Homework 1

Ulrich Atz

2021-02-28

Part 1: Method of Moments

We first need to recode our dataset according to specification a)-c). I assume that people with a HH income of $1 million or more do not participate in the survey.

```
d <- read_csv(file = "simulated_income_data.csv")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   X1 = col_double(),
##   panel_year = col_double(),
##   household_income = col_double(),
##   number_of_hh = col_double()
## )
```

```
d <- d %>%
  mutate(lowbin = recode(household_income,
  `3`= 1000, # Avoid numerical issues with log-transformations
  `4`= 5000,
  `6`= 8000,
  `8`= 10000,
  `10` = 12000,
  `11` = 15000,
  `13` = 20000,
  `15` = 25000,
  `16` = 30000,
  `17` = 35000,
  `18` = 40000,
  `19` = 45000,
  `21` = 50000,
  `23` = 60000,
  `26` = 70000,
  `27` = 100000, # No change across years
  `28` = 125000,
  `29` = 150000,
  `30` = 200000)
  )
```

```r
d <- d %>%
  mutate(highbin = recode(household_income,
  `3`= 4999,
  `4`= 7999,
  `6`= 9999,
  `8`= 11999,
  `10` = 14999,
  `11` = 19999,
  `13` = 24999,
  `15` = 29999,
  `16` = 34999,
  `17` = 39999,
  `18` = 44999,
  `19` = 49999,
  `21` = 59999,
  `23` = 69999,
  `26` = 99999, # category 27 below
  `28` = 149999,
  `29` = 199999,
  `30` = 1e6) # assume only people with HH income of $1M or less participate in the survey
  )

d <- d %>%
  mutate(highbin = case_when(
   household_income == 27 ~ 124999, # for years 2006-2009
   panel_year <= 2005 & household_income == 27 ~ 1e6,
   panel_year  > 2009 & household_income == 27 ~ 1e6,
   TRUE ~ highbin)
 )

d <- d %>%
  mutate(meanbin = (highbin + lowbin) / 2)

# Weight the survey by year
d <- d %>%
  group_by(panel_year) %>%
  mutate(n_year = sum(number_of_hh)) %>%
  mutate(weights = number_of_hh / mean(number_of_hh)) %>%
  mutate(lowbin_w = lowbin * weights,
         highbin_w = highbin * weights,
         meanbin_w = meanbin * weights)

# How balanced are the years?
d %>% summarize(n = last(n_year))
```

```
## # A tibble: 7 x 2
##   panel_year     n
## *      <dbl> <dbl>
## 1       2007  3324
## 2       2008  3162
## 3       2009  3202
## 4       2010  3432
## 5       2011  3294
## 6       2012  3306
## 7       2013  3288
```

```
# Bin probability for part 3
d <- d %>%
  mutate(pr = number_of_hh / n_year)
```

For the method of moments estimators we substitute the population values with the (unbiased) sample mean and variance.

Note: I have used and estimated σ^2 instead of σ throughout part 1 and 2.

The first moment is the mean:

```
bins <- c("lowbin", "highbin", "meanbin") %>% set_names

# For y (from Wikipedia)
mm_mean <- function(x) {
             x_bar <- weighted.mean(pull(d, x), d$weights)
             s2 <- Hmisc::wtd.var(pull(d, x), d$weights)
             log( x_bar^2 / sqrt(s2 + x_bar^2) )
           }

mm_mean <- map_dbl(bins, mm_mean) %>% round(2)
mm_mean
```

```
##  lowbin highbin meanbin
##   10.91   10.95   10.94
```

```
# For log(y)
mm_mean_log <- map_dbl(bins, ~ weighted.mean(log(pull(d, .))), d$weights) %>% round(2)
mm_mean_log
```

```
##  lowbin highbin meanbin
##   10.10   10.43   10.30
```

The second moment is the variance:

```
# For y (from Wikipedia)
mm_var <- function(x) {
             x_bar <- weighted.mean(pull(d, x), d$weights)
             s2 <- Hmisc::wtd.var(pull(d, x), d$weights)
```

```
            log(s2 / x_bar^2 + 1)
          }

mm_var <- map_dbl(bins,  mm_var) %>% round(3)
mm_var
```

```
##  lowbin highbin meanbin
##   0.290   0.904   0.594
```

```
# For log(y)
mm_var_log <- map_dbl(bins, ~ Hmisc::wtd.var(log(pull(d, .)), d$weights)) %>% round(3)
mm_var_log
```

```
##  lowbin highbin meanbin
##   0.713   0.678   0.674
```

Confidence intervals are calculated with the respective inverse distribution, chosen alpha, and standard errors; asymptotically with the normal distribution.

The issue is if we want the values for the original data (untransformed) because $\mathbb{E}(income) \neq e^{\mu}$ as well as for other paramters. The method for estimating standard errors for the values transformed back is not as clear, e.g. `EnvStats::elnormAlt` lists five different methods. Note that the log-normal CI should be asymmetric.

```
# Use the overall N for the standard errors
N <- sum(d$number_of_hh)

# Standard errors for y
mm_se <- map_dbl(bins, ~ sqrt(mm_var[.] / N)) %>% signif(3)
mm_se
```

```
##  lowbin highbin meanbin
## 0.00355 0.00627 0.00508
```

```
# Standard errors for log(y)
mm_se_log <- map_dbl(bins, ~ sqrt(mm_var_log[.] / N)) %>% signif(3)
mm_se_log
```

```
##  lowbin highbin meanbin
## 0.00557 0.00543 0.00541
```

```
# Standard errors for E(y)
mm_se_mean <- map_dbl(bins, ~ sqrt(mm_var[.] / N + mm_var[.]^2 / (2 * (N-1)))) %>% signif(3)
mm_se_mean
```

```
##  lowbin highbin meanbin
## 0.00380 0.00755 0.00579
```

Part 2: Maximum Likelihood

The log-normal distribution is defined as:

$$f(y) = \frac{1}{y\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{ln(y)-\mu}{\sigma}\right)^2}$$

The likelihood for the entire sample is:

$$L(\mu, \sigma^2) = \prod_{i=1}^{N} f(y_i|\mu, \sigma^2)$$

The log-likelihood for the entire sample is:

$$\ell(\mu, \sigma^2) = \sum_{i=1}^{N} \ln(f(y_i|\mu, \sigma^2)) = \sum_{i=1}^{N} \ell_i(\mu, \sigma^2)$$

The MLE estimator is then: $\widehat{\theta}_{MLE} = \arg\max_{\mu,\sigma^2} \ell(\mu, \sigma^2)$. The individual score S_i gives us two partial derivatives:

$$S_i = \frac{\partial \ln f(y_i)}{\partial \theta} = \nabla_{\mu,\sigma^2} \ln f(y_i)$$

We should note that the log-likelihood for the log-normal almost looks like the one for the normal distribution except for a constant term that does not depend on the parameters:

$$\ell(\mu, \sigma^2; y) = -\sum_i \ln y_i + \ell_{Normal}(\mu, \sigma^2; \ln(y))$$

Therefore, the MLE estimator for mean and variance look the same:

$$\widehat{\mu} = \frac{\sum \ln y_i}{n}, \qquad \widehat{\sigma}^2 = \frac{\sum (\ln y_i - \widehat{\mu})^2}{n}.$$

I implement them directly:

```
mle_mean <- map_dbl(bins, ~ weighted.mean(log(pull(d, .))), d$weights) %>% round(2)
mle_mean

##  lowbin highbin meanbin
##   10.10   10.43   10.30

mle_var <- map_dbl(bins, ~ Hmisc::wtd.var(log(pull(d, .)), d$weights, method = 'ML')) %>% round(3)
mle_var

##  lowbin highbin meanbin
##   0.707   0.672   0.669
```

The Hessian matrix is reproduced as follows[1] (where the cross-derivatives are zero):

[1] via https://scholarsarchive.byu.edu/cgi/viewcontent.cgi?article=2927&context=etd

$$\frac{\partial^2 \ell_i}{\partial \mu^2} = -\frac{1}{\widehat{\sigma}^2}$$

$$\frac{\partial^2 \ell_i}{\partial (\sigma^2)^2} = -\frac{\ln(y_i - \widehat{\mu})^2}{2 \cdot (\widehat{\sigma}^2)^3} = -\frac{1}{2 \cdot (\widehat{\sigma}^2)^2}$$

We know that $\ln(y_i)$ in the summation term is distributed as $\ln(y_i) \sim N(\widehat{\mu}, \widehat{\sigma}^2)$. We can re-parameterize and use the variance expansion to calculate its expectation σ^2.

We also know that the standard error in MLE can be estimated with the inverse of the Fisher information matrix: $SE(\widehat{\theta}) = \sqrt{diag[\mathbb{E}[-\mathcal{H}]^{-1}]}$. Thus,

$$SE(\widehat{\mu}) = \sqrt{\widehat{\sigma}^2/n}$$

and

$$SE(\widehat{\sigma}^2) = \sqrt{2 \cdot (\widehat{\sigma}^2)^2/n}$$

```
# Standard error for the sample mean
mle_se_mean <- map_dbl(bins, ~ sqrt(mle_var[.] / N)) %>% signif(3)
mle_se_mean

##  lowbin highbin meanbin
## 0.00554 0.00540 0.00539

# Standard error for the sample variance
mle_se_var <- map_dbl(bins, ~ mle_var[.] * sqrt(2 / N)) %>% signif(3)
mle_se_var

##  lowbin highbin meanbin
## 0.00659 0.00627 0.00624
```

The assumption appear to be not very sensitive to the different income bins. The exception is sometimes the lower bin because it does not depend on the assumed top income. The weighting makes, not surprisingly, a great difference.

Part 3: Generalized Method of Moments

Researchers looked into using more of the information contained with the bins with various methods (Eckernkemper and Gribisch 2020). Here, we want to estimate two parameters μ andσ^2 from a log-normal distribution. We have q sample moment conditions, one for each bin and year (121 total, here simplified to 19 bins):

$$g_{\bar{y}}(\theta) \equiv \frac{1}{N_t} \sum_{i=1}^{N_t} g(y_i|\theta) \approx \mathbf{0}$$

We define each moment per bin as suggested in the problem set, although we can use the grouped data by year in our data:

$$g^{\text{bin(18)}}(y|\mu,\sigma) = Pr(40,000 \leq y_i \leq 44,999|\mu,\sigma) - \frac{1}{N_t}\sum_{i=1}^{N_t} I(40,000 \leq y_i \leq 44,999|\mu,\sigma)$$

The theoretical bin membership probability is calculated as the difference of the cumulative distribution function $F(y) = \Phi(\frac{\ln(y)-\mu}{\sigma})$, that is $F(y^{upper}) - F(y^{lower})$.

The Jacobian is defined as the matrix $D \equiv \mathbb{E}[\frac{\partial g(y_i,\theta)}{\partial \theta}]$, which in this case is a 19×2 matrix.[2] We use the chain rule, so the derivative for μ it is:

[2] I average across the individual bins, which is somewhat imprecise for bin 27. Alternatively, I tried to estimate the GMM parameters with 121 moment conditions, which seems very similar but less stable.

$$\begin{aligned}\frac{\partial g_{\bar{y}}}{\partial \mu} &= \frac{F(y^{upper})}{\partial \mu} - \frac{F(y^{lower})}{\partial \mu} - 0 \\ &= \frac{-1}{\sigma}\phi(y^{upper}) + \frac{1}{\sigma}\phi(y^{lower})\end{aligned}$$

And for σ:

$$\begin{aligned}\frac{\partial g_{\bar{y}}}{\partial \sigma} &= \frac{F(y^{upper})}{\partial \sigma} - \frac{F(y^{lower})}{\partial \sigma} - 0 \\ &= -\frac{y^{upper}-\mu}{\sigma^2}\phi(y^{upper}) + \frac{y^{lower}-\mu}{\sigma^2}\phi(y^{lower})\end{aligned}$$

```r
# Prepare the subset of data needed
gmm_data <- d %>% group_by(household_income) %>%
  select(highbin, lowbin, pr) %>%
  summarize(across(everything(), mean)) %>%
  select(-household_income) %>%
  as.matrix
```

```
## Adding missing grouping variables: 'household_income'
```

```r
# The moment function
g_bar <- function(theta, x) {
  mu = theta[1]
  sigma = sqrt(theta[2])
  prob <- plnorm(x[ ,"highbin"], mu, sigma) - plnorm(x[ ,"lowbin"], mu, sigma)
  g <- as.matrix(prob - x[ ,"pr"])
  g
}

# The gradient function
grad_g <- function(theta, x) {
  mu = theta[1]
```

```
  sigma = sqrt(theta[2])
  del_mu <- -1/sigma * dlnorm(x[ ,"highbin"], mu, sigma) + 1/sigma * dlnorm(x[ ,"lowbin"], mu, sigma)
  del_sigma <- - (x[ ,"highbin"] - mu)/sigma^2 * dlnorm(x[ ,"highbin"], mu, sigma) +
    (x[ ,"lowbin"] - mu)/sigma^2 * dlnorm(x[ ,"lowbin"], mu, sigma)
 J <- cbind(del_mu, del_sigma)
 J
}
```

We can check whether the expectation of $g(y_i, \theta)$ is close to zero with previous estimates:

```
g_bar(c(mle_mean[3], mle_var[3]), gmm_data) %>% psych::describe()
```

```
##    vars  n mean   sd median trimmed  mad   min  max range skew kurtosis   se
## X1    1 19    0 0.07   0.01    0.01 0.04 -0.24 0.08  0.31   -2     4.12 0.02
```

The GMM estimator minimizes the following objective function:

$$\hat{\theta} = \arg\min_{\theta} \; g_{\bar{y}}(\theta)' \, W \, g_{\bar{y}}(\theta)$$

```
# This is extremely unstable.
g_analy <- gmm::gmm(g = g_bar,
              x = gmm_data,
              t0 = c(11, 0.5),
              gradv = grad_g,
              vcov = "iid",
              type = "twoStep")

g_analy
```

```
## Method
##  twoStep
##
## Objective function value:  0.0001382518
##
##   Theta[1]    Theta[2]
## 12.5958778   0.0014114
##
## Convergence code =  0
```

```
# Manual GMM first-step
gmm_fir <- function(theta, x = gmm_data) {
  G <- g_bar(theta, x)
  W <- diag(nrow(G)) # identity matrix
  Q = t(G) %*% W %*% G
}
```

```
theta_fir <- optim(c(10, 0.5), gmm_fir, method = c("BFGS"))$par
theta_fir
```

```
## [1] 11.2281238  0.3683312
```

```
gmm_mean_1 <- theta_fir[1]
gmm_var_1 <- theta_fir[2]
```

Griffiths and Hajargasht (2015) show, as far as I can tell, that the optimal weighting matrix $\widehat{W}$ is a diagonal matrix with the inverse of the population moments.

```
# Manual GMM second-step
W_hat <- function(x = gmm_data, par = theta_fir) {
  mu = par[1]
  sigma = sqrt(par[2])
  W <- diag(nrow(x))
  prob <- plnorm(x[ ,"highbin"], mu, sigma) - plnorm(x[ ,"lowbin"], mu, sigma)
  W_hat = W * (1 / prob)
  W_hat
}

gmm_sec <- function(theta, x = gmm_data) {
  G <- g_bar(theta, x)
  W <- W_hat(x)
  Q = t(G) %*% W %*% G
}

theta_sec <- optim(theta_fir, gmm_sec)$par
theta_sec
```

```
## [1] 11.300613  1.569673
```

```
gmm_mean_2 <- theta_sec[1] %>% round(2)
gmm_var_2 <- theta_sec[2] %>% round(2)
```

Finally, the standard errors for both parameters:

```
gmm_se_fun <- function(theta, x = gmm_data){
  N <- sum(d$number_of_hh)
  W <- W_hat(x)
  S <- solve(W)
  D <- grad_g(theta, x) %>% t()

  bread = solve(D %*% W %*% t(D))
  fill = D %*% W %*% S %*% t(W) %*% t(D)
  V =  bread %*% fill %*% bread
```

```
  se = sqrt(diag(V)) / N # somehow adjust for big N
  se
}

gmm_se <- gmm_se_fun(theta_sec) %>% set_names(c("se_mean", "se_var"))
gmm_se

##      se_mean       se_var
## 1.215255e-01 2.652388e-05

gmm_se_mean <- gmm_se[1] %>% round(2)
gmm_se_var <- gmm_se[2] %>% round(2)
```

For the table I restrict myself to the mean bins.

```
# Table 1
tibble(method = c("MM", "MLE", "GMM"),
       mean = c(mm_mean[3], mle_mean[3], gmm_mean_2),
       std.err.mean = c(mm_se[3], mle_se_mean[3], gmm_se_mean),
       var = c(mm_var[3], mle_var[3], gmm_var_2)) %>%
  kableExtra::kbl(booktabs = T)
```

method	mean	std.err.mean	var
MM	10.94	0.00508	0.594
MLE	10.30	0.00539	0.669
GMM	11.30	0.12000	1.570

```
# Table 2
m_means <- c(mm_mean[3], mle_mean[3], gmm_mean_2)
m_vars <- c(mm_var[3], mle_var[3], gmm_var_2)

# The big log-normal trap?!
means <- map2_dbl(m_means, m_vars, ~ exp(.x + .y / 2) )
medians <- map_dbl(m_means, ~ exp(.x) )
q10 <- map2_dbl(m_means, m_vars, ~ qlnorm(0.1, .x, sqrt(.y)) )
q90 <- map2_dbl(m_means, m_vars, ~ qlnorm(0.9, .x, sqrt(.y)) )

tibble(method = c("MM", "MLE", "GMM"),
       `mean income` = scales::dollar(round(means, -2)),
       `median income` = scales::dollar(round(medians, -2)),
       q10 = scales::dollar(round(q10, -2)),
       q90 = scales::dollar(round(q90, -2))) %>%
  kableExtra::kbl(booktabs = T, align = 'r')
```

method	mean income	median income	q10	q90
MM	$75,900	$56,400	$21,000	$151,400
MLE	$41,500	$29,700	$10,400	$84,800
GMM	$177,200	$80,800	$16,200	$402,600

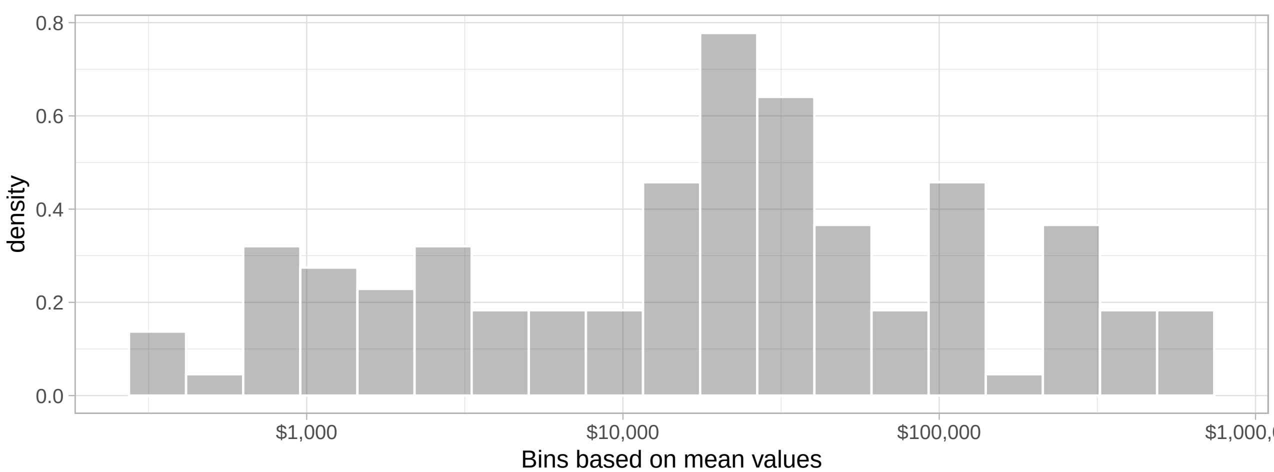
The three estimates vary widely: method of moments falls somewhere between the conservative MLE and the wide-ranged GMM. The standard errors for MM and MLE are similar, but GMM is larger by several magnitudes.

Part 4: Nonparametric Estimates

I use the mean bin estimate for the income distribution.

```
hist <- ggplot(d, aes(meanbin_w)) +
  geom_histogram(aes(y = stat(density)),
                 bins = 19,
                 color = "white",
                 alpha = 0.4) +
  scale_x_log10(labels = scales::dollar_format()) +
  xlab("Bins based on mean values") +
   theme_light()
```

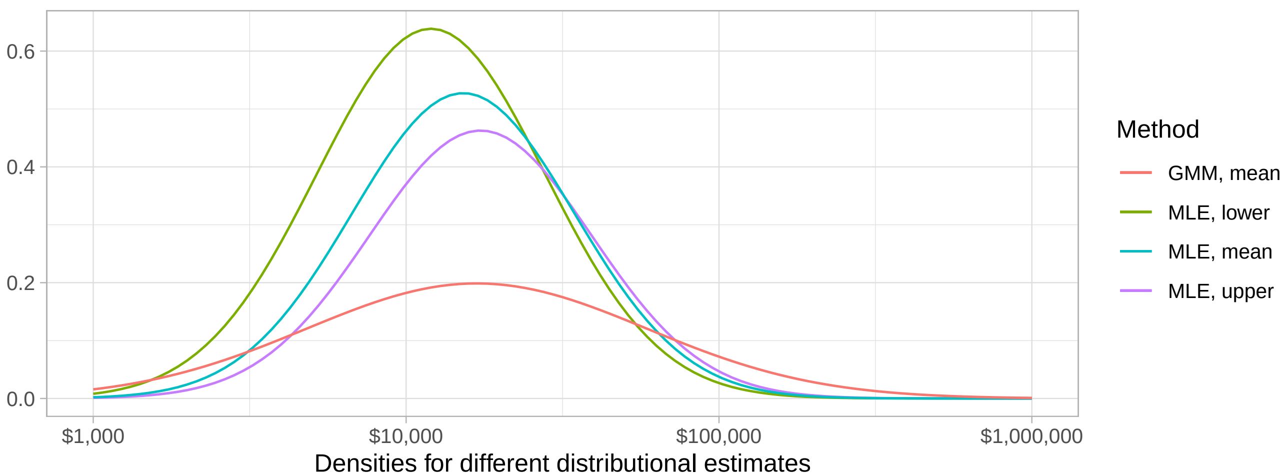
```
hist
```



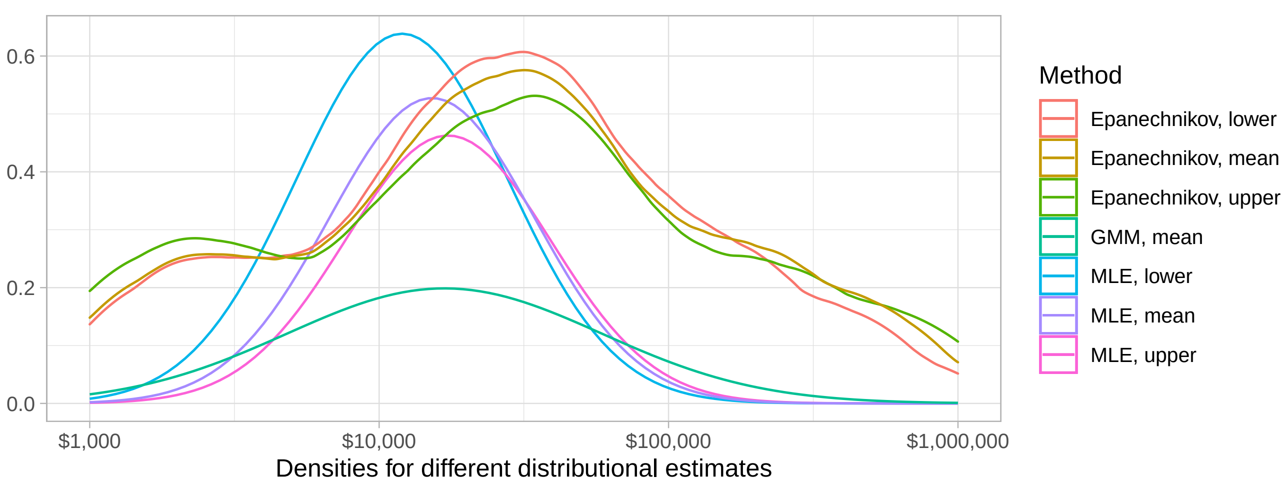

```
dens <- ggplot() +
  geom_function(fun = dlnorm, args = list(mle_mean[1], sqrt(mle_var[1])), aes(y = after_stat(y*N), color =
  geom_function(fun = dlnorm, args = list(mle_mean[2], sqrt(mle_var[2])), aes(y = after_stat(y*N), color =
  geom_function(fun = dlnorm, args = list(mle_mean[3], sqrt(mle_var[3])), aes(y = after_stat(y*N), color =
```

```
  geom_function(fun = dlnorm, args = list(gmm_mean_2, sqrt(gmm_var_2)), aes(y = after_stat(y*N), color = "
  scale_x_log10(labels = scales::dollar_format(), limits = c(1e3, 1e6)) +
  labs(x = "Densities for different distributional estimates",
       y = NULL,
       color = "Method") +
  theme_light()
```

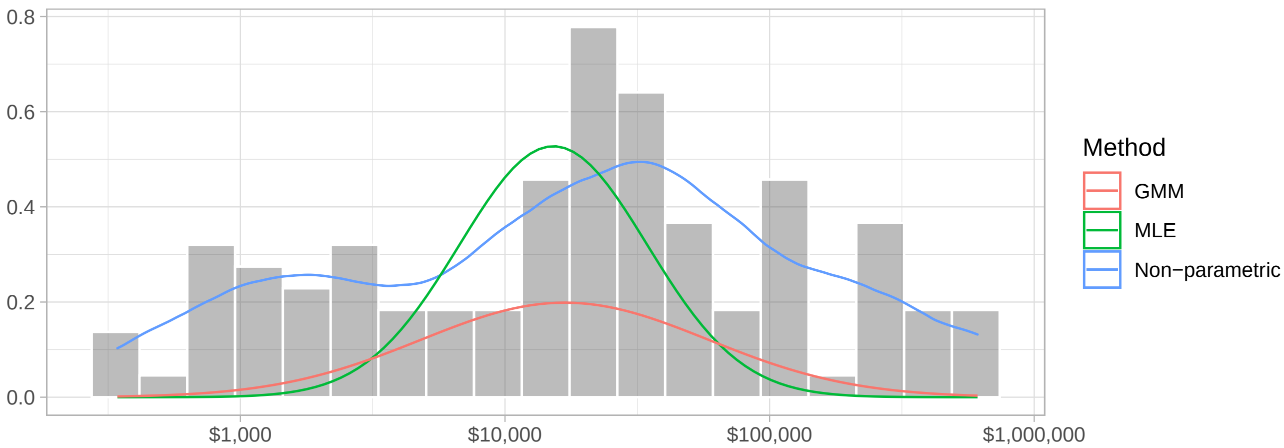
```
dens
```




```
dens +
  geom_density(data = d, kernel = "epanechnikov", aes(x = lowbin_w, color = "Epanechnikov, lower")) +
    geom_density(data = d, kernel = "epanechnikov", aes(x = highbin_w, color = "Epanechnikov, upper")) +
    geom_density(data = d, kernel = "epanechnikov", aes(x = meanbin_w, color = "Epanechnikov, mean"))
```



```
hist +
    geom_density(data = d, kernel = "epanechnikov", aes(x = meanbin_w, color = "Non-parametric"))  +
  geom_function(fun = dlnorm, args = list(mle_mean[3], sqrt(mle_var[3])), aes(y = after_stat(y*N), color =
  geom_function(fun = dlnorm, args = list(gmm_mean_2, sqrt(gmm_var_2)), aes(y = after_stat(y*N), color = "
    labs(x = "Income distribution estimated with three different methods on top of a histogram",
       y = NULL,
       color = "Method")
```



Income distribution estimated with three different methods on top of a histogram

In conclusion, we note:

- The empirical density appears to fit the slight bimodal distribution. Such a distribution cannot be perfectly replicated with a log-normal distribution.
- The MLE estimate is concentrated around a lower income median, has a lower standard deviation, and in that sense more robust statistics.
- The GMM estimate better approximates potentially the unknown tail of the income distribution, but at the cost of a larger variance and standard error.

References

Eckernkemper, Tobias, and Bastian Gribisch. 2020. "Classical and Bayesian Inference for Income Distributions Using Grouped Data." *Oxford Bulletin of Economics and Statistics* 83 (1): 32–65. https://doi.org/10.1111/obes.12396.

Griffiths, William, and Gholamreza Hajargasht. 2015. "On GMM Estimation of Distributions from Grouped Data." *Economics Letters* 126 (January): 122–26. https://doi.org/10.1016/j.econlet.2014.11.031.