Mathematical Modelling and Computing
DTU Compute

Bayesian Optimization for Hyperparameter Optimization in Linear Gaussian Models

Master Thesis

Bayesian Optimization for Hyperparameter Optimization in Linear Gaussian Models

Master Thesis
June, 2025

By
Frida Alamir Rønne

Supervised by
Martin Skovgaard Andersen, DTU Compute

Cover photo: Vibeke Hempler, 2012
Published by: DTU, DTU Compute, Building 303B, 2800 Kgs. Lyngby Denmark
www.compute.dtu.dk

Approval

Frida Alamir Rønne - s183978

Frida Alamir Rønne

Signature

23/6-2025

Date

Abstract

This report investigates the input estimation of linear Gaussian models using the properties of Gaussian processes. In particular, it demonstrates the advantages of using the Stable Spline (SS) kernel for system identification. In addition to this hyperparameter selection is performed by optimizing cost-functions corresponding to the Profile Marginalized Likelihood (PML) and Generalized Cross-Validation (GCV) criteria, where Bayesian optimization is utilized to efficiently locate their optimum point. Furthermore, an analytical approach based on Singular Value Decomposition (SVD) is introduced to reduce a variable from the cost-functions, significantly improving the computational efficiency of evaluating the cost-function and ultimately locating the optimum point. Additionally, the report explores how combining the PML and GCV criteria can lead to improved fit-scores. However, experimental results indicate no clear pattern for how the combination should be weighted.

Contents

1 Introduction

This report investigates how hyperparameter selection methods can be applied to input estimation in linear Gaussian models. If we consider a linear Gaussian model as a time-invariant single-input single-output system:

$$\boldsymbol{y} = \Phi\boldsymbol{\theta} + \boldsymbol{\varepsilon}, \tag{1.0.1}$$

where $\boldsymbol{y} \in \mathbb{R}^m$ is a known output, $\Phi \in \mathbb{R}^{m\times n}$ is a known input, $\boldsymbol{\theta} \in \mathbb{R}^n$ and $\boldsymbol{\varepsilon} \in \mathbb{R}^m$ are unknown and are assumed to be Gaussian distributed. The objective is to predict the unknown function-vector $\boldsymbol{\theta} \in \mathbb{R}^n$, referred to as the impulse response. With no prior assumption on $\boldsymbol{\theta}$ the impulse response can be estimated as a Maximum Likelihood (ML) problem by minimizing the least-squares (LS) problem :

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} ||\boldsymbol{y} - \Phi\boldsymbol{\theta}||_2^2. \tag{1.0.2}$$

Additionally, a common prior assumption for $\boldsymbol{\theta}$ is that it follows a zero-mean distribution with some covariance matrix $\sigma^2 K$ and similarly the noise $\boldsymbol{\varepsilon}$ also follows a zero-mean distribution with the covariance matrix $\sigma_n^2 I$. This leads us with two hyperparameters σ^2 and σ_n^2 to select when estimating $\boldsymbol{\theta}$. The impulse response is then estimated using Maximum a Posterior (MAP) estimation by minimizing the regularized least squares (RLS) problem, in the simplest case we assume $\sigma_n^2 = 1$ and $K = I$:

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} ||\boldsymbol{y} - \Phi\boldsymbol{\theta}||_2^2 + \frac{1}{2\sigma^2}||\boldsymbol{\theta}||_2^2, \tag{1.0.3}$$

However, in this project, the Stable Spline (SS) kernel [1] is chosen for constructing the covariance matrix K_ℓ, as it is well-known that it is advantageous in system identification. Note that using the SS kernel introduces a third hyperparameter ℓ. The MAP estimate then becomes:

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \frac{1}{2\sigma_n^2}||\boldsymbol{y} - \Phi\boldsymbol{\theta}||_2^2 + \frac{1}{2\sigma^2}\boldsymbol{\theta}^\top K_\ell^{-1}\boldsymbol{\theta}. \tag{1.0.4}$$

This leaves us with three hyperparameters to optimize over (σ_n^2, σ^2, ℓ). In this report, the hyperparameters are selected by optimizing the cost-functions corresponding to the Marginalized Likelihood and the Generalized Cross-Validation criteria. For this Bayesian optimization is used as an efficient method for optimizing these cost-functions. This method exploits the properties of Gaussian processes to infer the unknown function values – a topic that will be explored later in this report.

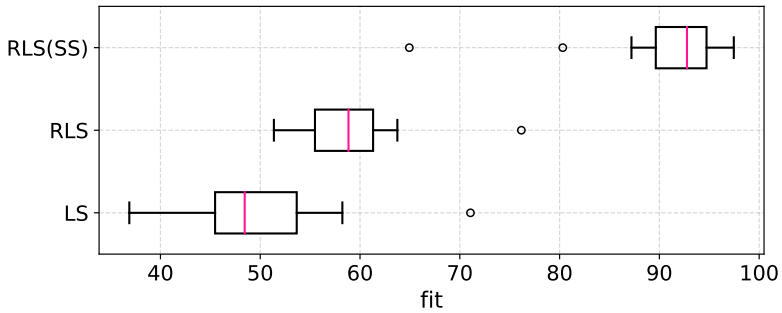


Figure 1.1: Distribution of fits for 20 systems using LS, RLS with $K = I$ and RLS with the SS kernel.

In Figure 1.1 we see how adding the regularized term improves the fits, furthermore when using the SS kernel the improvements are significantly better. However, obtaining an estimate for the impulse response is reliant on the hyperparameter optimization via the two cost-function mentioned. In addition, we in this report will investigate how combining these two cost-functions could possibly improve the fit.

2 Gaussian Processes

A central concept used in this report is the Gaussian process (GP).
The Gaussian probability distribution describes the distribution of one stochastic variable (a scalar) or more than one stochastic variable (listed in a vector). Considering a vector consisting of stochastic variables describing all the functions values of an unknown function is known as a Gaussian process. However, this vector would be infinitely long if it were to hold all function values, making it impossible to compute. Surprisingly, if we restrict our problem to a large finite number of function values the Gaussian process will give the same answer as if we had considered all the infinitely many points. It is therefore very advantageous to use Gaussian processes to do inference on unknown functions [2]. Gaussian processes is used to estimate the impulse response of our systems as it can be modelled as a linear Gaussian model, furthermore it is the foundation for Bayesian optimization which will be uncovered later in the report.

2.1 Noisy Observations

The following section is based on the theory presented in [2].
We now consider an unknown target function $f(\boldsymbol{x})$, where the input vector $\boldsymbol{x}$ holds p input variables. We then consider the case where we have m training points, such that X holds m vectors of length p thus $X \in \mathbb{R}^{p \times m}$, where the function values at these points are known. Furthermore we have m_* test point $X_* \in \mathbb{R}^{p \times m_*}$, where the function values are unknown. We denote the function values of the test values as $f_* = f(\boldsymbol{x}_*)$. We assume that the function values we obtain will be noisy versions of the true function values. We therefore denote the function values of the training points: $y = f(\boldsymbol{x}) + \varepsilon$, here ε is independent identical distributed Gaussian noise with variance σ_n^2. The covariance for the y is therefore:

$$\text{cov}(y) = K(X, X) + \sigma_n^2 I, \tag{2.1.1}$$

$K(\ ,\ )$ is a kernel function that takes X as input and computes the covariance matrix. There are different choices for the covariance function and furthermore the covariance function will hold one or more hyperparameters – how the hyperparameters are selected will affects the function values.

The joint distribution of the function values $\boldsymbol{f}_*$ and training points $\boldsymbol{y}$ is:

$$\begin{bmatrix} \mathbf{f}_* \\ \mathbf{y} \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X_*, X_*) & K(X_*, X) \\ K(X, X_*) & K(X, X) + \sigma_n^2 I \end{bmatrix}\right) \tag{2.1.2}$$

If we denote the mean vector, $\boldsymbol{\mu}$ and the covariance matrix, $\boldsymbol{\Sigma}$:

$$\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_a \\ \boldsymbol{\mu}_b \end{bmatrix}, \qquad \boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_{aa} & \boldsymbol{\Sigma}_{ab} \\ \boldsymbol{\Sigma}_{ba} & \boldsymbol{\Sigma}_{bb} \end{bmatrix} \tag{2.1.3}$$

We are able to obtain an expression for the mean and covariance for the conditional distribution $p(a|b)$ [3]:

$$\boldsymbol{\mu}_{a|b} = \boldsymbol{\mu}_a + \boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_{bb}^{-1}(\mathbf{x}_b - \boldsymbol{\mu}_b) \tag{2.1.4}$$

$$\boldsymbol{\Sigma}_{a|b} = \boldsymbol{\Sigma}_{aa} - \boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_{bb}^{-1}\boldsymbol{\Sigma}_{ba}. \tag{2.1.5}$$

Thus, the conditional distribution of the test function values given the observed training functions values (with added noise), $p(\boldsymbol{f}_*|\boldsymbol{y})$, is:

$$\boldsymbol{\mu}_{f_*|y} = K\left(X_*, X\right)\left[K(X,X) + \sigma_n^2 I\right]^{-1}\mathbf{y} \tag{2.1.6}$$

$$\boldsymbol{\Sigma}_{f_*|y} = K\left(X_*, X_*\right) - K\left(X_*, X\right)\left[K(X,X) + \sigma_n^2 I\right]^{-1} K\left(X, X_*\right), \tag{2.1.7}$$

based on the distribution in (2.1.2)

2.1.1 Predicting

In order to obtain useful prediction the hyperparametersmust be selected accordingly. However, beforehand it is necessary to evaluate our prediction for the functions values $\mu_{f_*|y}$, i.e. the mean at our test points, given our training points. Here it becomes advantageous to evaluate the marginal log likelihood of $p(\boldsymbol{y}|X)$ (note that maximizing this yields one method for selecting the hyperparameters which we will deal with in section 5). Given the conditional distribution of $\boldsymbol{y}|X \sim \mathcal{N}\left(\mathbf{0}, K(X,X) + \sigma_n^2 I\right)$, the marginal log likelihood is

$$\log p(\boldsymbol{y}|X) = -\frac{1}{2}\boldsymbol{y}^\top\left[K(X,X) + \sigma_n^2 I\right]^{-1}\boldsymbol{y} - \frac{1}{2}\log\left|K(X,X) + \sigma_n^2 I\right| - \frac{m}{2}\log 2\pi. \tag{2.1.8}$$

Note that here $K(X,X)$ holds a number of hyperparameters, furthermore the noise variance σ_n^2 also acts as an hyperparameter if it not already given.
When evaluating (2.1.8) we are able to optimize some terms making it computational more efficient. Firstly, we can use Cholesky factorization on the matrix $K = K(X,X) + \sigma_n^2 I$ as it is a symmetric matrix. The Cholesky factorization computes a lower triangle matrix L, for which it applies that $K = LL^\top$, where $L^\top$ is an upper triangle matrix. When computing the determinant of $K = LL^\top$ the properties of the Cholesky factorization are utilized:

$$\det(K) = \det(\mathrm{L}\mathrm{L}^\top) = \det(\mathrm{L})\det(\mathrm{L}^\top), \tag{2.1.9}$$

recalling that L and $L^\top$ are lower and upper triangle matrices the determinant can simply be expressed as the product of the diagonal elements, thus:

$$\det(\mathrm{L})\det(\mathrm{L}^\top) = \prod_{i=1}^{m} L_{ii}\prod_{i=1}^{m} L_{ii} = \prod_{i=1}^{m} L_{ii}^2. \tag{2.1.10}$$

Finally, the term $-\frac{1}{2}\log\left|K(X,X) + \sigma_n^2 I\right|$ can be rewritten as:

$$-\frac{1}{2}\log\prod_{i=1}^{m} L_{ii}^2 = -\sum_{i=1}^{m}\log L_{ii} \tag{2.1.11}$$

Utilizing this factorization we are able to evaluate $\boldsymbol{\mu}_{f_*|y}$, $\boldsymbol{\Sigma}_{f_*|y}$ and $p(\boldsymbol{y}|X)$ for one test point $\boldsymbol{x}_*$ as follows:

Algorithm 1 (X, $\boldsymbol{y}$, $K(\ ,\ )$, σ_n^2, $\boldsymbol{x}_*$)

1: $K \leftarrow K(X, X)$
2: $\boldsymbol{k}_* \leftarrow K(X, \boldsymbol{x}_*)$
3: $L \leftarrow$ Cholesky$(K + \sigma_n^2 I)$ ▷ Note: $LL^\top = K + \sigma_n^2 I$ and $L^{-\top}L^{-1} = [K + \sigma_n^2 I]^{-1}$
4: Solve $L\boldsymbol{u} = \boldsymbol{y} \quad (\boldsymbol{u} \leftarrow L^{-1}\boldsymbol{y})$
5: Solve $L^\top \boldsymbol{\alpha} = \boldsymbol{u} \quad (\boldsymbol{\alpha} \leftarrow L^{-\top}\boldsymbol{u})$
6: Solve $L\boldsymbol{v} = \boldsymbol{k}_* \quad (\boldsymbol{v} \leftarrow L^{-1}\boldsymbol{k}_*)$
7: $\bar{f}_* \leftarrow \boldsymbol{k}_*^\top \boldsymbol{\alpha}$ ▷ Evaluating (2.1.6), here $\boldsymbol{k}_*^\top \boldsymbol{\alpha} = \boldsymbol{k}_*^\top L^{-\top} L^{-1} \boldsymbol{y}$
8: $V[f_*] \leftarrow k(\boldsymbol{x_*}, \boldsymbol{x_*}) - ||\boldsymbol{v}||_2^2$ ▷ Evaluating (2.1.7), here $||\boldsymbol{v}||_2^2 = \boldsymbol{k}_* L^{-1} L^{-\top} \boldsymbol{k}_*^\top$
9: $\log p(\boldsymbol{y}|X) \leftarrow -\frac{1}{2}||\boldsymbol{u}||_2^2 - \sum_{i=1}^{p} \log L_{ii} - \frac{m}{2} \log 2\pi$ ▷ Evaluating (2.1.8), here $||\boldsymbol{u}||_2^2 \leftarrow \boldsymbol{y}^\top L^{-\top} L^{-1} \boldsymbol{y}$

Cholesky factorization allows more efficient evaluation of (2.1.6)–(2.1.8), but it introduces a bottleneck (line 3) due to its $\mathcal{O}(p^3)$ complexity, while the remaining steps (lines 4–6) are at most $\mathcal{O}(p^2)$. We hereby predict f by evaluating the mean $\bar{f}_*$ as this is the most probable estimate for f.

3 Covariance Functions

The following section is based on the theory presented in [2] .
As previously mentioned there are a number of different choices for the covariance function, $K(\ ,\ )$. In this section we will consider four different covariance functions, the Squared Exponential covariance function and three variations of the Matérn covariance function. Firstly, the Squared Exponential (SE) covariance function simply computes the covariance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ as:

$$\text{cov}_{SE}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma^2 \exp\left(-\frac{d^2}{2}\right), \qquad d = dist(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{|\boldsymbol{x}_i - \boldsymbol{x}_j|}{\ell}, \tag{3.0.1}$$

with hyperparameters σ^2 and ℓ. [2]

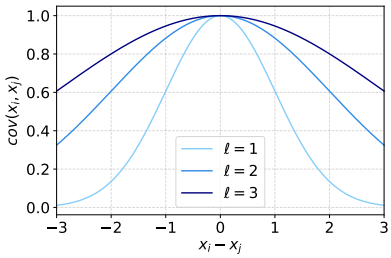


Figure 3.1: Squared Exponential function for different length scale ($\sigma^2 = 1$).

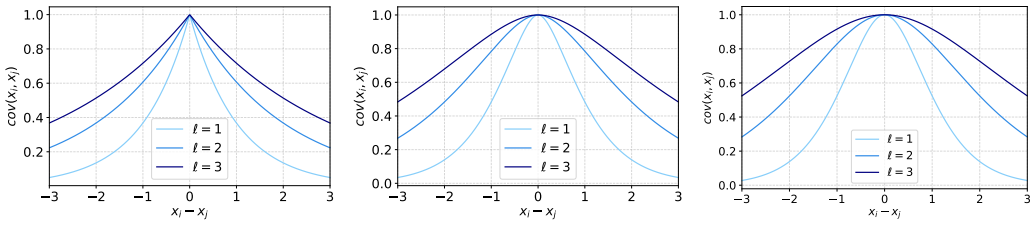
Here σ^2 is a scaling-factor and ℓ is a length-scale parameter that controls how we weight the distance in the covariance function. As seen in Figure 3.1 for $\ell \to \infty$ the covariance increases for $|x_i - x_j| \to \infty$.
For the Matérn covariance function there exist infinitely many extensions, however we in this sections consider the three simplest. The Matérn covariance function is defined by the parameters ν and p. The expression simplifies for $\nu = 1/2 + p$ for $p \in \mathbb{N}^+$, and we will consider the Matérn covariance for $\nu = 1/2,\ 3/2,\ 5/2$:

$$\text{cov}_{\nu=1/2}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma^2 \exp\left(-d\right), \tag{3.0.2}$$

$$\text{cov}_{\nu=3/2}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma^2 \left(1 + \sqrt{3}d\right) \exp\left(-\sqrt{3}d\right), \tag{3.0.3}$$

$$\text{cov}_{\nu=5/2}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma^2 \left(1 + \sqrt{3}d + \frac{5}{3}d^2\right) \exp\left(-\sqrt{5}d\right). \tag{3.0.4}$$

Note that d is the same distance function that is stated in (3.0.1).



(a) $\nu = 1/2$ (b) $\nu = 3/2$ (c) $\nu = 5/2$

Figure 3.2: Matérn Function for different length scale ($\sigma^2 = 1$).

When we consider how the ℓ affects the Matérn covaraince functions in Figure 3.2 we see a similar pattern as for the Squared Exponential function. We notice however that the Matérn Function for $\nu = 1/2$ is non-differentiable when $x_i - x_j = 0$.
We will now explore how the four different covariance functions behave when they are used to model the target function f, with 10 training points $(x, y = f(x))$ and no added noise.

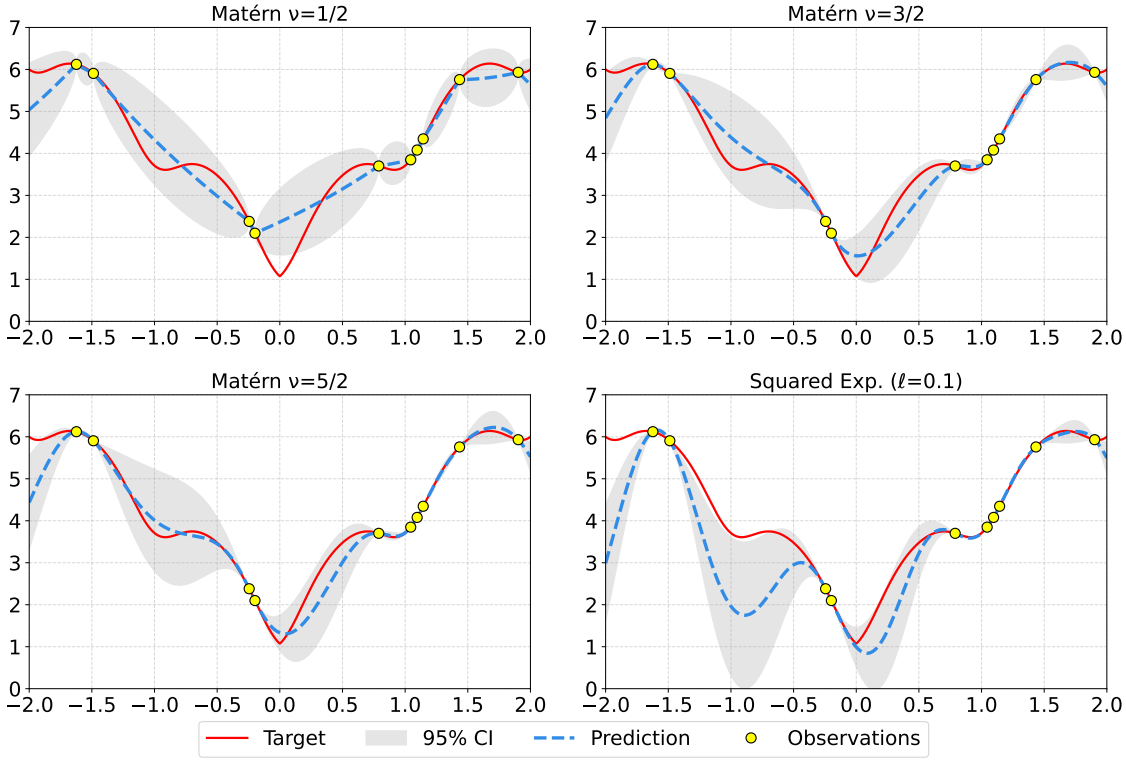


Figure 3.3: GP results using different covariance functions ($\sigma^2 = 1$).

When we consider the results seen in Figure 3.3, it seems that the Matérn covariance function for $\nu = 3/2$ and $\nu = 5/2$ are very similar, this is in coherence with how similar the covariance functions appear in Figure 3.2 (b) & (c). Furthermore, it is noticeable that when we use the Matérn covariance function for $\nu = 1/2$ the confidence intervals are relatively wider, this is also as expected if we consider Figure 3.2 (a) here we see that for small distances, in specific between -1 and 1, the covariance decreases faster yielding larger uncertainty. Lastly, it is worth mentioning that we use $\ell = 0.1$ when we use the Squared Exponential function, this is due to the fact that the distance function in (3.0.1) is squared and it is therefore not directly comparable to the Matérn covariance functions. When we consider the results from SE in Figure 3.3 we see that the covariance is more vulnerable to larger distances – between -1 and 0, where we have the largest distance we see a considerable large confidence interval compared to the rest.

4 Bayesian Optimization

Rather than modelling unknown functions, we now focus on finding the global optimum of these unknown functions. It is exactly the problem of finding the maximum or minimum of unknown functions that Bayesian optimization (BO) deals with. These unknown functions, where no prior assumptions are given, are known as ”black-box” functions and are often costly to evaluate. Bayesian optimization therefore proposes a sequential method for choosing the next points to evaluate in order to reach the optimum function value. Here the Gaussian Processes can be used as a tool to retrieve a probability distribution of the ”black-box” function, and thereby make an informed decision based on the confidence interval determined by the covariance function [4].

4.1 Acquisition Functions

When we need to decide where to evaluate the unknown function, f, we are faced with a trade-off between exploration and exploitation. Exploration: choosing a point for the purpose of minimize the uncertainty i.e. reducing the width of the confidence interval. When emphasizing exploration the function evaluation will most likely be more spread out and we hereby minimize the risk of getting caught in a local optimum. Exploitation: choosing a point for the purpose of possibly reaching the optimum point. Emphasizing exploitation will most likely result in the function evaluations being placed around a possible optimum. However, exploitation risks getting stuck in a local optimum. An example of emphasizing exploitation and exploration respectively is shown in Figure 4.1, for this instance it is seen that the optimum is reached in 6-7 function evaluations when exploitation is preferred, in contrast the optimum is not quite reached when exploration is preferred however the confidence interval is remarkably narrower.

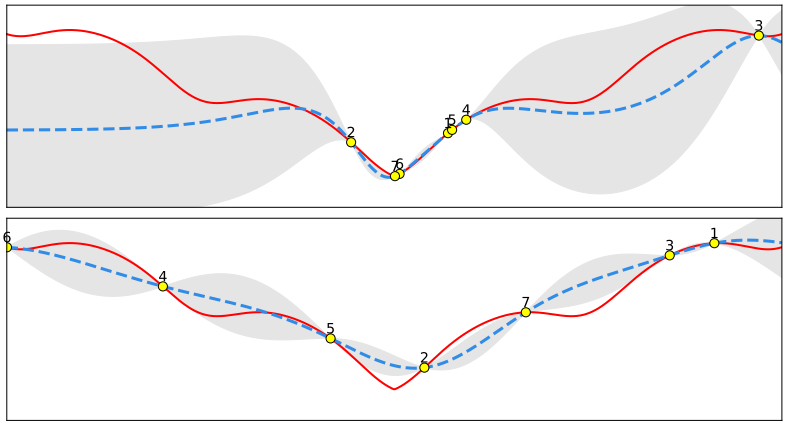


Figure 4.1: Top: Exploitation is preferred. Bottom: Exploration is preferred.

We will now consider three different Acquisition functions that can be used for choosing the next point to evaluate, by balancing the trade-off between exploitation and exploration.

4.1.1 Upper Confidence Bound (UCB)

The simplest approach is to choose the next point where the confidence interval has its upper bound.

$$\arg\max_{x} \left(\mathrm{UCB}\left(x\right)\right), \tag{4.1.1}$$

where

$$\text{UCB}\left(\boldsymbol{x}\right) = \boldsymbol{\mu}\left(\boldsymbol{x}\right) + \rho\boldsymbol{\sigma}\left(\boldsymbol{x}\right), \tag{4.1.2}$$

here ρ controls how the variance is weighted, i.e. for small ρ points where the expected value $\boldsymbol{\mu}(\boldsymbol{x})$ is large will be chosen, in contrast for large ρ points with large uncertainty $\boldsymbol{\sigma}(\boldsymbol{x})$ will be chosen.

4.1.2 Probability of Improvement (PI)

The purpose for the Probability of Improvement acquisition function is to chose point where the probability of improvement is largest. Therefore:

$$\arg\max_{\boldsymbol{x}} \left(\text{PI}\left(\boldsymbol{x}\right)\right), \tag{4.1.3}$$

where

$$\text{PI}\left(\boldsymbol{x}\right) = \Phi\left(\frac{\boldsymbol{\mu}\left(\boldsymbol{x}\right) - y_{+} - \xi}{\boldsymbol{\sigma}\left(\boldsymbol{x}\right)}\right), \tag{4.1.4}$$

here y_{+} is the largest observed function value thus far, ξ is a parameter controlling how the trade-off between exploration and exploitation. Large ξ weights exploration high, small ξ weights exploitation high.

4.1.3 Expected Improvement (EI)

Where the PI maximizes the probability of the improvement the Expected Improvement acquisition function focuses on the maximizing the magnitude of the expected improvement.

$$\text{EI}\left(\boldsymbol{x}_i\right) = \begin{cases} \left(\boldsymbol{\mu}\left(\boldsymbol{x}\right) - y_{+} - \xi\right)\Phi\left(\frac{\boldsymbol{\mu}(\boldsymbol{x}) - y_{+} - \xi}{\boldsymbol{\sigma}(\boldsymbol{x})}\right) + \boldsymbol{\sigma}\left(\boldsymbol{x}\right)\phi\left(\frac{\boldsymbol{\mu}(\boldsymbol{x}) - y_{+} - \xi}{\boldsymbol{\sigma}(\boldsymbol{x})}\right), & \text{if } \boldsymbol{\sigma}\left(\boldsymbol{x}\right) > 0 \\ \max(0, \boldsymbol{\mu}\left(\boldsymbol{x}\right) - y_{+} - \xi), & \text{if } \boldsymbol{\sigma}\left(\boldsymbol{x}\right) = 0 \end{cases} \tag{4.1.5}$$

The trade-off between expectation and exploitation is controlled by the parameter ξ. [4]

In this report we have chosen to exclusively use UCB as acquisition function when carrying out the numerical experiments.

5 Selecting the Hyperparameters

Two methods for selecting the hyperparameters will now be unfolded.

5.1 Profile Marginal Likelihood (PML)

If we consider (2.1.8), we can select the hyperparameters by minimizing the negative marginal log likelihood:

$$\arg\min_{\sigma_n^2,\sigma^2,\ell}(-\log p(\boldsymbol{y}|X)). \tag{5.1.1}$$

Using the notation $\sigma^2 K_\ell = K(X,X)$, moving the scaling-factor σ^2 outside the covariance function and noting that K holds the length-scale ℓ, we obtain the following,

$$-\log p(\boldsymbol{y}|X) = \frac{1}{2}\boldsymbol{y}^\top \left[\sigma^2 K_\ell + \sigma_n^2 I\right]^{-1}\boldsymbol{y} + \frac{1}{2}\log\left|\sigma^2 K_\ell + \sigma_n^2 I\right| + \frac{m}{2}\log 2\pi. \tag{5.1.2}$$

By performing a change of variables $\sigma_n^2/\sigma^2 = \lambda$, the expression

$$-\log p(\boldsymbol{y}|X) = \frac{1}{2}\boldsymbol{y}^\top \left[\left(\frac{\sigma^2}{\sigma^2}K_\ell + \frac{\sigma_n^2}{\sigma^2}I\right)\sigma^2\right]^{-1}\boldsymbol{y} + \frac{1}{2}\log\left|\left(\frac{\sigma^2}{\sigma^2}K_\ell + \frac{\sigma_n^2}{\sigma^2}I\right)\sigma^2\right| + \frac{m}{2}\log 2\pi, \tag{5.1.3}$$

becomes

$$-\log p(\boldsymbol{y}|X) = \frac{1}{2\sigma^2}\boldsymbol{y}^\top \left[K_\ell + \lambda I\right]^{-1}\boldsymbol{y} + \frac{1}{2}\log|K_\ell + \lambda I| + \frac{m}{2}\log 2\pi - \frac{m}{2}\log\sigma^{-2}. \tag{5.1.4}$$

The right-hand side is now a convex function of σ^{-2}, by setting the derivative of this function equal to zero we then obtain an expression for the optimal value of σ^2:

$$-\frac{m}{2}\sigma^2 + \frac{1}{2}\boldsymbol{y}^\top\left[K_\ell + \lambda I\right]^{-1}\boldsymbol{y} = 0 \Leftrightarrow \sigma^2 = \frac{\boldsymbol{y}^\top\left[K_\ell + \lambda I\right]^{-1}\boldsymbol{y}}{m}. \tag{5.1.5}$$

Substituting back σ^{-2} we obtain the Profile Marginal Likelihood objective function:

$$\mathrm{PML}(\lambda,\ell) = \frac{1}{2}\log|K_\ell + \lambda I| + \frac{m}{2}(\log 2\pi + 1) + \frac{m}{2}\log\frac{\boldsymbol{y}^\top\left[K_\ell + \lambda I\right]^{-1}\boldsymbol{y}}{m}. \tag{5.1.6}$$

By minimizing the PML criterion we are able select the hyperparameters.

5.2 Genereralized Cross-Validation (GCV)

Another way of selecting the hyperparameters is using Genereralized Cross-Validation (GCV), where the hyperparameters are selcted by minimizing the GCV criterion [5]:

$$\mathrm{GCV}(\lambda,\ell) = \frac{\frac{1}{m}||(I-H)\boldsymbol{y}||_2^2}{(\mathrm{Tr}(I-H)/m)^2}, \tag{5.2.1}$$

The matrix H is known as the hat matrix, which maps $\boldsymbol{y}$ to our estimate, $\hat{\boldsymbol{f}}$:

$$\hat{\boldsymbol{f}} = H\boldsymbol{y}. \tag{5.2.2}$$

When $\boldsymbol{y} \sim \mathcal{N}(\mathbf{0}, K_\ell + \sigma_n^2 I)$, there exist a vector $\boldsymbol{\alpha}$ for which

$$\boldsymbol{y} = (K + \sigma_n^2 I)\boldsymbol{\alpha} \Leftrightarrow \boldsymbol{\alpha} = (\sigma_n^2 I + K)^{-1}\boldsymbol{y}. \tag{5.2.3}$$

The estimate of $\boldsymbol{f}$ must be equal to the expected value given our test and training points as expressed in (2.1.6) and given $\boldsymbol{\alpha}$ the estimate of $\boldsymbol{f}$ for x_j is:

$$\hat{f}(x_j) = \sum_{i=1}^{m} \boldsymbol{\alpha}_i K(x_i, x_j). \tag{5.2.4}$$

Therefore the vector consisting of all estimates can be constructed as

$$\hat{\boldsymbol{f}} = \begin{bmatrix} \hat{f}(x_1) \\ \vdots \\ \hat{f}(x_m) \end{bmatrix} = K\boldsymbol{\alpha} = K(K + \sigma_n I)^{-1}\boldsymbol{y}, \tag{5.2.5}$$

thus,

$$H = K_\ell(K_\ell + \sigma_n^2 I)^{-1}. \tag{5.2.6}$$

Here H depends on the hyperparameters ℓ and σ_n^2.

6 State Space Model

We now consider a state space model (SSM), with no direct feed-through ($D = 0$), in discrete time:

$$\boldsymbol{x}_{t+1} = A\boldsymbol{x}_t + B\boldsymbol{u}_t \tag{6.0.1}$$

$$\boldsymbol{y}_t = C\boldsymbol{x}_t, \tag{6.0.2}$$

with input signal vector $\boldsymbol{u}$, state vector $\boldsymbol{x}$ and the output signal vector $\boldsymbol{y}$. The aim is now to model the impulse response, $\boldsymbol{\theta}_{\text{true}} \in \mathbb{R}^q$ of the SSM. In theory the impulse response is infinitely long, however in practice the length q is determined by the number of time steps it takes before it reaches a threshold that is sufficiently close to zero. When considering the 5 instances of the impulse response in Figure 6.1 we see that q varies a great deal and the impulse response appears very different for each instance. In this project we have chosen sample time of 0.3, therefore we reach $n = 200$ samples after 60 time steps.

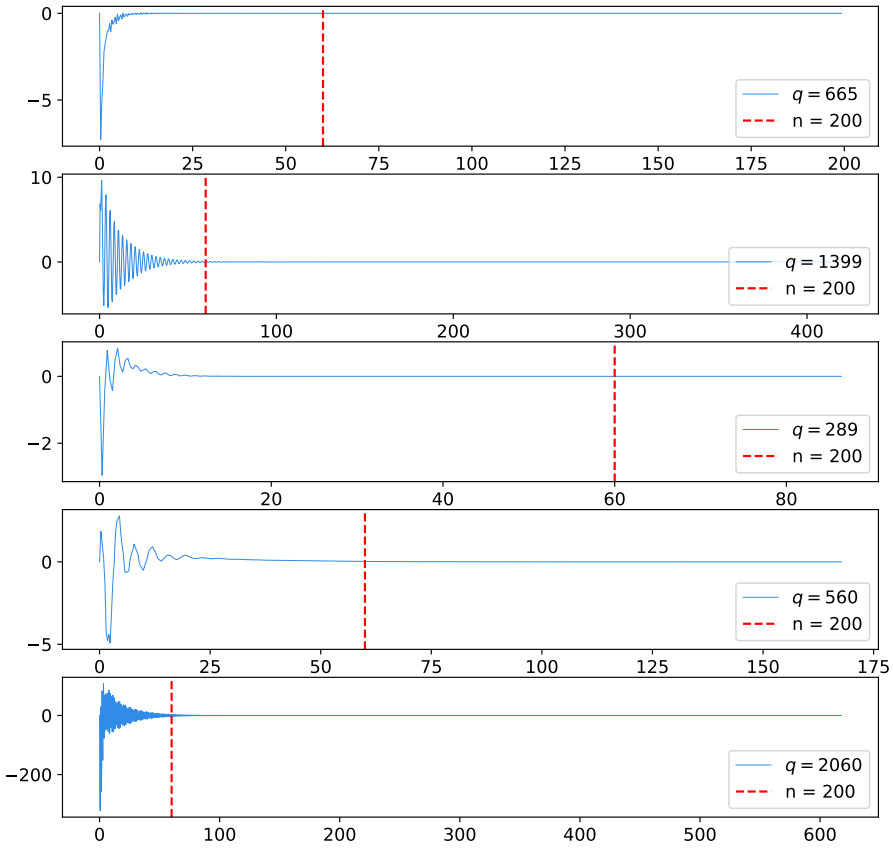


Figure 6.1: The impulse response $\boldsymbol{\theta}_{\text{true}}$ for 5 random instances of the SSM.

In order to estimate $\boldsymbol{\theta}$ we consider a model for the observed output signal vector $\boldsymbol{y}$ that can be described as (with added noise, ε):

$$\boldsymbol{y} = \Phi\boldsymbol{\theta} + \varepsilon, \tag{6.0.3}$$

$\Phi \in \mathbb{R}^{m\times n}$ is a Toeplitz matrix constructed using $\boldsymbol{u} \in \mathbb{R}^m$ such that for $\boldsymbol{u} = [0, u_1, u_2, \dots, u_{m-1}]$:

$$\Phi = \begin{bmatrix} 0 & 0 & \dots & 0 \\ u_1 & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & u_1 & 0 \\ \vdots & & \vdots & u_1 \\ u_{m-2} & \dots & & \vdots \\ u_{m-1} & u_{m-2} & \dots & u_{m-n} \end{bmatrix}. \tag{6.0.4}$$

The $\Phi \in \mathbb{R}^{m\times n}$ matrix is then used for computing a convolution of $\boldsymbol{\theta} \in \mathbb{R}^n$. Thus, Φ is our model input and $\boldsymbol{\theta}$ is our unknown function referred to as the impulse response. The objective is now to estimate this impulse response.

6.1 The Stable Spline Kernel

As mentioned before, this report uses the Stable Spline kernel for constructing the covariance matrix, as it is well-known that the SS kernel is appropriate for system identification [1].
The kernel is given by

$$\kappa_2^{SS}(s,t,\ell) = \kappa_2(e^{-\ell s}, e^{-\ell s}), \qquad \ell > 0, \qquad s,t \in [0,\infty[, \tag{6.1.1}$$

where $\kappa_p(s,t)$ is known as the Spline Kernel of order p (in this report $p = 2$), defined on $I \in \mathbb{R}$. Kernel warping ensures the stability [5], by performing a variable transformation mapping $I \to [0,1]$ – such that for $s,t = 0$ $\kappa^{SS} = 1$ and for $s,t \to \infty$ $\kappa^{SS} \to 0$. This property is the reason that the SS kernel is well-suited for system identification, as it enforces the prior assumption of stability —- i.e. that the impulse response decays toward zero. The rate of this decay is controlled by the hyperparameter ℓ.
The Stable Spline Kernel is used to construct the covariance matrix K_ℓ in the numerical experiments carried out in this report.

6.2 PML and GCV

Assume that $\boldsymbol{\theta}$ and ε are Gaussian random variables with distributions:

$$\boldsymbol{\varepsilon}|\sigma_n^2 \sim \mathcal{N}\left(\mathbf{0}, \sigma_n^2 I\right), \qquad \boldsymbol{\theta}|\sigma^2, \ell \sim \mathcal{N}(\mathbf{0}, \sigma^2 K_\ell), \tag{6.2.1}$$

where K_ℓ is a covariance matrix with hyperparameter ℓ. By maximizing the marginal likelihood $p(\boldsymbol{y}|\boldsymbol{\theta})$ where $\boldsymbol{y}|\boldsymbol{\theta} \sim \mathcal{N}\left(\mathbf{0}, \Phi K_\ell \Phi^\top + \sigma_n^2 I\right)$ we are able to select the hyperparameter in order to estimate our model (6.0.3), here we simply replace K_ℓ with $\Phi K_\ell \Phi^\top$ in (5.1.6) [6]:

$$\text{PML}(\lambda,\ell) = \frac{1}{2}\log\left|\Phi K_\ell \Phi^\top + \lambda I\right| + \frac{m}{2}(\log 2\pi + 1) + \frac{m}{2}\log\frac{\boldsymbol{y}^\top\left[\Phi K_\ell \Phi^\top + \lambda I\right]^{-1}\boldsymbol{y}}{m}. \tag{6.2.2}$$

Furthermore, when using GCV the hat matrix H, is determined by our model

$$\boldsymbol{y} = \Phi\boldsymbol{\theta} + \boldsymbol{\varepsilon} \tag{6.2.3}$$

thus the estimate for $\hat{\boldsymbol{y}}$ must be

$$\hat{\boldsymbol{y}} = \Phi\hat{\boldsymbol{\theta}}. \tag{6.2.4}$$

$\boldsymbol{\theta}$ is estimated via the MAP estimate:

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\text{argmin}} \frac{1}{2\sigma_n^2}||\boldsymbol{y} - \Phi\hat{\boldsymbol{\theta}}||_2^2 + \frac{1}{2\sigma^2}\hat{\boldsymbol{\theta}}^\top K_\ell^{-1}\hat{\boldsymbol{\theta}} \tag{6.2.5}$$

where the minimizer is found by setting the derivative with respect to $\hat{\boldsymbol{\theta}}$ equal to zero:

$$-\frac{1}{\sigma_n^2}\Phi^\top(\boldsymbol{y} - \Phi\hat{\boldsymbol{\theta}}) + \frac{1}{\sigma^2}K_\ell^{-1}\hat{\boldsymbol{\theta}} = 0 \tag{6.2.6}$$

performing the shift of variables $\frac{\sigma_n^2}{\sigma^2} = \lambda$ equates to

$$-\Phi\boldsymbol{y} + \Phi^\top\Phi\hat{\boldsymbol{\theta}} + \lambda K_\ell^{-1}\hat{\boldsymbol{\theta}} = 0 \Leftrightarrow (\Phi^\top\Phi + \lambda K_\ell^{-1})\hat{\boldsymbol{\theta}} = \Phi^\top\boldsymbol{y} \Leftrightarrow \hat{\boldsymbol{\theta}} = (\Phi^\top\Phi + \lambda K_\ell^{-1})^{-1}\Phi^\top\boldsymbol{y}. \tag{6.2.7}$$

Recalling our observed model, the estimate for $\boldsymbol{y}$ becomes

$$\hat{\boldsymbol{y}} = \Phi\hat{\boldsymbol{\theta}} = \Phi(\Phi^\top\Phi + \lambda K_\ell^{-1})^{-1}\Phi^\top\boldsymbol{y}, \tag{6.2.8}$$

thus the matrix that maps $\boldsymbol{y}$ to $\hat{\boldsymbol{y}}$ is:

$$H = \Phi(\Phi^\top\Phi + \lambda K_\ell^{-1})^{-1}\Phi^\top. \tag{6.2.9}$$

which is utilized in the GCV criterion [7]:

$$\text{GCV}(\lambda, \ell) = \frac{\frac{1}{m}||(I - H)\boldsymbol{y}||_2^2}{(\text{Tr}(I - H)/m)^2}. \tag{6.2.10}$$

6.3 Numerical Variable Reduction Using SVD

When computing PML and GCV we utilize $\tilde{\Phi} = \Phi L$ and compute a Singular Value Decomposition (SVD) of $\tilde{\Phi}$. Note that L stems from the Cholesky factorization of the covariance matrix $K_\ell = LL^\top$ and therefore depends on the hyperparameter ℓ.
For $\tilde{\Phi} \in \mathbb{R}^{m\times n}$, computing the SVD will result in

$$\tilde{\Phi} = U\Sigma V^\top, \tag{6.3.1}$$

where $U \in \mathbb{R}^{m\times m}$, $\Sigma \in \mathbb{R}^{m\times n}$ and $V \in \mathbb{R}^{n\times n}$. Here Σ is an rectangular singular value diagonal matrix and U and V are orthogonal matrices which entails that $U^\top U = I$ and $V^\top V = I$.
When we assume $m \geq n$ we get $U = \begin{bmatrix} U_1 & U_2 \end{bmatrix}$ with $U_1 \in \mathbb{R}^{m\times n}$ and $U_2 \in \mathbb{R}^{m\times(m-n)}$, $\Sigma = \begin{bmatrix} S \\ 0 \end{bmatrix}$, with $S \in \mathbb{R}^{n\times n}$ which then becomes a diagonal matrix consisting of singular values, $S = \text{diag}(s_1, s_2, \ldots, s_n)$. Thus a compact SVD becomes:

$$\tilde{\Phi} = U_1 S V^\top. \tag{6.3.2}$$

6.3.1 GCV

If we consider the numerator of the GCV criterion, $||(I - H)\boldsymbol{y}||_2^2$ we can write

$$||(I - H)\boldsymbol{y}||_2^2 = ||U^\top(I - H)U\tilde{\boldsymbol{y}}||_2^2, \tag{6.3.3}$$

as U is an orthogonal matrix and $\boldsymbol{y} = U\tilde{\boldsymbol{y}}$.
By reformulating H (6.2.9), replacing $K_\ell = LL^\top$ and utilizing $\tilde{\Phi} = \Phi L$ we get:

$$\begin{aligned} H =& \Phi(\Phi^\top\Phi + \lambda L^{-\top}L^{-1})^{-1}\Phi^\top && (6.3.4)\\ =& \Phi(L^{-\top}(L^\top\Phi^\top\Phi L + \lambda I)L^{-1})^{-1}\Phi^\top && (6.3.5)\\ =& \Phi(L^{-\top}(\tilde{\Phi}^\top\tilde{\Phi} + \lambda I)L^{-1})^{-1}\Phi^\top && (6.3.6)\\ =& \Phi L(\tilde{\Phi}^\top\tilde{\Phi} + \lambda I))^{-1}L^\top\Phi^\top && (6.3.7)\\ =& \tilde{\Phi}(\tilde{\Phi}^\top\tilde{\Phi} + \lambda I)^{-1}\tilde{\Phi}^\top. && (6.3.8) \end{aligned}$$

Furthermore, using an SVD (6.3.1), results in:

$$\begin{aligned} H &= U\Sigma V^\top(V\Sigma^\top U^\top U\Sigma V^\top + \lambda I)^{-1}V\Sigma^\top U^\top \\ &= U\Sigma V^\top(V\Sigma^\top\Sigma V^\top + \lambda I)^{-1}V\Sigma^\top U^\top \\ &= U\Sigma V^\top V(\Sigma^\top\Sigma + \lambda I)^{-1}V^\top V\Sigma^\top U^\top \\ &= U\Sigma(\Sigma^\top\Sigma + \lambda I)^{-1}\Sigma^\top U^\top. \end{aligned} \tag{6.3.9}$$

In conclusion, H can now be expressed as:

$$H = U\Sigma(\Sigma^\top\Sigma + \lambda I)^{-1}\Sigma^\top U^\top. \tag{6.3.10}$$

By substituting this expression for H in (6.3.3) we get:

$$\begin{aligned} ||U^\top(I-H)U\tilde{\boldsymbol{y}}||_2^2 &= ||U^\top(I - U\Sigma(\Sigma^\top\Sigma + \lambda I)^{-1}\Sigma^\top U^\top)U\tilde{\boldsymbol{y}}||_2^2 \\ &= ||U^\top U\tilde{\boldsymbol{y}} - U^\top U\Sigma(\Sigma^\top\Sigma + \lambda I)^{-1}\Sigma^\top U^\top U\tilde{\boldsymbol{y}}||_2^2 \\ &= ||\tilde{\boldsymbol{y}} - \Sigma(\Sigma^\top\Sigma + \lambda I)^{-1}\Sigma^\top\tilde{\boldsymbol{y}}||_2^2. \end{aligned} \tag{6.3.11}$$

Suppose $\tilde{\boldsymbol{y}} = \begin{bmatrix}\tilde{\boldsymbol{y}}_1\\ \tilde{\boldsymbol{y}}_2\end{bmatrix}$, where $\tilde{\boldsymbol{y}}_1 \in \mathbb{R}^n$ and $\tilde{\boldsymbol{y}}_2 \in \mathbb{R}^{m-n}$. When making use of the compact SVD (6.3.2), the expression inside the norm $||\cdot||_2^2$ can be expressed as:

$$\tilde{\boldsymbol{y}} - \Sigma(\Sigma^\top\Sigma + \lambda I)^{-1}\Sigma^\top\tilde{\boldsymbol{y}} = \begin{bmatrix}\tilde{\boldsymbol{y}}_1\\ \tilde{\boldsymbol{y}}_2\end{bmatrix} - \begin{bmatrix}S(S^\top S + \lambda I)^{-1}S\tilde{\boldsymbol{y}}_1\\ \mathbf{0}\end{bmatrix}. \tag{6.3.12}$$

This then simplifies to:

$$||(I-H)\boldsymbol{y}||_2^2 = ||\tilde{\boldsymbol{y}}_1 - S(S^2 + \lambda I)^{-1}S\tilde{\boldsymbol{y}}_1||_2^2 + ||\tilde{\boldsymbol{y}}_2||_2^2. \tag{6.3.13}$$

The term $||\tilde{\boldsymbol{y}}_2||_2^2$ can be reformulated, when

$$\begin{bmatrix}\tilde{\boldsymbol{y}}_1\\ \tilde{\boldsymbol{y}}_2\end{bmatrix} = \tilde{\boldsymbol{y}} = U^\top\boldsymbol{y} = \begin{bmatrix}U_1^\top\boldsymbol{y}\\ U_2^\top\boldsymbol{y}\end{bmatrix}, \tag{6.3.14}$$

therefore

$$||\tilde{\boldsymbol{y}}_2||_2^2 = ||U_2^\top\boldsymbol{y}||_2^2 = \boldsymbol{y}^\top U_2U_2^\top\boldsymbol{y}. \tag{6.3.15}$$

The identity matrix I can be constructed using U_1 and U_2:

$$\begin{aligned} I &= UU^\top \\ &= \begin{bmatrix} U_1 & U_2 \end{bmatrix}\begin{bmatrix} U_1 & U_2 \end{bmatrix}^\top \\ &= U_1U_1^\top + U_2U_2^\top \end{aligned} \tag{6.3.16}$$

Substituting this expression results in:

$$||\tilde{\boldsymbol{y}}_2||_2^2 = \boldsymbol{y}^\top (I - U_1 U_1^\top)\boldsymbol{y}. \tag{6.3.17}$$

In conclusion the numerator of the GCV criterion is now reduced to being expressed by only $\boldsymbol{y}$, S, λ and U_1:

$$||(I-H)\boldsymbol{y}||_2^2 = ||\tilde{\boldsymbol{y}}_1 - S(S^2+\lambda I)^{-1} S\tilde{\boldsymbol{y}}_1||_2^2 + \boldsymbol{y}^\top (I - U_1 U_1^\top)\boldsymbol{y}. \tag{6.3.18}$$

If we consider the denominator of the GCV criterion $\mathsf{Tr}(I-H)$ we can use the SVD by utilizing the expression obtained for H in (6.3.10)

$$\begin{aligned} \mathsf{Tr}(I-H) &= \mathsf{Tr}(U^\top (I-H)U) \\ &= \mathsf{Tr}(U^\top (I - U\Sigma(\Sigma^\top\Sigma + \lambda I)^{-1}\Sigma^\top U)U^\top) \\ &= \mathsf{Tr}(I - \Sigma(\Sigma^\top\Sigma + \lambda I)^{-1}\Sigma^\top), \end{aligned} \tag{6.3.19}$$

because we are left with only diagonal matrices the trace becomes

$$\mathsf{Tr}(I-H) = m - \sum_{i=1}^{n} \frac{s_i^2}{s_i^2 + \lambda} \tag{6.3.20}$$

which is simpler as it only requires $\mathsf{diag}(\Sigma) = \mathsf{diag}(S)$ and λ.

6.3.2 PML

We are also able to use SVD when computing the determinant in the PML criterion, $\log\left|\Phi K_\ell \Phi^\top + \lambda I\right|$, in (6.2.2) which leads to

$$\log\left|\Phi K_\ell \Phi^\top + \lambda I\right| = \log\left|\Phi L L^\top \Phi^\top + \lambda I\right| = \log\left|\tilde{\Phi}\tilde{\Phi}^\top + \lambda I\right| \tag{6.3.21}$$

If we add $\log\left|U^\top(\tilde{\Phi}\tilde{\Phi}^\top + \lambda I)U\right|$, as U is an orthogonal matrix, and we replace $\tilde{\Phi} = U\Sigma V^\top$ we obtain:

$$\begin{aligned} \log\left|U^\top(\tilde{\Phi}\tilde{\Phi}^\top + \lambda I)U\right| &= \log\left|U^\top(U\Sigma V^\top V\Sigma^\top U^\top + \lambda I)U\right| \\ &= \log\left|U^\top(U\Sigma\Sigma^\top U^\top + \lambda I)U\right| \\ &= \log\left|U^\top U\Sigma\Sigma^\top U^\top U + \lambda U^\top U)\right| \\ &= \log\left|\Sigma\Sigma^\top + \lambda I\right|. \end{aligned} \tag{6.3.22}$$

Considering the expression in $\log|\cdot|$

$$\Sigma\Sigma^\top = \begin{bmatrix} S \\ 0 \end{bmatrix}\begin{bmatrix} S & 0 \end{bmatrix} + \lambda I = \begin{bmatrix} \lambda I + S^2 & 0 \\ 0 & \lambda I \end{bmatrix}, \tag{6.3.23}$$

because we now have a diagonal matrix the log determinant becomes:

$$\log\left|\Sigma\Sigma^\top + \lambda I\right| = \sum_{i=1}^{n} \log(s_i^2 + \lambda) + (m-n)\log\lambda, \tag{6.3.24}$$

recalling that $S \in \mathbb{R}^{n\times n}$ therefore $\lambda I \in \mathbb{R}^{(m-n)\times(m-n)}$, the log determinant of this matrix becomes $(m-n)\log\lambda$.

Lastly, we consider the last term in (6.2.2), $\frac{n}{2}\log\frac{\boldsymbol{y}^\top\left[\Phi K_\ell\Phi^\top+\lambda I\right]^{-1}\boldsymbol{y}}{n}$ the numerator becomes:

$$\begin{aligned}
\boldsymbol{y}^\top\left[\Phi K_\ell\Phi^\top+\lambda I\right]^{-1}\boldsymbol{y} &= \boldsymbol{y}^\top\left[\Phi LL^\top\Phi^\top+\lambda I\right]^{-1}\boldsymbol{y}\\
&= \boldsymbol{y}^\top\left[\tilde{\Phi}\tilde{\Phi}^\top+\lambda I\right]^{-1}\boldsymbol{y}\\
&= \boldsymbol{y}^\top\left[U\Sigma V^\top V\Sigma^\top U^\top+\lambda I\right]^{-1}\boldsymbol{y}\\
&= \boldsymbol{y}^\top\left[U\Sigma\Sigma^\top U^\top+\lambda I\right]^{-1}\boldsymbol{y}\\
&= \boldsymbol{y}^\top U\left[\Sigma\Sigma^\top+\lambda I\right]^{-1}U^\top\boldsymbol{y}\\
&= \tilde{\boldsymbol{y}}^\top\left[\Sigma\Sigma^\top+\lambda I\right]^{-1}\tilde{\boldsymbol{y}}
\end{aligned} \tag{6.3.25}$$

Recalling that $\Sigma=\begin{bmatrix}S\\0\end{bmatrix}$

$$\begin{bmatrix}\tilde{\boldsymbol{y}}_1^\top & \tilde{\boldsymbol{y}}_2^\top\end{bmatrix}\begin{bmatrix}\left(S^2+\lambda I\right)^{-1} & 0\\0 & \lambda^{-1}I\end{bmatrix}\begin{bmatrix}\tilde{\boldsymbol{y}}_1\\\tilde{\boldsymbol{y}}_2\end{bmatrix}=\tilde{\boldsymbol{y}}_1^\top\left(S^2+\lambda I\right)^{-1}\tilde{\boldsymbol{y}}_1+\tilde{\boldsymbol{y}}_2^\top\lambda^{-1}I\tilde{\boldsymbol{y}}_2. \tag{6.3.26}$$

Noting the inverse matrix consist of only diagonal elements, we can rewrite the term as a sum:

$$\tilde{\boldsymbol{y}}_1^\top\left(S^2+\lambda I\right)^{-1}\tilde{\boldsymbol{y}}_1=\sum_{i=1}^{n}\frac{\tilde{y}_i^2}{s_i^2+\lambda} \tag{6.3.27}$$

The last terms can then be expressed only in terms of U_1 and $\boldsymbol{y}$ using (6.3.16)

$$\tilde{\boldsymbol{y}}_2^\top\lambda^{-1}I\tilde{\boldsymbol{y}}_2=\lambda^{-1}\boldsymbol{y}^\top U_2U_2^\top\boldsymbol{y}=\lambda^{-1}\boldsymbol{y}^\top(I-U_1U_1^\top)\boldsymbol{y}. \tag{6.3.28}$$

In conclusion, using SVD we can write

$$\boldsymbol{y}^\top\left[\Phi K_\ell\Phi^\top+\lambda I\right]^{-1}\boldsymbol{y}=\sum_{i=1}^{n}\frac{\tilde{y}_i^2}{s_i^2+\lambda}+\lambda^{-1}\boldsymbol{y}^\top(I-U_1U_1^\top)\boldsymbol{y} \tag{6.3.29}$$

We hereby are able to compute PML using only diag(S), λ, $\boldsymbol{y}$ and U_1.

6.4 Measure of Fit

In order to give an assessment of the quality of the estimate $\hat{\boldsymbol{\theta}}$, we compute the fit-score as:

$$\text{fit}(\hat{\boldsymbol{\theta}})=100\cdot\left(1-\frac{||\boldsymbol{\theta}_{\text{true}}-\hat{\boldsymbol{\theta}}||_2}{||\boldsymbol{\theta}_{\text{true}}-\mathbf{1}\mu||_2}\right),\qquad \mu=\frac{1}{q}\mathbf{1}^\top\boldsymbol{\theta}_{\text{true}}, \tag{6.4.1}$$

where $\boldsymbol{\theta}_{\text{true}}\in\mathbb{R}^q$. Thus, a good fit will yield a value close to 100.

6.5 Implementing the Hyperparameter Selection

As mentioned, this report uses Bayesian optimization for minimizing the cost-functions. As previously unfolded, using SVD allows us to compute the PML and GCV critetion in terms of diag(S), λ, $\boldsymbol{y}$ and U_1. This allows us to eliminate the hyperparameter λ by simply selecting it using grid search. This results in a function which only depends on ℓ for which Bayesian optimization is used for selecting ℓ. The target function for the BO is evaluated as follows:

Algorithm 2 TargetFunction

procedure TargetFunction($\ell, \boldsymbol{y}, \Phi, \lambda_{\text{min}}, \lambda_{\text{max}}$)
  $\boldsymbol{t} \leftarrow [0, 1, 2, \dots]$
  $K_\ell \leftarrow K(\boldsymbol{t}, \boldsymbol{t}, \ell)$
  $L \leftarrow \text{Cholesky}(K_\ell)$
  $\tilde{\Phi} \leftarrow \Phi L$
  $U_1, S, V^\top \leftarrow \text{SVD}(\tilde{\Phi})$
  $\boldsymbol{\lambda} \leftarrow [\lambda_{\text{min}}, \dots, \lambda_{\text{max}}]$
  for $i = 1 \dots p$ **do**
    $c_i \leftarrow \text{PML/GCV}(\boldsymbol{\lambda}_i, S, \boldsymbol{y}, U_1)$ ▷ Evaluating PML or GCV for a given ℓ in a grid for varying λ-values (Figure 6.2).
  return min$(\boldsymbol{c})$

Note that λ is selected each time the target function is evaluated. Figure 6.2 illustrates how, for a fixed value of ℓ, λ is chosen via grid search. This approach allows λ to be selected while performing the SVD only once, which is a significant computational advantage.

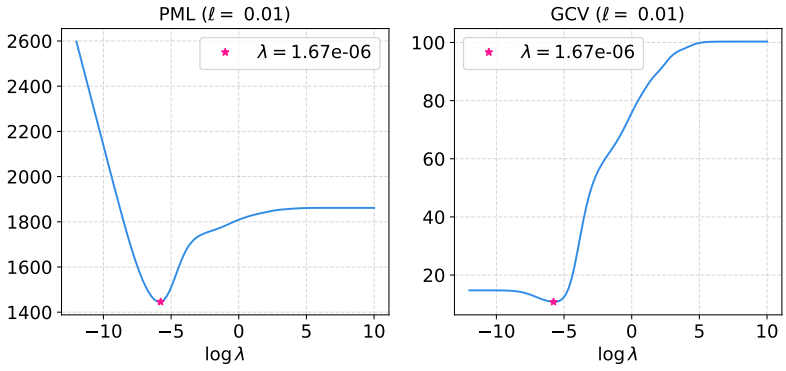


Figure 6.2: PML and GCV as a function of λ.

When running the Baysian optimization for either PML or GCV we are able to select ℓ as seen in Figure 6.3 and using the selected hyperparameter we can compute the estimated impulse response (Figure 6.4).

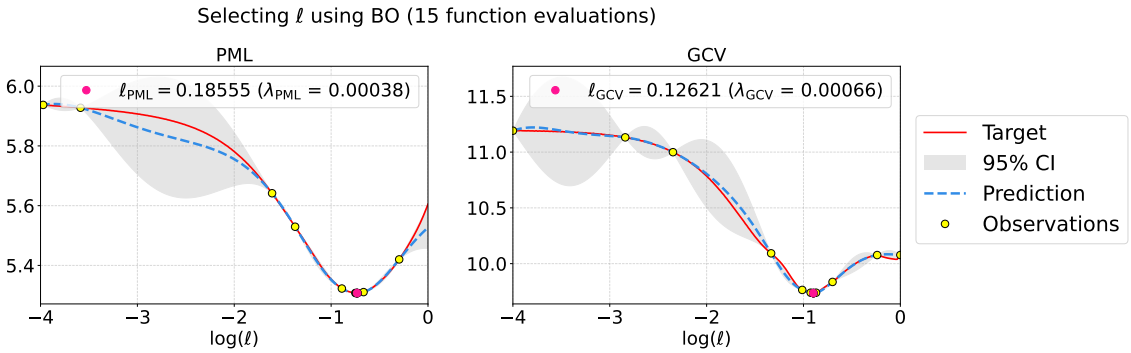


Figure 6.3: Results using PML and GCV for selecting λ and ℓ.

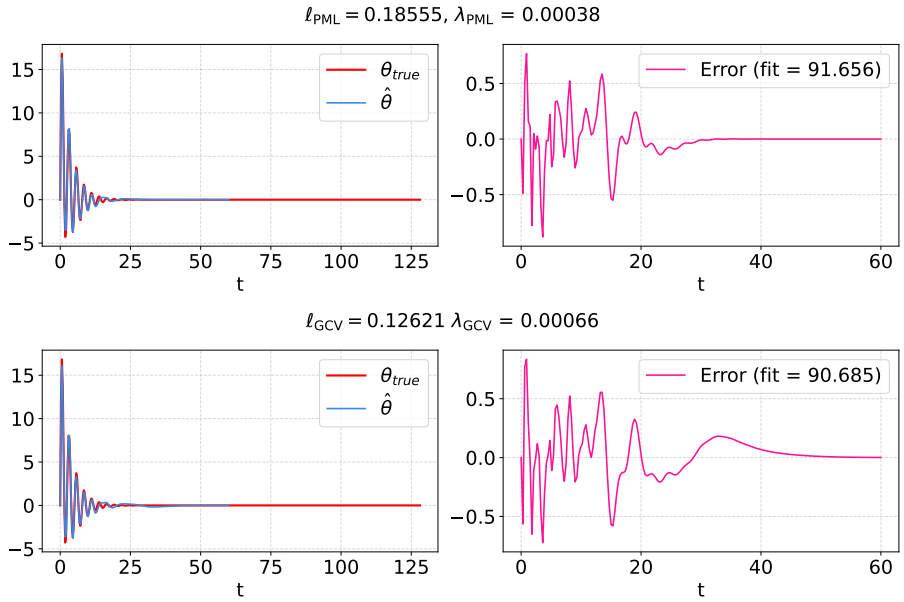


Figure 6.4: Estimated impulse response using PML and GCV.

Based on these results we see that for this instance we for both methods obtain fits close to 100 and in general the Baysian optimizer is able to select ℓ well in only 15 function evaluations.

6.6 Comparing Bayesian Optimization with Grid Search

The aim for using Bayesian optimization (BO) is to arrive at the optimum of the target function by choosing where to evaluate the target function next wisely, via. the chosen acquisition function (Section 4.1). With this method fewer time-costly function evaluations are needed and the assumption is therefore that BO will be more efficient. In order to investigate this assumption we compare the method to grid search, selecting ℓ by evaluating the target function an equal number of times (as BO) for ℓ in an evenly spaced logarithmic grid. In Figure 6.5 we see that BO does not necessarily yield better results with respect to the fit-score. However, we do notice that BO for PML is the most stable method furthermore we see that after 20 function evaluations we do not detect an increase in the results. We see the same picture when using GCV (see Appendix A.1). When we consider the target function for PML and GCV in Figure 6.3 the minimum appears well defined and we will therefore not expect a large number of function evaluations in order to obtain a good result. This is confirmed by the results shown in Figure 6.6 where it is clear that BO arrives at the optimum in far fewer function evaluations (the same applies for GCV see Appendix A.3).

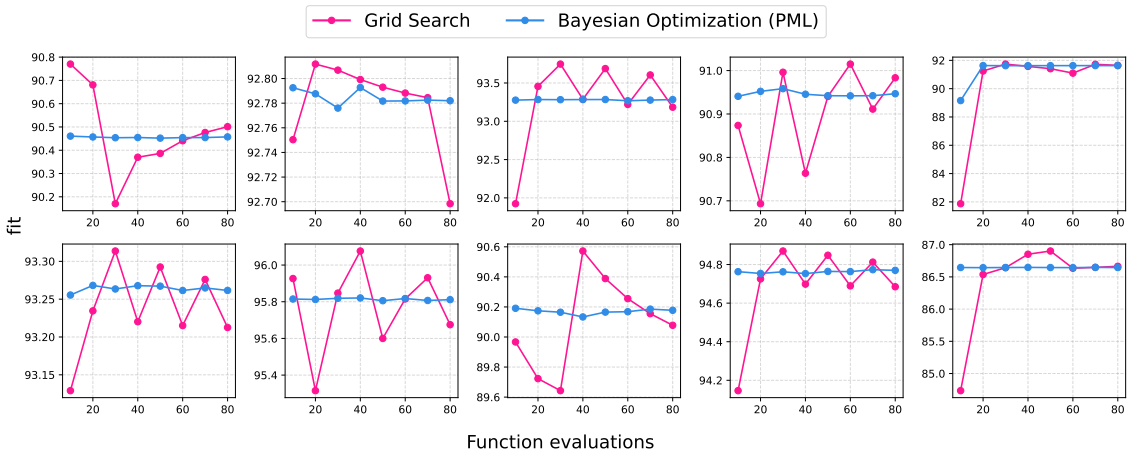


Figure 6.5: Fit for 10 SSM for varying numbers of function evaluations using PML.

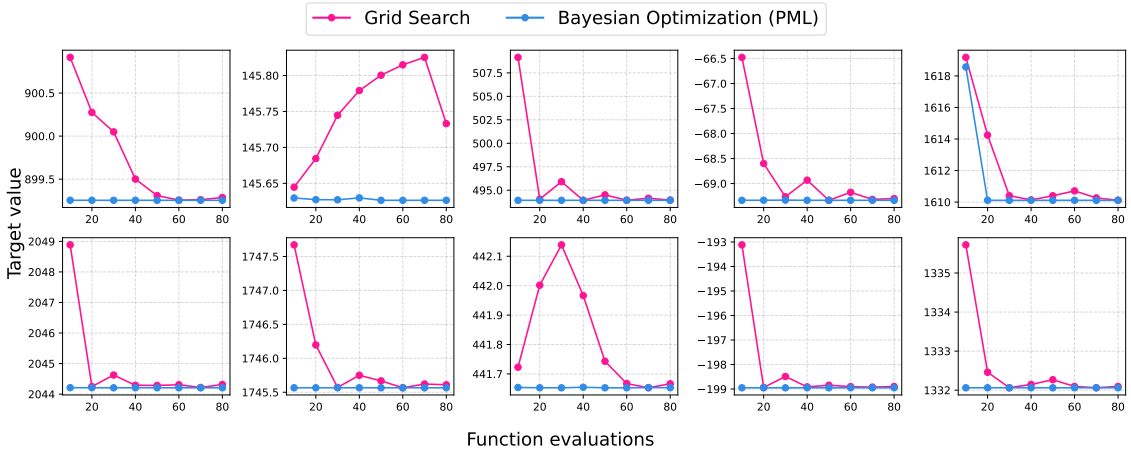


Figure 6.6: Minimum target function value for 10 SSM for varying numbers of function evaluations using PML.

In the following experiments we have chosen to use 15 function evaluations when using BO, where 5 function evaluations are chosen randomly i.e. emphasizing exploration, and 10 are chosen via the acquisition function $\rho = 2.576$ i.e. emphasizing the exploitation. We assume that the optimum is well-defined as seen in Figure 6.3 therefore we put the largest emphasis on exploitation.

7 Combining PML and GCV

In addition to using PML and GCV separately for selecting the hyperparameters, we will now investigate if combining the two functions can improve our results. We therefore introduce the parameter β that weights the two cost-function on a log-scale:

$$f(\beta) = \frac{2}{m} \cdot \text{PML}(\lambda, \ell) + \beta \cdot \text{GCV}(\lambda, \ell), \qquad \beta \in [10^{-7}, 10^{7}]. \tag{7.0.1}$$

Here β controls the combination of PML and GCV. Note that PML is multiplied with $2/m$ scaling PML in order to make the cost-functions output values more similar. In practice this is implemented similarly to algorithm 2 where line 9 becomes $c_i \leftarrow \frac{2}{m}\,\text{PML}(\boldsymbol{\lambda}_i, S, \boldsymbol{\theta}, U_1) + \beta\,\text{GCV}(\boldsymbol{\lambda}_i, S, \boldsymbol{\theta}, U_1)$.

7.1 Numerical Experiments

In order to investigate whether it is advantageous combining the two cost-function we compute the fit for a range of values for $\beta \in [10^{-7}, 10^{7}]$ for 50 random instances of the state space model. Note that for the numerical experiments carried out in this report $m = 500$, $n = 200$ and the signal-to-noise ratio (SNR) $= 10$.

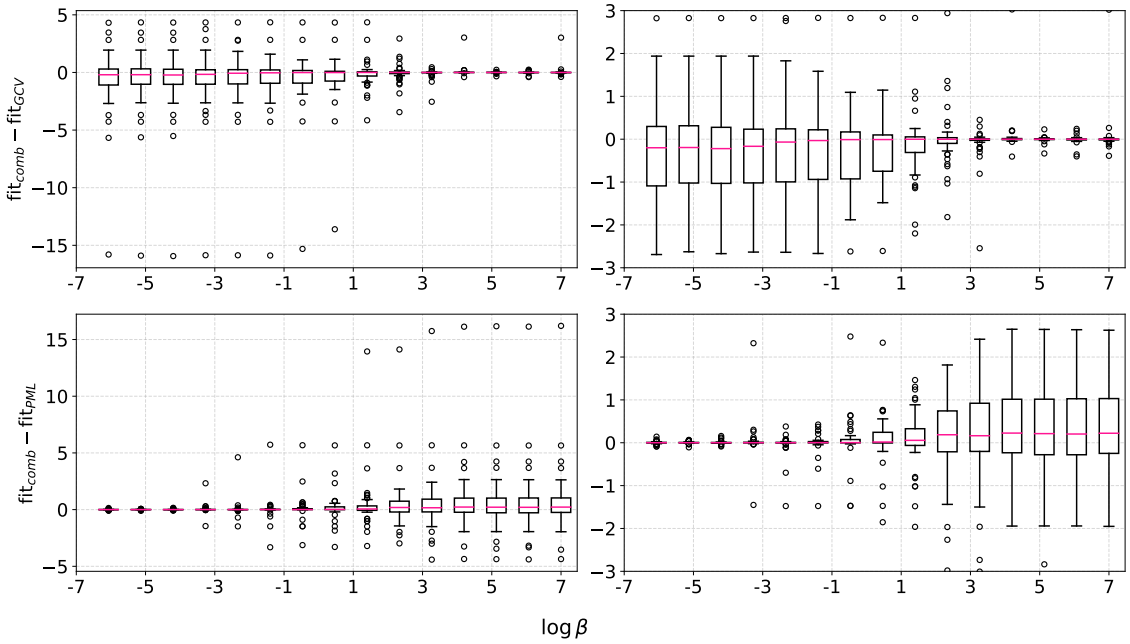


Figure 7.1: Difference in fit for 50 random instances of the SSM.

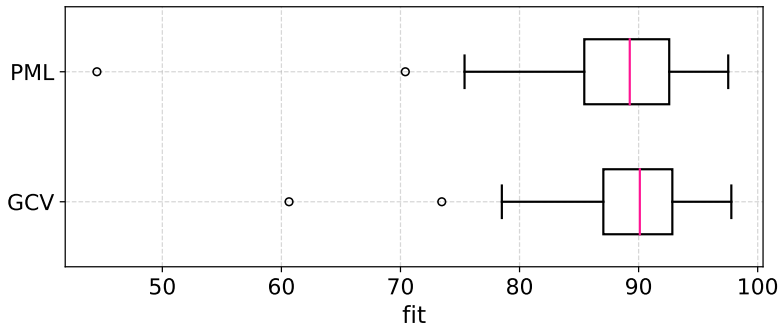


Figure 7.2: Distribution of fit for 50 random instances of the SSM using PML and GCV.

When considering the results shown in Figure 7.1 we see that combining the functions can yield better results. In specific if we emphasize GCV (large values of β), despite the results of using the methods separately yield similar results (Figure 7.2). However, even when considering the for "best" $\beta \approx 10^4$ we see that around 40 % of the instances also yield a negative improvement in fit.

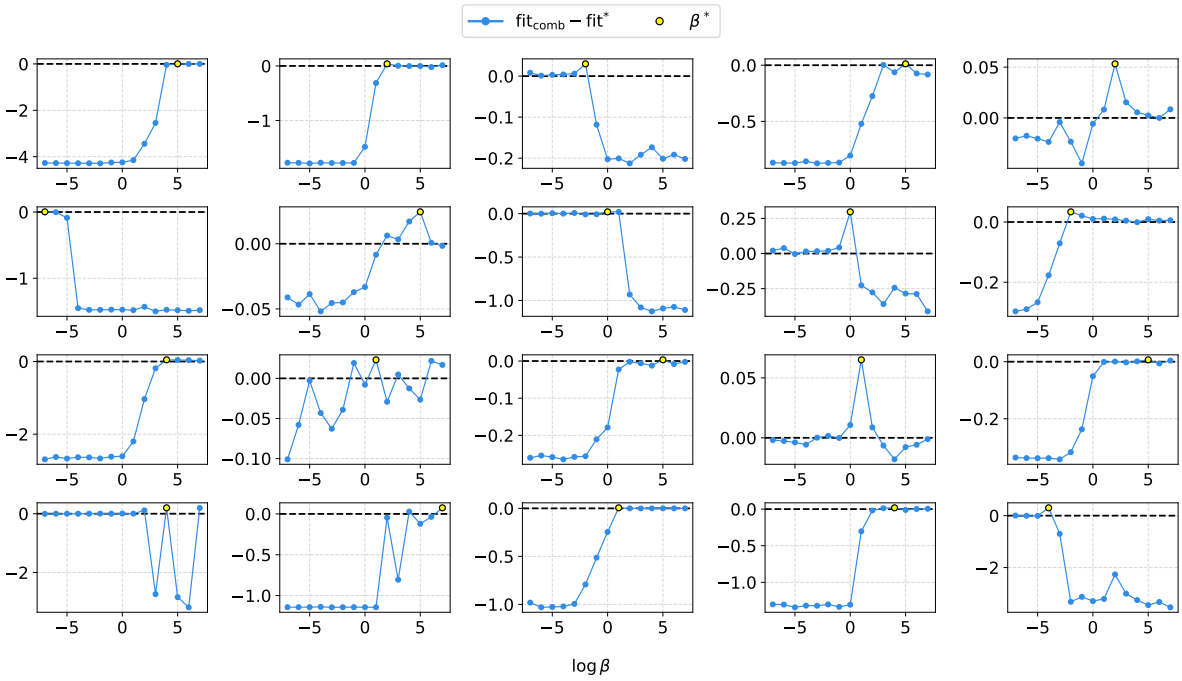


Figure 7.3: Difference in fit for 20 SSM for $\beta \in [10^{-7}, 10^{7}]$ (see Appendix A.4 for the remaining 30 SSM).

In Figure 7.3 & 7.4 we see that for almost all the instances of the SSM we are able to achieve an improvement in fit relatively to the best fit of GCV or PML. On the left we wee how the β-values are distributed in relation to the improvement of fit – here we see that 84% of the improvements are below 0.1. If we consider the distribution of β^*-values we see how there is no clear pattern showing how the "best" β should be chosen, however there is a majority of large β-values i.e. emphasizing GCV.

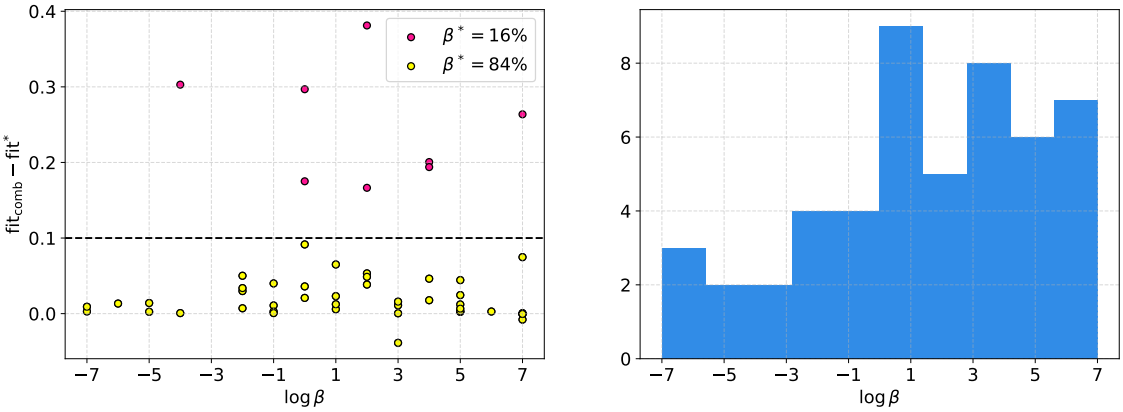


Figure 7.4: Distribution of best β^* and difference in fit for 50 SSM.

Previously, we have done experiments on training data with SNR $= 10$. By varying SNR and considering the improvement in fit relatively to the fit using exclusively PML, we see that noisier training data, i.e. small SNR, results in a larger range for improvement of fit. This is expected as less noisy data will yield better results, therefore the room for improvement by combining the methods decreases. In Figure 7.5 we see the difference for SNR $= 5$ & 30 (see Appendix A.2 for SNR $= 5, 10, 15, 20, 25, 30$).

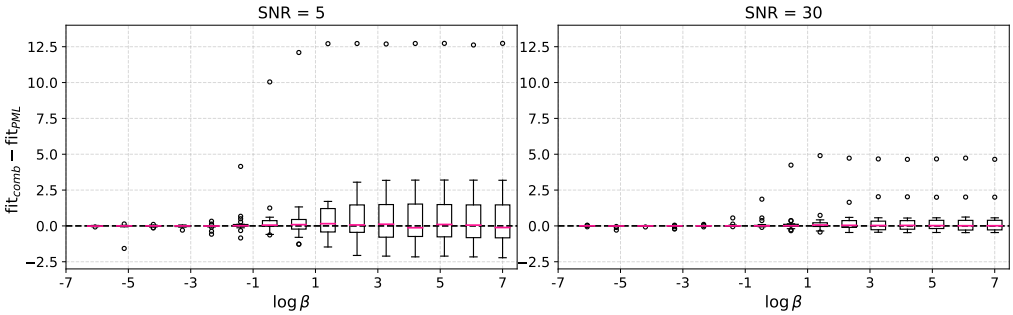


Figure 7.5: Difference in fit relatively to PML for 20 SSM for snr $= 5$ & 30.

8 Discussion

In this report it is confirmed that using the Stable Spline Kernel for estimating linear Gaussian models such as the impulse response for the state space model yield advantageous results. It is also found that using both PML and GCV for selecting the hyperparameters yield highly similar results for which we see high fit-scores. When using PML or GCV for selecting the hyperparameters we used grid search for selecting λ and Bayesian optimization for selecting ℓ. In addition, we investigated how using grid search for selecting ℓ, with an equal number of function evaluation, affect the results. When we considered the results side-by-side grid search yielded inconsistent results for the fit while increasing the number of function evaluation – in some cases grid search yielded better results, in some cases worse. In comparison, BO yielded fits that were mostly unaffected by increasing the number of function evaluations. This gave an incentive to use BO with only 15 function evaluations. In addition, we considered the two methods ability to minimize the cost-function and here it was clear that BO is more efficient both with respect to number of function evaluations and by minimizing the cost-function. Here we see a discrepancy between minimizing the cost-function and what fit-scores we get. This discrepancy is due to the fact that minimizing the PML or GCV cost-function is not directly equivalent to minimizing the fit-score.

The report also uncovers the hyperparameter λ can be eliminated analytically, allowing us to evaluate the cost-functions for a range of λ-values computing the SVD of $\tilde{\Phi}$ just once, which is a significant computational advantages. A target function only dependent on ℓ is obtained and here BO is used for selecting the ”best” ℓ. If we had not exploited this property BO could have been used for selecting both hyperparameter by minimizing the cost-functions in a 3-dimensional space, however we would have needed more costly function evaluation to reach the minimum.

A general weakness with the results is that computing the fit-score requires that the true impulse response is known and this is of course not the case in reality. The problem of knowing the true impulse response also arises when we experiment with combining the PML and GCV cost-functions. Because although we for the vast majority are able to improve the fit by combining the cost-functions, there is no clear pattern in which value of β leads to an improvement in fit, nor which value of β provides the largest improvement. For example, $\beta \approx 10^4$ appears to yield the best results overall, in spite of this 40% of the SSM yield negative improvement in fit for this β-value. This leads us with the problem of obtaining a measure for the fit without knowing the true impulse response. It would therefore have been meaningful to consider the observed output $\boldsymbol{y}$, in comparison with the estimated output, $\hat{\boldsymbol{y}} = \Phi\hat{\boldsymbol{\theta}}$, taking into account that $\boldsymbol{y}$ is the noisy version of the true output. This would have allowed us to investigate the results with a measure of fit that would be more applicable to a real scenario.

It could also have been insightful to explore other ways of parametrizing the combination of the cost-functions. We could have simply selected λ and ℓ using PML and GCV sepereately, then compute a combination of the values i.e. $\lambda_{\text{comb}} = \delta\lambda_{\text{PML}} + (1-\delta)\lambda_{\text{GCV}}$ and $\ell_{\text{comb}} = \gamma\ell_{\text{PML}} + (1-\gamma)\ell_{\text{GCV}}$ and have computed the fits for a range of values of $\delta, \gamma \in [0, 1]$. Once again the same weakness arises in regards to the measure of fit, however it would be useful if we were to see a clear pattern in how δ and γ should be selected.

In general, the results of combining the cost-functions have shown that there certainly are improvements to obtain however when considering the improvement for the ”best” β for 50 SSM 86% of them are below 0.1 then one could argue if the complexity of combining the functions and selecting the β-value is worth an improvement in the scale that we see in this report. This being said, it is shown that for the problem explored in this report it is correct to assume that the fit can be improved by combining PML and GCV. On the basis of that finding it would be interesting to do more experiments with the combined function in order to obtain larger improvements.

The results also showed that improving the fit by combining the cost-functions is most advantageous when the model output are more noisy. This observation can be used to decide whether combining the cost-functions is useful, if one is able to draw conclusion on the signal-to-noise ratio. Furthermore, it also goes to show that the results dealt with in this report are dependent on the choices made for the numerical experiments and would be different if for example the SNR had been chosen differently etc.

9 Conclusion

In this report the results demonstrate that the Stable Spline kernel is an effective prior for estimating the impulse response. Furthermore, both the PML and GCV criteria prove to be useful methods for hyperparameter selection. Here Bayesian optimization is an advantageous tool for optimizing these cost-functions as it is able to arrive at the optimum in few number of functions evaluations in comparison to grid search. Furthermore, we have found that we are able to obtain an improvement of fit when combining the two cost-functions. However, there is no clear pattern in how the cost-functions are weighed in relation to each other. Therefore the method is not directly applicable for new unknown systems. Furthermore, the numerical experiments carried out in this report show relatively small improvements however given that the result are dependent on the choices made for the numerical experiments one can assume that these could be improved as there still remains unexhausted ways to combine the cost-functions.

Bibliography

[1] Gianluigi Pillonetto and Giuseppe De Nicolao. “A new kernel-based approach for linear system identification”. In: *Automatica* 46.1 (2010), pp. 81–93. issn: 0005-1098. doi: https://doi.org/10.1016/j.automatica.2009.10.031. url: https://www.sciencedirect.com/science/article/pii/S0005109809004920.

[2] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[3] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. 1st ed. Springer, 2007. isbn: 0387310738.

[4] Roman Garnett. *Bayesian Optimization*. Cambridge University Press, 2023.

[5] Martin S. Andersen and Tianshi Chen. “Smoothing Splines and Rank Structured Matrices: Revisiting the Spline Kernel”. English. In: *SIAM Journal on Matrix Analysis and Applications* 41.2 (2020), pp. 389–412. issn: 0895-4798. doi: 10.1137/19m1267349.

[6] Tianshi Chen and Lennart Ljung. “Implementation of algorithms for tuning parameters in regularized least squares problems in system identification”. In: *Automatica* 49.7 (2013), pp. 2213–2220. issn: 0005-1098. doi: https://doi.org/10.1016/j.automatica.2013.03.030. url: https://www.sciencedirect.com/science/article/pii/S0005109813001994.

[7] Giulio Bottegal and Gianluigi Pillonetto. “The generalized cross validation filter”. In: *Automatica* 90 (2018), pp. 130–137. issn: 0005-1098. doi: https://doi.org/10.1016/j.automatica.2017.12.054. url: https://www.sciencedirect.com/science/article/pii/S0005109817306416.

A Additional Figures

A.1

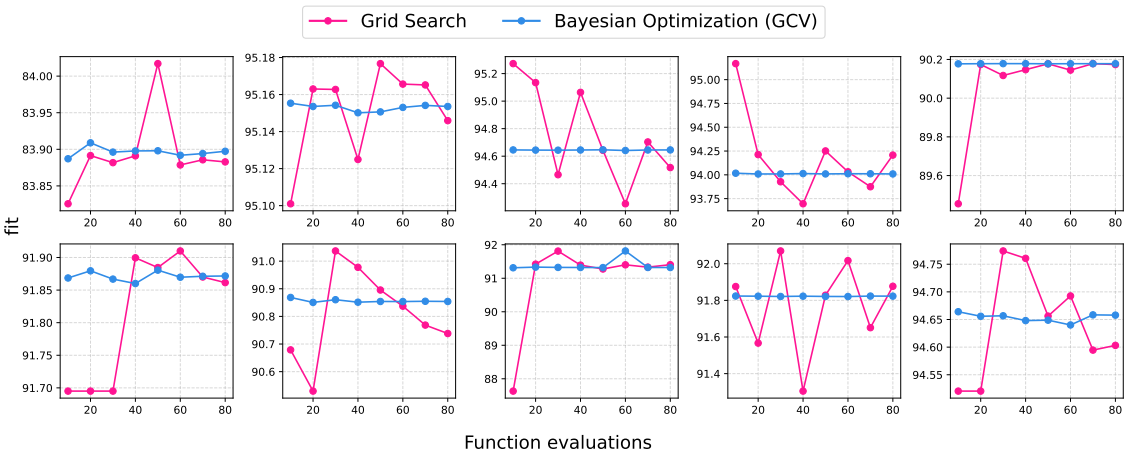


Figure A.1: Fit for 10 SSM for varying numbers of function evaluations using GCV.

A.2

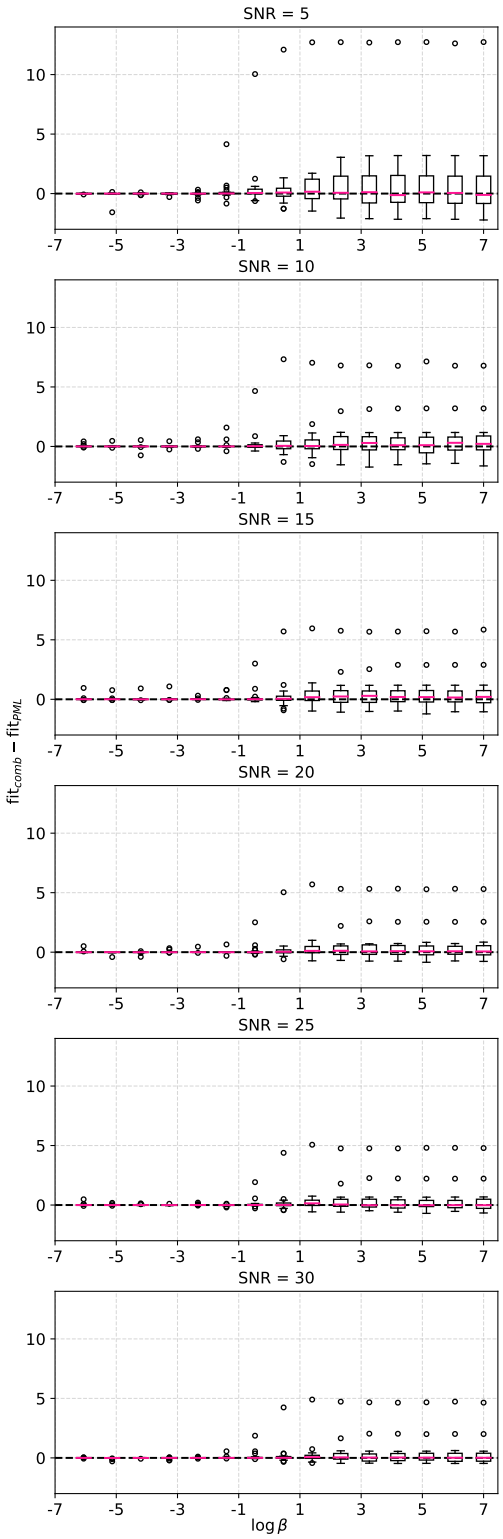


Figure A.2: Difference in fit relatively to PML for 20 SSM.

A.3

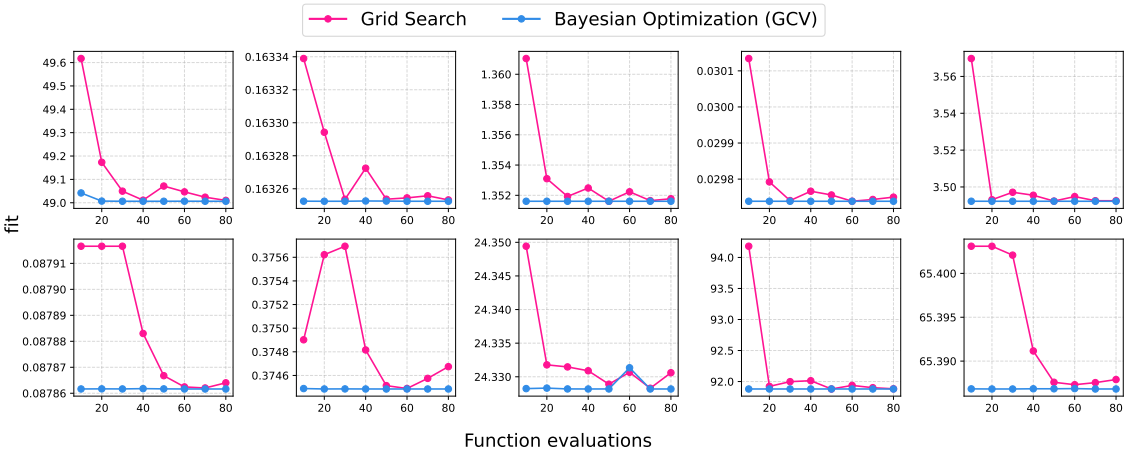


Figure A.3: Minimum target function value for 10 SSM for varying numbers of function evaluations using PML.

A.4

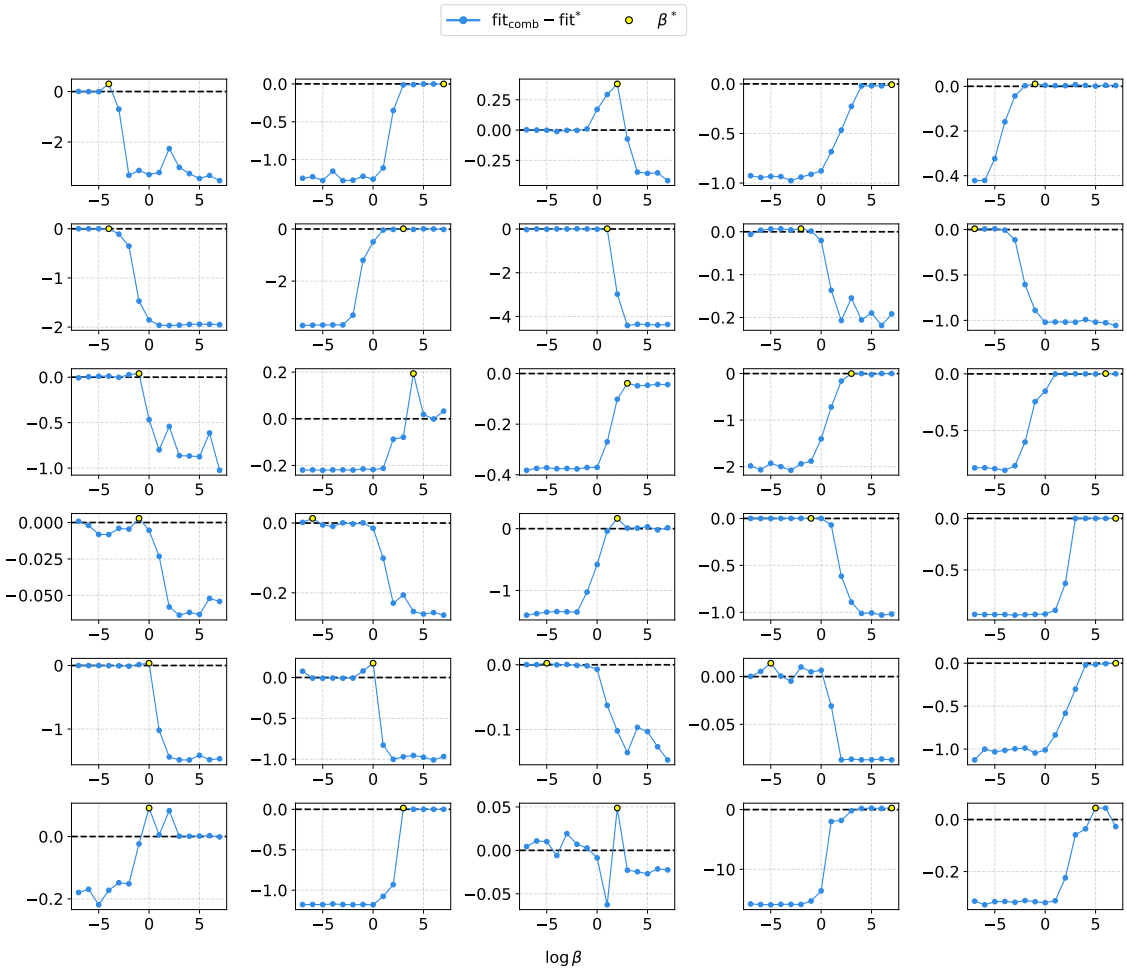


Figure A.4: Difference in fit as function of β for the remaining 30 SSM .

Technical
University of
Denmark

Building 303B
2800 Kgs. Lyngby
Tlf. 4525 1700

www.compute.dtu.dk