## Time Series Encoding (encode)

We use a sliding-window embedding to convert a univariate series into fixed-length vectors of length p. A feed-forward autoencoder is trained to minimize reconstruction error, and its bottleneck (k < p) provides a compact encoding that preserves salient information for downstream tasks.

This example shows how to transform a time series into fixed-size windows and train an autoencoder to learn a compact latent representation (p -> k) of these windows.

Prerequisites - R packages: daltoolbox, ggplot2 - Python with PyTorch accessible via reticulate (the backend is loaded by internal functions)

# Loading required packages  
library(daltoolbox)

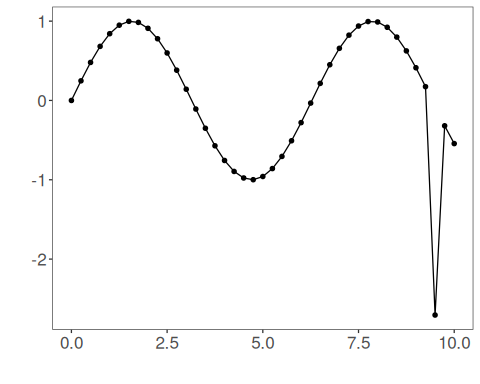
Series for study

data(tsd)  
tsd$y[39] <- tsd$y[39] \* 6 # inject a synthetic outlier for illustration in the plot

sw\_size <- 5 # sliding window size (p)  
ts <- ts\_data(tsd$y, sw\_size) # convert the series into windows with p columns  
ts\_head(ts, 3) # view the first 3 windows

## t4 t3 t2 t1 t0  
## [1,] 0.0000000 0.2474040 0.4794255 0.6816388 0.8414710  
## [2,] 0.2474040 0.4794255 0.6816388 0.8414710 0.9489846  
## [3,] 0.4794255 0.6816388 0.8414710 0.9489846 0.9974950

library(ggplot2)  
plot\_ts(x = tsd$x, y = tsd$y) + # series plot with the outlier peak  
 theme(text = element\_text(size = 16))



Data sampling

samp <- ts\_sample(ts, test\_size = 5) # hold out the last 5 windows for test  
train <- as.data.frame(samp$train)  
test <- as.data.frame(samp$test)

Train the model

auto <- autoenc\_e(5, 3) # reduce from 5 -> 3 dimensions (p -> k)  
auto <- fit(auto, train)

Encoding evaluation (train)

print(head(train)) # original windows (p columns)

## t4 t3 t2 t1 t0  
## 1 0.0000000 0.2474040 0.4794255 0.6816388 0.8414710  
## 2 0.2474040 0.4794255 0.6816388 0.8414710 0.9489846  
## 3 0.4794255 0.6816388 0.8414710 0.9489846 0.9974950  
## 4 0.6816388 0.8414710 0.9489846 0.9974950 0.9839859  
## 5 0.8414710 0.9489846 0.9974950 0.9839859 0.9092974  
## 6 0.9489846 0.9974950 0.9839859 0.9092974 0.7780732

result <- transform(auto, train) # encodings (k columns)  
print(head(result))

## [,1] [,2] [,3]  
## [1,] -0.02174866 -0.1305884 -0.7997297  
## [2,] 0.18436751 -0.3574348 -0.9403684  
## [3,] 0.39086729 -0.5806823 -1.0211574  
## [4,] 0.56393665 -0.7755992 -1.0237963  
## [5,] 0.68807191 -0.9300846 -0.9555943  
## [6,] 0.74946082 -1.0291867 -0.8240919

Encoding of the test set

print(head(test))

## t4 t3 t2 t1 t0  
## 1 0.9893582 0.9226042 0.7984871 0.6247240 0.4121185  
## 2 0.9226042 0.7984871 0.6247240 0.4121185 0.1738895  
## 3 0.7984871 0.6247240 0.4121185 0.1738895 -2.7054403  
## 4 0.6247240 0.4121185 0.1738895 -2.7054403 -0.3195192  
## 5 0.4121185 0.1738895 -2.7054403 -0.3195192 -0.5440211

result <- transform(auto, test)  
print(head(result))

## [,1] [,2] [,3]  
## [1,] 0.6982391 -1.0565972 -0.4407871  
## [2,] 0.5905741 -1.0150455 -0.1660231  
## [3,] 0.2947428 -1.8688312 1.7810478  
## [4,] -0.6640453 0.1703665 1.7040564  
## [5,] -0.6266151 -0.1774592 1.3481926

References - Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press. (Chapter on Autoencoders)