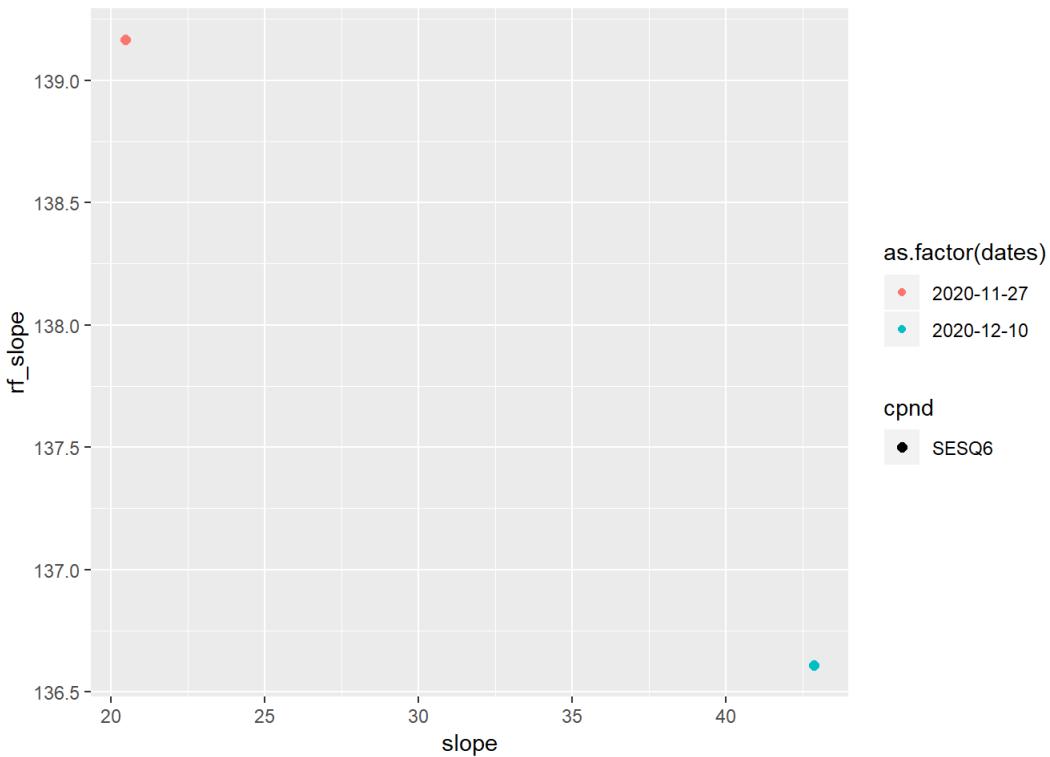
```
## Warning: Using size for a discrete variable is not advised.
```



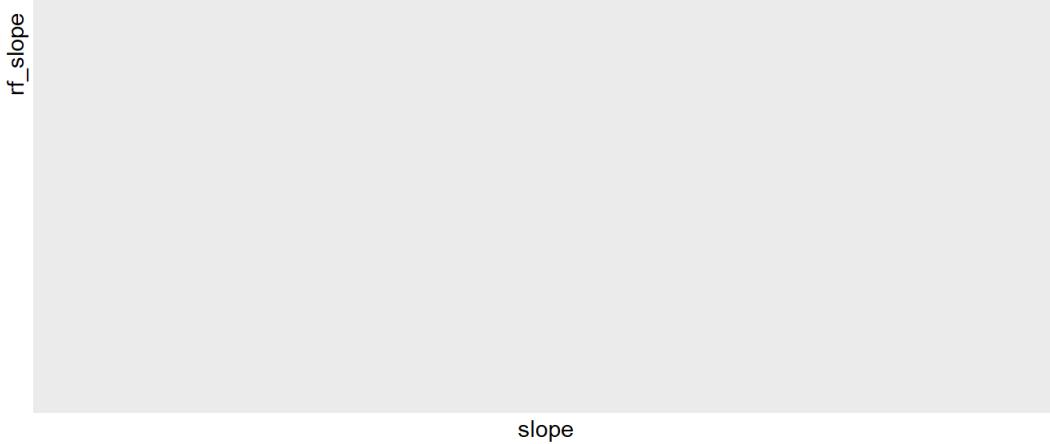
```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```

rf_slope

slope

```
## Warning: Using size for a discrete variable is not advised.
```

rf_slope

slope

```
## Warning: Using size for a discrete variable is not advised.
```

rf_slope

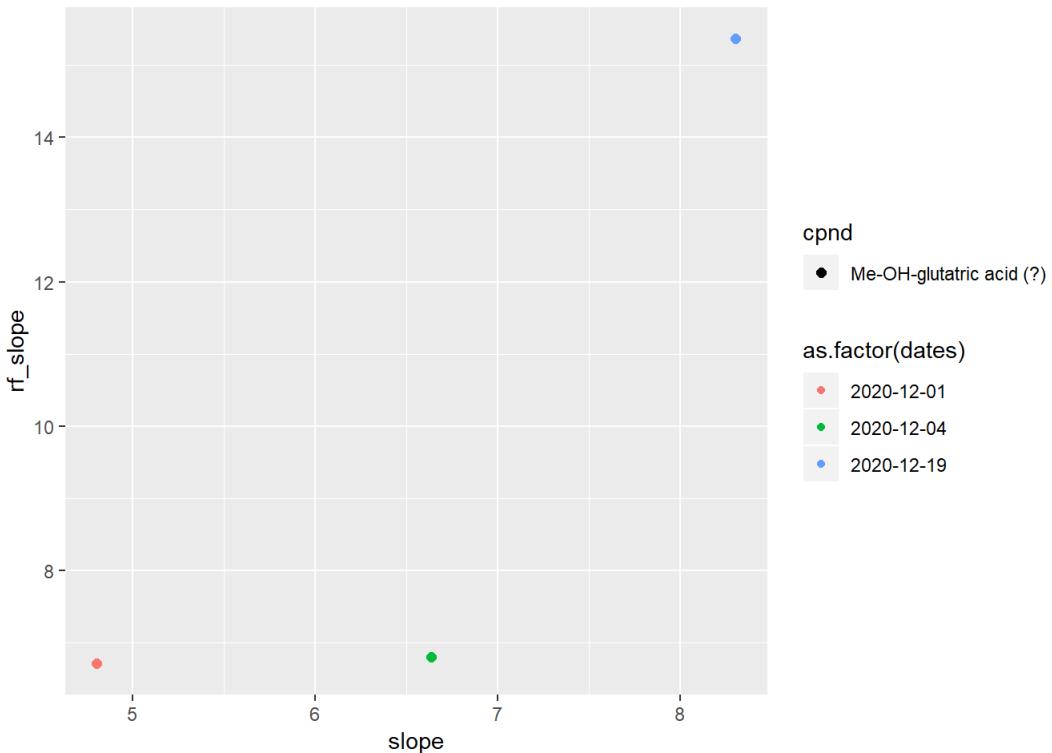
slope

```
## Warning: Using size for a discrete variable is not advised.
```

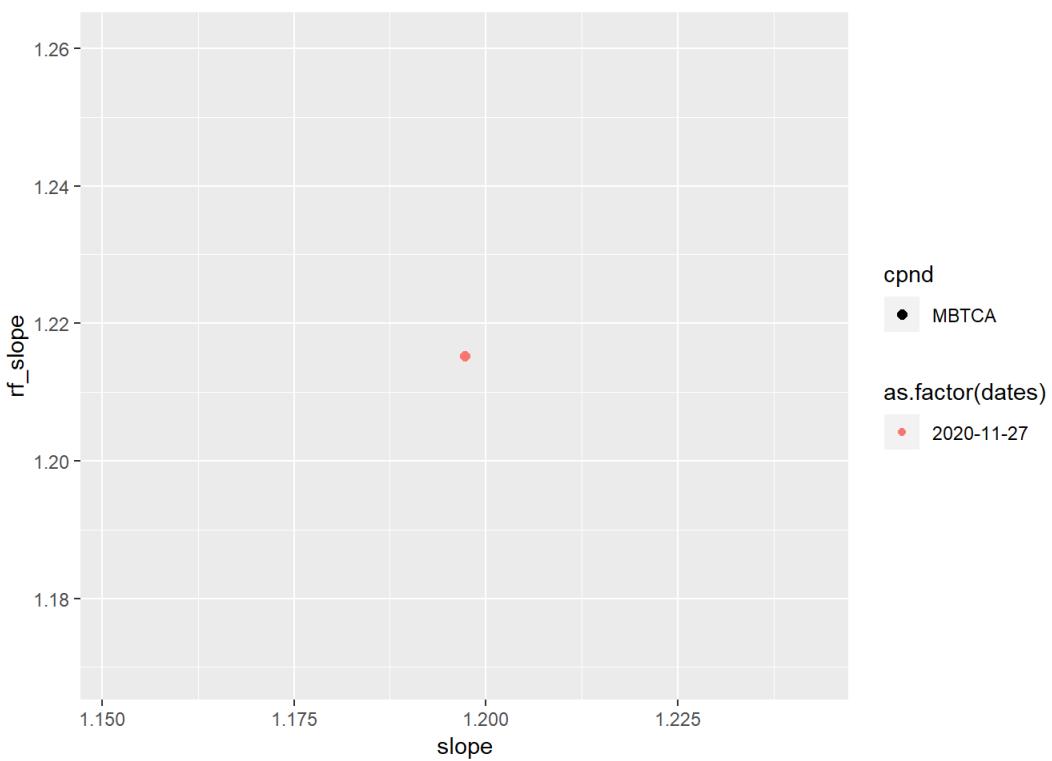
rf_slope

slope

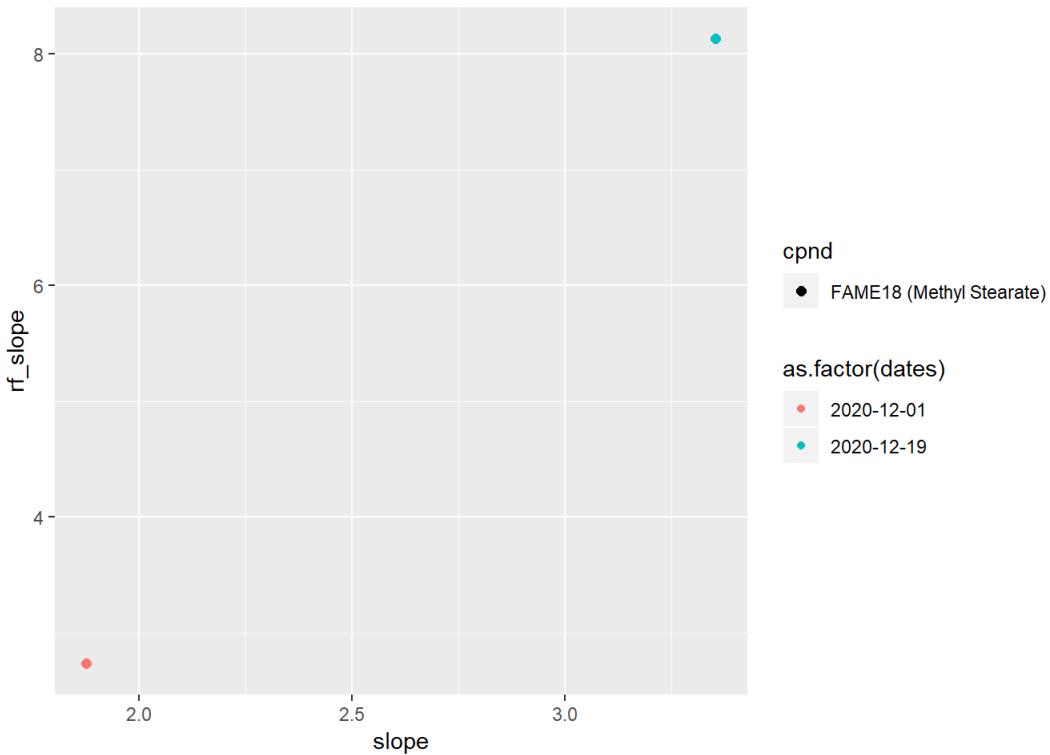
```
## Warning: Using size for a discrete variable is not advised.
```



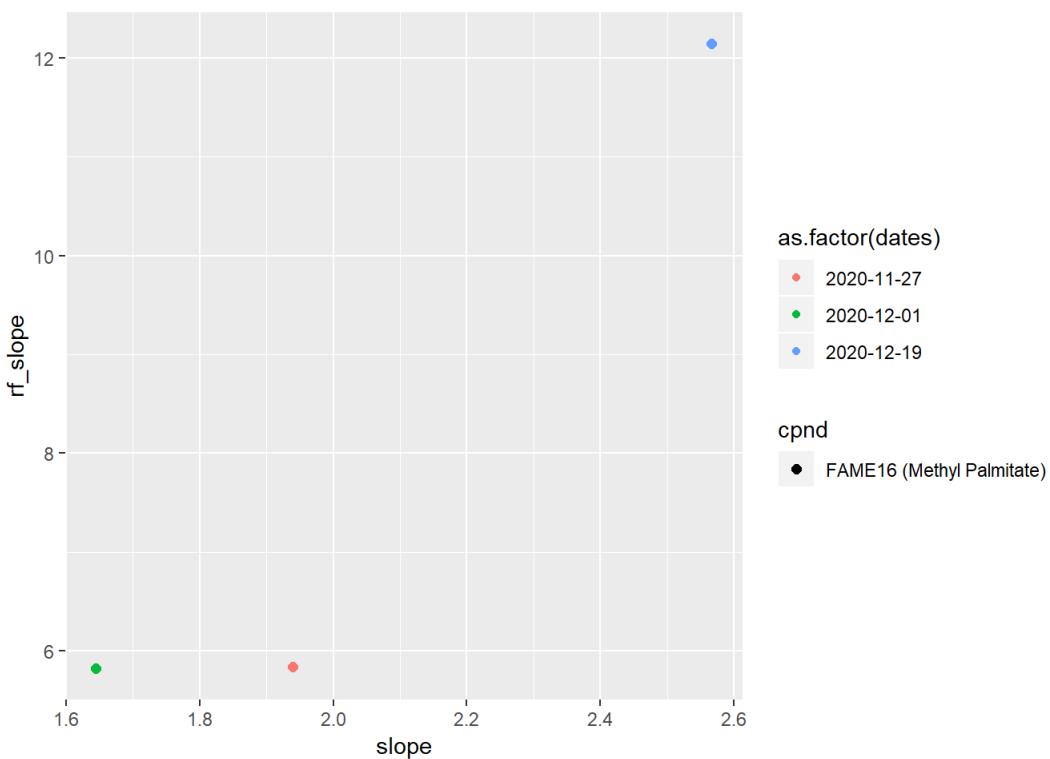
```
## Warning: Using size for a discrete variable is not advised.
```



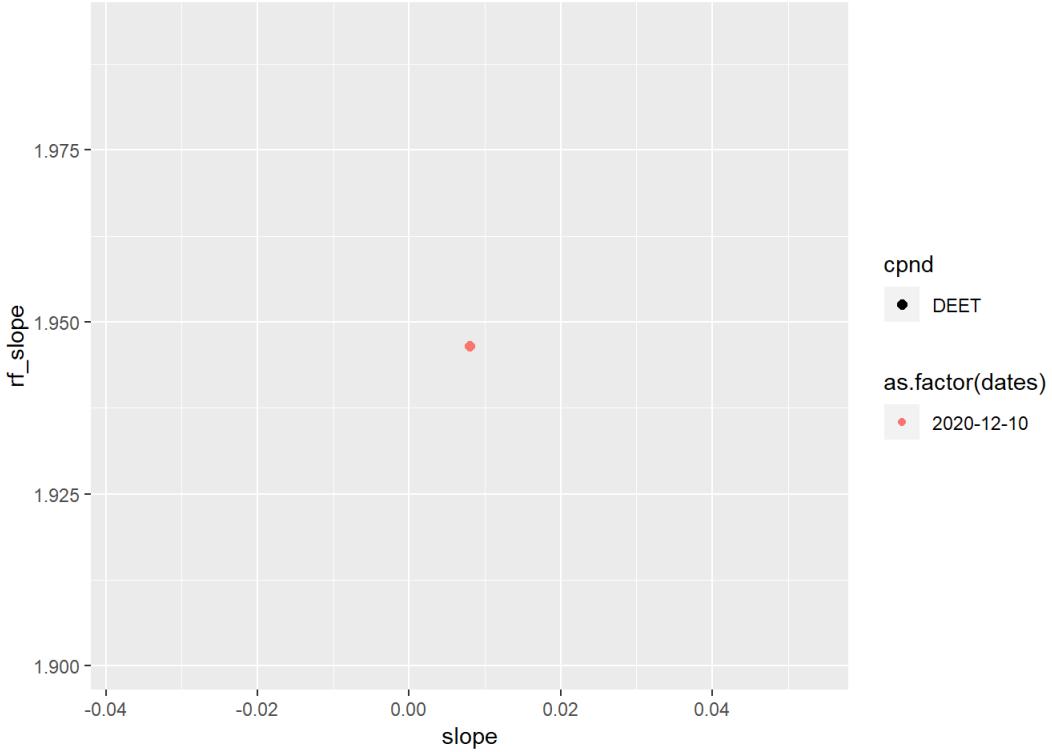
```
## Warning: Using size for a discrete variable is not advised.
```



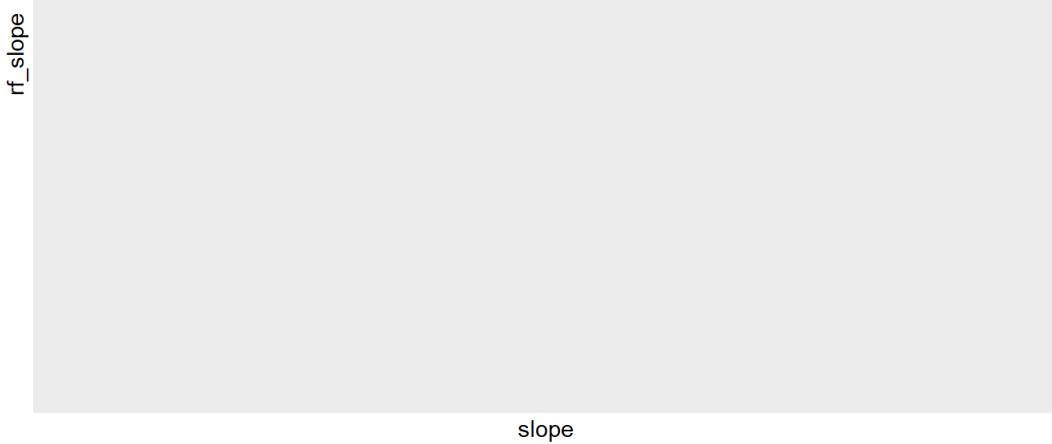
```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```

rf_slope

slope

```
## Warning: Using size for a discrete variable is not advised.
```

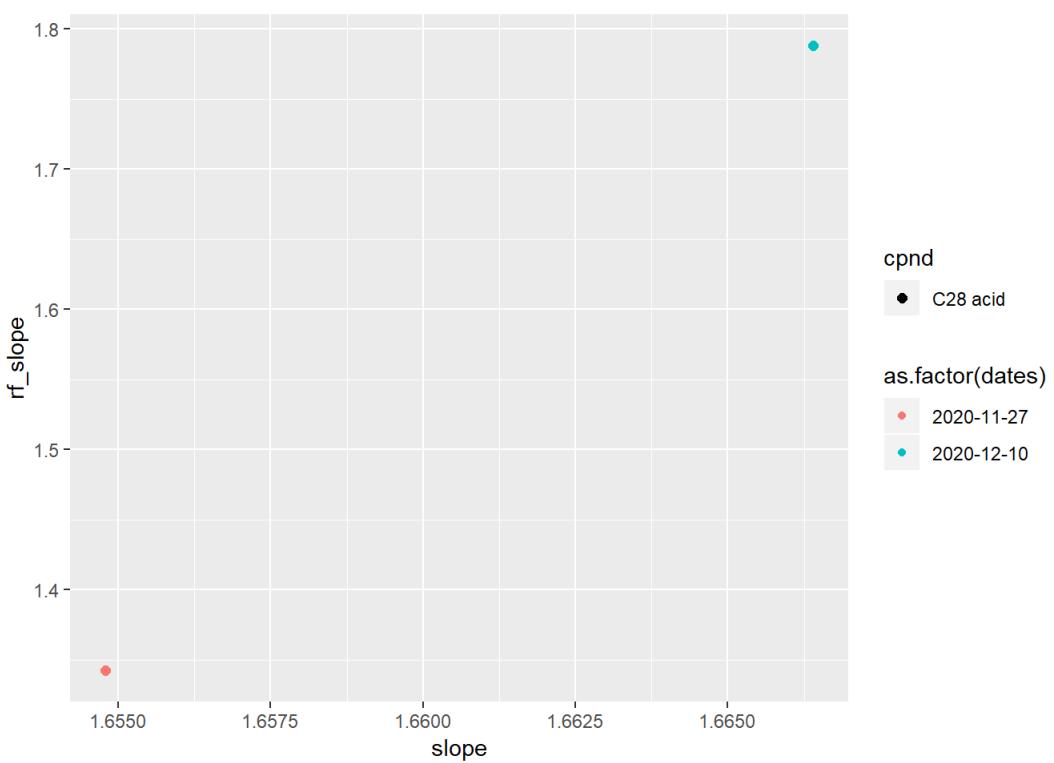
rf_slope

slope

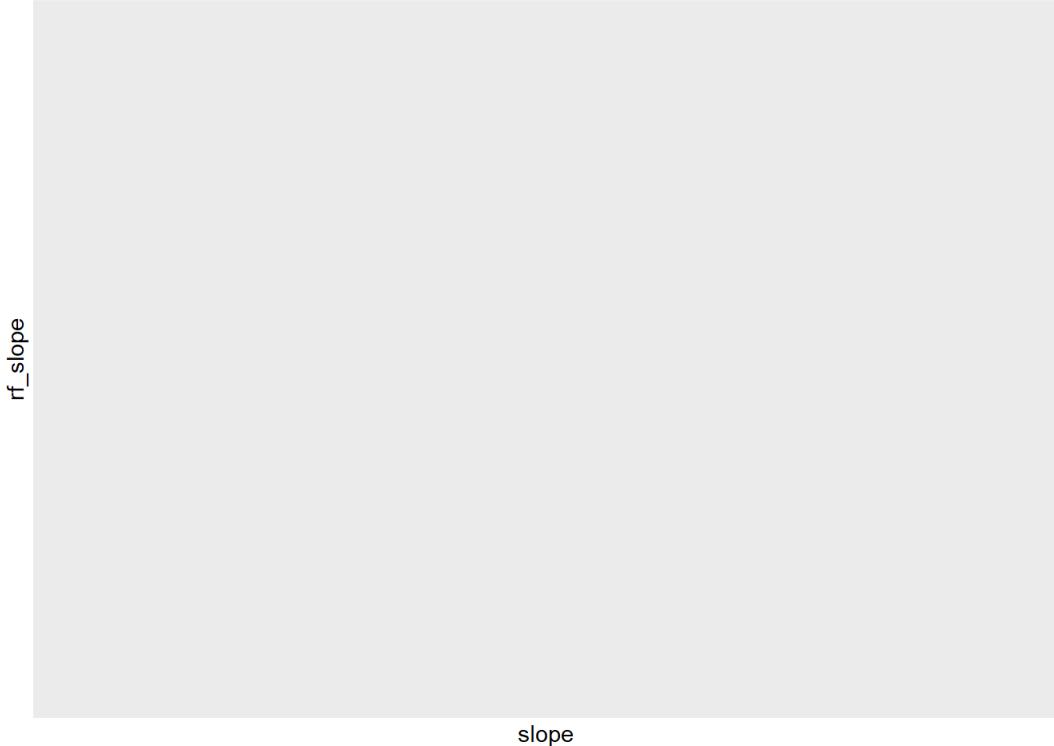
```
## Warning: Using size for a discrete variable is not advised.
```



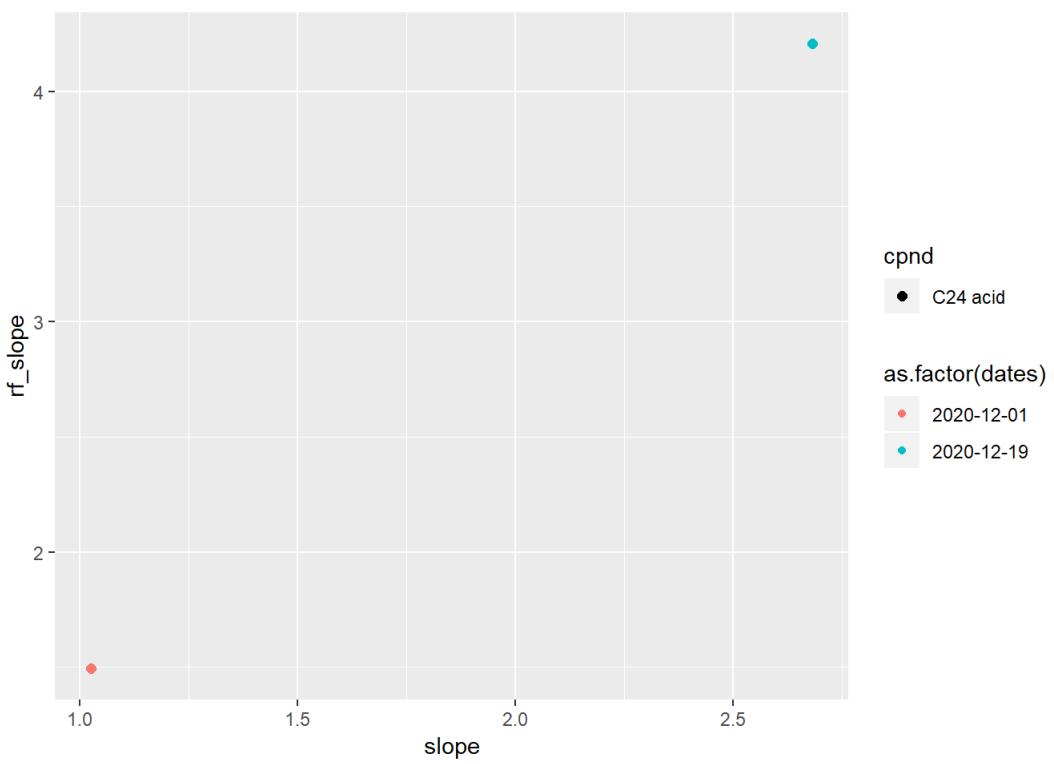
```
## Warning: Using size for a discrete variable is not advised.
```



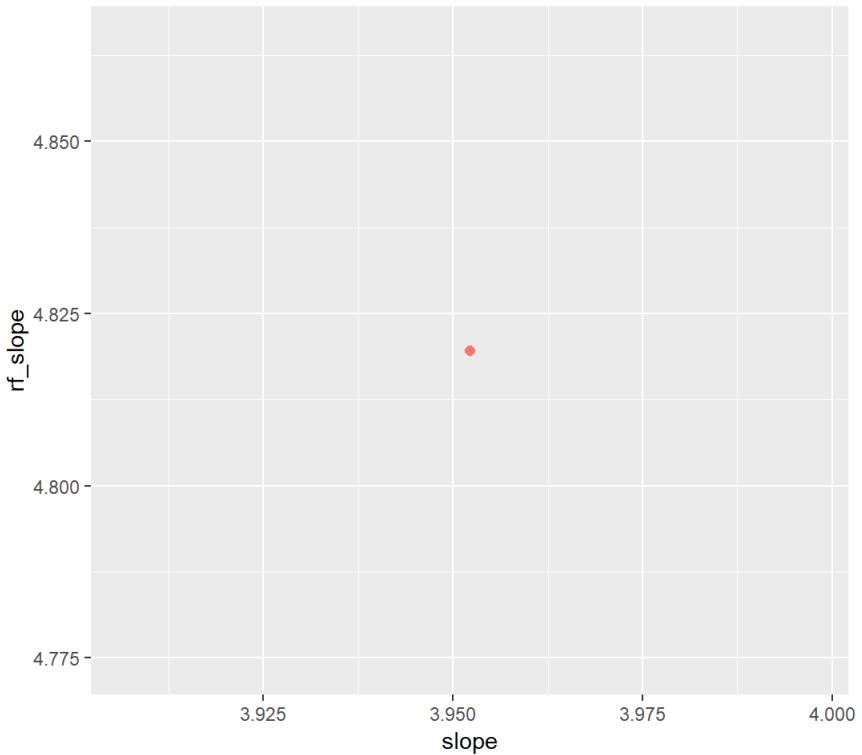
```
## Warning: Using size for a discrete variable is not advised.
```



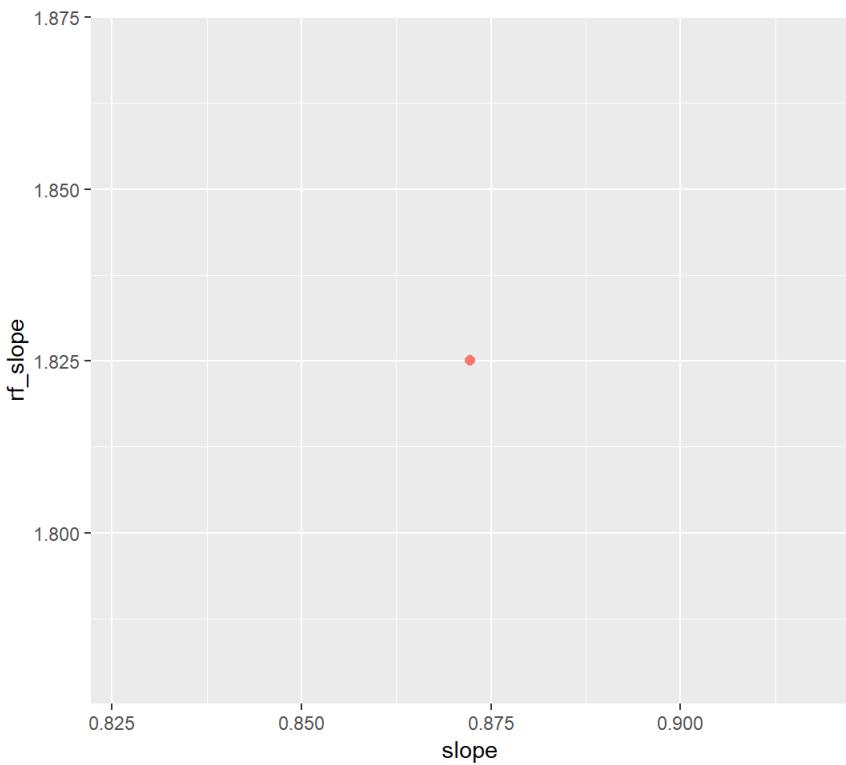
```
## Warning: Using size for a discrete variable is not advised.
```



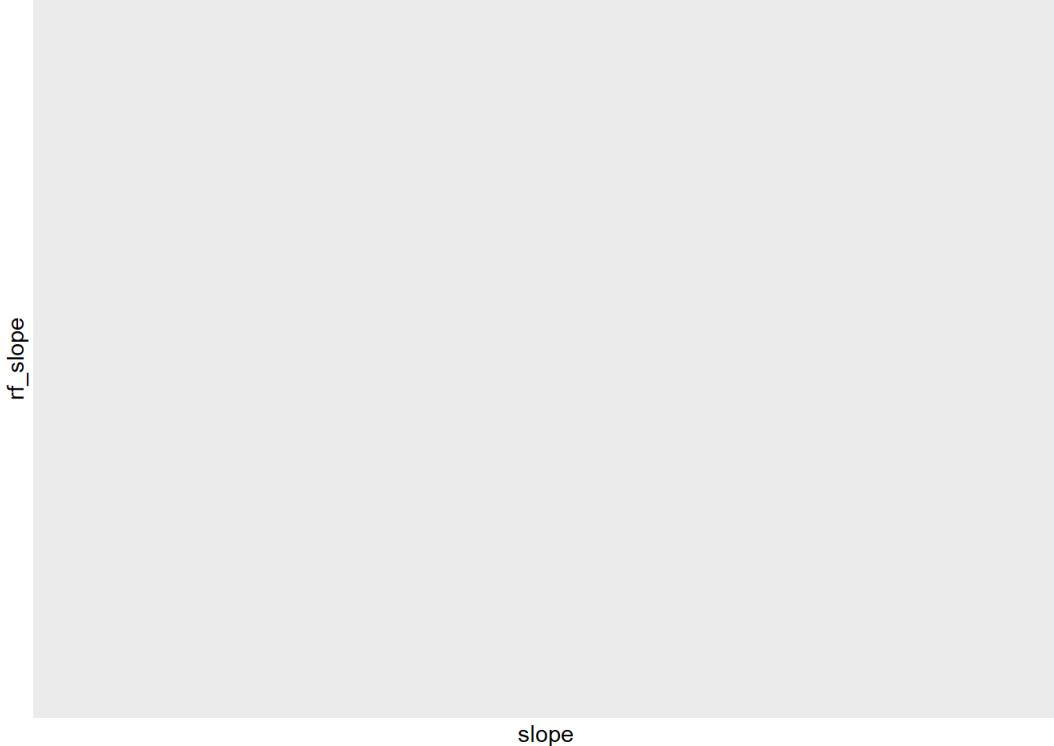
```
## Warning: Using size for a discrete variable is not advised.
```



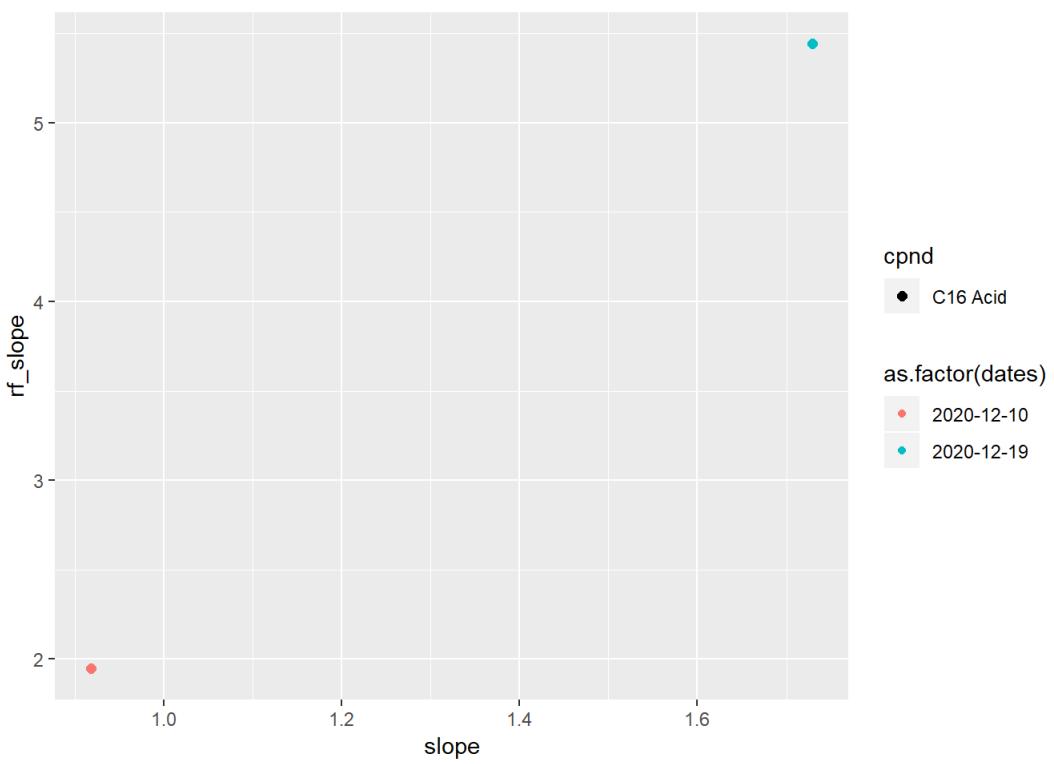
```
## Warning: Using size for a discrete variable is not advised.
```



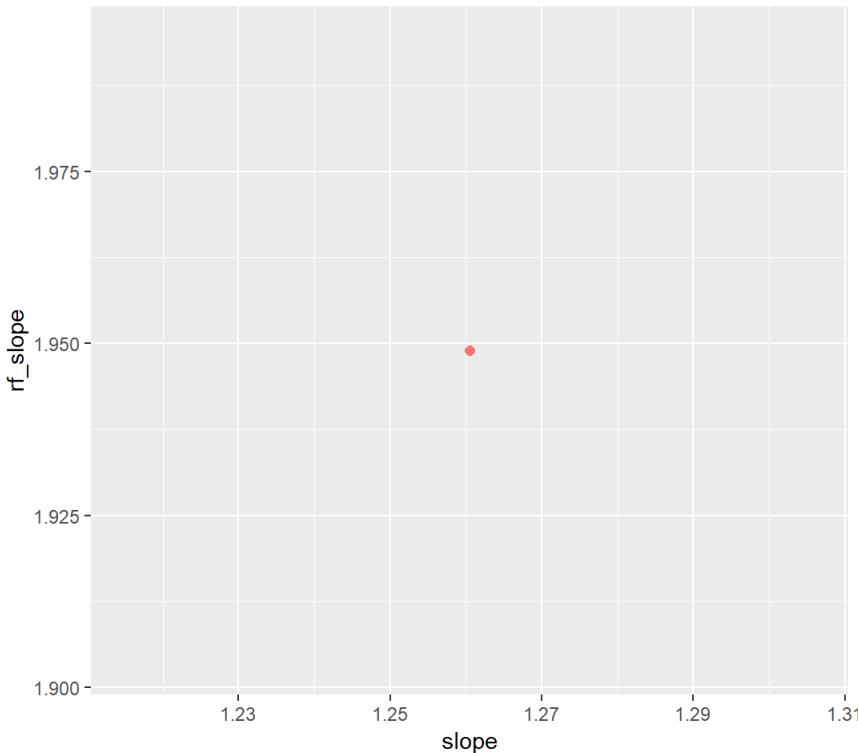
```
## Warning: Using size for a discrete variable is not advised.
```



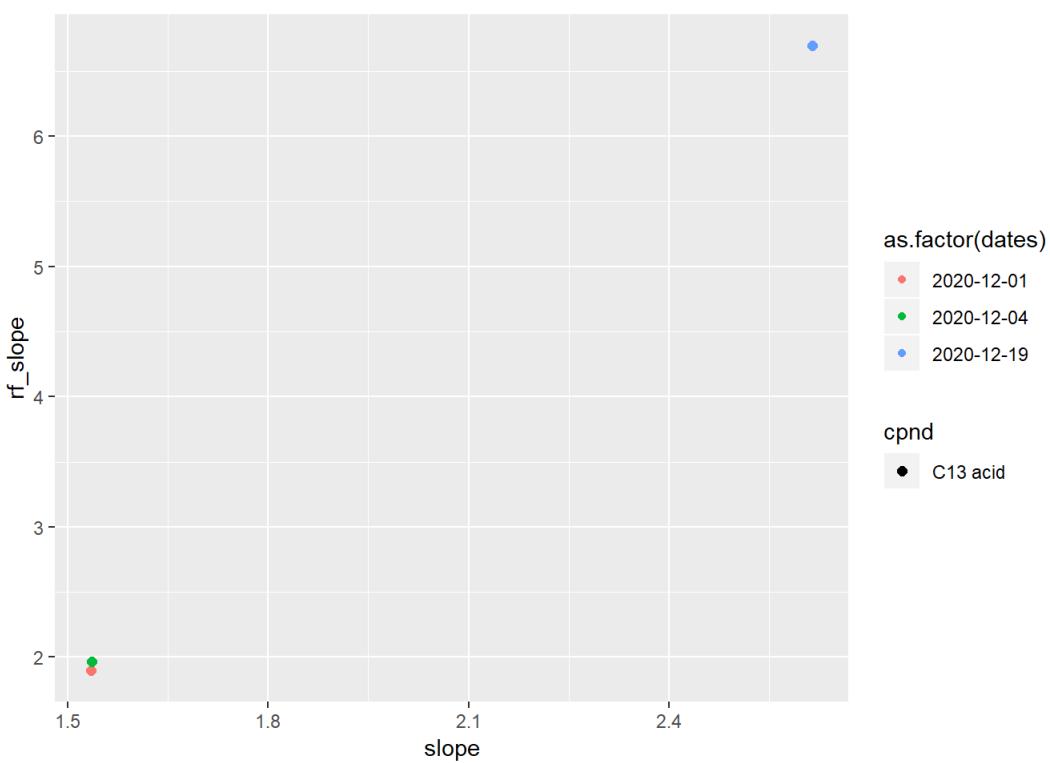
```
## Warning: Using size for a discrete variable is not advised.
```



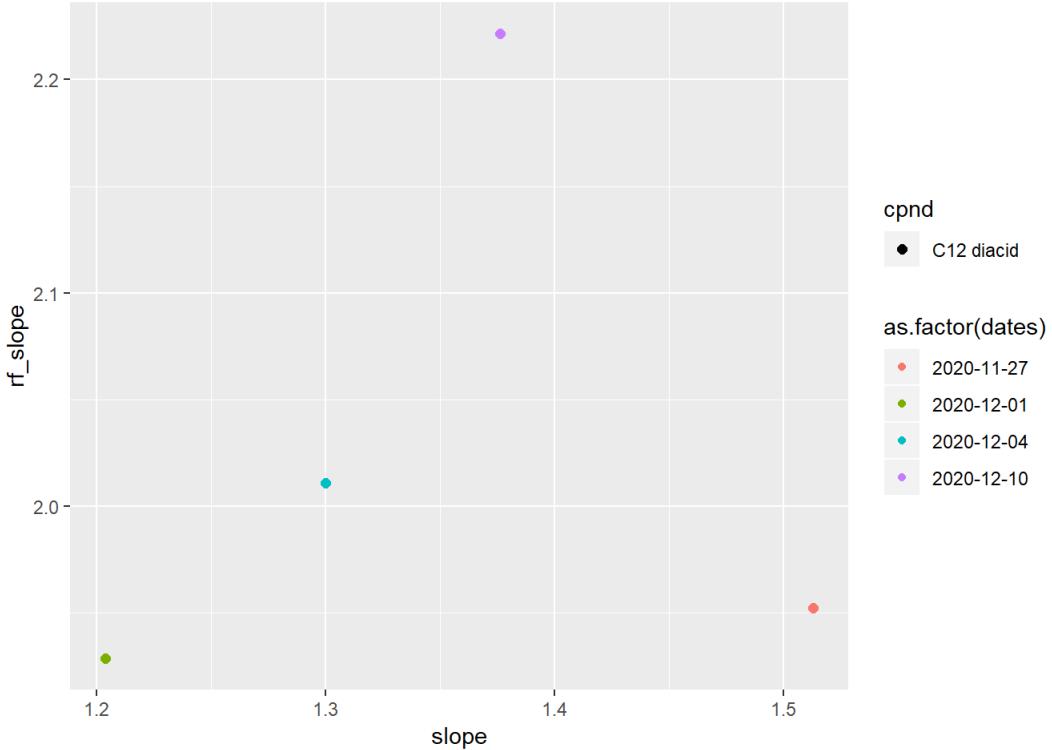
```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



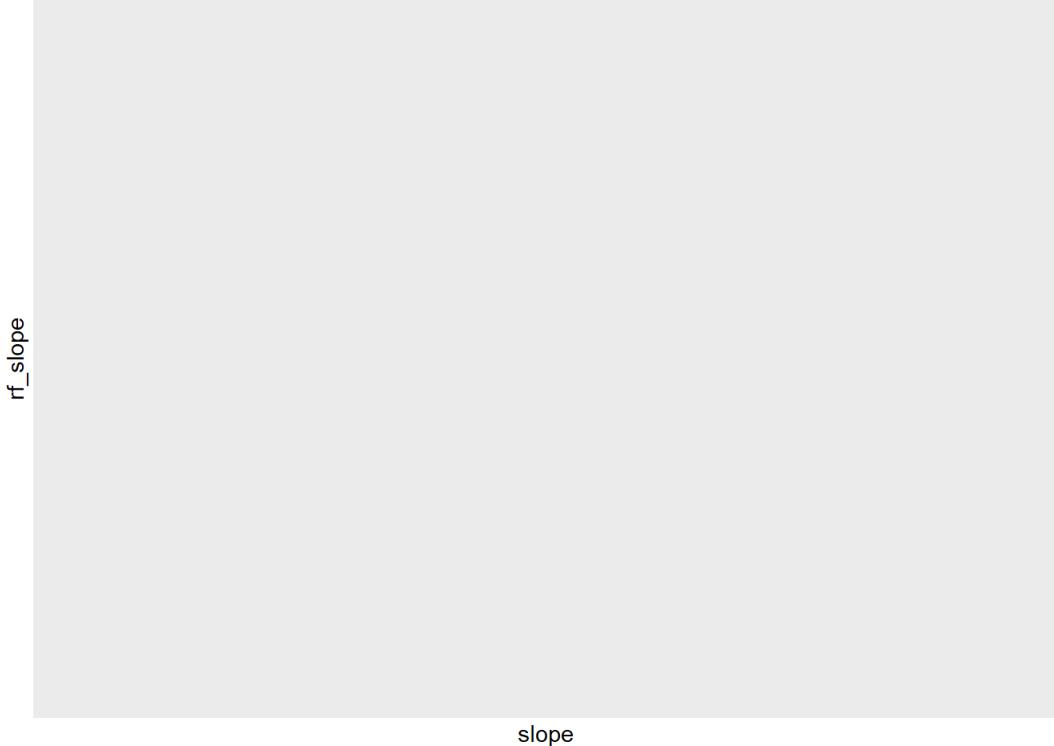
```
## Warning: Using size for a discrete variable is not advised.
```



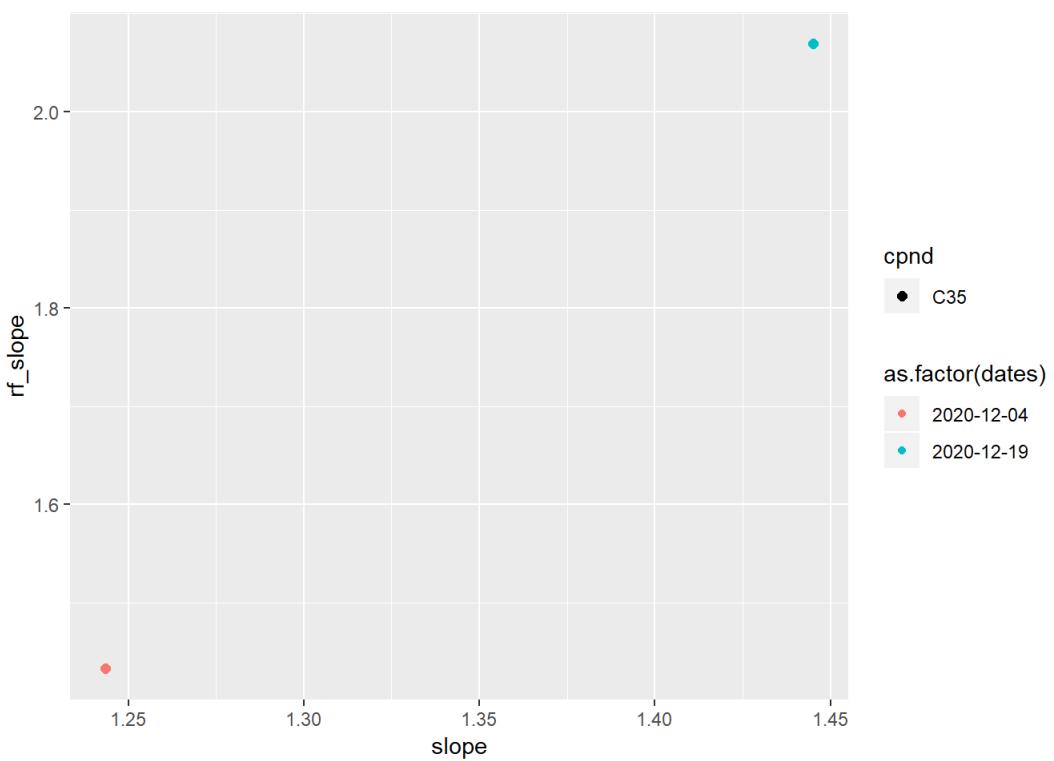
```
## Warning: Using size for a discrete variable is not advised.
```



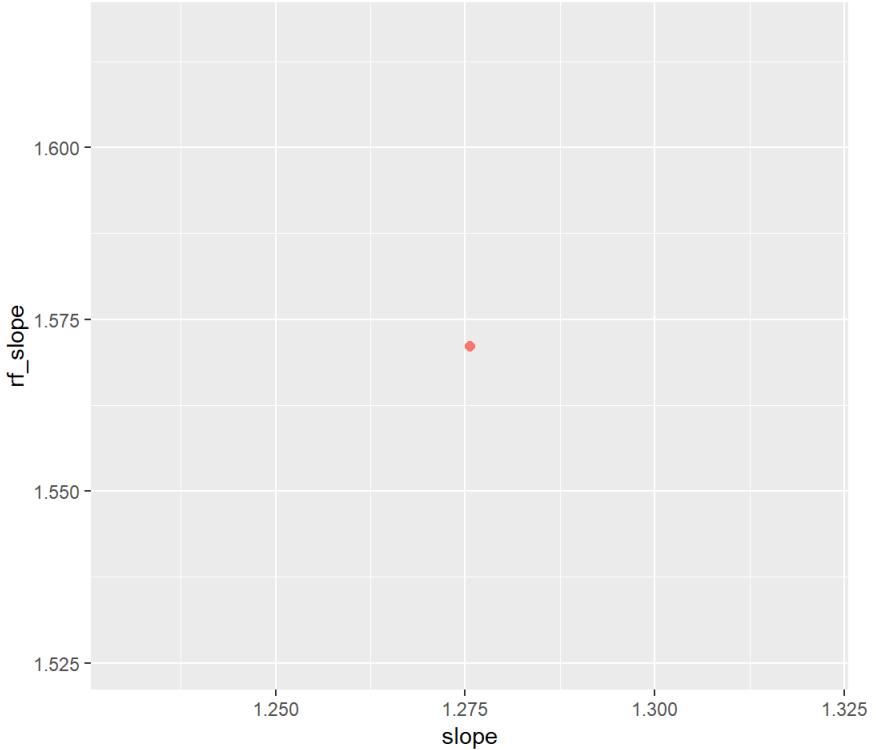
```
## Warning: Using size for a discrete variable is not advised.
```



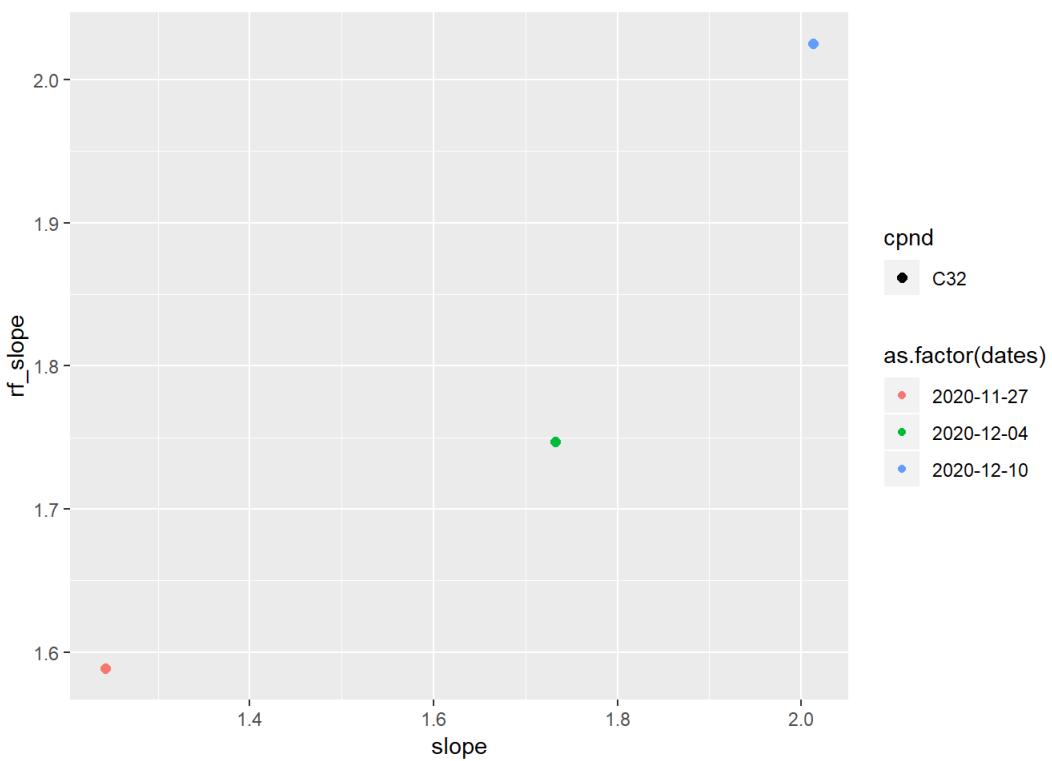
```
## Warning: Using size for a discrete variable is not advised.
```



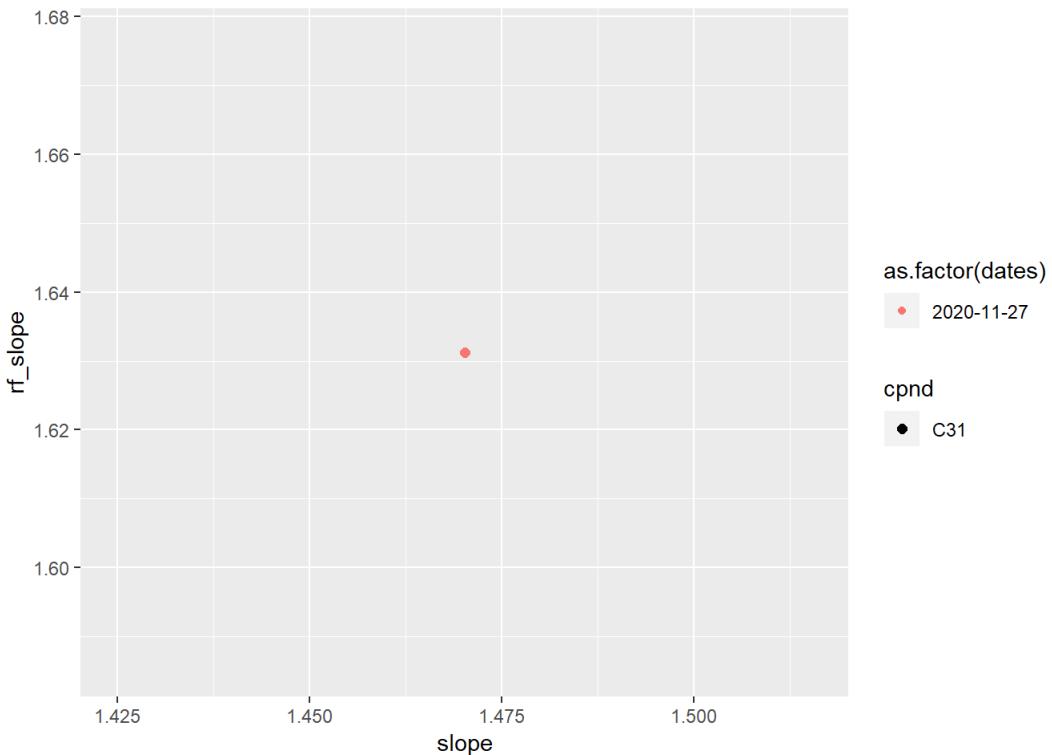
```
## Warning: Using size for a discrete variable is not advised.
```



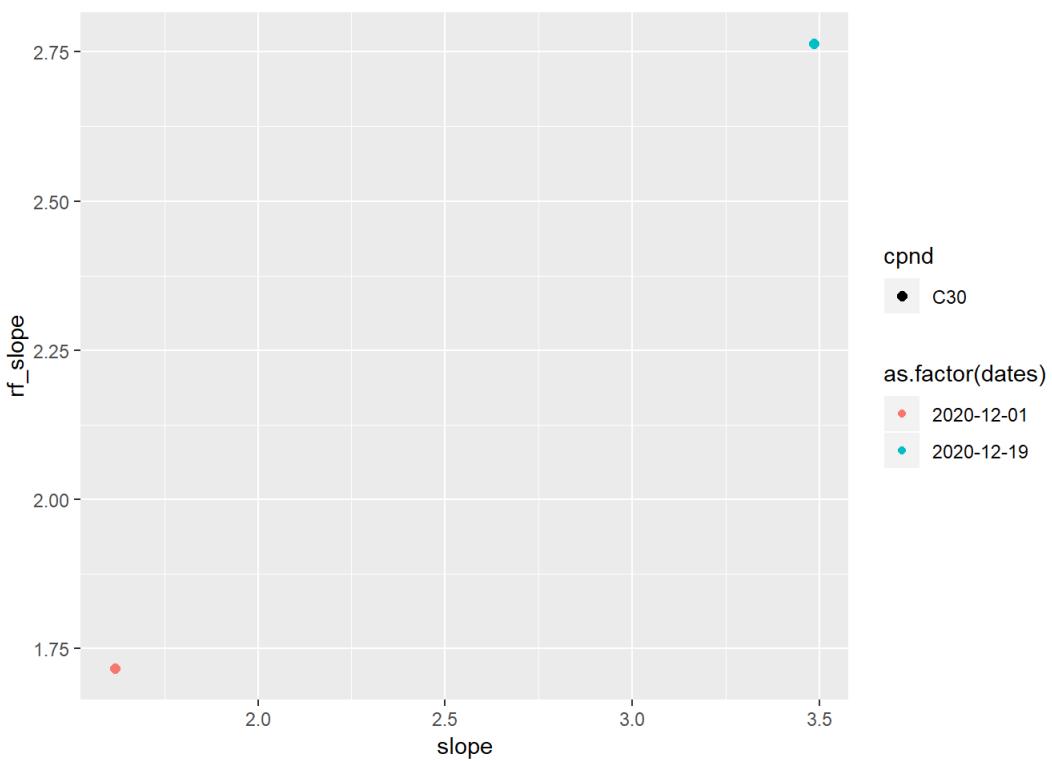
```
## Warning: Using size for a discrete variable is not advised.
```



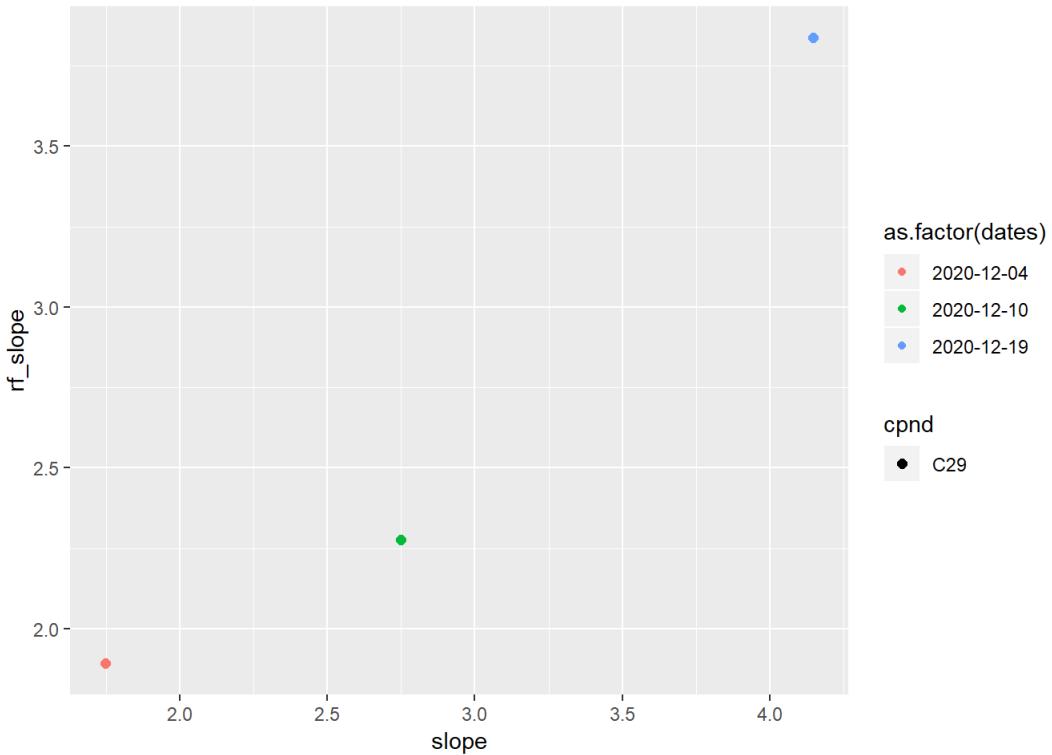
```
## Warning: Using size for a discrete variable is not advised.
```



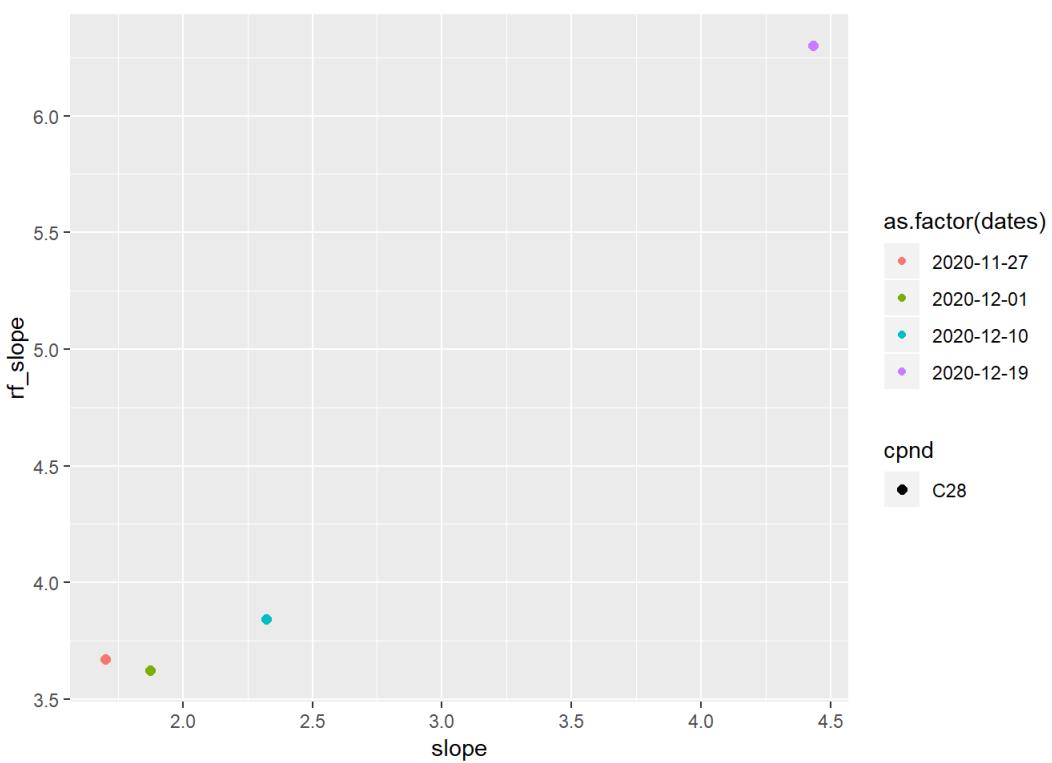
```
## Warning: Using size for a discrete variable is not advised.
```



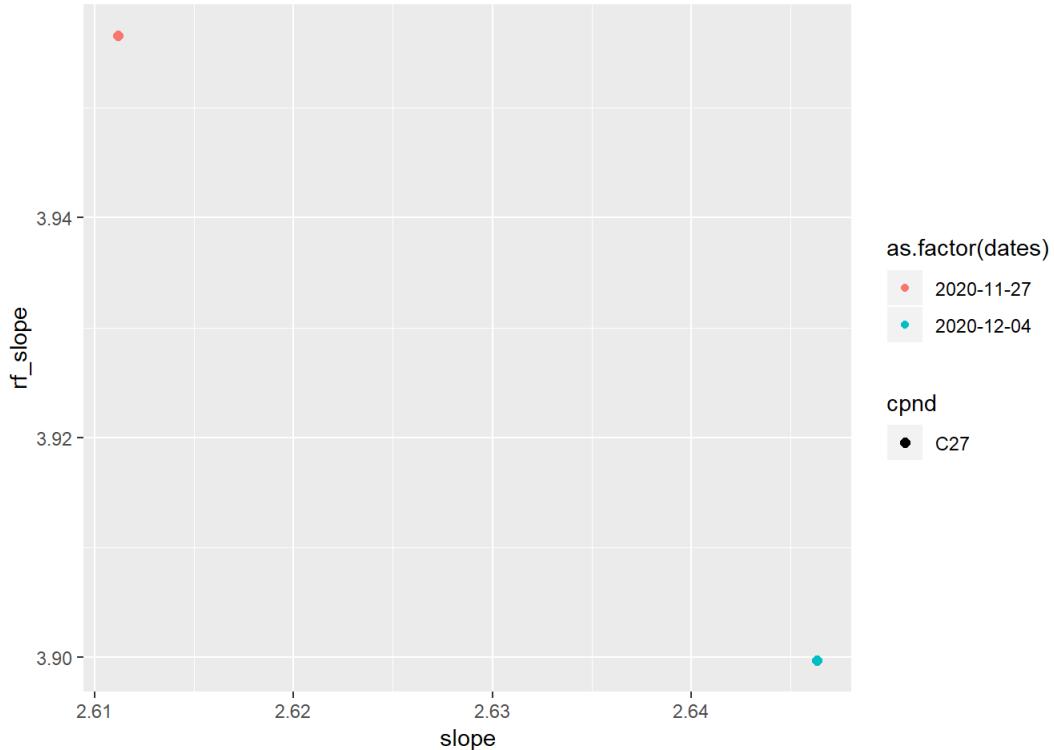
```
## Warning: Using size for a discrete variable is not advised.
```



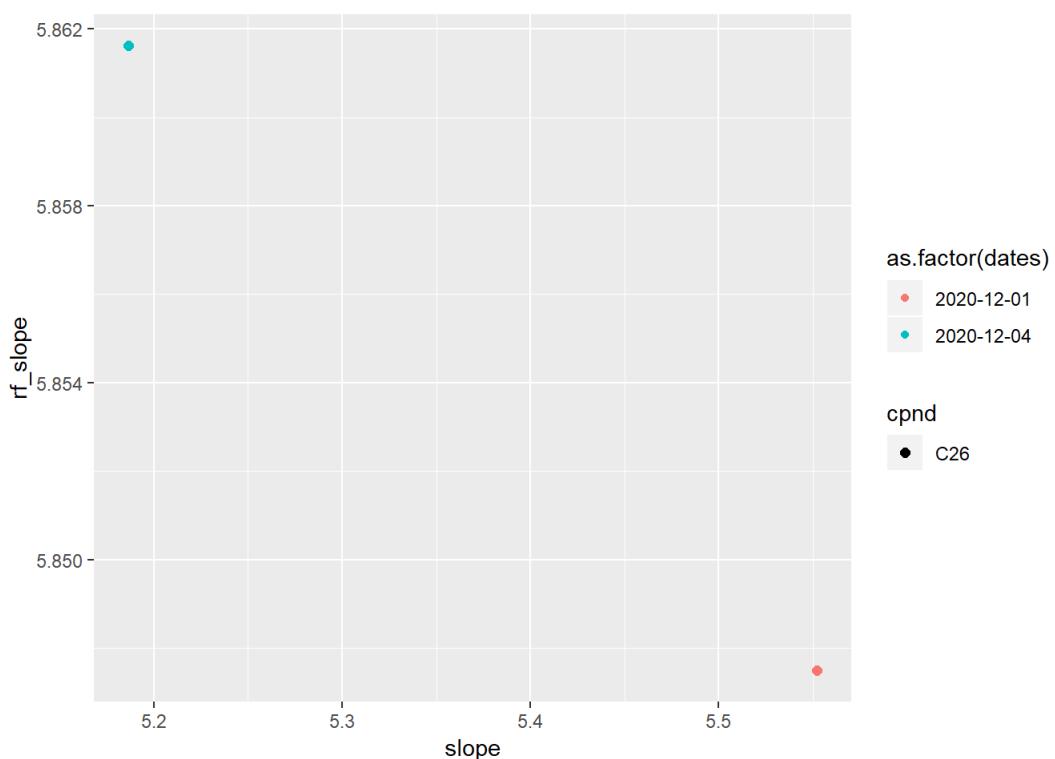
```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```

rf_slope

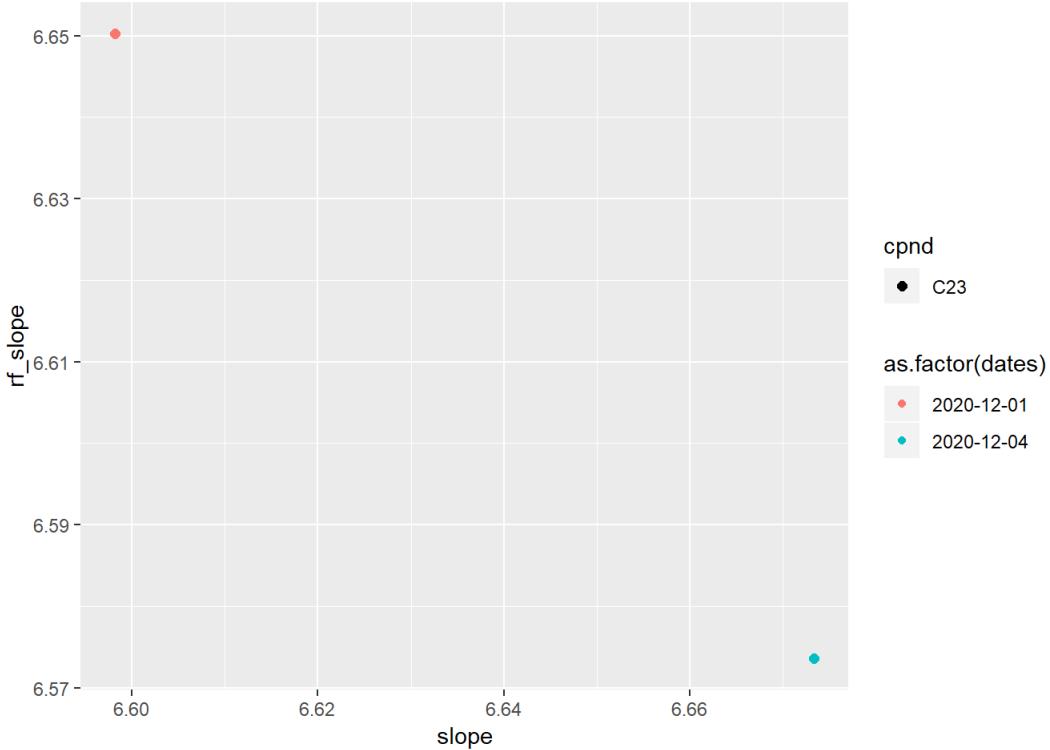
slope

```
## Warning: Using size for a discrete variable is not advised.
```

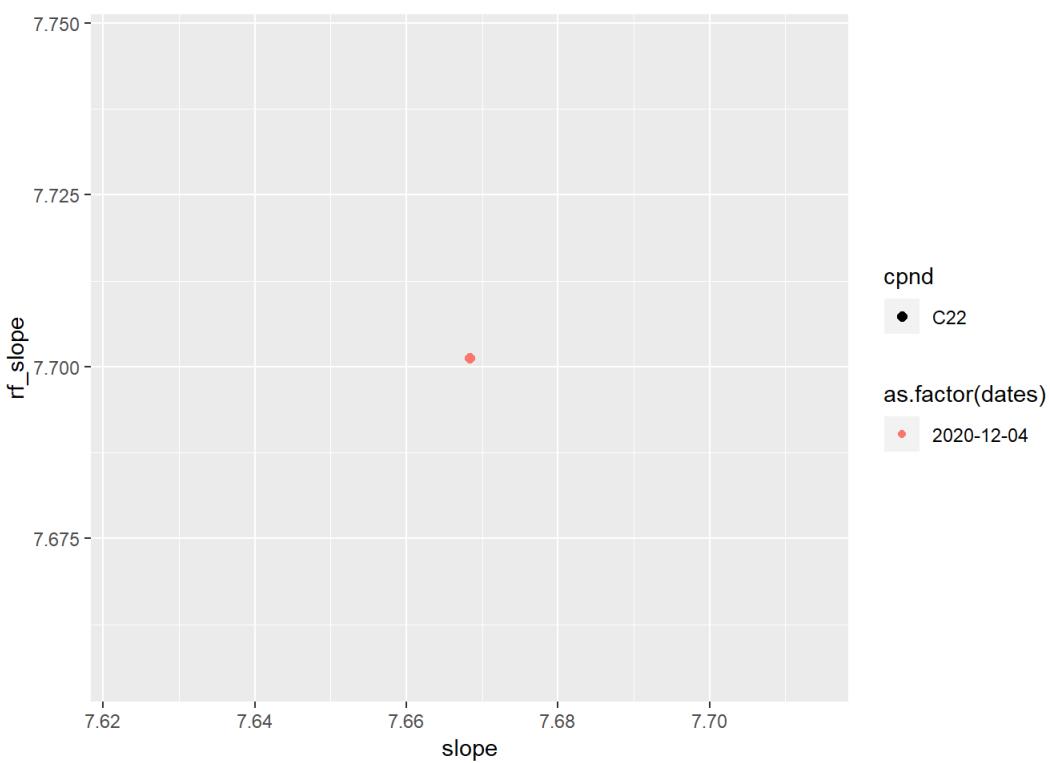
rf_slope

slope

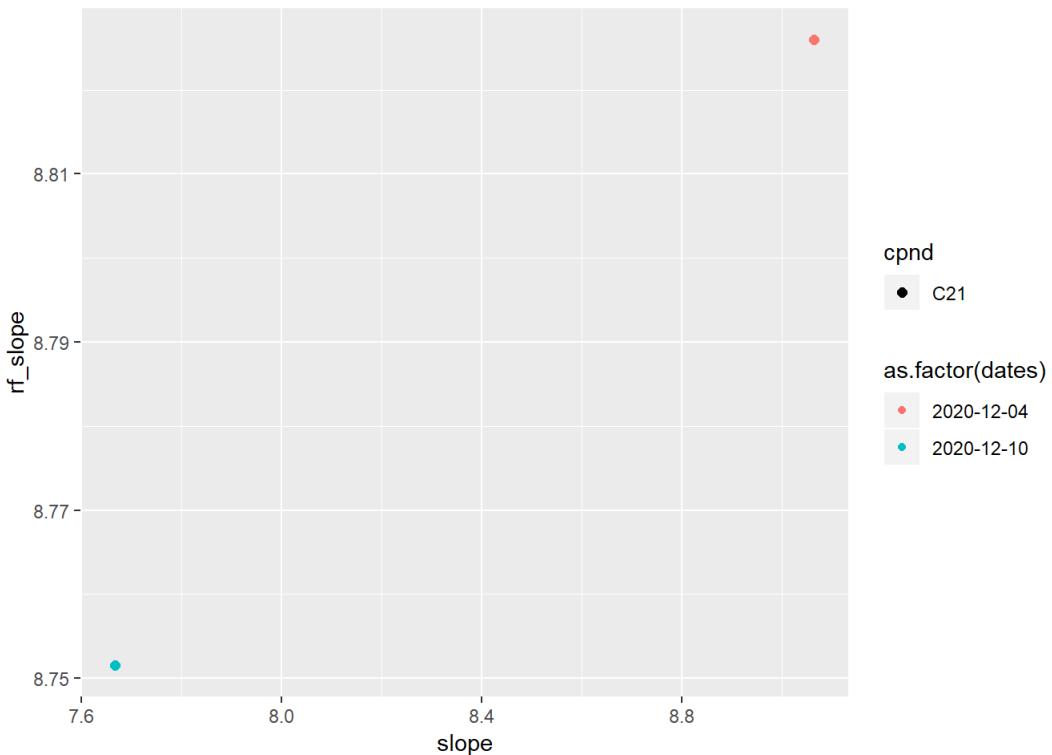
```
## Warning: Using size for a discrete variable is not advised.
```



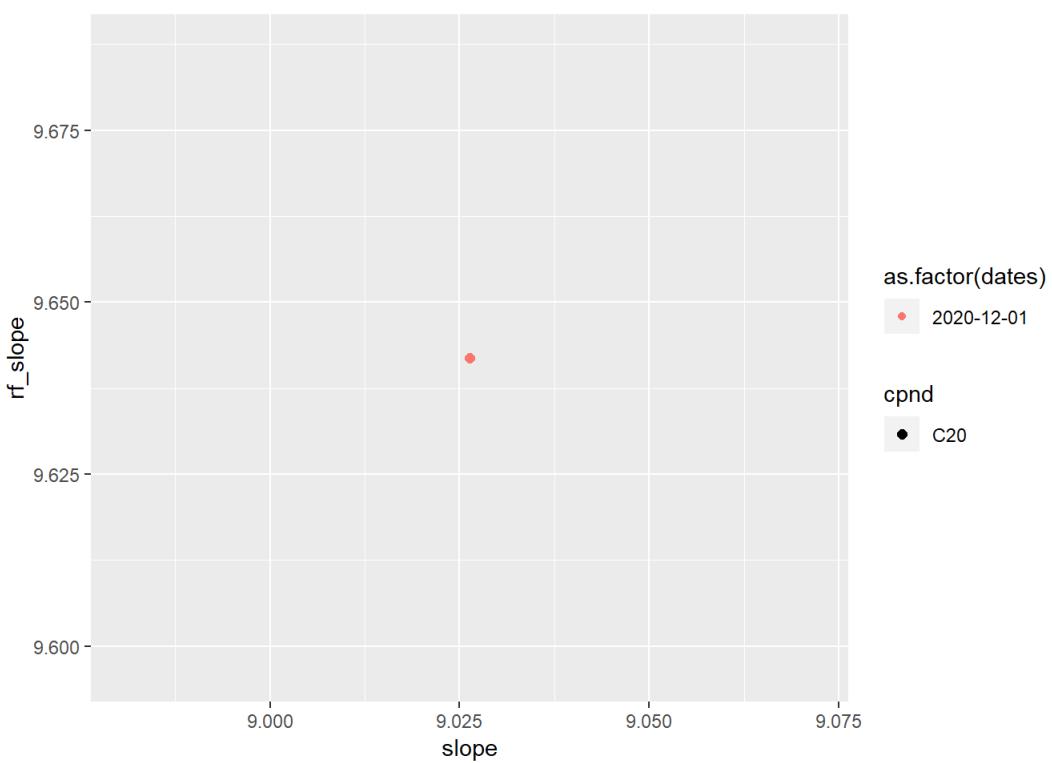
```
## Warning: Using size for a discrete variable is not advised.
```



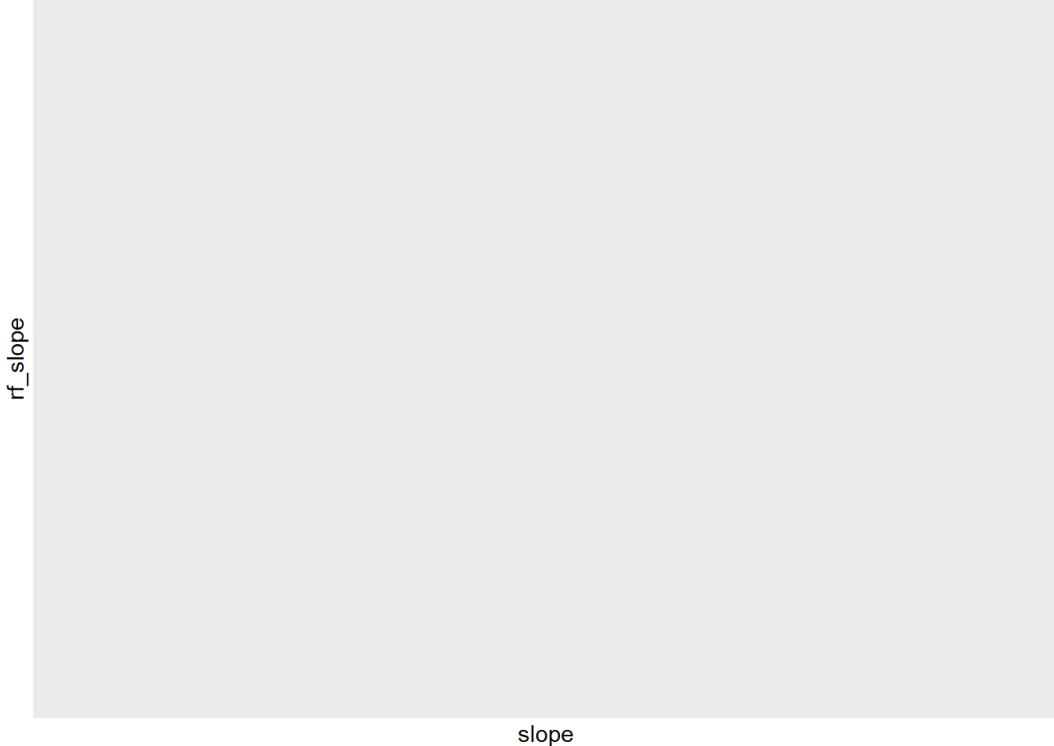
```
## Warning: Using size for a discrete variable is not advised.
```



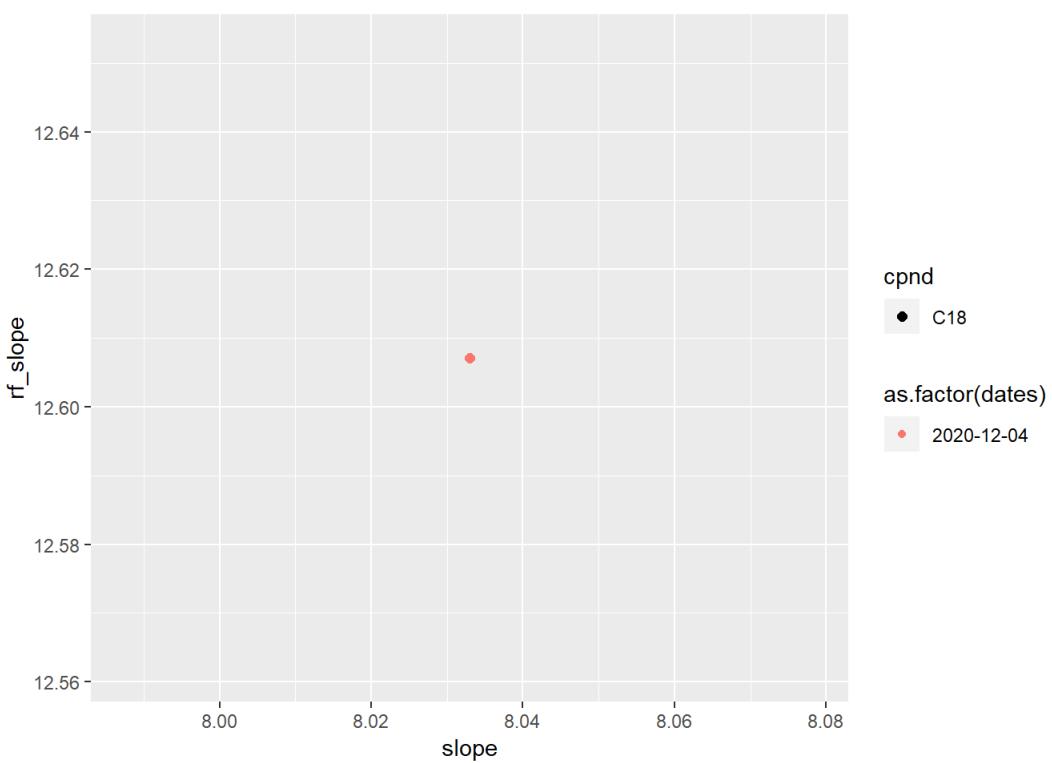
```
## Warning: Using size for a discrete variable is not advised.
```



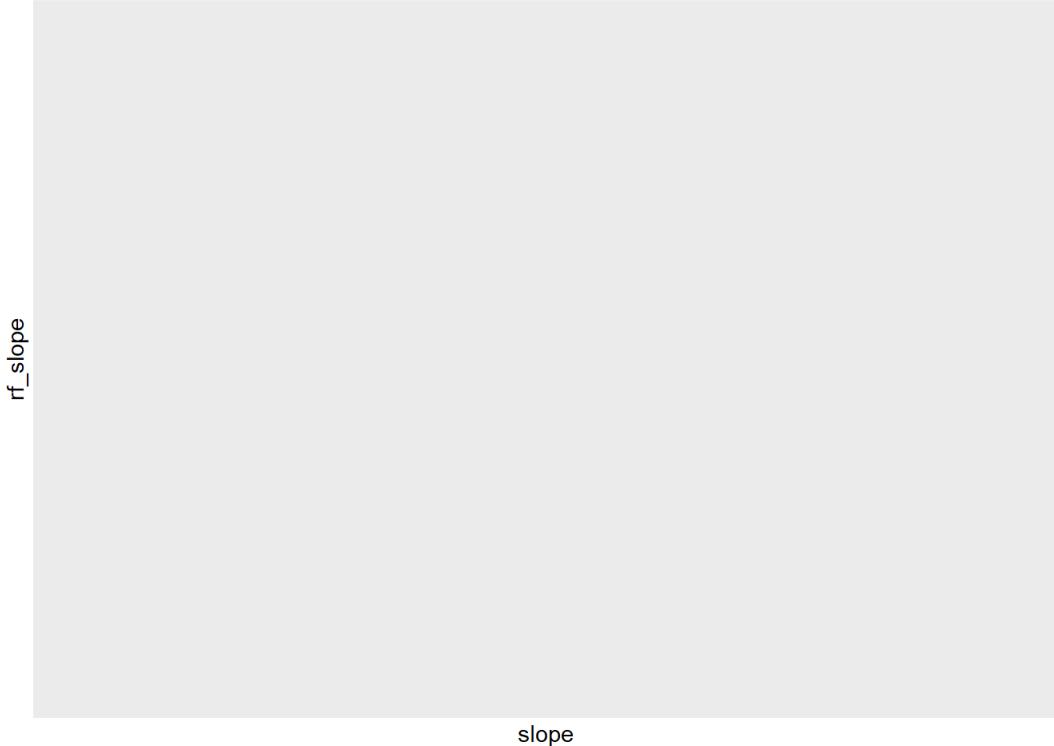
```
## Warning: Using size for a discrete variable is not advised.
```



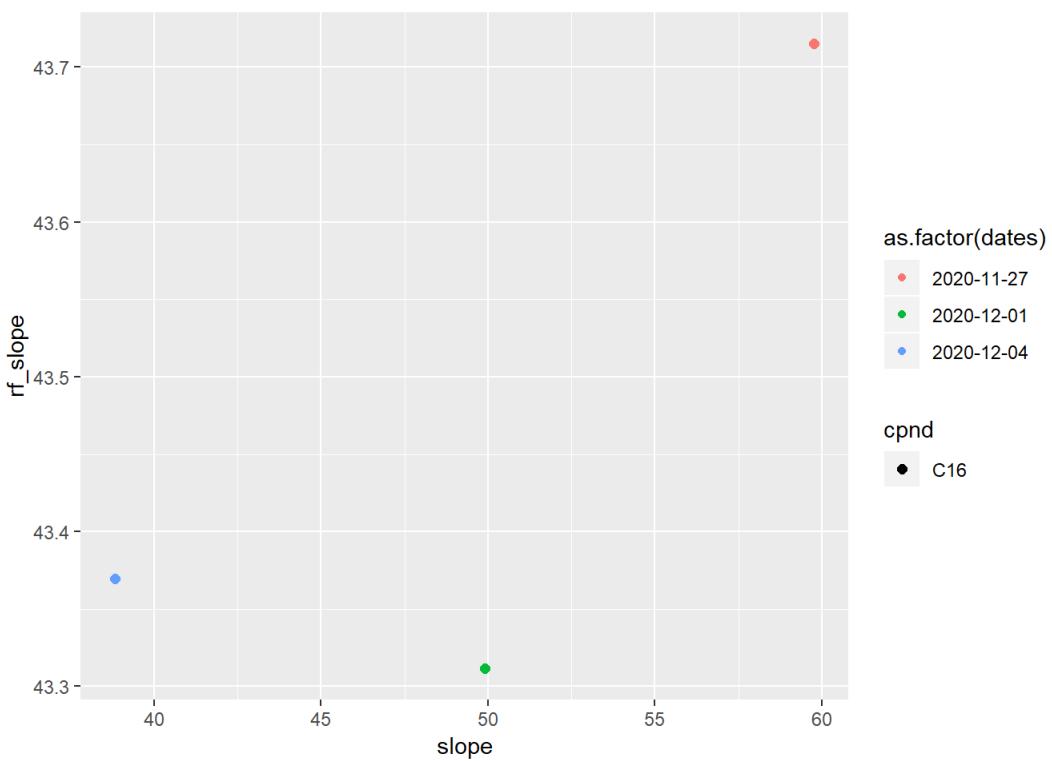
```
## Warning: Using size for a discrete variable is not advised.
```



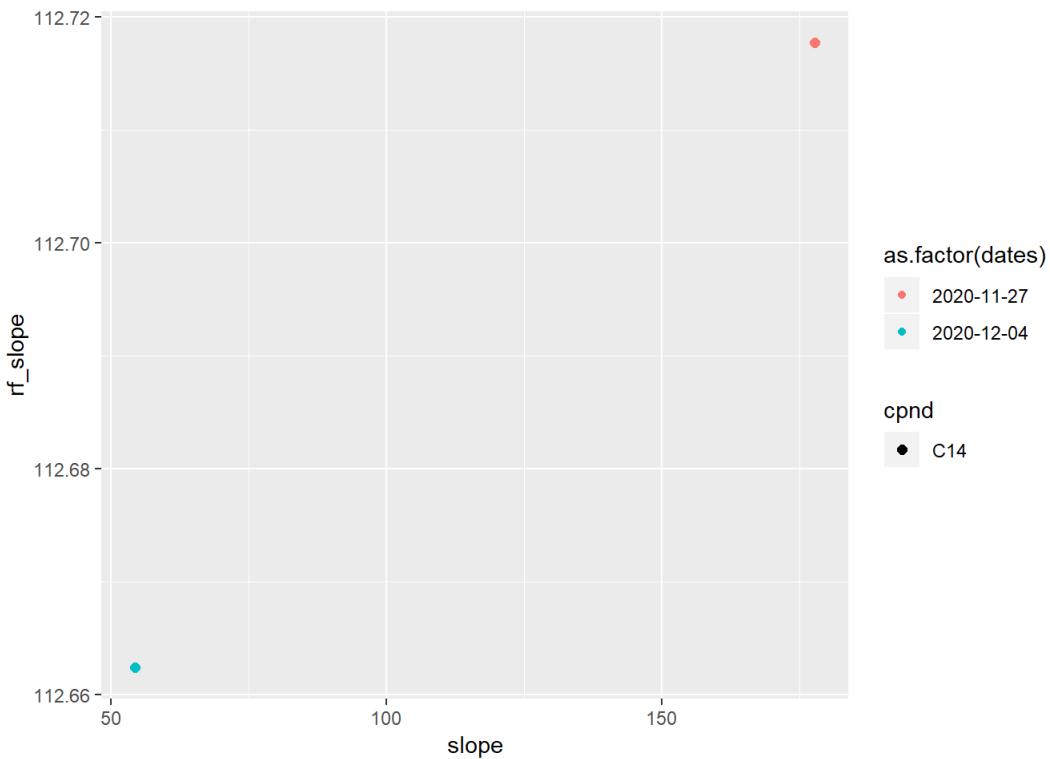
```
## Warning: Using size for a discrete variable is not advised.
```



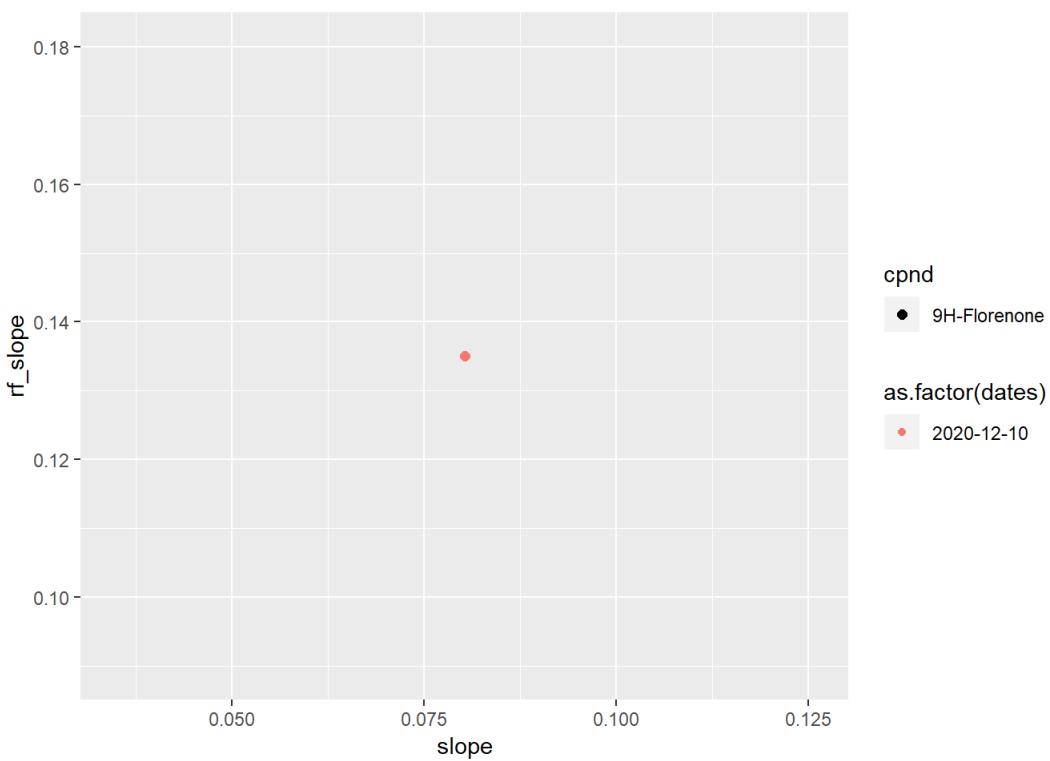
```
## Warning: Using size for a discrete variable is not advised.
```



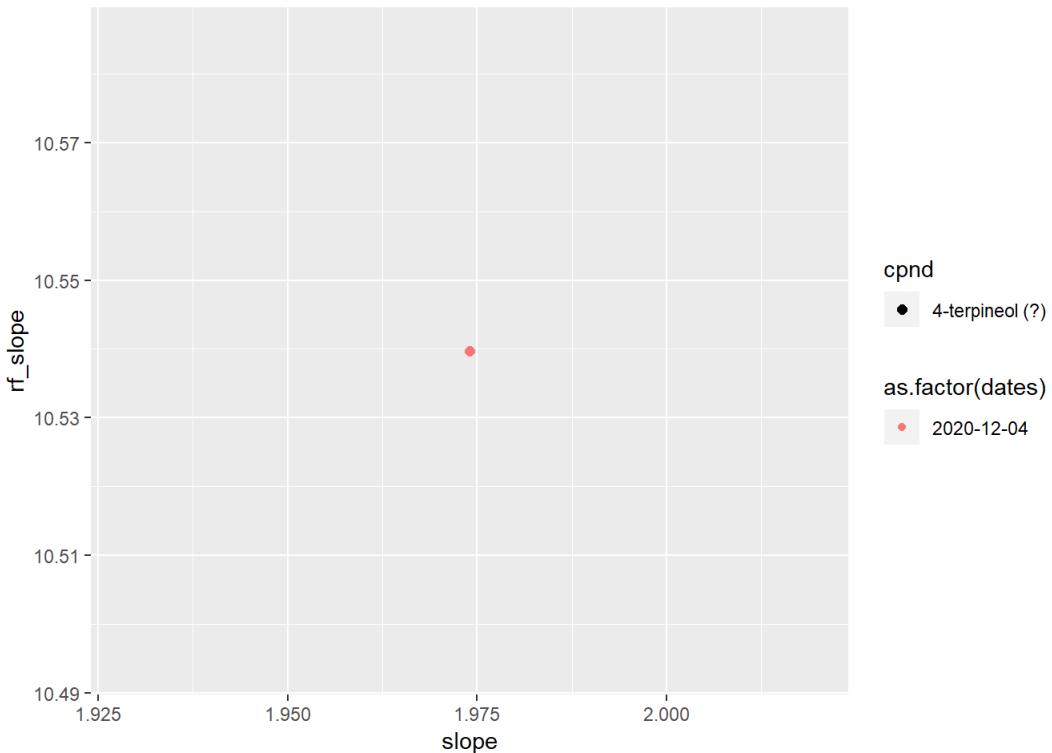
```
## Warning: Using size for a discrete variable is not advised.
```



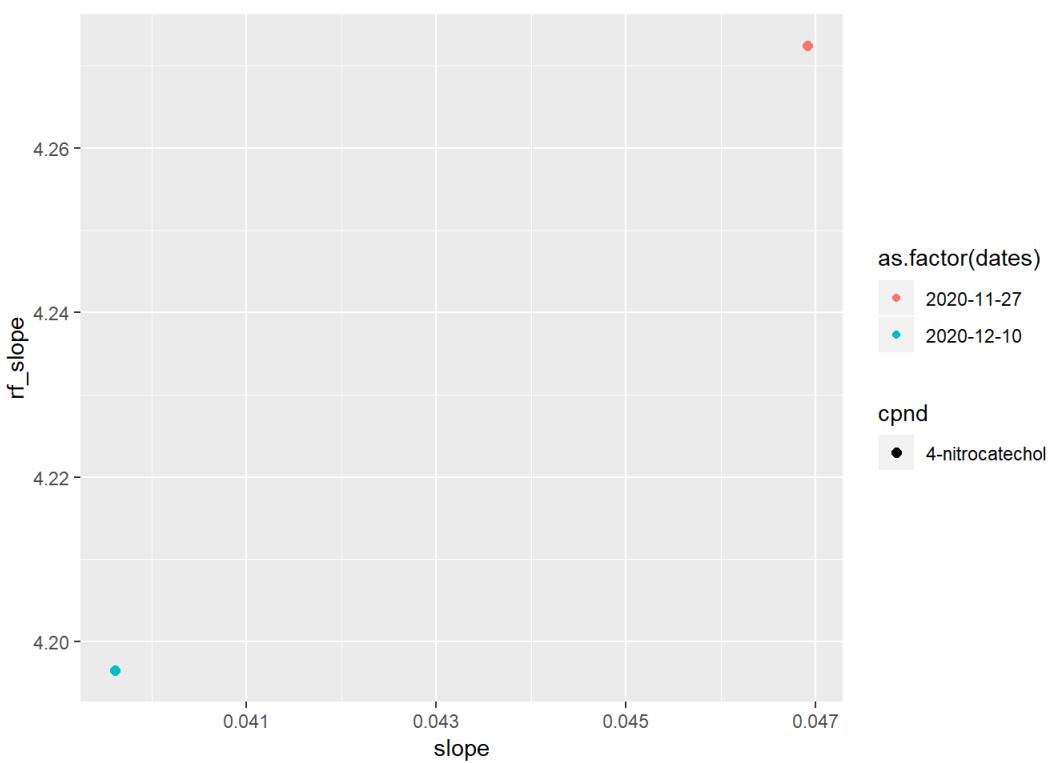
```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```

rf_slope

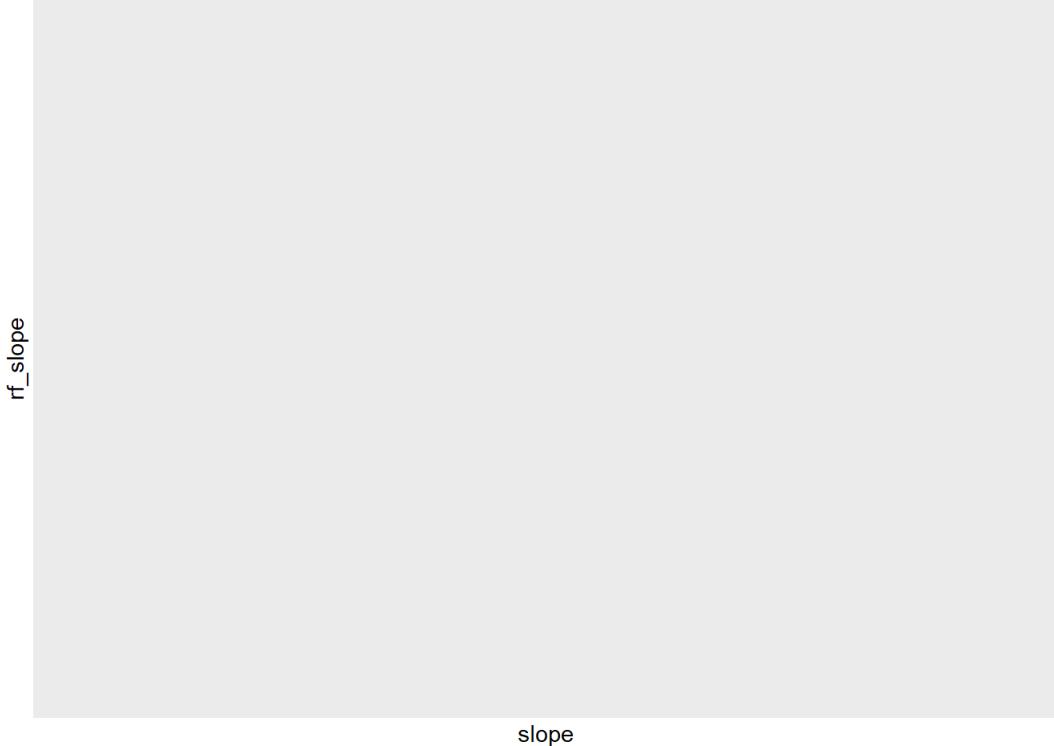
slope

```
## Warning: Using size for a discrete variable is not advised.
```

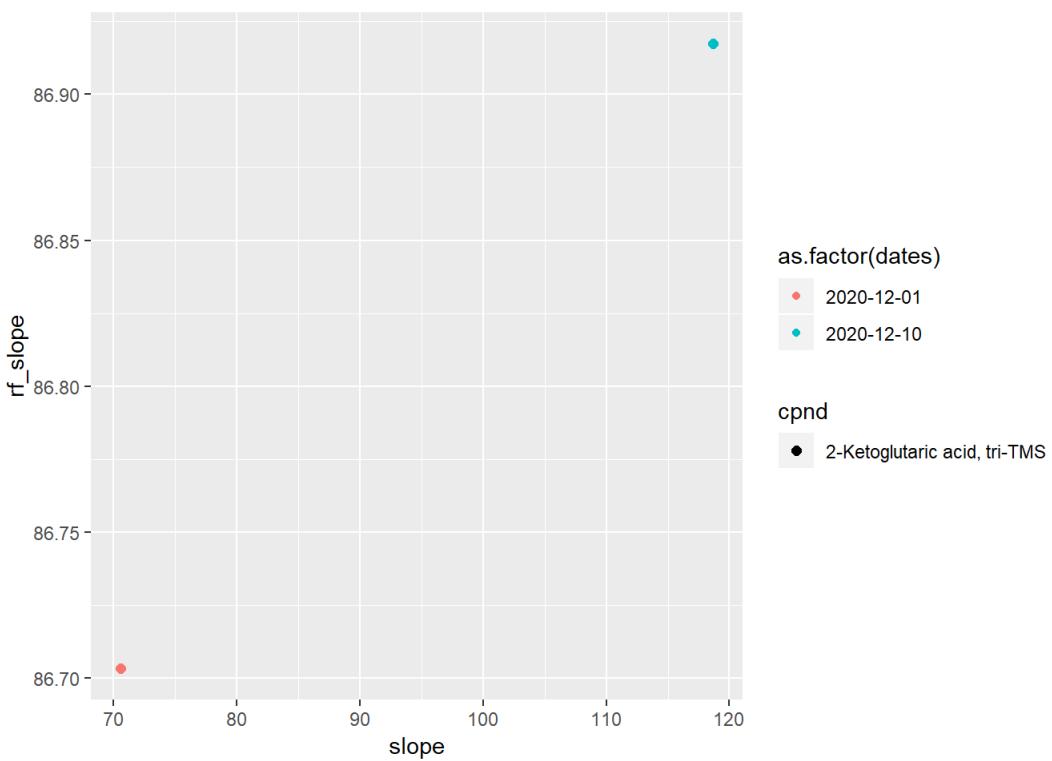
rf_slope

slope

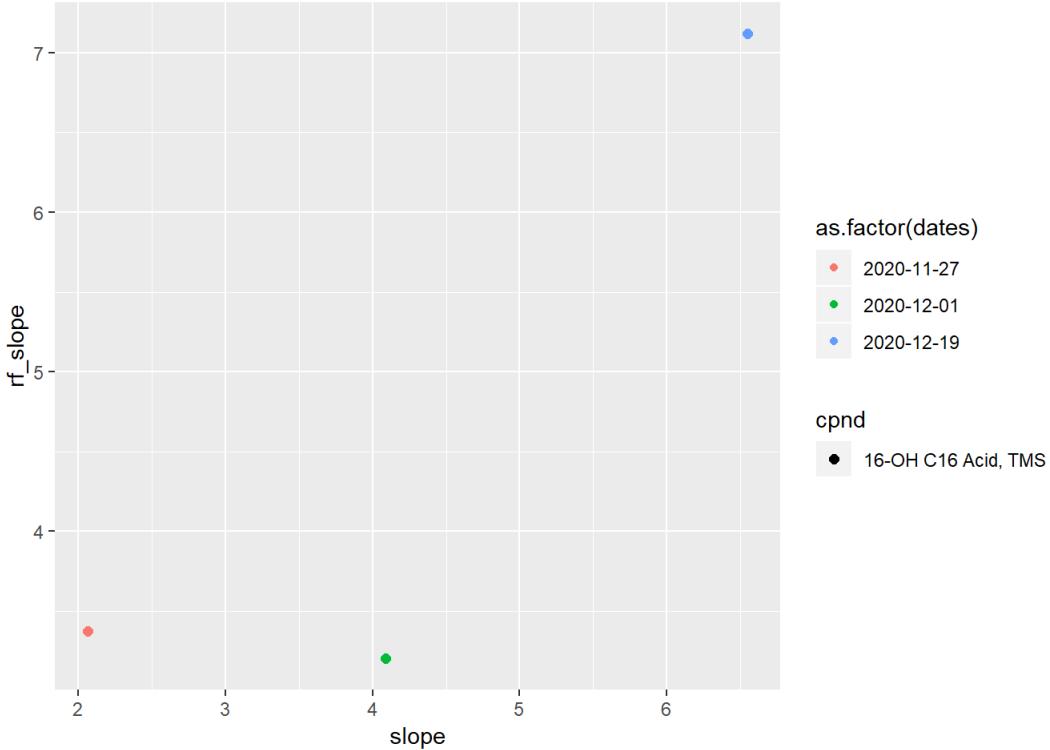
```
## Warning: Using size for a discrete variable is not advised.
```



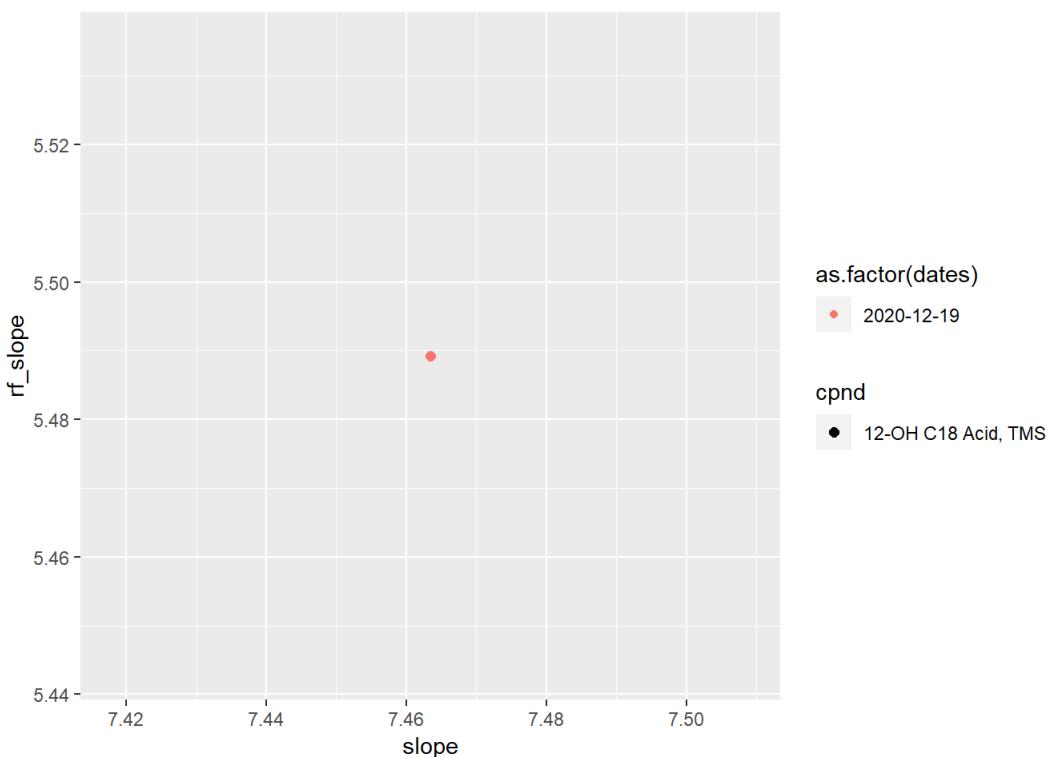
```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```

rf_slope

slope

```
## Warning: Using size for a discrete variable is not advised.
```

rf_slope

slope

```
## Warning: Using size for a discrete variable is not advised.
```

rf_slope

slope

```
## Warning: Using size for a discrete variable is not advised.
```

rf_slope

slope

```
## Warning: Using size for a discrete variable is not advised.
```

rf_slope

slope

```
## Warning: Using size for a discrete variable is not advised.
```

rf_slope

slope

```
## Warning: Using size for a discrete variable is not advised.
```

rf_slope

slope

```
## Warning: Using size for a discrete variable is not advised.
```

rf_slope

slope

```
## Warning: Using size for a discrete variable is not advised.
```

rf_slope

slope

```
#different way to visualize

rf_vis <- slopemod.test.withpred %>%
  dplyr::select(cpnd, dates, slope, rf_slope, r2) %>%
  gather(key = slope_cat, value = slope, slope, rf_slope)

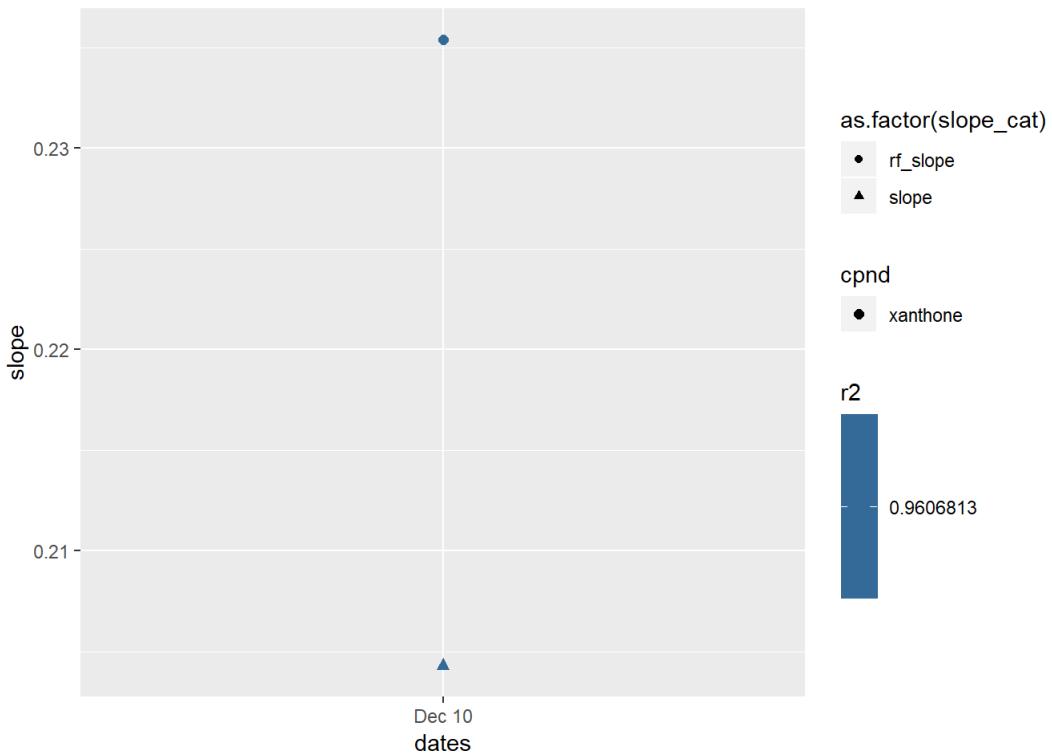
for(i in 1:nrow(cal_pts)){
  bid_temp <- cal_pts$Compound.Name[i]

  n_c_full_sub <- rf_vis %>%
    filter(cpnd == bid_temp)

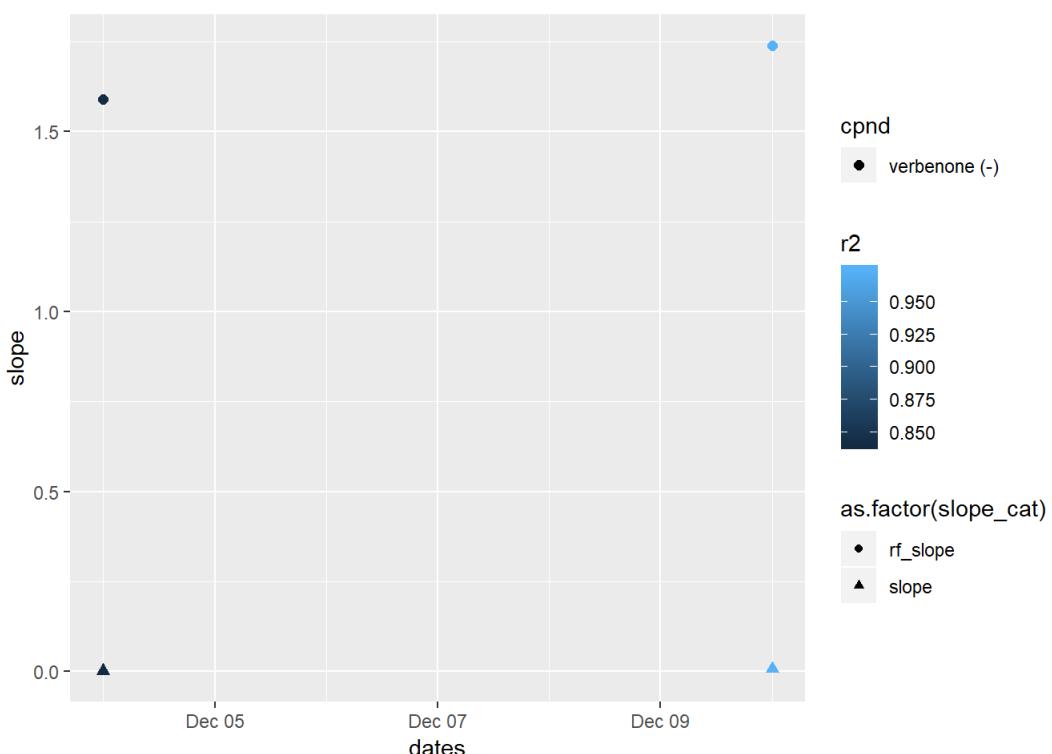
  p <- n_c_full_sub %>%
    ggplot(aes(x = dates, y = slope, shape = as.factor(slope_cat), color = r2, size = cpnd)) +
    geom_point()

  print(p)
}

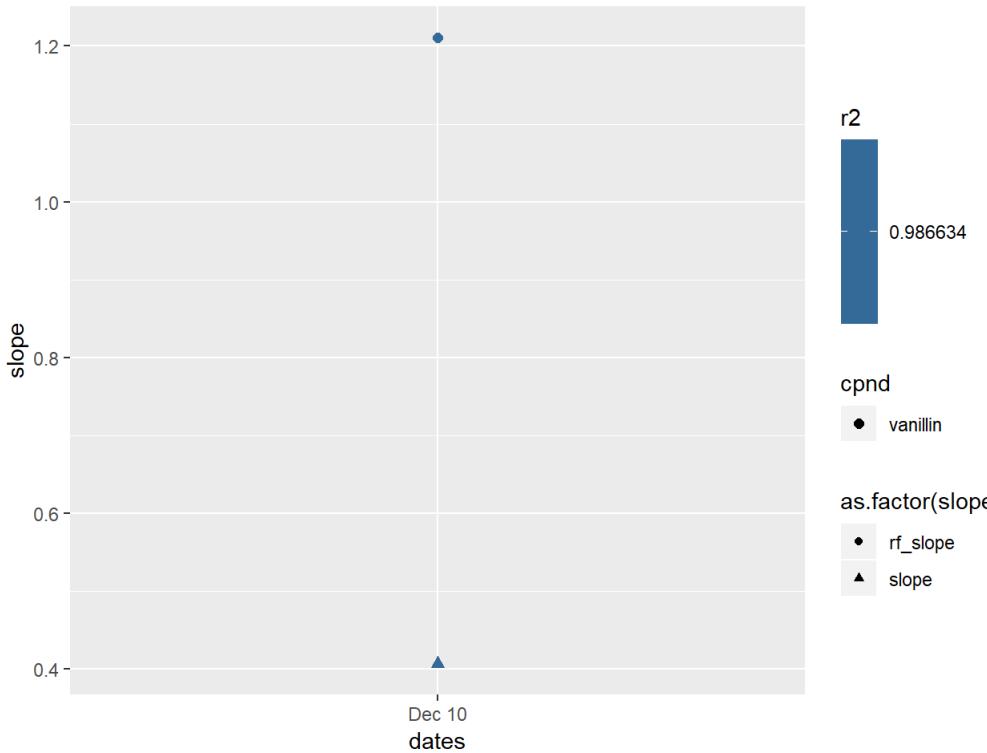
## Warning: Using size for a discrete variable is not advised.
```



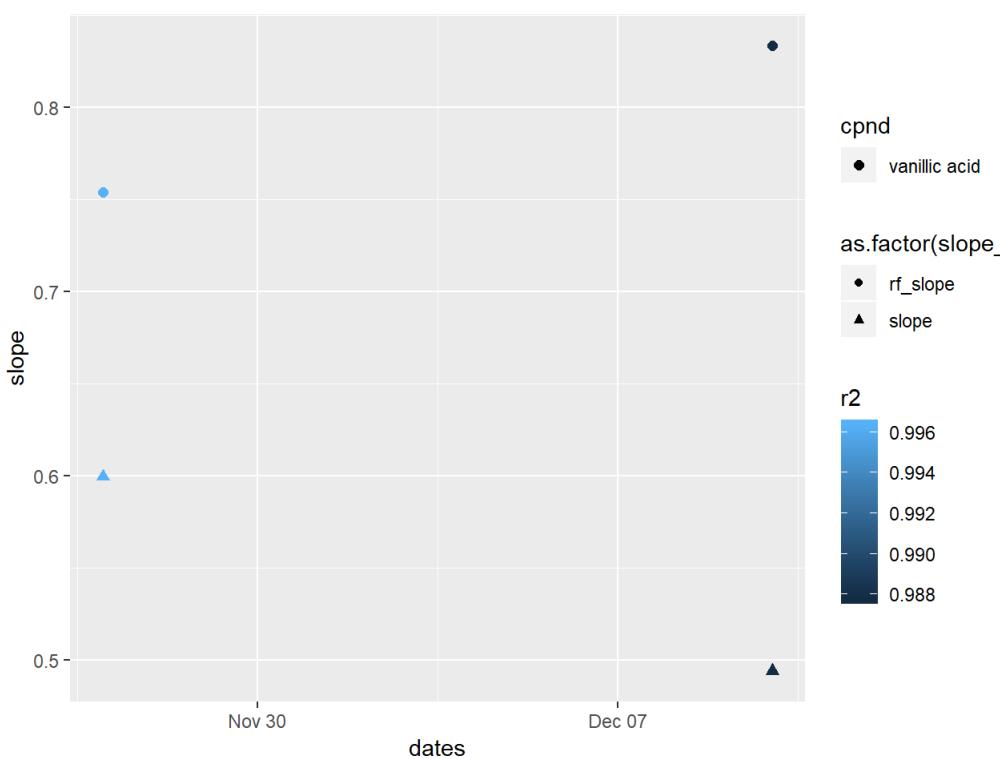
```
## Warning: Using size for a discrete variable is not advised.
```



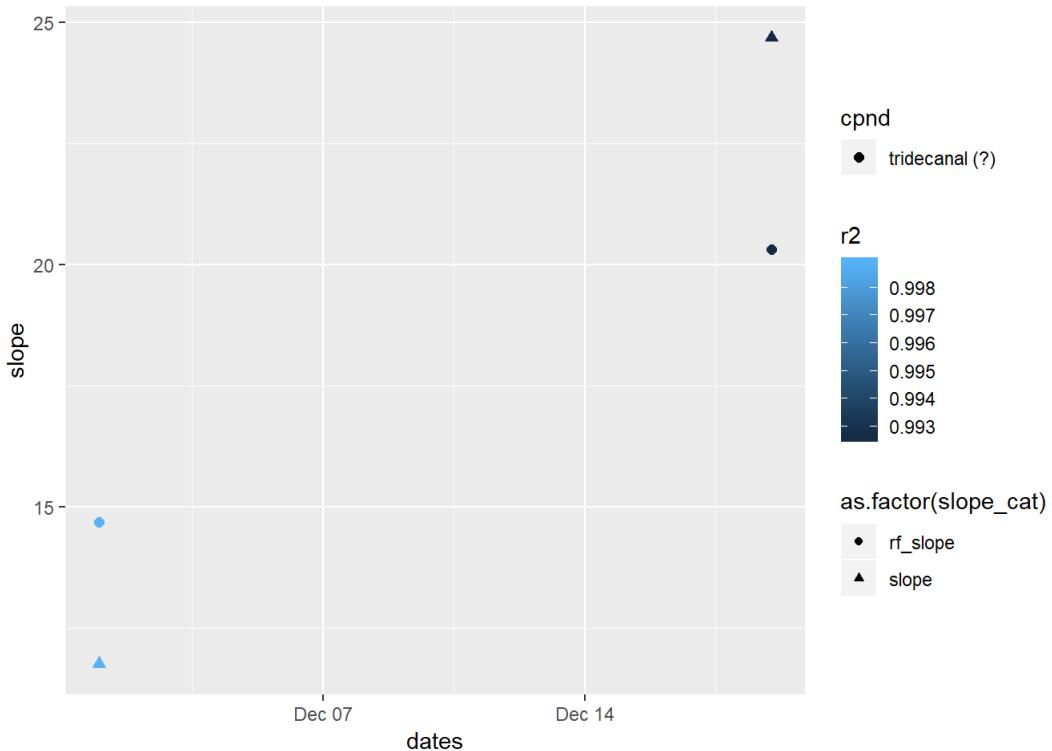
```
## Warning: Using size for a discrete variable is not advised.
```



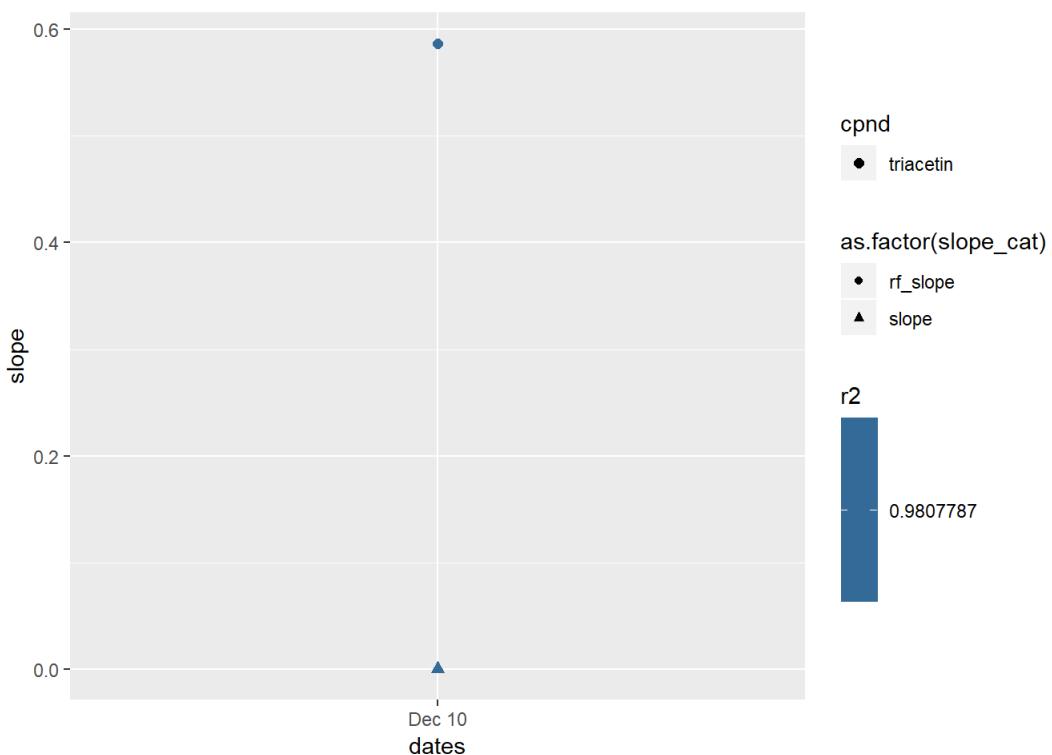
```
## Warning: Using size for a discrete variable is not advised.
```



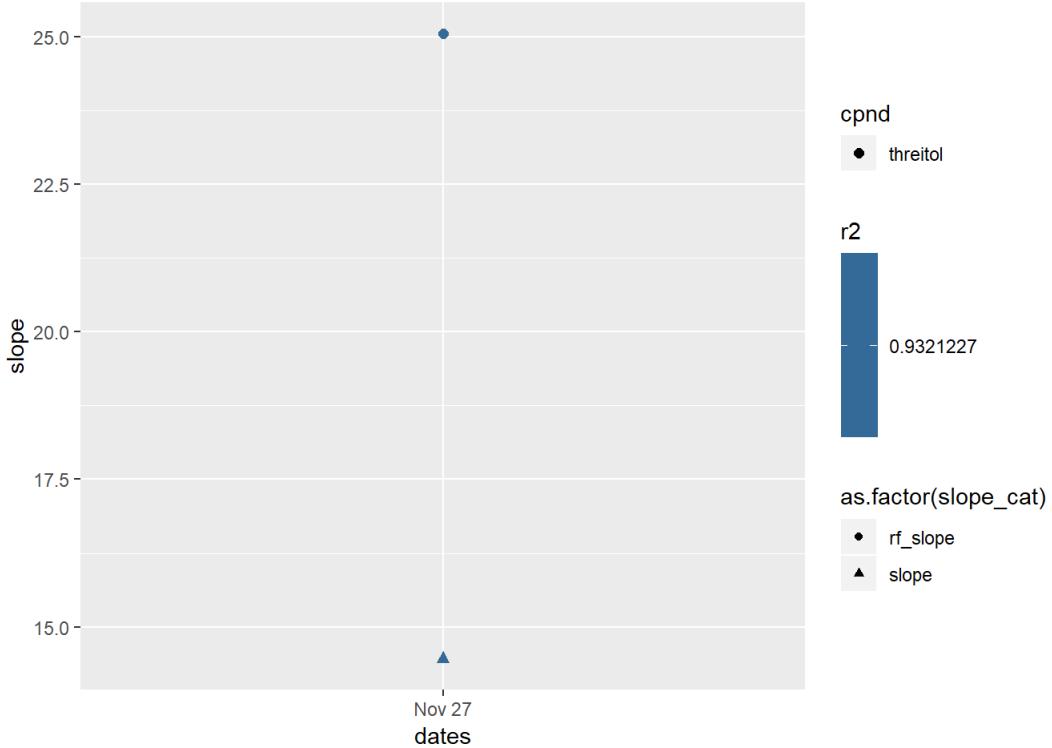
```
## Warning: Using size for a discrete variable is not advised.
```



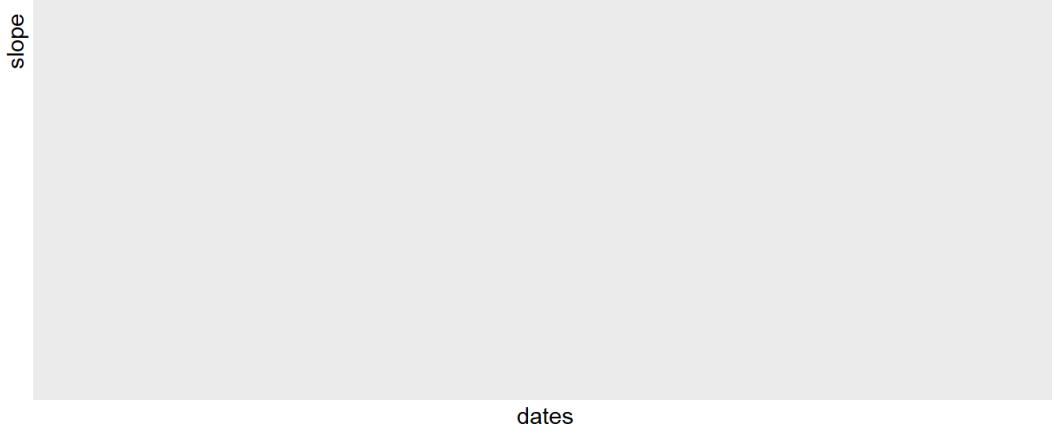
```
## Warning: Using size for a discrete variable is not advised.
```



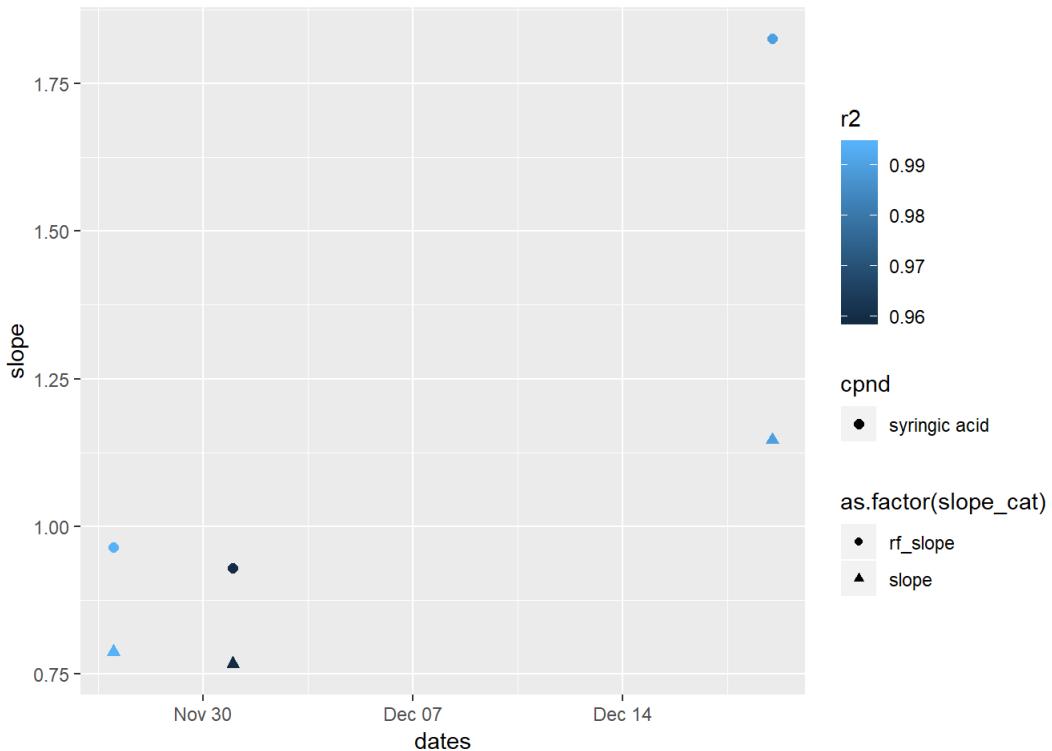
```
## Warning: Using size for a discrete variable is not advised.
```



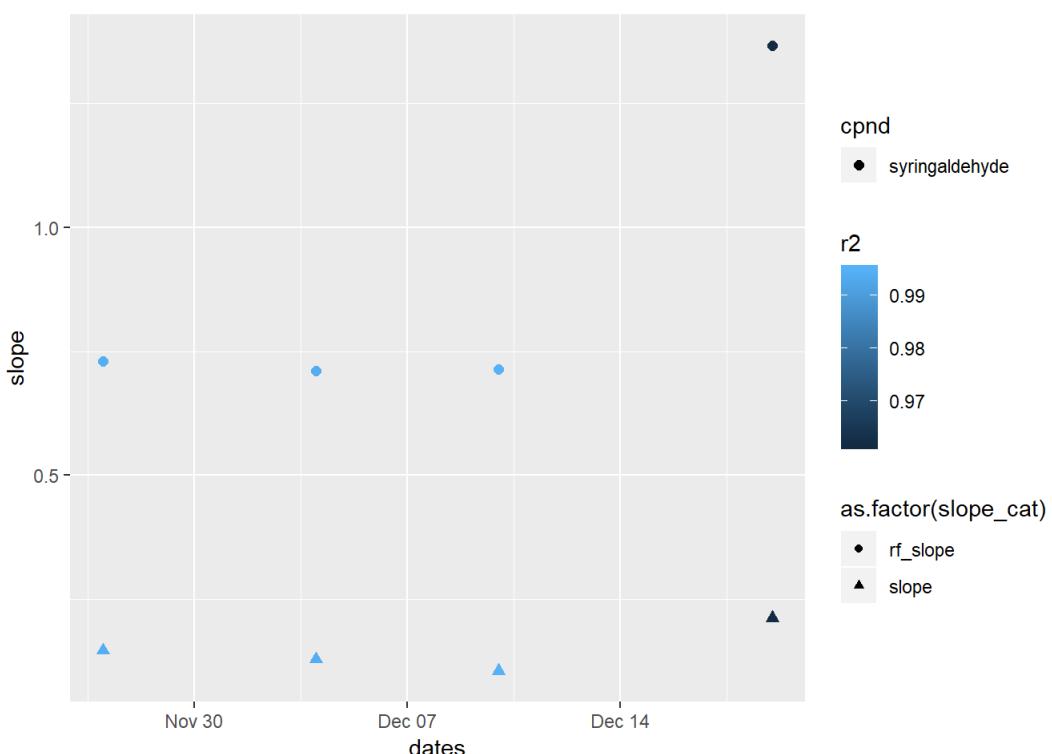
```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```

slope

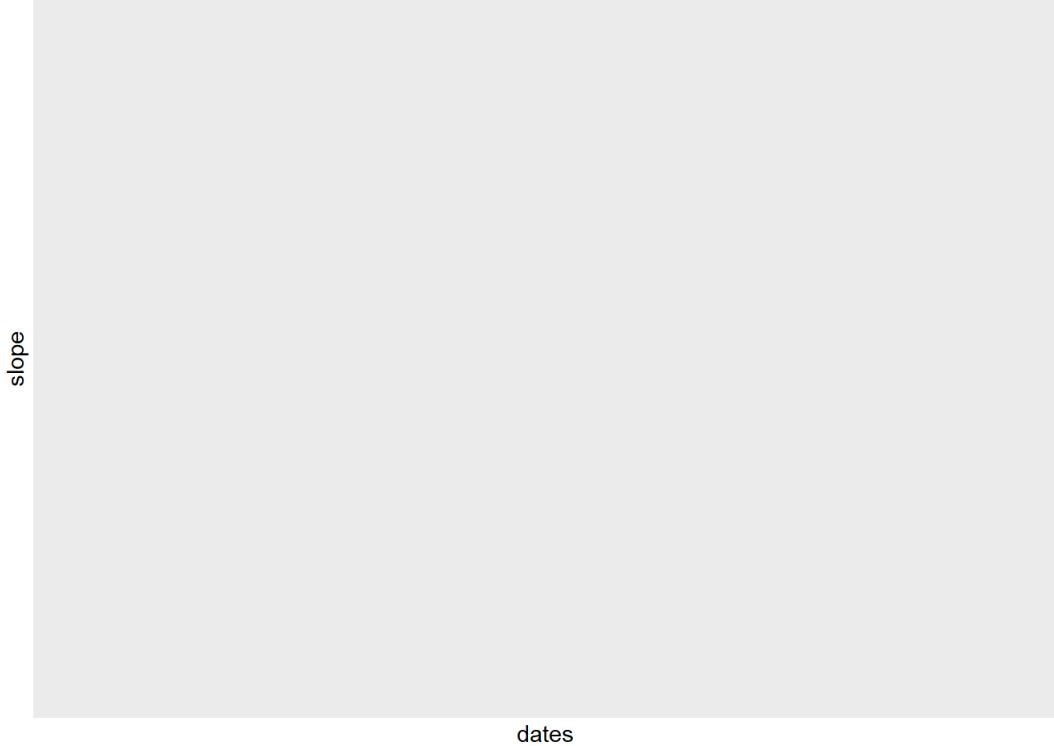
dates

```
## Warning: Using size for a discrete variable is not advised.
```

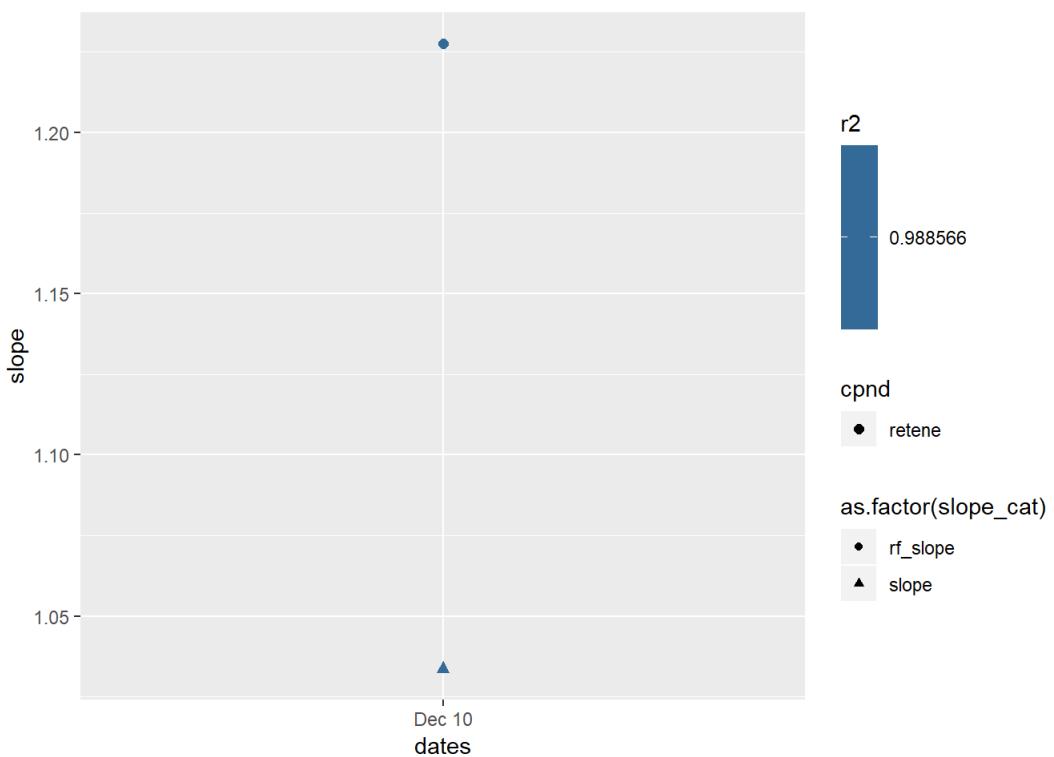
slope

dates

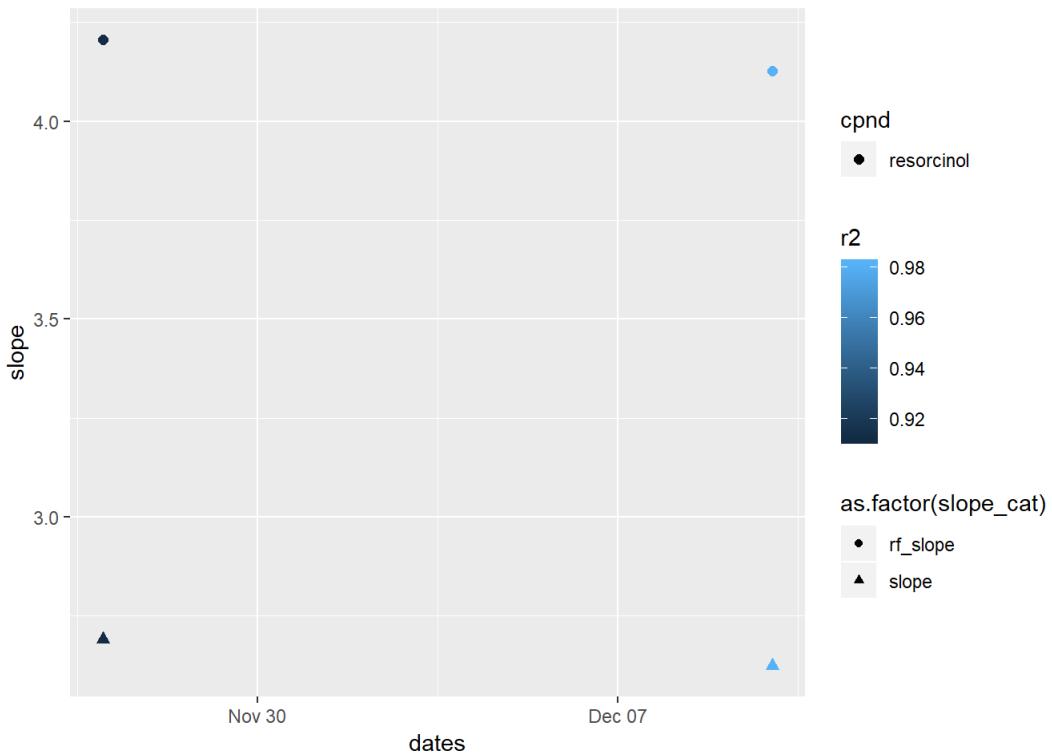
```
## Warning: Using size for a discrete variable is not advised.
```



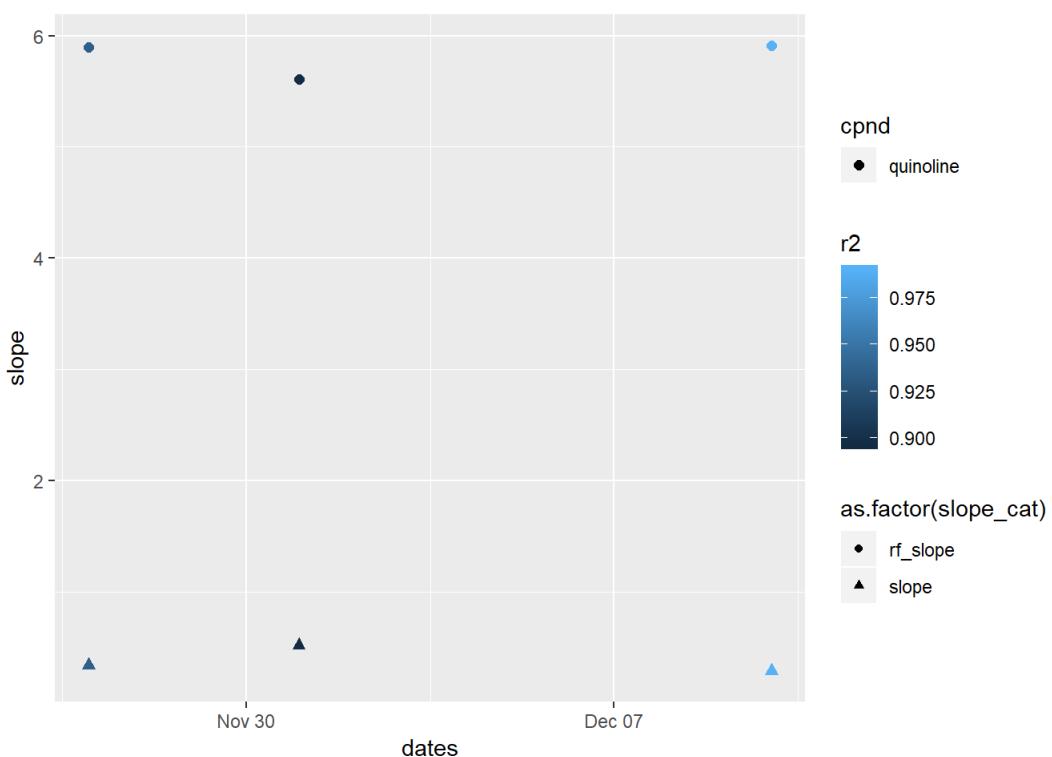
```
## Warning: Using size for a discrete variable is not advised.
```



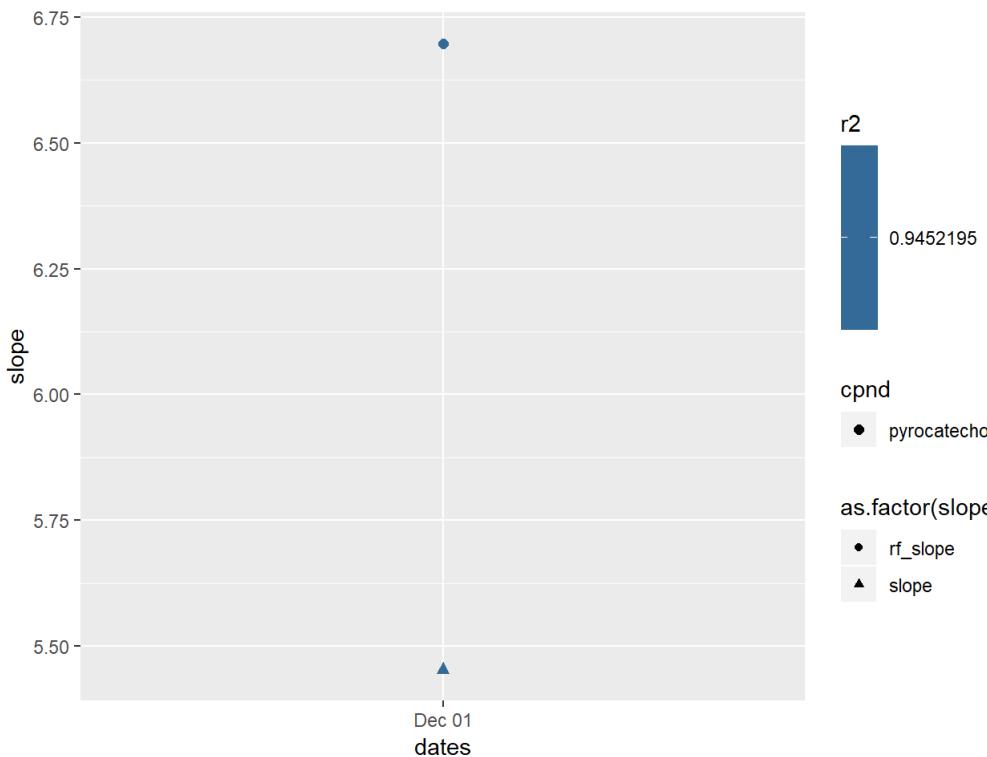
```
## Warning: Using size for a discrete variable is not advised.
```



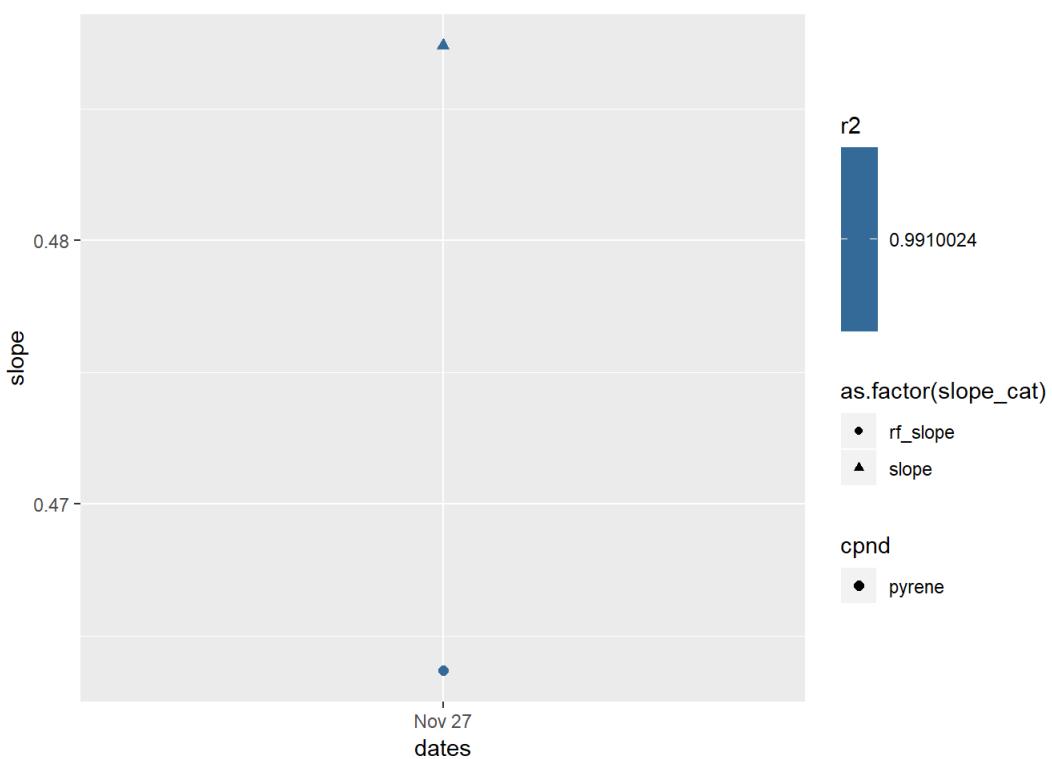
```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```

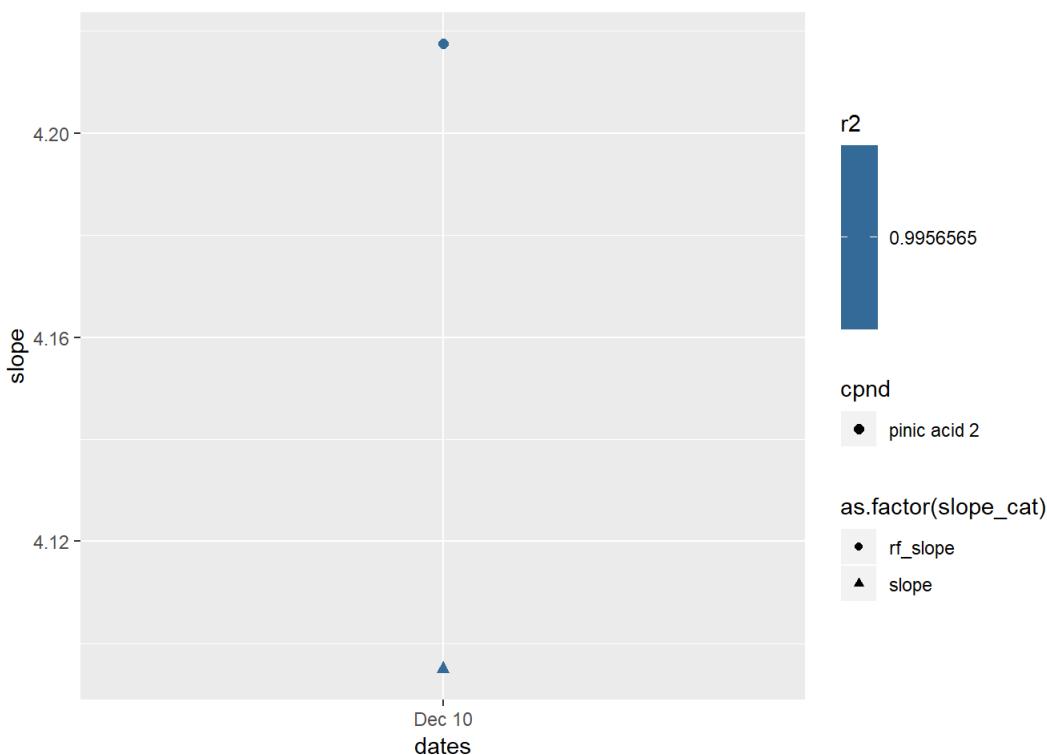


```
## Warning: Using size for a discrete variable is not advised.
```

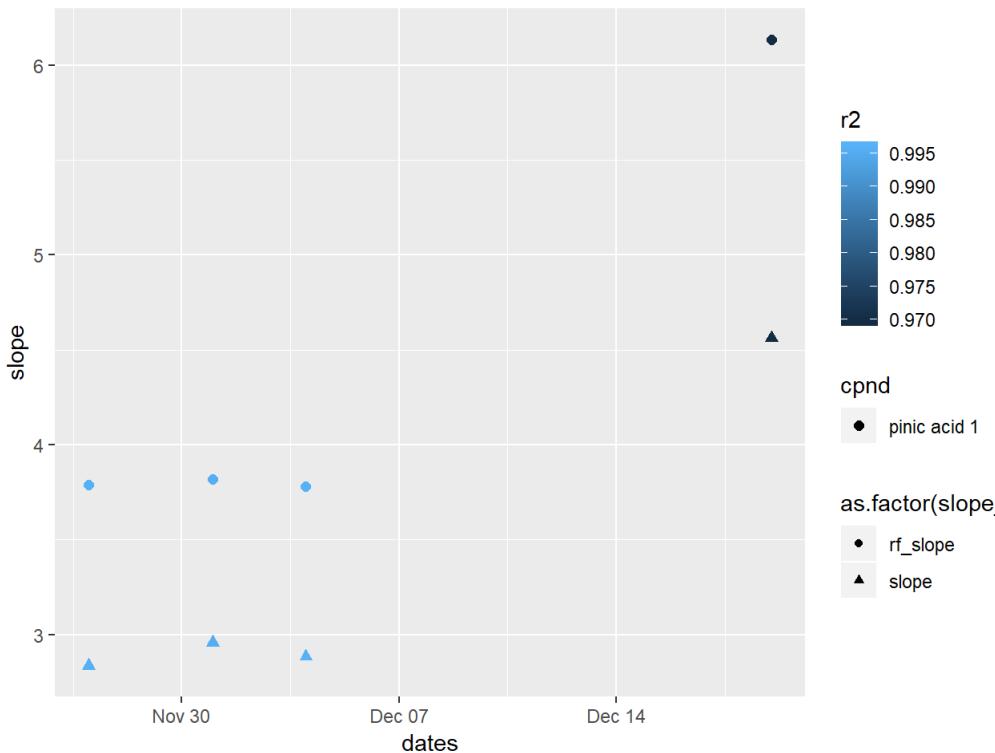
slope

dates

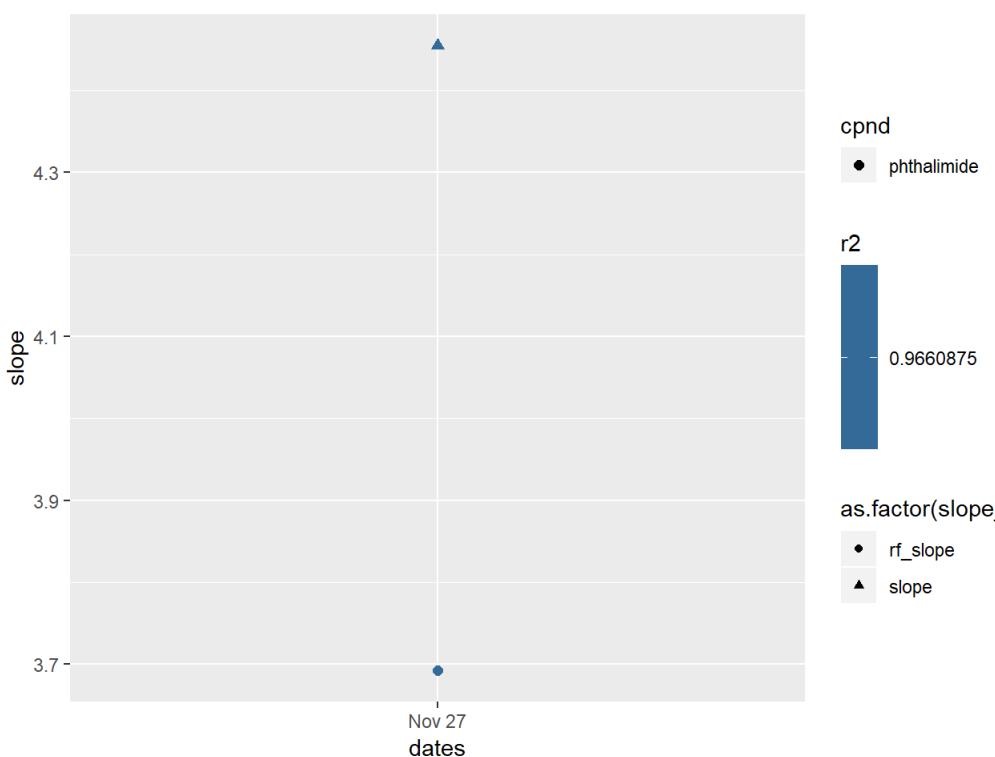
```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



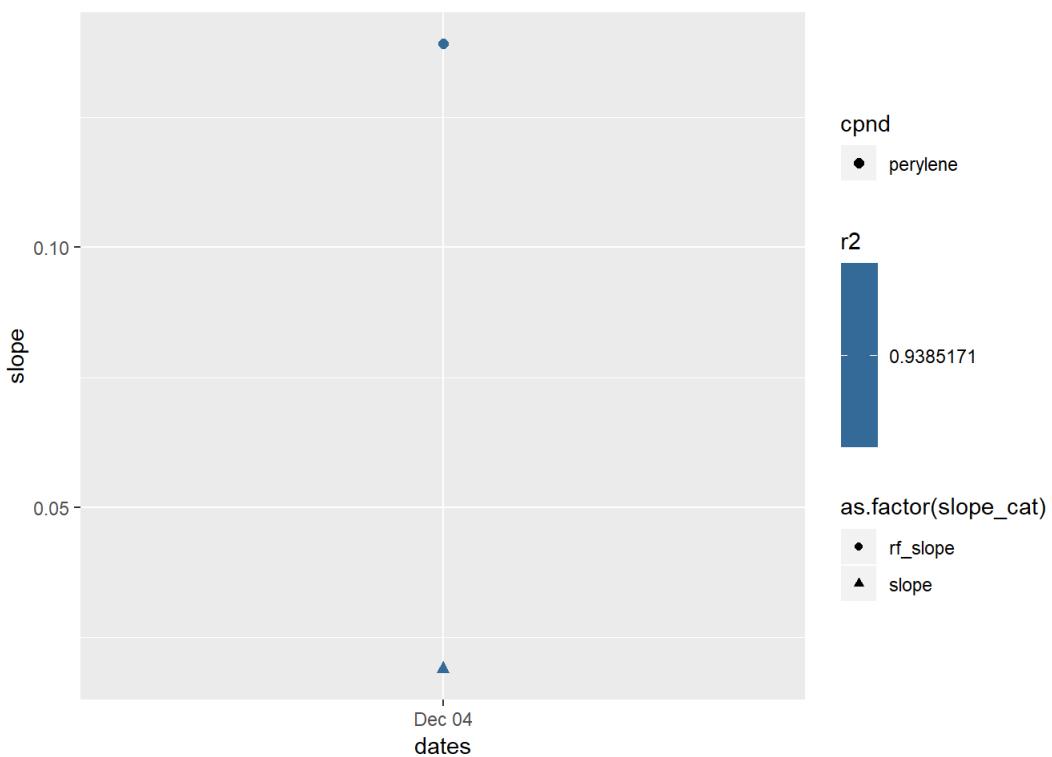
```
## Warning: Using size for a discrete variable is not advised.
```



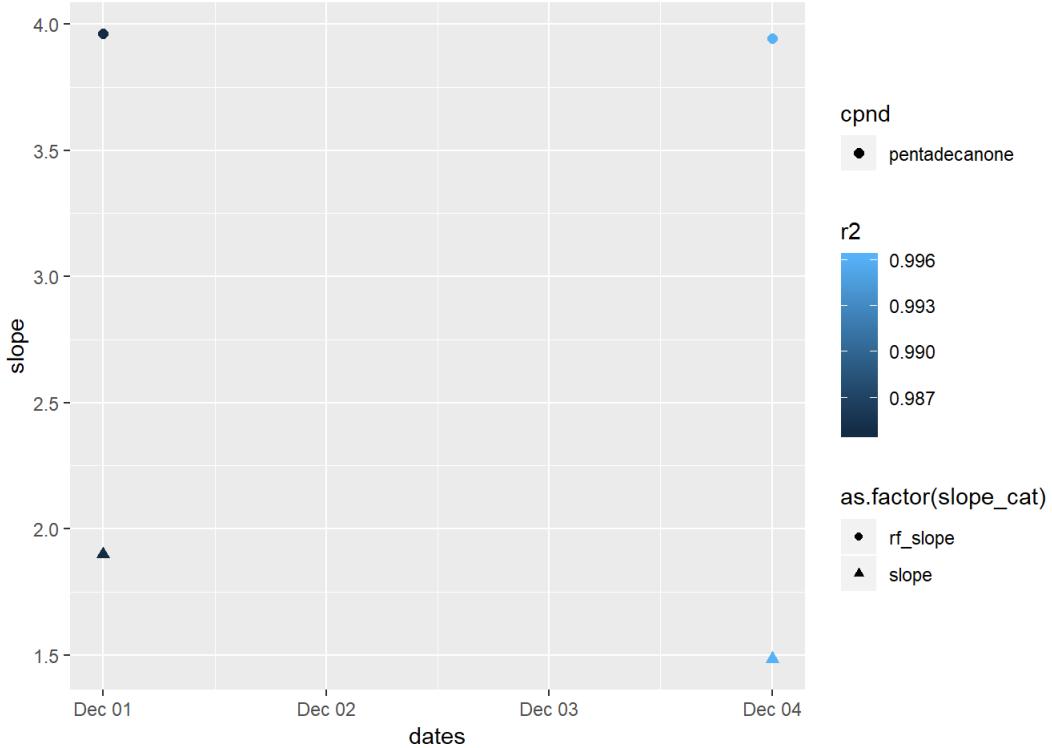
```
## Warning: Using size for a discrete variable is not advised.
```



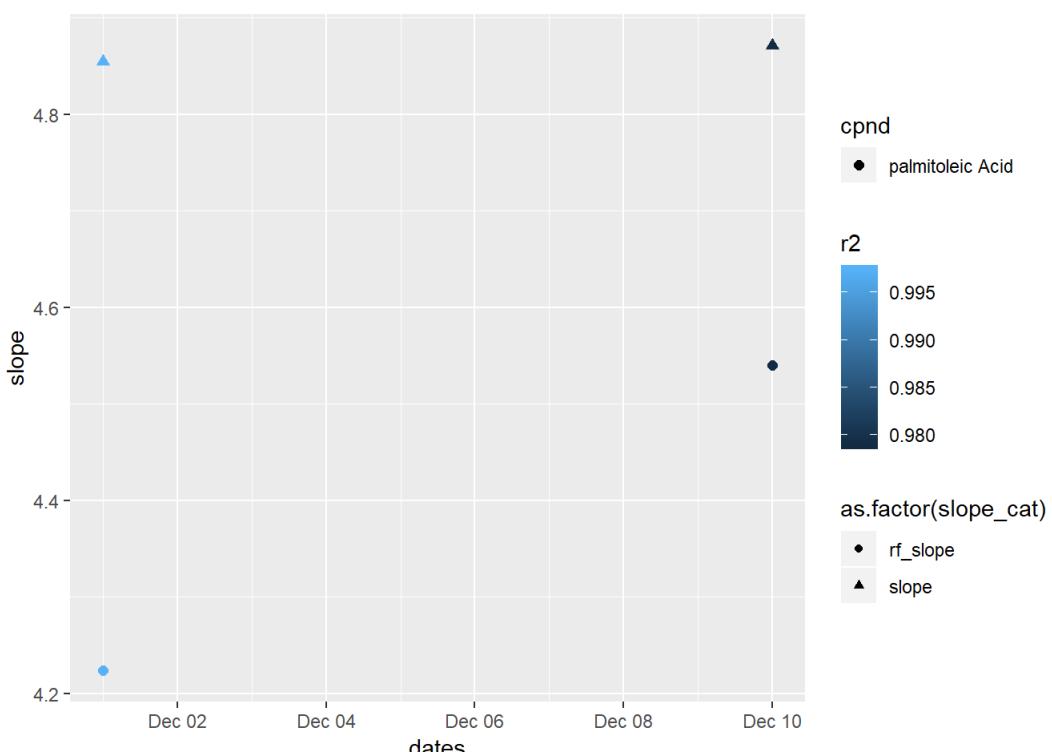
```
## Warning: Using size for a discrete variable is not advised.
```



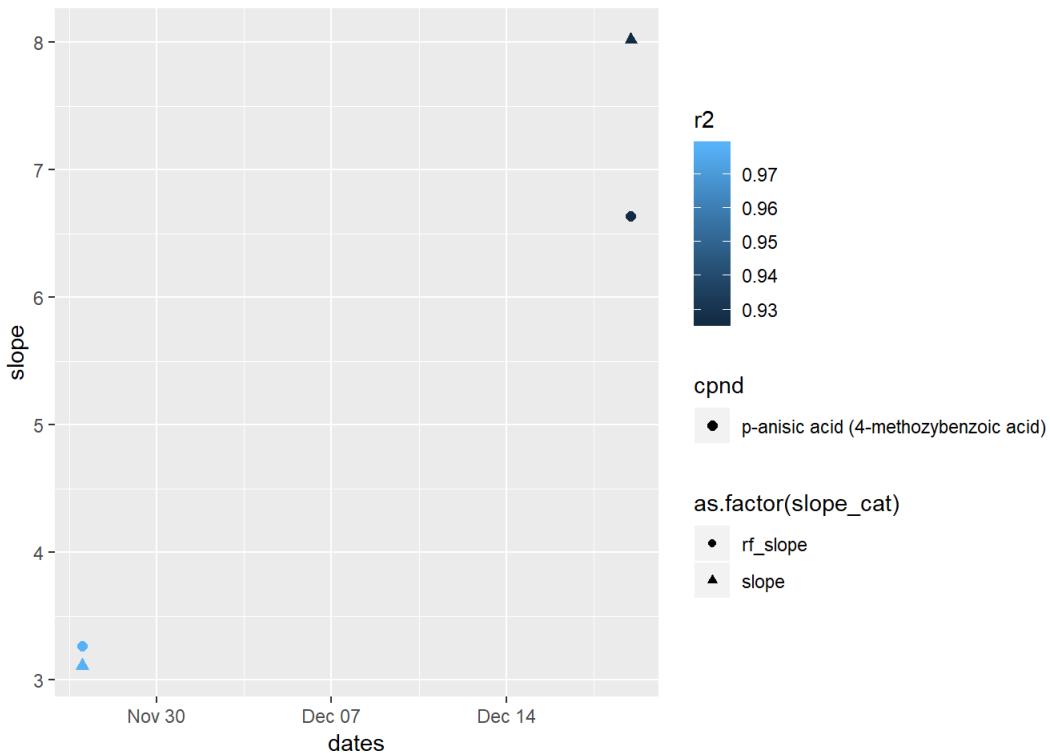
```
## Warning: Using size for a discrete variable is not advised.
```



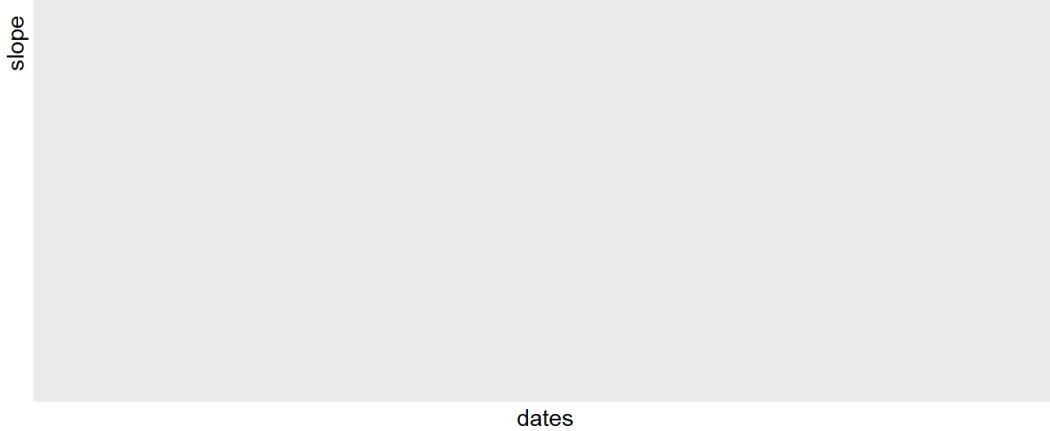
```
## Warning: Using size for a discrete variable is not advised.
```



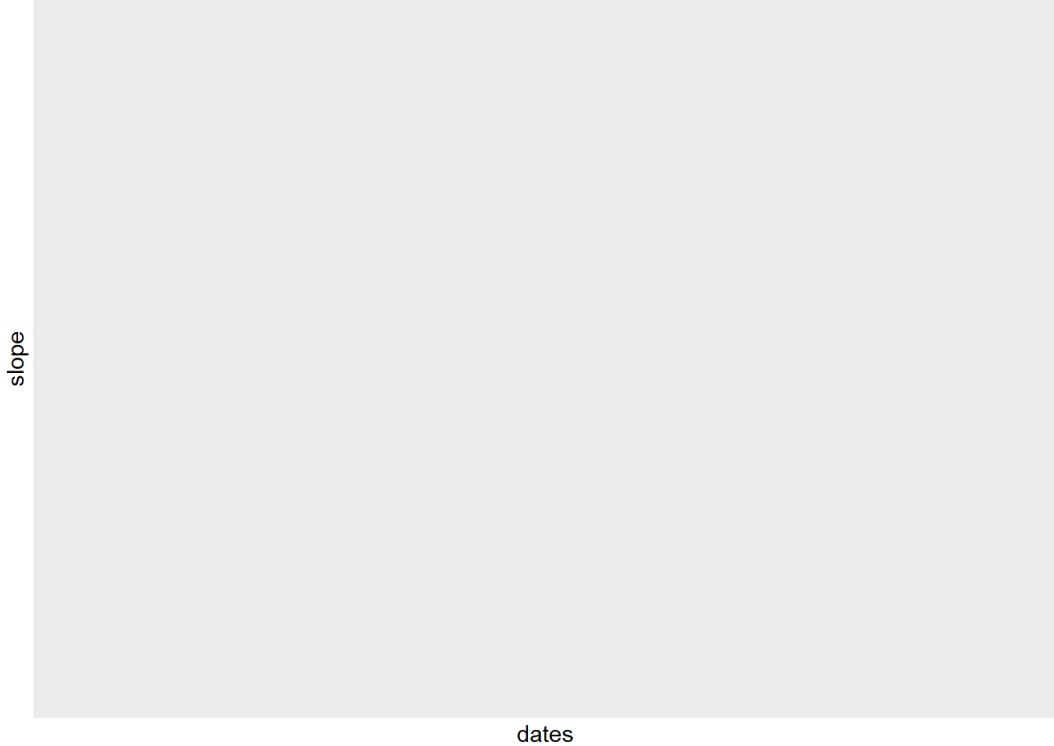
```
## Warning: Using size for a discrete variable is not advised.
```



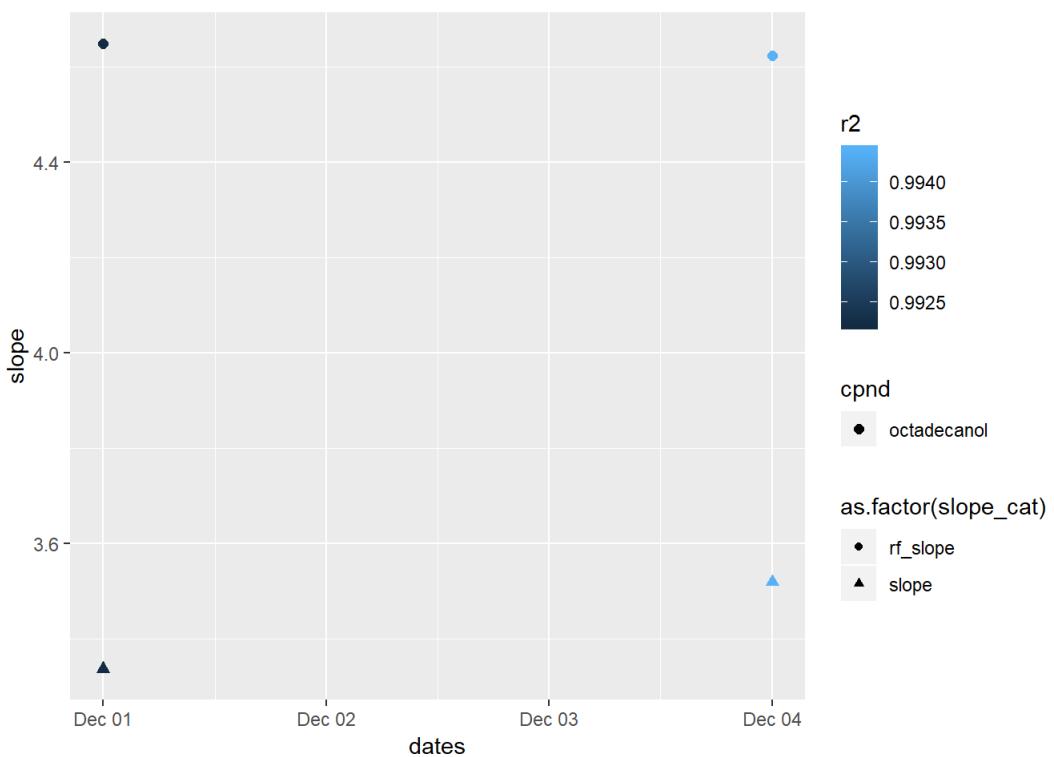
```
## Warning: Using size for a discrete variable is not advised.
```



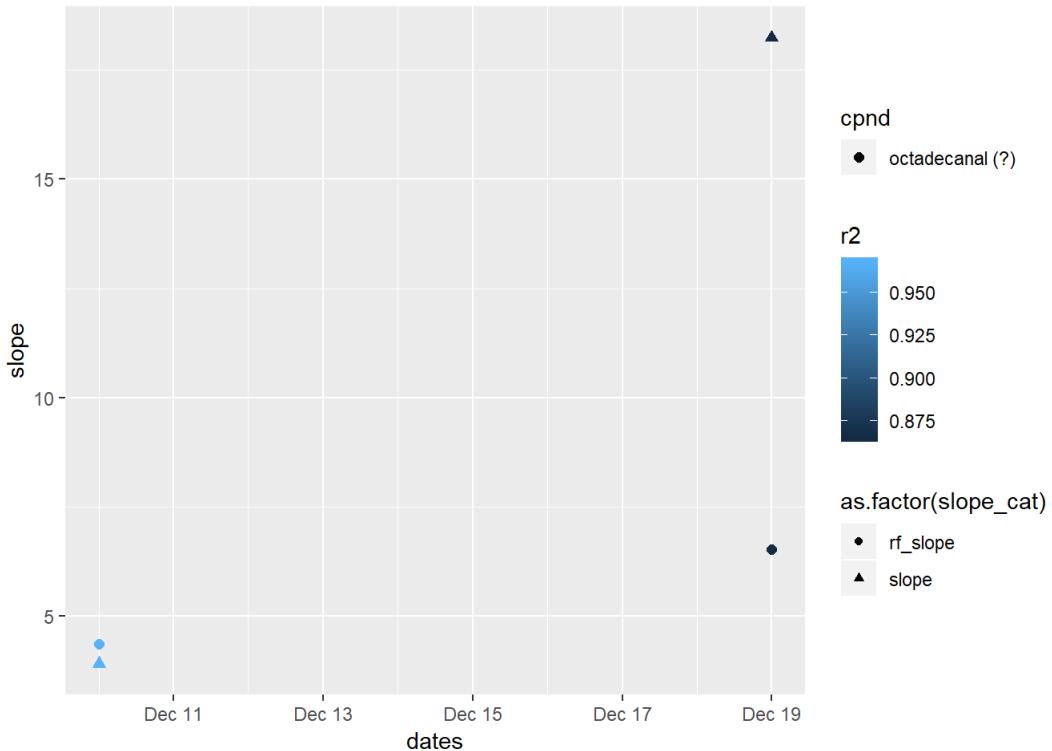
```
## Warning: Using size for a discrete variable is not advised.
```



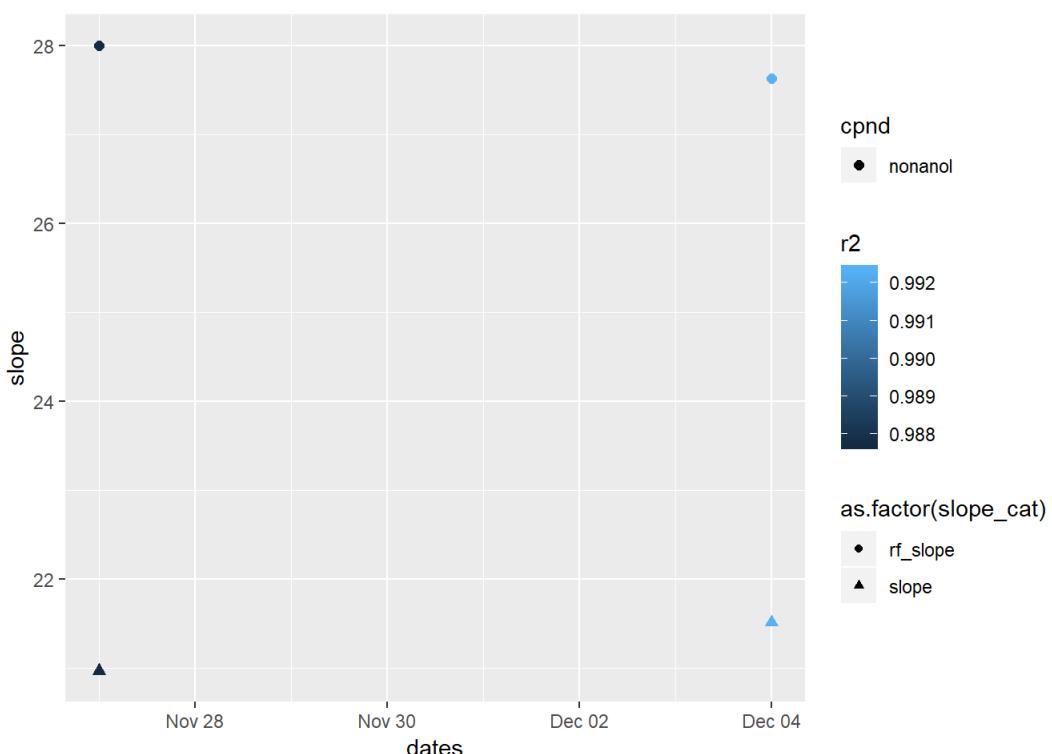
```
## Warning: Using size for a discrete variable is not advised.
```



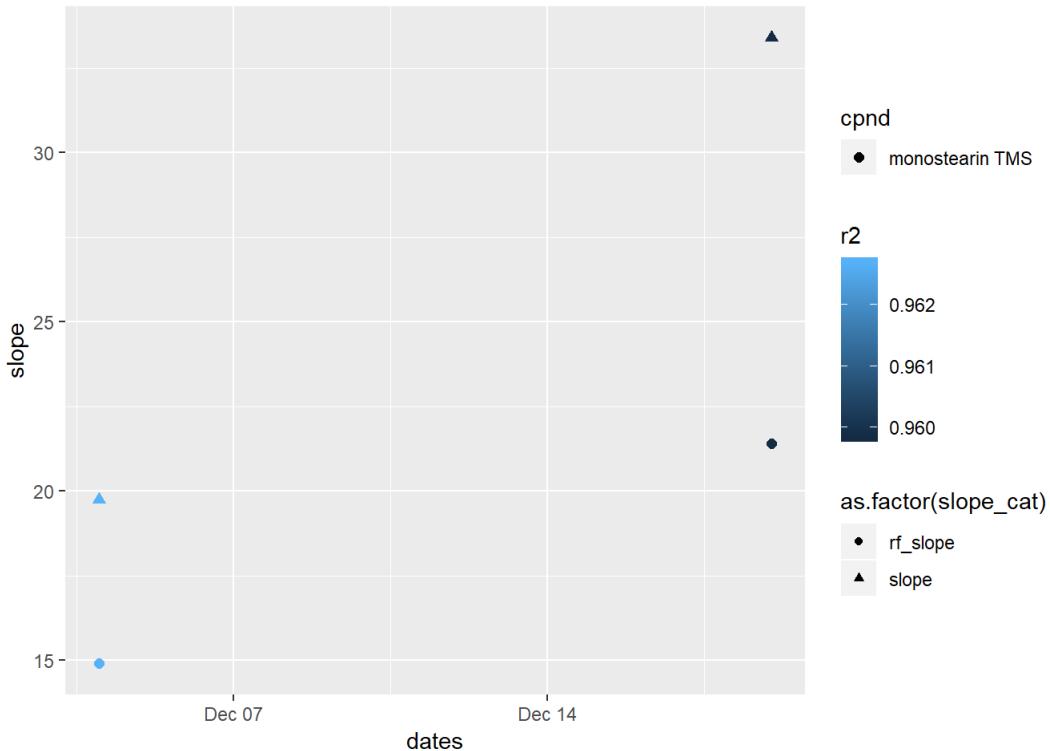
```
## Warning: Using size for a discrete variable is not advised.
```



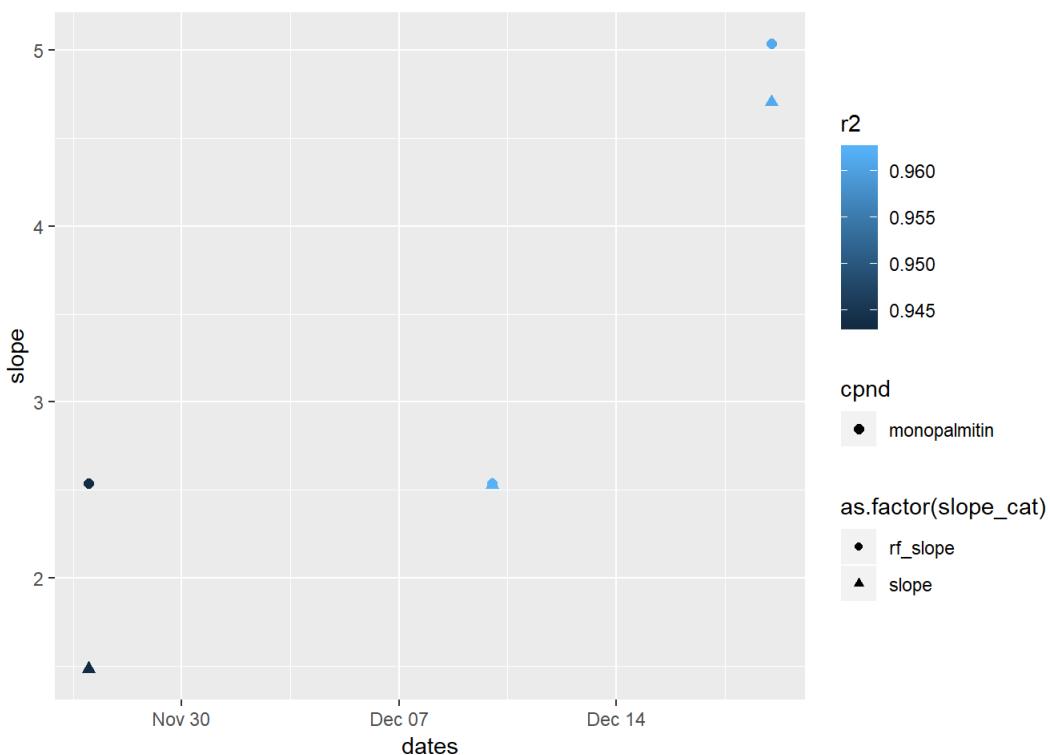
```
## Warning: Using size for a discrete variable is not advised.
```



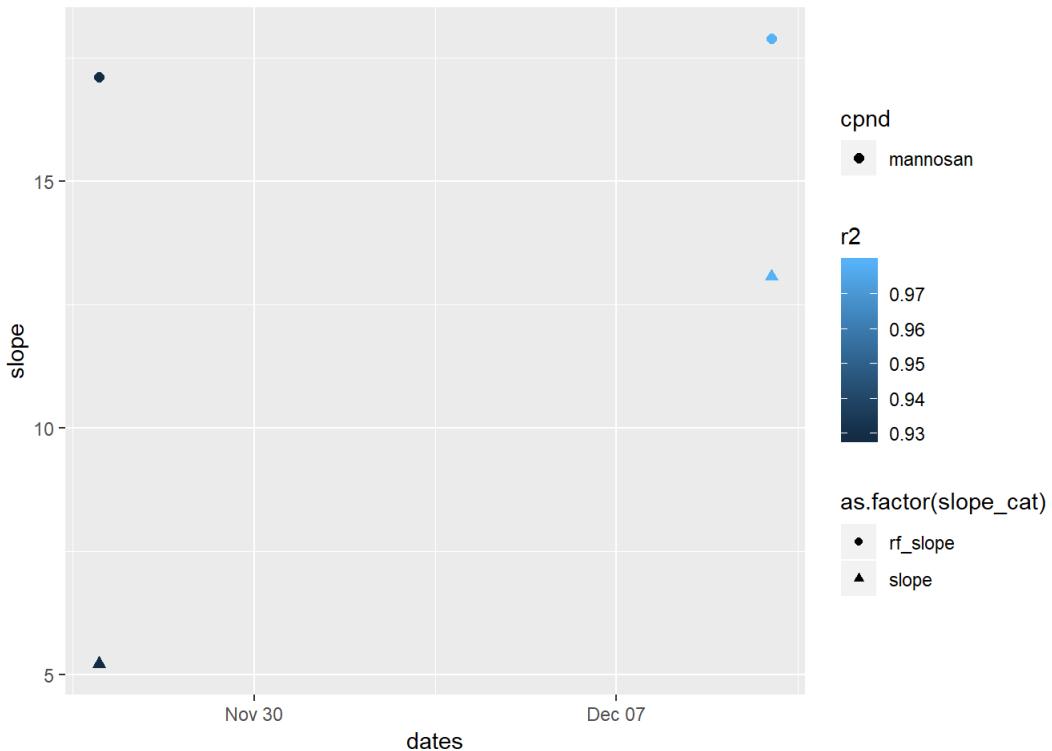
```
## Warning: Using size for a discrete variable is not advised.
```



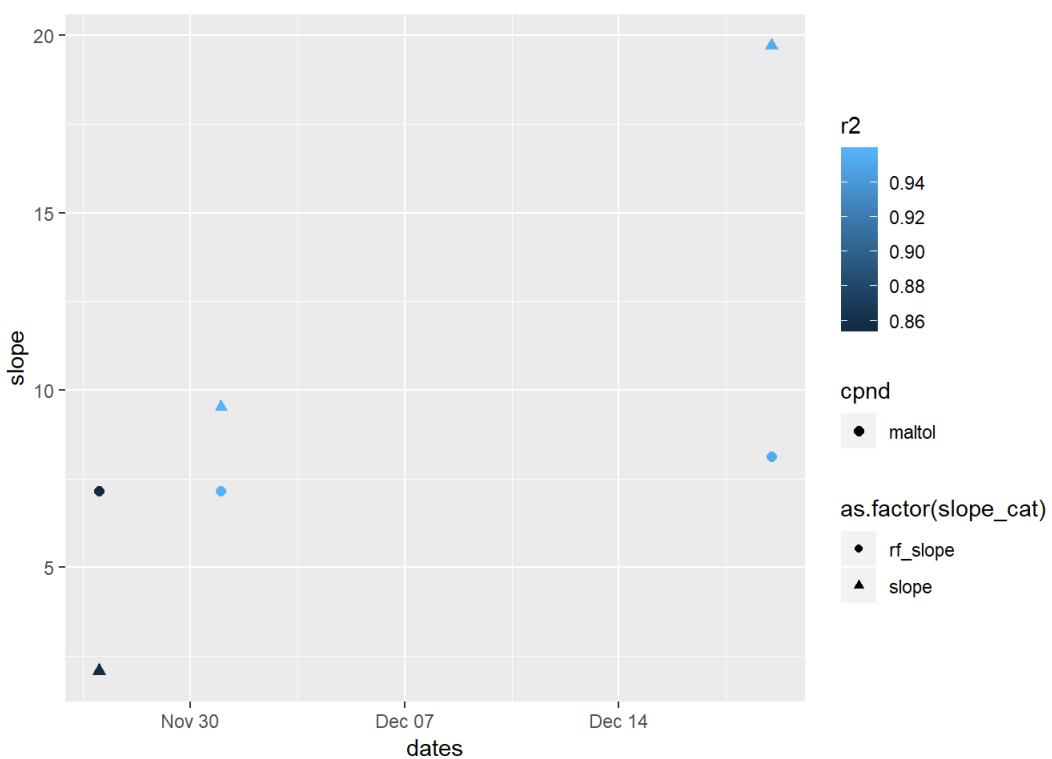
```
## Warning: Using size for a discrete variable is not advised.
```



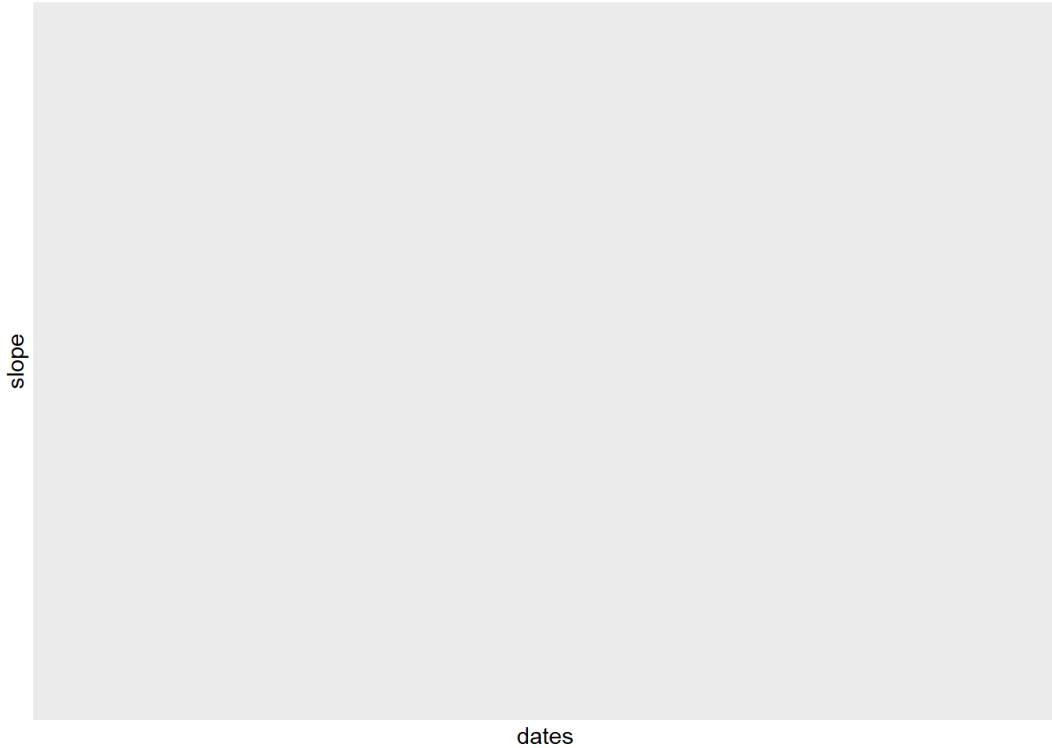
```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```

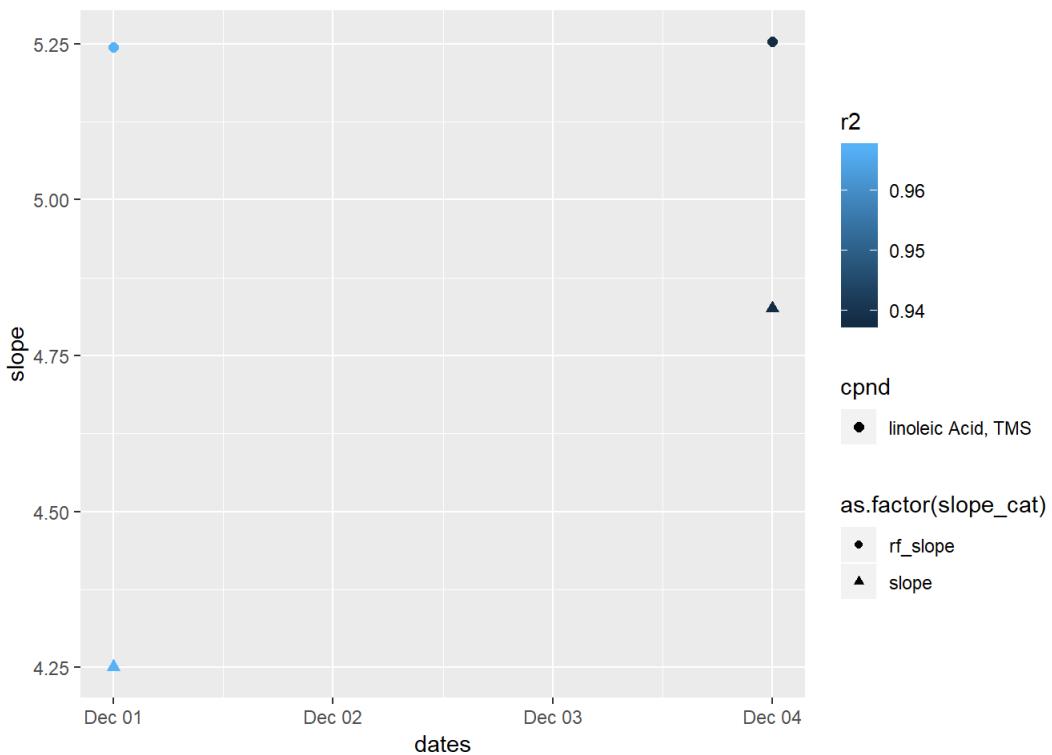


```
## Warning: Using size for a discrete variable is not advised.
```

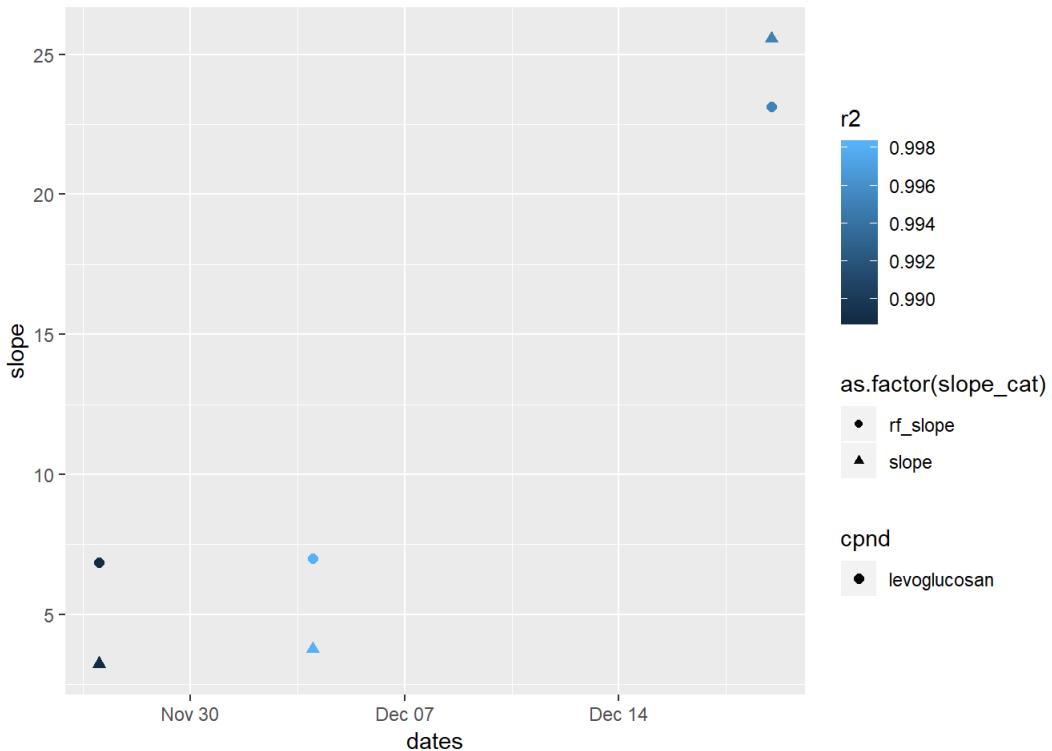


dates

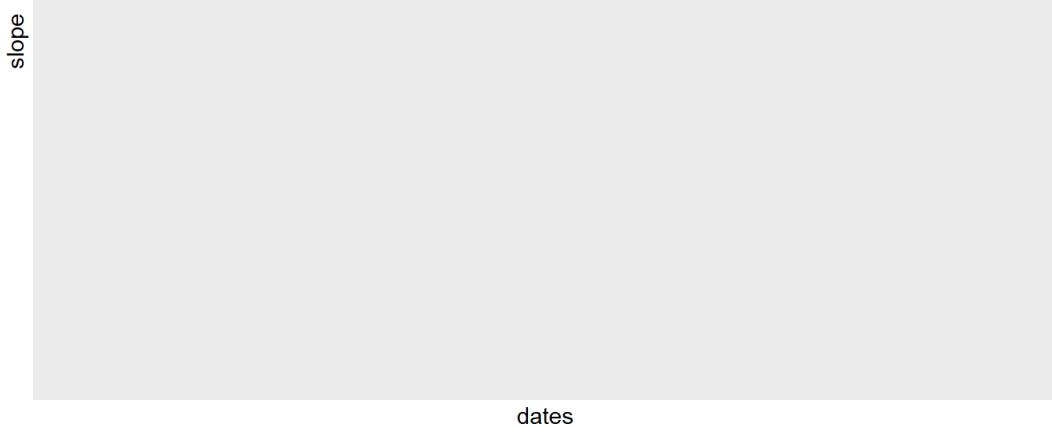
```
## Warning: Using size for a discrete variable is not advised.
```



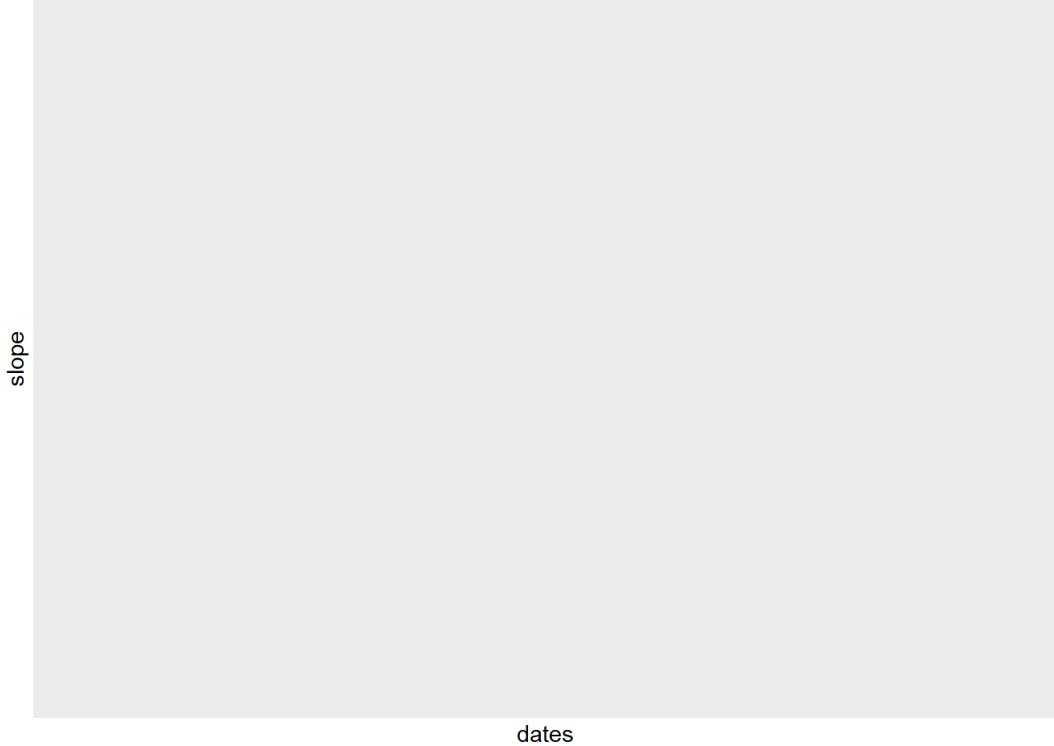
```
## Warning: Using size for a discrete variable is not advised.
```



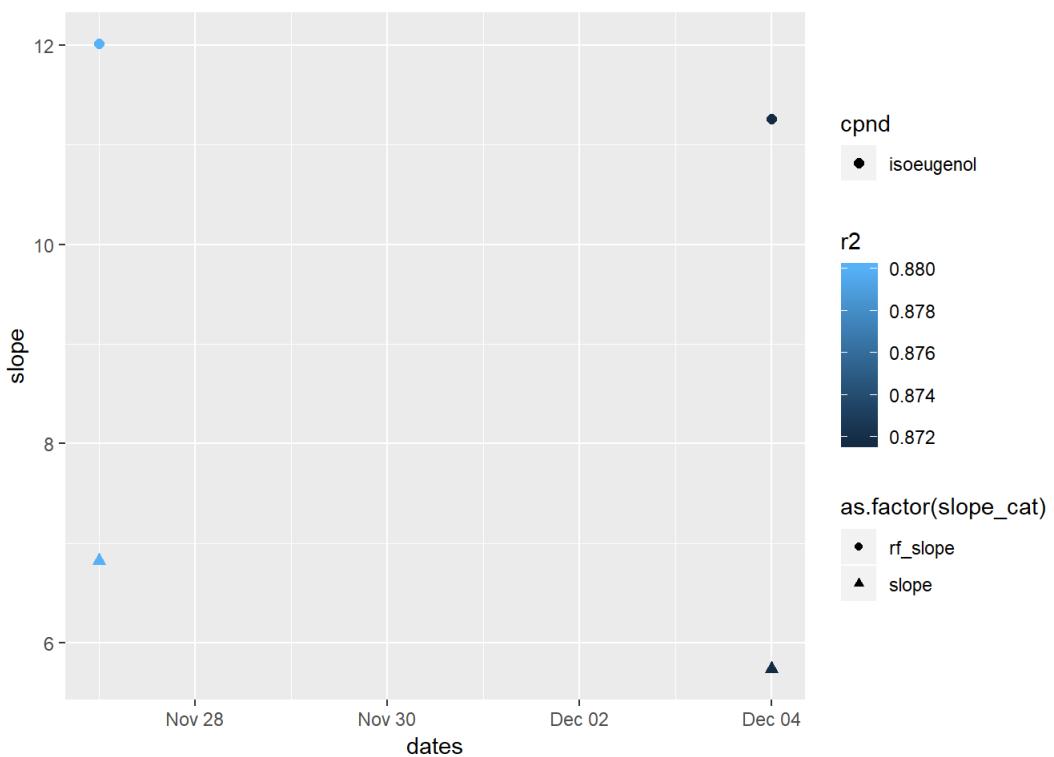
```
## Warning: Using size for a discrete variable is not advised.
```



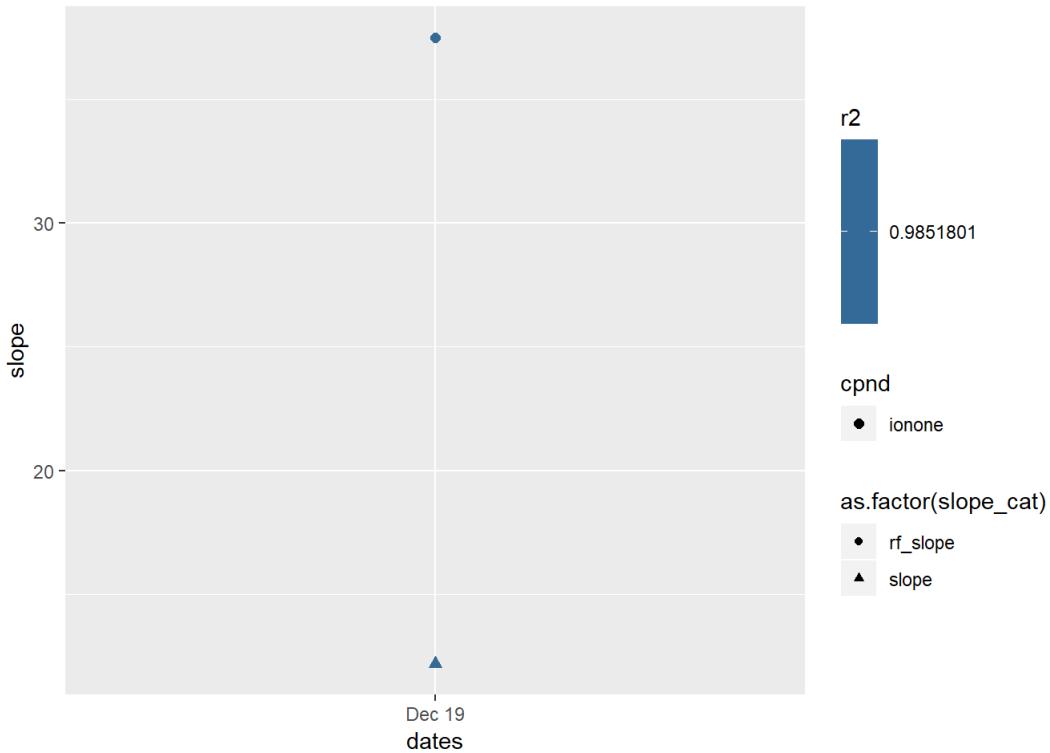
```
## Warning: Using size for a discrete variable is not advised.
```



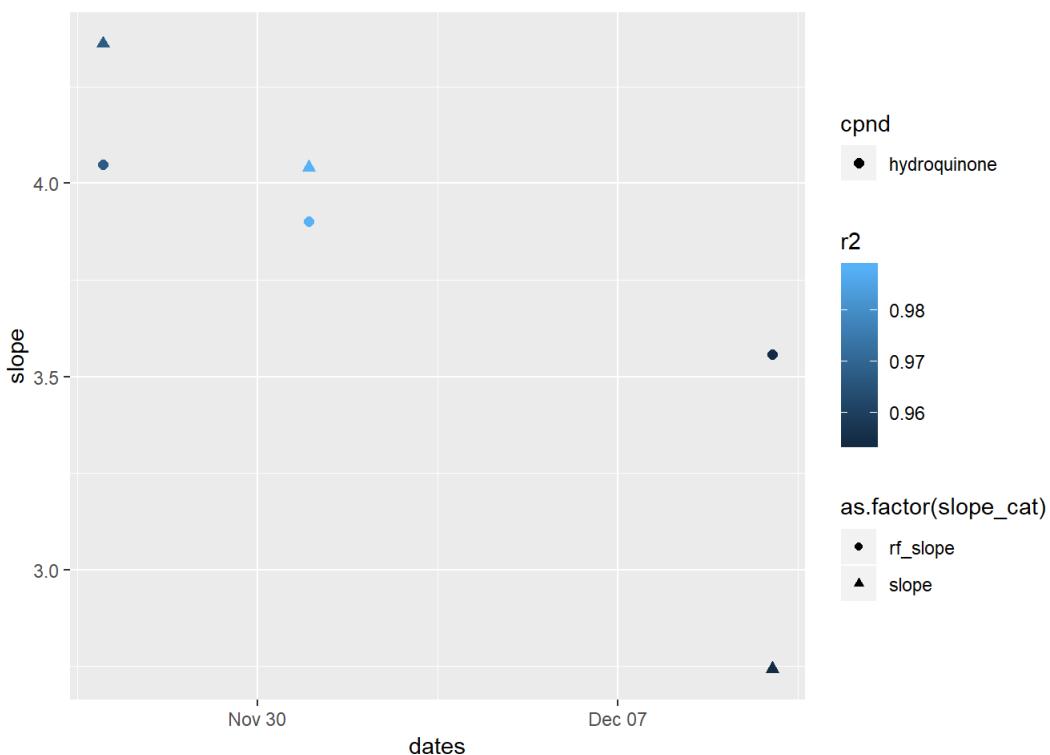
```
## Warning: Using size for a discrete variable is not advised.
```



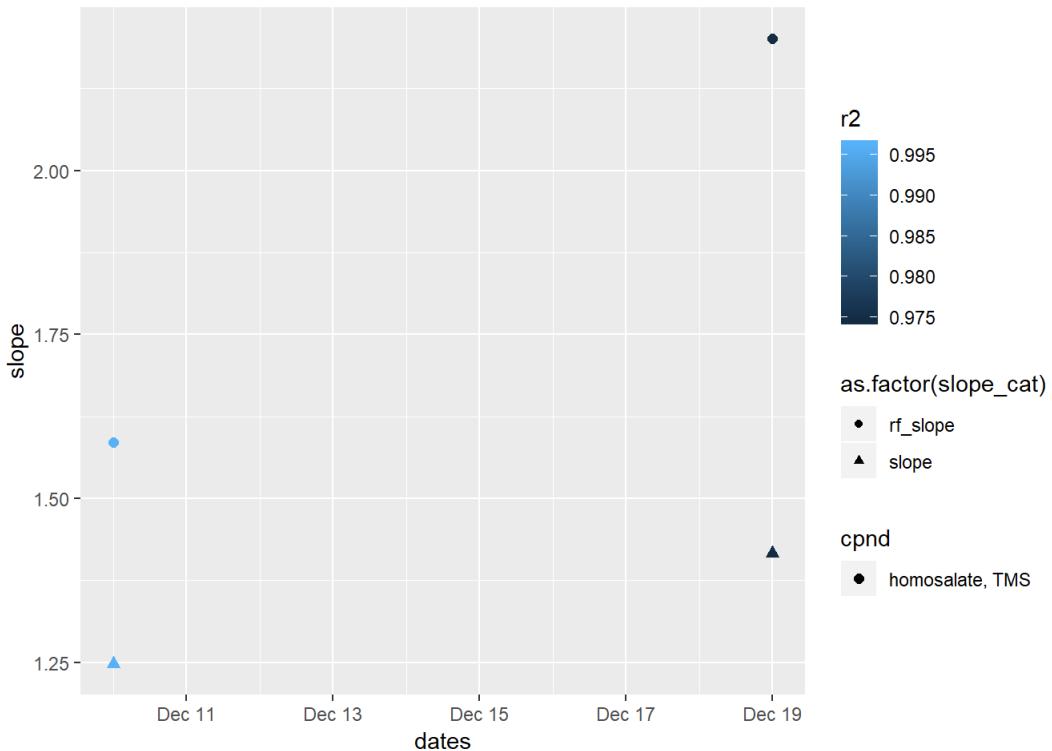
```
## Warning: Using size for a discrete variable is not advised.
```



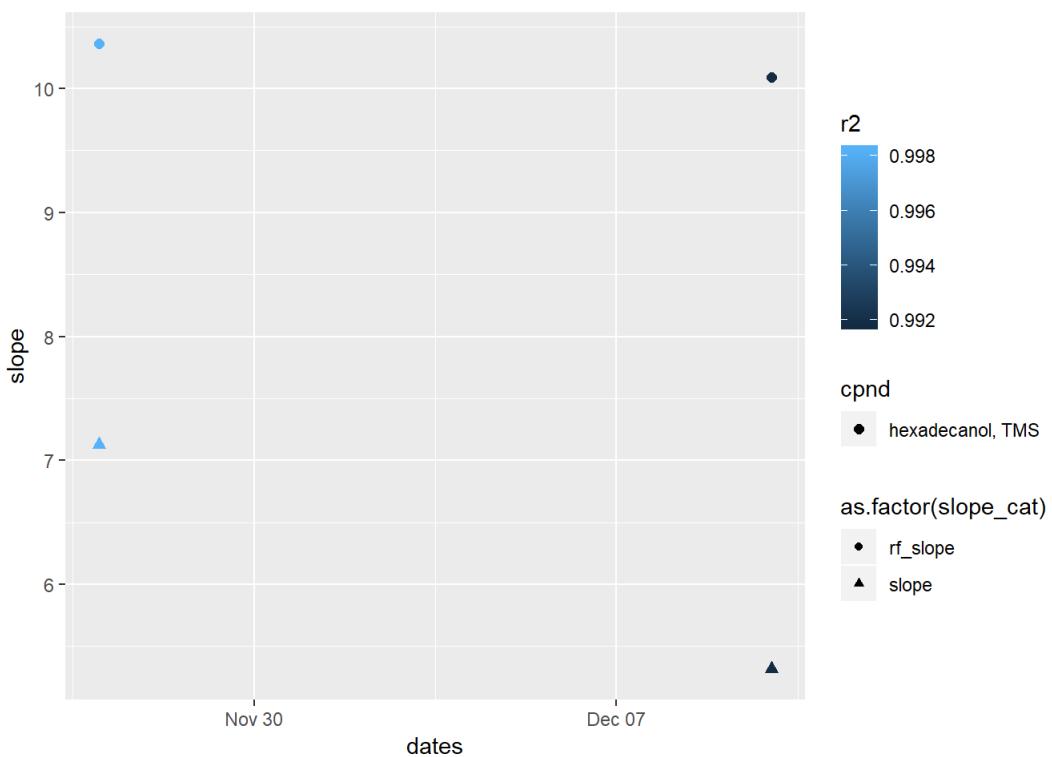
```
## Warning: Using size for a discrete variable is not advised.
```



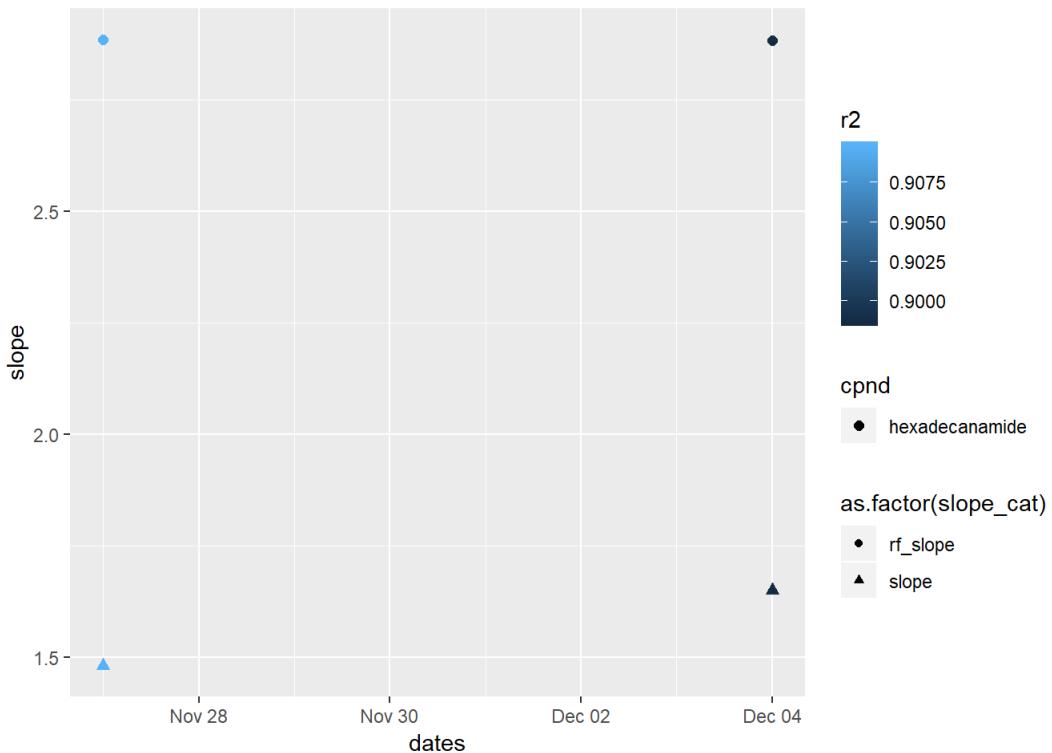
```
## Warning: Using size for a discrete variable is not advised.
```



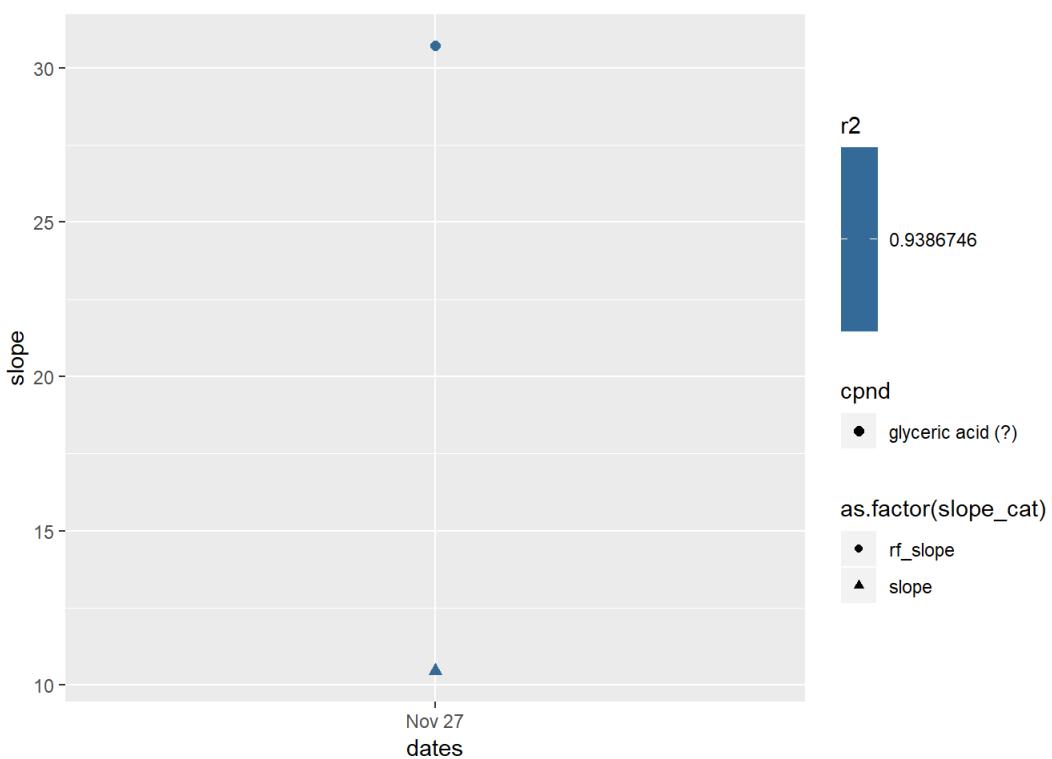
```
## Warning: Using size for a discrete variable is not advised.
```



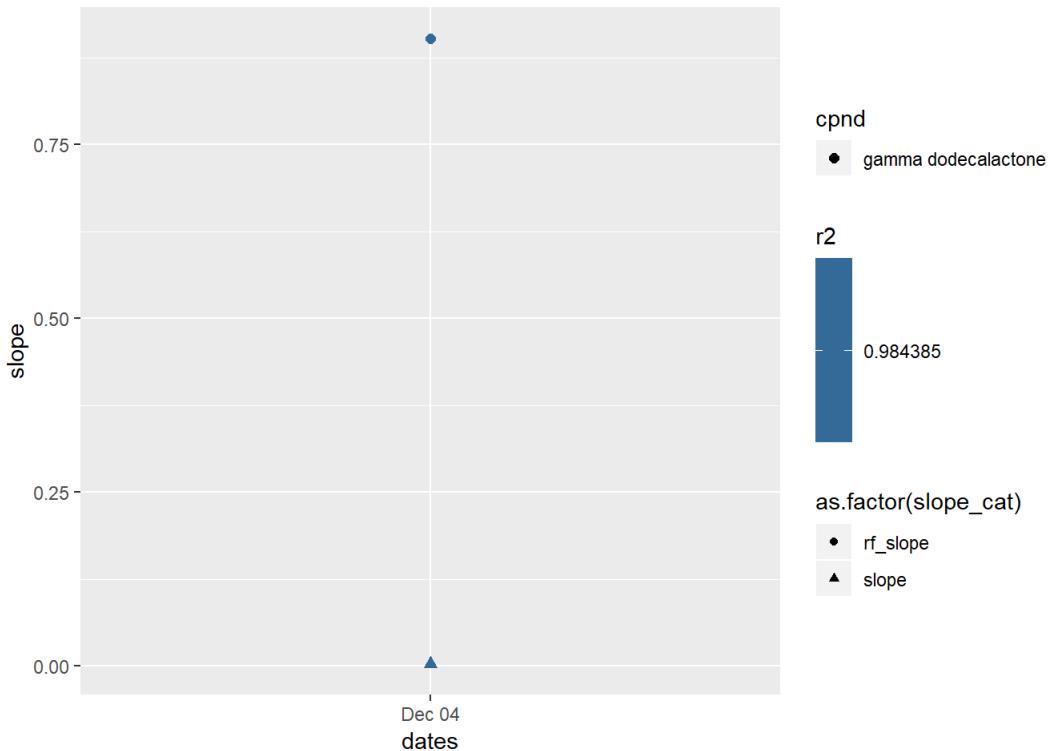
```
## Warning: Using size for a discrete variable is not advised.
```



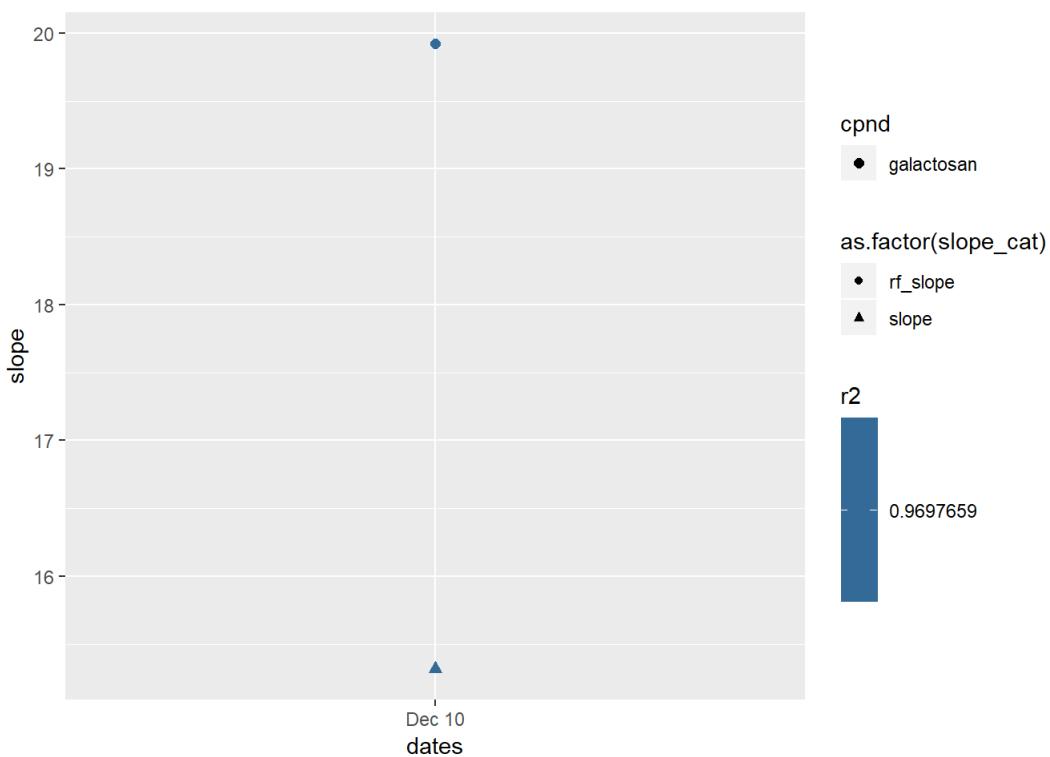
```
## Warning: Using size for a discrete variable is not advised.
```



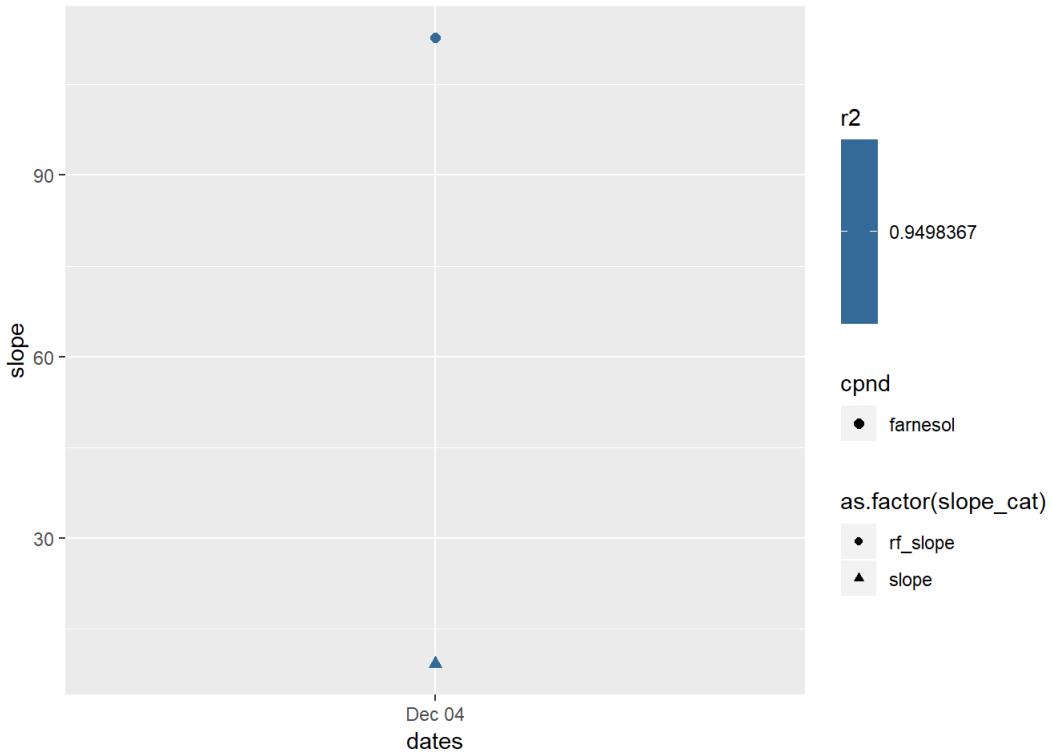
```
## Warning: Using size for a discrete variable is not advised.
```



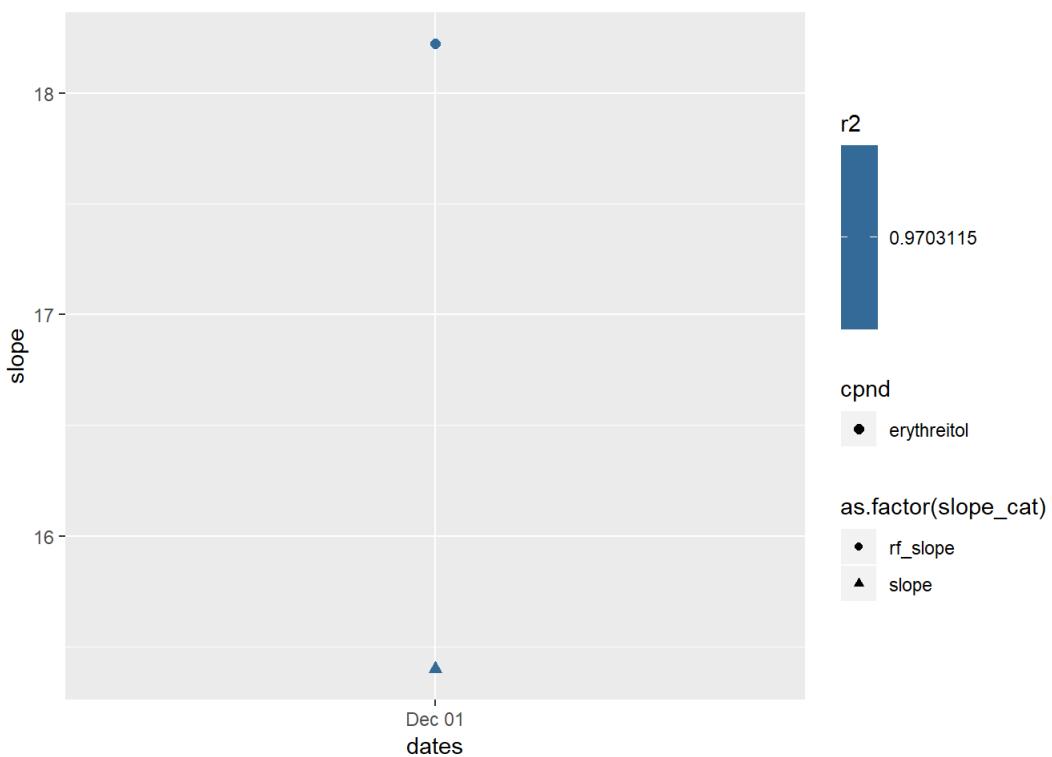
```
## Warning: Using size for a discrete variable is not advised.
```



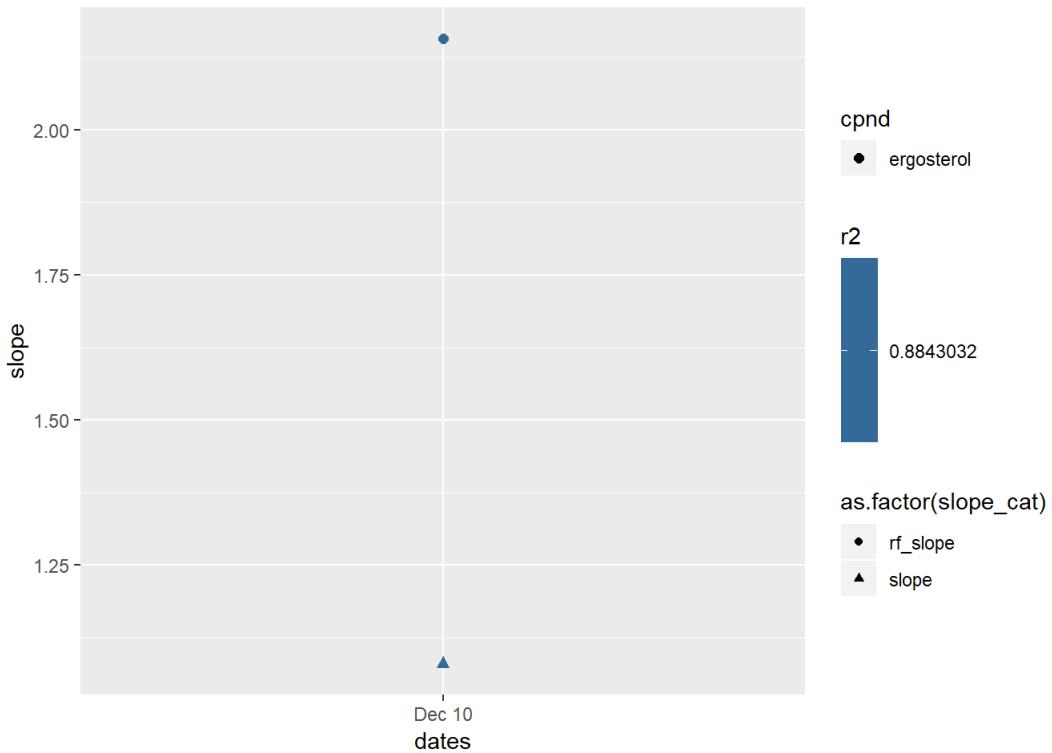
```
## Warning: Using size for a discrete variable is not advised.
```



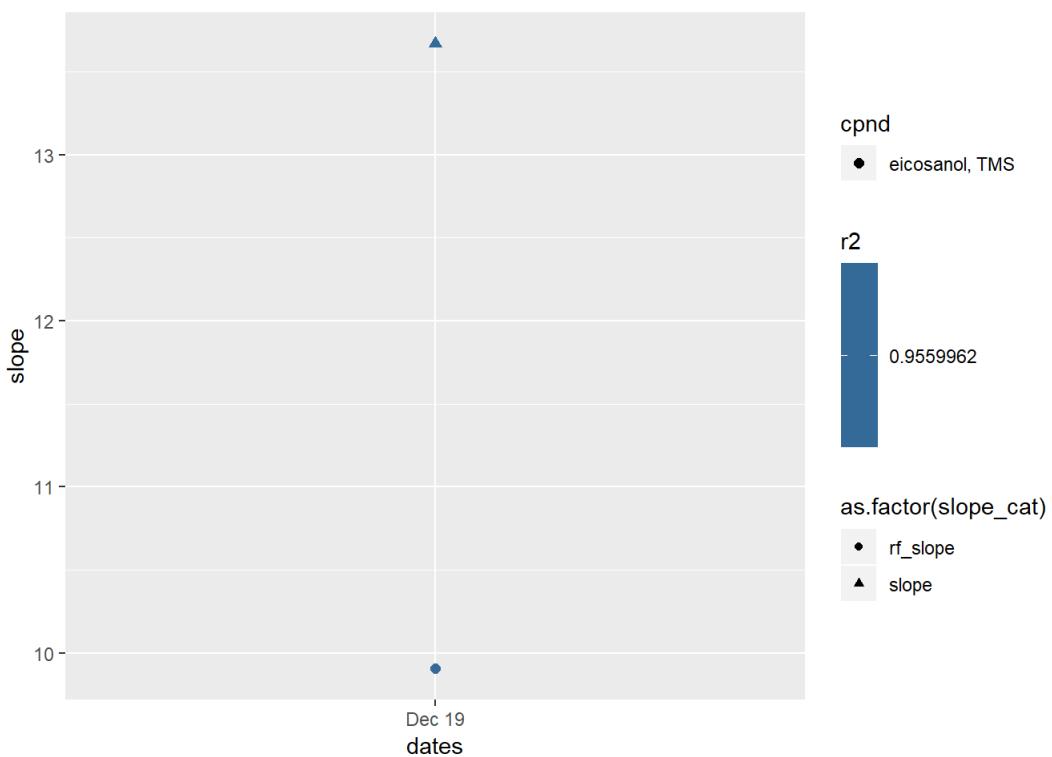
```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```

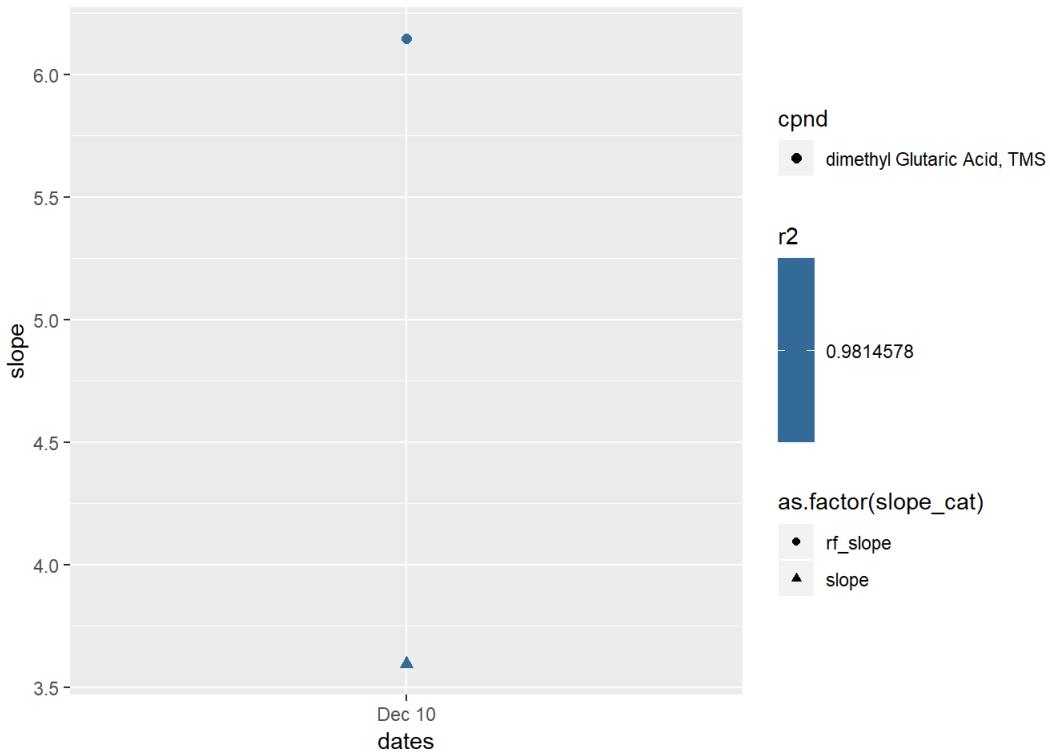


```
## Warning: Using size for a discrete variable is not advised.
```

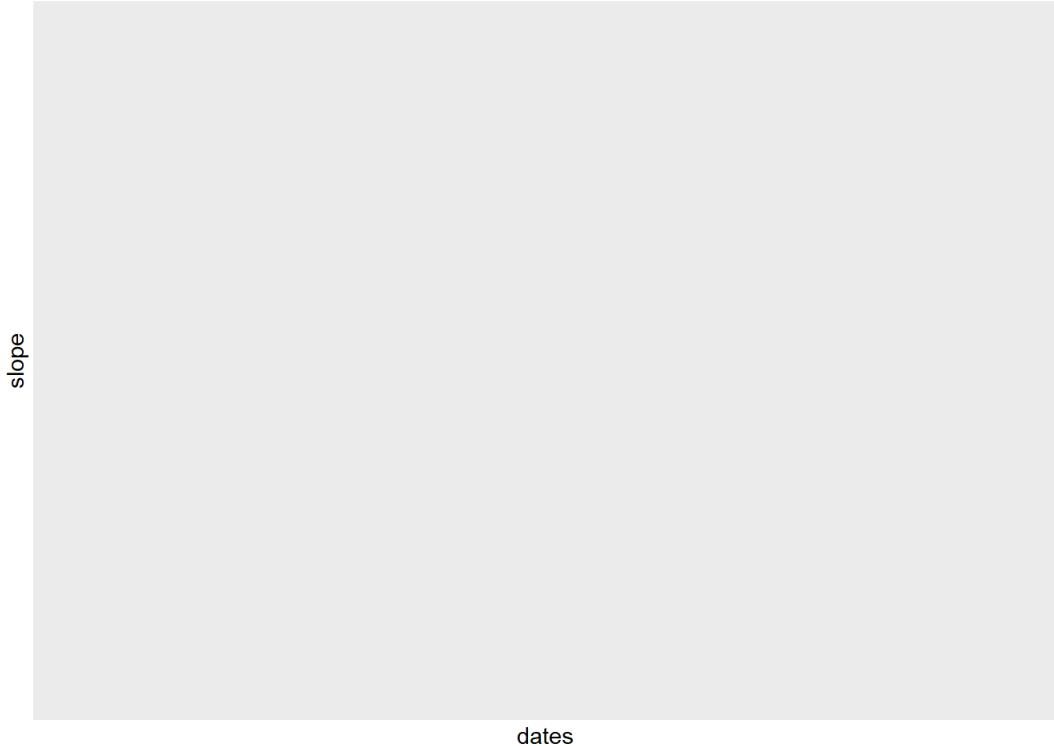
slope

dates

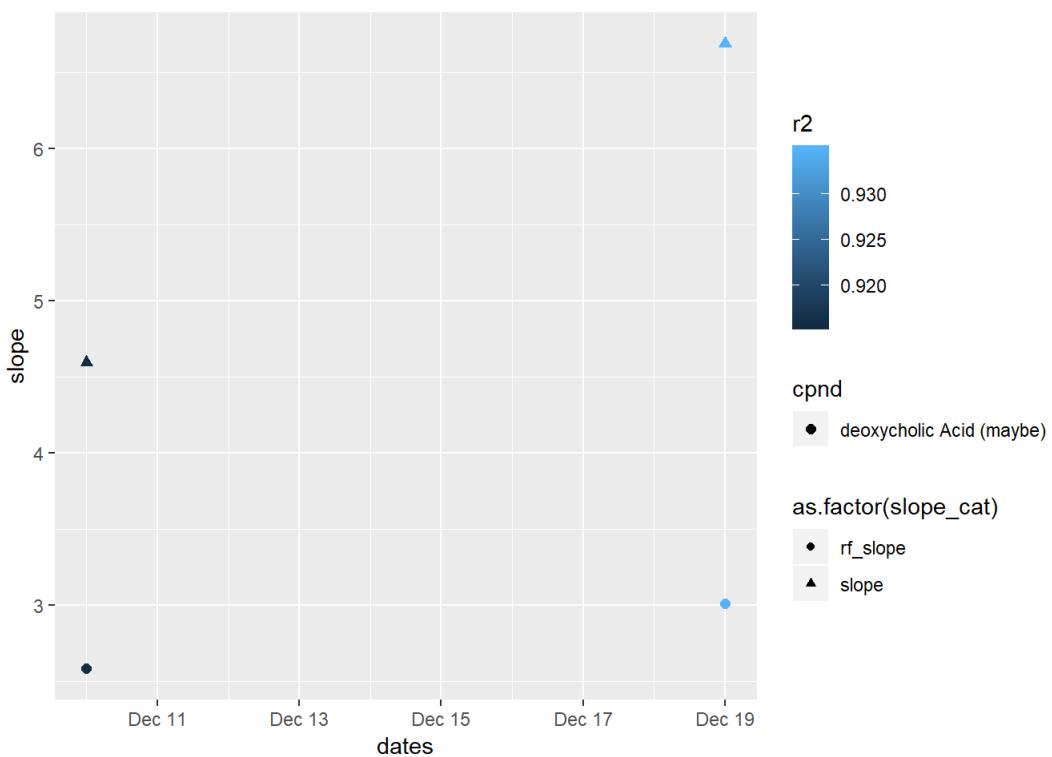
```
## Warning: Using size for a discrete variable is not advised.
```



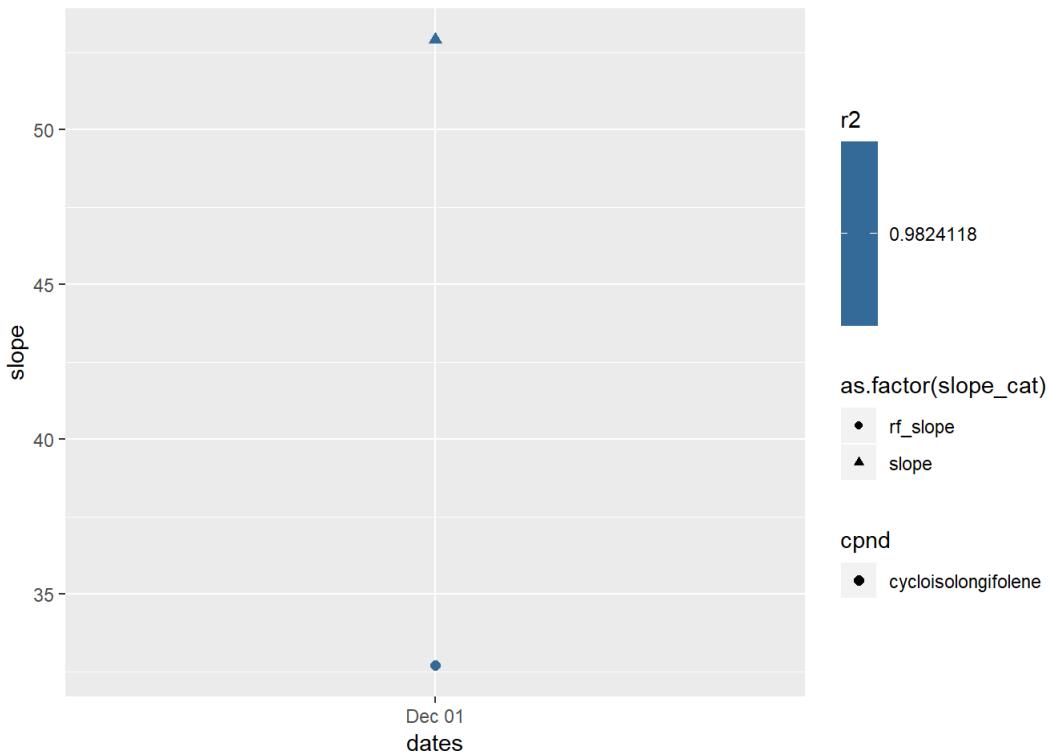
```
## Warning: Using size for a discrete variable is not advised.
```



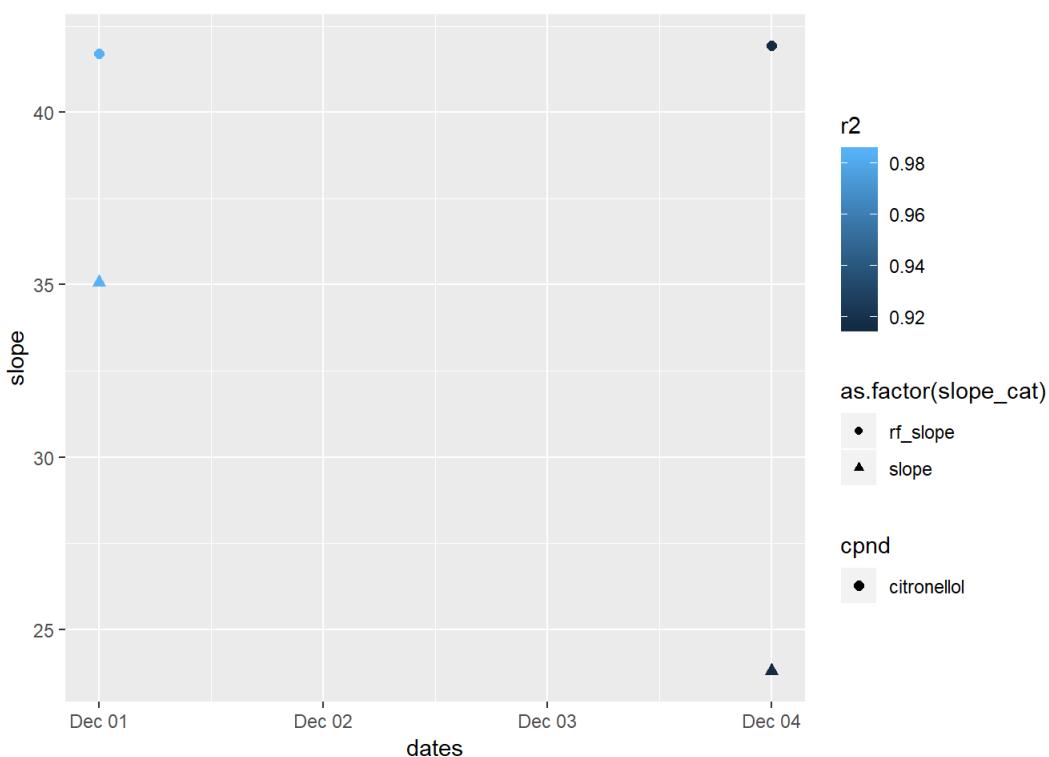
```
## Warning: Using size for a discrete variable is not advised.
```



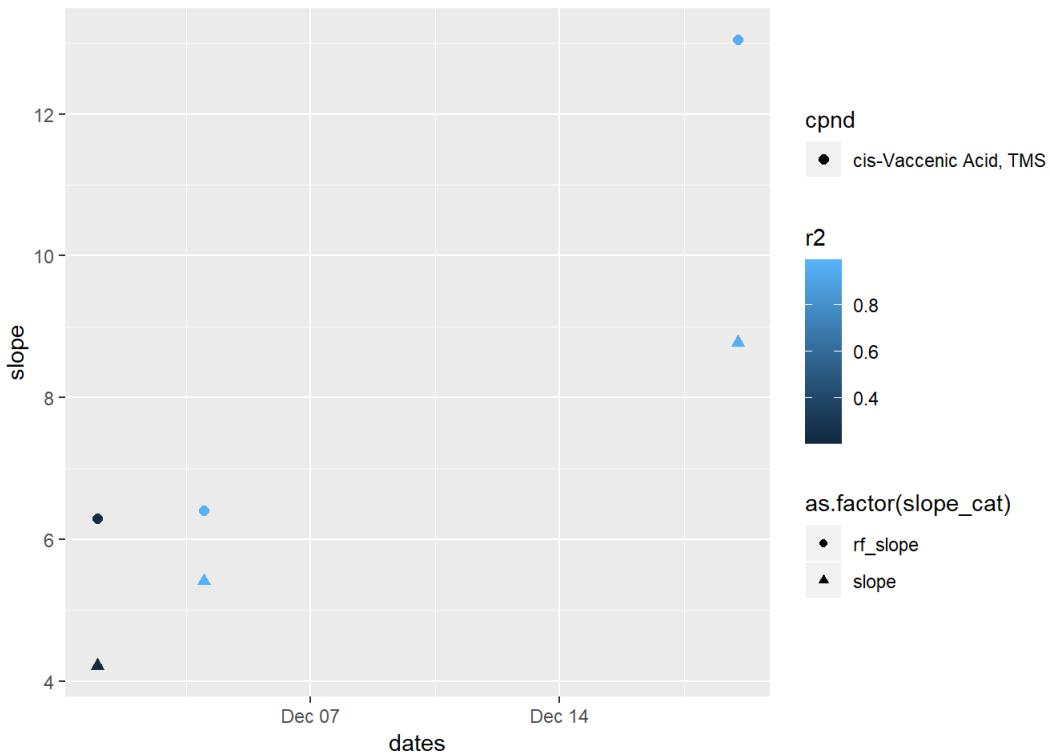
```
## Warning: Using size for a discrete variable is not advised.
```



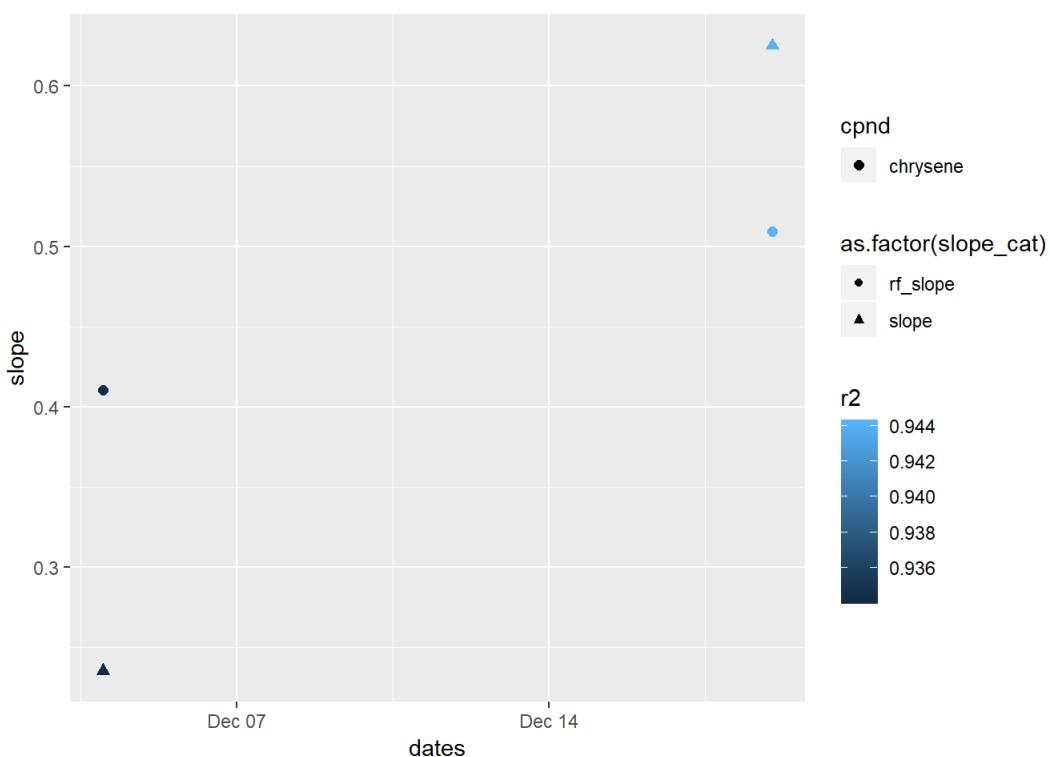
```
## Warning: Using size for a discrete variable is not advised.
```



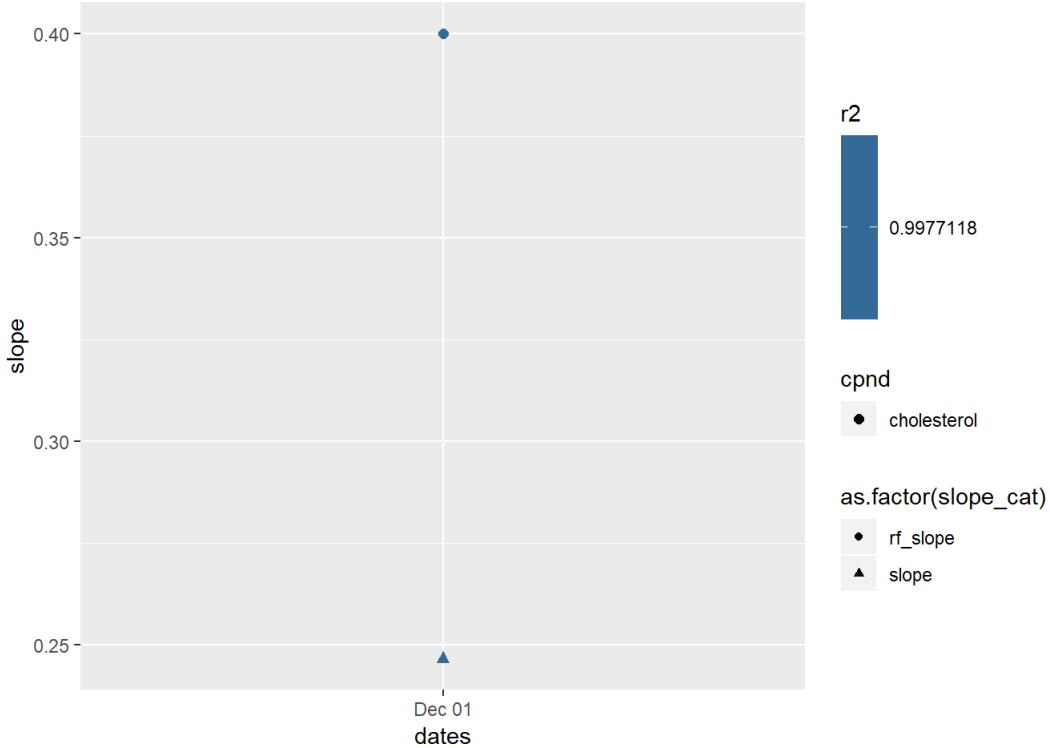
```
## Warning: Using size for a discrete variable is not advised.
```



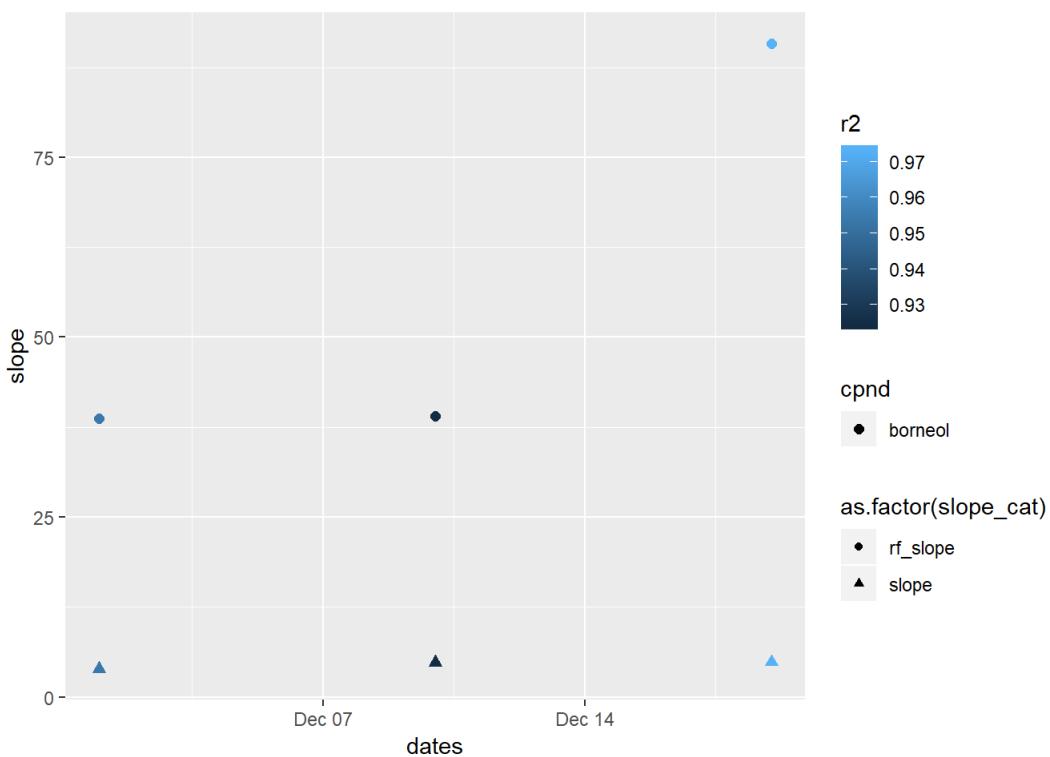
```
## Warning: Using size for a discrete variable is not advised.
```



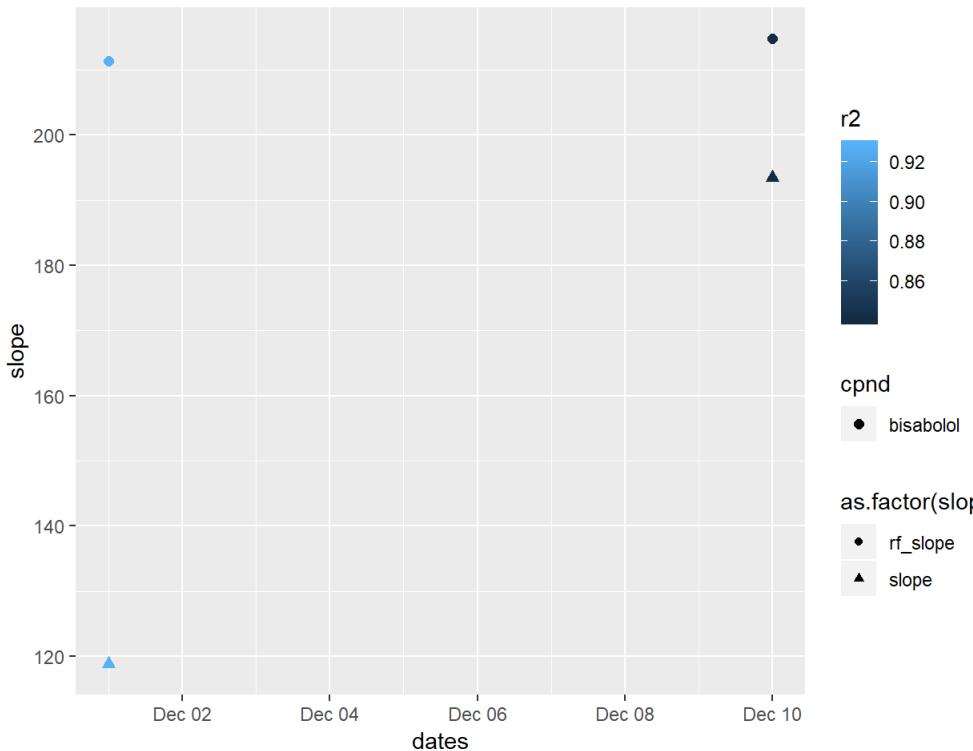
```
## Warning: Using size for a discrete variable is not advised.
```



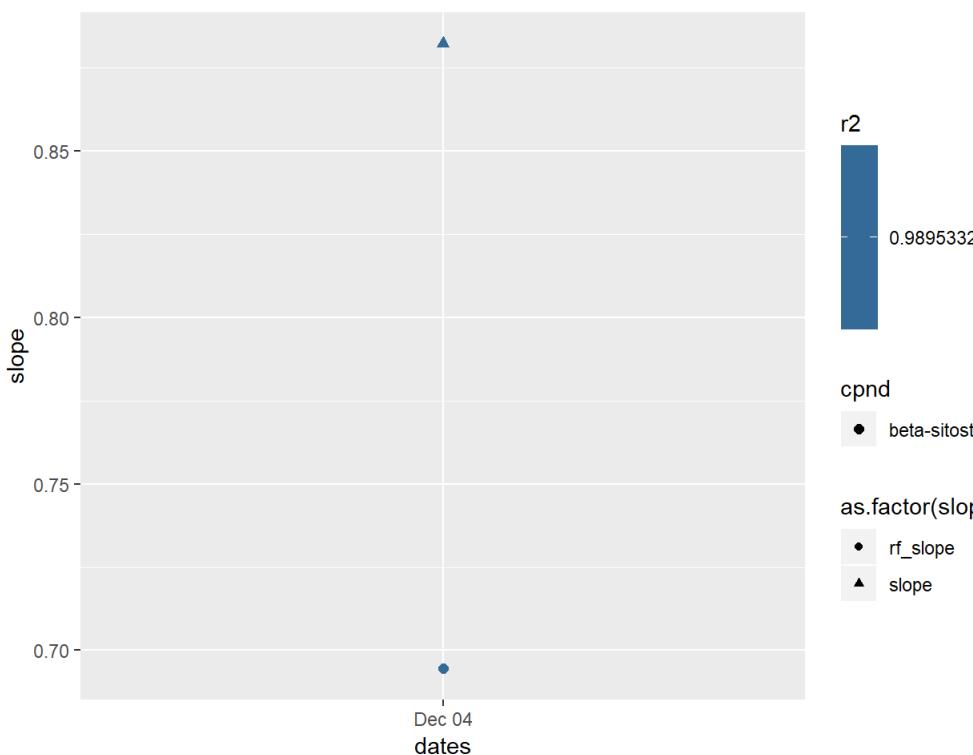
```
## Warning: Using size for a discrete variable is not advised.
```



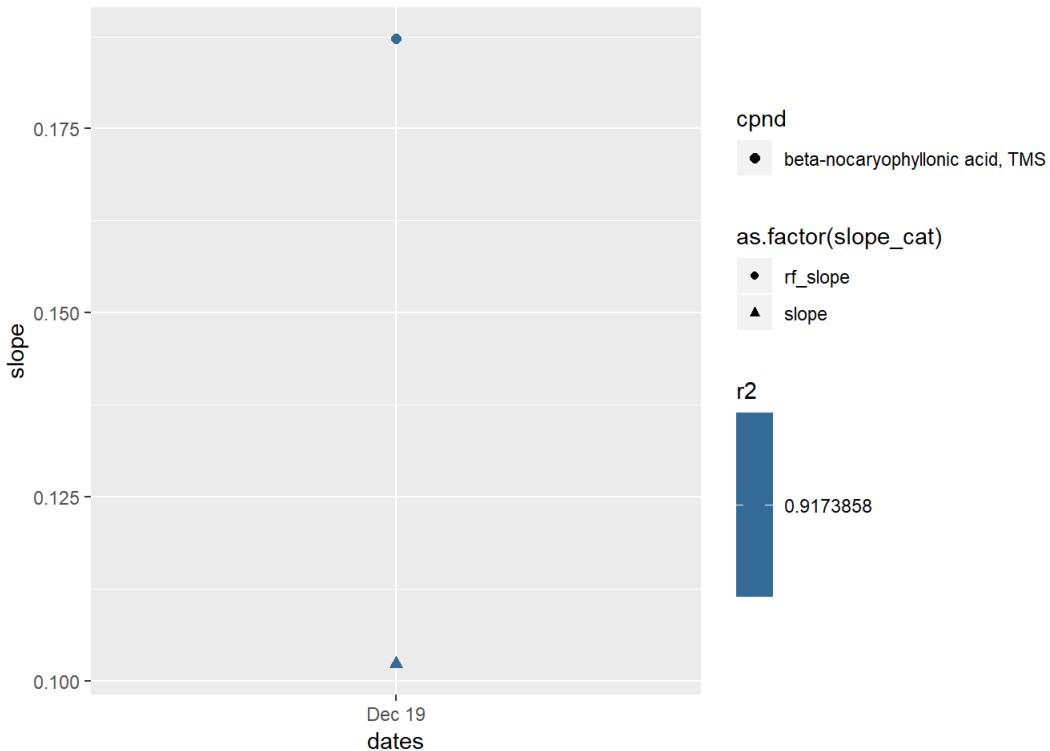
```
## Warning: Using size for a discrete variable is not advised.
```



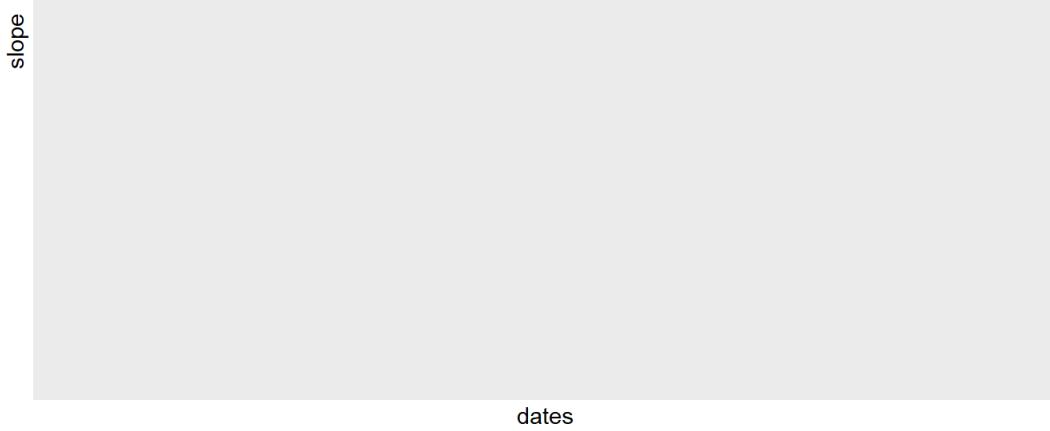
```
## Warning: Using size for a discrete variable is not advised.
```

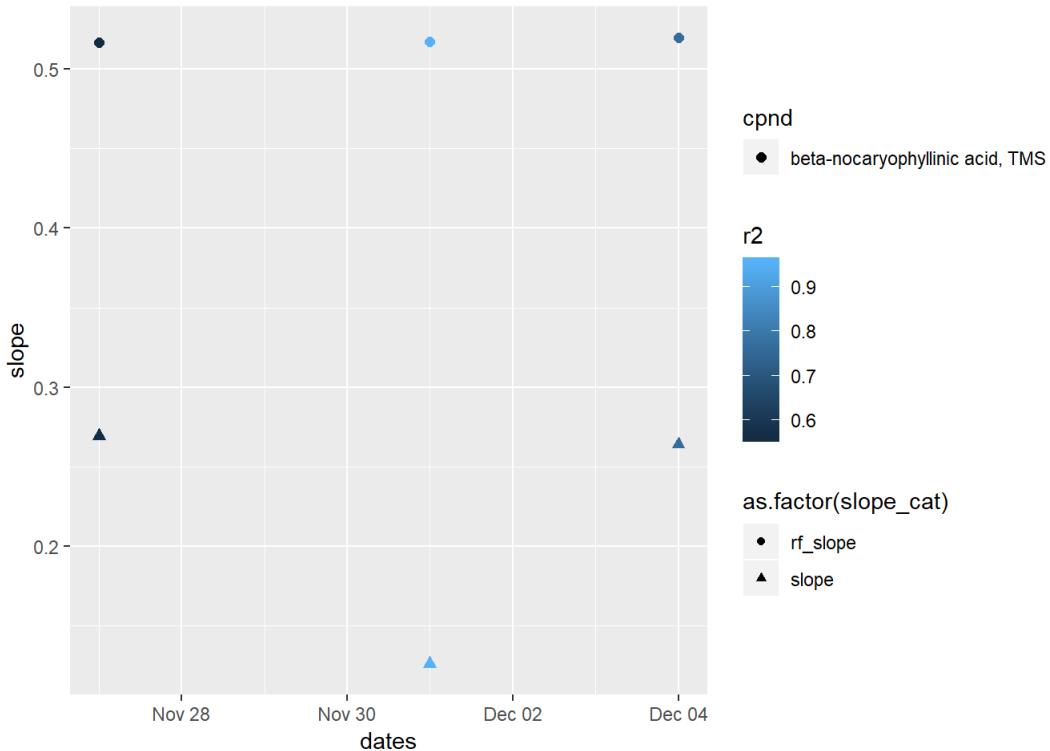


```
## Warning: Using size for a discrete variable is not advised.
```

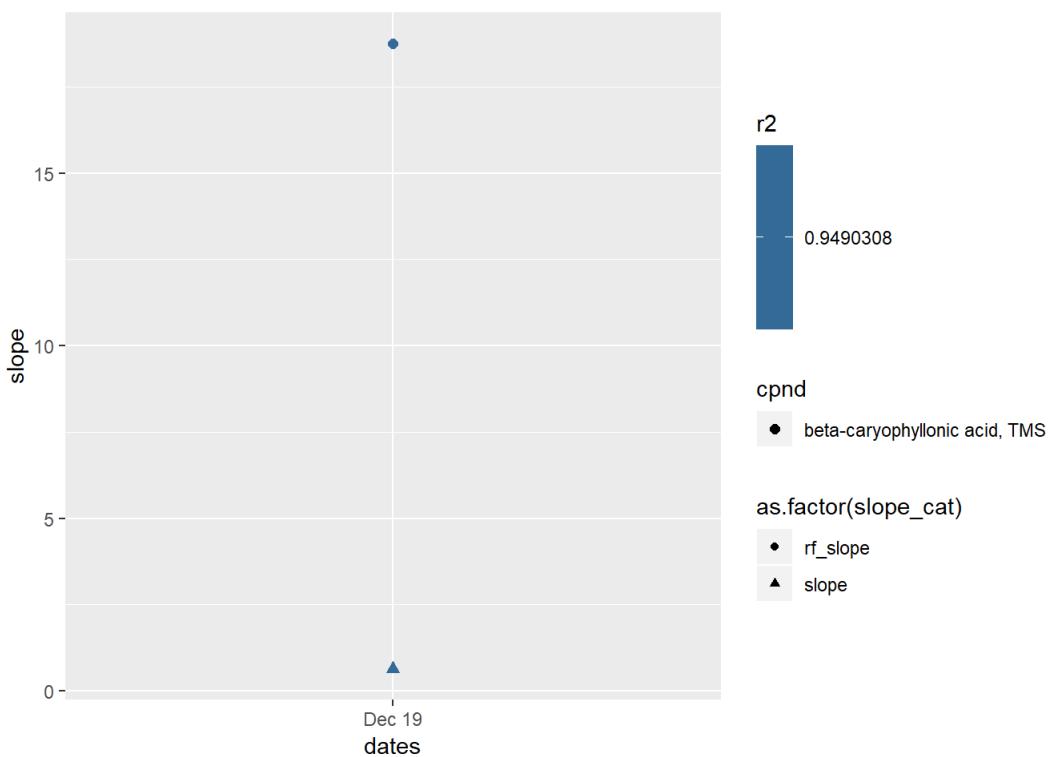


```
## Warning: Using size for a discrete variable is not advised.
```

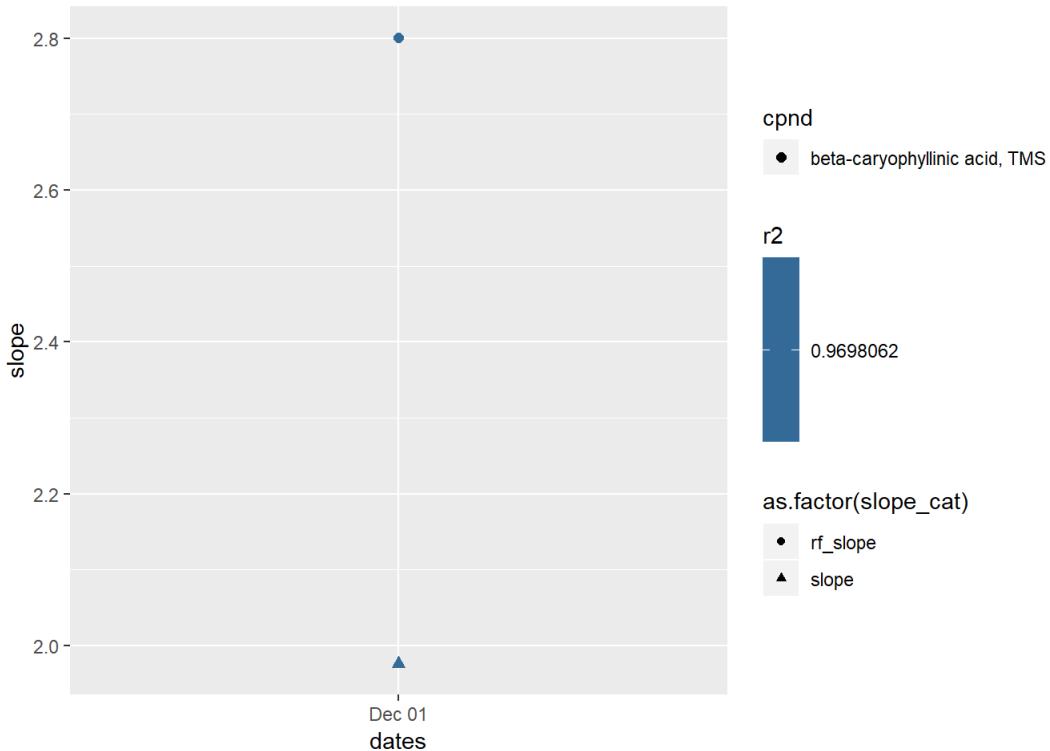




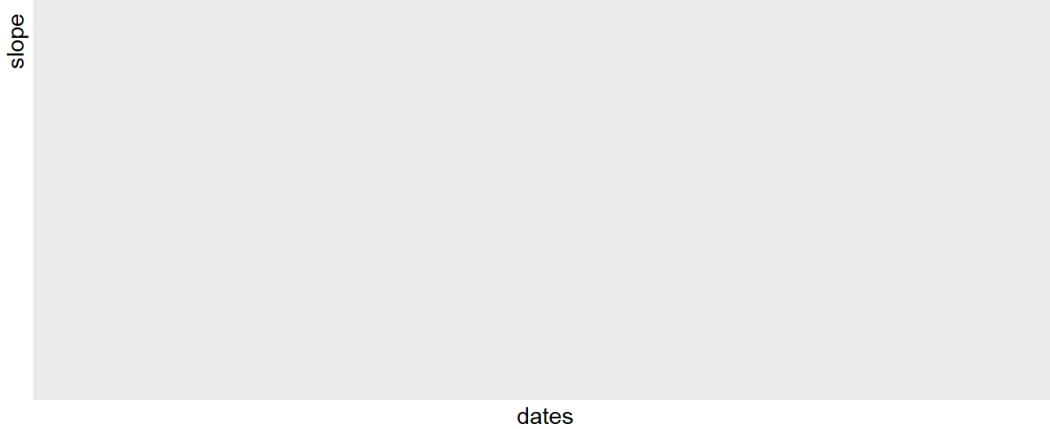
```
## Warning: Using size for a discrete variable is not advised.
```



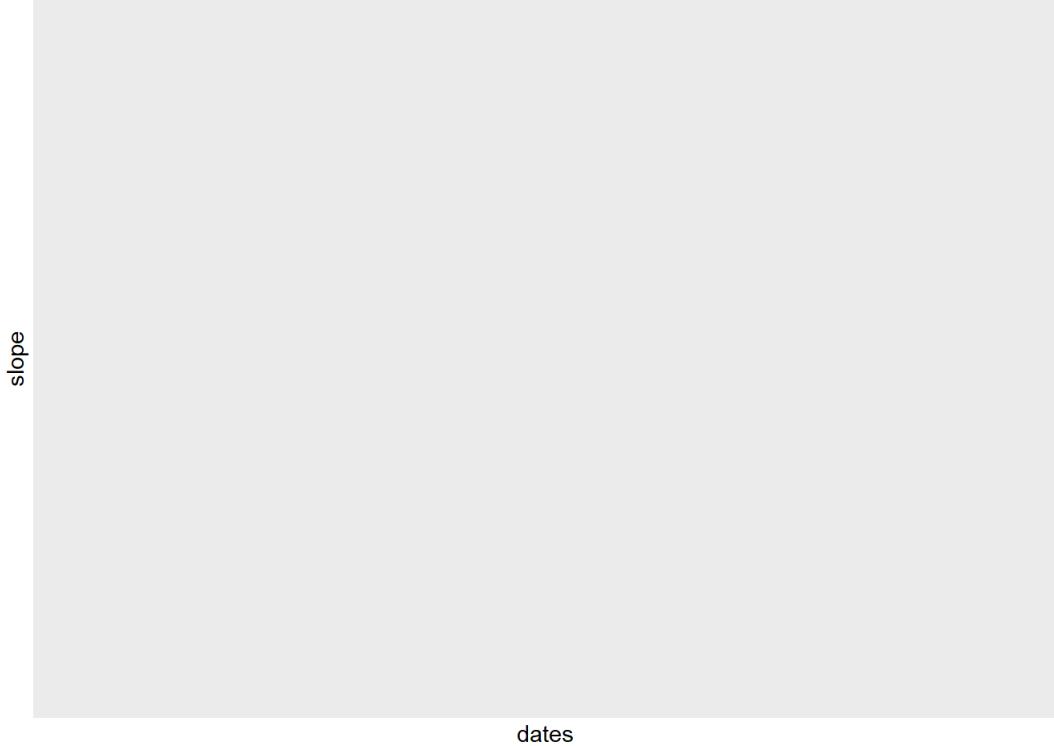
```
## Warning: Using size for a discrete variable is not advised.
```



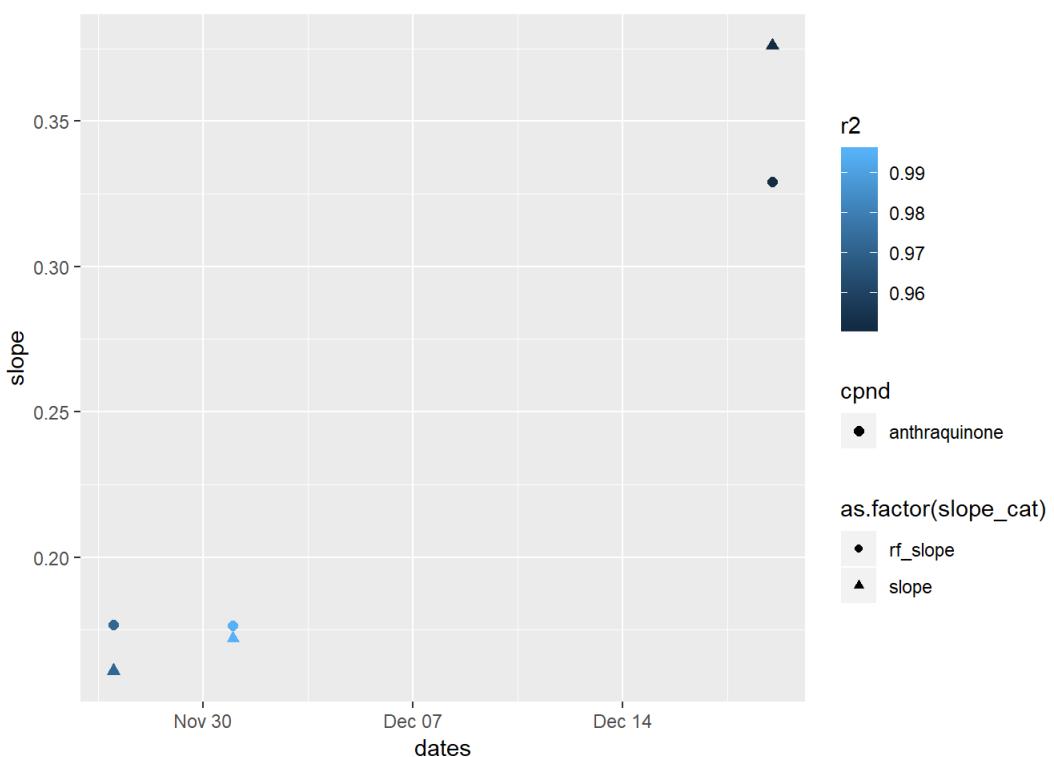
```
## Warning: Using size for a discrete variable is not advised.
```



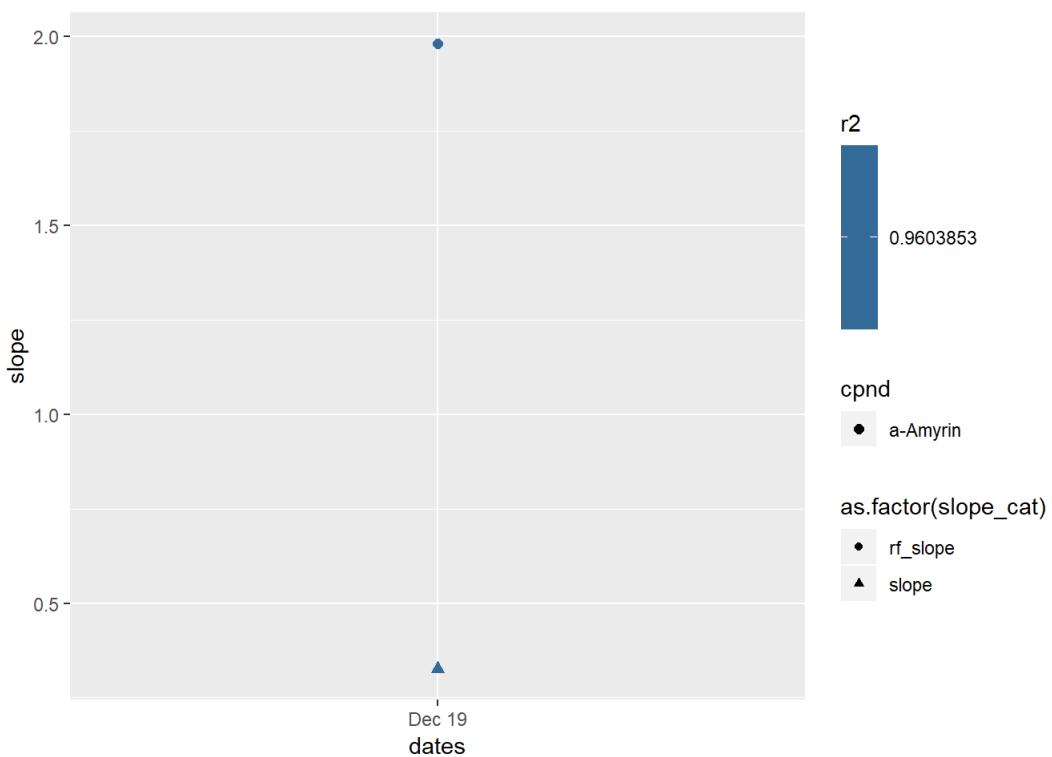
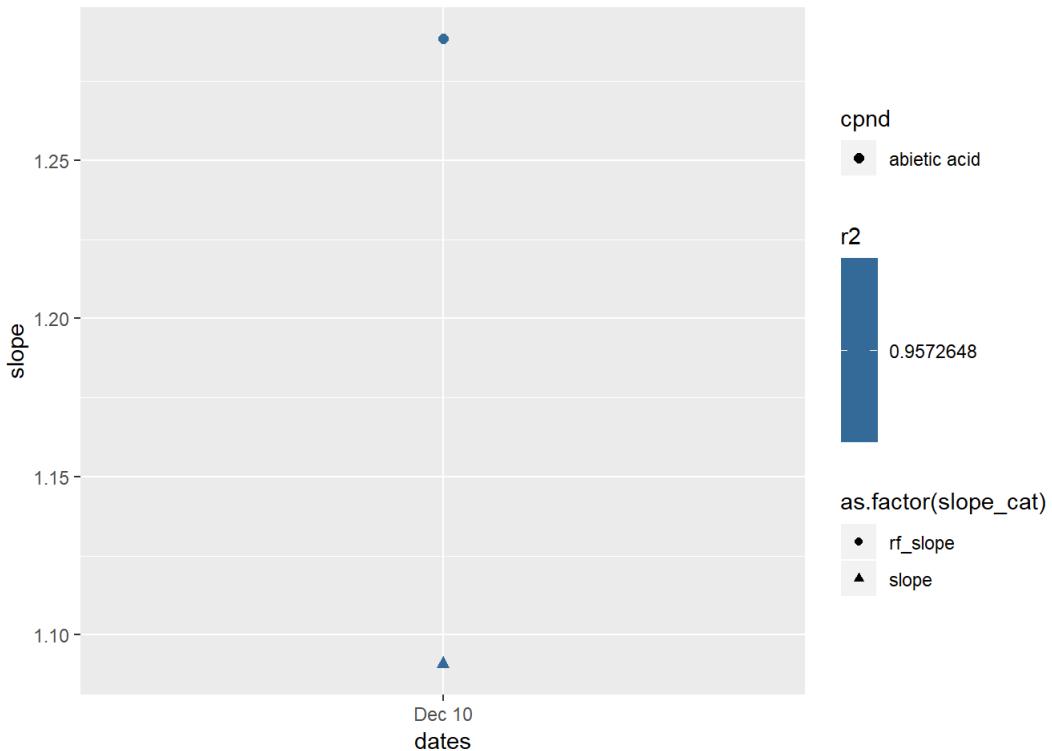
```
## Warning: Using size for a discrete variable is not advised.
```

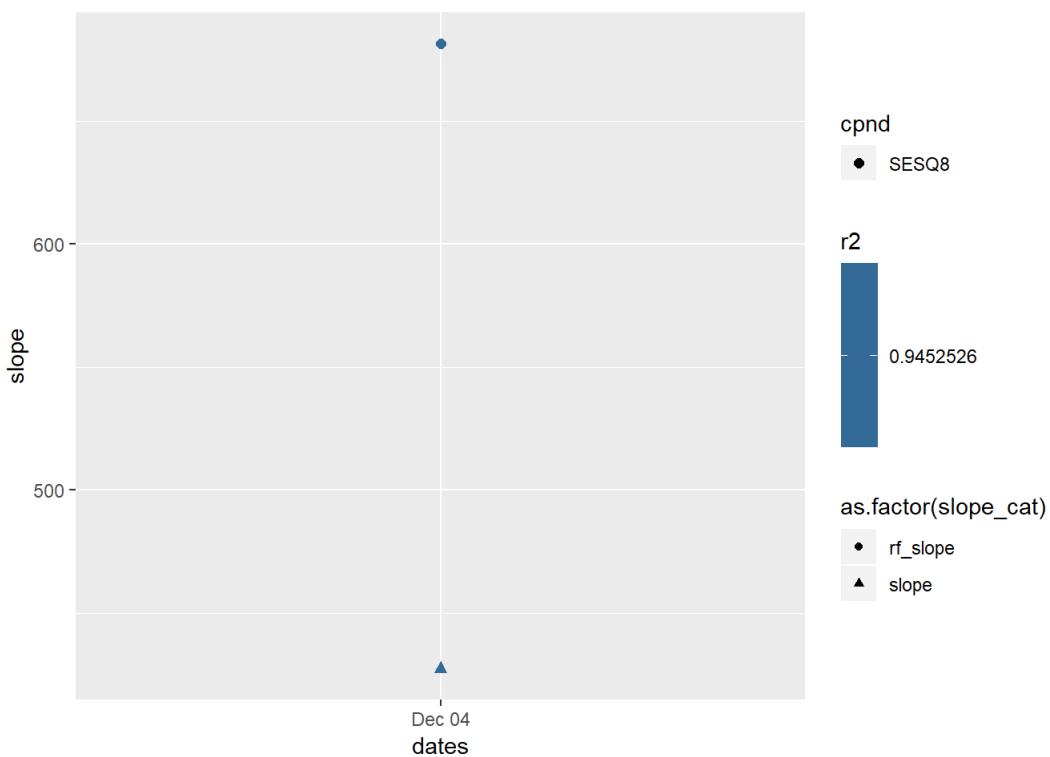
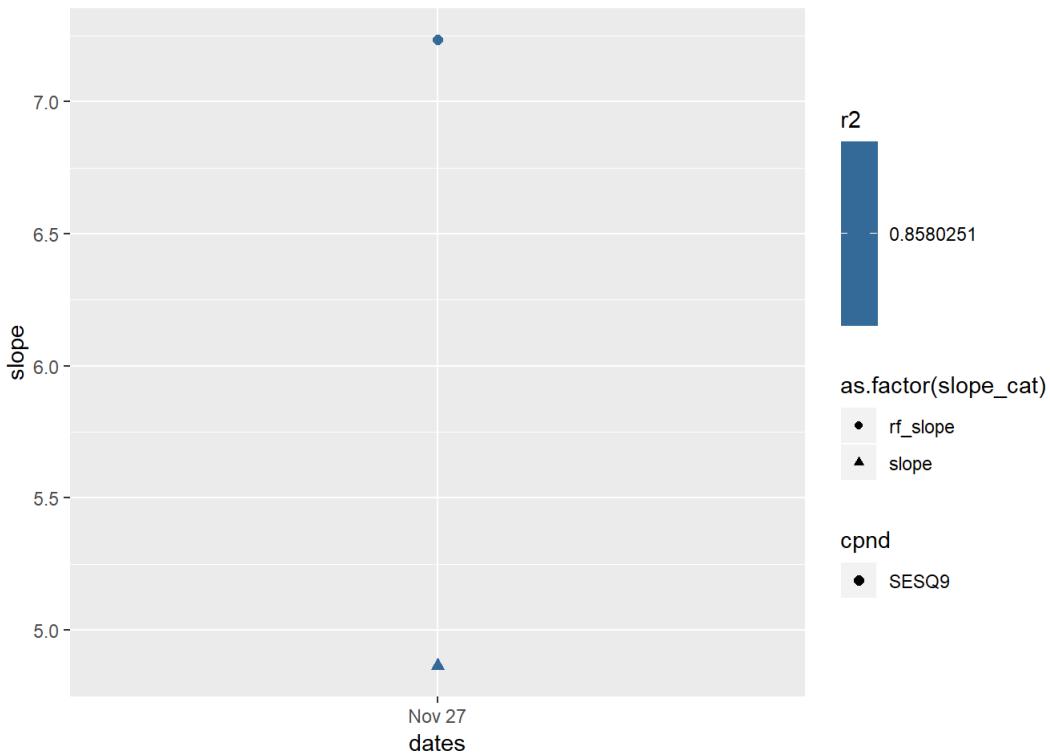


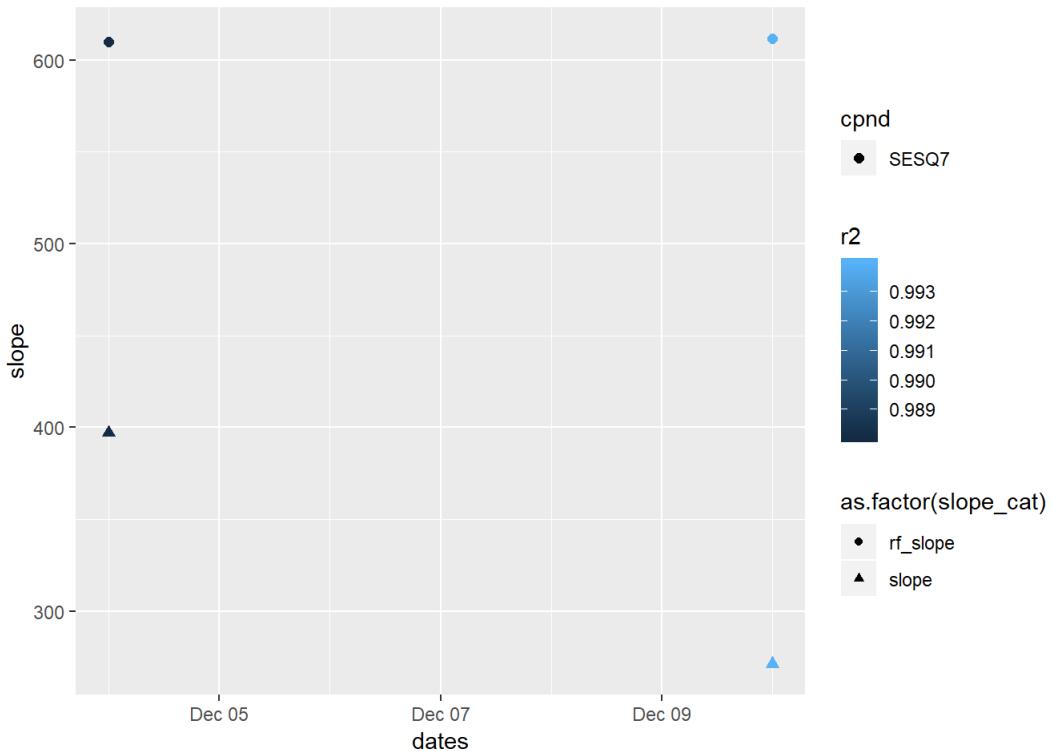
```
## Warning: Using size for a discrete variable is not advised.
```



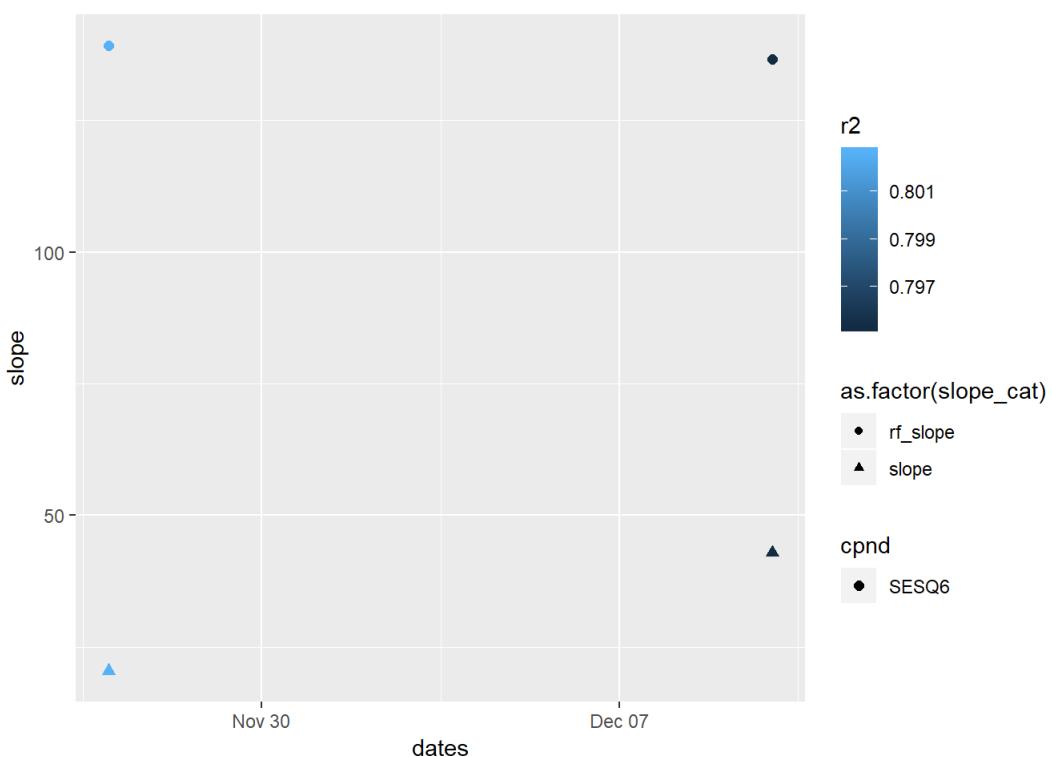
```
## Warning: Using size for a discrete variable is not advised.
```







```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```

slope

dates

```
## Warning: Using size for a discrete variable is not advised.
```

slope

dates

```
## Warning: Using size for a discrete variable is not advised.
```

slope

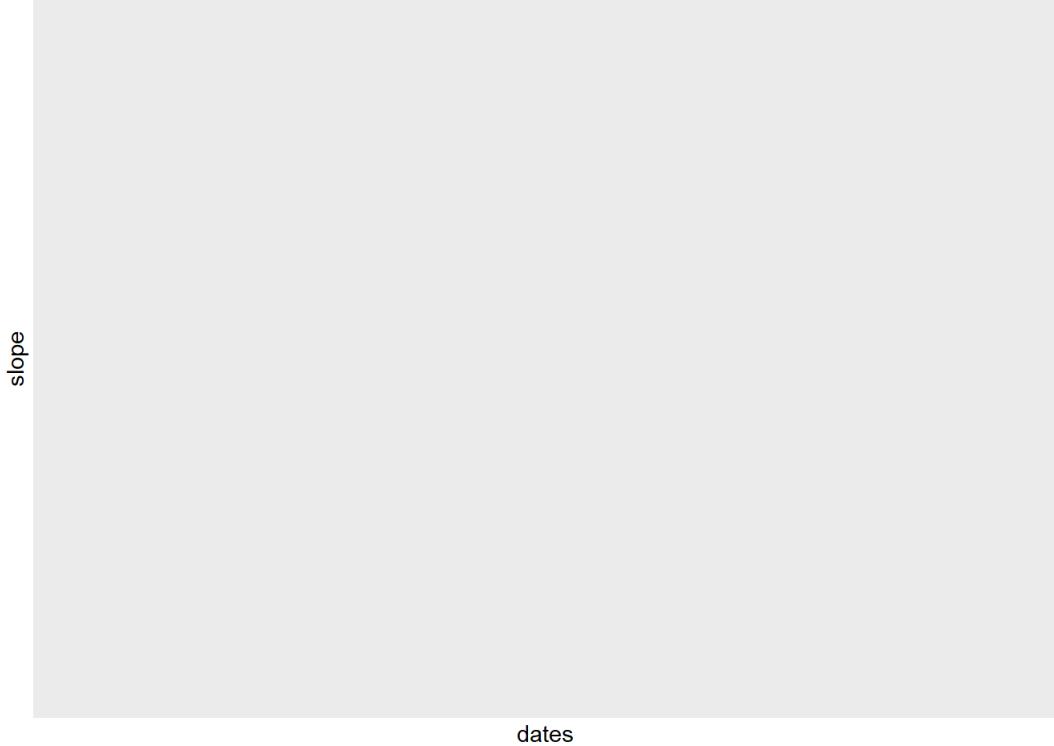
dates

```
## Warning: Using size for a discrete variable is not advised.
```

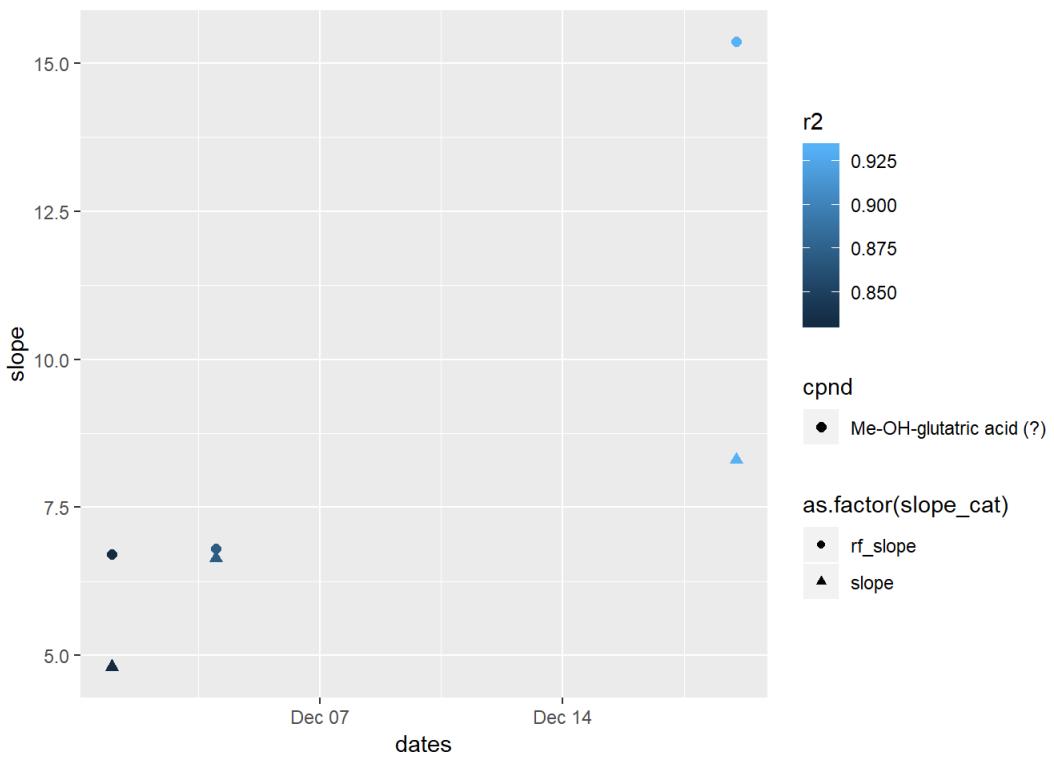
slope

dates

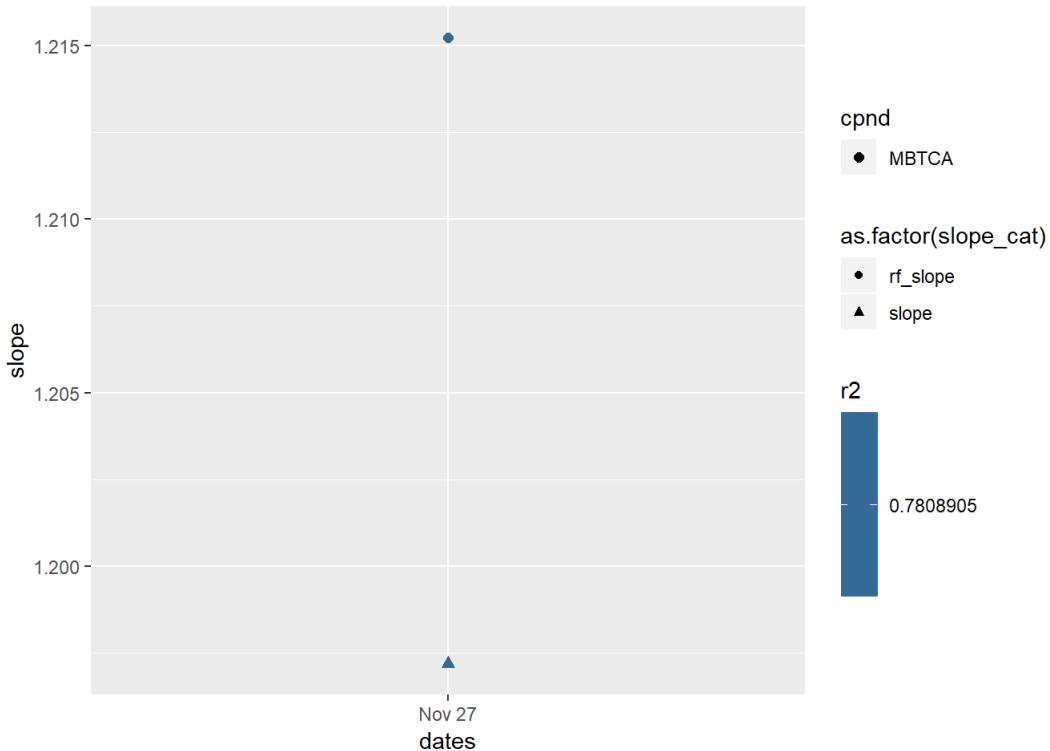
```
## Warning: Using size for a discrete variable is not advised.
```



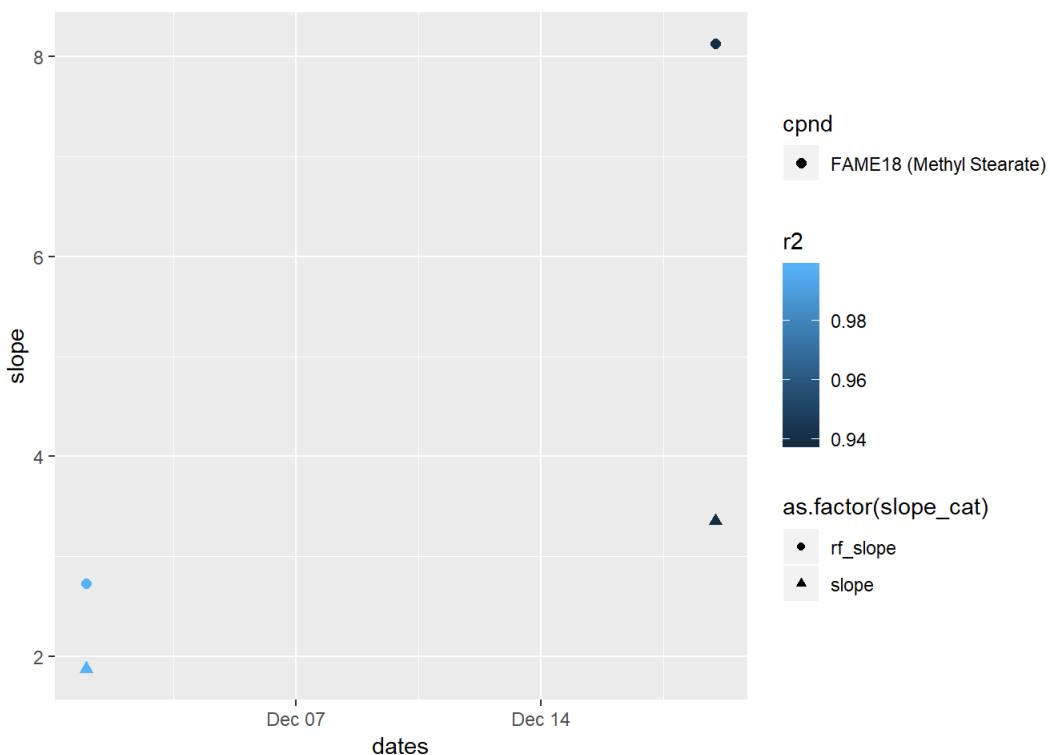
```
## Warning: Using size for a discrete variable is not advised.
```



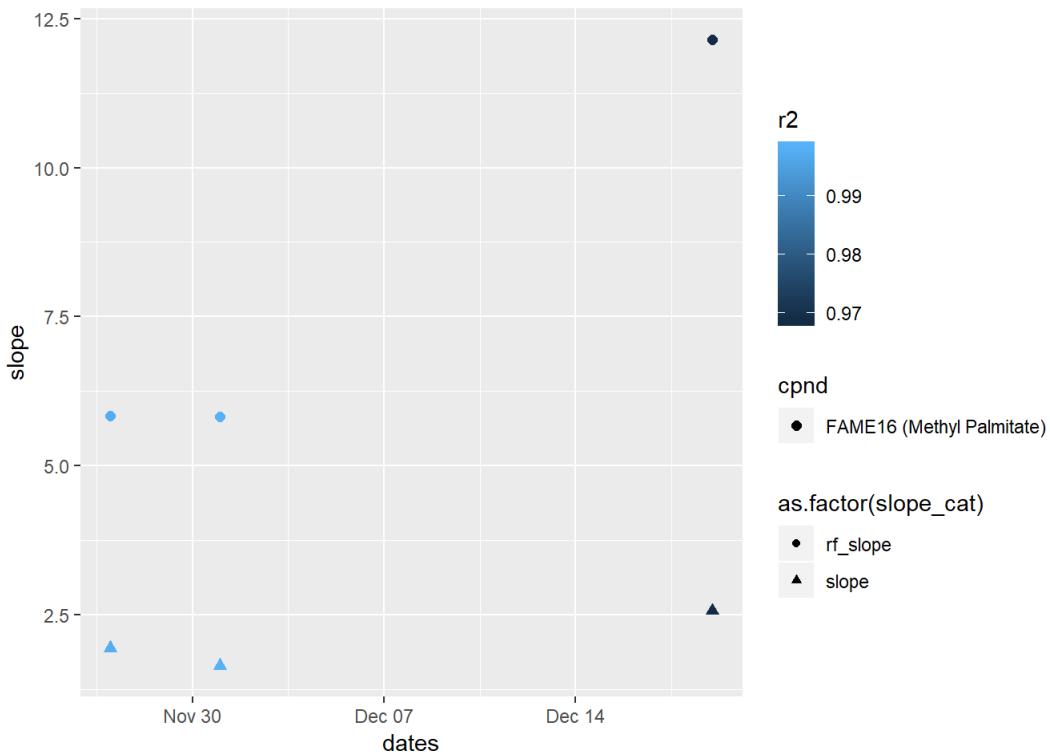
```
## Warning: Using size for a discrete variable is not advised.
```



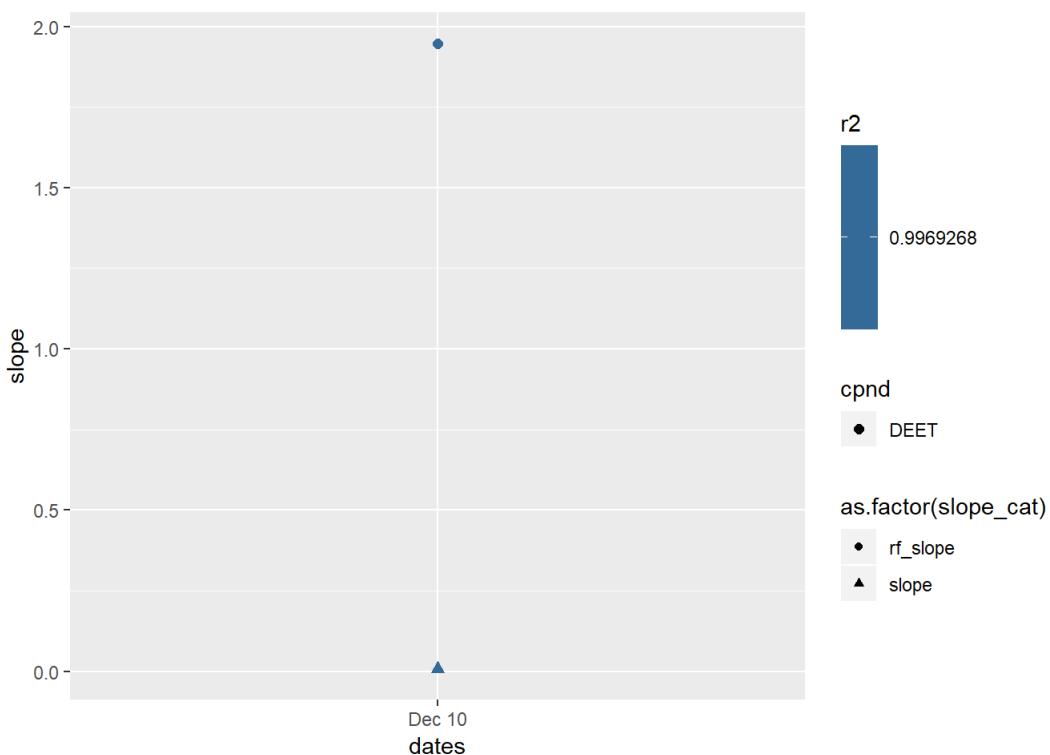
```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```

slope

dates

```
## Warning: Using size for a discrete variable is not advised.
```

slope

dates

```
## Warning: Using size for a discrete variable is not advised.
```

slope

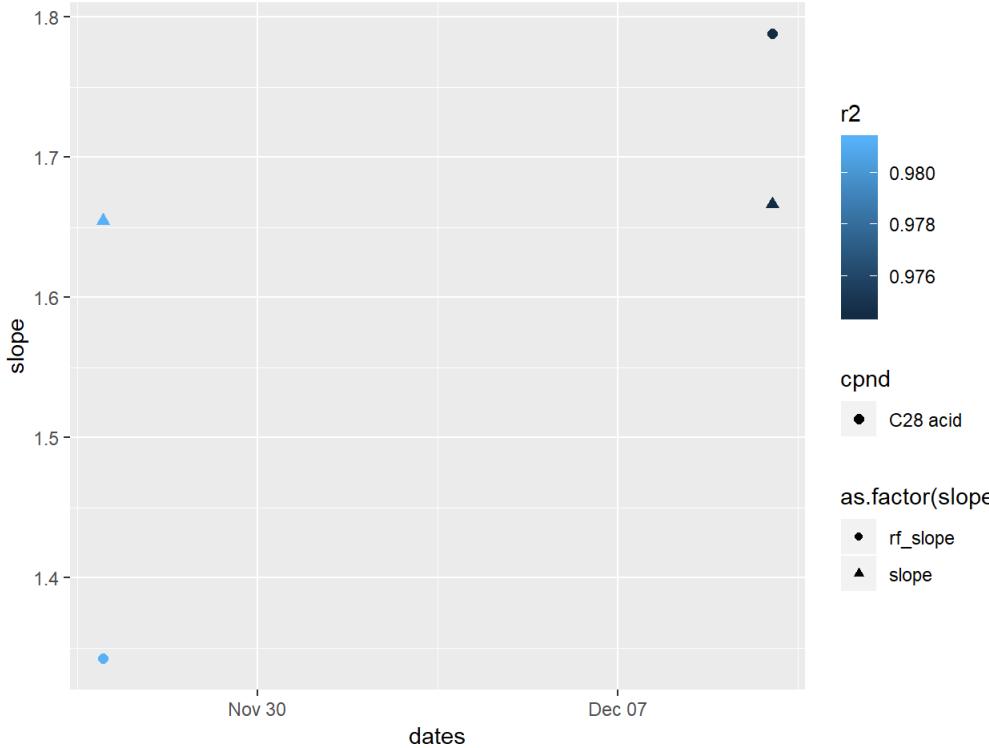
dates

```
## Warning: Using size for a discrete variable is not advised.
```

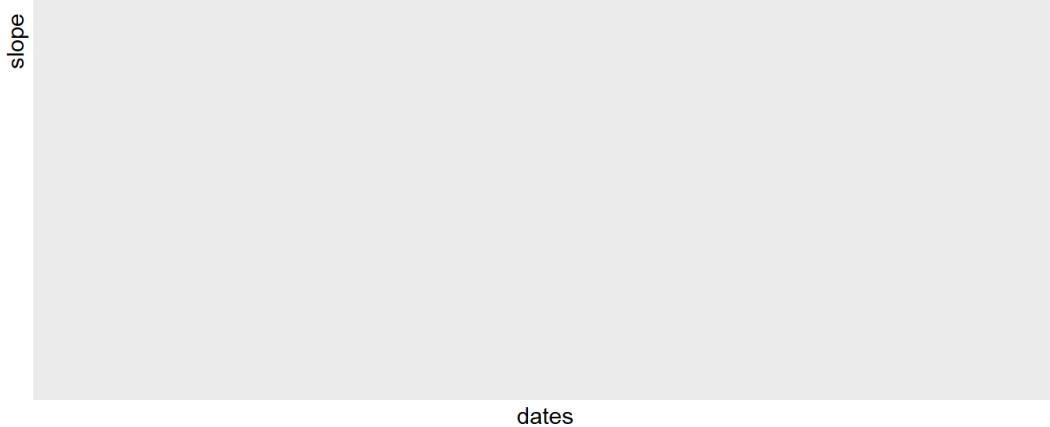
slope

dates

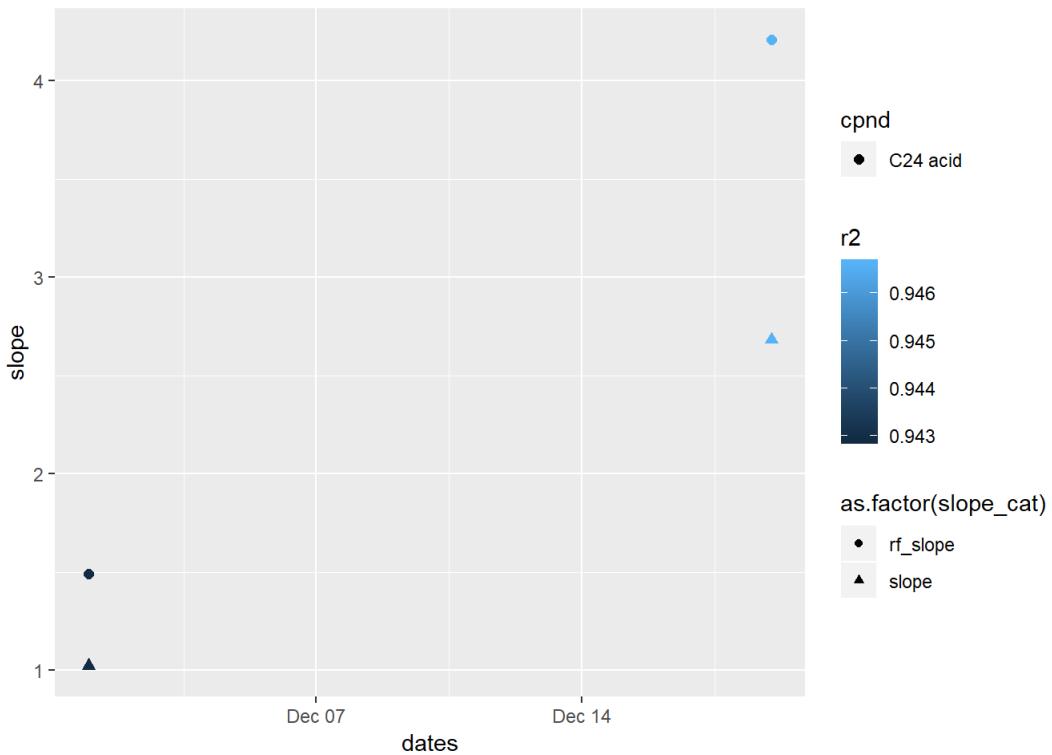
```
## Warning: Using size for a discrete variable is not advised.
```



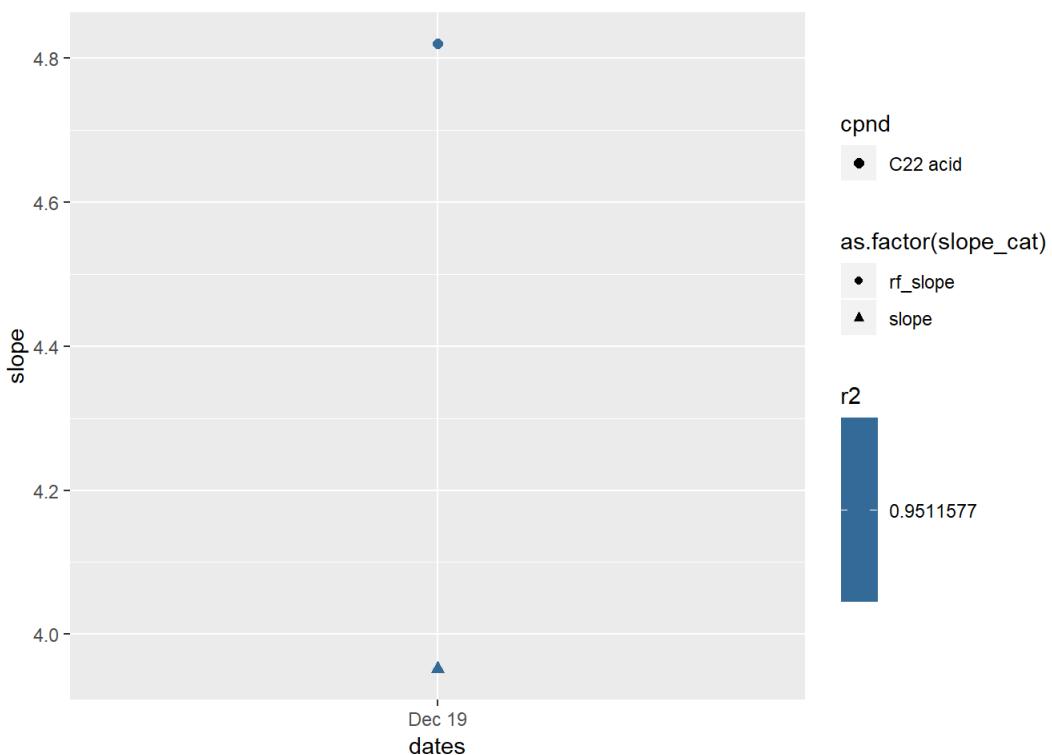
```
## Warning: Using size for a discrete variable is not advised.
```



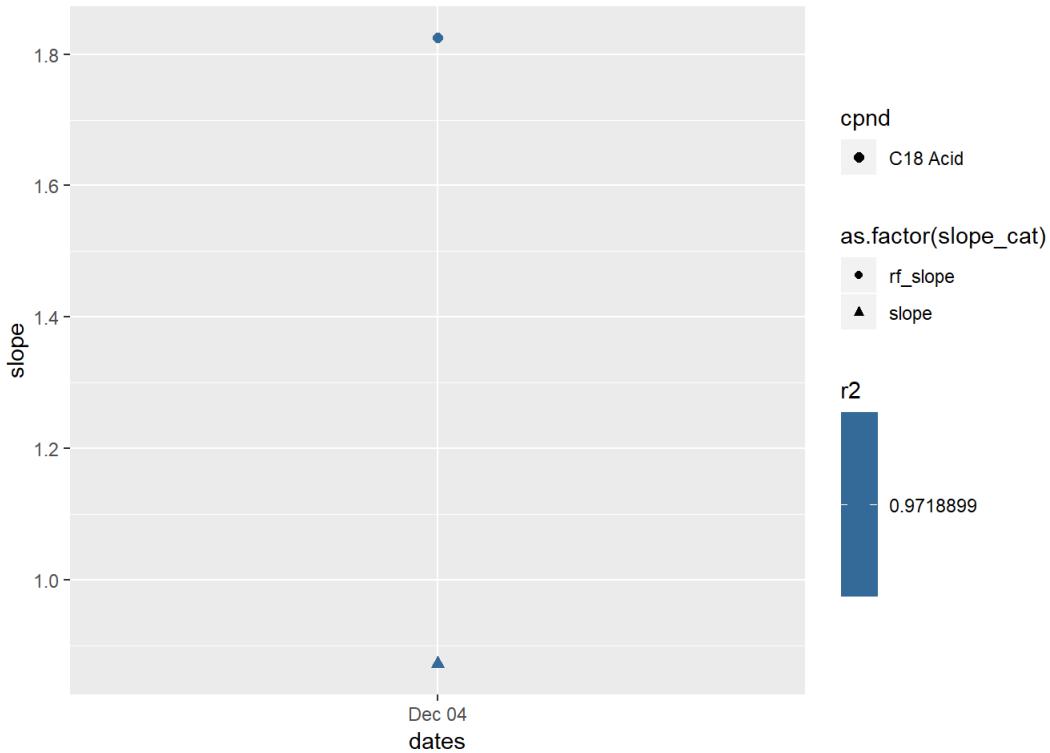
```
## Warning: Using size for a discrete variable is not advised.
```



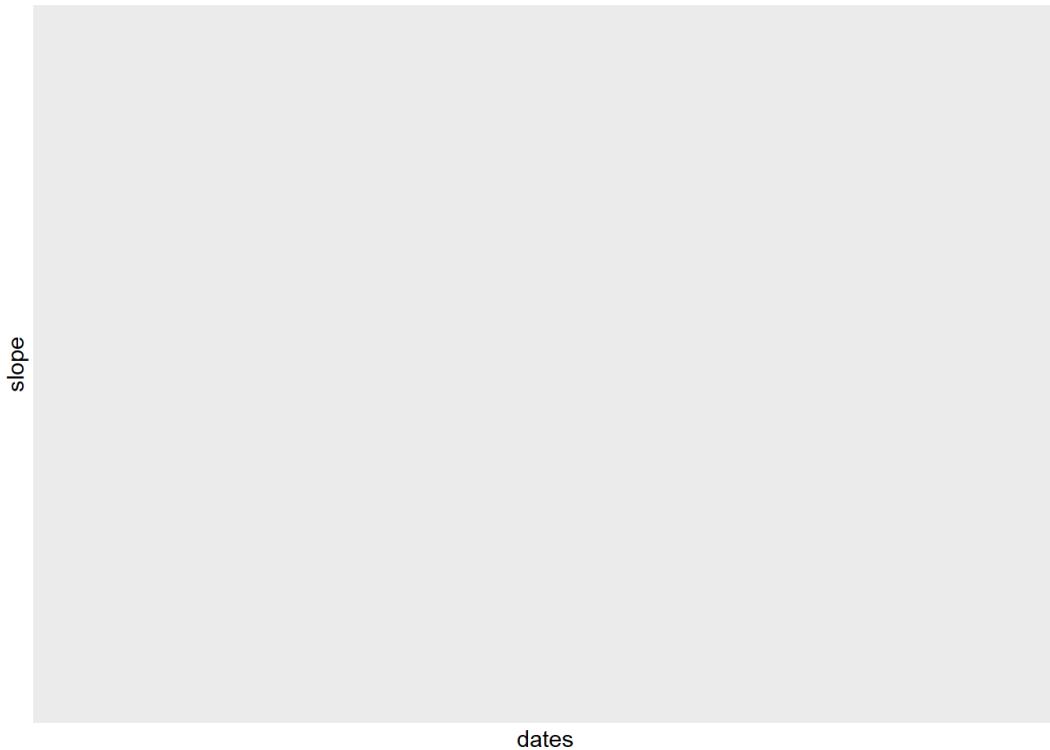
```
## Warning: Using size for a discrete variable is not advised.
```



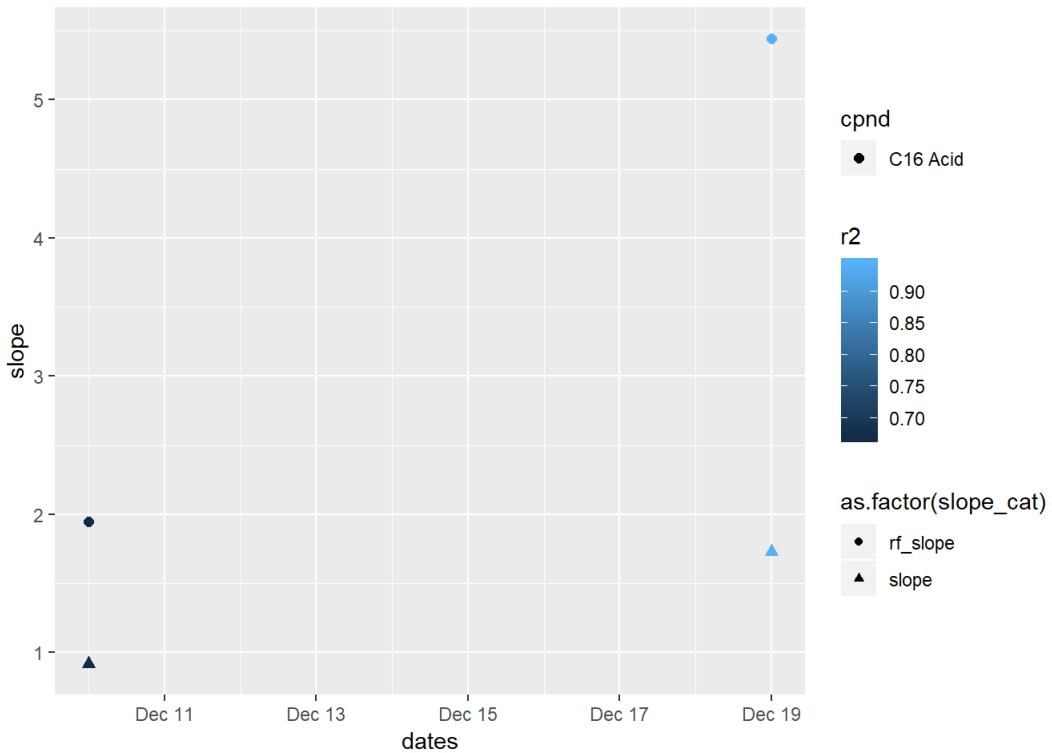
```
## Warning: Using size for a discrete variable is not advised.
```



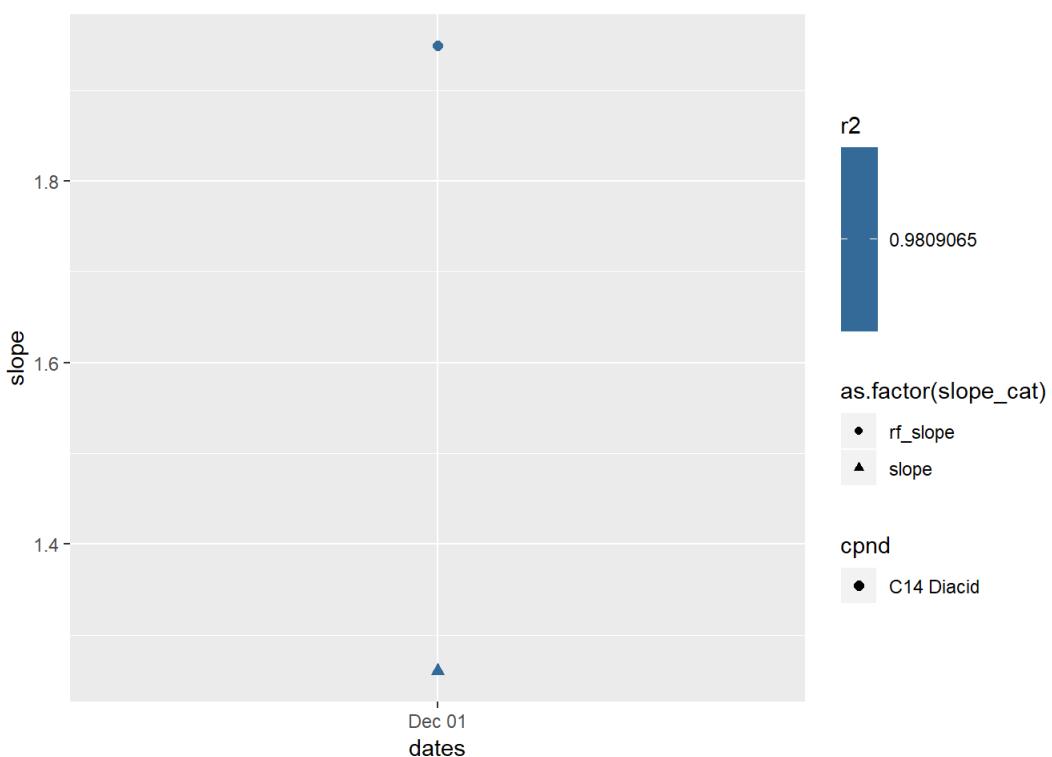
```
## Warning: Using size for a discrete variable is not advised.
```



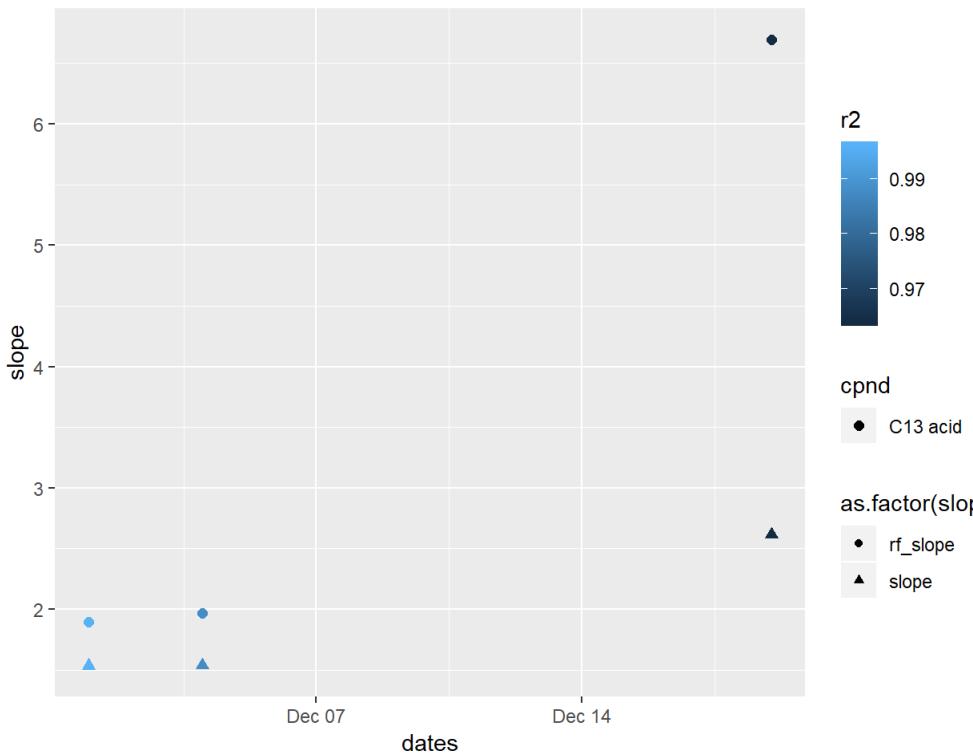
```
## Warning: Using size for a discrete variable is not advised.
```



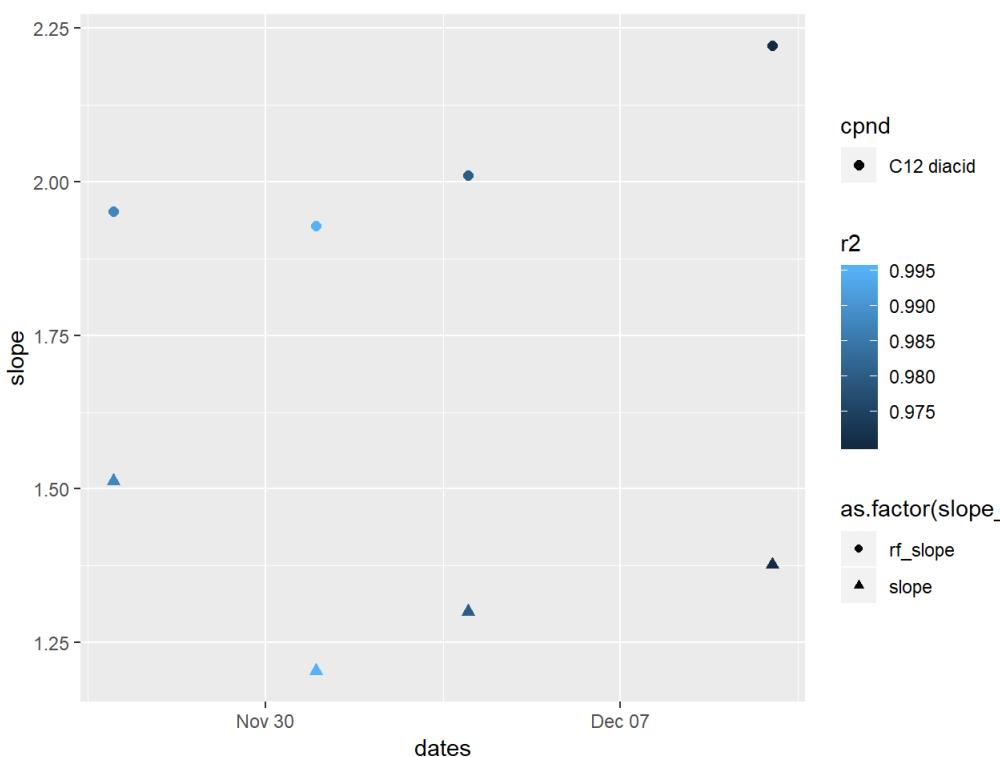
```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```

slope

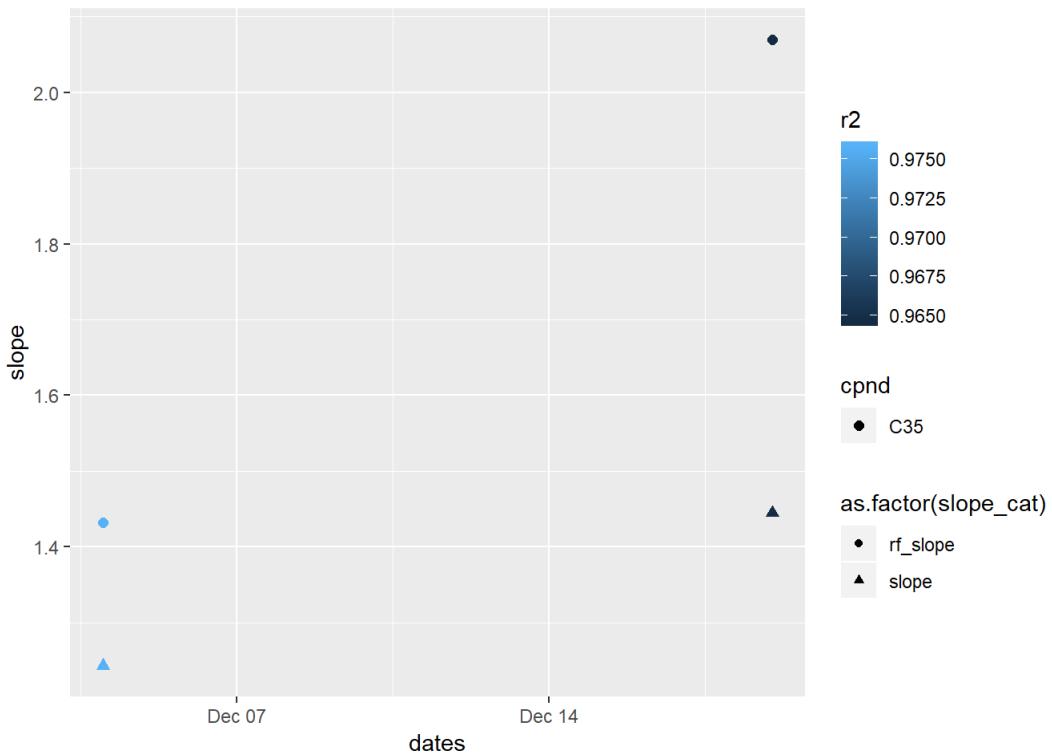
dates

```
## Warning: Using size for a discrete variable is not advised.
```

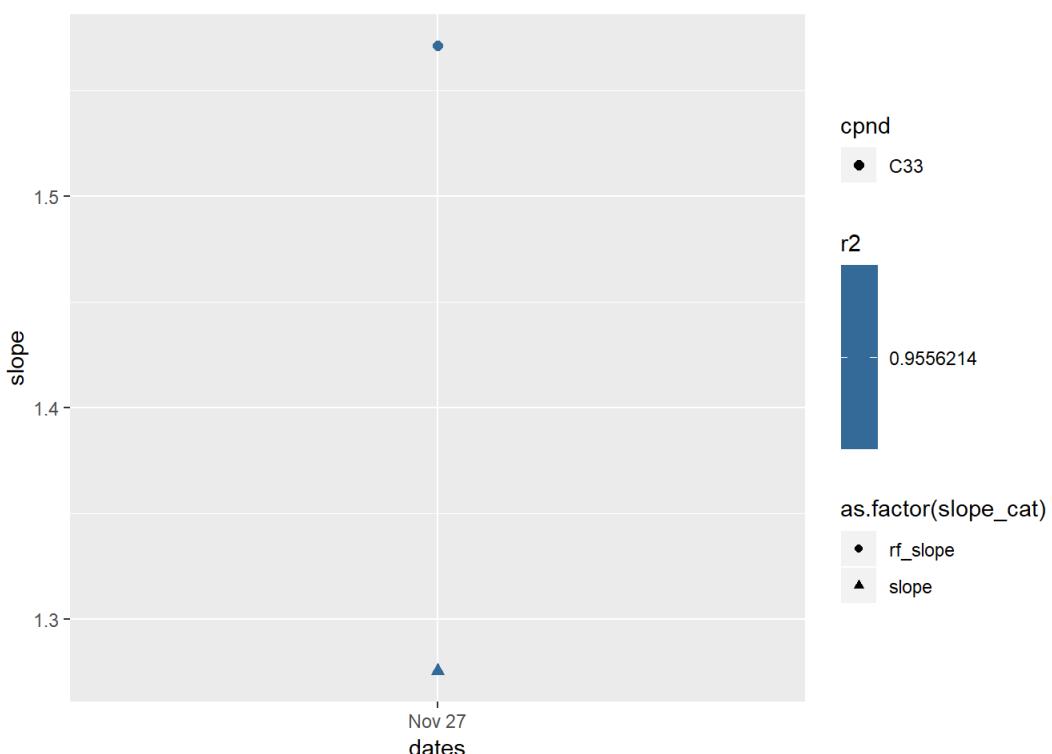
slope

dates

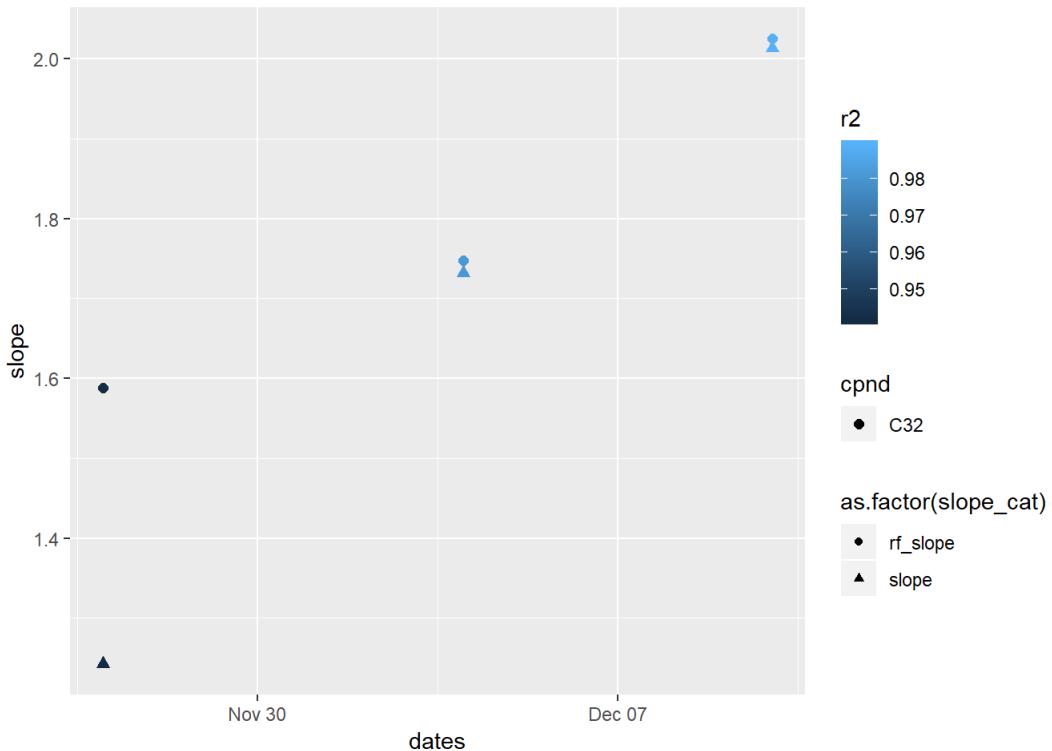
```
## Warning: Using size for a discrete variable is not advised.
```



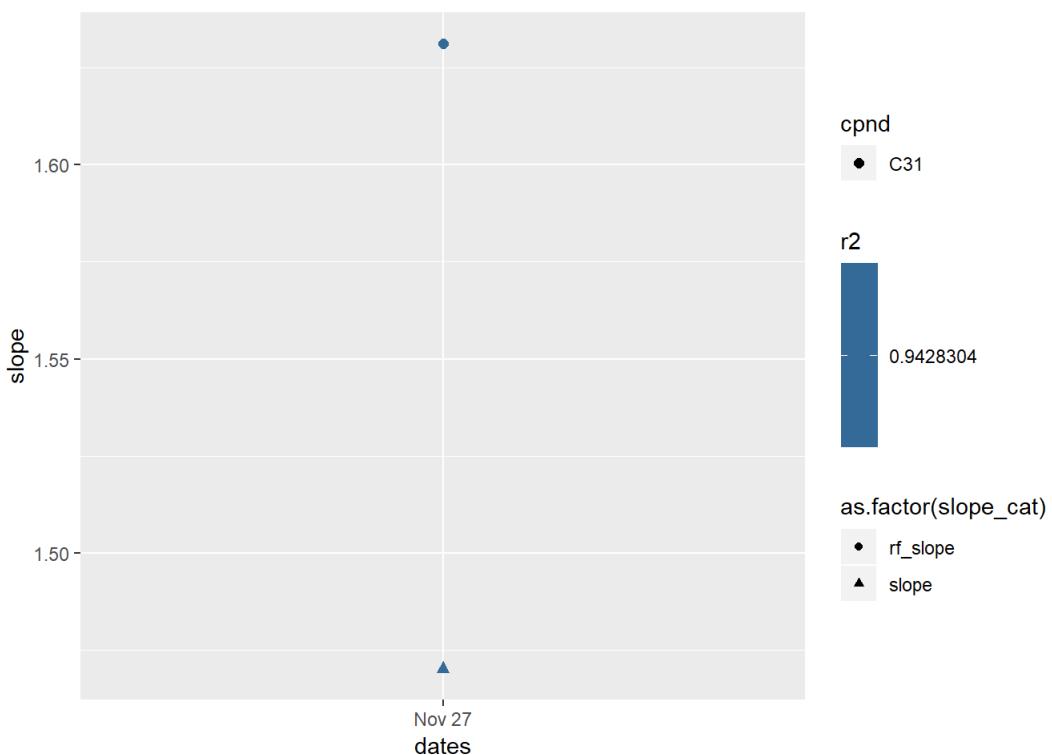
```
## Warning: Using size for a discrete variable is not advised.
```



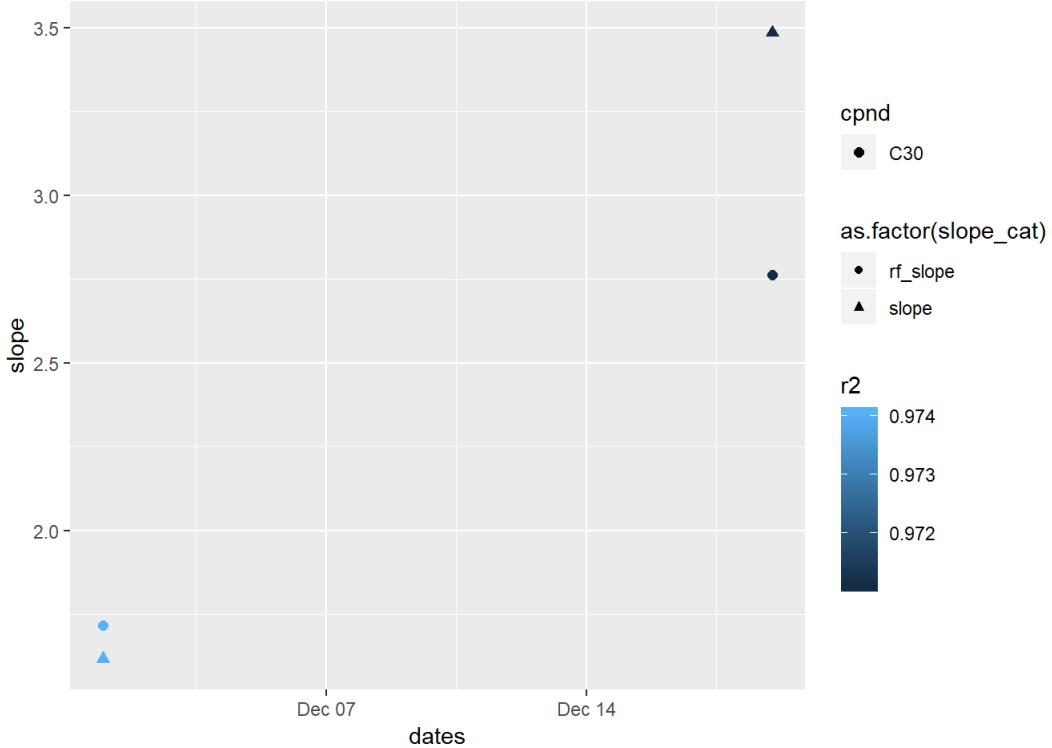
```
## Warning: Using size for a discrete variable is not advised.
```



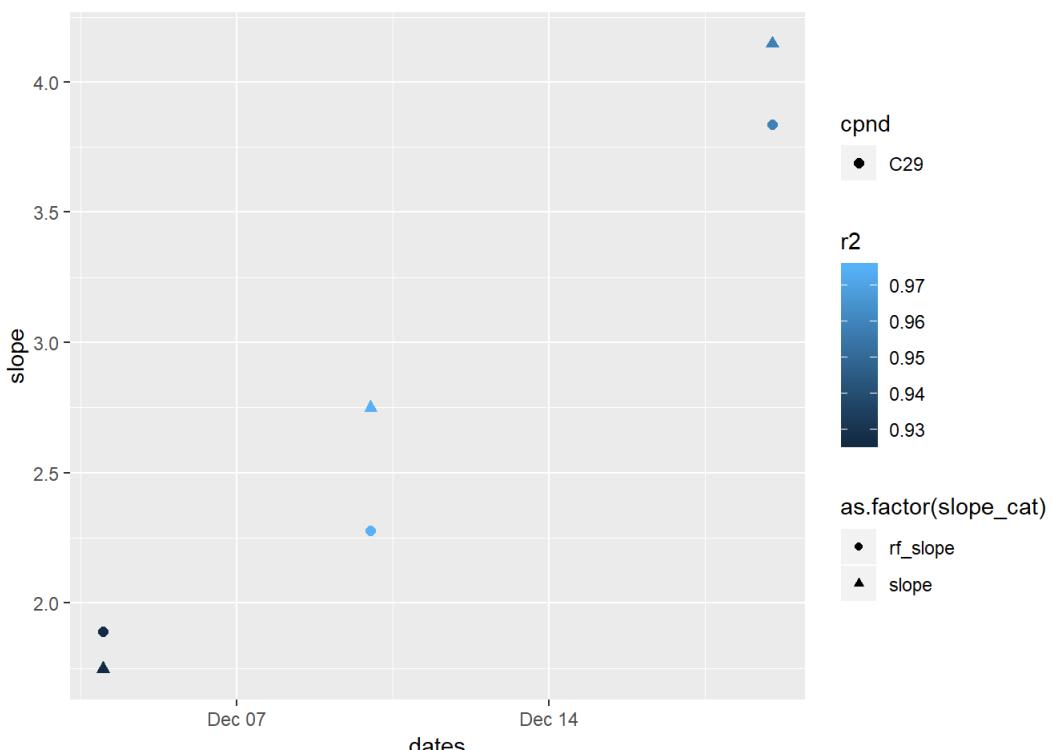
```
## Warning: Using size for a discrete variable is not advised.
```



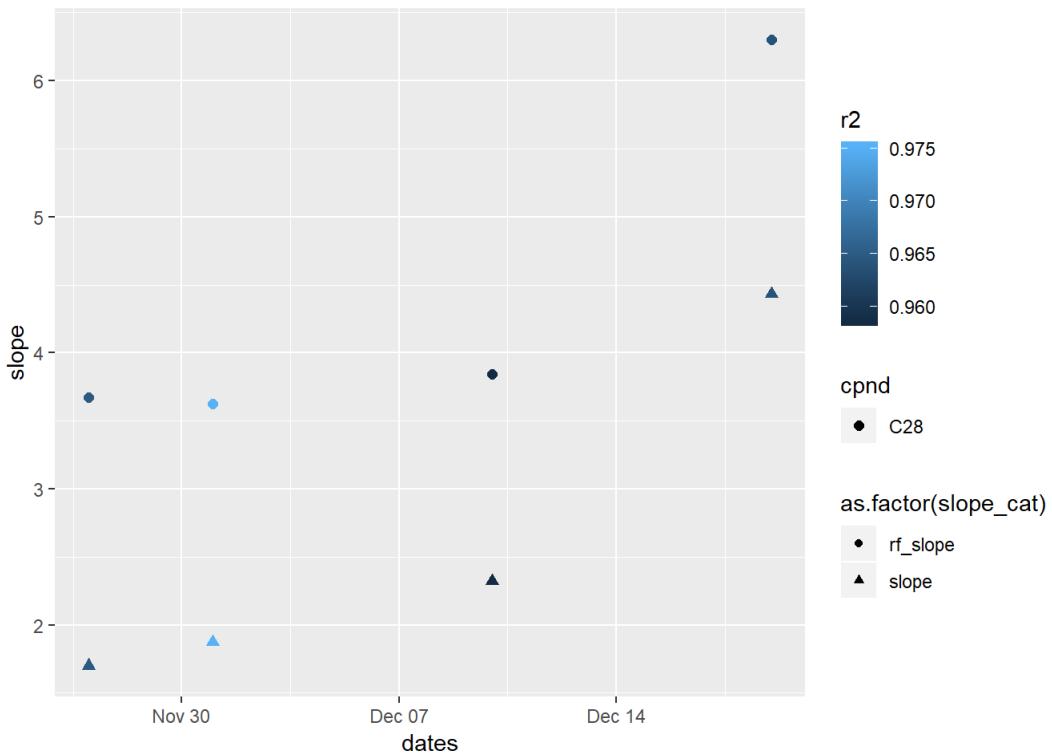
```
## Warning: Using size for a discrete variable is not advised.
```



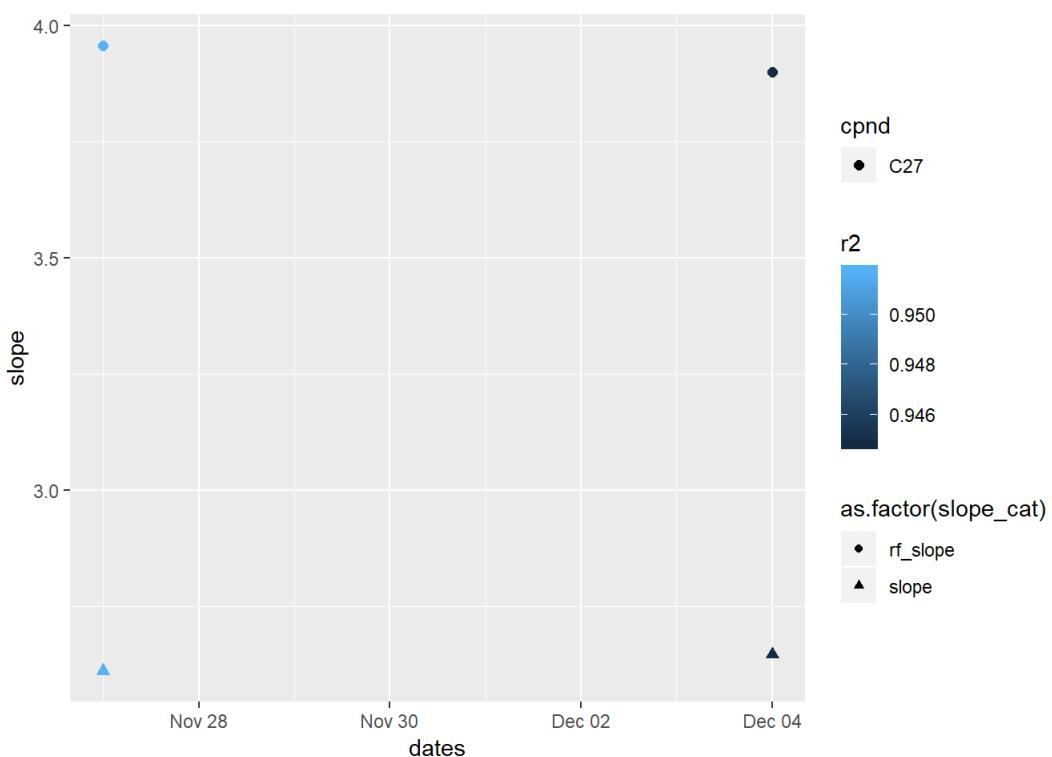
```
## Warning: Using size for a discrete variable is not advised.
```



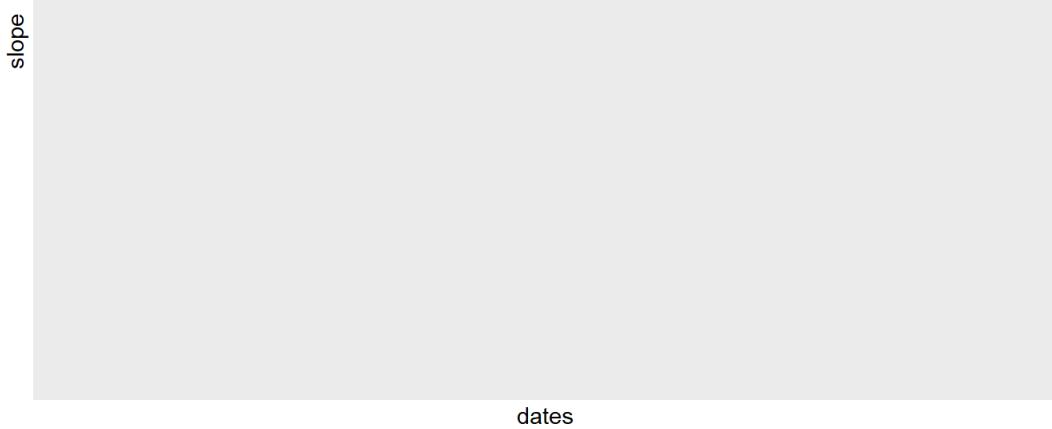
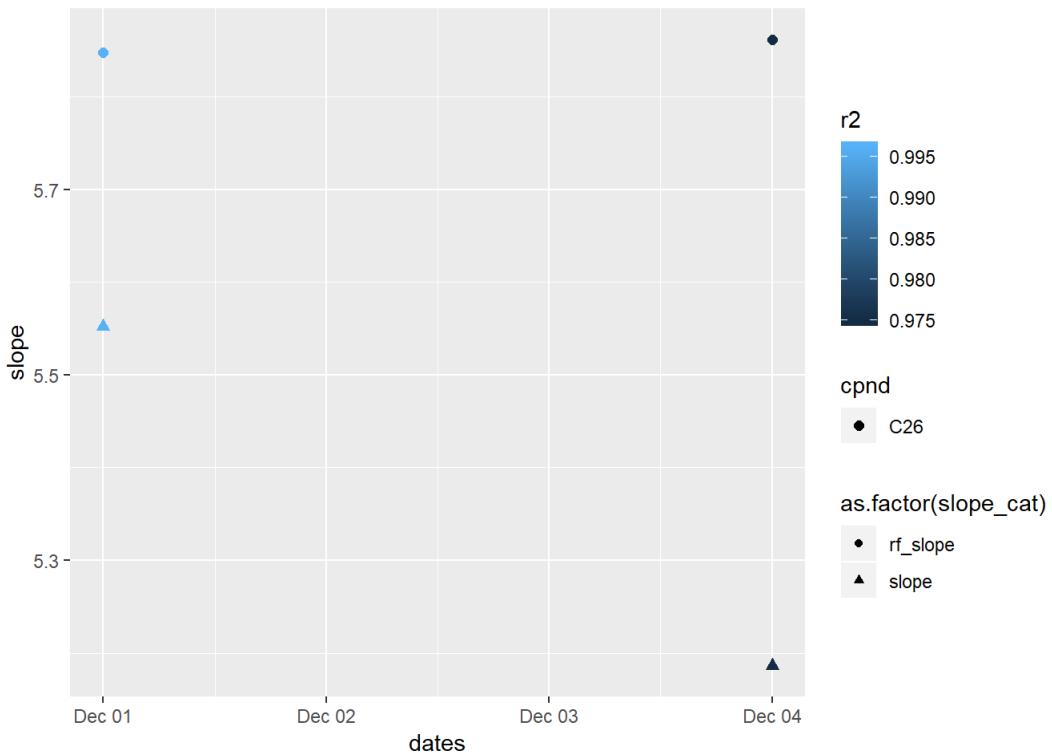
```
## Warning: Using size for a discrete variable is not advised.
```

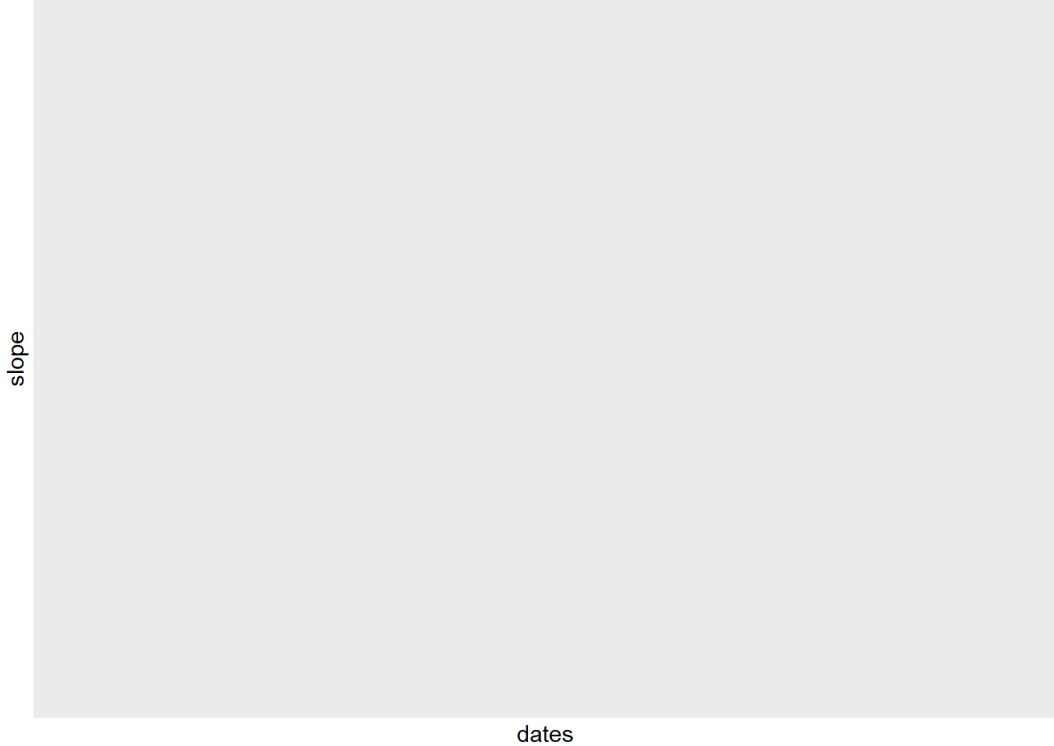


```
## Warning: Using size for a discrete variable is not advised.
```

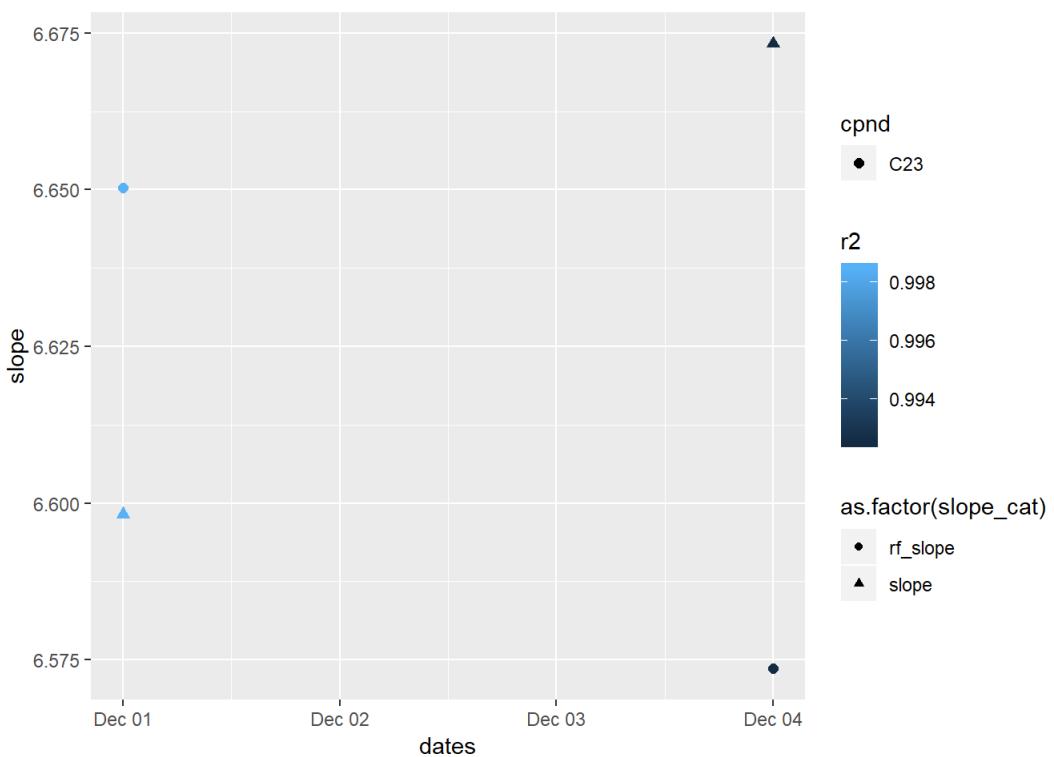


```
## Warning: Using size for a discrete variable is not advised.
```

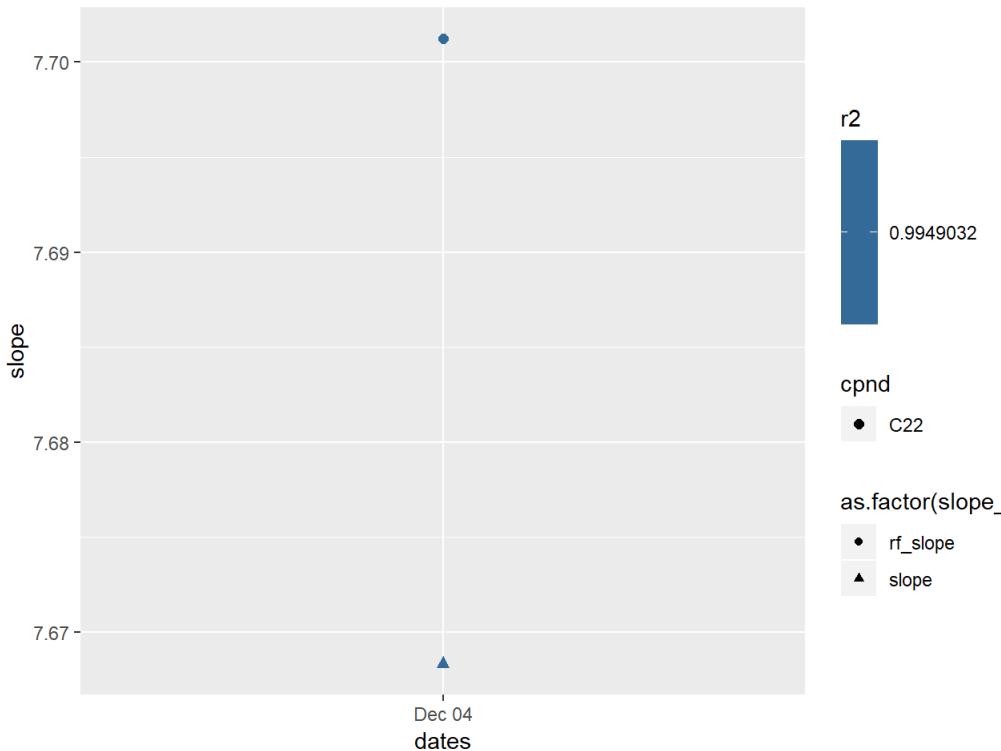




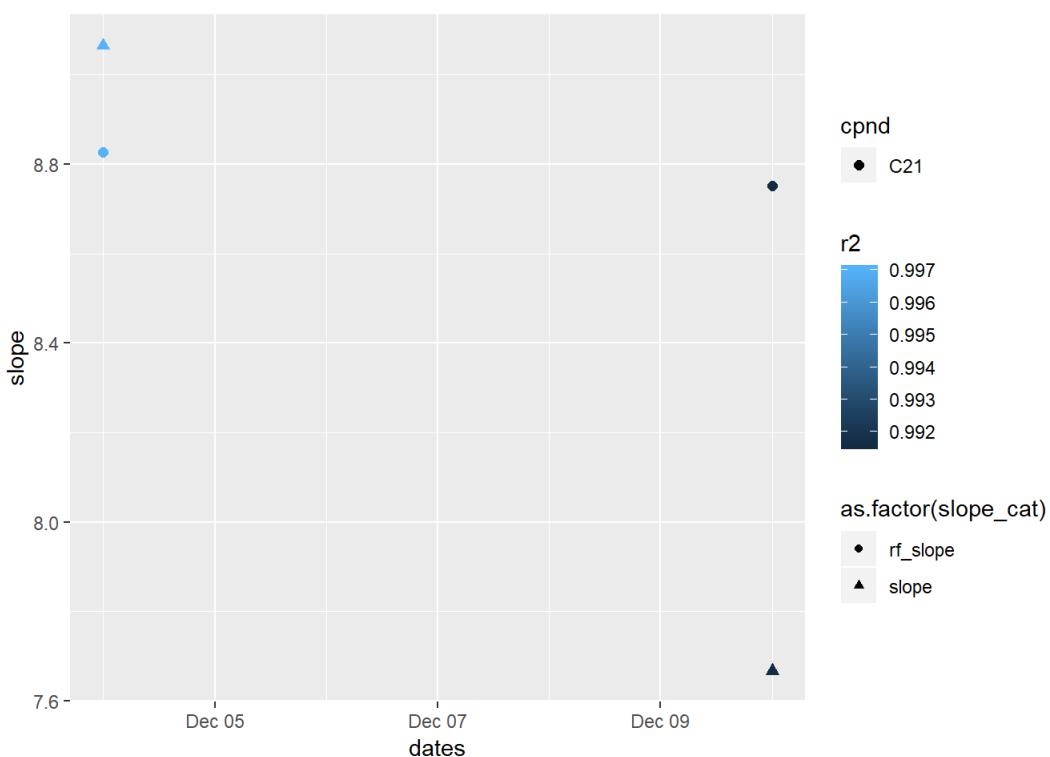
```
## Warning: Using size for a discrete variable is not advised.
```



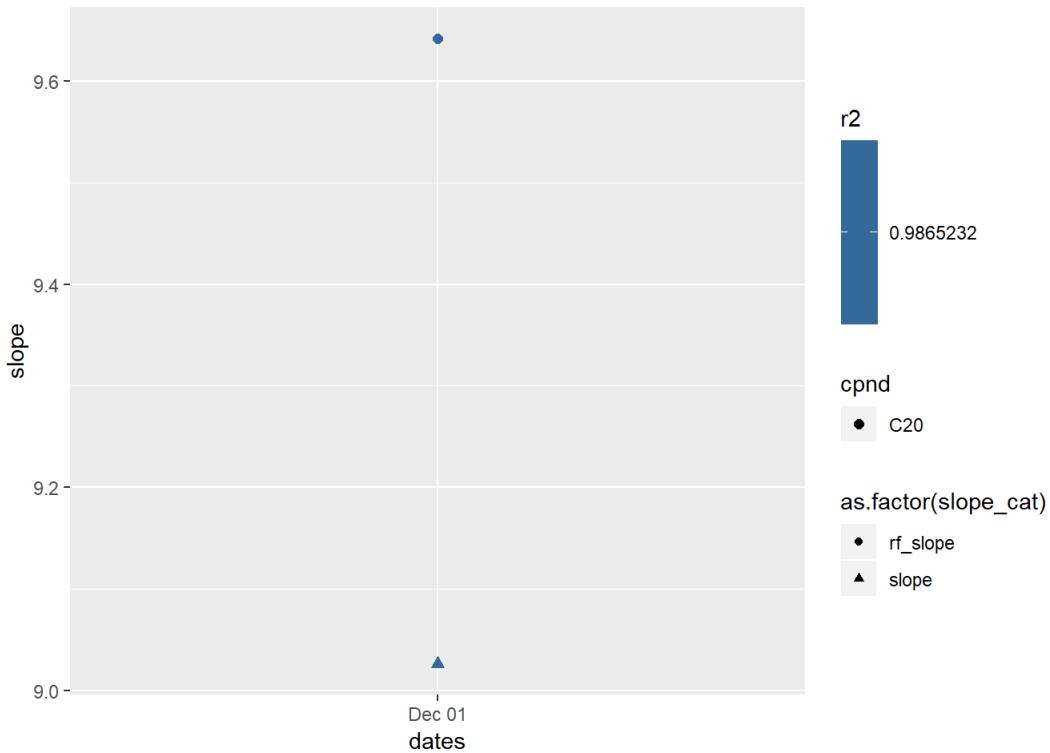
```
## Warning: Using size for a discrete variable is not advised.
```



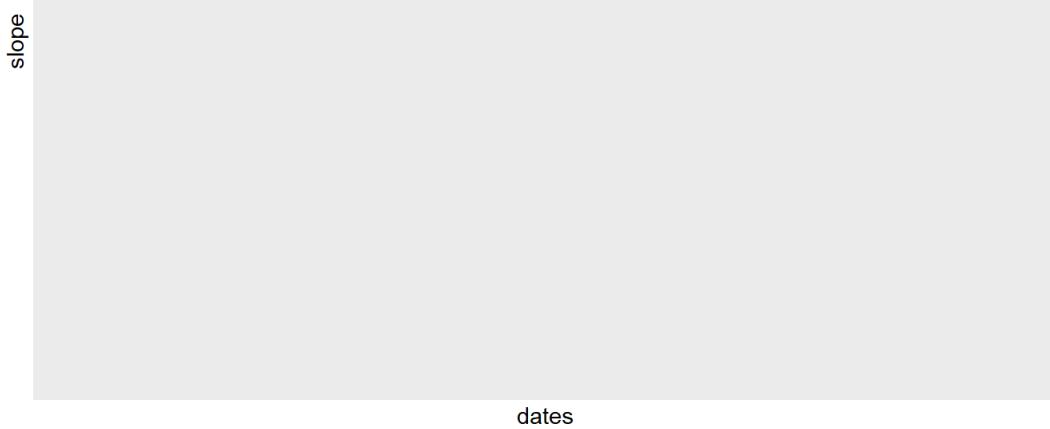
```
## Warning: Using size for a discrete variable is not advised.
```



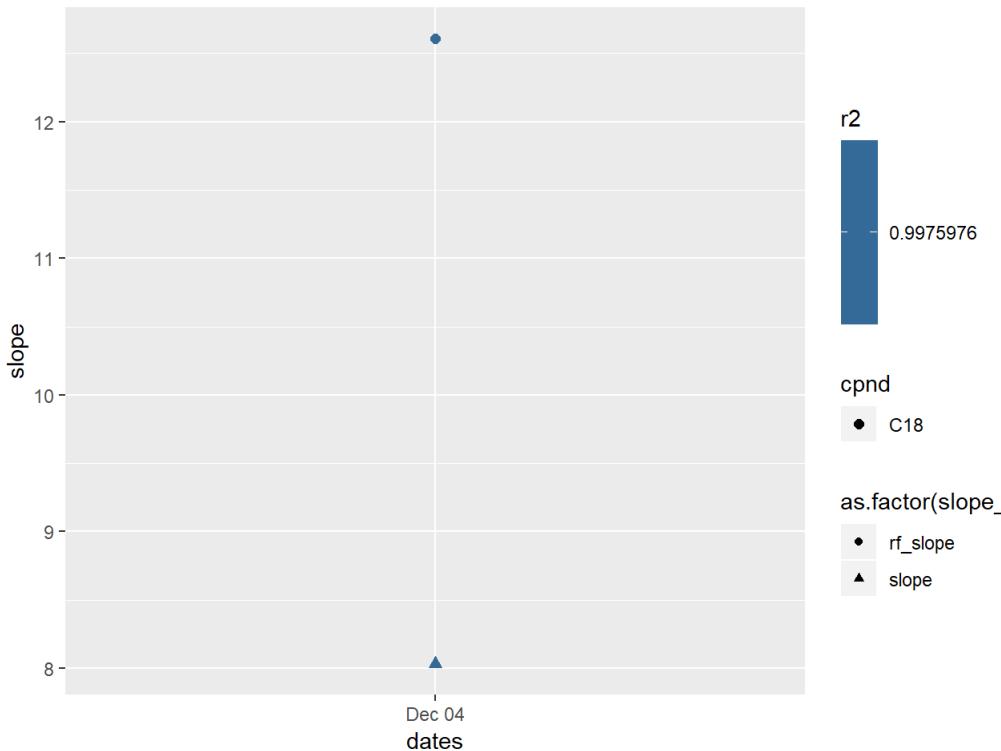
```
## Warning: Using size for a discrete variable is not advised.
```



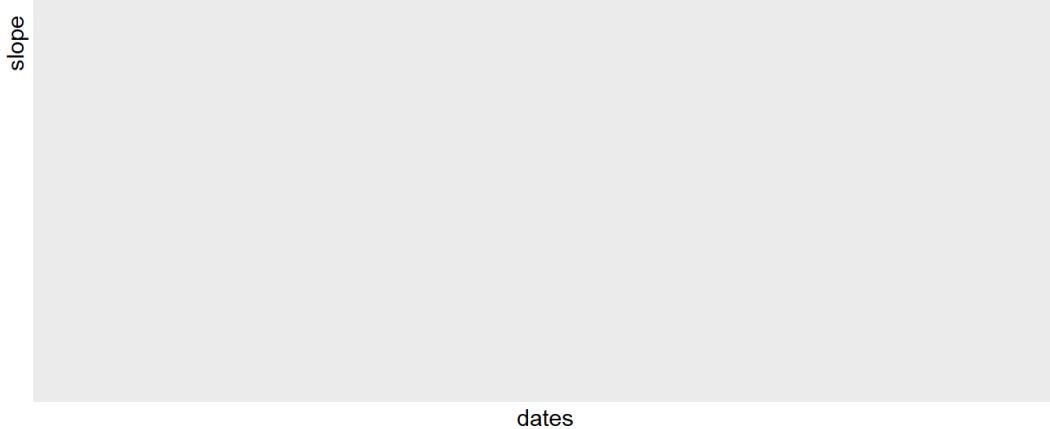
```
## Warning: Using size for a discrete variable is not advised.
```



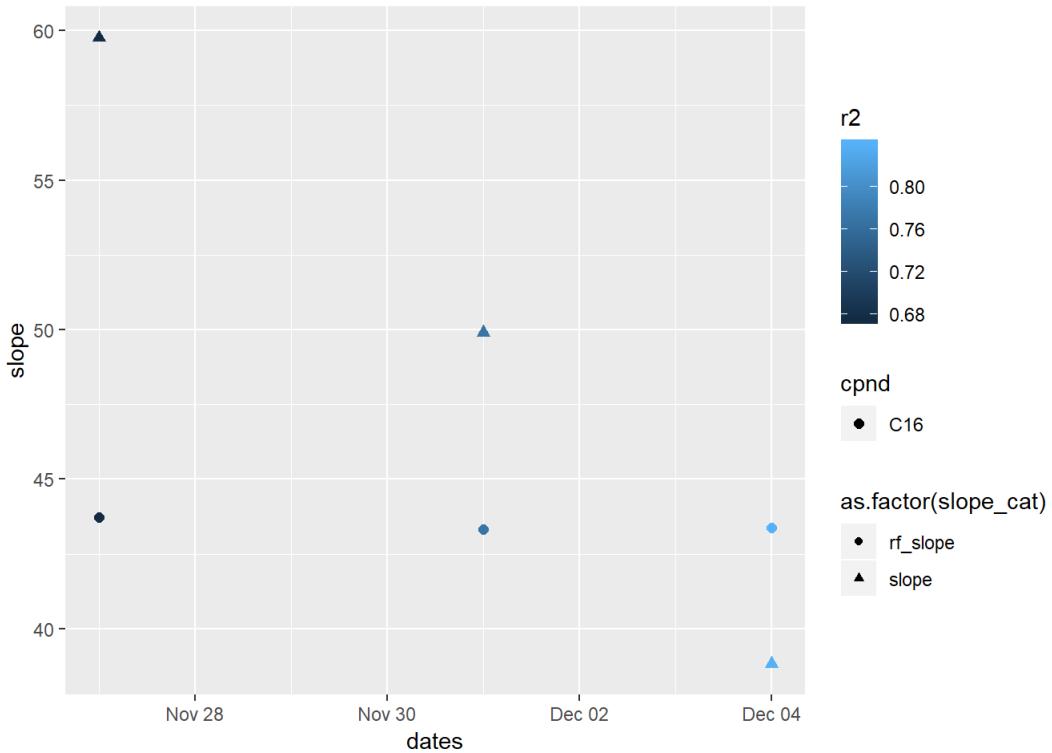
```
## Warning: Using size for a discrete variable is not advised.
```



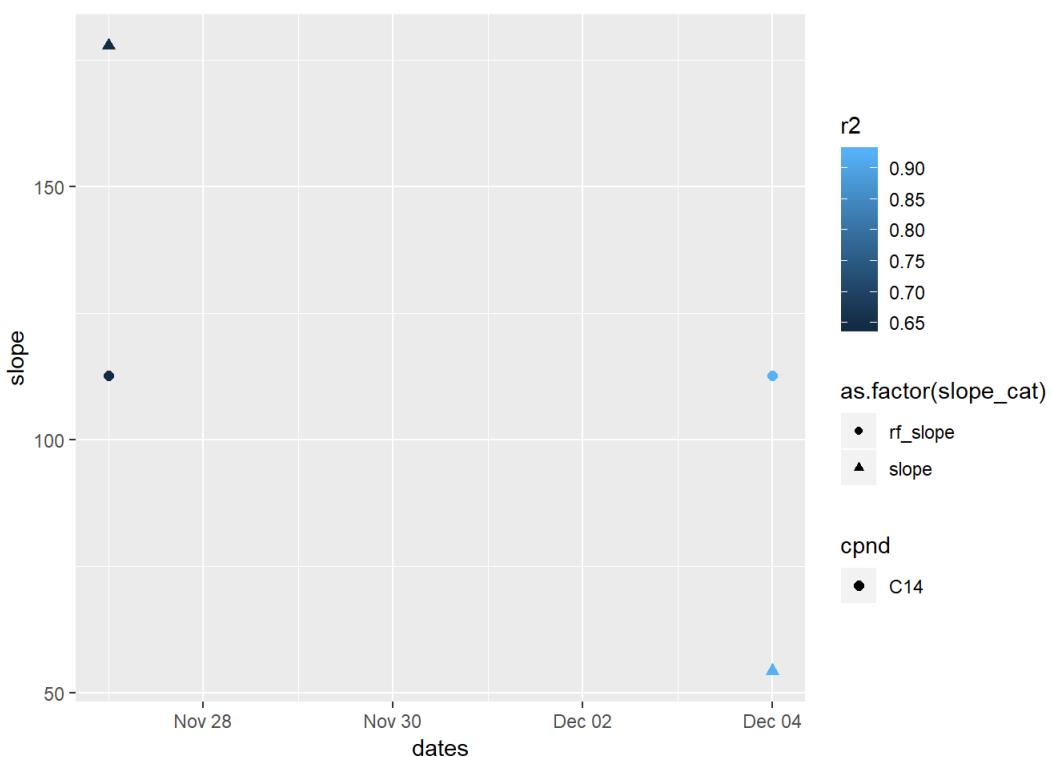
```
## Warning: Using size for a discrete variable is not advised.
```



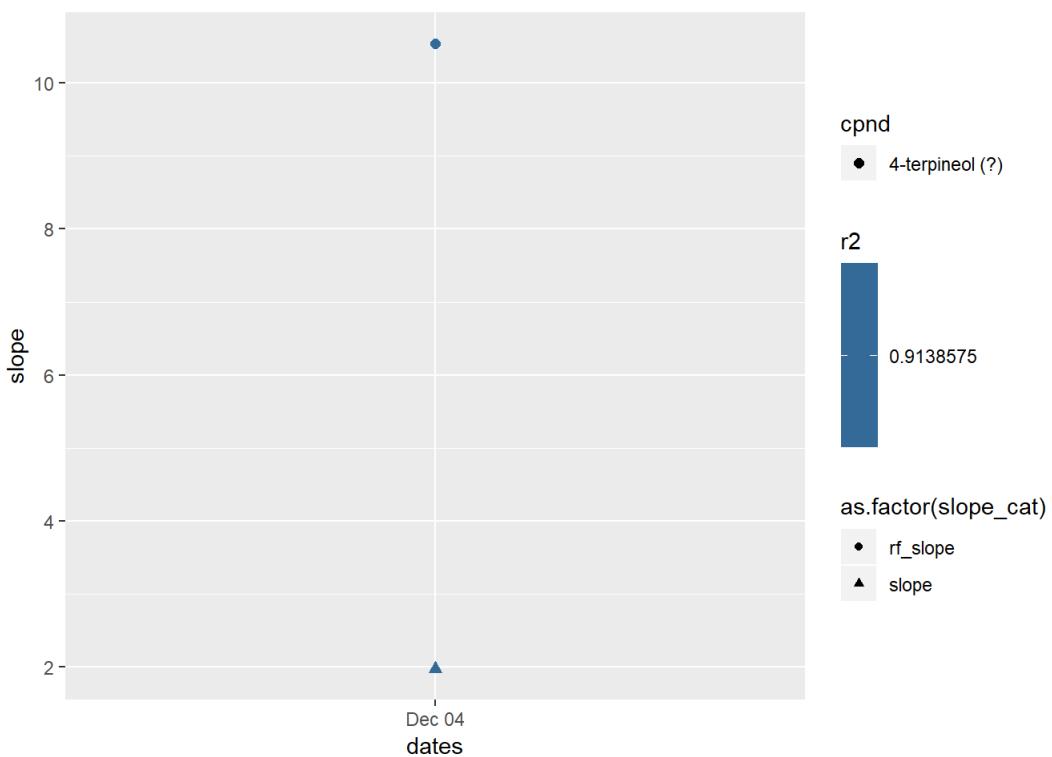
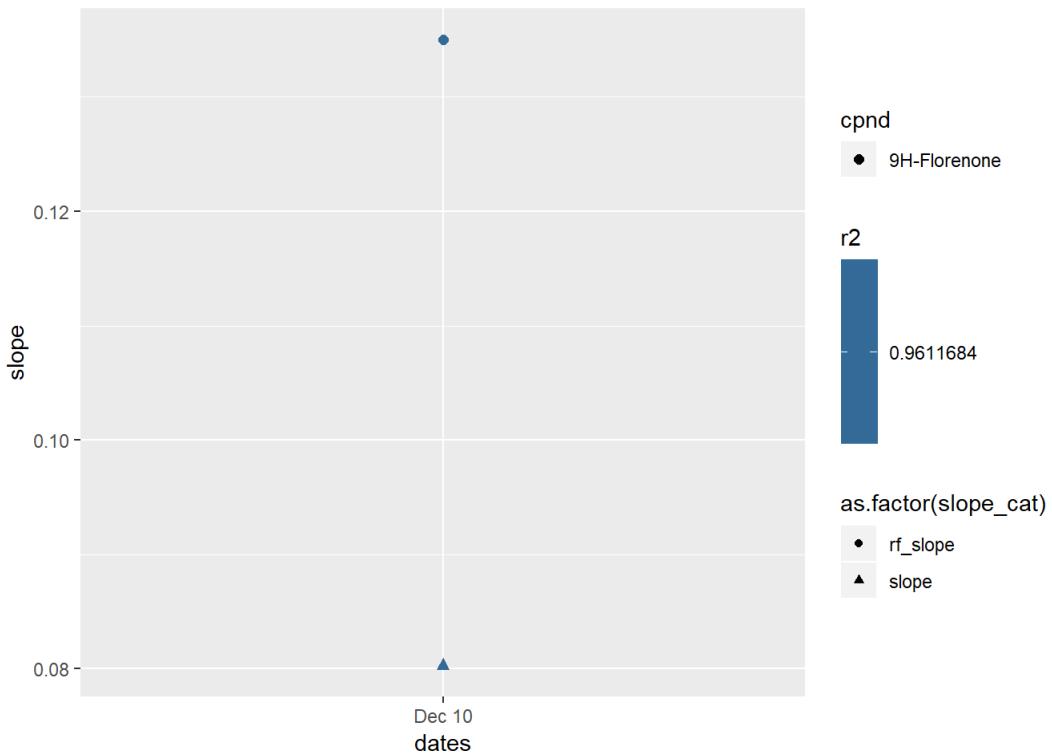
```
## Warning: Using size for a discrete variable is not advised.
```

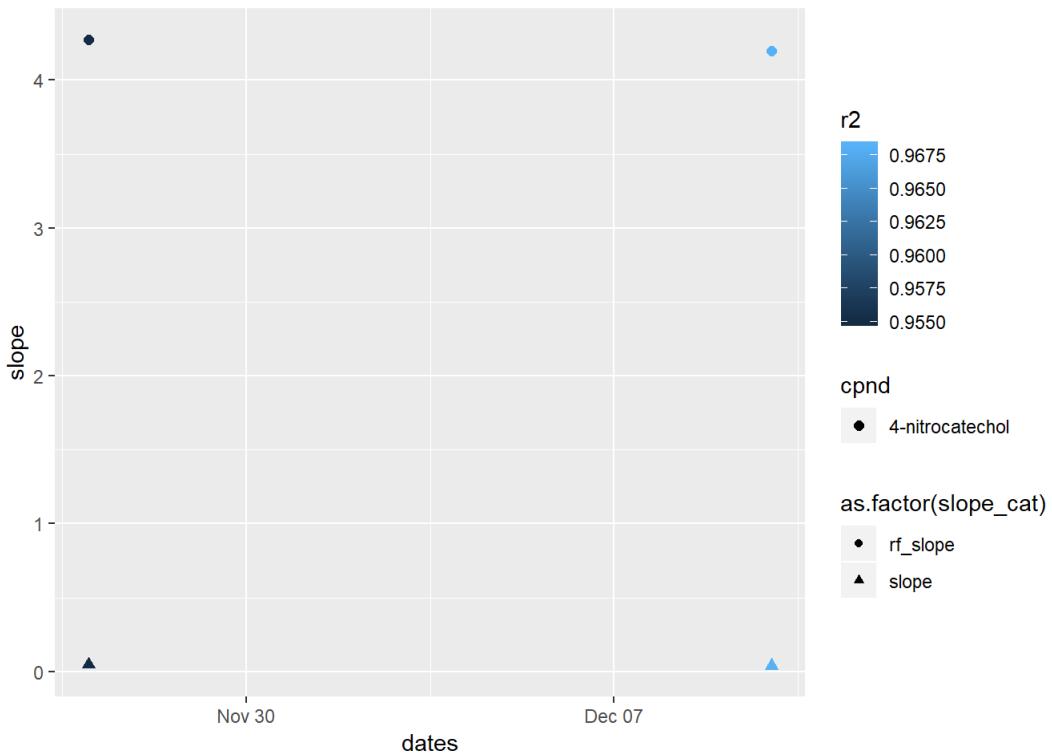


```
## Warning: Using size for a discrete variable is not advised.
```

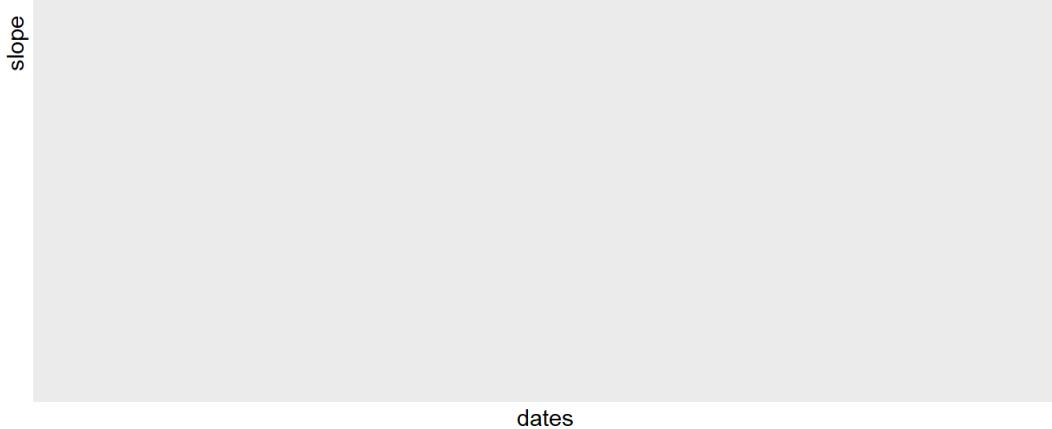


```
## Warning: Using size for a discrete variable is not advised.
```





```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```

slope

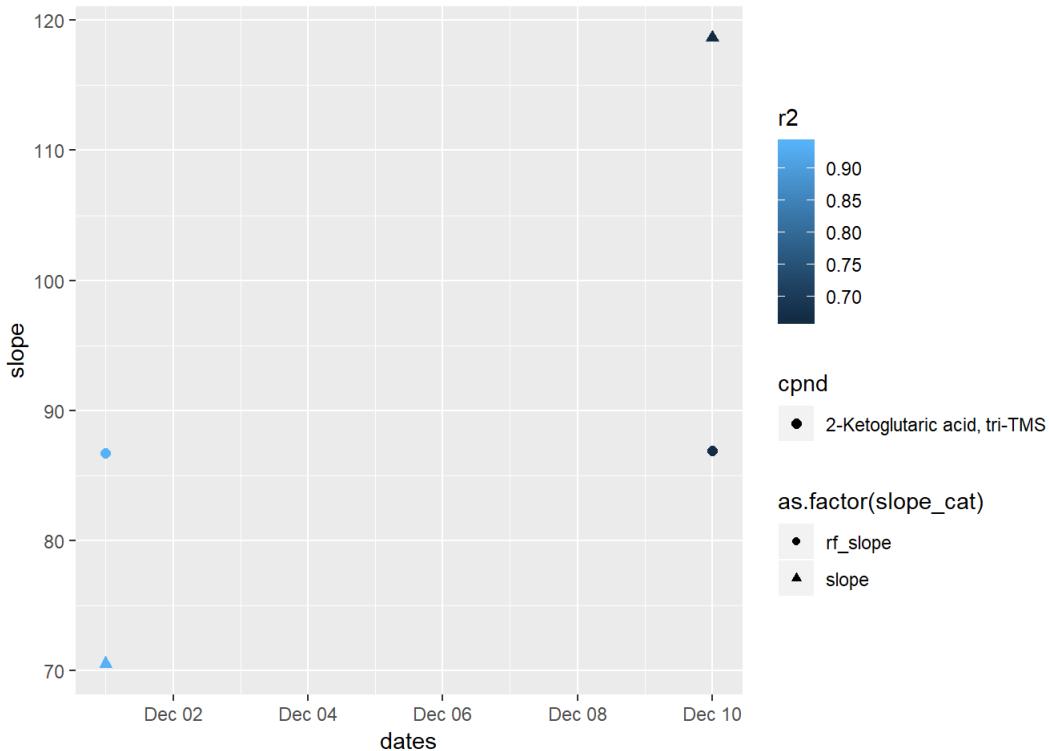
dates

```
## Warning: Using size for a discrete variable is not advised.
```

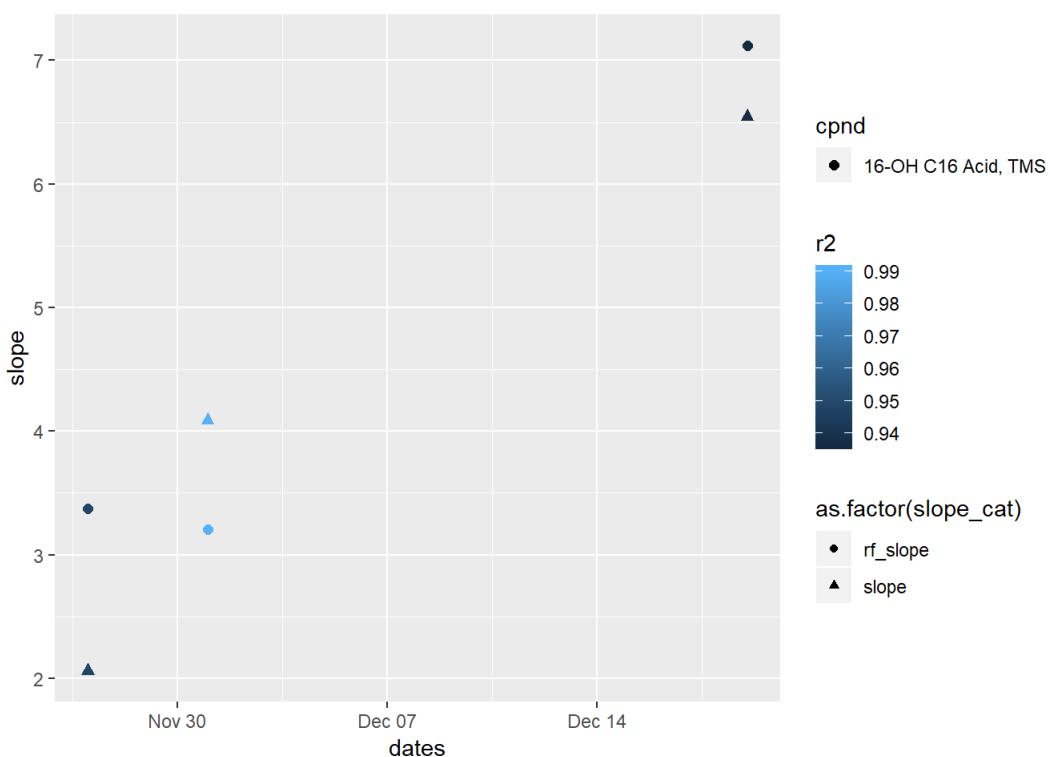
slope

dates

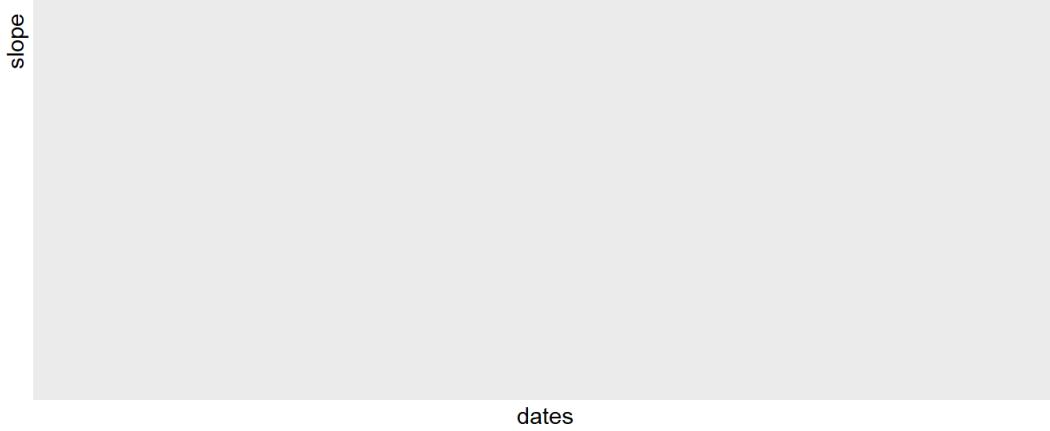
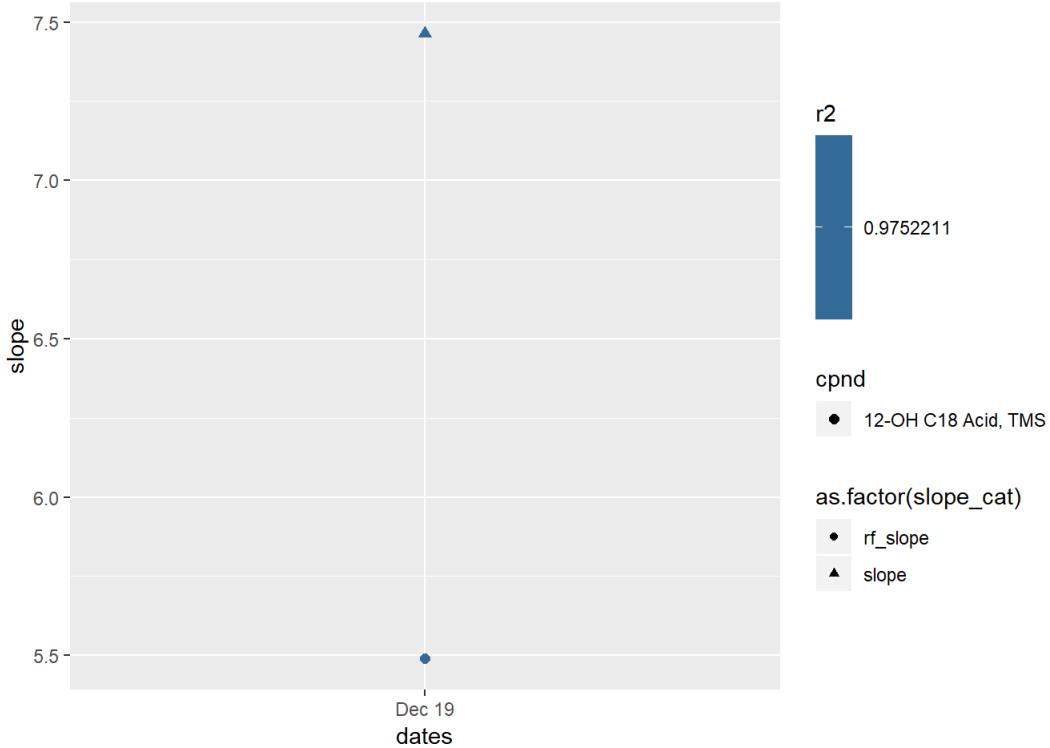
```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



```
## Warning: Using size for a discrete variable is not advised.
```



slope

dates

```
## Warning: Using size for a discrete variable is not advised.
```

slope

dates

```
## Warning: Using size for a discrete variable is not advised.
```

slope

dates

```
## Warning: Using size for a discrete variable is not advised.
```

slope

dates

```
## Warning: Using size for a discrete variable is not advised.
```

slope

dates

```
## Warning: Using size for a discrete variable is not advised.
```

slope

dates

```
## Warning: Using size for a discrete variable is not advised.
```

slope

dates

```
## Warning: Using size for a discrete variable is not advised.
```

slope

Viewing the predicted and

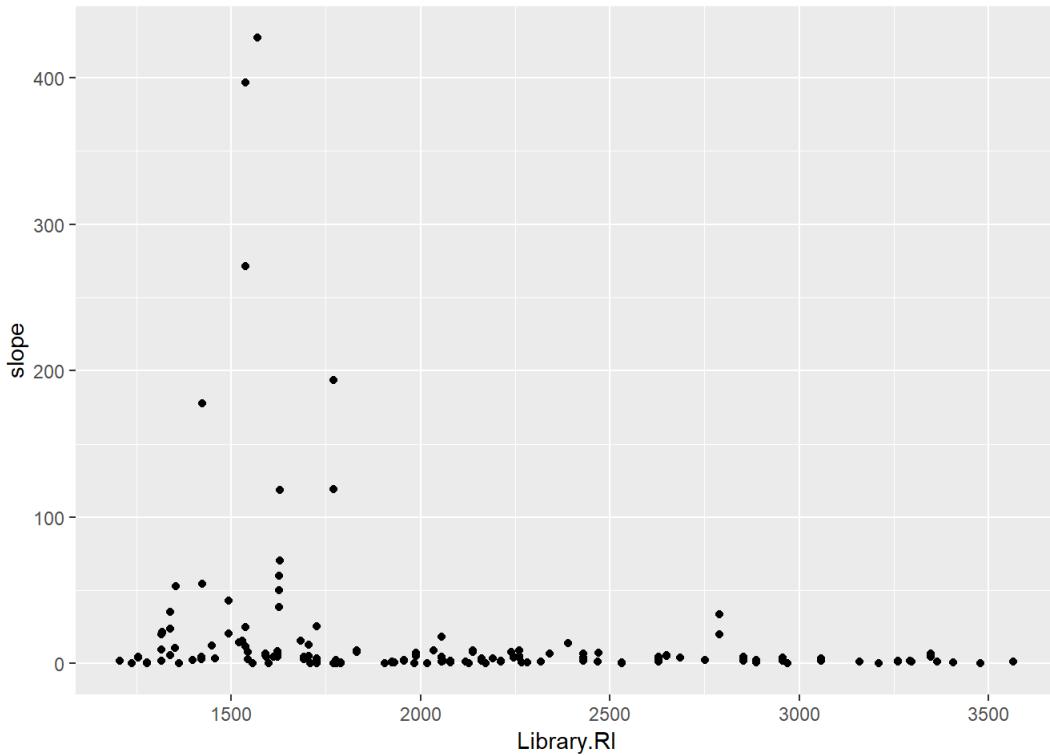
dates

actual slopes of the test set for the different times during the campaign indicates that for some the predictive capabilities are far better than for others.

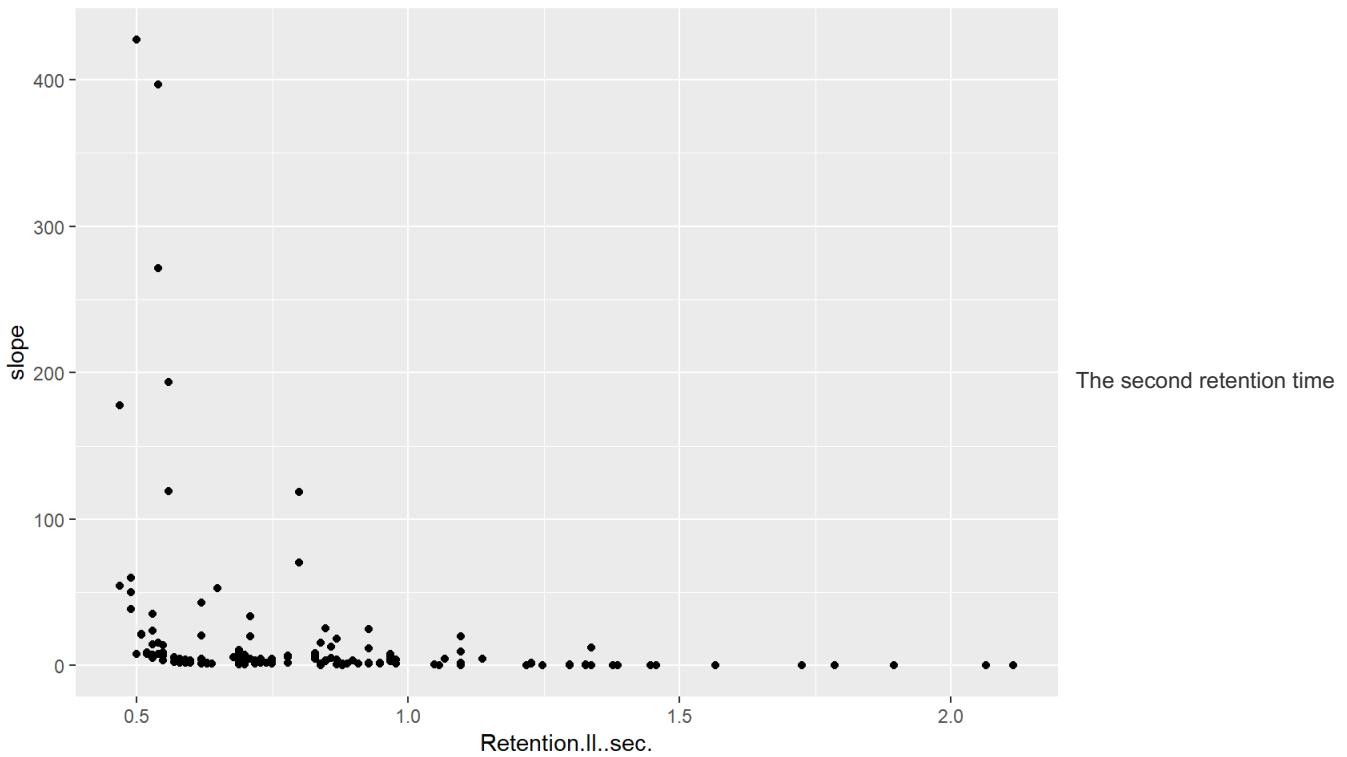
Return to Linear Modeling Possibilities

As the relationship between the injected and perceived volumes is presumed to be linear, a logical choice would be to improve the factor treatment to make a linear model perform better. A first take on the poor performance of the linear model is that the retention times are important but not in a linear fashion- this hypothesis is now explored. Visualizing slopes as a function of the retention times

```
slopemod.test.withpred %>%
  ggplot(aes(x = Library.RI, y = slope)) +
  geom_point()
```



```
slopemod.test.withpred %>%
  ggplot(aes(x = Retention.II..sec., y = slope)) +
  geom_point()
```



appears to follow a basically exponential decay, but the first retention time does not appear to adhere to any pattern that would be easily represented in a transformation for linear modeling.

```
ES.train.boost.s <- slopemod.train%>%
  mutate(loss_1 = as.factor(loss_1)) %>%
  mutate(loss_12 = as.factor(loss_12)) %>%
  mutate(loss_14 = as.factor(loss_14)) %>%
  mutate(loss_13 = as.factor(loss_13)) %>%
  mutate(loss_15 = as.factor(loss_15)) %>%
  mutate(loss_16 = as.factor(loss_16)) %>%
  mutate(loss_18 = as.factor(loss_18)) %>%
  mutate(loss_2 = as.factor(loss_2)) %>%
  mutate(loss_22 = as.factor(loss_22)) %>%
  mutate(loss_26 = as.factor(loss_26)) %>%
```

```

mutate(loss_28 = as.factor(loss_28)) %>%
mutate(loss_29 = as.factor(loss_29)) %>%
mutate(loss_30 = as.factor(loss_30)) %>%
mutate(loss_31 = as.factor(loss_31)) %>%
mutate(loss_42 = as.factor(loss_42)) %>%
mutate(loss_44 = as.factor(loss_44)) %>%
  mutate(loss_54 = as.factor(loss_54)) %>%
mutate(loss_56 = as.factor(loss_56)) %>%
mutate(loss_70 = as.factor(loss_70)) %>%
mutate(loss_74 = as.factor(loss_74)) %>%
mutate(mz_105 = as.factor(mz_105)) %>%
mutate(mz_113 = as.factor(mz_113)) %>%
mutate(mz_117 = as.factor(mz_117)) %>%
mutate(mz_129 = as.factor(mz_129)) %>%
mutate(mz_147 = as.factor(mz_147)) %>%
mutate(mz_41 = as.factor(mz_41)) %>%
mutate(mz_45 = as.factor(mz_45)) %>%
mutate(mz_43 = as.factor(mz_43)) %>%
  mutate(mz_55 = as.factor(mz_55)) %>%
mutate(mz_57 = as.factor(mz_57)) %>%
  mutate(mz_69 = as.factor(mz_69)) %>%
mutate(mz_71 = as.factor(mz_71)) %>%
mutate(mz_73 = as.factor(mz_73)) %>%
mutate(mz_75 = as.factor(mz_75)) %>%
mutate(mz_85 = as.factor(mz_85)) %>%
  mutate(mz_83 = as.factor(mz_83)) %>%
mutate(mz_81 = as.factor(mz_81)) %>%
mutate(mz_91 = as.factor(mz_91)) %>%
mutate(mz_93 = as.factor(mz_93)) %>%
  mutate(mz_99 = as.factor(mz_99)) %>%
mutate(dates = as.numeric(dates))

```

```

ES.test.boost.s <- slopemod.test%>%
  mutate(loss_1 = as.factor(loss_1)) %>%
mutate(loss_12 = as.factor(loss_12)) %>%
mutate(loss_14 = as.factor(loss_14)) %>%
  mutate(loss_13 = as.factor(loss_13)) %>%
mutate(loss_15 = as.factor(loss_15)) %>%
  mutate(loss_16 = as.factor(loss_16)) %>%
  mutate(loss_18 = as.factor(loss_18)) %>%
mutate(loss_2 = as.factor(loss_2)) %>%
mutate(loss_22 = as.factor(loss_22)) %>%
mutate(loss_26 = as.factor(loss_26)) %>%
mutate(loss_28 = as.factor(loss_28)) %>%
mutate(loss_29 = as.factor(loss_29)) %>%
mutate(loss_30 = as.factor(loss_30)) %>%
mutate(loss_31 = as.factor(loss_31)) %>%
mutate(loss_42 = as.factor(loss_42)) %>%
mutate(loss_44 = as.factor(loss_44)) %>%
  mutate(loss_54 = as.factor(loss_54)) %>%
mutate(loss_56 = as.factor(loss_56)) %>%
mutate(loss_70 = as.factor(loss_70)) %>%
mutate(loss_74 = as.factor(loss_74)) %>%
  mutate(mz_105 = as.factor(mz_105)) %>%
mutate(mz_113 = as.factor(mz_113)) %>%
mutate(mz_117 = as.factor(mz_117)) %>%
mutate(mz_129 = as.factor(mz_129)) %>%
mutate(mz_147 = as.factor(mz_147)) %>%
  mutate(mz_41 = as.factor(mz_41)) %>%
mutate(mz_45 = as.factor(mz_45)) %>%
mutate(mz_43 = as.factor(mz_43)) %>%
  mutate(mz_55 = as.factor(mz_55)) %>%
mutate(mz_57 = as.factor(mz_57)) %>%
  mutate(mz_69 = as.factor(mz_69)) %>%
mutate(mz_71 = as.factor(mz_71)) %>%
mutate(mz_73 = as.factor(mz_73)) %>%
mutate(mz_75 = as.factor(mz_75)) %>%
mutate(mz_85 = as.factor(mz_85)) %>%
  mutate(mz_83 = as.factor(mz_83)) %>%
mutate(mz_81 = as.factor(mz_81)) %>%
mutate(mz_91 = as.factor(mz_91)) %>%
mutate(mz_93 = as.factor(mz_93)) %>%
  mutate(mz_aa = as.factor(mz_aa)) %>%

```

```

mutate(dates = as.numeric(dates))

set.seed(144)
mod.boost <- gbm(slope ~ .,
                  data = ES.train.boost.s,
                  distribution = "gaussian",
                  n.trees = 1000,
                  interaction.depth = 5)

set.seed(144)
pred.boost <- predict(mod.boost, newdata = ES.test.boost.s, n.trees=1000)

SSE = sum((ES.test.boost.s$slope - pred.boost)^2)
SST = sum((ES.test.boost.s$slope - mean(ES.train.boost.s$slope))^2)
OSR2_boost = 1 - SSE/SST # is this model useful at all?

OSR2_boost_slope = OSR2(pred.boost, ES.train.boost.s$slope, ES.test.boost.s$slope)
#OSR2_boost_slope

mae_boosted.s <- MAE(obs = ES.test.boost.s$slope, pred = pred.boost)

```

A boosted model for the slopes performs slightly worse than the random forest model- the OSR squared is .35 and the MAE is 15.6.

For future model assessment, a tables are created indicating which compounds are well predicted and which are poorly predicted. This will hopefully enable assessment of why some compounds are being predicted so much more accurately than others.

```

pred <- pred.boost
slope <- ES.test.boost.s$slope
cpnd <- slopemod.test.o$cpnd

r2 <- ES.test.boost.s$r2

#comp <- data.frame(pred, slope, cpnd, r2)
comp <- data.frame(pred, slope, cpnd)

comp <- comp %>%
  mutate(diff = abs(pred-slope)) %>%
  mutate(pctdiff = diff/pred)

comp_badfit <- comp %>%
  filter(pctdiff > 1)

table(comp_badfit$cpnd)

```

```

##                               12-OH C18 Acid, TMS
##                               0
##                               16-OH C16 Acid, TMS
##                               0
##       2-Ketoglutaric acid, tri-TMS
##                               0
##       4-nitrocatechol
##                               0
##       4-terpineol (?)
##                               0
##       9H-Florenone
##                               0
##       a-Amyrin
##                               0
##       abietic acid
##                               0
##       anthraquinone
##                               0
##       beta-caryophyllinic acid, TMS
##                               0
##       beta-caryophyllonic acid, TMS
##                               0
##       beta-nocaryophyllinic acid, TMS
##                               0
##       beta-nocarvophyllonic acid, TMS

```

0
beta-sitosterol 0
bisabolol 0
borneol 0
C12 diacid 0
C13 acid 0
C14 0
C14 Diacid 0
C16 0
C16 Acid 0
C18 1
C18 Acid 0
C20 0
C21 0
C22 0
C22 acid 0
C23 0
C24 acid 0
C26 0
C27 0
C28 0
C28 acid 0
C29 0
C30 1
C30 0
C31 0
C32 0
C33 0
C35 0
cholesterol 0
chrysene 0
cis-Vaccenic Acid, TMS 1
citronellol 0
cycloisolongifolene 0
DEET 0
deoxycholic Acid (maybe) 0
dimethyl Glutaric Acid, TMS 0
eicosanol, TMS ^

U
ergosterol
0
erythreitol
0
FAME16 (Methyl Palmitate)
1
FAME18 (Methyl Stearate)
0
farnesol
0
galactosan
0
gamma dodecalactone
0
glyceric acid (?)
0
hexadecanamide
0
hexadecanol, TMS
1
homosalate, TMS
0
hydroquinone
0
ionone
0
isoeugenol
0
levoglucosan
0
linoleic Acid, TMS
0
maltol
1
mannosan
0
MBTCA
0
Me-OH-glutatric acid (?)
0
monopalmitin
0
monostearin TMS
0
nonanol
1
octadecanal (?)
0
octadecanol
0
p-anisic acid (4-methoxybenzoic acid)
0
palmitoleic Acid
1
pentadecanone
0
perylene
0
phthalimide
1
pinic acid 1
0
pinic acid 2
1
pyrene
0
pyrocatechol
0
quinoline
0
resorcinol
0

```

##          retene
##          0
##          SESQ6
##          0
##          SESQ7
##          0
##          SESQ8
##          0
##          SESQ9
##          1
##          syringaldehyde
##          0
##          syringic acid
##          0
##          threitol
##          0
##          triacetin
##          0
##          tridecanal (?)
##          0
##          vanilllic acid
##          0
##          vanillin
##          0
##          verbenone (-)
##          0
##          xanthone
##          0

```

```

comp_goodfit <- comp %>%
  filter(pctdiff < .5)

table(comp_goodfit$cpnd)

```

```

##          12-OH C18 Acid, TMS
##          1
##          16-OH C16 Acid, TMS
##          3
##          2-Ketoglutaric acid, tri-TMS
##          2
##          4-nitrocatechol
##          1
##          4-terpineol (?)
##          0
##          9H-Florenone
##          1
##          a-Amyrin
##          0
##          abietic acid
##          1
##          anthraquinone
##          3
##          beta-caryophyllinic acid, TMS
##          1
##          beta-caryophyllonic acid, TMS
##          0
##          beta-nocaryophyllinic acid, TMS
##          0
##          beta-nocaryophyllonic acid, TMS
##          1
##          beta-sitosterol
##          0
##          bisabolol
##          2
##          borneol
##          1
##          C12 diacid
##          4
##          C13 acid
##          0
##          ...
##          ...

```

C14
1
C14 Diacid
0
C16
2
C16 Acid
0
C18
0
C18 Acid
0
C20
0
C21
2
C22
0
C22 acid
0
C23
2
C24 acid
1
C26
2
C27
1
C28
2
C28 acid
1
C29
1
C30
1
C31
1
C32
1
C33
0
C35
0
cholesterol
0
chrysene
1
cis-Vaccenic Acid, TMS
1
citronellol
2
cycloisolongifolene
1
DEET
1
deoxycholic Acid (maybe)
2
dimethyl Glutaric Acid, TMS
0
eicosanol, TMS
1
ergosterol
1
erythreitol
1
FAME16 (Methyl Palmitate)
0
FAME18 (Methyl Stearate)
1
farnesol
0
galactosan

```

##          1
## gamma dodecalactone
##          1
## glyceric acid (?)
##          0
## hexadecanamide
##          0
## hexadecanol, TMS
##          1
## homosalate, TMS
##          1
## hydroquinone
##          3
## ionone
##          0
## isoeugenol
##          0
## levoglucosan
##          3
## linoleic Acid, TMS
##          2
## maltol
##          2
## mannosan
##          1
## MBTCA
##          1
## Me-OH-glutatric acid (?)
##          0
## monopalmitin
##          2
## monostearin TMS
##          1
## nonanol
##          0
## octadecanal (?)
##          2
## octadecanol
##          0
## p-anisic acid (4-methoxybenzoic acid)
##          1
## palmitoleic Acid
##          0
## pentadecanone
##          1
## perylene
##          0
## phthalimide
##          0
## pinic acid 1
##          4
## pinic acid 2
##          0
## pyrene
##          1
## pyrocatechol
##          1
## quinoline
##          3
## resorcinol
##          2
## retene
##          0
## SESQ6
##          0
## SESQ7
##          1
## SESQ8
##          1
## SESQ9
##          0
## syringaldehyde
##          4

```

```

##          syringic acid
##                               3
##          threitol
##                               1
##          triacetin
##                               1
##          tridecanal (?) 
##                               2
##          vanilllic acid
##                               1
##          vanillin
##                               1
##          verbenone (-)
##                               2
##          xanthone
##                               0

```

```

#comp_badfit %>%
  #ggplot(aes(x = abs(pred-slope), y = r2, color = cpnd)) +
  #geom_point()

```

```

test_slopes <- ES.test.boost.s %>%
  dplyr::select(cpnd, slope, r2, dates)

test_slopes$pslope <- RFPredictions

test_slopes <- test_slopes %>%
  mutate(dates = as.Date(dates, origin = "1970-01-01"))

Es.test.withslopes <- ES.test %>%
  left_join(test_slopes, by = c("Compound.Name"= "cpnd", "r_rundate" ="dates"))

```

TRYING TO FIGURE OUT NEAREST NEIGHBORS

As none of the previous methods achieved acceptably high performance, moving forward the best option would likely be assignment to an external standard by nearest neighbors. This option is preliminarily examined in the following section and will be further explored before a final method for external calibration is selected.

```

cal_curves_trimmed <- cal_curves %>%
  filter(slope > 0)

#cal_curves_t_slopes <- column_to_rrownames(cal_curves_trimmed, 'cpnd')

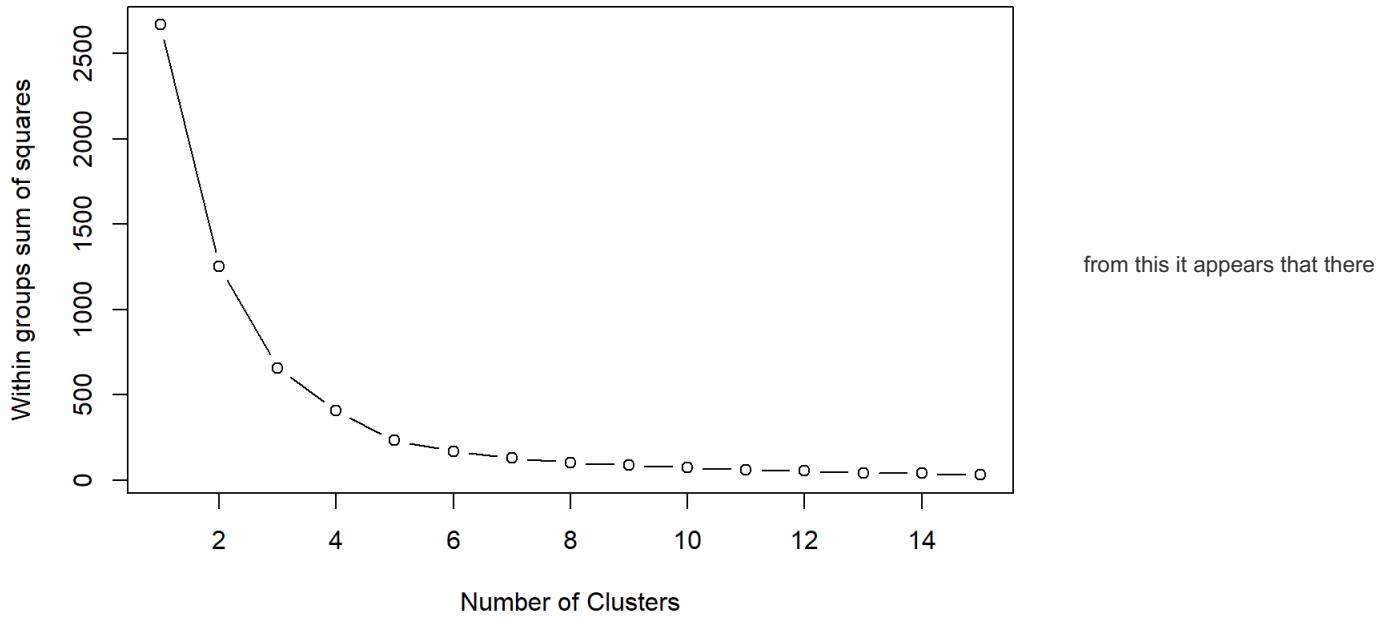
cal_curves_t_slopes <- cal_curves_trimmed %>%
  mutate(logslope = log(slope)) %>%
  dplyr::select(logslope)

wss <- (nrow(cal_curves_trimmed)-1)*sum(apply(cal_curves_t_slopes, 2, var))

for (i in 2:15) wss[i] <- sum(kmeans(cal_curves_t_slopes,
  centers=i)$withinss)

plot(1:15, wss, type="b", xlab="Number of Clusters",
  ylab="Within groups sum of squares")

```



are really only 4 categories that I should be using

```
fit <- kmeans(cal_curves_t_slopes, 4) # 5 cluster solution
# get cluster means
aggregate(cal_curves_t_slopes, by=list(fit$cluster), FUN=mean)
```

```
##   Group.1    logslope
## 1       1  1.2105531
## 2       2 -0.8673891
## 3       3  3.8992650
## 4       4 -4.4941055
```

```
# append cluster assignment
mydata <- data.frame(cal_curves_trimmed, fit$cluster)
```

not sure if that did anything

```

#install.packages("RANN")
library(RANN)
#install.packages("FNN")
library(FNN)

#note: c7 acid has no RI

locations <- cat %>%
  mutate(rt2scale = RT2 * 1000) %>%
  dplyr::select(Name, d_alkane_RTI, rt2scale) %>%
  filter(!is.na(d_alkane_RTI))

locations_t <- locations %>%
  dplyr::select(-Name)

test <- locations[2:3]
closest <- nn2(locations[2:3], k=3)

closest_2 <- closest$nn.idx[,2:3]

colnames(closest_2) <- c("n1_index", "n2_index")

locations <- cbind(locations, closest_2)

locations <- locations %>%
  mutate(n1 = Name) %>%
  mutate(n2 = Name)

for(i in 1:nrow(locations)){
  n1_index <- locations$n1_index[i]
  n2_index <- locations$n2_index[i]

  locations$n1[i] <- locations>Name[n1_index]
  locations$n2[i] <- locations>Name[n2_index]

}

#closest <- nn2(locations_t, k = 2, searchtype = "radius", radius = 0.001)
closest <- sapply(closest, cbind) %>% as_tibble

k <- knn(locations_t, locations_t, labels, k = 2, algorithm="cover_tree")
indices <- attr(k, "nn.index")

```

```

merged_cal_curves <- cal_curves %>%
  filter(slope > 0) %>%
  group_by(cpnd) %>%
  summarise(meanSlope = mean(slope)) %>%
  left_join(locations, by = c("cpnd"= "Name"))

```

Notes on future steps: try k nearest neighbors but kmeans might be better with small dataset take slope of predicted based on the random forest- will be better, more data

Appendix B: External Standard Compounds and Data
Example

RT1	RT2	Description	Column_T	Retention_d	alkane_Contributc	Num	Peak:MS
57.46	0.69	cis-Vaccenic Acid, TMS	SemiStdNF	2259	2260	Emily Barn	221 40 7; 41 207; 42 38; 43 144; 44
29.32	0.53	citronellol	SemiStdNF	1338	1339	Emily Barn	113 40 10; 41 213; 42 15; 43 44; 44
30.04	0.65	cycloisolongifolene	SemiStdNF	1355	1355	Emily Barn	136 40 11; 41 143; 42 8; 43 67; 44
78.9	0.97	deoxycholic Acid (maybe)	SemiStdNF	3347	3347	Emily Barn	308 40 2; 41 31; 42 4; 43 28; 44 10;
77.79	1.52	dibenz(ah)anthracene	SemiStdNF	3280	3280	Emily Barn	107 50 1; 51 1; 62 1; 63 3; 74 3;
34	0.9	dimethyl Glutaric Acid, TMS	SemiStdNF	1456	1457	Emily Barn	155 40 2; 41 17; 42 8; 43 36; 44 17;
48.66	0.62	dodecyl benzene	SemiStdNF	1920	1921	Emily Barn	69 40 2; 41 56; 42 7; 43 64; 44 2;
60.54	0.55	eicosanol, TMS	SemiStdNF	2390	2390	Emily Barn	114 40 2; 41 83; 42 13; 43 123; 44
78.06	0.84	ergosterol	SemiStdNF	3296	3297	Emily Barn	307 40 6; 41 179; 42 15; 43 179; 44
46.15	0.55	farnesol	SemiStdNF	1832	1832	Emily Barn	157 40 12; 41 297; 42 24; 43 139;
41.62	0.84	galactosan	SemiStdNF	1684	1684	Emily Barn	130 40 2; 41 4; 42 2; 43 15; 44 9;
42.42	1.9	gamma dodecalactone	SemiStdNF	1709	1710	Emily Barn	86 40 4; 41 83; 42 21; 43 60; 44 4;
42.61	1.97	beta-caryophyllene aldehyde	SemiStdNF	1715	1716	Emily Barn	138 40 14; 41 231; 42 22; 43 999;
36.82	0.54	SESQ7	SemiStdNF	1537	1538	Emily Barn	115 40 15; 41 539; 42 28; 43 36; 44
37.93	0.5	SESQ8	SemiStdNF	1569	1569	Emily Barn	111 40 11; 41 117; 42 9; 43 44; 44
52.47	0.93	beta-caryophyllinic acid, TMS	SemiStdNF	2060	2061	Emily Barn	209 40 1; 41 36; 42 4; 43 26; 44 8;
48.97	1.33	beta-caryophyllonic acid, TMS	SemiStdNF	1931	1932	Emily Barn	200 40 6; 41 93; 42 11; 43 384; 44
54.18	1.25	beta-nocaryophyllinic acid, TMS	SemiStdNF	2127	2127	Emily Barn	236 40 2; 41 46; 42 5; 43 34; 44 13;
43.9	0.27	beta-nocaryophyllone aldehyde	SemiStdNF	1757	1757	Emily Barn	146 40 15; 41 200; 42 21; 43 999;
50.49	1.73	beta-nocaryophyllonic acid, TMS	SemiStdNF	1985	1985	Emily Barn	213 40 8; 41 134; 42 15; 43 795; 44
79.89	0.87	beta-sitosterol	SemiStdNF	3406	3407	Emily Barn	286 40 6; 41 124; 42 15; 43 286; 44
44.32	0.56	bisabolol	SemiStdNF	1770	1771	Emily Barn	146 40 7; 41 168; 42 9; 43 39; 44 8;
25.78	0.62	borneol	SemiStdNF	1254	1254	Emily Barn	115 40 3; 41 79; 42 6; 43 42; 44 9;
76.61	0.84	cholesterol	SemiStdNF	3209	3209	Emily Barn	265 40 3; 41 136; 42 11; 43 183; 44
63.63	1.3	chrysene	SemiStdNF	2531	2532	Emily Barn	65 50 2; 51 1; 61 1; 62 3; 63 4;
27.34	0.72	C8 acid	SemiStdNF	1293	1293	Emily Barn	119 40 4; 41 59; 42 15; 43 51; 44
45.88	0.95	C9 Diacid (azelaic acid)	SemiStdNF	1822	1823	Emily Barn	172 40 1; 41 32; 42 10; 43 70; 44
31.22	0.69	C9 acid	SemiStdNF	1381	1382	Emily Barn	115 40 6; 41 75; 42 18; 43 70; 44
38.99	1.79	DEET	SemiStdNF	1600	1600	Emily Barn	55 40 1; 41 4; 42 8; 44 6; 50 3;
49.69	0.73	FAME16 (Methyl Palmitate)	SemiStdNF	1957	1957	Emily Barn	94 40 4; 41 134; 42 25; 43 181; 44
55.06	0.72	FAME18 (Methyl Stearate)	SemiStdNF	2161	2161	Emily Barn	105 40 4; 41 139; 42 25; 43 206; 44
44.51	0.98	MBTCA	SemiStdNF	1776	1777	Emily Barn	157 40 1; 41 24; 42 7; 43 31; 44 13;
39.72	0.83	Me-OH-glutatric acid (?)	SemiStdNF	1623	1623	Emily Barn	142 41 2; 42 2; 43 31; 44 12; 45 81;
81.14	1.06	a-Amyrin	SemiStdNF	3479	3479	Emily Barn	210 41 75; 42 4; 43 38; 44 3; 45 19;

62.26	0.84 abietic acid	SemiStdNF	2468	2468 Emily Barn	177 40 1; 41 35; 42 2; 43 31; 44 3;
51.37	1.45 anthraquinone	SemiStdNF	2017	2018 Emily Barn	88 49 5; 50 70; 51 17; 52 4; 53 5;
41.01	1.18 benzophenone	SemiStdNF	1664	1665 Emily Barn	44 49 3; 50 35; 51 121; 52 6; 53 1;
48.7	0.94 C10 diacid (sebacic acid)	SemiStdNF	1922	1922 Emily Barn	183 40 1; 41 40; 42 10; 43 83; 44
53.99	0.91 C12 diacid	SemiStdNF	2120	2120 Emily Barn	203 40 1; 41 44; 42 10; 43 56; 44
44.51	0.69 C13 acid	SemiStdNF	1776	1777 Emily Barn	127 40 2; 41 84; 42 16; 43 85; 44 8;
58.83	0.89 C14 Diacid	SemiStdNF	2317	2318 Emily Barn	229 40 2; 41 47; 42 9; 43 56; 44 12;
52.93	0.69 C16 Acid	SemiStdNF	2078	2079 Emily Barn	132 40 2; 41 98; 42 18; 43 122; 44
55.48	0.67 C17 acid	SemiStdNF	2177	2178 Emily Barn	157 40 2; 41 96; 42 18; 43 124; 44
57.95	0.7 C18 Acid	SemiStdNF	2280	2281 Emily Barn	135 40 2; 41 105; 42 19; 43 143; 44
66.83	0.69 C22 acid	SemiStdNF	2684	2685 Emily Barn	169 40 2; 41 104; 42 20; 43 174; 44
70.79	0.7 C24 acid	SemiStdNF	2886	2887 Emily Barn	171 40 2; 41 103; 42 18; 43 184; 44
74.48	0.72 C26 acid	SemiStdNF	3088	3088 Emily Barn	178 40 3; 41 111; 42 21; 43 218; 44
77.98	0.74 C28 acid	SemiStdNF	3291	3292 Emily Barn	185 40 2; 41 111; 42 20; 43 223; 44
23.15	0.77 C7 acid	User	0	Emily Barn	110 40 4; 41 45; 42 10; 43 57; 44
33.74	1.63 SESQ2	SemiStdNF	1449	1449 Emily Barn	85 42 4; 43 45; 44 4; 45 9; 47 9;
61.72	0.55 C24	SemiStdNF	2443	2444 Emily Barn	104 40 5; 41 199; 42 40; 43 440; 44
63.93	0.56 C25	SemiStdNF	2545	2546 Emily Barn	105 40 5; 41 187; 42 37; 43 449; 44
66.1	0.57 C26	SemiStdNF	2649	2649 Emily Barn	109 40 5; 41 186; 42 36; 43 439; 44
68.16	0.57 C27	SemiStdNF	2750	2751 Emily Barn	101 40 4; 41 179; 42 35; 43 442; 44
70.14	0.58 C28	SemiStdNF	2852	2853 Emily Barn	101 40 4; 41 170; 42 34; 43 429; 44
72.08	0.59 C29	SemiStdNF	2955	2956 Emily Barn	101 40 5; 41 167; 42 32; 43 430; 44
73.95	0.6 C30	SemiStdNF	3058	3058 Emily Barn	101 40 4; 41 160; 42 31; 43 427; 44
75.74	0.62 C31	SemiStdNF	3159	3159 Emily Barn	98 40 4; 41 156; 42 30; 43 427; 44
77.45	0.63 C32	SemiStdNF	3259	3260 Emily Barn	97 40 4; 41 147; 42 29; 43 420; 44
79.16	0.64 C33	SemiStdNF	3363	3363 Emily Barn	104 40 4; 41 144; 42 29; 43 412; 44
82.63	0.72 C35	SemiStdNF	3564	3565 Emily Barn	109 40 4; 41 133; 42 27; 43 386; 44
34.8	0.69 C10 carboxylic acid	SemiStdNF	1479	1480 Emily Barn	128 40 3; 41 89; 42 22; 43 81; 44
44.28	1.57 4-nitrocatechol	SemiStdNF	1769	1770 Emily Barn	85 42 1; 43 13; 44 7; 45 79; 46 5;
23.88	0.78 4-terpineol (?)	SemiStdNF	1206	1207 Emily Barn	92 40 16; 41 141; 42 20; 43 282;
44.55	1.39 9H-Florenone	SemiStdNF	1778	1778 Emily Barn	55 49 2; 50 13; 51 9; 52 2; 53 1;
32.82	0.47 C14	SemiStdNF	1422	1423 Emily Barn	69 40 9; 41 251; 42 57; 43 549; 44
39.83	0.49 C16	SemiStdNF	1626	1627 Emily Barn	73 40 7; 41 238; 42 51; 43 512; 44
43.07	0.49 C17	SemiStdNF	1730	1730 Emily Barn	78 40 7; 41 232; 42 48; 43 493; 44
46.11	0.5 C18	SemiStdNF	1830	1831 Emily Barn	82 40 7; 41 230; 42 47; 43 471; 44

49.04	0.51 C19	SemiStdNF	1934	1934 Emily Barn	86 40 7; 41 224; 42 46; 43 474; 44
51.79	0.52 C20	SemiStdNF	2034	2034 Emily Barn	89 40 7; 41 225; 42 46; 43 467; 44
54.45	0.52 C21	SemiStdNF	2137	2138 Emily Barn	93 40 7; 41 220; 42 44; 43 463; 44
56.96	0.54 C22	SemiStdNF	2238	2239 Emily Barn	97 40 7; 41 210; 42 42; 43 452; 44
59.4	0.55 C23	SemiStdNF	2341	2342 Emily Barn	104 40 5; 41 203; 42 39; 43 440; 44
62.29	0.7 12-OH C18 Acid, TMS	SemiStdNF	2470	2470 Emily Barn	180 41 38; 42 5; 43 39; 44 8; 45 51;
61.42	0.69 16-OH C16 Acid, TMS	SemiStdNF	2429	2430 Emily Barn	229 40 2; 41 73; 42 12; 43 82; 44
39.91	0.8 2-Ketoglutaric acid, tri-TMS	SemiStdNF	1629	1629 Emily Barn	136 42 1; 43 16; 44 16; 45 115; 46
36.4	0.88 3-5-dimethoxyphenol	SemiStdNF	1525	1526 Emily Barn	124 41 2; 42 1; 43 13; 44 4; 45 31;
58.26	1.45 4, 4 dimethoxy-benzophenone	SemiStdNF	2293	2294 Emily Barn	81 50 10; 51 8; 52 1; 53 3; 61 2;
40.59	0.89 4-hydroxybenzoic acid	SemiStdNF	1651	1651 Emily Barn	129 42 1; 43 15; 44 5; 45 61; 46 4;
29.93	0.69 glyceric acid (?)	SemiStdNF	1352	1353 Emily Barn	96 40 1; 41 2; 42 3; 43 24; 44 9;
56.36	1.23 hexadecanamide	SemiStdNF	2212	2213 Emily Barn	78 40 4; 41 103; 42 20; 43 132; 44
50.61	0.53 hexadecanol, TMS	SemiStdNF	1989	1989 Emily Barn	98 40 1; 41 55; 42 9; 43 69; 44 5;
52.32	0.93 homosalate, TMS	SemiStdNF	2054	2055 Emily Barn	93 41 27; 42 1; 43 13; 44 1; 45 9;
32.75	0.73 hydroquinone	SemiStdNF	1420	1421 Emily Barn	129 42 1; 43 18; 44 5; 45 67; 46 3;
33.74	1.34 ionone	SemiStdNF	1449	1449 Emily Barn	106 40 10; 41 69; 42 6; 43 243; 44
33.51	0.64 SESQ3	SemiStdNF	1442	1443 Emily Barn	109 40 13; 41 523; 42 1; 43 193; 45
33.81	0.66 SESQ4	SemiStdNF	1451	1451 Emily Barn	122 40 74; 41 541; 42 27; 43 77; 44
38.69	0.78 isoeugenol	SemiStdNF	1591	1591 Emily Barn	125 40 1; 41 2; 42 1; 43 8; 44 4;
60.43	0.87 isopimaric acid	SemiStdNF	2385	2385 Emily Barn	190 40 2; 41 63; 42 4; 43 20; 44 5;
36.56	1.33 ketopinic acid	SemiStdNF	1530	1530 Emily Barn	176 40 7; 41 96; 42 12; 43 70; 44
42.95	0.85 levoglucosan	SemiStdNF	1726	1727 Emily Barn	149 41 4; 42 1; 43 17; 44 10; 45 88;
57.12	0.7 linoleic Acid, TMS	SemiStdNF	2245	2245 Emily Barn	213 40 5; 41 205; 42 18; 43 85; 44
81.22	1.04 lupeol	SemiStdNF	3483	3484 Emily Barn	281 40 6; 41 207; 42 12; 43 114; 44
28.33	1.1 maltol	SemiStdNF	1316	1316 Emily Barn	88 40 2; 41 1; 42 6; 43 23; 44 4;
42.3	0.86 mannosan	SemiStdNF	1706	1706 Emily Barn	148 41 4; 42 1; 43 14; 44 10; 45 89;
65.68	0.75 monopalmitin	SemiStdNF	2628	2628 Emily Barn	183 40 2; 41 104; 42 20; 43 183; 44
68.92	0.71 monostearin TMS	SemiStdNF	2788	2789 Emily Barn	194 40 2; 41 119; 42 21; 43 212; 44
28.41	0.51 nonanol	SemiStdNF	1318	1318 Emily Barn	88 40 2; 41 35; 42 6; 43 31; 44 7;
52.36	0.87 octadecanal (?)	SemiStdNF	2056	2056 Emily Barn	180 40 15; 41 519; 42 87; 43 606;
55.82	0.55 octadecanol	SemiStdNF	2191	2191 Emily Barn	103 40 1; 41 65; 42 10; 43 89; 44 5;
51.71	0.85 octadecanone	SemiStdNF	2031	2031 Emily Barn	97 40 5; 41 167; 42 33; 43 297; 44
57.27	0.7 oleic Acid, TMS	SemiStdNF	2251	2252 Emily Barn	205 40 5; 41 196; 42 30; 43 146; 44
37.05	0.97 p-anisic acid (4-methoxybenzoic ac	SemiStdNF	1544	1544 Emily Barn	95 42 1; 43 10; 44 3; 45 27; 46 2;

52.36	0.71 palmitoleic Acid	SemiStdNF	2056	2056 Emily Barn	192 40 6; 41 182; 42 32; 43 122; 44
42.95	0.95 pentadecanone	SemiStdNF	1726	1727 Emily Barn	115 40 6; 41 153; 42 33; 43 548; 44
72.31	1.46 perylene	SemiStdNF	2967	2968 Emily Barn	63 50 1; 51 1; 62 2; 63 1; 74 4;
42.57	1.11 phthalic acid	SemiStdNF	1714	1715 Emily Barn	94 43 8; 44 5; 45 49; 46 3; 47 4;
38.76	1.14 phthalimide	SemiStdNF	1593	1593 Emily Barn	70 42 1; 43 7; 44 2; 45 10; 47 1;
41.89	0.97 pinic acid 1	SemiStdNF	1692	1693 Emily Barn	146 40 1; 41 22; 42 3; 43 19; 44 9;
42.04	0.98 pinic acid 2	SemiStdNF	1697	1698 Emily Barn	149 40 1; 41 20; 42 3; 43 17; 44 9;
37.28	1.47 pinonic acid	SemiStdNF	1550	1551 Emily Barn	132 40 3; 41 39; 42 6; 43 123; 44
55.33	1.22 pyrene	SemiStdNF	2171	2172 Emily Barn	48 50 2; 51 1; 61 2; 62 3; 63 3;
29.36	0.68 pyrocatechol	SemiStdNF	1339	1340 Emily Barn	70 42 1; 43 11; 44 5; 45 73; 46 3;
26.73	1.1 quinoline	SemiStdNF	1278	1278 Emily Barn	35 49 5; 50 43; 51 62; 52 16; 53 1;
31.99	0.73 resorcinol	SemiStdNF	1399	1399 Emily Barn	145 42 1; 43 18; 44 4; 45 65; 46 4;
32.18	0.57 SESQ1	SemiStdNF	1404	1404 Emily Barn	112 40 26; 41 148; 42 18; 43 68; 44
57.65	1.05 retene	SemiStdNF	2267	2268 Emily Barn	104 41 1; 50 2; 51 2; 62 3; 63 6;
51.75	1.54 sinapinaldehyde	SemiStdNF	2032	2033 Emily Barn	192 41 1; 42 2; 43 18; 44 6; 45 68;
70.44	0.6 squalene	SemiStdNF	2868	2869 Emily Barn	99 40 5; 41 176; 42 9; 43 13; 44 1;
78.86	0.86 stigmasterol	SemiStdNF	3344	3345 Emily Barn	248 40 2; 41 97; 42 5; 43 84; 44 5;
42.95	1.33 syringaldehyde	SemiStdNF	1726	1727 Emily Barn	115 41 1; 42 1; 43 11; 44 3; 45 31;
48.78	0.88 syringic acid	SemiStdNF	1924	1925 Emily Barn	187 42 1; 43 17; 44 7; 45 70; 46 3;
32.67	0.82 syringol	SemiStdNF	1418	1419 Emily Barn	104 42 1; 43 10; 44 3; 45 29; 46 1;
36.25	0.53 threitol	SemiStdNF	1521	1521 Emily Barn	124 41 4; 42 1; 43 10; 44 8; 45 67;
36.52	0.54 erythreitol	SemiStdNF	1528	1529 Emily Barn	133 41 5; 42 1; 43 11; 44 8; 45 68;
30.39	2.11 triacetin	SemiStdNF	1362	1363 Emily Barn	31 40 1; 41 2; 42 16; 43 999; 44
36.82	0.93 tridecanal (?)	SemiStdNF	1537	1538 Emily Barn	96 40 22; 41 629; 42 132; 43 670;
44.89	0.88 vanillic acid	SemiStdNF	1789	1789 Emily Barn	155 42 1; 43 16; 44 6; 45 64; 46 3;
37.55	1.34 vanillin	SemiStdNF	1558	1558 Emily Barn	103 41 1; 42 1; 43 14; 44 5; 45 40;
25.09	2.07 verbenone (-)	SemiStdNF	1237	1237 Emily Barn	93 40 38; 41 169; 42 33; 43 45; 44
48.24	1.38 xanthone	SemiStdNF	1906	1906 Emily Barn	64 49 2; 50 23; 51 7; 52 1; 53 5;
39.33	1.07 SESQ9	SemiStdNF	1610	1611 Emily Barn	124 41 8; 42 2; 43 31; 44 13; 45
35.3	0.62 SESQ6	SemiStdNF	1493	1494 Emily Barn	118 40 35; 41 450; 42 39; 43 89; 44
34.5	0.62 SESQ5	SemiStdNF	1471	1471 Emily Barn	122 40 28; 41 484; 42 65; 43 140;