Improvements to Bayesian Gene Activity State Estimation

CRAIG DISSELKOEN

Department of Mathematics, Statistics, and Computer Science, Dordt College  
Sioux Center, IA 51250, USA  
Email: Craig.Disselkoen@dordt.edu

NATHAN HEKMAN

Department of Mathematics, Statistics, and Computer Science, Dordt College  
Sioux Center, IA 51250, USA  
Email: nthnhkmn@dordt.edu

NATHAN TINTLE

Department of Mathematics, Statistics, and Computer Science, Dordt College  
Sioux Center, IA 51250, USA  
Email: Nathan.Tintle@dordt.edu

Abstract goes here. As of this writing, it hasn’t been written yet. I’m mostly just writing this in order to check that the formatting is correct; so I needed a couple sentences to put here and stand in for the abstract.

# Intro / Motivation

General theme: Explain insufficiencies in Bayesian 1.0; motivate need for our new methods, using specific examples where appropriate

## Bayesian 1.0

Review Bayesian 1.0 paper, its overall goals and motivations

## Operon confidence levels

Explain how operons are not all black-and-white or perfect confidence; and how this motivates incorporating operon confidence levels for the Bayesian method

## Outlier handling

Our overall mixture-modeling approach is designed to detect distinct clusters of expression data in each gene or operon, and specifically to identify an ‘active’ and an ‘inactive’ cluster, if they exist. By nature, this approach is highly sensitive to outliers – single data points, or especially small groups of data points located far from all the others. Depending on the distance from the rest of the gene or operon’s expression data, the distribution of that data, and other factors, often a small group of ‘outliers’ can have an enormous effect on the activity state estimates for the entire gene or operon.

For example, consider gene rhaB, whose expression data is shown in Figure 1(a). It has a series of outliers on the low end, which, as is clearly visible from the *MultiMM* fitted distribution in Figure 1(b), completely dictate the entire interpretation of the gene’s expression data. Specifically, the interpretation is that these outliers constitute the ‘inactive’ cluster, and in all of the other experimental conditions in the expression data, rhaB is ‘active’.

Sometimes this high sensitivity to outliers may be desired – perhaps the interpretation for rhaB given above and in Figure 1(b) is correct, and likewise in similar cases. It’s possible a gene could be ‘active’ in almost all the experimental conditions in the given expression data, and ‘inactive’ in only a few choice experimental conditions, perhaps where the experimental conditions targeted it; or vice versa. Other times, outliers in the observed expression data may result from measurement errors, equipment malfunctions, poorly controlled experiments, or other factors. It is difficult to computationally distinguish between outliers arising from the various possible causes, so it seems necessary to mitigate their influence as a precaution. This is a conservative approach, and may result in reduced sensitivity when a gene or operon actually does have a very small ‘active’ or ‘inactive’ cluster (and interpretations such as that in Figure 1(b) are actually correct). However, the conservative approach should be more robust to outliers resulting from non-biological causes; mitigating the prospect that these erroneous measurements could dictate the activity estimates for the entire gene or operon is well worth the reduced sensitivity to very small clusters in the expression data.

## Hedging changing-state classifications

In Bayesian 1.0, the *UniMM* and *MultiMM* methods performed an initial screening procedure to determine which genes appeared to be changing state, and which did not. In this paper we will refer to this as the “Classification Procedure”. Following the Classification Procedure, one of two very different approaches for generating activity state estimates was used, based on whether the gene (*UniMM*) or operon (*MultiMM*) was classified as “changing-state” or “not-changing-state”. The Classification Procedure used the Bayesian Information Criterion (BIC) to assess the fit of a 1-component (univariate, in the case of *UniMM*, or multivariate, in the case of *MultiMM*) Gaussian mixture distribution vs. a 2-component equal-variance mixture distribution. The 2-component distribution was assumed to be the better fit unless the evidence was statistically significant in favor of the 1-component model, as indicated by a BIC difference of at least 12 points between the two models, following [Raftery et al]. Then, genes for which the 2-component distribution was a better fit (or evidence was not statistically significant in favor of the 1-component model) were classified “changing-state”, and genes for which the 1-component distribution was a statistically-significantly better fit were classified “not-changing-state”. An alternate Classification Procedure would simply classify genes and operons based on the best BIC between the 1- and 2-component models, without assessing statistical significance. Hereafter, the Classification Procedures with the 12-point requirement (as published in [Bayesian 1.0]) will be referred to as *UniCP(12)* and *MultiCP(12)* in their gene-based and operon-based forms respectively; and the versions based on simply the best BIC will be referred to as *UniCP(0)* and *MultiCP(0)*. Furthermore, we will define the versions of *UniMM* utilizing the *UniCP(12)* and *UniCP(0)* Classification Procedures as *UniMM(12)* and *UniMM(0)* respectively; and we will likewise define *MultiMM(12)* and *MultiMM(0)*, utilizing *MultiCP(12)* and *MultiCP(0)*. In all respects other than their Classification Procedure, *UniMM(12)* and *UniMM(0)* are identical, as are *MultiMM(12)* and *MultiMM(0)*.

For some genes/operons, the two very different approaches for generating activity state estimates (chosen as a result of the Classification Procedure) produce similar estimates, so that regardless of how the gene/operon was classified, the end result is similar. For other genes/operons, the two approaches produce wildly different estimates, so that the classification of the gene/operon has an enormous impact on the generated activity state estimates. This is not necessarily undesirable behavior; the reason for the existence of the Classification Procedure in the first place is an acknowledgment that the changing-state approach may be wildly wrong for genes that are not changing state, and vice versa. However, it does produce some undesirable instability.

For example, consider gene ymjA, whose expression data is shown in Figure 2(a). This gene has nearly identical BIC (a difference of only 0.7 points) between a 1-component and a 2-component model, indicating that the classification is extremely borderline and uncertain. As it stands, the BIC is very slightly in favor of a 2-component model, so the *UniMM(0)* and *MultiMM(0)* gene activity estimates are generated based on the 2-component fit shown in Figure 2(b), producing the activity estimates shown in Figure 2(c). On the other hand, if the not-changing-state approach had been used, the imputation process would infer from the generally-low nature of ymjA’s expression data that it is more likely ‘inactive’ than ‘active’ in a majority of experiments, producing the activity estimates shown in Figure 2(d). Furthermore, note that in both Figure 2(c) and Figure 2(d), the generated activity state estimates indicate a high degree of confidence in their predictions; both approaches are quite confident in their respective conclusions, given their disparate assumptions. This confidence is misleading, as there is a significant amount of uncertainty present in the classification itself which should be reflected in the confidence of the activity state estimates.

In cases like this where the classification is so borderline and uncertain, and has such a large effect on the generated activity state estimates, the Bayesian 1.0 methods run an unacceptably high risk of making the wrong decision and erring wildly in their gene activity state estimates. Furthermore, this kind of unstable behavior, where a tiny change in the expression data could produce a massive change in the activity state estimates, is undesirable. Instead, we will introduce a modification to the algorithm that both (a) reduces this instability, providing a smoother transition between the two approaches rather than a sharp cutoff; and (b) “hedges” its estimates in cases of uncertain classifications, providing more balanced predictions and more accurately representing the uncertainty of the situation.

# Methods

## Real data sets

We use expression data comprising 4329 *E. coli* genes from 907 different microarray experiments in various conditions; details of this data and its processing are found in [Bayesian 1.0] and [the paper Bayesian 1.0 cited for this data]. *E. coli* operon predictions for 2648 operons, including 1895 single gene operons, were obtained from [Microbes Online], as in [Bayesian 1.0]. (OR: discuss RegulonDB and its operons, as well as strength-of-evidence.)

## Simulated data sets

For some experiments, we used simulated data obtained with the “*Uniform Method*” described in [Bayesian 1.0]. This simulated data is informed by the real *E. coli* expression data and operons described above, and designed to have similar properties, while having ‘known’ gene activity states (active/inactive). We will refer to this as the *Sim-Uniform E. coli* data.

In order to demonstrate our method’s performance in the face of uncertain operon classifications, we conducted several tests with alternate simulated data. For each of these tests, we simulated 100 3-gene sets over 907 experiments, where some proportion of the 100 gene sets were actually generated as operons (that is, from multivariate normal distributions and for inactive and active data points respectively, where is the 3x3 identity matrix), and the other gene sets were actually generated as independent genes (that is, each gene was independently generated from normal distributions and for inactive and active data points respectively). For each gene or operon, a random value between 0.2 and 0.8 was chosen for the mixing parameter. Finally, in each test we assigned confidence to each of the 100 gene sets. For example, in one of the tests we generated 70 gene sets that were actually operons and 30 gene sets that were actually independent genes, and assigned them all 70% confidence that they were an operon. In other words, in the simulation as with real data, the confidence that a gene set is an operon is designed to be a posterior probability that that is actually the case. We will refer to this simulated data set as the *Operon Confidence* data.

## Alternate gene activity state estimation methods

In this paper, we will use as baselines for comparison the same *MT*, *TT*, and *RB* methods described in [Bayesian 1.0]; the *UniMM(12)* and *MultiMM(12)* methods introduced there as *UniMM* and *MultiMM*; and the slight variants *UniMM(0)* and *MultiMM(0)* introduced above. All of these methods take expression data and possibly information about operon structure, and produce gene activity estimates between 0 and 1 inclusive for each gene in each experimental condition. As noted in [Bayesian 1.0], *MT* is a special case of the method used for this purpose by [GIMME]; *TT* is similar but allowing for an uncertain region as proposed by [Shlomi et al]; and *RB* is a continuous analog to *MT*, in the spirit of [GIM3E]. More implementation details are available in [Bayesian 1.0].

## Validation and statistical analysis

For simulated data, true gene activity states are known by nature of the simulation. For real data, predictions of gene activity using FVA [cite] on the *E. coli* iJO1366 metabolic model [cite] were obtained from [Bayesian 1.0].

For the purposes of evaluating the performance of the methods listed above as well as improved methods which we introduce in this paper, we use a variation of the “squared deviation approach” described in [Bayesian 1.0]. Our metric, which we will call the “mean square alignment”, is defined as

where and represent sets (usually matrices) of probabilities, and the mean is taken over all and . Usually, will be a set of gene activity state estimates from a method under test, and will be either the true gene activity states (on simulated data) or the FVA predictions (on real data). This metric is general enough, however, to compare any two sets of probabilities, not just gene activity state estimates. Higher mean square alignment indicates improved agreement (that is, less difference) between the sets.

As an alternate method of validation, we compare these methods versus experimentally measured reaction fluxes on a published set of 79 genes in 29 separate experimental conditions, using the data and methods described in [Bayesian 1.0].

## Operon confidence levels

In [Bayesian 1.0], the *UniMM* approach ignored operon structure, treating each gene independently, whereas the *MultiMM* approach incorporated operon structure by assuming operon information was perfectly accurate – assuming that all genes in an operon are always in the same activity state in any given experimental condition. Generalizing both of these approaches, here we propose a new approach, *ComboMM*, which takes as input not only the (proposed) operon structure, but also for each operon a single number ranging from 0 to 1, representing the confidence that is actually an operon rather than simply independent genes. More formally, is the posterior probability that the genes in are always in the same activity state in every experimental condition (the assumption made by Bayesian 1.0’s *MultiMM*), with the alternative being that is composed of completely independent genes (the assumption made by Bayesian 1.0’s *UniMM*). Under this definition, *UniMM* is a special case of *ComboMM* where for all ; and *MultiMM* is a special case of *ComboMM* where for all .

*ComboMM* therefore takes as input:

1. expression data for each gene and experimental condition
2. a partition of the genes , representing the proposed operon structure (that is, a many-to-one assignment of genes to operons, with every gene assigned to exactly one operon)
3. for each operon , a number ranging from 0 to 1 (explained above)

*ComboMM* first runs *UniMM* on the entire set of expression data (all genes and experiments), producing gene activity estimates for each gene and experimental condition , and then runs *MultiMM* on the same expression data, producing gene activity estimates . For each gene and experimental condition , *ComboMM*’s output gene activity estimates are then given by

where is the operon containing .

## Outlier handling

We propose a variety of methods for mitigating the influence of outliers in the expression data for a given gene. All of these involve some kind of preprocessing on each gene’s expression data, and possibly slight post-hoc adjustments to the activity state estimates for the gene.

Some of these outlier handling methods involve removing certain data points in the expression data as part of the preprocessing step, either for being too high or too low; we will call these data points “removed-high” and “removed-low” respectively. We impose the condition, implicit in [Bayesian 1.0], that any proposed method for generating activity state estimates must always produce valid activity state estimates between 0 and 1 for every gene in every condition; no methods are ever allowed to simply “drop” genes or conditions and produce no activity state estimate. This is partly to ensure fairness of comparison between methods. The following method-specific implementation details describe how these “removed” data points are handled by each method, and what activity state estimate these data points are eventually assigned:

* *MT*: “Removed” data points are not considered when determining medians. “Removed-high” data points will be assigned an activity state estimate of 1, and “removed-low” data points an estimate of 0. This maintains the invariant that all estimates produced by *MT* are either 0 or 1.
* *TT*: “Removed” data points are not considered when determining percentiles. All “removed” values will be assigned an activity state estimate of 0.5. This indicates our uncertainty surrounding these “removed” values, as we are unsure of the cause for any given extreme value, whether it is biological or non-biological (e.g. equipment malfunction). This also maintains the invariant that all estimates produced by *TT* are either 0, 1, or 0.5.
* *RB*: “Removed” data points are not considered when determining ranks, and are assigned an activity state estimate of 0.5.
* *MM* methods of all types (including *UniMM*, *MultiMM*, and *ComboMM*, with any Classification Procedure – e.g. both *UniMM(12)* and *UniMM(0)*, etc.): “Removed” data points are ignored for the entire Classification Procedure, model fitting procedure(s), and imputation procedure as appropriate. These procedures will produce, for each gene and experimental condition which has not been removed, activity state estimates as previously described. Then, each “removed-high” data point , where is the gene and is the experimental condition the point belongs to, will be assigned an activity state estimate equal to , that is, the highest activity state estimate assigned to any non-removed data point in the same gene. Likewise, each “removed-low” data point will be assigned the lowest activity state estimate assigned to any non-removed data point in the same gene. Also note that for procedures which treat operon observations as a unit (e.g. *MultiMM*’s procedure for changing-state operons, or *MultiMM*’s Classification Procedure itself), if in any given experimental condition any gene *G*’s expression value has been “removed-high”, then the expression values for all other genes in *G*’s operon will be “removed-high” in that experimental condition as well; and likewise for “removed-low” data points.

The outlier-handling approaches we will consider are the following:

* *Num(n)*: This approach, following [cite], simply removes the highest and lowest *n* observations from each gene. For instance, *Num(5)* removes the highest 5 observations and the lowest 5 observations from each gene. The removed observations will (in general) correspond to different experimental conditions for each gene. This approach is easy to implement and avoids drawing any conclusions whatsoever from the expression data prior to outlier handling.
* *Pct(p)*: This approach is nearly identical to *Num* except it removes the highest and lowest *p*% of observations from each gene; that is, the number of removed observations is dependent on the number of experimental conditions in the expression data.
* *SD(d)*: For each gene, this approach first calculates the mean and the standard deviation of the expression data for that gene, and then removes any observations greater than or less than . This is a standard approach for outlier determination in statistical analysis; see for instance [cite].
* *Wins(d)*: This approach, called “winsorization” [cite], is similar to *SD(d)*, but does not actually remove any observations. Instead, any observations greater than are re-assigned the expression value , and likewise any observations less than are re-assigned the expression value . Then, the activity-state-estimate-generation method of choice is run as normal; since no observations were removed, no post-hoc adjustments to the activity state estimates are necessary.

## Hedging changing-state classifications

In order to reduce the instability of the *MM* approaches with regards to classification, and also to “hedge” activity state estimates in cases of uncertain classification, we propose modifications to the *MM* approaches.

First, we propose a generalized definition of “Classification Procedure”:

A *Univariate* (or *Multivariate*) *Classification Procedure* is a function that takes expression data for one gene (or operon respectively) and maps it to a value () representing an estimate of the posterior probability that () is truly changing state.

All of the Classification Procedures discussed already fit this generalized definition. In particular, *UniCP(0)* and *UniCP(12)* fit the generalized definition of *Univariate Classification Procedure*, and *MultiCP(0)* and *MultiCP(12)* fit the generalized definition of *Multivariate Classification Procedure* – although all four of these procedures always generate (or ) , indicating not-changing-state or changing-state respectively. In contrast, our generalized definition now allows for fractional and uncertain classifications (between 0 and 1), based on probabilities.

Next, we propose the following generalized *UniMM* procedure defined in terms of an arbitrary Univariate Classification Procedure, so that any Classification Procedure desired may be used. In particular, when using *UniCP(0)* or *UniCP(12)* in Step 1, this entire generalized *UniMM* procedure itself reduces to *UniMM(0)* or *UniMM(12)* respectively. However, this generalized *UniMM* procedure allows us to utilize new Classification Procedures that may take advantage of fractional and uncertain classifications; we will propose one such Classification Procedure shortly.

Step 1. For each gene , use any Univariate Classification Procedure to obtain a value representing an estimate of the posterior probability that is truly changing state.

Step 2. For each gene , generate gene activity state estimates using the changing-state approach as described previously in [Bayesian 1.0], and as employed by *UniMM(12)* or *UniMM(0)* for genes classified changing-state. We will call this set of estimates . Also extract the parameters , , , and describing the best-fit 2-component equal-variance Gaussian distribution for .

Step 3. For each gene :

1. Find parameters and describing the best-fit 1-component Gaussian distribution for .
2. Make a list of all genes where either ( and ), or ( and ) (following the *UniMM* procedure for genes it classified as “not-changing-state”, described in [Bayesian 1.0]). Let the length of this list be .
3. Generate sets of gene activity state estimates for , one for each gene , as if ’s expression data came from ’s best-fit 2-component distribution. Let these estimates be called for each gene .

Step 4. For each gene , the final activity state estimates are calculated as

We emphasize again that this generalized *UniMM* method completely reduces to either *UniMM(12)* or *UniMM(0)* if you apply the appropriate Classification Procedure in Step 1. However, it also gives us the flexibility to assign fractional values if we wish, indicating uncertainty in classification. This is the key feature of this generalized method.

We now propose a third Univariate Classification Procedure which will take advantage of the capability to assign fractional or uncertain values. First, we define the “Normalized Bayes Factor” of a gene by

where is the natural log of the Bayes Factor, given ’s expression data, of the 1-component Gaussian distribution over the 2-component (equal-variance) Gaussian distribution (calculated by the R package Mclust [cite] as the difference in log-likelihood between the two models), and is the number of experimental conditions in the expression data. (We normalize by because experiments on simulated data show that for truly changing-state genes – that is, true 2-component Gaussian distributions – values of scale linearly with , the number of observations of the distribution. Detailed results not shown.) Because of how is defined, higher values of indicate higher likelihood that the gene is not-changing-state; typical values range from around 0 (for genes which appear to be not-changing-state) to negative without bound (for genes for which we have increasing certainty they are changing-state). The lowest for our real expression data is ; notably, the three genes in the araBAD operon (whose behavior on this dataset was discussed at length in [Bayesian 1.0]) make up three of the four lowest ’s on our data.

Then, we propose a mapping from to (for each gene ) given by a simple piecewise-linear function

where is some constant, and is a constant less than 0. From experiments on simulated data (detailed results not shown), we empirically observe that for truly not-changing-state genes (that is, truly 1-component Gaussian distributions), rarely falls below for any value of ; hence, since is defined as , we propose that is a reasonable choice for the parameter . This means that genes with values of less than (values of less than ) will be assigned . We also propose as a reasonable choice for the parameter , which means that genes with values of greater than (values of greater than ) will be assigned . We will call this Classification Procedure *UniCP(NBF)*, indicating that estimates are obtained via , and the version of *UniMM* which employs it we will call *UniMM(NBF)*.

In the same manner, we propose a generalized version of *MultiMM* defined in terms of an arbitrary Multivariate Classification Procedure, such that *MultiMM(12)* and *MultiMM(0)* are special cases. We maintain the same relationship between the generalized *MultiMM* and the generalized *UniMM*, as *MultiMM* had to *UniMM* in [Bayesian 1.0]. In particular, steps 3 and 4 of the generalized *MultiMM* are unchanged from the corresponding steps of the generalized *UniMM*; *MultiMM* performs these steps at the gene level rather than the operon level, as in [Bayesian 1.0]. A more complete description of the generalized *MultiMM* follows:

Step 1. For each operon , use any Multivariate Classification Procedure to obtain a value representing an estimate of the posterior probability that is truly changing state. Then assign for each gene in . That is, all genes in an operon receive the same .

Step 2. For each operon , generate gene activity state estimates exactly as in Step 2 of the generalized *UniMM*, but using the *MultiMM* algorithm for changing-state genes rather than the *UniMM* one.

Step 3. Identical to Step 3 of the generalized *UniMM*, producing a list of length of sets of estimates for each gene .

Step 4. As in the generalized *UniMM*, activity state estimates are calculated for each gene as

Step 5. Following [Bayesian 1.0], for each operon and each experimental condition we average the activity state estimates for each gene in the operon in that experimental condition, and assign the average as the final activity state estimate for each of the operon’s genes. This step ensures that we maintain the invariant that, in *MultiMM* methods, genes in the same operon are guaranteed to have identical activity state estimates.

We also propose *MultiCP(NBF)* (and thus *MultiMM(NBF)*) analogous to *UniCP(NBF)* (and *UniMM(NBF)*). *MultiCP(NBF)* is a Multivariate Classification Procedure which makes assignments for each operon based on a multivariate version of the Normalized Bayes Factor, treating the operon as a unit. This multivariate Normalized Bayes Factor is defined for each operon by

where is the multivariate version of ; is as before; is the exponentiation function with base ; and is the number of genes in the operon. Like the definition of , the definition of arises empirically. Extensive experimentation on simulated data shows that for truly changing-state operons – that is, true multivariate 2-component equal-variance Gaussian distributions – values of scale linearly with , and values of scale linearly with ; detailed results not shown. can also be defined as a transform of , specifically,

We propose a piecewise-linear mapping from to analogous to the mapping from to proposed earlier. In our experimentation (again, detailed results not shown), we observe that for truly not-changing-state genes (that is, true 1-component multivariate Gaussian distributions), appears largely independent of , and rarely falls below (notice that for this reduces to our previous estimate of for single genes); hence, we propose that

is a reasonable choice for the parameter in this case, and

is a reasonable choice for the parameter , being merely the previous choice for in terms of (that is, ) transformed to the scale by the transform above. Note that the new values of the parameters and , and hence the mapping function from to , vary according to the size of the operon under consideration.

# Results

## Operon confidence levels

Performance of the *ComboMM* method on the *Operon Confidence* data described above, as a function of , the proportion of the 100 gene sets actually generated as operons and also the confidence parameter assigned to each of the 100 gene sets, is shown in Figure 3. *ComboMM* is as good as, or better than, both *UniMM* and *MultiMM* in every test.

[ Performance on real data? Haven’t yet acquired operon confidence levels for real data. Presumably this would come from RegulonDB, and we would create a mapping from RegulonDB’s evidence levels or types to numeric estimates of operon confidence. ]

## Outlier handling

Performance of each of the gene activity state estimation methods with each of the outlier-handling approaches, on the real data, is given in Figure 4.

We can see an illustration of the effects of outlier-handling by returning to rhaB, the gene from Figure 1. In Figure 1 we saw that in the raw expression data for rhaB (no outlier handling), the series of unusually low observations was interpreted as an ‘inactive’ cluster, with the entire rest of the observations then interpreted as ‘active’. Figures 5(a) and 5(b) show the expression data for rhaB after preprocessing from *SD(4)* and *Wins(4)*, respectively; the series of unusually low observations was partially removed (by *SD(4)*) or made less extreme (by *Wins(4)*). After *SD(4)* preprocessing, the expression data for rhaB (shown in Figure 5(a)) was classified by both *MultiCP(12)* and *MultiCP(0)* (and *MultiCP(NBF)*) as not-changing-state, whereas the original expression data for rhaB (shown in Figure 1(a)) had been classified by those methods as changing-state, leading to the interpretation explained earlier. This leads to substantially different gene activity estimates, as shown for example in Figures 5(c) and 5(d). If *Wins(4)* preprocessing is used rather than *SD(4)*, a third interpretation results (see Figure 5(e)) in which rhaB is classified changing-state, but not due to the outliers on the left; instead, the overall right-skewness of the expression data is interpreted as a large ‘inactive’ cluster and a small ‘active’ cluster, producing the estimates shown in Figure 5(f).

## Hedging changing-state classifications

Figures 6 and 7 provide insights into the performance of each of the Classification Procedures on the *Sim-Uniform E. coli* data. Figure 6 shows the generated values from each procedure for the genes which were truly changing-state; we see that *UniCP(NBF)* and *MultiCP(NBF)* perform the best among univariate and multivariate procedures respectively, with *UniCP(0)* and *MultiCP(0)* especially underperforming. In contrast, Figure 7 shows the generated values from each procedure for the genes which were truly not-changing-state. In this case *UniCP(12)* and *MultiCP(12)* clearly perform worst; *UniCP(0)* and *MultiCP(0)*’s high performance here is offset by their poor performance in Figure 6, indicating an undue bias towards low values. *UniCP(NBF)* and *MultiCP(NBF)* provide the best balance of performance between Figures 6 and 7.

Table 1 quantifies this intuition, showing that *UniCP(NBF)* and *MultiCP(NBF)* provide the best overall performance (measured as mean square alignment between the values and the true classifications) among univariate and multivariate procedures, respectively.

Overall performance of the gene activity state estimation methods based on these Classification Procedures, on the *Sim-Uniform E. coli* data, is shown in Figure 8(a), with performance on the truly changing-state and truly not-changing-state subsets shown in Figures 8(b) and 8(c).

Performance on real data, as measured by alignment with FVA predictions, is shown in Figure 9. As discussed in [Bayesian 1.0], the FVA calls are very conservative…

# Future Work

* Multiple alternatives or more flexibility in operon confidence statements, as mentioned in my comments in 2.5.  Operon confidence levels
* Improvements to the NBF <-> C mapping (I do have some ideas about this which should improve the algorithm in all cases but *especially* with small numbers of experiments, where the current version is completely unworkable)
* TRN modeling
* Incorporate more data sources, branch out to more organisms, etc
* Rigorously incorporate these estimates into a version of FBA