

Repeat evaluation of ALFAM2 parameter set 1

Sasha D. Hafner

28 January 2024

Overview

This document for Armand Favrot presents a repeat of the evaluation of the ALFAM2 model with parameter set 1 originally presented in the 2019 paper introducing the model.

Packages and functions

```
library(data.table)
library(ALFAM2)
source('dfsumm.R')
source('model_stats.R')
```

See ALFAM2 version.

```
packageVersion('ALFAM2')
```

```
## [1] '3.58'
```

The problem you found with duplicated columns should not be present in this version, which is available from the dev branch now. But note that the function does not add dummy variable columns for “reference levels”, and both cattle manure and trailing hose are reference levels, so that behavior was not an error.

Input data

Get pmid from paper.

```
pmidcal <- fread('../data/S1_plot_codes_calibration.csv')
pmideval <- fread('../data/S2_plot_codes_evaluation.csv')
```

To associate these with measurements below, combine them in a data table here.

```
pmidcal[, datasub := 'cal']
pmideval[, datasub := 'eval']
pmid <- rbind(pmidcal, pmideval)
```

Load interval-level data. Note this is version 1.0 of ALFAM2 “database”. See <https://github.com/sashahafner/ALFAM2-data/tree/cfa0055e44907578bf40e0eac389e020f93dd6b1/data%20-%20ALFAM2%20output> for this version. Unfortunately I was just getting started with GitHub releases around this time in 2018, so there is no release for this version. Since then the situation has improved. Now you can always find the database version in `data-output/...` ([here](#)), and new versions get releases ([here](#)).

```
idat <- fread('../data/ALFAM2_interval.csv')
```

Fix changed variable name.

```
idat[, app.mthd := app.method]
```

Merge in pmid and subset keys.

```
dim(idat)
```

```
## [1] 30907 110
```

```
idat <- merge(idat, pmid, by = 'pmid')
dim(idat)
```

```
## [1] 12193 111
```

Trim to 78 hours.

```
idat <- idat[ct > 0 & ct < 78, ]
```

Check values.

```
dfsumm(idat[datasub == 'eval',.(pmid, app.mthd, app.rate, man.dm, man.source, air.temp, wind.2m,
man.ph, rain.rate, incorp)])
```

```
##
## 423 rows and 10 columns
## 413 unique rows
##
##          pmid  app.mthd app.rate  man.dm man.source air.temp
## Class      integer character  numeric numeric  character  numeric
## Minimum      195         bc      6.6    1.52         cat     0.3
## Maximum     1900         ts     58.2    10.7         pig     34.1
## Mean        1300        <NA>     29.3    6.07        <NA>     15.1
## Unique (excl. NA)  48         4      45     45         2     207
## Missing values    0         0      0      0         0      1
## Sorted        TRUE        FALSE    FALSE    FALSE    FALSE    FALSE
##
##          wind.2m  man.ph rain.rate  incorp
## Class      numeric numeric  numeric character
## Minimum      0.22    6.8      0      none
## Maximum     16.8    8.2      3.08    none
## Mean         3.08    7.38    0.0744    <NA>
## Unique (excl. NA)  267    22      44      1
## Missing values    2    39     134     0
## Sorted        FALSE    FALSE    FALSE    TRUE
##
```

```
dfsumm(idat[datasub == 'cal',.(pmid, app.mthd, app.rate, man.dm, man.source, air.temp, wind.2m,
man.ph, rain.rate, incorp)])
```

```
##
## 5501 rows and 10 columns
## 5386 unique rows
##
##          pmid  app.mthd app.rate  man.dm man.source air.temp
## Class      integer character  numeric numeric  character  numeric
## Minimum      182         bc      7.9      1         cat     -1.9
## Maximum     1900         ts     133    13.6         pig     35.2
## Mean        1350        <NA>     48.2    5.93        <NA>     13.1
## Unique (excl. NA)  490         4     202    208         2     958
## Missing values    0         0      0      0         0      3
## Sorted        TRUE        FALSE    FALSE    FALSE    FALSE    FALSE
```

```
##
##           wind.2m  man.ph  rain.rate    incorp
## Class      numeric numeric    numeric character
## Minimum      0.0513     6.4         0      deep
## Maximum      16.8      8.5         7.1    shallow
## Mean         3.09      7.32      0.0511    <NA>
## Unique (excl. NA) 1782     65        314      3
## Missing values    13     1053      1204      0
## Sorted        FALSE    FALSE      FALSE     FALSE
##
```

Fill in missing wind speed, air temperature, rainfall values. This is a thorny problem. Dropping values means leaving out emission intervals. Here they are set to the average by plot, or, for rain, zero.

```
idat[, wind.2m.ave := mean(wind.2m, na.omit = TRUE), by = pmid]
idat[, air.temp.ave := mean(air.temp, na.omit = TRUE), by = pmid]

idat[is.na(wind.2m), wind.2m := wind.2m.ave]
idat[is.na(air.temp), air.temp := air.temp.ave]

idat[is.na(rain.rate), rain.rate := 0]
idat[is.na(rain.cum), rain.cum := 0]
```

Check values.

```
dfsumm(idat[datasub == 'eval',.(pmid, app.mthd, app.rate, man.dm, man.source, air.temp, wind.2m,
                                man.ph, rain.rate, incorp)])
```

```
##
## 423 rows and 10 columns
## 413 unique rows
##           pmid  app.mthd  app.rate  man.dm  man.source  air.temp
## Class      integer character    numeric numeric    character    numeric
## Minimum      195         bc        6.6    1.52         cat        0.3
## Maximum     1900         ts       58.2    10.7         pig       34.1
## Mean        1300        <NA>       29.3    6.07        <NA>       15.1
## Unique (excl. NA)  48         4        45    45          2       208
## Missing values     0         0         0     0          0         0
## Sorted       TRUE        FALSE      FALSE    FALSE      FALSE      FALSE
##
```

```
##           wind.2m  man.ph  rain.rate    incorp
## Class      numeric numeric    numeric character
## Minimum      0.22     6.8         0      none
## Maximum      16.8     8.2        3.08    none
## Mean         3.09     7.38      0.0508    <NA>
## Unique (excl. NA)  269     22        44      1
## Missing values     0     39         0      0
## Sorted        FALSE    FALSE      FALSE     TRUE
##
```

```
dfsumm(idat[datasub == 'cal',.(pmid, app.mthd, app.rate, man.dm, man.source, air.temp, wind.2m,
                                man.ph, rain.rate, incorp)])
```

```
##
## 5501 rows and 10 columns
## 5386 unique rows
##           pmid  app.mthd  app.rate  man.dm  man.source  air.temp
```

```
## Class          integer character numeric numeric character numeric
## Minimum        182          bc      7.9      1          cat      -1.9
## Maximum        1900         ts      133     13.6         pig      35.2
## Mean           1350        <NA>     48.2     5.93        <NA>     13.1
## Unique (excl. NA) 490         4      202     208          2      960
## Missing values    0          0        0        0          0        0
## Sorted          TRUE        FALSE    FALSE    FALSE        FALSE    FALSE
##
##               wind.2m man.ph rain.rate   incorp
## Class          numeric numeric   numeric character
## Minimum        0.0513     6.4        0        deep
## Maximum        16.8       8.5        7.1    shallow
## Mean           3.09       7.32     0.0399     <NA>
## Unique (excl. NA) 1794     65        314        3
## Missing values    0      1053         0        0
## Sorted          FALSE    FALSE     FALSE    FALSE
##
```

Model application

Get parameters *without pH*. The pH parameters only apply to acidified slurry in set 1. This has changed in set 2 and the new set 3.

```
pars <- alfam2pars01[!grepl('man.ph', names(alfam2pars01))]
```

```
pars
```

```
##           int.f0           int.r1           int.r2           int.r3
##      -0.7364889      -1.1785848      -0.9543731      -2.9012937
##   app.mthd.os.f0      app.rate.f0           man.dm.f0   incorp.deep.f4
##      -1.1717859      -0.0134681           0.4074660      -3.6477259
## incorp.shallow.f4   app.mthd.bc.r1           man.dm.r1       air.temp.r1
##      -0.4121023           0.6283396      -0.0758220           0.0492777
##      wind.2m.r1       air.temp.r3   incorp.deep.r3   app.mthd.os.r3
##      0.0486651           0.0152419      -0.3838862      -0.1228830
##      rain.rate.r2       rain.cum.r3
##      0.4327281           -0.0300936
```

Generate predictions. Note `group` argument allows application to multiple plots. See the vignette for more details. So you can apply the function to any number of plots/locations with a single call. And in this version of the package `prep.dum = TRUE` by default (new argument name too).

```
pred <- alfam2(idat, pars = pars, app.name = 'tan.app', time.incorp = 'time.incorp', group = 'pmid')
```

```
## User-supplied parameters are being used.
```

```
## Incorporation applied for groups: 1500, 1501, 1506, 1515, 1516, 1517, 1518, 1754, 1757, 1760, 1761, ...
```

```
head(pred)
```

```
##   pmid app.mthd.ts app.mthd.bc app.mthd.os app.mthd.cs incorp.shallow
## 1  182           0           1           0           0           1
## 2  182           0           1           0           0           1
## 3  182           0           1           0           0           1
## 4  183           0           1           0           0           1
## 5  183           0           1           0           0           1
## 6  183           0           1           0           0           1
```

```
##   incorp.deep man.source.pig   ct   dt           f           s           e
## 1           0               0 4.00  4.00 2.267818e+00 104.28139 15.560788
## 2           0               0 21.00 17.00 1.500120e-02 103.48296 18.612039
## 3           0               0 44.75 23.75 3.504149e-07 101.03236 21.077637
## 4           0               0  6.00  6.00 1.722933e+00  52.38892  4.208149
## 5           0               0 20.50 14.50 9.638577e-02  52.43933  5.784289
## 6           0               0 45.20 24.70 5.636323e-04  51.01390  7.305541
##      e.int           j           er           f0           r1           r2           r3
## 1 15.560788 3.89019706 0.12743255 0.1731848 0.44709929 0.1110777 0.0010605843
## 2  3.051251 0.17948535 0.15242027 0.1731848 0.18412484 0.1110777 0.0009297968
## 3  2.465597 0.10381463 0.17261188 0.1731848 0.33701888 0.1120136 0.0010106235
## 4  4.208149 0.70135819 0.07215619 0.1427583 0.15147475 0.1110777 0.0012895630
## 5  1.576139 0.10869927 0.09918190 0.1427583 0.08777916 0.1110777 0.0011249975
## 6  1.521252 0.06158916 0.12526647 0.1427583 0.09708876 0.1110777 0.0011554081
##   f4 r5
## 1  1  0
## 2  1  0
## 3  1  0
## 4  1  0
## 5  1  0
## 6  1  0
```

See the dummy variables in the output.

Add predictions to the input variable data frame.

```
idat[, `:=` (j.NH3.pred = pred$j, e.cum.pred = pred$e, e.rel.pred = pred$er)]
```

And get final values (not beyond 78 hours).

```
idat[, ct.max := max(ct), by = pmid]
idat.final <- idat[ct == ct.max, ]
```

Model fit

Flux.

```
summ1 <- idat[, .(n = length(j.NH3),
                  me = me(j.NH3, j.NH3.pred),
                  mae = mae(j.NH3, j.NH3.pred),
                  mbe = mbe(j.NH3, j.NH3.pred)
                  ), by = datasub]

summ1
```

```
##   datasub   n      me      mae      mbe
## 1:    cal 5501 0.6865335 0.4530807 -0.19009152
## 2:    eval  423 0.6641409 0.4279645 -0.09822768
```

Cumulative emission.

```
summ2 <- idat.final[, .(n = length(e.cum),
                        me.cum = me(e.cum, e.cum.pred), me.rel = me(e.rel, e.rel.pred),
                        mae.cum = mae(e.cum, e.cum.pred), mae.rel = mae(e.rel, e.rel.pred),
                        mbe.cum = mbe(e.cum, e.cum.pred), mbe.rel = mbe(e.rel, e.rel.pred)
                        ), by = datasub]

summ2
```

```
##   datasub   n  me.cum  me.rel mae.cum  mae.rel  mbe.cum  mbe.rel
```

```
## 1:      cal 490 0.6095949 0.5284758 6.64300 0.1179259 -2.983639 -0.05565297
## 2:      eval  48 0.6184073 0.5806177 5.64609 0.1138731 -2.808678 -0.04425160

summ
```