[From Sarina]: Basically, Ashley says that in a single marginal model, the ATT is forced to equal the ATE (in the absence of interaction terms) because you are forcing the coefficients to be equal across exposure groups. For example if the coefficient for age is 3, then it is 3 for both the exposed and unexposed groups.

[From David]: My break in your paragraph here. Ashley's statement is not correct for non-linear models (e.g., logistic regression) due to non-collapsibility. This is illustrated in the attached code. You will in general get different answers for ATT and ATE even in a single outcome model with no interactions between treatment and covariates.

The Attached Code

Here is David's code, interspersed with my comments and thoughts

```r
# simulate data from linear/logistic models
n <- 1000
W <- runif(n)
A <- rbinom(n, 1, plogis(W))
binary_Y <- rbinom(n, 1, plogis(A + W + A*W))
continuous_Y <- A + W + A*W +rnorm(n)
data <- data.frame(W, A, binary_Y, continuous_Y)

# these are used throughout to compute g-comp
data_Ais1 <- data; data_Ais1$A <- 1
data_Ais0 <- data; data_Ais0$A <- 0


# case 1, single linear regression, no interactions
outcome_model <- glm(
    continuous_Y ~ A + W,
    data = data,
    family = gaussian()
)

pred_Ais1 <- predict(
    outcome_model, newdata = data_Ais1, type = "response"
)
pred_Ais0 <- predict(
    outcome_model, newdata = data_Ais0, type = "response"
)

# E[Y(1)] estimate (for ATE)
EY1_hat <- mean(pred_Ais1)
# E[Y(0)] estimate (for ATT)
EY0_hat <- mean(pred_Ais0)
# E[Y(1) | A = 1] estimate (for ATT)
EY1_Ais1_hat <- mean(pred_Ais1[A == 1])
# E[Y(0) | A = 1] estimate (for ATT)
EY0_Ais1_hat <- mean(pred_Ais0[A == 1])

ATE_hat <- EY1_hat - EY0_hat
ATT_hat <- EY1_Ais1_hat - EY0_Ais1_hat

# equivalent up to rounding error
ATE_hat - ATT_hat
```

```
## [1] 0
```

Yes, the above results are as I expected, and I might say that these are forced to be equivalent. That is, even though the true ATT and ATE are different, we estimate them to be the same here **because** we didn't include any interactions between A and W when we estimated the model. This is a form of misspecification bias.

```r
# case 2, single linear regression, interactions
outcome_model <- glm(
    continuous_Y ~ A*W,
    data = data,
    family = gaussian()
)

pred_Ais1 <- predict(
    outcome_model, newdata = data_Ais1, type = "response"
)
pred_Ais0 <- predict(
    outcome_model, newdata = data_Ais0, type = "response"
)

# E[Y(1)] estimate (for ATE)
EY1_hat <- mean(pred_Ais1)
# E[Y(0)] estimate (for ATT)
EY0_hat <- mean(pred_Ais0)
# E[Y(1) | A = 1] estimate (for ATT)
EY1_Ais1_hat <- mean(pred_Ais1[A == 1])
# note that this is EXACTLY THE SAME (up to rounding error)
# as the mean of Y in the treated (because outcome_model above
# contains an intercept)
EY1_Ais1_hat - mean(continuous_Y[A == 1])
```

```
## [1] 1.332268e-14
```

```r
# E[Y(0) | A = 1] estimate (for ATT)
EY0_Ais1_hat <- mean(pred_Ais0[A == 1])

ATE_hat <- EY1_hat - EY0_hat
ATT_hat <- EY1_Ais1_hat - EY0_Ais1_hat

# not equivalent
ATE_hat - ATT_hat
```

```
## [1] -0.01458256
```

Re above, again, as expected. The true ATT and ATE are different, and we get different results for them when we fit models that allow for these differences to occur.

```r
# case 3, linear regression for each treatment arm
# the same as a model where every term is interacted with treatment
# i.e., EXACTLY the same as case 2 above
outcome_model_Ais1 <- glm(
    continuous_Y ~ W,
    data = data[data$A == 1,],
    family = gaussian()
)

pred_Ais1 <- predict(
    outcome_model_Ais1, newdata = data, type = "response"
)

outcome_model_Ais0 <- glm(
    continuous_Y ~ W,
    data = data[data$A == 0,],
    family = gaussian()
)

pred_Ais0 <- predict(
    outcome_model_Ais0, newdata = data, type = "response"
)

# E[Y(1)] estimate (for ATE)
EY1_hat_two_models <- mean(pred_Ais1)
# E[Y(0)] estimate (for ATT)
EY0_hat_two_models <- mean(pred_Ais0)
# E[Y(1) | A = 1] estimate (for ATT)
EY1_Ais1_hat_two_models <- mean(pred_Ais1[A == 1])
# E[Y(0) | A = 1] estimate (for ATT)
EY0_Ais1_hat_two_models <- mean(pred_Ais0[A == 1])

ATE_hat_two_models <- EY1_hat_two_models - EY0_hat_two_models
ATT_hat_two_models <- EY1_Ais1_hat_two_models - EY0_Ais1_hat_two_models

# not equivalent
ATE_hat_two_models - ATT_hat_two_models
```

```
## [1] -0.01458256
```

```r
# but exactly the same (up to rounding error) as single model with trt * covariate interactions
ATE_hat_two_models - ATE_hat
```

```
## [1] 1.110223e-15
```

```r
ATT_hat_two_models - ATT_hat
```

```
## [1] 8.881784e-16
```

Re above, this is a great illustration. Again, as expected.

Now let's look at the code when using a logistic regression model. Before running code, let's emphasize that **we are fitting a logistic regression model (noncollapsibility) BUT we are estimating our effects on the risk difference scale (technically, strictly collapsible).**

```
## now repeat for logistic regression
# case 1, single logistic regression, no interactions
outcome_model <- glm(
    binary_Y ~ A + W,
    data = data,
    family = binomial()
)

pred_Ais1 <- predict(
    outcome_model, newdata = data_Ais1, type = "response"
)
pred_Ais0 <- predict(
    outcome_model, newdata = data_Ais0, type = "response"
)

# E[Y(1)] estimate (for ATE)
EY1_hat <- mean(pred_Ais1)
# E[Y(0)] estimate (for ATT)
EY0_hat <- mean(pred_Ais0)
# E[Y(1) | A = 1] estimate (for ATT)
EY1_Ais1_hat <- mean(pred_Ais1[A == 1])
# E[Y(0) | A = 1] estimate (for ATT)
EY0_Ais1_hat <- mean(pred_Ais0[A == 1])

ATE_hat <- EY1_hat - EY0_hat
ATT_hat <- EY1_Ais1_hat - EY0_Ais1_hat

# NOT equivalent due to non-collapsibility
ATE_hat - ATT_hat
```

```
## [1] 0.00414348
```

```
ATE_hat
```

```
## [1] 0.2624782
```

```
ATT_hat
```

```
## [1] 0.2583347
```

```
ATE_hat_OR <- (EY1_hat/(1 - EY1_hat))/(EY0_hat/(1 - EY0_hat))
ATT_hat_OR <- ((EY1_Ais1_hat)/(1 - EY1_Ais1_hat))/(EY0_Ais1_hat/(1 - EY0_Ais1_hat))

ATE_hat_OR - ATT_hat_OR
```

```
## [1] -0.005205947
```

```
ATE_hat_OR
```

```
## [1] 4.749602
```

```
ATT_hat_OR
```

```
## [1] 4.754808
```