Modified ALCC, R code description.

Method · October 2020

DOI: 10.10171/CP16444

Adrián Alejandro Correndo
Kansas State University

Modified ALCC

Adrian A. Correndo

10/14/2020

This code was prepared as a tutorial for potential users of the Modified Arcsine-log Calibration Curve (ALCC) detailed in Correndo et al. (2017) with the purpose of studying relationships between relative yield and soil test values.

Instructions for users

1. Load your soil test value (STV) and relative yield (RY) data as vectors of a data.frame.
2. Specify into the function -ALCC()- which vectors correspond to RY and STV.
3. Specify your relative yield target (e.g. 90%) and desired confidence level (e.g. 0.95 for 1 - alpha(0.05)). Used for the estimation of critical soil test value (CSTV) and lower and upper confidence limits.
4. Run.
5. Check results and adjust plot as desired.
6. Please, refer any question to Adrian Correndo, correndo@agro.uba.ar - correndo@ksu.edu.

Note: RY should be expressed relative to a maximum in order to obtain values bounded at %100. Otherwise, arcsine transformation doesn't work.

References

Correndo, A.A., F. Salvagiotti, F.O. García, y F.H. Gutiérrez-Boem. 2017. A modification of the arcsine-log calibration curve for analysing soil test value - relative yield relationships. Crop & Pasture Science 68 (3): 297-304, doi: 10.10171/CP16444

Libraries

```
# Install if needed
# install.packages("tidyverse")
# tidyverse contains "dplyr" for data wrangling and "ggplot2" for creating plots
library(tidyverse)
```

Data

```
# Example 1 dataset
data_1 = data.frame("RY" = c(65,80,85,88,90,94,93,96,97,95,98,100,99,99,100),
                    "STV" = c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15))

# Example 2 dataframe. Imported from csv file
# It can be easily replaced with your own csv file
data_2 = read.csv(file = "data_test.csv")
```

ALCC function, definition

```
ALCC <- function(RY, STV, target, confidence){
  n = length(RY) # Sample size
  df = n - 2 # Degrees of freedom
  prob = 1-((1-confidence)/1/2) # Probability for t-dist
  tvalue = qt(p=prob, df = df) # Student-t value
  arcsine = function(RY)asin(sqrt(RY/100))
  arc_RY = arcsine(RY) - arcsine(target) # RY transformation (centered to target)
  ln_STV = log(STV) # STV natural log transformation
  r = cor(ln_STV, arc_RY) # Pearson correlation (r)
  p_value = cor.test(ln_STV,arc_RY)$p.value # p-value of r
  slope = sd(ln_STV)/sd(arc_RY) # SMA slope for ln_STV ~ arc_RY
  intercept = mean(ln_STV) - (mean(arc_RY)*slope) # Intercept
  Line = intercept + slope * arc_RY# Fitted ln_STV for observed RY
  CSTV = exp(intercept) # Critical STV for specified RY-target and confidence (1-alpha)
  MSE = sum((Line-ln_STV)^2)/df # Mean Square Error of ln_STV
  SSx = sum((mean(arc_RY)-arc_RY)^2) # Sum of Squares of arc_RY
  SE_int = sqrt(MSE*((1/n)+ ((mean(arc_RY)^2)/SSx))) # Standard Error intercept
  CSTV_lower = exp(intercept - (tvalue * SE_int)) # Lower limit of CSTV
  CSTV_upper = exp(intercept + (tvalue * SE_int)) # Upper limit of CSTV
  new_RY = seq(min(RY),100, by=0.2) # New RY vector up to %100 to fit curve
  new_arc_RY = arcsine(new_RY) - arcsine(target) # Transforming new_RY vector
  fitted_Line = intercept + slope * new_arc_RY# Fitted ln_STV for new_RY
  fitted_STV = exp(fitted_Line) # Fitted ln_STV for new_RY
  # Outcome
  results <- list("Pearson r" = r,
                  "p_value" = p_value,
                  "CSTV" = CSTV,
                  "Lower_limit" = CSTV_lower, "Upper_limit" = CSTV_upper,
                  "Curve" = list("Fitted_RY" = new_RY, "Fitted_STV" = fitted_STV))

  return(results)
}
```

Fit example 1

```
# Fit the ALCC model
# RY target = 90%, confidence level = 0.95, replace with your desired values
fit_example_1 = ALCC(RY = data_1$RY,STV = data_1$STV, target=90,confidence = 0.95)

fit_example_1
```

```
## $`Pearson r`
## [1] 0.9682908
##
## $p_value
## [1] 3.296044e-09
##
## $CSTV
## [1] 4.478476
##
## $Lower_limit
## [1] 3.947041
```

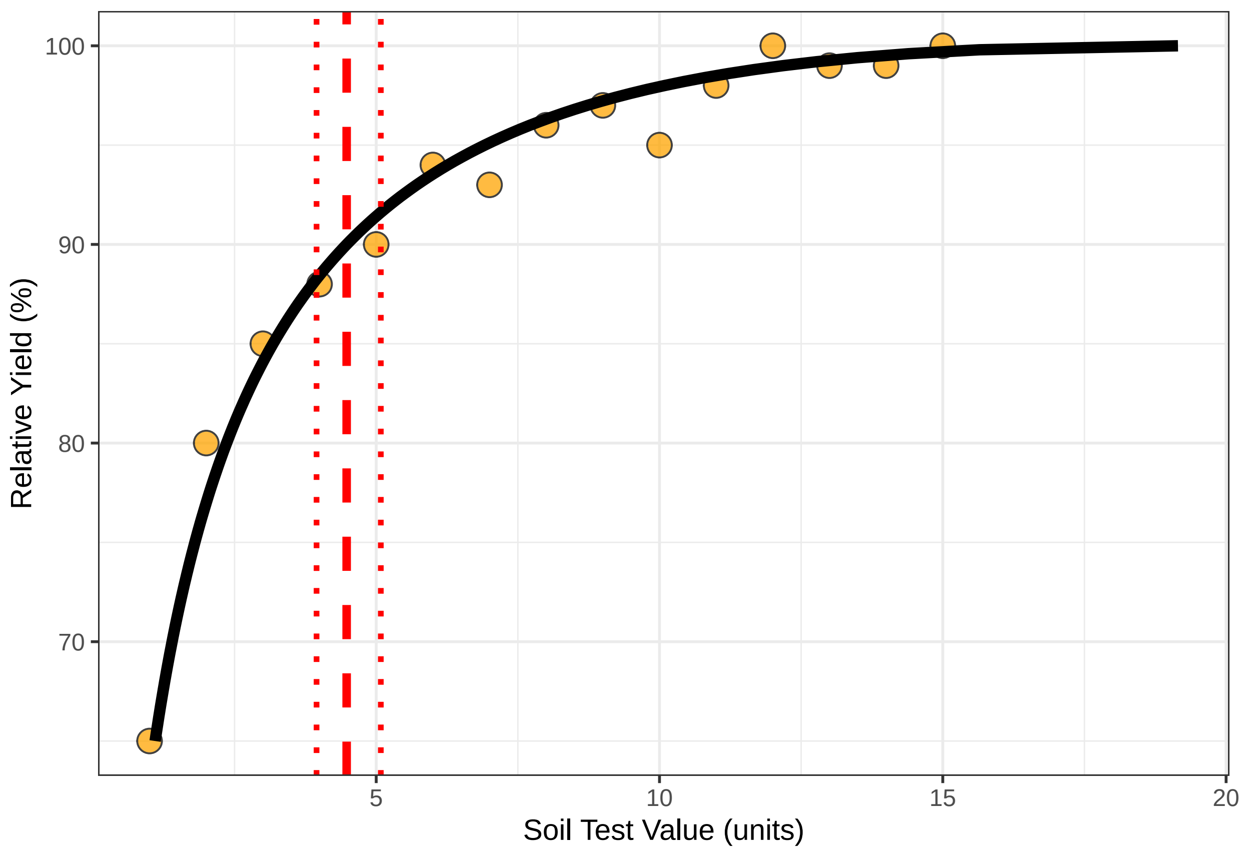
```
##
## $Upper_limit
## [1] 5.081463
##
## $Curve
## $Curve$Fitted_RY
##   [1]  65.0  65.2  65.4  65.6  65.8  66.0  66.2  66.4  66.6  66.8  67.0  67.2
##  [13]  67.4  67.6  67.8  68.0  68.2  68.4  68.6  68.8  69.0  69.2  69.4  69.6
##  [25]  69.8  70.0  70.2  70.4  70.6  70.8  71.0  71.2  71.4  71.6  71.8  72.0
##  [37]  72.2  72.4  72.6  72.8  73.0  73.2  73.4  73.6  73.8  74.0  74.2  74.4
##  [49]  74.6  74.8  75.0  75.2  75.4  75.6  75.8  76.0  76.2  76.4  76.6  76.8
##  [61]  77.0  77.2  77.4  77.6  77.8  78.0  78.2  78.4  78.6  78.8  79.0  79.2
##  [73]  79.4  79.6  79.8  80.0  80.2  80.4  80.6  80.8  81.0  81.2  81.4  81.6
##  [85]  81.8  82.0  82.2  82.4  82.6  82.8  83.0  83.2  83.4  83.6  83.8  84.0
##  [97]  84.2  84.4  84.6  84.8  85.0  85.2  85.4  85.6  85.8  86.0  86.2  86.4
## [109]  86.6  86.8  87.0  87.2  87.4  87.6  87.8  88.0  88.2  88.4  88.6  88.8
## [121]  89.0  89.2  89.4  89.6  89.8  90.0  90.2  90.4  90.6  90.8  91.0  91.2
## [133]  91.4  91.6  91.8  92.0  92.2  92.4  92.6  92.8  93.0  93.2  93.4  93.6
## [145]  93.8  94.0  94.2  94.4  94.6  94.8  95.0  95.2  95.4  95.6  95.8  96.0
## [157]  96.2  96.4  96.6  96.8  97.0  97.2  97.4  97.6  97.8  98.0  98.2  98.4
## [169]  98.6  98.8  99.0  99.2  99.4  99.6  99.8 100.0
##
## $Curve$Fitted_STV
##   [1]  1.097927  1.108379  1.118944  1.129625  1.140423  1.151339  1.162376
##   [8]  1.173535  1.184817  1.196225  1.207760  1.219425  1.231221  1.243150
##  [15]  1.255214  1.267415  1.279755  1.292236  1.304860  1.317630  1.330548
##  [22]  1.343616  1.356836  1.370210  1.383742  1.397433  1.411286  1.425303
##  [29]  1.439488  1.453842  1.468369  1.483071  1.497951  1.513012  1.528257
##  [36]  1.543689  1.559311  1.575127  1.591138  1.607350  1.623765  1.640386
##  [43]  1.657217  1.674262  1.691525  1.709008  1.726717  1.744655  1.762826
##  [50]  1.781234  1.799883  1.818779  1.837924  1.857325  1.876985  1.896910
##  [57]  1.917103  1.937572  1.958319  1.979352  2.000675  2.022293  2.044214
##  [64]  2.066441  2.088983  2.111844  2.135031  2.158551  2.182410  2.206616
##  [71]  2.231175  2.256095  2.281384  2.307048  2.333097  2.359537  2.386379
##  [78]  2.413629  2.441298  2.469395  2.497929  2.526910  2.556347  2.586252
##  [85]  2.616636  2.647508  2.678882  2.710768  2.743179  2.776127  2.809626
##  [92]  2.843690  2.878331  2.913565  2.949407  2.985872  3.022976  3.060736
##  [99]  3.099169  3.138294  3.178128  3.218691  3.260004  3.302086  3.344960
## [106]  3.388649  3.433176  3.478564  3.524841  3.572032  3.620165  3.669269
## [113]  3.719374  3.770512  3.822716  3.876020  3.930460  3.986075  4.042904
## [120]  4.100988  4.160372  4.221100  4.283223  4.346790  4.411856  4.478476
## [127]  4.546710  4.616622  4.688278  4.761749  4.837110  4.914440  4.993823
## [134]  5.075351  5.159117  5.245225  5.333784  5.424910  5.518728  5.615374
## [141]  5.714992  5.817737  5.923779  6.033299  6.146495  6.263582  6.384794
## [148]  6.510388  6.640644  6.775870  6.916408  7.062634  7.214969  7.373879
## [155]  7.539891  7.713596  7.895664  8.086860  8.288063  8.500287  8.724716
## [162]  8.962742  9.216022  9.486552  9.776766 10.089688 10.429142 10.800084
## [169] 11.209119 11.665380 12.182103 12.779738 13.492946 14.389790 15.647308
## [176] 19.150544
```

Plot example 1

```
# Extracting curve data as a data.frame to plot
curve_example1 = data.frame("Fitted_STV" = fit_example_1[["Curve"]][["Fitted_STV"]],
```

```r
                              "Fitted_RY" = fit_example_1[["Curve"]][["Fitted_RY"]])

# Plot
data_1 %>% ggplot()+
  # Points
  geom_point(aes(x = STV, y = RY), fill = "orange", shape = 21, size = 4, alpha = 0.75)+
  # Fitted ALCC
  geom_line(data = curve_example1, aes(x= Fitted_STV, y = Fitted_RY), size = 2)+
  # Critical value
  geom_vline(xintercept = fit_example_1$CSTV, col = "red", size = 1.5, linetype = "dashed")+
  # Confidence limits
  geom_vline(xintercept = fit_example_1$Lower_limit, col = "red", size = 1, linetype = "dotted")+
  geom_vline(xintercept = fit_example_1$Upper_limit, col = "red", size = 1, linetype = "dotted")+
  # Axis titles
  labs(x = "Soil Test Value (units)", y = "Relative Yield (%)")+
  theme_bw()+
  theme()
```



Fit example 2

```r
# Fit the ALCC model
# target = 90, confidence = 0.95, replace with your desired values
fit_example_2 = ALCC(RY = data_2$RY,STV = data_2$STV, target=90,confidence = 0.95)

fit_example_2
```

```
## $`Pearson r`
```

```
## [1] 0.7164928
##
## $p_value
## [1] 7.314913e-23
##
## $CSTV
## [1] 23.25457
##
## $Lower_limit
## [1] 21.57156
##
## $Upper_limit
## [1] 25.06888
##
## $Curve
## $Curve$Fitted_RY
##   [1]  12.0  12.2  12.4  12.6  12.8  13.0  13.2  13.4  13.6  13.8  14.0  14.2
##  [13]  14.4  14.6  14.8  15.0  15.2  15.4  15.6  15.8  16.0  16.2  16.4  16.6
##  [25]  16.8  17.0  17.2  17.4  17.6  17.8  18.0  18.2  18.4  18.6  18.8  19.0
##  [37]  19.2  19.4  19.6  19.8  20.0  20.2  20.4  20.6  20.8  21.0  21.2  21.4
##  [49]  21.6  21.8  22.0  22.2  22.4  22.6  22.8  23.0  23.2  23.4  23.6  23.8
##  [61]  24.0  24.2  24.4  24.6  24.8  25.0  25.2  25.4  25.6  25.8  26.0  26.2
##  [73]  26.4  26.6  26.8  27.0  27.2  27.4  27.6  27.8  28.0  28.2  28.4  28.6
##  [85]  28.8  29.0  29.2  29.4  29.6  29.8  30.0  30.2  30.4  30.6  30.8  31.0
##  [97]  31.2  31.4  31.6  31.8  32.0  32.2  32.4  32.6  32.8  33.0  33.2  33.4
## [109]  33.6  33.8  34.0  34.2  34.4  34.6  34.8  35.0  35.2  35.4  35.6  35.8
## [121]  36.0  36.2  36.4  36.6  36.8  37.0  37.2  37.4  37.6  37.8  38.0  38.2
## [133]  38.4  38.6  38.8  39.0  39.2  39.4  39.6  39.8  40.0  40.2  40.4  40.6
## [145]  40.8  41.0  41.2  41.4  41.6  41.8  42.0  42.2  42.4  42.6  42.8  43.0
## [157]  43.2  43.4  43.6  43.8  44.0  44.2  44.4  44.6  44.8  45.0  45.2  45.4
## [169]  45.6  45.8  46.0  46.2  46.4  46.6  46.8  47.0  47.2  47.4  47.6  47.8
## [181]  48.0  48.2  48.4  48.6  48.8  49.0  49.2  49.4  49.6  49.8  50.0  50.2
## [193]  50.4  50.6  50.8  51.0  51.2  51.4  51.6  51.8  52.0  52.2  52.4  52.6
## [205]  52.8  53.0  53.2  53.4  53.6  53.8  54.0  54.2  54.4  54.6  54.8  55.0
## [217]  55.2  55.4  55.6  55.8  56.0  56.2  56.4  56.6  56.8  57.0  57.2  57.4
## [229]  57.6  57.8  58.0  58.2  58.4  58.6  58.8  59.0  59.2  59.4  59.6  59.8
## [241]  60.0  60.2  60.4  60.6  60.8  61.0  61.2  61.4  61.6  61.8  62.0  62.2
## [253]  62.4  62.6  62.8  63.0  63.2  63.4  63.6  63.8  64.0  64.2  64.4  64.6
## [265]  64.8  65.0  65.2  65.4  65.6  65.8  66.0  66.2  66.4  66.6  66.8  67.0
## [277]  67.2  67.4  67.6  67.8  68.0  68.2  68.4  68.6  68.8  69.0  69.2  69.4
## [289]  69.6  69.8  70.0  70.2  70.4  70.6  70.8  71.0  71.2  71.4  71.6  71.8
## [301]  72.0  72.2  72.4  72.6  72.8  73.0  73.2  73.4  73.6  73.8  74.0  74.2
## [313]  74.4  74.6  74.8  75.0  75.2  75.4  75.6  75.8  76.0  76.2  76.4  76.6
## [325]  76.8  77.0  77.2  77.4  77.6  77.8  78.0  78.2  78.4  78.6  78.8  79.0
## [337]  79.2  79.4  79.6  79.8  80.0  80.2  80.4  80.6  80.8  81.0  81.2  81.4
## [349]  81.6  81.8  82.0  82.2  82.4  82.6  82.8  83.0  83.2  83.4  83.6  83.8
## [361]  84.0  84.2  84.4  84.6  84.8  85.0  85.2  85.4  85.6  85.8  86.0  86.2
## [373]  86.4  86.6  86.8  87.0  87.2  87.4  87.6  87.8  88.0  88.2  88.4  88.6
## [385]  88.8  89.0  89.2  89.4  89.6  89.8  90.0  90.2  90.4  90.6  90.8  91.0
## [397]  91.2  91.4  91.6  91.8  92.0  92.2  92.4  92.6  92.8  93.0  93.2  93.4
## [409]  93.6  93.8  94.0  94.2  94.4  94.6  94.8  95.0  95.2  95.4  95.6  95.8
## [421]  96.0  96.2  96.4  96.6  96.8  97.0  97.2  97.4  97.6  97.8  98.0  98.2
## [433]  98.4  98.6  98.8  99.0  99.2  99.4  99.6  99.8 100.0
##
```

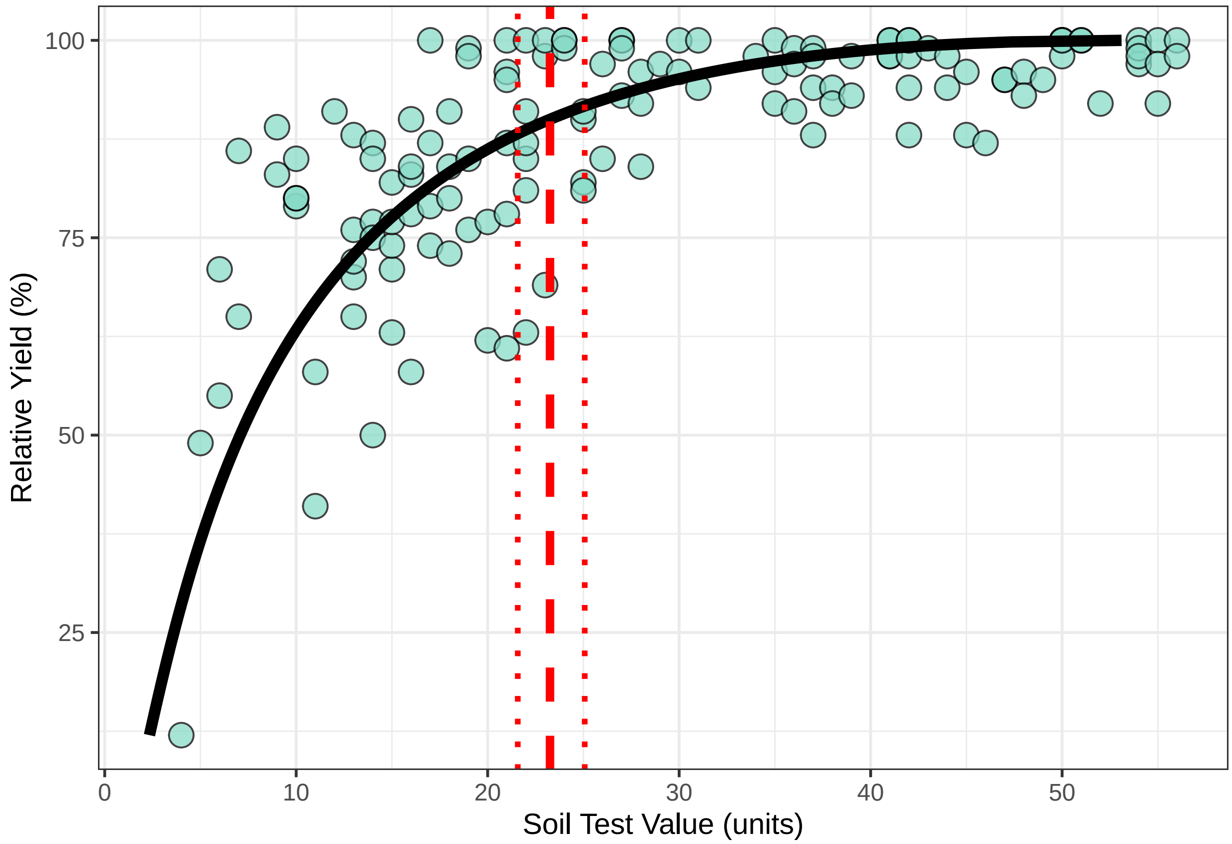
```
## $Curve$Fitted_STV
##   [1]  2.336883  2.355345  2.373822  2.392314  2.410824  2.429353  2.447902
##   [8]  2.466472  2.485066  2.503683  2.522326  2.540994  2.559691  2.578416
##  [15]  2.597171  2.615957  2.634775  2.653625  2.672510  2.691430  2.710385
##  [22]  2.729377  2.748407  2.767476  2.786584  2.805733  2.824922  2.844155
##  [29]  2.863430  2.882748  2.902112  2.921521  2.940976  2.960478  2.980028
##  [36]  2.999627  3.019275  3.038973  3.058722  3.078522  3.098374  3.118280
##  [43]  3.138239  3.158253  3.178322  3.198446  3.218628  3.238866  3.259162
##  [50]  3.279517  3.299931  3.320405  3.340940  3.361536  3.382194  3.402915
##  [57]  3.423698  3.444546  3.465459  3.486436  3.507480  3.528590  3.549767
##  [64]  3.571012  3.592326  3.613709  3.635161  3.656685  3.678279  3.699945
##  [71]  3.721683  3.743495  3.765380  3.787340  3.809375  3.831485  3.853672
##  [78]  3.875936  3.898277  3.920697  3.943195  3.965774  3.988432  4.011172
##  [85]  4.033993  4.056896  4.079883  4.102953  4.126107  4.149347  4.172672
##  [92]  4.196083  4.219582  4.243168  4.266843  4.290606  4.314460  4.338404
##  [99]  4.362439  4.386567  4.410787  4.435100  4.459507  4.484009  4.508607
## [106]  4.533301  4.558092  4.582980  4.607967  4.633054  4.658240  4.683527
## [113]  4.708916  4.734406  4.760000  4.785698  4.811500  4.837408  4.863422
## [120]  4.889542  4.915771  4.942108  4.968555  4.995111  5.021779  5.048559
## [127]  5.075451  5.102457  5.129578  5.156813  5.184165  5.211634  5.239220
## [134]  5.266926  5.294751  5.322696  5.350763  5.378953  5.407266  5.435703
## [141]  5.464265  5.492953  5.521768  5.550712  5.579784  5.608986  5.638319
## [148]  5.667784  5.697382  5.727114  5.756981  5.786984  5.817124  5.847402
## [155]  5.877819  5.908377  5.939075  5.969916  6.000901  6.032030  6.063304
## [162]  6.094726  6.126295  6.158014  6.189883  6.221903  6.254076  6.286402
## [169]  6.318884  6.351522  6.384317  6.417271  6.450385  6.483660  6.517098
## [176]  6.550699  6.584466  6.618399  6.652499  6.686769  6.721209  6.755821
## [183]  6.790607  6.825566  6.860702  6.896016  6.931508  6.967181  7.003035
## [190]  7.039073  7.075296  7.111705  7.148302  7.185089  7.222066  7.259237
## [197]  7.296601  7.334162  7.371920  7.409878  7.448036  7.486397  7.524963
## [204]  7.563734  7.602714  7.641903  7.681304  7.720918  7.760747  7.800794
## [211]  7.841059  7.881546  7.922255  7.963189  8.004350  8.045739  8.087360
## [218]  8.129213  8.171302  8.213628  8.256193  8.298999  8.342049  8.385345
## [225]  8.428890  8.472684  8.516732  8.561034  8.605594  8.650414  8.695497
## [232]  8.740844  8.786458  8.832342  8.878499  8.924930  8.971640  9.018629
## [239]  9.065902  9.113461  9.161308  9.209446  9.257879  9.306609  9.355639
## [246]  9.404972  9.454611  9.504560  9.554821  9.605397  9.656292  9.707509
## [253]  9.759051  9.810922  9.863125  9.915663  9.968541 10.021761 10.075327
## [260] 10.129242 10.183511 10.238138 10.293126 10.348478 10.404200 10.460295
## [267] 10.516766 10.573619 10.630857 10.688485 10.746507 10.804928 10.863751
## [274] 10.922982 10.982625 11.042685 11.103166 11.164074 11.225413 11.287189
## [281] 11.349407 11.412071 11.475188 11.538762 11.602799 11.667304 11.732284
## [288] 11.797744 11.863690 11.930128 11.997064 12.064504 12.132455 12.200922
## [295] 12.269914 12.339435 12.409494 12.480096 12.551250 12.622962 12.695240
## [302] 12.768091 12.841522 12.915543 12.990159 13.065381 13.141215 13.217670
## [309] 13.294756 13.372480 13.450851 13.529880 13.609574 13.689945 13.771001
## [316] 13.852752 13.935209 14.018381 14.102280 14.186917 14.272302 14.358448
## [323] 14.445364 14.533064 14.621560 14.710863 14.800987 14.891945 14.983749
## [330] 15.076415 15.169955 15.264384 15.359717 15.455968 15.553154 15.651289
## [337] 15.750391 15.850475 15.951559 16.053661 16.156798 16.260989 16.366253
## [344] 16.472609 16.580077 16.688679 16.798434 16.909365 17.021495 17.134845
## [351] 17.249441 17.365306 17.482466 17.600946 17.720773 17.841975 17.964580
## [358] 18.088618 18.214118 18.341112 18.469631 18.599711 18.731384 18.864686
## [365] 18.999655 19.136328 19.274745 19.414946 19.556974 19.700873 19.846689
```

```
## [372] 19.994468 20.144260 20.296115 20.450087 20.606231 20.764604 20.925266
## [379] 21.088278 21.253707 21.421618 21.592084 21.765177 21.940974 22.119557
## [386] 22.301010 22.485420 22.672882 22.863491 23.057350 23.254567 23.455255
## [393] 23.659532 23.867524 24.079364 24.295192 24.515156 24.739412 24.968128
## [400] 25.201479 25.439655 25.682855 25.931293 26.185196 26.444810 26.710396
## [407] 26.982235 27.260631 27.545908 27.838422 28.138553 28.446718 28.763369
## [414] 29.089000 29.424154 29.769426 30.125473 30.493024 30.872890 31.265974
## [421] 31.673293 32.095997 32.535388 32.992958 33.470424 33.969784 34.493376
## [428] 35.043972 35.624899 36.240199 36.894871 37.595217 38.349366 39.168111
## [435] 40.066313 41.065417 42.198418 43.520998 45.142004 47.343203 53.102988
```

Plotting example 2

```r
# Extracting curve data as a data.frame to plot
curve_example2 = data.frame("Fitted_STV" = fit_example_2[["Curve"]][["Fitted_STV"]],
                            "Fitted_RY" = fit_example_2[["Curve"]][["Fitted_RY"]])

# Plot
data_2 %>% ggplot()+
  # Points
  geom_point(aes(x = STV, y = RY), fill = "#88dbc8", shape = 21, size = 4, alpha = 0.75)+
  # Fitted ALCC
  geom_line(data = curve_example2, aes(x= Fitted_STV, y = Fitted_RY), size = 2)+
  # Critical value
  geom_vline(xintercept = fit_example_2$CSTV, col = "red", size = 1.5, linetype = "dashed")+
  # Confidence limits
  geom_vline(xintercept = fit_example_2$Lower_limit, col = "red", size = 1, linetype = "dotted")+
  geom_vline(xintercept = fit_example_2$Upper_limit, col = "red", size = 1, linetype = "dotted")+
  # Axis titles
  labs(x = "Soil Test Value (units)", y = "Relative Yield (%)")+
  theme_bw()+
  theme()
```


Relative Yield (%)
100
75
50
25
0
10
20
30
40
50
Soil Test Value (units)