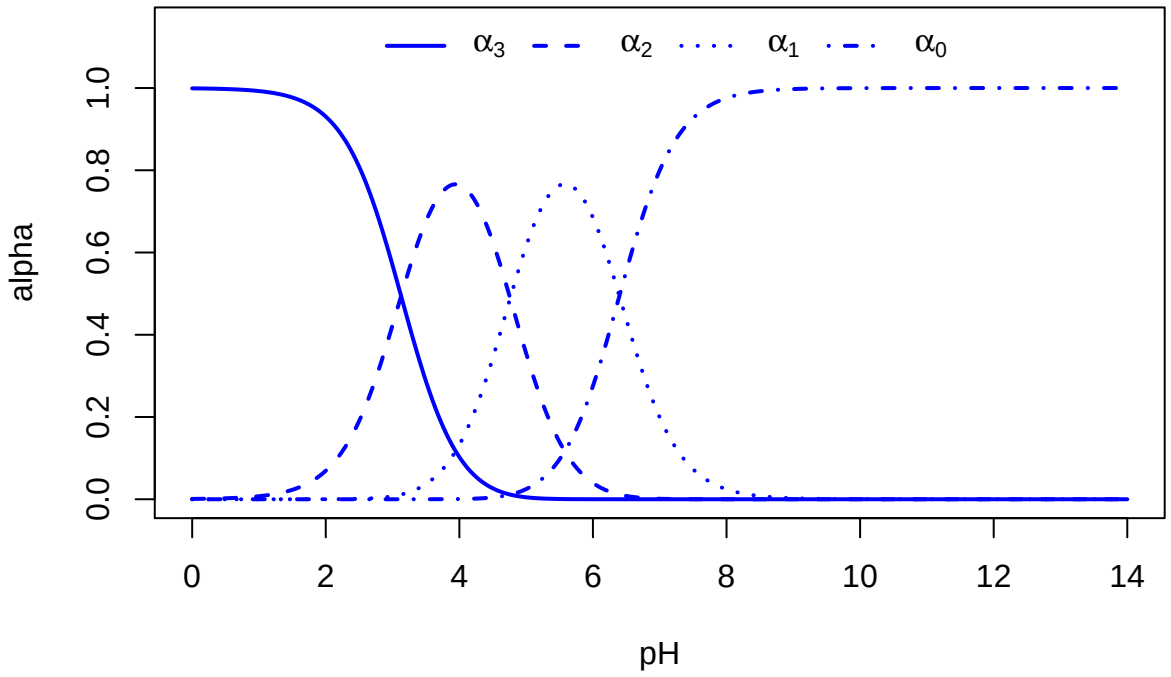
alpha_pH.R

davidharvey

Mon Feb 15 10:27:30 2016

```
# function to plot alpha diagrams for weak acids
# pka_list: list of pka values, ideally in order of most acidic to
#           least acidic, but sorted later; default values
#           are for citric acid
# pH_range: min to max; defaults to 0 to 14
# pH_step: increments along pH axis for calculating alpha values;
#          defaults to 0.01
# plot_alpha: logical value; if TRUE (the default), plots alpha diagram
# species: option to enter name of weak acid for main title of plot;
#          defaults to NULL, which supresses main title
# labels: option to enter vector of labels for legend; defaults to
#         NULL, which uses a default legend

alpha_pH = function(pka_list = c(3.128, 4.761, 6.396),
                pH_range = c(0,14), pH_step = 0.01, plot_alpha = TRUE,
                species = NULL, labels = NULL){

  # initial set-up
  # ensures that the pKa values are in order; adds an initial Ka of 1
  # to associate with the denominator's H+ term; creates counter, n,
  # for the number of alpha values; and creates vector of pH values and
  # concentrations of H+

  pkas = sort(pka_list)
  pkas = c(0, pkas)
  n = length(pkas)
  ph = seq(pH_range[1], pH_range[2], pH_step)
  h = 10^-ph

  # calculate denominator
  # uses cumprod() to create vector with the general structure of 1,
  # ka1, ka2*ka2, etc; sets up vector to store the denominator's
  # value for each pH; and calculates and stores the denomiator's
  # values for each pH

  ka = cumprod(10^-pkas)
  denom = rep(0, length(h))
  for (i in 1:length(h)) {
    for (j in 1:n) {
      denom[i] = denom[i] + h[i]^(n-j) * ka[j]
    }
  }

  # calculate alpha
  # create data frame to store alpha values and add pH values to first
  # column; add names to data frame's columns; and calculates and store
  # alpha values for each pH
```

```
  alpha.df = data.frame(matrix(0, ncol = n + 1, nrow = length(h)))
  alpha.df[ , 1] = ph
  colnames(alpha.df)[1] = "pH"
  for (j in 1:n) {
    colnames(alpha.df)[j + 1] = paste("alpha", n - j, sep = "")
    for (i in 1:length(h)) {
      alpha.df[i, j + 1] = h[i]^(n-j) * ka[j]/denom[i]
    }
  }

  # create alpha plot
  # if requested, returns a plot; adds a main title if name of species
  # is provided; adds a legend using default labels or provided labels

  if (plot_alpha == TRUE) {
  matplot(alpha.df$pH, as.matrix(alpha.df[ , 2:(n + 1)]),
          type = "l", lty = c(1:6), lwd = 2,
          col = c(rep("blue", 6), rep("red", 6)),
          xlim = c(pH_range[1], pH_range[2]), ylim = c(0,1.15),
          xlab = "pH", ylab = "alpha")
    if (is.null(species) == FALSE){
      title(main = paste("alpha plot for ", species, sep = ""))
    }
    if (is.null(labels) == TRUE){
      labels = rep("", n)
      for (i in 1:n) {
        num.protons = n - i
        labels[i] = eval(substitute(expression(alpha[I]),
                                    list(I = num.protons)))
      }
      legend(x = "top",
             legend = labels, lty = c(1:6),
             col = c(rep("blue", 6), rep("red", 6)), lwd = 2,
             bty = "n", horiz = TRUE)
    } else {
      legend(x = "top",
             legend = labels, lty = c(1:6),
             col = c(rep("blue", 6), rep("red", 6)), lwd = 2,
             bty = "n", horiz = TRUE, cex = 1 - 0.2*n/2)
    }
  }

  # returns data: data frame containing pH values and alpha values

  invisible(alpha.df)
}

# sample code to show output

x = alpha_pH(species = "citric acid")
```




```
head(x)
```

```
##     pH    alpha3       alpha2       alpha1       alpha0
## 1 0.00 0.9992558 0.0007441778 1.290258e-08 5.184140e-15
## 2 0.01 0.9992385 0.0007614987 1.351043e-08 5.554809e-15
## 3 0.02 0.9992208 0.0007792224 1.414691e-08 5.951980e-15
## 4 0.03 0.9992026 0.0007973584 1.481336e-08 6.377546e-15
## 5 0.04 0.9991841 0.0008159161 1.551120e-08 6.833536e-15
## 6 0.05 0.9991651 0.0008349054 1.624192e-08 7.322127e-15
```