Functional Deep Learning With Application to Forecasting Precipitation in Macau

Kuichen Shao Jiatai Wu

Department of Mathematics
Faculty of Science and Technology
University of Macau

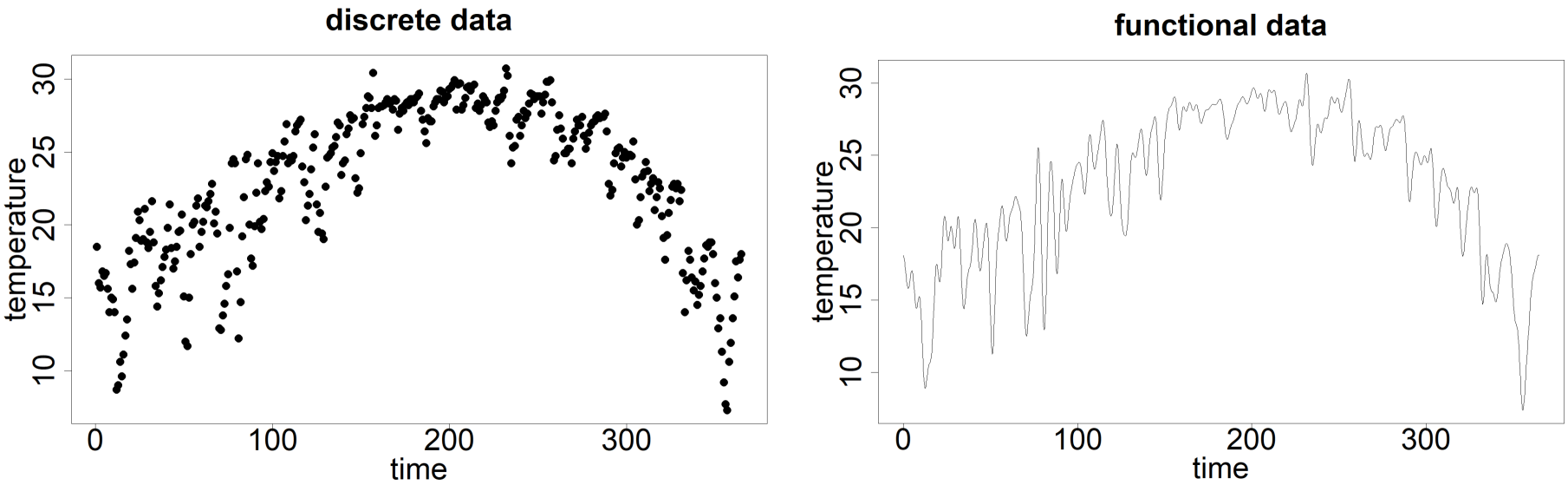
Final Year Project Presentation

May 3, 2023

Presentation Overview

What is functional data

An example on traditional discrete data and functional data:



(a) (b)

Figure: The daily average temperature in Macau (1999)

Basis and estimation

Basis function expansion:

$$x(t) = \sum_{k=1}^{K} c_k \phi_k(t) = \boldsymbol{c}' \boldsymbol{\phi}(t)$$

Type of basis function systems: constant basis, monomial basis, splines and Fourier series. The Fourier series are:

$$\begin{aligned}
\phi_1(t) &= 1 \\
\phi_2(t) &= \sin(\omega t) \\
\phi_3(t) &= \cos(\omega t) \\
\phi_4(t) &= \sin(2\omega t) \\
\phi_5(t) &= \cos(2\omega t) \\
&\vdots \\
\textit{where} \quad \omega &= 2\pi / T.
\end{aligned}$$

Basis and estimation

The loss function for coefficients:

$$F(\boldsymbol{c}) = \sum_j [y_j - x(t_j)]^2 + \lambda \int [Lx(t)]^2 dt$$
$$= \sum_j [y_j - \boldsymbol{c}'\phi(t_j)]^2 + \lambda \boldsymbol{c}'(\int [L\phi(t)L\phi'(t)]dt)\boldsymbol{c}$$

where L (linear differential operator): a linear combination of derivatives like $Lx = D^2x$ and harmonic accelerarion $L = \omega^2 D + D^3$.

The coefficients of smoothing can be estimated by least square method:

$$\hat{\boldsymbol{c}} = (\boldsymbol{\Phi}'\boldsymbol{\Phi} + \lambda R)^{-1}\boldsymbol{\Phi}' y,$$

where

$$R = \int L\phi(t)L\phi'(t)\mathrm{d}t.$$

Functional linear model

- Comparison:
  Traditional LM: $y_i = \sum_{j=0}^{p} \beta_j x_{ij} + \epsilon_i$.
  Functional LM: $y_i = \alpha_0 + \boldsymbol{z}_i' \boldsymbol{\alpha} + \sum_{j=1}^{q} \int \beta_j(t) x_{ij}(t)\, dt + \epsilon_i$.

- Expansion of functional coefficients:

$$\beta(t) = \sum_{k}^{K} c_k \phi_k(t) = \boldsymbol{c}' \boldsymbol{\phi}(t).$$

Functional linear model

- Loss function:

$$SSE\left(\alpha,\beta\right)=\sum_{i=1}^{N}\left[y_i-\boldsymbol{z}_i'\boldsymbol{\alpha}-\sum_{j=1}^{q}\int x_{ij}\left(t\right)\beta_j\left(t\right)dt\right]^2$$

$$=\sum_{i=1}^{N}\left[y_i-\boldsymbol{z}_i'\boldsymbol{\alpha}-\sum_{j=1}^{q}c_{ij}\int x_{ij}\left(t\right)\phi_j\left(t\right)dt\right]^2$$

$$\text{PENSSE}_\lambda\left(\boldsymbol{\alpha},\boldsymbol{\beta}\right)=\sum_{i=1}^{N}\left[y_i-z_i'\alpha-\sum_{j}\int x_{ij}\left(t\right)\beta_j\left(t\right)dt\right]^2$$

$$+\lambda\left(\int\left[L\beta'\left(t\right)L\beta(t)\right]dt\right).$$

Functional linear model

- Matrix form:

$$\mathbf{Z} = \begin{bmatrix} \mathbf{z_1'} & \int x_{11}(t)\,\Phi_1(t)\,dt & \dots & \int x_{1q}(t)\,\Phi_q(t)\,dt \\ \vdots & \ddots & & \vdots \\ \mathbf{z_n'} & \int x_{n1}(t)\,\Phi_1(t)\,dt & \dots & \int x_{nq}(t)\,\Phi_q(t)\,dt \end{bmatrix},$$

$$\boldsymbol{b} = \begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{c} \end{bmatrix}$$

$$\boldsymbol{y} = \boldsymbol{Z}\boldsymbol{b} + \epsilon$$

- Estimation of coefficient:

$$\hat{\boldsymbol{b}} = \left(\boldsymbol{Z}'\boldsymbol{Z}\right)^{-1}\boldsymbol{Z}'\boldsymbol{y}$$

$$\hat{\boldsymbol{b}} = \left(\boldsymbol{Z}'\boldsymbol{Z} + \boldsymbol{R}(\lambda)\right)^{-1}\boldsymbol{Z}'\boldsymbol{y}$$

Functional Principal Component Analysis

	Multivariate PCA	Functional PCA
Maximization	$Var(a^T X)$	$Var(\int \xi(t) X_i(t) dt)$
Subjection1	$\|\|\boldsymbol{a}\|\| = 1$	$\int \xi(t)^2 \mathrm{d}t = 1$
Subjection2	$\boldsymbol{a}_j^T \boldsymbol{a}_k = 0, k < j$	$\int \xi_j(t)\xi_k(t)\mathrm{d}t = 0, k < j$
Eigenvalue/Eigenvector	$\Sigma \boldsymbol{a}_j = \mu_j \boldsymbol{a}_j$	$\int v(s,t)\xi_j(t)dt = \mu_j \xi_j(s)$
Score	$c_{ij} = a_j^T x_i$	$c_{ij} = \int \xi_j(t) x_i(t) dt$
FVE	$\frac{\sum_{j=1}^q \mu_j}{\sum \mu_j}$	$\frac{\sum_{j=1}^q \mu_j}{\sum \mu_j}$

Table: Comparison between multivariate PCA and functional PCA

Note: $v(s,t) = \frac{1}{N-1} \sum_i [x_i(s) - \bar{x}(s)][x_i(t) - \bar{x}(t)]$

Functional Score Regression

Multivariate PCA: dimension reduction among different variables.

Functional PCA: dimension reducntion within one variable.

Functional Score Regression: combination of multivariate PCA and Functional Linear Model

1. PCA in multivariate data perspective.
2. Smooth their principal component scores and get functional score.
3. Replace functional scores to original functional variables in the FLM.

Artificial Neural Network

Without taking functional data into consideration

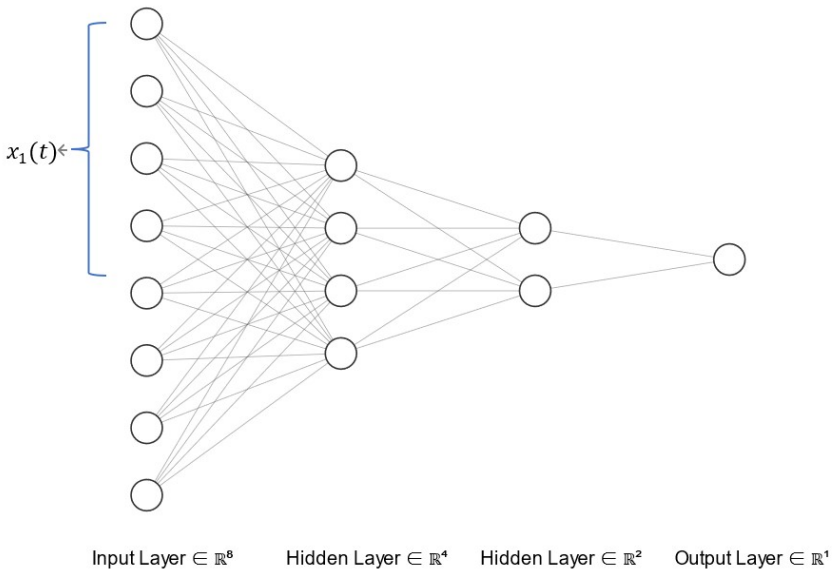


Figure: ANN

- Advantage:
  Easy to understand and implement
- Disadvantage:
  1. Too many parameters.
  2. Lack of interpretation for weight function.

Functional Neural Network

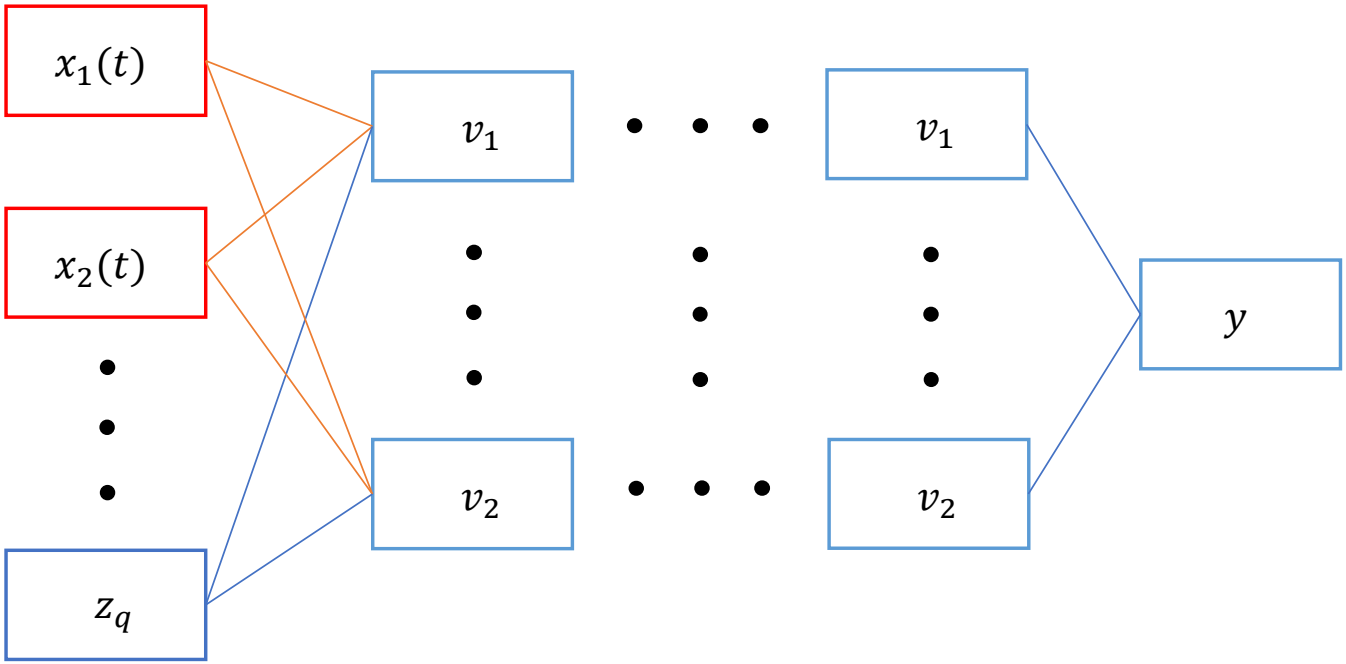


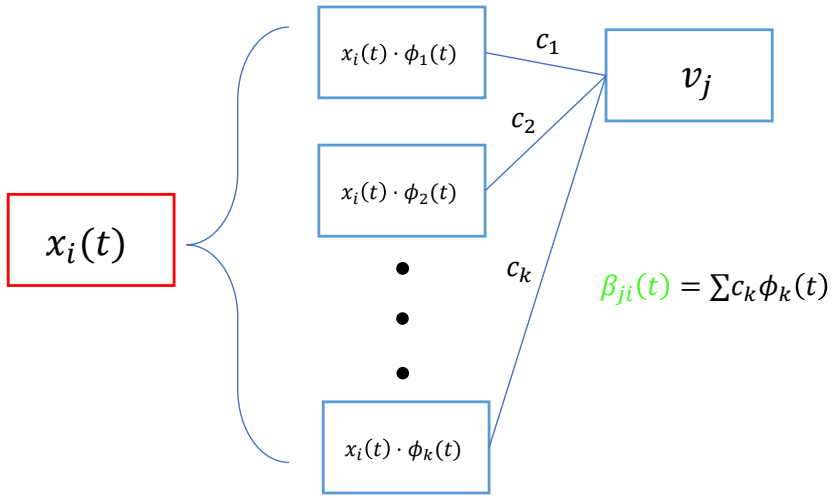
Figure: FuncNN

Introduce functional data into neural network accroding to FuncNN (Thind et al. ,2022). They do some manipulations in the first layer of network and then follow the same rules of traditional network. Formulate neuron v as:

$$v_i^{(1)} = g\Big(\sum_{k=1}^{K}\int_{\mathcal{T}}\beta_{ik}(t)x_k(t)\mathrm{d}t + \sum_{j=1}^{J} w_{ij}^{(1)}z_j + b_i^{(1)}\Big)$$

Functional Neural Network

Details about the first layer and weight function are shown in graph:



Weight function can be expressed by basis function expansion:

$$\beta_{ji}(t) = \sum_{k=1}^{K} c_{ijk}\phi_{ijk}(t)$$

Then neuron v is formulated as:

$$v_i^{(1)} = g\Big(\sum_{j=1}^{J}\sum_{k=1}^{K}\int_{\mathcal{T}} \phi_{ijk}(t)x_j(t)\mathrm{d}t + \sum_{p=1}^{P} w_{ip}^{(1)}z_p + b_i^{(1)}\Big)$$

Data description

- Source: Macau Meteorological and Geophysical Bureau
- Data details: daily average temperature, daily average relative humidity, daily average wind speed, daily average sea level pressure, daily total sunshine duration, and daily average precipitation
- Variable: response and covariates
  - Scalar Response: precipitation
  - Functional Covariates: temperature, humidity, pressure, sunshine, windspeed
- Time interval: 1999.01.01-2022.12.31.
  24 years as 24 functional observations with 365 time points.
  5 functional covarites and 1 scalar response.

Choosing number of basis functions

We choose Fourier Series as bases and the number of basis funtions is determined by AIC + BIC, take covariate temperature as example:

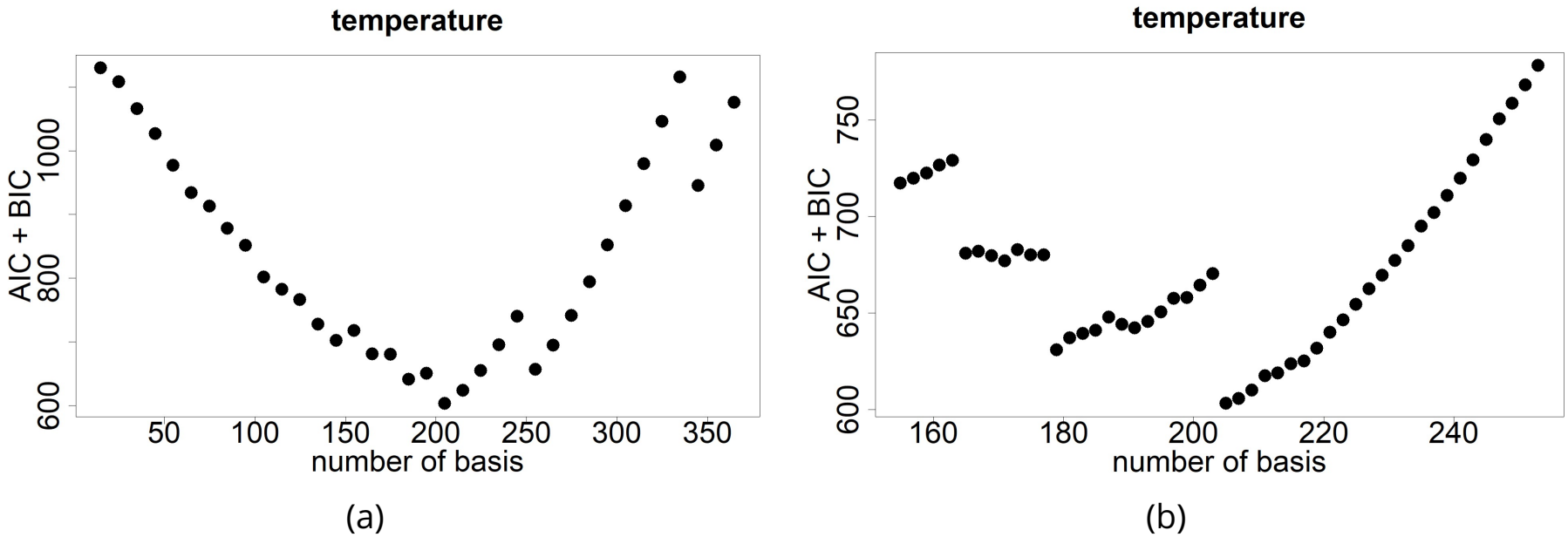


Figure: The value of AIC + BIC corresponding to each $log_{10}(\lambda)$ for Temperature

Functional Linear Model (FLM)

After we smooth the data, we plug in functional data obtained into 4 models discussed before, implememnt the models with more details.

For functional linear model:

- Due to sample size, we can only take 3 basis functions for each covariate, otherwise $Z'Z$ is a singular matrix, i.e. not invertible. And the results is terrible.

- So it is necessary to add a penalty term, then $Z'Z + R(\lambda)$ is invertible for more than 3 basis functions.

Functional Score Regression (FSR)

We take PCA of five covariates in 24 years and the total variance explained by different PCs are shown in picture.

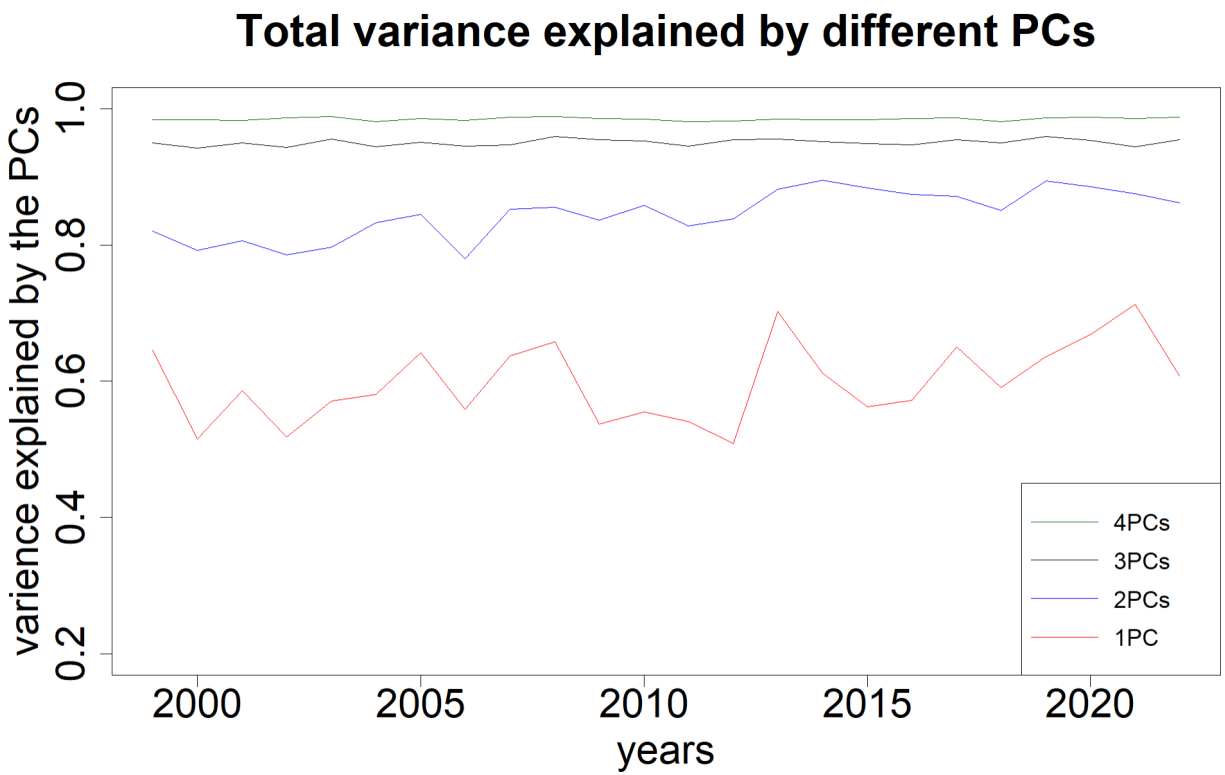


Figure: FVE of PCs

We choose the first three functional scores as our covariates.

Functional Neural Network (FuncNN)

Hyperparameters of FuncNN

- number of layers
- number of neurons per layer
- activation function: ReLU, sigmoid, linear, LeakyReLU, tanh
- method to avoid overfitting: dropout and early stop

Measurement techniques

Techniqes we use to measure the models:

- Training set and test set split
- $MSPE_{CV} = \sum_{b=1}^{B} \sum_{I \in S_b} \frac{\left(\hat{y}^{(-b)} - y_I\right)^2}{N}$, S_b is the b-th partition of the training data set and $\hat{y}^{(-b)}, I \in S_b$ is the prediction in the set. Here we set b as 5, namely we use five-fold cross-validation.
- $R^2 = 1 - \sum_{I=1}^{N} (y_I - \hat{y}_I)^2 / \sum_{I=1}^{N} (y_I - \bar{y})^2$.

Outcomes of different models

For the current year regression problem:

	Training *MSE*	*MSPE*$_{cv}$	*Test* *MSE*	*Test* R^2
FLM without penalty	0.20	—	4.07	-1.83
FLM with penalty	1.02	1.92	1.43	0.003
FSR	0.65	1.28	0.95	0.34
FuncNN	0.33	1.50	0.70	0.51

Table: Result for the current year regression problem

For the next year's prediction problem (autoregression):

	Training *MSE*	*MSPE*$_{cv}$	*Test* *MSE*	*Test* R^2
FLM without penalty	—	—	—	—
FLM with penalty	1.09	0.89	1.73	-0.20
FSR	1.02	1.32	1.28	0.11
FuncNN	0.79	1.48	1.05	0.27

Table: Result for the next year prediction problem

Conclusion and further work

- Conclusion: FuncNN performs well and the reason may be the relation between response and covariates is not linear

- Further work:
  - Few criteria to decide number of basis functions in the process of smoothing
  - Few methods to deal with the problem of multicollinearity
  - Few formal significance test of functional coefficients
  - FuncNN deals with functional data only in the first layer.

References

1. Cuesta-Albertos, J. A., García-Portugués, E., Febrero-Bande, M., & González-Manteiga, W. (2019). Goodness-of-fit tests for the functional linear model based on randomly projected empirical processes. The Annals of Statistics, 47(1), 439-467.
2. Ferraty F, Vieu P (2006). Nonparametric Functional Data Analysis. Springer-Verlag, New York.
3. Heinrichs, F., Heim, M., & Weber, C. (2023). Functional Neural Networks: Shift invariant models for functional data with applications to EEG classification. arXiv preprint arXiv:2301.05869.
4. Henderson, B. (2006). Exploring between site differences in water quality trends: a functional data analysis approach. Environmetrics: The official journal of the International Environmetrics Society, 17(1), 65-80.
5. Martínez, J., Saavedra, Á., García-Nieto, P. J., Piñeiro, J. I., Iglesias, C., Taboada, J., ...& Pastor, J. (2014). Air quality parameters outliers detection using functional data analysis in the Langreo urban area (Northern Spain). Applied Mathematics and Computation, 241, 1-10.
6. Perdices, D., de Vergara, J. E. L., & Ramos, J. (2021). Deep-FDA: Using functional data analysis and neural networks to characterize network services time series. IEEE Transactions on Network and Service Management, 18(1), 986-999.
7. Preda, C., Saporta, G., and Lévéder, C. (2007), "PLS Classification of Functional Data," Computational Statistics, 22, 223–235.
8. Ramsay, J. O., Hooker, G., & Graves, S. (2009). Functional data analysis with R and MATLAB (Vol. 43). Springer Science & Business Media.
9. Rumelhart, D., Hinton, G., and Williams, R. (1985), "Learning Internal Representations by Error Propagation," Technical Report, California Univ San Diego La Jolla Inst for Cognitive Science.
10. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research, 15(1), 1929-1958.
11. Thind, B., Multani, K., & Cao, J. (2022). Deep Learning With Functional Inputs. Journal of Computational and Graphical Statistics, 00(0), 1-10.
12. Wang, J.-L., Chiou, J.-M., & Muller, H.-G. (2015). Functional data analysis. Annual Review of Statistics and Its Application, 2(1), 1–41.
13. Yao, J., Mueller, J., & Wang, J. L. (2021, July). Deep learning for functional data analysis with adaptive basis layers. In International Conference on Machine Learning (pp. 11898-11908). PMLR.
14. Yao, Y., Rosasco, L., & Caponnetto, A. (2007). On early stopping in gradient descent learning. Constructive Approximation, 26(2), 289-315.

Thanks for Listening

Linear Differential Operater

$Lx(t)$ is a linear differential operator on $x(t)$. It can be defined as:

$$Lx(t) = \beta_0(t)x(t) + \beta_1(t)Dx(t) + \cdots + \beta_{m-1}(t)D_{m-1}x(t) + D_m x(t)$$

We choose $L = \omega^2 D + D^3$ since we are working on a group of periodical data with a known period, and this penalty can substantially penalize the high-order terms in the Fourier Series. We can see for any periodical data $a_j \sin(j\omega t) + b_j \cos(j\omega t)$ with known frequency ω, its linear differential operator:

$$L\left[a_j \sin\left(j\omega t\right) + b_j \cos\left(j\omega t\right)\right] = \omega^2 j\left(1 - j^2\right)\left[a_j \cos\left(j\omega t\right) - b_j \sin\left(j\omega t\right)\right].$$

We observe that when j = 1, there is no penalization. As j increase, the increase of penalty is proportional to $j^2(1 - j^2)^2$ which is very large. Thus, the penalty constricts the form into $a\sin(\omega t) + b\cos(\omega t)$.

GCV

Generalized Cross Validation (GCV):

$$GCV(\lambda) = \left(\frac{n}{n - df(\lambda)}\right)\left(\frac{SSE}{n - df(\lambda)}\right).$$

Let

$$\boldsymbol{R} = \int L\phi(t)\, L\phi'(t)\, dt,$$

If we design the hat matrix as:

$$\boldsymbol{H} = \boldsymbol{\Phi}\left(\boldsymbol{\Phi}'\boldsymbol{\Phi} + \lambda \boldsymbol{R}\right)^{-1}\boldsymbol{\Phi}',$$

then the degree of freedom as a function of λ can be calculated by:

$$df(\lambda) = \text{trace}\left[\boldsymbol{H}(\lambda)\right].$$

Penalty matrix R for functional linear model:

$$\begin{bmatrix} \lambda_0 I & \dots & \dots & \dots \\ 0 & \lambda_1 R_1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda_q R_q \end{bmatrix}$$

Functional Response for FLM

The functional linear model can also deal with the functional response using a concurrent model that is estimating functional response by the covariates at the same time points. The model can be described as:

$$\boldsymbol{y}(t) = \boldsymbol{Z}(t)\beta(t) + \epsilon(t),$$

where $\boldsymbol{y}(t)$ is a functional vector with length N containing N functional response and $\boldsymbol{Z}(t)$ is a functional covariate matrix with dimension $N \times p$. The loss function is similar but only replaces the square term with the inner product on the vector. Then solve the regression by normal equations:

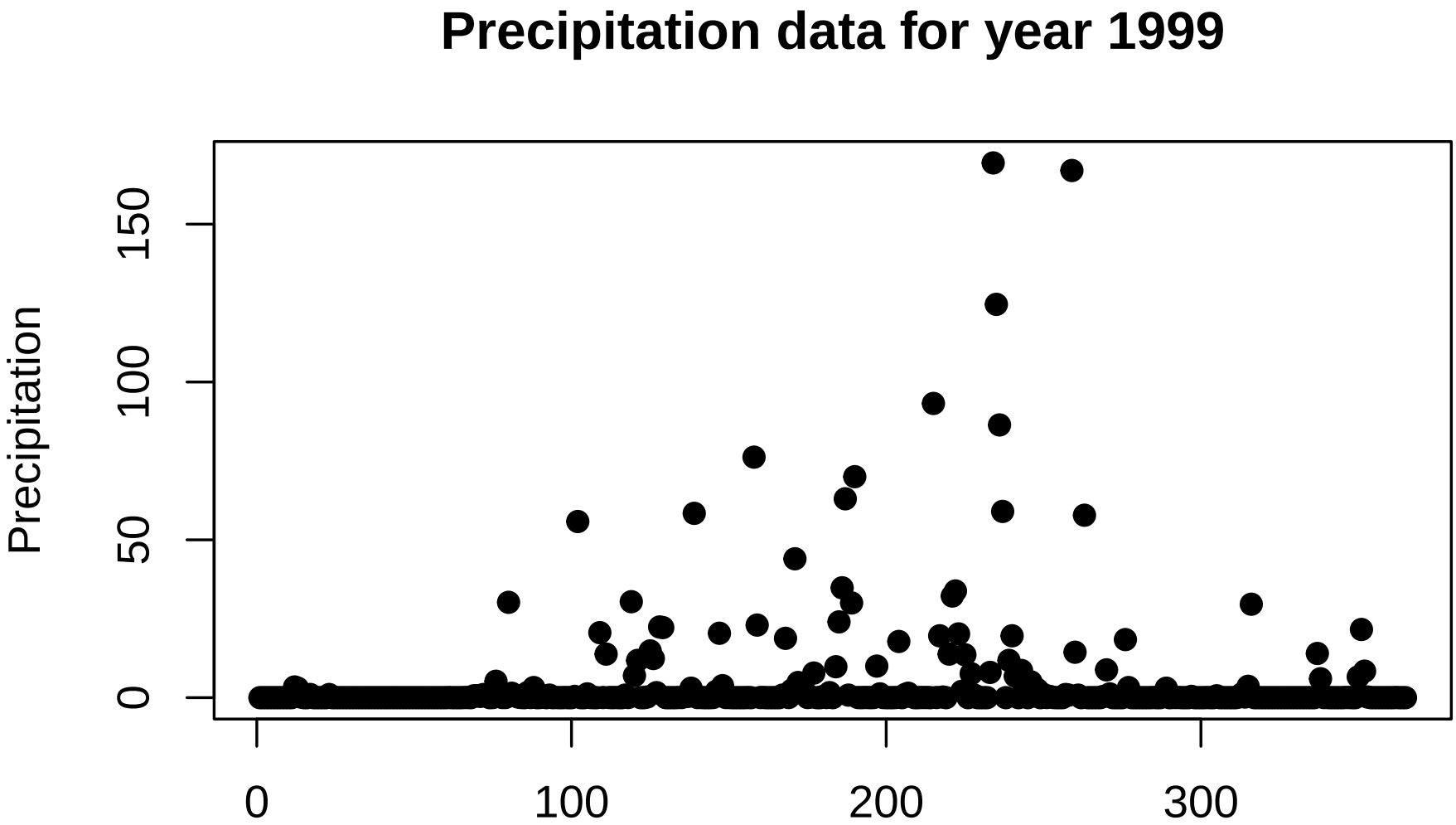
$$\left[\int \boldsymbol{\Theta}'(t)\boldsymbol{Z}'(t)\boldsymbol{Z}(t)\boldsymbol{\Theta}(t)\,dt + \boldsymbol{R}(\lambda)\right]^2 \hat{\boldsymbol{b}} = \left[\int \boldsymbol{\Theta}'(t)\boldsymbol{Z}'(t)\boldsymbol{y}(t)\,dt\right].$$

where $\boldsymbol{\Theta}(t)$ is the basis function matrix of functional weight $\beta(t)$, and $\hat{\boldsymbol{b}}$ is the corresponding coefficient scalar vector.

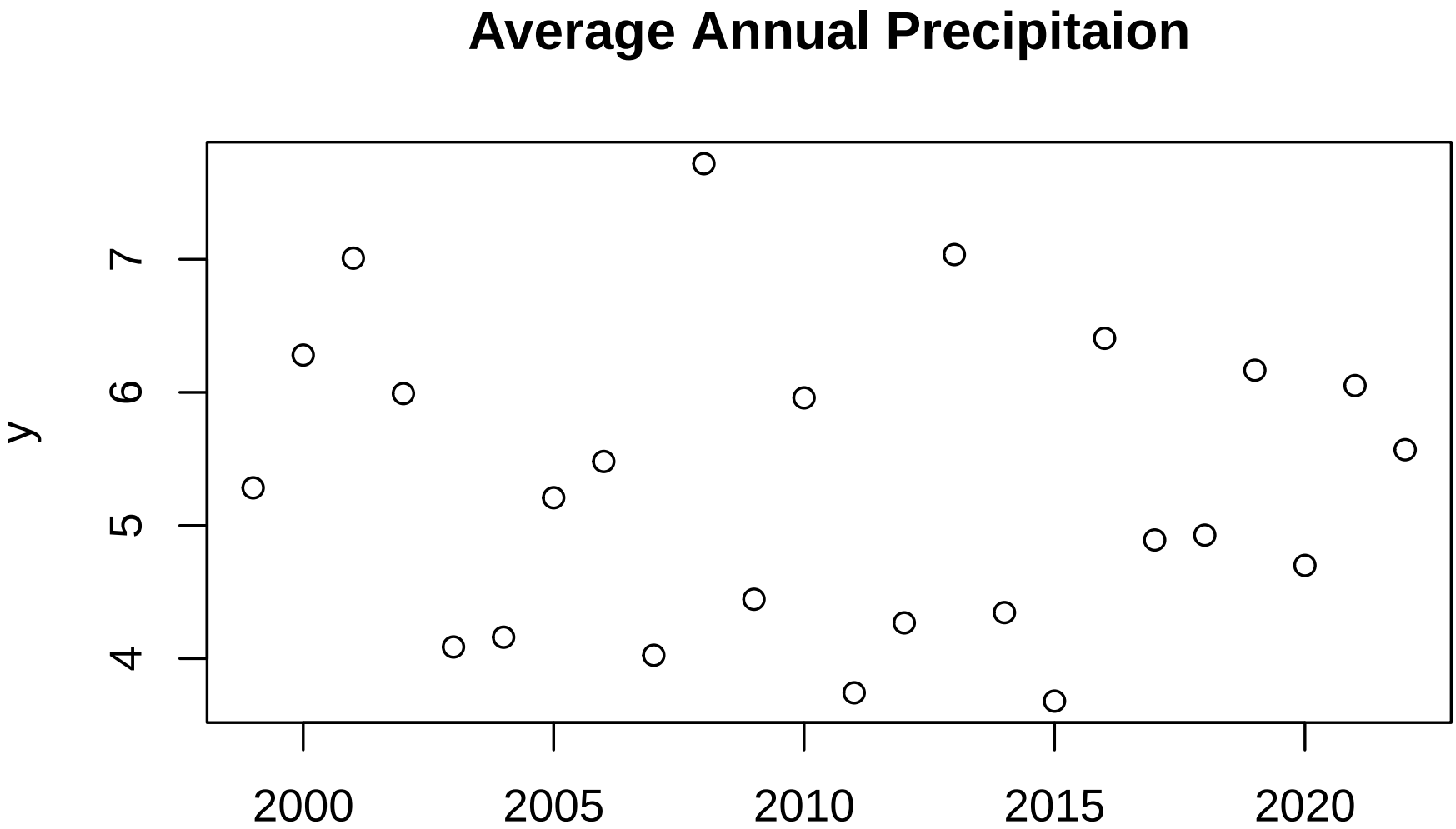
Functional Response for FuncNN

FuncNN can also deal with the functional response and its method is quite straightforward if we know how to deal with functional covariates. It still expresses weight functions to functional response as basis functions expansion, takes the inner product of response and basis functions as output neurons, trains the network, and finally output coefficients indicating the weight functions.

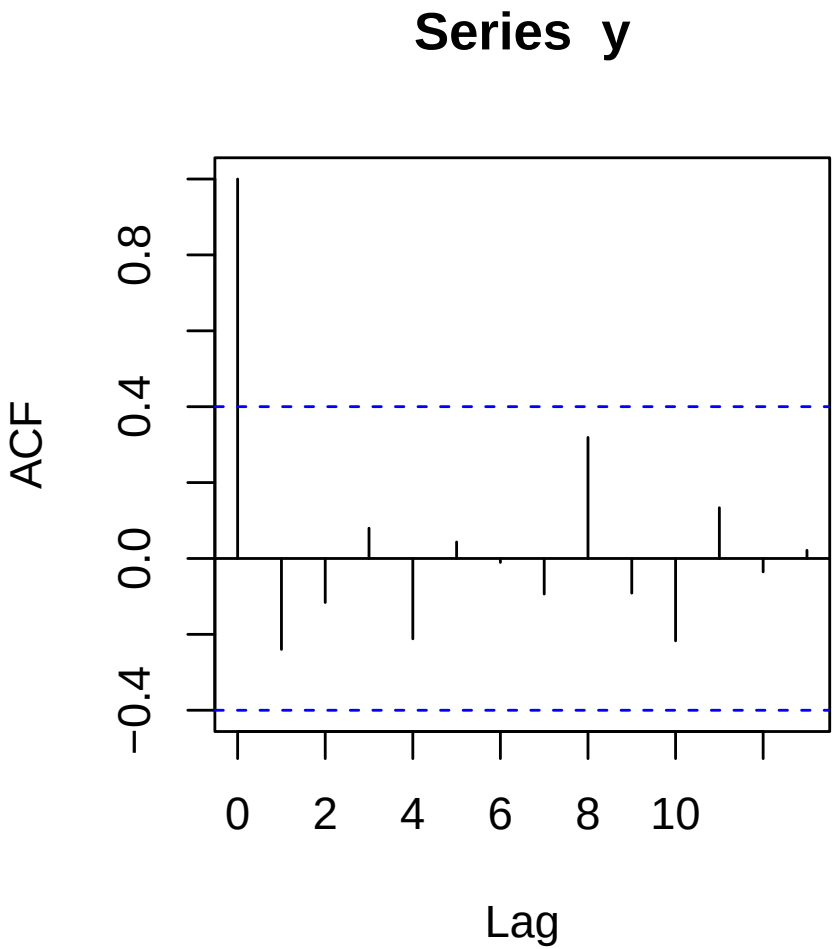
Precipitation data

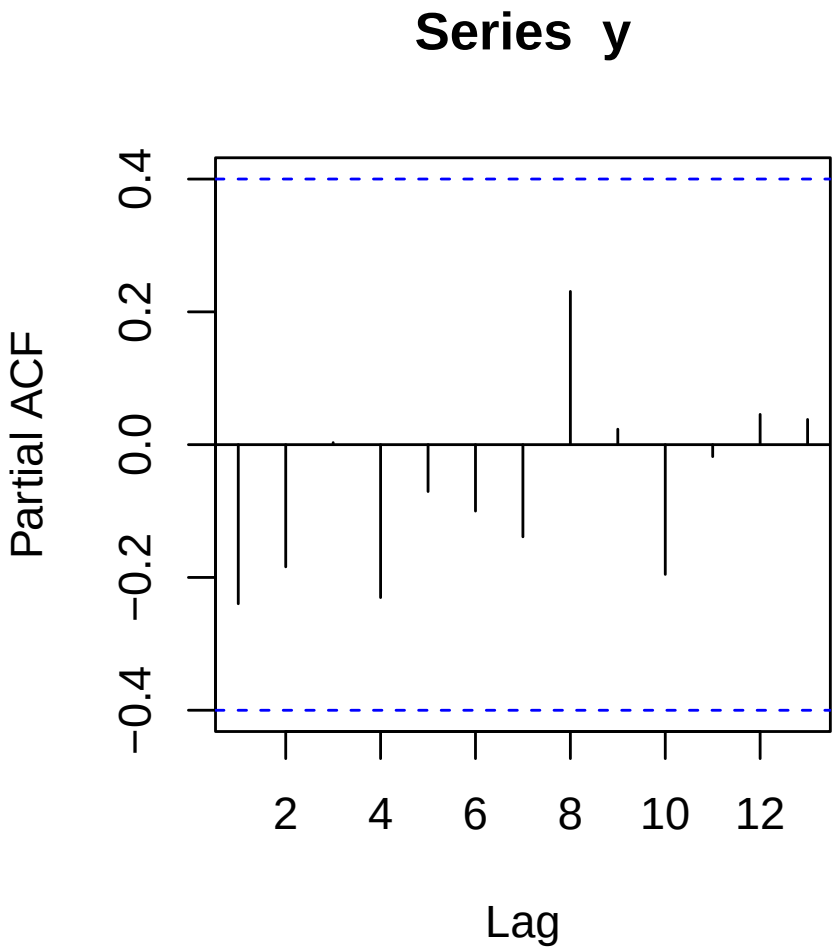


Precipitation data



Precipitation data





Drop out method I

Dropout addresses overfitting by randomly ”dropping out” (setting to zero) some neurons during training. This forces the network to learn more robust features, as no single neuron can rely on the presence of another neuron to make predictions. During training, each neuron in the network has a probability p of being ”dropped out.” This means that the neuron’s output is set to zero with probability p, or left unchanged with probability $1 - p$. The value of p is typically set between 0.1 and 0.5.
To implement this method, we generate a Bernoulli distribution with probability p to get a set of (0,1) samples, the number of samples equals to the number of neurons in a hidden layer. If the neuron corresponds to 0, drop it, else remain that neural. The following 2 graphs show how dropout was implemented in a one-hidden layer neural network.

Drop out method II

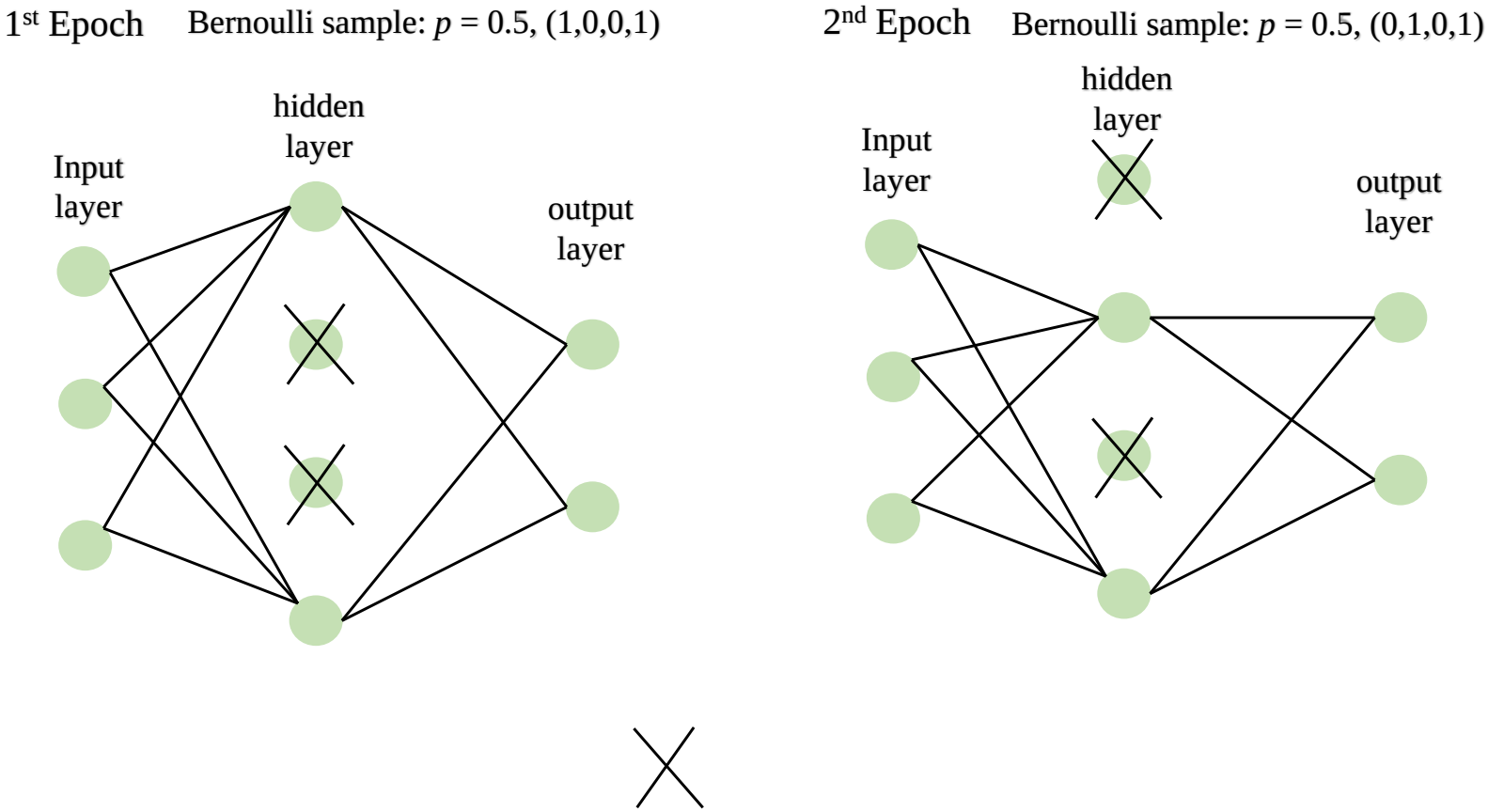


Figure: The principle of how dropout works

Early stop method I

The idea behind early stopping is to monitor the network's performance on a validation set during training and stop the training process once the validation error starts to increase, indicating that the network is starting to overfit the training data.
In the beginning, data set is separated into a training set and a validation set randomly. As the network is trained, the training error generally decreases over time, while the validation error initially decreases but then starts to increase again. This is because the network starts to overfit the training data, and the improvements in the training error do not generalize to new data.

Early stop method II

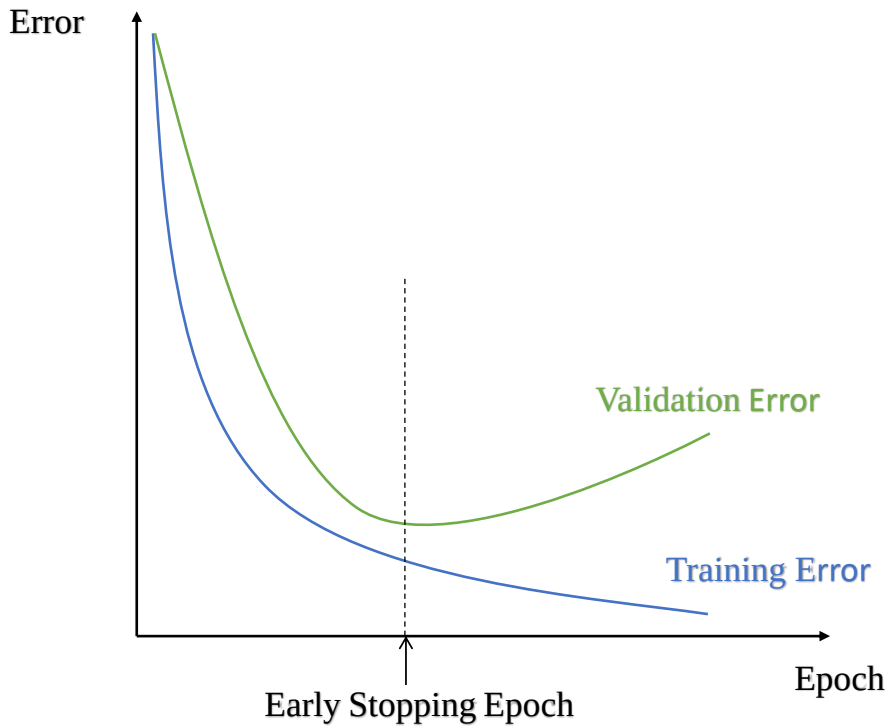


Figure: The principle of how early stop works

Early stop method III

The early stopping point is indicated by the dashed line in the graph. This is the point at which the validation error is at its minimum. At this point, the network has achieved the best possible generalization performance on the validation set. Any further training would cause the network to overfit the training data, resulting in worse generalization performance. Therefore, the training process is stopped at the early stopping point, and the network parameters at this point are used as the final trained model. This model has achieved the best possible generalization performance on the validation set and can be used to make predictions on new data.