ENA Model Comparison Vignette

Zachari Swiecki

05/10/2021

The following demonstrates comparing a given ENA model to a distribution of randomized models. A motivating example and explanation of the test method can be found in: Swiecki, Z. (in press). The expected value test: A new statistical warrant for theoretical saturation. Paper accepted to the Third Annual International Conference on Quantitative Ethnography.

First, create the observed ENA model. Note that the model comparisons use the full adjacency vector for each unit, so we only need `ena.accumulate.data` and the normalization function. Here, I use the data built into the rENA package.

```
data(RS.data)

codeNames = c('Data','Technical.Constraints','Performance.Parameters',
  'Client.and.Consultant.Requests','Design.Reasoning','Collaboration');

accum = ena.accumulate.data(
  units = RS.data[,c("UserName","Condition")],
  conversation = RS.data[,c("Condition","GroupName")],
  metadata = RS.data[,c("CONFIDENCE.Change","CONFIDENCE.Pre","CONFIDENCE.Post")],
  codes = RS.data[,codeNames],
  window.size.back = 4
)

obs.lineweights = as.matrix(accum$connection.counts) %>% rENA::fun_sphere_norm()
```

Next, create a list of randomized datasets and produce the adjacency vectors associated with each using `rep.random.lws`. You can vary the percentage of lines in a given dataset that are randomized using the percent parameter. Here, I am creating 1000 randomized datasets where all lines have been randomized. Note that this will take several minutes to run, so I've provided a set of pre-calculated randomized data to load.

```
# rand.data = rep.random.lws(dataset = RS.data,
#                   speakerCol = "UserName",
#                   codeCols = codeNames,
#                   unitCols = c("UserName","Condition"),
#                   convoCols = c("Condition","GroupName"),
#                   metaCols= c("CONFIDENCE.Change","CONFIDENCE.Pre","CONFIDENCE.Post"),
#                   model = "E",
#                   window = 5,
#                   reps = 1000,
#                 percent = 100
#                 )

load("~/Rprojects/model-comparisons-v2/data/rand.data.ex.Rdata")
```

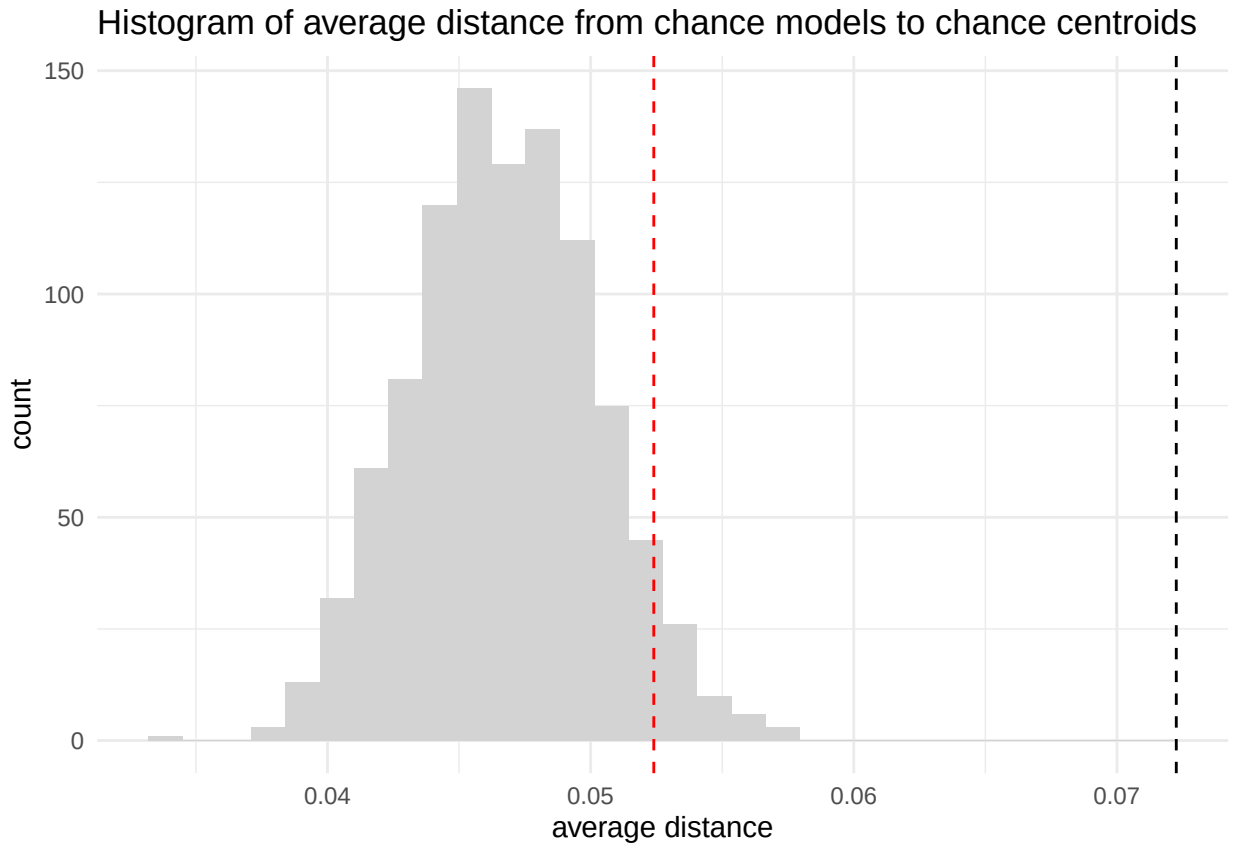
Now run the model comparison test for the model overall.

```
res.overall = ena.compare.models(observedMod = obs.lineweights,
                                 simMods = rand.data,
                                 method = "euclidean")
```

Plot the results. Here, the red line is the 95th percentile for the distribution of average distances between the chance models and chance-based centroids. The black line marks the average distance between the observed models and the chance-based centroids. The results show that the observed distance is greater than 95% of the distances under the null hypothesis. Thus, the overall model is significantly different from chance.

```
hplot = ggplot(data.frame(res.overall$distribution),aes(x = res.overall$distribution)) + geom_histogram
  geom_vline(
    xintercept=quantile(res.overall$distribution,probs = 0.95),
    linetype = "dashed",color = "red") + xlab("average distance") + ggtitle("Histogram of average distar
  theme_minimal()

hplot
```



We can conduct a similar test for a given individual's model (i.e., adjacency vector). Here, we see that the first unit's model is not significantly different than chance.

```
unit.pos = 1
unit.lws = obs.lineweights[unit.pos,] #the first unit
my_element = function(x) x[unit.pos,]
sim.unit.lws = map(rand.data,my_element)

res.indiv = ena.compare.model.indiv(observedMod = unit.lws,simMods = sim.unit.lws,method = "euclidean")

hplot.ind = ggplot(data.frame(res.indiv$distribution),aes(x = res.indiv$distribution)) +
```

```
  geom_histogram(fill = "light grey") +
  geom_vline(xintercept=quantile(res.indiv$distribution,probs = 0.95), linetype = "dashed",color = "red
  geom_vline(xintercept = res.indiv$statistic, linetype = "dashed",color = "black") +
  theme_minimal() +
  xlab("distance") +
  ggtitle(label = "Histogram of distance from chance models to chance centroid")

hplot.ind
```