Atkinson Inequality Index and Utility Family

Fan Wang

2020-06-12

Contents

1 Atkinson Inequality Index

Go to the **RMD**, **R**, **PDF**, or **HTML** version of this file. Go back to fan's REconTools Package, R Code Examples Repository (bookdown site), or Intro Stats with R Repository (bookdown site).

1.1 Atkinson Inequality Measures

Atkinson (JET, 1970) studies five standard inequality measures. Atkinson finds that given the same income data across countries, different inequality measure lead to different rankings of which country is more unequal. Atkinson develops an measure of inequality that changes depending on an inequality aversion parameter.

$$\text{Atkinson Inequality} = A\left(\left\{Y_i\right\}_{i=1}^{N}, \lambda\right) = 1 - \left(\sum_{i=1}^{N} \frac{1}{N}\left(\frac{Y_i}{\sum_{j=1}^{N}\left(\frac{Y_j}{N}\right)}\right)^{\lambda}\right)^{\frac{1}{\lambda}} \in [0,1]$$

$A\left(\left\{Y_i\right\}_{i=1}^{N}, \lambda\right)$ equals to zero is perfect equality. 1 is Perfect inequality. If $\lambda = 1$, the inequality measure is always equal to 0 because the planner does not care abouot inequality anymore.

1.2 Atkinson Inequality Function

Programming up the equation above, we have, given a sample of data:

```r
# Formula
ffi_atkinson_ineq <- function(ar_data, fl_rho) {
  ar_data_demean <- ar_data/mean(ar_data)
  it_len <- length(ar_data_demean)
  fl_atkinson <- 1 - sum(ar_data_demean^{fl_rho}*(1/it_len))^(1/fl_rho)
  return(fl_atkinson)
}
```

When each element of the data array has weight, we have:

```r
# Formula
ffi_atkinson_random_var_ineq <- function(ar_data, ar_prob_data, fl_rho) {
  #' @param ar_data array sorted array values
  #' @param ar_prob_data array probability mass for each element along `ar_data`, sums to 1
  #' @param fl_rho float inequality aversion parameter fl_rho = 1 for planner
  #' without inequality aversion. fl_rho = -infinity for fully inequality averse.

  fl_mean <- sum(ar_data*ar_prob_data);
  fl_atkinson <- 1 - (sum(ar_prob_data*(ar_data^{fl_rho}))^(1/fl_rho))/fl_mean
  return(fl_atkinson)
}
```

1.3 Atkinson Inequality Examples

Given a vector of observables, compute the Atkinson inequality measure given different inequality aversion.

1.3.1 Data Samples and Weighted Data

The ρ preference vector.

```r
# Preference Vector
ar_rho <- 1 - (10^(c(seq(-2.0,2.0, length.out=30))))
ar_rho <- unique(ar_rho)
mt_rho <- matrix(ar_rho, nrow=length(ar_rho), ncol=1)
```

Sampled version for N sampled points, random, uniform and one-rich.

```r
# Random normal Data Vector (not equal outcomes)
set.seed(123)
it_sample_N <- 30
fl_rnorm_mean <- 100
fl_rnorm_sd <- 6
ar_data_rnorm <- rnorm(it_sample_N, mean=fl_rnorm_mean, sd=fl_rnorm_sd)
# Uniform Data Vector (Equal)
ar_data_unif <- rep(1, length(ar_data_rnorm))

# One Rich (last person has income equal to the sum of all others*100)
ar_data_onerich <- rep(0.1, length(ar_data_rnorm))
ar_data_onerich[length(ar_data_onerich)] = sum(head(ar_data_onerich,-1))*10
```

Given the same distributions, random, uniform and one-rich, generate discrete random variable versions below. We approximate continuous normal with discrete binomial:

```r
# Use binomial to approximate normal
fl_p_binom <- 1 - fl_rnorm_sd^2/fl_rnorm_mean
fl_n_binom <- round(fl_rnorm_mean^2/(fl_rnorm_mean - fl_rnorm_sd^2))
fl_binom_mean <- fl_n_binom*fl_p_binom
fl_binom_sd <- sqrt(fl_n_binom*fl_p_binom*(1-fl_p_binom))
# drv = discrete random variable
ar_drv_rbinom_xval <- seq(1, fl_n_binom)
ar_drv_rbinom_prob <- dbinom(ar_drv_rbinom_xval, size=fl_n_binom, prob=fl_p_binom)
```

```
# ignore weight at x=0
ar_drv_rbinom_prob <- ar_drv_rbinom_prob/sum(ar_drv_rbinom_prob)
```

Additionally, for the one-rich vector created earlier, now consider several probability mass over them, change the weight assigned to the richest person.

```
# This should be the same as the unweighted version
ar_drv_onerich_prob_unif <- rep(1/it_sample_N, it_sample_N)
# This puts almost no weight on the last rich person
# richlswgt = rich less weight
ar_drv_onerich_prob_richlswgt <- ar_drv_onerich_prob_unif
ar_drv_onerich_prob_richlswgt[it_sample_N] <- (1/it_sample_N)*0.1
ar_drv_onerich_prob_richlswgt <- ar_drv_onerich_prob_richlswgt/sum(ar_drv_onerich_prob_richlswgt)
# This puts more weight on the rich person
# richmrwgt = rich more weight
ar_drv_onerich_prob_richmrwgt <- ar_drv_onerich_prob_unif
ar_drv_onerich_prob_richmrwgt[it_sample_N] <- (1/it_sample_N)*10
ar_drv_onerich_prob_richmrwgt <- ar_drv_onerich_prob_richmrwgt/sum(ar_drv_onerich_prob_richmrwgt)
```

1.3.2 Testing Atkinson Index at Single rho Value

Atkinson index with $\rho = -1$, which is a planner with some aversion towards inequality, this is equivalent to CRRA=2.

Testing with normal sample draw vs normal approximated with binomial discrete random variable:

```
# ATK = 0.05372126
ffi_atkinson_ineq(ar_data_rnorm, -1)
```

```
## [1] 0.00335581
```

```
# ATK = 0.03443246
ffi_atkinson_random_var_ineq(ar_drv_rbinom_xval, ar_drv_rbinom_prob, -1)
```

```
## [1] 0.003642523
```

```
# ATK = 0
ffi_atkinson_ineq(ar_data_unif, -1)
```

```
## [1] 0
```

Testing with

```
# ATK = 0.89, sample
ffi_atkinson_ineq(ar_data_onerich, -1)
```

```
## [1] 0.9027248
```

```
# ATK = 0.89, drv, uniform weight
ffi_atkinson_random_var_ineq(ar_data_onerich, ar_drv_onerich_prob_unif, -1)
```

```
## [1] 0.9027248
```

```
# ATK = 0.49, drv, less weight on rich
ffi_atkinson_random_var_ineq(ar_data_onerich, ar_drv_onerich_prob_richlswgt, -1)
```

```
## [1] 0.4965518
```

```
# ATK = 0.97, drv, more weight on rich
ffi_atkinson_random_var_ineq(ar_data_onerich, ar_drv_onerich_prob_richmrwgt, -1)
```

```
## [1] 0.9821147
```

Create vector of inequality aversion parameters and graph legends.

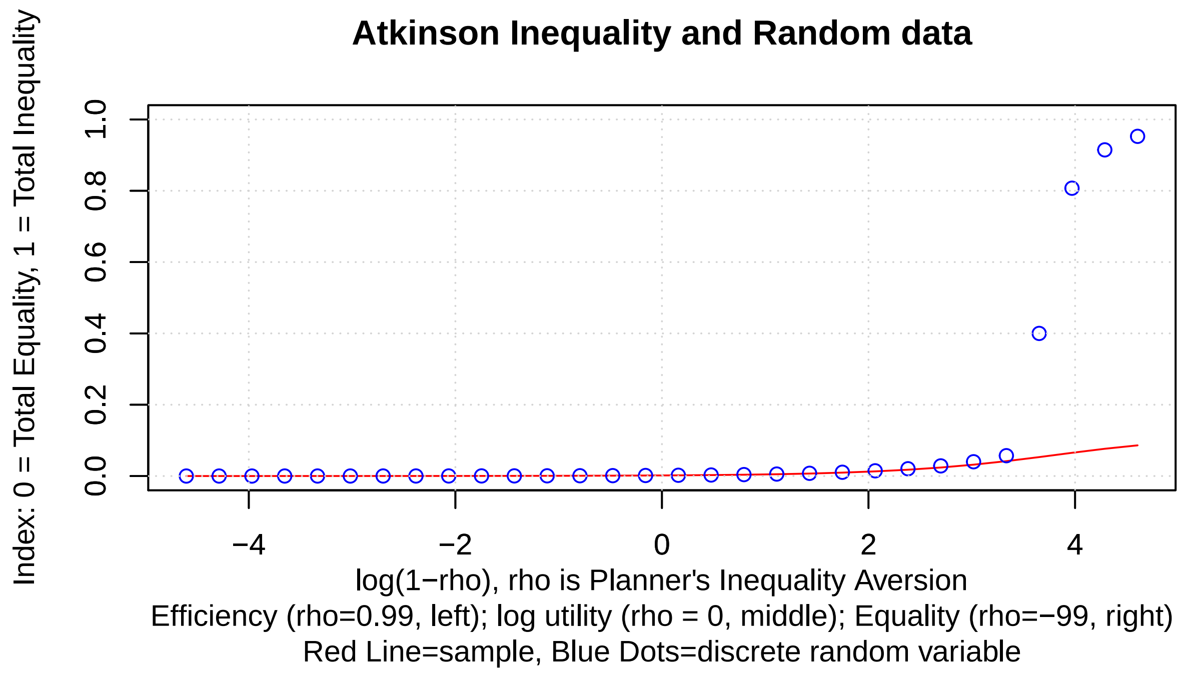
```
ar_log_1_minus_rho <- log(1-ar_rho)
st_x_label <- 'log(1-rho), rho is Planner\'s Inequality Aversion\nEfficiency (rho=0.99, left); log utili
st_y_label <- 'Index: 0 = Total Equality, 1 = Total Inequality'
```

1.3.3 Atkinson Inequality and Normally Distributed Data

How does Atkinson Inequality measure change with respect to a vector of normal random data as inequality aversion shifts? Note that in the example below, the bionomial approximated version is very similar to the normally drawn sample version. until rho beocmes very negative.

At very negative rho values, the binomial approximation has very tiny, but positive mass for all small values starting from 0, the normally drawn sample has no mass at those points. The Atkinson inequality planner increasingly only cares about the individual with the lowest value from the binomial approximated version, and given the low value of those individuals compared to others, despite having no mass, the inequality index is almost 1.

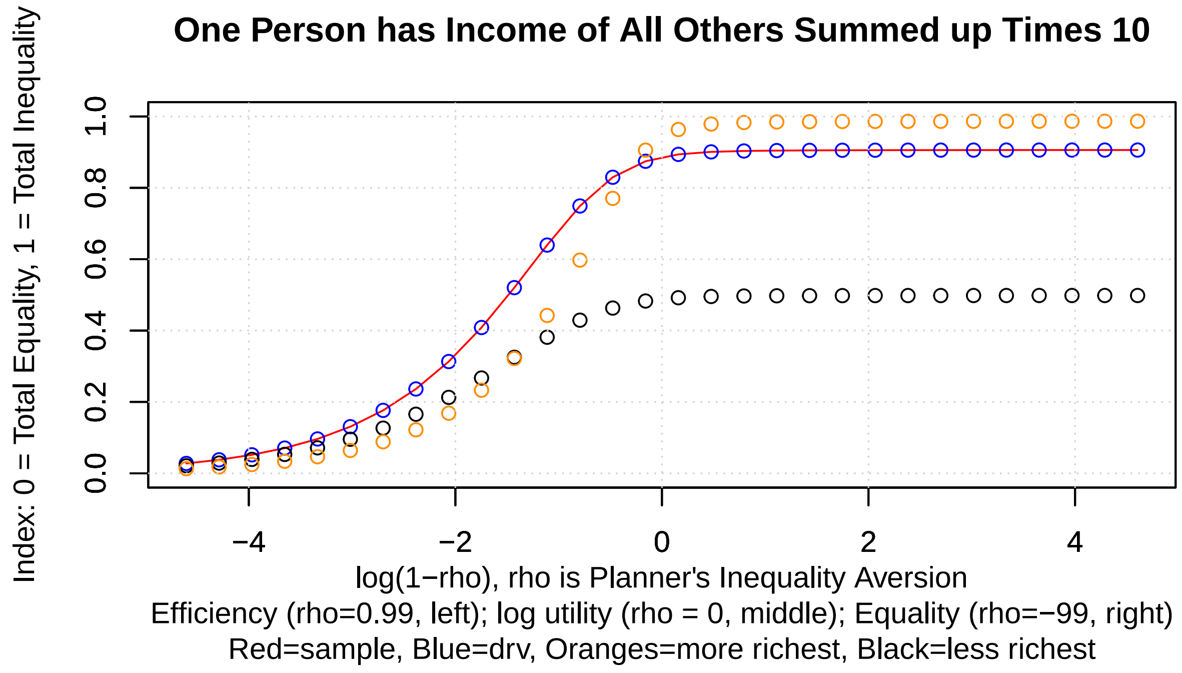
```
ar_ylim = c(0,1)
# First line
par(new=FALSE)
ar_atkinson_sample <- apply(mt_rho, 1, function(row){
  ffi_atkinson_ineq(ar_data_rnorm, row[1])})
plot(ar_log_1_minus_rho, ar_atkinson_sample,
     ylim = ar_ylim, xlab = st_x_label, ylab = st_y_label,
     type="l", col = 'red')
# Second line
par(new=T)
ar_atkinson_drv <- apply(mt_rho, 1, function(row){
  ffi_atkinson_random_var_ineq(ar_drv_rbinom_xval, ar_drv_rbinom_prob, row[1])})
plot(ar_log_1_minus_rho, ar_atkinson_drv,
     ylim = ar_ylim, xlab = '', ylab = '',
     type="p", col = 'blue')
# Title
title(main = 'Atkinson Inequality and Random data',
      sub = 'Red Line=sample, Blue Dots=discrete random variable')
grid()
```



1.3.4 Atkinson Inequality with an Extremely Wealthy Individual

Now with the one person has the wealth of all others in the vector times 10.

```r
# First line
par(new=FALSE)
ar_atkinson <- apply(mt_rho, 1, function(row){ffi_atkinson_ineq(
  ar_data_onerich, row[1])})
plot(ar_log_1_minus_rho, ar_atkinson,
     ylim = ar_ylim, xlab = st_x_label, ylab = st_y_label,
     type="l", col = 'red')
# Second line
par(new=T)
ar_atkinson_drv <- apply(mt_rho, 1, function(row){ffi_atkinson_random_var_ineq(
  ar_data_onerich, ar_drv_onerich_prob_unif, row[1])})
plot(ar_log_1_minus_rho, ar_atkinson_drv,
     ylim = ar_ylim, xlab = '', ylab = '',
     type="p", col = 'blue')
# Third line
par(new=T)
ar_atkinson_drv_richlswgt <- apply(mt_rho, 1, function(row){ffi_atkinson_random_var_ineq(
  ar_data_onerich, ar_drv_onerich_prob_richlswgt, row[1])})
plot(ar_log_1_minus_rho, ar_atkinson_drv_richlswgt,
     ylim = ar_ylim, xlab = '', ylab = '',
     type="p", col = 'black')
# Fourth line
par(new=T)
ar_atkinson_drv_richmrwgt <- apply(mt_rho, 1, function(row){ffi_atkinson_random_var_ineq(
  ar_data_onerich, ar_drv_onerich_prob_richmrwgt, row[1])})
plot(ar_log_1_minus_rho, ar_atkinson_drv_richmrwgt,
     ylim = ar_ylim, xlab = '', ylab = '',
     type="p", col = 'darkorange')
# Title
title(main = 'One Person has Income of All Others Summed up Times 10',
      sub  = 'Red=sample, Blue=drv, Oranges=more richest, Black=less richest')
grid()
```



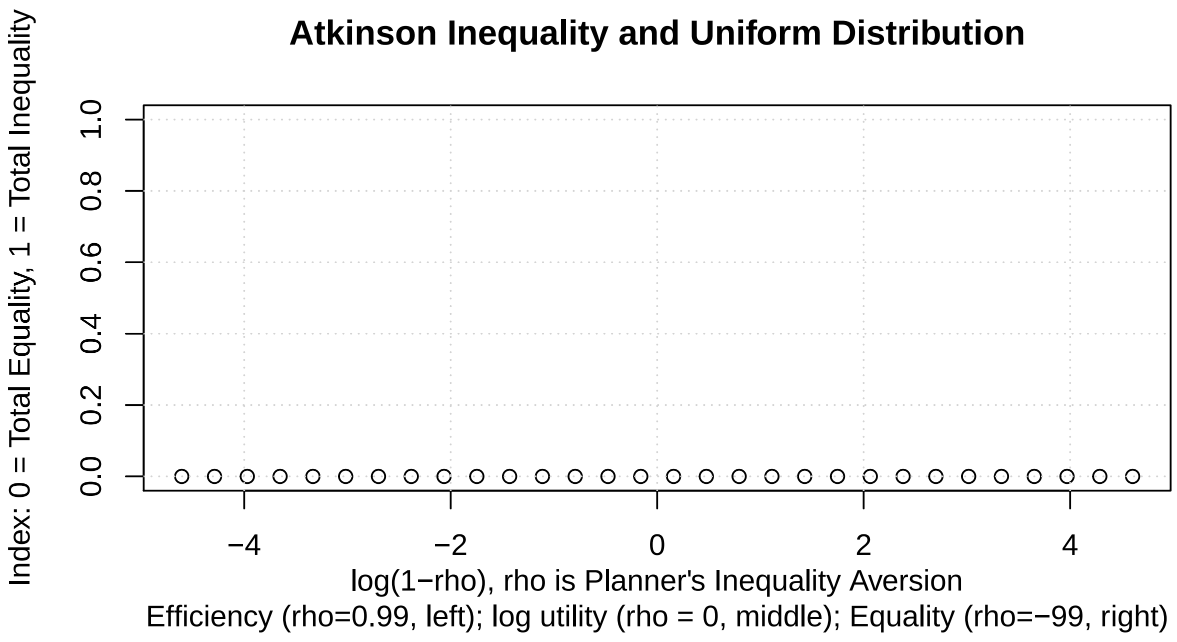
1.3.5 Atkinson Inequality with an Uniform Distribution

The Uniform Results, since allocations are uniform, zero for all.

```
par(new=FALSE)
ffi_atkinson_ineq(ar_data_unif, -1)
```

```
## [1] 0
```

```
ar_atkinson <- apply(mt_rho, 1, function(row){ffi_atkinson_ineq(ar_data_unif, row[1])})
plot(ar_log_1_minus_rho, ar_atkinson, ylim = ar_ylim, xlab = st_x_label, ylab = st_y_label)
title(main = 'Atkinson Inequality and Uniform Distribution')
grid()
```



1.4 Analyzing Equation Mechanics

How does the Aktinson Family utility function work? THe Atkinson Family Utility has the following functional form.

$$V^{\text{social}} = \left(\alpha \cdot A^{\lambda} + \beta \cdot B^{\lambda}\right)^{\frac{1}{\lambda}}$$

Several key issues here:

1. V^{social} is the utility of some social planner
2. A and B are allocations for Alex and Ben.
3. α and β are biases that a social planner has for Alex and Ben: $\alpha + \beta = 1$, $\alpha > 0$, and $\beta > 0$
4. $-\infty < \lambda \leq 1$ is a measure of inequality aversion
   - $\lambda = 1$ is when the planner cares about weighted total allocations (efficient, Utilitarian)
   - $\lambda = -\infty$ is when the planner cares about only the minimum between A and B allocations (equality, Rawlsian)

What if only care about Alex? Clearly, if the planner only cares about Ben, $\beta = 1$, then:

$$V^{\text{social}} = \left(B^{\lambda}\right)^{\frac{1}{\lambda}} = B$$

Clearly, regardless of the value of λ, as B increases V increases. What Happens to V when A or B increases? What is the derivative of V with respect to A or B?

$$\frac{\partial V}{\partial A} = \frac{1}{\lambda}\left(\alpha A^{\lambda} + \beta B^{\lambda}\right)^{\frac{1}{\lambda}-1} \cdot \lambda \alpha A^{\lambda - 1}$$

$$\frac{\partial V}{\partial A} = \left(\alpha A^{\lambda} + \beta B^{\lambda}\right)^{\frac{1-\lambda}{\lambda}} \cdot \alpha A^{\lambda - 1} > 0$$

Note that $\frac{\partial V}{\partial A} > 0$. When $\lambda < 0$, $Z^{\lambda} > 0$. For example $10^{-2} = \frac{1}{100}$. And For example $0.1^{\frac{3}{-2}} = \frac{1}{0.1^{1.5}}$. Still Positive.

While the overall V increases with increasing A, but if we did not have the outter power term, the situation is different. In particular, when $\lambda < 0$:

$$\text{if } \lambda < 0 \text{ then } \frac{d\left(\alpha A^{\lambda} + \beta B^{\lambda}\right)}{dA} = \alpha \lambda A^{\lambda - 1} < 0$$

Without the outter $\frac{1}{\lambda}$ power, negative λ would lead to decreasing weighted sum. But:

$$\text{if } \lambda < 0 \text{ then } \frac{dG^{\frac{1}{\lambda}}}{dG} = \frac{1}{\lambda} \cdot G^{\frac{1-\lambda}{\lambda}} < 0$$

so when G is increasing and $\lambda < 0$, V would decrease. But when $G(A, B)$ is decreasing, as is the case with increasing A when $\lambda < 0$, V will actually increase. This confirms that $\frac{\partial V}{\partial A} > 0$ for $\lambda < 0$. The result is symmetric for $\lambda > 0$.

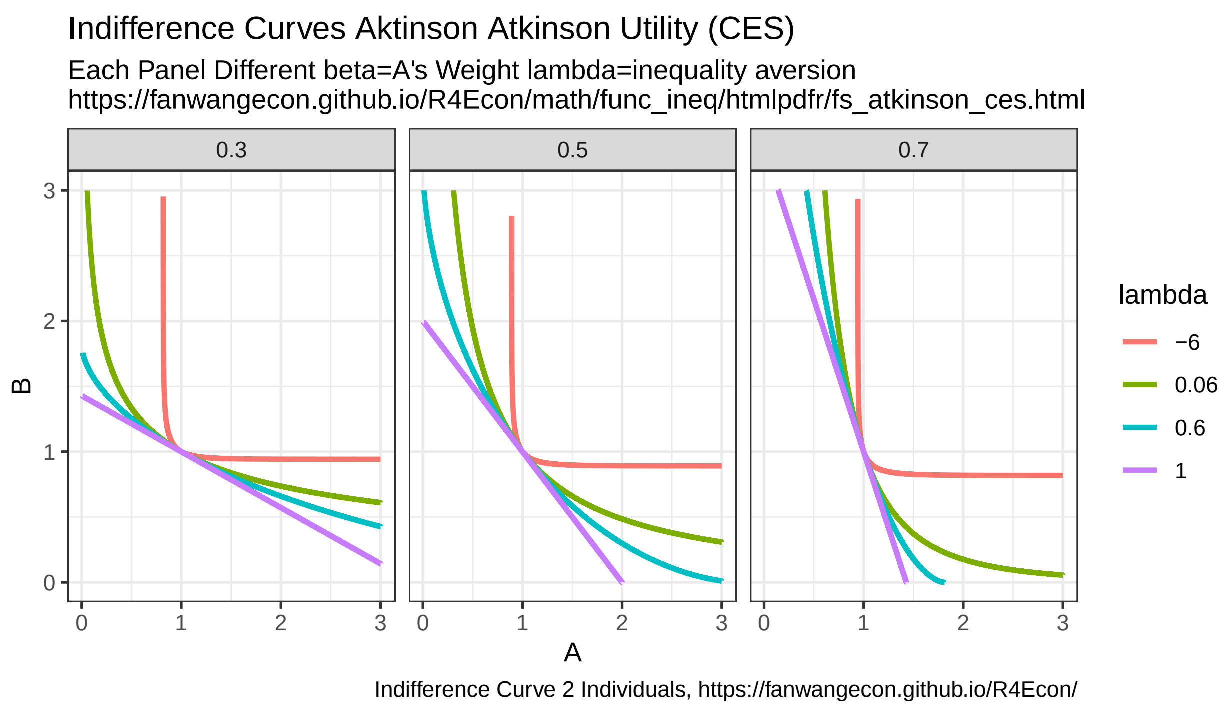
1.5 Indifference Curve Graph

Given V^*, we can show the combinations of A and B points that provide the same utility. We want to be able to potentially draw multiple indifference curves at the same time. Note that indifference curves are defined by α, λ only. Each indifference curve is a set of A and B coordinates. So to generate multiple indifference curves means to generate many sets of A, B associated with different planner preferences, and then these could be graphed out.

```
# A as x-axis, need bounds on A
fl_A_min = 0.01
fl_A_max = 3
it_A_grid = 50000

# Define parameters
# ar_lambda <- 1 - (10^(c(seq(-2,2, length.out=3))))
ar_lambda <- c(1, 0.6, 0.06, -6)
ar_beta <- seq(0.25, 0.75, length.out = 3)
ar_beta <- c(0.3, 0.5, 0.7)
ar_v_star <- seq(1, 2, length.out = 1)
tb_pref <- as_tibble(cbind(ar_lambda)) %>%
  expand_grid(ar_beta) %>% expand_grid(ar_v_star) %>%
  rename_all(~c('lambda', 'beta', 'vstar')) %>%
  rowid_to_column(var = "indiff_id")

# Generate indifference points with apply and anonymous function
# tb_pref, whatever is selected from it, must be all numeric
# if there are strings, would cause conversion error.
ls_df_indiff <- apply(tb_pref, 1, function(x){
  indiff_id <- x[1]
  lambda <- x[2]
  beta <- x[3]
  vstar <- x[4]
  ar_fl_A_indiff <- seq(fl_A_min, fl_A_max, length.out=it_A_grid)
  ar_fl_B_indiff <- (((vstar^lambda) -
                      (beta*ar_fl_A_indiff^(lambda)))/(1-beta))^(1/lambda)
  mt_A_B_indiff <- cbind(indiff_id, lambda, beta, vstar,
                         ar_fl_A_indiff, ar_fl_B_indiff)
  colnames(mt_A_B_indiff) <- c('indiff_id', 'lambda', 'beta', 'vstar',
                                'indiff_A', 'indiff_B')
  tb_A_B_indiff <- as_tibble(mt_A_B_indiff) %>%
    rowid_to_column(var = "A_grid_id") %>%
    filter(indiff_B >= 0 & indiff_B <= max(ar_fl_A_indiff))
  return(tb_A_B_indiff)
})
df_indiff <- do.call(rbind, ls_df_indiff) %>% drop_na()
```

Note that many more A grid points are needed to fully plot out the leontief line.

```
# Labeling
st_title <- paste0('Indifference Curves Aktinson Atkinson Utility (CES)')
st_subtitle <- paste0('Each Panel Different beta=A\'s Weight lambda=inequality aversion\n',
                      'https://fanwangecon.github.io/',
                      'R4Econ/math/func_ineq/htmlpdfr/fs_atkinson_ces.html')
st_caption <- paste0('Indifference Curve 2 Individuals, ',
                     'https://fanwangecon.github.io/R4Econ/')
st_x_label <- 'A'
st_y_label <- 'B'

# Graphing
plt_indiff <-
  df_indiff %>% mutate(lambda = as_factor(lambda),
                       beta = as_factor(beta),
                       vstar = as_factor(vstar)) %>%
  ggplot(aes(x=indiff_A, y=indiff_B,
             colour=lambda)) +
  facet_wrap( ~ beta) +
  geom_line(size=1) +
  labs(title = st_title, subtitle = st_subtitle,
       x = st_x_label, y = st_y_label, caption = st_caption) +
  theme_bw()

# show
print(plt_indiff)
```


Indifference Curves Aktinson Atkinson Utility (CES)
Each Panel Different beta=A's Weight lambda=inequality aversion
https://fanwangecon.github.io/R4Econ/math/func_ineq/htmlpdfr/fs_atkinson_ces.html
0.3
0.5
0.7
B
A
0
1
2
3
lambda
−6
0.06
0.6
1
Indifference Curve 2 Individuals, https://fanwangecon.github.io/R4Econ/