Assignment 5: Local Poisson Regression

Ian Wallgren, Wanchang Zhang, Lavinia Hriscu, Victor Jimenez

In this assignment we will use a generalized non-parametric regression model to study the relation between two variables in a human development dataset.

```r
countries<-read.csv2(file="HDI.2017.subset.csv",row.names = 1)
attach(countries)
le.fm.r = round(le.fm)
```

In general, we consider a bivariate random variable (X, Y) with joint distribution such that:

$$(Y|X = x) \sim f(y; m(x), \psi) = f(y; g^{-1}(\theta(x)), \psi); \quad \psi \in \mathbb{R}^p$$

where $m(x) = E(Y|X = x) \in C^2$ for which an invertible link function $g(\cdot)$ exists, such that $\theta(x) \in C^2$ is free of constraints. Then, we estimate $\theta(x)$ locally by maximizing the expected log-likelihood function:

$$l_t(x; h) = \sum_{i=1}^{n} w_i(x; h) l_t^i(x)$$

where $w_i^t \propto K\left(\frac{t-x_i}{h}\right)$ are the weights for the contributions of every data-point at the local computation of the likelihood. A higher h means that more points are to consider for the construction of the local estimator, thus yielding a low complexity, low flexibility estimator with risk of underfitting the data if its value is too high. When larger values of h are considered, the opposite happens.

$Y = 0, 1, 2, \ldots$

1. Bandwidth choice for the local Poisson regression

$\log Y =$

In this case, $(X, Y) \sim \text{Poiss}(\lambda(x))$ and $\lambda(x) = E(Y|X = x)$ is to be estimated, and a link function is not necessary as $\lambda(x) \in \mathbb{R}$ is already free of constraints. We will then use local Poisson regression and will focus the assignment in the choice of the bandwidth of the kernel such that the expected log-likelihood of an independent observation is maximized:

$\lambda(x) > 0$

$\log(\lambda(x)) \in R$

$$h_{CV} = \arg\max_h l_{CV}(h) = \arg\min_h -\frac{1}{n} \sum_{i=1}^{n} \log\left(\hat{\mathbb{P}}_h^{(-i)}(Y = y_i|X = x_i)\right)$$

Where $\hat{\mathbb{P}}_h^{(-i)}$ is an approximation of the probability mass function of the Poisson distribution of our data where the i-th variable has been omitted. The full expression would be:

$$\log\left(\mathbb{P}(Y = y_i|X = x_i)\right) = \log\left(e^{-\lambda_i} \frac{\lambda_i^{y_i}}{y_i!}\right) = -\lambda_i + y_i \log(\lambda_i) - \log(y_i!)$$

The function that computes $\log\left(\hat{\mathbb{P}}_h^{(-i)}(Y = y_i|X = x_i)\right)$ for every datapoint and yields $l_{CV}(h)$ is the following:

```r
loglik.CV.poisson <- function(x, y, h){
  n <- length(x)
  lambda.i <- sapply(1:n,
                  function(i, x, y, h){
                    sm.poisson(x=x[-i], y=y[-i], h=h, eval.points=x[i],display='none')$estimate
                  }, x, y, h)
  return(-sum(y * log(lambda.i) - lambda.i - log(factorial(y))) / n)
}
```

Probably dont need to divide n, the script of loglik.CV.binomial also does not need to divide n

Now, we can perform LOO-CV by applying the previous function recursively for every h that we want to consider. If no range of bandwidths to consider is provided, the function estimates a suitable range with the function `h.select` of the library `sm`.

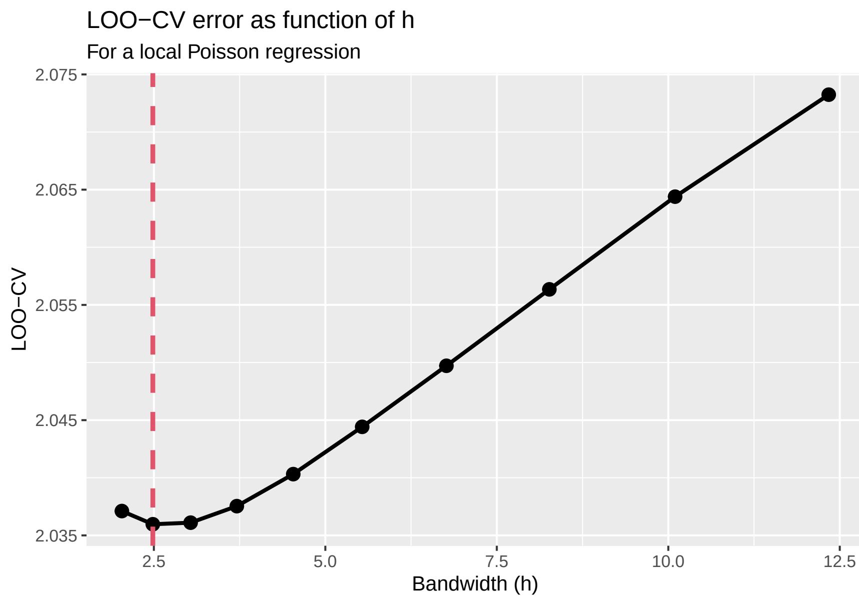
```r
h.cv.sm.poisson <- function(x, y, rg.h=NULL, l.h=10, method=loglik.CV.poisson){
  cv.h <- numeric(l.h)
  if (is.null(rg.h)){
    hh <- c(h.select(x,y,method="cv"),
            h.select(x,y,method="aicc"))#,hcv(x,y))
      rg.h <- range(hh)*c(1/1.1, 1.5)
  }
  i <- 0
  gr.h <- exp(seq(log(rg.h[1]), log(rg.h[2]), l=l.h))
  for (h in gr.h){
    i <- i + 1
    cv.h[i] <- method(x, y, h)
  }
  return(list(h = gr.h,
              cv.h = cv.h,
              h.cv = gr.h[which.min(cv.h)])
  )
}
```

2. Local Poisson regression for Country Development Data

We will fit a local Poisson regression with the functions provided earlier to model `le.fm.r` (the rounded value of `le.fm`) IDK WHAT DOEST IT MEAN, which is non-negative integer, so it is suitable for the model, as a function of `Life.expec`, which is the life expectancy by country.

First, we obtain the optimal bandwidth value `h.cv` using `h.cv.sm.poisson`:

```r
cv.result = h.cv.sm.poisson(Life.expec, le.fm.r)
```




```
h.cv = cv.result$h.cv
h.cv
```

```
## [1] 2.485047
```

With this value of h, we will perform the regression with `sm.poisson`:

```
cv.pois = sm.poisson(x=Life.expec, y=le.fm.r, h=h.cv, eval.points=Life.expec,display='none')
```