Modeling with Mixtures-of-Gammas: Some Practical and Computational Considerations

Tasks for Bill

- The pdf for a k-component mixture-of-gammas distribution is

$$g(x; \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\lambda}) = \sum_{j=1}^{k} \lambda_j f(x; \alpha_j, \beta_j),$$

  such that f is the pdf for a gamma distribution, $\sum_{j=1}^{k} \lambda_j = 1$, and $\lambda_j > 0$ for all j. In the above notation, we have $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_k)^{\mathrm{T}}$, $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_k)^{\mathrm{T}}$, and $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_k)^{\mathrm{T}}$. For our study, we will be looking only at 2-component or 3-component mixture models; i.e., $k \in \{2, 3\}$.

- Load the `mixtools` package and then read in the `gammamixEM2.R` file that I provided you.

- We will onsider these 12 settings:

  - **Condition 1**: $k = 2$, $\boldsymbol{\alpha} = (2, 5)^{\mathrm{T}}$, $\boldsymbol{\beta} = (3, 4)^{\mathrm{T}}$, and $\boldsymbol{\lambda} = (0.5, 0.5)^{\mathrm{T}}$
  - **Condition 2**: $k = 2$, $\boldsymbol{\alpha} = (2, 5)^{\mathrm{T}}$, $\boldsymbol{\beta} = (3, 4)^{\mathrm{T}}$, and $\boldsymbol{\lambda} = (0.2, 0.8)^{\mathrm{T}}$
  - **Condition 3**: $k = 2$, $\boldsymbol{\alpha} = (1, 10)^{\mathrm{T}}$, $\boldsymbol{\beta} = (1, 1)^{\mathrm{T}}$, and $\boldsymbol{\lambda} = (0.5, 0.5)^{\mathrm{T}}$
  - **Condition 4**: $k = 2$, $\boldsymbol{\alpha} = (1, 10)^{\mathrm{T}}$, $\boldsymbol{\beta} = (1, 1)^{\mathrm{T}}$, and $\boldsymbol{\lambda} = (0.2, 0.8)^{\mathrm{T}}$
  - **Condition 5**: $k = 2$, $\boldsymbol{\alpha} = (2, 30)^{\mathrm{T}}$, $\boldsymbol{\beta} = (3, 2)^{\mathrm{T}}$, and $\boldsymbol{\lambda} = (0.5, 0.5)^{\mathrm{T}}$
  - **Condition 6**: $k = 2$, $\boldsymbol{\alpha} = (2, 30)^{\mathrm{T}}$, $\boldsymbol{\beta} = (3, 2)^{\mathrm{T}}$, and $\boldsymbol{\lambda} = (0.2, 0.8)^{\mathrm{T}}$
  - **Condition 7**: $k = 3$, $\boldsymbol{\alpha} = (2, 5, 6)^{\mathrm{T}}$, $\boldsymbol{\beta} = (3, 5, 7)^{\mathrm{T}}$, and $\boldsymbol{\lambda} = (1/3, 1/3, 1/3)^{\mathrm{T}}$
  - **Condition 8**: $k = 3$, $\boldsymbol{\alpha} = (2, 5, 6)^{\mathrm{T}}$, $\boldsymbol{\beta} = (3, 5, 7)^{\mathrm{T}}$, and $\boldsymbol{\lambda} = (0.2, 0.3, 0.5)^{\mathrm{T}}$
  - **Condition 9**: $k = 3$, $\boldsymbol{\alpha} = (1, 20, 50)^{\mathrm{T}}$, $\boldsymbol{\beta} = (2, 4, 3)^{\mathrm{T}}$, and $\boldsymbol{\lambda} = (0.2, 0.3, 0.5)^{\mathrm{T}}$
  - **Condition 10**: $k = 3$, $\boldsymbol{\alpha} = (1, 20, 50)^{\mathrm{T}}$, $\boldsymbol{\beta} = (2, 4, 3)^{\mathrm{T}}$, and $\boldsymbol{\lambda} = (0.2, 0.3, 0.5)^{\mathrm{T}}$
  - **Condition 11**: $k = 3$, $\boldsymbol{\alpha} = (2, 50, 180)^{\mathrm{T}}$, $\boldsymbol{\beta} = (1, 2, 3)^{\mathrm{T}}$, and $\boldsymbol{\lambda} = (0.2, 0.3, 0.5)^{\mathrm{T}}$
  - **Condition 12**: $k = 3$, $\boldsymbol{\alpha} = (2, 50, 180)^{\mathrm{T}}$, $\boldsymbol{\beta} = (1, 2, 3)^{\mathrm{T}}$, and $\boldsymbol{\lambda} = (0.2, 0.3, 0.5)^{\mathrm{T}}$

- For each of the 12 conditions, we will do the following:

  - Generate $B = 5000$ samples of size n, for $n \in \{100, 250, 500\}$. (First, try doing this for $B = 5$ to make sure your code runs properly.) Make sure you set a seed at the beginning of your simulation.

- For each set of samples that you generate, estimate the mixture-of-gammas model using 4 different strategies:

  1. Specify the starting values in `gammamixEM2` using the parameter values for the simulation.
  2. Do not specify starting values for any of the parameters in `gammamixEM2`. Run the algorithm 10 times and retain the output that has the best log-likelihood value; i.e., the fit that has the largest log-likelihood value.
  3. Transform your simulated data by taking the cube root. Then, fit a mixture-of-normals distribution to the data using the `normalmixEM` function. Do a preliminary hard classification of each observation to a component, which will effectively partition the simulated data into k groups. Do this classification by finding the maximum posterior membership probability for each observation. Then, fit a separate gamma distribution to each of the k groups. Use these estimates as the respective starting values in `gammamixEM2`; i.e., `alpha` and `beta`. Moreover, compute the proportion of observations you classified into each of the k groups. These proportions will serve as starting values for `lambda` in `gammamixEM2`.
  4. Redo the previous setting, but set `alpha` to the true parameter values and set `fix.alpha=TRUE` in `gammamixEM2`.

- For each of the 4 strategies above, keep a `list` of your output. Construct each `list` to be of length 5000 such that each element of the list is a matrix with the estimated parameter values and the final log-likelihood. Specifically, if `out` is your output, then collect your output as `new.out <- rbind(out$gamma.pars,out$lambda,out$loglik)`. Note the fourth row of the output matrix will simply be the log-likelihood repeated k times.

- In your lists, make sure you post-process the output by ordering the columns based on their estimates of the component means. Specifically, if you have `new.out` as defined above, then reorder the columns using `new.out <- new.out[,order(new.out[1,]*new.out[2,])]`. We do this to avoid the label switching problem in mixture estimation.

- In your simulations, set the convergence criterion argument in `gammamixEM2` to `epsilon=1e-5`.

- Make sure that your simulation can recover in case of an unintended error in the EM algorithm. One way to do this is to enclose everything within a `while` statement. Then, wrap the `try` function with option `silent=FALSE` around the `gammamixEM2`. Test to see if the output is of class `"try-error"`. If it is, prompt your loop to return to the previous iteration. If not, let the loop increment appropriately.

- In total, there are 12 mixture conditions, 3 different sample sizes, and 4 different starting value strategies. Therefore, you are running a total of $12 \times 3 \times 4 = 144$ simulation conditions.

- For each of the 144 simulation conditions, calculate the MSEs from the true parameter value as well as the mean and standard deviation of your parameter estimates; i.e., the MC standard deviation of your 5000 estimates will provide an estimate of the standard errors for each parameter.
- You can use Teng's old code as a general framework, but there are errors in what he did. Try to streamline and organize your code so that it is easy for others to disseminate. Again, only use Teng's code for general guidance.
- Try to parallelize your code and/or run it on one of the University's Unix clusters.