ARE 212
Problem Set 1, Second Half
Scott Kjorlien, Thor Larson, Sarah O'Brien, Bora Ozaltun, Garrison Schlauch

2. Exercises

(1)

Lets first remember the theorem:

—

Theorem 1. *Let X take values in $E \subseteq \mathbb{R}^k$ be a $k \times 1$ absolutely continuous random vector with a probability density function $f_X(x)$. Let $g : E \to \mathbb{R}^k$ be a one-to-one, continuously differentiable function. Let E_0 be an open subset of E such that $P(X \in E_0) = 1$ and such that the Jacobian of g is non-zero on E_0.*

Then the $k \times 1$ random vector $Y = g(X)$ is absolutely continuous with probability density function given by:

$$f_Y(y) = f_X(g^{-1}(y))|\frac{\partial g^{-1}(y)}{\partial y}| \ \ for \ \ y \in g(E_0) \equiv Y_0$$

—

For this question we are given that (x, y) are independent and continuously distributed random variables with densities f_x, f_y. Setting $z = x + y$, we are asked to write down a formula for f_z. Lets set up the problem so that we can use the theorem given above:

- Since (x, y) are independent, $f_{(x,y)} = f_x f_y$.
- Define a function $g : E \to \mathbb{R}^2$ where $E \subseteq \mathbb{R}^2$ such that $\begin{bmatrix} z \\ k \end{bmatrix} = g(x, y) = \begin{bmatrix} x + y \\ y \end{bmatrix}$.
- Define the support of (x, y) to be to be $E = \begin{pmatrix} E_1 \\ E_2 \end{pmatrix} \subseteq \mathbb{R}^2$. Finally, define $g(E) = \begin{pmatrix} A \\ E_2 \end{pmatrix}$, where E_1 might not be the same as A.

Now we can get the inverse of g to be $g^{-1}(z, k) = \begin{bmatrix} z - k \\ k \end{bmatrix}$. The Jacobian of $g^{-1}(z, k)$ can then be calculated as:

$$\frac{\partial g^{-1}(z,k)}{\partial (z,k)} = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix}$$

meaning the determinant is $|\frac{\partial g^{-1}(z,k)}{\partial (z,k)}| = 1$. By the Theorem above, we can finally calculate the density of (z, k):

$$f_{(z,k)}(z, k) = f_{(x,y)}(z - k, k) = f_x(z - k) f_y(k) \ \ k \in E_2 \ z \in A$$

which finally leads us to the distribution of z by integrating out k (which is just y):

$$f_z(z) = \int_A f_x(z - k) f_y(k) dk$$

.

(2)

We are given that s is a discrete random variable and x is a continuous random variable. Assuming that s and x are independent, we will show that the convolution has a continuous cdf. Let $m = s + x$, the random variable that is a result of the convolution. So we can get the cdf to be:

$$P(m \leq a) = P(s + x \leq a) = \sum_{i \in supp(s)} P(s + x \leq a | s = i) P(s = i) = \sum_{i \in supp(s)} P(x \leq a - i) P(s = i)$$

Notice that $P(x \leq a - i)$ is the probability that the random variable x is less than $a - i$. Since x is a continuous random variable, its cdf will be continuous as well. The weighted sum of continuous functions is also continuous, meaning that $P(m \leq a)$ must be continuous. This proves that the cdf is continuous.

Note that this doesn't guarantee continuity or even existence of the pdf of z. We would further need to assume that the pdf of x exists and is continuous to guarantee that the pdf of m is continuous. This requires additional assumptions as not all continuous random variables have pdfs and not all continuous random variables that do have pdfs have continuous pdfs.

Because of this last point, the code provided in class is a bit misleading. Even though we will get a continuous cdf as a result of the convolution, we don't have any guarantees that we will get a continuous pdf. The figure at the end of the notebook plots the pdf, which is a bit misleading since we don't know if the pdf exists or is even continuous. We can however plot the cdf and it will be continuous for all convolutions of discrete and continuous random variables.

(3)

In order for the matrix multiplication $\mathbf{A}\mathbf{A}^{-1}\mathbf{A}$ to be well-defined, $\mathbf{A}^{-}$ must be an $n \times m$ matrix. Since any matrix multiplied by a zero matrix is itself a zero matrix, $\mathbf{A}^{-1}$ can in fact be any $n \times m$ matrix.

(4)

(a)

y is a scalar random variable while $\mathbf{x}$ is a vector random variable.

(b)

y is a single realization of a scalar random variable while $\mathbf{x}$ is similarly a single realization.

(c)

$\mathbf{y}$ is a vector of N realizations, while $\mathbf{X}$ is a matrix of realizations.

(5)

(1)

Let $\mathbf{A}$ be an $m \times n$ matrix of zeros. Note that we cannot apply the fact in the problem since $\mathbf{A}$ has rank $r = 0$. So for a matrix $\mathbf{B}$ to satisfy the four conditions in the problem, it must be that (i) $\mathbf{B}$ is an $n \times m$ matrix (in order for matrix multiplication to be well-defined), and (ii) $\mathbf{B}$ is a matrix of zeros. To see that (ii) must be true, note that $\mathbf{BAB} = \mathbf{0}$, where $\mathbf{0}$ is an $n \times m$ matrix of zeros; hence, in order to satisfy the second property listed in the problem, $\mathbf{B} = \mathbf{0}$. Therefore, $\mathbf{A}^+$ is an $n \times m$ matrix of zeros.

(2)

Suppose $\mathbf{X}$ is an $n \times m$ matrix with full column rank r. Then $r = m > 0$ and we can apply the fact given in the problem. Let $\mathbf{L} = \mathbf{X}$ and $\mathbf{R} = \mathbf{I}_m$ (the $m \times m$ identity matrix). Then $\mathbf{L}$ is an $n \times r$ full column rank matrix, $\mathbf{R}$ is an $r \times m$ full row rank matrix, and $\mathbf{A} = \mathbf{LR}$; hence, $\mathbf{LR}$ is a full rank factorization of $\mathbf{X}$. Then applying the fact,

$$\mathbf{X}^+ = \mathbf{R}^\top(\mathbf{L}^\top\mathbf{X}\mathbf{R}^\top)^{-1}\mathbf{L}^\top = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top.$$

Moreover,

$$\mathbf{X}^+\mathbf{X} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{X} = \mathbf{I}_m.$$

(3)

Let $\mathbf{y} = \mathbf{X}b + \mathbf{u}$ where $\mathbf{X}^\top u = 0$ and $\mathbf{X}$ has full column rank. Pre-multiplying both sides by $\mathbf{X}^+$:

$$\begin{aligned}
&\mathbf{X}^+\mathbf{y} = \mathbf{X}^+\mathbf{X}b + \mathbf{X}^+\mathbf{u} \\
\Rightarrow\ &\mathbf{X}^+\mathbf{y} = \mathbf{X}^+\mathbf{X}b + (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{u} \\
\Rightarrow\ &\mathbf{X}^+\mathbf{y} = \mathbf{X}^+\mathbf{X}b \quad (\text{since } \mathbf{X}^\top u = 0) \\
&\Rightarrow\ b = \mathbf{X}^+\mathbf{y} \quad (\text{since } \mathbf{X}^+\mathbf{X} = \mathbf{I})
\end{aligned}$$

3. Convolutions

(1)

This appears to be a repetition of Exercise (2).

(2)

The class for this section is named `ConvolveDiscrete` and can be found in `PS1/code/classes.py`. Please refer to our notebook for plots and test cases.

Let x and y be discrete independent random variables and $z = x + y$. Then the pmf of z can be written as

$$\begin{aligned}
p_z(z) &= P(z = z) \\
&= \sum_{x \in \Omega_x} P(z = z, x = x) \quad \text{[law of total probability]} \\
&= \sum_{x \in \Omega_x} P(y = z - x, x = x) \quad [(z = z, x = x) \text{ equivalent to } (y = z - x, x = x)] \\
&= \sum_{x \in \Omega_x} P(y = z - x) P(x = x) \quad [x \text{ and } y \text{ are independent}] \\
&= \sum_{x \in \Omega_x} p_y(z - x) p_x(x)
\end{aligned}$$

and the cdf of z can be written as

$$\begin{aligned}
F_z(z) &= P(z \leq z) \\
&= \sum_{x \in \Omega_x} P(z \leq z, x = x) \quad \text{[law of total probability]} \\
&= \sum_{x \in \Omega_x} P(y \leq z - x, x = x) \quad [(z \leq z, x = x) \text{ equivalent to } (y \leq z - x, x = x)] \\
&= \sum_{x \in \Omega_x} P(y \leq z - x) P(x = x) \quad [x \text{ and } y \text{ are independent}] \\
&= \sum_{x \in \Omega_x} F_y(z - x) p_x(x)
\end{aligned}$$

(3)

The class for this section is named `ConvolveContinuous` and can be found in `PS1/code/classes.py`. Please refer to our notebook for plots and test cases.

Let x and y be continuous independent random variables and $z = x + y$. Then the cdf of z can be written as

$$\begin{aligned}
F_z(z) &= P(z \leq z) \\
&= P(x + y \leq z) \quad \text{[definition of } z\text{]} \\
&= \int_{x \in \Omega_x} P(x + y \leq z | x = x) f_x(x) \quad \text{[law of total probability]} \\
&= \int_{x \in \Omega_x} P(x + y \leq z | x = x) f_x(x) \quad \text{[since we're conditioning on } x = x\text{]} \\
&= \int_{x \in \Omega_x} P(y \leq z - x | x = x) f_x(x) \\
&= \int_{x \in \Omega_x} P(y \leq z - x) f_x(x) \quad [x \text{ and } y \text{ are independent}] \\
&= \int_{x \in \Omega_x} F_y(z - x) f_x(x).
\end{aligned}$$

To get the pdf, we can differentiate the cdf with respect to z:

$$f_z(z) = \frac{d}{dz} F_z(z) = \int_{x \in \Omega_x} f_y(z - x) f_x(x).$$

4. General Weighted Linear Regression

OLS

- This estimator belongs to the class of general weighted linear regressions.
- Estimator: $\beta = (X'X)^{-1}X'y$
- $T' = X'$
- Yes, T is random

GLS

- This estimator belongs to the class of general weighted linear regressions.
- Estimator: $\beta = (X'\Omega^{-1}X)^{-1}X'\Omega^{-1}y$
- $T' = X'\Omega^{-1}$
- T is random because X is random. However, note that Ω fixed and chosen by the researcher in GLS.[1]

IV

- This estimator belongs to the class of general weighted linear regressions.
- Estimator: $\beta = (Z'X)^{-1}Z'y$
- $T' = Z'$
- T is random because Z is random.

2SLS

- This estimator belongs to the class of general weighted linear regressions.
- Estimator: $\beta = (X'P_ZX)^{-1}X'P_Zy$, where $P_Z = Z(Z'Z)^{-1}Z'$
- $T' = X'P_Z$
- T is random because X and Z are random.

Logit (via MLE)

- The logit model is not compatible with the model assumed for GLS.
- The functional form is nonlinear - the functional form does not follow the $y = X\beta + u$ structure that is assumed in the set up of the problem.

1. Note: FGLS is used when the researcher does not know the exact structure of the error terms, and instead estimates Ω.

5. Simultaneous Equations

(1)

We are given that y is a $N \times k$ matrix. One configuration of dimensions that would make the algebra conformable is:

- X is $N \times m$.
- β is $m \times k$
- u is $N \times k$
- T is $N \times l$

which would then give us $T'Y$ of dimension $l \times k$ and $T'X$ of dimension $l \times m$. Further, if we assume that $T'X$ is full column rank (which would imply $l \geq m$), then $(T'X)^+(T'X) = I$ and β can be estimated by $(T'X)^+(T'Y)$.

(2)

Unless $k = 1$, the code developed in `weighted_regression.ipynb` can't be used as it assumes β is $m \times 1$ and y and u are $N \times 1$.

(3)

In `PS1/code/functions.py`, there is a function titled `pt_5_6()`, which extends the weighted regression notebook. There, we are able to estimate β with k dependent variables, m independent variables, and a sample size of N. This function is executed in the jupyter notebook that can be found at `PS1/code/pset1_notebook.ipynb` under section 5.

(4)

We are given that $y = X\beta + u$ with $ET'u = 0$. Consider the following form, $T'y = T'X\beta + T'u$, which can be rearranged as $(T'X)^+T'(y - u) = (T'X)^+T'X\beta$. If we assume $(T'X)^+T'X$ is I almost surely, then we can write it as $\beta = (T'X)^+T'(y - u)$. So in finite sample, $\hat{\beta}$ is going to be distributed as a transformation of the random variables (T, X, y, u). The exact distribution is dependent on the distributions of (T, X, y, u) which aren't assumed in this problem and the transformation.

6. SUR

(1)

The problem description gives us that u is $N \times k$ (from comformability). Denote $u_{i,k}$ as the i^{th} row and j^{th} column of u. Now the covariance matrix $cov(u|X) = 0$ is the collection of all cross covariances within u. If we think of the "N" dimension as the different equations and the "k" dimension as the different observations, we are often interested in specifying if there is correlation accross equations and/or observations.

Homoskedasticity usually refers to the assumption that we have no correlation across observations. So $E[u_{i,k}, u_{i,l}|X] = 0$ for all $k, l \in \{1, 2, \ldots, N\}$ and $i \in \{1, 2, \ldots, k\}$. Whereas we might want to assume that equations are correlated, such that $E[u_{i,k}, u_{j,k}|X] = \sigma_{i,j,k}$ for all $k \in \{1, 2, \ldots, N\}$ and $i, j \in \{1, 2, \ldots, k\}$[2]. So now lets look at the covariance matrix, $cov(u|X) = E[uu^T|X] = \Omega$. From matrix multiplication, we know that $\Omega_{i,j} = E[\sum_{l=1}^{k} u_{i,l}u_{j,l}|X] = \sum_{l=1}^{k} E[u_{i,l}u_{j,l}|X]$. Notice that across observation correlations do not show up in Ω, verifying the description that Ω is a generalization of the assumption of homoskedasticity.

(2)

We adapted the code at part 5 to set X = T. The function that generates the data and runs the estimation procedure is the `pt_5_6` function in the `PS1/code/functions.py` file. The code is run in the jupyter notebook that can be found in `PS1/code/pset1_notebook.ipynb` under section 6.

(3)

When solving the system of equations with $X = T$, the OLS estimates of each equation are the same as the estimates in Part (2). This result can also be seen in the notebook output[3]. Note though, setting $X = T$ isn't the optimal estimator if we know the covariance structure. For known Ω, the FGLS estimator where $T = X\Omega^{-1}$ would be a better estimator, as it is a more efficient estimator.

2. Note that it is sufficient to look at the expectation since $Cov(a, b) = E[a, b] - E[a]E[b]$ and $E[u_{i,j}] = 0$.
3. Same location and line as the 6.2. code references.

7. Food Expenditures in India

Note: The code for this section can be found in two places:

- The code to generate the figures can be found in the jupyter notebook located at `PS1/code/pset1_notebook.ipynb` of the GitHub repository. The block of code for this section is under "7. Food Expenditures in India".
- The `KernelDensityEstimator` class can be found at `PS1/code/classes.py`.

(1)

In Figure 1, we plot the gaussian kernel with two different bandwidth sizes: (1) h=10000 and (2) h=100. Notice that the fine bandwidth fits the data much closer, but has potentially undesirable variation around the peak and right-tail. On the other hand, h=10000 is smooth but misses most of the distribution. In order to tune the bandwidth level, we used Silverman's rule for determining the bandwith and plotted the kernel density in Figure 2. Here we see that the density fits the distribution well, not having the kinks or flatness that the previous two bandwidth levels had. Overall a drawback of this method is how dependent the resulting density is on the bandwidth and kernel that is used.

Note that a better way of choosing a bandwidth would be to use the leave-one-out estimator as described in the lecture to perform hyper-parameter tuning. However, since this would take more time, and the problem stated that we don't need to do anything formal, we decided not to do this.

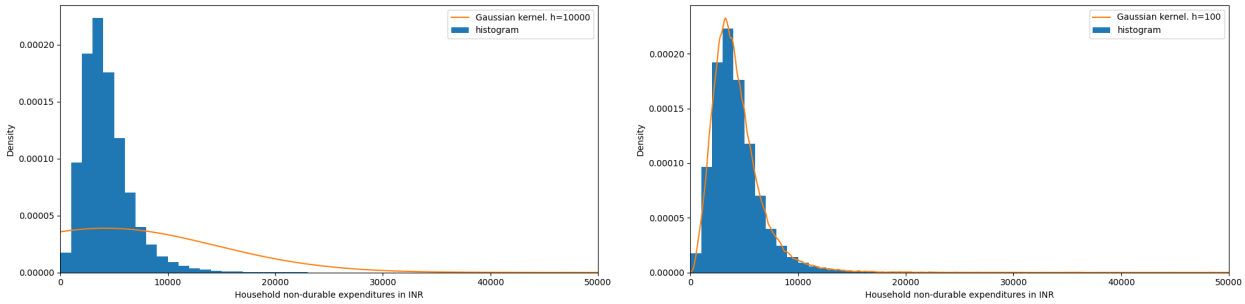


Figure 1: Kernel density estimation using a gaussian kernel with different bandwidth sizes. The sub-figure on the left, uses a bandwidth of 10000, which is very coarse and leads to a not desiarable fit. The sub-figure to the right has a very fine bandwidth of 100 which leads to more kinks than desirable.

(2)

From the previous section, our favorite kernel and bandwidth is the gaussian kernel with the Silverman bandwidth. For the transformation, we want to transform the variable x into $y = g(x) = \log(x)$. Note that $supp(x) = [0, \infty)$, but $supp(g(x)) = R$. Also note that $g^{-1}(y) = e^y$ and $\frac{\partial g^{-1}(y)}{\partial y} = e^y$. So from the inverse Jacobian rule:

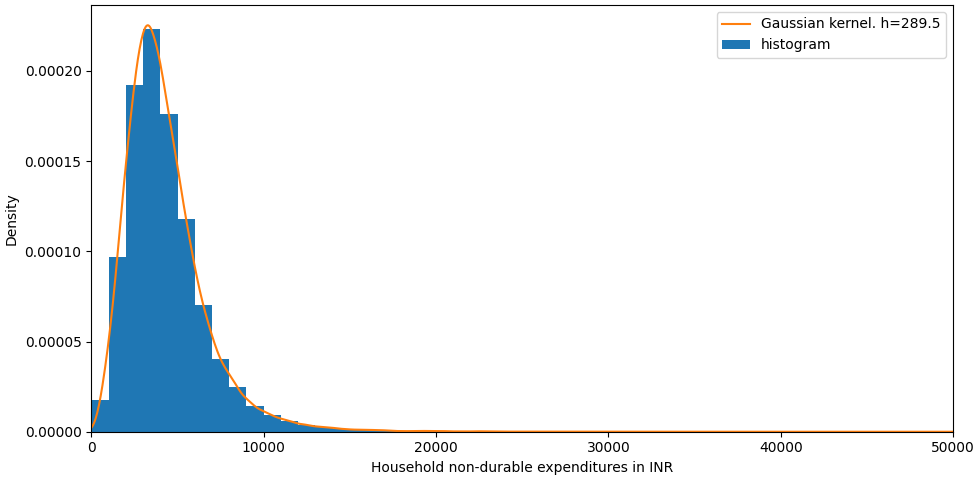


Figure 2: Kernel density estimation using a gaussian kernel with the bandwidth size set to Silverman's rule.

$$f_Y(y) = f_X(g^{-1}(y))|\frac{\partial g^{-1}(y)}{\partial y}| = f_X(e^y)e^y \quad y \in R$$

Replacing $f_X(\cdot)$ with $\hat{f}^h(x)$ and setting h to be the Silverman bandwidth, we can generate kernel density. Figure 3 plots the transformed density, it is the orange solid line.

(3)

Choosing the Gaussian kernel and the Silverman's bandwidth on the logged data, we generated the KDE and plotted it using red dots in Figure 3. We can see that both approaches provide an almost identical KDE distribution.

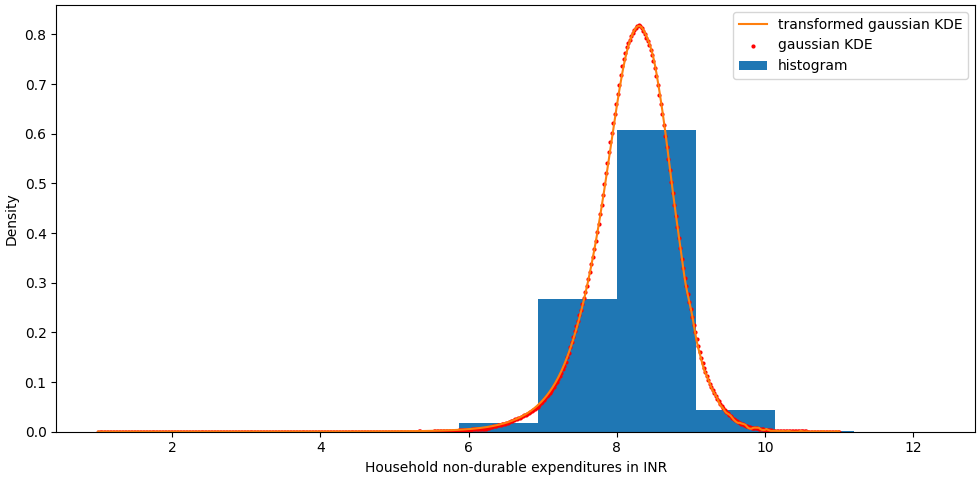


Figure 3: Kernel density estimation using a gaussian kernel. The variable is logged household expenditures. The two estimates are for the transformed and the directly estimated KDE.

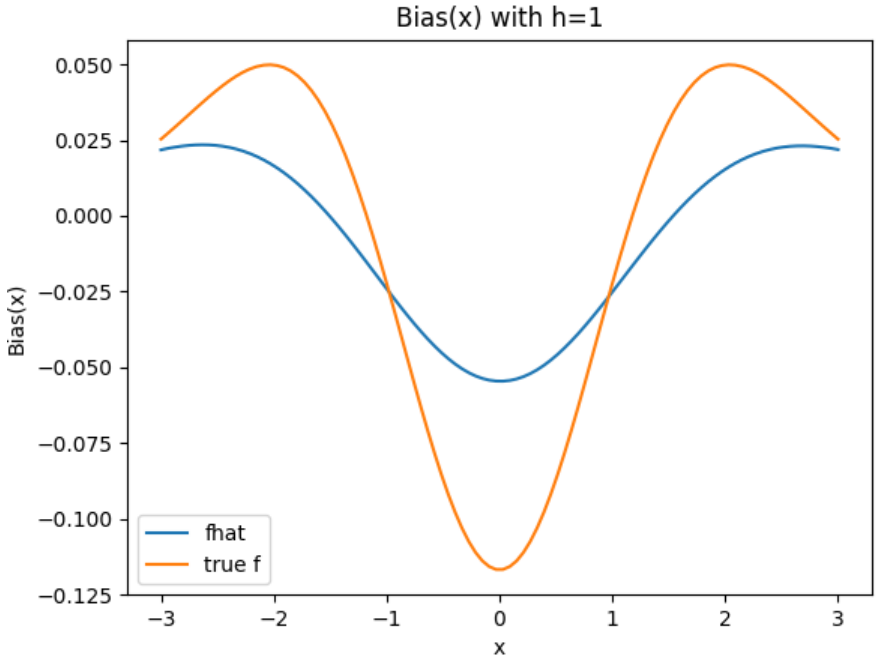
8. Kernel Bias Estimator

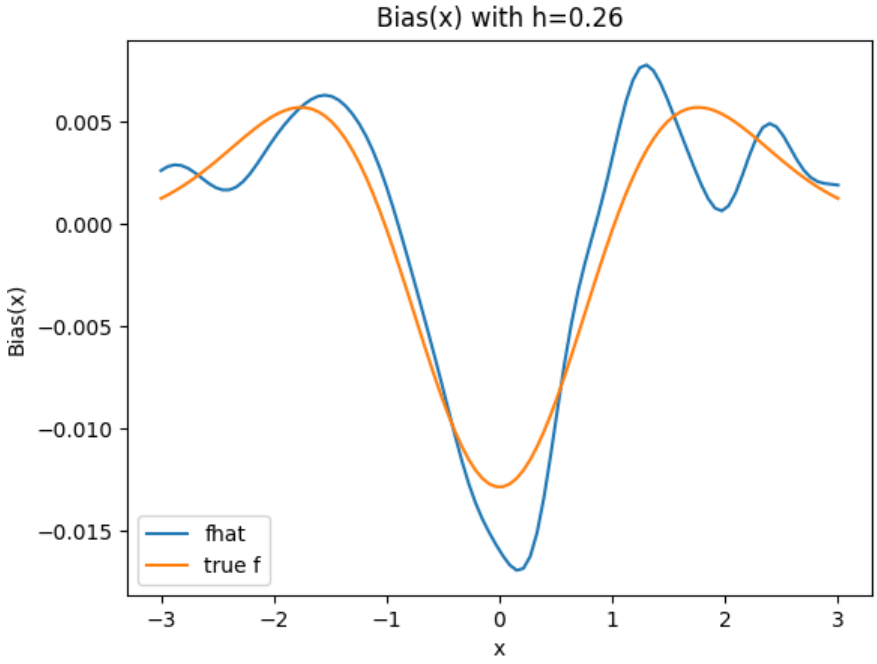
In our Jupyter Notebook, we generate a standard normal variable and use KDE to estimate the density. The bias of the KDE estimator is

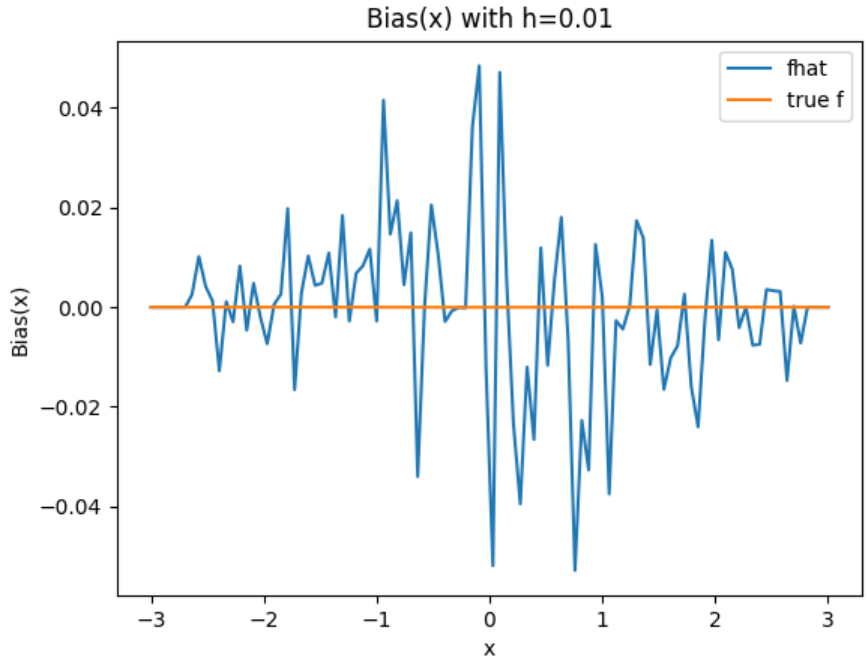
$$\begin{aligned} Bias(x) &= E[\hat{f}(x)] - f(x) \\ &= \int k(u) f(x + hu) du - f(x) \end{aligned}$$

As shown in class. Note that as $h \to 0$, the bias term above goes to 0. The problem asks us to consider estimating $Bias(x)$ using $\hat{f}(x)$, where $\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^{n} k(\frac{X_i - x}{h})$.

Speaking to Ethan after class, our understanding was that we should analyze what happens to the bias using simulations. The following three plots show the difference between bias (as functions of x) when we use the true density and the estimated density under three different bandwiths. Note that the Silverman bandwith is close to 0.25.







There are a few interesting things to note in the plots above. Importantly, under the Silverman optimal bandwidth, we can see that the estimate $\hat{bias}(x)$ is close to the true $bias(x)$ when we use $\hat{f}$. However, with a value for h that is either too high or too low, our estimate of bias deteriorates. A particularly interesting problem is that as $h \to 0$, $bias(x) \to 0$, but when we use use $\hat{f}$, $\hat{bias}(x) \not\to 0$.