Spiral Matching Example Using NNS Clustering

Install NNS

We need the latest version of NNS available on GitHub.

```
require(devtools); install_github('OVVO-Financial/NNS',ref = "NNS-Beta-Version")
require(NNS)
```

```
## Warning: package 'rgl' was built under R version 3.3.3
```

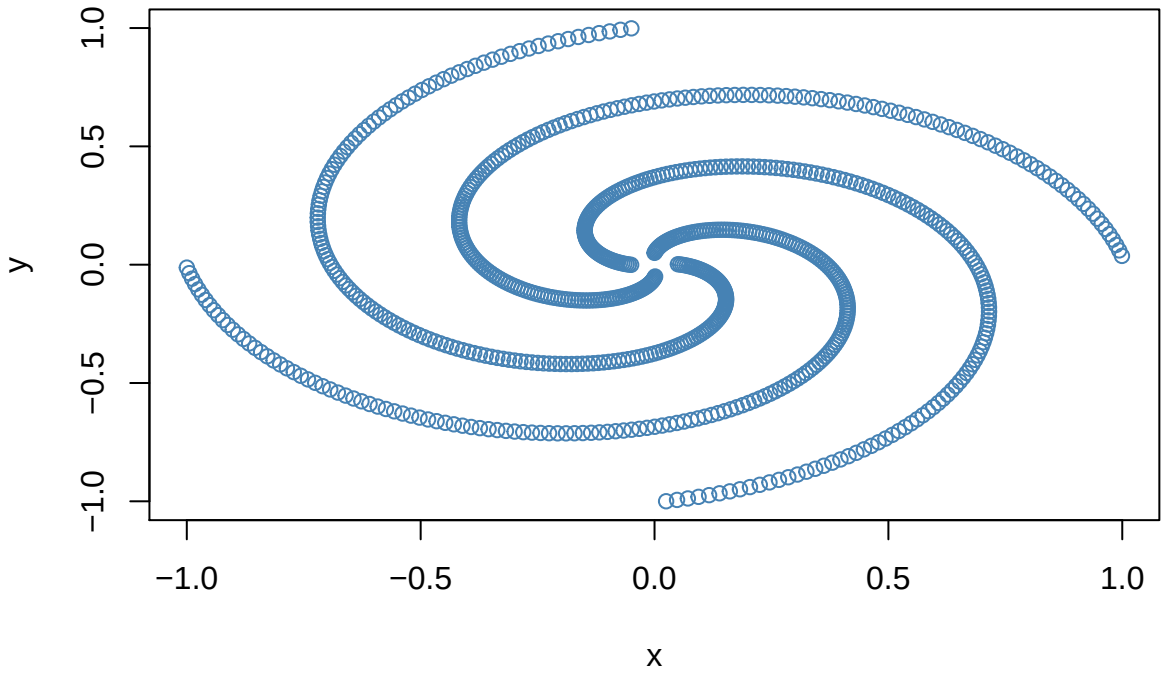
```
## Warning: package 'stringr' was built under R version 3.3.3
```

Data

Let's create a spiral dataset.

```
N <- 200 # number of points per class
D <- 2 # dimensionality
K <- 4 # number of classes
X <- data.frame() # data matrix (each row = single example)
y <- data.frame() # class labels
set.seed(123)
for (j in (1:K)){
  r <- seq(0.05,1,length.out = N) # radius
  t <- seq((j-1)*4.7,j*4.7, length.out = N)# theta
  Xtemp <- data.frame(x =r*sin(t) , y = r*cos(t))
  ytemp <- data.frame(matrix(j, N, 1))
  X <- rbind(X, Xtemp)
  y <- rbind(y, ytemp)}

data <- cbind(X,y)
colnames(data) <- c(colnames(X), 'label')
plot(data[,1],data[,2],col='steelblue',xlab='x',ylab='y')
```



More Data

Next we will create two more spirals. The second spiral will have a noise term added to the first spiral. The third spiral will have a different seed and the same noise term added.

```
X <- data.frame() # data matrix (each row = single example)
y <- data.frame() # class labels
set.seed(123)
for (j in (1:K)){
  r <- seq(0.05,1,length.out = N) # radius
  t <- seq((j-1)*4.7,j*4.7, length.out = N) + rnorm(N, sd = 0.1) # theta
  Xtemp <- data.frame(x =r*sin(t) , y = r*cos(t))
  ytemp <- data.frame(matrix(j, N, 1))
  X <- rbind(X, Xtemp)
  y <- rbind(y, ytemp)}

data2 <- cbind(X,y)
colnames(data2) <- c(colnames(X), 'label')

X <- data.frame() # data matrix (each row = single example)
y <- data.frame() # class labels
set.seed(1234)
for (j in (1:K)){
  r <- seq(0.05,1,length.out = N) # radius
  t <- seq((j-1)*4.7,j*4.7, length.out = N) + rnorm(N, sd = 0.1) # theta
```

```
  Xtemp <- data.frame(x =r*sin(t) , y = r*cos(t))
  ytemp <- data.frame(matrix(j, N, 1))
  X <- rbind(X, Xtemp)
  y <- rbind(y, ytemp)}

data3 <- cbind(X,y)
colnames(data3) <- c(colnames(X), 'label')
```

Simple Correlation Analysis

Does a simple correlation distinguish these spirals from one another? No, unfortunately all of the x's are correlated and all of the y's are correlated.

```
options(digits=4)
cor(cbind(data$x,data2$x,data3$x,data$y,data2$y,data3$y))
```

```
##            [,1]      [,2]      [,3]      [,4]      [,5]       [,6]
## [1,]  1.0000000  0.995402  0.994848 -0.000668 -0.003216 -0.0004052
## [2,]  0.9954025  1.000000  0.991165 -0.004372 -0.006835 -0.0036174
## [3,]  0.9948483  0.991165  1.000000 -0.005880 -0.008046 -0.0051770
## [4,] -0.0006680 -0.004372 -0.005880  1.000000  0.994805  0.9945635
## [5,] -0.0032162 -0.006835 -0.008046  0.994805  1.000000  0.9898847
## [6,] -0.0004052 -0.003617 -0.005177  0.994563  0.989885  1.0000000
```

How about nonlinear NNS correlation?

```
options(digits=4)
as.matrix(NNS.cor(cbind(data$x,data2$x,data3$x,data$y,data2$y,data3$y)))
```

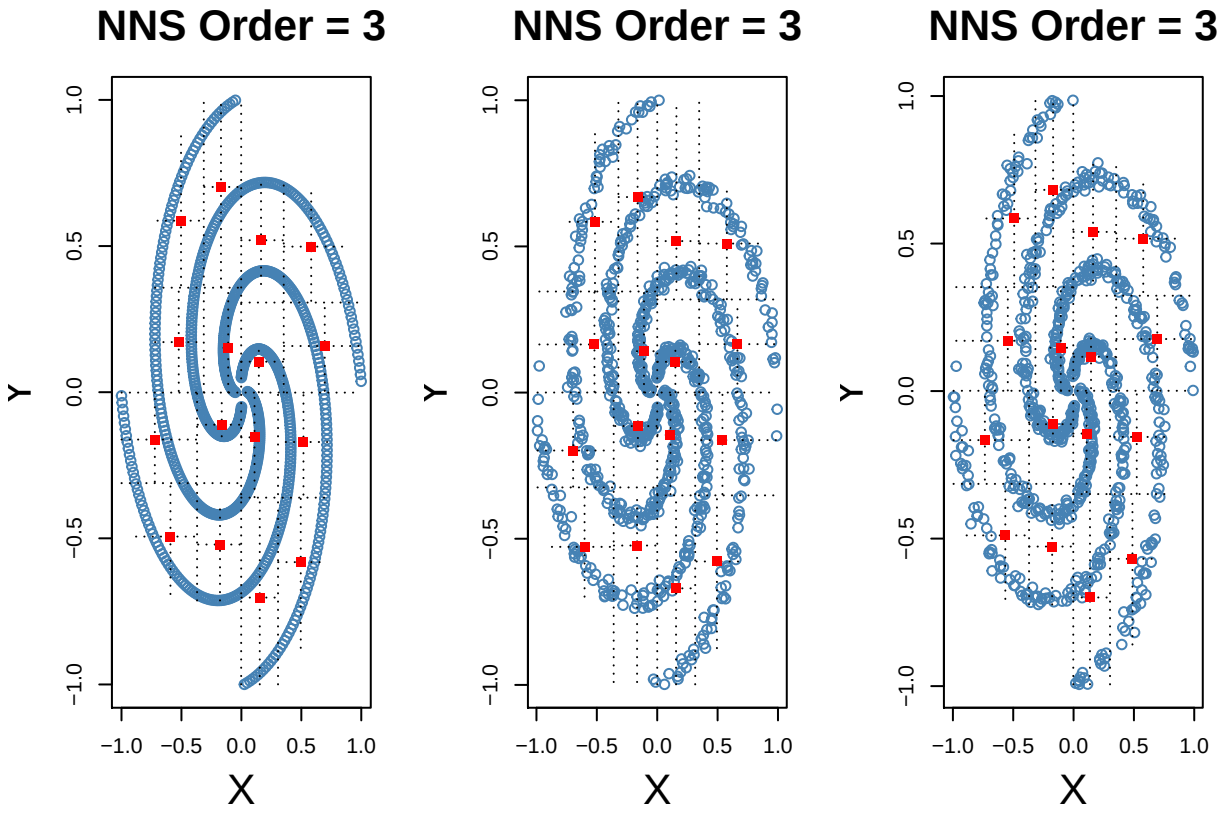
```
##          [,1]    [,2]     [,3]     [,4]     [,5]     [,6]
## [1,]  1.00000  0.9778  0.97996 -0.02565  0.02666 -0.02647
## [2,]  0.97778  1.0000  0.96811 -0.13928 -0.08060 -0.13563
## [3,]  0.97996  0.9681  1.00000 -0.11853 -0.05826 -0.10397
## [4,] -0.02565 -0.1393 -0.11853  1.00000  0.96459  0.97178
## [5,]  0.02666 -0.0806 -0.05826  0.96459  1.00000  0.95416
## [6,] -0.02647 -0.1356 -0.10397  0.97178  0.95416  1.00000
```

Nope, still correlated. We need a different feature to distinguish these spirals.

NNS Clustering

Next let's compare all 3 different spirals using the partition map in NNS. Visually, this is not too different from an analyst placing dots on fingerprints at feature locations.

```
par(mfrow=c(1,3))
order=3
z=NNS.part(data$x,data$y,Voronoi = T,order=order)$regression.points
z2=NNS.part(data2$x,data2$y,Voronoi = T,order=order)$regression.points
z3=NNS.part(data3$x,data3$y,Voronoi = T,order=order)$regression.points
```



Distances

Now we can compare the regression points of each sprial to ascertain whether they have the same structure. Euclidean distances are used to reduce the (x, y) coordinates to a single distance metric.

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{1}$$

Central Coordinates

Our first step requires a central point for each set of coordinates. Transforming our equation to:

$$d = \sqrt{(x - \bar{x})^2 + (y - \bar{y})^2} \tag{2}$$

We subtract our column means from each point, transforming our data tables:

```
rr<- seq_len(nrow(z))
for(j in 2:3) set(z,rr,j,z[[j]]-mean(z[[j]]))
for(j in 2:3) set(z2,rr,j,z2[[j]]-mean(z2[[j]]))
for(j in 2:3) set(z3,rr,j,z3[[j]]-mean(z3[[j]]))
```

Squared Distances from Central Coordinates

Borrowing from Pythagoras, we then square the distances from the central coordinates of each set, sum them, and take their square root.

```
z[,Dist := sqrt(sum(x^2,y^2)),by=quadrant]
z2[,Dist := sqrt(sum(x^2,y^2)),by=quadrant]
z3[,Dist := sqrt(sum(x^2,y^2)),by=quadrant]
```

Hypoteneuse

Following our (and Euclid's) Pythagorean guideline, we take the square root of the summed squares for the hypotenuese, or distance from the central point.

```
dis.1=z[,Dist]
dis.2=z2[,Dist]
dis.3=z3[,Dist]
```

Comparison

Identical geometric structures (the constellation of NNS points) will have perfectly correlated distances. But, if the images and constellations are rotated, how do we reconcile the alignment? Sorting the distances should still yield identical structures...

Pearson Correlation:

```
cor(cbind(sort(dis.1),sort(dis.2),sort(dis.3)))
```

```
##         [,1]   [,2]   [,3]
## [1,] 1.0000 0.9976 0.9988
## [2,] 0.9976 1.0000 0.9973
## [3,] 0.9988 0.9973 1.0000
```

We can see how Pearson correlation fails miserably at this task since it considers spiral 3 a near perfect match to spirals 1 and 2.

NNS Correlation:

```
NNS.cor(cbind(sort(dis.1),sort(dis.2),sort(dis.3)))
```

```
##       [,1]  [,2]  [,3]
## [1,] 1.000 0.875 0.625
## [2,] 0.875 1.000 0.750
## [3,] 0.625 0.750 1.000
```

Voila, the same seed sprials (1&2) are a match! The nonlinear NNS correlation measure is able to retain the very high structure identification while refraining from classifying the distinct 3rd spiral as the same as the other two.

More NNS:

To learn more about NNS statistics and their theoretical foundations, see "*Nonlinear Nonparametric Statistics: Using Partial Moments*" available on Amazon: http://a.co/5bpHvUg

Check back to see more NNS examples posted on GitHub: https://github.com/OVVO-Financial/NNS/tree/NNS-Beta-Version/examples

See link for blog post highlighting this dataset: https://www.r-bloggers.com/build-your-own-neural-network-classifier-in-r/