PSEUDOCODE

Initial conditions/setup

```
Set n (network size)
Set k (no of nodes per cave)
Set number-of-caves = n/(k+1)
Set ɸ (proportion rewired)
Set number-static-traits (S) = static attributes (randomly create 5 numbers - binary 0 or 1)
Set number-dynamic-traits (O) = dynamic attributes (randomly create 20 numbers - binary 0 or 1)
Set urn = an empty list that will contain all the selected opinions of a given agent's neighbors related to a given trait
Calculate (expected) social distance
```

Calculate social distance

Agent i and *j* at time *t*:

dij,t = $\sqrt{\sum_{m\epsilon s}(s_{mi} - s_{mj})^2 + \sum_{m\epsilon o}(o_{mi,t} - o_{mj,t})^2}$

if t = 0, distance is given by the randomly distributed traits. This is the expected distance (E(d))

Calculate weight

Agent i and *j* at time *t*:

wij,t = E(d)-$d_{ij,t}$

Run the model/go:

```
Randomly choose an agent (asynchronous update)
    Update dij
    Update wij
    Fill the urn and select an opinion
```

Fill the urn and select an opinion:

```
If expected distances and current distances are different:
    Set probability of adoption Pj,t = $|w_{ij,t}|$ / (∑A(I,k)=1* $|w_{ik,t}|$), where Pj,t is probability that agent i adopt agent j's opinion; A(i,k) = 1 when agents i and k are connected and A(i,k) = 0 when agents i and k are not connected
        If probability is large (when compared to a random number between 0 and 1) and neighbor's weight is negative, randomly put either a 0 or a 1 in the urn with probability of 10%
        If probability is large (when compared to a random number between 0 and 1) and neighbor's weight is poisitve, put the neighbor's opinion in the urn
            Randomly select an opinion from the urn
            Go to the next neighbor
                Repeat process until all urns/dynamic traits of the focal agent are compared
```