9.4 treatment interactions

Chris Parrish

July 3, 2016

Contents

9.4 treatment interactions

reference:
- ARM chapter 09, github

```
library(arm)          # for se.coef
library(rstan)
rstan_options(auto_write = TRUE)
options(mc.cores = parallel::detectCores())
library(ggplot2)
```

treatment interactions

data

```
### Data
source("electric_grade4.data.R", echo = TRUE)
```

```
##
## > N <- 42
##
## > post_test <- c(116.2, 116.9, 106.9, 104.6, 114.2,
## +     113.6, 116.6, 114.8, 114.9, 111, 113.9, 115.6, 116.2, 119.6,
## +     109.6, 122, 109.7, 112. .... [TRUNCATED]
##
## > pre_test <- c(105.9, 100.8, 91.7, 97.5, 106.5, 107.4,
```

```
## +     111.4, 110, 106.9, 106.7, 104.1, 109.7, 110.6, 115.2, 101.9,
## +     119.8, 108.8, 104.6 .... [TRUNCATED]
##
## > grade <- c(4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
## +     4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
## +     4, 4, 4, 4, 4, 4, 4, 4, 4,  .... [TRUNCATED]
##
## > treatment <- c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
## +     1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
## +     0, 0, 0, 0, 0, 0, 0, 0, 0, .... [TRUNCATED]
```

model

electric_tr.stan

```
data {
  int<lower=0> N;
  vector[N] post_test;
  vector[N] treatment;
}
parameters {
  vector[2] beta;
  real<lower=0> sigma;
}
model {
  post_test ~ normal(beta[1] + beta[2] * treatment, sigma);
}
```

fit

```
### Model with only treatment indicator: post_test ~ treatment
data.list <- c("N", "post_test", "pre_test", "treatment")
electric_tr.sf <- stan(file='electric_tr.stan', data=data.list,
                       iter=1000, chains=4)
print(electric_tr.sf)
```

```
## Inference for Stan model: electric_tr.
## 4 chains, each with iter=1000; warmup=500; thin=1;
## post-warmup draws per chain=500, total post-warmup draws=2000.
##
##           mean se_mean   sd   2.5%    25%    50%    75%  97.5% n_eff Rhat
## beta[1] 110.25    0.04 1.30 107.59 109.42 110.25 111.10 112.74   971 1.00
## beta[2]   3.88    0.06 1.85   0.21   2.66   3.85   5.09   7.73   889 1.00
## sigma     6.16    0.02 0.71   4.94   5.65   6.09   6.61   7.71   982 1.00
## lp__    -94.70    0.04 1.28 -97.97 -95.34 -94.34 -93.74 -93.22  1026 1.01
##
## Samples were drawn using NUTS(diag_e) at Sun Jul 10 13:41:55 2016.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
##  The estimated Bayesian Fraction of Missing Information is a measure of
##  the efficiency of the sampler with values close to 1 being ideal.
##  For each chain, these estimates are
```

```
##  0.8 0.9 1.1 1
```

model

electric_trpre.stan

```
data {
  int<lower=0> N;
  vector[N] post_test;
  vector[N] treatment;
  vector[N] pre_test;
}
parameters {
  vector[3] beta;
  real<lower=0> sigma;
}
model {
  post_test ~ normal(beta[1] + beta[2] * treatment + beta[3] * pre_test, sigma);
}
```

fit

```
### Model controlling for pre-test: post_test ~ treatment + pre_test
electric_trpre.sf <- stan(file='electric_trpre.stan', data=data.list,
                          iter=1000, chains=4)
print(electric_trpre.sf)
```

```
## Inference for Stan model: electric_trpre.
## 4 chains, each with iter=1000; warmup=500; thin=1;
## post-warmup draws per chain=500, total post-warmup draws=2000.
##
##           mean se_mean   sd   2.5%    25%    50%    75%  97.5% n_eff Rhat
## beta[1]  42.30    0.16 4.33  33.56  39.42  42.40  45.23  50.62   734 1.00
## beta[2]   1.66    0.03 0.69   0.29   1.20   1.68   2.09   3.05   759 1.00
## beta[3]   0.65    0.00 0.04   0.58   0.62   0.65   0.68   0.74   714 1.00
## sigma     2.25    0.01 0.26   1.82   2.07   2.22   2.41   2.84   959 1.00
## lp__    -53.55    0.06 1.45 -57.18 -54.20 -53.24 -52.50 -51.73   583 1.01
##
## Samples were drawn using NUTS(diag_e) at Sun Jul 10 13:41:59 2016.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
##  The estimated Bayesian Fraction of Missing Information is a measure of
##  the efficiency of the sampler with values close to 1 being ideal.
##  For each chain, these estimates are
##  1.2 1 0.8 1.1
```

model

electric_inter.stan

```
data {
```

```
  int<lower=0> N;
  vector[N] post_test;
  vector[N] treatment;
  vector[N] pre_test;
}
transformed data {
  vector[N] inter;           // interaction
  inter = treatment .* pre_test;
}
parameters {
  vector[4] beta;
  real<lower=0> sigma;
}
model {
  post_test ~ normal(beta[1] + beta[2] * treatment + beta[3] * pre_test
                     + beta[4] * inter, sigma);
}
```

fit

```
### Model with interaction: post_test ~ pre_test + treatment + pre_test:treatment
electric_inter.sf <- stan(file='electric_inter.stan', data=data.list,
                          iter=1000, chains=4)
print(electric_inter.sf, pars = c("beta", "lp__"))
```

```
## Inference for Stan model: electric_inter.
## 4 chains, each with iter=1000; warmup=500; thin=1;
## post-warmup draws per chain=500, total post-warmup draws=2000.
##
##           mean se_mean    sd   2.5%    25%    50%    75%  97.5% n_eff Rhat
## beta[1]  38.20    0.21  5.04  28.23  35.00  38.20  41.60  47.80   577 1.01
## beta[2]  17.12    0.42 10.01  -2.75  10.68  17.09  23.45  37.09   577 1.01
## beta[3]   0.69    0.00  0.05   0.60   0.66   0.69   0.72   0.79   569 1.01
## beta[4]  -0.14    0.00  0.09  -0.33  -0.20  -0.14  -0.08   0.04   574 1.01
## lp__    -52.69    0.07  1.65 -56.68 -53.56 -52.35 -51.46 -50.53   492 1.00
##
## Samples were drawn using NUTS(diag_e) at Sun Jul 10 13:42:03 2016.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
##  The estimated Bayesian Fraction of Missing Information is a measure of
##  the efficiency of the sampler with values close to 1 being ideal.
##  For each chain, these estimates are
##  0.9 0.9 0.9 1.2
```

figure 9.7

```
## Figure 9.7
inter.ggdf <- data.frame(c(), c(), c(), c(), c(), c(), c(), c()) # empty data frame
for (i in 1:4) {
    source(paste("electric_grade", i, ".data.R", sep = ""))
    temp <- data.frame(post_test, pre_test, grade, treatment)
```

```
    data.list <- c("N", "post_test", "pre_test", "treatment")
    sf <- stan(file='electric_inter.stan', data=data.list,
               iter=1000, chains=4)
    beta.post <- extract(sf, "beta")$beta
    beta.mean <- colMeans(beta.post)
    temp$beta1 <- beta.mean[1]
    temp$beta2 <- beta.mean[2]
    temp$beta3 <- beta.mean[3]
    temp$beta4 <- beta.mean[4]
    inter.ggdf <- rbind(inter.ggdf, temp)
}
inter.ggdf$grade <- factor(inter.ggdf$grade)
levels(inter.ggdf$grade) <- paste("Grade", levels(inter.ggdf$grade))
inter.ggdf$treatment <- factor(inter.ggdf$treatment)
```

```
p1 <- ggplot(inter.ggdf, aes(x = pre_test, y = post_test)) +
    geom_point(aes(shape = treatment)) +
    scale_shape_manual(values = c(16, 1)) +
    geom_abline(aes(intercept = beta1 + beta2 * (as.numeric(treatment)-1),
                    slope = beta3 + beta4 * (as.numeric(treatment)-1),
                    linetype = treatment)) +
    scale_linetype_manual(values = c(2, 1)) +
    facet_grid(. ~ grade) +
    scale_x_continuous(expression(paste("pre-test, ", x[i])),
                       limits = c(0, 125)) +
    scale_y_continuous(expression(paste("post-test, ", y[i])),
                       limits = c(0, 125)) +
    theme(legend.position = "none")
print(p1)
```

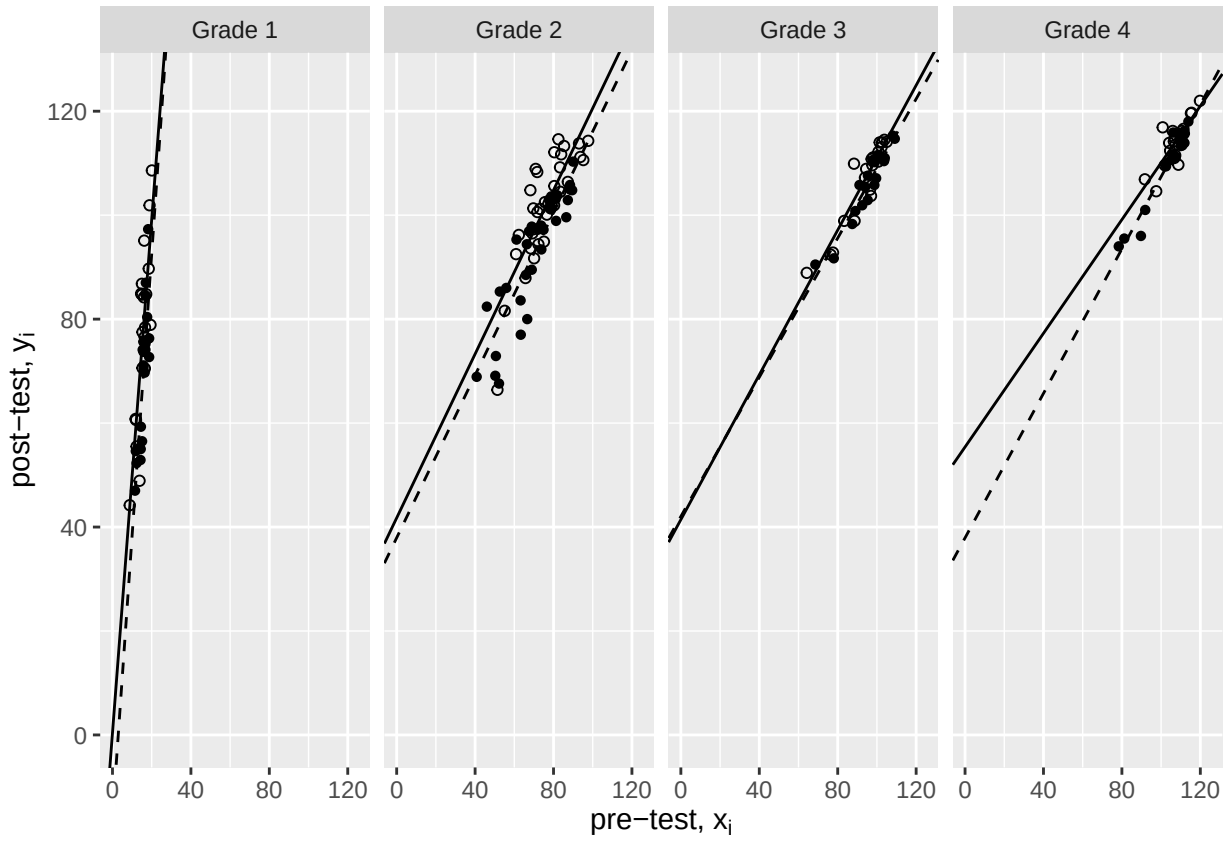
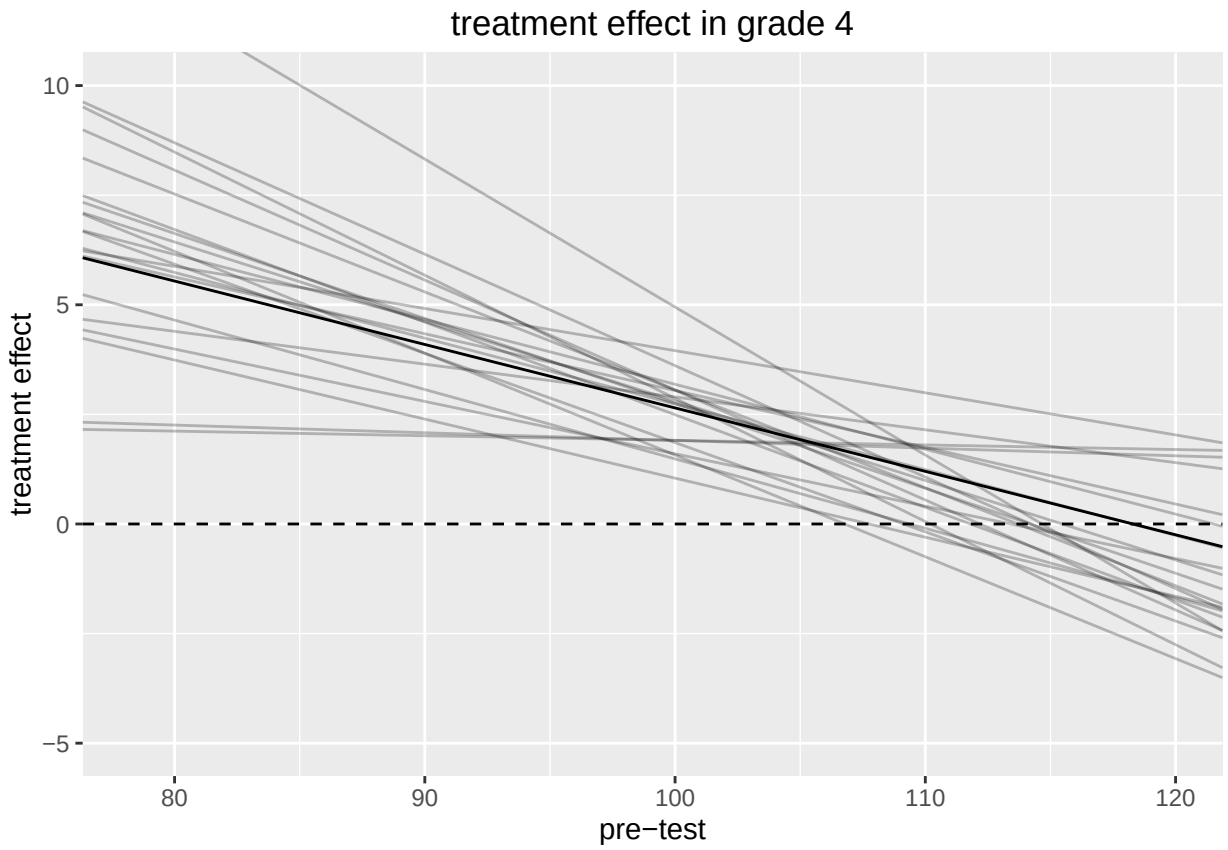


figure 9.8

```
## Uncertainty (Figure 9.8)
source("electric_grade4.data.R")
beta.post <- extract(electric_inter.sf, "beta")$beta
beta.mean <- colMeans(beta.post)
n <- 20
ndx <- sample(nrow(beta.post), n)
uncertainty.ggdf <- data.frame(sampled_int = beta.post[ndx, 2],
                               sampled_slope = beta.post[ndx, 4],
                               id = ndx)
# dev.new()
p2 <- ggplot(data.frame(pre_test),
             aes(x = pre_test, y = beta.mean[2] + beta.mean[4] * pre_test)) +
    geom_line(size = 0) +
    geom_abline(aes(intercept = beta.mean[2], slope = beta.mean[4])) +
    geom_abline(aes(intercept = sampled_int, slope = sampled_slope, group = id),
                data = uncertainty.ggdf, alpha = 0.25) +
    geom_hline(yintercept = 0, linetype = "dashed") +
    scale_x_continuous("pre-test") +
    scale_y_continuous("treatment effect", limits = c(-5, 10)) +
    ggtitle("treatment effect in grade 4")
print(p2)
```



compute the average treatment effect & summarize

```
## Compute the average treatment effect & summarize
n.iter <- nrow(beta.post)
effect <- array(NA, c(n.iter, N))
for (i in 1:n.iter)
    effect[i,] <- beta.post[i,2] + beta.post[i,4] * pre_test
avg.effect <- rowMeans(effect)
print(c(mean(avg.effect), sd(avg.effect)))
```

```
## [1] 1.8127455 0.6595629
```