Supplement 1

Flattening of the pain intensity relationship

Peter Kamerman

Last knitted: 26 June 2020

Contents

---

1 Introduction

The use of pain intensity cut-offs for study inclusion in clinical trial (typically a lower cut-off of 4/10 on an 11-point numerical pain rating scale) has two consequences. Firstly, the cut-off artificially raises the baseline mean pain score of the sample above that of the mean score of the population from which the sample was drawn. Secondly, unless the correlation between two sequential measurements is 1, there should be a "flattening" of the relationship between the first and subsequent measurements. This "flattening" means that the cut-off has a disproportionate effect on the mean baseline pain intensity compared to subsequent measurements.

This script demonstrates the "flattening" of the relationship between two sequential pain measurements in a hypothetical placebo group in a clinical trial for the management of neuropathic pain, in the absence of any cut-off. For both the baseline and subsequent measurement (V1 and V2, respectively), the mean pain intensity was set at 6.2 (SD 1.7); typical baseline center and spread values for placebo groups in placebo-controlled trials for the management of neuropathic pain when there is no cut-off inclusion criterion (see Supplement 2). By using the same mean (SD) pain intensity for both measurements I have assumed that the pain is, on average, stable over the time period between the two measurements.

For the purposes of this simulation, I examined the relationship between the two measurements (V1 and V2) over a range of correlation values, namely: r = 1.0, 0.8, 0.5, and 0.2.

Also, I constructed the models under two conditions: *constrained* and *unconstrained* pain intensity ratings. For the *unconstrained* models I took the raw values obtained through sampling from a bivariate normal distribution. In the *constrained* model, values from the unconstrained simulation were bounded to the range of the numerical pain rating scale (NRS), namely, 0 to 10. Specifically, in the case of measurement 1 (V1), values were constrained between 1 and 10 to model pain ratings of participants entering a placebo-controlled clinical trial, who may be expected to have at least some pain (minimum of 1/10 on an NRS) on study entry. In comparison, measurement 2 (V2) values were constrained between 0 and 10 because pain intensity could conceivably take on any value between 0 and 10 on an NRS during later measurements.

For reproducibility, I set the random seed to *"2019"* for all random sampling procedures.

Note: Because of random sampling the mean (SD) of the samples differ slightly from the population mean (SD).

---

2 Generate 2x2 covariance matrices

Generate covariance matrices using an SD of 1.7 and correlation of r = 1.0, 0.8, 0.5, and 0.2.

```r
# Generate 2*2 correlation matrices
cor_10 <- matrix(c(1, 1, 1, 1), ncol = 2)
cor_08 <- matrix(c(1, 0.8, 0.8, 1), ncol = 2)
cor_05 <- matrix(c(1, 0.5, 0.5, 1), ncol = 2)
cor_02 <- matrix(c(1, 0.2, 0.2, 1), ncol = 2)

# Set the standard deviation
std <- c(1.7, 1.7)

# Generate 2*2 covariance matrices using the correlation matrices and SD
cov_10 <- cor2cov(cor.mat = cor_10,
                  sd = std)
cov_10
```

```
##      [,1] [,2]
## [1,] 2.89 2.89
## [2,] 2.89 2.89
```

```r
cov_08 <- cor2cov(cor.mat = cor_08,
                  sd = std)
cov_08
```

```
##       [,1]  [,2]
## [1,] 2.890 2.312
## [2,] 2.312 2.890
```

```r
cov_05 <- cor2cov(cor.mat = cor_05,
                  sd = std)
cov_05
```

```
##       [,1]  [,2]
## [1,] 2.890 1.445
## [2,] 1.445 2.890
```

```
cov_02 <- cor2cov(cor.mat = cor_02,
                  sd = std)
cov_02
```

```
##       [,1]  [,2]
## [1,] 2.890 0.578
## [2,] 0.578 2.890
```

---

3 Correlation: r = 1.0

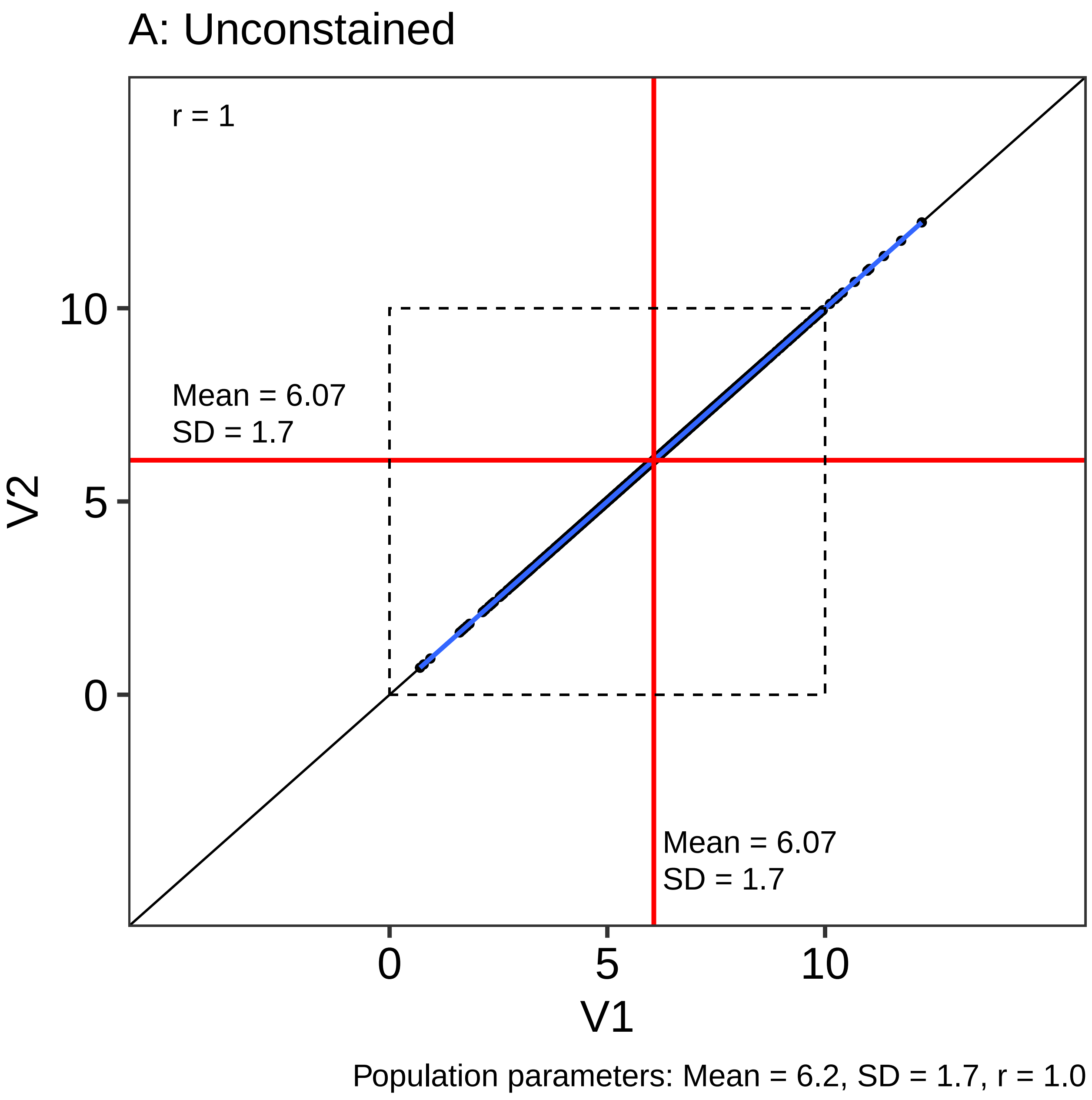
3.1 Unconstrained data

```
# Set the random seed for reproducibility
set.seed(2019)

# Generate the data (1000 pairs from a bivariate normal distribution)
cor_10.base <- as.data.frame(mvrnorm(n = 1000, mu = c(6.2, 6.2), Sigma = cov_10))

# Plot unconstrained data
ggplot(data = cor_10.base) +
    aes(x = V1, y = V2) +
    geom_point() +
    geom_abline(slope = 1, intercept = 0) +
    geom_smooth(method = 'lm',
                se = FALSE) +
    geom_hline(yintercept = mean(cor_10.base$V2),
               colour = 'red', size = 1) +
    geom_vline(xintercept = mean(cor_10.base$V1),
               colour = 'red', size = 1) +
    geom_rect(ymin = 0, ymax = 10,
              xmin = 0, xmax = 10,
              colour = '#000000',
              alpha = 0,
              linetype = 2) +
    annotate(geom = 'text', x = -5, y = 15, hjust = 0, size = 5,
             label = str_glue("r = {round(cor(cor_10.base$V1,
                             cor_10.base$V2), 2)}")) +
    annotate(geom = 'text', x = -5, y = mean(cor_10.base$V2) + 1.7,
             hjust = 0, size = 5,
             label = str_glue("Mean = {round(mean(cor_10.base$V2), 2)}")) +
    annotate(geom = 'text', x = -5, y = mean(cor_10.base$V2) + 0.75,
             hjust = 0, size = 5,
             label = str_glue("SD = {round(sd(cor_10.base$V2),2)}")) +
    annotate(geom = 'text', x = mean(cor_10.base$V1) + 0.2, y = -3.8,
             hjust = 0, size = 5,
             label = str_glue("Mean = {round(mean(cor_10.base$V1), 2)}")) +
    annotate(geom = 'text', x = mean(cor_10.base$V1) + 0.2, y = -4.75,
              hjust = 0, size = 5,
              label = str_glue("SD = {round(sd(cor_10.base$V1), 2)}")) +
    labs(title = 'A: Unconstained',
         caption = 'Population parameters: Mean = 6.2, SD = 1.7, r = 1.0') +
    scale_y_continuous(limits = c(-5, 15),
                       breaks = c(0, 5, 10)) +
    scale_x_continuous(limits = c(-5, 15),
                       breaks = c(0, 5, 10)) +
    theme(plot.caption = element_text(size = 14))
```

```
## `geom_smooth()` using formula 'y ~ x'
```



```
# Linear regression
summary(lm(V2 ~ V1, data = cor_10.base))
```

```
## Warning in summary.lm(lm(V2 ~ V1, data = cor_10.base)): essentially perfect fit:
## summary may be unreliable
```

```
##
## Call:
## lm(formula = V2 ~ V1, data = cor_10.base)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -5.232e-14  3.000e-18  4.900e-17  9.800e-17  1.548e-15
##
## Coefficients:
##               Estimate Std. Error    t value Pr(>|t|)
## (Intercept) -5.393e-15  1.945e-16 -2.773e+01   <2e-16 ***
## V1           1.000e+00  3.087e-17  3.240e+16   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.662e-15 on 998 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 1.05e+33 on 1 and 998 DF,  p-value: < 2.2e-16
```

```r
Confint(lm(V2 ~ V1, data = cor_10.base))
```

```
## Warning in summary.lm(object, ...): essentially perfect fit: summary may be
## unreliable
```

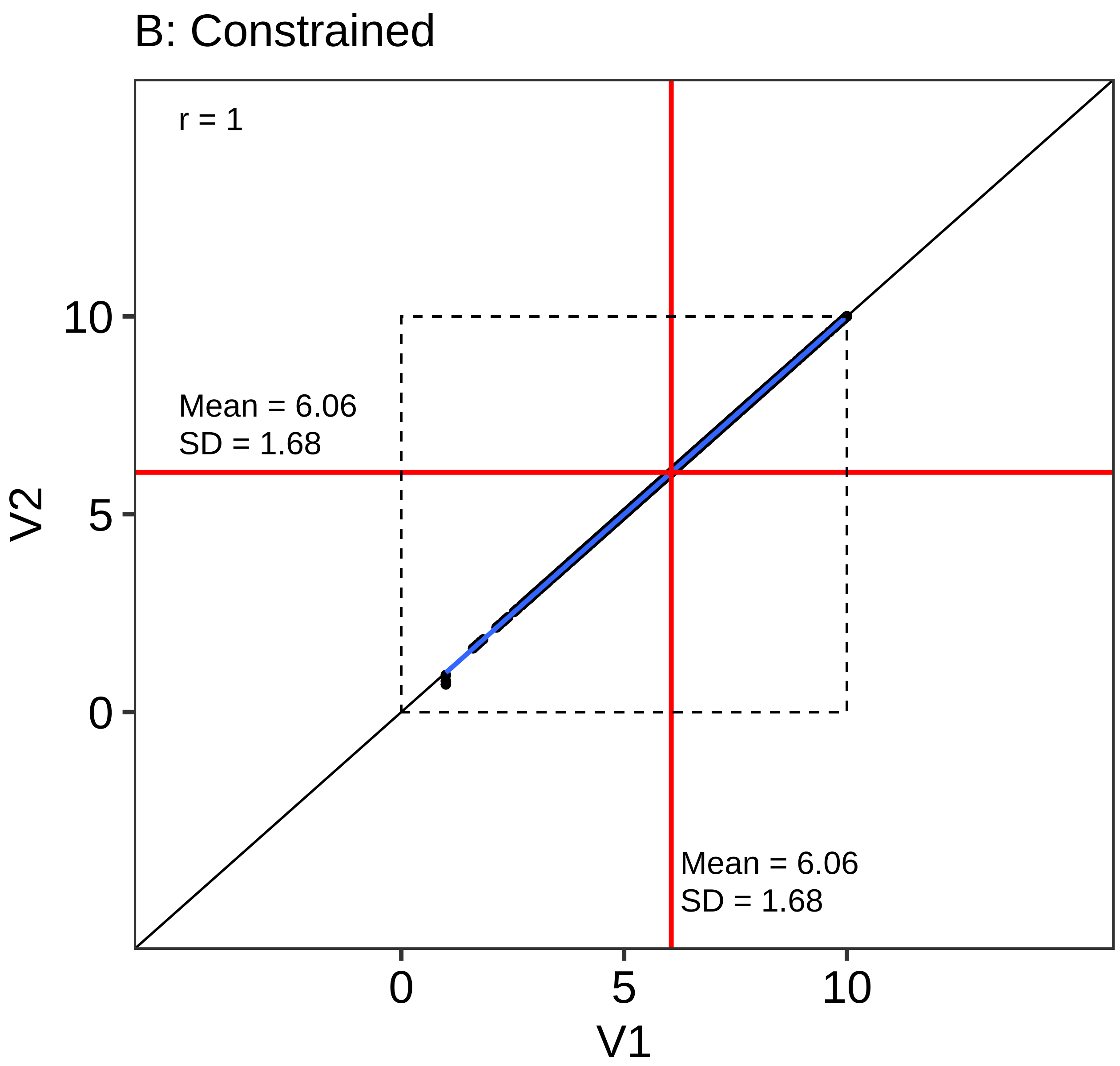
```
##                 Estimate         2.5 %        97.5 %
## (Intercept) -5.39264e-15 -5.774318e-15 -5.010962e-15
## V1           1.00000e+00  1.000000e+00  1.000000e+00
```

3.2 Constrained data

```r
# Constrain data
cor_10.constrained <- cor_10.base %>%
    mutate(V1 = case_when(
                V1 < 1 ~ 1,
                V1 > 10 ~ 10,
                TRUE ~ V1)) %>%
    mutate(V2 = case_when(
                V2 < 0 ~ 0,
                V2 > 10 ~ 10,
                TRUE ~ V2)) %>%
    mutate(group = 'No threshold')

# Plot constrained data
ggplot(data = cor_10.constrained) +
    aes(x = V1, y = V2) +
    geom_point() +
    geom_abline(slope = 1, intercept = 0) +
    geom_smooth(method = 'lm',
                 se = FALSE) +
    geom_hline(yintercept = mean(cor_10.constrained$V2),
                colour = 'red', size = 1) +
    geom_vline(xintercept = mean(cor_10.constrained$V1),
               colour = 'red', size = 1) +
    geom_rect(ymin = 0, ymax = 10,
              xmin = 0, xmax = 10,
              colour = '#000000',
              alpha = 0,
              linetype = 2) +
    annotate(geom = 'text', x = -5, y = 15, hjust = 0, size = 5,
             label = str_glue("r = {round(cor(cor_10.constrained$V1,
                                           cor_10.constrained$V2), 2)}")) +
    annotate(geom = 'text', x = -5, y = mean(cor_10.constrained$V2) + 1.7,
             hjust = 0, size = 5,
             label = str_glue("Mean = {round(mean(cor_10.constrained$V2), 2)}")) +
    annotate(geom = 'text', x = -5, y = mean(cor_10.constrained$V2) + 0.75,
             hjust = 0, size = 5,
             label = str_glue("SD = {round(sd(cor_10.constrained$V2),2)}")) +
    annotate(geom = 'text', x = mean(cor_10.constrained$V1) + 0.2, y = -3.8,
             hjust = 0, size = 5,
             label = str_glue("Mean = {round(mean(cor_10.constrained$V1), 2)}")) +
    annotate(geom = 'text', x = mean(cor_10.constrained$V1) + 0.2, y = -4.75,
             hjust = 0, size = 5,
             label = str_glue("SD = {round(sd(cor_10.constrained$V1), 2)}")) +
    labs(title = 'B: Constrained',
         caption = 'Population parameters: Mean = 6.2, SD = 1.7, r = 1.0') +
    scale_y_continuous(limits = c(-5, 15),
                       breaks = c(0, 5, 10)) +
    scale_x_continuous(limits = c(-5, 15),
                       breaks = c(0, 5, 10)) +
    theme(plot.caption = element_text(size = 14))
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Population parameters: Mean = 6.2, SD = 1.7, r = 1.0

```r
# Linear regression
summary(lm(V2 ~ V1, data = cor_10.constrained))
```

```
##
## Call:
## lm(formula = V2 ~ V1, data = cor_10.constrained)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.295454 -0.000643  0.000662  0.001717  0.005245
##
## Coefficients:
##               Estimate Std. Error  t value Pr(>|t|)
## (Intercept) -0.0069328  0.0013969   -4.963 8.16e-07 ***
## V1           1.0010482  0.0002222 4504.746  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01178 on 998 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 2.029e+07 on 1 and 998 DF,  p-value: < 2.2e-16
```

```
Confint(lm(V2 ~ V1, data = cor_10.constrained))
```

```
##                 Estimate        2.5 %       97.5 %
## (Intercept) -0.006932785 -0.009674038 -0.004191532
## V1           1.001048195  1.000612122  1.001484269
```

---

4 Correlation: r = 0.8

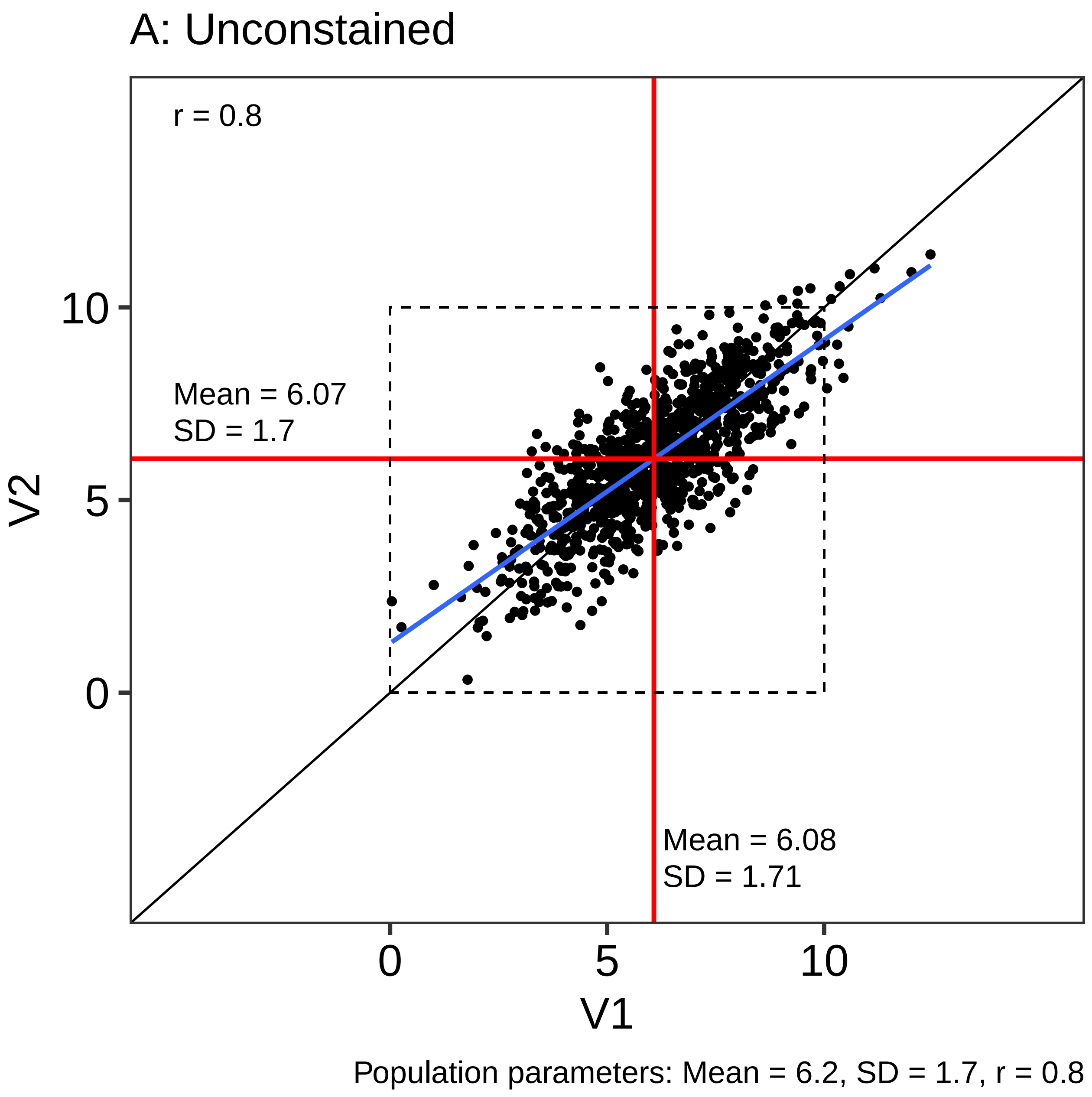
4.1 Unconstrained data

```
# Set the random seed for reproducibility
set.seed(2019)

# Generate the data (1000 pairs from a bivariate normal distribution)
cor_08.base <- as.data.frame(mvrnorm(n = 1000, mu = c(6.2, 6.2), Sigma = cov_08))

# Plot unconstrained data
ggplot(data = cor_08.base) +
    aes(x = V1, y = V2) +
    geom_point() +
    geom_abline(slope = 1, intercept = 0) +
    geom_smooth(method = 'lm',
                se = FALSE) +
    geom_hline(yintercept = mean(cor_08.base$V2),
               colour = 'red', size = 1) +
    geom_vline(xintercept = mean(cor_08.base$V1),
               colour = 'red', size = 1) +
    geom_rect(ymin = 0, ymax = 10,
              xmin = 0, xmax = 10,
              colour = '#000000',
              alpha = 0,
              linetype = 2) +
    annotate(geom = 'text', x = -5, y = 15, hjust = 0, size = 5,
             label = str_glue("r = {round(cor(cor_08.base$V1,
                                cor_08.base$V2), 2)}")) +
    annotate(geom = 'text', x = -5, y = mean(cor_08.base$V2) + 1.7,
             hjust = 0, size = 5,
             label = str_glue("Mean = {round(mean(cor_08.base$V2), 2)}")) +
    annotate(geom = 'text', x = -5, y = mean(cor_08.base$V2) + 0.75,
             hjust = 0, size = 5,
             label = str_glue("SD = {round(sd(cor_08.base$V2),2)}")) +
    annotate(geom = 'text', x = mean(cor_08.base$V1) + 0.2, y = -3.8,
             hjust = 0, size = 5,
             label = str_glue("Mean = {round(mean(cor_08.base$V1), 2)}")) +
    annotate(geom = 'text', x = mean(cor_08.base$V1) + 0.2, y = -4.75,
             hjust = 0, size = 5,
             label = str_glue("SD = {round(sd(cor_08.base$V1), 2)}")) +
    labs(title = 'A: Unconstained',
         caption = 'Population parameters: Mean = 6.2, SD = 1.7, r = 0.8') +
    scale_y_continuous(limits = c(-5, 15),
                       breaks = c(0, 5, 10)) +
    scale_x_continuous(limits = c(-5, 15),
                       breaks = c(0, 5, 10)) +
    theme(plot.caption = element_text(size = 14))
```

```
## `geom_smooth()` using formula 'y ~ x'
```



```r
# Linear regression
summary(lm(V2 ~ V1, data = cor_08.base))
```

```
##
## Call:
## lm(formula = V2 ~ V1, data = cor_08.base)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.9824 -0.6632  0.0201  0.6799  3.3484
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.28292    0.12006   10.69   <2e-16 ***
## V1           0.78780    0.01901   41.43   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.03 on 998 degrees of freedom
## Multiple R-squared:  0.6323, Adjusted R-squared:  0.632
## F-statistic:  1716 on 1 and 998 DF,  p-value: < 2.2e-16
```

```r
Confint(lm(V2 ~ V1, data = cor_08.base))
```

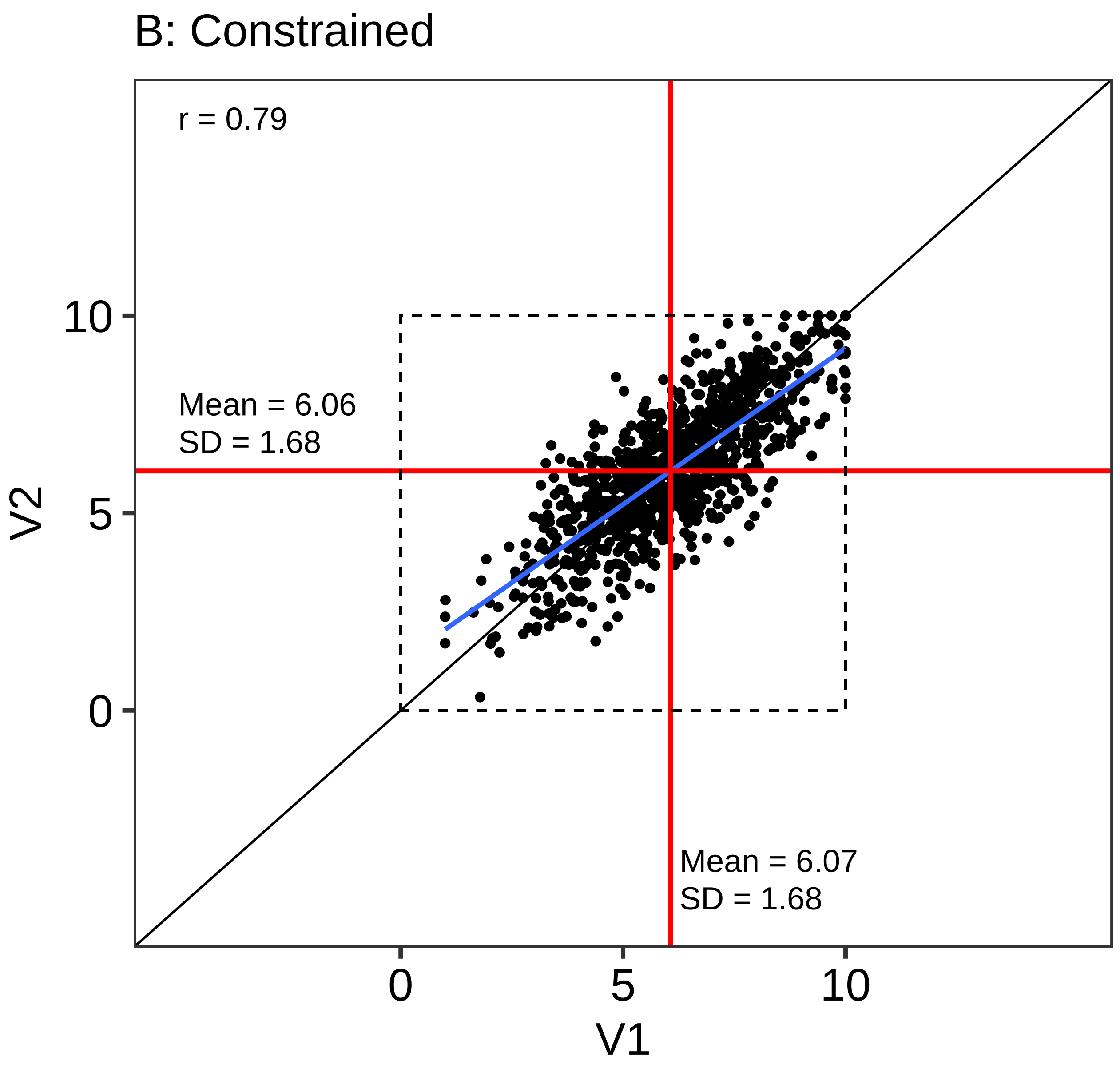
```
##              Estimate     2.5 %    97.5 %
## (Intercept) 1.2829201 1.0473188 1.5185214
## V1          0.7878005 0.7504867 0.8251143
```

4.2 Constrained data

```r
# Constrain data
cor_08.constrained <- cor_08.base %>%
    mutate(V1 = case_when(
                V1 < 1 ~ 1,
                V1 > 10 ~ 10,
                TRUE ~ V1)) %>%
    mutate(V2 = case_when(
                V2 < 0 ~ 0,
                V2 > 10 ~ 10,
                TRUE ~ V2)) %>%
    mutate(group = 'No threshold')

# Plot constrained data
ggplot(data = cor_08.constrained) +
    aes(x = V1, y = V2) +
    geom_point() +
    geom_abline(slope = 1, intercept = 0) +
    geom_smooth(method = 'lm',
                se = FALSE) +
    geom_hline(yintercept = mean(cor_08.constrained$V2),
               colour = 'red', size = 1) +
    geom_vline(xintercept = mean(cor_08.constrained$V1),
               colour = 'red', size = 1) +
    geom_rect(ymin = 0, ymax = 10,
              xmin = 0, xmax = 10,
              colour = '#000000',
              alpha = 0,
              linetype = 2) +
    annotate(geom = 'text', x = -5, y = 15, hjust = 0, size = 5,
             label = str_glue("r = {round(cor(cor_08.constrained$V1,
                                 cor_08.constrained$V2), 2)}")) +
    annotate(geom = 'text', x = -5, y = mean(cor_08.constrained$V2) + 1.7,
             hjust = 0, size = 5,
             label = str_glue("Mean = {round(mean(cor_08.constrained$V2), 2)}")) +
    annotate(geom = 'text', x = -5, y = mean(cor_08.constrained$V2) + 0.75,
             hjust = 0, size = 5,
             label = str_glue("SD = {round(sd(cor_08.constrained$V2),2)}")) +
    annotate(geom = 'text', x = mean(cor_08.constrained$V1) + 0.2, y = -3.8,
             hjust = 0, size = 5,
             label = str_glue("Mean = {round(mean(cor_08.constrained$V1), 2)}")) +
    annotate(geom = 'text', x = mean(cor_08.constrained$V1) + 0.2, y = -4.75,
             hjust = 0, size = 5,
             label = str_glue("SD = {round(sd(cor_08.constrained$V1), 2)}")) +
    labs(title = 'B: Constrained',
         caption = 'Population parameters: Mean = 6.2, SD = 1.7, r = 0.8') +
    scale_y_continuous(limits = c(-5, 15),
                       breaks = c(0, 5, 10)) +
    scale_x_continuous(limits = c(-5, 15),
                       breaks = c(0, 5, 10)) +
    theme(plot.caption = element_text(size = 14))
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Population parameters: Mean = 6.2, SD = 1.7, r = 0.8

```r
# Linear regression
summary(lm(V2 ~ V1, data = cor_08.constrained))
```

```
##
## Call:
## lm(formula = V2 ~ V1, data = cor_08.constrained)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.9756 -0.6595  0.0179  0.6843  3.3534
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.25863    0.12171   10.34   <2e-16 ***
## V1           0.79179    0.01933   40.97   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.027 on 998 degrees of freedom
## Multiple R-squared:  0.6271, Adjusted R-squared:  0.6267
## F-statistic:  1678 on 1 and 998 DF,  p-value: < 2.2e-16
```

```
Confint(lm(V2 ~ V1, data = cor_08.constrained))
```

```
##              Estimate     2.5 %    97.5 %
## (Intercept) 1.2586257 1.0197797 1.4974718
## V1          0.7917882 0.7538616 0.8297147
```

---

5 Correlation: r = 0.5

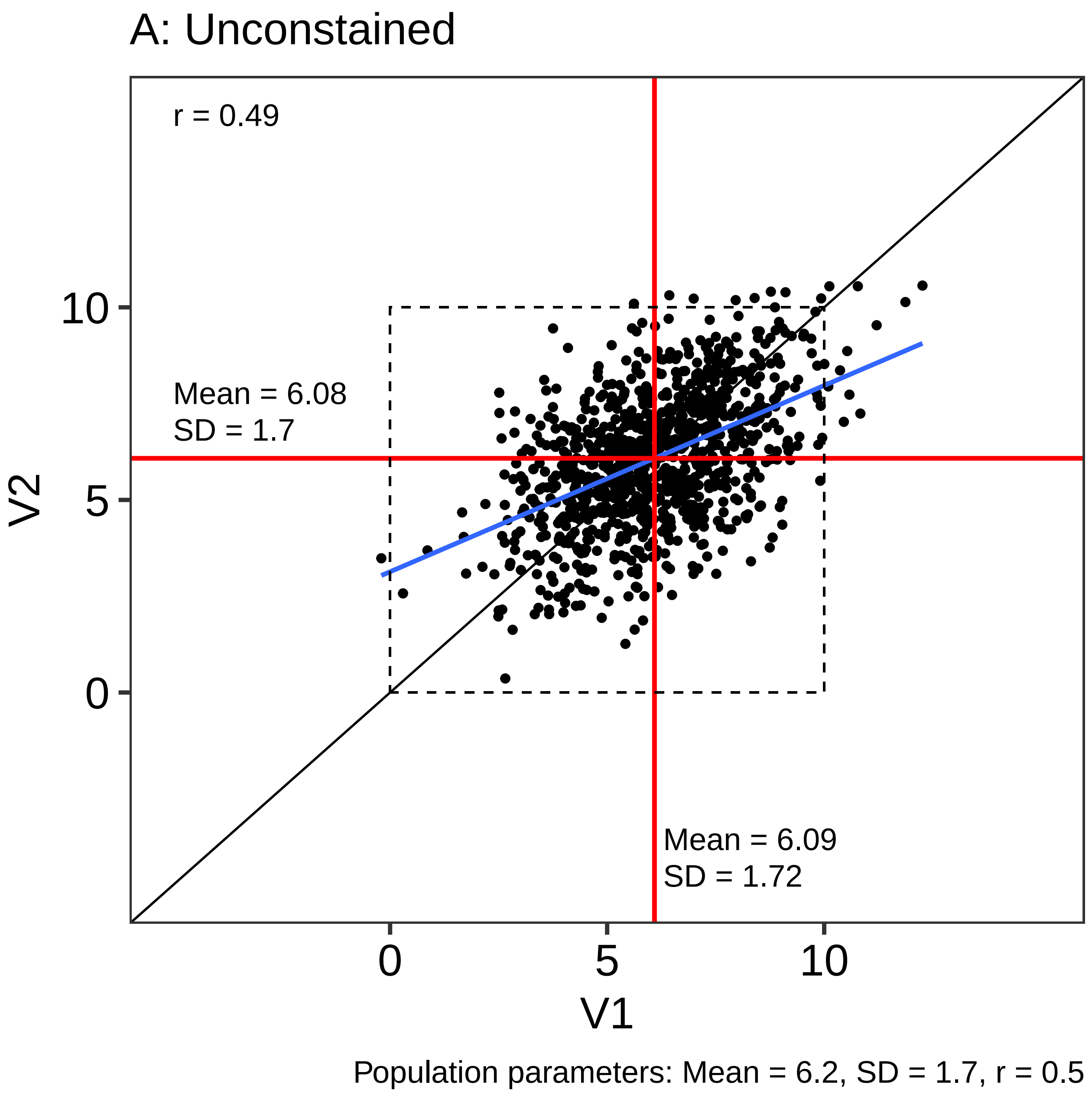
5.1 Unconstrained data

```
# Set the random seed for reproducibility
set.seed(2019)

# Generate the data (1000 pairs from a bivariate normal distribution)
cor_05.base <- as.data.frame(mvrnorm(n = 1000, mu = c(6.2, 6.2), Sigma = cov_05))

# Plot unconstrained data
ggplot(data = cor_05.base) +
    aes(x = V1, y = V2) +
    geom_point() +
    geom_abline(slope = 1, intercept = 0) +
    geom_smooth(method = 'lm',
                se = FALSE) +
    geom_hline(yintercept = mean(cor_05.base$V2),
               colour = 'red', size = 1) +
    geom_vline(xintercept = mean(cor_05.base$V1),
               colour = 'red', size = 1) +
    geom_rect(ymin = 0, ymax = 10,
              xmin = 0, xmax = 10,
              colour = '#000000',
              alpha = 0,
              linetype = 2) +
    annotate(geom = 'text', x = -5, y = 15, hjust = 0, size = 5,
             label = str_glue("r = {round(cor(cor_05.base$V1,
                                     cor_05.base$V2), 2)}")) +
    annotate(geom = 'text', x = -5, y = mean(cor_05.base$V2) + 1.7,
             hjust = 0, size = 5,
             label = str_glue("Mean = {round(mean(cor_05.base$V2), 2)}")) +
    annotate(geom = 'text', x = -5, y = mean(cor_05.base$V2) + 0.75,
             hjust = 0, size = 5,
             label = str_glue("SD = {round(sd(cor_05.base$V2),2)}")) +
    annotate(geom = 'text', x = mean(cor_05.base$V1) + 0.2, y = -3.8,
             hjust = 0, size = 5,
             label = str_glue("Mean = {round(mean(cor_05.base$V1), 2)}")) +
    annotate(geom = 'text', x = mean(cor_05.base$V1) + 0.2, y = -4.75,
             hjust = 0, size = 5,
             label = str_glue("SD = {round(sd(cor_05.base$V1), 2)}")) +
    labs(title = 'A: Unconstained',
         caption = 'Population parameters: Mean = 6.2, SD = 1.7, r = 0.5') +
    scale_y_continuous(limits = c(-5, 15),
                       breaks = c(0, 5, 10)) +
    scale_x_continuous(limits = c(-5, 15),
                       breaks = c(0, 5, 10)) +
    theme(plot.caption = element_text(size = 14))
```

```
## `geom_smooth()` using formula 'y ~ x'
```



```r
# Linear regression
summary(lm(V2 ~ V1, data = cor_05.base))
```

```
##
## Call:
## lm(formula = V2 ~ V1, data = cor_05.base)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.4932 -0.9854  0.0613  0.9882  4.5026
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.13520    0.17229   18.20   <2e-16 ***
## V1           0.48347    0.02723   17.76   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.481 on 998 degrees of freedom
## Multiple R-squared:  0.2401, Adjusted R-squared:  0.2394
## F-statistic: 315.4 on 1 and 998 DF,  p-value: < 2.2e-16
```

```r
Confint(lm(V2 ~ V1, data = cor_05.base))
```

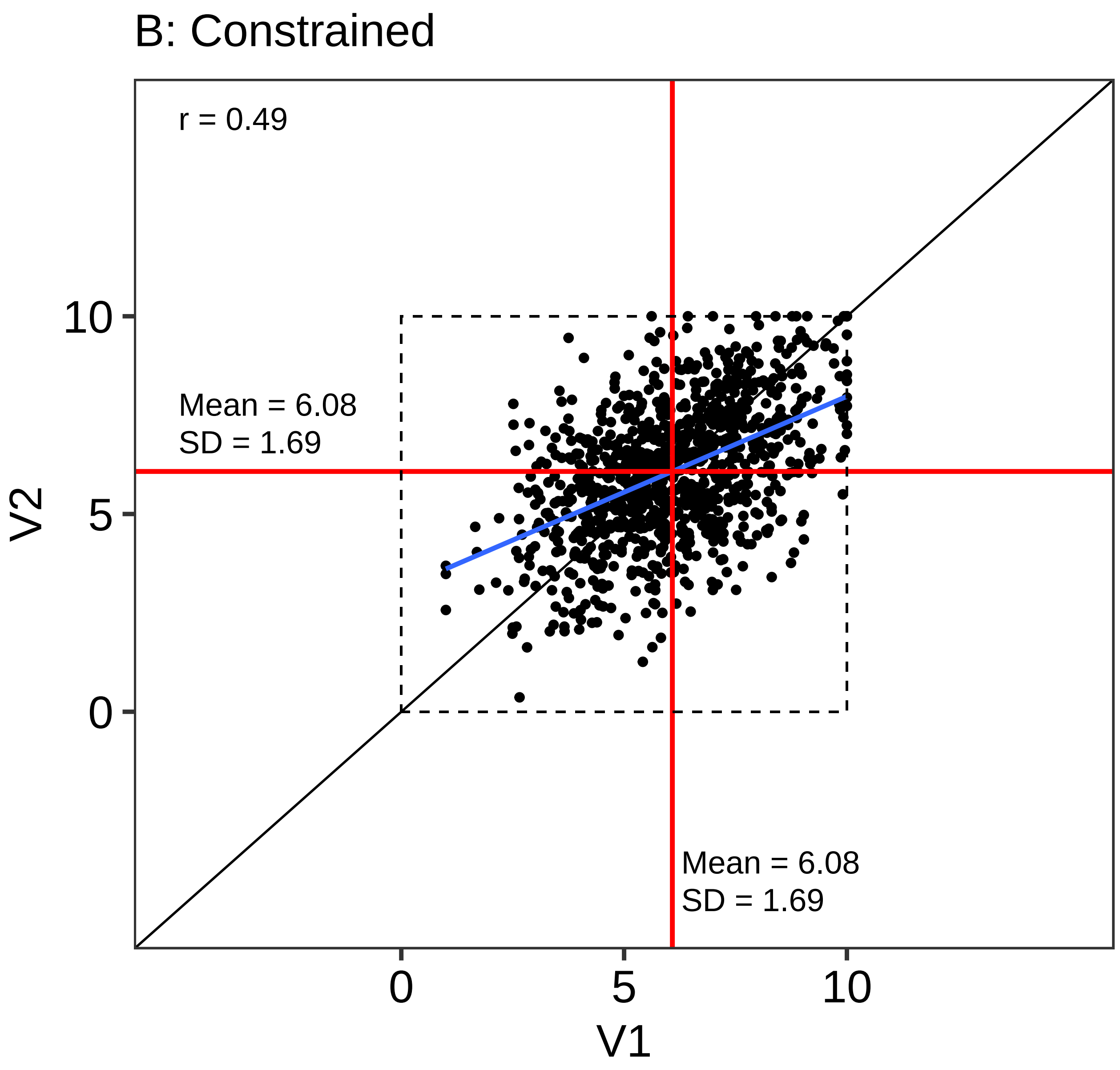
```
##              Estimate     2.5 %    97.5 %
## (Intercept) 3.1352016 2.7971072 3.4732960
## V1          0.4834716 0.4300464 0.5368969
```

5.2 Constrained data

```r
# Constrain data
cor_05.constrained <- cor_05.base %>%
    mutate(V1 = case_when(
                V1 < 1 ~ 1,
                V1 > 10 ~ 10,
                TRUE ~ V1)) %>%
    mutate(V2 = case_when(
                V2 < 0 ~ 0,
                V2 > 10 ~ 10,
                TRUE ~ V2)) %>%
    mutate(group = 'No threshold')

# Plot constrained data
ggplot(data = cor_05.constrained) +
    aes(x = V1, y = V2) +
    geom_point() +
    geom_abline(slope = 1, intercept = 0) +
    geom_smooth(method = 'lm',
                se = FALSE) +
    geom_hline(yintercept = mean(cor_05.constrained$V2),
               colour = 'red', size = 1) +
    geom_vline(xintercept = mean(cor_05.constrained$V1),
               colour = 'red', size = 1) +
    geom_rect(ymin = 0, ymax = 10,
              xmin = 0, xmax = 10,
              colour = '#000000',
              alpha = 0,
              linetype = 2) +
    annotate(geom = 'text', x = -5, y = 15, hjust = 0, size = 5,
             label = str_glue("r = {round(cor(cor_05.constrained$V1,
                                 cor_05.constrained$V2), 2)}")) +
    annotate(geom = 'text', x = -5, y = mean(cor_05.constrained$V2) + 1.7,
             hjust = 0, size = 5,
             label = str_glue("Mean = {round(mean(cor_05.constrained$V2), 2)}")) +
    annotate(geom = 'text', x = -5, y = mean(cor_05.constrained$V2) + 0.75,
             hjust = 0, size = 5,
             label = str_glue("SD = {round(sd(cor_05.constrained$V2),2)}")) +
    annotate(geom = 'text', x = mean(cor_05.constrained$V1) + 0.2, y = -3.8,
             hjust = 0, size = 5,
             label = str_glue("Mean = {round(mean(cor_05.constrained$V1), 2)}")) +
    annotate(geom = 'text', x = mean(cor_05.constrained$V1) + 0.2, y = -4.75,
             hjust = 0, size = 5,
             label = str_glue("SD = {round(sd(cor_05.constrained$V1), 2)}")) +
    labs(title = 'B: Constrained',
         caption = 'Population parameters: Mean = 6.2, SD = 1.7, r = 0.5') +
    scale_y_continuous(limits = c(-5, 15),
                       breaks = c(0, 5, 10)) +
    scale_x_continuous(limits = c(-5, 15),
                       breaks = c(0, 5, 10)) +
    theme(plot.caption = element_text(size = 14))
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Population parameters: Mean = 6.2, SD = 1.7, r = 0.5

```
# Linear regression
summary(lm(V2 ~ V1, data = cor_05.constrained))
```

```
##
## Call:
## lm(formula = V2 ~ V1, data = cor_05.constrained)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.4919 -0.9759  0.0602  0.9929  4.5058
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.12758    0.17463   17.91   <2e-16 ***
## V1            0.48465    0.02766   17.52   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.477 on 998 degrees of freedom
## Multiple R-squared:  0.2352, Adjusted R-squared:  0.2345
## F-statistic:   307 on 1 and 998 DF,  p-value: < 2.2e-16
```

```
Confint(lm(V2 ~ V1, data = cor_05.constrained))
```

```
##              Estimate     2.5 %    97.5 %
## (Intercept) 3.1275846 2.7848912 3.4702781
## V1          0.4846463 0.4303639 0.5389286
```

---

6 Correlation: r = 0.2

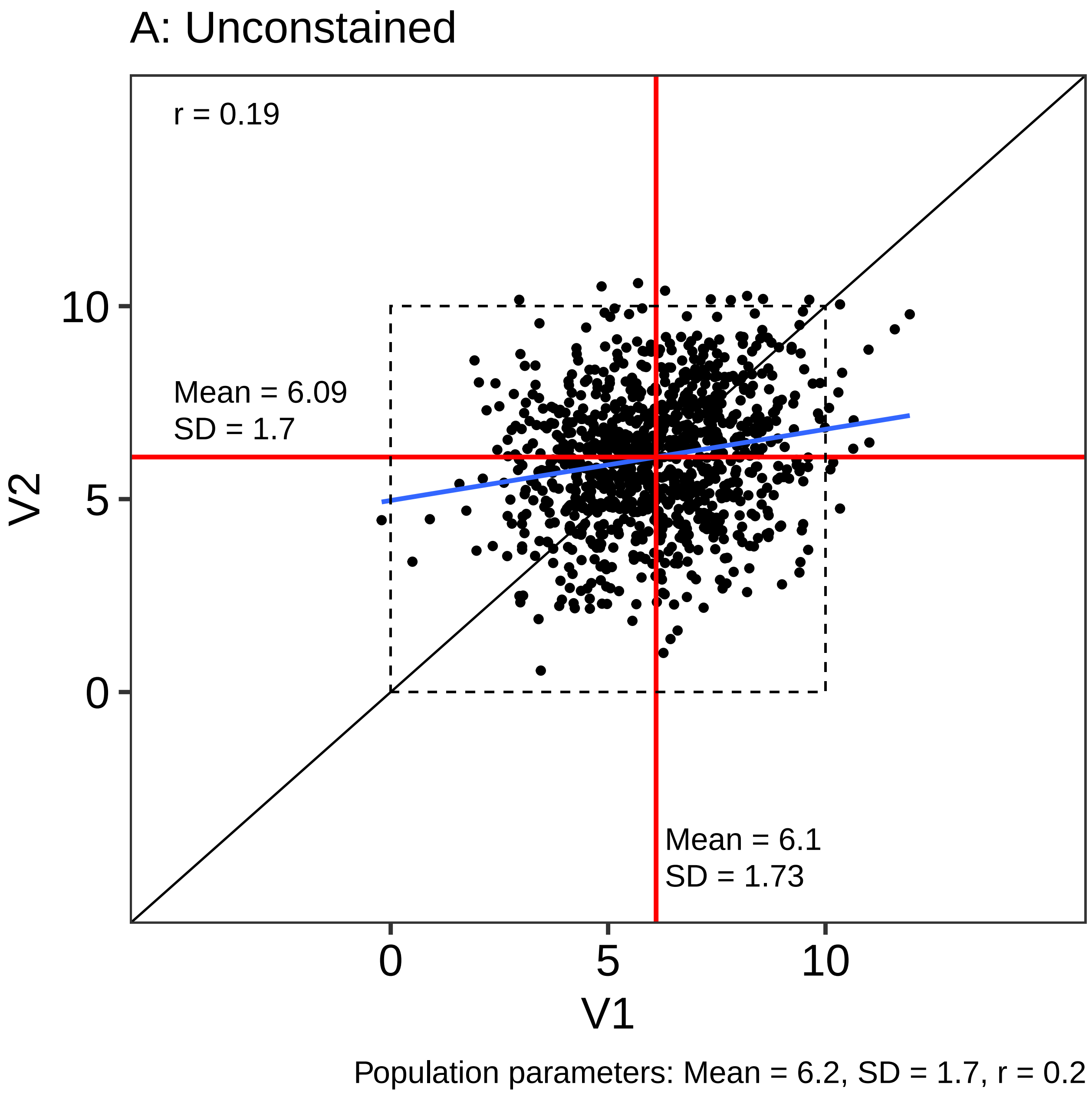
6.1 Unconstrained data

```
# Set the random seed for reproducibility
set.seed(2019)

# Generate the data
cor_02.base <- as.data.frame(mvrnorm(n = 1000, mu = c(6.2, 6.2), Sigma = cov_02))

# Plot unconstrained data
ggplot(data = cor_02.base) +
    aes(x = V1, y = V2) +
    geom_point() +
    geom_abline(slope = 1, intercept = 0) +
    geom_smooth(method = 'lm',
                se = FALSE) +
    geom_hline(yintercept = mean(cor_02.base$V2),
               colour = 'red', size = 1) +
    geom_vline(xintercept = mean(cor_02.base$V1),
               colour = 'red', size = 1) +
    geom_rect(ymin = 0, ymax = 10,
              xmin = 0, xmax = 10,
              colour = '#000000',
              alpha = 0,
              linetype = 2) +
    annotate(geom = 'text', x = -5, y = 15, hjust = 0, size = 5,
             label = str_glue("r = {round(cor(cor_02.base$V1,
                                  cor_02.base$V2), 2)}")) +
    annotate(geom = 'text', x = -5, y = mean(cor_02.base$V2) + 1.7,
             hjust = 0, size = 5,
             label = str_glue("Mean = {round(mean(cor_02.base$V2), 2)}")) +
    annotate(geom = 'text', x = -5, y = mean(cor_02.base$V2) + 0.75,
             hjust = 0, size = 5,
             label = str_glue("SD = {round(sd(cor_02.base$V2),2)}")) +
    annotate(geom = 'text', x = mean(cor_02.base$V1) + 0.2, y = -3.8,
             hjust = 0, size = 5,
             label = str_glue("Mean = {round(mean(cor_02.base$V1), 2)}")) +
    annotate(geom = 'text', x = mean(cor_02.base$V1) + 0.2, y = -4.75,
             hjust = 0, size = 5,
             label = str_glue("SD = {round(sd(cor_02.base$V1), 2)}")) +
    labs(title = 'A: Unconstained',
         caption = 'Population parameters: Mean = 6.2, SD = 1.7, r = 0.2') +
    scale_y_continuous(limits = c(-5, 15),
                       breaks = c(0, 5, 10)) +
    scale_x_continuous(limits = c(-5, 15),
                       breaks = c(0, 5, 10)) +
    theme(plot.caption = element_text(size = 14))
```

```
## `geom_smooth()` using formula 'y ~ x'
```



```r
# Linear regression
summary(lm(V2 ~ V1, data = cor_02.base))
```

```
##
## Call:
## lm(formula = V2 ~ V1, data = cor_02.base)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.1078 -1.0744  0.0558  1.1482  4.6550
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.96474    0.19426  25.557  < 2e-16 ***
## V1           0.18442    0.03063   6.021 2.43e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.67 on 998 degrees of freedom
## Multiple R-squared:  0.03506,    Adjusted R-squared:  0.03409
## F-statistic: 36.26 on 1 and 998 DF,  p-value: 2.427e-09
```

```r
Confint(lm(V2 ~ V1, data = cor_02.base))
```

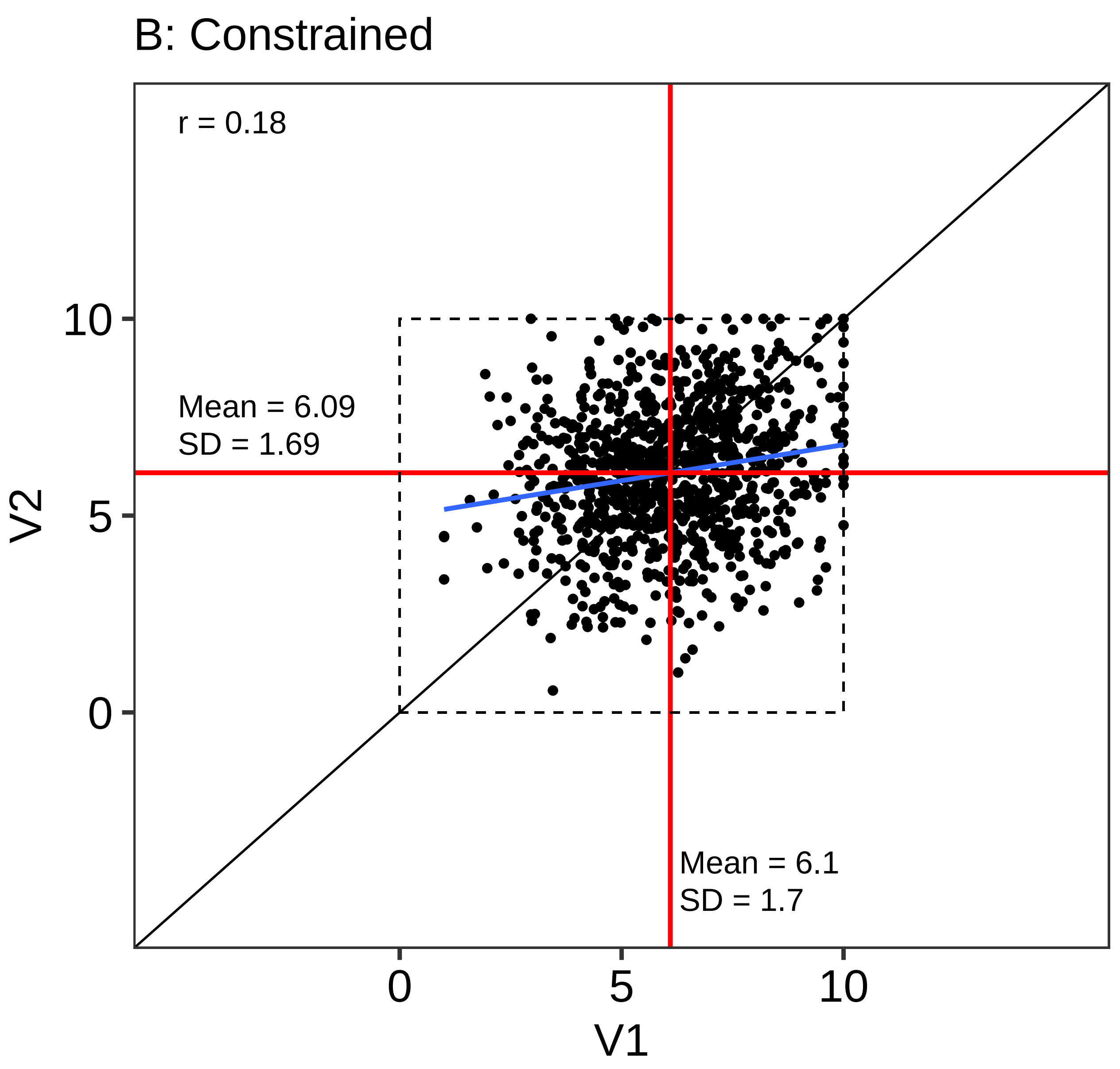
```
##              Estimate     2.5 %    97.5 %
## (Intercept) 4.9647355 4.5835301 5.3459409
## V1          0.1844196 0.1243185 0.2445207
```

6.2 Constrained data

```r
# Constrain data
cor_02.constrained <- cor_02.base %>%
    mutate(V1 = case_when(
                V1 < 1 ~ 1,
                V1 > 10 ~ 10,
                TRUE ~ V1)) %>%
    mutate(V2 = case_when(
                V2 < 0 ~ 0,
                V2 > 10 ~ 10,
                TRUE ~ V2)) %>%
    mutate(group = 'No threshold')

# Plot constrained data
ggplot(data = cor_02.constrained) +
    aes(x = V1, y = V2) +
    geom_point() +
    geom_abline(slope = 1, intercept = 0) +
    geom_smooth(method = 'lm',
                 se = FALSE) +
    geom_hline(yintercept = mean(cor_02.constrained$V2),
                colour = 'red', size = 1) +
    geom_vline(xintercept = mean(cor_02.constrained$V1),
                colour = 'red', size = 1) +
    geom_rect(ymin = 0, ymax = 10,
              xmin = 0, xmax = 10,
              colour = '#000000',
              alpha = 0,
              linetype = 2) +
    annotate(geom = 'text', x = -5, y = 15, hjust = 0, size = 5,
             label = str_glue("r = {round(cor(cor_02.constrained$V1,
                              cor_02.constrained$V2), 2)}")) +
    annotate(geom = 'text', x = -5, y = mean(cor_02.constrained$V2) + 1.7,
             hjust = 0, size = 5,
             label = str_glue("Mean = {round(mean(cor_02.constrained$V2), 2)}")) +
    annotate(geom = 'text', x = -5, y = mean(cor_02.constrained$V2) + 0.75,
             hjust = 0, size = 5,
             label = str_glue("SD = {round(sd(cor_02.constrained$V2),2)}")) +
    annotate(geom = 'text', x = mean(cor_02.constrained$V1) + 0.2, y = -3.8,
             hjust = 0, size = 5,
             label = str_glue("Mean = {round(mean(cor_02.constrained$V1), 2)}")) +
    annotate(geom = 'text', x = mean(cor_02.constrained$V1) + 0.2, y = -4.75,
             hjust = 0, size = 5,
             label = str_glue("SD = {round(sd(cor_02.constrained$V1), 2)}")) +
    labs(title = 'B: Constrained',
         caption = 'Population parameters: Mean = 6.2, SD = 1.7, r = 0.2') +
    scale_y_continuous(limits = c(-5, 15),
                       breaks = c(0, 5, 10)) +
    scale_x_continuous(limits = c(-5, 15),
                       breaks = c(0, 5, 10)) +
    theme(plot.caption = element_text(size = 14))
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Population parameters: Mean = 6.2, SD = 1.7, r = 0.2

```r
# Linear regression
summary(lm(V2 ~ V1, data = cor_02.constrained))
```

```
##
## Call:
## lm(formula = V2 ~ V1, data = cor_02.constrained)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.1061 -1.0728  0.0554  1.1495  4.4858
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.97445    0.19646  25.321  < 2e-16 ***
## V1           0.18259    0.03104   5.882 5.52e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.665 on 998 degrees of freedom
## Multiple R-squared:  0.03351,    Adjusted R-squared:  0.03254
## F-statistic:  34.6 on 1 and 998 DF,  p-value: 5.524e-09
```

```
Confint(lm(V2 ~ V1, data = cor_02.constrained))
```

```
##              Estimate     2.5 %    97.5 %
## (Intercept) 4.9744475 4.5889347 5.3599602
## V1          0.1825921 0.1216767 0.2435075
```

---

7 Publication plots

The output of the code is plotted to a ‘PNG’ graphics file.

```
# r = 1.0
plot_1 <- ggplot(data = cor_10.constrained) +
    aes(x = V1, y = V2) +
    geom_point() +
    geom_abline(slope = 1, intercept = 0) +
    geom_smooth(method = 'lm',
                se = FALSE,
                size = 1.5) +
    geom_hline(yintercept = mean(cor_10.constrained$V2),
               colour = 'red') +
    geom_vline(xintercept = mean(cor_10.constrained$V1),
               colour = 'red') +
    geom_rect(ymin = 0, ymax = 10,
              xmin = 0, xmax = 10,
              colour = '#000000',
              alpha = 0,
              linetype = 2) +
    annotate(geom = 'text', x = -5, y = 15, hjust = 0, size = 6,
             label = str_glue("r = {round(cor(cor_10.constrained$V1,
                                cor_10.constrained$V2), 2)}")) +
    annotate(geom = 'text', x = -5, y = mean(cor_10.constrained$V2) + 1.7,
             hjust = 0, size = 4,
             label = str_glue("Mean = {round(mean(cor_10.constrained$V2), 2)}")) +
    annotate(geom = 'text', x = -5, y = mean(cor_10.constrained$V2) + 0.75,
             hjust = 0, size = 4,
             label = str_glue("SD = {round(sd(cor_10.constrained$V2),2)}")) +
    annotate(geom = 'text', x = mean(cor_10.constrained$V1) + 0.2, y = -3.8,
             hjust = 0, size = 4,
             label = str_glue("Mean = {round(mean(cor_10.constrained$V1), 2)}")) +
    annotate(geom = 'text', x = mean(cor_10.constrained$V1) + 0.2, y = -4.75,
             hjust = 0, size = 4,
             label = str_glue("SD = {round(sd(cor_10.constrained$V1), 2)}")) +
    labs(subtitle = 'Population parameters: Mean = 6.2, SD = 1.7, r = 1.0') +
    scale_y_continuous(limits = c(-5, 15),
                       breaks = c(0, 5, 10)) +
    scale_x_continuous(limits = c(-5, 15),
                       breaks = c(0, 5, 10)) +
    theme_bw(base_size = 18) +
    theme(axis.text = element_text(colour = '#000000'),
          axis.title.x = element_blank(),
          panel.grid = element_blank(),
          plot.subtitle = element_text(size = 14))

# r = 0.8
plot_2 <- ggplot(data = cor_08.constrained) +
    aes(x = V1, y = V2) +
    geom_point() +
    geom_abline(slope = 1, intercept = 0) +
    geom_smooth(method = 'lm',
                se = FALSE,
                size = 1.5) +
    geom_hline(yintercept = mean(cor_08.constrained$V2),
               colour = 'red') +
    geom_vline(xintercept = mean(cor_08.constrained$V1),
               colour = 'red') +
    geom_rect(ymin = 0, ymax = 10,
              xmin = 0, xmax = 10,
              colour = '#000000',
              alpha = 0,
              linetype = 2) +
    annotate(geom = 'text', x = -5, y = 15, hjust = 0, size = 6,
             label = str_glue("r = {round(cor(cor_08.constrained$V1,
                              cor_08.constrained$V2), 2)}")) +
    annotate(geom = 'text', x = -5, y = mean(cor_08.constrained$V2) + 1.7,
             hjust = 0, size = 4,
             label = str_glue("Mean = {round(mean(cor_08.constrained$V2), 2)}")) +
    annotate(geom = 'text', x = -5, y = mean(cor_08.constrained$V2) + 0.75,
             hjust = 0, size = 4,
             label = str_glue("SD = {round(sd(cor_08.constrained$V2),2)}")) +
    annotate(geom = 'text', x = mean(cor_08.constrained$V1) + 0.2, y = -3.8,
             hjust = 0, size = 4,
             label = str_glue("Mean = {round(mean(cor_08.constrained$V1), 2)}")) +
    annotate(geom = 'text', x = mean(cor_08.constrained$V1) + 0.2, y = -4.75,
             hjust = 0, size = 4,
             label = str_glue("SD = {round(sd(cor_08.constrained$V1), 2)}")) +
    labs(subtitle = 'Population parameters: Mean = 6.2, SD = 1.7, r = 0.8') +
    scale_y_continuous(limits = c(-5, 15),
                       breaks = c(0, 5, 10)) +
    scale_x_continuous(limits = c(-5, 15),
                       breaks = c(0, 5, 10)) +
    theme_bw(base_size = 18) +
    theme(axis.text = element_text(colour = '#000000'),
          axis.title = element_blank(),
          panel.grid = element_blank(),
          plot.subtitle = element_text(size = 14))

# r = 0.5
```

```r
plot_3 <- ggplot(data = cor_05.constrained) +
    aes(x = V1, y = V2) +
    geom_point() +
    geom_abline(slope = 1, intercept = 0) +
    geom_smooth(method = 'lm',
                se = FALSE,
                size = 1.5) +
    geom_hline(yintercept = mean(cor_05.constrained$V2),
               colour = 'red') +
    geom_vline(xintercept = mean(cor_05.constrained$V1),
               colour = 'red') +
    geom_rect(ymin = 0, ymax = 10,
              xmin = 0, xmax = 10,
              colour = '#000000',
              alpha = 0,
              linetype = 2) +
    annotate(geom = 'text', x = -5, y = 15, hjust = 0, size = 6,
             label = str_glue("r = {round(cor(cor_05.constrained$V1,
                                 cor_05.constrained$V2), 2)}")) +
    annotate(geom = 'text', x = -5, y = mean(cor_05.constrained$V2) + 1.7,
             hjust = 0, size = 4,
             label = str_glue("Mean = {round(mean(cor_05.constrained$V2), 2)}")) +
    annotate(geom = 'text', x = -5, y = mean(cor_05.constrained$V2) + 0.75,
             hjust = 0, size = 4,
             label = str_glue("SD = {round(sd(cor_05.constrained$V2),2)}")) +
    annotate(geom = 'text', x = mean(cor_05.constrained$V1) + 0.2, y = -3.8,
             hjust = 0, size = 4,
             label = str_glue("Mean = {round(mean(cor_05.constrained$V1), 2)}")) +
    annotate(geom = 'text', x = mean(cor_05.constrained$V1) + 0.2, y = -4.75,
             hjust = 0, size = 4,
             label = str_glue("SD = {round(sd(cor_05.constrained$V1), 2)}")) +
    labs(subtitle = 'Population parameters: Mean = 6.2, SD = 1.7, r = 0.5') +
    scale_y_continuous(limits = c(-5, 15),
                       breaks = c(0, 5, 10)) +
    scale_x_continuous(limits = c(-5, 15),
                       breaks = c(0, 5, 10)) +
    theme_bw(base_size = 18) +
    theme(axis.text = element_text(colour = '#000000'),
          panel.grid = element_blank(),
          plot.subtitle = element_text(size = 14))

# r = 0.2
plot_4 <- ggplot(data = cor_02.constrained) +
    aes(x = V1, y = V2) +
    geom_point() +
    geom_abline(slope = 1, intercept = 0) +
    geom_smooth(method = 'lm',
                se = FALSE,
                size = 1.5) +
    geom_hline(yintercept = mean(cor_02.constrained$V2),
               colour = 'red') +
    geom_vline(xintercept = mean(cor_02.constrained$V1),
               colour = 'red') +
    geom_rect(ymin = 0, ymax = 10,
              xmin = 0, xmax = 10,
              colour = '#000000',
              alpha = 0,
              linetype = 2) +
    annotate(geom = 'text', x = -5, y = 15, hjust = 0, size = 6,
             label = str_glue("r = {round(cor(cor_02.constrained$V1,
                              cor_02.constrained$V2), 2)}")) +
    annotate(geom = 'text', x = -5, y = mean(cor_02.constrained$V2) + 1.7,
             hjust = 0, size = 4,
             label = str_glue("Mean = {round(mean(cor_02.constrained$V2), 2)}")) +
    annotate(geom = 'text', x = -5, y = mean(cor_02.constrained$V2) + 0.75,
             hjust = 0, size = 4,
             label = str_glue("SD = {round(sd(cor_02.constrained$V2),2)}")) +
    annotate(geom = 'text', x = mean(cor_02.constrained$V1) + 0.2, y = -3.8,
             hjust = 0, size = 4,
             label = str_glue("Mean = {round(mean(cor_02.constrained$V1), 2)}")) +
    annotate(geom = 'text', x = mean(cor_02.constrained$V1) + 0.2, y = -4.75,
             hjust = 0, size = 4,
             label = str_glue("SD = {round(sd(cor_02.constrained$V1), 2)}")) +
    labs(subtitle = 'Population parameters: Mean = 6.2, SD = 1.7, r = 0.2') +
    scale_y_continuous(limits = c(-5, 15),
                       breaks = c(0, 5, 10)) +
    scale_x_continuous(limits = c(-5, 15),
                       breaks = c(0, 5, 10)) +
    theme_bw(base_size = 18) +
    theme(axis.text = element_text(colour = '#000000'),
          axis.title.y = element_blank(),
          panel.grid = element_blank(),
          plot.subtitle = element_text(size = 14))

# Patchwork
patch <- plot_1 + plot_2 + plot_3 + plot_4 +
  plot_layout(ncol = 2) + plot_annotation(tag_levels = 'A')

ggsave(filename = 'figures/figure_1.png',
       plot = patch,
       width = 11.5,
       height = 11.5)
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```

---

8 Session information

```r
sessionInfo()
```

```
## R version 4.0.2 (2020-06-22)
## Platform: x86_64-apple-darwin17.0 (64-bit)
## Running under: macOS Catalina 10.15.5
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.0/Resources/lib/libRblas.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.0/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] patchwork_1.0.1 car_3.0-8       carData_3.0-4   MBESS_4.7.0
##  [5] MASS_7.3-51.6   magrittr_1.5    forcats_0.5.0   stringr_1.4.0
##  [9] dplyr_1.0.0     purrr_0.3.4     readr_1.3.1     tidyr_1.1.0
## [13] tibble_3.0.1    ggplot2_3.3.2   tidyverse_1.3.0
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_1.0.4.6      lubridate_1.7.9   lattice_0.20-41   assertthat_0.2.1
##  [5] digest_0.6.25     R6_2.4.1          cellranger_1.1.0  backports_1.1.8
##  [9] reprex_0.3.0      evaluate_0.14     httr_1.4.1        pillar_1.4.4
## [13] rlang_0.4.6       curl_4.3          readxl_1.3.1      rstudioapi_0.11
## [17] data.table_1.12.8 blob_1.2.1        Matrix_1.2-18     rmarkdown_2.3
## [21] splines_4.0.2     foreign_0.8-80    munsell_0.5.0     broom_0.5.6
## [25] compiler_4.0.2    modelr_0.1.8      xfun_0.15         pkgconfig_2.0.3
## [29] mgcv_1.8-31       htmltools_0.5.0   tidyselect_1.1.0  rio_0.5.16
## [33] fansi_0.4.1       crayon_1.3.4      dbplyr_1.4.4      withr_2.2.0
## [37] grid_4.0.2        nlme_3.1-148      jsonlite_1.6.1    gtable_0.3.0
## [41] lifecycle_0.2.0   DBI_1.1.0         scales_1.1.1      zip_2.0.4
## [45] cli_2.0.2         stringi_1.4.6     farver_2.0.3      fs_1.4.1
## [49] xml2_1.3.2        ellipsis_0.3.1    generics_0.0.2    vctrs_0.3.1
## [53] openxlsx_4.1.5    tools_4.0.2       glue_1.4.1        hms_0.5.3
## [57] abind_1.4-5       yaml_2.2.1        colorspace_1.4-1  rvest_0.3.5
## [61] knitr_1.29        haven_2.3.1
```