train(dataMaj, dataMin)
- Fit dataMin for neighbourhood search
- for all x ∈ dataMin
- Search and store neighbourhood indices NI(x) for x
- end loop for x
- initialize
- store dataMin
- for epoch ∈ {0 .. epochs}
- for x ∈ dataMin
- minBatch ← NMB(x)
- majBatch ← NMB(x)
- convSamples ← predict(minBatch)
- concatSample ← (covSamples | majBatch)
- trainFit(discriminator, concatSample)
- trainFit(generator, concatSample)
- end loop for x
- end loop for epoch

generateSynthData(N)
- $M \leftarrow 1 + \lfloor N / |dataMin| \rfloor$
- synthSet ← ∅
- for each x ∈ dataMin:
- B ← ∅
- for $k \in \{ 0 .. \lfloor M / GEN \rfloor \}$:
- $B \leftarrow B \bigcup predict(NMB(x))$
- end loop for k
- $synthSet \leftarrow synthSet \bigcup B[1..M]$
- end loop for x
- ← synthSet[1 ..N]

NMB_old(x)
- Fit dataMin for neighbourhood search
- Search neighbourhood indices NI(x) for x
- shuffle NI(x)
- ← { x_i ∈ dataMin : i ∈ NI(x) }

NMB_new(x)
- Take indices NI(x) for x from stored list.
- shuffle NI(x)
- ← { x_i ∈ dataMin : i ∈ NI(x) }

BMB_old
- Fit dataMaj for neighbourhood search
- bmbi ← Search neighbourhood indices NI(x) for all x ∈ dataMin
- ← GEN random points from dataMaj ignoring the value of bmbi

BMB_new
- ← GEN random points from dataMaj

Legend:
- unchanged operation
- removed operation
- call to changed functions
- affected loop
- affected function
- new operations

Speed-test:

without changes:
total: #9 (453.6)

with changnes:
total: #9 (434.6)
train: #9 (338.0s)
generateSynthData: #348 (88.2s)

NMB: #6501 (1.7s)
BMB: #3480 (0.3s)
predict@generateSynthData: #3012 (87.4s)

Changes gives speedup ~ 20s

Major time is lost during training and for predicting.