Project 4 Causal Inference

Group 7

Daizy Lam, Qiao Li, Chuanchuan Liu, Yingyao Wu, Serena Yuan

Causal Effects

Suppose we have a random sample of size N from a large population. For each unit i in the random sample, we use $T_i \in \{0,1\}$ to denote whether the unit i received the treatment of interest. Let $Y_i(0)$ indicates the outcome that the unit i was under control while $Y_i(1)$ indicates the outcome under treatment. For unit i the treatment effect is $Y_i(1) - Y_i(0)$. We are interested in the average effect of the treatment in the whole population (to simplify the notation, we suppress subscript i for unit):

$$\Delta = E(Y_1 - Y_0) = E(Y_1) - E(Y_0)$$

Introduction

In this project, our goal is to compare Regression Estimate, Stratification, and Weighted Regression where PS is based on L2 penalized logistic regression. We implement and evaluate these three causal inference algorithms on treatment datasets (low dimensional/ high dimensional). ATE estimation is used to evaluate and compare the efficiency of the causal inference algorithms.

Load Data

```r
lowDim <- read.csv("../data/lowDim_dataset.csv")
highDim <- read.csv("../data/highDim_dataset.csv")
```

Regression Estimate (with no need of propensity score)

This algorithm builds regression model fit to control groups and treatment groups, and then makes predictions based on the model in order to calculate ATE.

$$\hat{\Delta}(reg) = \frac{1}{N}\sum(\hat{m}_1(X_i) - \hat{m}_0(X_i))$$

Algorithm

```r
Regression_Estimation <- function(df){

  start_time <- Sys.time()
  # regression model for control groups (A=0)
  model0 <- glm(formula=Y~., data=subset(df[which(df$A==0),],select=-c(A)))
  # regression model for treatment groups (A=1)
  model1 <- glm(formula=Y~., data=subset(df[which(df$A==1),],select=-c(A)))

  X = subset(df, select=-c(Y,A)) #input data for prediction

  #prediction using model0
  Y0 <- predict(model0, newdata=X)
  #prediction using model1
  Y1 <- predict(model1, newdata=X)

  # calculate ATE
  ATE <- mean(Y1-Y0)

  # calculate running time
  end_time <- Sys.time()
  running_time <- end_time - start_time

  return (list(ATE=ATE, running_time=running_time))
}
```

Summary

- ATE

```r
calc_ate_low <- Regression_Estimation(lowDim)$ATE
calc_ate_high <- Regression_Estimation(highDim)$ATE
cat("ATE for Low Dimension Data is", calc_ate_low, "\n")
```

```
## ATE for Low Dimension Data is 2.125138
```

```r
cat("ATE for High Dimension Data is", calc_ate_high)
```

```
## ATE for High Dimension Data is -57.42659
```

- Running Time

```r
running_time_low <- Regression_Estimation(lowDim)$running_time
running_time_high <- Regression_Estimation(highDim)$running_time
cat("Running Time for Low Dimension Data is", running_time_low, "sec.","\n")
```

```
## Running Time for Low Dimension Data is 0.00982213 sec.
```

```r
cat("Running Time for High Dimension Data is", running_time_high, "sec.")
```

```
## Running Time for High Dimension Data is 0.2331991 sec.
```

- Performance

```r
true_ate_low = 2.0901
true_ate_high = -54.8558
performance_low <- abs(true_ate_low - calc_ate_low)
performance_high <- abs(true_ate_high - calc_ate_high)
cat("Performance for Low Dimension Data is", performance_low, "\n" )
```

```
## Performance for Low Dimension Data is 0.03503807
```

```r
cat("Performance for High Dimension Data is", performance_high)
```

```
## Performance for High Dimension Data is 2.570787
```

- Result

```r
RE <- matrix(c(calc_ate_high,calc_ate_low,
               running_time_high,running_time_low,
               performance_high,performance_low),
             ncol = 3, nrow=2)
colnames(RE) = c("ATE","Computational Efficiency","Performance")
rownames(RE) = c("High Dimension", "Low Dimension")
RE
```

```
##                       ATE Computational Efficiency Performance
## High Dimension -57.426587               0.23319912  2.57078732
## Low Dimension    2.125138               0.00982213  0.03503807
```

Propensity Score

L2 Penalized Logistic Regression

The logistic regression model represents the class-conditional probabilities through a linear function of the predictors:

$$logit[Pr(T = 1|X)] = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p$$

$$Pr(T = 1|X) = \frac{1}{1 + e^{-(\beta_0+\beta_1 x_1+\cdots+\beta_p x_p)}}$$

To avoid overfitting of the logistic regression model, we introduce regularization term to decrease the model variance in the loss function Q.

In order to achieve this, we modifying the loss function with a penalty term which effectively shrinks the estimates of the coefficients. The regularization term is the "L2 norm":

$$Q = -\frac{1}{n}\sum[y_i(\beta_0 + \beta_1 x_1 + ... + \beta_p x_p) + log(1 + \exp(\beta_0 + \beta_1 x_1 + ... + \beta_p x_p))] + \lambda \sum \beta_j^2$$

```r
propensity_score <- function(data){
  #L2 penalized logistic regression was used to estimate propensity scores
  #Input:
  #data - the dataframe we want to evaluate
  #Return:
  #p_score - propensity score
  #Run time for calculating p_score
  set.seed(0)
  start_time <- Sys.time()
  A<-data$A
  Y<-data$Y

  glm <- cv.glmnet(data.matrix(data[,-c(1,2)]), A, family = "binomial", alpha = 0)
  p_score <- predict(glm$glmnet.fit,
                     s = glm$lambda.min,
                     newx = data.matrix(data[,-c(1,2)]),
                     type = "response")

  end_time <- Sys.time()
  running_time = end_time - start_time
  return(list(p_score,running_time))
}
```

Propensity Scores

```
# propensity scores
ps_high = propensity_score(highDim)[[1]]
ps_low = propensity_score(lowDim)[[1]]
# running times
rt_high = propensity_score(highDim)[[2]]
rt_low = propensity_score(lowDim)[[2]]
```

Stratification

Stratification (sometimes referred to as subclassification) is commonly used in observational studies to control for systematic differences between the control and treated groups. This technique consists of grouping subjects into strata determined by observed background characteristics. (D'Agostino, 1998)

We will estimate the Average Treatment Effect (ATE) using stratification based on L2 penalized Logistic Regression propensity scores. The algorithm will be based on the following equation:

$$\hat{\Delta}_S = \sum_{j=1}^{K} \frac{N_j}{N} \{N_{1j}^{-1} \sum_{i=1}^{N} T_i Y_i I(\hat{e}_i \in \hat{Q}_j) - N_{0j}^{-1} \sum_{i=1}^{N} (1 - T_i) Y_i I(\hat{e}_i \in \hat{Q}_j)\}$$

where K is the number of strata, N_j is the number of individuals in stratum j, N_{1j} is the number of "treated" individuals in stratum j and N_{0j} is the number of "controlled" individuals in stratum j. $\hat{Q}_j = (\hat{q}_{j-1}, q_j]$ is the interval from (j-1)th sample quantile to jth sample quantile of the estimated propensity scores. (Lunceford and Davidian, 2004)

Estimate Average Treatment Effect (ATE)

We first determine the number of strata K and create equally spaced intervals starting from 0 to 1. In the example below we are using K = 5, as suggested by Rosenbaum and Rubin who indicated that creating five strata removes 90 per cent of the bias due to the stratifying variable or covariate (D'Agostino, 1998), and the resulting intervals to be constructed into quantiles would be (0, 0.2, 0.4, 0.6, 0.8, 1).

Then we form K = 5 strata according to the sample quantiles of the $\hat{e}_i$ (estimated propensity scores), i = 1, . . . , N (sample size/number of observations), where the jth sample quantile $\hat{q}_j$, for j = 1, . . . , K, is such that the proportion of $\hat{e}_i \leq \hat{q}_j$ is roughly j/K, $\hat{q}_0 = 0$ and $\hat{q}_K = 1$. (Lunceford and Davidian, 2004)

```
K = 5 # number of strata
q = seq(0, 1, by = 1/K) # sample quantiles to be constructed below
```

Within each stratum, we subset the observations for the specific strata, i.e. observations whose propensity scores fall in the interval $(q_{j-1}, q_j]$ for the jth strata. We then split the subsetted data set into "treated" and "controlled" groups using the binary treatment indicator 'A', and calculate a weighted sum of the outcome 'Y' for each group within the stratum.

Then the estimated ATE would be the summation over all K strata of the weighted sum of the difference on the "treated" and "controlled" groups as described above.

```r
stratification = function(df, ps){
  tm_start = Sys.time()
  # split into K strata
  stratum = rep(NA, length(q))
  for (i in 1:length(q)){
    stratum[i] = quantile(ps, q[i])
  }
  # ATE
  ate_strata = rep(NA, K)
  for (j in 1:K){
    # select observations whose propensity score is within (q_{j-1},q_j]
    curr.obs = df[which(stratum[j] < ps & ps <= stratum[j+1]),]

    # subset/select treatment and control groups
    treatment = curr.obs[curr.obs$A == 1,]
    control = curr.obs[curr.obs$A == 0,]

    # calculate sum within stratum
    treatment_sum = sum(treatment$A * treatment$Y) / nrow(treatment)
    control_sum = sum((1 - control$A) * control$Y) / nrow(control)
    ate_strata[j] = (nrow(curr.obs)/nrow(df)) * (treatment_sum - control_sum)
  }
  output = sum(ate_strata)
  tm_end = Sys.time()
  cat("The estimated ATE is", output, "\n")
  cat("Run time is", (tm_end - tm_start), "s")
  return(list(output, (tm_end - tm_start)))
}
```

High Dimension Data Set

```r
strata_high = stratification(highDim, ps_high)
```

```
## The estimated ATE is -57.93487
## Run time is 0.02276015 s
```

Low Dimension Data Set

```r
strata_low = stratification(lowDim, ps_low)
```

```
## The estimated ATE is 2.376539
## Run time is 0.003199816 s
```

Performance

- True ATE for High Dimension Data Set: -54.8558

- True ATE for Low Dimension Data Set: 2.0901

```r
true_ate_high = -54.8558
true_ate_low = 2.0901
cat("Performance for High Dimension Data is", abs(true_ate_high - strata_high[[1]]), "\n")
```

```
## Performance for High Dimension Data is 3.079069
```

```r
cat("Performance for Low Dimension Data is", abs(true_ate_low - strata_low[[1]]))
```

```
## Performance for Low Dimension Data is 0.2864389
```

Summary

```
##                 Performance Algorithm.Run.Time Propensity.Score.Run.Time
## High Dimension   3.0790690        0.022760153                  5.922544
## Low Dimension    0.2864389        0.003199816                  0.228431
```

Weighted Regression

Weighted least square estimation of regression function:

$$Y_i = \alpha_0 + \tau \cdot T_i + \alpha_1' \cdot Z_i + \alpha_2'(Z_i - \bar{Z}) \cdot T_i + \epsilon_i$$

where the weights are of inverse propensity weighting. Z_i are a subset of covariates X_i with sample average $\bar{Z}$, and τ is an estimate for ATE.

We will use the following method to select Z:

Estimate linear regressions $Y_i = \beta_{k0} + \beta_{k1}T_i + \beta_{k2} \cdot X_{ik} + \epsilon_i$

The weights are given by $\hat{\omega}(t,x) = \frac{t}{\hat{e}(x)} + \frac{1-t}{1-\hat{e}(x)}$.

Then estimate K linear regressions of type $Y_i = \beta_{k0} + \beta_{k1} \cdot T_i + \beta_{k2} \cdot X_{ik} + \epsilon_i$ and then calculate the t-statistic for the test of null hypothesis that the slope $\beta_{k2}$is equal to zero. We select the subset where the t-statistic is significant. So select for Z all covariates with a t-statistic which is larger in absolute value than t_{reg}. Then, in the final regression, include all covariates in this subset, and use the weights as the values $\hat{\omega}(t,x)$.

```r
#assign variables for data frame
X_low <- data.matrix(lowDim[,-c(1,2)])
A_low <- data.matrix(lowDim[,2])
X_high <- data.matrix(highDim[,-c(1,2)])
A_high <- data.matrix(highDim[,2])
```

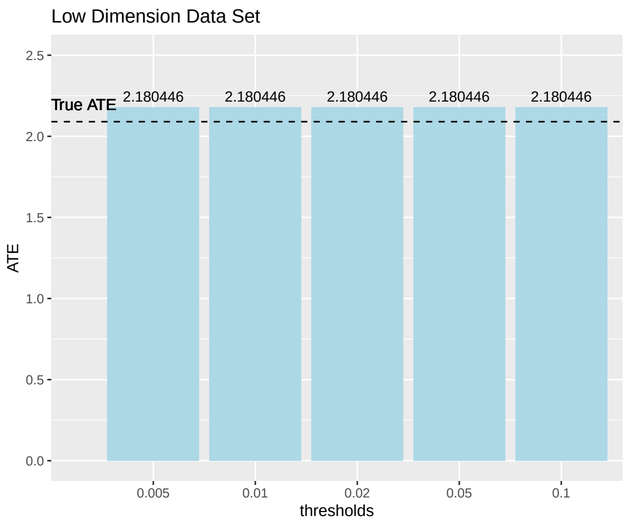
```r
Weighted_Regression <- function(X,A,df, threshold, ps){
  start_time <- Sys.time()
  # Finding weights
  t<- as.numeric(A)
  wt<- t/(ps) + (1-t)/(1-ps)

  # Estimate linear regression
  Y <- df$Y
  model<- lm(Y~., data = df)
  feature_z <- summary(model)$coef[,4][-c(1:2)]<threshold
  Z <- as.data.frame(cbind(A,X[,feature_z]))
  Z<-sapply(Z, as.numeric)

  #Weighted Regression
  weighted_reg <- lm(Y ~ Z, weights = wt)

  #coef of T is an estimate for ATE
  ATE<- coef(weighted_reg)[2]
  end_time <- Sys.time()
  running_time <- end_time - start_time
  cat("The estimated ATE is", ATE, "\n")
  cat("Run time is", running_time , "s \n")
  return (list(ATE=ATE, running_time=running_time))

}
```

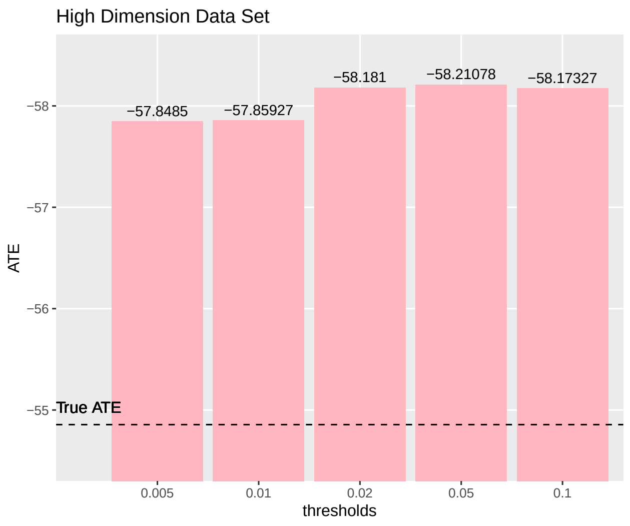
Different Thresholds for Best ATE

```r
thresholds <- c(0.1, 0.05, 0.02, 0.01, 0.005)

ate_scores_low <- rep(NA,length(thresholds))
ate_scores_high <- rep(NA,length(thresholds))

true_ate_high = -54.8558
true_ate_low = 2.0901

count = 1
for (t in thresholds) {
  wreg_low = Weighted_Regression(X_low, A_low, lowDim, t, ps_low)
  wreg_high = Weighted_Regression(X_high, A_high, highDim, t, ps_high)
  diff_low = abs(true_ate_low - wreg_low[[1]])
  diff_high = abs(true_ate_high - wreg_high[[1]])
  ate_scores_low[count] = diff_low
  ate_scores_high[count] = diff_high
  count = count + 1
}
```

```
## The estimated ATE is 2.180446
## Run time is 0.01921916 s
## The estimated ATE is -58.17327
## Run time is 0.09485197 s
## The estimated ATE is 2.180446
## Run time is 0.004462004 s
## The estimated ATE is -58.21078
## Run time is 0.08925104 s
## The estimated ATE is 2.180446
## Run time is 0.01634002 s
## The estimated ATE is -58.181
## Run time is 0.08331108 s
## The estimated ATE is 2.180446
## Run time is 0.005799055 s
## The estimated ATE is -57.85927
## Run time is 0.08955693 s
## The estimated ATE is 2.180446
## Run time is 0.004212141 s
## The estimated ATE is -57.8485
## Run time is 0.0906539 s
```

Visualization





Set threshold to the optimal value:

```
opt_low <- which.min(ate_scores_low)
opt_high <- which.min(ate_scores_high)
threshold_low <- thresholds[opt_low]
threshold_high <- thresholds[opt_high]
```

```
Weighted_Regression_low = Weighted_Regression(X_low,A_low,lowDim,threshold_low, ps_low)
```

```
## The estimated ATE is 2.180446
## Run time is 0.004544973 s
```

```
Weighted_Regression_high = Weighted_Regression(X_high,A_high,highDim, threshold_high, ps_high)
```

```
## The estimated ATE is -57.8485
## Run time is 0.0865159 s
```

```
true_ate_high = -54.8558
true_ate_low = 2.0901
cat("Performance for High Dimension Data is",
    abs(true_ate_high - Weighted_Regression_high[[1]]), "\n")
```

```
## Performance for High Dimension Data is 2.992701
```

```
cat("Performance for Low Dimension Data is",
    abs(true_ate_low -Weighted_Regression_low[[1]]))
```

```
## Performance for Low Dimension Data is 0.09034573
```

Summary

```
##                  Performance Algorithm.Run.Time Propensity.Score.Run.Time
## High Dimension  2.99270117        0.086515903                  5.922544
## Low Dimension   0.09034573        0.004544973                  0.228431
```

Model Comparison

For evaluating the performance of the three algorithms, we computed their absolute errors (absolute difference of the estimated and true ATEs), and for evaluating the computational efficiency we look at the system running time.

Performance: absolute error of the true and estimated ATE

Computational Efficiency: run time in seconds

Results in Tables

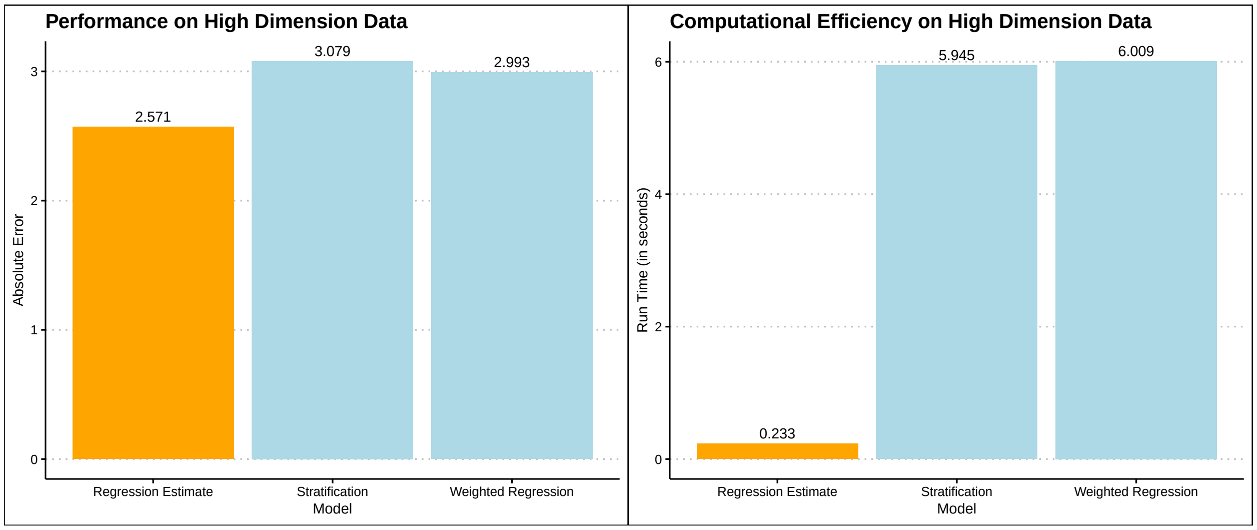
High Dimension Dataset

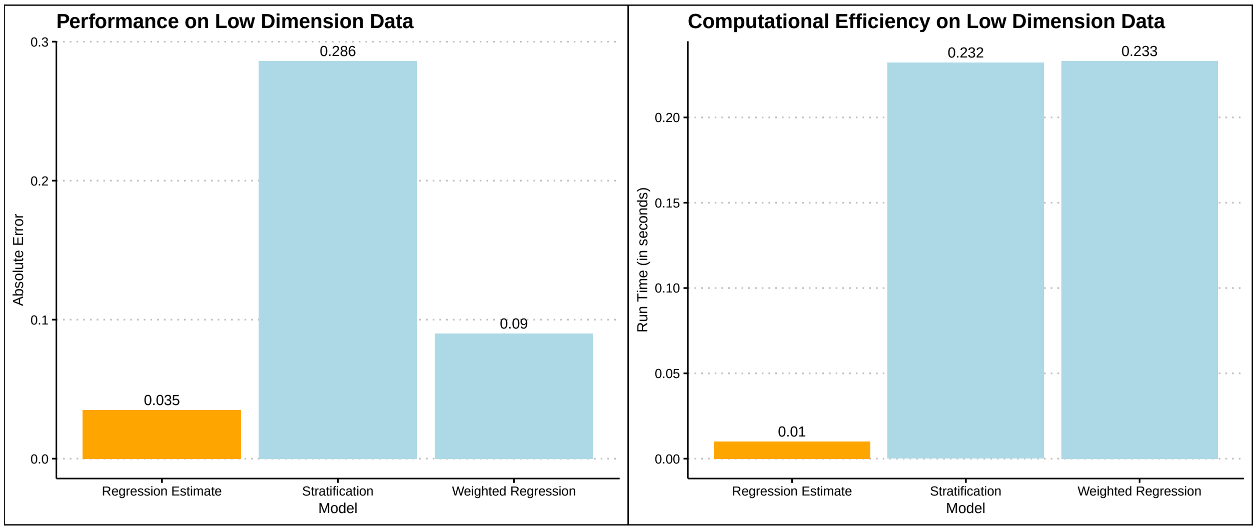
```
##                             ATE Absolute Error Algorithm Run Time
## Stratification      -57.93487       3.079069         0.02276015
## Regression Estimate -57.42659       2.570787         0.23319912
## Weighted Regression -57.84850       2.992701         0.08651590
##                     Propensity Score Run Time
## Stratification                       5.922544
## Regression Estimate                        NA
## Weighted Regression                  5.922544
```

Low Dimension Dataset

```
##                            ATE Absolute Error Algorithm Run Time
## Stratification      2.376539     0.28643894        0.003199816
## Regression Estimate 2.125138     0.03503807        0.009822130
## Weighted Regression 2.180446     0.09034573        0.004544973
##                     Propensity Score Run Time
## Stratification                       0.228431
## Regression Estimate                        NA
## Weighted Regression                  0.228431
```

Results in Visualization





Conclusion

In conclusion, we explored the three causal inference algorithms (Regression Estimate, Stratification and Weighted Regression with PS based on L2 penalized logistic regression) on two data sets (low dimensional/ high dimensional) and computed the estimated ATE for each model. We evaluated the algorithms performance by calculating the absolute differences between the true ATE and estimated ATE and compare the computational efficiency by comparing the system run time.

We identified the regression estimate has the best performance on both high dimensional and low dimensional data with the lowest absolute error with fast system run time in compare to the other algorithms.

References

D'Agostino, R. B. (1998). Propensity score methods for bias reduction in the comparison of a treatment to a non-randomized control group. Statistics in Medicine, 17(19), 2265-2281. doi:10.1002/(sici)1097-0258(19981015)17:193.0.co;2-b

Lunceford, J. K.; Davidian, M. (2004). Stratification and weighting via the propensity score in estimation of causal treatment effects: A comparative study. Statistics in Medicine, 23(19), 2937-2960. doi:10.1002/sim.1903

Friedman, Jerome, Trevor Hastie, and Rob Tibshirani. 2010. "Regularization Paths for Generalized Linear Models via Coordinate Descent." Journal of Statistical Software 33 (1): 1.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman. 2009. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer Science & Business Media.

Westreich, Daniel, Justin Lessler, and Michele Jonsson Funk. 2010. "Propensity Score Estimation: Neural Networks, Support Vector Machines, Decision Trees (Cart), and Meta-Classifiers as Alternatives to Logistic Regression." Journal of Clinical Epidemiology 63 (8): 826–33.