

fig2_ptp1b

2025-06-12

PTP1B

```
analyze_ddg_vs_esm1v <- function(ddg_file, esm_file, protein_name = "Protein") {  
  # Load data  
  test_ddg <- fread(ddg_file)  
  test_esm <- fread(esm_file)  
  
  # Prepare ESM1v column  
  colnames(test_esm)[2] <- "ESM1v"  
  
  # Construct variant column in ddg data  
  test_ddg[, new_position := pos + 1]  
  test_ddg[, variant := paste0(wtAA, new_position, mutAA)]  
  
  # Merge on variant  
  test_df <- merge(test_ddg, test_esm, by = "variant")  
  
  # Rename ddG column  
  test_df <- test_df %>% dplyr::rename(ddG_pred = `ddG (kcal/mol)`)  
  
  # Spearman correlation  
  spearman_rho <- cor.test(test_df$ddG_pred, test_df$ESM1v, method = "spearman")  
  rho_value <- round(spearman_rho$estimate, 2)  
  
  # Plot  
  p <- ggplot(test_df, aes(x = ddG_pred, y = ESM1v)) +  
    geom_bin2d(bins = 100) +  
    scale_fill_continuous(type = "viridis") +  
    theme_classic() +  
    labs(  
      x = "Predicted ddGf",  
      y = "ESM1v",  
      title = protein_name,  
      subtitle = paste("Spearman's rho =", rho_value)  
    ) +  
    theme(  
      text = element_text(size = 12),  
      legend.position = "right"  
    )  
  
  # Output  
  list(  

```

```

    spearman_rho = spearman_rho,
    plot = p,
    merged_data = test_df
  )
}
ptp1b_pred <- analyze_ddg_vs_esm1v(
  ddg_file = "/Users/xl7/Documents/0.Projects/01.protein-seq-evo-v1/data/decay_pdb/PTP1B/AF-P18031-F1-m
  esm_file = "/Users/xl7/Documents/0.Projects/00.large_supplements/ESM1v_proteome_wide/all_ESM1v/P18031
  protein_name = "PTP1B"
)

```

```

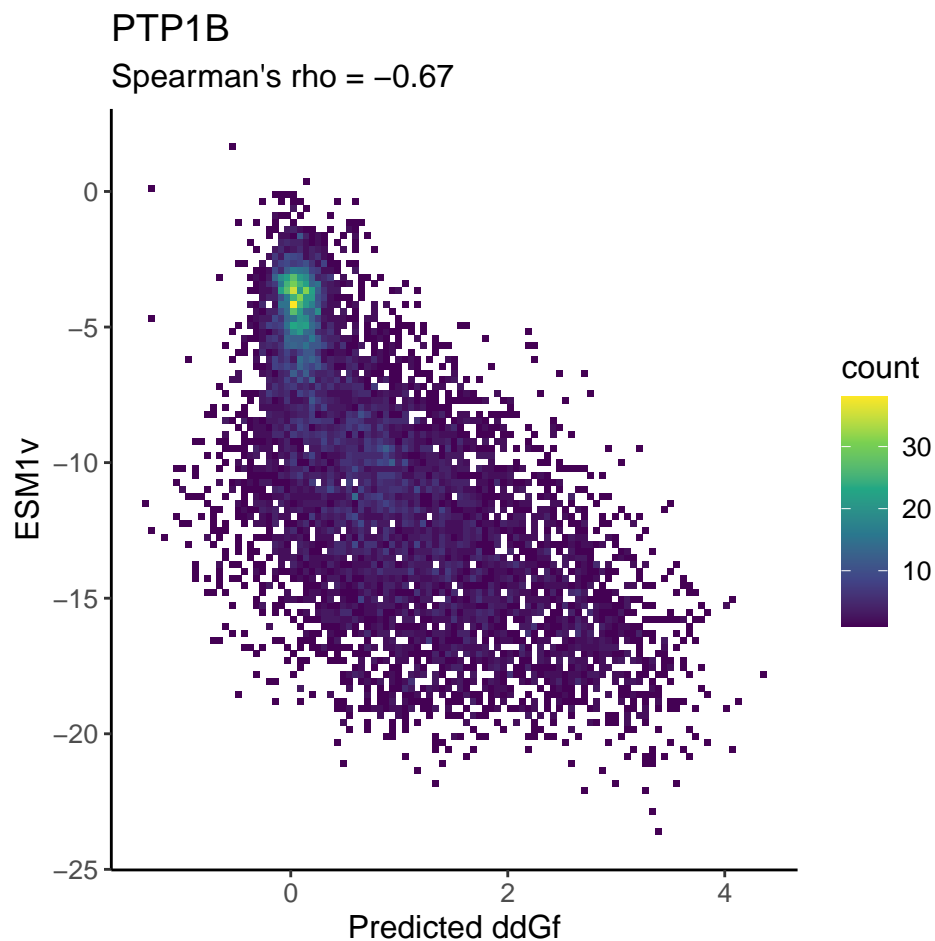
## Warning in cor.test.default(test_df$ddG_pred, test_df$ESM1v, method =
## "spearman"): Cannot compute exact p-value with ties

```

```

#result$spearman_rho #-0.5410593
p0 <- ptp1b_pred$plot
ggsave("/Users/xl7/Documents/0.Projects/01.protein-seq-evo-v1/figs/panels/fig2_ptp1b_p0.pdf",
  plot = p0, width = 5, height = 5, dpi = 300)
p0

```



```

plot_loess_residuals <- function(test_df, active_site_positions,
                                span = 0.7, protein_name = "ptp1b") {

  # Filter out active site positions
  test_df_fil <- test_df %>% filter(!new_position %in% active_site_positions)

  # Fit loess model on filtered data
  loess_fit <- loess(ESM1v ~ ddG_pred, data = test_df_fil, span = span, family = "symmetric")

  # Predict on all data
  test_df$fitted_pred <- predict(loess_fit, newdata = test_df)
  test_df$residuals_pred <- test_df$ESM1v - test_df$fitted_pred

  # Fit line data for the smooth curve
  fit_line_df <- data.frame(
    ddG_pred = seq(0,
                    max(test_df$ddG_pred, na.rm = TRUE),
                    length.out = 200)
  )
  fit_line_df$ESM1v <- predict(loess_fit, newdata = fit_line_df)

  # Spearman correlation
  spearman_result <- suppressWarnings(cor.test(test_df$ddG_pred, test_df$ESM1v, method = "spearman"))
  spearman_rho <- spearman_result$estimate
  spearman_p <- spearman_result$p.value

  # Axis and color scale limits
  xlim_vals <- range(test_df$ddG_pred, na.rm = TRUE)
  ylim_vals <- range(test_df$ESM1v, na.rm = TRUE)
  resid_limit <- max(abs(test_df$residuals_pred), na.rm = TRUE)

  # Main plot
  p <- ggplot(test_df, aes(x = ddG_pred, y = ESM1v, color = residuals_pred)) +
    geom_point(size = 2, alpha = 0.35) +
    geom_line(data = fit_line_df, aes(x = ddG_pred, y = ESM1v),
              inherit.aes = FALSE, color = "black", linewidth = 0.6) +
    labs(
      title = paste0(protein_name, ": ", nrow(test_df), " mutations"),
      subtitle = paste0("Spearman's rho = ", round(spearman_rho, 2)),
      x = "ThermoMPNN predicted ddGf",
      y = "ESM1v Pathogenicity",
      color = "ESM1v-ddGf residuals"
    ) +
    theme_classic() +
    xlim(xlim_vals) +
    ylim(ylim_vals) +
    scale_color_gradient2(
      low = "red", mid = "grey", high = "blue", midpoint = 0,
      limits = c(-resid_limit, resid_limit), name = "Residuals"
    ) +
    theme(legend.position = "left")

  # Add marginal density plots

```

```

p_marginal <- ggMarginal(
  p,
  type = "density",
  margins = "both",
  groupColour = FALSE,
  groupFill = FALSE,
  size = 10,
  colour = "grey",
  fill = "lightgrey"
)

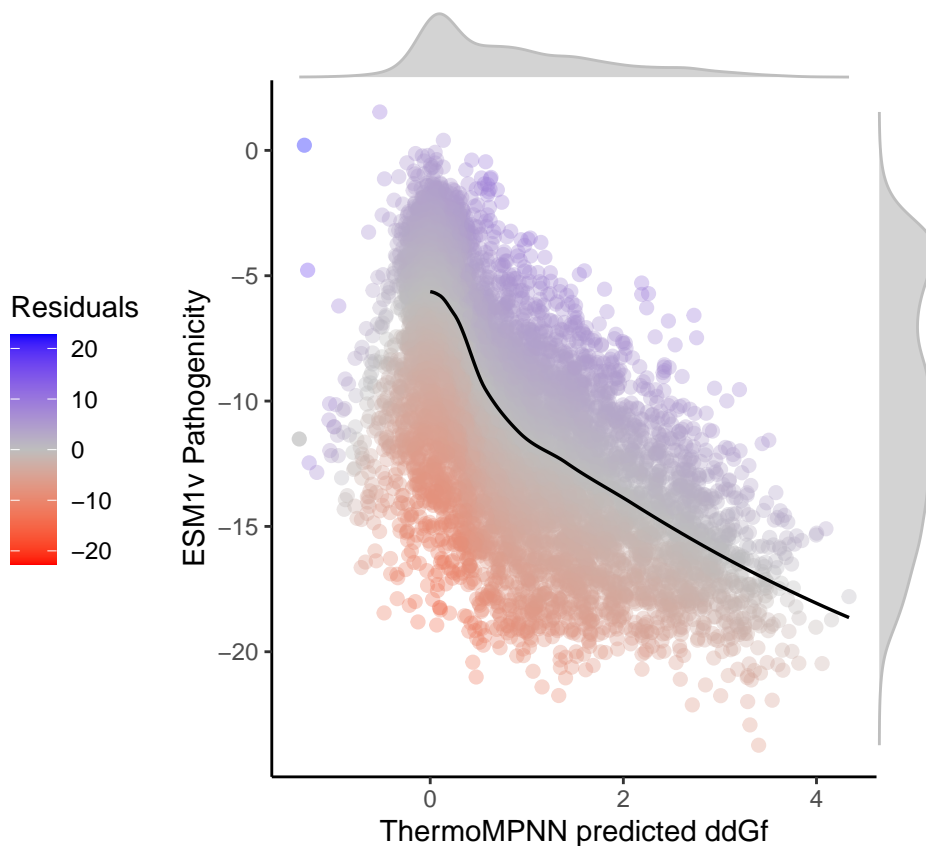
return(list(
  plot = p_marginal,
  data = test_df
))
}

ptp1b_pred_residual <- plot_loess_residuais(ptp1b_pred$merged_data, active_site_positions = c(21 , 23,
p1 <- ptp1b_pred_residual$plot
ggsave("/Users/xl7/Documents/0.Projects/01.protein-seq-evo-v1/figs/panels/fig2_ptp1b_p1.pdf",
  plot = p1, width = 5, height = 4, dpi = 300)
p1

```

PTP1B: 8265 mutations

Spearman's rho = -0.67



```

map_loess_residuals_to_pdb <- function(test_df, pdb_path, output_pdb_path) {

  # 1. Compute median residuals per position
  median_residuals <- test_df %>%
    group_by(new_position) %>%
    summarise(median_residuals = median(residuals_pred, na.rm = TRUE), .groups = "drop")

  # 2. Read in PDB
  pdb <- read.pdb(pdb_path)

  # 3. Initialize new B-factor vector
  new_b_factors <- pdb$atom$b

  # 4. Map residuals to matching residue numbers in the PDB
  for (i in seq_len(nrow(median_residuals))) {
    pos <- median_residuals$new_position[i]
    val <- median_residuals$median_residuals[i]
    matching_indices <- which(pdb$atom$resno == pos)
    new_b_factors[matching_indices] <- val
  }

  # 5. Replace non-matching indices with outlier value (e.g., 999)
  matched_positions <- unique(median_residuals$new_position)
  non_matching_indices <- which(!(pdb$atom$resno %in% matched_positions))
  new_b_factors[non_matching_indices] <- 999

  # 6. Assign and save new PDB
  pdb$atom$b <- new_b_factors
  write.pdb(pdb, file = output_pdb_path)

  # Optional: return summary
  return(list(
    min_residual = min(median_residuals$median_residuals, na.rm = TRUE),
    max_residual = max(median_residuals$median_residuals, na.rm = TRUE),
    length(non_matching_indices),
    output_file = output_pdb_path
  ))
}

pdb_residual <- map_loess_residuals_to_pdb(
  test_df = ptp1b_pred_residual$data,
  pdb_path = "~/Documents/0.Projects/01.protein-seq-evo-v1/data/decay_pdb/PTP1B/1pty.pdb",
  output_pdb_path = "~/Documents/0.Projects/01.protein-seq-evo-v1/data/decay_pdb/PTP1B/1pty_loess_resid
)

## PDB has ALT records, taking A only, rm.alt=TRUE

print(pdb_residual)

## $min_residual
## [1] -9.084527
##
## $max_residual

```

```

## [1] 7.893668
##
## [[3]]
## [1] 152
##
## $output_file
## [1] "~/Documents/0.Projects/01.protein-seq-evo-v1/data/decay_pdb/PTP1B/1pty_loess_residual.pdb"

#chimeraX
#color byattribute a:bfactor #4 & sel target csab palette -9.1,red:0,white:8,blue

# --- Read PDB and extract protein/ligand atoms ---
pdb <- read.pdb("~/Documents/0.Projects/01.protein-seq-evo-v1/data/decay_pdb/PTP1B/1pty.pdb", rm.alt = TRUE)

## PDB has ALT records, taking A only, rm.alt=TRUE

protein_ca <- pdb$atom[pdb$atom$elety == "CA" & pdb$atom$resid != "PTR" & pdb$atom$resid != "MG" & pdb$atom$resid != "MG", ]

ligand_atoms <- pdb$atom[pdb$atom$resid == "PTR" & pdb$atom$type == "HETATM", ]

# --- Compute minimum distance to ligand for each CA atom ---
protein_ca$min_dist_to_ligand <- apply(protein_ca, 1, function(atom) {
  dists <- sqrt((as.numeric(atom["x"]) - ligand_atoms$x)^2 +
    (as.numeric(atom["y"]) - ligand_atoms$y)^2 +
    (as.numeric(atom["z"]) - ligand_atoms$z)^2)
  min(dists)
})

# --- Merge with prediction data ---
merged_df <- merge(ptp1b_pred_residual$data, protein_ca, by.x = "new_position", by.y = "resno") %>%
  filter(residuals_pred <= 0)

# --- Residue-level median residuals ---
merged_df_residue <- merged_df %>%
  group_by(new_position) %>%
  summarise(loess_residual_avg = median(residuals_pred, na.rm = TRUE), .groups = "drop") %>%
  left_join(protein_ca, by = c("new_position" = "resno"))

# --- Exclude orthosteric sites for fitting ---
#orthosteric_sites <- c(24, 46, 48, 49, 182, 215, 216, 217, 219, 220, 221, 254, 258, 259, 262)
orthosteric_sites <- c(21, 23, 24, 25, 26, 28, 29, 45, 46, 47, 48, 49, 50, 85, 86, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221, 222, 223, 224, 225, 226, 227, 228, 229, 230, 231, 232, 233, 234, 235, 236, 237, 238, 239, 240, 241, 242, 243, 244, 245, 246, 247, 248, 249, 250, 251, 252, 253, 254, 255, 256, 257, 258, 259, 260, 261, 262, 263, 264, 265, 266, 267, 268, 269, 270, 271, 272, 273, 274, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284, 285, 286, 287, 288, 289, 290, 291, 292, 293, 294, 295, 296, 297, 298, 299, 300)

ortho_cutoff <- max(merged_df_residue %>% filter(new_position %in% orthosteric_sites) %>% pull(min_dist_to_ligand))
#merged_df_residue %>% filter(min_dist_to_ligand <= ortho_cutoff) %>% pull(new_position)
#21, 23, 24, 25, 26, 28, 29, 45, 46, 47, 48, 49, 50, 85, 86, 110, 111, 120, 180, 181, 182

merged_df_residue_fil <- merged_df_residue %>%
  filter(!new_position %in% orthosteric_sites)

# --- Fit exponential model ---
exp_model <- nlsLM(
  abs(loess_residual_avg) ~ a * exp(-b * min_dist_to_ligand),
  data = merged_df_residue_fil,

```

```

    start = list(a = 1, b = 0.1)
  )
exp_model

## Nonlinear regression model
##   model: abs(loess_residual_avg) ~ a * exp(-b * min_dist_to_ligand)
##   data: merged_df_residue
##       a       b
## 4.70160 0.03512
## residual sum-of-squares: 790.5
##
## Number of iterations to convergence: 8
## Achieved convergence tolerance: 1.49e-08

# --- Prediction grid ---
x_vals <- seq(min(merged_df_residue$min_dist_to_ligand),
              max(merged_df_residue$min_dist_to_ligand), length.out = 200)

# --- Bootstrapping for confidence intervals ---
set.seed(11)
boot_params <- replicate(1000, {
  samp <- merged_df_residue[sample(nrow(merged_df_residue), replace = TRUE), ]
  fit <- try(nlsLM(abs(loess_residual_avg) ~ a * exp(-b * min_dist_to_ligand),
                  data = samp, start = list(a = 1, b = 0.1)), silent = TRUE)
  if (inherits(fit, "try-error")) c(NA, NA) else coef(fit)
})
boot_params <- t(boot_params)[complete.cases(t(boot_params)), ]

boot_preds <- apply(boot_params, 1, function(p) p[1] * exp(-p[2] * x_vals))
fit_df_residue <- data.frame(
  min_dist_to_ligand = x_vals,
  loess_residual_pred = predict(exp_model, newdata = data.frame(min_dist_to_ligand = x_vals)),
  lower = apply(boot_preds, 1, quantile, probs = 0.025),
  upper = apply(boot_preds, 1, quantile, probs = 0.975)
)

# --- Model parameter extraction and derived quantity ---
model_summary <- summary(exp_model)
coefs <- coef(exp_model)
se <- model_summary$coefficients[, "Std. Error"]

residue_a <- c(coefs["a"],
               coefs["a"] - 1.96 * se["a"],
               coefs["a"] + 1.96 * se["a"])

residue_b <- c(coefs["b"],
               coefs["b"] - 1.96 * se["b"],
               coefs["b"] + 1.96 * se["b"])

half_d <- log(2) / coefs["b"]
half_d_ci <- quantile(log(2) / boot_params[, "b"], probs = c(0.025, 0.975))
residue_half_d <- c(half_d, half_d_ci)

```

```

cat("Parameter a (intercept):", residue_a, "\n")

## Parameter a (intercept): 4.701599 3.948771 5.454427

cat("Parameter b (decay rate):", residue_b, "\n")

## Parameter b (decay rate): 0.03511624 0.02477671 0.04545577

cat("Half-distance (log(2)/b):", residue_half_d, "\n")

## Half-distance (log(2)/b): 19.73865 14.42729 31.73776

# Number of iterations to convergence: 8
# Achieved convergence tolerance: 1.49e-08
# Parameter a (intercept): 4.701599 3.948771 5.454427
# Parameter b (decay rate): 0.03511624 0.02477671 0.04545577
# Half-distance (log(2)/b): 19.73865 14.42729 31.73776

# --- Annotate site types ---
merged_df_residue <- merged_df_residue %>%
  mutate(site_type = if_else(new_position %in% orthosteric_sites, "orthosteric", "non-orthosteric"))

# --- Plot ---
orange_labs <- orthosteric_sites
cyan_labs <- c(189,192,193,196,197,200,276,280,281,282)
all_labs <- union(orange_labs, cyan_labs)

p2 <- ggplot(merged_df_residue, aes(x = min_dist_to_ligand, y = abs(loess_residual_avg))) +

  # Unlabeled points
  geom_point(data = subset(merged_df_residue, !new_position %in% all_labs),
    aes(color = site_type), size = 2, alpha = 0.5) +

  # CI ribbon
  geom_ribbon(
    data = fit_df_residue,
    aes(x = min_dist_to_ligand, ymin = lower, ymax = upper),
    inherit.aes = FALSE,
    fill = "grey70",
    alpha = 0.3) +

  # Main fit line
  geom_line(
    data = fit_df_residue,
    aes(x = min_dist_to_ligand, y = loess_residual_pred),
    inherit.aes = FALSE,
    color = "black")+

  # Labeled orange points
  geom_point(data = subset(merged_df_residue, new_position %in% orange_labs),

```



```

        shape = 16, size = 2, color = "orange") +
# Labeled cyan points
geom_point(data = subset(merged_df_residue, new_position %in% cyan_labs),
           shape = 17, size = 3, color = "cyan3") +
geom_text_repel(data = subset(merged_df_residue, new_position %in% cyan_labs),
               aes(label = new_position), color = "black") +

# Reference line
geom_vline(xintercept = max(merged_df_residue %>% filter(site_type == "orthosteric") %>% pull(min_dis
               linetype = "dashed", color = "slategrey") +

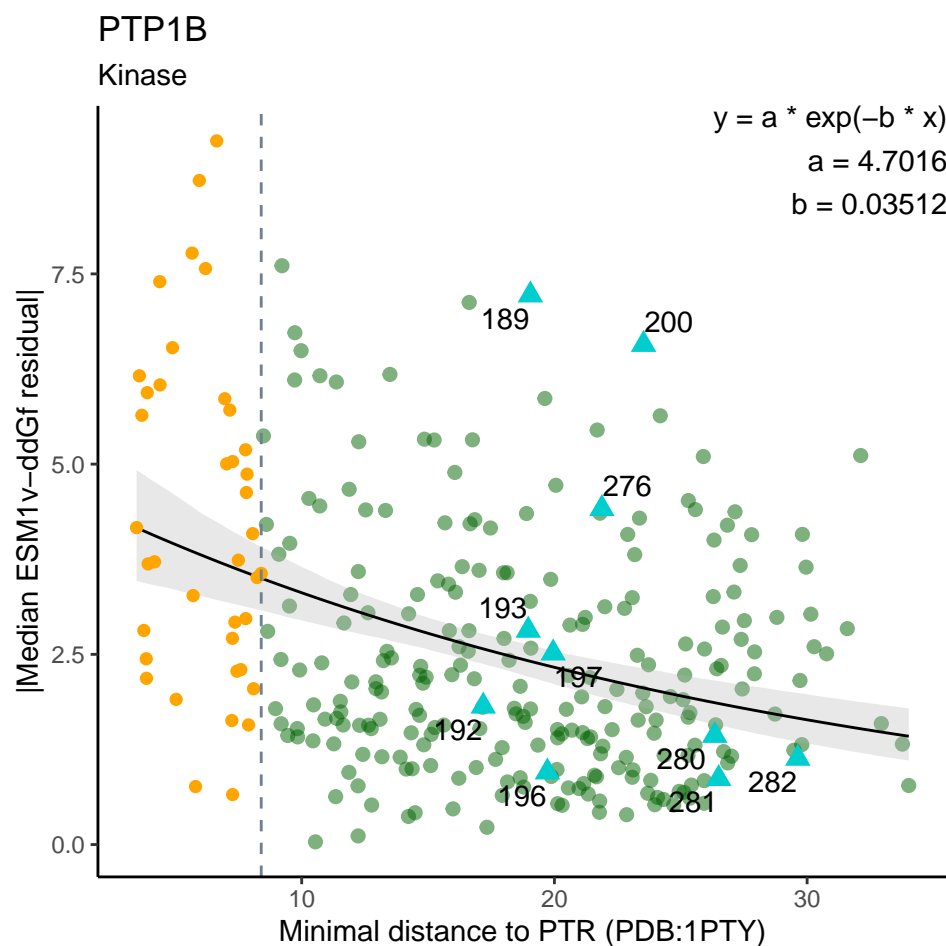
# Labels and theme
labs(
  title = "PTP1B",
  subtitle = "Kinase",
  x = "Minimal distance to PTR (PDB:1PTY)",
  y = "|Median ESM1v-ddGf residual|"
) +
theme_classic() +
theme(legend.position = "none") +
scale_color_manual(values = c("non-orthosteric" = "darkgreen", "orthosteric" = "orange")) +

annotate("text", x = Inf, y = Inf, hjust = 1, vjust = 1,
          label = sprintf("y = a * exp(-b * x)\na = %.4f\nb = %.5f", coefs["a"], coefs["b"]),
          size = 4, color = "black", hjust = 0)

## Warning: Duplicated aesthetics after name standardisation: hjust

ggsave("/Users/xl7/Documents/0.Projects/01.protein-seq-evo-v1/figs/panels/fig2_ptp1b_p2.pdf",
       plot = p2, width = 4, height = 4, dpi = 300)
p2

```



```
lm_model <- lm(log(abs(loess_residual_avg)) ~ min_dist_to_ligand, data = merged_df_residue)
summary(lm_model)
```

```
##
## Call:
## lm(formula = log(abs(loess_residual_avg)) ~ min_dist_to_ligand,
##     data = merged_df_residue)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -4.2385 -0.4141  0.0382  0.5617  1.3128
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)      1.16943    0.11482   10.185 < 2e-16 ***
## min_dist_to_ligand -0.02548    0.00608  -4.191 3.71e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.746 on 285 degrees of freedom
## Multiple R-squared:  0.05804,    Adjusted R-squared:  0.05474
## F-statistic: 17.56 on 1 and 285 DF, p-value: 3.714e-05
```

```

# Call:
# lm(formula = log(abs(loess_residual_avg)) ~ min_dist_to_ligand,
#     data = merged_df_residue)
#
# Residuals:
#      Min       1Q   Median       3Q      Max
# -4.2385 -0.4141  0.0382  0.5617  1.3128
#
# Coefficients:
#              Estimate Std. Error t value Pr(>|t|)
# (Intercept)      1.16943     0.11482  10.185 < 2e-16 ***
# min_dist_to_ligand -0.02548     0.00608  -4.191 3.71e-05 ***
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 0.746 on 285 degrees of freedom
# Multiple R-squared:  0.05804, Adjusted R-squared:  0.05474
# F-statistic: 17.56 on 1 and 285 DF,  p-value: 3.714e-05

```

```
allosteric_sites <- c(189,192,193,196,197,200,276,280,281,282)
```

```
merged_df_residue <- merged_df_residue %>%
```

```
  mutate(site_class = case_when(
    new_position %in% orthosteric_sites ~ "orthosteric",
    new_position %in% allosteric_sites ~ "allosteric",
    TRUE ~ "other"
  ))
```

```
label_df <- merged_df_residue %>%
```

```
  group_by(site_class) %>%
  summarise(
    n = n(),
    median_val = median(abs(loess_residual_avg), na.rm = TRUE),
    .groups = "drop"
  )
```

```
merged_df_residue$site_class <- factor(merged_df_residue$site_class, levels = c("orthosteric", "allosteric", "other"))
```

```

p3 <- ggplot(merged_df_residue, aes(x = site_class, y = abs(loess_residual_avg), fill = site_class)) +
  geom_violin(trim = FALSE, scale = "width", alpha = 0.8, color = NA) +
  geom_jitter(width = 0.15, size = 2, alpha = 0.7, color = "lightgrey") +
  stat_summary(fun = median, geom = "crossbar", width = 0.4, color = "black", fatten = 1) +
  stat_summary(fun = median, geom = "point", shape = 23, size = 2, fill = "black", color = "black", stroke = "black") +
  # Add sample size n=xxx above each group
  geom_text(
    data = label_df,
    aes(x = site_class, y = max(abs(merged_df_residue$loess_residual_avg)) * 1.1,
      label = paste0("n = ", n)),
    inherit.aes = FALSE,
    size = 4
  ) +
  geom_text(
    data = label_df,

```

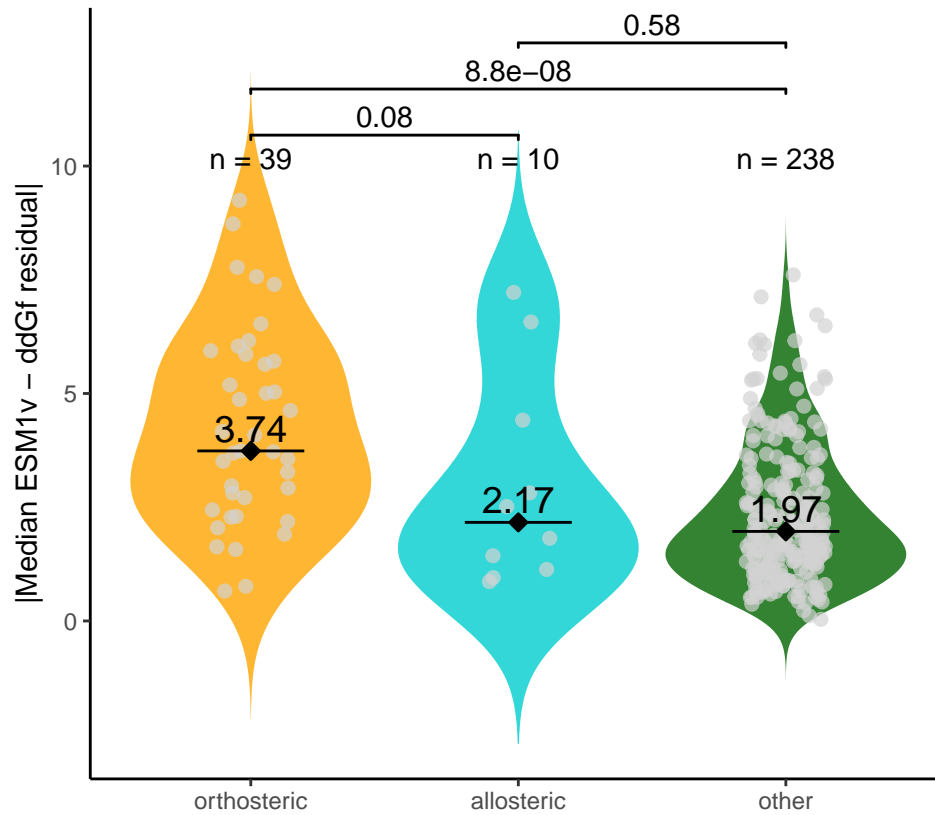
```

aes(x = site_class, y = median_val + 0.5, label = sprintf("%.2f", median_val)),
inherit.aes = FALSE,
size = 5
) +
# Significance bars
geom_signif(
  comparisons = list(
    c("orthosteric", "allosteric"),
    c("orthosteric", "other"),
    c("allosteric", "other")
  ),
  map_signif_level = FALSE,
  test = "wilcox.test",
  step_increase = 0.1,
  tip_length = 0.01
) +

# Labels and theme
labs(
  title = "PTP1B",
  subtitle = "",
  x = "",
  y = "|Median ESM1v - ddGf residual|"
) +
scale_fill_manual(values = c(
  "orthosteric" = "orange",
  "allosteric" = "cyan3",
  "other" = "darkgreen"
)) +
theme_classic() +
theme(legend.position = "none")
ggsave("/Users/xl7/Documents/0.Projects/01.protein-seq-evo-v1/figs/panels/fig2_ptp1b_p3.pdf",
  plot = p3, width = 3, height = 4, dpi = 300)
p3

```

PTP1B



```
set.seed(11)

# Get fixed per-residue medians
fixed_df <- merged_df_residue %>%
  filter(site_class %in% c("orthosteric", "allosteric")) %>%
  mutate(median_resid = abs(loess_residual_avg)) %>%
  dplyr::select(site_class, median_resid)

# Bootstrap medians for 'other' group
n_sample <- sum(merged_df_residue$site_class == "allosteric")

boot_medians_other <- map_dfr(1:1000, function(i) {
  sampled <- merged_df_residue %>%
    filter(site_class == "other") %>%
    slice_sample(n = n_sample)

  tibble(
    site_class = "other",
    median_resid = median(abs(sampled$loess_residual_avg), na.rm = TRUE),
    replicate = i
  )
})
```

```

# Combine all
plot_df <- bind_rows(
  fixed_df %>% mutate(replicate = NA),
  boot_medians_other
)
head(plot_df)

## # A tibble: 6 x 3
##   site_class median_resid replicate
##   <chr>         <dbl>         <int>
## 1 orthosteric     1.57             NA
## 2 orthosteric     0.657            NA
## 3 orthosteric     2.44             NA
## 4 orthosteric     0.762            NA
## 5 orthosteric     2.71             NA
## 6 orthosteric     1.63             NA

# Labels
label_df <- plot_df %>%
  group_by(site_class) %>%
  summarise(
    n = n(),
    median_val = median(median_resid),
    y_max = max(median_resid),
    .groups = "drop"
  )

label_df <- label_df %>%
  mutate(n_label = case_when(
    site_class == "other" ~ "bootstrapped 1000 times",
    TRUE ~ paste0("n = ", n)
  ))

plot_df$site_class <- factor(plot_df$site_class, levels = c("orthosteric", "allosteric", "other"))

# Plot
p4 <- ggplot(plot_df, aes(x = site_class, y = median_resid, fill = site_class)) +
  geom_violin(data = plot_df,
    trim = FALSE, scale = "width", alpha = 0.8, color = NA) +

  geom_jitter(data = plot_df,
    width = 0.15, size = 2, alpha = 0.7, color = "lightgrey") +

  stat_summary(fun = median, geom = "crossbar", width = 0.4, color = "black", fatten = 1) +
  stat_summary(fun = median, geom = "point", shape = 23, size = 2.5,
    fill = "black", color = "black", stroke = 0.7) +

  geom_text(
    data = label_df,
    aes(x = site_class, y = y_max * 1.1, label = n_label),
    inherit.aes = FALSE,
    size = 4) +

```

```

geom_text(
  data = label_df,
  aes(x = site_class, y = median_val + 0.25, label = sprintf(" %.2f", median_val)),
  inherit.aes = FALSE,
  size = 4
) +

geom_signif(
  comparisons = list(
    c("orthosteric", "other"),
    c("allosteric", "other"),
    c("allosteric", "orthosteric")
  ),
  test = "wilcox.test",
  map_signif_level = FALSE,
  step_increase = 0.1,
  tip_length = 0.01
) +

labs(
  title = "PTP1B",
  subtitle = "",
  x = "",
  y = "|Median residual (ESM1v - ddGf)|"
) +
scale_fill_manual(values = c(
  "orthosteric" = "orange",
  "allosteric" = "cyan",
  "other" = "darkgreen"
)) +
theme_classic() +
theme(legend.position = "none")

ggsave("/Users/xl7/Documents/0.Projects/01.protein-seq-evo-v1/figs/panels/fig2_ptp1b_p4.pdf",
  plot = p4, width = 3, height = 4, dpi = 300)
p4

```

PTP1B

