

mkin -
Routines for fitting kinetic models with one or more
state variables to chemical degradation data

Johannes Ranke
Product Safety
Harlan Laboratories Ltd.
Zelgliweg 1, CH-4452 Itingen, Switzerland

September 10, 2010

Abstract

In the regulatory evaluation of chemical substances like plant protection products (pesticides), biocides and other chemicals, degradation data play an important role. For the evaluation of pesticide degradation experiments, detailed guidance has been developed, based on nonlinear optimisation. The R add-on package **mkin** implements fitting some of the models recommended in this guidance from within R and calculates some statistical measures for data series within one or more compartments, for parent and metabolites.

Contents

1	Introduction	1
2	Example	1
2.1	Data format	1
2.2	Model definition	2
2.3	Fitting the model	3

Key words: Kinetics, FOCUS, nonlinear optimisation

1 Introduction

Many approaches are possible regarding the evaluation of chemical degradation data. The **kinfit** package ([Ranke, 2010a](#)) in R ([R Development Core Team, 2010](#)) implements the approach recommended in the kinetics report provided by the FORum for Co-ordination of pesticide fate models and their USe ([FOCUS Work Group on Degradation Kinetics, 2006](#)) for simple data series for one parent compound in one compartment.

The **mkim** package ([Ranke, 2010b](#)) extends this approach to data series with metabolites and more than one compartment and includes the possibility for back reactions.

2 Example

In the following, requirements for data formatting are explained. Then the procedure for fitting the four kinetic models recommended by the FOCUS group to an example dataset for parent only given in the FOCUS kinetics report is illustrated. The explanations are kept rather verbose in order to lower the barrier for R newcomers.

2.1 Data format

The following listing shows example dataset C from the FOCUS kinetics report as distributed with the **mkim** package

```
R> library("mkim")
R> FOCUS_2006_C

  name time value
1 parent    0  85.1
2 parent    1  57.9
3 parent    3  29.9
```

```

4 parent      7  14.6
5 parent     14   9.7
6 parent     28   6.6
7 parent     63   4.0
8 parent     91   3.9
9 parent    119   0.6

```

Note that the data needs to be in the format of a data frame containing a variable **name** specifying the observed variable, indicating the compound name and, if applicable, the compartment, a variable **time** containing sampling times, and a numeric variable **value** specifying the observed value of the variable. If a further variable **error** is present, this will be used to give different weights to the data points (the higher the error, the lower the weight, see the help page of the **modCost** function of the **FME** package ([Soetaert and Petzoldt, 2010](#))). Replicate measurements are not recorded in extra columns but simply appended, leading to multiple occurrences of the sampling times **time**.

Small to medium size dataset can be conveniently entered directly as R code as shown in the following listing

```

R> example_data <- data.frame(
+   name = rep("parent", 9),
+   time = c(0, 1, 3, 7, 14, 28, 63, 91, 119),
+   value = c(85.1, 57.9, 29.9, 14.6, 9.7, 6.6, 4, 3.9, 0.6)
+ )

```

2.2 Model definition

The next task is to define the model to be fitted to the data. In order to facilitate this task, a convenience function **mkmod** is available.

```

R> SF0 <- mkmod(parent = list(type = "SF0"))
R> SFORB <- mkmod(parent = list(type = "SFORB"))
R> SF0_SF0 <- mkmod(
+   parent = list(type = "SF0", to = "m1", sink = TRUE),
+   m1 = list(type = "SF0"))
R> SFORB_SF0 <- mkmod(
+   parent = list(type = "SFORB", to = "m1", sink = TRUE),
+   m1 = list(type = "SF0"))

```

The model definitions given above define sets of linear first-order ordinary differential equations. In these cases, a coefficient matrix is also returned.

Other models that include time on the right-hand side of the differential equation are the first-order multi-compartment (FOMC) model and the Hockey-Stick (HS) model. At present, only the FOMC model can only be used, and only for the parent compound.

2.3 Fitting the model

Then the model parameters should be fitted to the data. The function `mkinfitt` internally creates a cost function using `modCost` from the **FME** package and the produces a fit using `modFit` from the same package. In cases of linear first-order differential equations, the solution used for calculating the cost function is based on the fundamental system of the coefficient matrix, as proposed by [Bates and Watts \(1988\)](#).

```
R> # Do not show significance stars as they interfere with vignette generation
R> options(show.signif.stars = FALSE)
R> SFO.fit <- mkinfit(SFO, FOCUS_2006_C)
```

```
Model cost at call 1 : 4718.953
Model cost at call 4 : 4718.953
Model cost at call 5 : 572.4065
Model cost at call 7 : 572.4065
Model cost at call 8 : 236.2068
Model cost at call 9 : 236.2068
Model cost at call 11 : 198.9361
Model cost at call 12 : 198.9361
Model cost at call 14 : 196.6776
Model cost at call 15 : 196.6776
Model cost at call 16 : 196.6776
Model cost at call 17 : 196.5420
Model cost at call 18 : 196.5420
Model cost at call 19 : 196.5420
Model cost at call 20 : 196.5339
Model cost at call 21 : 196.5339
Model cost at call 22 : 196.5339
Model cost at call 23 : 196.5334
Model cost at call 25 : 196.5334
Model cost at call 26 : 196.5334
Model cost at call 28 : 196.5334
Model cost at call 29 : 196.5334
Model cost at call 33 : 196.5334
```

```
R> summary(SFO.fit)
```

Equations:

```
[1] d_parent = - k_parent_sink * parent
```

Starting values for optimised parameters:

	initial	type	lower	upper
parent_0	100.0	state	0	Inf
k_parent_sink	0.1	deparm	0	Inf

Fixed parameter values:

None

Optimised parameters:

	Estimate	Std. Error	t value	Pr(>t)
parent_0	82.4920	4.7402	17.402	2.54e-07
k_parent_sink	0.3061	0.0459	6.668	0.000143

Residual standard error: 5.299 on 7 degrees of freedom

Chi2 error levels in percent:

	err.min	n.optim	df
All data	15.84	2	7
parent	15.84	2	7

Estimated disappearance times:

	DT50	DT90
parent	2.265	7.523

Estimated formation fractions:

	ff
parent_sink	1

Data:

time	variable	observed	predicted	residual
0	parent	85.1	82.49198372280005742	2.608
1	parent	57.9	60.74234531039351026	-2.842
3	parent	29.9	32.93450756424957149	-3.035
7	parent	14.6	9.68211886535106103	4.918
14	parent	9.7	1.13639436818243311	8.564
28	parent	6.6	0.01565475942942832	6.584
63	parent	4.0	0.00000034869017577	4.000
91	parent	3.9	0.00000000006617202	3.900
119	parent	0.6	0.00000000000001256	0.600

R> SFORB.fit <- mkinfit(SFORB, FOCUS_2006_C)

Model cost at call	1	:	7044.136
Model cost at call	4	:	7044.136
Model cost at call	7	:	3460.19
Model cost at call	9	:	3460.19
Model cost at call	11	:	3460.190
Model cost at call	13	:	312.9928
Model cost at call	15	:	312.9928
Model cost at call	17	:	312.9928
Model cost at call	18	:	27.14677
Model cost at call	20	:	27.14677
Model cost at call	23	:	4.437654
Model cost at call	25	:	4.437654
Model cost at call	28	:	4.362927
Model cost at call	31	:	4.362927
Model cost at call	33	:	4.362715

```

Model cost at call 38 : 4.362714
Model cost at call 43 : 4.362714
Model cost at call 48 : 4.362714
Model cost at call 52 : 4.362714

```

```
R> summary(SFORB.fit)
```

Equations:

```

[1] d_parent_free = - k_parent_free_sink * parent_free - k_parent_free_bound * parent_free
[2] d_parent_bound = + k_parent_free_bound * parent_free - k_parent_bound_free * parent_free

```

Starting values for optimised parameters:

	initial	type	lower	upper
parent_free_0	100.0	state	0	Inf
k_parent_free_sink	0.1	deparm	0	Inf
k_parent_free_bound	0.1	deparm	0	Inf
k_parent_bound_free	0.1	deparm	0	Inf

Fixed parameter values:

	value	type
parent_bound	0	state

Optimised parameters:

	Estimate	Std. Error	t value	Pr(>t)
parent_free_0	85.002737	0.890671	95.437	1.20e-09
k_parent_free_sink	0.395044	0.014308	27.610	5.83e-07
k_parent_free_bound	0.061599	0.007289	8.451	0.000190
k_parent_bound_free	0.020764	0.003752	5.533	0.001322

Residual standard error: 0.9341 on 5 degrees of freedom

Chi2 error levels in percent:

	err.min	n.optim	df
All data	2.662	4	5
parent	2.662	4	5

Estimated disappearance times:

	DT50	DT90
parent	1.887	21.25

Estimated formation fractions:

	ff
parent_free_sink	1

Data:

time	variable	observed	predicted	residual
0	parent	85.1	85.003	0.09726
1	parent	57.9	58.039	-0.13912
3	parent	29.9	30.054	-0.15351

7	parent	14.6	13.866	0.73388
14	parent	9.7	9.787	-0.08657
28	parent	6.6	7.532	-0.93205
63	parent	4.0	4.033	-0.03269
91	parent	3.9	2.447	1.45348
119	parent	0.6	1.484	-0.88424

```
R> SFO_SFO.fit <- mkinfit(SFO_SFO, FOCUS_2006_D, plot=TRUE)
```

```
Model cost at call 1 : 18994.29
Model cost at call 3 : 18994.29
Model cost at call 7 : 10641.45
Model cost at call 8 : 10641.45
Model cost at call 12 : 7145.673
Model cost at call 14 : 7145.672
Model cost at call 17 : 411.979
Model cost at call 18 : 411.9788
Model cost at call 22 : 371.2202
Model cost at call 23 : 371.2201
Model cost at call 27 : 371.2134
Model cost at call 29 : 371.2134
Model cost at call 31 : 371.2134
Model cost at call 32 : 371.2134
```

```
R> summary(SFO_SFO.fit, data=FALSE)
```

Equations:

```
[1] d_parent = - k_parent_sink * parent - k_parent_m1 * parent
[2] d_m1 = - k_m1_sink * m1 + k_parent_m1 * parent
```

Starting values for optimised parameters:

	initial	type	lower	upper
parent_0	100.0	state	0	Inf
k_parent_sink	0.1	deparm	0	Inf
k_m1_sink	0.1	deparm	0	Inf
k_parent_m1	0.1	deparm	0	Inf

Fixed parameter values:

	value	type
m1	0	state

Optimised parameters:

	Estimate	Std. Error	t value	Pr(>t)
parent_0	9.960e+01	1.614e+00	61.720	< 2e-16
k_parent_sink	4.792e-02	3.750e-03	12.777	3.05e-15
k_m1_sink	5.261e-03	7.159e-04	7.349	5.76e-09
k_parent_m1	5.078e-02	2.094e-03	24.248	< 2e-16

Residual standard error: 3.211 on 36 degrees of freedom

Chi2 error levels in percent:

	err.min	n.optim	df
All data	6.565	4	16
parent	6.827	3	6
m1	4.748	1	10

Estimated disappearance times:

	DT50	DT90
parent	7.023	23.33
m1	131.761	437.70

Estimated formation fractions:

	ff
parent_sink	0.4855
parent_m1	0.5145
m1_sink	1.0000

```
R> SFORB_SF0.fit <- mkinfit(SFORB_SF0, FOCUS_2006_D, plot=TRUE)
```

Model cost at call	1	:	16413.78
Model cost at call	3	:	16413.78
Model cost at call	10	:	3061.057
Model cost at call	11	:	3061.056
Model cost at call	17	:	392.3445
Model cost at call	18	:	392.3445
Model cost at call	20	:	392.3445
Model cost at call	22	:	392.3445
Model cost at call	25	:	354.9441
Model cost at call	27	:	354.9441
Model cost at call	31	:	354.9441
Model cost at call	33	:	354.6558
Model cost at call	35	:	354.6558
Model cost at call	39	:	354.6558
Model cost at call	40	:	352.5951
Model cost at call	42	:	352.5951
Model cost at call	46	:	352.5951
Model cost at call	48	:	352.2550
Model cost at call	50	:	352.2550
Model cost at call	54	:	352.2550
Model cost at call	56	:	352.2078
Model cost at call	58	:	352.2078
Model cost at call	62	:	352.2078
Model cost at call	65	:	352.2058
Model cost at call	66	:	352.2058
Model cost at call	68	:	352.2058
Model cost at call	72	:	352.205
Model cost at call	74	:	352.205
Model cost at call	80	:	352.2048
Model cost at call	83	:	352.2048


```

Model cost at call 87 : 352.2048
Model cost at call 89 : 352.2048
Model cost at call 94 : 352.2048
Model cost at call 97 : 352.2048
Model cost at call 102 : 352.2048
Model cost at call 104 : 352.2048
Model cost at call 106 : 352.2048

```

```
R> summary(SFORB_SFO.fit, data=FALSE)
```

Equations:

```

[1] d_parent_free = - k_parent_free_sink * parent_free - k_parent_free_bound * parent_free
[2] d_parent_bound = + k_parent_free_bound * parent_free - k_parent_bound_free * parent_free
[3] d_m1 = - k_m1_sink * m1 + k_parent_free_m1 * parent_free

```

Starting values for optimised parameters:

	initial	type	lower	upper
parent_free_0	100.0	state	0	Inf
k_parent_free_sink	0.1	deparm	0	Inf
k_parent_free_bound	0.1	deparm	0	Inf
k_parent_bound_free	0.1	deparm	0	Inf
k_m1_sink	0.1	deparm	0	Inf
k_parent_free_m1	0.1	deparm	0	Inf

Fixed parameter values:

	value	type
parent_bound	0	state
m1	0	state

Optimised parameters:

	Estimate	Std. Error	t value	Pr(>t)
parent_free_0	1.011e+02	2.020e+00	50.033	< 2e-16
k_parent_free_sink	6.407e-02	2.691e-02	2.381	0.01150
k_parent_free_bound	1.680e-01	5.143e-01	0.327	0.37297
k_parent_bound_free	5.238e-01	8.545e-01	0.613	0.27199
k_m1_sink	5.213e-03	7.210e-04	7.230	1.14e-08
k_parent_free_m1	6.562e-02	2.542e-02	2.581	0.00716

Residual standard error: 3.219 on 34 degrees of freedom

Chi2 error levels in percent:

	err.min	n.optim	df
All data	6.645	6	14
parent	7.207	5	4
m1	5.123	1	10

Estimated disappearance times:

	DT50	DT90
parent	6.805	24.05

```
m1      132.971 441.72
```

Estimated formation fractions:

```
      ff  
parent_free_sink 0.494  
parent_free_m1   0.506  
m1_sink          1.000
```

References

D. Bates and D. Watts. *Nonlinear regression and its applications*. Wiley-Interscience, 1988.

FOCUS Work Group on Degradation Kinetics. *Guidance Document on Estimating Persistence and Degradation Kinetics from Environmental Fate Studies on Pesticides in EU Registration. Report of the FOCUS Work Group on Degradation Kinetics*, 2006. URL <http://focus.jrc.ec.europa.eu/dk>. EC Document Reference Sanco/10058/2005 version 2.0.

R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2010. URL <http://www.R-project.org>. ISBN 3-900051-07-0.

Johannes Ranke. *kinfit: Routines for fitting simple kinetic models to chemical degradation data*, 2010a. URL <http://CRAN.R-project.org>.

Johannes Ranke. *mkln: Routines for fitting kinetic models with one or more state variables to chemical degradation data*, 2010b. URL <http://CRAN.R-project.org>.

Karline Soetaert and Thomas Petzoldt. Inverse modelling, sensitivity and monte carlo analysis in R using package FME. *Journal of Statistical Software*, 33(3):1–28, 2010. URL <http://www.jstatsoft.org/v33/i03/>.