A beginner's guide to fitting ordinary different equations to data

Sang Woo Park Benjamin M. Bolker David J. D. Earn *

December 29, 2020

Target audience: students and researchers with some degree of mathematical sophistication who aren't necessarily great with statistics. They know a reasonable amount about ODEs, probably a little bit about numerical integration, maybe next to nothing about maximum likelihood/process vs observation error/state space models/MCMC/etc..

Target journal: R Journal? JSS? Bull Math Biol? Methods in Ecol & Evol?

1 Introduction

Mathematical models based on ordinary different equations (ODEs) are ubiquitous in a vast array of application areas. ⟨*DJDE: refs? maybe not necessary.*⟩ ⟨*BMB: "in science and engineering?"*⟩ ⟨*SWP: I like "in science and engineering"*⟩ Researchers studying ODE models are often highly skilled in dynamical systems theory, but many lack experience estimating the parameters of their models from data. When confronted with observations from the real world, how should one go about fitting an ODE model to the data?

In principle, the answer is straightforward: examine all possible solutions of the ODEs (considering all possible parameter values and initial states) and identify the solution curve that is closest to the observed data. This approach is sometimes feasible, for simple enough problems. But more often than not, fitting an ODE to data is a challenging statistical and computational problem. In particular:

- The dimension of the combined of parameter and initial states is rarely less than 4 and often much larger, making a naïve search for the best fitting solution computationally overwhelming.
- Different measures of goodness of fit may identify different "best" solutions.
- Multiple solutions, associated with different regions of parameter space, may fit the data equally well, or nearly equally well.
- Statistical inference — testing for clear evidence of the operation of a particular process, attaching a measure of confidence to a parameter estimates, or putting confidence intervals on predictions — requires additional knowledge or assumptions

*Department of Mathematics & Statistics, McMaster University, Hamilton, Ontario, Canada

about the stochastic processes that generate the variation of observed data around the best-fit solution.

As one attempts to address these issues, more subtle challenges arise and one can quickly reach the limits of the state of the art. Making meaningful mechanistic inferences from ODE fits to data can be difficult, especially when fitting complex models to small, noisy data sets.

At this point you may want to give up and work on something else, but we hope you won't. The situation isn't *that* bad. Our goal here is to provide—and to present a pedagogical introduction to—an `R` package that makes fitting many ODEs to data relatively painless. Simultaneously, we aim to explain the uncertainties and limitations inherent in ODE-fitting, and provide convenient tools and references for users who wish to delve more deeply into the subject.

⟨DJDE: The general style I'm imagining is similar to Numerical Recipes (Press et al., 1992)*, i.e., provide tools that work but also explain clearly and concisely why they work, what they are doing, and under what circumstances they can be expected to fail.⟩*

2 Trajectory matching: from simple to complex

2.1 Least-squares fitting

In order to find a solution of an ODE model that is closest to the observed data, we have to first decide on a measure of the distance between the solution and the data. One of the most basic measures we can use is the sum of squared differences between the two. The least-squares solution of an ODE model (i.e., the solution that minimizes the sum of squared differences) will visually resemble the patterns in the data to some extent; therefore, we can expect it give us a biologically sensible set of parameters.

Suppose we have (1) a gradient function that represents the derivatives $\frac{d\boldsymbol{y}}{dt}$; (2) a method for numerical integration; (3) a set of observations $\boldsymbol{y}(t_i)$ where $i = 1, \ldots, n$ (for simplicity, from a single time series with n observations); (4) a method for numerical minimization of an objective function. Define a trajectory (a solution of the ODE for specified parameters $\boldsymbol{\beta}$ [and initial conditions] and time points t_i) as $\hat{\boldsymbol{y}}(t_i|\boldsymbol{\beta})$. *⟨BMB: notation???⟩⟨SWP: changed some notations⟩* Then, the method of least-squared fitting seeks to find a set of parametrs, $\hat{\boldsymbol{\beta}}$, that minimize the following objective function: $\sum_{i=1}^{n} \left(\boldsymbol{y}(t_i) - \hat{\boldsymbol{y}}(t_i|\boldsymbol{\beta})\right))^2$. Such set of parameters can be formally defined as follows:

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \sum_{i=1}^{n} \left(\boldsymbol{y}(t_i) - \hat{\boldsymbol{y}}(t_i|\boldsymbol{\beta})\right))^2 . \tag{1}$$

The actual process of finding $\hat{\beta}$ will depend on the minimizer of our choice.

gradient functions should be straightforward (you know what you want to model, right? ...)

observations as noted above, we're assuming a single time series for now. Regularity of sampling isn't much of a problem. It's OK to measure a subset of the state variables (e.g. just prevalence of infected in an SIR model, or just predators in a pred-prey model). May need some care to make sure you're comparing observations with ODE properly (e.g. do observations represent measurements of state variables, rates, or integrals of rates? Measuring cumulative properties may have tricky statistical consequences later ...)

integrator for simple problems, anything should do. Open question as to whether simple/stable (e.g. RK4) is better than something more sophisticated (e.g. LSODA) ⟨*BMB: in general, the easier the problem, the less the details matter!*⟩

minimizer similar to integrator, i.e. details don't matter if the problem is easy. Derivative-free methods such as Nelder-Mead are robust but slower; derivative-based methods are effective *if sensitivity equations are used* (see below). Some minimizers such as Levenberg-Marquardt are particularly tuned to least-squares (although derivs still matter)

initial conditions if all variables are observed, we can pretend the initial observations as known without error and set $\hat{\boldsymbol{y}}(0) = \boldsymbol{y}(0)$. Otherwise we have to treat ICs as auxiliary parameters.

⟨*SWP: Do we just want to put definitions here? Not sure what you're exactly looking for here...*⟩
⟨*SWP: Is it OK to extract data from a figure and put it in the package as long as we cite it? This seems like a good example here...*⟩ For example, consider a time series of human tumor cell (GI-101A) growth within a nude mice (Worschech et al., 2009); the time series data was extracted from the original article (Figure 1A, control data) using WebPlotDigitizer and is included in the `fitode` package (see `tumorgrowth`).

```
plot(tumorgrowth)
library(fitode)
```

Murphy et al. (2016) analyzed this time series by fitting seven different ODE models (i.e., exponential, Mendelsohn, logistic, linear, surface, Bertalanffy, and Gompertz) via least-squares and comparing their model predictions. Here, we demonstrate the fitting of the exponential growth model.

In order to fit the exponential growth model using least-squares, we have to set up the ODE model using an `odemodel` function in `fitode`. In particular, we have to specify gradient functions (`model` argument), an observation model (`observation` argument), initial conditions (`initial` argument), and parameters (`par` argument) of the model. The gradient function is simple:

$$\frac{dA}{dt} = rA, \tag{2}$$

where r is the exponential growth rate. In `fitode`, gradient functions are expressed as a list of *formulas* (the left hand side (LHS) of the formula determines the state variable and the right hand side (RHS) of the formula determines its gradient function):

```
model=list(A ~ r*A)
```

Likewise, observation model should specify that we want to minimize the squared sum of differences between the observed `volume` and the trajectory of `A` (the LHS of the formula determines the observed variable and the RHS of the formula determines the objective function):

```
observation=list(volume ~ ols(mean=A))
```

We have to specify the initial condition for our state variable:

```
initial=list(A~A0)
```

Finally, we have to specify all parameters (including the initial conditions) of the model. In this case, we have two parameters: the exponential growth rate, `r`, and the initial condition, `A0`.

```
par=c("r", "A0")
```

Putting them altogether, we can create an `odemodel` object as follows:

```
exp_model <- odemodel(
  model=list(A ~ r*A),
  observation=list(volume ~ ols(mean=A)),
  initial=list(A~A0),
  par=c("r", "A0")
)
```

Before we can start fitting this model, we have to specify a starting parameter for the optimization. In principle, we can try out multiple starting parameters to test for convergence; this is not necessary for sufficiently simple models. For now, we will leave the discussion on starting parameters and pick a starting parameter based on visual inspections:

```
start <- c(r=0.03, A0=150)
ss <- simulate(exp_model, parms=start, times=tumorgrowth$day)
plot(tumorgrowth)
lines(A~times, data=ss)
```

To fit the model, we have to call the `odemodel` object through the `fitode` function.

```
tumor_exp_fit <- fitode(
  model=exp_model,
  data=tumorgrowth,
  start=start,
  tcol="day"
)
```

Once the model has been fitted, we can use the `plot` function to plot the estimated trajectory along with the data:

```
plot(tumor_exp_fit)
```

Finally, we can obtain the parameter estimates by using the `coef` function:

```
coef(tumor_exp_fit)
```

You might think that this fit is good enough and decide to carry on with our analysis of the ODE model using the estimated parameters. While least-squares fits provide solutions that are in some sense "optimal" (although the Gauss-Markov theorem only applies to *linear* models), but don't provide any scope for inference. For example, how much confidence can we have in these parameter estimates? That said, by avoiding inference, you also avoid having to think much about the characteristics of the observations (independence, distribution, etc.). We hope that you think about these characteristics carefully and continue to follow us through this guide.

2.2 Gaussian errors

If we're willing to assume that the residuals $(\boldsymbol{y}(t_i) - \hat{\boldsymbol{y}}(t_i|\boldsymbol{\beta}))$ are *independent*, *identically distributed* Gaussian random variables – i.e.,

$$\boldsymbol{y}(t_i) \sim \mathcal{N}(\hat{\boldsymbol{y}}(t_i|\boldsymbol{\beta}), \sigma^2), \tag{3}$$

where σ^2 is the variance of the Gaussian distribution – then we can start making statistical inferences: confidence intervals on parameters, confidence and prediction intervals on predictions, etc.. Assuming independence of successive residuals typically means we are assuming that the noise in the data is *observation error*; that is, the underlying process is deterministic (and smooth), and all the noise in the data comes from (independent) imperfect observations of the process.

Under these assumptions, the likelihood (i.e., the probability of the observed data) can be written as follows:

$$\mathcal{L}(\boldsymbol{y}(t_i)|\boldsymbol{\beta}) = \prod_{i=1}^{n} \left(\frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2\sigma^2} \left(\boldsymbol{y}(t_i) - \hat{\boldsymbol{y}}(t_i|\boldsymbol{\beta})\right)^2 \right) \right), \tag{4}$$

where the parameter set $\boldsymbol{\beta}$ now includes parameters of the ODE model, initial conditions, and the variance term σ^2 of the Gaussian distribution as an auxilliary parameter (i.e., $\sigma^2 \in \boldsymbol{\beta}$). Our goal is to find a set of parameters $\boldsymbol{\beta}$ that maximizes the likelihood; such set of parameters is referred to as the maximum likelihood estimate (MLE). Equivalently, the MLE can be expressed as a parameter that minimizes the negative log-likelihood function, which is now our new measure for the distance between the solution and the data:

$$\hat{\beta} = \arg\min_{\boldsymbol{\beta}} \left(-\frac{n}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_{i=1}^{n} \left(\boldsymbol{y}(t_i) - \hat{\boldsymbol{y}}(t_i|\boldsymbol{\beta})\right)^2 \right), \tag{5}$$

which shows that the Gaussian errors are equivalent to *scaled* least-squares errors. Using the negative log-likelihood also results in a more stable numerical computation and useful statistical properties.

There are several ways in which we can fit an ODE model under the Gaussian assumption. First, we can estimate all parameters at once.

- we can fit via explicit Gaussian (with or without normalization terms), profiled Gaussian (set $\hat{\sigma} = \text{RSS}/n$), or quasi-likelihood (fit with $\sigma = 1$, then estimate $\hat{\sigma}$ from final value of RSS)
- once we have a Gaussian fit, we can use standard methods (Fisher information/Wald) to estimate parameter CIs [still assuming iid Gaussian]
- for CIs on predictions or forecasts, use delta method or [whatever it's called when we sample from a MVN sampling distribution of parameters; par. bootstrap? PPI?] or importance sampling or ...
- now we are assuming independence, so measuring cumulative variables will result in incorrect inference (cf. ODE papers in epidemiology that do this carelessly??)

EXAMPLE?

```
update(tumor_exp_fit,
       observation=list(volume ~ dnorm(mean=A, sd=sigma)),
       start=c(coef(exp_fit), sigma=1),
       par=c("r", "A0", "sigma"))
```

2.3 non-Gaussian errors

Once we've gotten this far, it's fairly easy to extend the model to cover conditional distributions other than Gaussian. For example, we often have count data that would naturally be modeled by a Poisson or negative binomial distribution.

EXAMPLE? (does `fitode` have an `update` method? that would be a cute way to make compact examples)

This doesn't raise very many additional problems.

3 Increasing robustness and efficiency

So far we've assumed that the numerical methods should Just Work. People often have really crappy data (e.g., very short time series, complex models). In R, we're typically depending on derivatives computed by finite differences.

3.1 Sensitivity equations

Don't forget to cite Raue et al. (2013).

3.2 Link functions

aka, transforming parameters to unconstrained scales. Don't know if this deserves its own section, but it's a good idea. Generally improves scaling, applicability of Wald approximations, robustness.

The only caveat (besides increased complexity) is that if a parameter ever ends up on the boundary of its feasible space, then transforming to an unconstrained space will push

the optimum to infinity, leading to problems . . . box constraints may be better in this case, although Wald-based inference will break in any case.

3.3 Self-starting models

`fitode` doesn't do this at present; demote to an "other" paragraph?

3.4 Compiled models

`fitode` doesn't do this at present; demote to an "other" paragraph? See deSolve vignette on compiled gradient functions: also cf. `nimble`, `odeintr`, `cOde`, `dMod`, `rodeo`, others?

Speed is important when using simulation-based methods (SMC, ABC, etc.; see below) or when fitting a large number of series. (Sensitivity-based methods can help improve speed, but not as much as compilation.)

3.5 Multi-start methods

Slow (hence faster integration would be nice) but deals with multi-modality. Again cf. Raue et al. (2013): Latin hypercube, Sobol, etc.. They discuss diagnosis of multi-modality as well.

3.6 Other

Should we include regularization here? Box constraints are a cheaper/easier possibility. Regularization would make a nice lead-up to Bayesian methods.

4 Beyond frequentist models

⟨BMB: hard to decide on the best order: beyond-freq (i.e. Bayesian) then beyond-trajectories or vice versa? Reasons you would need these do overlap, although not completely⟩

Discuss advantages of Bayesian approach (priors, integration over uncertainty). Need to think about sampling (Gibbs, Metrop-Hastings, HMC . . .)

Stan, boms (??), `deBInfer`

5 Beyond trajectory-matching

5.1 gradient matching etc. (process error only?)

`CollocInfer`, papers by Hooker/Ellner/etc.; Kim McAuley

5.2 state-space models

Huge range here. Sequential Monte Carlo (pomp etc.). Kalman/extended Kalman filters? Heydari et al. (2014) (SDEs + Bayesian + Kalman filter)

Mention ABC/synthetic likelihood?

5.3 random effects/mixed models

What do we want to say here? Easier in Bayesian-toolbox contexts such as Stan. `nlmeODE`; Tornøe et al. (2004) . . .

6 Package comparison

7 Miscellaneous

- Appropriate citations to PKPD literature, e.g. `RxODE`, Wang et al. (2016) (although this looks like it is a simulation rather than fitting tool?). In many cases they use particular models with nonlinear but known solutions, e.g. two-compartment linear models.
- cite that mixed-ODE paper we reviewed.
- Discuss identifiability/estimability?

References

Heydari, J., C. Lawless, D. A. Lydall, and D. J. Wilkinson (2014, August). Fast Bayesian parameter estimation for stochastic logistic growth models. *Bio Systems 122*, 55–72.

Murphy, H., H. Jaafari, and H. M. Dobrovolny (2016). Differences in predictions of ODE models of tumor growth: a cautionary example. *BMC cancer 16*(1), 163.

Press, W. H., S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery (1992). *Numerical Recipes in C: The Art of Scientific Computing* (2nd ed.). New York: Cambridge University Press.

Raue, A., M. Schilling, J. Bachmann, A. Matteson, M. Schelke, D. Kaschek, S. Hug, C. Kreutz, B. D. Harms, F. J. Theis, U. Klingmüller, and J. Timmer (2013, September). Lessons Learned from Quantitative Dynamical Modeling in Systems Biology. *PLOS ONE 8*(9), e74335.

Tornøe, C. W., H. Agersø, E. N. Jonsson, H. Madsen, and H. A. Nielsen (2004, October). Non-linear mixed-effects pharmacokinetic/pharmacodynamic modelling in NLME using differential equations. *Computer Methods and Programs in Biomedicine 76*(1), 31–40.

Wang, W., K. M. Hallow, and D. A. James (2016). A Tutorial on RxODE: Simulating Differential Equation Pharmacometric Models in R. *CPT: Pharmacometrics & Systems Pharmacology 5*(1), 3–10.

Worschech, A., N. Chen, A. Y. Yong, Q. Zhang, Z. Pos, S. Weibel, V. Raab, M. Sabatino, A. Monaco, H. Liu, et al. (2009). Systemic treatment of xenografts with vaccinia virus GLV-1h68 reveals the immunologic facet of oncolytic therapy. *BMC genomics 10*(1), 301.