768 Assignment 1*

Keith Roberts; 400279646

2024-09-20

Author's Note: This assignment was completed using lecture materials and the course textbook (Racine, 2019). Generative AI (Chat-GPT4o) was consulted to fix formatting errors and syntax bugs. However, all work is original and is my own. All solutions, including any errors they may contain, are entirely my own.

Discrete Probability and Cumulative Probability Functions Questions and Solutions

1. I wish to demonstrate that the unordered kernel estimator of $p(x)$ that uses Aitchison and Aitken's unordered kernel function is *proper* (i.e., it is non-negative and it sums to one over all $x \in \{0, 1, \dots, c-1\}$).

The unordered kernel function estimator of $\hat{p}(x)$ is given by $\hat{p}(x) = \frac{1}{n}\sum_{i=1}^{n} L(X_i, x, \lambda)$ where the unordered kernel function $L(X_i, x, \lambda)$ defined by

$$L(X_i, x, \lambda) = \begin{cases} 1 & \text{if } X_i = x \\ \frac{\lambda}{c-1} & \text{otherwise,} \end{cases}$$

and where $\lambda \in [0, \frac{c-1}{c}]$.

To show that the estimator is proper, two properties must be shown:

i) The estimator $\hat{p}(x)$ is non-negative and,

ii) The estimator sums to 1 over all $x \in \{0, 1, ..., c-1\}$

I will proceed with showing each of these properties hold in succession.

[*] Keiths-MBP-2, x86_64, root

i. Note that the sign of $\widehat{p}(x) = \frac{1}{n}\sum_{i=1}^{n} L(X_i, x, \lambda)$ depends on the sign of $L(X_i, x, \lambda)$. For any $X_i = x \in \{0, 1, 2, \ldots, c-1\}$ the function $L(X_i, x, \lambda) = 1 - \lambda \geq 0$ since $\lambda \in [0, \frac{\lambda}{c-1}]$ and $\frac{c-1}{c} < 1$. For $X_i \neq x$ we have $L(X_i, x, \lambda) = \frac{\lambda}{c-1} \geq 0$ since $\lambda \in [0, \frac{c-1}{c}]$ and $c > 1$. Therefore, the estimator $\widehat{p}(x)$ is non-negative.

ii. To show that the estimator sums to 1 over all $x \in 0, 1, ..., c-1$, we have

$$\begin{aligned}
\sum_{x=0}^{c-1} \widehat{p}(x) &= \sum_{x=0}^{c-1} \frac{1}{n} \sum_{i=1}^{n} L(X_i, x, \lambda) \\
&= \frac{1}{n} \sum_{i=1}^{n} \sum_{x=0}^{c-1} L(X_i, x, \lambda) \\
&= \frac{1}{n} \sum_{i=1}^{n} \left((1-\lambda) + \frac{(c-1)\lambda}{c-1} \right) \\
&= \frac{1}{n} \sum_{i=1}^{n} (1) \\
&= \frac{n}{n} \\
&= 1
\end{aligned}$$

Hence property ii. is shown and, along with i. above, the estimator $\widehat{p}(x)$ is proper.

2

. Consider the unordered kernel estimator of $p(x)$ that uses Aitchison and Aitken's unordered kernel function.

i. Express the MSE of $\hat{p}(x)$ in terms of the MSE of $p_n(x)$ and constants Λ_1, Λ_2, and Λ_3 similar to those that were defined in Chapter 1 of the textbook.

The $MSE[\hat{p}(x)]$ is given as $Var[\hat{p}(x)] + (Bias[\hat{p}(x)])^2$

First, assume that $\{X_1, \dots, X_n\}$ represents n independent random draws from the probability distribution $p(x)$. Then the expected value of $\hat{p}(x)$ is given as

$$\begin{aligned}
\mathbb{E}[\hat{p}(x)] &= \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}[L(X_i, x, \lambda)] \\
&= p(x) + \lambda\left(\frac{1 - cp(x)}{c-1}\right) \\
&= p(x) + \lambda\Lambda_1
\end{aligned}$$

Utilizing this expectation, we can now proceed to find the Variance of $\hat{p}(x)$ as a first full step in our derivation of the $MSE[\hat{p}(x)]$.

$$\begin{aligned}
Var[\hat{p}(x)] &= \mathbb{E}[(\hat{p}(x) - \mathbb{E}[\hat{p}(x)])^2] \\
&= \mathbb{E}\left[\left(\frac{1}{n}\sum_{i=1}^{n}L(X_i, x, \lambda) - \mathbb{E}[\frac{1}{n}\sum_{i=1}^{n}L(X_i, x, \lambda)]\right)^2\right] \\
&= \frac{1}{n^2}\left(\sum_{i=1}^{n}\mathbb{E}[\eta_i^2] \ + \ \underbrace{\sum_{i=1}^{n}\sum_{i\neq j}\mathbb{E}[\eta_i\eta_j]}_{=0}\right) \\
&= \frac{1}{n^2}\sum_{i=1}^{n}\mathbb{E}[\eta_i^2] \\
&= \frac{1}{n}\mathbb{E}[L(X_1, x, \lambda) - \mathbb{E}[L(X_1, x, \lambda)]]^2 \\
&= \frac{1}{n}(\mathbb{E}[L^2(X_1, x, \lambda)] - \mathbb{E}[L(X_1, x, \lambda)]^2)
\end{aligned}$$

Note that, as in Racine (2019, p. 14),

$$\begin{aligned}\mathbb{E}L^2(X_1, x, \lambda) &= \sum_{t \in \mathcal{D}} L^2(t, x, \lambda)p(t) \\ &= p(x) - 2\lambda p(x) + \lambda^2 \Lambda_2\end{aligned}$$

$$\text{where, } \Lambda_2 = \frac{1 + c^2 p(x) - 2cp(x)}{(c-1)^2}$$

Again following Racine (2019, p. 14) and applying the expression directly above, I can simplify the variance function significantly.

$$\begin{aligned}Var[\widehat{p}(x)] &= \frac{1}{n}(\mathbb{E}[L^2(X_1, x, \lambda)] - \mathbb{E}[L(X_1, x, \lambda)]^2) \\ &= \frac{1}{n}(p(x) - 2\lambda p(x) + \lambda^2 \Lambda_2 - (p(x) + \lambda \Lambda_1)^2) \\ &= \frac{p(x)(1-p(x))}{n} - \frac{2\lambda}{n}\Lambda_3 + \frac{\lambda^2}{n}(\Lambda_2 - \Lambda_1^2)\end{aligned}$$

$$Var[\widehat{p}(x)] = \frac{p(x)(1-p(x))}{n}\left(1 - \frac{\lambda c}{(c-1)}\right)^2$$

The above expression for $Var[\widehat{p}(x)]$ includes the multiplicative fraction $\frac{p(x)(1-p(x))}{n}$ which is equivalent to the $MSE[p_n(x)]$. Thus, I now have an expression for $Var[\widehat{p}(x)]$ in terms of $MSE[p_n(x)$ and I will now turn to deriving the second term in the $MSE[\widehat{p}(x)]$ namely, $\left(Bias[\widehat{p}(x)]\right)^2$.

$$\begin{aligned}\left(Bias[\widehat{p}(x)]\right)^2 &= \mathbb{E}[\widehat{p}(x)] - p(x) \\ &= \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} L(X_i, x, \lambda) - p(x)\right] \\ &= \lambda\left(\frac{1 - cp(x)}{c-1}\right) \\ &= \lambda \Lambda_1\end{aligned}$$

Assembling all pieces, the $MSE[\widehat{p}(x)]$ can be expressed in terms of $MSE[\widehat{p}_n(x)]$ Λ_1, Λ_2, and Λ_3 as follows:

$$\begin{aligned}
MSE[\widehat{p}(x)] &= Var[\widehat{p}(x)] + \big(Bias[\widehat{p}(x)]\big)^2 \\
&= \frac{p(x)(1-p(x))}{n}\left(1-\frac{\lambda c}{(c-1)}\right)^2 \\
&= MSE[p_n(x)]\left(1-\frac{\lambda c}{(c-1)}\right)^2 \\
&= MSE[p_n(x)] - \frac{2\lambda c p(x)(1-p(x))}{n(c-1)} + \frac{\lambda^2 c^2 p(x)(1-p(x))}{n(c-1)^2}
\end{aligned}$$

$$MSE[\widehat{p}(x)] = MSE[p_n(x)] + \lambda^2\Lambda_1^2 - \frac{2\lambda\Lambda_3}{n} + \frac{\lambda^2}{n}(\Lambda_2 - \Lambda_1^2)$$

ii. A comparison of the finite sample performance of $\widehat{p}(x)$ and that of $p_n(x)$ revolves around the magnitudes of Λ_1, Λ_2, and $\Lambda_3 = p(x)(1+\Lambda_1)$. Suppose that X has a discrete uniform distribution (i.e., $p(x) = 1/c$ for all $x \in \mathcal{D}$). Express $\mathrm{MSE}\,\widehat{p}(x) - \mathrm{MSE}\,p_n(x)$ in terms of n, c, and λ and determine its sign for *any* λ.

The finite sample performance can be measured by the difference $MSE[\widehat{p}(x)] - MSE[p_n(x)]$. If X has a discrete uniform distribution where $p(x) = \frac{1}{c}\ \forall\ x \in \mathcal{D}$ then, we have the following:

$$\begin{aligned}
MSE[\widehat{p}(x)] - MSE[p_n(x)] &= \lambda^2\Lambda_1^2 - \frac{2\lambda\Lambda_3}{n} + \frac{\lambda^2}{n}(\Lambda_2 - \Lambda_1^2) \\
&= 0 - \frac{2\lambda\frac{1}{c}}{n} + \frac{\lambda^2}{n}(\frac{1}{c-1} - 0) \\
&= -\frac{2\lambda}{nc} + \frac{\lambda^2}{n(c-1)} \\
&= \frac{\lambda(c\lambda - 2c + 2)}{nc(c-1)}
\end{aligned}$$

Clearly the sign of the difference above is determined by the sign of $\lambda(c\lambda - 2c + 2)$ which has 3 potential cases.

If $\lambda = 0$, then $MSE[\widehat{p}(x)] - MSE[p_n(x)] = 0$.

If $\lambda > \frac{2c-2}{c}$ then $\lambda(c\lambda - 2c + 2) > 0$ and $MSE[\widehat{p}(x)] - MSE[p_n(x)] > 0$.

Finally, if $\lambda < \frac{2c-2}{c}$ then $\lambda(c\lambda - 2c + 2) < 0$ and $MSE[\widehat{p}(x)] - MSE[p_n(x)] < 0$.

3. Consider the probability function $p(x)$ for the unordered discrete random variable $X \in \mathcal{D} = \{0, 1, \ldots, c{-}1\}$, where $c \geq 2$ represents the number of unique outcomes. Let $\{X_i\}_{i=1}^{n}$ represent i.i.d. draws from a distribution with unknown $p(x)$. The kernel estimator of $p(x)$ is given by

$$\widehat{p}(x) = \frac{1}{n}\sum_{i=1}^{n} L(X_i, x, \lambda)$$

Where $L(\cdot)$ is an unordered kernel function defined as

$$L(X_i, x, \lambda) = \begin{cases} 1 & \text{if } X_i = x \\ \lambda & \text{otherwise} \end{cases}$$

and where $\lambda \in [0, 1]$.

i. Now I wish to derive the bias of this estimator $\widehat{p}(x) = \frac{1}{n}\sum_{i=1}^{n} L(X_i, x, \lambda)$

$$\begin{aligned} Bias[\widehat{p}(x)] &= \mathbb{E}[\widehat{p}(x)] - p(x) \\ &= \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} L(X_i, x, \lambda)\right] - p(x) \\ &= \mathbb{E}[L(X_i, x, \lambda)] - p(x) \text{ by i.i.d} \\ &= \sum_{t \in \mathcal{D}} L(t, x, \lambda)p(t) - p(x) \\ &= \left[L((t = x), x, \lambda)p(x) + \sum_{t \neq x} L(t, x, \lambda)\right] - p(x) \\ &= p(x) + \lambda(1 - p(x)) - p(x) \end{aligned}$$

$$\therefore Bias[\widehat{p}(x)] = \lambda(1 - p(x))$$

ii. Next I wish to derive the variance of this estimator

$$
\begin{aligned}
Var[\widehat{p}(x)] &= \mathbb{E}\big[\big(\widehat{p}(x) - \mathbb{E}[\widehat{p}(x)]\big)^2\big] \\
&= \frac{1}{n}\mathbb{E}[\eta_i^2] \text{ by i.i.d} \\
&= \frac{1}{n}\Big(\mathbb{E}\big[L^2(X_i, x, \lambda) - \big(\mathbb{E}[L(X_i, x, \lambda)]\big)^2\big]\Big) \\
&= \frac{1}{n}\Big(p(x) + \lambda^2(1-p(x)) - (p(x) + \lambda(1-p(x))^2\Big)
\end{aligned}
$$

$$
\therefore Var[\widehat{p}(x)] = \frac{1}{n}\Big(p(x) + \lambda^2(1-p(x)) - (p(x))^2 - 2\lambda p(x)(1-p(x)) - [\lambda(1-p(x))]^2\Big)
$$

iii. Using the SMSE as my criterion, I now wish to derive the optimal smoothing parameter for this estimator.

First note that the $SMSE = \sum_x MSE$, where $MSE = Var[\widehat{p}(x)] + \big(Bias[\widehat{p}(x)]\big)^2$

Using the pieces previously derived in parts i. and ii. above we find that the SMSE is,

$$
SMSE = \sum_x \left(\frac{1}{n}(p(x) + \lambda^2(1-p(x)) - [p(x)]^2 - 2\lambda p(x)(1-p(x)) - \lambda^2(1-p(x))^2) + \lambda^2(1-p(x))^2\right)
$$

Now the optimal smoothing parameter is found by taking the First Order Condition of this equation wrt λ.

$$
\begin{aligned}
\frac{\partial SMSE}{\partial \lambda} &= \sum_x \left(\frac{1}{n}(2\lambda(1-p(x)) - 2p(x)(1-p(x)) - 2\lambda(1-p(x))^2) + 2\lambda(1-p(x))^2\right) = 0 \\
&= \lambda \sum_x \left((1-p(x)) - (1-p(x))^2 + n(1-p(x))^2\right) = \sum_x p(x)(1-p(x))
\end{aligned}
$$

$$
\therefore \lambda^* = \frac{\sum_x p(x)(1-p(x))}{\sum_x (p(x) + n(1-p(x)))}
$$

iv. Finally, I wish to determine whether or not this estimator is a proper probability function estimator. Now that we have an optimal λ^*, our unordered kernel function $L(X_i, x, \lambda^*) \geq 0 \ \forall \ X_i \in \mathcal{D}$ since $0 \leq p(x) \leq 1$ and therefore $\widehat{p}(x)$ is non-negative. However, the sum of $\widehat{p}(x) \ \forall x \in \mathcal{D} \neq 0$, which is shown below.

$$\begin{aligned}\sum_{x \in \mathcal{D}} \widehat{p}(x) &= \sum_{x} \sum_{i=1}^{n} L(X_i, x, \lambda) \\ &= \frac{1}{n} \sum_{i=1}^{n} (1 + (c-1)\lambda^*) \\ &= \frac{1}{n} \sum_{i=1}^{n} \left(1 + (c-1)\left(\frac{\sum_x p(x)(1-p(x))}{\sum_x \left(p(x) + n(1-p(x))\right)}\right)\right) \neq 1\end{aligned}$$

And therefore, $\widehat{p}(x)$ is not a proper probability function estimator.

4. Consider an *ordered* random variable with discrete support, $X \in \mathcal{D} = \{0, 1\}$, so that the number of outcomes is $c = 2$. Consider the kernel estimator of $p(x) = \Pr(X = x)$ defined by

$$\hat{p}(x) = \frac{1}{n} \sum_{i=1}^{n} l(X_i, x, \lambda),$$

where $l(\cdot)$ is an ordered kernel function defined by

$$l(X_i, x, \lambda) = \lambda^{d_{xi}} / \Lambda_{xi}$$

where $0 \leq \lambda \leq 1$, $d_{xi} = |x - X_i|$, and the normalizing factor $\Lambda_{xi} = \sum_{x \in \mathcal{D}} \lambda^{d_{xi}}$ is tailored to the particular value of $X_i \in \mathcal{D}$.

Presume that you have n independent random draws $\{X_1, X_2, \dots, X_n\}$ from the probability distribution $p(x)$.

i. How many values can this kernel assume? What is the value of the kernel when $X_i = x$? How about when $X_i \neq x$? Is distance taken into account?

Since $X_i \in \mathcal{D} = \{0, 1\}$, the absolute distance d_{xi} can only take on two values: 0 or 1. Hence, we see that when $X_i = x,\ d_{xi} = 0$ and when $X_i \neq x,\ d_{xi} = 1$. Then, by plugging these case values in the ordered kernel function we have $l(X_i = x, x, \lambda) = \frac{\lambda^0}{\Lambda_{xi}} = \frac{1}{1+\lambda}$ when $X_i = x$. Conversely, when $X_i \neq x$ we have the value of the order kernel function $l(X_i \neq x, x, \lambda) = \frac{\lambda}{\Lambda_{xi}} = \frac{\lambda}{1+\lambda}$.

Distance is taken into account. We can see above that the distance parameter directly influences the kernel since the kernel adjusts the weights of the observations based on how close X_i is to x. Intuitively, closer observations obtain a higher relative weighting.

ii. Now I will derive the bias of this estimator and show that the *leading* bias term is of $O(\lambda)$, which mirrors the result for the unordered case. I will Presume that $0 \leq \lambda < 1$ so that, at a certain point in the proof, I can express $1/(1 + \lambda)$ as the infinite series $1 - \lambda + \lambda^2 - \lambda^3 + \dots$ (Note that a hint is given: first get to the point where you have $E\hat{p}(x) = p(x) + \dots$ and where you have collected the terms involving $\lambda/(1+\lambda)$, then use this approximation so that you can write $\lambda/(1 + \lambda) = \lambda - \lambda^2 + \lambda^3 - \dots$).

To derive the bias of the estimator we need to derive the expected value of the estimator.

$$\begin{aligned}
\mathbb{E}[\widehat{p}(x)] &= \mathbb{E}\bigg[\frac{1}{n}\sum_{i=1}^{n} l(X_i, x, \lambda)\bigg] \\
&= \mathbb{E}\big[l(X_i, x, \lambda)\big] \\
&= p(x)l(X_i = x, x, \lambda) + (1 - p(x))l(X_i \neq x, x, \lambda) \\
&= p(x)\frac{1}{(1+\lambda)} + (1 - p(x))\frac{\lambda}{1+\lambda} \\
&= \frac{p(x) + \lambda(1 - p(x))}{1+\lambda}
\end{aligned}$$

Now, invoking the hints in the details of the question and recognizing that $1/(1+\ lambda) \approx 1 - \lambda + \lambda^2 - \lambda^3 + ...$ and that $\lambda/(1+\lambda) \approx \lambda - \lambda^2 + \lambda^3 - ...$ I can now write the above as follows,

$$\begin{aligned}
\mathbb{E}[\widehat{p}(x)] &= \big(p(x) + \lambda(1 - p(x))\big)\big(1 - \lambda + \lambda^2 - \lambda^3 + ...\big) \\
&= p(x) + \lambda(1 - p(x)) - \lambda p(x) + O(\lambda^2)
\end{aligned}$$

$$\begin{aligned}
\therefore\ Bias[\widehat{p}(x)] &= \mathbb{E}[\widehat{p}(x)] - p(x) \\
&= \lambda(1 - p(x)) + O(\lambda^2)
\end{aligned}$$

iii. Next I wish to derive the variance of this estimator up to terms of order $O(\lambda^2)$.

The variance of the estimator is given as

$$\begin{aligned}
Var[\widehat{p}(x)] &= \frac{1}{n}Var(l(X_i, x, \lambda) \\
\text{where}:\ \ l(X_i, x, \lambda) &= \frac{1}{1+\lambda}\ \ w.p.\ p(x)\ \text{or} \\
l(X_i, x, \lambda) &= \frac{\lambda}{1+\lambda}\ \ w.p.\ 1 - p(x)
\end{aligned}$$

$$\begin{aligned}
\text{Hence}:\ \ Var[l(X_i, x, \lambda)] &= \mathbb{E}\big[l(X_i, x, \lambda)^2\big] - \big(\mathbb{E}\big[l(X_i, x, \lambda)\big]\big)^2 \\
&= p(x)\bigg(\frac{1}{1+\lambda}\bigg)^2 + (1 - p(x))\bigg(\frac{\lambda}{1+\lambda}\bigg)^2 - \big(\mathbb{E}\big[l(X_i, x, \lambda)\big]\big)^2 \\
&= \frac{p(x) + \lambda^2(1 - p(x))}{(1+\lambda)^2} - \big(\mathbb{E}\big[l(X_i, x, \lambda)\big]\big)^2 \\
&= \frac{p(x) + \lambda^2(1 - p(x))}{(1+\lambda)^2} - \bigg(\frac{p(x) + \lambda(1 - p(x))}{(1+\lambda)}\bigg)^2
\end{aligned}$$

Which finally simplifies (after expansion) to the following,

$$
\begin{aligned}
Var[l(X_i, x, \lambda)] &= \frac{1}{(1+\lambda)^2}\Bigg(p(x) + \lambda^2(1-p(x)) - (p(x))^2 - 2\lambda(1-p(x)) - \lambda^2(1-p(x))\Bigg) \\
&= \frac{(1-p(x))}{(1+\lambda)^2}\Bigg(p(x) + \lambda^2 p(x) - 2\lambda\Bigg) \\
&= \frac{p(x)(1-p(x))}{(1+\lambda)^2}\Bigg(1 + \lambda^2 - \frac{2\lambda}{p(x)}\Bigg) \\
\therefore Var[\widehat{p}(x)] &= \frac{1}{n}Var[l(X_i, x, \lambda)] = \frac{p(x)(1-p(x))}{n(1+\lambda)^2}\Bigg(1 + \lambda^2 - \frac{2\lambda}{p(x)}\Bigg)
\end{aligned}
$$

iv. Now I wish to derive the MSE and the SMSE of this estimator.

$$
\begin{aligned}
MSE[\widehat{p}(x)] &= Var[\widehat{p}(x)] + \Big(Bias[p(x)]\Big)^2 \\
&= \frac{p(x)(1-p(x))}{n(1+\lambda)^2}\Bigg(1 + \lambda^2 - \frac{2\lambda}{p(x)}\Bigg) + (\lambda(1-p(x))^2)
\end{aligned}
$$

Similarly one finds the SMSE as $SMSE = \sum_{x\in\mathcal{D}}\Bigg[\frac{p(x)(1-p(x))}{n(1+\lambda)^2}\Bigg(1+\lambda^2-\frac{2\lambda}{p(x)}\Bigg)+(\lambda(1-p(x))^2)\Bigg]$

v. What is the optimal smoothing parameter?

The optimal λ^* is derived from solving the first order condition for the minimization of the SMSE wrt to λ.

$$
\frac{\partial SMSE}{\partial \lambda} = \sum_{x\in\mathcal{D}}\Bigg[\frac{p(x)(1-p(x))}{n(1+\lambda)^2}\Bigg(\lambda - \frac{1}{p(x)}\Bigg) - \frac{2p(x)(1-p(x))}{n(1+\lambda)^3}\Bigg(1 + \lambda^2 - \frac{2\lambda}{p(x)}\Bigg)\Bigg] = 0
$$

The solution for the optimal smoothing parameter will solve this first order necessary condition for λ^*. However, I have not been able to find a closed form solution. My suspicion is that there was an error made previously in the derivations that has caused this issue. Unfortunately, this is where I must stop until I am able to correct the errors made previously, should this indeed be the reason for my troubles.

5. Code up a Monte Carlo simulation that compares the SMSE performance of $p_n(x)$ and $\hat{p}(x)$, where the latter uses Aitchison and Aitken's unordered kernel function with three alternatively chosen smoothing parameters:

   i. The SMSE-optimal λ that uses the *true* (unknown in general) probabilities

   ii. The SMSE-optimal λ that uses plug-in estimates $p_n(x)$ of the probabilities

   iii. The likelihood cross-validated λ

   Run two simulations – one where the probabilities differ substantially across the $x \in \mathcal{D}$ and another where they are the discrete uniform $p(x) = 1/c$. Conduct $M = 1000$ Monte Carlo replications and consider the following probabilities and methods for generating the random samples:

Code and Solutions

```r
# Load the necessary library
library(np)
```

```
Warning in .recacheSubclasses(def@className, def, env): undefined subclass
"numericVector" of class "Mnumeric"; definition not updated
```

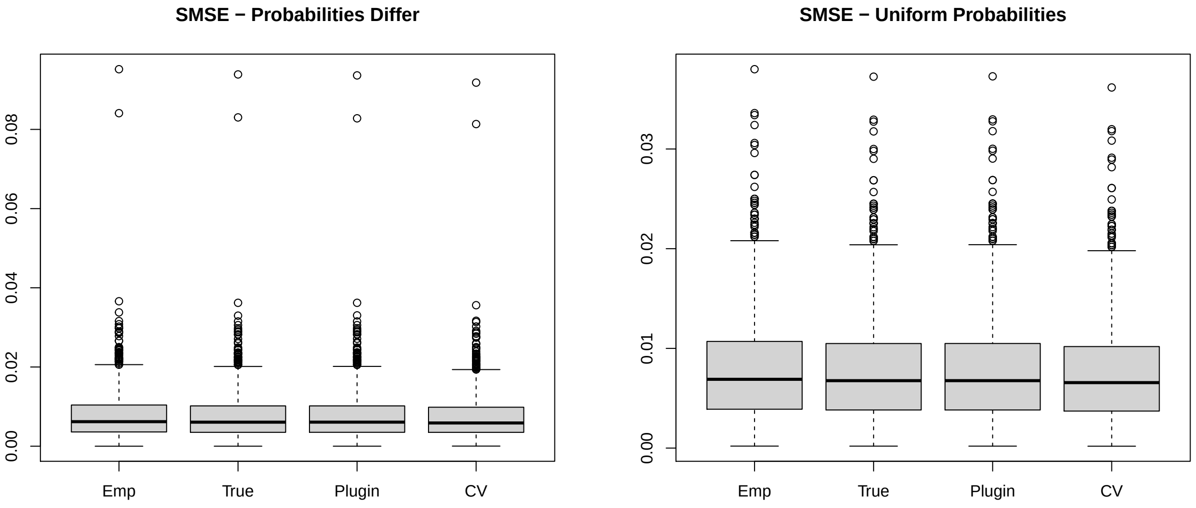
```
Nonparametric Kernel Methods for Mixed Datatypes (version 0.60-11)
[vignette("np_faq",package="np") provides answers to frequently asked questions]
[vignette("np",package="np") an overview]
[vignette("entropy_np",package="np") an overview of entropy-based methods]
```

```r
options(np.messages=FALSE)

# Define the function to compute the SMSE-optimal lambda using the true probabilities
lambda_mse_opt <- function(p) {
  # Calculate lambda_opt
  num <- sum(p * (1 - p))
  denom <- sum(1 + p^2 - 2 * p)
  lambda_opt <- num / (denom + num)
  return(lambda_opt)
}

# Set seed
set.seed(42)

# Sample size and number of Monte Carlo reps
```

```r
n <- 100
M <- 1000

# Define probability vectors
p_diff <- c(0.07, 0.13, 0.20, 0.27, 0.33)
p_uniform <- rep(1/5, 5)

# Storage for results
results <- list()

# Function to perform Monte Carlo simulation
monte_carlo_simulation <- function(p, n, M) {
  smse.p.n <- numeric(M)
  smse.p.hat.opt.true <- numeric(M)
  smse.p.hat.opt.plugin <- numeric(M)
  smse.p.hat.cv.ml <- numeric(M)

  for (m in 1:M) {
    # Generate a random sample on the support {0,1,...c-1}
    X <- sample(0:(length(p) - 1), n, replace = TRUE, prob = p)
    D <- 0:(length(p) - 1)  # Support of X

    # Empirical probability estimate (counts of X)
    p_n <- table(X) / n

    # Optimal lambda using the true probabilities
    lambda_opt_true <- lambda_mse_opt(p)
    p_hat_opt_true <- (table(factor(X)) + lambda_opt_true) / (n + lambda_opt_true * length(D

    # Optimal lambda using the plug-in estimates
    lambda_opt_plugin <- lambda_mse_opt(as.numeric(p_n))
    p_hat_opt_plugin <- (table(factor(X)) + lambda_opt_plugin) / (n + lambda_opt_plugin * len

    # Cross-validated lambda (without npudistbw)
    # Use a fixed lambda or experiment with other cross-validation techniques
    lambda_cv <- 0.5  # For now, a fixed value, can be adjusted
    p_hat_cv_ml <- (table(factor(X)) + lambda_cv) / (n + lambda_cv * length(D))

    # Summed mean square error
    smse.p.n[m] <- sum((as.numeric(p_n) - p)^2)
    smse.p.hat.opt.true[m] <- sum((as.numeric(p_hat_opt_true) - p)^2)
    smse.p.hat.opt.plugin[m] <- sum((as.numeric(p_hat_opt_plugin) - p)^2)
    smse.p.hat.cv.ml[m] <- sum((as.numeric(p_hat_cv_ml) - p)^2)
  }

  # Create a data frame with vectors of SMSE for each estimator
  smse <- data.frame(
    p.n = smse.p.n,
    p.hat.opt.true = smse.p.hat.opt.true,
    p.hat.opt.plugin = smse.p.hat.opt.plugin,
    p.hat.cv.ml = smse.p.hat.cv.ml
  )

  return(smse)
}

# Run simulations for both scenarios
results$diff <- monte_carlo_simulation(p_diff, n, M)
```

```
Warning in as.numeric(p_n) - p: longer object length is not a multiple of
shorter object length

Warning in as.numeric(p_hat_opt_true) - p: longer object length is not a
multiple of shorter object length

Warning in as.numeric(p_hat_opt_plugin) - p: longer object length is not a
multiple of shorter object length

Warning in as.numeric(p_hat_cv_ml) - p: longer object length is not a multiple
of shorter object length

Warning in as.numeric(p_n) - p: longer object length is not a multiple of
shorter object length

Warning in as.numeric(p_hat_opt_true) - p: longer object length is not a
multiple of shorter object length

Warning in as.numeric(p_hat_opt_plugin) - p: longer object length is not a
multiple of shorter object length

Warning in as.numeric(p_hat_cv_ml) - p: longer object length is not a multiple
of shorter object length
```

```
results$uniform <- monte_carlo_simulation(p_uniform, n, M)

# Boxplots of results
par(mfrow = c(1, 2))
boxplot(results$diff, main = "SMSE - Probabilities Differ", names = c("Emp",
                    "True", "Plugin", "CV"), cex.axis = 1, cex.main = 1.2 )
boxplot(results$uniform, main = "SMSE - Uniform Probabilities", names = c("Emp","True", "Plug
```



```
# Summarize the results in tables
summary_diff <- apply(results$diff, 2, summary)
summary_uniform <- apply(results$uniform, 2, summary)

print("Summary - Pr Differ")
```

```
[1] "Summary - Pr Differ"
```

```
print(summary_diff)
```

```
              p.n p.hat.opt.true p.hat.opt.plugin  p.hat.cv.ml
Min.    0.0000000   3.825037e-06     3.825037e-06 2.593694e-05
1st Qu. 0.0036000   3.505509e-03     3.510992e-03 3.509578e-03
Median  0.0062000   6.070377e-03     6.077710e-03 5.867698e-03
Mean    0.0078733   7.739187e-03     7.743627e-03 7.542582e-03
3rd Qu. 0.0104000   1.017359e-02     1.017438e-02 9.848662e-03
Max.    0.0952000   9.388717e-02     9.363793e-02 9.181416e-02
```

```
print("Summary - Uniform Pr")
```

```
[1] "Summary - Uniform Pr"
```

```
print(summary_uniform)
```

```
              p.n p.hat.opt.true p.hat.opt.plugin  p.hat.cv.ml
Min.    0.0002000   0.0001960592     0.0001960602 0.0001903629
1st Qu. 0.0039500   0.0038721694     0.0038725482 0.0037596669
Median  0.0069000   0.0067640427     0.0067651984 0.0065675193
Mean    0.0080588   0.0079000098     0.0079023585 0.0076704819
3rd Qu. 0.0106500   0.0104401529     0.0104429058 0.0101368233
Max.    0.0380000   0.0372512499     0.0372863129 0.0361689471
```

Explanation

I expected that the SMSE-optimal lambda using true probabilities would perform the best illustrated by lower SMSE values. However, I thought it might be reasonable to assume that the plug-in and cross-validated methods would yield higher, albeit close SMSE values compared to the true probability estimator.

This seemed to be a correct intuition since the SMSE-optimal lambda using the true probabilities indeed performs better (with a lower SMSE) than the plug-in and cross-validated methods. But, you can also see that the plug-in estimator is not far off in value, showing that it serves as a reasonable approximation as well.

Recall that the Stein effect refers to a phenomenon where certain shrinkage estimators outperform traditional unbiased estimators, in terms of mean squared error (MSE). Here, the SMSE-optimal lambdaestimators introduce bias through smoothing (shrinkage toward more central probability values). Checking the values of the SMSE estimators computed above, we note that across all estimators in the uniform case the SMSE is indeed lower than the "true probability" model. However, in the p.diff case, only some SMSE values were smaller. Thus, there might be a small Stein effect at work in the p.diff case but it seems that the result is negligible. These values can be shown in the tables below:

```
dplyr::bind_rows(
  data.frame(
    method = "True",
    mean = mean(results$diff$p.hat.opt.true),
    sd = sd(results$diff$p.hat.opt.true)
  ),
  data.frame(
    method = "Plugin",
    mean = mean(results$diff$p.hat.opt.plugin),
    sd = sd(results$diff$p.hat.opt.plugin)
  ),
  data.frame(
    method = "CV",
    mean = mean(results$diff$p.hat.cv.ml),
    sd = sd(results$diff$p.hat.cv.ml)
  )
)
```

```
  method        mean          sd
1   True 0.007739187 0.006719745
2 Plugin 0.007743627 0.006717332
3     CV 0.007542582 0.006557349
```

```
dplyr::bind_rows(
  data.frame(
    method = "True",
    mean = mean(results$uniform$p.hat.opt.true),
    sd = sd(results$uniform$p.hat.opt.true)
  ),
  data.frame(
    method = "Plugin",
    mean = mean(results$uniform$p.hat.opt.plugin),
    sd = sd(results$uniform$p.hat.opt.plugin)
  ),
  data.frame(
    method = "CV",
    mean = mean(results$uniform$p.hat.cv.ml),
    sd = sd(results$uniform$p.hat.cv.ml)
  )
)
```

```
  method        mean          sd
1   True 0.007900010 0.005532706
2 Plugin 0.007902359 0.005535973
3     CV 0.007670482 0.005371958
```