Plotting all the results

Amy Pitts

2/19/2022

Jimmy needs to give me the right esitamte of effect right now here is what I am using

```
smaller_true_ATE <- 0.15
bigger_true_ATE <- 0.3
```

```
load("./output_data/binary_scen_1.RData")
load("./output_data/binary_scen_2.RData")
load("./output_data/binary_scen_3.RData")
load("./output_data/binary_scen_4.RData")
load("./output_data/binary_scen_5.RData")
load("./output_data/binary_scen_6.RData")
# load("./output_data/binary_scen_13.RData")
# load("./output_data/binary_scen_14.RData")
load("./output_data/binary_scen_15.RData")
load("./output_data/binary_scen_16.RData")
load("./output_data/binary_scen_17.RData")
load("./output_data/binary_scen_18.RData")
```

```
all_scenarios <- tibble(
  id = c(1:18),
  n_sample = c(rep(1000, 6), rep(10000, 6), rep(100, 6)),
  desired_prop = rep(c(0.1, 0.1, 0.2, 0.2, 0.3, 0.3),3),
  beta1 = rep(c(0.767, 1.587),9),
  beta0 = rep(c(0.422, 0.873, 0.46, 0.952, 0.499, 1.032), 3)
)
```

Get all the odd numbers $\beta_1 = 0.767$

```
binary_final_odd <- binary_scen_1 %>% mutate(n_sample = 1000, beta1 = 0.767, desired_prop = 0.1) %>%
  bind_rows(binary_scen_3 %>% mutate(n_sample = 1000, beta1 = 0.767, desired_prop = 0.2)) %>%
 bind_rows(binary_scen_5 %>% mutate(n_sample = 1000, beta1 = 0.767, desired_prop = 0.3)) %>%
 # rbind(binary_scen_13 %>% mutate(n_sample = 100, beta1 = 0.767, desired_prop = 0.1)) %>%
 bind_rows(binary_scen_15 %>% mutate(n_sample = 100, beta1 = 0.767, desired_prop = 0.2)) %>%
 bind_rows(binary_scen_17 %>% mutate(n_sample = 100, beta1 = 0.767, desired_prop = 0.3))

binary_final_odd <- binary_final_odd %>%
  mutate(
    ATE_bias = ATE - smaller_true_ATE,
    empirical_bias = empirical_mean - smaller_true_ATE,
    boot_type = ifelse(boot_type == 0, "Simple", "Complex")
  )
```

Get all the even numbers $\beta_1 = 1.587$

```
binary_final_even <- binary_scen_2 %>% mutate(n_sample = 1000, beta1 = 1.587, desired_prop = 0.1) %>%
  bind_rows(binary_scen_4 %>% mutate(n_sample = 1000, beta1 = 1.587, desired_prop = 0.2)) %>%
 bind_rows(binary_scen_6 %>% mutate(n_sample = 1000, beta1 = 1.587, desired_prop = 0.3)) %>%
 # bind_rows(binary_scen_14 %>% mutate(n_sample = 100, beta1 = 1.587, desired_prop = 0.1)) %>%
 bind_rows(binary_scen_16 %>% mutate(n_sample = 100, beta1 = 1.587, desired_prop = 0.2)) %>%
 bind_rows(binary_scen_18 %>% mutate(n_sample = 100, beta1 = 1.587, desired_prop = 0.3))


binary_final_even <- binary_final_even %>%
  mutate(
    ATE_bias = ATE - bigger_true_ATE,
    empirical_bias = empirical_mean - bigger_true_ATE,
    boot_type = ifelse(boot_type == 0, "Simple", "Complex")
  )
```

```
binary_final <- binary_final_even %>% bind_rows(binary_final_odd)
```

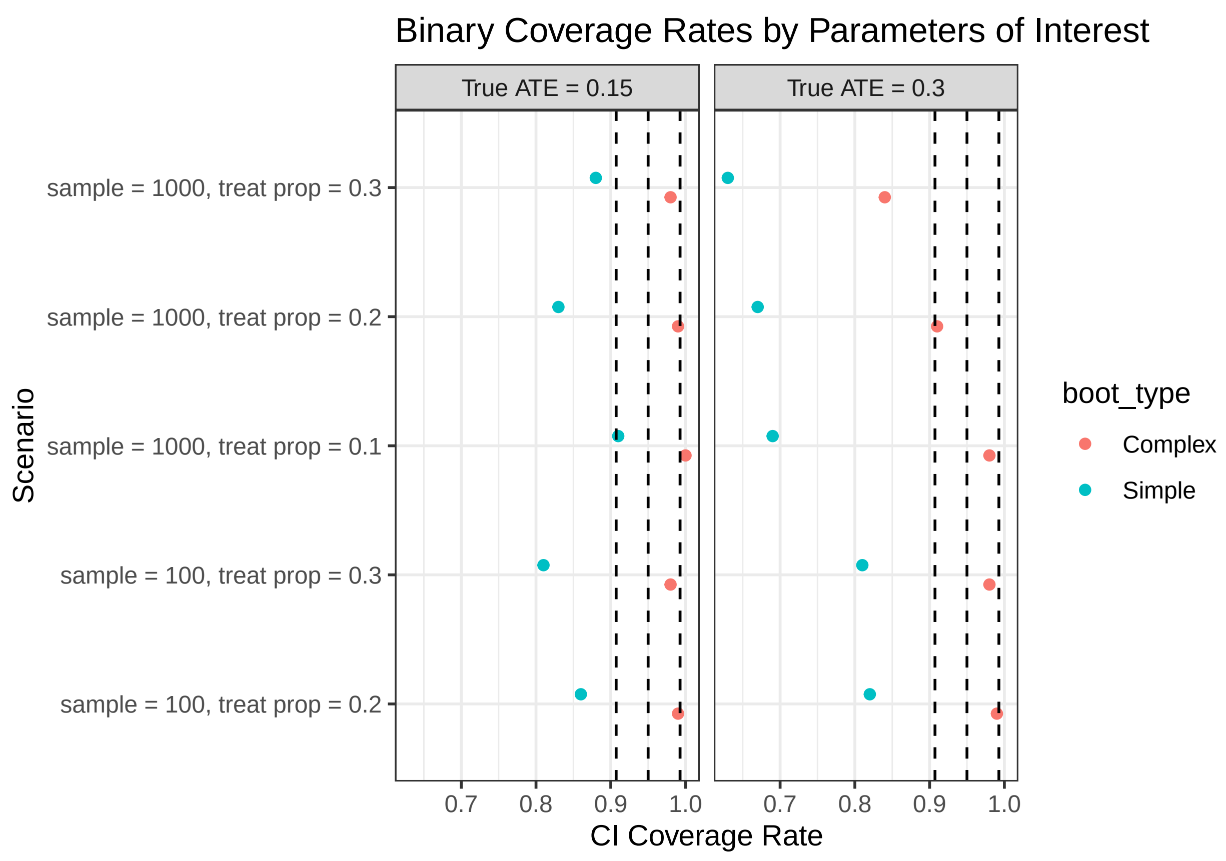
Coverage rate!

```
cr_df<- binary_final %>%
  mutate(scenario = factor(scenario),
         new_name = str_c("sample = ", n_sample, ", treat prop = ", desired_prop),
         treat_effect = ifelse(beta1 == 0.767, "True ATE = 0.15", "True ATE = 0.3")) %>%
  group_by( new_name, treat_effect, boot_type ) %>%
  summarize(cr = sum(covered)/ 100)
```

```
## `summarise()` has grouped output by 'new_name', 'treat_effect'. You can override using the `.groups`
```

name the scenarios by sample size and treat prop and facet by the treatment effect

```
ggplot(cr_df, aes(x=cr, y=new_name, color=boot_type)) +
  geom_point(position=position_dodge(0.3))+
  geom_vline(xintercept=0.9927, linetype="dashed", color = "black") +
   geom_vline(xintercept=0.9072, linetype="dashed", color = "black") +
   geom_vline(xintercept=0.95, linetype="dashed", color = "black") +
  facet_grid(~treat_effect) +
  labs(
   title = "Binary Coverage Rates by Parameters of Interest",
   y = "Scenario",
   x = "CI Coverage Rate"
  )+theme_bw()
```



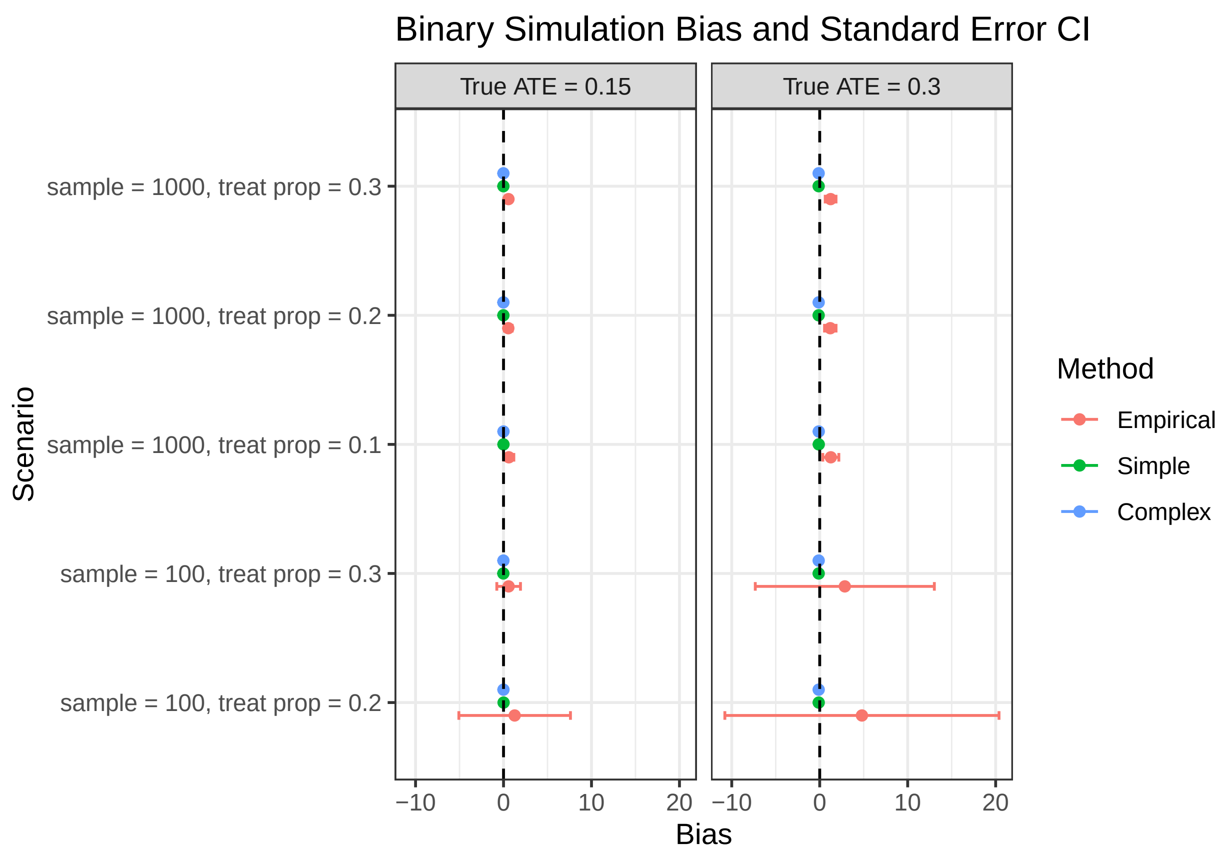
Bias

```
empirical_data <- binary_final %>%
  mutate(
    scenario = factor(scenario),
    new_name = str_c("sample = ", n_sample, ", treat prop = ", desired_prop),
    treat_effect = ifelse(beta1 == 0.767, "True ATE = 0.15", "True ATE = 0.3")
  ) %>%
  select(scenario, new_name, treat_effect, empirical_bias, empirical_se) %>%
  unique() %>%
  rename(
    ATE_bias  = empirical_bias,
    bias_se = empirical_se
  ) %>%
  mutate(
    Method = "Empirical"
  )

boot_data <- binary_final %>%
  mutate(scenario = factor(scenario),
         new_name = str_c("sample = ", n_sample, ", treat prop = ", desired_prop),
         treat_effect = ifelse(beta1 == 0.767, "True ATE = 0.15", "True ATE = 0.3"),
         Method = boot_type
         ) %>%
  group_by(Method, scenario, new_name, treat_effect ) %>%
  summarize(
    ATE_bias = mean(ATE_bias),
    bias_se = mean(sd_ATE)
  )
```

```
## `summarise()` has grouped output by 'Method', 'scenario', 'new_name'. You can override using the `.g
```

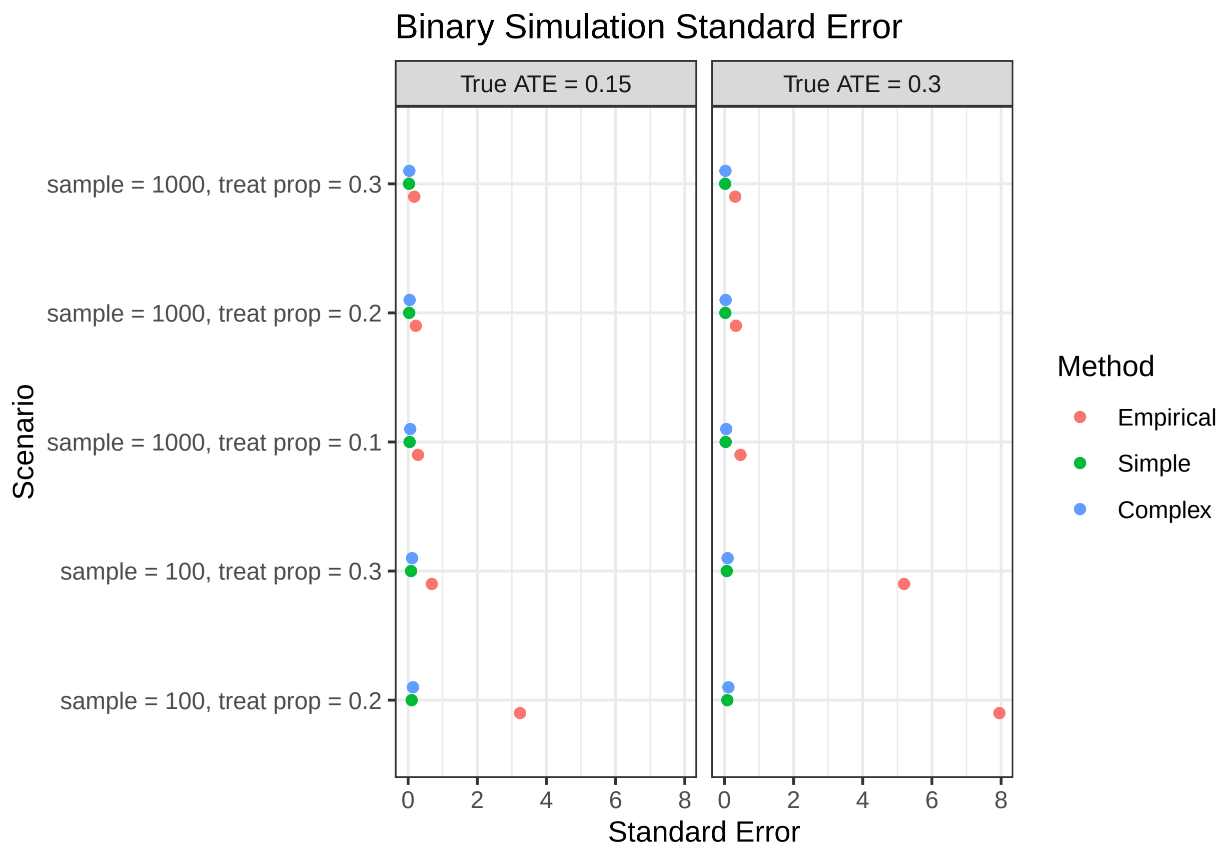
```
bind_rows(boot_data,empirical_data ) %>%
  mutate(
    Method = factor(Method, levels = c("Empirical", "Simple", "Complex"))
  ) %>%
  ggplot( aes(x=ATE_bias, y=new_name, color=Method)) +
    geom_point(position=position_dodge(0.3))+
    geom_errorbar(aes(xmin=ATE_bias-1.96*bias_se, xmax=ATE_bias+1.96*bias_se), width=.2,
                 position=position_dodge(0.3)) +
    geom_vline(xintercept=0, linetype="dashed", color = "black") +
    facet_grid(~treat_effect) +
    labs(
     title = "Binary Simulation Bias and Standard Error CI",
     y = "Scenario",
     x = "Bias"
    )+theme_bw()
```



Standard Error

```
bind_rows(boot_data,empirical_data ) %>%
  mutate(
    Method = factor(Method, levels = c("Empirical", "Simple", "Complex"))
  ) %>%
```

```r
ggplot( aes(x=bias_se, y=new_name, color=Method)) +
  geom_point(position=position_dodge(0.3))+
  facet_grid(~treat_effect) +
  labs(
   title = "Binary Simulation Standard Error",
   y = "Scenario",
   x = "Standard Error"
  )+theme_bw()
```


Binary Simulation Standard Error
True ATE = 0.15
True ATE = 0.3
sample = 1000, treat prop = 0.3
sample = 1000, treat prop = 0.2
sample = 1000, treat prop = 0.1
sample = 100, treat prop = 0.3
sample = 100, treat prop = 0.2
Scenario
Standard Error
Method
Empirical
Simple
Complex