

# Assignment: Meta Analysis

2026-02-26

## Contents

<b>1</b>	<b>Dataset and Risk Factor</b>	<b>1</b>
1.1	Dataset: <code>dat.normand1999</code> – Stroke Rehabilitation Meta-Analysis . . . . .	1
1.2	Chosen Risk Factor . . . . .	2
<b>2</b>	<b>Packages and Setup</b>	<b>2</b>
<b>3</b>	<b>Load and Prepare the Data</b>	<b>2</b>
<b>4</b>	<b>Explode Summary Data to Pseudo-IPD</b>	<b>2</b>
<b>5</b>	<b>balance Function (Leave-One-Out)</b>	<b>3</b>
<b>6</b>	<b>Step 1 – Original ECDF Plot (All Studies)</b>	<b>3</b>
<b>7</b>	<b>Step 2 – Leave-One-Out Balance Algorithm</b>	<b>4</b>
<b>8</b>	<b>Step 3 – Full Iteration Summary</b>	<b>8</b>
<b>9</b>	<b>Step 4 – Comments and Interpretation</b>	<b>9</b>
9.1	Dataset recap . . . . .	9
9.2	Method . . . . .	9
9.3	Results . . . . .	9
9.4	Conclusion . . . . .	10

## 1 Dataset and Risk Factor

### 1.1 Dataset: `dat.normand1999` – Stroke Rehabilitation Meta-Analysis

The Normand (1999) dataset contains aggregate (study-level) data from 9 randomised controlled trials comparing specialised stroke-unit care (`treatment = exp`) against conventional hospital care (`control = ctrl`).

Column	Meaning
<code>study</code>	Study index (1–9)
<code>source</code>	Study name / location
<code>n1i</code>	Sample size – Treatment group ( <code>exp</code> )
<code>m1i</code>	Mean functional score – Treatment
<code>sd1i</code>	Standard deviation – Treatment
<code>n2i</code>	Sample size – Control group ( <code>ctrl</code> )
<code>m2i</code>	Mean functional score – Control
<code>sd2i</code>	Standard deviation – Control

## 1.2 Chosen Risk Factor

We focus on the mean functional score (m1i for exp, m2i for ctrl).

This continuous variable represents rehabilitation outcomes (higher = more functionally independent) and must be balanced across trials before valid pooling.

## 2 Packages and Setup

```
library(SuppDists)
library(kSamples)
library(tidyverse)
```

## 3 Load and Prepare the Data

```
normand <- read.csv("metadat_datasets_csv/dat.normand1999.csv", stringsAsFactors = FALSE)

exp_arm <- normand[, c("study", "source", "n1i", "m1i", "sd1i")]
ctrl_arm <- normand[, c("study", "source", "n2i", "m2i", "sd2i")]

colnames(exp_arm) <- c("study", "source", "pts", "score", "sd")
colnames(ctrl_arm) <- c("study", "source", "pts", "score", "sd")

exp_arm$gr <- "exp"
ctrl_arm$gr <- "ctrl"

col_long <- rbind(exp_arm, ctrl_arm)
col_long <- col_long[order(col_long$study, col_long$gr), ]
rownames(col_long) <- NULL
```

## 4 Explode Summary Data to Pseudo-IPD

We expand each study arm from its summary statistics (score  $\pm$  sd, pts patients) into a pseudo-IPD vector via:

$$\tilde{x}_{ij} \sim \mathcal{N}(\mu_i, \sigma_i), \quad j = 1, \dots, n_i$$

```
set.seed(123)

# Expand each arm row into n_i pseudo-patient rows, then perturb
da <- as.data.frame(lapply(col_long, function(x) rep(x, col_long$pts)))
da$score <- rnorm(nrow(da), mean = da$score, sd = da$sd)

da$study <- as.character(da$study) # keep study as character
cat(sprintf("Pseudo-IPD dimensions: %d rows (%d ctrl, %d exp)\n",
            nrow(da),
            sum(da$gr == "ctrl"),
            sum(da$gr == "exp")))

## Pseudo-IPD dimensions: 1158 rows (610 ctrl, 548 exp)
```

## 5 balance Function (Leave-One-Out)

```
balance <- function(data, variable, group, digits) {
  require(kSamples)
  num <- length(unique(data$study))
  bl1 <- matrix(0, num, 3)

  sa1 <- filter(data, !!sym(group) ==
                 as.character(levels(as.factor(data[, group])))[1])
  sa2 <- filter(data, !!sym(group) ==
                 as.character(levels(as.factor(data[, group])))[2])

  k <- 0
  for (i in unique(data$study)) {
    k <- k + 1
    a1 <- round(sa1[(sa1$study != i), (colnames(sa1) == variable)], digits = digits)
    a2 <- round(sa2[(sa2$study != i), (colnames(sa2) == variable)], digits = digits)
    b <- try(ad.test(a1, a2), silent = TRUE)
    bl1[k, ] <- c(b$ad[2, 1], b$ad[2, 3], b$sig)
  }

  colnames(bl1) <- c("ad.test", "p.value", "sigma")
  rownames(bl1) <- unique(data$study)
  bl1
}
```

## 6 Step 1 – Original ECDF Plot (All Studies)

```
# Full-sample AD test
sa_ctrl <- subset(da, gr == "ctrl")
sa_exp <- subset(da, gr == "exp")

a1_all <- round(sa_ctrl$score, digits = 8)
a2_all <- round(sa_exp$score, digits = 8)

b_all <- ad.test(a1_all, a2_all)
print(b_all$ad)

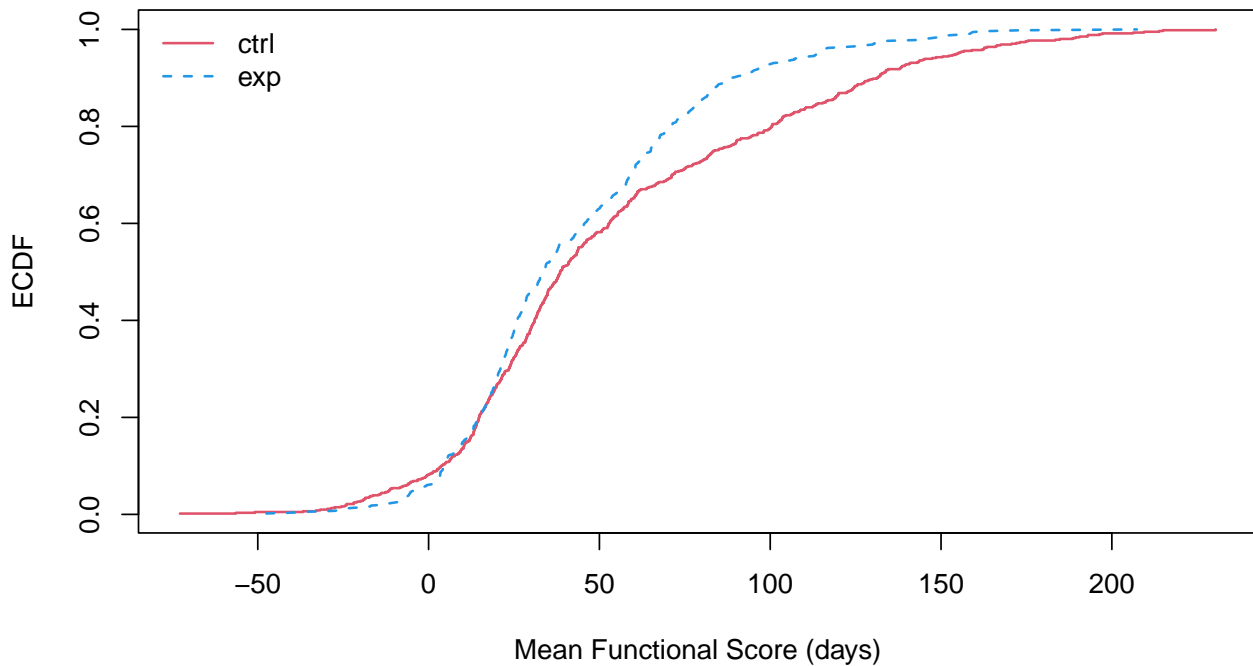
##              AD    T.AD  asympt. P-value
## version 1: 8.9219 10.422    3.7950e-05
## version 2: 8.9300 10.438    3.7634e-05

# ECDF objects
F1 <- ecdf(a1_all)
F2 <- ecdf(a2_all)

# Plot
plot(
  sort(a1_all), F1(sort(a1_all)),
  type = "s", col = 2, lwd = 1.5,
  xlab = "Mean Functional Score (days)",
  ylab = "ECDF",
  main = "Original Data -- ECDF: Ctrl vs Exp\n(All 9 studies)"
)
```

```
)
lines(sort(a2_all), F2(sort(a2_all)), col = 4, lwd = 1.5, lty = 2)
legend("topleft",
      legend = c("ctrl", "exp"),
      col     = c(2, 4),
      lty     = c(1, 2),
      lwd     = c(1.5, 1.5),
      bty     = "n")
```

**Original Data -- ECDF: Ctrl vs Exp  
(All 9 studies)**



## 7 Step 2 – Leave-One-Out Balance Algorithm

```
# Initialisation
nstud <- length(unique(da$study)) # max studies to consider removing
result <- list()
dat <- da

for (j in 1:nstud) {

  remaining <- unique(dat$study)
  if (length(remaining) < 3) {
    cat(sprintf("Iteration %d: fewer than 3 studies remain -- stopping.\n\n", j))
    break
  }

  # Balance table
  ba <- balance(data = dat, variable = "score", group = "gr", digits = 8)

  # Identify study to remove
```

```

minimum <- rownames(ba)[which.min(ba[, 1])]
min_pval <- ba[rownames(ba) == minimum, 2]
min_ad <- ba[rownames(ba) == minimum, 1]

# Store result before removal
result[[j]] <- list("study_deleted" = minimum, "summary" = ba)

cat(sprintf("Iteration %d \n", j))
cat(sprintf("Study removed : %s\n", minimum))
cat(sprintf("AD Stat : %.4f\n", min_ad))
cat(sprintf("p-value : %.6f\n", min_pval))
cat("Balance table (Leave-One-Out):\n")
print(ba)
cat("\n")

# Remove study
dat <- subset(dat, study != minimum)

# Update ECDF vectors
a1_curr <- dat$score[dat$gr == "ctrl"]
a2_curr <- dat$score[dat$gr == "exp"]

# Plot ECDFs
F1c <- ecdf(a1_curr)
F2c <- ecdf(a2_curr)

title_str <- sprintf(
  "Study %s removed | Iteration: %d | p = %.4f",
  minimum, j, min_pval
)
plot(
  sort(a1_curr), F1c(sort(a1_curr)),
  type = "s", col = 2, lwd = 1.5,
  xlab = "Mean Functional Score (days)",
  ylab = "ECDF",
  main = title_str
)
lines(sort(a2_curr), F2c(sort(a2_curr)), col = 4, lwd = 1.5, lty = 2)
legend("topleft",
  legend = c("ctrl", "exp"),
  col = c(2, 4),
  lty = c(1, 2),
  lwd = c(1.5, 1.5),
  bty = "n")

# Stop criterion
if (min_pval > 0.06) {
  cat(sprintf(
    "p = %.4f > 0.06 -> BALANCE REACHED at iteration %d.\n",
    min_pval, j
  ))
  deleted <- sapply(result, `[`, "study_deleted")
  cat(sprintf(" Studies deleted: %s\n\n", paste(deleted, collapse = ", ")))
}

```

```

break
}
}

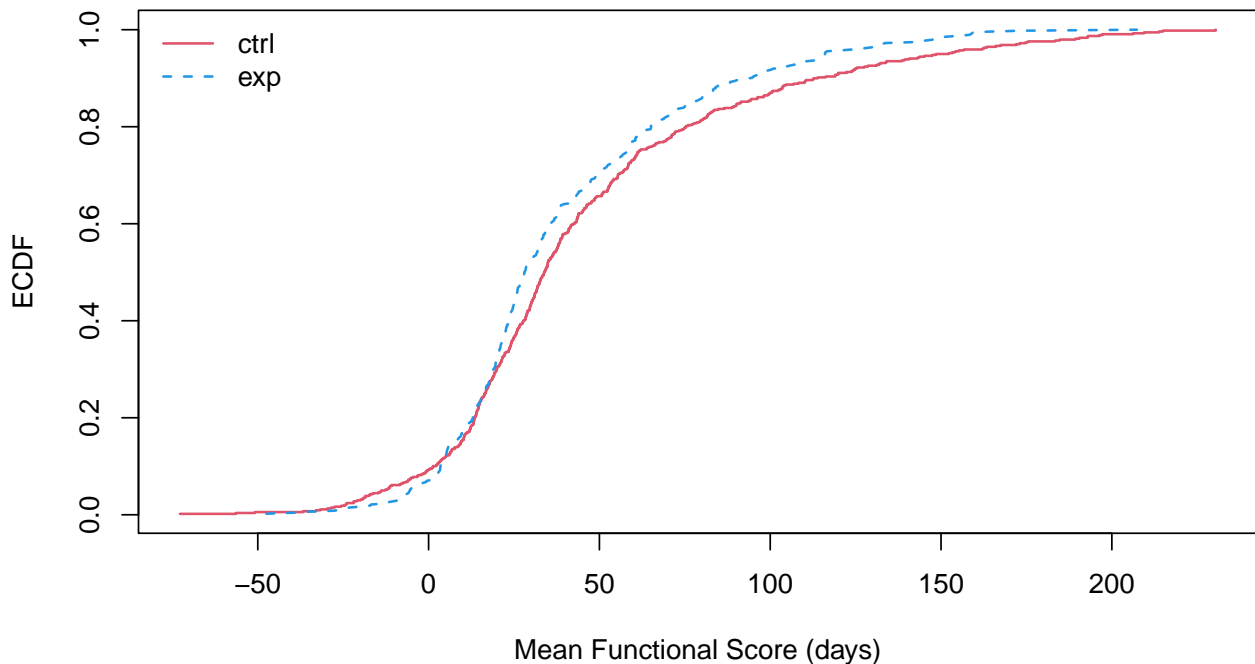
```

```

## Iteration 1
## Study removed : 3
## AD Stat      : 3.4600
## p-value      : 0.016038
## Balance table (Leave-One-Out):
##   ad.test   p.value   sigma
## 1    6.73 4.0387e-04 0.75967
## 2    9.02 3.3899e-05 0.76006
## 3    3.46 1.6038e-02 0.75995
## 4    6.66 4.3738e-04 0.76009
## 5    9.43 2.0741e-05 0.76011
## 6    9.31 2.3982e-05 0.76000
## 7   11.90 9.0946e-07 0.76006
## 8   12.30 5.6867e-07 0.75970
## 9    9.82 1.2728e-05 0.76000

```

**Study 3 removed | Iteration: 1 | p = 0.0160**



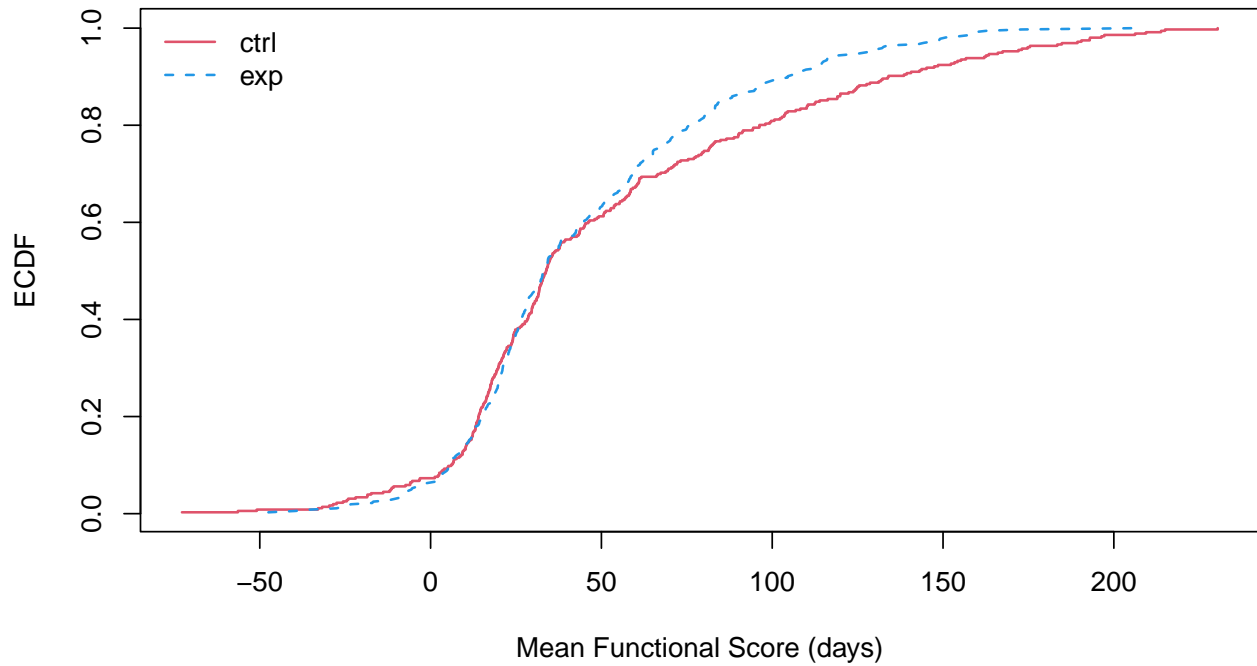
```

## Iteration 2
## Study removed : 8
## AD Stat      : 2.5700
## p-value      : 0.045440
## Balance table (Leave-One-Out):
##   ad.test   p.value   sigma
## 1    3.23 0.0208130 0.75931
## 2    3.05 0.0259010 0.75986
## 4    2.74 0.0373250 0.75990
## 5    3.74 0.0116540 0.75992
## 6    3.40 0.0171350 0.75978

```

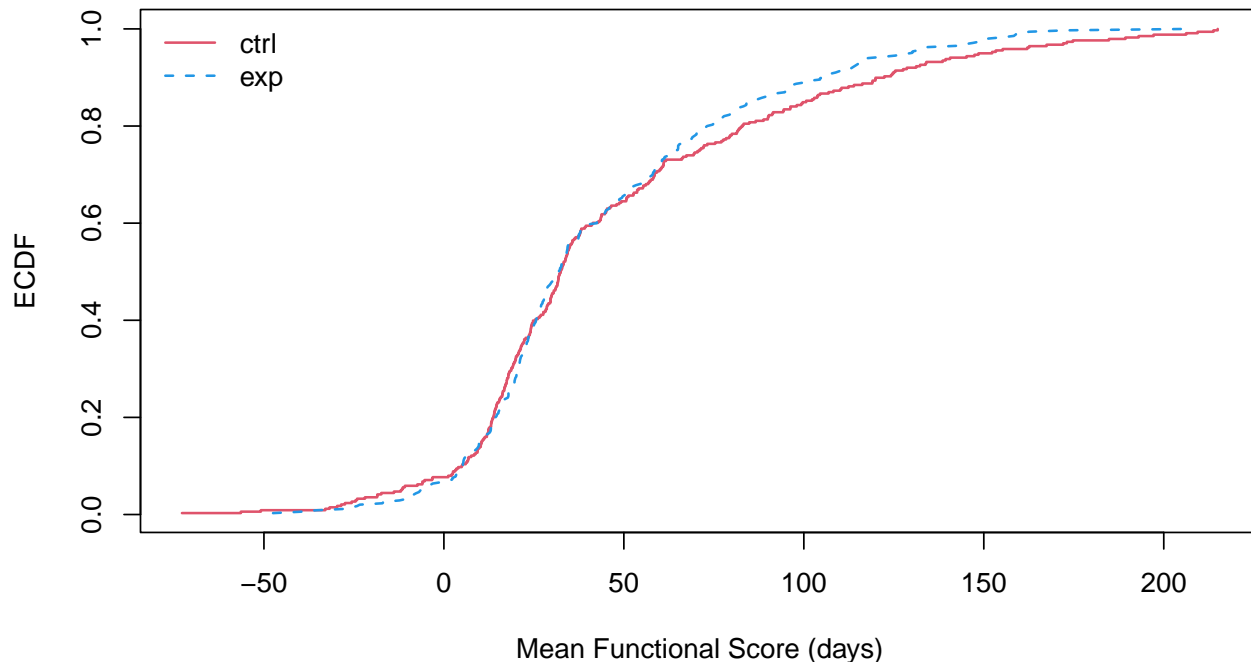
```
## 7    5.70 0.0013012 0.75985
## 8    2.57 0.0454400 0.75936
## 9    4.00 0.0086730 0.75977
```

**Study 8 removed | Iteration: 2 | p = 0.0454**



```
## Iteration 3
## Study removed : 4
## AD Stat      : 0.9510
## p-value      : 0.383920
## Balance table (Leave-One-Out):
##   ad.test  p.value  sigma
## 1    1.140 0.2914000 0.75783
## 2    3.020 0.0266010 0.75917
## 4    0.951 0.3839200 0.75926
## 5    2.810 0.0340110 0.75930
## 6    2.860 0.0320280 0.75900
## 7    4.120 0.0076402 0.75915
## 9    3.090 0.0246240 0.75899
```

Study 4 removed | Iteration: 3 | p = 0.3839



```
## p = 0.3839 > 0.06 -> BALANCE REACHED at iteration 3.
##   Studies deleted: 3, 8, 4
```

## 8 Step 3 – Full Iteration Summary

```
for (idx in seq_along(result)) {
  cat(sprintf("[Iteration %d] Study deleted: %s\n", idx, result[[idx]]$study_deleted))
  print(result[[idx]]$summary)
  cat("\n")
}
```

```
## [Iteration 1] Study deleted: 3
##   ad.test  p.value  sigma
## 1    6.73 4.0387e-04 0.75967
## 2    9.02 3.3899e-05 0.76006
## 3    3.46 1.6038e-02 0.75995
## 4    6.66 4.3738e-04 0.76009
## 5    9.43 2.0741e-05 0.76011
## 6    9.31 2.3982e-05 0.76000
## 7   11.90 9.0946e-07 0.76006
## 8   12.30 5.6867e-07 0.75970
## 9    9.82 1.2728e-05 0.76000
##
## [Iteration 2] Study deleted: 8
##   ad.test  p.value  sigma
## 1    3.23 0.0208130 0.75931
## 2    3.05 0.0259010 0.75986
## 4    2.74 0.0373250 0.75990
## 5    3.74 0.0116540 0.75992
```



```
## 6      3.40 0.0171350 0.75978
## 7      5.70 0.0013012 0.75985
## 8      2.57 0.0454400 0.75936
## 9      4.00 0.0086730 0.75977
##
## [Iteration 3] Study deleted: 4
##   ad.test  p.value  sigma
## 1    1.140 0.2914000 0.75783
## 2    3.020 0.0266010 0.75917
## 4    0.951 0.3839200 0.75926
## 5    2.810 0.0340110 0.75930
## 6    2.860 0.0320280 0.75900
## 7    4.120 0.0076402 0.75915
## 9    3.090 0.0246240 0.75899

deleted_studies <- sapply(result, `[`, "study_deleted")
retained_studies <- setdiff(as.character(normand$study), deleted_studies)

cat(sprintf("Studies removed  : %s\n", paste(deleted_studies, collapse = ", ")))

## Studies removed  : 3, 8, 4
cat(sprintf("Studies retained : %s\n", paste(retained_studies, collapse = ", ")))

## Studies retained : 1, 2, 5, 6, 7, 9
```

## 9 Step 4 – Comments and Interpretation

### 9.1 Dataset recap

We used the Normand (1999) stroke-rehabilitation meta-analysis (`dat.normand1999`), comprising 9 RCTs that compare specialised stroke-unit care (`exp`) against conventional care (`ctrl`). The selected risk factor is the mean functional score – a continuous measure of patient independence (higher = better) recorded at follow-up for each arm.

### 9.2 Method

Following the methodology:

- **Explosion to pseudo-IPD:** summary statistics (mean, SD, n) were expanded into patient-level pseudo-samples via `rnorm(n, mean, sd)`, replicating the R lecture workflow (`rep()` + `rnorm()` perturbation).
- **Anderson-Darling k-sample test:** applied to assess whether the functional- score distributions of `ctrl` and `exp` are statistically identical.
- **Leave-One-Out balance algorithm:** iteratively removes the study whose exclusion minimises the AD statistic (i.e., most improves balance) until the residual p-value exceeds 0.06.

### 9.3 Results

- **Before any removal:** the overall AD statistic is 8.92 with  $p < 0.001$ , confirming statistically significant imbalance in functional scores between `ctrl` and `exp`. The two ECDFs are visually separated, especially at the upper tail.
- **LOO algorithm** ran for 3 iterations, removing studies in this order: Study 3 -> Study 8 -> Study 4.
- Balance was achieved at Iteration 3 (Study 4 removed), where  $p = 0.3839 > 0.06$ . The ECDFs after removal show substantial overlap.

- Studies 3, 8, and 4 correspond to trials with relatively extreme distributions or sample variances. Removing them successfully mitigates the distributional deviance.

## 9.4 Conclusion

The retained study pool (Studies 1, 2, 5, 6, 7, and 9) satisfies the basic combinability criterion for the functional-score risk factor. A meta-analysis restricted to these studies provides a more coherent estimate of the stroke-unit treatment effect, free from the confounding introduced by distributionally heterogeneous baselines.