1. Write a function named `tp420` in R that plots a given vector of data as a time series. Your code should take as input a vector `x` of data and output a plot of `x` versus time `t`. Time `t` should range from 1 to the length of the vector `x`.

2. Write a function named `fd420` that performs the “first-difference” operation on a given vector `x`. Your code should take as input a vector `x` of data and output a vector of first-differences computed from `x`. Your function should also plot (using `tp420`) the returned vector as a time series.

Recall that the `i`th first-difference is the difference between the `(i + 1)`th and the `i`th elements of `x`. Check to make sure that the returned vector has length that is one smaller than the length of `x`.

3. Write a function named `sacf420` that calculates the sample autocorrelation function from a given vector `x`. Your code should take as input a vector `x` of data and output a vector of sample autocorrelations computed from `x`. Your function should also plot (using `tp420`) the returned vector as a time series.

Recall that the `i`-lag sample autocorrelation

\[ \hat{\rho}(i) = \frac{\hat{\gamma}(i)}{\hat{\gamma}(0)} \quad i = 1, 2, \ldots, n - 1 \]

where \( \hat{\gamma}(j) \) is the `j`-lag sample covariance and `n` is the length of `x`. Your code should calculate \( \hat{\gamma}(j) \) explicitly, that is, you are not allowed to use the built-in function in R. Check to make sure that all returned correlations are less than 1 in absolute value.

4. Write a function named `arpcoeff420` that estimates and returns the parameters for an AR(p) model. Your code will take as input a vector `x` of data and a positive integer `p`, and return parameters fitted from an AR(p) model.

Recall the AR(p) model:

\[ Y_t - \bar{Y} = \pi_1(Y_{t-1} - \bar{Y}) + \pi_2(Y_{t-2} - \bar{Y}) + \ldots + \pi_p(Y_{t-p} - \bar{Y}) + Z_t. \]  

(1)

Your code will thus return a vector containing estimates of the `(p + 1)` parameters \( \bar{Y}, \pi_1, \pi_2, \ldots, \pi_p \).
As noted on p. 61 of your textbook, the parameters $\pi_1, \pi_2, \ldots, \pi_p$ can be estimated by fitting data to the model

$$Y_t - \bar{x} = \pi_1(Y_{t-1} - \bar{x}) + \pi_2(Y_{t-2} - \bar{x}) + \ldots + \pi_p(Y_{t-p} - \bar{x}) + Z_t$$

(2)

treating it as an ordinary linear regression model, where $\bar{x}$ is the sample mean of the data in $x$. Accordingly, your code should have the following steps.

(a) Compute $\bar{x}$ as the sample mean of data in $x$. This is your estimate $\hat{\mu}$ of $\mu$.

(b) Form the “data” matrix $D$ for the linear regression in (2). If the length of the original data vector $x$ is $n$, then the data matrix $D$ will have $n - p$ rows, each having $p$ columns. Looking at (2), we see that the first column of $D$ will have the data $x_p - \bar{x}, x_{p+1} - \bar{x}, \ldots, x_{n-1} - \bar{x}$. Likewise, the second column of $D$ will have the data $x_{p-1} - \bar{x}, x_p - \bar{x}, \ldots, x_{n-2} - \bar{x}$. The last (or $p$th) column of $D$ will have the data $x_1 - \bar{x}, x_2 - \bar{x}, \ldots, x_{n-p} - \bar{x}$.

(c) Form the dependent variable vector $Y$ for the linear regression in (2). Again looking at (2), we see that $Y$ should be a $(n - p)$ vector containing the data $x_{p+1} - \bar{x}, x_{p+2} - \bar{x}, \ldots, x_n - \bar{x}$.

(d) Perform linear regression with data matrix $D$ and dependent variable $Y$. Recall that the coefficients of such a linear regression are given by $\hat{\pi} = (D^T D)^{-1} D^T Y$. In other words, $\hat{\pi}$ is the solution to the linear equation $(D^T D) \hat{\pi} = D^T Y$. (In R, the solution $z$ to the linear equation $Az = b$ can be obtained using “solve(A,b).” Also, in R, the transpose of a matrix $M$ is obtained as $t(M)$ and the product of two matrices $M$ and $N$ is obtained as $M \%*\% N$.)

(e) Check to make sure that the $\hat{\pi}$ obtained in the previous step has $p$ elements. Concatenate $\hat{\mu}$ to the beginning of this vector and return the concatenated vector as output.

5. Write a function named `spacf420` that estimates and returns sample partial autocorrelation function. Your code should take as input a data vector $x$ and a positive integer $r$, and output a vector containing the first $r$ sample partial autocorrelations. Your function should also plot (using `tp420`) the returned vector as a time series.

Recall that the $i$th sample partial autocorrelation from data $x$ is simply the last fitted coefficient $\hat{\pi}_i$ when fitting an AR(i) model to the data $x$. (It measures the excess correlation not accounted for by an AR(i-1) model.) So, your function will have a loop running from 1 through $r$. During the $j$th step of this loop, you will call the function `arpcoeff420` to fit an AR($j$) model, and pick up the last fitted coefficient $\hat{\pi}_j$ from the fitted model.