## Variational Autoencoder (encode)

This example uses a Variational Autoencoder (VAE) to learn latent representations of time-series windows. The VAE reduces from p to k dimensions and regularizes the latent space to approximate a target distribution (e.g., standard normal) via a KL term.

Prerequisites - Python with PyTorch accessible via reticulate - R packages: daltoolbox, tspredit, daltoolboxdp, ggplot2

Quick notes - Loss: reconstruction + KL divergence between the latent distribution and the prior. - Useful for generating continuous, well-behaved representations in latent space.

# Installing example dependencies (if needed)  
#install.packages("tspredit")  
#install.packages("daltoolboxdp")

# Loading required packages  
library(daltoolbox)  
library(tspredit)  
library(daltoolboxdp)  
library(ggplot2)

# Example dataset (series -> windows)  
data(tsd)  
  
sw\_size <- 5 # sliding window size (p)  
ts <- ts\_data(tsd$y, sw\_size) # convert series into windows with p columns  
  
ts\_head(ts)

## 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

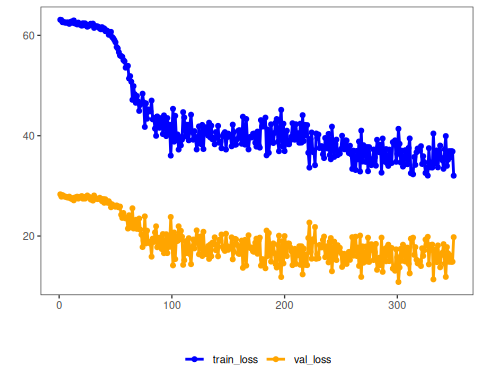
# Normalization (min-max by group)  
preproc <- ts\_norm\_gminmax()  
preproc <- fit(preproc, ts)  
ts <- transform(preproc, ts)  
  
ts\_head(ts)

## t4 t3 t2 t1 t0  
## [1,] 0.5004502 0.6243512 0.7405486 0.8418178 0.9218625  
## [2,] 0.6243512 0.7405486 0.8418178 0.9218625 0.9757058  
## [3,] 0.7405486 0.8418178 0.9218625 0.9757058 1.0000000  
## [4,] 0.8418178 0.9218625 0.9757058 1.0000000 0.9932346  
## [5,] 0.9218625 0.9757058 1.0000000 0.9932346 0.9558303  
## [6,] 0.9757058 1.0000000 0.9932346 0.9558303 0.8901126

# Train/test split  
samp <- ts\_sample(ts, test\_size = 10)  
train <- as.data.frame(samp$train)  
test <- as.data.frame(samp$test)

# Creating the VAE: reduce from 5 -> 3 dimensions (p -> k)  
# - num\_epochs: fewer epochs may suffice given the additional KL term  
auto <- autoenc\_variational\_e(5, 3, num\_epochs = 350)  
  
# Training the model  
auto <- fit(auto, train)

# Learning curves (total loss per epoch)  
fit\_loss <- data.frame(  
 x = 1:length(auto$train\_loss),  
 train\_loss = auto$train\_loss,  
 val\_loss = auto$val\_loss  
)  
grf <- plot\_series(fit\_loss, colors = c('Blue', 'Orange'))  
plot(grf)



# Testing the VAE (encoding)  
# Show samples from the test set and the encoding (k columns)  
print(head(test))

## t4 t3 t2 t1 t0  
## 1 0.7258342 0.8294719 0.9126527 0.9702046 0.9985496  
## 2 0.8294719 0.9126527 0.9702046 0.9985496 0.9959251  
## 3 0.9126527 0.9702046 0.9985496 0.9959251 0.9624944  
## 4 0.9702046 0.9985496 0.9959251 0.9624944 0.9003360  
## 5 0.9985496 0.9959251 0.9624944 0.9003360 0.8133146  
## 6 0.9959251 0.9624944 0.9003360 0.8133146 0.7068409

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

## [,1] [,2] [,3] [,4] [,5] [,6]  
## [1,] -0.05441125 0.2229130 -0.06870109 -0.01014933 0.0023267455 0.001223829  
## [2,] -0.01811695 0.2521625 -0.02667102 -0.01229730 -0.0014747716 0.003800359  
## [3,] 0.03062835 0.2499319 0.02800947 -0.01343963 -0.0029293373 0.004565883  
## [4,] 0.07587761 0.2209470 0.07767614 -0.01319459 -0.0036799908 0.004955817  
## [5,] 0.10965651 0.1711112 0.11656408 -0.01234447 -0.0028897002 0.004129704  
## [6,] 0.12758264 0.1116226 0.13405931 -0.01322078 0.0004401989 -0.001453571