Chapter 4: Comparison of Regularized ML and SM Density Estimators in $\mathcal{Q}_{\mathrm{ker}}$

After discussing the early stopping SM density estimator and comparing it with the penalized SM density estimator in the preceding chapter, we now turn to comparing these two kinds of regularized SM density estimators with the penalized ML density estimator. In particular, we would like to understand what differences the regularized SM density estimators and the penalized ML density estimator have and why they have such differences.

In Section 4.1, we focus on the penalized ML density estimator. We will establish the existence and the uniqueness of the minimizer of the penalized NLL loss functional. In spite of its existence, similar to the discussion in Gu and Qiu (1993), such a minimizer in an infinite-dimensional RKHS is *not* computable. Instead, we seek a finite-dimensional subspace of $\mathcal{H}$ to approximate this minimizer. In order to ensure the comparability of SM and ML density estimators, we minimize the (penalized) SM loss functional over this finite-dimensional subspace as well, and discuss how to compute the penalized and early stopping SM density estimators in it in Section 4.2. We then discuss the similarities and differences between regularized SM density estimators and penalized ML density estimator via numerical examples in Section 4.3. Finally, we explain why they have the observed differences in Section 4.4.

4.1 Penalized ML Density Estimator

In this section, we look at the penalized ML density estimator, which is obtained by minimizing

$$\widehat{J}_{\mathrm{NLL}}(f) + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2, \qquad \text{subject to } f \in \mathcal{F}, \tag{4.1}$$

where $\widehat{J}_{\mathrm{NLL}}(f) = A(f) - \frac{1}{n}\sum_{i=1}^n f(X_i)$ is the NLL loss functional we have seen in Chapter 2, and $\lambda \geq 0$ is the penalty parameter. Note that in (4.1), we choose the penalty functional to be $\widetilde{P}(f) := \frac{1}{2}\|f\|_{\mathcal{H}}^2$, for all $f \in \mathcal{H}$, which is to ensure the comparability with the regularized SM density estimators.

The following proposition established the existence and the uniqueness of the minimizer of (4.1).

Proposition 4.1. *Under Assumptions **(A1)** - **(A4)** in Chapter 2, for every* $\lambda > 0$*,* (4.1) *has a unique minimizer in* $\mathcal{F}$*.*

The proof of Proposition 4.1 is given in Section 4.5.1.

4.1.1 Failure of the Representer Theorem

Note that Proposition 4.1 only shows (4.1) has a unique minimizer, but gives no indication what such a minimizer is like.

Since minimizing (4.1) is a convex minimization problem over an infinite-dimensional RKHS, a classic tool of characterizing its minimizer is the representer theorem (Kimeldorf and Wahba, 1971; Schölkopf, Herbrich, and Smola, 2001), which states that, if $\widetilde{J} : \mathcal{H} \to \mathbb{R}$ is a convex loss functional that depends *only* on the evaluation of $f \in \mathcal{H}$ at data points $X_1, \cdots, X_n$, then

$$\min_{f\in\mathcal{H}}\left\{\widetilde{J}(f) + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2\right\} = \min_{f\in\mathrm{Span}\{k(X_1,\cdot),\cdots,k(X_n,\cdot)\}}\left\{\widetilde{J}(f) + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2\right\};$$

in other words, the minimizer of $\widetilde{J}(f) + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2$ over $\mathcal{H}$ can be represented as a linear combination of $k(X_1, \cdot), \cdots, k(X_n, \cdot)$. Here, we only consider a specific form of the penalty functional $\frac{\lambda}{2}\|f\|_{\mathcal{H}}^2$. The representer theorem covers the case where a general penalty functional is used; see Schölkopf, Herbrich, and Smola (2001) for a discussion.

The representer theorem is a powerful tool and reduces a convex minimization problem in an infinite-dimensional RKHS to the one in a finite-dimensional subspace, which effectively reduces the computational burden and improves the efficiency. However, note that, due the presence of A in (4.1), $\widehat{J}_{\mathrm{NLL}}$ does not only depend on the evaluation of f at data points $X_1, \cdots, X_n$, but also on that of f at *all* points in $\mathcal{X}$. This suggests the representer theorem may fail in characterizing the minimizer of (4.1). In fact, as the following example demonstrates, the representer theorem does fail.

Example 4.1. Let $\mathcal{X} = \mathbb{R}^2$ and the kernel function be $k(x, y) = (x^\top y)^2$ for all $x, y \in \mathbb{R}^2$, i.e., the homogeneous polynomial kernel of degree 2. We choose the base density to be $\mu(x) = \frac{1}{2\pi}\exp(-\frac{\|x\|_2^2}{2})$ for all $x \in \mathbb{R}^2$, i.e., the pdf of 2-dimensional Gaussian distribution with the zero mean and the identity covariance matrix. We fix $n = 1$ and let the single data point be $X_1 = (a, 0)^\top \in \mathbb{R}^2$. Also, let $\lambda > 0$.

Then, it can be shown that the minimizer of the corresponding penalized NLL loss functional

$$A(f) - f(X_1) + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2, \qquad \text{subject to } f \in \mathcal{F}, \tag{4.2}$$

takes on the form of

$$f^*(x) = b^* x_1^2 + c^* x_2^2, \qquad \text{for all } x = (x_1, x_2)^\top \in \mathbb{R}^2,$$

for some $b^* \neq 0$ and $c^* \neq 0$, but $\mathrm{Span}\{k(X_1, \cdot)\} \cap \mathcal{F}$ contains all functions of the form

$$f(x) = \gamma a^2 x_1^2, \quad \text{for all } x := (x_1, x_2)^\top \in \mathbb{R}^2 \text{ and } \gamma \in \mathbb{R} \text{ satisfying } \gamma a^2 < \frac{1}{2}.$$

Thus, $f^* \notin \text{Span}\{k(X_1, \cdot)\} \cap \mathcal{F}$ and we conclude the representer theorem fails.

Details about this example can be found in Section 4.5.2. ▶

4.1.2 Construction of a Finite-dimensional Approximating Space

Now, we have seen the representer theorem fails in characterizing the minimizer of (4.1). In addition, as is discussed by Gu and Qiu (1993), despite its existence and the uniqueness, such a minimizer in an infinite-dimensional $\mathcal{H}$ is *not* computable. In order to compute the penalized ML density estimator, we propose to seek a finite-dimensional subspace in $\mathcal{H}$ to approximate the minimizer of (4.1), which is the focus of the current section.

One idea is to choose the following n-dimensional subspace spanned by the kernel functions centered at $X_1, \cdots, X_n$,

$$\left\{ f \;\middle|\; f := \sum_{i=1}^{n} \beta_i k(X_i, \cdot), \beta_1, \cdots, \beta_n \in \mathbb{R} \right\}, \tag{4.3}$$

as the approximating space. This is the approach taken by Gu and Qiu (1993) and Gu (1993).

We propose a different approach here. Suppose $\mathcal{X} \subseteq \mathbb{R}$ for the moment and start with a set of grid points over $\mathcal{X}$, denoted by $\mathbf{X}_1 := \{w_1, \cdots, w_m\}$, that covers the range of data. We minimize the objective functional in (4.1) over

$$\widetilde{\mathcal{H}}_1 := \left\{ f \;\middle|\; f := \sum_{j=1}^{m} \beta_j k(w_j, \cdot), \beta_j \in \mathbb{R}, w_j \in \mathbf{X}_1, \text{ for all } j = 1, \cdots, m \right\}.$$

Then, we insert an additional grid point at the midpoint of two adjacent points in $\mathbf{X}_1$, forming

$$\mathbf{X}_2 := \left\{ w_1, w_{1.5}, w_2, \cdots, w_{m-1}, w_{\frac{2m-1}{2}}, w_m \right\},$$

where $w_{\frac{2j-1}{2}} = \frac{1}{2}(w_j + w_{j+1})$ for all $j = 1, \cdots, m-1$, and minimize (4.1) over

$$\widetilde{\mathcal{H}}_2 := \left\{ f \;\middle|\; f := \sum_{j \in \{1, 1.5, 2, \cdots, m\}} \beta_j k(w_j, \cdot), \beta_j \in \mathbb{R}, w_j \in \mathbf{X}_2, \text{ for all } j = 1, 1.5, 2, \cdots, m \right\}.$$

Let $\tilde{f}_{\mathrm{ML},\ell} := \arg\min_{f \in \widetilde{\mathcal{H}}_\ell} \big\{ \widehat{J}_{\mathrm{NLL}}(f) + \frac{\lambda}{2} \|f\|_{\mathcal{H}}^2 \big\}$ for $\ell = 1, 2$. Then, we compute the following relative difference

$$\left| \frac{\big[\widehat{J}_{\mathrm{NLL}}(\tilde{f}_{\mathrm{ML},2}) + \frac{\lambda}{2}\|\tilde{f}_{\mathrm{ML},2}\|_{\mathcal{H}}^2\big] - \big[\widehat{J}_{\mathrm{NLL}}(\tilde{f}_{\mathrm{ML},1}) + \frac{\lambda}{2}\|\tilde{f}_{\mathrm{ML},1}\|_{\mathcal{H}}^2\big]}{\widehat{J}_{\mathrm{NLL}}(\tilde{f}_{\mathrm{ML},1}) + \frac{\lambda}{2}\|\tilde{f}_{\mathrm{ML},1}\|_{\mathcal{H}}^2} \right|. \tag{4.4}$$

Since $\mathbf{X}_1 \subset \mathbf{X}_2$ and $\widetilde{\mathcal{H}}_1 \subset \widetilde{\mathcal{H}}_2$, we must have $\widehat{J}_{\mathrm{NLL}}(\tilde{f}_{\mathrm{ML},1}) + \frac{\lambda}{2}\|\tilde{f}_{\mathrm{ML},1}\|_{\mathcal{H}}^2 \geq \widehat{J}_{\mathrm{NLL}}(\tilde{f}_{\mathrm{ML},2}) + \frac{\lambda}{2}\|\tilde{f}_{\mathrm{ML},2}\|_{\mathcal{H}}^2$.

We repeat the process described above: keep inserting additional grid points at the midpoint of two adjacent points, minimize the penalized NLL loss functional over the new finite-dimensional subspace with the updated grid points, and stop until the relative difference of the values of the penalized NLL loss functional between two consecutive sets of grid points does not exceed a pre-specified tolerance parameter. The algorithm is provided in Algorithm 4.1.

If $\mathcal{X} \subseteq \mathbb{R}^d$ for $d > 1$, we can generalize the procedure described above accordingly. However, one has to be careful when adding additional points, since, with $d > 1$, each grid point has up to 2^d adjacent points, and the number of grid points being added each time can grow exponentially. The computational burden will also increase accordingly as d increases. This is one example of the infamous curse of dimensionality and is an unsatisfactory feature of our approach.

One advantage of our approach, comparing to the approach by Gu and Qiu (1993) and Gu (1993), is that our approach can yield an even smaller minimum of the penalized NLL loss functional than theirs, as we will see via numerical examples in Section 4.1.4.

Algorithm 4.1 Determining the finite-dimensional subspace of approximating the minimizer of $\widehat{J}_{\mathrm{NLL}}(f) + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2$ over $f \in \mathcal{H}$

```
Require:
    • $X_1, \cdots, X_n$, the data from $p_0$;
    • $\mathbf{X}_1 := \{w_1, \cdots, w_m\}$, the first set of grid points;
    • $\epsilon > 0$, the tolerance parameter.
Let $\widetilde{\mathcal{H}}_1 = \big\{f \mid f := \sum_j \beta_j k(w_j, \cdot), \beta_j \in \mathbb{R}, w_j \in \mathbf{X}_1, \text{ for all } j = 1, 2, \cdots, m\big\}$ and compute $\tilde{f}_{\mathrm{ML},1} := \arg\min_{f \in \widetilde{\mathcal{H}}_1} \big\{\widehat{J}_{\mathrm{NLL}}(f) + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2\big\}$;
Form $\mathbf{X}_2$, which is obtained by adding a grid point at the midpoint of two adjacent points in $\mathbf{X}_1$;
Compute $\tilde{f}_{\mathrm{ML},2} := \arg\min_{f \in \widetilde{\mathcal{H}}_2} \big\{\widehat{J}_{\mathrm{NLL}}(f) + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2\big\}$, where $\widetilde{\mathcal{H}}_2 := \{f \mid f := \sum_j \beta_j k(w_j, \cdot), \beta_j \in \mathbb{R}, w_j \in \mathbf{X}_2\}$;
Compute error using (4.4);
while error $> \epsilon$ do
    Let $\mathbf{X}_1 = \mathbf{X}_2$, and form a new $\mathbf{X}_2$ by adding a grid point at the midpoint of two adjacent points in $\mathbf{X}_1$;
    Let $\tilde{f}_{\mathrm{ML},1} = \tilde{f}_{\mathrm{ML},2}$, and compute $\tilde{f}_{\mathrm{ML},2} := \arg\min_{f \in \widetilde{\mathcal{H}}_2} \big\{\widehat{J}_{\mathrm{NLL}}(f) + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2\big\}$, where $\widetilde{\mathcal{H}}_2 := \{f \mid f := \sum_j \beta_j k(w_j, \cdot), \beta_j \in \mathbb{R}, w_j \in \mathbf{X}_2\}$;
    Update error using (4.4).
return $\mathbf{X}_1$.
```

In our description of the process above, one key component we have not discussed is how to compute the minimizer of the penalized NLL loss functional for a given finite-dimensional approximating subspace. As we have discussed in Chapter 2, the main difficulty is to handle A and its derivatives efficiently. In the next section, we propose an algorithm to approximate the values of A and its derivatives and compute the minimizer of the penalized NLL loss functional.

4.1.3 Computation of the Minimizer of the Penalized NLL Loss Functional

Our goal of this section is to provide an adaptive algorithm to compute the minimizer of $\widehat{J}_{\mathrm{NLL}}(f) + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2$ over a given finite-dimensional approximating subspace

$$\widetilde{\mathcal{H}} := \left\{ f \;\middle|\; f := \sum_{j=1}^{m} \beta_j k(w_j, \cdot), \beta_j \in \mathbb{R} \text{ for all } j = 1, 2, \cdots, m \right\},$$

where $\{w_1, \cdots, w_m\} \subset \mathcal{X}$ is a set of specified grid points.

Since any $f \in \widetilde{\mathcal{H}}$ can be written as $f = \sum_{j=1}^{m} \beta_j k(w_j, \cdot)$, we can rewrite the penalized NLL loss functional as

$$\widetilde{J}_{\mathrm{NLL},\lambda}(\boldsymbol{\beta}) := \widehat{J}_{\mathrm{NLL}}(f) + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2 = \widetilde{A}(\boldsymbol{\beta}) - \boldsymbol{\beta}^\top \left(\frac{1}{n}\mathbf{K}_1 \mathbf{1}_n\right) + \frac{\lambda}{2}\boldsymbol{\beta}^\top \mathbf{K}_2 \boldsymbol{\beta}, \tag{4.5}$$

for all $\boldsymbol{\beta} := (\beta_1, \cdots, \beta_m)^\top \in \mathbb{R}^m$, where

$$\widetilde{A}(\boldsymbol{\beta}) := A(f) = \log\left(\int_{\mathcal{X}} \mu(x) \exp\left(\sum_{j=1}^{m} \beta_j k(w_j, x)\right) \mathrm{d}x\right),$$

the (j, i)-entry of $\mathbf{K}_1 \in \mathbb{R}^{m\times n}$ is $k(w_j, X_i)$, the (j, j')-entry of $\mathbf{K}_2 \in \mathbb{R}^{m\times m}$ is $k(w_j, w_{j'})$, for all $j, j' = 1, \cdots, m$, $i = 1, \cdots, n$, and $\mathbf{1}_n := (1, \cdots, 1)^\top \in \mathbb{R}^n$.

By a similar argument to Proposition 4.1, we can show $\widetilde{J}_{\mathrm{NLL},\lambda}$ has a unique minimizer in $\mathbb{R}^m$. To compute this minimizer, we use the gradient descent algorithm with the constant step size. Starting from $\boldsymbol{\beta}_{\mathrm{NLL}}^{(0)} \in \mathbb{R}^m$, the gradient descent iterates are

$$\boldsymbol{\beta}_{\mathrm{NLL}}^{(t+1)} = \boldsymbol{\beta}_{\mathrm{NLL}}^{(t)} - \tau \nabla \widetilde{J}_{\mathrm{NLL},\lambda}(\boldsymbol{\beta}_{\mathrm{NLL}}^{(t)}), \qquad \text{for all } t \in \mathbb{N}_0,$$

where $\tau > 0$ is an appropriately chosen constant step size, and

$$\nabla \widetilde{J}_{\mathrm{NLL},\lambda}(\boldsymbol{\beta}) = \nabla \widetilde{A}(\boldsymbol{\beta}) - \frac{1}{n}\mathbf{K}_1\mathbf{1}_n + \lambda \mathbf{K}_2\boldsymbol{\beta}. \tag{4.6}$$

We terminate the algorithm when $\|\nabla \widetilde{J}_{\mathrm{NLL},\lambda}(\boldsymbol{\beta}_{\mathrm{NLL}}^{(t)})\|_2/\sqrt{m}$ is less than a pre-specified tolerance parameter.

The j-th component of $\nabla \widetilde{A}(\boldsymbol{\beta})$ in (4.6) is

$$\int_{\mathcal{X}} k(w_j, x)\mu(x)\exp\left(\sum_{\ell=1}^{m} \beta_\ell k(w_\ell, x) - \widetilde{A}(\boldsymbol{\beta})\right)\mathrm{d}x, \tag{4.7}$$

which is $\mathbb{E}_{q_f}[k(w_j, X)]$ with $f \in \widetilde{\mathcal{H}}$, for all $j = 1, \cdots, m$. Exact computation of (4.7) can be difficult. We propose the batch Monte Carlo method in the next section to approximate them.

4.1.3.1 Batch Monte Carlo Approximation of $\nabla \widetilde{A}(\boldsymbol{\beta})$

Since μ is a density function, let $Y_1, \cdots, Y_B$ be i.i.d samples from μ. We can approximate $\exp(\widetilde{A}(\boldsymbol{\beta}))$ and (4.7) using the Monte Carlo method by

$$\widehat{\exp(\widetilde{A}(\boldsymbol{\beta}))} := \frac{1}{B}\sum_{b=1}^{B} \exp\left(\sum_{\ell=1}^{m} \beta_\ell k(w_\ell, Y_b)\right) \tag{4.8}$$

and

$$\frac{1}{B}\sum_{b=1}^{B} k(w_j, Y_b)\exp\left(\sum_{\ell=1}^{m} \beta_\ell k(w_\ell, Y_b) - \log\bigl(\widehat{\exp(\widetilde{A}(\boldsymbol{\beta}))}\bigr)\right), \tag{4.9}$$

respectively.

What we have so far is just the *vanilla* Monte Carlo method. It requires us to choose an appropriate value of B so that we can obtain satisfactory approximations of the desired quantities. This may not be a trivial task in practice.

In order to better choose the value of B under different scenarios and control the quality of the approximations, we propose the batch Monte Carlo method. The main idea is to keep drawing a batch of random samples from μ and updating the approximations until the standard deviation of approximations is less than a prespecified tolerance parameter (see Steps 4, 10, 15 and 21 in Algorithm 4.2 for the calculation of the stopping criterion). The details are given in Algorithm 4.2.

Algorithm 4.2 Batch Monte Carlo method

```
    Require:
      • $w_1, \cdots, w_m$, grid points at which kernel functions are centered;
      • a sampler to draw i.i.d samples from $\mu$;
      • $\boldsymbol{\beta}$, the coefficient vector of kernel functions;
      • $B$, batch size;
      • $\epsilon > 0$, tolerance parameter to determine when to stop sampling.
    /* Approximation of $\exp(\widetilde{A}(\boldsymbol{\beta}))$ */
 1: Draw $Y_1, \cdots, Y_B$ from $\mu$ and set batch_cnt $= 1$;        ▷ Draw the first batch
 2: Approximate $\exp(\widetilde{A}(\boldsymbol{\beta}))$ using $Y_1, \cdots, Y_B$ according to (4.8); let the resulting
    approximation be output1;
 3: Set mean_sq $= \frac{1}{B}\sum_{b=1}^{B} \exp\big(2\sum_{\ell=1}^{m} \beta_\ell k(w_\ell, Y_b)\big)$;
 4: Set se1 $= \sqrt{\texttt{mean\_sq} - \texttt{output1}^2}$;
 5: while se1 $> \epsilon$ do
 6:     Draw $Y_1, \cdots, Y_B$ from $\mu$;                              ▷ Draw more batches
 7:     Approximate $\exp(\widetilde{A}(\boldsymbol{\beta}))$ using $Y_1, \cdots, Y_B$ according to (4.8); let the resulting
    approximation be output1_inter;
 8:     Update

          output1 $=$ (output1_inter $+$ output1 $\times$ batch_cnt)/(batch_cnt $+ 1$);

 9:     Update

    mean_sq $= \left(\frac{1}{B}\sum_{b=1}^{B}\exp\left(2\sum_{\ell=1}^{m}\beta_\ell k(w_\ell, Y_b)\right) + \texttt{mean\_sq} \times \texttt{batch\_cnt}\right) \Big/ (\texttt{batch\_cnt} + 1)$;

10:     Update se1 $= \sqrt{\texttt{mean\_sq} - \texttt{output1}^2}$;
11:     Update batch_cnt $=$ batch_cnt $+ 1$;
    /* Approximation of $\exp(\widetilde{A}(\boldsymbol{\beta}))$ is output1 */
```

Algorithm 2 Batch Monte Carlo method (continued)

```
/* Approximation of $\nabla\widetilde{A}(\boldsymbol{\beta})$ */
Draw $Y_1, \cdots, Y_B$ from $\mu$ and set $\texttt{batch\_cnt} = 1$;    ▷ *Draw the first batch*
Approximate $[\nabla\widetilde{A}(\boldsymbol{\beta})]_j$ using $Y_1, \cdots, Y_B$ according to (4.9), for all $j = 1, \cdots, m$; let the resulting approximation of $\nabla\widetilde{A}(\boldsymbol{\beta})$ be $\texttt{output2}$;
Let $\texttt{mean\_sq}_j = \frac{1}{B}\sum_{b=1}^{B}\big(k(w_j, Y_b)\exp\big(\sum_{\ell=1}^{m}\beta_\ell k(w_\ell, Y_b) - \log(\texttt{output1})\big)\big)^2$ for all $j = 1, \cdots, m$;
Let $\texttt{se\_grad}_j = \sqrt{\texttt{mean\_sq}_j - \texttt{output2}_j^2}$ and $\texttt{se\_grad} = \sqrt{\frac{1}{m}\sum_{j=1}^{m}\texttt{se\_grad}_j^2}$;
while $\texttt{se\_grad} > \epsilon$ do
    Draw $Y_1, \cdots, Y_B$ from $\mu$;    ▷ *Draw more batches*
    Approximate $[\nabla\widetilde{A}(\boldsymbol{\beta})]_j$ using $Y_1, \cdots, Y_B$ according to (4.9), for all $j = 1, \cdots, m$; let the resulting approximation of $\nabla\widetilde{A}(\boldsymbol{\beta})$ be $\texttt{output2\_inter}$;
    Update
        $$\texttt{output2} = \big(\texttt{output2\_inter} + \texttt{output2} \times \texttt{batch\_cnt}\big)\big/(\texttt{batch\_out} + 1);$$
    Update
        $$\texttt{mean\_sq}_j = \left(\frac{1}{B}\sum_{b=1}^{B}\left(k(w_j, Y_b)\exp\left(\sum_{\ell=1}^{m}\beta_\ell k(w_\ell, Y_b) - \log(\texttt{output1})\right)\right)^2 + \texttt{mean\_sq}_j \times \texttt{batch\_out}\right)\Bigg/(\texttt{batch\_out} + 1),$$
    for all $j = 1, \cdots, m$;
    Update
        $$\texttt{se\_grad}_j = \sqrt{\texttt{mean\_sq}_j - \texttt{output2}_j^2}, \quad \text{and} \quad \texttt{se\_grad} = \sqrt{\frac{1}{m}\sum_{j=1}^{m}\texttt{se\_grad}_j^2};$$
    Update $\texttt{batch\_cnt} = \texttt{batch\_cnt} + 1$;
return $\texttt{output1} \in \mathbb{R}$ and $\texttt{output2} \in \mathbb{R}^m$.
```

4.1.3.2 Gradient Descent Algorithm to Minimize $\widetilde{J}_{\mathrm{NLL},\lambda}$

With the batch Monte Carlo method described in the preceding section, we provide the complete gradient descent algorithm to minimize $\widetilde{J}_{\mathrm{NLL},\lambda}$ over $\mathbb{R}^m$ in Algorithm 4.3.

Algorithm 4.3 Gradient descent algorithm to minimize $\widetilde{J}_{\mathrm{NLL},\lambda}$ over $\mathbb{R}^m$

```
Require:
  • $X_1, \cdots, X_n$, data;
  • $w_1, \cdots, w_m$, points at which kernel functions are centered;
  • $\lambda \geq 0$, penalty parameter;
  • $\boldsymbol{\beta}_{\mathrm{NLL}}^{(0)} \in \mathbb{R}^m$, starting point;
  • $\tau > 0$, step size;
  • $\epsilon > 0$, the tolerance parameter to determine when to terminate the gradient
    descent algorithm.
Compute $\mathbf{K}_1 \in \mathbb{R}^{m \times n}$ and $\mathbf{K}_2 \in \mathbb{R}^{m \times m}$ that appear in (4.5);
Set $\texttt{coef} = \boldsymbol{\beta}_{\mathrm{NLL}}^{(0)}$;
Compute $\nabla \widetilde{A}(\texttt{coef})$ using Algorithm 4.2;
Compute $\nabla \widetilde{J}_{\mathrm{NLL},\lambda}(\texttt{coef})$ by (4.6);
Compute $\texttt{error} = \|\nabla \widetilde{J}_{\mathrm{NLL},\lambda}(\texttt{coef})\|_2 / \sqrt{m}$;
while $\texttt{error} > \varepsilon$ do
    Update $\texttt{coef} = \texttt{coef} - \tau \nabla \widetilde{J}_{\mathrm{NLL},\lambda}(\texttt{coef})$;
    Compute $\nabla \widetilde{A}(\texttt{coef})$ using Algorithm 4.2;
    Compute $\nabla \widetilde{J}_{\mathrm{NLL},\lambda}(\texttt{coef})$ by (4.6);
    Update $\texttt{error} = \|\nabla \widetilde{J}_{\mathrm{NLL},\lambda}(\texttt{coef})\|_2 / \sqrt{m}$.
return coef.
```

4.1.4 Numerical Illustration

We use the `waiting` variable in the Old Faithful Geyser dataset to illustrate the algorithms introduced in the preceding two sections — one on seeking an appropriate finite-dimensional approximating space, and the other on computing the minimizer of the penalized NLL loss functional.

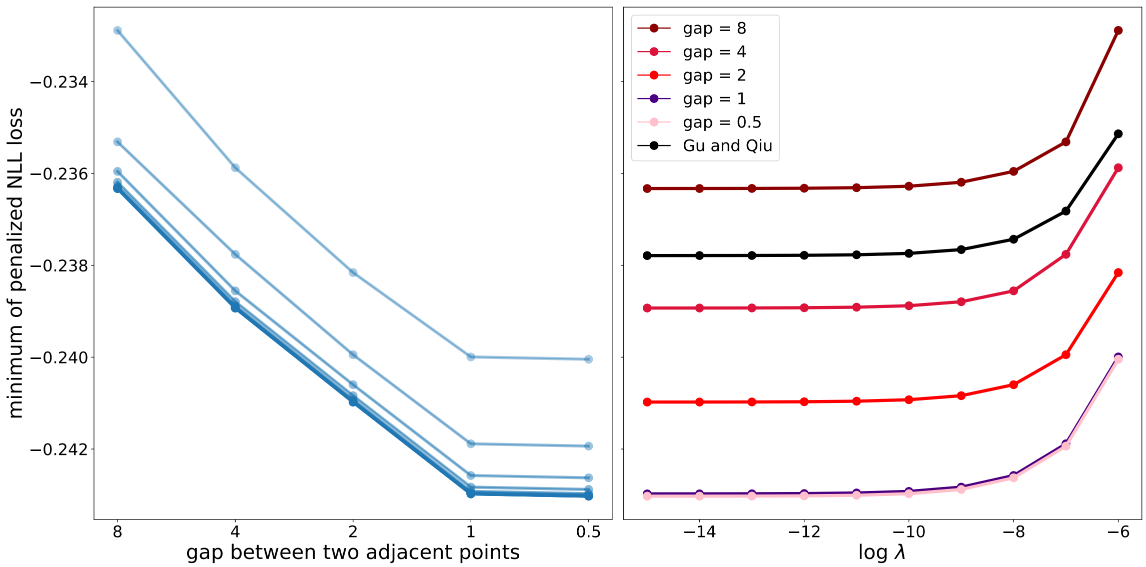

Figure 4.1: Left panel: the minimum of $\widetilde{J}_{\mathrm{NLL},\lambda}$ against the gap between two adjacent points at which kernel functions are centered in different choices of finite-dimensional approximating subspace. Different opacity indicates different values of λ, and the more opaque line indicates the smaller λ value. Right panel: the minimum of $\widetilde{J}_{\mathrm{NLL},\lambda}$ against different values of $\log \lambda$.

We start from the set of grid points $\{1, 9, 17, \cdots, 201\}$, where the difference between two adjacent grid points is 8. Note that this starting set of grid points contains 26 points and covers the range of raw data. In approximating $\nabla \widetilde{A}$ using Algorithm 4.2, we choose the batch size to be $B = 5000$ and the tolerance parameter to be 10^{-3}. In applying Algorithm 4.3, we choose the tolerance parameter to be 10^{-4}. The gradient descent algorithm always starts from the zero vector. The values of λ we choose are $e^{-15}, e^{-14}, \cdots, e^{-6}$.

The left panel of Figure 4.1 shows the minimum of $\widetilde{J}_{\mathrm{NLL},\lambda}$ against the gap between two adjacent grid points at which kernel functions are centered in different choices of finite-dimensional approximating subspace. For each value of λ, we see that, as the gap between two adjacent grid points becomes smaller and smaller, the minimum of

$\widetilde{J}_{\text{NLL},\lambda}$ keeps decreasing. When the difference between two adjacent grid points is 1, if we further insert additional grid points, the further reduction on the minimum of $\widetilde{J}_{\text{NLL},\lambda}$ becomes negligible. Thus, for this particular `waiting` data, we use the following subspace as our final choice of the finite-dimensional approximating subspace

$$\widetilde{\mathcal{H}} := \left\{ f \,\middle|\, f := \sum_{j=1}^{201} \beta_j k(w_j, \cdot), \beta_j \in \mathbb{R}, w_j = j, \text{ for all } j = 1, \cdots, 201 \right\}. \tag{4.10}$$

The right panel of Figure 4.1 shows the minimum of $\widetilde{J}_{\text{NLL},\lambda}$ against different values of $\log \lambda$. Similar to our observations from the left panel, as the gap between two adjacent grid points becomes smaller and smaller, the minimum of $\widetilde{J}_{\text{NLL},\lambda}$ keeps decreasing for all choices of λ. Comparing the minimums of $\widetilde{J}_{\text{NLL},\lambda}$ over the finite-dimensional subspaces with the gap between two adjacent grid points being 1 and 0.5, the difference is very tiny. In addition, the black curve in this panel shows the minimums of $\widetilde{J}_{\text{NLL},\lambda}$ using the finite-dimensional approximating subspace (4.3) that Gu and Qiu (1993) and Gu (1993) proposed. It is obvious that our approach yields a smaller minimum of $\widetilde{J}_{\text{NLL},\lambda}$ than their approach does, implying the superiority of our approach.

4.2 Regularized SM Density Estimators with $f \in \widetilde{\mathcal{H}}$

We minimize the penalized NLL loss functional over a finite-dimensional approximating space. Since our ultimate goal of this chapter is to compare the regularized SM density estimators and the penalized ML density estimator and understand their similarities and differences, in order to ensure the comparability, we also minimize the (penalized) SM loss functional over the same finite-dimensional subspace of $\mathcal{H}$. We describe how to achieve this in the current section.

Throughout this section, we again let

$$\widetilde{\mathcal{H}} := \left\{ f \;\middle|\; f := \sum_{j=1}^{m} \beta_j k(w_j, \cdot), \beta_j \in \mathbb{R} \text{ for all } j = 1, 2, \cdots, m \right\}$$

be a finite-dimensional approximating subspace, where $\{w_1, \cdots, w_m\} \subset \mathcal{X}$ is a set of specified grid points. We first derive the expression of $\widehat{J}_{\mathrm{SM}}(f)$ with $f \in \widetilde{\mathcal{H}}$, where $\widehat{J}_{\mathrm{SM}}$ is the SM loss functional given by (2.13) in Chapter 2.

Proposition 4.2. *Let $f = \sum_{j=1}^{m} w_j k(w_j, \cdot) \in \widetilde{\mathcal{H}}$. Then, the SM loss functional can be written as*

$$\widetilde{J}_{\mathrm{SM}}(\boldsymbol{\beta}) := \frac{1}{2}\boldsymbol{\beta}^\top \left(\frac{1}{n}\mathbf{S}\mathbf{S}^\top\right)\boldsymbol{\beta} - \boldsymbol{\beta}^\top \mathbf{t},$$

where $\boldsymbol{\beta} := (\beta_1, \cdots, \beta_m)^\top \in \mathbb{R}^m$, the $(j, (i-1)d + u)$-entry of $\mathbf{S} \in \mathbb{R}^{m \times (nd)}$ is $\langle k(w_j, \cdot), \partial_u k(X_i, \cdot)\rangle_{\mathcal{H}} = \partial_u k(X_i, w_j)$, and the j-th entry of $\mathbf{t} \in \mathbb{R}^m$ is

$$\langle \hat{z}, k(w_j, \cdot)\rangle = -\frac{1}{n}\sum_{i=1}^{n}\sum_{u=1}^{d}\big(\partial_u^2 k(X_i, w_j) + (\partial_u \log \mu)(X_i)\partial_u k(X_i, w_j)\big),$$

for all $j = 1, \cdots, m$, $i = 1, \cdots, n$ and $u = 1, \cdots, d$.

Proof of Proposition 4.2 can be found in Section 4.5.3.

4.2.1 Penalized SM Density Estimator with $f \in \widetilde{\mathcal{H}}$

In order to compute the penalized SM density estimator under the constraint $f \in \widetilde{\mathcal{H}}$, we minimize the penalized SM loss functional $\widehat{J}_{\mathrm{SM}}(f) + \frac{\rho}{2}\|f\|_{\mathcal{H}}^2$ subject to $f \in \widetilde{\mathcal{H}}$. The following proposition establishes the existence and the uniqueness of this minimizer, and discusses how to compute it.

Proposition 4.3. *The minimizer of $\widehat{J}_{\mathrm{SM}}(f) + \frac{\rho}{2}\|f\|_{\mathcal{H}}^2$ subject to $f \in \widetilde{\mathcal{H}}$, denoted by $\tilde{f}_{\mathrm{SM}}^{(\rho)}$, exists and is unique, and is given by*

$$\tilde{f}_{\mathrm{SM}}^{(\rho)} = \sum_{j=1}^{m} \beta_j^{(\rho)} k(w_j, \cdot),$$

where $\boldsymbol{\beta}_{\mathrm{SM}}^{(\rho)} := (\beta_1^{(\rho)}, \beta_2^{(\rho)}, \cdots, \beta_m^{(\rho)})^\top \in \mathbb{R}^m$ *satisfies*

$$\left(\frac{1}{n}\mathbf{S}\mathbf{S}^\top + \lambda \mathbf{K}_2\right)\boldsymbol{\beta}_{\mathrm{SM}}^{(\rho)} = \mathbf{t},$$

where $\mathbf{S} \in \mathbb{R}^{m\times(nd)}$ *and* $\mathbf{t} \in \mathbb{R}^m$ *are the same as those defined in Proposition 4.2, and* $\mathbf{K}_2 \in \mathbb{R}^{m\times m}$ *is the same as the one used in* (4.5)*.*

Proof of Proposition 4.3 can be found in Section 4.5.4.

4.2.2 Early Stopping SM Density Estimator with $f \in \widetilde{\mathcal{H}}$

We now discuss how to compute the early stopping SM density estimator with $f \in \widetilde{\mathcal{H}}$. We apply the gradient descent algorithm to minimizing $\widetilde{J}_{\mathrm{SM}}$ over $\mathbb{R}^m$. The corresponding gradient descent iterates are

$$\boldsymbol{\beta}_{\mathrm{SM}}^{(t+1)} = \boldsymbol{\beta}_{\mathrm{SM}}^{(t)} - \tau[\mathbf{M}\boldsymbol{\beta}_{\mathrm{SM}}^{(t)} - \mathbf{t}] = (\mathbf{I} - \tau\mathbf{M})\boldsymbol{\beta}_{\mathrm{SM}}^{(t)} + \tau\mathbf{t}, \tag{4.11}$$

where $\mathbf{M} := \frac{1}{n}\mathbf{S}\mathbf{S}^\top$ for notational simplicity, $\boldsymbol{\beta}_{\mathrm{SM}}^{(0)} \in \mathbb{R}^m$ is the starting point of the algorithm, and $\tau > 0$ is the appropriately chosen constant step size.

The following proposition links $\boldsymbol{\beta}_{\mathrm{SM}}^{(t)}$ to $\boldsymbol{\beta}_{\mathrm{SM}}^{(0)}$.

Proposition 4.4. *For all* $t \in \mathbb{N}_0$*, we have*

$$\boldsymbol{\beta}_{\mathrm{SM}}^{(t+1)} = (\mathbf{I} - \tau\mathbf{M})^{t+1}\boldsymbol{\beta}_{\mathrm{SM}}^{(0)} + \tau\sum_{\ell=0}^{t}(\mathbf{I} - \tau\mathbf{M})^{\ell}\mathbf{t}. \tag{4.12}$$

If, in particular, we choose $\boldsymbol{\beta}_{\mathrm{SM}}^{(0)} = \mathbf{0}_m$*, the* m*-dimensional zero vector, we have*

$$\boldsymbol{\beta}_{\mathrm{SM}}^{(t+1)} = \tau\mathbf{Q}\widetilde{\boldsymbol{\Lambda}}^{(t+1)}\mathbf{Q}^\top\mathbf{t}, \qquad \textit{for all } t \in \mathbb{N}_0, \tag{4.13}$$

where $\mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^\top = \mathbf{M}$ *be the eigen-decomposition of* $\mathbf{M}$*,* $\mathbf{Q}$ *is an orthogonal matrix,* $\boldsymbol{\Lambda}$ *is a diagonal matrix with all eigenvalues of* $\mathbf{M}$*,* $\lambda_1 \geq \lambda_2 \cdots \geq \lambda_m$*, on the diagonal, and* $\widetilde{\boldsymbol{\Lambda}}^{(t+1)}$ *is the diagonal matrix with elements on the diagonal being*

$$\tilde{\lambda}_j^{(t+1)} = \begin{cases} \dfrac{1-(1-\tau\lambda_j)^{t+1}}{\tau\lambda_j}, & \textit{if } \lambda_j \neq 0, \\ t+1, & \textit{if } \lambda_j = 0, \end{cases}$$

for all $j = 1, \cdots, m$ *and* $t \in \mathbb{N}_0$*.*

Proof of Proposition 4.4 can be found in Section 4.5.5.

Using Proposition 4.4, the $(t+1)$-st gradient descent iterate of $\widehat{J}_{\mathrm{SM}}$ in $\widetilde{\mathcal{H}}$ is

$$\sum_{j=1}^{m} \beta_j^{(t+1)} k(w_j, \cdot),$$

where $\beta_j^{(t+1)}$ is the j-th component of $\boldsymbol{\beta}_{\mathrm{SM}}^{(t+1)}$.

4.3 Comparison of Regularized ML and SM Density Estimators

With the algorithms of minimizing the penalized NLL loss functional and the (penalized) SM loss functional over a finite-dimensional approximating subspace of $\mathcal{H}$ shown in the preceding sections, we now use numerical examples to compare the corresponding density estimators and understand their similarities and differences.

We again use the `waiting` variable in the Old Faithful Geyser dataset. Figure 4.2 shows the resulting density estimates with different choices of penalty parameters or numbers of iterations. All these density estimates are computed over the finite-dimensional subspace (4.10). We use the same $\mathcal{X}$, μ, and k as in Chapter 3.

From the second and third rows of Figure 4.2, we see that, similar to our findings in Chapter 3, two kinds of regularized SM density estimates are still qualitatively very similar.

The penalized ML and regularized SM density estimates are qualitatively very similar when there is a large or intermediate amount of regularization. However, they are very different when there is very small regularization (small penalty parameter values or a large number of iterations). Specifically, when a very small regularization is imposed, regularized SM density estimates contains a bump or becomes a spike at the isolated observation, but penalized ML density estimates do not, even when there is no penalty (the case when $\lambda = 0$).

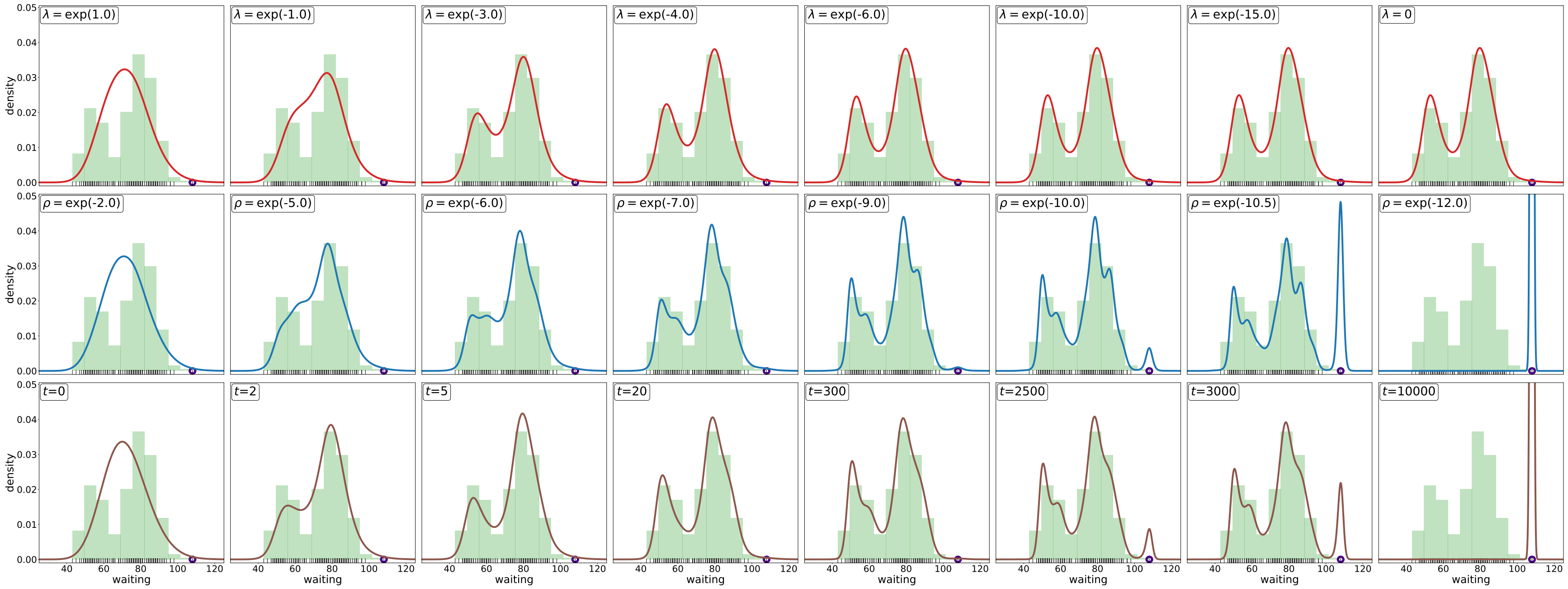

Figure 4.2: Penalized ML (first row), penalized SM (second row), and early stopping SM (third row) density estimates of the `waiting` variable. Histogram of data with the bin width chosen by the Freedman-Diaconis rule is shown in green. The rug plot indicates the location of data and the purple circle indicates the location of the isolated observation 108.

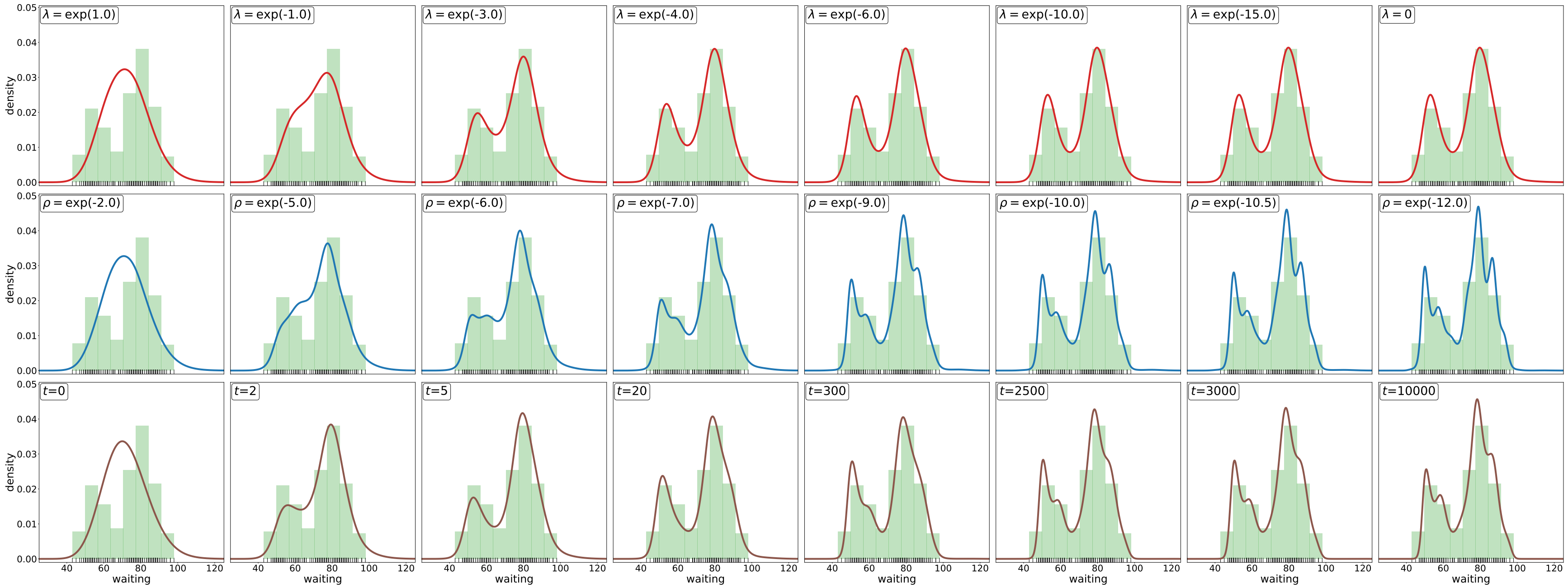

Figure 4.3: Penalized ML (first row), penalized SM (second row), and early stopping SM (third row) density estimates of the `waiting` variable with the isolated observation 108 removed. Histogram of data with the bin width chosen by the Freedman-Diaconis rule is shown in green. The rug plot indicates the location of data.

If we remove this isolated observation 108, as is shown in Figure 4.3, regularized SM density estimates do not contain a spike when the regularization is small.

Based on these numerical examples, we conclude that the regularized SM density estimators are very sensitive to the presence of an isolated observation, especially when there is a small amount of regularization. Additional numerical examples confirm this.

4.4 Discussion on the Presence of a Spike in SM Density Estimates

We now attempt to explain why the regularized SM density estimates put a spike at the isolated observation when regularization is very small. With $d = 1$, we can write the SM loss functional as

$$\widehat{L}_{\text{SM}}(q_f) = \underbrace{\frac{1}{n}\sum_{i \neq i^*}\left(\frac{1}{2}\big((\log q_f)'(X_i)\big)^2 + (\log q_f)''(X_i)\right)}_{=:\text{(I)}} + \underbrace{\frac{1}{n}\left(\frac{1}{2}\big((\log q_f)'(X_{i^*})\big)^2 + (\log q_f)''(X_{i^*})\right)}_{=:\text{(II)}},$$

where X_{i^*} denotes the isolated observation.

Recall that the penalized SM density estimator is obtained by minimizing $\widehat{L}_{\text{SM}}(q_f) + \frac{\rho}{2}\|f\|_{\mathcal{H}}^2$. When ρ is tiny, we are effectively minimizing the $\widehat{L}_{\text{SM}}$ part. Notice $\log q_f$ is a linear combination of $\log \mu$ and Gaussian kernel functions centered at a set of grid points or the first two derivatives of Gaussian kernels centered at data points (depending on which basis functions we use), and these Gaussian kernel functions and derivatives of Gaussian kernel functions are local basis functions. Also, note that a spike is essentially a local maximum. Consequently, putting a spike in $\log q_f$ at X_{i^*} has the effects of forcing $((\log q_f)'(X_{i^*}))^2 \approx 0$ and reducing the value of $(\log q_f)''(X_{i^*})$

and, hence, that of (II) a lot, without affecting (I) much. Thus, putting a spike at X_{i^*} can reduce the value of $\widehat{L}_{\mathrm{SM}}$, coinciding with the goal of minimizing $\widehat{L}_{\mathrm{SM}}$.

A similar explanation holds for the early stopping SM density estimator. As we keep running the gradient descent algorithm, we are searching for a $q_f \in \mathcal{Q}_{\mathrm{ker}}$ that can reduce the value of $\widehat{L}_{\mathrm{SM}}$ as much as possible. With an isolated observation X_{i^*} being present and local basis functions being used, putting a spike at X_{i^*} can achieve this goal.

4.5 Proofs

4.5.1 Proof of Proposition 4.1

Proof of Proposition 4.1. First, by **(A2)** in Chapter 2, we know $\mathcal{F} = \mathcal{H}$ so that minimizing (4.1) over $\mathcal{H}$ is an unconstrained minimization problem.

By **(A3)** and Theorem 2.2 in Chapter 2, we know A is strictly convex. In addition, since $\frac{1}{n}\sum_{i=1}^{n} f(X_i) = \frac{1}{n}\sum_{i=1}^{n}\langle f, k(X_i, \cdot)\rangle$ is convex in f, we conclude $\widehat{J}_{\mathrm{NLL}}$ is convex. In addition, note that the functional $f \mapsto \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2$ is strongly convex with constant $\lambda > 0$. It follows that the objective functional in (4.1) is strongly convex with constant $\lambda > 0$ (Proposition 10.8 in Bauschke and Combettes, 2011). Finally, Corollary 11.17 in Bauschke and Combettes (2011) implies (4.1) has exactly one minimizer in $\mathcal{H}$. ■

4.5.2 Details about Example 4.1

We provide details about Example 4.1 here, and will follow three steps:

Step 1. Identify the RKHS $\mathcal{H}$ associated with the choice of k and the corresponding natural parameter space $\mathcal{F}$;

Step 2. Identify $\mathrm{Span}\{k(X_1, \cdot)\}$ and $\mathrm{Span}\{k(X_1, \cdot)\} \cap \mathcal{F}$;

Step 3. Explcitly compute the minimizer of (4.2) over $\mathcal{F}$ and show this minimizer does not reside in $\mathrm{Span}\{k(X_1, \cdot)\} \cap \mathcal{F}$.

Step 1. Letting $x = (x_1, x_2)^\top \in \mathbb{R}^2$ and $y = (y_1, y_2)^\top \in \mathbb{R}^2$ be arbitrary, we have

$$k(x, y) = (x^\top y)^2 = x_1^2 y_1^2 + 2x_1x_2y_1y_2 + x_2^2y_2^2 = \langle \Phi(x), \Phi(y)\rangle, \qquad \text{for all } x, y \in \mathbb{R},$$

where $\Phi : \mathbb{R}^2 \to \mathbb{R}^3$ denotes the feature map of k and is given by

$$\Phi(x) = \left(x_1^2, \sqrt{2}x_1x_2, x_2^2\right)^\top,$$

and the inner product $\langle \cdot , \cdot \rangle$ is the dot product in $\mathbb{R}^3$. Then, the resulting $\mathcal{H}$ contains all functions mapping from $\mathbb{R}^2$ to $\mathbb{R}$ of the form

$$f(x) = w_1 x_1^2 + \sqrt{2} w_2 x_1 x_2 + w_3 x_2^2, \qquad \text{for all } x = (x_1, x_2)^\top \in \mathbb{R}^2,$$

for some $w_1, w_2, w_3 \in \mathbb{R}$ with the RKHS norm of f being $\|f\|_{\mathcal{H}} = \sqrt{w_1^2 + w_2^2 + w_3^2}$ (Theorem 4.21 in Steinwart and Christmann, 2008).

We now derive the natural parameter space $\mathcal{F}$. Let $f(x) = w_1 x_1^2 + \sqrt{2} w_2 x_1 x_2 + w_3 x_2^2$ for some $w_1, w_2, w_3 \in \mathbb{R}$ and all $x = (x_1, x_2)^\top \in \mathbb{R}^2$. By simple algebra, we have

$$e^{A(f)} = \int\int_{\mathbb{R}^2} \frac{1}{2\pi} \exp\Big(-\frac{\|x\|_2^2}{2}\Big) \exp\big(f(x)\big) \mathrm{d}x_1 \mathrm{d}x_2 = |W|^{1/2},$$

where $|W|$ denotes the determinant of the matrix W,

$$W^{-1} = \begin{pmatrix} 1 - 2w_1 & -\sqrt{2} w_2 \\ -\sqrt{2} w_2 & 1 - 2w_3 \end{pmatrix},$$

and we assume $(1 - 2w_1)(1 - 2w_3) - 2w_2^2 > 0$. Then,

$$A(f) = \log\big(|W|^{1/2}\big) = -\frac{1}{2} \log\big((1 - 2w_1)(1 - 2w_3) - 2w_2^2\big).$$

Note that $A(f) < \infty$ if and only if W is positive definite if and only if $1 - 2w_1 > 0$ and $1 - 2w_3 > 0$ and $(1 - 2w_1)(1 - 2w_3) - 2w_2^2 > 0$ if and only if $w_1 < \frac{1}{2}$ and $w_3 < \frac{1}{2}$ and $(1 - 2w_1)(1 - 2w_3) - 2w_2^2 > 0$.

As the conclusion of Step 1, we have

$$\begin{aligned} \mathcal{H} = \Big\{ f : \mathbb{R}^2 \to \mathbb{R} \,\Big|\, & f(x) := w_1 x_1^2 + \sqrt{2} w_2 x_1 x_2 + w_3 x_2^2 \text{ for all } x = (x_1, x_2)^\top \in \mathbb{R}^2, \\ & w_1, w_2, w_3 \in \mathbb{R} \Big\}, \end{aligned} \tag{4.14}$$

$$\begin{aligned} \mathcal{F} = \Big\{ f : \mathbb{R}^2 \to \mathbb{R} \,\Big|\, & f(x) = w_1 x_1^2 + \sqrt{2} w_2 x_1 x_2 + w_3 x_2^2 \text{ for all } x = (x_1, x_2)^\top \in \mathbb{R}^2, \\ & w_1 < \frac{1}{2}, w_3 < \frac{1}{2}, (1 - 2w_1)(1 - 2w_3) - 2w_2^2 > 0, w_1, w_2, w_3 \in \mathbb{R} \Big\}. \end{aligned} \tag{4.15}$$

Step 2. With $X_1 = (a, 0)^\top$, it is easy to see $\mathrm{Span}\{k(X_1, \cdot)\}$ contains all functions of the form

$$f(x) = \gamma k(X_1, x) = \gamma (X_1^\top x)^2 = \gamma a^2 x_1^2, \qquad \text{for all } x := (x_1, x_2)^\top \in \mathbb{R}^2 \text{ and } \gamma \in \mathbb{R}.$$

Then, we can characterize $\mathrm{Span}\{k(X_1, \cdot)\} \cap \mathcal{F}$ as

$$\mathrm{Span}\{k(X_1, \cdot)\} \cap \mathcal{F} = \left\{ f : \mathbb{R}^2 \to \mathbb{R} \;\middle|\; f(x) = \gamma a^2 x_1^2 \text{ for all } x = (x_1, x_2)^\top \in \mathbb{R}^2, \gamma a^2 < \frac{1}{2} \right\}.$$

Step 3. Let $f \in \mathcal{F}$ be

$$f(x) = w_1 x_1^2 + \sqrt{2} w_2 x_1 x_2 + w_3 x_2^2, \qquad \text{for all } x = (x_1, x_2)^\top \in \mathbb{R}^2,$$

where $w_1, w_2, w_3 \in \mathbb{R}$ satisfy the constraints in (4.15). Then, with $\lambda > 0$, $\widehat{J}_{\mathrm{NLL}}(f) + \frac{\lambda}{2} \|f\|_{\mathcal{H}}^2$ becomes

$$\widetilde{J}_\lambda(w_1, w_2, w_3) := -\frac{1}{2} \log\big((1 - 2w_1)(1 - 2w_3) - 2w_2^2\big) - w_1 a^2 + \frac{\lambda}{2}(w_1^2 + w_2^2 + w_3^2).$$

By calculus, we have

$$\nabla \widetilde{J}_\lambda(w_1, w_2, w_3) = \begin{pmatrix} \frac{1-2w_3}{(1-2w_1)(1-2w_3)-2w_2^2} - a^2 + \lambda w_1 \\ \frac{2w_2}{(1-2w_1)(1-2w_3)-2w_2^2} + \lambda w_2 \\ \frac{1-2w_1}{(1-2w_1)(1-2w_3)-2w_2^2} + \lambda w_3 \end{pmatrix}.$$

Any stationary points, denoted by (w_1^*, w_2^*, w_3^*), must satisfy $\nabla \widetilde{J}_\lambda(w_1^*, w_2^*, w_3^*) = 0$, from which we obtain the following system of equations

$$\begin{cases} (1 - 2w_3^*) + (\lambda w_1^* - a^2)\big((1 - 2w_1^*)(1 - 2w_3^*) - 2{w_2^*}^2\big) & = 0, \\ 2w_2^* + \lambda w_2^*\big((1 - 2w_1^*)(1 - 2w_3^*) - 2{w_2^*}^2\big) & = 0, \\ (1 - 2w_1^*) + \lambda w_3^*\big((1 - 2w_1^*)(1 - 2w_3^*) - 2{w_2^*}^2\big) & = 0. \end{cases}$$

Solving the preceding system for (w_1^*, w_2^*, w_3^*) yields the following solutions:

$$\left(\frac{1}{2}, 0, \frac{1}{2}\right), \quad \big(b_1^*, 0, c_1^*\big), \quad \big(b_1^*, 0, c_2^*\big), \quad \big(b_2^*, 0, c_1^*\big), \quad \big(b_2^*, 0, c_2^*\big), \tag{4.16}$$

where

$$b_1^* := \frac{(\lambda + 2a^2) + \sqrt{(\lambda - 2a^2)^2 + 8\lambda}}{4\lambda}, \qquad b_2^* := \frac{-(\lambda + 2a^2) + \sqrt{(\lambda - 2a^2)^2 + 8\lambda}}{-4\lambda},$$

$$c_1^* := \frac{\lambda + \sqrt{\lambda^2 + 8\lambda}}{4\lambda}, \qquad c_2^* := \frac{\lambda - \sqrt{\lambda^2 + 8\lambda}}{4\lambda}.$$

We next check whether the corresponding functions belong to $\mathcal{F}$ or not.

First, notice that the pair of $(w_1^*, w_2^*, w_3^*) = (\frac{1}{2}, 0, \frac{1}{2})$ leads to the function $f(x) = \frac{1}{2}(x_1^2 + x_2^2)$, which does *not* belong to $\mathcal{F}$ since $w_1^* = \frac{1}{2}$ and $w_3^* = \frac{1}{2}$. We discard it.

In addition, since $\lambda > 0$, we have

$$b_1^* = \frac{(\lambda + 2a^2) + \sqrt{(\lambda - 2a^2)^2 + 8\lambda}}{4\lambda} > \frac{(\lambda + 2a^2) + |\lambda - 2a^2|}{4\lambda}.$$

Consider the following two cases:

(i) If $2a^2 - \lambda \geq 0$, that is, $a^2/\lambda \geq \frac{1}{2}$, we have $b_1^* > \frac{\lambda+2a^2+2a^2-\lambda}{4\lambda} = \frac{a^2}{\lambda} \geq \frac{1}{2}$;

(ii) If $2a^2 - \lambda < 0$, we have $b_1^* > \frac{\lambda+2a^2+\lambda-2a^2}{4\lambda} = \frac{2\lambda}{4\lambda} \geq \frac{1}{2}$.

In both cases, we have $b_1^* > \frac{1}{2}$, and any function f with $w_1^* = b_1^*$ cannot belong to $\mathcal{F}$.

Next, we show $b_2^* < \frac{1}{2}$, which is equivalent to showing $1 - 2b_2^* > 0$. Since

$$\begin{aligned} 1 - 2b_2^* &= 1 - \frac{-(\lambda + 2a^2) + \sqrt{(\lambda - 2a^2)^2 + 8\lambda}}{-2\lambda} \\ &= 1 + \frac{\sqrt{(\lambda - 2a^2)^2 + 8\lambda} - (\lambda + 2a^2)}{2\lambda} \\ &> 1 + \frac{|\lambda - 2a^2| - (\lambda + 2a^2)}{2\lambda}. \end{aligned}$$

Again, we consider the following two cases:

(i) If $2a^2 - \lambda \geq 0$, we have $1 - 2b_2^* > 1 + \frac{2a^2-\lambda-\lambda-2a^2}{2\lambda} = 1 + \frac{-2\lambda}{2\lambda} = 0$;

(ii) If $2a^2 - \lambda < 0$, that is, $0 \leq a^2/\lambda < \frac{1}{2}$, we have $1 - 2b_2^* > 1 + \frac{\lambda-2a^2-\lambda-2a^2}{2\lambda} = 1 - \frac{2a^2}{\lambda} > 0$.

As for c_1^* and c_2^*, since $\lambda > 0$, we have

$$c_1^* = \frac{\lambda + \sqrt{\lambda^2 + 8\lambda}}{4\lambda} > \frac{\lambda + \sqrt{\lambda^2}}{4\lambda} = \frac{1}{2},$$

$$c_2^* = \frac{\lambda - \sqrt{\lambda^2 + 8\lambda}}{4\lambda} < \frac{\lambda}{4\lambda} = \frac{1}{4} < \frac{1}{2}.$$

That is, any function f with $w_3^* = c_1^*$ cannot belong to $\mathcal{F}$.

As a conclusion, the only solution in (4.16) such that the resulting f belongs to $\mathcal{F}$ is $(w_1^*, w_2^*, w_3^*) = (b_2^*, 0, c_2^*)$.

In addition, it is easy to see that the Hessian matrix of $\widetilde{J}_\lambda$ at $(w_1^*, w_2^*, w_3^*) = (b_2^*, 0, c_2^*)$ is

$$\begin{pmatrix} \frac{2(1-c_2^*)^2}{(1-2b_2^*)^2(1-2c_2^*)^2} + \lambda & 0 & 0 \\ 0 & \frac{2(1-b_2^*)(1-c_2^*)}{(1-2b_2^*)^2(1-2c_2^*)^2} + \lambda & 0 \\ 0 & 0 & \frac{2(1-b_2^*)^2}{(1-2b_2^*)^2(1-2c_2^*)^2} + \lambda \end{pmatrix},$$

which is positive definite. We conclude that $\widetilde{J}_\lambda$ achieves the minimum at $(w_1^*, w_2^*, w_3^*) = (b_2^*, 0, c_2^*)$, and the corresponding function, $f^* := \arg\min_{f \in \mathcal{F}} \{ \widehat{J}_{\mathrm{NLL}}(f) + \frac{\lambda}{2} \|f\|_{\mathcal{H}}^2 \}$, is

$$f^*(x) = b_2^* x_1^2 + c_2^* x_2^2, \qquad \text{for all } x = (x_1, x_2)^\top \in \mathbb{R}^2.$$

Since $c_2^* \neq 0$ for all $\lambda > 0$, f^* does *not* belong to $\mathrm{Span}\{k(X_1, \cdot)\} \cap \mathcal{F}$. ■

4.5.3 Proof of Proposition 4.2

Proof of Proposition 4.2. Since $f \in \widetilde{\mathcal{H}}$ takes on the form $f = \sum_{j=1}^m \beta_j k(w_j, \cdot)$ for $\beta_1, \cdots, \beta_m \in \mathbb{R}$, we first have

$$\begin{aligned} &\langle f, (\partial_u k(X_i, \cdot) \otimes \partial_u k(X_i, \cdot)) f \rangle_{\mathcal{H}} \\ &= \left\langle \sum_{\ell=1}^m \beta_\ell k(w_\ell, \cdot), (\partial_u k(X_i, \cdot) \otimes \partial_u k(X_i, \cdot)) \left(\sum_{j=1}^m \beta_j k(w_j, \cdot) \right) \right\rangle_{\mathcal{H}} \\ &= \sum_{\ell=1}^m \sum_{j=1}^m \beta_\ell \beta_j \langle \partial_u k(X_i, \cdot), k(w_\ell, \cdot) \rangle_{\mathcal{H}} \langle \partial_u k(X_i, \cdot), k(w_j, \cdot) \rangle_{\mathcal{H}} \\ &= \sum_{\ell=1}^m \sum_{j=1}^m \beta_\ell \beta_j \partial_u k(X_i, w_\ell) \partial_u k(X_i, w_j), \end{aligned}$$

where the last equality is due to the reproducing property of the partial derivative of k (see Proposition A.5 in Appendix A). Then, since $\widehat{C} = \frac{1}{n} \sum_{i=1}^n \sum_{u=1}^d \partial_u k(X_i, \cdot) \otimes$

$\partial_u k(X_i, \cdot)$, by the linearity of the inner product, we have

$$\begin{aligned}
\langle f, \widehat{C} f\rangle_{\mathcal{H}} &= \frac{1}{n}\sum_{i=1}^{n}\sum_{u=1}^{d}\big\langle f, (\partial_u k(X_i,\cdot)\otimes \partial_u k(X_i,\cdot)) f\big\rangle_{\mathcal{H}} \\
&= \frac{1}{n}\sum_{i=1}^{n}\sum_{u=1}^{d}\sum_{\ell=1}^{m}\sum_{j=1}^{m}\beta_\ell\beta_j\partial_u k(X_i, w_\ell)\partial_u k(X_i, w_j) \\
&= \boldsymbol{\beta}^\top\left(\frac{1}{n}\mathbf{S}\mathbf{S}^\top\right)\boldsymbol{\beta}.
\end{aligned}$$

As for $\langle f, \hat{z}\rangle_{\mathcal{H}}$, we have the following

$$\begin{aligned}
\langle f, \hat{z}\rangle_{\mathcal{H}} &= \left\langle \sum_{j=1}^{m}\beta_j k(w_j, \cdot), -\frac{1}{n}\sum_{i=1}^{n}\sum_{u=1}^{d}\big(\partial_u^2 k(X_i, \cdot) + (\partial_u \log\mu)(X_i)\partial_u k(X_i, \cdot)\big)\right\rangle_{\mathcal{H}} \\
&= \sum_{j=1}^{m}\beta_j\left(-\frac{1}{n}\sum_{i=1}^{n}\sum_{u=1}^{d}\big(\partial_u^2 k(X_i, w_j) + (\partial_u \log\mu)(X_i)\partial_u k(X_i, w_j)\big)\right) \\
&= \boldsymbol{\beta}^\top\mathbf{t},
\end{aligned}$$

where we use the reproducing property of the partial derivative of k in the second equality. The desired result follows by combining all pieces above together. ■

4.5.4 Proof of Proposition 4.3

Proof of Proposition 4.3. Recall from Chapters 2 and 3 that the operator $\widehat{C}: \mathcal{H}\to\mathcal{H}$ is positive semidefinite, $\widehat{J}_{\mathrm{SM}}$ is convex over $\mathcal{H}$. In addition, note that the functional $f \mapsto \frac{\rho}{2}\|f\|_{\mathcal{H}}^2$ is strongly convex with constant ρ. We conclude that the penalized SM loss functional $f \mapsto \widehat{J}_{\mathrm{SM}}(f) + \frac{\rho}{2}\|f\|_{\mathcal{H}}^2$ is strongly convex with parameter $\lambda > 0$, and that it has exactly one minimizer (Corollary 11.17 in Bauschke and Combettes, 2011).

With $f = \sum_{j=1}^{m}\beta_j k(w_j, \cdot) \in \widetilde{\mathcal{H}}$, we have $\frac{\rho}{2}\|f\|_{\mathcal{H}}^2 = \frac{\rho}{2}\boldsymbol{\beta}^\top\mathbf{K}_2\boldsymbol{\beta}$. Thus, together with Proposition 4.2, we can write the functional $\widehat{J}_{\mathrm{SM}}(f) + \frac{\rho}{2}\|f\|_{\mathcal{H}}^2$ with $f \in \widetilde{\mathcal{H}}$ as

$$\widetilde{J}_{\mathrm{SM},\rho}(\boldsymbol{\beta}) := \frac{1}{2}\boldsymbol{\beta}^\top\left(\frac{1}{n}\mathbf{S}\mathbf{S}^\top\right)\boldsymbol{\beta} - \boldsymbol{\beta}^\top\mathbf{t} + \frac{\rho}{2}\boldsymbol{\beta}^\top\mathbf{K}_2\boldsymbol{\beta}.$$

Then, $\boldsymbol{\beta}_{\mathrm{SM}}^{(\rho)} := \arg\min_{\boldsymbol{\beta}\in\mathbb{R}^m}\widetilde{J}_{\mathrm{SM},\rho}(\boldsymbol{\beta})$ must satisfy the first-order optimality condition

$$\left(\frac{1}{n}\mathbf{S}\mathbf{S}^\top + \rho\mathbf{K}_2\right)\boldsymbol{\beta}_{\mathrm{SM}}^{(\rho)} = \mathbf{t}.$$

As a consequence, $\arg\min_{f\in\widetilde{\mathcal{H}}}\{\widehat{J}_{\mathrm{SM}}(f)+\frac{\rho}{2}\|f\|_{\mathcal{H}}^2\}$ is $\sum_{j=1}^m \beta_j^{(\rho)}k(w_j,\,\cdot)$, where $\beta_j^{(\rho)}$ is the j-th element of $\boldsymbol{\beta}_{\mathrm{SM}}^{(\rho)}$. ■

4.5.5 Proof of Proposition 4.4

Proof of Proposition 4.4. We first prove (4.12) by induction. When $t = 0$, from (4.11), we have

$$\boldsymbol{\beta}_{\mathrm{SM}}^{(1)} = (\mathbf{I}-\tau\mathbf{M})\boldsymbol{\beta}_{\mathrm{SM}}^{(0)} + \tau\mathbf{t}. \tag{4.17}$$

From (4.12), we plug in $t = 0$ and obtain

$$\boldsymbol{\beta}_{\mathrm{SM}}^{(1)} = (\mathbf{I}-\tau\mathbf{M})\boldsymbol{\beta}_{\mathrm{SM}}^{(0)} + \tau\sum_{\ell=0}^{0}(\mathbf{I}-\tau\mathbf{M})^{i}\mathbf{t} = (\mathbf{I}-\tau\mathbf{M})\boldsymbol{\beta}_{\mathrm{SM}}^{(0)} + \tau\mathbf{t},$$

which is identical to (4.17).

Now, supposing (4.12) holds for $t = s$, we are going to show it also holds for $t = s+1$. Note the following

$$\begin{aligned}
\boldsymbol{\beta}_{\mathrm{SM}}^{(s+1)} &= (\mathbf{I}-\tau\mathbf{M})\boldsymbol{\beta}_{\mathrm{SM}}^{(s)} + \tau\mathbf{t} \\
&= (\mathbf{I}-\tau\mathbf{M})\left[(\mathbf{I}-\tau\mathbf{M})^{s}\boldsymbol{\beta}_{\mathrm{SM}}^{(0)} + \tau\sum_{\ell=0}^{s-1}(\mathbf{I}-\tau\mathbf{M})^{\ell}\mathbf{t}\right] + \tau\mathbf{t} \\
&= (\mathbf{I}-\tau\mathbf{M})^{s+1}\boldsymbol{\beta}_{\mathrm{SM}}^{(0)} + \tau\sum_{\ell=0}^{s-1}(\mathbf{I}-\tau\mathbf{M})^{\ell+1}\mathbf{t} + \tau\mathbf{t} \\
&= (\mathbf{I}-\tau\mathbf{M})^{s+1}\boldsymbol{\beta}_{\mathrm{SM}}^{(0)} + \tau\sum_{\ell=1}^{s}(\mathbf{I}-\tau\mathbf{M})^{\ell}\mathbf{t} + \tau\mathbf{t} \\
&= (\mathbf{I}-\tau\mathbf{M})^{s+1}\boldsymbol{\beta}_{\mathrm{SM}}^{(0)} + \tau\sum_{\ell=0}^{s}(\mathbf{I}-\tau\mathbf{M})^{\ell}\mathbf{t},
\end{aligned}$$

which is the desired result.

We now prove (4.13). With $\boldsymbol{\beta}_{\mathrm{SM}}^{(0)} = \mathbf{0}_m$, we can simplify (4.12) to

$$\boldsymbol{\beta}_{\mathrm{SM}}^{(t+1)} = \tau\sum_{\ell=0}^{t}(\mathbf{I}-\tau\mathbf{M})^{\ell}\mathbf{t}, \qquad \text{for all } t\in\mathbb{N}_0.$$

Let $\mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^\top = \mathbf{M}$ be the eigen-decomposition of $\mathbf{M}$, where $\mathbf{Q}$ and $\boldsymbol{\Lambda}$ are defined in the proposition. Then,

$$\mathbf{I} - \tau\mathbf{M} = \mathbf{I} - \tau\mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^\top = \mathbf{Q}\mathbf{Q}^\top - \tau\mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^\top = \mathbf{Q}(\mathbf{I} - \tau\boldsymbol{\Lambda})\mathbf{Q}^\top,$$

and we have

$$\sum_{\ell=0}^{t}(\mathbf{I} - \tau\mathbf{M})^\ell = \sum_{\ell=0}^{t}\mathbf{Q}(\mathbf{I} - \tau\boldsymbol{\Lambda})^\ell\mathbf{Q}^\top = \mathbf{Q}\left[\sum_{\ell=0}^{t}(\mathbf{I} - \tau\boldsymbol{\Lambda})^\ell\right]\mathbf{Q}^\top.$$

Now, if $\lambda_j \neq 0$, we have

$$\sum_{\ell=0}^{t}(1 - \tau\lambda_j)^\ell = \frac{1 - (1 - \tau\lambda_j)^{t+1}}{\tau\lambda_j}; \tag{4.18}$$

and if $\lambda_j = 0$, we have

$$\sum_{\ell=0}^{t}(1 - \tau\lambda_j)^\ell = t + 1. \tag{4.19}$$

Therefore, if we let $\widetilde{\boldsymbol{\Lambda}}^{(t+1)}$ be the diagonal matrix with elements on the diagonal being (4.18) and (4.19), the desired result follows. ■