Modeling Biological Intensity Measurements with Sicegar

Ethan Ashby

5/26/2020

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 (Sicegar) for fitting sigmoidal and impulse models to biological intensity measurements. This package was initially developed to fit S-shaped curves to fluorescence generated by GFP produced by viruses as they affected cells. However, this method is purportedly applicable to many high-throughput biological tasks, and hosts a number of advantages compared to the other modeling approach considered (ImpulseDE2). The model fits expression models using least squares and then compares AIC's between a constant fit and the models to determine whether there is any signal. If there is signal, the algorithm proceeds through a 7-step criteria (AIC cutoffs, thresholds for initial and final expression, convergence, etc) to determine if the fit is best described by a sigmoid, impulse, or ambiguous fit. 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.

My two chief concerns with this method are as follows:
1. A least squares fitting approach assumes a gaussian (normal) noise structure about the signal. These read count data may adhere to an approximately normal distribution in some cases but not in others (e.x. low counts). Therefore, applying a least squares fitting method to this scenario may be theoretically untenable.
2. This method is only capable of fitting models to upregulated profiles. The code must be adapted to fit downregulated profiles as well.

The chief benefits of this method are as follows:
1. The gaussian assumption (while possibly simplistic) is more interepretable and allows for the application of a bunch of statistically well-developed ideas downstream of model fitting. The algorithm outputs standard errors for all the parameters, which can be extremely useful in assessing how much confidence we have in the parameter estimates. 2. The models appear to produce much better fits that the ImpulseDE2 approach, even with random initializations. 3. Sicegar parameterizes the sigmoid model with only 3 parameters, helping to reduce overfitting.

How to work with Sicegar

Two functions that may be very useful in the sicegar are fitAndCategorize(), which fits a sigmoid and impulse (double sigmoid) model to the data and goes through the decision process to classify the gene as either sigmoid or impulse. figureModelCurves() plots these fits in plots that are compatible with ggplot2.

It's important to note that Sicegar does all it's modeling on scaled data; the expression and time are both converted to a [0,1] scale. The resulting optimized parameters can be backtransformed to their original units by using scaling parameters in the 'Model' object. Calling this fitted model returns a number of useful things:

the parameter estimates, scaling parameters, parameter estimates in the data's actual units, standard errors of the parameter estimates, likelihoods, asymptotes, and more.

```
## # A tibble: 18 x 2
##    feature                   number_of_genes
##    <chr>                               <int>
##  1 AS_CDS                               4386
##  2 AS_gene                                45
##  3 AS_IGR                               2476
##  4 AS_mobile_genetic_element              49
##  5 AS_ncRNA                               67
##  6 AS_origin_of_replication                1
##  7 AS_rRNA                                22
##  8 AS_sequence_feature                    11
##  9 AS_tRNA                                89
## 10 CDS                                  4386
## 11 gene                                   45
## 12 IGR                                  2476
## 13 mobile_genetic_element                 49
## 14 ncRNA                                  67
## 15 origin_of_replication                   1
## 16 rRNA                                   22
## 17 sequence_feature                       11
## 18 tRNA                                   89
```

```
## Joining, by = c("Geneid", "JH01_A01", "JH01_A02", "JH01_A03", "JH01_A04", "JH01_A05", "JH01_A06", "J
```

```
## Joining, by = c("Geneid", "JH01_A01", "JH01_A02", "JH01_A03", "JH01_A04", "JH01_A05", "JH01_A06", "J
```

```
## Joining, by = c("Geneid", "JH01_A01", "JH01_A02", "JH01_A03", "JH01_A04", "JH01_A05", "JH01_A06", "J
## Joining, by = c("Geneid", "JH01_A01", "JH01_A02", "JH01_A03", "JH01_A04", "JH01_A05", "JH01_A06", "J
## Joining, by = c("Geneid", "JH01_A01", "JH01_A02", "JH01_A03", "JH01_A04", "JH01_A05", "JH01_A06", "J
```

```
## estimating size factors
```

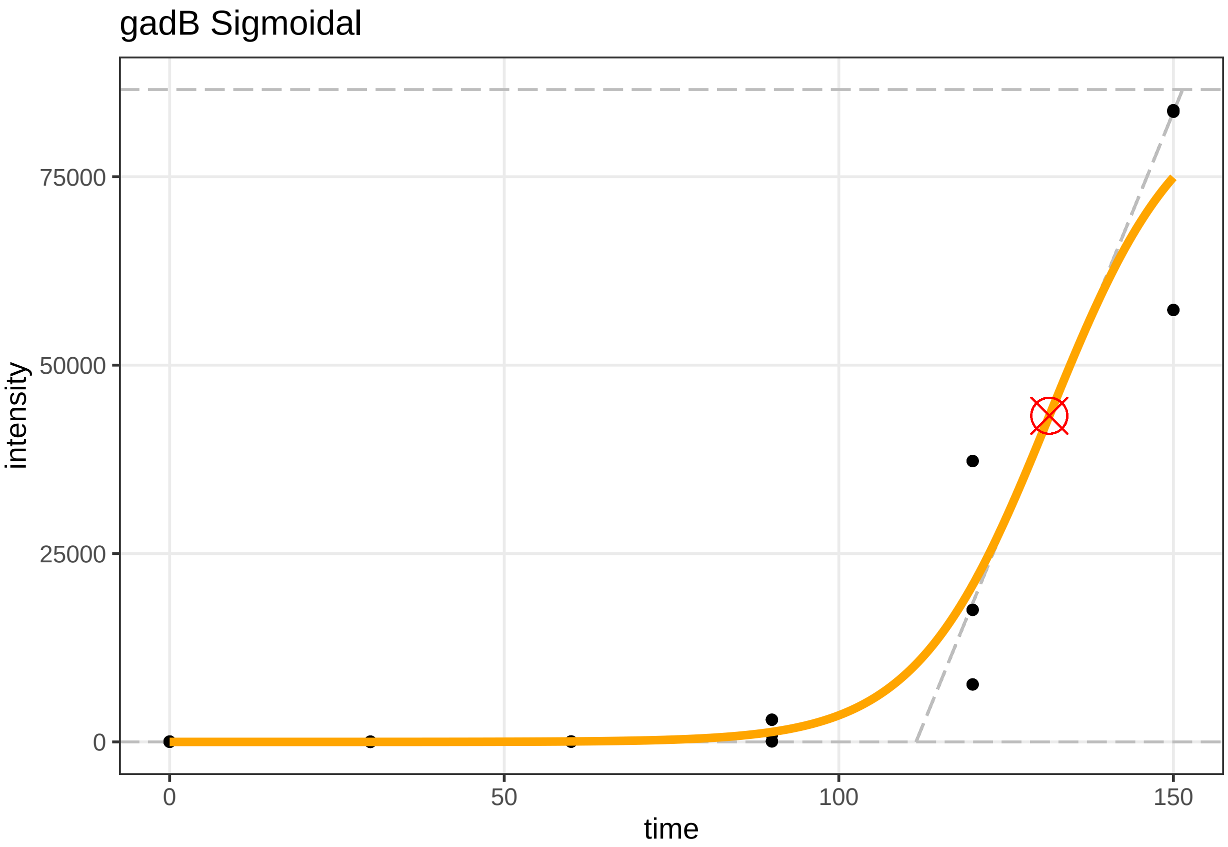
```
## estimating dispersions
```

```
## gene-wise dispersion estimates
```

```
## mean-dispersion relationship
```

```
## final dispersion estimates
```

```
## fitting model and testing
```



```
##                                      [,1]
## maximum_N_Estimate                   "1.03268"
## maximum_Std_Error                    "0.6261551"
## maximum_t_value                      "1.64924"
## maximum_Pr_t                         "0.1198779"
## slopeParam_N_Estimate                "15.06204"
## slopeParam_Std_Error                 "18.22164"
## slopeParam_t_value                   "0.8266016"
## slopeParam_Pr_t                      "0.4214176"
## midPoint_N_Estimate                  "0.8763859"
## midPoint_Std_Error                   "0.1484414"
## midPoint_t_value                     "5.903918"
## midPoint_Pr_t                        "2.895542e-05"
## residual_Sum_of_Squares              "0.131609"
## log_likelihood                       "18.72373"
## AIC_value                            "-29.44747"
## BIC_value                            "-25.88598"
## isThisaFit                           "TRUE"
## startVector.maximum                  "1.15839"
## startVector.slopeParam               "62.45002"
## startVector.midPoint                 "-0.07090508"
## dataScalingParameters.timeRange      "150"
## dataScalingParameters.intensityMin   "0"
## dataScalingParameters.intensityMax   "83828.85"
## dataScalingParameters.intensityRange "83828.85"
## model                                "sigmoidal"
## additionalParameters                 "TRUE"
## maximum_Estimate                     "86568.39"
## slopeParam_Estimate                  "0.1004136"
## midPoint_Estimate                    "131.4579"
## dataInputName                        NA
## betterFit                            "2"
## correctFit                           "20"
## totalFit                             "25"
## maximum_x                            NA
## maximum_y                            "86568.39"
## midPoint_x                           "131.4579"
## midPoint_y                           "43284.19"
## slope                                "2173.16"
## incrementTime                        "39.83525"
## startPoint_x                         "111.5403"
## startPoint_y                         "0"
## reachMaximum_x                       "151.3755"
## reachMaximum_y                       "86568.39"
```

For genes that are upregulated like gadB, the model appears to fit the data well and provides useful output regarding parameter values and their standard errors. However, for genes like talB (impulse and downregulated), the model doesn't know how to fit them well. This may warrant investigation: how to tweak the code to include downregulaetd genes. There are also many useful vignettes that come with the package that can explain the functions in more detail.

The last important thing to mention is that there are many transformations that have been developed to normalize RNA-seq data for linear modeling. In my attempts to normalize the data to apply the least squares approach, I successfully did psuedocounts and a voom transform. However, fitting models to these transformed data means that the sigmoid and impulse models no longer fit. More consideration may need to be invested in this step.