If you're a Bayesian you can do everything that God forbids (- Willem Heiser)

Joachim Vandekerckhove

Unorthodox things you can do if you're a Bayesian

- Accumulate evidence for the absence of an effect

Unorthodox things you can do if you're a Bayesian

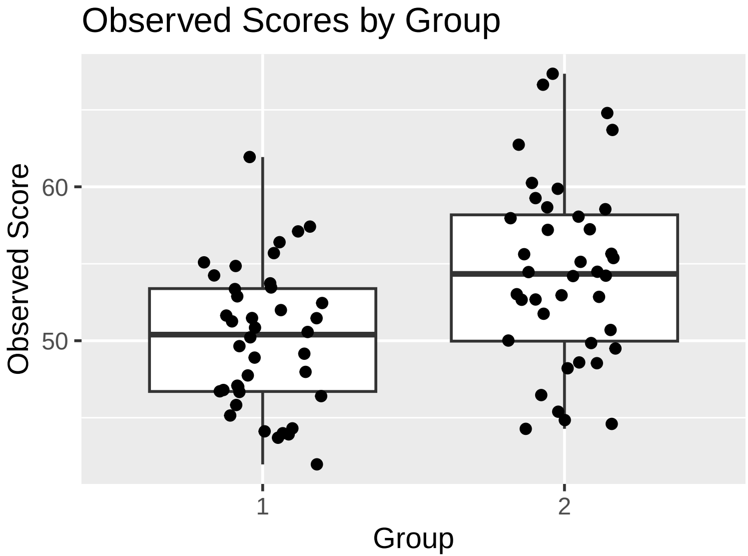
- Accumulate evidence for the absence of an effect
- Draw valid conclusions with almost no data

Unorthodox things you can do if you’re a Bayesian

- Accumulate evidence for the absence of an effect
- Draw valid conclusions with almost no data
- Say things about means without knowing much about the basic units

Means without knowing much about the basic units

```r
N <- 80
group <- rep(1:2, each = N/2) # Make two groups
# True score for each participant depends on group
true_score <-
  rnorm(N,
        mean = rep(c(50, 55), each = N/2),
        sd   = 2)
# Observations are noisy
observed_score <- true_score +
  rnorm(N, mean = 0, sd = 5)

data <- data.frame(participant = 1:N,
                   group = factor(group),
                   true_score, observed_score)
```


Observed Scores by Group
Observed Score
60
50
1
2
Group

```
model_string <- "
model {
  for (i in 1:N) {
    # Likelihood
    observed_score[i] ~ dnorm(true_score[i], tau)
    # Hierarchical model
    true_score[i]     ~ dnorm(mu[group[i]], tau_true)
  }
  diff_mu ~ dnorm(0, 0.001)
  mu[1]   ~ dunif(0, 100)
  mu[2] <- mu[1] + diff_mu
  tau      ~ dgamma(0.1, 0.1)
  tau_true ~ dgamma(0.1, 0.1)
}
"
```

```r
data_list <- list(
  N = N,
  observed_score = data$observed_score,
  group = as.numeric(data$group)
)

# Initial values
inits <- function() {
  list(
    tau        = rgamma(1, 0.1, 0.1),
    tau_true   = rgamma(1, 0.1, 0.1),
    true_score = runif(N, 0, 100)
  )
}
```

```r
jags_model <- jags.model(textConnection(model_string),
                         data     = data_list,
                         inits    =     inits,
                         n.chains =         3 ,
                         n.adapt  =      1000 )
```

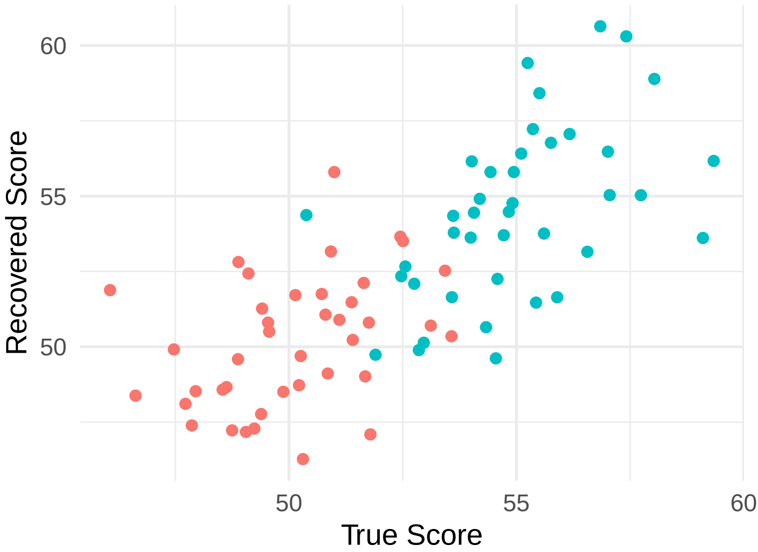
```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 80
##    Unobserved stochastic nodes: 84
##    Total graph size: 250
##
## Initializing model
```

```r
update(jags_model, n.iter = 1000)
samples <- coda.samples(jags_model,
                        variable.names = c("diff_mu",
                                           "true_score"),
                        n.iter = 5000)
```

```r
# Summarize the results
summary_diff_mu <- summary(samples)$
  statistics["diff_mu", ]
print(summary_diff_mu)
```

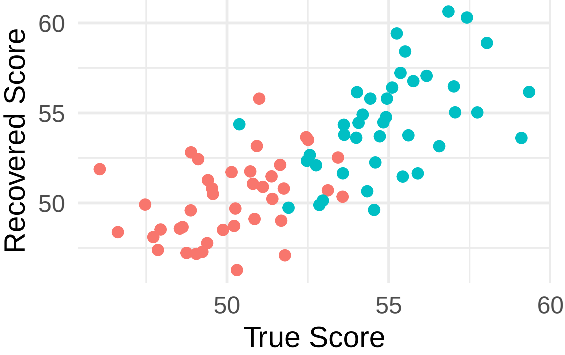
```
##           Mean             SD       Naive SE Time-series SE
##    4.314157098    1.203333239    0.009825175    0.058018522
```

```r
# Extract true scores from the posterior samples
true_scores_posterior <- as.data.frame(as.matrix(samples)) %>%
  select(starts_with("true_score")) %>%
  apply(2, mean)

# Add the recovered true scores to the data frame
data$recovered_true_score <- true_scores_posterior

# Scatter plot of true scores vs recovered true scores
p2 <- ggplot(data, aes(x = true_score, y = recovered_true_score,
                       color = group)) +
  geom_point() +
  labs(x = "True Score", y = "Recovered Score") +
  theme_minimal() +
  theme(legend.position = "none")
```


60
55
50
Recovered Score
50
55
60
True Score

Hierarchical recovery beats individual recovery

The difference between groups is well recovered as 4.3141571 ± 1.2033332, even though the true scores within each group are poorly recovered.



Perform inference with no actual data

Suppose these were scores from a test (and suppose the two groups are an "on-track" group and an "advanced" group). One student from the advanced group missed class. What do we know about student $N + 1$?

```r
model_string <- "
model {
  for (i in 1:N) {
    # Likelihood
    observed_score[i] ~ dnorm(true_score[i], tau)
    # Hierarchical model
    true_score[i]     ~ dnorm(mu[group[i]], tau_true)
  }
  diff_mu ~ dnorm(0, 0.001)
  mu[1]   ~ dunif(0, 100)
  mu[2] <- mu[1] + diff_mu
  tau       ~ dgamma(0.1, 0.1)
  tau_true ~ dgamma(0.1, 0.1)

  true_score[N+1] ~ dnorm(mu[2], tau_true)
  observed_score[N+1] ~ dnorm(true_score[N+1], tau)
}
```

```r
data_list <- list(
  N = N,
  observed_score = c(data$observed_score, NA),
  group = as.numeric(data$group)
)

# Initial values
inits <- function() {
  list(
    tau        = rgamma(1, 0.1, 0.1),
    tau_true   = rgamma(1, 0.1, 0.1),
    true_score = runif(N+1, 0, 100)
  )
}
```

```r
jags_model <- jags.model(textConnection(model_string),
                         data     = data_list,
                         inits    =     inits,
                         n.chains =         3 ,
                         n.adapt  =      1000 )
```

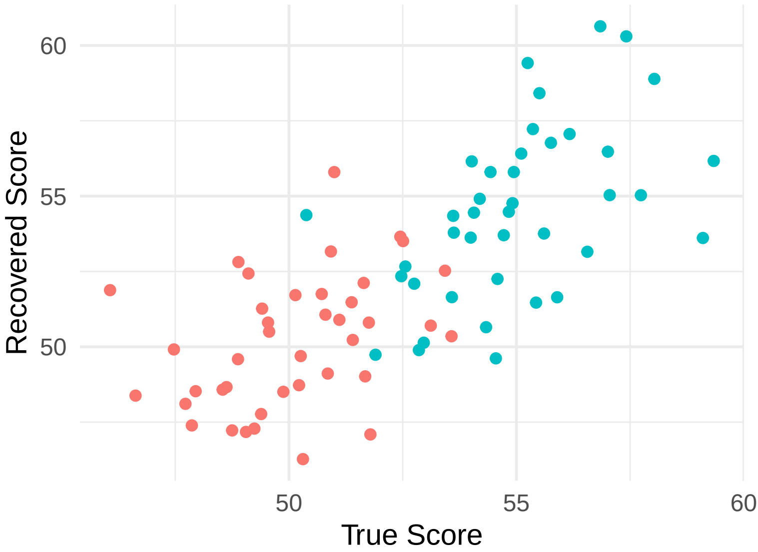
```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 80
##    Unobserved stochastic nodes: 86
##    Total graph size: 252
##
## Initializing model
```

```r
update(jags_model, n.iter = 1000)
samples <- coda.samples(jags_model,
                        variable.names = c("diff_mu",
                                           "true_score",
                                           "observed_score"),
                        n.iter = 5000)
```

```r
# Summarize the results
summary_new <- summary(samples)$
  statistics[c("true_score[81]","observed_score[81]"),
             c("Mean", "SD")]
print(summary_new)
```

```
##                        Mean       SD
## true_score[81]     54.41803 3.645873
## observed_score[81] 54.44141 5.458713
```

```r
# Extract true scores from the posterior samples
true_scores_posterior_new <- as.data.frame(as.matrix(samples)) %>%
  select(starts_with("true_score[81]"))
```


60
55
50
Recovered Score
50
55
60
True Score