RSDexport

Yichen Han

2024-08-10

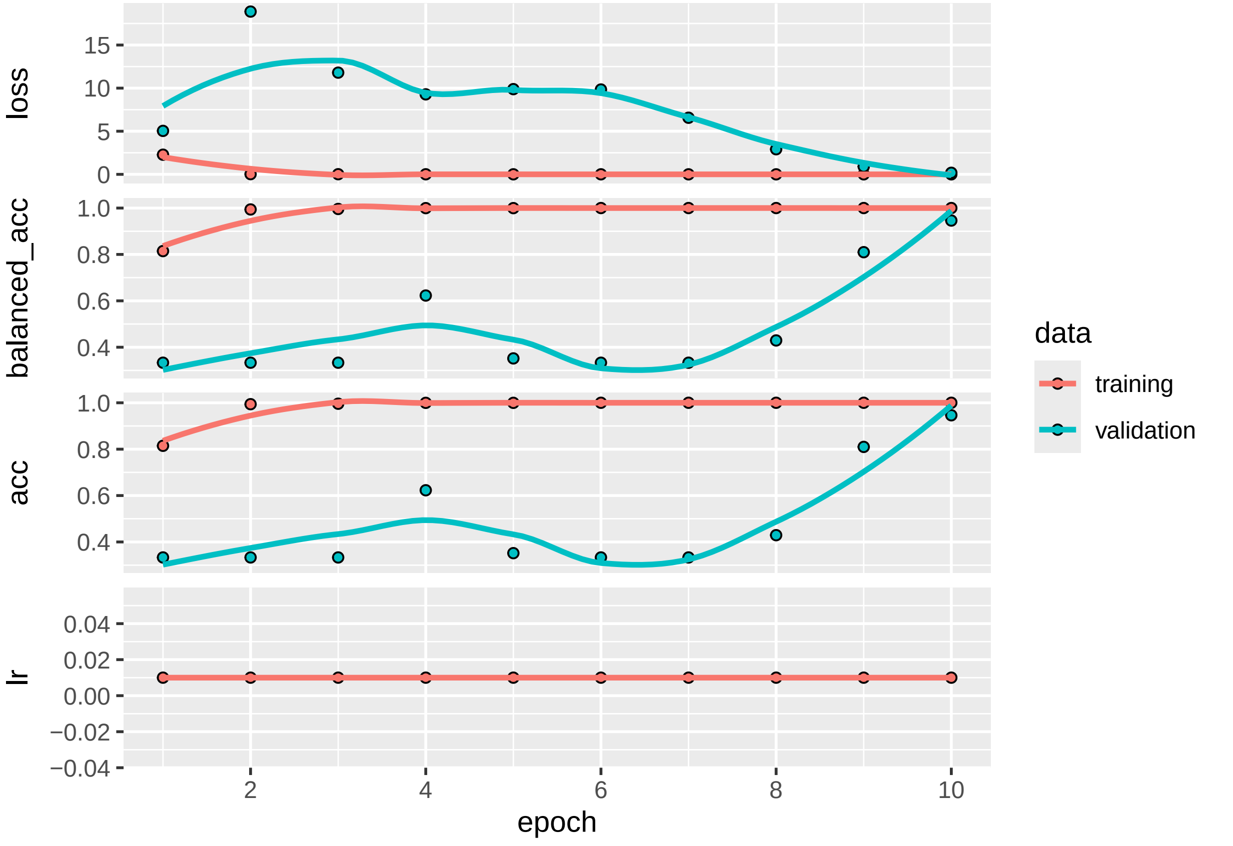
Experiment 39

Spec.region excluded for “abnormal”. Functional change not guaranteed.

Call: `GenePermutation(triplets, keyed, num.perm=15000, min.subs=10, max.subs=30, spec.region=30:60)`

Model: `maxlen = 477, batch_size = 64, steps_per_epoch = 45, epochs = 10, step = c(1, 1, 1)`

```r
# load outputs/hist39.rds
hist39 <- readRDS(file="outputs/hist39.rds")
plot(hist39)
```



```r
checkpoint_path <- file.path("checkpoints")
dir_path <- file.path("outputs")
```

```
run_name <- "rsd-permutation_39"
model <- load_cp(paste0(checkpoint_path, "/", run_name),
                 cp_filter = "last_ep")
```

```
## Using checkpoint checkpoints/rsd-permutation_39/Ep.010-val_loss0.18-val_acc0.946.hdf5
```

Model at last Epoch

```
path_special_test <- file.path("special/test")
path_normal_test <- file.path("normal/test")
path_abnormal_test <- file.path("abnormal/test")
eval_model <- evaluate_model(path_input = c(path_normal_test,
  path_abnormal_test, path_special_test),
  model = model,
  batch_size = 64,
  step = 5,
  vocabulary_label = list(c("normal", "abnormal", "special")),
  number_batches = 10,
  mode = "label_folder",
  verbose = FALSE
)

eval_model
```

```
## [[1]]
## [[1]]$confusion_matrix
##             Truth
## Prediction normal abnormal special
##   normal      254       26       0
##   abnormal      0      213       0
##   special       2       17     256
##
## [[1]]$accuracy
## [1] 0.9414062
##
## [[1]]$categorical_crossentropy_loss
## [1] 0.1702538
##
## [[1]]$AUC
## NULL
##
## [[1]]$AUPRC
## NULL
```

```
special_seq <- permute_sequence(triplets, type="ok", min.subs=5, max.subs=30,
                                dict=codon.dict, spec.cond=FALSE, spec.region=NULL)
special_seq <- permute_sequence(special_seq, type="func", min.subs=5, max.subs=30,
                                dict=codon.dict, spec.cond=TRUE, spec.region=30:60)
special_seq <- paste(special_seq, collapse = "")
onehot_instance <-  seq_encoding_label(char_sequence = special_seq,
                                          maxlen = 477,
                                          start_ind = 1,
                                          vocabulary = c("A", "C", "G", "T"))
head(onehot_instance[1,,])
```

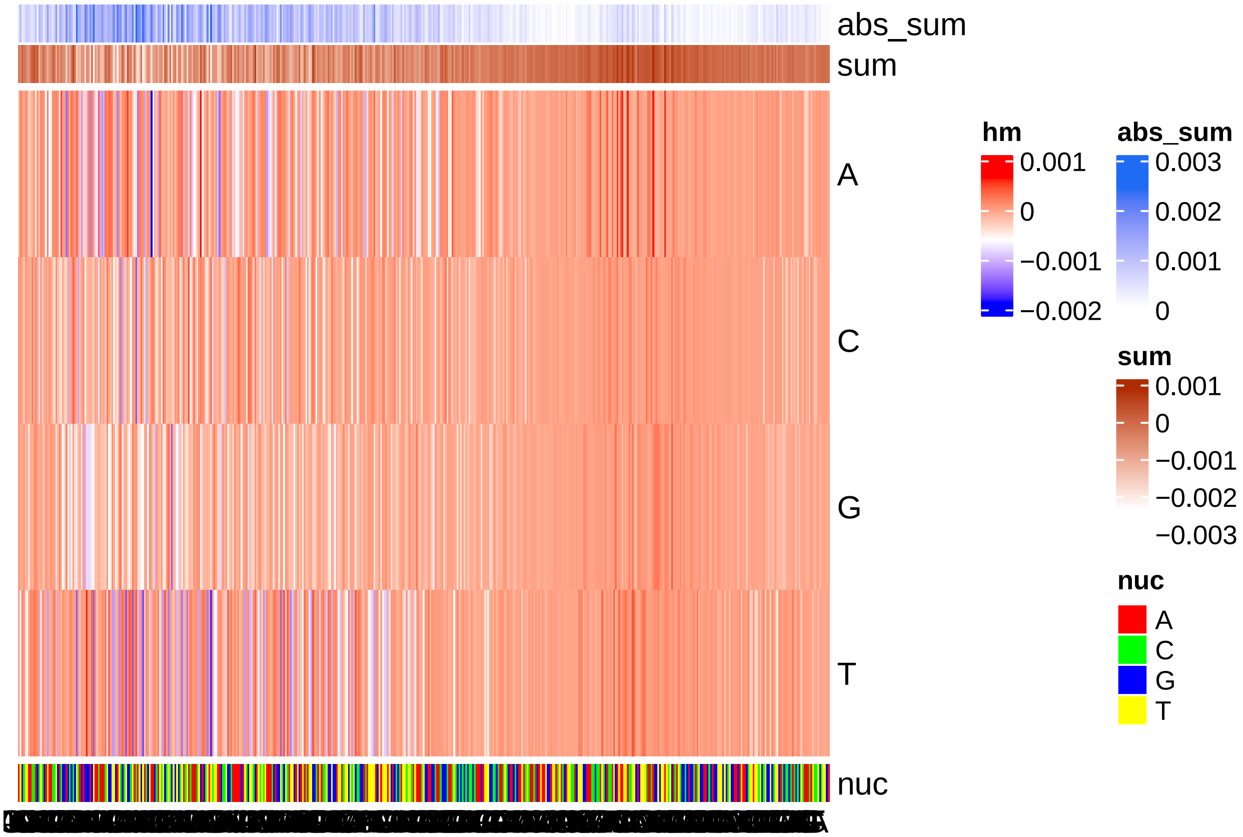
```
##      [,1] [,2] [,3] [,4]
## [1,]    1    0    0    0
## [2,]    0    0    0    1
## [3,]    0    0    1    0
## [4,]    0    1    0    0
## [5,]    0    0    0    1
## [6,]    0    0    0    1
```

```
pred <- predict(model, onehot_instance, verbose = 0)
pred
```

```
##              [,1]         [,2] [,3]
## [1,] 2.989441e-09 3.559504e-18    1
```

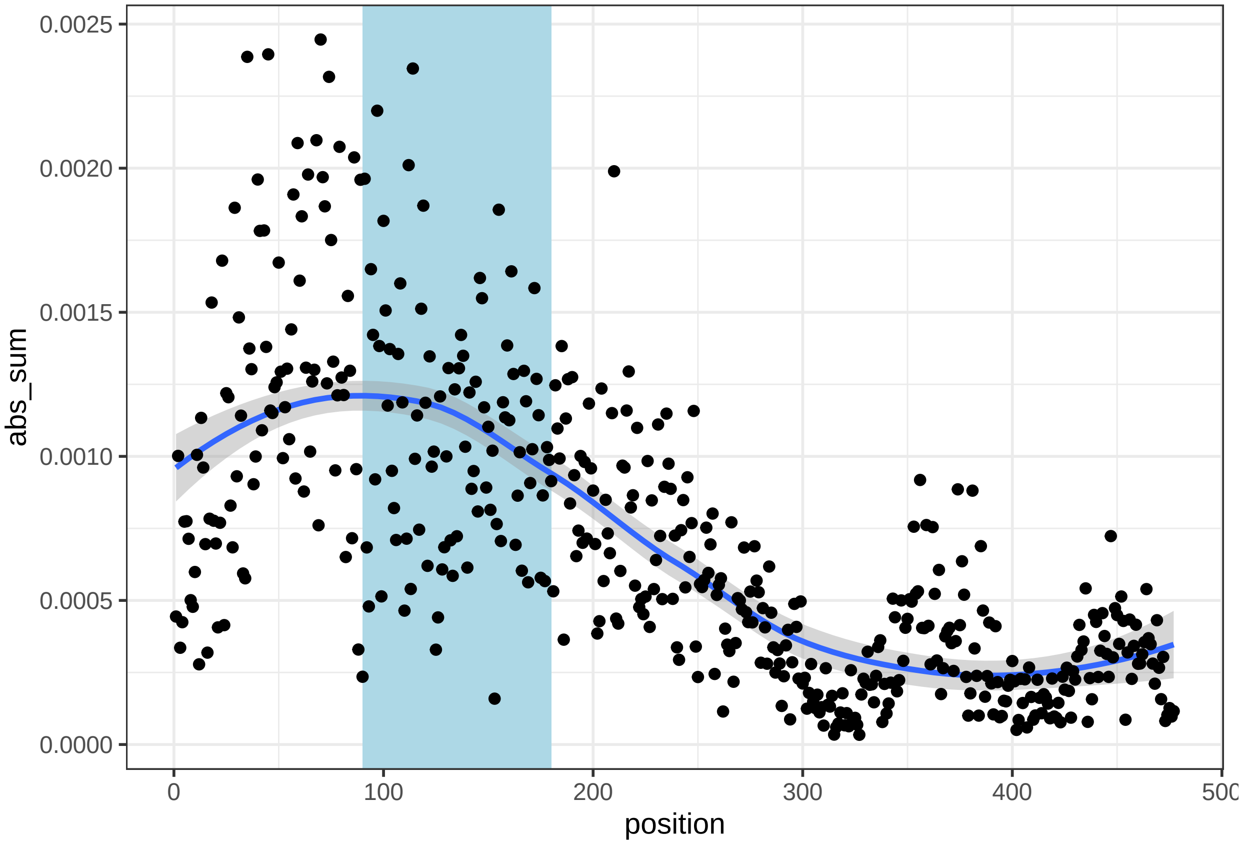
```
ig <- integrated_gradients(
  input_seq = onehot_instance,
  baseline_type = "shuffle",
  target_class_idx = 2,
  model = model,
  num_baseline_repeats = 50)

heatmaps_integrated_grad(integrated_grads = ig,
                         input_seq = onehot_instance)
```

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



```
abs_sum <- rowSums(abs(as.array(ig)))
df <- data.frame(abs_sum = abs_sum, position = 1 : 477)
ggplot(df, aes(x = position, y = abs_sum)) + geom_rect(aes(xmin = 90, xmax = 180, ymin = -Inf, ymax = I
```

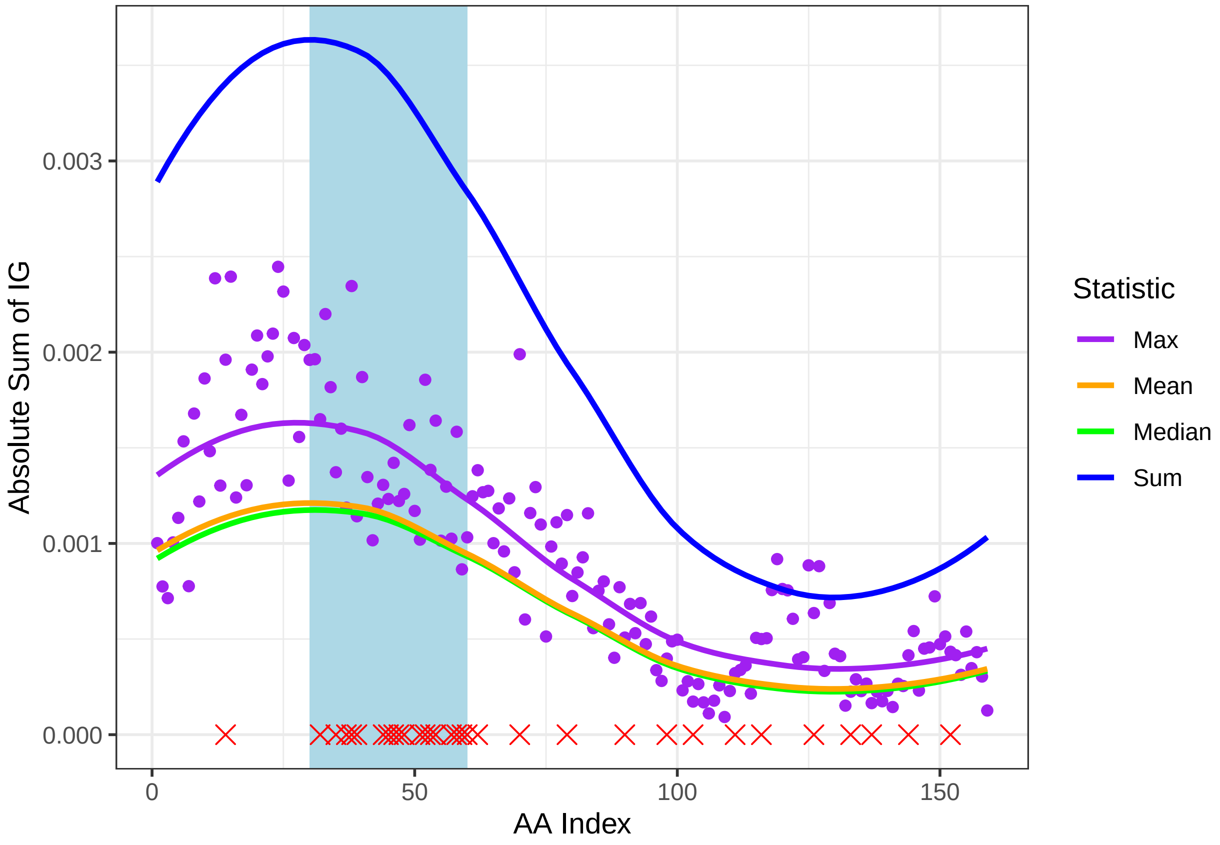
```
## ‘geom_smooth()‘ using method = ’loess’ and formula = ’y ~ x’
```



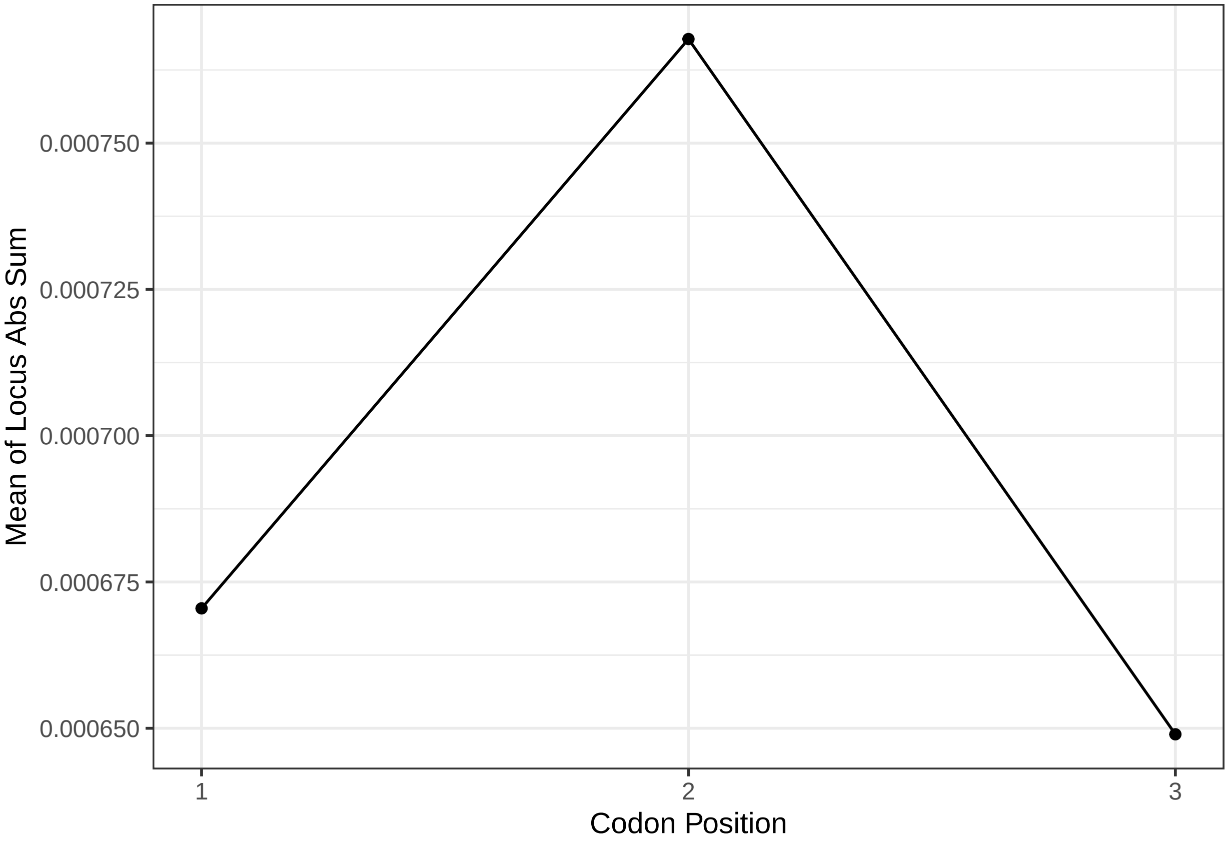
```
# abs_sum <- rowSums(abs(as.array(ig)))
#df <- data.frame(abs_sum = abs_sum, position = 1 : 477)
#ggplot(df, aes(x = position, y = abs_sum)) + geom_rect(aes(xmin = 90, xmax = 180, ymin = -Inf, ymax = 

# modification: df should be summed every 3 rows, position reassigned accordingly.
df_mod <- df %>%
  mutate(group = rep(1:(nrow(df) / 3), each = 3)) %>%
  group_by(group) %>%
  summarise(
    abs_sum_sum = sum(abs_sum),
    abs_sum_median = median(abs_sum),
    abs_sum_mean = mean(abs_sum),
    abs_sum_max = max(abs_sum)
  )

# Split special_seq (a string) into triplets (every third character)
chars <- strsplit(special_seq, "")[[1]]

# Create triplets
trip_spec <- sapply(seq(1, length(chars), by = 3), function(i) {
  paste(chars[i:min(i+2, length(chars))], collapse = "")
})

# Get the index of sequences at which trip_spec != triplets
index <- which(trip_spec != triplets)
```

```r
# Plot the results with different smooth lines for sum, median, mean, and max
ggplot(df_mod, aes(x = group)) +
  geom_rect(aes(xmin = 30, xmax = 60, ymin = -Inf, ymax = Inf), fill = "lightblue", alpha = 0.2) +
  # draw points of max
  geom_point(aes(x = group, y = abs_sum_max), color = "purple") +
  geom_smooth(aes(y = abs_sum_sum, color = "Sum"), method = "auto", se = FALSE) +
  geom_smooth(aes(y = abs_sum_median, color = "Median"), method = "auto", se = FALSE) +
  geom_smooth(aes(y = abs_sum_mean, color = "Mean"), method = "auto", se = FALSE) +
  geom_smooth(aes(y = abs_sum_max, color = "Max"), method = "auto", se = FALSE) +
  scale_color_manual(values = c("Sum" = "blue", "Median" = "green", "Mean" = "orange", "Max" = "purple"
  geom_point(data = data.frame(group = index), aes(x = group, y = 0), shape = 4, size = 3, color = "red
  theme_bw() +
  labs(y = "Absolute Sum of IG", color = "Statistic", x = "AA Index")
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

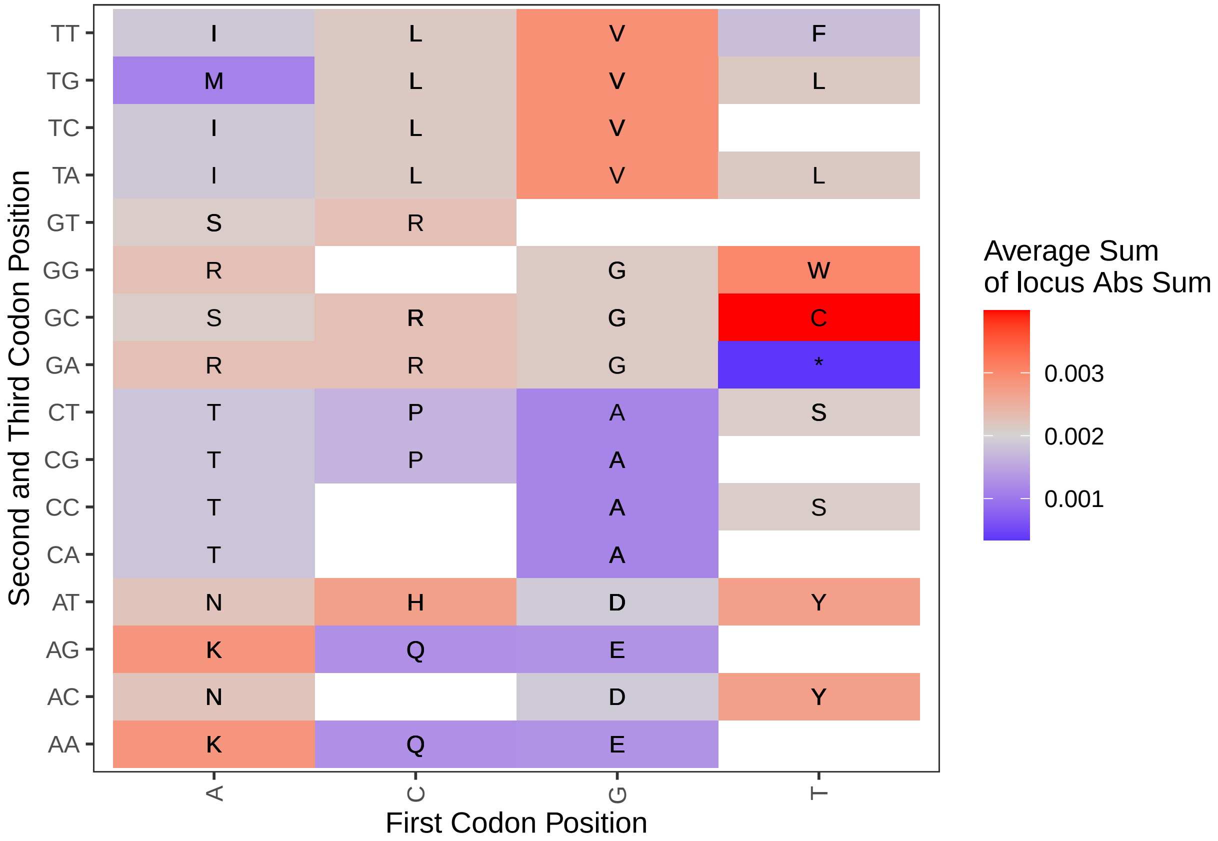


```r
# calculate mean/med of first, second, and third codon positions
mean_codon <- sapply(1:3, function(i) {
  mean(df$abs_sum[seq(i, 477, by = 3)])
})
# plot with line. x:1,2,3, y: mean_codon
ggplot(data = data.frame(x = 1:3, y = mean_codon), aes(x = x, y = y)) +
  geom_line() +
  geom_point() +
  theme_bw() +
  labs(x = "Codon Position", y = "Mean of Locus Abs Sum") +
  # x tick only 1,2,3, integer
  scale_x_continuous(breaks = 1:3, minor_breaks = NULL, labels = c("1", "2", "3"))
```

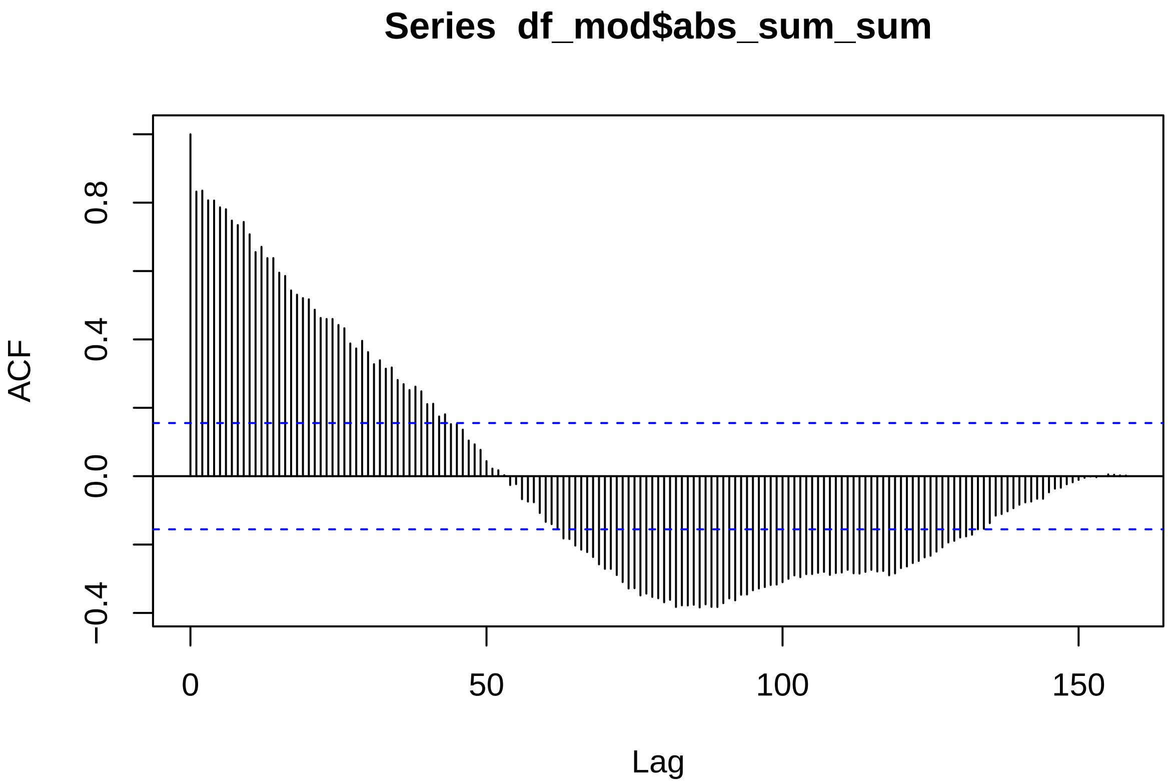


NB: plot generated with considerable randomness and variance.

```
keyed_spec <- triplets_keying(trip_spec)
df_keyed <- cbind(keyed_spec, trip_spec, df_mod)
# mutate two colmuns: cod1 is the first character in trip_spec, cod23 is the 2-3 chars in trip_spec
df_keyed <- df_keyed %>%
  mutate(cod1 = substr(trip_spec, 1, 1),
         cod23 = substr(trip_spec, 2, 3))
df_keyed <- df_keyed %>%
  group_by(keyed_spec) %>%
  mutate(value = mean(abs_sum_sum))

ggplot(df_keyed, aes(x = cod1, y = cod23, fill = value)) +
  geom_tile() +
  geom_text(aes(label = keyed_spec), color = "black", size = 3) +
  scale_fill_gradient2(low = "blue", mid = "lightgrey", high = "red", midpoint = 0.002) +
  theme_bw() +
  labs(fill = "Average Sum\nof locus Abs Sum",
       x = "First Codon Position",
       y = "Second and Third Codon Position") +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1),
        strip.text = element_text(face = "bold"),
        panel.grid.major = element_blank(), panel.grid.minor = element_blank())
```



```
cor_values <- acf(df_mod$abs_sum_sum, lag.max = 159, plot = TRUE)
```

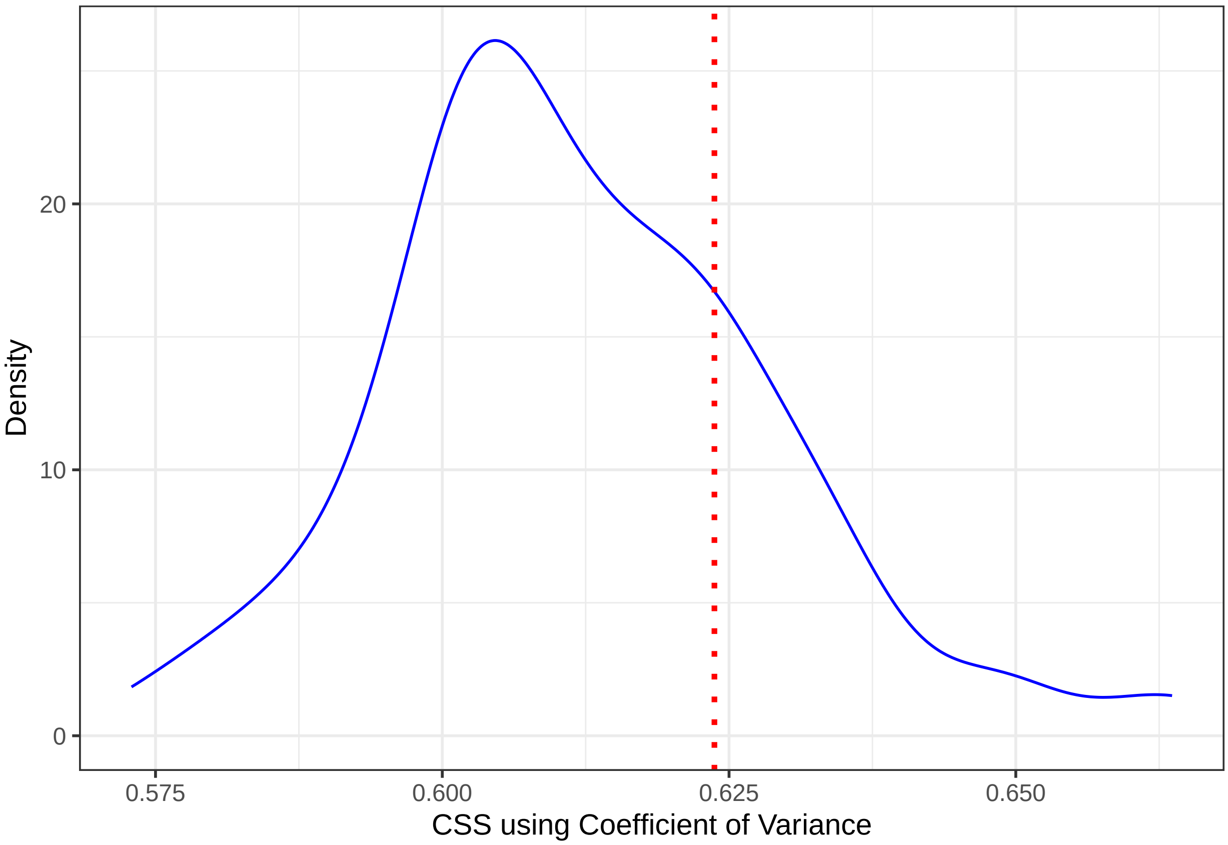


```r
instance <- permute_sequence(triplets, type="ok", min.subs=10, max.subs=30,
                             dict=codon.dict, spec.cond=FALSE, spec.region=NULL)
instance <- permute_sequence(instance, type="func", min.subs=10, max.subs=30,
                             dict=codon.dict, spec.cond=TRUE, spec.region=30:60)
# begin
instance_pasted <- paste(instance, collapse = "")
chars <- strsplit(instance_pasted, "")[[1]]
trip_instance <- sapply(seq(1, length(chars), by = 3), function(i) {
  paste(chars[i:min(i+2, length(chars))], collapse = "")
})
keyed_instance <- triplets_keying(trip_instance)
onehot_instance <-  seq_encoding_label(char_sequence = instance_pasted,
                                       maxlen = 477,
                                       start_ind = 1,
                                       vocabulary = c("A", "C", "G", "T"))
ig_instance <- integrated_gradients(
  input_seq = onehot_instance,
  baseline_type = "shuffle",
  target_class_idx = 3,
  model = model,
  num_baseline_repeats = 50)
abs_sum <- rowSums(abs(as.array(ig_instance)))
df <- data.frame(abs_sum = abs_sum, position = 1 : 477)
df_mod <- df %>%
  mutate(group = rep(1:(nrow(df) / 3), each = 3)) %>%
  group_by(group) %>%
  summarise(
    abs_sum_mean = mean(abs_sum)
  )
```

```r
df_keyed <- cbind(keyed_instance, trip_instance, df_mod)
# delete row if keyed_instance in "M","W"
df_keyed <- df_keyed %>%
  filter(!(keyed_instance %in% c("M", "W", "*")))
# groupby keyed_instance, calculate std of abs_sum_mean using summary
df_std <- df_keyed %>%
  group_by(keyed_instance) %>%
  summarise(std = sd(abs_sum_mean) / mean(abs_sum_mean))
css_instance_init <- mean(df_std$std, na.rm=TRUE)

# use: permute_sequence(instance, type="ok", min.subs=30, max.subs=80,
#                       dict=codon.dict, spec.cond=FALSE, spec.region=NULL)
# to create 100 instances, and repeat everything from "#begin"
# for each permuted instance.
# Save all css_instance, excluding the original one, into css vector
```

```r
css_values <- numeric(100)

# Loop to create 100 permuted instances
for (i in 1:100) {
  # Create a permuted instance with the specified parameters
  permuted_instance <- permute_sequence(instance, type = "ok", min.subs = 10, max.subs = 30,
                                        dict = codon.dict, spec.cond = FALSE, spec.region = NULL)

  # Start the process for the new permuted instance
  permuted_instance <- paste(permuted_instance, collapse = "")
  chars <- strsplit(permuted_instance, "")[[1]]
  trip_instance <- sapply(seq(1, length(chars), by = 3), function(i) {
    paste(chars[i:min(i+2, length(chars))], collapse = "")
  })
  keyed_instance <- triplets_keying(trip_instance)
  onehot_instance <- seq_encoding_label(char_sequence = permuted_instance,
                                        maxlen = 477,
                                        start_ind = 1,
                                        vocabulary = c("A", "C", "G", "T"))

  ig_instance <- integrated_gradients(
    input_seq = onehot_instance,
    baseline_type = "shuffle",
    target_class_idx = 3,
    model = model,
    num_baseline_repeats = 50
  )

  abs_sum <- rowSums(abs(as.array(ig_instance)))
  df <- data.frame(abs_sum = abs_sum, position = 1:477)
  df_mod <- df %>%
    mutate(group = rep(1:(nrow(df) / 3), each = 3)) %>%
    group_by(group) %>%
    summarise(
      abs_sum_mean = mean(abs_sum)
    )
  df_keyed <- cbind(keyed_instance, trip_instance, df_mod)

  # Delete rows where keyed_instance is "M", "W", or "*"
  df_keyed <- df_keyed %>%
    filter(!(keyed_instance %in% c("M", "W", "*")))

  # Group by keyed_instance and calculate the standard deviation of abs_sum_mean
  df_std <- df_keyed %>%
    group_by(keyed_instance) %>%
    summarise(std = sd(abs_sum_mean) / mean(abs_sum_mean))

  # Calculate the css_instance for this permuted instance
  css_instance <- mean(df_std$std, na.rm=TRUE)

  # Store the css_instance in the vector
  css_values[i] <- css_instance
}
```

```
# density curve of css_values
dfcss <- data.frame(CSS = css_values)
ggplot(dfcss, aes(x = CSS)) +
  geom_density(color = "blue", alpha = 0.5) +
  theme_bw() +
  # a dotted vertical line of x=css_instance_init
  geom_vline(xintercept = css_instance_init, linetype = "dotted", color = "red", lwd=1) +
  labs(x = "CSS using Coefficient of Variance", y = "Density")
```


Density
CSS using Coefficient of Variance