# ECON 899B - PS2

Alex von Hafften

11/22/2021

## Part 1 - Quadrature Integration

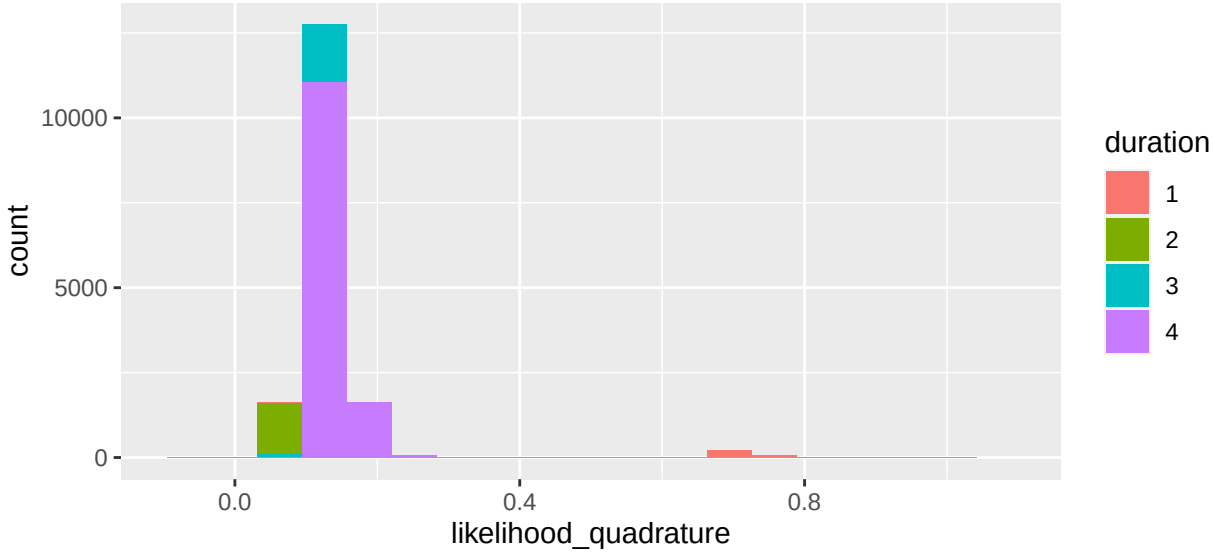
To implement the quadrature integration, I did not understand the provided equations from the problem set, so I derived the equations myself:

$$
\begin{aligned}
& P(T_i|X_i, Z_{it}, \theta) \\
&= \begin{cases}
P(\alpha_0 + X_i\beta + Z_{i0}\gamma + \varepsilon_{i0} > 0) & \text{if } T_i = 1 \\
P(\alpha_0 + X_i\beta + Z_{i0}\gamma + \varepsilon_{i0} < 0, \alpha_0 + X_i\beta + Z_{i1}\gamma + \varepsilon_{i1} > 0) & \text{if } T_i = 2 \\
P(\alpha_0 + X_i\beta + Z_{i0}\gamma + \varepsilon_{i0} < 0, \alpha_0 + X_i\beta + Z_{i1}\gamma + \varepsilon_{i1} < 0, \alpha_0 + X_i\beta + Z_{i2}\gamma + \varepsilon_{i2} > 0) & \text{if } T_i = 3 \\
P(\alpha_0 + X_i\beta + Z_{i0}\gamma + \varepsilon_{i0} < 0, \alpha_0 + X_i\beta + Z_{i1}\gamma + \varepsilon_{i1} < 0, \alpha_0 + X_i\beta + Z_{i2}\gamma + \varepsilon_{i2} < 0) & \text{if } T_i = 4
\end{cases} \\
&= \begin{cases}
\Phi((-\alpha_0 - X_i\beta - Z_{i0}\gamma)/\sigma_0) & \text{if } T_i = 1 \\
\int_{-\infty}^{-\alpha_0 - X_i\beta - Z_{i0}\gamma} \phi(\frac{\varepsilon_{i0}}{\sigma_0}) \frac{1}{\sigma_0} [1 - \Phi(-\alpha_1 - X_i\beta - Z_{i1}\gamma - \rho\varepsilon_{i0})] d\varepsilon_{i0} & \text{if } T_i = 2 \\
\int_{-\infty}^{-\alpha_0 - X_i\beta - Z_{i0}\gamma} \int_{-\infty}^{-\alpha_1 - X_i\beta - Z_{1i}\gamma} \phi(\frac{\varepsilon_{i0}}{\sigma_0}) \frac{1}{\sigma_0} \phi(\varepsilon_{i1} - \rho\varepsilon_{i1}) [1 - \Phi(-\alpha_1 - X_i\beta - Z_{i1}\gamma - \rho\varepsilon_{i0})] d\varepsilon_{i1} d\varepsilon_{i0} & \text{if } T_i = 3 \\
\int_{-\infty}^{-\alpha_0 - X_i\beta - Z_{i0}\gamma} \int_{-\infty}^{-\alpha_1 - X_i\beta - Z_{1i}\gamma} \phi(\frac{\varepsilon_{i0}}{\sigma_0}) \frac{1}{\sigma_0} \phi(\varepsilon_{i1} - \rho\varepsilon_{i1}) \Phi(-\alpha_1 - X_i\beta - Z_{i1}\gamma - \rho\varepsilon_{i0}) d\varepsilon_{i1} d\varepsilon_{i0} & \text{if } T_i = 4
\end{cases}
\end{aligned}
$$

The resulting estimated log-likelihood is:

```
## [1] -34812.7
```

A histogram of the estimated likelihoods by duration are below:

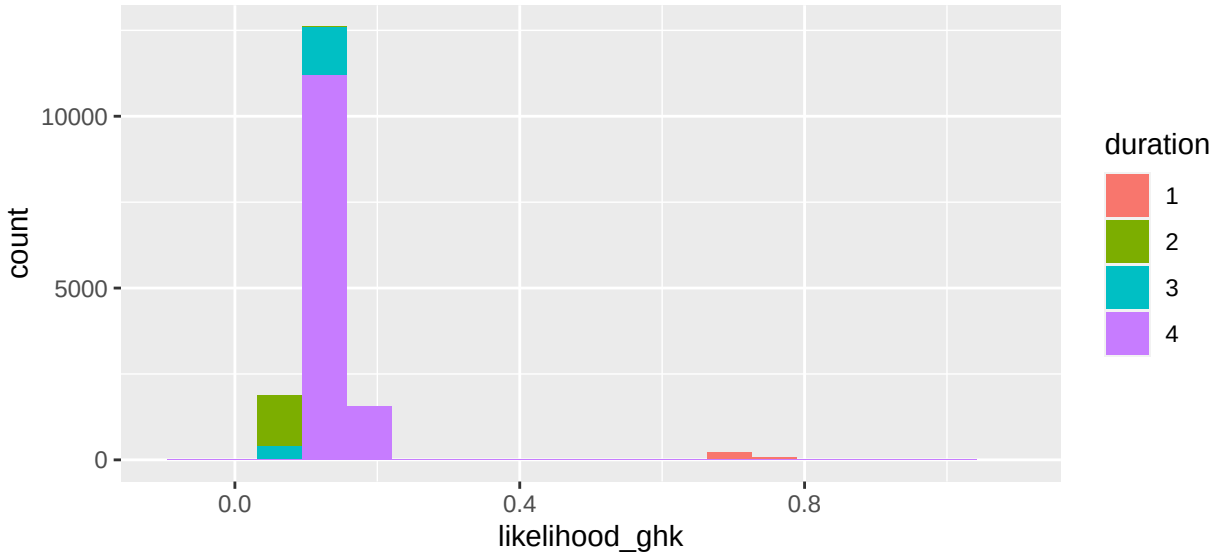


## Part 2 - GHK Method

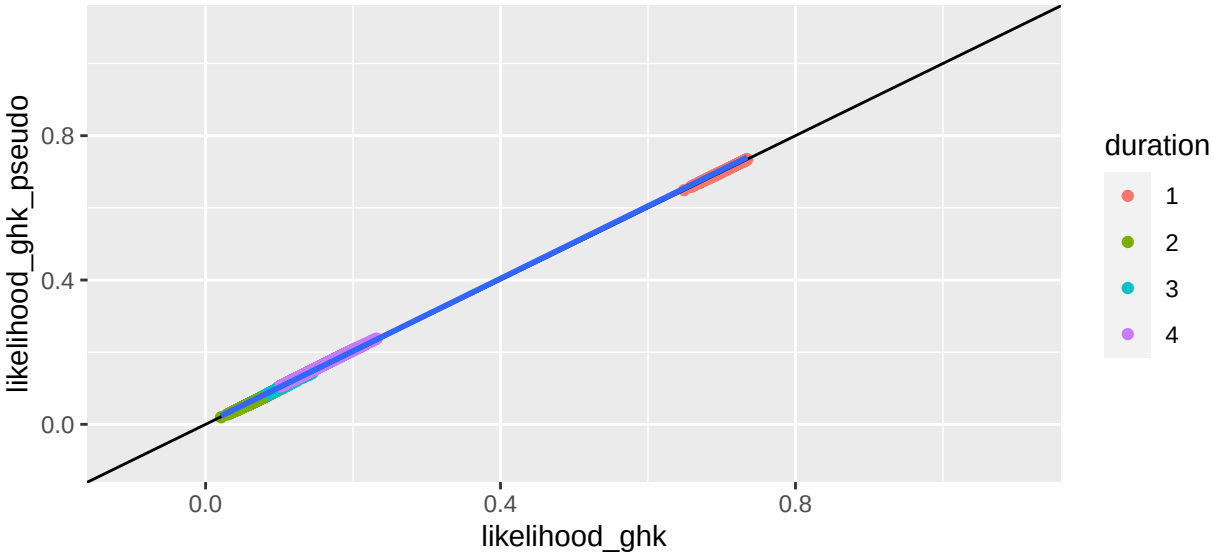
Using the GHK method, the estimated log-likelihood is:

```
## [1] -34718.16
```

The histogram of the estimated likelihoods by duration are below:

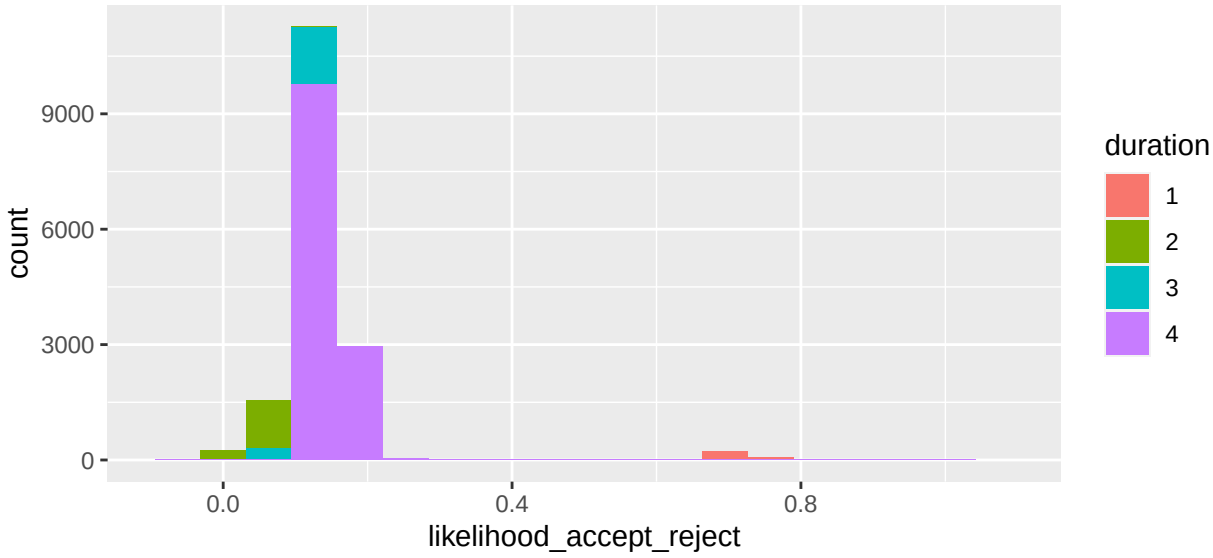



In the above histogram, I use Halton sequences to generate the simulations. In comparison, I also used Julia's built-in pseudo random number generation in the GHK method. The estimates are numerically different, but there's effectively no meaningful differences. The black line is a 45 degree line and the blue line is a ordinary least squares regression line.

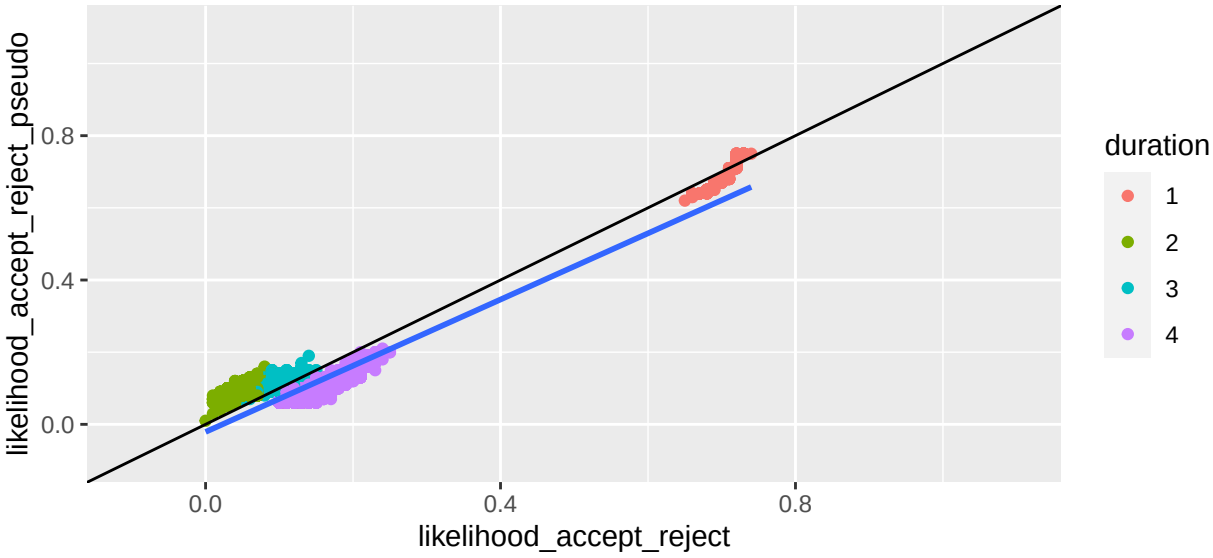


## Part 3 - Accept-Reject Method

Using the accept-reject method, some of the estimated likelihoods are zero, so the estimated log-likelihood is negative infinity. This is a very undesirable feature of these method because low likelihood observation will have large impact on the overall estimated log-likelihood. The histogram of the likelihoods are below:




In the above histogram, I use Halton sequences to generate the simulations. In comparison, I also used Julia's built-in pseudo random number generation in the accept-reject method. Unlike using Halton sequences for the GHK method, there is a noticeable difference between using Halton sequences and pseudo-random numbers. Using Halton sequences results in higher probability estiamtes than using pseudo random numbers. This is consistent with Halton sequences having better coverage than pseudo random numbers.



## Part 4 - Comparison

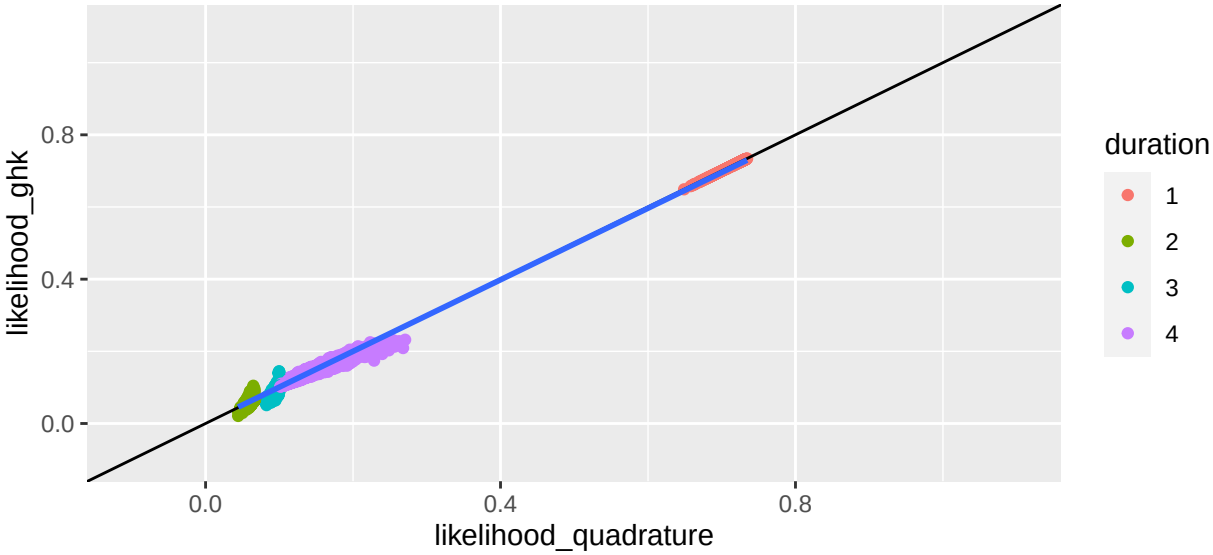
First, let's compare summary statistics by method and duration.

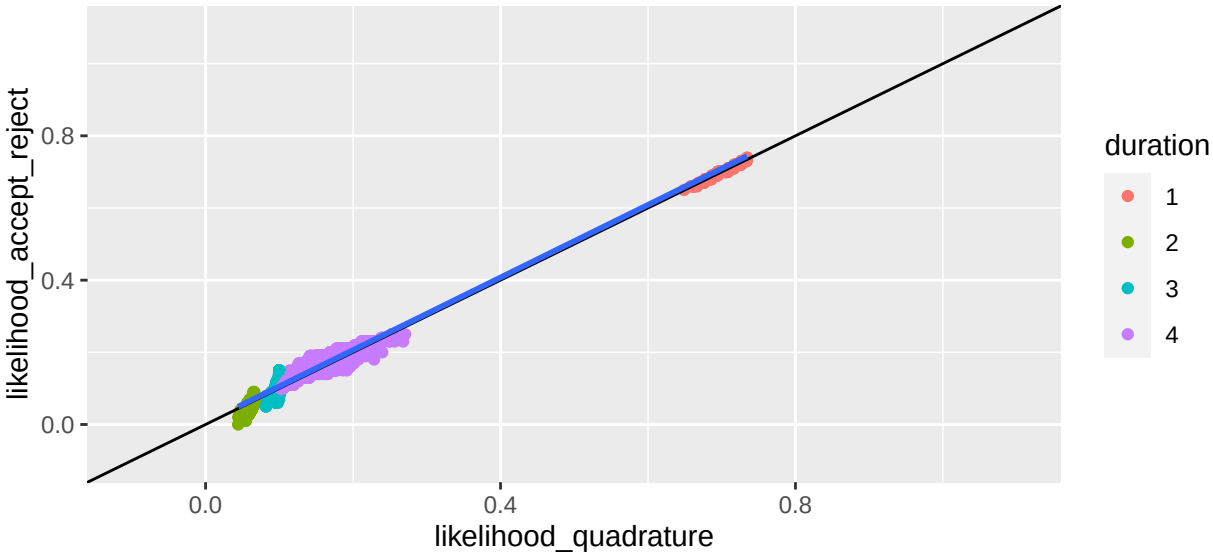
| name | min | first_quartile | mean | median | third_quartile | max |
|---|---|---|---|---|---|---|
| likelihood_accept_reject | 0.000 | 0.110 | 0.136 | 0.120 | 0.150 | 0.740 |
| likelihood_ghk | 0.021 | 0.106 | 0.131 | 0.118 | 0.138 | 0.734 |
| likelihood_quadrature | 0.044 | 0.103 | 0.130 | 0.115 | 0.137 | 0.734 |

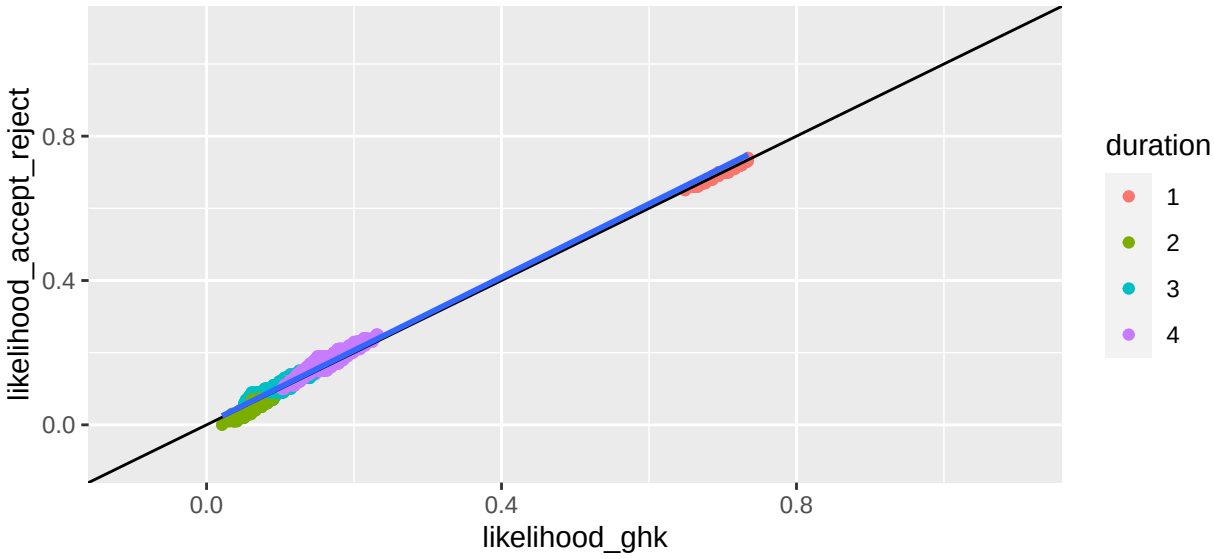
Second, let's consider how the likelihood estimates differ across computational methods. The correlation of the estimated likelihoods across all methods is very high. In particular, the quadrature method and GHK method are very highly correlated while the accept-reject method is slightly less.

| | likelihood_quadrature | likelihood_ghk | likelihood_accept_reject |
|---|---|---|---|
| likelihood_quadrature | 1.000 | 0.998 | 0.992 |
| likelihood_ghk | 0.998 | 1.000 | 0.994 |
| likelihood_accept_reject | 0.992 | 0.994 | 1.000 |

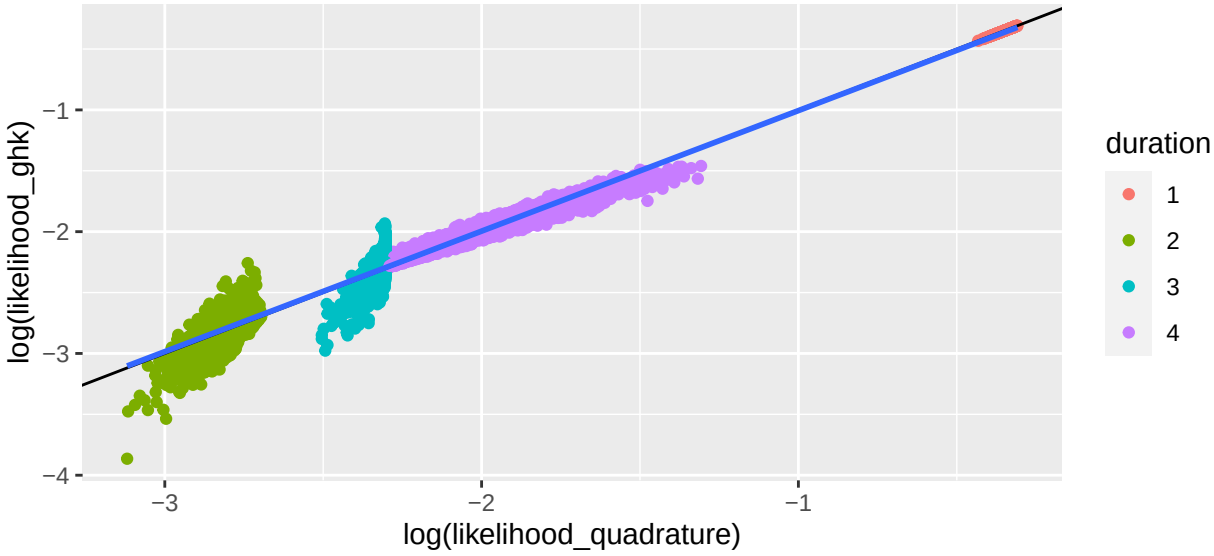
Below are three scatterplots that pairwise compare these methods. Generally, the quadrature and GHK methods produce very similar estimated likelihoods while the accept-reject method produces slightly noisier estimates.

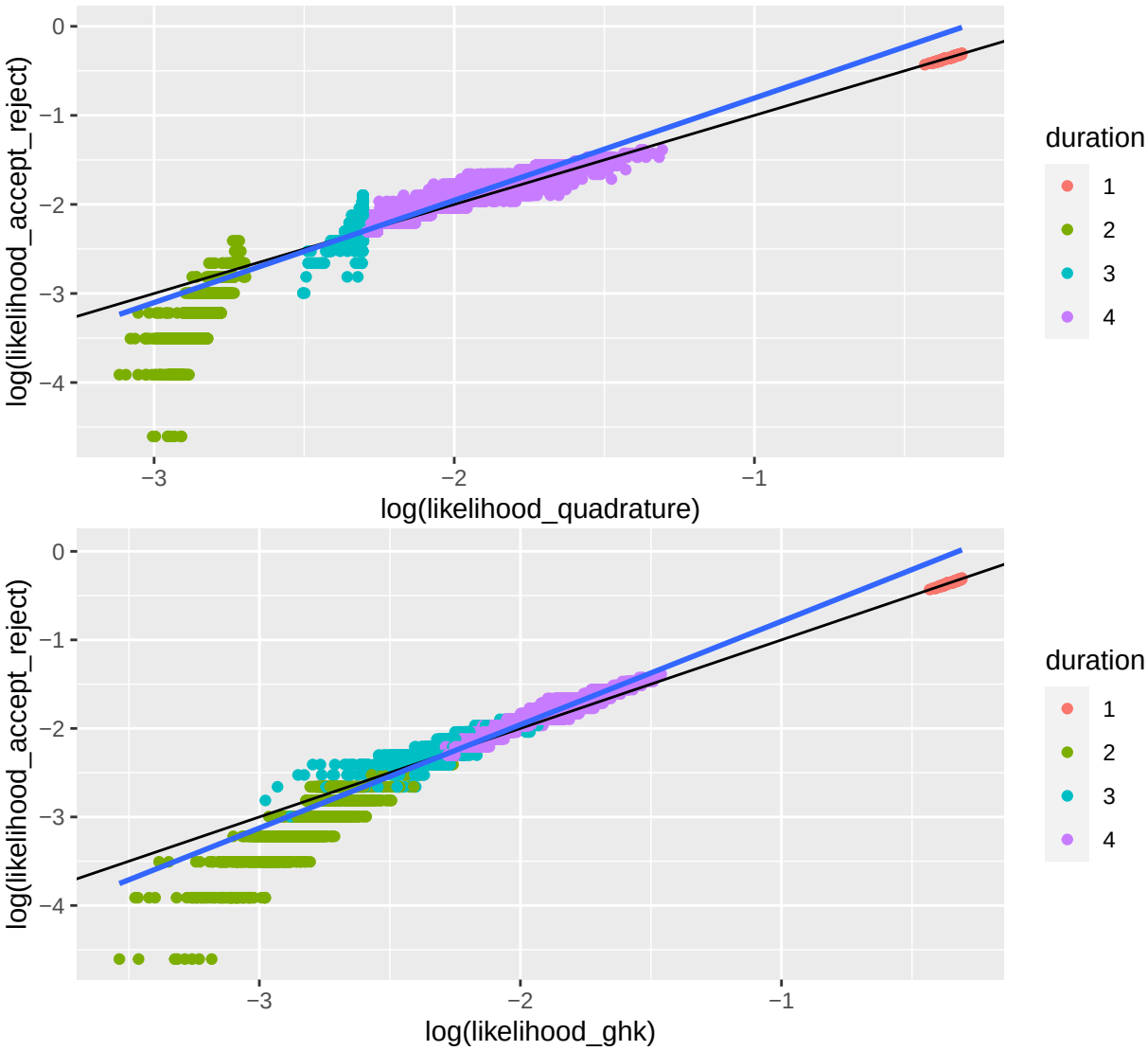





When we're applying these methods, we're maximizing log-likelihood. So I create scatterplots comparing the observation-level log-likelihood estimates; this requires that we drop the observations with zero likelihood under accept-reject. First, we can see some noise between quadrature and GHK, but the regression line is right on the 45 degree line. Second, from these scatterplots, we see that accept-reject underweights low likelihood observations compared to the quadrature and GHK estimates (i.e. the regression line is below the 45 degree line on the left hand side).






Third, let's consider the runtime across these techniques. For all of the techniques, my code distributed the computation of the observation-level likelihood over four processes. The quadrature integration takes about 27 seconds per run, the GHK method takes about 10 seconds per run, and the accept-reject method takes about 8 seconds per run. This comparison suggests that, at least in this application, the GHK method is preferred because the precision matches the quadrature method, but takes less than half the amount of time per run.

## Part 5 - Optimization

I ran into some issues getting my LBFGS optimization algorithm working correctly and the code is still running. The current parameter estimates are in the table below:

| variable | estimate |
|---|---|
| log_likelihood | -11537.288 |
| alpha_0 | -0.692 |
| alpha_1 | -0.444 |
| alpha_2 | -0.263 |
| score_0 | -0.170 |
| rate_spread | 0.163 |
| i_large_loan | 0.265 |
| i_medium_loan | 0.184 |
| i_refinance | 0.024 |
| age_r | 0.162 |
| cltv | -0.358 |
| dti | -0.014 |
| cu | 0.129 |
| first_mort_r | 0.101 |
| i_FHA | 0.182 |
| i_open_year2 | 0.385 |
| i_open_year3 | 0.142 |
| i_open_year4 | 0.028 |
| i_open_year5 | -0.006 |
| gamma | -0.038 |
| rho | -0.151 |

## Appendix - Code Description

I've attached a zip folder with my code. There are four scripts `01_toolbox.jl`, `02_likelihood.jl`, `98_test.jl`, and `99_run.jl`. I wrote the code in such a way that the observation-level likelihood computations are handled are distributed. In my case, I use four processes. The `01_toolbox.jl` script contains the functions that are sent to all worker processes. The `02_likelihood.jl` script contains the functions that setup the computation up by the main process. `98_test.jl` is a test file. The major test that I did was to compute the likelihood for each observation for $T_i = 1, 2, 3, 4$ to test that the likelihoods added up to one. Finally, `99_run.jl` runs reads in the data, sets up additional processes, and calls functions in `02_likelihood.jl`.