fitting ddm

2023-10-22

R Markdown

Lets try and fit this in Stan Hierarchically!

```
trials = 300
subjects = 10

mu_alpha = 2
sd_alpha = 0.5

mu_delta = 2
sd_delta = 0.5

mu_beta = 0.5
sd_beta = 1

mu_tau = 0.3
sd_tau = 0.1

alphas = array(NA,subjects)
deltas = array(NA,subjects)
betas = array(NA,subjects)
taus = array(NA,subjects)
trials = rep(trials,subjects)

alphas = truncnorm::rtruncnorm(subjects,0,Inf,mu_alpha, sd_alpha)
deltas = rnorm(subjects,mu_delta, sd_delta)
betas = brms::inv_logit_scaled(rnorm(subjects,mu_beta, sd_beta))
taus = truncnorm::rtruncnorm(subjects,0,Inf,mu_tau, sd_tau)



apply_rwiener <- function(index, n, alpha, delta, beta, tau) {

  result <- rwiener(n = n,
                    alpha = alpha,
                    delta = delta,
                    beta = beta,
                    tau = tau)

  result_df <- data.frame(index = index,
                          alpha = alpha,
                          trials = n,
                          beta = beta,
```

```
                          tau = tau,
                          delta = delta,
                          result)
  return(result_df)
}
```

```
# Applying the function to each element of the vectors
results_list <- mapply(apply_rwiener,
                       index = 1:length(trials),
                       n = trials,
                       alpha = alphas,
                       delta = deltas,
                       beta = betas,
                       tau = taus,
                       SIMPLIFY = FALSE)

#getting the results in a dataframe
result_df <- do.call(rbind, results_list)




mod = cmdstanr::cmdstan_model(here::here("stan_scripts","Hierarchical Models","Hier_ddm.stan"))

data_stan = list(trials = nrow(result_df),
                 S = length(unique(result_df$index)),
                 S_id = result_df$index,
                 minRT = result_df %>% group_by(index) %>% summarize(minrt = min(q)) %>% .$minrt,
                 RT = result_df$q,
                 resp = result_df %>% .$resp
)

# fit <- mod$sample(
#     data = data_stan,
#     chains = 4,
#     parallel_chains = 4,
#     adapt_delta = 0.9,
#     refresh = 50,
#     max_treedepth = 12,
#     init = 0)

#fit$save_object(here::here("report","Hierarchical models","ddm","workspace","model.RDS"))
fit <- readRDS("~/HDDM_stan/report/Hierarchical models/ddm/workspace/model.RDS")



replacements <- c("gm[1]" = "mu_delta",
                  "gm[2]" = "mu_alpha",
                  "gm[3]" = "mu_beta",
                  "gm[4]" = "mu_tau",
                  "tau_u[1]" = "sd_delta",
                  "tau_u[2]" = "sd_alpha",
                  "tau_u[3]" = "sd_beta",
```

```
                "tau_u[4]" = "sd_tau"
                )
```

Lets look at the summary of the model Note that: 1 is delta 2 is log(alpha) 3 is logit(beta) 4 is logit(tau)/min(RT)

```
flextable::flextable(as_draws_df(fit$summary()) %>%   mutate(variable = case_when(
    variable %in% names(replacements) ~ replacements[variable],
    TRUE ~ variable
  )) %>% mutate_if(is.numeric, round, digits = 2) %>% head(9))
```

```
## Warning: fonts used in ‘flextable‘ are ignored because the ‘pdflatex‘ engine is
## used and not ‘xelatex‘ or ‘lualatex‘. You can avoid this warning by using the
## ‘set_flextable_defaults(fonts_ignore=TRUE)‘ command or use a compatible engine
## by defining ‘latex_engine: xelatex‘ in the YAML header of the R Markdown
## document.
```

variable	mean	median	sd	mad	q5	q95	rhat	ess_bulk	ess_tail
lp___	1,148.60	1,148.81	6.58	6.55	1,137.28	1,158.97	1.01	856.42	1,865.06
mu_delta	1.95	1.95	0.19	0.17	1.66	2.25	1.00	2,099.46	2,304.36
mu_alpha	0.64	0.63	0.15	0.14	0.40	0.89	1.00	1,801.14	1,853.92
mu_beta	0.68	0.70	0.29	0.27	0.20	1.14	1.00	1,709.33	2,313.20
mu_tau	2.17	2.18	0.37	0.35	1.52	2.76	1.00	2,179.94	2,695.57
sd_delta	0.53	0.50	0.18	0.15	0.31	0.86	1.00	1,892.26	2,534.40
sd_alpha	0.44	0.41	0.15	0.12	0.26	0.72	1.00	1,917.22	2,293.16
sd_beta	0.94	0.90	0.25	0.22	0.62	1.41	1.00	1,660.02	2,526.06
sd_tau	1.19	1.14	0.30	0.27	0.78	1.74	1.00	2,086.78	2,683.40

Prior posterior updates

```
variables = c("gm[1]","gm[2]","gm[3]","gm[4]",
              "tau_u[1]","tau_u[2]","tau_u[3]","tau_u[4]")



posteriors = as_draws_df(fit$draws(variables = variables)) %>%
  select(variables) %>%
  pivot_longer(everything())%>% mutate(posterior = T)
```

```
## Warning: Using an external vector in selections was deprecated in tidyselect 1.1.0.
## i Please use ‘all_of()‘ or ‘any_of()‘ instead.
##   # Was:
##   data %>% select(variables)
##
##   # Now:
##   data %>% select(all_of(variables))
```

```
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

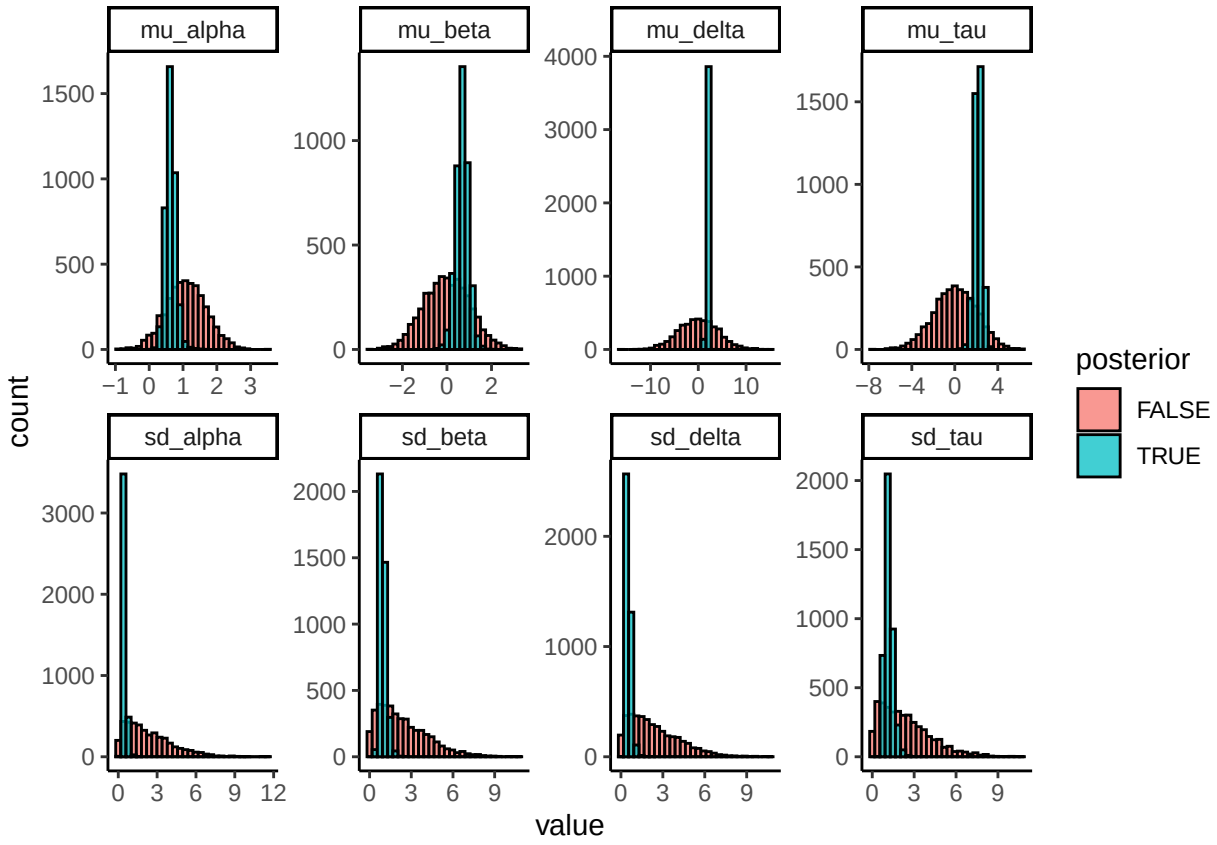
```
## Warning: Dropping 'draws_df' class as required metadata was removed.
```

```
prior_variables = paste0("prior_",variables)
priors = as_draws_df(fit$draws(variables = prior_variables)) %>%
  select(prior_variables) %>%
  pivot_longer(everything()) %>% mutate(posterior = F) %>%
  mutate(name = gsub("prior_","",name))
```

```
## Warning: Using an external vector in selections was deprecated in tidyselect 1.1.0.
## i Please use `all_of()` or `any_of()` instead.
##   # Was:
##   data %>% select(prior_variables)
##
##   # Now:
##   data %>% select(all_of(prior_variables))
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## Dropping 'draws_df' class as required metadata was removed.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
rbind(priors,posteriors) %>%
  mutate(name = case_when(
    name %in% names(replacements) ~ replacements[name],
    TRUE ~ name
  )) %>%
  ggplot(aes(x = value, fill = posterior))+
  geom_histogram(alpha = 0.75, position = 'identity', col = "black")+
  facet_wrap(~name, scales = "free", ncol = 4)+
  theme_classic()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
## make it ready to combine with single subject fits:

test = rbind(priors,posteriors) %>%
  mutate(name = case_when(
    name %in% names(replacements) ~ replacements[name],
    TRUE ~ name),
    variable = substr(name, 4, nchar(name)),
    ID = NA) %>%
  mutate(value = ifelse(variable == "tau", brms::inv_logit_scaled(value),value))
```

```
source(here::here("report","Hierarchical models","ddm","scripts","plots.R"))

variables = c("tau","alpha","delta","beta")


posteriors = as_draws_df(fit$draws(variables = variables)) %>%
  select(contains(variables)) %>% mutate(posterior = T)
```

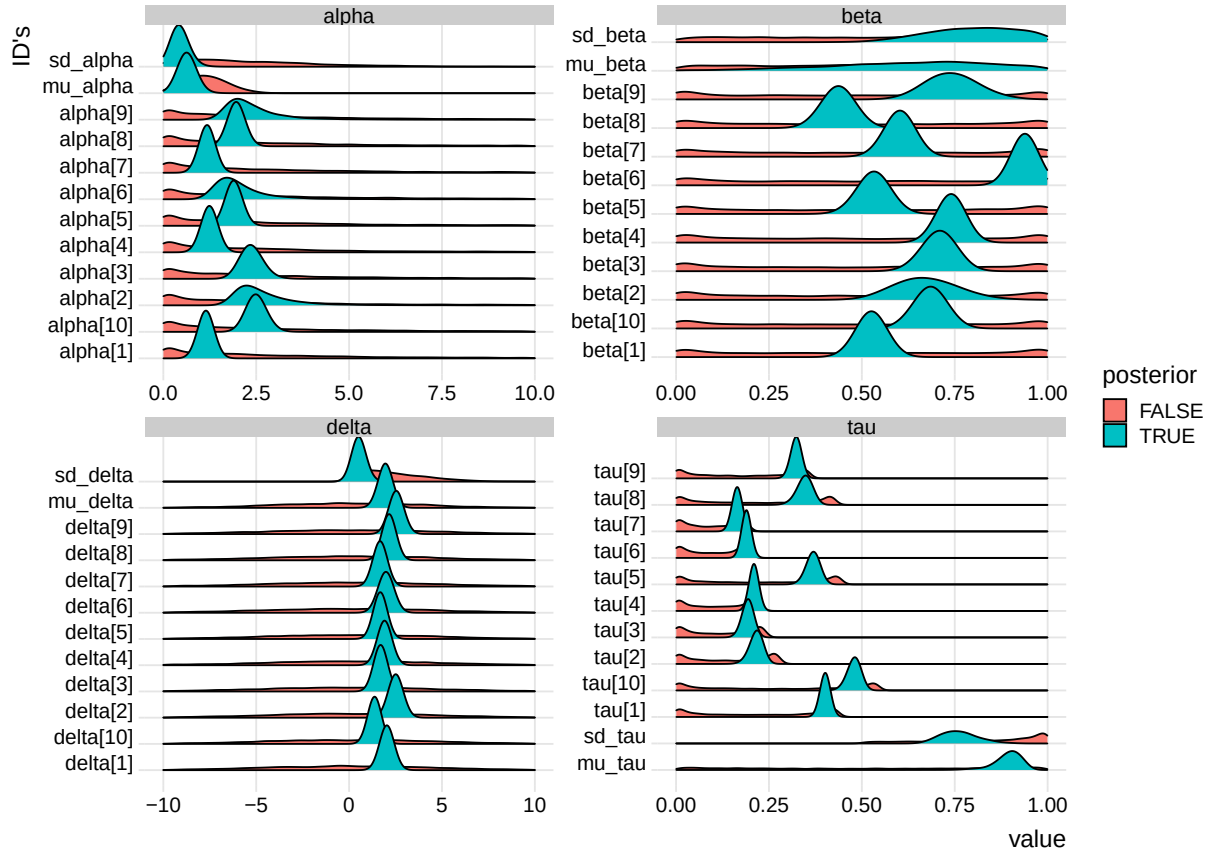
```
## Warning: Dropping 'draws_df' class as required metadata was removed.
```

```
prior_variables = paste0("prior_",variables)
priors = as_draws_df(fit$draws(variables = prior_variables)) %>%
  select(-c(.chain,.iteration,.draw)) %>% mutate(posterior = F) %>%
  rename_all(~sub("^prior_", "", .))
```

```
## Warning: Dropping 'draws_df' class as required metadata was removed.
```

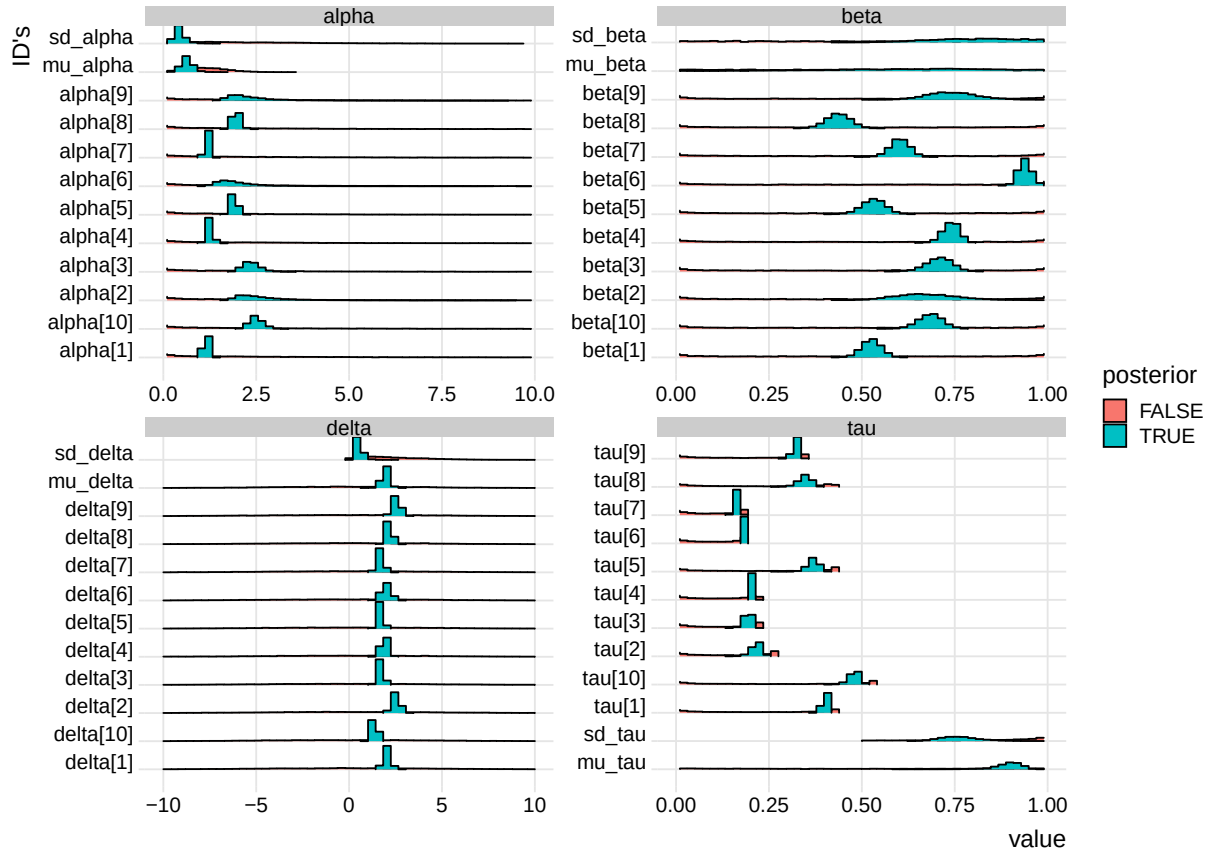
```
#Density plot
rbind(posteriors,priors) %>% pivot_longer(starts_with(variables))  %>%
  mutate(variable = sub("\\[\\d+\\]", "", name), ID = as.numeric(sub(".*\\[(\\d+)\\]", "\\1", name))) %
  rbind(.,test) %>%
  ggplot(aes(x = value, y = name, fill = posterior))+
    geom_density_ridges() +
    theme_ridges()+ylab("ID's")+
  facet_wrap_custom(~variable, scales = "free", scale_overrides = list(
    scale_override(1, scale_x_continuous(limits = c(0, 10))),
    scale_override(2, scale_x_continuous(limits = c(0, 1))),
    scale_override(3, scale_x_continuous(limits = c(-10, 10))),
    scale_override(4, scale_x_continuous(limits = c(0, 1)))
  ))
```

```
## Picking joint bandwidth of 0.221

## Picking joint bandwidth of 0.0343

## Picking joint bandwidth of 0.361

## Picking joint bandwidth of 0.0132

## Warning: Removed 19734 rows containing non-finite values
## (`stat_density_ridges()`).
```



```
# Histogram
rbind(posteriors,priors) %>% pivot_longer(starts_with(variables))  %>%
  mutate(variable = sub("\\[\\d+\\]", "", name), ID = as.numeric(sub(".*\\[(\\d+)\\]", "\\1", name))) %>
  rbind(.,test) %>%
  ggplot(aes(x = value, y = name, fill = posterior))+
  geom_density_ridges(stat = "binline", bins = 50, scale = 0.95, draw_baseline = FALSE)+
    theme_ridges()+ylab("ID's")+
  facet_wrap_custom(~variable, scales = "free", scale_overrides = list(
    scale_override(1, scale_x_continuous(limits = c(0, 10))),
    scale_override(2, scale_x_continuous(limits = c(0, 1))),
    scale_override(3, scale_x_continuous(limits = c(-10, 10))),
    scale_override(4, scale_x_continuous(limits = c(0, 1)))
  ))
```

```
## Warning: Removed 19734 rows containing non-finite values (`stat_binline()`).
```

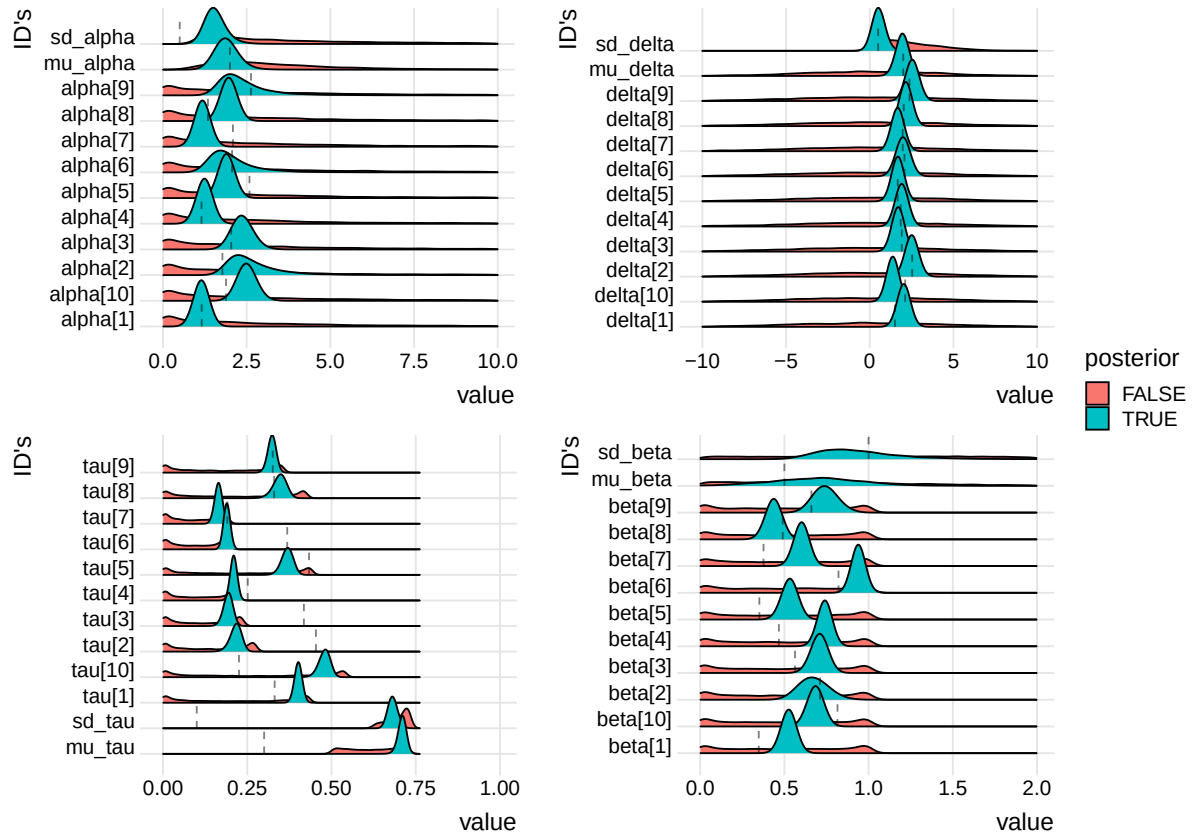



```
variables = c("alpha","tau","delta","beta")

plot_list <- list()

for(variable1 in variables){


  global_estimates = data.frame(ID = NA,
```

```
                                name = c(paste0("mu_",variable1), paste0("sd_",variable1)),
                                value = c(get(paste0("mu_",variable1)), get(paste0("sd_",variable1))),
                                variable = variable1,
                                posterior = NA)


  real_data = result_df %>% mutate(trials = NULL, q = NULL , resp = NULL) %>%
    pivot_longer(cols = c("alpha","tau","delta","beta"), names_to = "variable") %>%
    rename(ID = index) %>%
    distinct() %>%
    mutate(name = paste(variable, "[", ID, "]", sep = ""), posterior = NA) %>%
    filter(variable == variable1) %>%
    rbind(., global_estimates)



plotdata = rbind(posteriors,priors) %>% pivot_longer(starts_with(variables))  %>%
    mutate(variable = sub("\\[\\d+\\]", "", name), ID = as.numeric(sub(".*\\[(\\d+)\\]", "\\1", name)))
    rbind(.,test) %>%
    filter(variable == variable1) %>%
    mutate(value = ifelse(name == "mu_alpha", exp(value), value),
           value = ifelse(name == "sd_alpha", exp(value), value),
           value = ifelse(name == "mu_tau", inv_logit_scaled(value), value),
           value = ifelse(name == "sd_tau", inv_logit_scaled(value), value)
           )


  plot_list[[variable1]] = plotdata %>% ggplot(aes(x = value, y = name,fill = posterior))+
      geom_density_ridges() +
      theme_ridges()+
    geom_segment(data = real_data, aes(x = value, xend = value, y = as.numeric(as.factor(name)),yend =
      {if(variable1 == "alpha")scale_x_continuous(lim = c(0,10))}+
      {if(variable1 == "tau")scale_x_continuous(lim = c(0,1))}+
      {if(variable1 == "delta")scale_x_continuous(lim = c(-10,10))}+
      {if(variable1 == "beta")scale_x_continuous(lim = c(0,2))}+
      ylab("ID's")

    # facet_wrap_custom(~variable, scales = "free", scale_overrides = list(
    #   scale_override(1, scale_x_continuous(limits = c(0, 10))),
    #   scale_override(2, scale_x_continuous(limits = c(0, 1))),
    #   scale_override(3, scale_x_continuous(limits = c(-10, 10))),
    #   scale_override(4, scale_x_continuous(limits = c(0, 1)))
    # ))

}


# Create a 2x2 matrix layout
library(patchwork)
((plot_list[["alpha"]]+plot_list[["delta"]])/
(plot_list[["tau"]]+plot_list[["beta"]]))+plot_layout(guides = "collect")
```

```
## Picking joint bandwidth of 0.236
```

```
## Warning: Removed 11903 rows containing non-finite values
## (`stat_density_ridges()`).

## Picking joint bandwidth of 0.361

## Warning: Removed 1944 rows containing non-finite values
## (`stat_density_ridges()`).

## Picking joint bandwidth of 0.0101

## Picking joint bandwidth of 0.0388

## Warning: Removed 4195 rows containing non-finite values
## (`stat_density_ridges()`).
```



Predictive checks? #One problem if alpha gets to high the Rweiner runs kind of forever so setting a max of 8 on alpha here.

```
n_draws = trials[1]

drawv = rbinom(n_draws,4000,extraDistr::rprop(n_draws,1,0.5))
```

```r
parameters = as_draws_df(fit$draws()) %>%
  select(matches(c("alpha","tau","beta","delta"))) %>%
  mutate(tau_raw = NULL) %>%
  select(-starts_with("prior_")) %>%
  mutate(draw = 1:nrow(.)) %>%
  slice(draww) %>%
  pivot_longer(-draw)%>%
  separate(name, into = c("Parameter", "ID"), sep = "\\[|\\]") %>%
  filter(Parameter != "tau_u")
```

```
## Warning: Dropping 'draws_df' class as required metadata was removed.
```

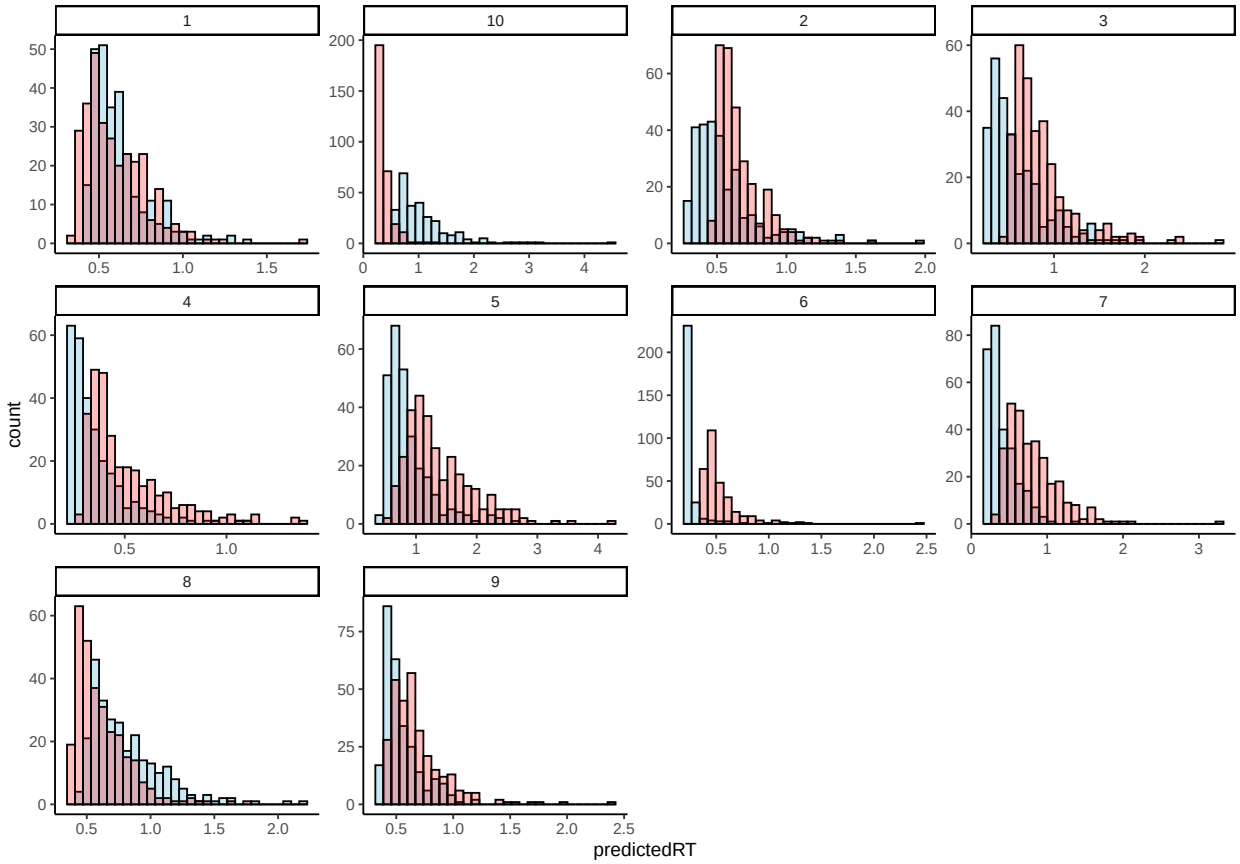
```
## Warning: Expected 2 pieces. Additional pieces discarded in 13068 rows [1, 2, 3, 4, 5, 6,
## 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, ...].
```

```r
parameters %>% mutate(value = ifelse((Parameter == "alpha" & value > 8), 8, value)) %>%
  pivot_wider(names_from = Parameter, values_from = value,values_fn = {mean}) %>%
  rowwise() %>% mutate(predictedRT = RWiener::rwiener(1,alpha,tau,beta,delta)[[1]]) %>%
  ggplot(aes(predictedRT))+
  geom_histogram(fill = "lightblue2", alpha = 0.7, col = "black")+
  facet_wrap(~ID, scales = "free")+
  theme_classic()+
  geom_histogram(data = result_df %>% rename(ID = index), aes(x = q), alpha = 0.25, fill = "red", col =
```

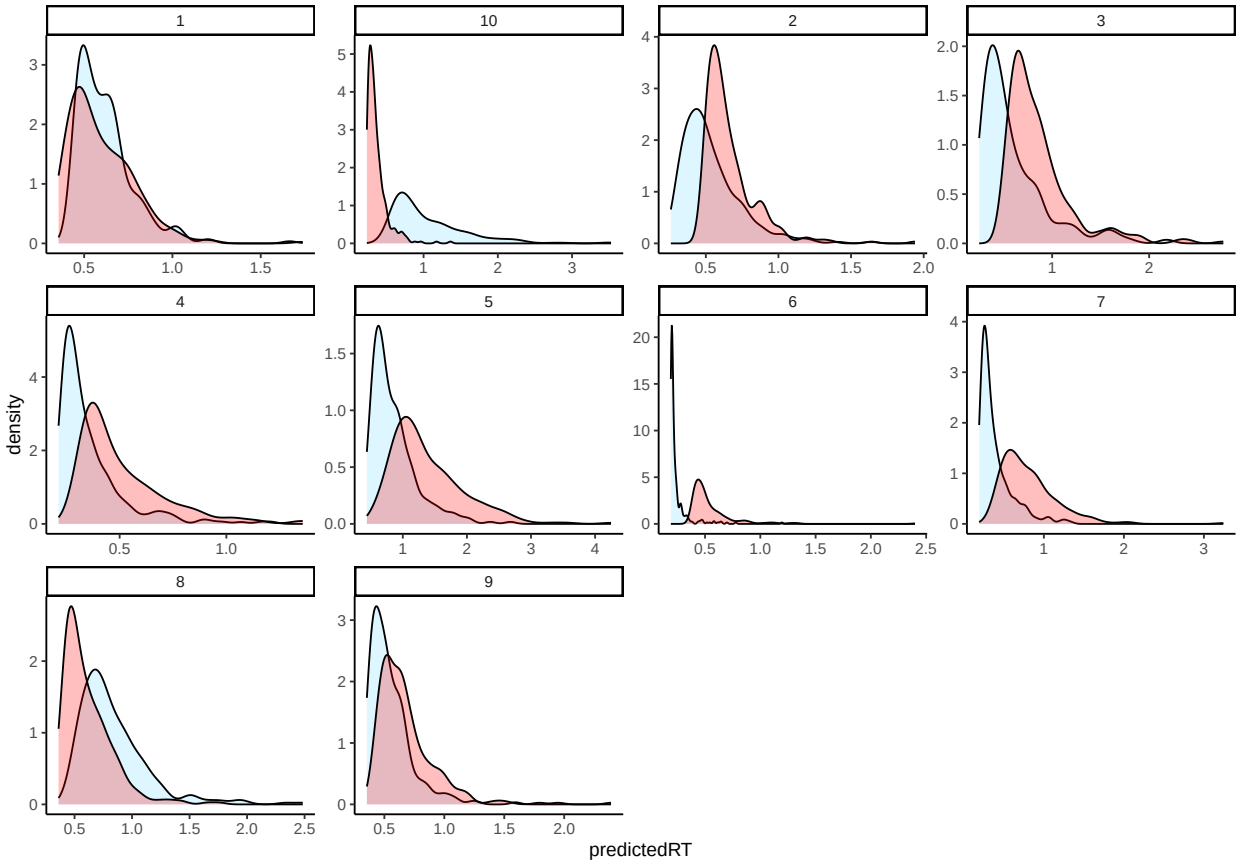
```
## Warning: Combining variables of class <character> and <integer> was deprecated in
## ggplot2 3.4.0.
## i Please ensure your variables are compatible before plotting (location:
##   `combine_vars()`)
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
## Warning: Combining variables of class <integer> and <character> was deprecated in
## ggplot2 3.4.0.
## i Please ensure your variables are compatible before plotting (location:
##   `combine_vars()`)
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
parameters %>% pivot_wider(names_from = Parameter, values_from = value,values_fn = {mean}) %>%
  rowwise() %>%
  mutate(predictedRT = RWiener::rwiener(1,alpha,tau,beta,delta)[[1]],
         predictedresp = RWiener::rwiener(1,alpha,tau,beta,delta)[[2]]) %>%
  ggplot(aes(predictedRT), col = "black")+
  geom_density(fill = "lightblue1", alpha = 0.5, col = "black")+
  facet_wrap(~ID, scales = "free")+
  theme_classic()+
  geom_density(data = result_df %>% rename(ID = index), aes(x = q), alpha = 0.25, fill = "red", col = "
```



Lets do some parameter recovery on this model!

fit model

```
get_simulation = function(parameters){

trials = parameters$trials
subjects = parameters$subjects

mu_alpha = parameters$mu_alpha
sd_alpha = parameters$sd_alpha

mu_delta = parameters$mu_delta
sd_delta = parameters$sd_delta

mu_beta = parameters$mu_beta
sd_beta = parameters$sd_beta

mu_tau = parameters$mu_tau
sd_tau = parameters$sd_tau

alphas = array(NA,subjects)
deltas = array(NA,subjects)
```

```r
betas = array(NA,subjects)
taus = array(NA,subjects)
trials = rep(trials,subjects)


alphas = truncnorm::rtruncnorm(subjects,0.01,Inf,mu_alpha, sd_alpha)
deltas = rnorm(subjects,mu_delta, sd_delta)
betas = brms::inv_logit_scaled(rnorm(subjects,mu_beta, sd_beta))
taus = truncnorm::rtruncnorm(subjects,0.01,Inf,mu_tau, sd_tau)



apply_rwiener <- function(index, n, alpha, delta, beta, tau) {

  result <- rwiener(n = n,
                    alpha = alpha,
                    delta = delta,
                    beta = beta,
                    tau = tau)

  result_df <- data.frame(index = index,
                          alpha = alpha,
                          trials = n,
                          beta = beta,
                          tau = tau,
                          delta = delta,
                          result)
  return(result_df)
}

# Applying the function to each element of the vectors
results_list <- mapply(apply_rwiener,
                       index = 1:length(trials),
                       n = trials,
                       alpha = alphas,
                       delta = deltas,
                       beta = betas,
                       tau = taus,
                       SIMPLIFY = FALSE)

#getting the results in a dataframe
result_df <- do.call(rbind, results_list)
return(result_df)
}




fit_model = function(parameters){
  id = parameters$id

  data = get_simulation(parameters)
```

```
mod = cmdstanr::cmdstan_model(here::here("stan_scripts","Hier","Hier_ddm.stan"))

data_stan = list(trials = nrow(result_df),
                 S = length(unique(result_df$index)),
                 S_id = result_df$index,
                 minRT = result_df %>% group_by(index) %>% summarize(minrt = min(q)) %>% .$minrt,
                 RT = result_df$q,
                 resp = result_df %>% .$resp
)

fit <- mod$sample(
  data = data_stan,
  chains = 4,
  parallel_chains = 4,
  adapt_delta = 0.9,
  refresh = 50,
  max_treedepth = 12,
  init = 0)


global_estimates = c("gm[1]","gm[2]","gm[3]","gm[4]",
              "tau_u[1]","tau_u[2]","tau_u[3]","tau_u[4]")


subject_estimates = c("tau","alpha","delta","beta")

pattern <- paste0("(", paste(subject_estimates, collapse = "|"), ")\\[\\d+\\]")


diag = data.frame(fit$diagnostic_summary(), id)


posteriors_global_estimates = as_draws_df(fit$summary()) %>%
  filter(variable %in% global_estimates) %>% mutate(trials = trials[[1]],
                                                    subjects = subjects,
                                                    real_mu_alpha = mu_alpha,
                                                    real_sd_alpha = sd_alpha,
                                                    real_mu_beta = mu_beta,
                                                    real_sd_beta = sd_beta,
                                                    real_mu_tau = mu_tau,
                                                    real_sd_tau = sd_tau,
                                                    real_mu_delta = mu_delta,
                                                    real_sd_delta = sd_delta,
                                                    )



posteriors_subject_estimates = as_draws_df(fit$summary()) %>%
  mutate(extracted_variable = str_extract(as_draws_df(fit$summary()) %>% .$variable, pattern)) %>%
  drop_na() %>% mutate(num_div = diag$num_divergent,
                               tree_depth = diag$num_max_treedepth,
                               real_alpha = parameters$alpha,
                               real_delta = parameters$delta,
```

```
                                real_beta = parameters$beta,
                                real_tau = parameters$tau,
                                trials = parameters$trials,
                                id = id) %>% select(-contains("."))



  return(list(posteriors_global_estimates,posteriors_subject_estimates, diag))
}
```

```
trials = seq(50,100,by = 25)
subjects = seq(5,25,by = 10)

mu_alpha = seq(2,3,by = 1)
sd_alpha = seq(1,2,by = 1)

mu_delta = seq(-2,2,by = 2)
sd_delta = seq(1,2,by = 1)

mu_beta = seq(0.4,0.6,by = 0.2)
sd_beta = seq(1,2,by = 1)

mu_tau = seq(0.3,0.4,by = 0.1)
sd_tau = seq(0.1,0.2,by = 0.1)


replicate = 1:1

parameters = expand.grid(mu_alpha = mu_alpha,
                         sd_alpha = sd_alpha,
                         mu_delta = mu_delta,
                         sd_delta = sd_delta,
                         mu_beta = mu_beta,
                         sd_beta = sd_beta,
                         mu_tau = mu_tau,
                         sd_tau = sd_tau,
                         subjects = subjects,
                         trials = trials,
                         replicate = replicate) %>%
  mutate(id = 1:nrow(.))


data_list <- split(parameters, parameters$id)

# fit_model(data_list[[1]])
```

```
# cores = availableCores()-1
#
# plan(multisession, workers = 10)
#
# possfit_model = possibly(.f = fit_model, otherwise = "Error")
#
# results <- future_map(data_list, ~possfit_model(.x), .progress = TRUE, .options = furrr_options(seed
```

```
# error_indices <- which(results == "Error")
#
# unique(error_indices)
#
# results2 = results[results != "Error"]
#
# results = NULL
```