ImpulseDE2 modeling of TC RNA-Seq data with random initializations

Ethan Ashby

5/29/2020

Overview

Sigmoidal and Impulse models are S-shaped curves that are descriptive of a variety of biological responses, and are parameterized by biologically meaningful parameters. Particularly, we are interested in how these models can be applied to gene expression measurements over time in response to environmental stimuli.

Here we focus on an algorithm (ImpulseDE2) for fitting sigmoidal and impulse models to gene-wise expression measurements for the purpose of differential expression testing. The model fits gene-wise expression models using a non-linear optimization approach and then compares the signficance of each model to a constant model (no signal) and to each other using a log ratio test. The data we are working with is a 150 minute time-course RNA-sequencing experiment of *E. coli* cells in response to cell starvation. We are particularly interested in the onset time (t_1) parameter of these models, as they can be used as a proxy for when genes are 'turning on' in response to a stimuli and can be used to test the hypothesis that sensitivity/insensitivity to RpoS could be a mechanism for controlling order and timing or gene expression in response to stress.

For some reason- probably due to the sparseness of our time point measurements and the algorithm's poor initialization procedure for the parameter values- the algorithm is reaching local minima in its optimization procedure, leading to models that do not accurately reflect the data they are trying to fit.

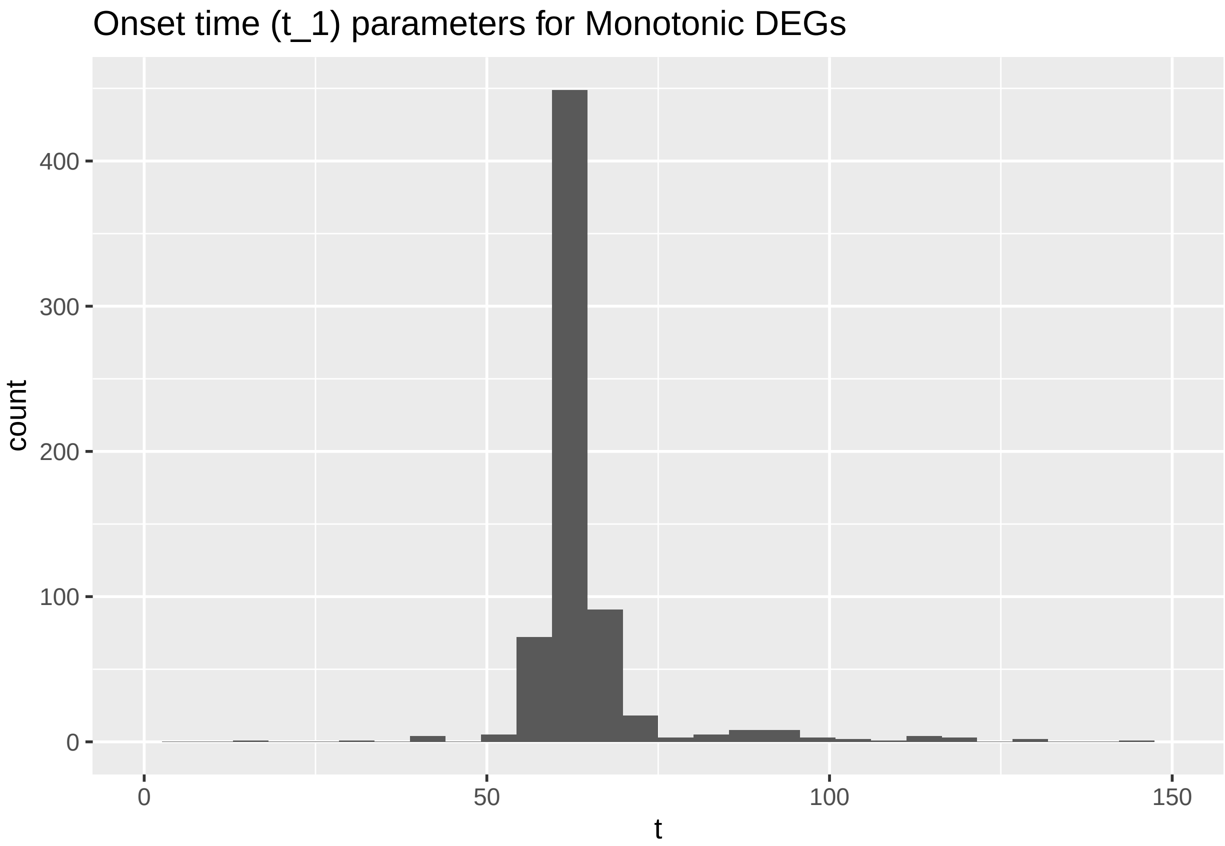
This code will walk you through the problem we encountered and updates I've made to the initialization procedure to get better (albeit not perfect) models that should provide more accurate estimation of the onset time parameters.

Out-of-the-box ImpulseDE2 has problems modeling gene expression profiles

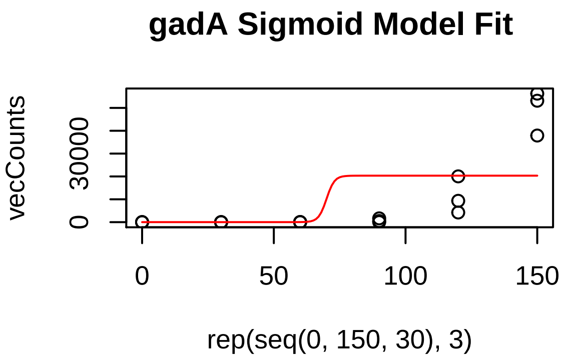
When we fit models to genes using the out-of-the-box ImpulseDE2 approach, we get lists of genes identified as monotonically differentially expressed (i.e. can be described by the sigmoid model), transiently differentially expressed (i.e. can be described by the impulse model), or ambiguous. When we take the monotonically differentially expressed genes and extract their sigmoidal model parameters, and plot a histogram of the t_1 (onset time) parameters, the approach suggests that the vast majority of genes (>90%) are turning on around 60 minutes. Initially, we believed that this suggested that the hypothesis that sensitivity was controlling timing was unsupported. In fact, it looked like large groups of genes were all turning on at around the same time.

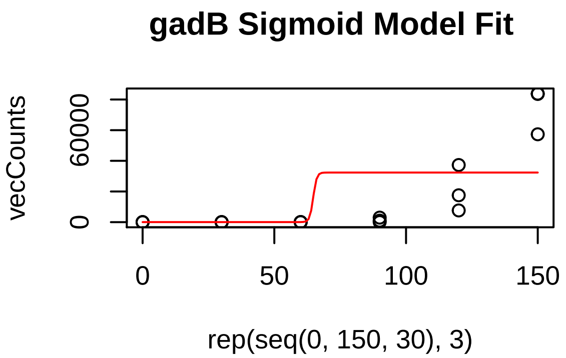
But a look closer shows that the models fit by this ImpulseDE2 approach were actually quite poor in terms of their fit to the data. Shown are plots of three related genes, gadA, gadB, and gadC, which show clear monotonic behavior that should be explainable by the sigmoid model. However, the sigmoid model fits get it all wrong, for these genes and presumably for others! The onset time, slope of expression induction, and final expression levels are not at all reflected in our data. So we cannot ascribe any confidence to the out-of-the-box parameter estimates… our quest for better modeling of these functions continues.

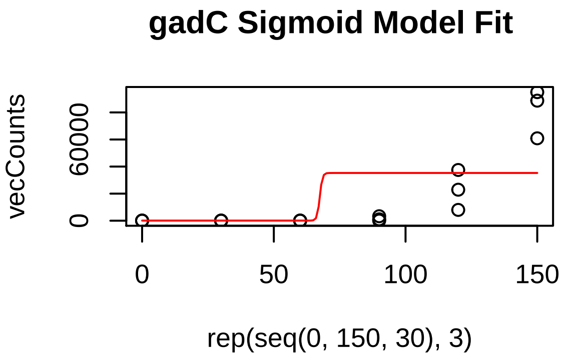
```
## gene-wise dispersion estimates

## mean-dispersion relationship
## final dispersion estimates

## [1] "Corrected 1 DESEq2 dispersion estimates which to avoid variance overestimation and loss of disc

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning: Removed 4 rows containing non-finite values (stat_bin).

## Warning: Removed 2 rows containing missing values (geom_bar).
```



Onset time (t_1) parameters for Monotonic DEGs







Random parameter initializations

Like you, I was surprised to see the ImpulseDE2 method work so poorly out of the box in fitting descriptive models to the data. I checked the source code and couldn't find an immediately wrong with their method, but I noticed that the way the authors created parameter initializations looked suspect. Here's how they initialized all the parameters:

$$\beta = 1, h_0 = \max(\bar{x}_0, \bar{x}_{30}), h_1 = \min(\bar{x}_{60}, \bar{x}_{90}, \bar{x}_{120}, \bar{x}_{150}), t_1 = 60$$

Where $\bar{x}_i$ denotes the mean expression at time point i. 60 was chosen as the initialization value for t_1 since that was the "middle timepoint". While these initialization values are not inherently bad, the challenge is that nonlinear curve fitting algorithms in high throughput settings generally converge to local minina that are undesirable.

In order to circumvent this challenge, many randomly generated initialization values can be generated prior to performing the optimization procedure. Then output models comparing the resulting models for the highest log-likelihood can be used to select the best-fitting solution across the initialization grid.

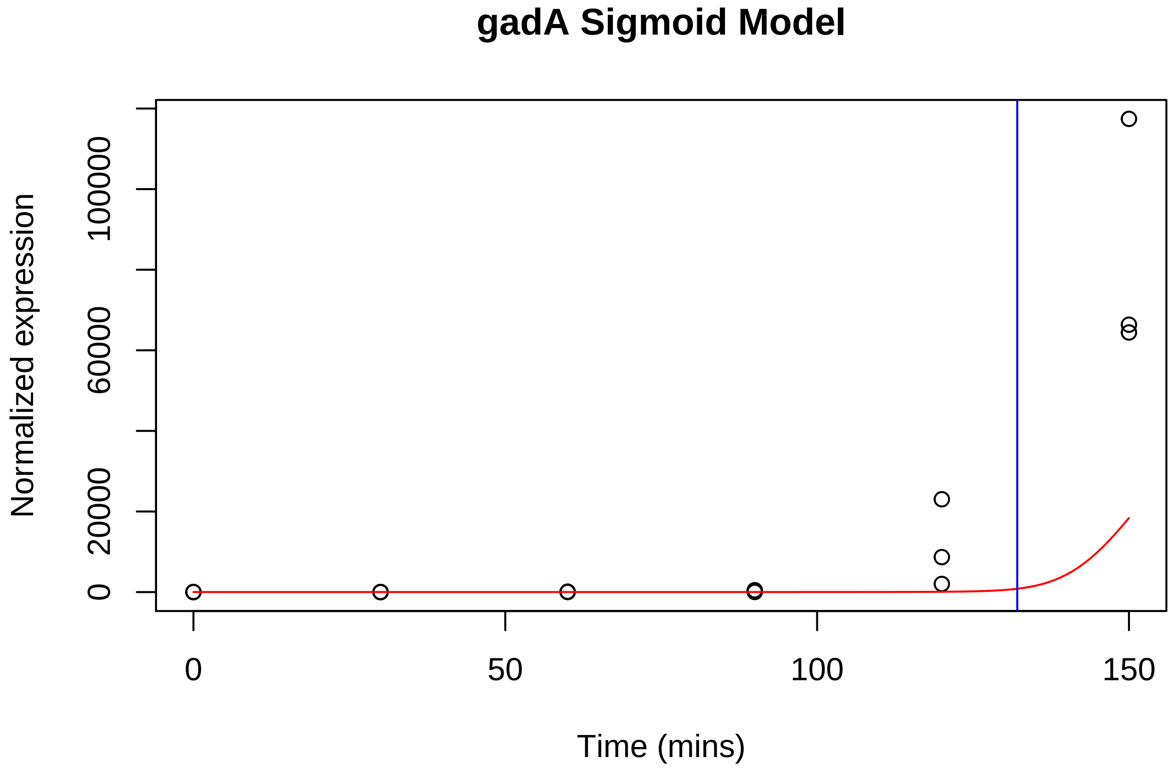
All the functions in the ImpulseDE2 are loaded into the global environment. Here are a few that will be useful.

- evalLogLikSigmoid()
  - In the 'srcImpulseDE2_CostFunctionsFit.R' file. Evaluates the log likelihood of a sigmoid model with particular parameters. Critical part of the model fitting procedure
- evalSigmoid()
  - In the 'srcImpulseDE2_evalSigmoid.R' file. Computes values (read counts) of sigmoid model given parameters (Beta, log(h_0), log(h_1), t_1)
- estimateSigmoidParam()

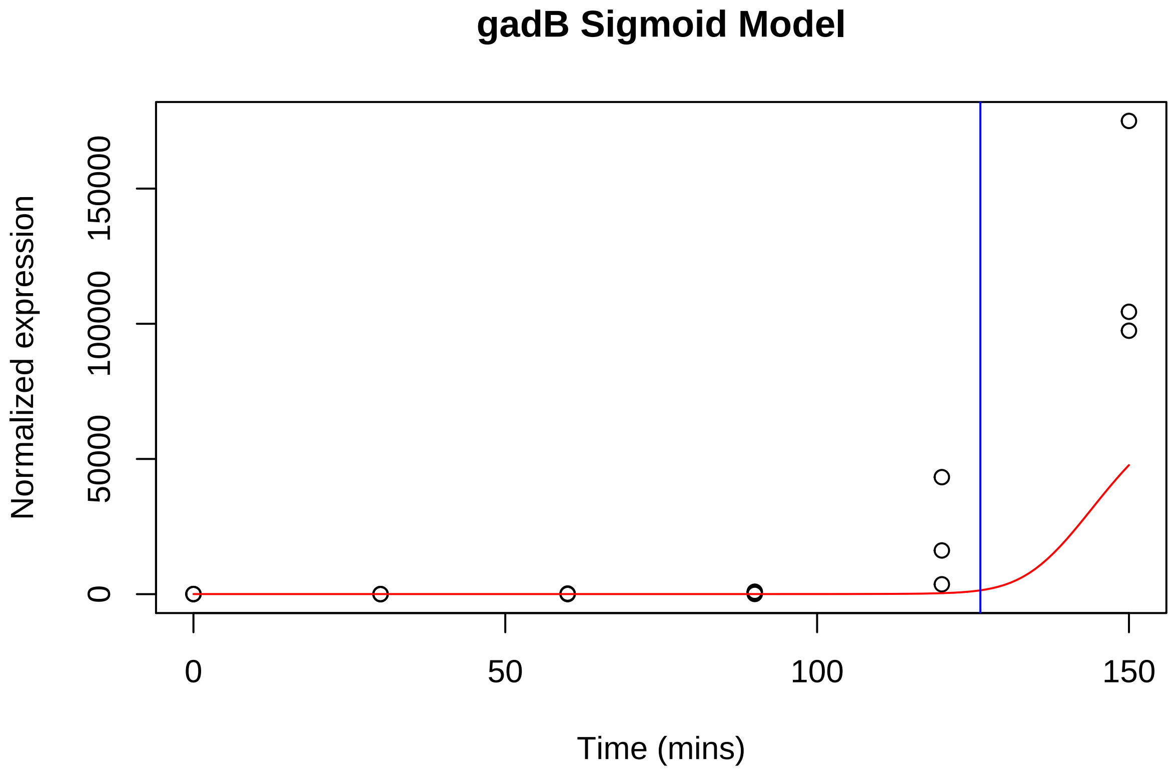
  - In the ‘srcImpulseDE2__fitSigmoid.R’ file. Generates initialization values (guesses) for sigmoid parameters.
- fitSigmoidModel()
  - In the ‘srcImpulseDE2__fitSigmoid.R’ file. Using optim() function from stats package with method=“BFGS”, fits the parameters to the data for a singel gene.
- fitSigmoidGene()
  - In the ‘srcImpulseDE2__fitSigmoid.R’ file. Wrapper for fitSigmoidModel() that determines whether the model should be up or downregulated.

Here is my attempt at improving the model by employing many random initializations. This method will be analogous to fitSigmoidModel() with an additional plotting feature. Arguments are explained below:

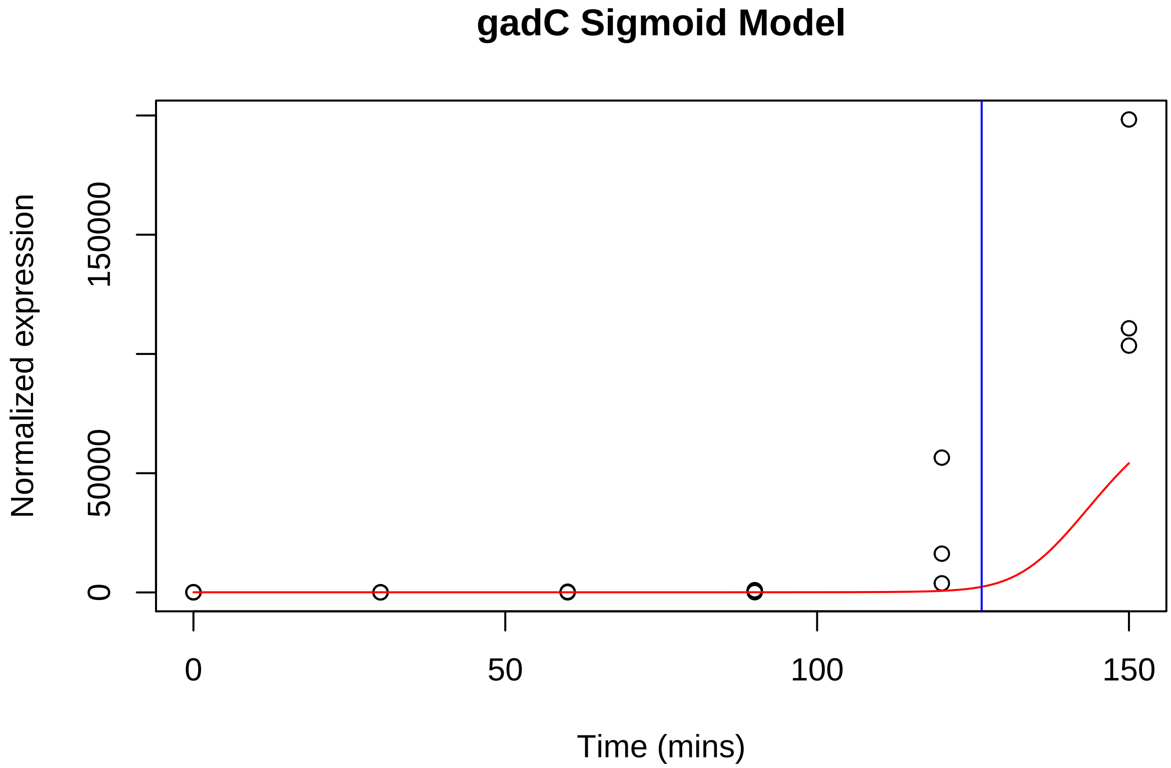
- gene=NULL: option to put in a gene of interest, will extract all useful information for you
- vecCounts: count data to fit model to (unnormalized)
- scaDisp: dispersion parameter for a gene
- vecSizeFactors: size factors for CASE samples (correspond to “JH01”)
- vecTimepointsUnique: vector of timepoints seq(0,150,30)
- vecidxTimepoint: vector connecting timepoints to counts rep(1:6,3)
- lsvecidxBatch=NULL: batch effects
- MAXIT=1000: maximum number of iterations in optimization
- init_num=200: number of parameter initializations that you want
- seed=47: random seed
- RELTOL=10ˆ-9: convergence tolerance
- verbose=FALSE: display messages during fitting?
- plot=TRUE: display a plot of the counts and fitted sigmoid function be displayed
- NOTE: method is now “L-BFGS-B” where I can put bounds on the parameters to be fit.



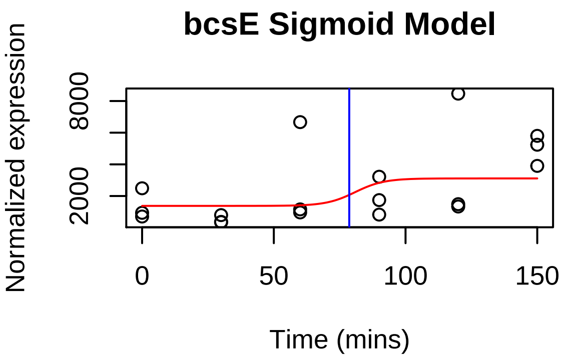
```
## [1]   0.1054577   2.5128489  10.9243257 132.0869029
```

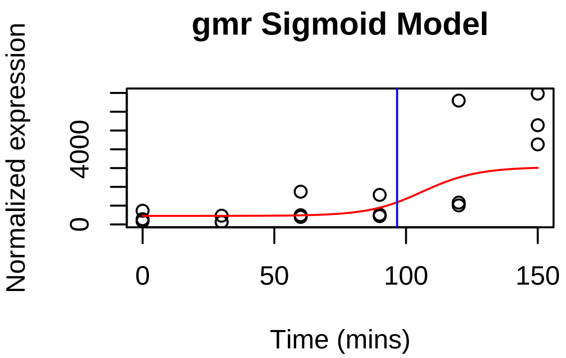


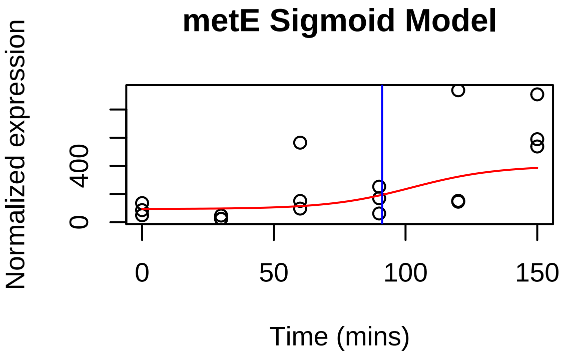
```
## [1]   0.1172034   3.2657577  11.2332274 126.1713304
```

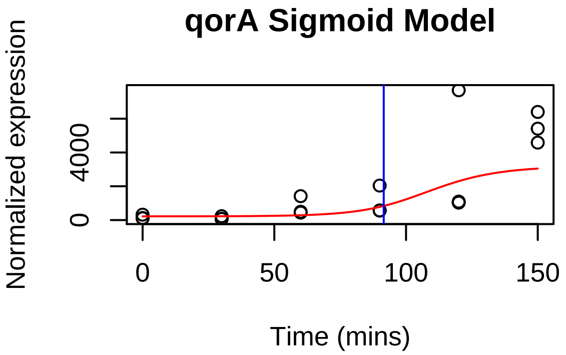


```
## [1]   0.1169405   4.1966794  11.3240408 126.3854052
```









As illustrated in these plots, the estimation of the onset time parameter (t_1 shown by the blue line on the plots) looks much more appropriate considering the shape of the data. However, the final expression value is still way off where it should be. The final optimization procedure should probably be adapted from this one, but this should give a pretty good idea of where the actual t_1 parameters lie.

I believe this method doesn't need a wrapper to decide whether an upregulated or downregulated fit is better, because the function will automatically generate initializations where $h_0 > h_1$ and other cases where $h_0 < h_1$, and the log likelihood selection criteria ensures that the best fit is returned.

Improvements to this method

1. Better estimation of the h_1 parameter: it looks like we're consistently missing the steady state expression below. Perhaps fixing the h_1 value by taking the mean at the final time point, and then optimizing the fit of the three remaining parameters would help improve the fit.
2. Randomly generated parameter estimates bounds. I randomly generated parameters like so:

   $$\beta = rand[0, 20], \log(h_0) = rand[0, 20], \log(h_1) = rand[0, 20], t_1 = rand[0, 150],$$

   More thoughout could be put into how we're generating these random initializations so that more useful outputs are created more often.
3. Look at the source code again to understand why log(h_0) and log(h_1) are being used and whether this is influencing the estimation of t_1. Also, I think there may be a problem in the source code where the function is being fit to normalized counts and the log likelihood is being evaluated for the unnormalized counts. This problem should be corrected.
4. Gene-wise model fitting is probably running into problems with overfitting. Is there a way to share information between genes for better parameter estimation? Perhaps an alternative parameterization of the sigmoid model (in Sicegar where there are only 3 parameters to be estimated) could be useful here.