Asset Prices

```
data.file <- file.path(data.dir, "Reproducible Finance.csv")

symbols <- c("SPY", "EFA", "IJS", "EEM", "AGG")

# Yahoo! Finance

prices <- getSymbols(symbols,
             from = "2012-12-31",
             to = "2017-12-31",
             auto.assign = T,
             warnings = F) %>%
          map(~Ad(get(.))) %>%
  reduce(merge) %>%
  `colnames<-`(symbols)
```

```
'getSymbols' currently uses auto.assign=TRUE by default, but will
use auto.assign=FALSE in 0.5-0. You will still be able to use
'loadSymbols' to automatically load data. getOption("getSymbols.env")
and getOption("getSymbols.auto.assign") will still be checked for
alternate defaults.

This message is shown once per session and may be disabled by setting
options("getSymbols.warning4.0"=FALSE). See ?getSymbols for details.
```

```
# CSV

prices <- read_csv(data.file,
                   col_types =
                     cols(date =
                            col_date(format = "%Y-%m-%d"))) %>%
  tk_xts(date_var = date)
```

```
Warning in tk_xts_.data.frame(data = data, select = select, date_var =
date_var, : Non-numeric columns being dropped: date
```

```
# Excel

prices <-
  read_excel("Reproducible Finance.xlsx",
             col_types = c("text", "numeric",
                           "numeric", "numeric",
                           "numeric", "numeric")) %>%
  mutate(date = ymd(date)) %>%
  tk_xts(date_var = date)
```

```
Warning in tk_xts_.data.frame(data = data, select = select, date_var =
date_var, : Non-numeric columns being dropped: date
```

```
# Verfiy Import
head(prices, 3)
```

```
                SPY      EFA      IJS      EEM      AGG
2012-12-31 128.3092 49.16410 75.06590 39.89233 98.19626
2013-01-02 131.5977 49.92501 77.12553 40.67489 98.08131
2013-01-03 131.3004 49.44080 77.02349 40.38705 97.83374
```

```
# convert to monthly prices.

prices_monthly <- to.monthly(prices,
                             indexAt = "lastof",
                             OHLC = F)

head(prices_monthly)
```

```
                SPY      EFA      IJS      EEM      AGG
2012-12-31 128.3092 49.16410 75.06590 39.89233 98.19626
2013-01-31 134.8773 50.99717 79.08315 39.77539 97.58625
2013-02-28 136.5982 50.34004 80.37274 38.86691 98.16285
2013-03-31 141.7850 50.99717 83.67441 38.47113 98.25957
2013-04-30 144.5090 53.55654 83.77677 38.93888 99.21130
2013-05-31 147.9209 51.93964 87.36826 37.05894 97.22598
```

```
# Convert to monthly returns, xts.

asset_returns_xts <-
  Return.calculate(prices_monthly,
                   method = "log") %>%
  na.omit()

head(asset_returns_xts, 3)
```

```
                  SPY         EFA        IJS          EEM          AGG
2013-01-31 0.04992297  0.03660636 0.05213343 -0.002935495 -0.006231517
2013-02-28 0.01267831 -0.01296938 0.01617522 -0.023105260  0.005891222
2013-03-31 0.03726793  0.01296938 0.04025808 -0.010235026  0.000984796
```

```
# Convert to monthly returns, dplyr.

asset_returns_dplyr_byhand <-
  prices %>%
  to.monthly(indexAt = "lastof", OHLC = F) %>%
  # convert the index to a date
  data.frame(date = index(.)) %>%
```

```
  # now remove the index because it got converted to row names
  remove_rownames() %>%
  gather(asset, prices, - date) %>%
  group_by(asset) %>%
  mutate(returns = (log(prices) - log(lag(prices)))) %>%
  select(-prices) %>%
  spread(asset, returns) %>%
  select(date, symbols)

head(asset_returns_dplyr_byhand)
```

```
# A tibble: 6 x 6
  date            SPY     EFA      IJS      EEM       AGG
  <date>        <dbl>   <dbl>    <dbl>    <dbl>     <dbl>
1 2012-12-31 NA      NA      NA       NA       NA
2 2013-01-31  0.0499  0.0366  0.0521  -0.00294 -0.00623
3 2013-02-28  0.0127 -0.0130  0.0162  -0.0231   0.00589
4 2013-03-31  0.0373  0.0130  0.0403  -0.0102   0.000985
5 2013-04-30  0.0190  0.0490  0.00122  0.0121   0.00964
6 2013-05-31  0.0233 -0.0307  0.0420  -0.0495  -0.0202
```

```
asset_returns_dplyr_byhand <- asset_returns_dplyr_byhand %>%
  na.omit()

head(asset_returns_dplyr_byhand)
```

```
# A tibble: 6 x 6
  date            SPY     EFA      IJS      EEM       AGG
  <date>        <dbl>   <dbl>    <dbl>    <dbl>     <dbl>
1 2013-01-31  0.0499  0.0366  0.0521  -0.00294 -0.00623
2 2013-02-28  0.0127 -0.0130  0.0162  -0.0231   0.00589
3 2013-03-31  0.0373  0.0130  0.0403  -0.0102   0.000985
4 2013-04-30  0.0190  0.0490  0.00122  0.0121   0.00964
5 2013-05-31  0.0233 -0.0307  0.0420  -0.0495  -0.0202
6 2013-06-30 -0.0134 -0.0272 -0.00140 -0.0547  -0.0158
```

```
# convert to monthly returns, tidyquant.

asset_returns_tq_builtin <-
  prices %>%
  tk_tbl(preserve_index = T,
         rename_index = "date") %>%
  gather(asset, prices, -date) %>%
  group_by(asset) %>%
  tq_transmute(mutate_fun = periodReturn,
               period = "monthly",
```

```
              type = "log") %>%
  spread(asset, monthly.returns) %>%
  select(date, symbols) %>%
  slice(-1)

head(asset_returns_tq_builtin)
```

```
# A tibble: 6 x 6
  date            SPY     EFA      IJS      EEM       AGG
  <date>        <dbl>   <dbl>    <dbl>    <dbl>     <dbl>
1 2013-01-31  0.0499  0.0366  0.0521  -0.00294 -0.00623
2 2013-02-28  0.0127 -0.0130  0.0162  -0.0231   0.00589
3 2013-03-28  0.0373  0.0130  0.0403  -0.0102   0.000985
4 2013-04-30  0.0190  0.0490  0.00122  0.0121   0.00964
5 2013-05-31  0.0233 -0.0307  0.0420  -0.0495  -0.0202
6 2013-06-28 -0.0134 -0.0272 -0.00140 -0.0547  -0.0158
```

```
# convert to monthly returns, tibbletime.

asset_returns_tbltime <-
  prices %>%
  tk_tbl(preserve_index = T,
         rename_index = "date") %>%
  # this is the tibbletime function
  as_tbl_time(index = date) %>%
  as_period(period = "monthly",
            side = "end") %>%
  gather(asset, returns, - date) %>%
  group_by(asset) %>%
  tq_transmute(mutate_fun = periodReturn,
               type = "log") %>%
  spread(asset, monthly.returns) %>%
  select(date, symbols) %>%
  slice(-1)

head(asset_returns_tbltime)
```

```
# A time tibble: 6 x 6
# Index: date
  date            SPY     EFA      IJS      EEM       AGG
  <date>        <dbl>   <dbl>    <dbl>    <dbl>     <dbl>
1 2013-01-31  0.0499  0.0366  0.0521  -0.00294 -0.00623
2 2013-02-28  0.0127 -0.0130  0.0162  -0.0231   0.00589
3 2013-03-28  0.0373  0.0130  0.0403  -0.0102   0.000985
4 2013-04-30  0.0190  0.0490  0.00122  0.0121   0.00964
```

```
5 2013-05-31  0.0233 -0.0307  0.0420  -0.0495  -0.0202
6 2013-06-28 -0.0134 -0.0272 -0.00140 -0.0547  -0.0158
```

```
asset_returns_long <-
  asset_returns_dplyr_byhand %>%
  gather(asset, returns, - date) %>%
  group_by(asset)

head(asset_returns_long)
```

```
# A tibble: 6 x 3
# Groups:   asset [1]
  date       asset returns
  <date>     <chr>   <dbl>
1 2013-01-31 SPY    0.0499
2 2013-02-28 SPY    0.0127
3 2013-03-31 SPY    0.0373
4 2013-04-30 SPY    0.0190
5 2013-05-31 SPY    0.0233
6 2013-06-30 SPY   -0.0134
```

```
# visualize return data


highchart(type = "stock") %>%
  hc_title(text = "Monthly Log Returns") %>%
  hc_add_series(asset_returns_xts[, symbols[1]],
                name = symbols[1]) %>%
  hc_add_series(asset_returns_xts[, symbols[2]],
                name = symbols[2]) %>%
  hc_add_series(asset_returns_xts[, symbols[3]],
                name = symbols[3]) %>%
  hc_add_series(asset_returns_xts[, symbols[4]],
                name = symbols[4]) %>%
  hc_add_series(asset_returns_xts[, symbols[5]],
                name = symbols[5]) %>%
  hc_add_theme(hc_theme_flat()) %>%
  hc_navigator(enabled = F) %>%
  hc_scrollbar(enabled = F) %>%
  hc_exporting(enabled = T) %>%
  hc_legend(enabled = T)
```

```
PhantomJS not found. You can install it with webshot::install_phantomjs(). If it is installed,
```

```
hc_hist_fun <- function(n = 1, object, color ) {
  hc_hist <- hist(object[, symbols[n]],
                  breaks = 50,
```

```
                  plot = F)

  hchart(hc_hist, color = color) %>%
    hc_title(text =
               paste(symbols[n],
                     "Log Returns Distribution",
                     sep = " ")) %>%
    hc_add_theme(hc_theme_flat()) %>%
    hc_exporting(enabled = F) %>%
    hc_legend(enabled = F)
}

hc_hist_fun(1, asset_returns_xts, "cornflowerblue")
```

```
hc_hist_fun(2, asset_returns_xts, "green")
```

```
hc_hist_fun(3, asset_returns_xts, "pink")
```

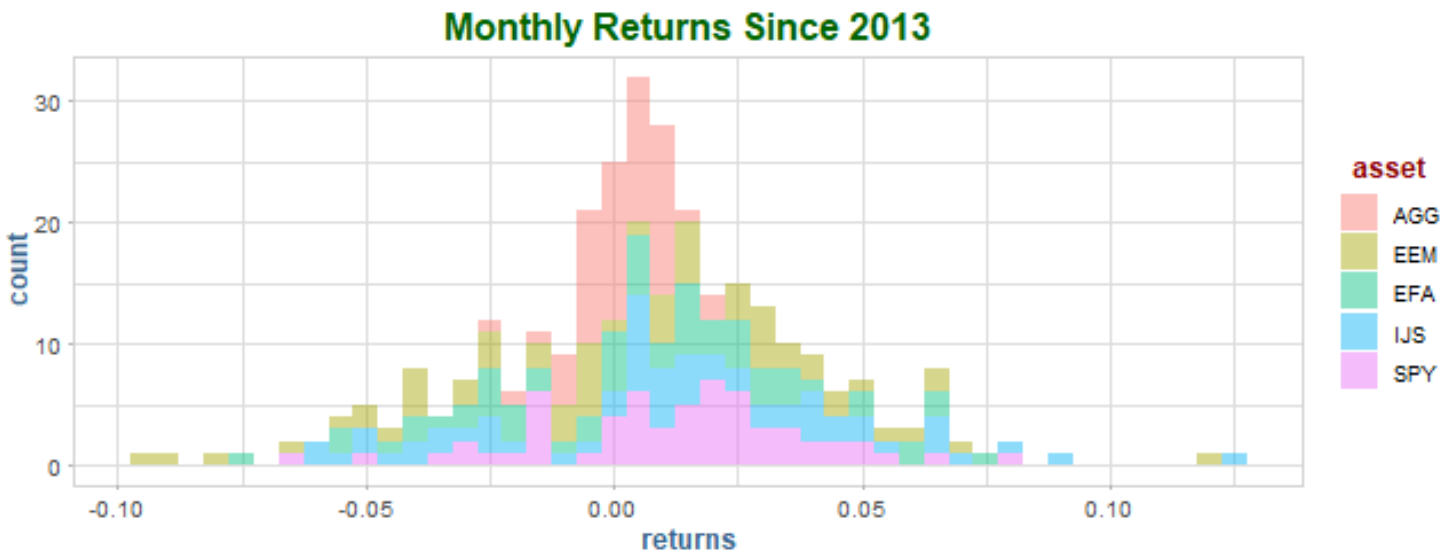
```
hc_hist_fun(4, asset_returns_xts, "purple")
```

```
hc_hist_fun(5, asset_returns_xts, "yellow")
```

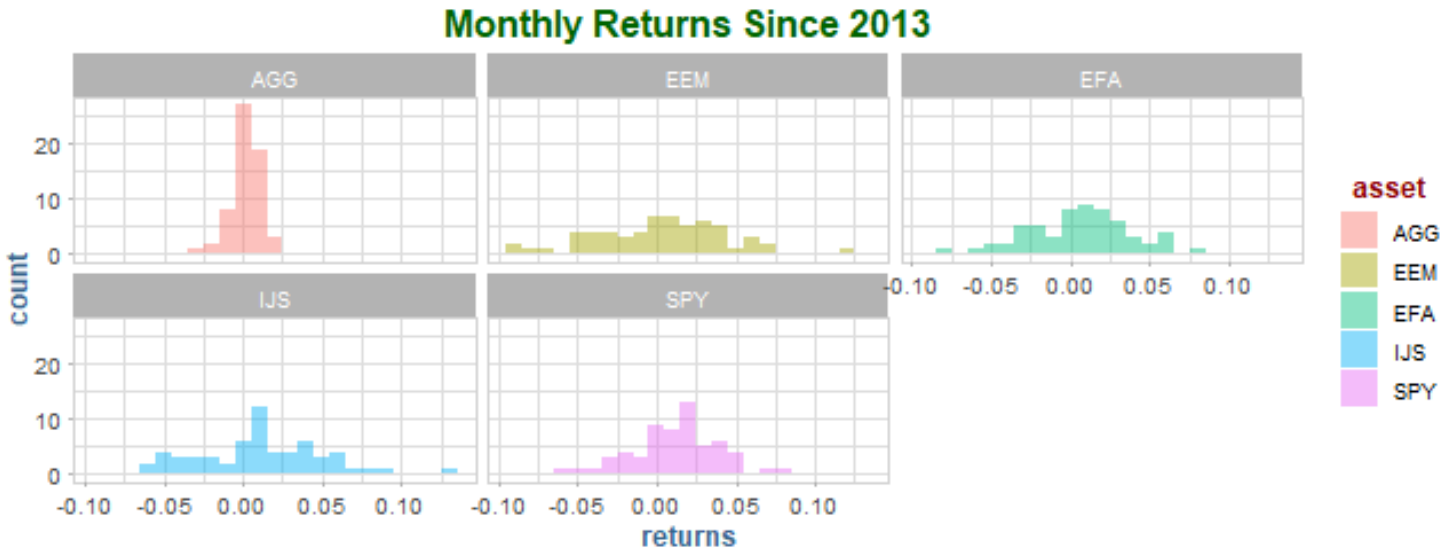
```
map(1:5, hc_hist_fun, asset_returns_xts, "blue")
```

```
[[1]]

[[2]]

[[3]]

[[4]]

[[5]]
```

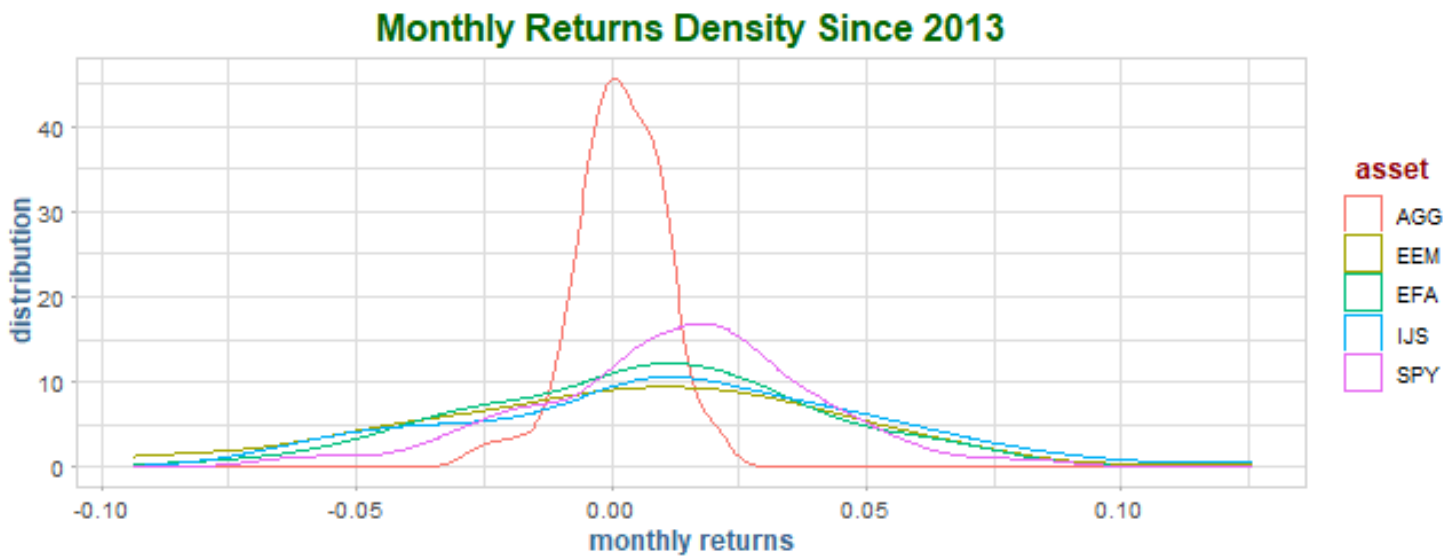
```
asset_returns_long %>%
  ggplot(aes(x = returns, fill = asset)) +
  geom_histogram(alpha = 0.45, binwidth = 0.005) +
  ggtitle("Monthly Returns Since 2013") +
  theme_update(plot.title = element_text(hjust = 0.5))
```



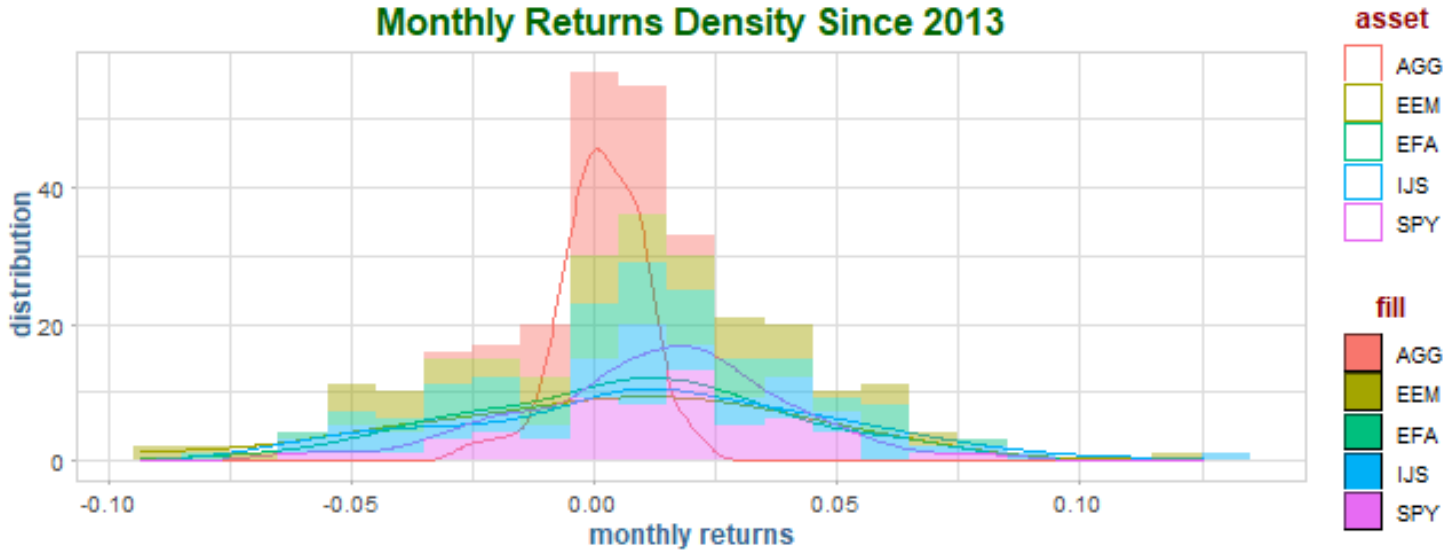

```
asset_returns_long %>%
  ggplot(aes(x = returns, fill = asset)) +
  geom_histogram(alpha = 0.45, binwidth = 0.01) +
  facet_wrap(~asset) +
  ggtitle("Monthly Returns Since 2013") +
  theme_update(plot.title = element_text(hjust = 0.5))
```




```
asset_returns_long %>%
  ggplot(aes(x = returns, colour = asset)) +
  geom_density(alpha = 1) +
  ggtitle("Monthly Returns Density Since 2013") +
  xlab("monthly returns") +
  ylab("distribution") +
  theme_update(plot.title = element_text(hjust = 0.5))
```




```
asset_returns_long %>%
  ggplot(aes(x = returns)) +
  geom_density(aes(color = asset), alpha = 1) +
  geom_histogram(aes(fill = asset), alpha = 0.45, binwidth = 0.01) +
  ggtitle("Monthly Returns Density Since 2013") +
  xlab("monthly returns") +
  ylab("distribution") +
  theme_update(plot.title = element_text(hjust = 0.5))
```




```
asset_returns_long %>%
  ggplot(aes(x = returns)) +
  geom_density(aes(color = asset), alpha = 1) +
```

```
  geom_histogram(aes(fill = asset), alpha = 0.45, binwidth = 0.01) +
  facet_wrap(~asset) +
  ggtitle("Monthly Returns Density Since 2013") +
  xlab("monthly returns") +
  ylab("distribution") +
  theme_update(plot.title = element_text(hjust = 0.5))
```