Test of ALFAM2 closed-form solution

Sasha D. Hafner

11 March, 2024

Overview

The `alfam2()` function uses a closed-form solution to calculate emission over time. This document has tests of two closed-form solutions directly without use of the ALFAM2 package. The solutions are defined here as R functions. They are tested by comparison to numerical results.

Model structure

With a sink for slow pool S, structure is given in Fig. 1.

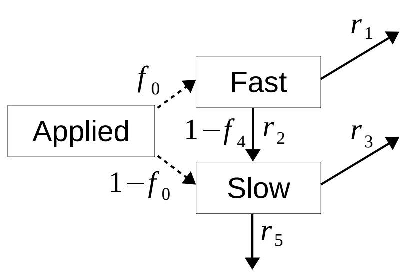


Figure 1: Structure of ALFAM2 model.

So derivatives are:

```
df/dt <-  -r1 * F - r2 * F
ds/dt <- r2 * F - r3 * S - r5 * S
de/dt <- r1 * F + r3 * S
```

Prep

Packages.

```r
library(data.table)
library(ggplot2)
library(deSolve)
```

Solution

Solution A, used in package, originally from Valdemar.

```r
ALFAMa <- function(t, app, f0, r1, r2, r3, r5) {

  rf <- r1 + r2
  rs <- r3 + r5
  rd <- rf - rs

  fi <- f0 * app
  si <- (1 - f0) * app

  f <- fi * exp(-rf * t)
  s <- exp(-rs * t) * (si - (exp(-rd * t) - 1) / rd * r2 * fi)

  ef <- (1 - exp(-rf * t)) * r1 / rf * fi
  es <- (fi + si - f - s - ef) / rs * r3
  e <- ef + es

  return(data.table(time = t, app = app, f0 = f0, r1 = r1, r2 = r2,
                    r3 = r3, r5 = r5, f = f, s = s, e = e))
}
```

Solution B, from Paul.

```r
ALFAMb <- function(t, app, f0, r1, r2, r3, r5) {

  rf <- r1 + r2
  rs <- r3 + r5
  rd <- rf - rs
  fi <- f0 * app
  si <- (1 - f0) * app

  f <- fi * exp(-rf * t)
  s <- (si - (exp(-rd * t) - 1) * r2 * fi / rd) * exp(-rs * t)

  ef <- fi * (r1 / rf) * (1 - exp(-rf * t))
  es <- (si + fi * r2 / rd) * (r3 / rs) * (1 - exp(-rs * t)) -
        (fi * r2 / rd) * (r3 / rf) * (1 - exp(-rf * t))

  e <- ef + es

  return(data.table(time = t, app = app, f0 = f0, r1 = r1, r2 = r2,
                    r3 = r3, r5 = r5, f = f, s = s, e = e))
}
```

Predictions

Input data.

```r
dat <- data.table(time = 0:168)
```

Parameter values.

```r
r1 <- 0.03
r2 <- 0.01
r3 <- 0.001
r5 <- 0.05
```

```
f0 <- 0.3
```

Closed-form solution A.

```
predcfa <- ALFAMa(t = dat[, time], app = 100, f0, r1, r2, r3, r5)
```

And B.

```
predcfb <- ALFAMb(t = dat[, time], app = 100, f0, r1, r2, r3, r5)
```

Numerical solution uses `lsoda()`.

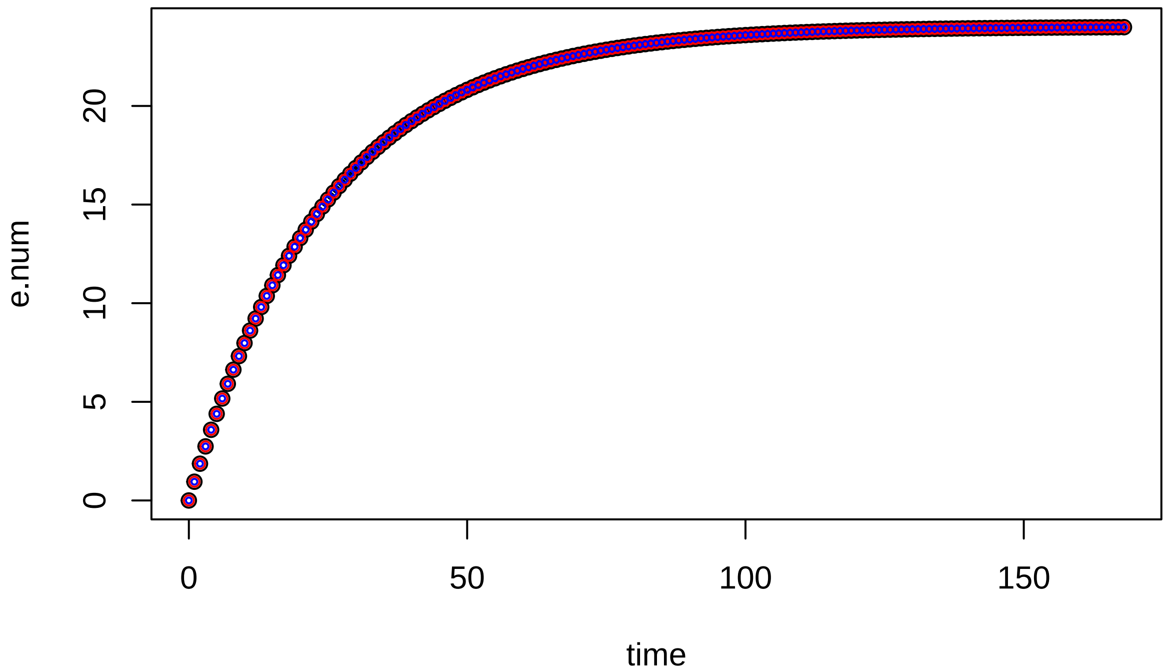
```
pars <- c(r1 = r1, r2 = r2, r3 = r3, r5 = r5, f0 = f0)
app <- 100
y <- c(f = pars['f0'] * app, s = (1 - pars['f0']) * app, e = 0)
names(y) <- c('f', 's', 'e')

rates <- function(t, x, parms) {
  r1 <- parms['r1']
  r2 <- parms['r2']
  r3 <- parms['r3']
  r5 <- parms['r5']

  f <- x[1]
  s <- x[2]

  dfdt <- as.numeric(-r1 * f - r2 * f)
  dsdt <- as.numeric(r2 * f - r3 * s - r5 * s)
  dedt <- as.numeric(r1 * f + r3 * s)

  return(list(c(dfdt, dsdt, dedt)))

}

prednum <- data.table(lsoda(y = y, times = dat[, time], func = rates, parms = pars))
```

Combine results.

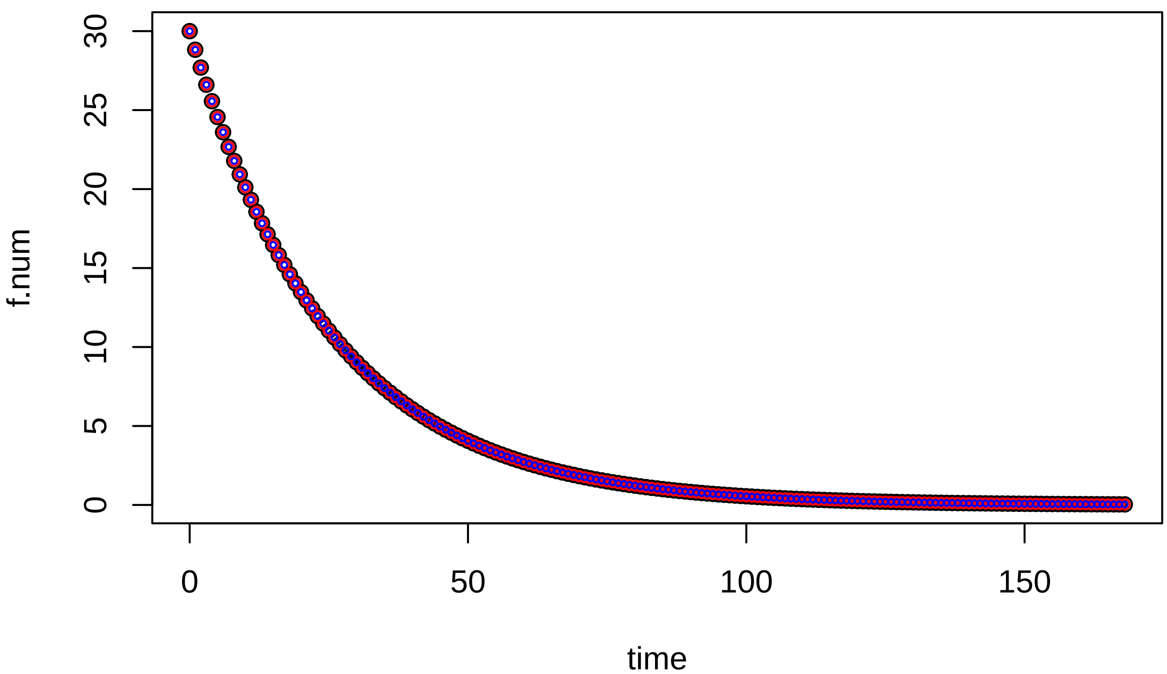
```
pred <- merge(predcfa, prednum, by = 'time', suffixes = c('', '.num'))
pred <- merge(pred, predcfb, by = 'time', suffixes = c('.cfa', '.cfb'))
```

Compare.

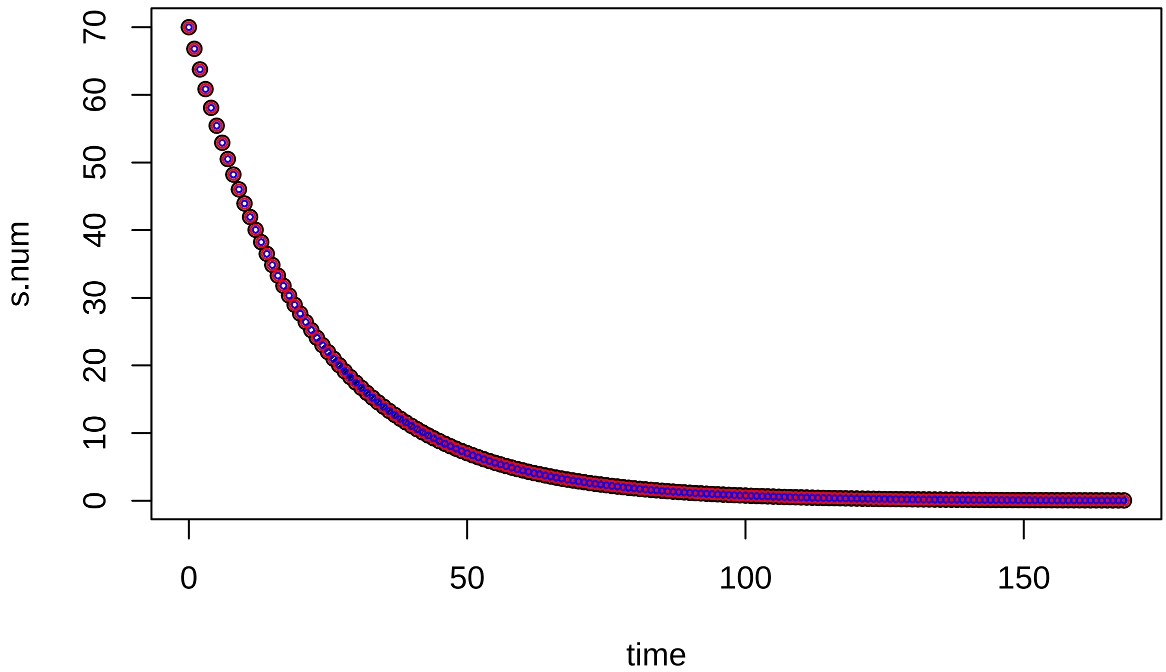
```
plot(e.num ~ time, data = pred)
points(e.cfa ~ time, data = pred, col = 'red', cex = 0.7)
points(e.cfb ~ time, data = pred, col = 'blue', cex = 0.4)
```




```r
plot(f.num ~ time, data = pred)
points(f.cfa ~ time, data = pred, col = 'red', cex = 0.7)
points(f.cfb ~ time, data = pred, col = 'blue', cex = 0.4)
```




```r
plot(s.num ~ time, data = pred)
points(s.cfa ~ time, data = pred, col = 'red', cex = 0.7)
points(s.cfb ~ time, data = pred, col = 'blue', cex = 0.4)
```



Conclusion

Both closed-form solutions seem to be accurate.

Something else: try to blow up solution

```
ALFAMa(t = 168, app = 100, f0 = 0.5, r1 = 0.1, r2 = 0.1, r3 = 0.01, r5 = 0.01)
```

```
##    time app  f0  r1  r2   r3   r5            f        s        e
## 1:  168 100 0.5 0.1 0.1 0.01 0.01 1.278425e-13 2.701631 61.14918
```

```
ALFAMa(t = 168, app = 100, f0 = 0.5, r1 = 1E7, r2 = 0.1, r3 = 0.01, r5 = 0.01)
```

```
##    time app  f0    r1  r2   r3   r5 f        s        e
## 1:  168 100 0.5 1e+07 0.1 0.01 0.01 0 1.736763 74.13162
```

```
ALFAMa(t = 168, app = 100, f0 = 0.5, r1 = 0.1, r2 = 1E7, r3 = 0.01, r5 = 0.01)
```

```
##    time app  f0  r1    r2   r3   r5 f        s        e
## 1:  168 100 0.5 0.1 1e+07 0.01 0.01 0 3.473526 48.26324
```

```
ALFAMa(t = 168, app = 100, f0 = 0.5, r1 = 0.1, r2 = 0.1, r3 = 1E7, r5 = 0.01)
```

```
##    time app  f0  r1  r2    r3   r5            f   s   e
## 1:  168 100 0.5 0.1 0.1 1e+07 0.01 1.278425e-13 NaN NaN
```

```
ALFAMa(t = 168, app = 100, f0 = 0.5, r1 = 0.1, r2 = 0.1, r3 = 0.01, r5 = 1E7)
```

```
##    time app  f0  r1  r2   r3    r5            f   s   e
## 1:  168 100 0.5 0.1 0.1 0.01 1e+07 1.278425e-13 NaN NaN
```

Problems are with r3 and r5, presumably through exp(… + rd).

Paul's solution will still give emission because it does not use `s` for calculation of `e`:

```r
ALFAMb(t = 168, app = 100, f0 = 0.5, r1 = 0.1, r2 = 0.1, r3 = 0.01, r5 = 0.01)
```

```
##    time app  f0  r1  r2   r3   r5            f        s        e
## 1:  168 100 0.5 0.1 0.1 0.01 0.01 1.278425e-13 2.701631 61.14918
```

```r
ALFAMb(t = 168, app = 100, f0 = 0.5, r1 = 1E7, r2 = 0.1, r3 = 0.01, r5 = 0.01)
```

```
##    time app  f0    r1  r2   r3   r5 f        s        e
## 1:  168 100 0.5 1e+07 0.1 0.01 0.01 0 1.736763 74.13162
```

```r
ALFAMb(t = 168, app = 100, f0 = 0.5, r1 = 0.1, r2 = 1E7, r3 = 0.01, r5 = 0.01)
```

```
##    time app  f0  r1    r2   r3   r5 f        s        e
## 1:  168 100 0.5 0.1 1e+07 0.01 0.01 0 3.473526 48.26324
```

```r
ALFAMb(t = 168, app = 100, f0 = 0.5, r1 = 0.1, r2 = 0.1, r3 = 1E7, r5 = 0.01)
```

```
##    time app  f0  r1  r2    r3   r5            f   s   e
## 1:  168 100 0.5 0.1 0.1 1e+07 0.01 1.278425e-13 NaN 100
```

```r
ALFAMb(t = 168, app = 100, f0 = 0.5, r1 = 0.1, r2 = 0.1, r3 = 0.01, r5 = 1E7)
```

```
##    time app  f0  r1  r2   r3    r5            f   s  e
## 1:  168 100 0.5 0.1 0.1 0.01 1e+07 1.278425e-13 NaN 25
```

A work-around for Valdemar's solution:

```r
ALFAMa2 <- function(t, app, f0, r1, r2, r3, r5) {

  rf <- r1 + r2
  rs <- r3 + r5
  rd <- rf - rs

  fi <- f0 * app
  si <- (1 - f0) * app

  erdt <- exp(-rd * t)
  if (erdt > .Machine$double.xmax) erdt <- .Machine$double.xmax / 10

  f <- fi * exp(-rf * t)
  s <- exp(-rs * t) * (si - (erdt - 1) / rd * r2 * fi)

  ef <- (1 - exp(-rf * t)) * r1 / rf * fi
  es <- (fi + si - f - s - ef) / rs * r3
  e <- ef + es

  return(data.table(time = t, app = app, f0 = f0, r1 = r1, r2 = r2,
                    r3 = r3, r5 = r5, f = f, s = s, e = e))
}
```

```r
ALFAMa2(t = 168, app = 100, f0 = 0.5, r1 = 0.1, r2 = 0.1, r3 = 0.01, r5 = 0.01)
```

```
##    time app  f0  r1  r2   r3   r5            f        s        e
## 1:  168 100 0.5 0.1 0.1 0.01 0.01 1.278425e-13 2.701631 61.14918
```

```r
ALFAMa2(t = 168, app = 100, f0 = 0.5, r1 = 1E7, r2 = 0.1, r3 = 0.01, r5 = 0.01)
```

```
##    time app  f0    r1  r2   r3   r5 f        s        e
## 1:  168 100 0.5 1e+07 0.1 0.01 0.01 0 1.736763 74.13162
```

```r
ALFAMa2(t = 168, app = 100, f0 = 0.5, r1 = 0.1, r2 = 1E7, r3 = 0.01, r5 = 0.01)
```

```
##    time app  f0  r1    r2   r3   r5 f        s        e
## 1:  168 100 0.5 0.1 1e+07 0.01 0.01 0 3.473526 48.26324
```

```r
ALFAMa2(t = 168, app = 100, f0 = 0.5, r1 = 0.1, r2 = 0.1, r3 = 1E7, r5 = 0.01)
```

```
##    time app  f0  r1  r2    r3   r5            f s   e
## 1:  168 100 0.5 0.1 0.1 1e+07 0.01 1.278425e-13 0 100
```

```r
ALFAMa2(t = 168, app = 100, f0 = 0.5, r1 = 0.1, r2 = 0.1, r3 = 0.01, r5 = 1E7)
```

```
##    time app  f0  r1  r2   r3    r5            f s  e
## 1:  168 100 0.5 0.1 0.1 0.01 1e+07 1.278425e-13 0 25
```