Homework Assignment 2: The Advantages of Backwardness

36-402, Data Analysis, Spring 2012

Due 31 January 2012

Many theories of economic growth say that it's easier for poor countries to grow faster than rich countries — "catching up", or the "advantages of backwardness". One argument for this is that poor countries can grow by copying existing, successful technologies and ways of doing business from rich ones. But rich countries are already using those technologies, so they can only grow by finding new ones, and copying is faster than innovation. So, all else being equal, poor countries should grow faster than rich ones. One way to check this is to look at how growth rates are related to other economic variables.

Our data for examining this will be taken from the "Penn World Table" (`http://pwt.econ.upenn.edu/php_site/pwt_index.php`), for selected countries and years. The data file is `penn-select.csv` on the class website. Each row of this table gives, for a given country and a five-year period, the starting year, the initial population of the country, the initial gross domestic product (GDP)[1] per capita (adjusted for inflation and local purchasing power), the average annual growth rate of GDP over that period, the average population growth rate, the average percentage of GDP devoted to investment, and the average percentage ratio of trade (imports plus exports) to GDP[2].

We will use the `np` package on CRAN to do kernel regression.[3] Install it, and load the data file `penn-select.csv` (link on the class website).

[1]Annual gross domestic product is the total value of all goods and services produced in the country in a given year. It has some pathologies — an earthquake which breaks everyone's windows *increases* GDP by the value of the repairs — but it's a standard measure of economic output.

[2]The Penn tables call this variable "openness". It can be bigger than 100, if, for instance, a country re-exports lots of its imports.

[3]In addition to the examples in Chapter 4 of the notes, the package has good help files, and a tutorial at `http://www.jstatsoft.org/v27/i05`.

1. (5 points) Fit a linear model of `gdp.growth` on `log(gdp)`. What is the coefficient? What does it suggest about catching-up?

2. (5 points) Fit a linear model of `gdp.growth` on `log(gdp)`, `pop.growth`, `invest` and `trade`. What is the coefficient on `log(gdp)`? What does it suggest about catching-up?

3. (5 points) It is sometimes suggested that the catching-up effect only works for countries which are open to more-developed economies. Add an interaction between `log(gdp)` and `trade` to the model from Problem 2. What are the relevant coefficients? What do they suggest about catching-up?

4. (15 points) Use data-set splitting, as in Chapter 3, to decide which of these three linear models predicts best. (You can adapt the code from that chapter or write your own.) Which one is the winner?

5. (15 points) The `npreg` function in the `np` package does kernel regression. By default, it uses a combination of cross-validation and sophisticated but very slow optimization to pick the best bandwidth. In this problem, we will force it to use fixed bandwidths, and do the cross-validation ourselves.

   ```
   penn.0.1 <- npreg(gdp.growth~log(gdp),bws=0.1,data=penn)
   ```

   does a kernel regression of `growth` on `log(gdp)`, using the default kernel (which is Gaussian) and bandwidth 0.1. (You don't have to call the data `penn`.) You can run `fitted`, `predict`, etc., on the output of `npreg` just as you can on the output of `lm`. (There are more examples of using `npreg` in Chapter 4.)

   The code at the end of this assignment (also online) uses five-fold cross-validation to estimate the mean-squared error for the five bandwidths $0.1, 0.2, 0.3, 0.4, 0.5$. Use it to create a plot of MSE versus bandwidth. Add to the same plot the MSEs of those five bandwidths on the *whole* data. What bandwidth predicts best?

6. (10 points) Make a scatterplot of `log(gdp)` versus `growth`. Add the line for the linear model from problem 1. Add the fitted values for the kernel curve with the best bandwidth (according to the previous problem). What does this suggest about catching up?

   (There are at least two ways to get the fitted values for the kernel regression, using `fitted` or `predict`.)

7. (5 points) `npreg` will also do kernel regressions with multiple input variables. This time, use the built-in bandwidth selector:

   ```
   penn.npr <- npreg(gdp.growth ~ log(gdp) + pop.growth + invest
     + trade, data=penn, tol=0.1, ftol=0.1)
   ```

(The last two arguments tell the bandwidth selector to not be very hard to optimize; it may still take several minutes.) What are the selected bandwidths? (Use `summary`.)

8. (5 points) Explain why we cannot add an interaction between `log(gdp)` and `trade` to the nonparametric regression from the previous problem.

9. (10 points) Find the 33$^{\text{rd}}$ and 67$^{\text{th}}$ percentiles of `pop.growth`, `invest` and `trade`. There are eight combinations of these values. For each combination, plot the predicted growth rate versus the initial GDP, over the range of observed GDP values, under the linear model you picked in problem 4, and the kernel regression from problem 7. (You can do this with `predict`, but there are probably ways.) Describe what the two sets of curves suggest about catching-up.

10. (15 points) To chose between the linear model and the kernel regression, use cross-validation again. Modify the code provided to use five-fold cross-validation to get CV MSEs for both the linear regression and for the nonparametric regression (with automatic bandwidth selection). Which predicts better?

11. (10 points) Based on your analysis, does the data support the idea of catching up, undermine it, support its happening under certain conditions, or provide no evidence either way? Describe one additional variable which could be added to the data to the analysis. (As always, explain your answers.)

```
# Compare predictive ability of different bandwidths using k-fold CV
  # Inputs: number of folds, vector of bandwidths, dataframe
  # Presumes: data frame contains variables called "gdp.growth" and "gdp"
  # Output: vector of cross-validated MSEs for the different bandwidths
  # The default bandwidths here are NOT good ones for other problems
cv.growth.folds <- function(nfolds=5, bandwidths <- (1:5)/10, df=penn) {
  require(np)
  case.folds <- rep(1:nfolds,length.out=nrow(df))
    # divide the cases as evenly as possible
  case.folds <- sample(case.folds) # randomly permute the order
  fold.mses <- matrix(0,nrow=nfolds,ncol=length(bandwidths))
  colnames(fold.mses) = as.character(bandwidths)
     # By naming the columns, we'll won't have to keep track of which bandwidth
     # is in which position
  for (fold in 1:nfolds) {
    # What are the training cases and what are the test cases?
    train <- df[case.folds!=fold,]
    test <- df[case.folds==fold,]
    for (bw in bandwidths) {
      # Fit to the training set
      # First create a "bandwidth object" with the fixed bandwidth
      current.npr.bw <- npregbw(gdp.growth ~ log(gdp), data=train, bws=bw,
        bandwidth.compute=FALSE)
      # Now actually use it to create the kernel regression
      current.npr <- npreg(bws=current.npr.bw)
      # Predict on the test set
      predictions <- predict(current.npr, newdata=test)
      # What's the mean-squared error?
      fold.mses[fold,paste(bw)] <- mean((test$gdp.growth - predictions)^2)
      # Using paste() here lets us access the column with the right name...
    }
  }
  # Average the MSEs
  bandwidths.cv.mses <- colMeans(fold.mses)
  return(bandwidths.cv.mses)
}
```