Meta-learning Implicit Neural Representation for Sparse Time Series Functional Data Analysis

Bofan Chen
Department of Pure Mathematics and Mathematical Statistics
University of Cambridge
cbfcbf.byron@gmail.com

Abstract

Sparse time series data are everywhere in our daily life. However it is hard to analyze them by traditional statistical methods. Here, we propose MetaINR, a method to learning the implicit neural representation (INR) given sparse obsevations of time series data. Based on this, estimations of mean function, covariance operator, and principal components can be implemented easily. The method introduce the meta-learning to the functional data analysis framework, which greatly outperforms the traditional baseline.

1 Introduction

Sparse time series data is prevalent in daily life, such as medical data measured at different times for patients or height data collected at various stages of adolescence. However, the sparsity and irregularity of such data make traditional statistical methods less suitable. Additionally, the consideration of continuity and differentiability properties in most time series data pose challenges for multivariable statistical analysis.

Recent research suggests that meta-learning exhibits great advantages in handling sparse data. It can learn commonalities among similar tasks, enabling few-shot learning or even one-shot learning on extremely sparse training sets. Besides, statistical analysis based on functional data has been hugely developed for dealing with continuous time series problems recently.

In this paper, we introduce a novel approach, metaINR, which combines meta-learning and functional data analysis to address the estimation challenges posed by sparse time series data. Through this method, we use meta-learning to realize the implicit neural representation (INR) to reconstruct the time series function for each sample based on sparse time series observations. Furthermore, we can estimate the mean, covariance, and principal components of the distribution of these samples. Therefore, MetaINR method presents a fresh solution for addressing issues related to sparse time series data.

2 Related works

2.1 Functional Data

Multivariate data versus functional data Traditional multivariate statistics take finite-dimensional vectors as the objects of study to analyze their statistical properties, such as mean vectors and covariance matrices. This approach is appropriate and easy to compute when the dimension is relatively small. However, if the dimension of the space in which the objects are analyzed is increased to infinite, for example, in L^2 space, traditional methods face certain challenges. In such functional spaces, the focus is not only on the values of vectors in a particular dimension but also on properties

Preprint. Under review.

such as continuity, smoothness and derivatives. At the same time, as the space grows to infinite dimensions, many useful tools in multivariate statistics, such as probability density functions, will lose their meaning. It is the differences in statistical properties and people's focus between infinite-dimensional and finite-dimensional spaces, that have led to widespread attention to functional data analysis (FDA) among researchers.

Time series as a representative example of functional data Time series is a typical object that needs to be addressed in an infinite-dimensional function space. This is because time series data collected in reality often exhibit continuity, and, in certain situations, people may focus on topics such as derivatives and time alignment (registration). Many previous studies have relied on multivariate statistics in the analysis of time series, such as [mention some specific examples]... From the perspective of function approximation, these multivariate-statistics-based studies can indeed reflect certain properties of time series. However, in many cases, this approach is challenging to generalize, as is the case with sparse time series data studied in this paper.

Sparse times series data analysis When the observation frequency of time series is low and irregular, sparse time series data is formed. We can represent the sparse dataset of n samples as

$$D = \{(t_{ij}, y_{ij})_{j=1}^{J_i}\}_{i=1}^{n},$$

where $y_{ij} = x_i(t_{ij}), i = 1, \cdots, n, j = 1, \cdots, J_i$ and J_i is the total observations of the i-th sample. This type of data is particularly common in the medical field, as the frequency of medical visits varies from person to person, with irregular patterns. When it comes to analyzing such "messy" data using multivariate statistics, people often find themselves at a loss. However, in functional data analysis, one can employ the method of local polynomial regression to estimate the mean function and covariance operator of functional data. Specifically, given our sparse observations of dataset D, we can define the loss function to estimate mean function $\mu(t)$

$$L_t^{\mu}(\beta) = \sum_{i=1}^{n}\sum_{j=1}^{J_i}\left[y_{ij} - \sum_{r=0}^{R}\beta_r\,(t - t_{ij})^r\right]^2 K\left(\frac{t - t_{ij}}{h}\right),$$

where $K(\cdot)$ is the kernel function and h is the bandwidth which shoud be carefully chosen based on data. By minimizing this function, we can get $\beta_t = (\beta_0(t), \cdots, \beta_R(t)) = arg\,min L_t^{\mu}(\beta)$. The estimator of mean function cam be defined as $\hat{\mu}(t) = \beta_0(t)$. Similarly, to estimate the covariance kernel $c(t, s)$, we define the loss function

$$L_{t,s}^{c}(\beta) = \sum_{i=1}^{n}\sum_{1\le j,l,\le J_i} K_{t,s}(\frac{t - t_{ij}}{h}, \frac{s - t_{il}}{h})\,[g(t_{ij}, t_{il}) - f\,(\beta, (t, s), (t_{ij}, t_{il}))]^2$$

where

$$g(t_{ij}, t_{il}) = (y_{ij} - \hat{\mu}(t_{ij}))(y_{il} - \hat{\mu}(t_{il}))$$
$$f\,(\beta, (t, s), (t_{ij}, t_{il})) = \beta_0 + \beta_{11}(t - t_{ij}) + \beta_{12}(s - t_{il})$$

By minimization of $L_{t,s}^{c}(\beta)$, we obtain $(\beta_0(t), \beta_{11}(t), \beta_{12}(t)) = arg\,min L_{t,s}^{c}(\beta)$. Then the estimator of the covariance kernel is defined as $\hat{c}(t, s) = \beta_0(t)$. Based on the estimation of mean and covariance, we can use the method of Principal Analysis by Conditional Expectation (PACE) to reconstruct each sample and compute their scores (see details in the appendix). Comparing to using local polynomial regression directly to reconstruct samples, It is noticeable that the reconstruction of the sample makes use of the total of information of all observations, but it hugely relies on the assumption that scores are normally distributed. In the following text, I will compare the estimates obtained based on this method as the baseline with the meta-learning-based approach proposed in this paper.

2.2 Implicit Neural Representation

Traditional grid-based representation versus implicit neural representation Traditionally, the representation of functional data $y(t)$ involves using a standard grid-based method, wherein a finite set of coordinates and corresponding function values are given as pairs (t_i, y_i). However, this approach has certain drawbacks: the original functional data in an infinite-dimensional space is discretized into a finite-dimensional space. This not only hampers operations like differentiation but also significantly impacts the resolution of the function within a smaller time interval. With the emergence of deep learning, it has been discovered that utilizing deep neural networks can be an effective way to represent functions implicitly. Implicit neural representation (INR) parameterize the functional data by deep neural networks:

$$y(t) = F_{\phi}(t),$$

where $F_{\phi}(\cdot)$ is a deep neural network parameterized by ϕ. This type of continuous representation not only proves to be more memory-efficient but also enables higher resolution and the handling of more complex operations on functional data.

SIREN architecture for time series Many prior works have utilized ReLU-based multilayer perceptrons (MLP) as the neural network architecture for Implicit Neural Representations (INR). However, ReLU-based INR often lack the ability to capture fine details in signal and struggle to effectively express higher-order derivatives. In addressing this limitation, [1] introduced the SIREN architecture as an alternative to traditional ReLU-MLPs:

$$\Phi(\mathbf{x}) = \mathbf{W}_n\left(\phi_{n-1} \circ \phi_{n-2} \circ \ldots \circ \phi_0\right)(\mathbf{x}) + \mathbf{b}_n, \quad \mathbf{x}_i \mapsto \phi_i\left(\mathbf{x}_i\right) = \sin\left(\mathbf{W}_i \mathbf{x}_i + \mathbf{b}_i\right)$$

SIREN employs the sine function as the activation function, effectively capturing intricate details in functions. A remarkable observation is that even after taking derivatives of a function represented using SIREN, it can still be expressed using the SIREN architecture. While previous research predominantly applied SIREN to represent images and videos, its application in time series representation has been limited. This paper explores the utilization of SIREN in the analysis of time series and demonstates its superiority in time series INR.

2.3 Meta-learning

Meta-learning framework Meta-learning was initially introduced to enhance the efficiency of machine learning, often referred to as "learning to learn". By employing the deep learning model $F_{\phi}(\cdot)$ to address similar tasks, the "shared characteristics" among these tasks can be leveraged to expedite the machine learning process. To fomulate, given the observation dataset D^* of the target task and the observation dataset $D = \bigcup_{i=1}^{n} D_i$ of n similar tasks, we need to find a meta-function to effectively learn the parameters of the given task, namely $\phi^* = f_\theta(D, D^*)$, where θ is the meta-parameter of the meta-function $f_\theta(\cdot)$. Accordingly, the original target task can be solved as $F_{\phi^*}(\cdot)$. In fact, the representation of meta-function $f_\theta(\cdot)$ can be diverse. There are two main classs of methods: Parameterized Policy Gradient Methods (PPG) and Black Box Methods.

Parameterized Policy Gradient Methods and MAML The PPG method use an iterative algorithm to represent f_θ, which has an innerloop of the form:

$$\phi_{j+1}^* = \phi_j^* - \alpha_\theta \nabla_{\phi_j^*} \hat{L}_\theta\left(D^*, F_{\phi_j^*}\right),$$

with the initial value $\phi_0^* = \phi_0^*(\theta)$. This is very intuitive because it is similar to the traditional gradient decent method to optimize. The only difference is that θ should be carefully determined by the observations of other tasks D in this scenario. One important example of PPG method is MAML, where $\phi_0^*(\theta) = \theta$ and $\hat{L}_\theta$ can be any loss function independent of θ. MAML can be regard as learning an initial value for a task distribution, which retains the gradient decent framework of traditional optimization algorithms.

Black Box Methods Since we only care about the target function $F_{\phi}(\cdot)$ rather than the value of parameters ϕ, an alternative way to treat $f_\theta(\cdot)$ is to see it as a black box. Based on this idea, we can rewrite the target function:

$$F_{\phi}(t) = F_{f_\theta(D,D^*)}(t) = g_\theta(t, D, D^*)$$

Here we do not need to explicitly specify the value of ϕ, and treat the meta-learning task as another "big" task, which adds all the training materials into input. Because of the sequential properties of the datasets, sequential model architectures such as RNN are used in g_θ, which unavoidably change the initial architecture F_ϕ. In contrast, PPG methods use the meta-parameters θ to find the task parameters ϕ explicitly, which maintain the initial network architecture. Therefore, black box methods have inductive bias from data while PPG methods have inductive bias in network structure.

In the context of time series Implicit Neural Representation (INR), selecting an appropriate network structure is crucial, as it directly influences properties such as continuity and differentiability. Hence, in this paper, we opt for MAML as our meta-learning method for the INR task.

3 Method

Overview of MetaINR Given sparse observations of n similar time series, the goal of MetaINR is to reconstruct each sample as accurately as possible and identify the principal components (PC) of these time series. As discussed in the previous section, in traditional approaches, we first use local polynomial regression to estimate the mean and covariance of these time series. Under the assumption that scores follow a normal distribution, the PACE method can then be employed to estimate each principal component and reconstruct each sample. While this two-step estimation method is indeed superior to directly reconstructing samples using local polynomials as it leverages information from all samples, it still has the notable drawback that the assumption that scores follow a normal distribution is too restrictive. Here, we choose an alternative way by using the method of meta-learning, which involves three main steps:

- Learning the meta-model: use SIREN architecture and MAML algorithm to learn the meta-model for the INR of given time series.
- Recovery of samples: calculate the SIREN-based INR efficiently for each time series sample based on the mata-model.
- Functional data estimation: use INR of each sample to estimate the mean function, covariance kernel and principal components.

We now discuss each step.

Learning the meta-model The tasks here are realizing the implicit neural representation of samples by the SIREN architecture $F_\phi(t)$. Given the sparse observations of n similiar time series $D = \{(t_{ij}, y_{ij})_{j=1}^{J_i}\}_{i=1}^{n}$, we use MAML algorithm 1 to learning the meta-model of these tasks, whose initial value of parameters are denoted by ϕ_0. The MAML optimization process to find ϕ_0 can be expressed as follows:

Algorithm 1 Model-Agnostic Meta-Learning for Time Series Implicit Neural Representation

Input: $D = \{(t_{ij}, y_{ij})_{j=1}^{J_i}\}_{i=1}^{n}$: the sparse observations for tasks
Input: α, β : step size hyperparameters
  randomly initialize θ
  while not done **do**
    for all i **do**
      $\theta^0 \leftarrow \theta$
      for $q = 0$ to $Q-1$ **do**
        Evaluate $\nabla_{\theta^q}\mathcal{L}_i\left(F_{\theta^q}\right)$ using support set $D_i^{\text{support}} = (t_{ij}, y_{ij})_{j=1}^{\lfloor 0.8 * J_i \rfloor}$
        Compute adapted parameters with gradient descent: $\theta_i^{q+1} = \theta_i^q - \alpha \nabla_{\theta_i^q}\mathcal{L}_i\left(F_{\theta_i^q}\right)$
      end for
      Compute the loss $\mathcal{L}_i\left(F_{\theta_i^Q}\right)$ in sample i using query set $D_i^{\text{query}} = (t_{ij}, y_{ij})_{j=\lfloor 0.8 * J_i \rfloor}^{J_i}$
    end for
    Compute the total loss $\mathcal{L} = \sum_{i=1}^{n}\mathcal{L}_i\left(F_{\theta_i^Q}\right)$
    Update meta-parameters $\theta \leftarrow \theta - \beta \nabla_\theta \mathcal{L}$
  end while
  $\phi_0 \leftarrow \theta$
Output: the meta-parameters ϕ_0

Recovery of samples Based on this meta-model, we can use gradient decent algorithm2 to calculate the optimal parameters ϕ_i^* for the sample i and hence obtain the INR $F_{\phi_i^*}(t)$.

Functional data estimation Based on the estimated INR $\hat{y}_i(t) = F_{\phi_i^*}(t)$, we subsequently estimate the mean function $\hat{\mu}(t)$ and covariance kernel $\hat{c}(t, s)$ as follows:

$$\hat{\mu}(t) = \frac{1}{n}\sum_{i=1}^{n}\hat{y}_i(t)$$

Algorithm 2 Implicit Neural Representation Learning for Target Sample

Input: $D^* = (t_{ij}, y_{ij})_{j=1}^{J^*}$: the sparse observations for the target task
Input: α : step size hyperparameters
initialize $\phi \leftarrow \phi_0$
while not done **do**
Evaluate $\nabla_\phi \mathcal{L}(F_\phi)$ using support set $D^* = (t_j^*, y_j^*)_{j=1}^{J^*}$
Update parameters with gradient descent: $\phi = \phi - \alpha \nabla_\phi \mathcal{L}(F_\phi)$
end while
$\phi^* \leftarrow \phi$
Output: ϕ^*: the parameters of INR for the target model

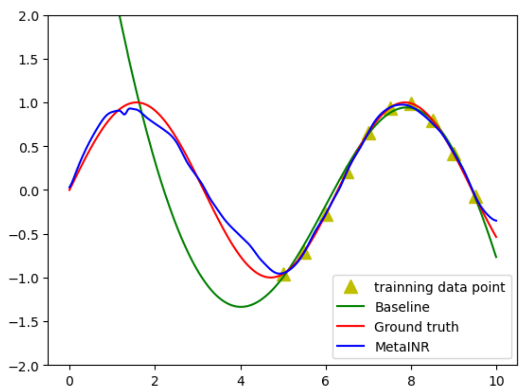

Figure 1: Learning periodicity

$$\hat{c}(t, s) = \frac{1}{n} \sum_{i=1}^{n} [\hat{y}_i(t)) - \hat{\mu}(t)][\hat{y}_i(s) - \hat{\mu}(s)]$$

According to the covariance operator, we can further estimate its principal components $\hat{\phi}_k$ by calculate the eigenfunctions of $\hat{c}(t, s)$.

In our approach, we view meta-learning not merely as an efficient learning method but as a means to learn commonalities among a large number of tasks. Leveraging this property, the metaINR $\hat{y}_i(t) = F_{\phi_i^*}(t)$ implicitly contain the information of the whole datasets. Consequently, it provides a robust estimation for sparse time series data.

4 Experiments

4.1 Synthetic data

4.1.1 what meta-learning learned in timeseries

global+differential property lead to a better PC estimation

1. periodicity

2. differential equations: No?

3. average phenomenon for meta prior model

4. based on information from whole dataset to get a estimation

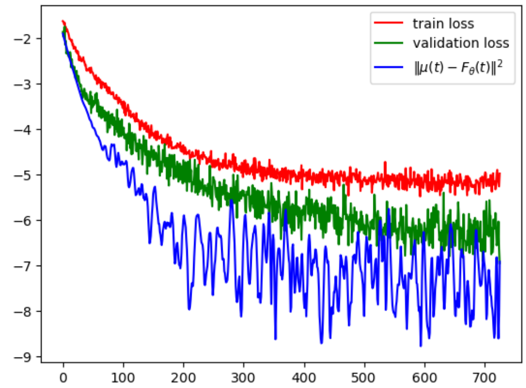

Figure 2: Learning mean

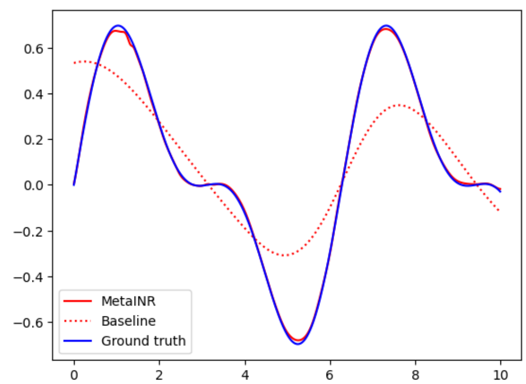

Figure 3: Mean estimation

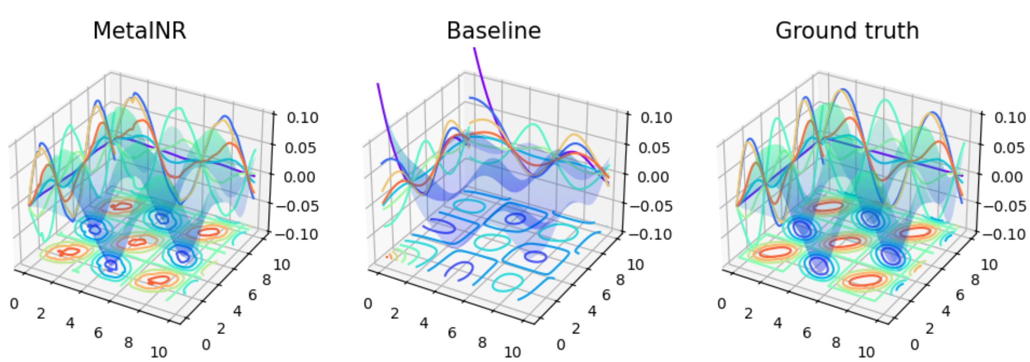

Figure 4: Covariance estimation

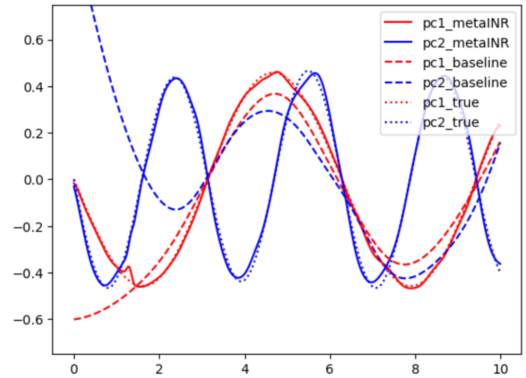

Figure 5: FPCA

4.1.2 Mean estimation

4.1.3 Covariance estimation

4.1.4 fPCA

4.2 Real-world data

5 Conclusion

6 Appendix

6.1 Traditional FDA for sparse functional data

6.2 Brief introduciton to SIREN

References

References

[1] V. Sitzmann, J. Martel, A. Bergman, D. Lindell, and G. Wetzstein, "Implicit neural representations with periodic activation functions," *Advances in neural information processing systems*, vol. 33, pp. 7462–7473, 2020.