Question 2: Semi-Parametric, Structural Estimation

Suppose that the utility of the wife over participation and consumption follows the functional form:

$$U(C, P; \epsilon) = C + x'\alpha(1 - P) + \beta(1 - P)C + \epsilon(1 - P)$$

and the wage equation is

$$w(z, \xi) = z'\gamma + \xi$$

As shown in class, the observed wage equation can be written as

$$w(z, \xi) = z'\gamma + M(\Pr(P = 1 \mid y, z, x)) + u$$

Using the sub-sample of married women between the ages of 25 and 55, implement the following estimation steps

(a) Non-parametrically estimate $\Pr(P = 1 \mid y, z, x)$ using a kernel regression where z includes completed education and age and x includes a constant, age and current number of children

For this question, we have decided to use **R** with the **np** package and manually for the calculation. In R there is the package **np** written by Jeffrey S. Racine and Tristen Hayfield for the estimation of nonparametric (and semiparametric) kernel methods with built-in function we use for estimation. We also use **R** to manually calculate the results.

Initially we start by subsetting the data for women with the characteristics given by the question and we create a new definition of education to make it easier to work.

```
# Question 2 ------------------------------------------------------------

# Utility function
# U(c, P; e) = c + x'a*(1-P) + b(1-P)c + e(1-P)
# c: consumption
# Participation index (=1 if works)
# x: vector of covariates
# r threshold value


# 2 - a) Estimation Particiaption probabilities ---------------------------
# wage equation:
# w(z,eta) = z'y + M(Pr(P =1| y,z,x )) + u

# Subseting for married women between 25-55
# We also  subset to not include NA or negative
```

```r
# values for each variable
  m_women_25_55 <- data_ps1 %>%
    filter(age %in% 25:55, marst %in% c(1,2), hhincome >=0,
           !is.na(non_labor_income),
           !is.na(age),
           !is.na(nchild)) %>%
    mutate(n_child = as.numeric(nchild),
           educ = as.numeric(educ)) %>%
    filter(educd > 1 & educd != 999) %>%  # drop if educd <= 1 or educd == 999
    mutate(education = -1) %>%  # create new column 'education' with -1 as initial value
    mutate(education = case_when(
      educd == 2 ~ 0,
      educd == 14 ~ 1,
      educd == 15 ~ 2,
      educd == 13 ~ 2.5,
      educd == 16 ~ 3,
      educd == 17 ~ 4,
      educd == 22 ~ 5,
      educd == 21 ~ 5.5,
      educd == 23 ~ 6,
      educd == 20 ~ 6.5,
      educd == 25 ~ 7,
      educd == 24 ~ 7.5,
      educd == 26 ~ 8,
      educd == 30 ~ 9,
      educd == 40 ~ 10,
      educd == 50 ~ 11,
      educd == 60 ~ 12,
      educd == 61 ~ 12,
      educd == 62 ~ 12,
      educd == 63 ~ 12,
      educd == 64 ~ 12,
      educd == 65 ~ 12,
      educd == 70 ~ 13,
      educd == 71 ~ 13,
      educd == 80 ~ 14,
      educd == 90 ~ 15,
      educd == 100 ~ 16,
      educd == 101 ~ 16,
      educd == 110 ~ 17,
      educd == 111 ~ 18,
```

```
      educd == 112 ~ 19,
      educd == 113 ~ 20,
      educd == 114 ~ 20,
      educd == 116 ~ 20,
      TRUE ~ education  # if none of the above conditions are true, keep existing value
    )) %>%
    filter(education != -1)
```

Since we do not have enough processing power, we will use the sub samples taken randomly without replacement. We will take two samples of size 1000, 5000.

```
# Take a subsample  because my laptop does not have enough computing
# power
n_1 = 1000 # Size of subsample 1
n_2 = 5000 # Size of subsample 2


# First subsample n= 1000
m_women_25_55_n_1 <- m_women_25_55[sample(nrow(m_women_25_55), n_1),] %>%
                                        mutate(educ = as.factor(educ),
                                               age = as.numeric(age),
                                               nchild = n_child)

# Second subsample n= 5000
m_women_25_55_n_2 <- m_women_25_55[sample(nrow(m_women_25_55), n_2),] %>%
  mutate(educ = as.factor(educ),
         age = as.numeric(age),
         nchild = n_child)
```

Now, we finally calculate the optimal bandwidth using the linear cross-validation and using a Gaussian mutivariate kernel function. The function perform 5 iterations of the whole process in order to find the the best bandwidth.

```
# Step 1: Non-parametric estimation of Pr(P = 1 | y,z,x) ------------------
# a)  Non-parametrically estimate Pr(P = 1|y, z, x) using a kernel regression
  # where z includes completed education and age and x includes a constant,
  # age and current number of children and y is non labor income

# Bandwidth estimation with Linear Cross-Validation
bw_par_n1 <- npregbw(formula = labor_par ~ non_labor_income  + educ + age + nchild ,
                  data = m_women_25_55_n_1, regtype = "lc",
                  ckertype = "gaussian")
```

```r
bw_par_n2 <- npregbw(formula = labor_par ~ non_labor_income  + educ + age + nchild ,
                     data = m_women_25_55_n_2, regtype = "lc",
                     ckertype = "gaussian")
```

With the following code we get the results

```r
# Results
resultados_n1 <- npreg(bw_par_n1) # first sample
resultados_n2 <- npreg(bw_par_n2) # second sample
```

Table 1: Non-parametric estimation bandwidth of P using np Package - local constant

Sample	NL income h	Educ h	Age h	nchild h
1000 obs.	8837.935	0.817675	4.98267	0.8327307
5000 obs.	854086.6	0.5154519	3.6600053	2.836688

Now, we finally calculate the predicted probabilities of participation using the code chunk below. We also can see the table with descriptive statistics of predicted P. We note that falls in range 0 to 1, and mean of them are near 0.94 in every sample.

Table 2: Descriptive stats of the predicted P -np

Sample	Mean	Sd	min	max
1000 obs.	0.9419313	0.06002464	0	1
5000 obs.	0.9483481	0.02694107	0.9247883	0.9709707

Now, we do it again but manually. We will use the Epechnikov kernel and Silverman's Rule of Thumb.

We start by defining functions that will be used in this problem. We define the function to perform the epanechnikov_kernel that receives as input

```r
# Write auxiliary functions to manually calculate Nadaraya-Watson
# and the kernel estimator

# Function to calculate the multivariate Epanechnikov kernel
epanechnikov_kernel <- function(x, y, h) {
  d <- x - y
  t <- sqrt(diag(tcrossprod(d, solve(h) %*% d)))
  ifelse(t < 1, 0.75 * (1 - t^2), 0)
```

```
}

# define the multivariate Nadaraya-Watson estimator
nadaraya_watson <- function(X,y, H){
  # Number of observations
  n <- nrow(X)

  # Vector of predicted y
  y_hat <- rep(0, n)

  # Loop to calculate m(X - x_i)
  for (i in 1:n) {
    kernel_weights <- rep(0, n)
    for (j in 1:n) {
      kernel_weights[j] <- epanechnikov_kernel(X[j,], X[i,], H)
    }
    y_hat[i] <- sum(kernel_weights * y) / sum(kernel_weights)
  }

  # Print the estimated values of y
  y_hat
}


# Estimate the optimal bandwidth using Silverman's rule of thumb
Silverman <- function(X){
  n <- nrow(X) # Number of rows
  d <- ncol(X) # Number of columns
  std <- apply(X, 2, sd)  # Sd
  H <- diag(d) * std * (4 / (d + 2))^(1/(d + 4) ) * (n ^(-1*(1/(d+4))))
  return(H)
  } # Returns the diagonal matrix for bandwidth
```

Then, we define the vectors and matrices X, Y, Z and P (predicted participation probability) to represents the variables given by the question and add the suffix n1 or n2 to represent. See the comments in the code for a complete description of the name of each variable. Also note T is column bind of matrices X,Y,Z„ to represent observed labor particpation we use L and H is the Silverman's Bandwidth.

```
# Subset matrices
# X variables age education
```

```r
X_n1 <- as.matrix(cbind(const = rep(1, times = nrow(m_women_25_55_n_1)),m_women_25_55_n_1[
X_n2 <- as.matrix(cbind(const = rep(1, times = nrow(m_women_25_55_n_2)),m_women_25_55_n_2[

# Z variables completed education and age
Z_n1 <- as.matrix(m_women_25_55_n_1[, c("age", "education")])
Z_n2 <- as.matrix(m_women_25_55_n_2[, c("age", "education")])

# Y non labor income
Y_n1 <- as.vector(m_women_25_55_n_1[,"non_labor_income"])
Y_n2 <- as.vector(m_women_25_55_n_2[,"non_labor_income"])

# Matrix containing everything to estimate P
T_n1 <- as.matrix(cbind(X_n1[,-1], Z_n1, Y_n1))
T_n2 <- as.matrix(cbind(X_n2[,-1], Z_n2, Y_n2))

# Matrix of labor participation
L_n1 <- as.vector(m_women_25_55_n_1[,"labor_par"])
L_n2 <- as.vector(m_women_25_55_n_2[,"labor_par"])

# bandwith Matrix according to Silverman
H_n1 <- Silverman(T_n1)
H_n2 <- Silverman(T_n2)

# Estimation of P
P_n1 <- nadaraya_watson(X = T_n1, y = L_n1, H = H_n1)
P_n2 <- nadaraya_watson(X = T_n2, y = L_n2, H = H_n2)
```

We get the following bandwidths and descriptive statistics for the estimation

Table 3: Non-parametric estimation Silverman bandwidth of P using manual - local constant

Sample	Non-labor income h	Educ h	Age h	nchild h
1000 obs.	55729.29	1.515574	3.726225	0.5274278
5000 obs.	60356.78	1.211947	3.077618	0.4272879

Table 4: Descriptive stats of the predicted P - manual

Sample	Mean	Sd	min	max
1000 obs.	0.952886	0.02333114	0.90723	0.98723
5000 obs.	0.9488085	0.01610487	0.9025235	0.9697569

(b) Take your predicted working probabilities estimated in part (a) $\widehat{\Pr}(P = 1 \mid y, z, x)$ in the sample over which you implemented the non-parametric regression in part (a). Use Robinson's partial regression model to estimate γ and M. See pg. 62 in the Nonparametrics6.pdf file located in the Readings subfolder of the shared Dropbox folder.

In **R** we can easily perform the Robinson's Partial Regression by using the following command:

```
# Semi parametric model
robinson_reg_n1 <- npplreg(as.numeric(real_wage) ~ educ + age | predicted_p, data = m_wome
robinson_reg_n2 <- npplreg(as.numeric(real_wage) ~ educ + age | predicted_p, data = m_wome
robinson_reg_n3 <- npplreg(as.numeric(real_wage) ~ educ + age | predicted_p, data = m_wome

summary(robinson_reg_n1)
summary(robinson_reg_n2)
summary(robinson_reg_n3)
```

Table 5: Estimation of gamma coefficients using np

Variable	1000 obs.	5000 obs.
educ	2934.344	2613.985
age	141.0934	-372.652

For the manual case we define functions for the univariate case o local constant estimation because our original functions were for multivariate form. We start by defining the Epanechnikov kernel in univariate case, and the nadaraya-watson estimator in univariate case.

```
# We will need to define additional functions to perform in the univariate case
# Define the kernel function (Epanechnikov kernel)
epa_uni <- function(x, x0, h) {
  u <- abs((x - x0) / h) # argument
  k <- 0.75 * (1 - u^2) * as.numeric(abs(u) <= 1) # value
  return(k)
}

# Univariate NW Regression
nw_regression <- function(x0, x, y, h) {
  k <- epa_uni(x, x0, h)
  sum(k * y) / sum(k)
}
```

We then define the bandwidth for each sample. as we can see below in the code.

```
# Bandwidth for P
HP_1 <-  sd(P_n1) * (4 / (1 + 2))^(1/(1 + 4) ) * (1000 ^(-1*(1/(1+4))))
HP_2 <-  sd(P_n2) * (4 / (1 + 2))^(1/(1 + 4) ) * (5000 ^(-1*(1/(1+4))))
```

Table 6: Silverman's Bandwidth for predicted P - manual

Bandwidth	1000 Obs.	5000 Obs.
h	0.006207599	0.003105639

We use the function sapply and nw_regression to non-parametrically regress each variable and estimate it to obtain $\hat{g}_t, t \in \{y, X\}$ that will be used to to calculate $e_y = y - g_y$ and $e_X = X - g_x$.

```
# First Calculate gy
# First sample
gy_n1 <- sapply(P_n1, nw_regression, x = P_n1, y = W_n1, h = HP_1 )
plot(P_n1, W_n1)
lines(P_n1, gy_n1, col = "blue", lwd = 2)

# Second sample
gy_n2 <- sapply(P_n2, nw_regression, x = P_n2, y = W_n2, h = HP_2)
plot(P_n2, W_n2)
lines(P_n2, gy_n2, col = "blue", lwd = 2)

# Second: calculate gx
gx_n1 <- as.matrix(cbind(sapply(P_n1, nw_regression, x = P_n1, y = Z_n1[,1], h = HP_1 ), s
gx_n2 <- as.matrix(cbind(sapply(P_n2, nw_regression, x = P_n2, y = Z_n2[,1], h = HP_2 ), s


# Calculate residual e_y = y - g_y, e_x = X- g_x
ey_n1 <- W_n1 - gy_n1 # First sample
ex_n1 <- Z_n1 - gx_n1

ey_n2 <- W_n2 - gy_n2 # Second samples
ex_n2 <- Z_n2 - gx_n2
```

We plot below for each variable and sample the nonparametric regression.

With the calculated residuals we regress them to obtain $\hat{\gamma}$ manually.

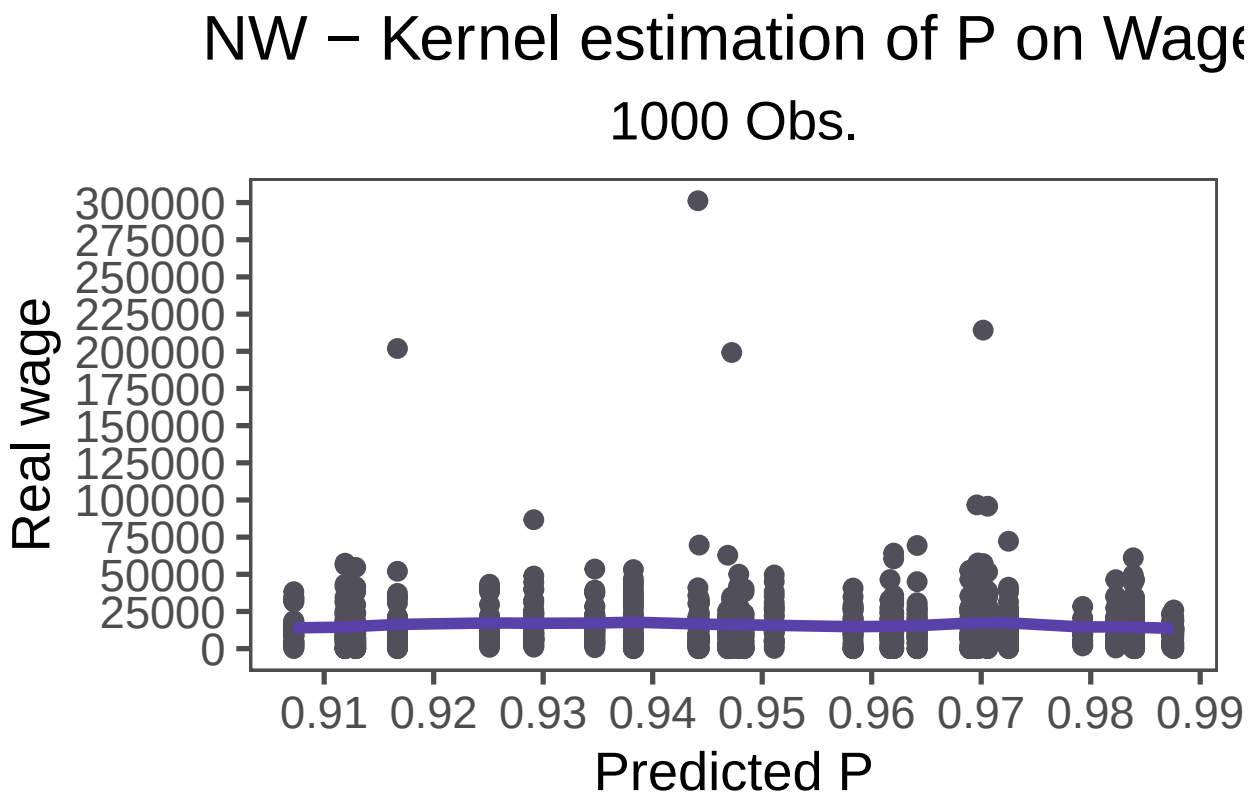

Figure 6: NP regression - Wages on P (1000 obs.)

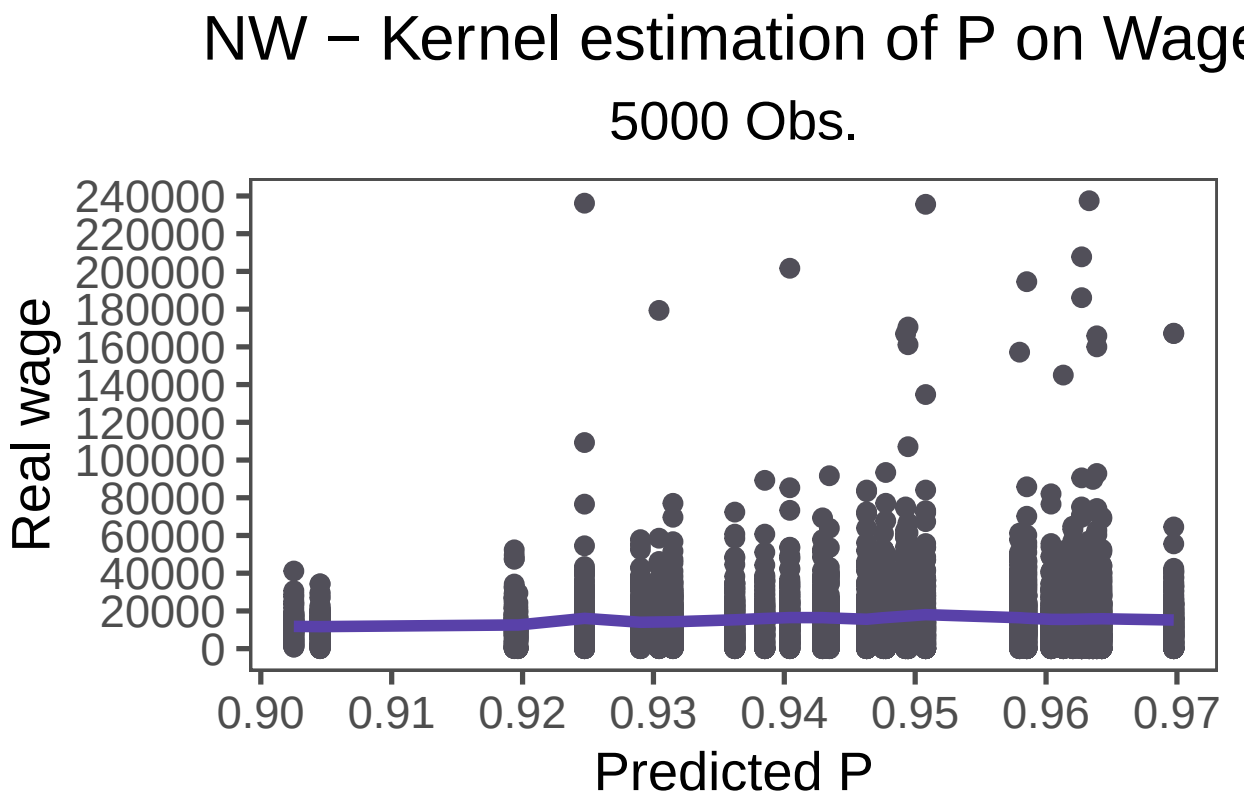

Figure 7: NP regression - Wages on P (5000 obs.)

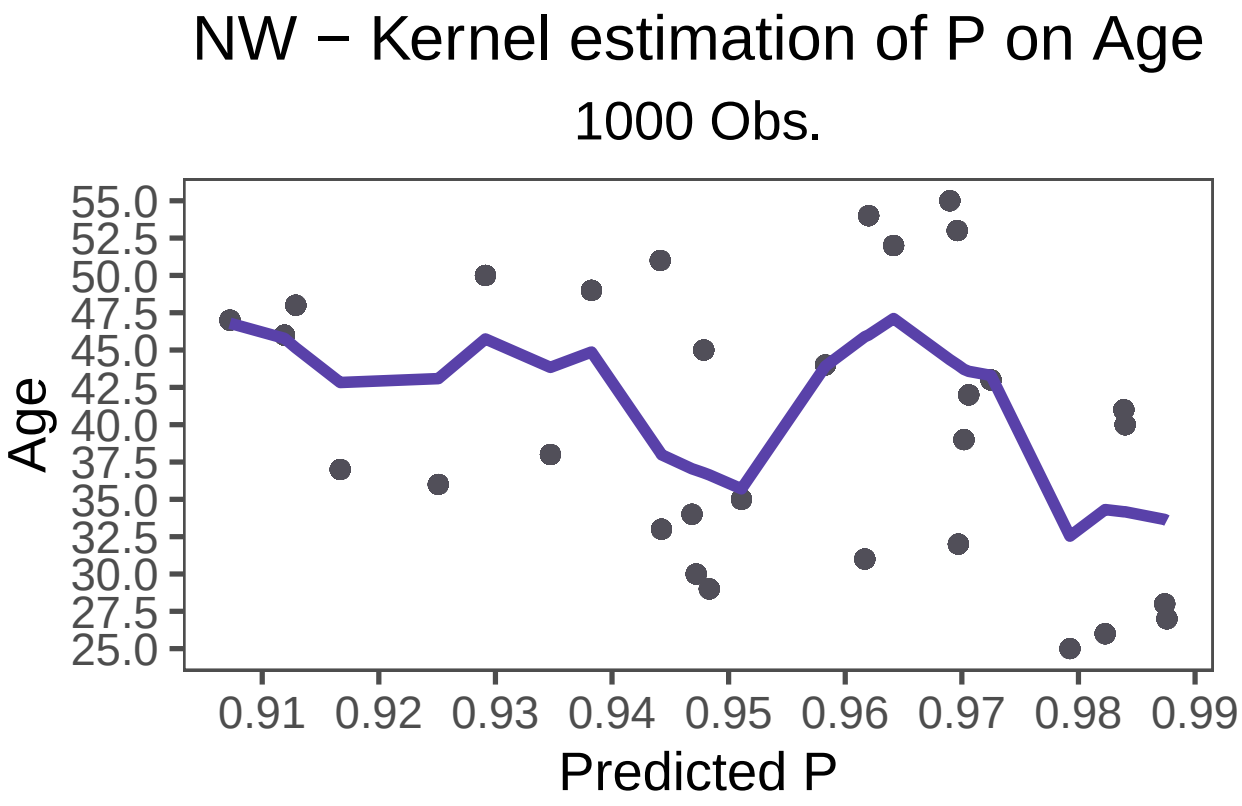

Figure 8: NP regression - Age on P (1000 obs.)

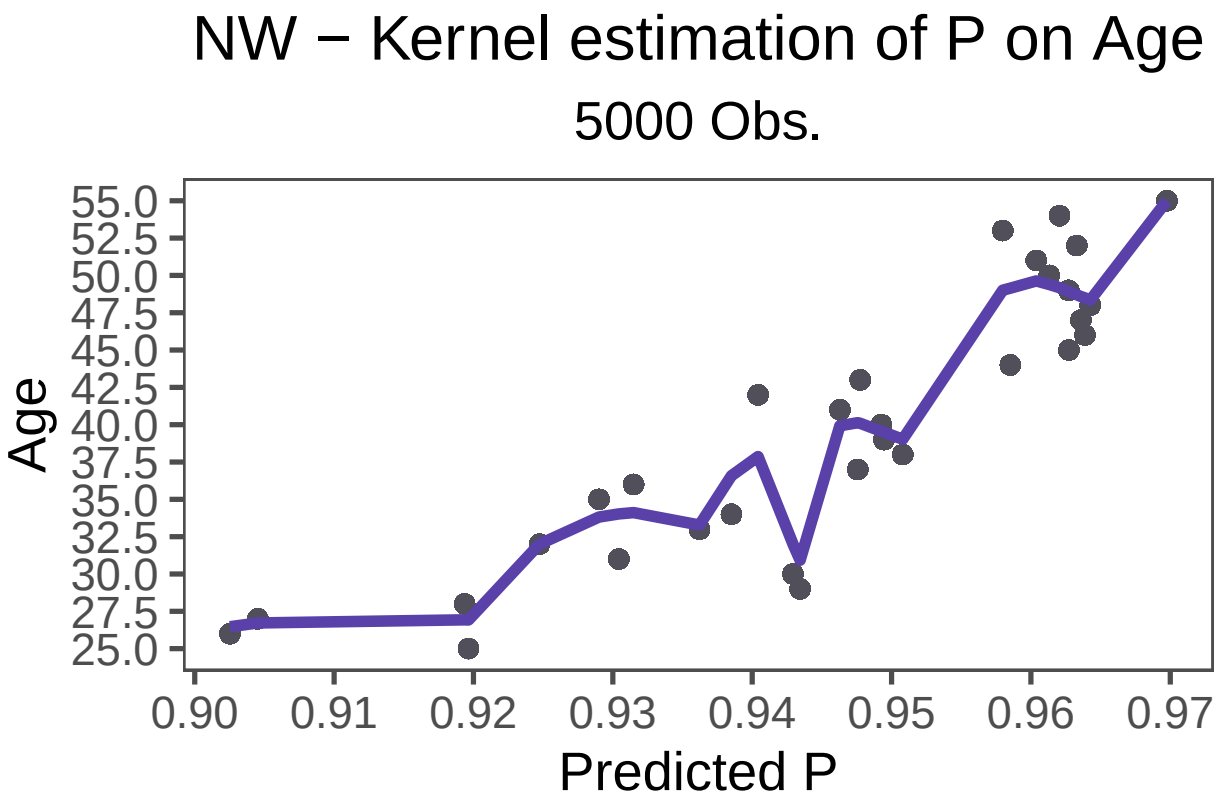

Figure 9: NP regression - Age on P (5000 obs.)

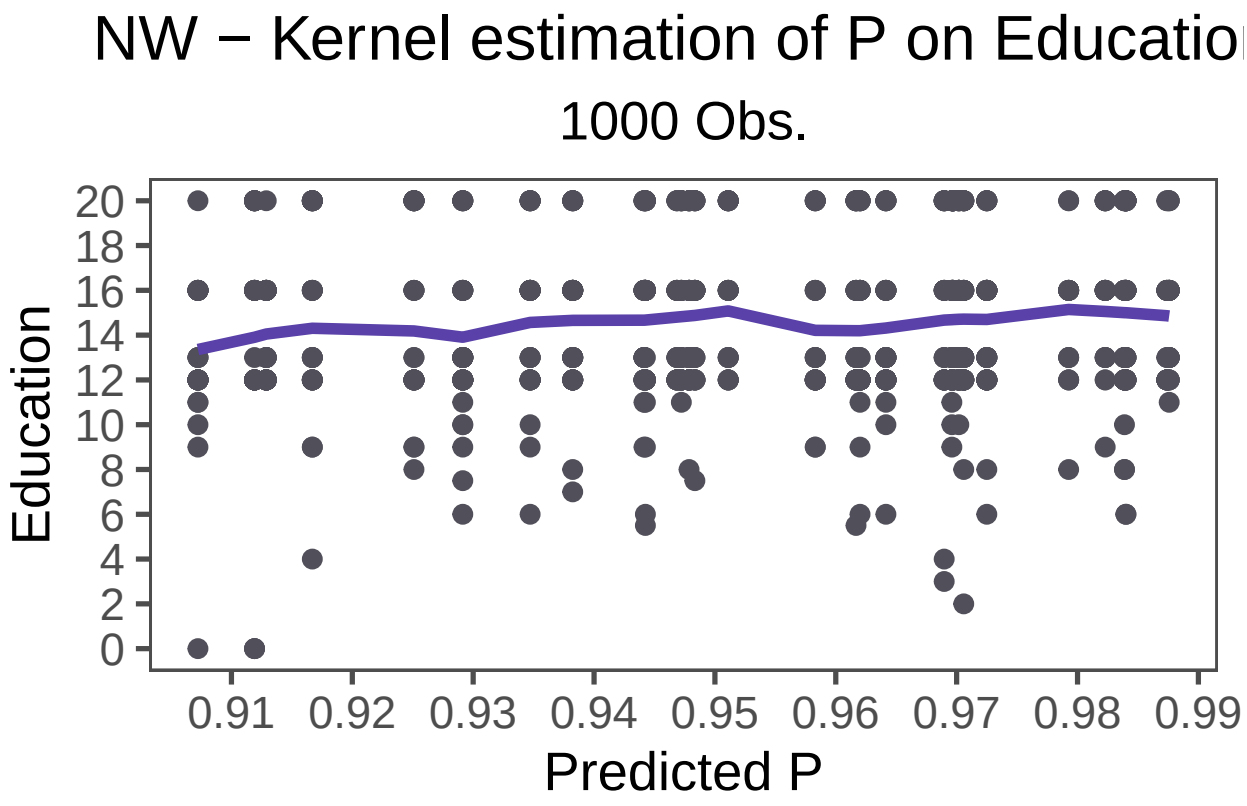

Figure 10: NP regression - Education on P (1000 obs.)

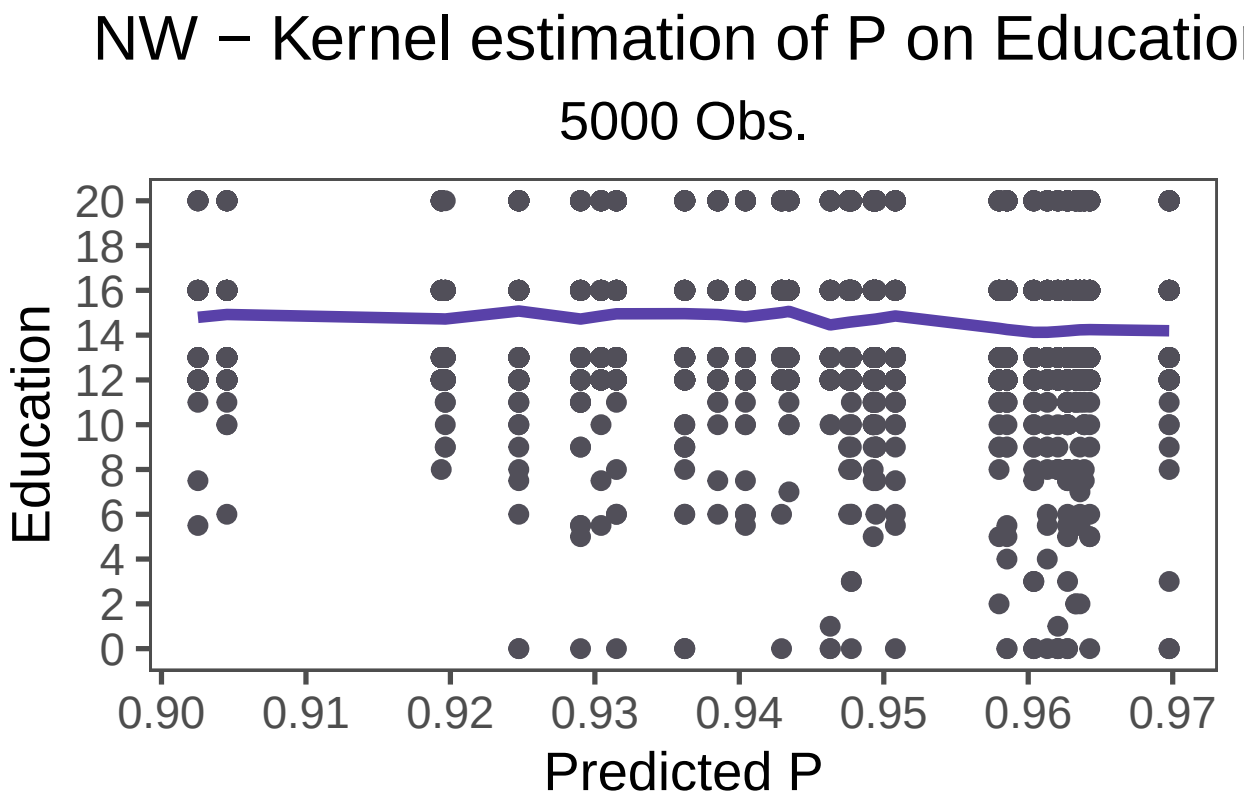

Figure 11: NP regression - Education on P (5000 obs.)

```
# Finaly, OLS residuals
gamma_n1 <- solve(t(ex_n1) %*% ex_n1) %*% t(ex_n1) %*% ey_n1
gamma_n2 <- solve(t(ex_n2) %*% ex_n2) %*% t(ex_n2) %*% ey_n2
```

We get the following results:

Table 7: Estimation of gamma coefficients manually

Variable	1000 obs.	5000 obs.
educ	1928.1576	1541.002
age	173.31590	38.54569

(c) Use your estimates for γ obtained in part (b) to estimate α and β using the Klein and Spady single index estimator. Be careful to adjust the estimator to account for the fact that $\Pr(P = 1 \mid y, z, x) = 1 - F\left(x'\alpha + \beta y - \widehat{\gamma}' z\right)$.

The built-in Klein and Spady estimation does not permit to change the betas, so for this question we will only manually perform the estimation using a loglikelihood approach.

(d) Do you fail to reject the theoretical implications of the model? Discuss.