Lab Notebook

Alex Cope

August 2016

Contents

Grant Writing notes

1 NSF Proposals by David Smelser

1.1 What does OSP do?

Review proposals for compliance w/ federal, state, etc policies/regulations
Reviews all documents within proposal
provides feedback
Average NSF proposal is 60 pages
https://osp.utk.edu/

1.2 Cayuse SP

Tells institution as whole what you're working on (especially concerned with intellectual property) Once sections complete and submitted, notifies Department and college the the proposal needs to be approved Should arrive in OSP bt business days before submission date. Wilhelm says it takes him over an hour

1.3 Single copy documents

List of all people you have worked with in your career, including on projects, papers, textbooks, etc. Always evolving document. Lifetime conflict with advisors. Post-doctoral sponsors no longer officially considered in conflict. Sometimes, you have to look into pubmed to find where people work. Wilhelm says this is changing so a conflict is only between senior author and first author. Can petition to have certain people not review proposal based on things like completely opposite views on the science involved.

1.4 Project summary

One page, no more than that Overview, statement of intellectual merit, statement on broader impact Write in 3rd person Should not be an abstract

1.5 Biosketches

Should change to best reflect products most closely related to project proposal
Ex) If want funding for REU, maybe include papers that had undergraduate authors
Have any further questions, contact David Smelser at dsmelser@utk.edu

Mathematics and Statistics Notes

1 Chapter 7: Model Assessment and Selection

Taken from *The Elements of Statistical Learning*, 2nd ed, Hastie et al.

1.1 7.2 Bias, Variance, and Model Complexity

Let L(Y,$\hat{f}$(X)) be a loss function (can be squared error, absolute, etc), with X being a vector input, Y being a target vector, and $\hat{f}$(X) being estimated from some training set. Let $\mathcal{T}$ represent a training set. Then the test error is

$$Err_{\mathcal{T}} = E\left[L\left(Y, \hat{f}(X)\right)|\mathcal{T}\right]$$

which is the prediction error over an independent sample. Note that X and Y are drawn from their joint distribution. The expected test error is

$$Err = E\left[L\left(Y, \hat{f}(X)\right)\right] = E\left[Err_{\mathcal{T}}\right]$$

Err is better for statistical analysis and most methods estimate this quantity.

Training error:

$$\overline{err} = \frac{1}{N}\sum_{i=1}^{N} L\left(y_i, \hat{f}(x_i)\right)$$

As model complexity increases, training data is used more and adapts to more complicated information extracted from the model; however, this drives down the bias but increases the variance. Want to minimize expected test error.

Training error is not a good estimate of test error. In fact, a model with a training error of 0 is an overfit, meaning it will perform poorly when working with other data sets.

What if we are working with categorical data. The situation is very much the same. Let $L\left(G, \hat{G}(X)\right)$ (G = $\hat{G}$(X) = $argmax_k\ \hat{p}_k$(X)) be a loss function, where G is a categorical response taking one of K values. Let $p_k(x)$ be the probability that G = k given the input X. A loss function in this case might be 0-1 loss or something more complex (see pg. 221).

Training error for this type of data is

$$\overline{err} = -\frac{2}{N}\sum_{i=1}^{N} log\hat{p}_{g_i}(x_i)$$

The log-likelihood can serve as a loss-function for Poisson, gamma, exponential, and log-normal. From my understanding, this is essentially what ROC and similar models do when sampling gene expression values. Let $Pr_{\theta(X)}$(Y) be the density of Y, indexed by θ(X), then

$$L\left(Y, \theta(X)\right) = -2 \cdot logPr_{\theta(X)}(Y)$$

1.2 7.3 The Bias-Variance Decomposition

Let Y = f(X) + ϵ, where the average value of ϵ is 0 and the variance is equal to σ_ϵ^2. Then with a regression $\hat{f}$(X) at point x_0 with squared error loss, the expected prediction error is

$$Err(x_0) = IrreducibleError + Bias^2 + Variance$$

where the first term is the variance of the target around true mean f(x_0), the second is the amount by which the estimate differs from the true mean, and the third is the expected squared deviation of $\hat{f}(x_0)$ around its mean. Generally, the more complex $\hat{f}$, the lower the squared bias but the higher the variance.

The biggest thing to take from this section is that as model complexity goes up, squared bias generally decreases, but variance increases.

1.3 7.4 Optimism of the Training Error Rate

Typically, the training error is less than the true error because same data used to fit and assess the model.

Define the *in-sample error* to be

$$Err_{in} = \frac{1}{N}\sum_{i=1}^{N} E_{Y^0}\left[L\left(Y_i^0, \hat{f}(x_i)\right)|\mathcal{T}\right]$$

Y^0 indicates N new responses for N training points. Let the *optimism* and its corresponding average over training sets $\mathbf{y}$, ω, be

$$op \equiv Err_{in} - \overline{err}$$
$$\omega \equiv E_{\mathbf{y}}(op)$$

which is usually a positive term since the training error is usually driven down by fitting/assessment approach. Usually can only estimate ω. For many common loss functions (including squared error),

$$\omega = \frac{2}{N}\sum_{i=1}^{N} Cov\left(\hat{y}_i, y_i\right)$$

In summary, the expected value of the in-sample error is the sum of expected value of the training error over training sets $\mathbf{y}$ and ω,

$$E_{\mathbf{y}}(Err_{in}) = E_{\mathbf{y}}(\overline{err}) + \frac{2}{N}\sum_{i=1}^{N} Cov\left(\hat{y}_i, y_i\right)$$

If $\hat{y}_i$ is obtained via linear fit with d inputs, then,

$$\sum_{i=1}^{N} Cov\left(\hat{y}_i, y_i\right) = d\sigma_\epsilon^2$$
$$\rightarrow E_{\mathbf{y}}(Err_{in}) = E_{\mathbf{y}}(\overline{err}) + 2 \cdot \frac{d}{N}\sigma_\epsilon^2$$

Optimism increases linearly with number of inputs, but decreases as training set increases in size.

1.4 7.5 Estimates of In-Sample Prediction Error

Let d parameters be fit under squared error loss, then you get the C_p statistic,

$$C_p = \overline{err} + 2 \cdot \frac{d}{N}\hat{\sigma}_\epsilon^2$$

Note: $\hat{\sigma}_\epsilon^2$ is estimate of the noise variance.

Akaike information criterion (AIC):

Used to estimate Err_{in} when using log-likelihood loss function.

$$-2 \cdot E\left[logPr_{\hat{\theta}}(Y)\right] \approx -\frac{2}{N} \cdot E\left[loglik\right] + 2 \cdot \frac{d}{N}$$
$$\hat{\theta} = MLE(\theta)$$
$$loglik = \sum_{i=1}^{N} logPr_{\hat{\theta}}(Y)$$

where Pr_θ(Y) is a family of densities for Y (one of which is the true density).

When selecting a model via AIC, choose one that minimizes AIC. It is important to note that when using nonlinear models, d must be replaced with some measurement of model complexity.

In general, let $\mathcal{F}$ be a set of models f_α(x), where α is a tuning parameter. Then,

$$AIC(\alpha) = \overline{err}(\alpha) + 2 \cdot \frac{d(\alpha)}{N}\hat{\sigma}_\epsilon^2$$

Basically, AIC estimates the test error curve, so you want to find the $\hat{\alpha}$ that minimizes it.

Book notes that if basis functions are chosen adaptively, this whole thing collapses.

Example) You have p total inputs, but best fitting linear model takes $d < p$ inputs, optimism will exceed (2d/N)σ_ϵ^2.

1.5 7.7 The Bayesian Approach and BIC

Like AIC, applicable where fitting occurs via maximizing log-likelihood. Generic form of BIC:

$$BIC = -2 \cdot loglik + (logN) \cdot d$$

Under a Gaussian model (σ_ϵ^2 is known), $-2 \cdot loglik$ equals up to a constant $\sum_i^N (y_i - \hat{f}(x_i))/\sigma_\epsilon^2$. For a squared error loss, this is $N \cdot \overline{err}/\sigma_\epsilon^2$

$$\begin{aligned} &\rightarrow BIC = \frac{N}{\sigma_\epsilon^2}\left[\overline{err} + (logN) \cdot \frac{d}{N}\sigma_\epsilon^2\right] \\ &\rightarrow BIC \propto AIC \end{aligned}$$

BIC tends to penalize complex models, but as $N \rightarrow \infty$, the probability BIC picks correct model goes to 1.

2 Chapter 6: Model Checking

2.1 6.1 The place of model checking in applied Bayesian statistics

Sensitivity Analysis - how much do posterior inferences change in our model versus other models

2.2 6.2 Do the inferences from the model make sense?

Checking model via external validation
• Use model to make predictions, collect actual data, do they match up?
• Usually need to check model **before** getting new data

2.3 6.3 Posterior predictive checking

To measure discrepancy between models and data, define test quantity T(y,θ). Think test statistic in frequentist hypothesis testing, which depends only on data. To measure lack of fit of data with respect to posterior predictive distribution can be measured as p-value of test quantity. In a Bayesian framework, the p-value is

$$\begin{aligned} p_B &= Pr(T(y^{rep}, \theta) \geq T(y, \theta)|y) \\ &= \iint I_{T(y^{rep},\theta) \geq T(y,\theta)} p(y^{rep}|\theta) p(\theta|y) dy^{rep} d\theta \end{aligned}$$

which translates to the probability a set of replicated data could be more extreme than the observed (**how is this different from the frequentist's definition?**). A p-value can be calculated via simulations: drawing from joint posterior distribution, the p-value is simply the ration of

$$T(y^{reps}, \theta) \geq T(y, \theta^s)$$

where the y^{reps} are drawn from the simulated values of θ^s.

3 Chapter 7: Evaluating, comparing, and expanding models

3.1 7.1 Measures of predictive accuracy

Measures of predictive accuracy should be guided by the application: • point prediction - single value reported as prediction of future observation (mean squared error)
• probabilistic prediction - report inferences about $\tilde{y}$ that takes into account full uncertainty

over $\tilde{y}$

Log-likelihood, $\log p(y|\theta)$, is a probabilistic prediction. Proportional to the mean squared error if model is normal with constant variance. With large sample sizes, minimizing Kullback-Leibler information is the same as maximizing expected log-likelihood. Use log-likelihood because of its generality and because we are interested in summarizing the model fit to data (prior density is not relevant in computing predictive accuracy).

Ideal measure of model fit: predictive performance for new data (external validation).

Let:
- f be the true model
- y be the observed data
- $\tilde{y}$ be the future/alternative data

Out-of-sample predictive fit for new data point $\tilde{y}_i$ is

$$\begin{aligned}\log p_{post}(\tilde{y}_i) &= \log E_{post}(p(\tilde{y}_i|\theta)) \\ &= \log \int p(\tilde{y}_i|\theta)p_{post}(\theta)d\theta\end{aligned}$$

where $p_{post}(\tilde{y}_i$ is the predictive density and $p_{post}(\theta)$ is the posterior.

Expected out-of-sample log-likelihood, since future data unknown:

$$E_f(\log p_{post}(\tilde{y}_i)) = \int (\log p_{post}(\tilde{y}_i))f(\tilde{y}_i)d\tilde{y}$$

Generally, θ is not known, so can't get $\log p(y|\theta)$. Want to summarize predictive accuracy with

$$\log \prod p_{post}(y_i) = \sum_{i=1}^{n} \log \int p(y_i|\theta)p_{post}(\theta)d\theta$$

which is usually computed in practice by generating $\theta^s, s = 1, ..., S$ from $p_{post}(\theta)$ and then

$$\sum_{i=1}^{n} \log \left(\frac{1}{S} \sum_{s=1}^{S} p(y_i|\theta^s) \right)$$

3.2 7.2 Information criteria and cross-validation

Information criteria typically defined based on deviance $= -2 \log p(y|\hat{\theta})$

Interested in prediction accuracy for model validation and comparing models. **From pg. 170 of BDA: "When different models have the same number of parameters estimated in the same way, one might simply compare their best-fit log predictive densities directly,"...So in my current case, can I just compare the log-likelihoods of the different ROC results?**

Can approximate out-of-sample predictive accuracy using existing data:

1) *Within-sample predictive accuracy* - Naive estimate of log-likelihood for new data is log-likelihood of existing data. In general, overestimate

2) *Adjusted within-sample predictive accuracy* - Corrections for bias in computed log-likelihood based on existing data and is approximately unbiased. Corrects bias by subtracting off biases produced via number of (effective) parameters being fit. Reasonable, but correct at best only

in expectation

3) Cross-validation

Gelman et al recommend using WAIC as a measure of predictive performance, which is a more Bayesian approach than AIC or DIC. WAIC has the property that it averages over the posterior distribution, as opposed to conditioning on a point estimate.

$$p_{WAIC} = 2\sum_{i=1}^{n} \left(\log\left(E_{post} p(y_i|\theta)\right) - E_{post}(\log p(y_i|\theta))\right)$$

$$\textit{computed } p_{WAIC} = 2\sum_{i=1}^{n} \left(\log\left(\frac{1}{S}\sum_{s=1}^{S} p(y_i|\theta^s)\right) - \frac{1}{S}\sum_{s=1}^{S} \log p(y_i|\theta^s)\right)$$

They also DO NOT recommend using BIC, as the goal of this criterion is to approximate the marginal probability density of the data p(y) under the model, which can then be used to estimate relative posterior probabilities.

Computer Science Notes

1 Working with Pandas (Python package)

Biology Notes

1 Journal club: Integrative proteomic profiling of ovarian cancer cell lines reveals precursor cell associated proteins and functional status

Difficult to perform mechanistic studies with primary tissue, so need cellular models. Models need to be representative of tumor.

Genomic/Transriptomic analyses do not necessarily reflect proteomic profile.

Use streamline MS-based proteomics approach for molecular subtype characterization.

Many OvCa cell lines have been assigned to the wrong tissue of origin.

229,000 unique peptide sequences → 11,070 distinct protein groups (FDR ¡ 1

Look into Label-free protein quantification.

Used clustering based on expression of 8,500 distinct proteins quantified in at least 10 of 30 cell lines, which resulted in three main groups:
1) Group 1: cells lines previously reported to represent HGSOC cell lines based on genomic profiles
2) Group 2: unlikely HGSOC lines
3) Group 3: Note how some of the cell lines in this group are related, but doesn't address how all of them are related, so why is this not split up into multiple groups

Expected two cell lines to cluster in group 1 based on genomic similarity to HGSOC tumors, but instead were in group 3. They hypothesize that there is a discriminating feature of these tumors only visible at protein level.

Used principal component analysis at whole-proteome levels, which also showed three groupings. Determined 67 proteins with highest discriminating power between groups using feature selection and SVMs.

1.1 Sarvesh's presentation

Incorrect tissue origin: • cell lines representative of tumor
• 30 years since ovarian cell lines have been established
• cross-contamination

Goal of study: Integrated/streamlined MS-based proteomics approach for linking proteomic profiles from cell lines, tumor tissues and primary cells for evaluation of OvCa cellular model systems.

Functional signature are at protein and metabolite level.

Longer chromatography dilutes sample more, For narrow peak, MS might scan twice and then able to get new peak. Broader peak will be scanned by MS more, becomes redundant.

2 For Grant Writing

2.1 Environmental shaping of codon usage and functional adaptation across microbial communities (Roller et al, Nucleic Acid Research, 2013)

Environmental shaping of codon usage and functional adaptation across microbial communities (Roller et al, Nucleic Acid Research, 2013)

Metagenomics uses high-throughput sequencing to sequence genomes directly from environmental sample, but no culturing.
Problem: how do we know which genes go which microbe?
In these samples, there is usually a large degree of horizontal gene transfer → greater within species genetic diversity

Analysis pipelines focused in two directions:
1) Classify functions of genes based on orthology databases, rank importance based on abundance
2) Estimating phyletic distribution of microbial species based on similarity of searches against known sequences

Microbial communities share environmental factors, have similar tRNA pools to facilitate horizontal gene transfer. Fast growth rates introduce strong bias in codon usage at level of whole metagenomes. Claim CUB reflects selection for optimization of highly expressed genes.

Methods: Synonymous codon frequencies normalized per amino acid. Use intraclass correlation coefficient (ICC) to measure differences in codon frequencies between synonymous codons ($ICC = 0$ implies large difference).

Ribosomal protein genes served as reference set because highly expressed.

Calculated Measure Independent of Length and Composition (MILC):

$$MILC = \frac{\sum_a M_a}{L} - C$$
$$M_a = 2\sum_c O_c \ln \frac{O_c}{E_c} = 2\sum_c O_c \ln f_c g_c$$
$$C = \frac{\sum_a (r_a - 1)}{L} - 0.5$$

C serves as correction for estimation of overall bias in shorter sequences, r_a is number of synonymous codons for an amino acid. For each gene, calculated MILC compared to (meta)genome CUB and reference set defined by ribosomal proteins.

Found that within-species CUB is more variable between metagenomes than between species in same metagenome. CUB in metagenomes reflects CUB in single microbial species.

Environmental shaping of codon usage and functional adaptation across microbial communities (Roller et al, Nucleic Acid Research, 2013)

2.2 Community-wide analysis of microbial genome sequence signatures,(Dick et al, Genome Biology, 2009)

Use genome signatures for sequence classification, need to account for diversity in population, dynamics (like horizontal gene transfer), and environmental factors.

3 Origin and evolution of splicesomal introns, (Rogozin et al, Biology Direct, 2012)

Introns first: protein coding genes contained introns at very early stages of life, role in facilitating recombination of sequences

Introns-late: introns only emerged in eukaryotes, new introns accumulating

Core of spliceosome (five small nuclear ribonucleoproteins, or snRNPs) is conserved in well-characterized eukaryotes.

Intron densities vary widely among eukaryotes, but when plotted against intron length, two clusters do form. When density is greater than 3 introns per kbp, there seems to be correlation between intron density and length of introns.

Note that many evolutionary studies focus on highly conserved genes, so conclusions regarding intron stasis could be result of sampling bias.

In intron-poor genes of single-cell eukaryotes, intron position is biased towards 5 end. Selectionist interpretation is is these introns are more often involved in intron-mediated functions. The author's also say that in intron-rich genes are more uniformly distributed.

Analyses suggests mechanisms of intron loss and gain are likely to be different, reverse transcription involved in loss.

Continue reading at Reconstruction of evolution of exon-intron structure of eukaryote genes

Progress Tracker

1 8-25-2016

1.1 Reading

1. Li *et al.* Whole genome analysis of non-optimal codon usage in secretory signal sequences of *Streptoyces coelicolor*. *Biosystems*. 2006
2. Zalucki *et al.* Secretory signal sequence non-optimal codons are required for expression and export of b-lactamase. *Biochemical and Biophysical Research Communications*. 2007
3. Read Chapter 2 out of Lynch 2007.

At Cedric's recommendation, I re-ran the *E. coli* simulations with more samples (10,000 up to 50,000). He pointed out that the frequency plots for the codon usage were a little flat, which would be reflected in the traces not converging. The traces looked okay, but I decided it wouldn't hurt to rerun them with a few more samples. The plots looked almost identical, so it seems like 10,000 samples is sufficient for the *E. coli* genome. One of the peptides that looks particularly flat is for Lysine (K). The AAA codons is favored at a frequency of 0.8, regardless of logϕ. In *E. coli*, AAA is the best initiator of translation. I wonder if this plays a role in the strong bias for AAA.
Also ran for *C. bescii* and received the data on *E. coli* gene expression from the Li 2014 paper. Mallory Ladd (in Bob's lab) asked me to write some scripts from her for a project she is working on last week. She reminded me on today, so I decided to finish that up. Most of the scripts were based on work I did during my rotation in Bob's lab, but her files were in a different format. This required me to make modifications to my previous scripts, which turned out to be more annoying than I thought it would be.

2 8-26-2016

Finished up the scripts for Mallory and ran them for her. When I have a chance during the weekend, I will spend time learning how to use LaTeX, grading the short essays I assigned my students on the role of computation in unlocking biological systems, and finish preparing for lab on Monday.

Also, touched base with Steve regarding the protein clustering project he and I worked on during my rotation. He is collaborating on a similar project with Dr. Barerra over in BCMB and said he is planning on giving the project to an undergrad. After the undergrad has done some initial calculations, he said I'm welcome to help out on creating some "relatively low hanging" simulation code. Depending on how work is progressing with my other projects, I would like to help out on this where I can.

2.1 Goals for next week

1. Compare gene expression results from ROC simulations with results from Li 2014.
2. Look at gene expression results for the genes with predicted signal peptides. See if the majority of them are low expression genes (ie. non-optimal codon usage could be largely due to mutation biases).

3. Begin thinking about how to handle house-keeping genes in current models.
4. Resume independent-study of Bayesian Data Analysis.

3 8-29-2016

Next steps for analysis:
1) Fit model to genome consisting of genes with the first 35 amino acids removed, which should eliminate most of the signal peptide components. I would expect that if signal peptides have evolved under different selective pressures, then the overall model fitting would improve. However, given the large size of most genes relative to the signal peptides, I don't know how much impact removing these will have even if the signal peptide region and the rest of the gene are evolving differently.

2) Fit model to just genes with predicted signal peptides and do a separate model fitting to genes without predicted signal peptides. If genes with signal peptides are evolving differently, I would expect to see a drop off in the fitting relative to the genes without signal peptides. The number of genes with predicted signal peptides is roughly 10 percent of the E. coli genome (425 genes), so I don't know if this will be enough to get accurate fittings. Maybe I could create a random subset of 425 genes without signal peptides and fit the model to this subset in order to eliminate size as a variable in the analysis.

3) I would like to perform analysis with the mixture models that we discussed yesterday. It makes sense to me to treat genes with signal peptides vs those without as separate subpopulations within the genomes. I'm also wondering if it would be possible to go a level deeper in this analysis and treat individual codons as a member of a signal peptide vs a non-signal peptide. To me, this seems like a more accurate approach since it is the regions within the gene that could be evolving differently. However, my understanding of mixture models is limited and my knowledge of the current implementation in ROC even less so. Currently reading Gelman's chapter on Finite Mixture Models in order to improve my understanding.

4 8-30-2016

Submitted the above to Mike here are his responses.

1) Only one way to find out. Key question is how will you compare the quality of the model fits?

Me: Read up on model checking in Gelman's book

2) Creating a 'control' set makes sense, but note that the quality of the fit is quantitatively described by the posterior parameter intervals (more info, tighter intervals) and measures of model fit based on the unscaled probabilities of the MCMC samples.

3) For the first part, I would agree this would be a good step. If you use the mixture model approach, you can initially designate genes into particular category and let the algorithm update these designations. Preliminary work suggests that using estimates of ϕ's when fitting the model helps with the categorization.

Me: Have this from Li et al (2014, Cell).

For the second part, I agree that trying to apply separate models to different parts of the gene would be nice. How to do this in the current framework will take some thought. Note that I am interested in a similar type of analysis at the level of separate introns for alternatively spliced genes.

Completed a run of the truncated genome, but a problem occurred when trying to plot these functions. Mike suggested I look into writing the objects to a file. Looked into the parameterObject.r class and found a writeParameterObject function, which seems to accomplish this task. Added it to my standard template script for running ROC. Now if something happens

when plotting or I want to go back to do more analysis on a run, this object will be saved to a file.

Mike noted that I need to understand where the sources of the data come from. Li et al (2014, Cell) performs their subexperiments using ribosome footprinting. The mRNA levels they provide are RPKM values, which are normalized by the length of the gene.

5 8-31-2016

Towards the end of the day, got the following error while attempting to fit ROC to a genome consisting of only genes containing signal peptides. When the the function to plot the CUB plot was called, I got a memory allocation issue with the following traceback:

*** caught segfault *** address 0xfffffffffffffff8, cause 'memory not mapped'
Traceback:
1: .External(list(name = "CppMethod__invoke_notvoid", address = ¡pointer: 0x2cdce60¿, dll = list(name = "Rcpp", path = "/usr/lib/R/site-library/Rcpp/libs/Rcpp.so", dynamicLookup = TRUE, handle = ¡pointer: 0x2df0a00¿, info = ¡pointer: 0x7f3a5ebf9860¿), numParameters = -1L), ¡pointer: 0x238f690¿, ¡pointer: 0x23a9af0¿, .pointer, ...)
2: genome$getGenomeForGeneIndices(genes.in.mixture, simulated)
3: plot.Rcpp_ROCModel(model, genome, samples = samples * 0.1, mixture = 1, main = "E.coli Codon Usage Plot")
4: plot(model, genome, samples = samples * 0.1, mixture = 1, main = "E.coli Codon Usage Plot")

I ran this on Gauley with the most up-to-date version of the RibModel. I moved this file over to my computer with a slightly old version and did not get this error. I sent Holis all the information I could in hopes that he could figure out what is going on.

6 9-1-2016

Based on my conversations with Mike, it seemed like if I wanted to continue a fitting, I could load Parameter and MCMC objects from previous runs. I have no problem doing this with just a MCMC object, but if I try to continue a fitting using a loaded Parameter object, I get another memory allocation issue with the following traceback:
*** caught segfault *** address (nil), cause 'memory not mapped'
Traceback:
1: .External(list(name = "CppMethod__invoke_void", address = ¡pointer: 0x2715d50¿, dll = list(name = "Rcpp", path = "/usr/local/lib/R/site-library/Rcpp/libs/Rcpp.so", dynamicLookup = TRUE, handle = ¡pointer: 0x28a8f20¿, info = ¡pointer: 0x7f02b49b4b40¿), numParameters = -1L), ¡pointer: 0x406d990¿, ¡pointer: 0x32e5060¿, .pointer, ...)
2: mcmc$run(genome, model, ncores, divergence.iteration)
3: runMCMC.Rcpp_MCMCAlgorithm(mcmc, genome, model, 4)
4: runMCMC(mcmc, genome, model, 4)
5: system.time(runMCMC(mcmc, genome, model, 4))
6: eval(expr, envir, enclos)
7: eval(ei, envir)
8: withVisible(eval(ei, envir))

Since I've been wanting to get into the code a little more, I figured trying to debug this myself wouldn't be a bad idea. Based on what I found, it seems like the writeParameterObject() function in R is intended to create a Parameter object that will be used **just** for future data analysis, not a starting point for a new run. Maybe that is what was intended, but I think this

is bad software practice. If you have two objects of the same class, then they should work the same way.

The place where things are breaking is at the call for ROCModel::updateTracesWithInitialValues(Genome &genome). The source of this error seems to be that many of the variables that are needed for fitting a model are initialized in the initParameterSet() function. This function is only called from the constructors of the Parameter subclasses; as a result, they will only be initialized when the initializeParameterObject() is called in R.

The loaded Parameter object does not see groupList, which is an array containing the amino acid letter ids (ie. K for lysine). This can be fixed by moving the initialization of this list to the Parameter.h file. I still need to do some digging into this.

7 9-2-2016

I continued some of the debugging from yesterday. I also was unfortunate enough to encounter the error from August 31st again on my local machine. However, I have not had much luck consistently generating this error, so what the cause of it is baffling me.

However, I did find a fairly significant error in Genome::getGenomeForGeneIndicesR(), which returns a genome consisting of the genes in a mixture. In the plotModelObject.R (generates CUB plots), it looks like it pulls out the genes in a mixture and passes into Genome::getGenomeForGeneIndicesR a list of the indices for these genes. The problem is these indices are based on an R vector (starts at 1), but the code in Genome::getGenomeForGeneIndicesR() forgets to subtract off 1 to convert the indices to C++ array indices (starting at 0). In the case of a mixture model with only 1 category, this just means the genome returned will be missing the fist gene and contain a blank gene in the last position in the gene array. However, when fitting using mixture categories, this means you could be missing a lot of genes. I'm not surprised this error was not found sooner because of C/C++ not returning an IndexOutOfBounds error when accessing array/vector elements via array[index]. It is generally safer when working with std::vector¡¿ to use vector.at(index), as this will return an IndexOutOfBounds error. However, it is also slower.

I talked with Bob before I called it a day. I explained to him some of the problems with previous analysis of codon usage bias for signal peptides in *E. coli* (ie. failure to account for affects of varying gene expression) and showed him some of the plots I generated. I also explained to him the next steps I want to take with the analysis. He said he was happy about what I have done so far and where I plan on going with this project.

7.1 Plans for Weekend

Continue reading "Part 2: Fundamentals of Bayesian Data Analysis" from *Bayesian Data Analysis*, 3rd ed, Gelman et al and "Chapter 7: Model Assessment and Selection" from *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed, Hastie et al.

Also review the chapter on numerical integration from *Numerical Methods*, 4th ed, Faires and Burden. This thing about dividing a Gamma distribution up into quantiles and taking the average value of each just seems wrong to me.

8 9-6-2016

Focused on learning about model checking. I just want to get a solid base in this area by the end of this week so I can start implementing some model checks. So far, looking like the Bayesian Information Criterion (BIC) might be a good start, but I want to read up on it a little bit more. I know Gelman et al talks about it and Hastie et al cite the original paper, which also might be worth taking a look at.

Last week, Mike noted that sometimes it is hard to tell if the traces are actually converging based on the plots. I think adding a moving average function might make this task a little bit easier.

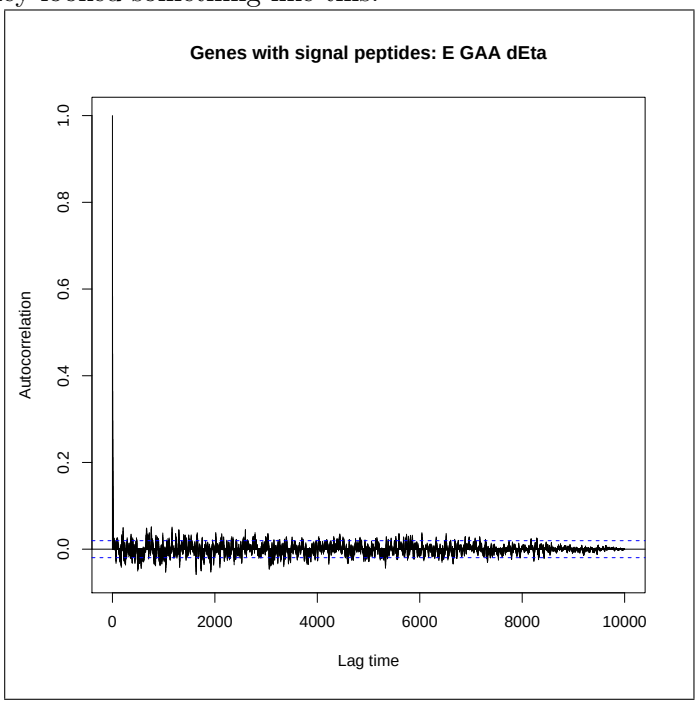
In one of the earlier SignalP paper, the authors note that SignalP should not be used to identify extracellular proteins; instead, researchers should use SecretomeP (by the same group). This brings up a good point: most of the papers that have done analysis of codon usage bias in signal peptides have seemed to imply that signal peptide = extracellular. It might be worth dividing up those with signal peptides into two groups: those that are predicted to be extracellular via SecretomeP and those that are not and fit these in ROC with a mixture model.

9 9-7-2016

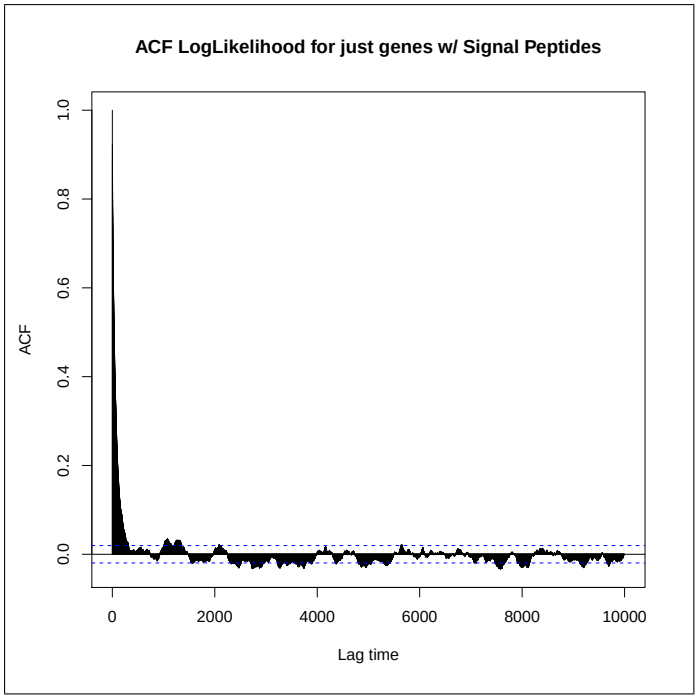
Talk to Cedric about autocorrelation functions. Look at Geweke scores.

10 9-11-2016

Since 9-7-2016: • Set up a Github repository for Bob's lab. Should make it easier for us to keep track of the scripts we write and make accessing them easier. I also commented/documented some of the scripts I had previously written to make them more accessible to my labmates who have little programming experience.
• Got a little bit of experience working with the Python package, Pandas. Supposed to be good for data analysis, which is why Yaojin has suggested teaching it in LFSC 507.
• Looked into autocorrelation functions and MCMC checking methods, including the Geweke score.
• Performed an autocorrelation on the $\Delta\eta$ and δM traces using R's acf function. Generally, they looked something like this.



• I also ran the acf for the loglikelihood trace and obtained results that generally looked like the following.



• Started NSF Graduate Fellowship application.

11 11-17-2016

11.1 Summary of work since last update

October was mainly driven by NSF GRFP application, colloquium presentation for Grant Writing, and writing NSF-style proposal for Grant Writing class. Time dedicated to TAing for the LFSC 507 class has not been insignificant as I had to learn the Python packages NumPy, SciPy, and Pandas for some of my lectures.
However, I was able to make significant progress on the Signal Peptide project (see section "Signal Peptide Project - Progress as of 11-17-2016" for details).

Mike's comments regarding current results:
• Overall, it seems like selection on CUB within signal peptide regions is in the same direction as the rest of the genome. However, there might be some differences in the strength of selection for certain codons.

• Standard linear regressions were used for generating the plots found in the Signal Peptide Project section. Weighted least squares regression would be a more valid approach due to the deterministic variables in these cases are not equally precise. I can modify the function I used to generate these plots to take in a boolean variable specifying whether or not the user wants to use ordinary or weighted least squares and perform the appropriate calculations. (Note: need to look into weighted least squares first).

• When treating Φ values as given and not estimating them, treated them as the true Φ value, ie. variation in Φ is 0. This is certainly not the case. A better approach might be to calculate the average standard deviation for each Φ from the different samples and use these to allow the genes to vary when estimating $\Delta\eta$. I will need to measure the autocorrelation of the Φ traces to make sure the samples are independent of the previous samples.

• Results for the signal peptides and first 31 codons in genes lacking a signal peptide (Fig-

ures 5a and 7a) seem more consistent with selection against nonsense errors in the 5' regions of the genes. To be more certain of this, instead of doing first 31 codons for all genes, can calculate mean and variance of signal peptide length and use these to draw cutoff points for splitting the genes. Signal peptide regions are generally 20 to 40 amino acids long. An alternative approach is take the first 20 codons,truncate out the next 20, and then take the the rest of the gene.

• Would be helpful if plots comparing $\Delta\eta$ values between mixtures generated by module plotParameterObject.R also included error bars.

Other current work:
1) The plotModelObject.R module cannot generate the codon usage frequency plots (as function of $log\Phi$) when not estimating expression. I started fixing this on 11-16-2016. Part of the problem with the current implementation is it relies on making calculations from the last samples of the traces. When not estimating a parameter, the traces for that parameter are not updated, resulting in the traces being filled with the value 0. I'm modifying the code to initialize the Φ traces with the initial values provided by the user. If no values are provided, the traces will be initialized with 0.0, as they are now. The advantage of this is it allows us to handle this situation of not estimating parameters without requiring a bunch of if-else statements to the code.
2) In Bob's lab, I promised to assist Mallory Ladd with the data analysis for a project she is currently working on and will present at a conference in a few weeks (with the promise of being listed as a co-author on the project). This project involves looking at liquid chromatography tandem mass spectrometry (LC-MS/MS) measurements of metabolites in soil water samples. Metabolomics data is usually more complicated than proteomics data because of the greater variability in the chemical structures of metabolites relative to peptides.
3) I need to finish modifying the current ribModel framework to treat Φ as a mixture of lognormal distributions. I need to implement the traces to keep track of the additional hyperparameters approximated by the model and actually test the code to make sure it works.

Other stuff:
• Assisting older students in GST with their preliminary exam by providing feedback on proposals/presentations. This has given me some insight into the GST prelim process. The combination of this and Grant Writing should result in my own prelim going smoothly next year...hopefully.
• Signed up for Outreach on Darwin Day website. I suggested last night at the meeting for Darwin Day that we consider doing something similar to "Pint of Science" which will allow general public to interact with scientists in a more informal context. I already harassed a couple of my closer friends in GST and they seem interested in helping out. • Final draft of proposal for Grant Writing is due November 22nd by 5 pm, so I've been working on making some edits to that the past couple of days.
Plans for the coming month (ordered by my current priority): 1) Make improvements to statistical approach and plots based on Mike's suggestions. Bob has said he would also like me to look at signal peptides in *C. thermocellum* and C. bescii, but I expect we will find similar results to *E. coli*. Given the relatively short time it takes to fit ROC to data, it wouldn't hurt to perform the additional runs. 2) Assist Mallory with data analysis. Poster presentation is in a few weeks. We have exchanged a few papers regarding metabolomics and current data analysis tools. We are meeting tomorrow (11-18-2016) to discuss ideas. 3) Finally finish Φ mixture distribution.

Signal Peptide Project - Progress as of 11-17-2016

All graphs in the following sections are of $\Delta\eta$ values, which represent selection. Dotted lines in all graphs represent y=x.

1 Three categories: No signal peptide, signal peptides, mature peptides (See Figure 1)

Treated mutation as shared between the categories. Gene expression was kept constant. Mixture Element 1 and Mixture Element 3 correspond to genes with no signal peptides and mature peptides. Mixture Element 2 is the signal peptide region. As can be seen, there is a strong linear correlation and slope of 0.93 between the genes with no signal peptides and mature peptides, which is expected. Ideally, this would be closer to 1. I wonder if the fact that genes with a signal peptide region are generally lower expression genes (mean Φ value of 0.95) is causing a slight drop in the estimates of $\Delta\eta$ values for codons in these genes. Relative to the signal peptide regions, there is a drop-off in correlation and the slopes are much different from 1 for both mature peptides and signal peptides. However, this could be because of increases in variation introduced by the small regions of signal peptides.

2 Three categories: Same as above, but simulated genome (See Figure 2)

My thought was that if the decreased correlation seen in Figure 1 between signal peptides and the other regions of the genome is because of increased variation, then signal peptides simulated under codon-specific parameters generated from the whole genome (minus the signal peptide regions) will produce similar results. Results for a fitting to a simulated genome are shown in Figure 2. This resulted in a slightly improved correlation between $\Delta\eta$ values for the three categories. The slopes for the comparisons to the $\Delta\eta$ values for signal peptides improve slightly (ie. get closer to 1). However, the slope of the comparison between $\Delta\eta$ values for the genes without a signal peptide and mature peptides decreases from 0.93 to 0.85.

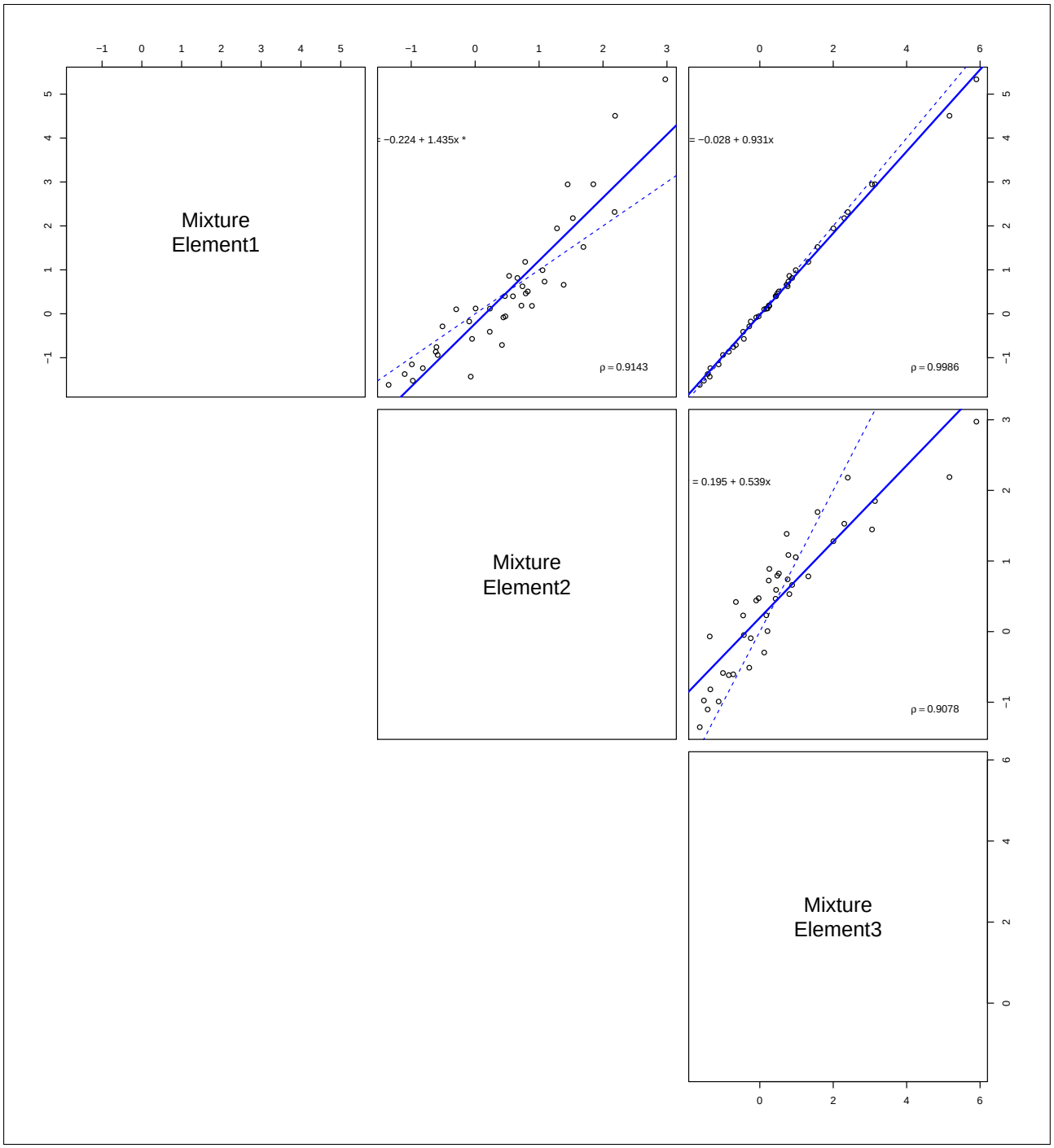


Figure 1: Three mixture model fit using *E. coli* K12 MG1655 genome. Mixture Element 1 represents genes without a predicted signal peptide. Mixture Element 2 represents signal peptide regions. Mixture Element 3 represents the mature peptide regions.

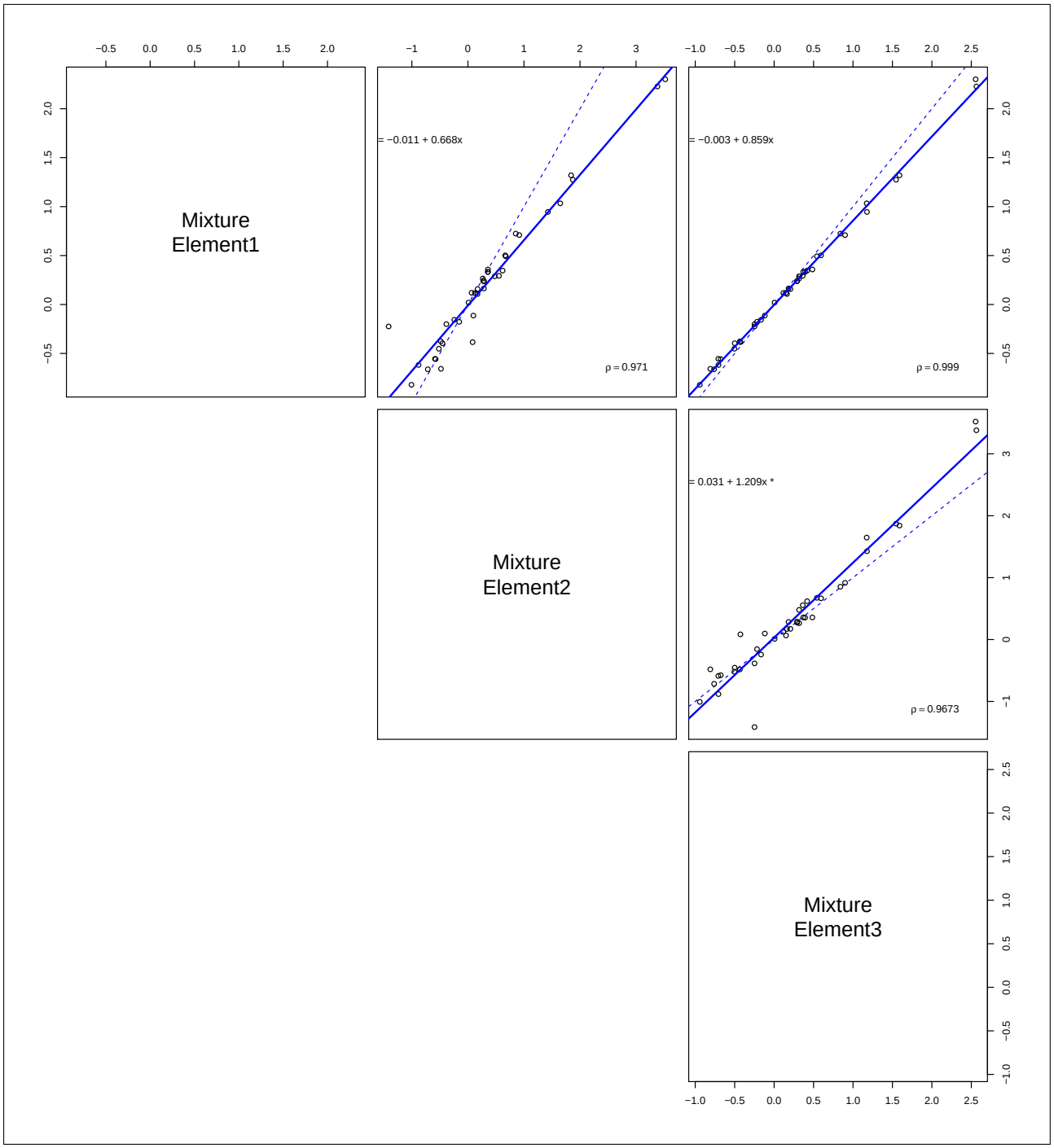


Figure 2: Three mixture model fit using simulated genome based on parameters estimated from whole *E. coli* genome minus signal peptide regions. Mixture Element 1 represents genes without a predicted signal peptide. Mixture Element 2 represents signal peptide regions. Mixture Element 3 represents the mature peptide regions.

3 Other model fittings

Description:
ROC() = Fit ROC-SEMPPR to the data provided
NoSp = Gene with no signal peptide
SP = Signal Peptide
MP = Mature Peptide
"=" = Treat as same category
"!=" = Not treated as same category
"Together" = Just means treated as same category

All cases in which regions are assumed to be different categories are set to share mutation bias terms.
1. ROC(MP) $\rightarrow \Phi$ (obtain gene expression values for genes with signal peptides)
2. ROC(NoSP = MP) and ROC(NoSP != MP)
3. ROC(NoSP = SP) and ROC(NoSP != SP)
4. ROC(MP = SP $|\Phi$) and ROC(MP != SP $|\Phi$)
5. ROC(Simulated MP = Simulated SP $|\Phi$) and ROC(Simulated MP != Simulated SP $|\Phi$)
6. ROC(First 31 codons of NoSP gene = Rest of Gene $|\Phi$) and ROC(First 31 codons of NoSP!= Rest of Gene $|\Phi$)

3.1 ROC(NoSp = MP) and ROC(NoSp != MP) (See Figure 3)

As expected, the estimates of $\Delta\eta$ for NoSP and MP, when treated as together and as separate categories, demonstrate a strong positive correlation and a practically 1-1 relationship. In addition, $\Delta\eta$ estimates generally have tight confidence intervals. This provides evidence that genes without signal peptides and mature peptide regions are under similar selective pressures.

3.2 ROC(NoSp = SP) and ROC(NoSp != SP) (See Figure 4)

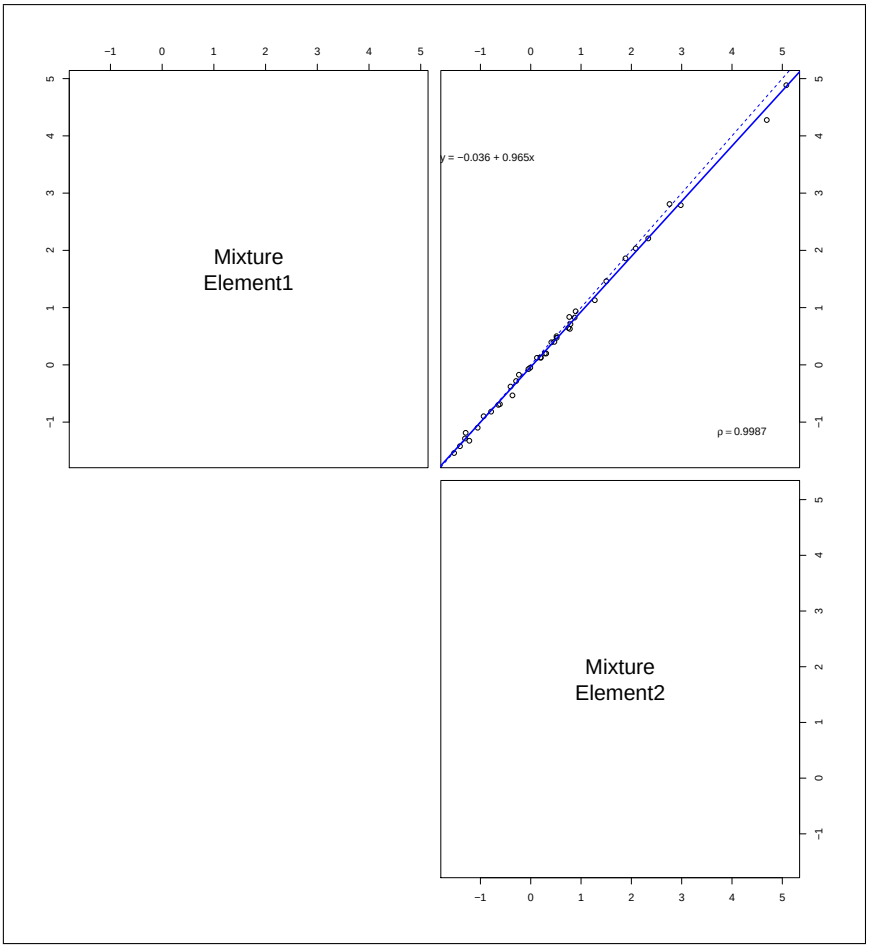
Figure 4b shows a very linear relationship between the $\Delta\eta$ values when NoSP and SP are treated as the same category and the $\Delta\eta$ values of just NoSP. Figure 4a and Figure 4c show that SP regions have a similar direction of selection to NoSP (ie. codons with high $\Delta\eta$ values in NoSP have high $\Delta\eta$ in SP), but the strength of the selection might differ.

3.3 ROC(MP = SP $|\Phi$) and ROC(MP != SP $|\Phi$) (See Figure 5)

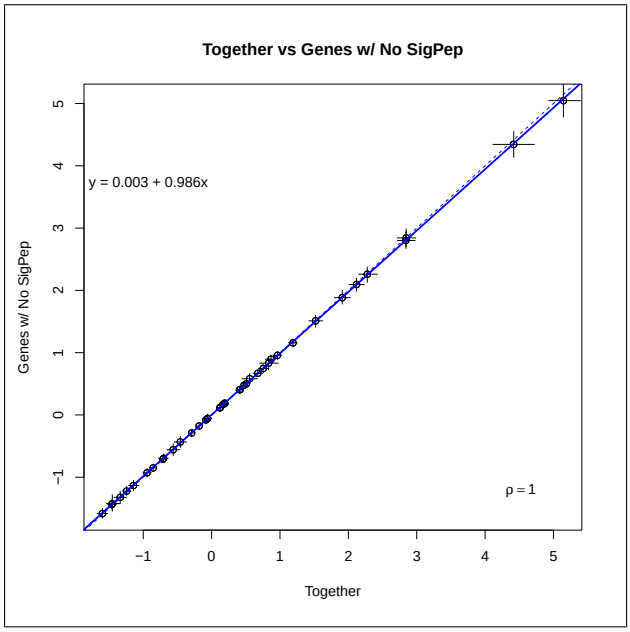
Due to the limited amount of data, Φ values were set to constant for these fittings. Interestingly, doing so seemed to cause the $\Delta\eta$ values of the signal peptide regions to be much more constrained than when MP = SP (Note that the scale for SP only goes up to 1.5, while when MP = SP, values go up to 4). Overall, there is still a positive linear correlation between $\Delta\eta$ values between MP and SP.

3.4 ROC(Simulated MP = Simulated SP $|\Phi$) and ROC(Simulated MP != Simulated SP $|\Phi$) (See Figure 6)

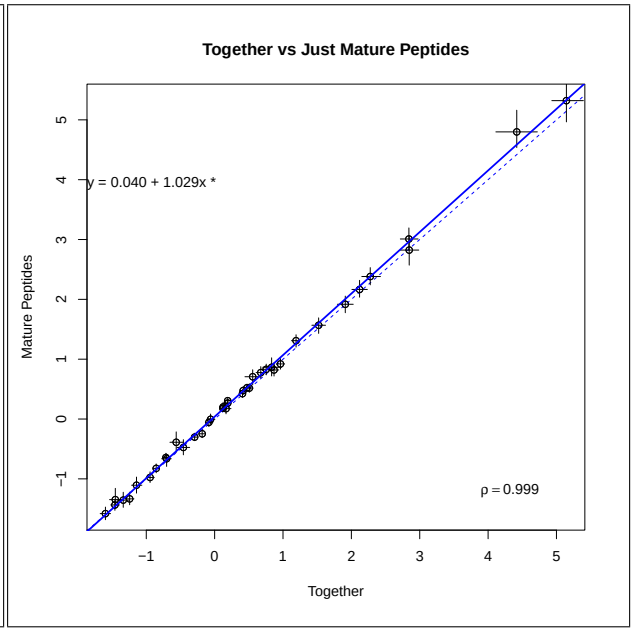
Using the SP and MP from the simulated genome, I did a similar fit to the previous one. Results show a stronger linear correlation, and slope is a little closer to 1. However, the scale of the $\Delta\eta$ values for the signal peptide regions is much greater (up to approx 6.0). This makes me wonder if I made a mistake. I will need to double check my work here.



(a) $\Delta\eta$ comparisons from treating NoSP and MP as separate categories. Mixture element 1 represents genes not predicted to have a signal peptide. Mixture element 2 represents mature peptides.

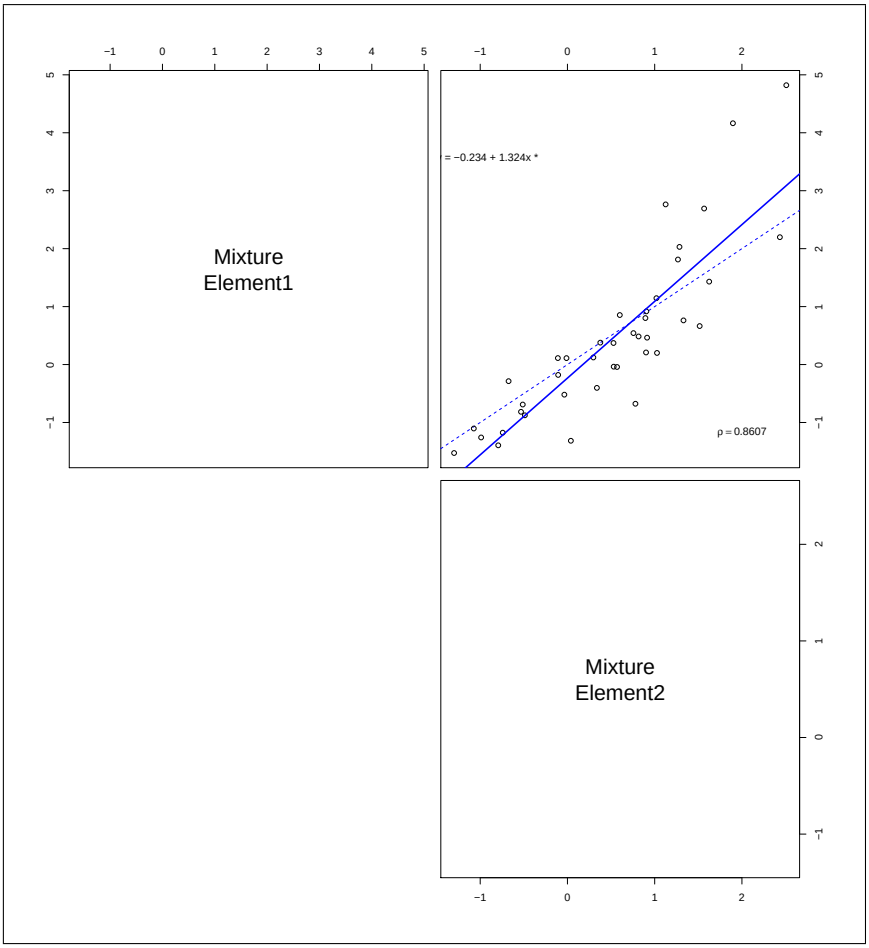


(b) Comparison between $\Delta\eta$ values when NoSP and MP are treated as same category and when NoSP is treated as own category.

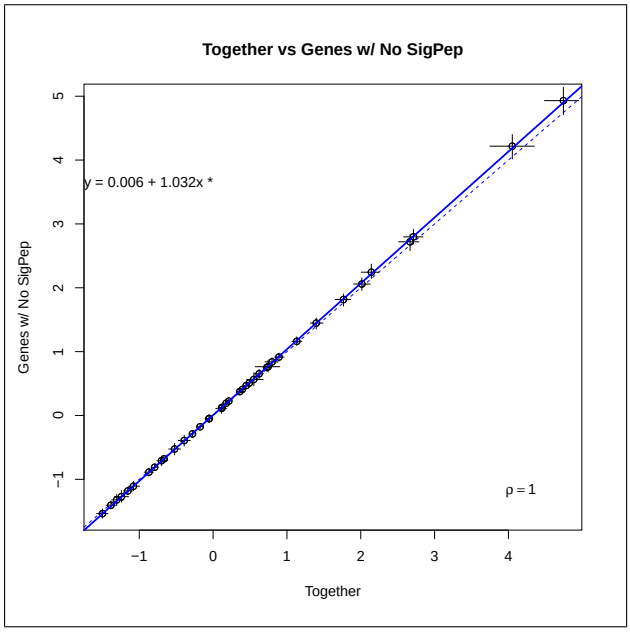


(c) Comparison between $\Delta\eta$ values when NoSP and MP are treated as same category and when MP is treated as own category.

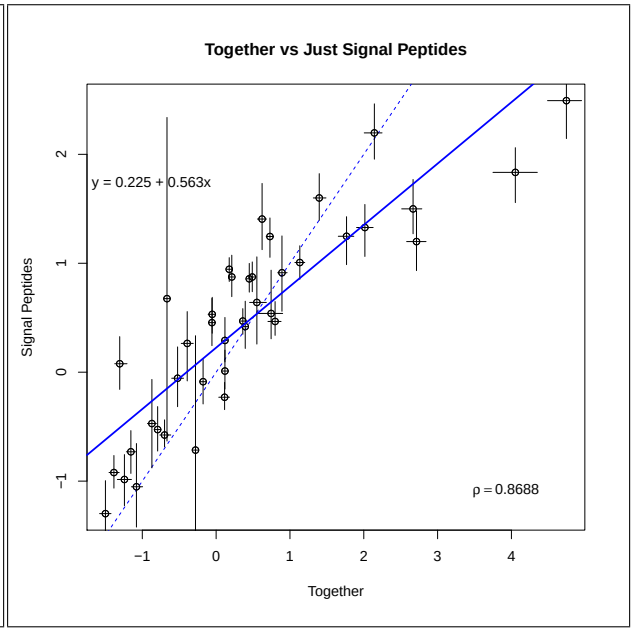
Figure 3: Results from ROC(NoSP = MP) and ROC(NoSP != MP)



(a) $\Delta\eta$ comparisons from treating NoSP and SP as separate categories. Mixture element 1 represents genes not predicted to have a signal peptide. Mixture element 2 represents signal peptides.

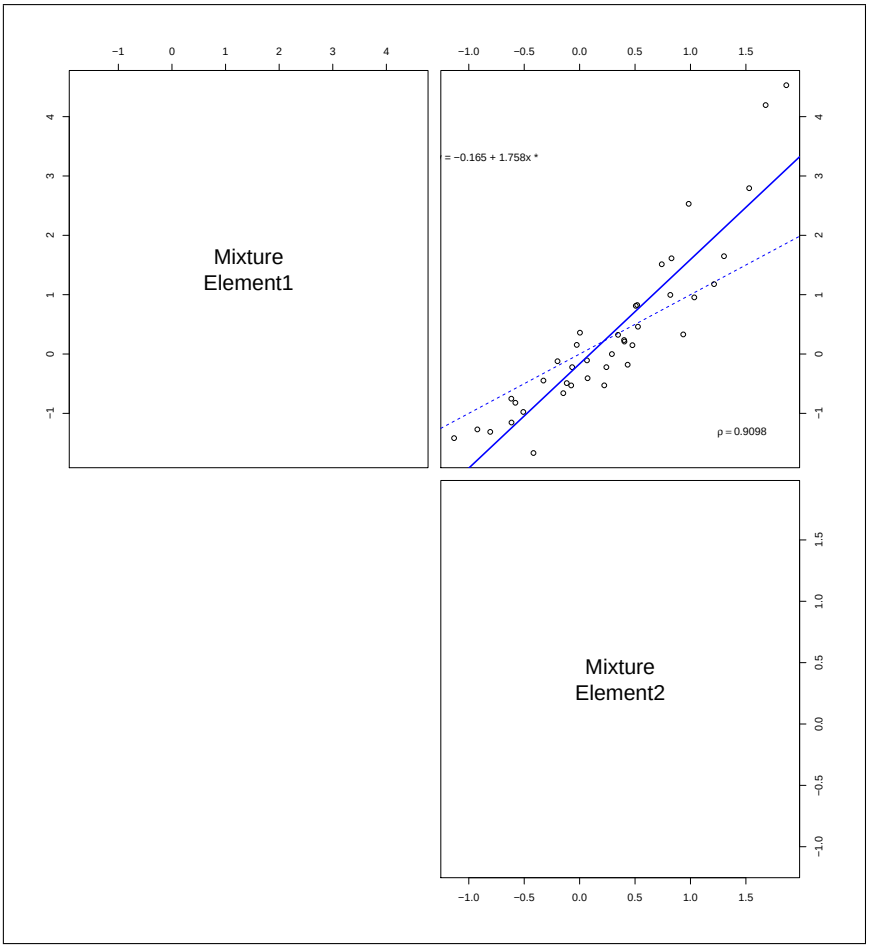


(b) Comparison between $\Delta\eta$ values when NoSP and SP are treated as same category and when NoSP is treated as own category.

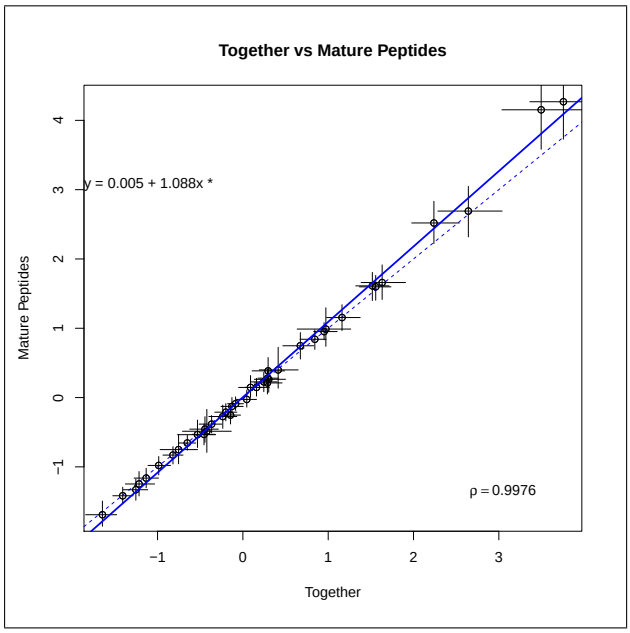


(c) Comparison between $\Delta\eta$ values when NoSP and SP are treated as same category and when SP is treated as own category.

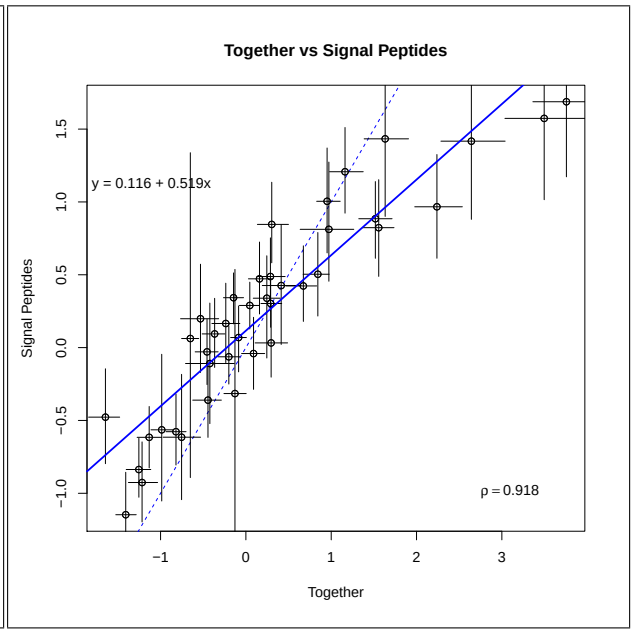
Figure 4: Results from ROC(NoSP = SP) and ROC(NoSP != SP)



(a) $\Delta\eta$ comparisons from treating MP and SP as separate categories. Mixture element 1 represents mature peptides. Mixture element 2 represents signal peptides.

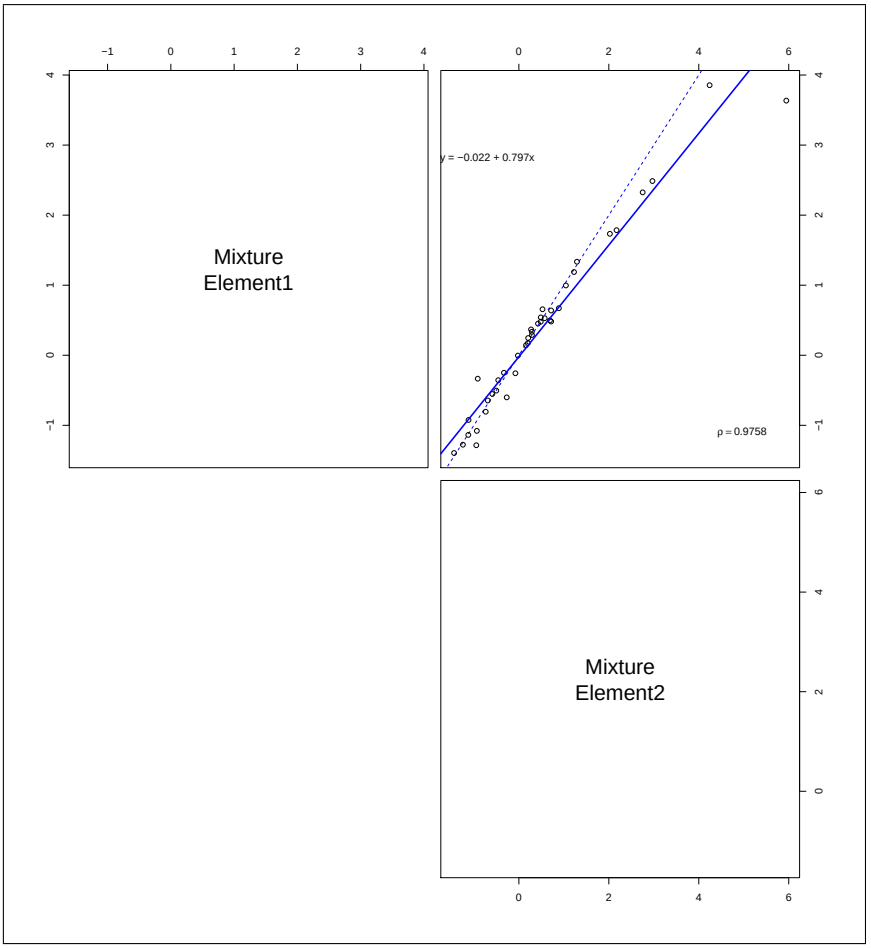


(b) Comparison between $\Delta\eta$ values when MP and SP are treated as same category and when MP is treated as own category.

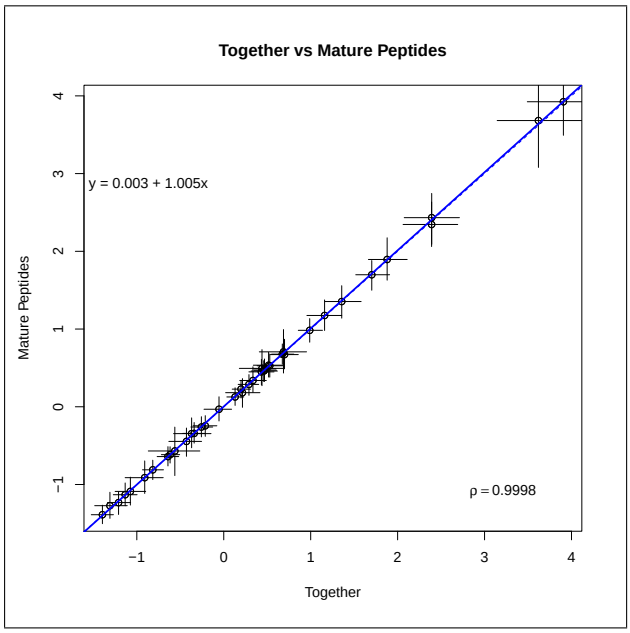


(c) Comparison between $\Delta\eta$ values when MP and SP are treated as same category and when SP is treated as own category.

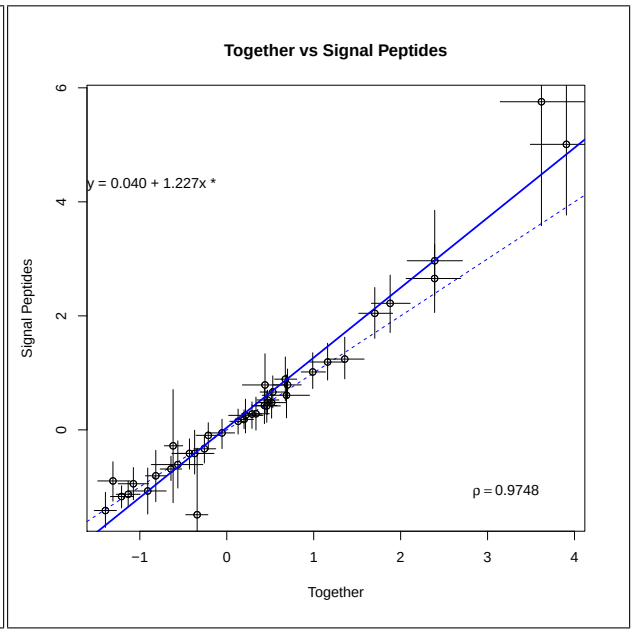
Figure 5: Results from ROC(MP = SP) and ROC(MP != SP)



(a) $\Delta\eta$ comparisons from treating MP and SP as separate categories.Mixture element 1 represents mature peptides. Mixture element 2 represents signal peptides.



(b) Comparison between $\Delta\eta$ values when MP and SP are treated as same category and when MP is treated as own category.



(c) Comparison between $\Delta\eta$ values when MP and SP are treated as same category and when SP is treated as own category.

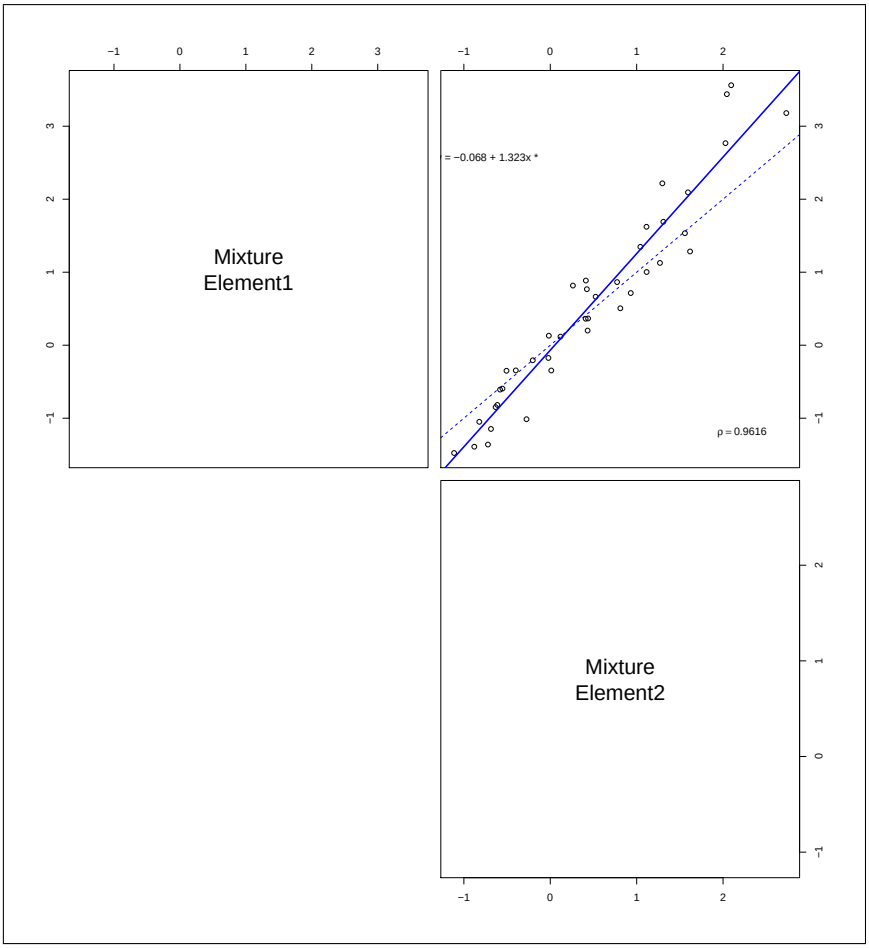
Figure 6: Results from ROC(Simulated MP = Simulated SP $|\Phi$) and ROC(Simulated MP != Simulated SP $|\Phi$)

3.5 ROC(First 31 codons of NoSP genes = Rest of gene $|\Phi$) and ROC(First 31 codons of NoSP genes != Rest of gene $|\Phi$) (See Figure 7)

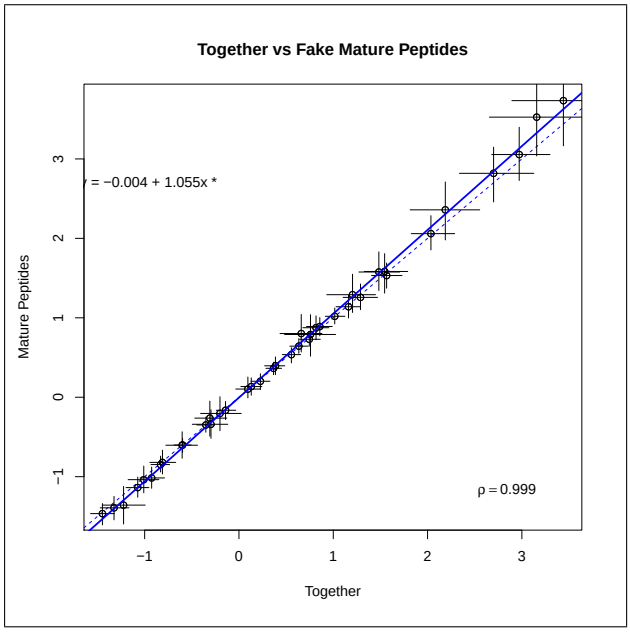
One potential question is whether or not there are general differences between codon usage in the 5' regions of genes and the rest of the genome. A random set of genes not predicted to have signal peptides were selected such that the mean and variation of gene expression was comparable to the set of genes predicted to have signal peptides (0.95 and 0.42, respectively). This produced a set of 361 genes (compared to 425 genes with a signal peptide). Each gene was split into two sections: the first 31 codons and the rest of the gene. Again, there is a clear positive linear correlation.

3.6 Comparisons between mature peptides and signal peptides (See Figure 8)

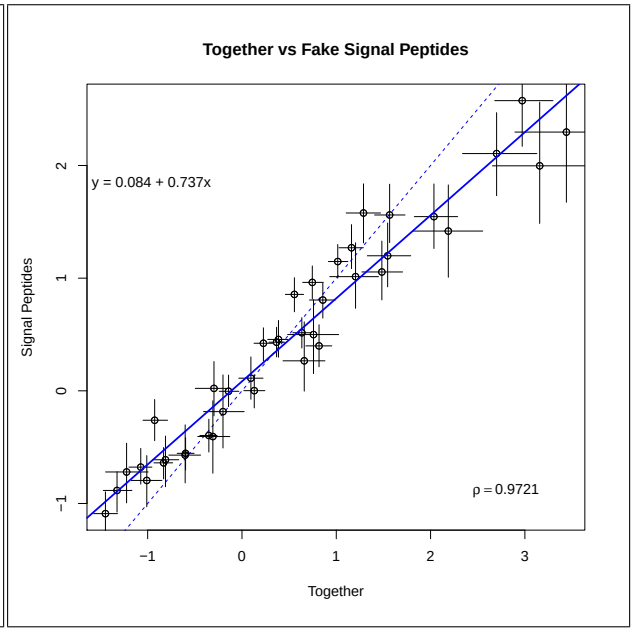
Below are graphs comparing the $\Delta\eta$ estimates for MP and SP from the 3 fittings that estimated $\Delta\eta$ values for these regions while considered separate categories.



(a) $\Delta\eta$ comparisons from treating first 31 codons and rest of gene as separate categories.Mixture element 1 represents the rest of the gene. Mixture element 2 represents the first 31 codons.



(b) Comparison between $\Delta\eta$ values when first 31 genes and rest of gene are treated as same category and when rest of gene is treated as own category.



(c) Comparison between $\Delta\eta$ values when first 31 genes and rest of gene are treated as same category and when first 31 codons is treated as own category.

Figure 7: Results from ROC(First 31 codons of NoSP gene = Rest of Gene $|\Phi$) and ROC(First 31 codons of NoSP!= Rest of Gene $|\Phi$)

(a) Comparison of $\Delta\eta$ values for MP from ROC(NoSP!=MP) (x-axis) and ROC(MP!=SP|Φ) (y-axis)

(b) Comparison of $\Delta\eta$ values for SP from ROC(NoSP!=SP) (x-axis) and ROC(MP!=SP|Φ) (y-axis)

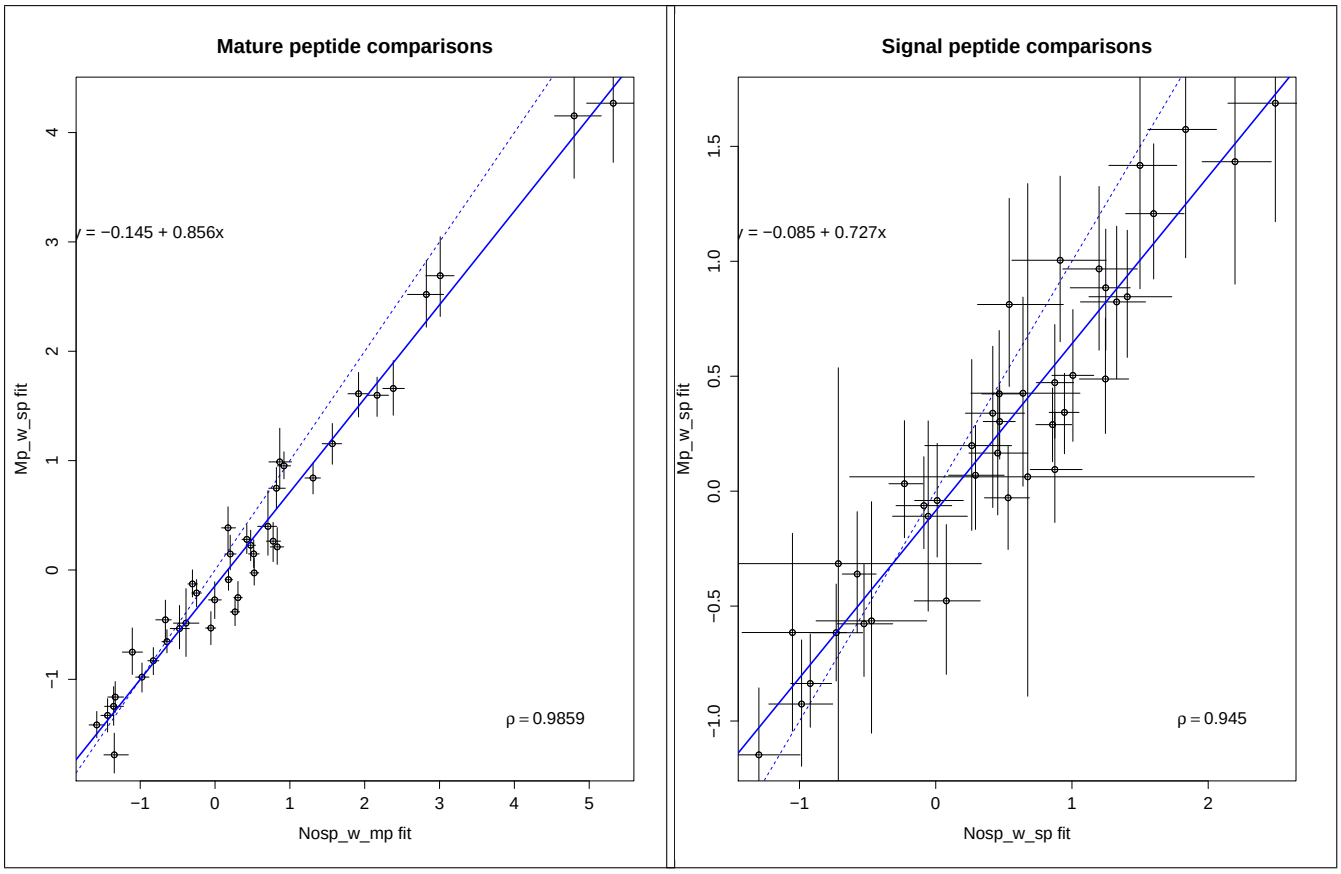


Figure 8: Comparing $\Delta\eta$ values for MP and SP from ROC(NoSP!=SP), ROC(NoSP!=MP), and ROC(MP!=SP|Φ)

4 My thoughts and next steps

As expected, it looks like genes without signal peptides and mature peptides have very similar codon usage patterns and the strength of selection on these codons is generally the same. Overall, the positive correlations between $\Delta\eta$ values for actual signal peptides and the rest of the genome suggests to me that, in *E. coli*, codon selection in signal peptides is fairly consistent with the rest of the genome. However, based on comparisons to simulated data and the 5' end of genes, it appears there are some differences in the strength of selection for certain codons, as evidence by greater variations from the line y = x.

I think my next step will be to replicate the analysis described above using a set of filtered signal peptides. SignalP 4.0 returns a value that essentially measures a confidence value for whether or not the gene actually contains a signal peptide. It is possible some of the lower scoring signal peptides are not actually signal peptides, which could be throwing off some of the results. For now, I will choose a SignalP score of 0.75 (out of 1) as my cutoff, leaving me with 267 genes to work with.