CIIL analysis

1 Define functions

```
create_bar_plot <- function(data, var) {

  plt_data =  data %>%
              group_by(Model) %>%                        # Group data by factors
              summarise(meanVal = mean(!!sym(var)), ci.lower = bootES(!!sym(var))$bou
nds[1], ci.upper = bootES(!!sym(var))$bounds[2])

plot_ly(data = plt_data, x = ~Model, y = ~meanVal, type = 'bar', color = ~Model, erro
r_y = list(symmetric=F, type = "data", array=~ci.upper-meanVal, arrayminus=~meanVal-c
i.lower, color="black"))
}

check_for_normality <- function(data, var) {
  results_by_model <- data %>%
  summarise(
    W_statistic = shapiro.test(!!sym(var))$statistic,    # Shapiro-Wilk W statistic
    p_value = shapiro.test(!!sym(var))$p.value           # p-value
  )
  print(results_by_model)
}

run_anova <- function(data, var) {
  aov_model <- aov_ez(id = "ID",
                    dv = var,
                    within = "Model",
                    data = data)
  print(summary(aov_model))

  printContrasts <- function(expr, model) {
   contrast(emmeans(model, expr), method="pairwise", adjust = "bonferroni")
  }
  printContrasts("Model", aov_model)
}
```

2 Preparing data

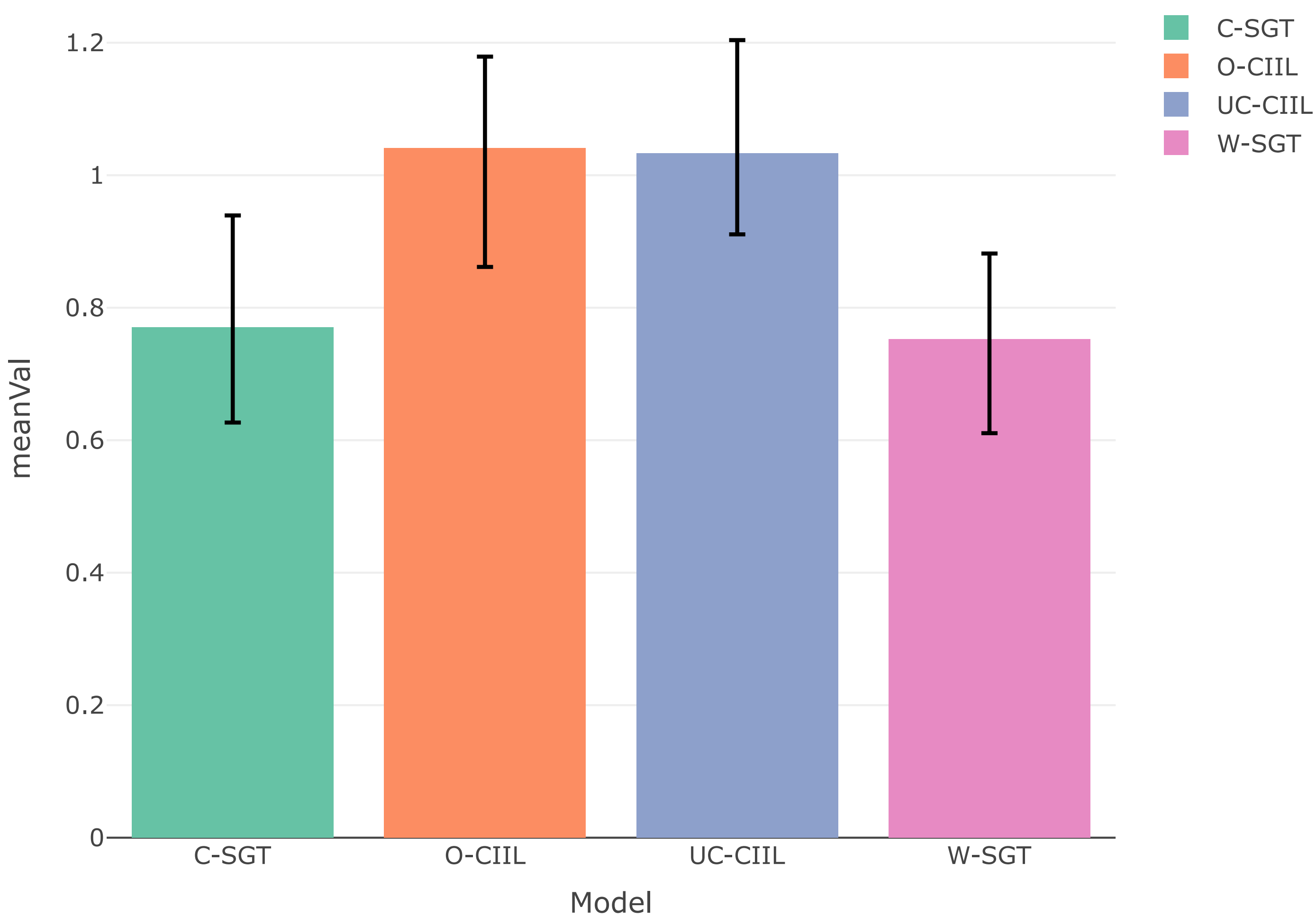
```
data = read.table("stats.csv", header=TRUE, sep=",")
data$Model <- as.factor(data$Model)
```

3 Evaluate Throughput

3.1 Plot throughput

Represent a bar chart with 95% CI for each model and block

```
create_bar_plot(data, "Throughput")
```



3.2 Check for normality

```
check_for_normality(data, "Throughput")
```

```
##   W_statistic   p_value
## 1   0.9835679 0.5525809
```

3.3 Run an Anova

```
run_anova(data, "Throughput")
```

```
##
## Univariate Type III Repeated-Measures ANOVA Assuming Sphericity
##
##             Sum Sq num Df Error SS den Df F value    Pr(>F)
## (Intercept) 51.776      1   5.0718     15 153.131 2.837e-09 ***
## Model        1.216      3   0.8936     45  20.409 1.697e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Mauchly Tests for Sphericity
##
##       Test statistic p-value
## Model        0.67422 0.36907
##
##
## Greenhouse-Geisser and Huynh-Feldt Corrections
##  for Departure from Sphericity
##
##        GG eps Pr(>F[GG])
## Model 0.78594  4.207e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##          HF eps   Pr(>F[HF])
## Model 0.9419507 4.046554e-08
```

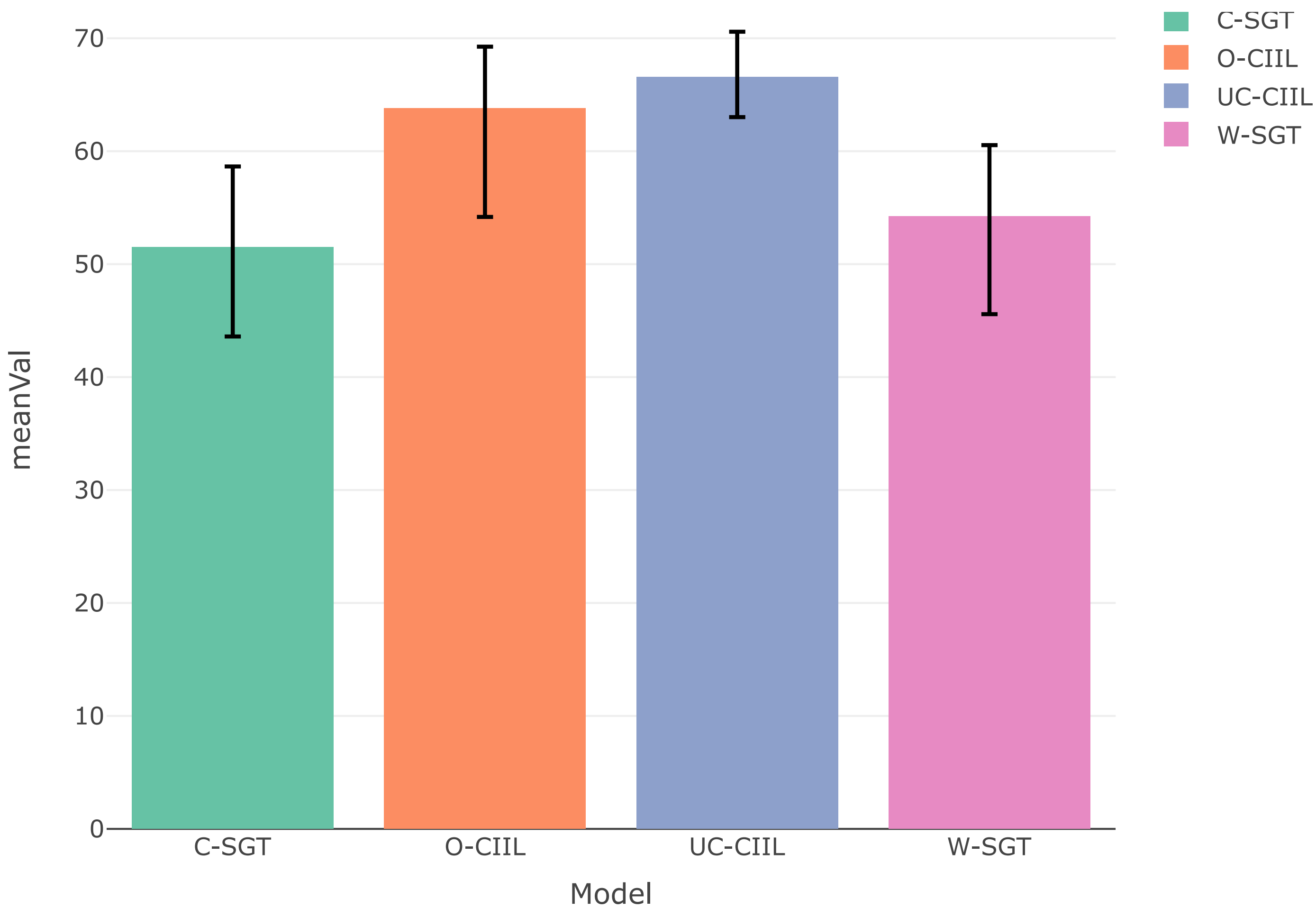
```
##  contrast         estimate     SE df t.ratio p.value
##  C.SGT - O.CIIL    -0.2704 0.0638 15  -4.235  0.0043
##  C.SGT - UC.CIIL   -0.2622 0.0520 15  -5.045  0.0009
##  C.SGT - W.SGT      0.0180 0.0399 15   0.450  1.0000
##  O.CIIL - UC.CIIL   0.0082 0.0471 15   0.174  1.0000
##  O.CIIL - W.SGT     0.2884 0.0502 15   5.740  0.0002
##  UC.CIIL - W.SGT    0.2802 0.0421 15   6.650  <.0001
##
## P value adjustment: bonferroni method for 6 tests
```

4 Evaluate Path Efficiency

4.1 Plot path efficiency

Represent a bar chart with 95% CI for each model and block

```
create_bar_plot(data, "PathEfficiency")
```



4.2 Check for normality

```
check_for_normality(data, "PathEfficiency")
```

```
##   W_statistic      p_value
## 1   0.9239228 0.0007277742
```

4.3 Run a Friedman Test

```
friedman.test(PathEfficiency ~ Model | ID, data = data)
```

```
##
##  Friedman rank sum test
##
## data:  PathEfficiency and Model and ID
## Friedman chi-squared = 23.1, df = 3, p-value = 3.849e-05
```

```
frdAllPairsNemenyiTest(data$PathEfficiency, data$Model, blocks = data$ID)
```

```
##
##   Pairwise comparisons using Nemenyi-Wilcoxon-Wilcox all-pairs test for a two-way b
alanced complete block design
```

```
## data: y, groups and blocks
```

```
##         C-SGT   O-CIIL  UC-CIIL
## O-CIIL  0.00210 -       -
## UC-CIIL 0.00023 0.94719 -
## W-SGT   0.69233 0.06554 0.01381
```

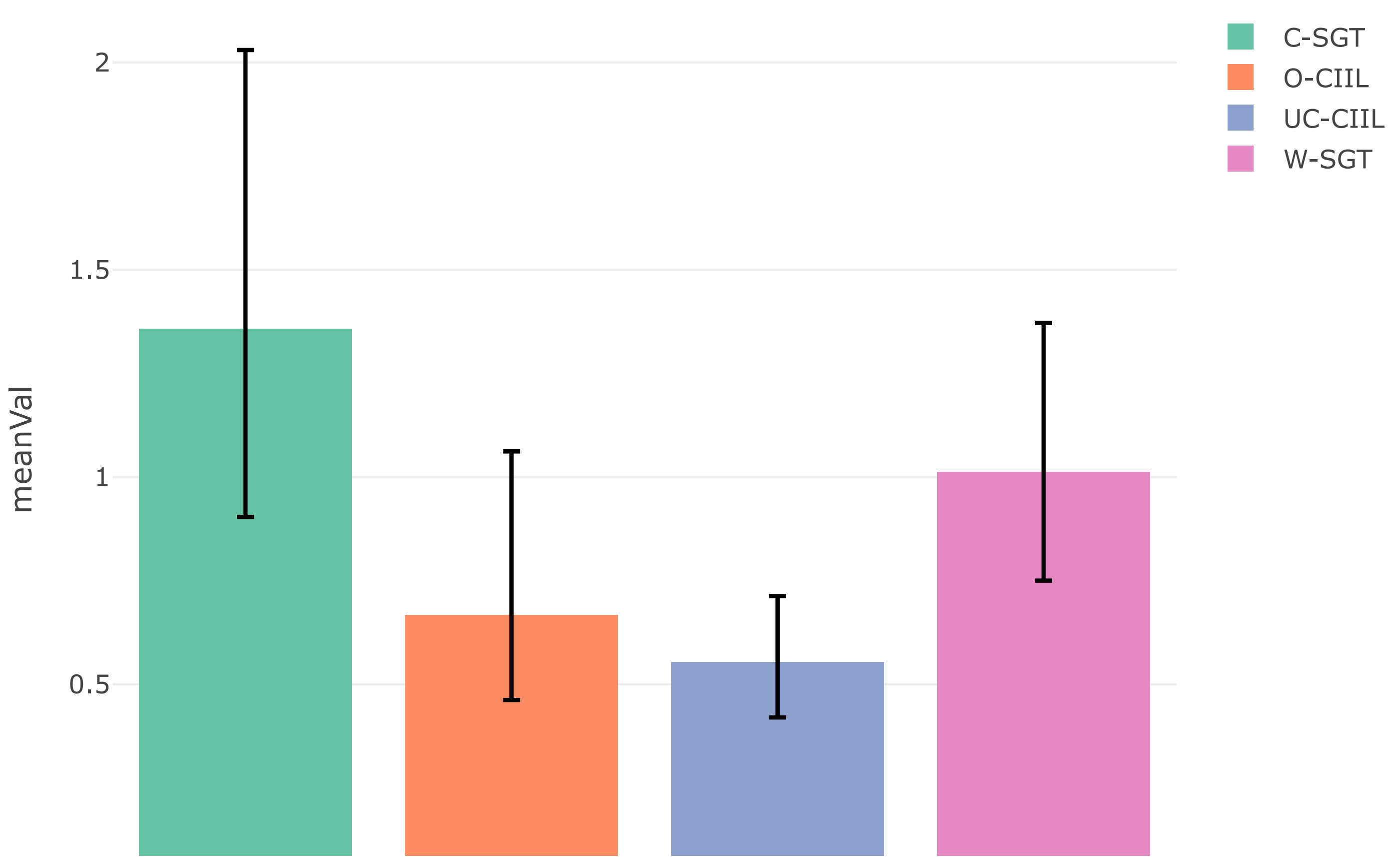
```
##
## P value adjustment method: single-step
```

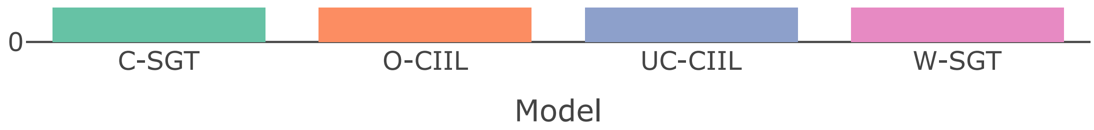
5 Evaluate Overshoots

5.1 Plot overshoots

Represent a bar chart with 95% CI for each model and block

```
create_bar_plot(data, "Overshoots")
```





5.2 Check for normality

```
check_for_normality(data, "Overshoots")
```

```
##   W_statistic      p_value
## 1    0.772511 1.402409e-08
```

5.3 Run a Friedman Test

```
friedman.test(Overshoots ~ Model | ID, data = data)
```

```
## 
##  Friedman rank sum test
## 
## data:  Overshoots and Model and ID
## Friedman chi-squared = 17.025, df = 3, p-value = 0.0006984
```

```
frdAllPairsNemenyiTest(data$Overshoots, data$Model, blocks = data$ID)
```

```
## 
##  Pairwise comparisons using Nemenyi-Wilcoxon-Wilcox all-pairs test for a two-way b
alanced complete block design
```

```
## data: y, groups and blocks
```

```
##         C-SGT  O-CIIL UC-CIIL
## O-CIIL  0.0655 -      -      
## UC-CIIL 0.0089 0.9030 -      
## W-SGT   0.9991 0.0458 0.0056
```

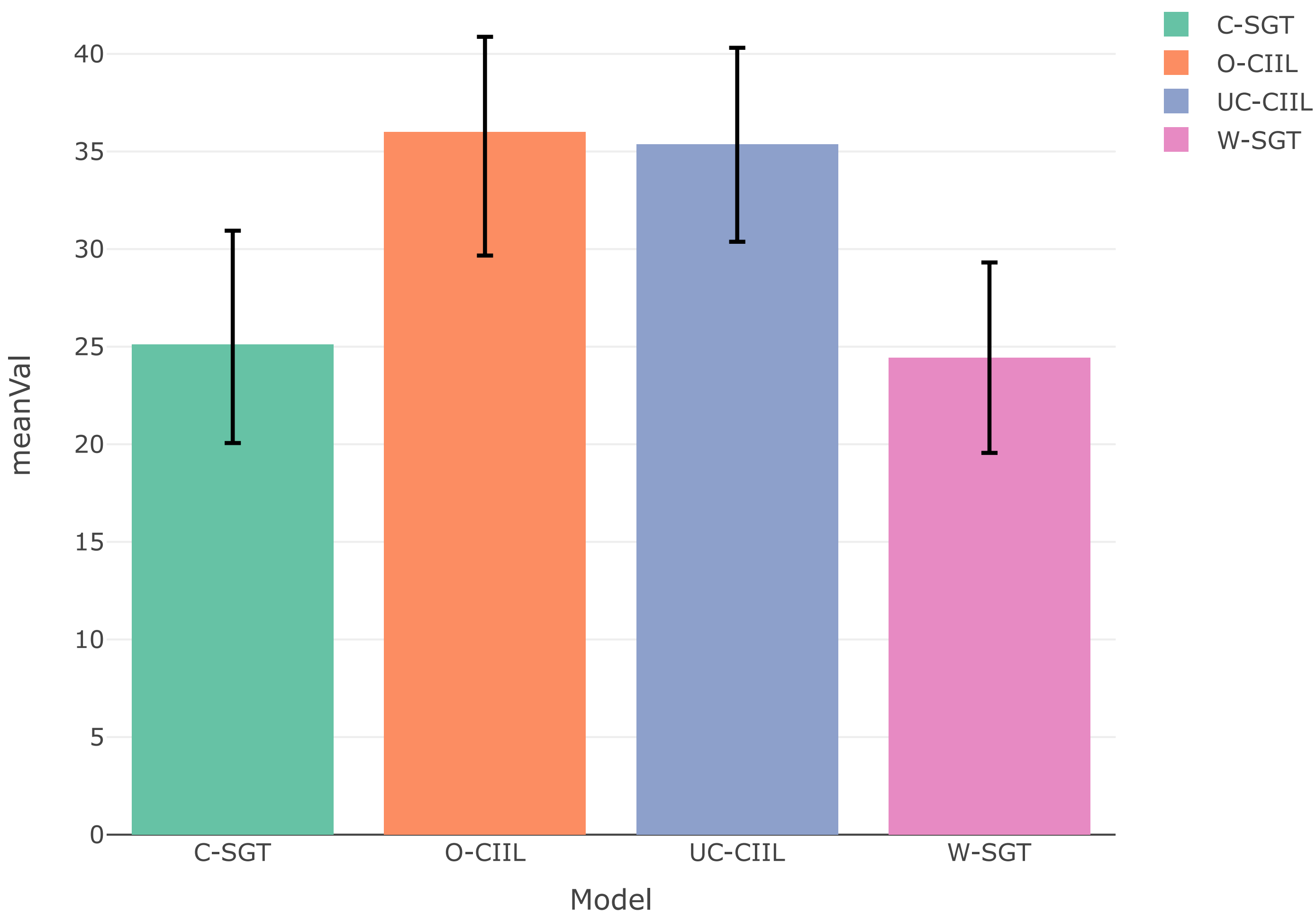
```
## 
## P value adjustment method: single-step
```

6 Evaluate Trials

6.1 Plot trials

Represent a bar chart with 95% CI for each model and block

```
create_bar_plot(data, "Trials")
```



6.2 Check for normality

```
check_for_normality(data, "Trials")
```

```
##   W_statistic    p_value
## 1   0.9668082 0.08243196
```

6.3 Run a Friedman Test

```
run_anova(data, "Trials")
```

```
##
## Univariate Type III Repeated-Measures ANOVA Assuming Sphericity
##
##              Sum Sq num Df Error SS den Df F value    Pr(>F)
## (Intercept)  58504      1   6374.2     15 137.672 5.873e-09 ***
## Model         1910      3   1131.2     45  25.328 9.463e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Mauchly Tests for Sphericity
##
##       Test statistic p-value
## Model        0.61319 0.24407
##
##
## Greenhouse-Geisser and Huynh-Feldt Corrections
##  for Departure from Sphericity
##
##        GG eps Pr(>F[GG])
## Model 0.74304  9.199e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##          HF eps   Pr(>F[HF])
## Model 0.8787179 8.178381e-09
```

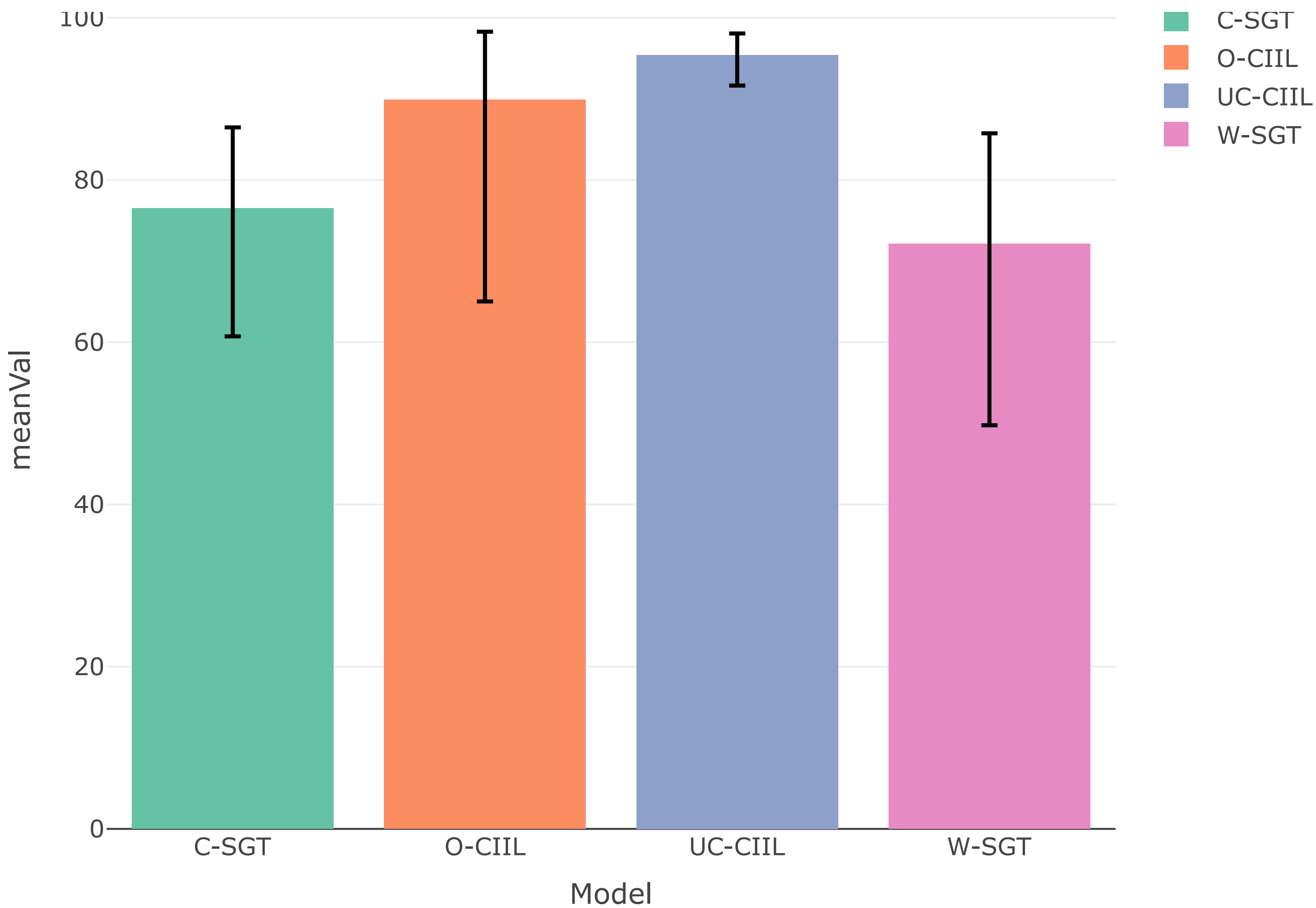
```
##  contrast         estimate   SE df t.ratio p.value
##  C.SGT - O.CIIL    -10.875 2.21 15  -4.918  0.0011
##  C.SGT - UC.CIIL   -10.250 1.89 15  -5.431  0.0004
##  C.SGT - W.SGT       0.688 1.31 15   0.524  1.0000
##  O.CIIL - UC.CIIL    0.625 1.51 15   0.414  1.0000
##  O.CIIL - W.SGT     11.562 1.89 15   6.130  0.0001
##  UC.CIIL - W.SGT    10.938 1.68 15   6.493  0.0001
##
## P value adjustment: bonferroni method for 6 tests
```

7 Evaluate CompletionRate

7.1 Plot trials

Represent a bar chart with 95% CI for each model and block

```
create_bar_plot(data, "CompletionRate")
```



7.2 Check for normality

```
check_for_normality(data, "CompletionRate")
```

```
##   W_statistic      p_value
## 1   0.6561094 5.696399e-11
```

7.3 Run a Friedman Test

```
friedman.test(CompletionRate ~ Model | ID, data = data)
```

```
##
##  Friedman rank sum test
##
## data:  CompletionRate and Model and ID
## Friedman chi-squared = 20.613, df = 3, p-value = 0.0001267
```

```
frdAllPairsNemenyiTest(data$CompletionRate, data$Model, blocks = data$ID)
```

```
##
##  Pairwise comparisons using Nemenyi-Wilcoxon-Wilcox all-pairs test for a two-way b
alanced complete block design
```

```
## data: y, groups and blocks
```

```
##        C-SGT  O-CIIL UC-CIIL
## O-CIIL  0.1685 -      -
## UC-CIIL 0.0210 0.8443 -
## W-SGT   0.9472 0.0458 0.0035
```

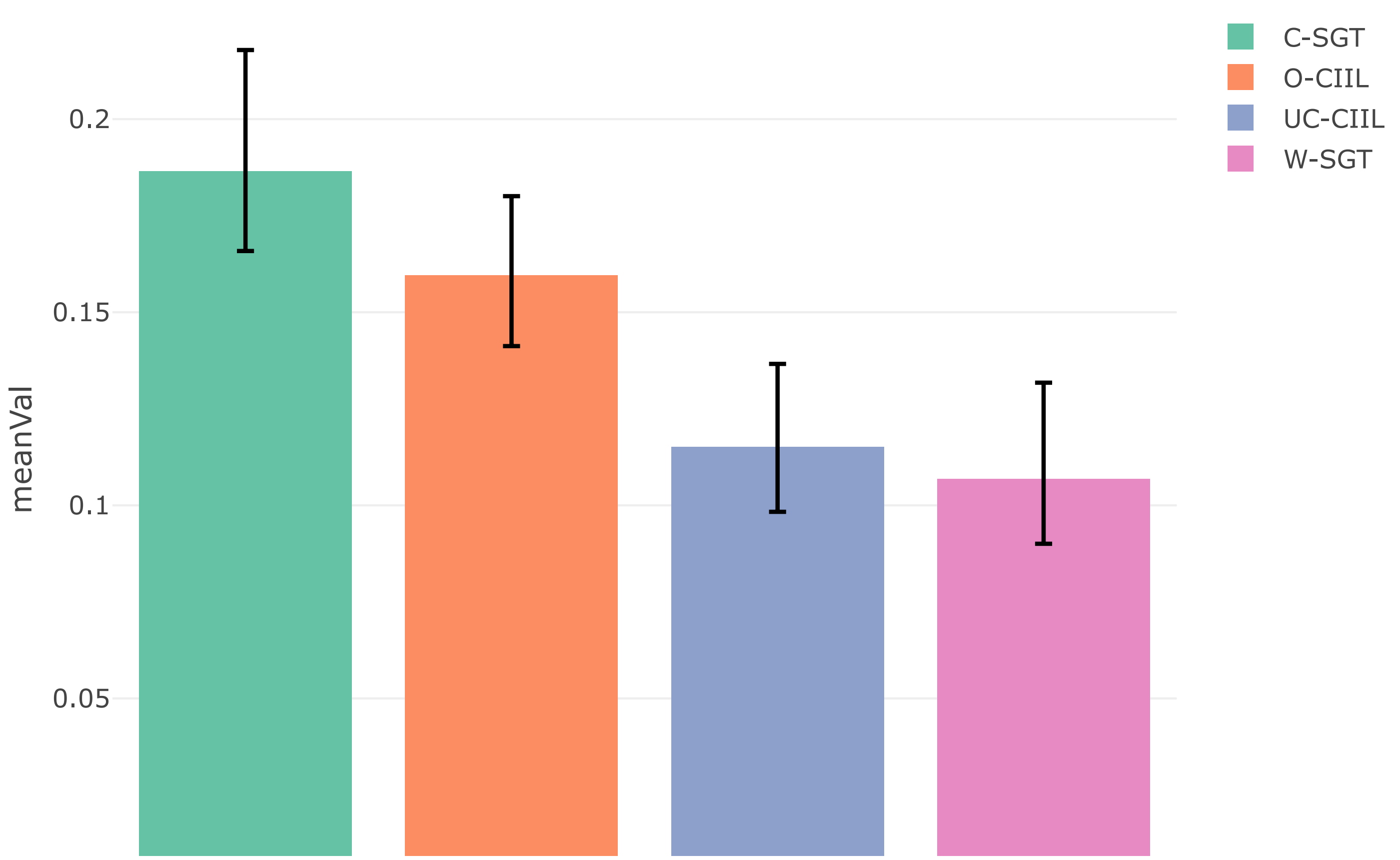
```
##
## P value adjustment method: single-step
```

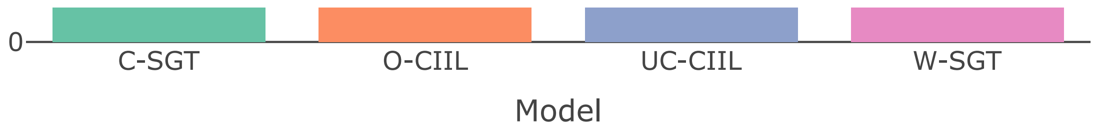
8 Evaluate ActionInterference

8.1 Plot trials

Represent a bar chart with 95% CI for each model and block

```
create_bar_plot(data, "ActionInterference")
```





8.2 Check for normality

```
check_for_normality(data, "ActionInterference")
```

```
##   W_statistic    p_value
## 1   0.9600903 0.03691513
```

8.3 Run a Friedman Test

```
friedman.test(ActionInterference ~ Model | ID, data = data)
```

```
## 
##  Friedman rank sum test
## 
## data:  ActionInterference and Model and ID
## Friedman chi-squared = 29.025, df = 3, p-value = 2.213e-06
```

```
frdAllPairsNemenyiTest(data$ActionInterference, data$Model, blocks = data$ID)
```

```
## 
##  Pairwise comparisons using Nemenyi-Wilcoxon-Wilcox all-pairs test for a two-way b
alanced complete block design
```

```
## data: y, groups and blocks
```

```
##         C-SGT   O-CIIL  UC-CIIL
## O-CIIL  0.60619 -       -      
## UC-CIIL 0.00023 0.02102 -      
## W-SGT   3.7e-05 0.00560 0.97662
```

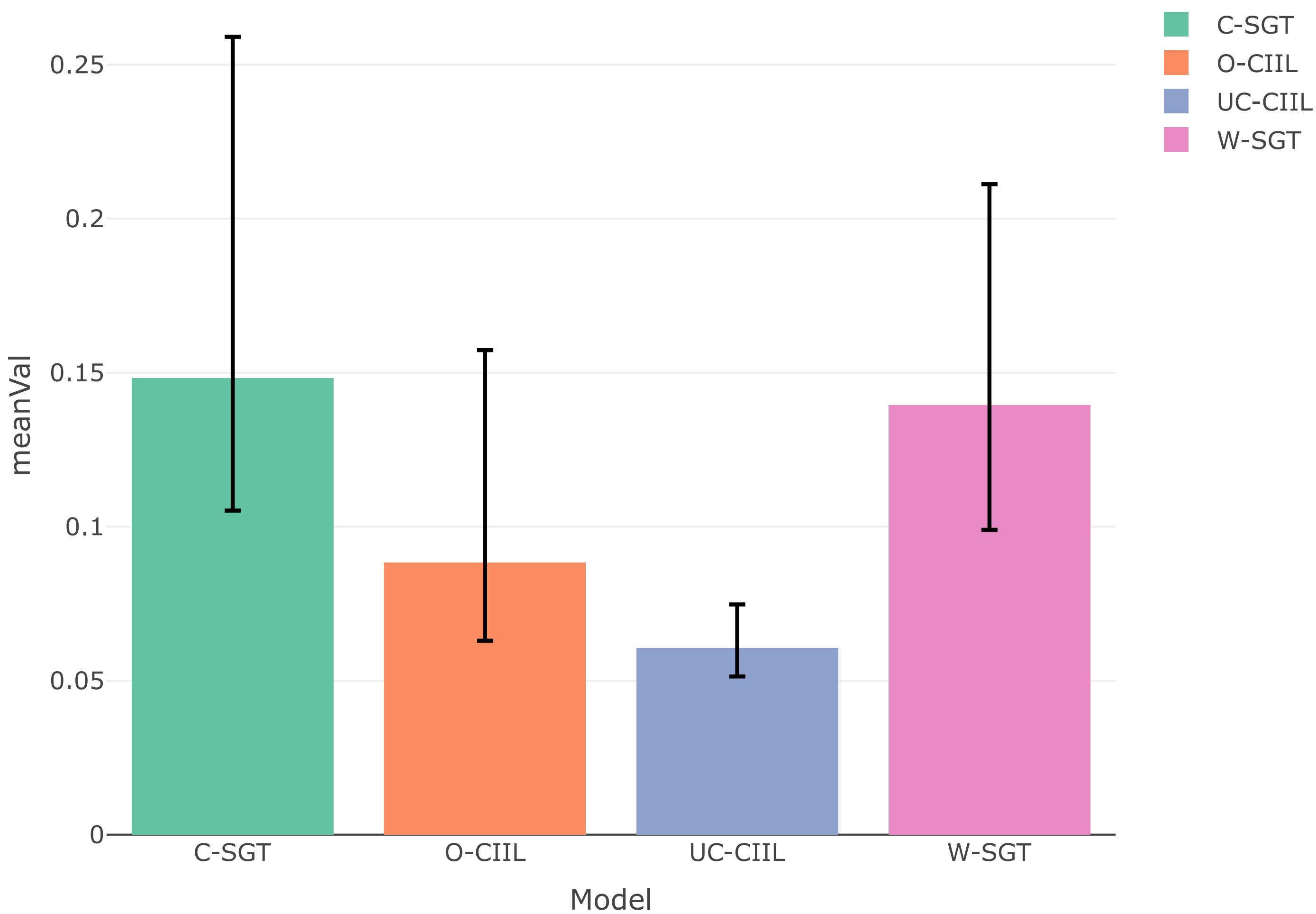
```
## 
## P value adjustment method: single-step
```

9 Evaluate Drift

9.1 Plot trials

Represent a bar chart with 95% CI for each model and block

```
create_bar_plot(data, "Drift")
```



9.2 Check for normality

```
check_for_normality(data, "Drift")
```

```
##   W_statistic      p_value
## 1   0.6476207 4.016901e-11
```

9.3 Run a Friedman Test

```
friedman.test(Drift ~ Model | ID, data = data)
```

```
##
##  Friedman rank sum test
##
## data:  Drift and Model and ID
## Friedman chi-squared = 27.675, df = 3, p-value = 4.25e-06
```

```
frdAllPairsNemenyiTest(data$Drift, data$Model, blocks = data$ID)
```

```
##
##  Pairwise comparisons using Nemenyi-Wilcoxon-Wilcox all-pairs test for a two-way b
alanced complete block design
```

```
## data: y, groups and blocks
```

```
##         C-SGT   O-CIIL UC-CIIL
## O-CIIL  0.168   -      -
## UC-CIIL 3.7e-05 0.066  -
## W-SGT   1.000   0.168  3.7e-05
```

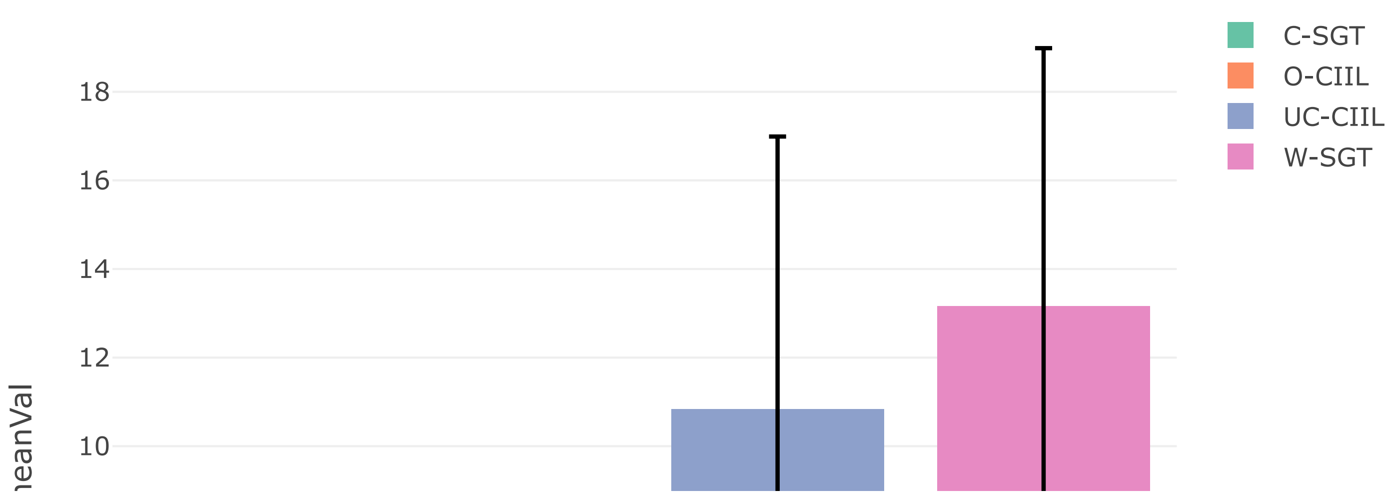
```
##
## P value adjustment method: single-step
```

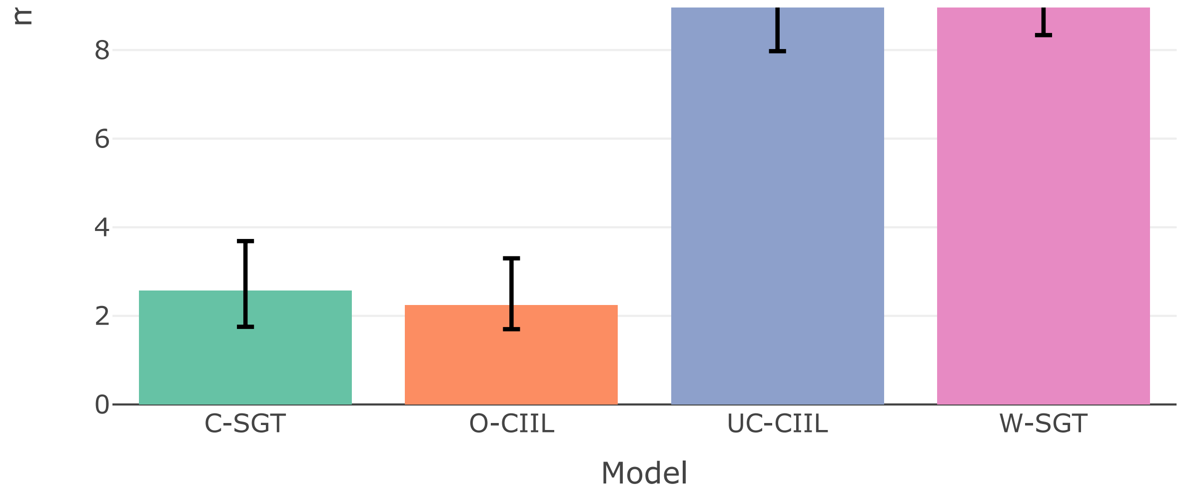
10 Evaluate CostOfSimultaneity

10.1 Plot trials

Represent a bar chart with 95% CI for each model and block

```
create_bar_plot(data, "CostOfSimultaneity")
```





10.2 Check for normality

```
check_for_normality(data, "CostOfSimultaneity")
```

```
##   W_statistic      p_value
## 1   0.7430927 3.007962e-09
```

10.3 Run a Friedman Test

```
friedman.test(CostOfSimultaneity ~ Model | ID, data = data)
```

```
##
##  Friedman rank sum test
##
## data:  CostOfSimultaneity and Model and ID
## Friedman chi-squared = 36.225, df = 3, p-value = 6.711e-08
```

```
frdAllPairsNemenyiTest(data$CostOfSimultaneity, data$Model, blocks = data$ID)
```

```
##
##  Pairwise comparisons using Nemenyi-Wilcoxon-Wilcox all-pairs test for a two-way b
alanced complete block design
```

```
## data: y, groups and blocks
```

```
##         C-SGT   O-CIIL  UC-CIIL
## O-CIIL  0.97662 -       -
## UC-CIIL 0.00023 3.7e-05 -
## W-SGT   0.00042 6.9e-05 0.99908
```

```
##
## P value adjustment method: single-step
```