Building Stalk Portfolios with R

Robert Hickman

2020-05-17

```
library(tidyverse)
library(MASS)

lapply(c("filter", "select"), conflicted::conflict_prefer, "dplyr")
```

```
## [[1]]
## NULL
##
## [[2]]
## NULL
```

```
source("../../static/files/turnips/turnip_funs.R")
```

```
#the four patterns
states <- c("fluctuating", "large_spike", "decreasing", "small_spike")

#build the transition matrix between the states
transition_matrix <- matrix(
  c(
    0.2, 0.3, 0.15, 0.35,
    0.5, 0.05, 0.2, 0.25,
    0.25, 0.45, 0.05, 0.25,
    0.45, 0.25, 0.15, 0.15
  ),
  nrow = 4, byrow = TRUE)
#name the current (rows) and next (cols) states
rownames(transition_matrix) <- states
colnames(transition_matrix) <- states

transition_matrix
```

```
##             fluctuating large_spike decreasing small_spike
## fluctuating        0.20        0.30       0.15        0.35
## large_spike        0.50        0.05       0.20        0.25
## decreasing         0.25        0.45       0.05        0.25
## small_spike        0.45        0.25       0.15        0.15
```

```
left_eigen <- ginv(eigen(transition_matrix)$vectors)[1,]
pattern_likelihood_analytic <- left_eigen / sum(left_eigen)
names(pattern_likelihood_analytic) <- states

pattern_likelihood_analytic
```

```
## fluctuating large_spike  decreasing small_spike
```

```
##    0.3462773    0.2473628    0.1476074    0.2587525
```

Of course, we can also do this using Hamiltonian Monte Carlo methods but just simulating a few sets of independent weeks

```
#transition probabilities
transition_df <- as.data.frame(transition_matrix) %>%
  rownames_to_column(var = "current_state") %>%
  pivot_longer(cols = states, names_to = "next_state", values_to = "prob") %>%
  group_by(current_state) %>%
  mutate(cum_prob = cumsum(prob)) %>%
  ungroup()

#get the next pattern from the current pattern
find_next_pattern <- function(pattern, rng, transitions = transition_df) {
  next_transition <- transitions %>%
    #find possible patterns
    dplyr::filter(current_state == pattern & cum_prob > rng) %>%
    #take top row
    .[1,]
  #next state is that pattern
  next_state <- next_transition$next_state
}

#run forward for prop_forward weeks for each run to check convergence
transition_patterns <- function(initial_pattern, prop_forward) {
  patterns <- c()
  pattern <- initial_pattern
  #run n times
  for(runs in seq(prop_forward)) {
    pattern <- find_next_pattern(pattern, runif(1))
    patterns <- append(patterns, pattern)
  }
  #return as df
  df <- data.frame(
    initial_pattern,
    pattern = as.character(patterns),
    t = 1:prop_forward
  )
  return(df)
}

#repeat sims n times
simulation_reps <- 100
#how many weeks to run each sim for
prop_forward = 10
#run the sims
pattern_likelihood <- states %>%
  rep(simulation_reps) %>%
  map_df(., transition_patterns, prop_forward) %>%
  group_by(pattern) %>%
  summarise(prob = n() / (simulation_reps *  prop_forward * length(states)))

pattern_likelihood
```

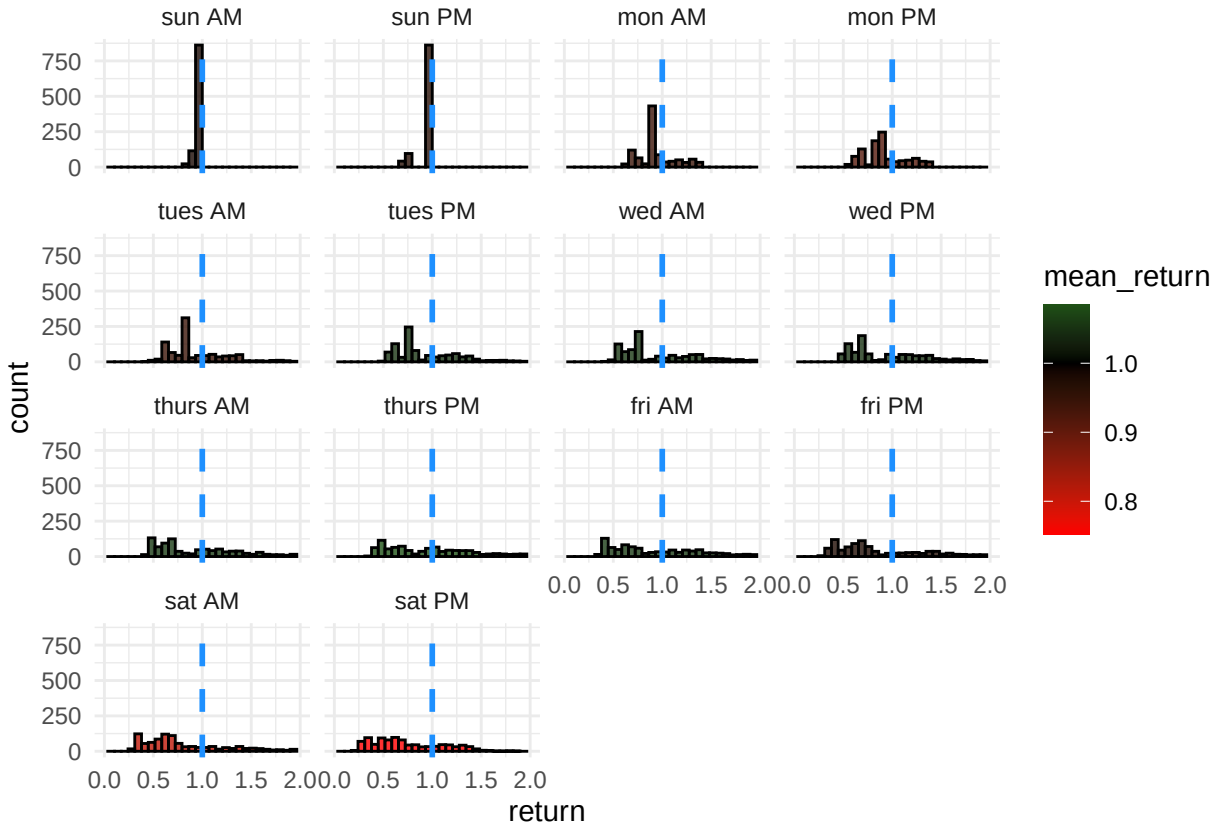
```
## # A tibble: 4 x 2
##   pattern      prob
##   <chr>       <dbl>
## 1 decreasing  0.145
## 2 fluctuating 0.352
## 3 large_spike 0.238
## 4 small_spike 0.265
```

```
p1 <- pattern_likelihood_analytic %>%
  as.data.frame() %>%
  rownames_to_column("pattern") %>%
  left_join(pattern_likelihood, by = "pattern") %>%
  rename(hmc = "prob", analytic = ".") %>%
  pivot_longer(c("hmc", "analytic"), names_to = "calc", values_to = "prob") %>%
  ggplot(aes(x = pattern, y = prob, group = calc)) +
  geom_bar(stat = "identity", position = "dodge", aes(fill = calc), colour = "black") +
  scale_fill_manual(values = c("dodgerblue", "orange")) +
  labs(
    title = "",
    subtitle = "",
    x = "week's prices pattern",
    y = "probability"
  ) +
  theme_minimal()
```

```
#simulate a week of prices given a pattern
simulate_week <- function(pattern, epochs) {
  #set up prices vector
  sunday_price <- sample(90:110, 1)
  initial_prices <- c(rep(sunday_price, 2), rep(0, 12))

  #simulate pattern
  if(pattern == "decreasing") {
    week_prices <- sim_decreasing(prices = initial_prices)
  } else if(pattern == "fluctuating") {
    week_prices <- sim_fluctuating(prices = initial_prices, first_epochs = c(sample(0:6, 1), sample(2:3
  } else if(pattern == "large_spike") {
    week_prices <- sim_largespike(prices = initial_prices, rate = runif(1, 0.85, 0.95), first_peak = sa
  } else if(pattern == "small_spike") {
    week_prices <- sim_smallspike(prices = initial_prices, first_peak = sample(1:8, 1))
  }

  #arrange df
  weekly_prices <- data.frame(
    day = epochs,
    buy_price = sunday_price,
    price = week_prices
  )

  return(weekly_prices)
}
```

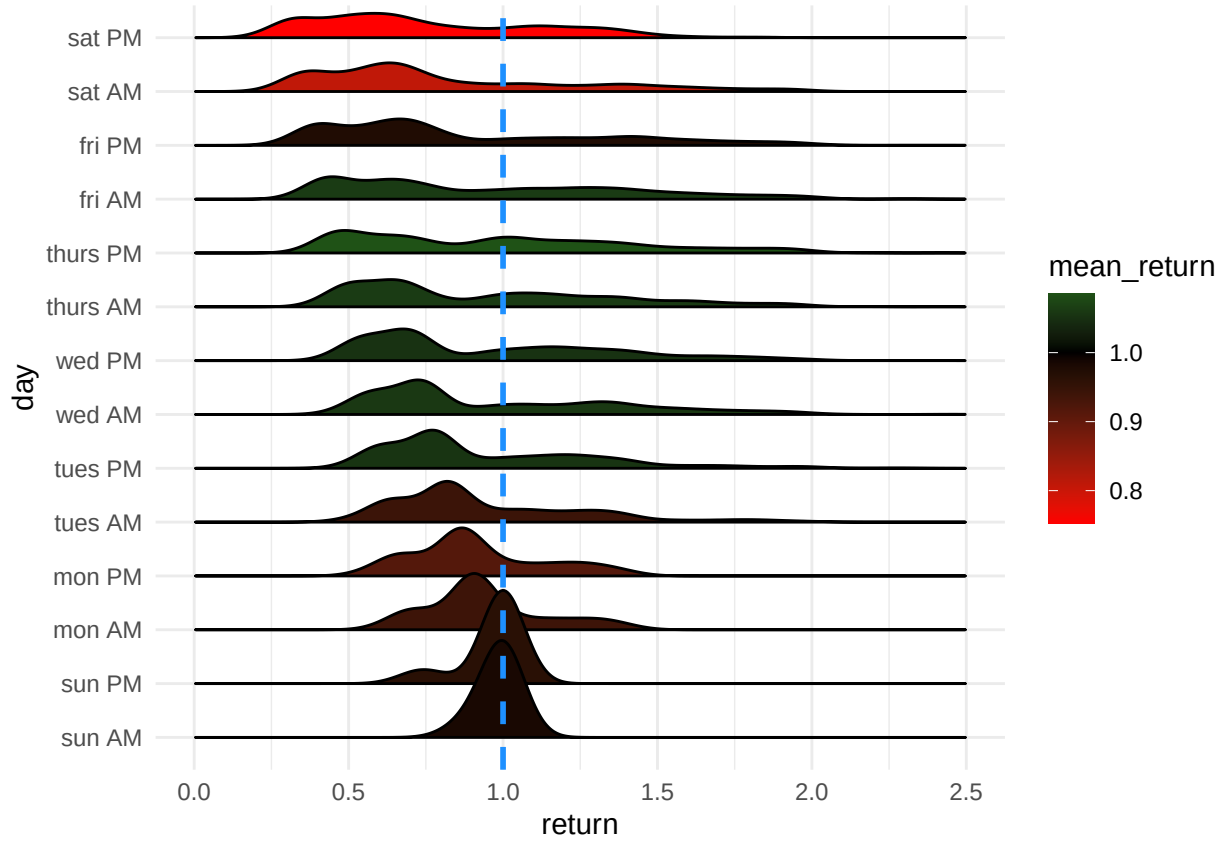
```
week <- c("sun", "mon", "tues", "wed", "thurs", "fri", "sat")
epochs <- paste(rep(week, each = 2), rep(c("AM", "PM"), 7))
```

```
simulation_reps <- 1000
prices <- pattern_likelihood %>%
  #sample patterns by likelihood
  dplyr::sample_n(simulation_reps, weight = prob, replace = TRUE) %>%
  .$pattern %>%
  map_df(., simulate_week, epochs) %>%
  mutate(return = price / buy_price,
         day = factor(day, levels = epochs)) %>%
  dplyr::filter(!grepl("sunday", day)) %>%
  group_by(day) %>%
  mutate(mean_return = mean(return)) %>%
  ungroup()

p2 <- ggplot(prices, aes(x = return, fill = mean_return)) +
  geom_histogram(alpha = 0.8, colour = "black") +
  geom_vline(xintercept = 1, linetype = "dashed", colour = "dodgerblue", size = 1) +
  scale_fill_gradient2(low = "red", high = "green", mid = "black", midpoint = 1) +
  scale_x_continuous(limits = c(0, 2)) +
  theme_minimal() +
  facet_wrap(~day)

p2
```




```
library(ggridges)

p3 <- ggplot(prices, aes(x = return, y = day, fill = mean_return)) +
```

```
  geom_density_ridges2() +
  geom_vline(xintercept = 1, linetype = "dashed", colour = "dodgerblue", size = 1) +
  scale_fill_gradient2(low = "red", high = "green", mid = "black", midpoint = 1) +
  scale_x_continuous(limits = c(0, 2.5)) +
  theme_minimal()

p3
```




```
monthly_interest <- 1.005

interest_df <- data.frame(day = factor(epochs, levels = epochs)) %>%
  mutate(interest_days = rep(0:6, each = 2)) %>%
  mutate(interest_gained = (1 * (monthly_interest ^ (1/30)) ^ interest_days) -1)

interest_df
```

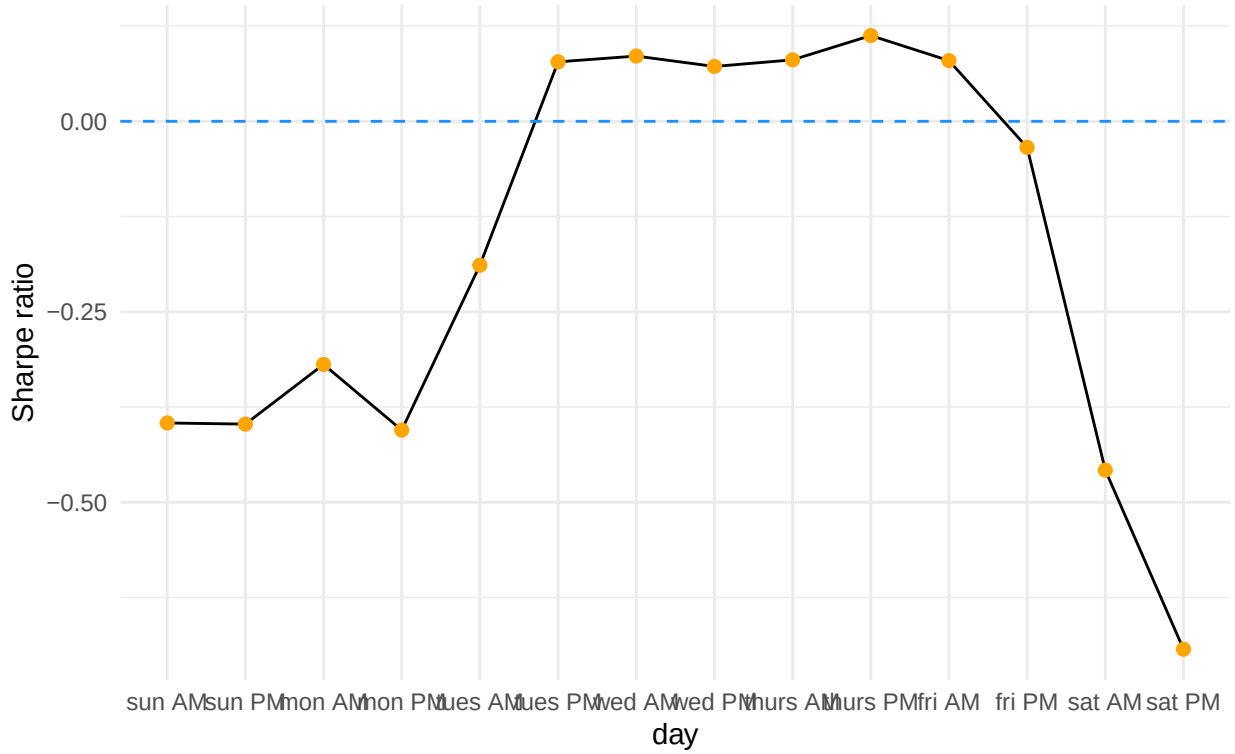
```
##          day interest_days interest_gained
## 1     sun AM             0    0.0000000000
## 2     sun PM             0    0.0000000000
## 3     mon AM             1    0.0001662652
## 4     mon PM             1    0.0001662652
## 5    tues AM             2    0.0003325581
## 6    tues PM             2    0.0003325581
## 7     wed AM             3    0.0004988785
## 8     wed PM             3    0.0004988785
## 9   thurs AM             4    0.0006652267
```

```
## 10 thurs PM             4    0.0006652267
## 11   fri AM             5    0.0008316025
## 12   fri PM             5    0.0008316025
## 13   sat AM             6    0.0009980060
## 14   sat PM             6    0.0009980060
```

```
mean_returns <- prices %>%
  left_join(interest_df, by = "day") %>%
  mutate(excess_return = (return - 1) - interest_gained) %>%
  group_by(day) %>%
  summarise(mean_excess = mean(excess_return),
            sd_excess = sd(excess_return)) %>%
  mutate(sharpe_ratio = mean_excess / sd_excess)

dplyr::select(mean_returns, day, sharpe_ratio)
```

```
## # A tibble: 14 x 2
##    day      sharpe_ratio
##    <fct>           <dbl>
##  1 sun AM        -0.396
##  2 sun PM        -0.397
##  3 mon AM        -0.319
##  4 mon PM        -0.405
##  5 tues AM       -0.189
##  6 tues PM        0.0780
##  7 wed AM         0.0856
##  8 wed PM         0.0720
##  9 thurs AM       0.0806
## 10 thurs PM       0.113
## 11 fri AM         0.0796
## 12 fri PM        -0.0342
## 13 sat AM        -0.458
## 14 sat PM        -0.693
```

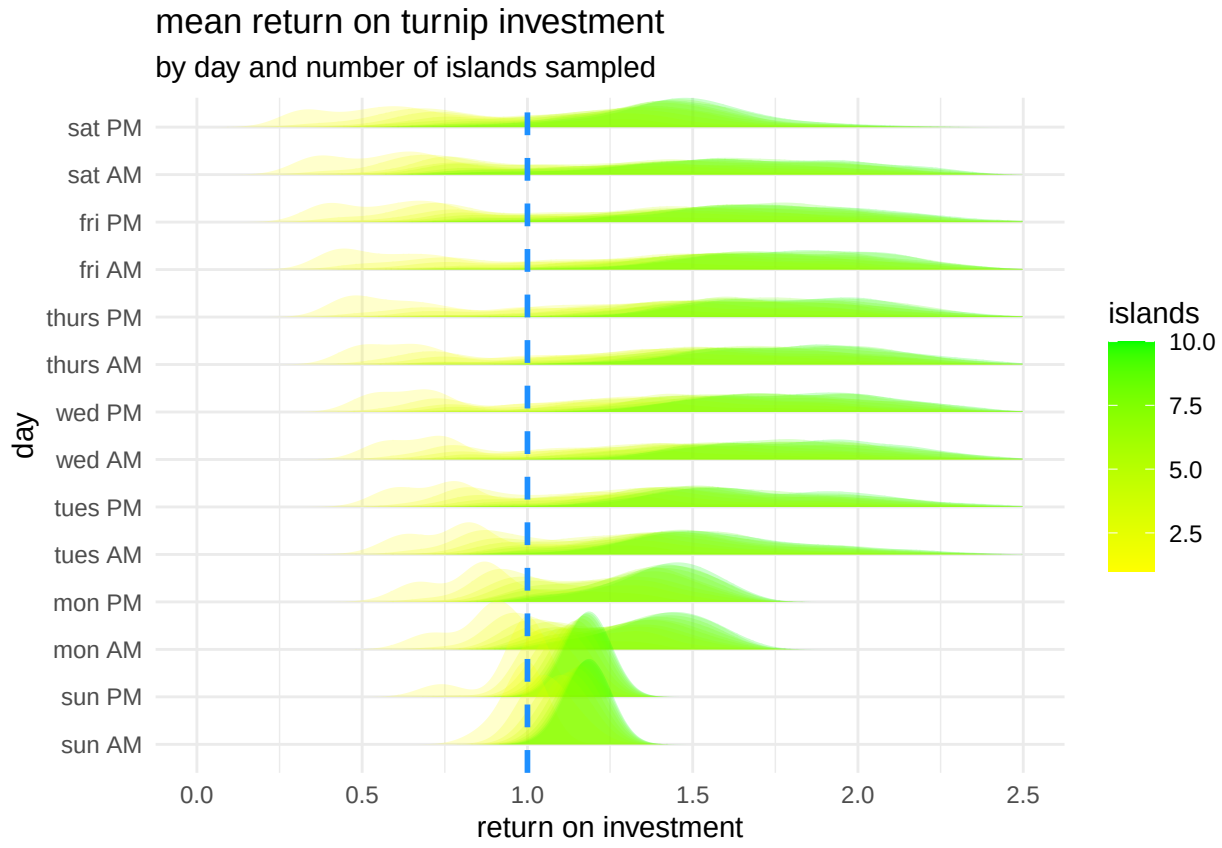
```
p4 <- ggplot(mean_returns, aes(x = day, y = sharpe_ratio, group = 1)) +
  geom_line() +
  geom_point(size = 2, colour = "orange") +
  geom_hline(yintercept = 0, linetype = "dashed", colour = "dodgerblue") +
  labs(
    title = "",
    subtitle = "",
    x = "day",
    y = "Sharpe ratio"
  ) +
  theme_minimal()

p4
```




```
#do the same for n islands
simulate_open_economies <- function(islands, pattern_likelihood, epochs) {
  patterns <- pattern_likelihood %>%
    dplyr::sample_n(islands, weight = prob, replace = TRUE) %>%
    .$pattern

  #munge
  all_prices <- map_df(patterns, simulate_week, epochs) %>%
    #will always buy at lowest and sell at highest
    mutate(buy_price = min(buy_price)) %>%
    group_by(day) %>%
    mutate(sell_price = max(price)) %>%
    dplyr::select(day, buy_price, sell_price) %>%
    unique() %>%
    mutate(islands)
}

#run on 1:n islands
n_islands <- 10
open_prices <- rep(1:n_islands, simulation_reps) %>%
  map_df(simulate_open_economies, pattern_likelihood, epochs) %>%
  group_by(islands) %>%
  mutate(return = sell_price / buy_price,
         day = factor(day, levels = epochs)) %>%
  dplyr::filter(!grepl("sunday", day)) %>%
  group_by(islands, day) %>%
```

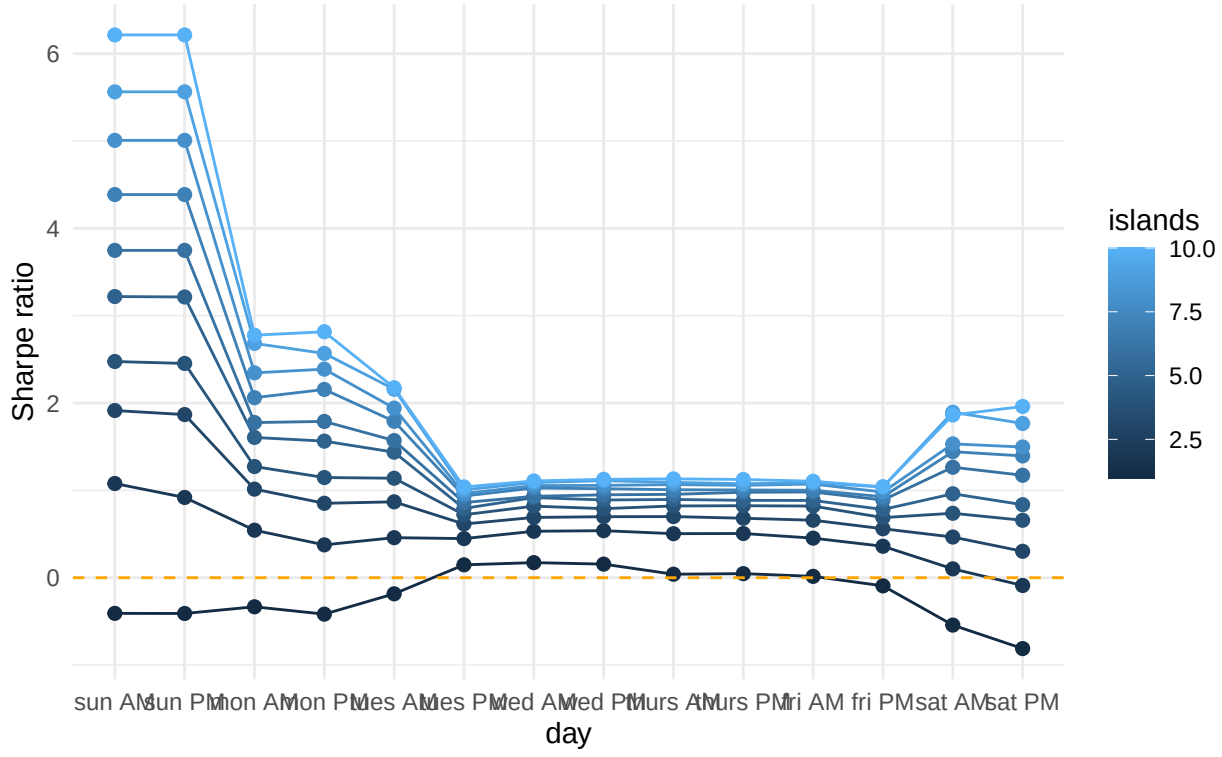
```
  mutate(mean_return = mean(return)) %>%
  ungroup()
```

```
p5 <- ggplot(open_prices, aes(x = return, y = day, group = paste(day, islands), fill = islands)) +
  geom_density_ridges2(alpha = 0.2, colour = NA) +
  geom_vline(xintercept = 1, linetype = "dashed", colour = "dodgerblue", size = 1) +
  scale_fill_gradient(low = "yellow", high = "green") +
  scale_x_continuous(limits = c(0, 2.5)) +
  labs(
    title = "mean return on turnip investment",
    subtitle = "by day and number of islands sampled",
    x = "return on investment",
    y = "day"
  ) +
  theme_minimal()

p5
```



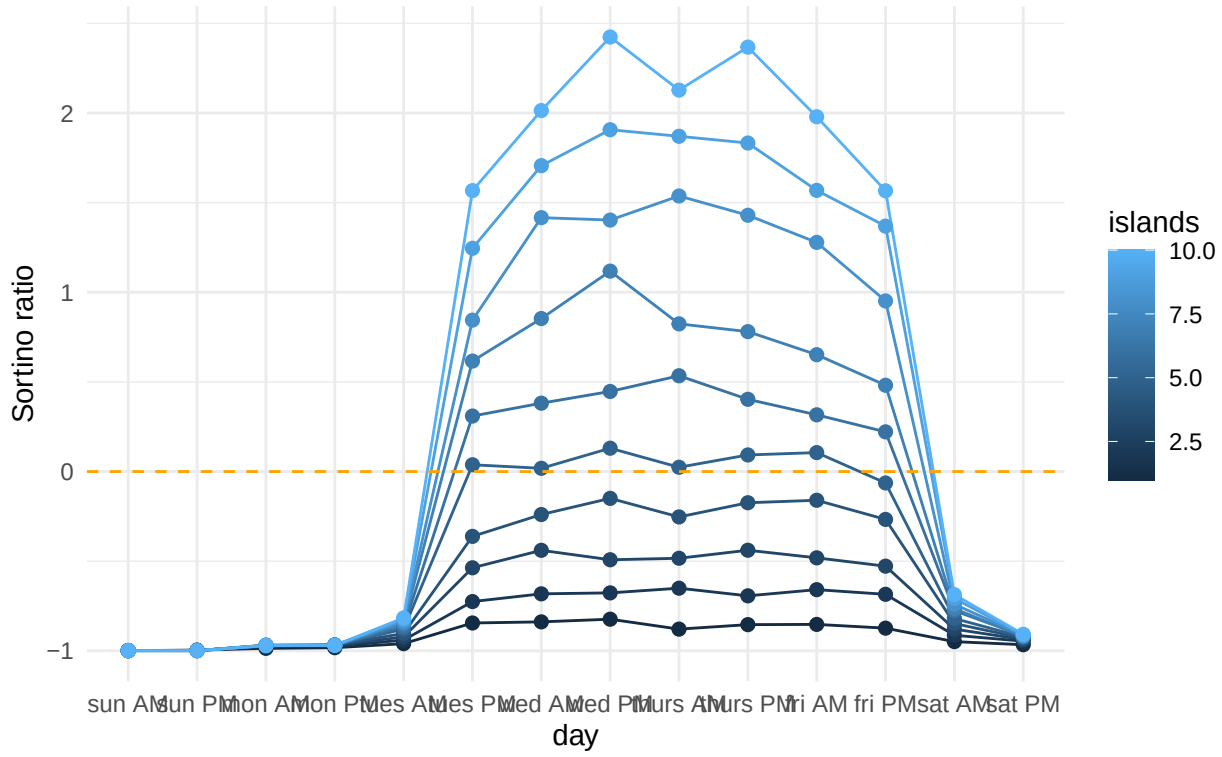

```
monthly_interest <- 1

interest_df <- data.frame(day = factor(epochs, levels = epochs)) %>%
  mutate(interest_days = rep(0:6, each = 2)) %>%
  mutate(intrest_gained = (1 * (monthly_interest ^ (1/30)) ^ interest_days) -1)

mean_returns <- open_prices %>%
  left_join(interest_df, by = "day") %>%
```

```
  mutate(excess_return = (return - 1) - intrest_gained) %>%
  group_by(islands, day) %>%
  summarise(mean_excess = mean(excess_return),
            sd_excess = sd(excess_return)) %>%
  mutate(sharpe_ratio = mean_excess / sd_excess)

p6 <- ggplot(mean_returns, aes(x = day, y = sharpe_ratio, group = islands, colour = islands)) +
  geom_line() +
  geom_point(size = 2) +
  geom_hline(yintercept = 0, linetype = "dashed", colour = "orange") +
  labs(
    title = "",
    subtitle = "",
    x = "day",
    y = "Sharpe ratio"
  ) +
  theme_minimal()

p6
```




```
MAR <- 1

sortino_ratio <- open_prices %>%
  group_by(day, islands) %>%
  mutate(excess_return = return - 1) %>%
  summarise(
```

```
    mean_excess = mean(excess_return - MAR),
    downside = sum((MAR - excess_return[excess_return < MAR])^2/n())
  ) %>%
  mutate(sortino_ratio = mean_excess / sqrt(downside))

p7 <- ggplot(sortino_ratio, aes(x = day, y = sortino_ratio, group = islands, colour = islands)) +
  geom_line() +
  geom_point(size = 2) +
  geom_hline(yintercept = 0, linetype = "dashed", colour = "orange") +
  labs(
    title = "",
    subtitle = "",
    x = "day",
    y = "Sortino ratio"
  ) +
  theme_minimal()

p7
```




```
MARs <- c(-0.1, 0, 0.1, 0.5, 1, 2, 4, 7)

mulptiple_sortinos <- map_dfr(seq(length(MARs)), ~open_prices) %>%
  mutate(MAR = rep(MARs, each = nrow(open_prices))) %>%
  select(day, islands, return, MAR) %>%
  group_by(day, islands, MAR) %>%
  mutate(excess_return = return - 1) %>%
```

```
  summarise(
    mean_excess = mean(excess_return - MAR),
    downside = sum((MAR - excess_return[excess_return < MAR])^2/n())
  ) %>%
  mutate(sortino_ratio = mean_excess / sqrt(downside))

p8 <- ggplot(mulptiple_sortinos, aes(x = day, y = sortino_ratio, group = islands, colour = islands)) +
  geom_line() +
  geom_point(size = 2) +
  geom_hline(yintercept = 0, linetype = "dashed", colour = "orange") +
  labs(
    title = "",
    subtitle = "",
    x = "day",
    y = "Sortino ratio"
  ) +
  theme_minimal() +
  facet_wrap(~MAR, scales = "free")

p8
```