One Earth, Volume *2*

Supplemental Information

Cultural Evolution of Sustainable Behaviors: Pro-environmental Tipping Points in an Agent-Based Model

Roope Oskari Kaaronen and Nikita Strelkovskii

Supplemental Experimental Procedures

Software

In this research article and NetLogo (version 6.1.0) model, we use NetLogo's native BehaviorSpace tool for parameter sweeping, and NetLogo's BehaviorSearch for Genetic Algorithms [1]. We use R [2] and R Studio [3] and R packages tidyverse [4], factoextra [5], Hmisc [6], plyr [7], RColorBrewer [8], reshape2 [9], gridExtra [10] and nlrx [11] for data analysis and visualisation.

ODD Protocol

The following model description follows the ODD (Overview, Design concepts, Details) protocol for describing agent-based models [12,13].

1. Purpose

This model illustrates the cultural evolution of pro-environmental behaviour patterns. It shows how collective behaviour patterns evolve from interactions between agents and agents (in a social network) as well as agents and the affordances within a niche. More specifically, the cultural evolution of behaviour patterns is understood in this model as a product of:

1. The landscape of affordances (action opportunities) provided by the material environment,
2. Individual learning and habituation,
3. Social learning and network structure,
4. Personal states (such as habits and attitudes), and
5. Cultural niche construction, or the modulation of affordances within a niche.

More particularly, the model illustrates how changes in the landscape of affordances [14] can trigger nonlinear changes in collective behaviour patterns. The model also shows how several behavioural cultures can emerge from the same environment and even within the same network.

The model is an elaboration of Kurt Lewin's [15] heuristic equation, $B = f(P, E)$, where behaviour (B) is a function (f) of the person (P) and the environment (E). The model introduces several feedback loops (1–5 above) to Lewin's equation, and thus provides a framework for studying the evolution of dynamical and complex behavioural systems over time. The model should be considered an abstract model, since many of its parameters are unspecifiable due to limits to current understanding of human (social) behaviour. However, the model can be tuned to replicate real-world macro patterns, and be used as a sandbox environment to locate tipping points in social systems. In the present manuscript, for example, we use the model to reproduce real-world patterns of bicycle and car use in Copenhagen.

2. Entities, state variables, and scales

The model includes three types of agents: human individuals, represented by mobile circle-shaped agents (or 'turtles' in NetLogo lingo), affordances (static patches that occupy grid cells) and links (which connect agents in a social network).

Individuals: Agents represent a single human being, located within a broader collective social network and ecological niche. Each individual has two personal states. These personal states correspond to the individual's probability of engaging with a specific kind of affordance. Affordances are opportunities for action provided by the environment. The two personal states in this model are *pro-env* and *non-env*. The former, *pro-env*, defines the probability of an

individual to engage with pro-environmental affordances, and the latter, *non-env*, defines the probability of an individual to engage with non-environmental affordances.

The personal states of individual agents are sampled from a normal distribution with mean values *initial-pro* (for *pro-env*) and *initial-non* (for *non-env*), and SD 0.15. This standard deviation is roughly in line with empirical data related to environmental attitudes and self-reported behaviours [16]. Owing to the model's probabilistic representation of human behaviour, the values of *pro-env* and *non-env* must be bounded between 0 and 1. More specifically, the model assigns individual boundaries for the *pro-env* and *non-env* of each agent. The bounds are sampled from a normal distribution with mean values 0.2 (lower bound) and 0.8 (upper bound), with SD 0.05.

Individuals are coloured based on their personal states. This is purely cosmetic, but it aids in noticing changes in personal states. If *pro-env* > *non-env*, the agent is coloured black. If *non-env* > *pro-env*, the agent is coloured red.

Links: Individual agents are embedded in a social network which is connected by links. The model supports four types of networks: the Klemm-Eguíluz model (highly clustered scale-free network), the Watts–Strogatz model (small-world network), the Barabási–Albert model (scale-free network with preferential attachment) and the Erdős–Rényi model (random network). All network edges (links) are undirected (bidirectional).

The default network choice is the Klemm-Eguíluz model [17]. The Klemm-Eguíluz algorithm generates a network based on a finite memory of the nodes (agents), creating a highly clustered and scale-free network (see Figures S2–S4). The Klemm-Eguíluz model was chosen since it represents two features we know to characterize social systems: Societies have hubs (the network degree distribution follows a power law distribution, i.e. it has scale-free properties) and societies have highly clustered local communities (social networks have high clustering

coefficients) (ibid.). See Klemm and Eguíluz [17] and Caparrini [18] for descriptions of how Klemm-Eguíluz model works, as well as Prettejohn et al. [section 3.4 in 19] for useful pseudocode. We set the default Klemm-Eguíluz model's parameter *m0* (initial number of agents) to 5 and μ (probability to connect with low degree nodes) to 0.9.

Figure S1. Class diagram (UML).

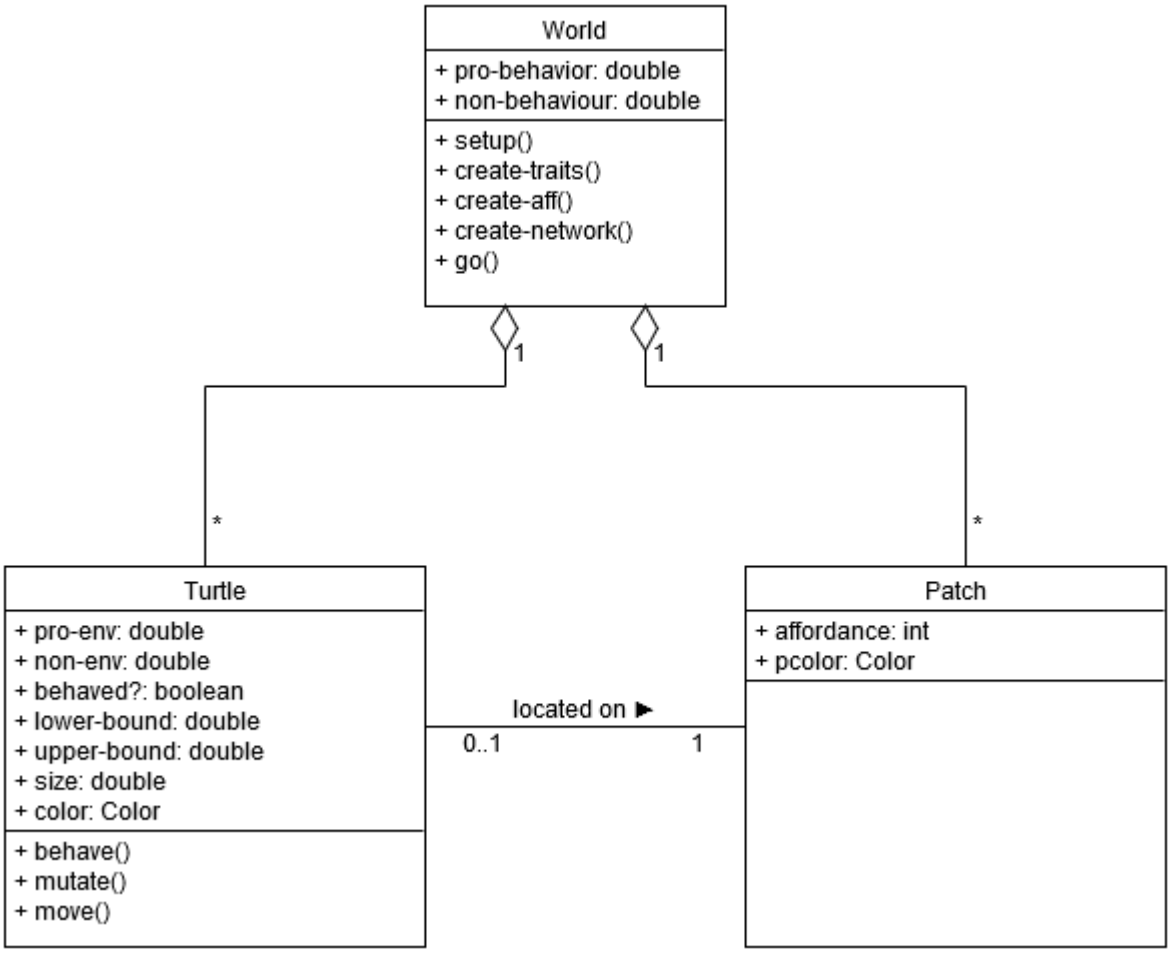


Figure S2. Network degree distribution. A representative plot of the network degree distribution from a single model run with 300 agents. Notice how some agents have amounts of links that greatly exceed the mean (black dashed line) and median (red dashed line).

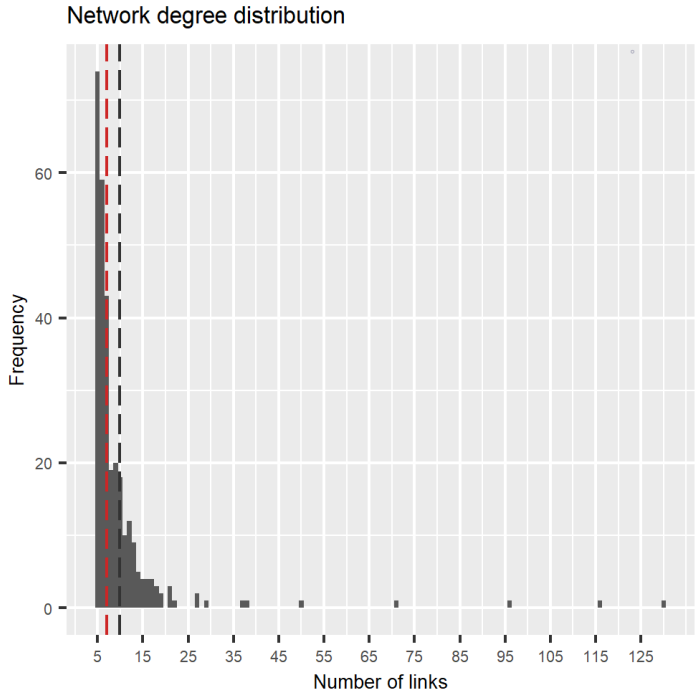


Figure S3. Cumulative network degree distribution. 1000 simulations (total of 300,000 agents) on a logarithmic scale. Notice the scale-free density distribution and relative infrequency of agents with above 150 direct links. Mean links are signified by the black dashed line and median by the red dashed line.

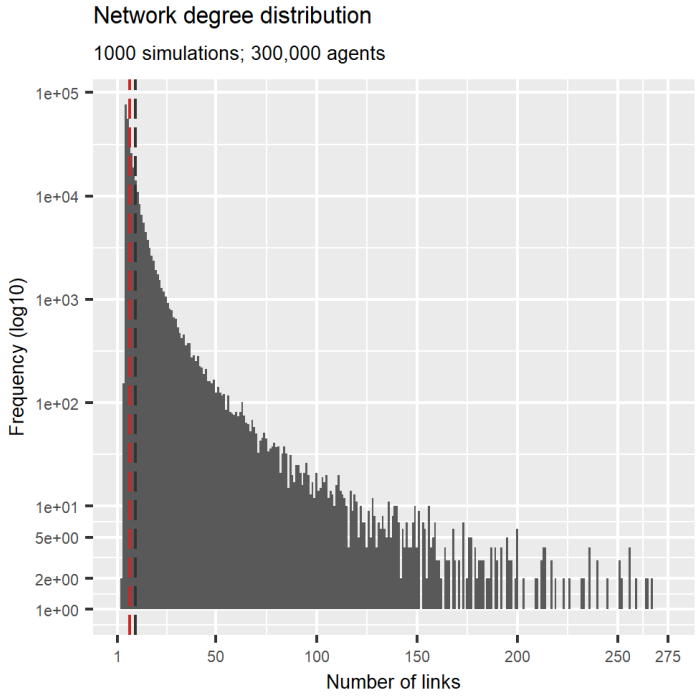
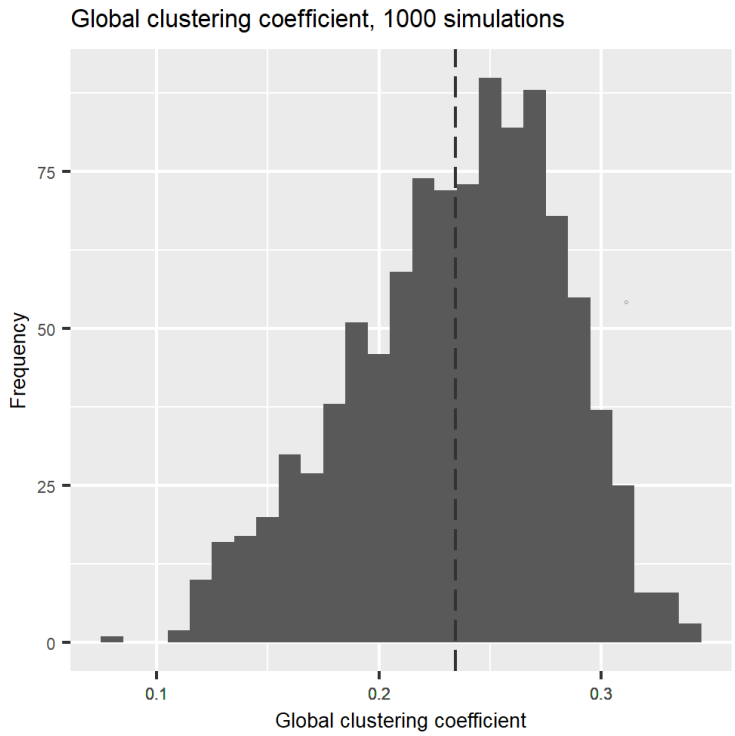


Figure S4. Global clustering coefficients. Histogram with 1000 runs with 100 agents. Global clustering coefficients are calculated based on triplets of nodes. Triplets are three nodes which are connected either by two (open triplet) or three (closed tripled) edges (links). The global cluster coefficient is the number of closed triplets in a network divided by the total number of triplets. Dashed line is at the mean global clustering coefficient, 0.24.



Patches (environment): Patches represent the action-opportunities, or affordances, within the environment. An affordance is the functional relevance of the environment for an individual. The model has two affordances: One represents an opportunity for pro-environmental behaviour (represented by a violet patch) and one represents an opportunity for environmentally harmful behaviour (sky-blue patch). The latter are from here on referred to as non-environmental affordances. The affordances of the environment are therefore binary in this model, even though nothing prevents the addition of more kinds of affordances. Affordance-patches occupy the two-dimensional grid of the model. The grid wraps horizontally and vertically (i.e., it is torus-shaped). The total area of the grid is an arbitrary 201x201 patches.

Scales: The model can be adapted to represent different spatial and temporal scales. One time-step can be understood to either represent one instance of behaviour per agent, or a collection of behaviours. In the abstract version of the model, the spatial and temporal scales are not specifically defined. In empirical validation, the spatial area of the model represents the city centre of Copenhagen, with each tick representing one day.

3. Process overview and scheduling

The submodels of the model are described in more detail and pseudocode in the *Submodels* section. In this section, we describe a brief process overview.

Setup: The model begins with a setup phase where the patches, agents and links are created. Ticks are reset after the setup, so all setup processes occur before the first timestep.

First, the social network (agents and links) is created. This will create a network with individuals specified by the parameter *number-of-agents*.

Second, each agent is assigned two personal states, *pro-env* and *non-env*.

Third, affordances are created. Affordances are binary patches-own variables: value 0 signifies a non-environmental affordance, and value 1 a pro-environmental affordance. First, all patches are assigned with a non-environmental affordance (and coloured sky-blue). Subsequently, the proportion of patches designated by the parameter *pro-amount* are turned into pro-environmental affordances. Therefore, the parameter *pro-amount* corresponds to the initial proportion of pro-environmental affordances within the total landscape of affordances.

Go: The ‘Go’ procedure is the heart of the model.

First, agents behave. If the agent is on a pro-environmental affordance, it will interact with it with the probability of P(*pro-env*). For example, if an agent’s personal state *pro-env* is 0.5, it has a 50% chance of interacting with a pro-environmental affordance.

Likewise, if the agent is on a non-environmental affordance, it will interact with it with the probability of P(*non-env*). Again, if an agent’s personal state non-env is 0.7, it has a 70% chance of interacting with a non-environmental affordance.

A while-loop ensures that each agent behaves once every turn. Each agent owns a binary value, *behaved?*, which signifies whether it has behaved, or actualized an affordance, during the current tick. If *behaved?* is TRUE, the agent will stop attempting to behave after completing the behaviour commands (including steps 1–5 below).

Once an agent behaves successfully, a sequence of procedures launched in the following order.

1. If the agent behaved pro-environmentally (i.e., it actualizes a pro-environmental affordance), it will increase its current personal state *pro-env* by the amount of *asocial-learning* and decrease its current *non-env* by the amount of *asocial-learning*.

   Conversely, if the agent behaved non-environmentally (i.e., it actualizes a non-environmental affordance), it will increase its current *non-env* by the amount of *asocial-learning* and decrease its current *pro-env* by the amount of *asocial-learning*.

2. If *niche-construction* is TRUE (niche construction is turned on) and if the agent behaved pro-environmentally, with probability *construct-pro* it will ask one of the eight patches in its Moore neighbourhood to turn into a pro-environmental affordance (which is then coloured in violet). *construct-pro* therefore defines the rate of pro-environmental

niche construction. The procedure is identical for non-environmental niche construction (following non-environmental behaviour), whose rate is defined by *construct-non.* Rates of niche construction are controlled for number-of-agents. This way, adding more agents to the simulations does not add to the rate of overall niche construction. This is necessary because the area (grid) of the model is held constant.

3. If *networks* is TRUE and if the agent behaved pro-environmentally, it will engage in social learning with its network neighbours (the agents to which it is directly connected to by a link). Following pro-environmental behaviour, the agent will ask its network neighbours to increase their current *pro-env* by the amount specified by parameter *social-learning*, as well as to decrease their current *non-env* by the amount specified by parameter *social-learning*. Again, the procedure is similar after non-environmental behaviour, except this results in an increase of *non-env* and decrease of *pro-env* by the amount of *social-learning*.

4. The agent will bound its personal states *pro-env* and *non-env*. If the agent's personal state is above its upper bound or below its lower bound, it will set its personal state to its upper and lower bound, respectively.

5. If mutate? Is TRUE, at each tick, the pro-env and non-env of all agents have a chance of mutating. The default probability for mutation (*mutate-prob*) is 0.005, and the default rate for mutation (*mutate-rate*) is 0.05. The probabilities for increasing or decreasing pro-env and non-env values (of all agents) are equal, i.e. mutation is not biased to any direction.

After each behaviour or attempt to behave, agents move in a random forward direction between 45 degrees right and 45 degrees left from their current heading. In one tick (time-

step) agents will continue moving until they have behaved, i.e. until they have successfully interacted with an affordance.

The aforementioned steps are sequential: An agent completes the full set of actions before passing on control to the next agent. The order of agents is read in a random order on each tick.

4. Design concepts

Basic principles.

The model design elaborates on social psychologist Kurt Lewin's [15] heuristic equation: $B = f(P, E)$. Here, behaviour (B) is a function (f) of the person (P) and its environment (E).

The model adds five dimensions of detail into Lewin's equation.

1. The environment affords a variety of opportunities for action, or affordances ($E \rightarrow B$).
2. Behaviour modulates personal states through processes of habituation and individual learning ($B \rightarrow P$).
3. Personal states, such as habits and intentions, drive behaviour ($P \rightarrow B$).
4. Behaviour shapes the environment through processes of niche construction ($B \rightarrow E$).
5. Feedback loops 1–4 all occur within a social network where behaviour is transmitted via social learning ($B_{myself} \rightarrow P_{neighbors}$ and $B_{neighbors} \rightarrow P_{myself}$).

These assumptions are elaborated in detail in the manuscript's section Model Assumptions.

The basic principles can be summarized as follows: Through processes of individual and social learning as well as niche construction, any behaviour at time t will have an effect on the behaviour of an agent and other agents at time $t+1$. The model therefore presents a dynamical

systems approach to the emergence of human behaviour, where the unit of study is a tightly coupled human-environment system – a dynamical system which evolves over time and can behave in nonlinear ways due to positive feedback-loops.

Emergence.

The model produces a complex and dynamical system which exhibits several kinds of emergent behaviour.

Firstly, the model displays nonlinearities in the development of behavioural cultures (collective behaviour habits). The behaviour of the agents in the network can be steady for long periods of time, only to be followed by abrupt phase transitions into new states (this is illustrated in more detail in the Results section of the manuscript).

Second, the model illustrates how two different behavioural cultures can emerge from the same environment, and even in the same social network. This is a macro-level pattern that is known (from studies of cultural evolution) to occur in real-world societies [20].

Third, the model has several leverage points. For instance, a small change (e.g., 5–10%) in the initial composition of affordances in the landscape can have radical effects on the evolution of the behavioural cultures. Thus, in a way which is typical to complex emergent systems, the model is sensitive to initial conditions, which makes its evolution difficult to predict at certain parameter ranges.

Fourth, whilst the model always starts with a random composition of the affordance landscape, this landscape gets more structured over time as individuals construct the niche around them.

Adaptation.

Through processes of individual and social learning, agents adapt their personal states to their behaviour and to their immediate social environment. Moreover, agents construct their environment to be more predictable by constructing niches which are in line with past behaviour.

Objectives.

Agents engage in active attempts to behave successfully (actualize an affordance) and to create an environment where past behaviour patterns are increasingly more likely.

Learning.

The model includes two learning processes, individual and social learning. Individual (asocial) learning occurs after behaviour and affects only the agent who behaved. Individual learning is thus a product of individual behaviour. Social learning occurs in the social network an agent is embedded in.

The rates of individual and social learning depend on the chosen representation of behaviours and time-units. Realistic rates of individual and social learning are therefore difficult to specify. However, by studying real-world patterns, it might be possible to infer reasonably accurate rates of social and individual learning (see section Empirical Validation of the manuscript).

Prediction.

Agents do not estimate future conditions or consequences of their decisions.

Sensing.

Agents sense the (colour of the) patch they are currently on as well as their network neighbours and neighbours' behaviour. Agents also sense their physical vicinity, i.e. the patches in their Moore neighbourhood (the 8 patches surrounding the patch they are currently on).

Interaction.

After behaving, agents interact with their network neighbours. This involves both influencing the network neighbours as well as being influenced by each network neighbour (both defined by the rate of *social-learning*). Niche construction also influences the behaviour of other agents, and is thus an indirect form of social interaction.

Stochasticity.

The following processes rely on random sampling:

The initial personal states of agents are sampled from a normal distribution (see section 2 of ODD protocol above). The initial configuration of affordances on the grid is random (the proportion of pro-environmental affordances, however, is fixed by the parameter *pro-amount*). The movement of agents on the grid is a random walk through the landscape of affordances. Each instance of behaviour and niche construction makes use of a floating random number generator. The model supports the use of a fixed random seed for replicability (if *random-seed?* is TRUE, a random seed can be fixed with the *rseed* parameter).

Collectives.

Individuals belong to a social network and construct their niche, as defined above. Individuals take part in shaping the collective network and niche which, in turn, shapes their behaviour.

Observation.

Observation generally involves tracking mean or specific values over time. The most relevant variables are the global variables *pro-behavior* and *non-behavior*, which track the total amount of pro-environmental and non-environmental behaviour during each tick.

Parameter sweeps are conducted via NetLogo's native BehaviorSpace tool.

5. Initialization

The initialization of the model is allowed to vary among simulations. Since many values, such as the personal states of agents, are randomly sampled, each model run will differ from the next even when run with the same parameter values.

However, the model supports the use of a fixed random seed for replicability (if *random-seed?* is TRUE, a random seed can be fixed with the rseed parameter).

The initial state of the model at $t = 0$ will depend on the parameters *initial-pro, initial-non, pro-amount* and the network parameters (*networks*, *network-type*) as defined above.

In the abstract version of the model, the initial states are arbitrary. The abstract model can be used to study the dynamics and sensitivities of the model's general structure.

In empirical validation, the initial states of the model are tuned to reproduce real-world patterns, or the cycling and driving habits of people in central Copenhagen.

6. Input data

The model does not use input from external sources such as data files or other models.

7. Submodels

In the following, the processes mentioned in *Process overview and scheduling* (above) are described in more detail in pseudocode, flowcharts (UML diagrams) and natural language. Pseudocode is written by editing NetLogo code to resemble natural language. Whilst the descriptions below are comprehensive, please also refer to the fully annotated model code for details. The following section documents the *SETUP* submodels (*Social network, Personal states and Affordances*) and the *GO* submodels (*Behavior* and *Mutate*). *Behavior* includes descriptions of the processes of individual learning, niche construction and social learning.

SETUP

Social network

Since fully a full description of the Klemm-Eguíluz model would require a chapter-length analysis, we refer the reader to Caparrini's Complex Networks Toolbox [18] for a description of the Klemm-Eguíluz small-world-scale-free network (we adapted, with permission, Caparrini's code for the present model). A full pseudocode description of the Klemm-Eguíluz model is openly accessible in Prettejohn, Berryman and McDonnell's [19] chapter '3.4 Klemm and Eguílez

Small-World-Scale-Free Network'. A full mathematical description of the model is also available in Klemm-Eguíluz' original work [17].

Personal states

Personal states are created in the model setup. In pseudocode,

```
to set personal states
for each turtle in the list of all turtles [
  Set pro-env: sample a random value from a normal distribution with
  mean of initial-pro and a standard deviation of 0.15.
  Set non-env: sample a random value from a normal distribution with
  mean of initial-non and a standard deviation of 0.15.
  Set lower-bound: Set a lower bound for non-env and pro-env from a
  random normal distribution with mean 0.2 and SD 0.05.
  Set upper-bound: Set an upper bound for non-env and pro-env from a
  random normal distribution with mean 0.8 and SD 0.05
  ]
  end
```

Affordances

Affordances are patches-own variables. Affordances are created with the following procedure (pseudocode):

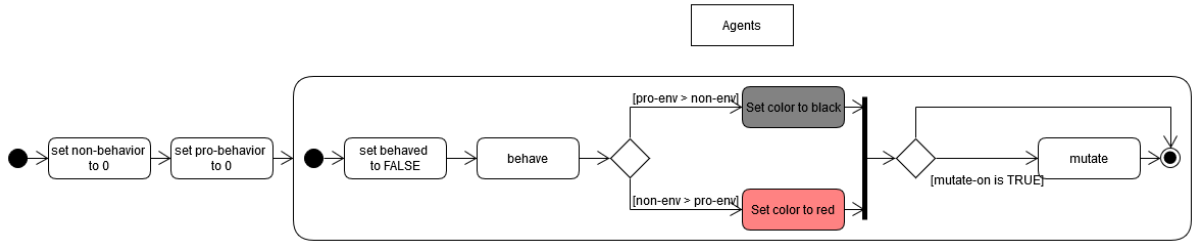
```
to create affordances
let total-patches be total count of patches
  ask all patches [
      set affordance to 0 ;; non-evironmental affordance
```

```
      set color to sky-blue ]
  ask n-of (total-patches * pro-amount) patches [
      set affordance to 1 ;; pro-environmental affordance
      set color to violet]
end
```

GO

The go-procedure begins with each agent resetting their global *pro-behavior* and *non-behavior* variables to 0 (these global variables measure the total pro- and non-environmental behaviours of all agents at the end of each tick). Then, agents set their *behaved?* variable (turtles-own variable) to FALSE. The *behaved?* variable ensures that each agent behaves (either pro- or non-environmentally) only once during a tick. After this, agents behave.

Figure S5. Go procedure, activity diagram (UML).



Behavior

This submodel is the heart of the model. It defines how agents interact with the environment and other agents. Since the procedure is identical for both pro-environmental and non-environmental behaviours, only pro-environmental behaviour is described here. To implement non-environmental behaviour, simply duplicate the code and replace ‘pro-environmental’

(value 1) patch with ‘non-environmental’ (value 0), ‘violet’ with ‘sky-blue’, and *pro-env* with *non-env* (and vice versa, *non-env* with *pro-env*). The processes of habituation, niche construction and social learning are included in this submodel, and are described below in pseudocode.

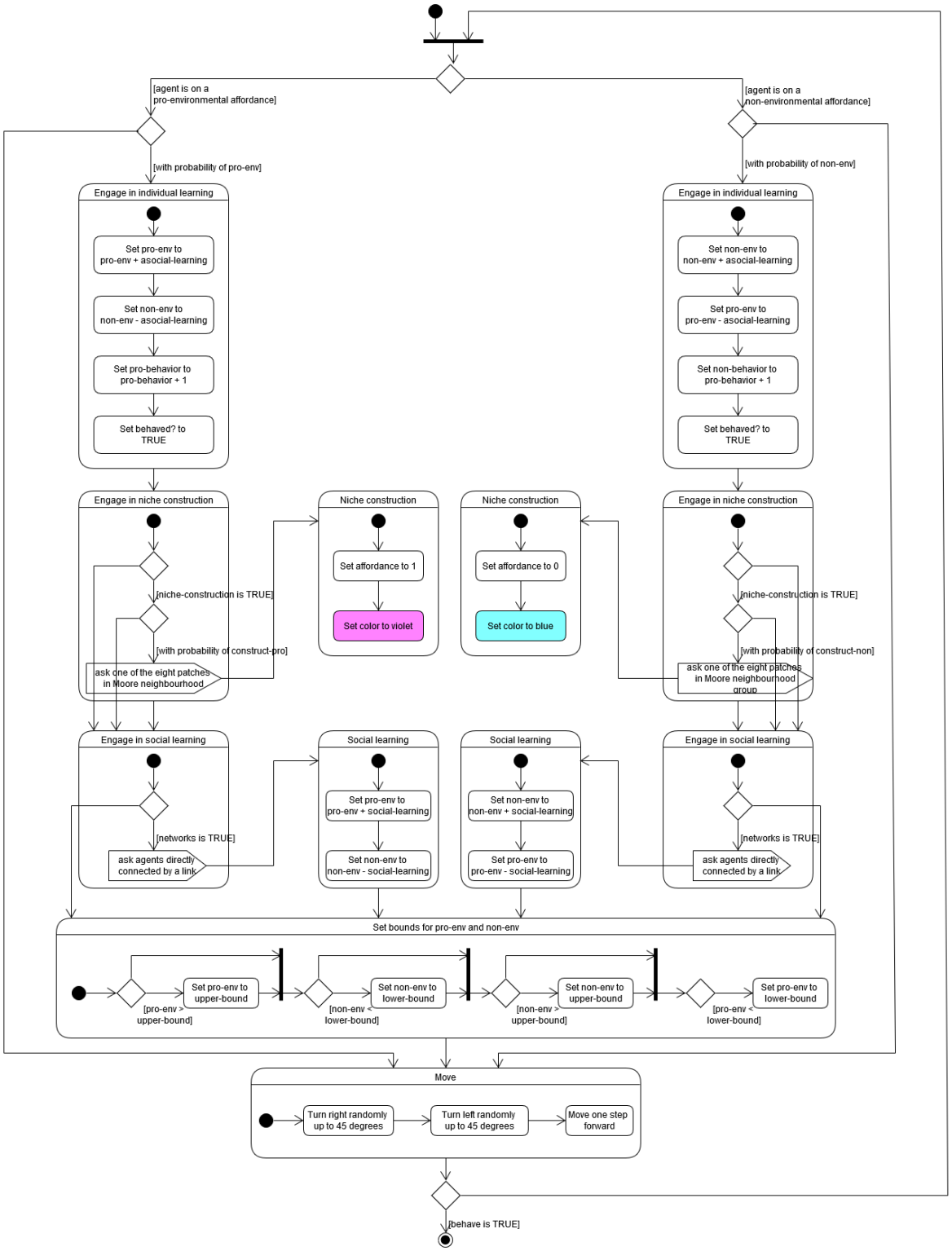
```
to behave
 while behaved? is FALSE [ ;; Start of while-loop
      if the patch the agent is currently on is pro-environmental
      and random-floating number in range [0,1] is smaller than
      pro-env [
;; Engage in individual learning
      set pro-env to (pro-env + asocial-learning)
      set non-env to (non-env - asocial-learning)
      set pro-behavior to (pro-behavior + 1)
      set behaved? to TRUE
;; And still complete the following commands (we are still in the
while-loop)

;; Engage in niche construction
if niche-construction is TRUE [
      if random-floating number in range [0,1] is smaller than
      (construct-pro / number-of-agents) [
        ask one-of patches in Moore neighborhood [
           set affordance to 1
           set color to violet ]
              ]
           ]
```

```
;; Engage in social learning
if networks is TRUE [
      ask link-neighbors [
        set pro-env to (pro-env + social-learning)
        set non-env to (non-env - social-learning)
            ]
     ]
]

;; Set bounds for pro-env and non-env
if pro-env > upper-bound [set pro-env to upper-bound]
if non-env < lower-bound [set non-env to lower-bound]
if non-env > upper-bound  [set non-env to upper-bound]
if pro-env < lower-bound [set pro-env to lower-bound]

;; Finally, move.
turn right randomly up to 45 degrees
turn left randomly up to 45 degrees
move one step forward
] ;; End of while-loop, and end the behave procedure
end
```

Mutate

```
to mutate

if mutate-on? = TRUE [

let mutate-probability 0.005

let mutate-rate 0.05

if random-floating number in range [0,1] is smaller than mutate-
probability [

    ask turtles [ set pro-env to (pro-env + mutate-rate)]]

if random-floating number in range [0,1] is smaller than mutate-
probability [

    ask turtles [ set non-env to (non-env - mutate-rate) ]]

;; ...and so on for all four possible configurations (mutation is
not biased to any direction.)

if random-floating number in range [0,1] is smaller than mutate-
probability [

    ask turtles [ set non-env to (non-env + mutate-rate) ]]

  if random-floating number in range [0,1] is smaller than mutate-
probability [

    ask turtles [ set pro-env to (pro-env - mutate-rate) ]]

]

end
```

Figure S6. The ‘behave’ submodel, activity diagram (UML).



Sensitivity Analysis

Local Sensitivity Analysis: OFAT Testing

We begin by testing our model's sensitivities based on one-factor-at-a-time (OFAT) sensitivity analysis. OFAT sensitivity analysis 'consists of selecting a base parameter setting (nominal set) and varying one parameter at a time while keeping all other parameters fixed' [21]. It is therefore referred to as a local sensitivity analysis method. For local sensitivity testing, we use the parameter values as defined by Table S3 (the abstract model run), since its output is arguably more intuitive to understand (than the parameter values used for empirical validation), and it is much less computationally demanding. For data visualisation, we use raincloud plots [22], which illustrate the distribution of data points (in this case, the proportion of pro-environmental behaviour at the final timestep, 2000) and a boxplot with medians and ± 1 standard deviations. Since the mechanism for initial-pro and initial-non, as well as construct-pro and construct-non, are identical, only the pro-environmental variants of these parameters are analysed. This produces a total of 7 plots, shown below.

Figure S7. Sensitivity test 1. The model is especially sensitive to the initial proportion of pro-environmental affordances. This is, however, expected on the basis of results such as Figures 2A and 2B. At extreme values such as when pro-amount is larger than 0.75, most agents will behave pro-environmentally.

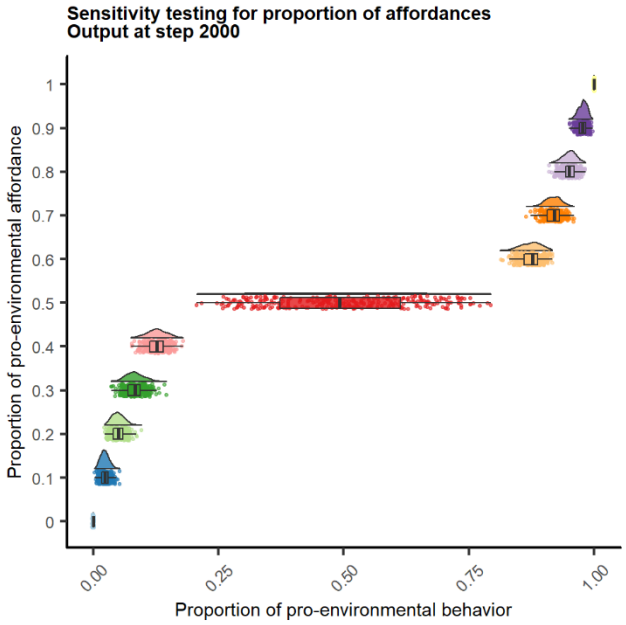


Figure S8. Sensitivity test 2. The model is particularly robust against changes in the rate of individual (asocial) learning.

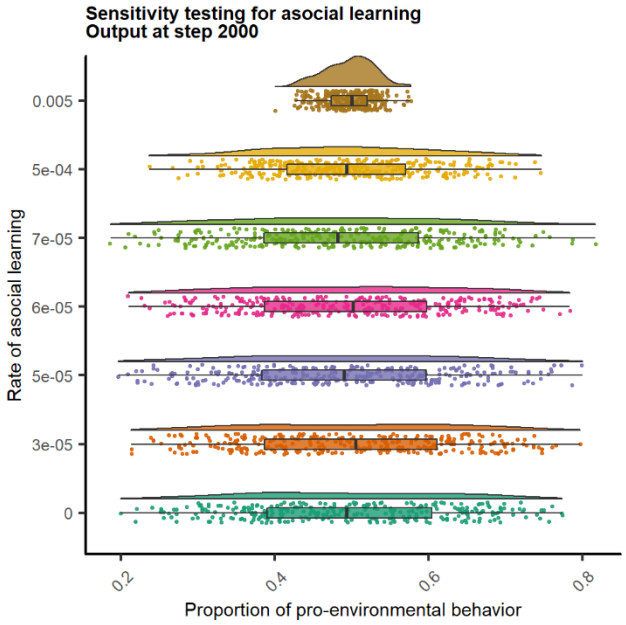


Figure S9. Sensitivity test 3. Higher rates of pro-environmental niche construction will lead to more extreme results in the adoption of pro-environmental behaviour. This effect was also seen and explained in the Results section of the present manuscript.

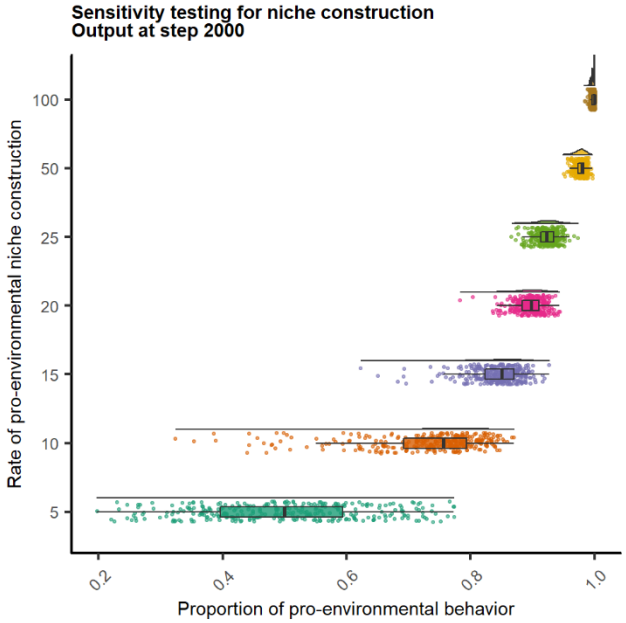


Figure S10. Sensitivity test 4. The network density (minimum degree of connection, or *m0* in the Klemm-Eguíluz model) has a notable effect on outcomes in pro-environmental behaviours. The reasoning is intuitive: When networks are denser, more social learning and transmission occurs, which leads to more polarized end results as the society of agents converges into a uniform behavioural unit or culture (notice how the density distribution of degree connection 20 approaches what seems like a bimodal distribution).

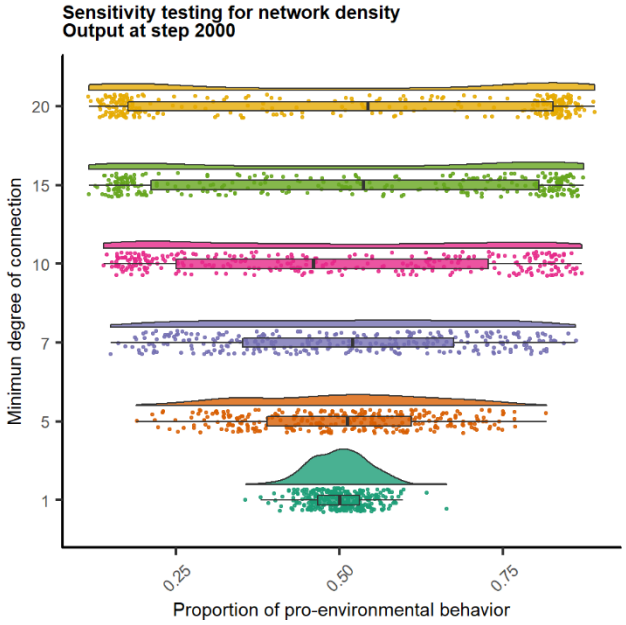


Figure S11. Sensitivity test 5. Importantly, the model is robust against the total number of agents. Due to computational constraints, we do not run the model with over 1000 agents. When the model has over 100 agents, the results are similar. The default value for number-of-agents, 300, can thus be justified.

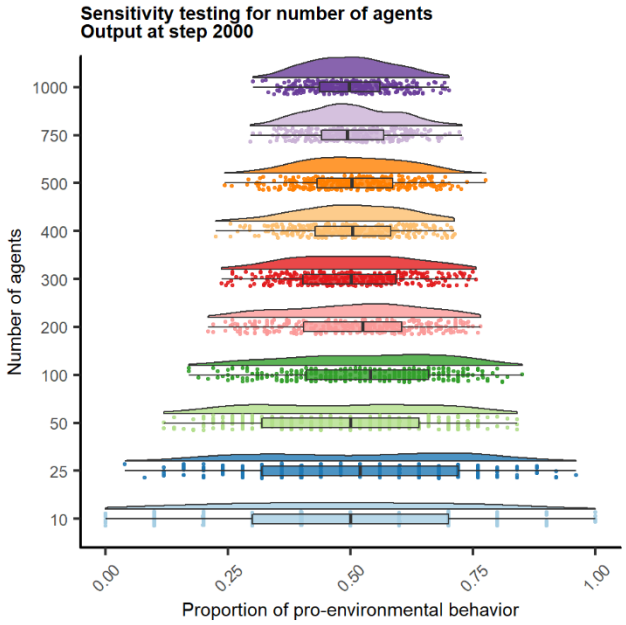


Figure S12. Sensitivity test 6. The effect of initial pro-environmental personal states on the outcome of the model is considerable, and similar in logic to the initial composition of affordances (Figure S7). Notice, however, that in global sensitivity testing, this effect is shown to be less robust when other parameters are allowed to vary.

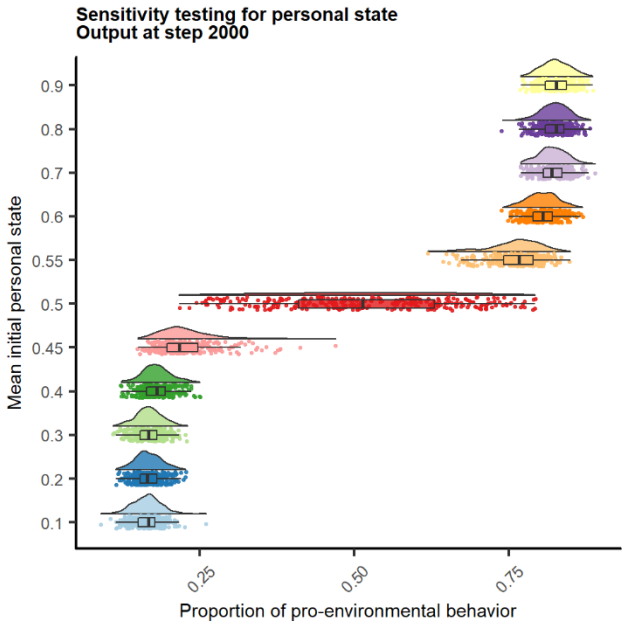
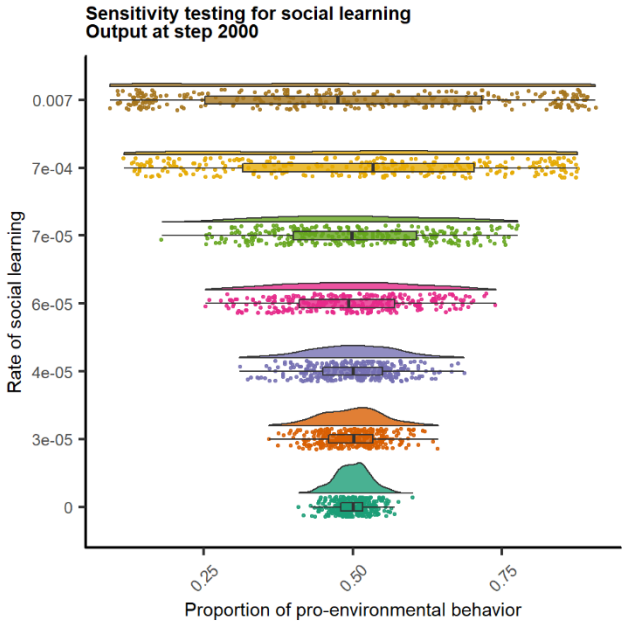


Figure S13. Sensitivity test 7. Similarly to Figure S10 (network density), the rate of social learning has a considerable effect on model outcomes, particularly at extreme values (i.e., ten- or twentyfold to the rate used in the Results section). The model is quite robust against more moderate changes in the rate of social learning. Again, the reasoning is intuitive: The more the rate of social learning is increased, the more social transmission occurs, which leads to more polarized end results as the society of agents converges into a uniform behavioural unit or culture.



Global sensitivity analysis: Latin hypercube sampling

We use Latin hypercube sampling (LHS) as our method for global sensitivity analysis. LHS ensures that each of the model's input variables have all portions of their distribution represented by input values [23]. LHS is simply a K-dimensional extension of Latin square sampling (ibid.), and is commonly used for global sensitivity testing [24]. See e.g. [24] or [23] for more details on LHS. We use the R package nlrx [11] to generate our Latin hypercube samples. We sample our input values from the ranges specified in Table S1. The values were selected on the basis of the OFAT sensitivity tests. We excluded extreme parameter values (which would lead to very predictable and extreme model results, such as when pro-amount is close to 1), but still allow the model to run on a wide range of input values.

Table S1. Parameter ranges for global sensitivity analysis.

Model parameter	**Range**
number-of-agents	[100, 1000]
social-learning	[0.0002, 0.0008]
asocial-learning	[0.0002, 0.0008]
pro-amount	[0.33, 0.66]
initial-pro	[0.33, 0.66]
initial-non	[0.33, 0.66]
construct-non	[0, 10]
construct-pro	[0, 10]
network-param	[3, 7]
mu	0.9

Figure S14. Sensitivity test 8. 300 parameter sets are sampled from the ranges specified in Table S1. The model is run 5 times on each parameter sample, with a different random seed. The lines in this plot illustrate the range of the outcomes of each parameter sample, from min value to max value. Overall, the model has a clear tendency of converging to a state of either high or low pro-environmental behaviour. This is unsurprising, given the results seen in Figures 2–4. This effect will be less drastic if the model is run for less than 2000 ticks or if the range of parameters such as pro-amount is decreased.

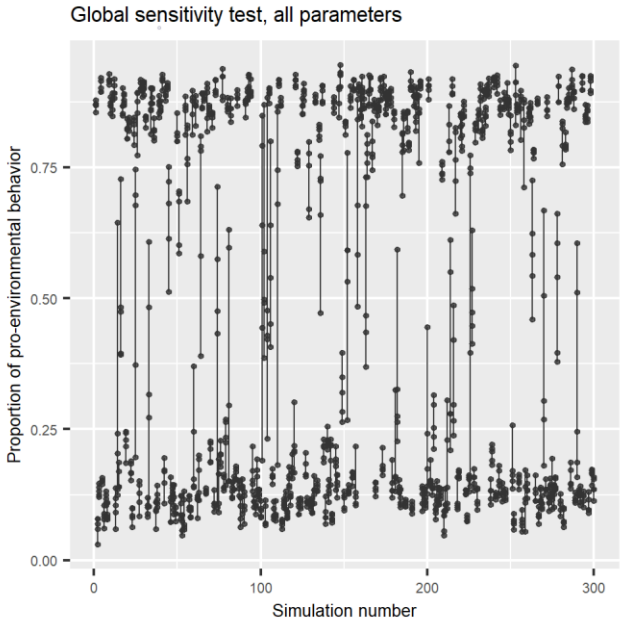


Figure S15. Sensitivity test 9. Even when all other parameters are allowed to vary freely, the nonlinear effect of pro-environmental affordances on pro-environmental behaviour remains. This figure therefore illustrates that the phase transition effect seen in Figures 2A and 2B is very robust.

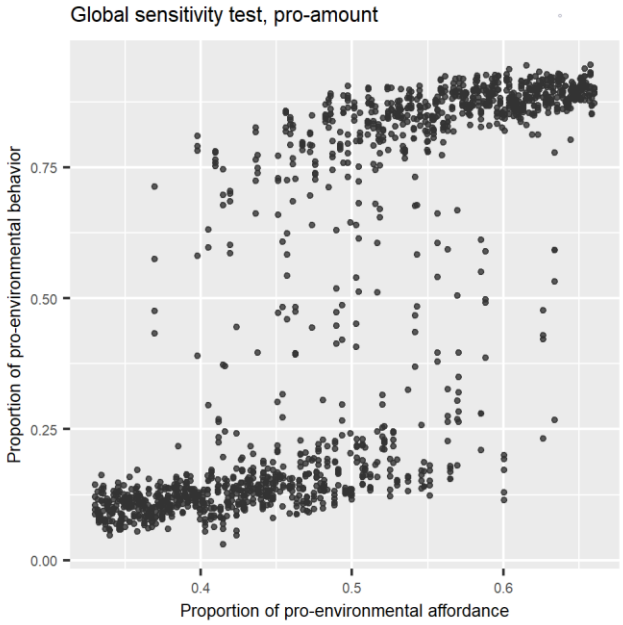
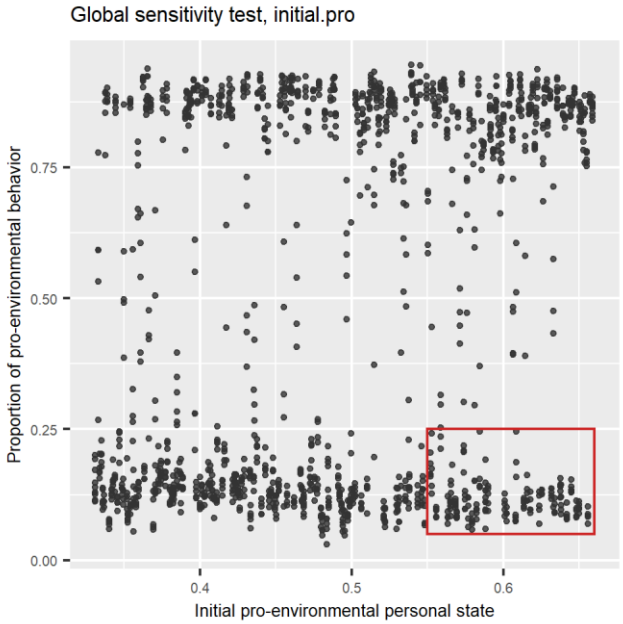


Figure S16. Sensitivity test 10. When other parameters are allowed to vary, initial-pro (the mean initial pro-environmental personal state) has a less apparent effect on behaviours than seen in Figure S12 (where an OFAT test was run on initial-pro). Notice how initial pro-environmental personal states often do not translate into sustained pro-environmental behaviour (highlighted by the red box). This is most likely because of either a lack of pro-environmental affordances, or the interference of a high initial-non value (i.e., counteracting personal states).



Supplemental References

1. Wilensky, U. (2010). NetLogo. https://ccl.northwestern.edu/netlogo/.

2. R Core Team (2019). R: A language and environment for statistical computing. https://www.r-project.org/.

3. RStudio Team (2015). RStudio: Integrated development for R. http://www.rstudio.com.

4. Wickham, H. (2017). The tidyverse. https://www.tidyverse.org/.

5. Kassambara, A., and Mundt, F. (2016). Factoextra: Extract and visualize the results of multivariate data analyses. https://cran.r-project.org/web/packages/factoextra/index.html.

6. Harrell Jr, F.E., and Dupont, C. (2008). Hmisc: Harrell miscellaneous. https://cran.r-project.org/web/packages/Hmisc/index.html.

7. Wickham, H. (2011). The split-apply-combine strategy for data analysis. J. Stat. Softw. *40*, 1–29.

8. Neuwirth, E. (2014). RColorBrewer: ColorBrewer palettes. R package version 1.1-2. https://cran.r-project.org/package=RColorBrewer.

9. Wickham, H. (2012). reshape2: Flexibly reshape data: a reboot of the reshape package. https://cran.r-project.org/web/packages/reshape2/index.html.

10. Auguie, B., and Antonov, A. (2017). gridExtra: Miscellaneous Functions for "Grid" Graphics. https://cran.r-project.org/web/packages/gridExtra/index.html.

11. Salecker, J., Sciaini, M., Meyer, K.M., and Wiegand, K. (2019). The nlrx r package: A next-generation framework for reproducible NetLogo model analyses. Methods Ecol. Evol. *10*, 1854–1863.

12. Grimm, V., Berger, U., Bastiansen, F., Eliassen, S., Ginot, V., Giske, J., Goss-Custard, J., Grand, T., Heinz, S.K., and Huse, G. (2006). A standard protocol for describing individual-based and agent-based models. Ecol. Model. *198*, 115–126.

13. Grimm, V., Berger, U., DeAngelis, D.L., Polhill, J.G., Giske, J., and Railsback, S.F. (2010). The ODD protocol: A review and first update. Ecol. Model. *221*, 2760–2768.

14. Rietveld, E., and Kiverstein, J. (2014). A rich landscape of affordances. Ecol. Psychol. *26*, 325–352.

15. Lewin, K. (1936). Principles of topological psychology (Read Books Ltd).

16. Chan, K.K.W. (1996). Environmental attitudes and behaviour of secondary school students in Hong Kong. Environmentalist *16*, 297–306.

17. Klemm, K., and Eguiluz, V.M. (2002). Highly clustered scale-free networks. Phys. Rev. E *65*, 036123.

18. Caparrini, F.S. (2018). Complex Networks Toolbox (NetLogo). http://www.cs.us.es/~fsancho/?e=162.

19. Prettejohn, B.J., Berryman, M.J., and McDonnell, M.D. (2011). Methods for generating complex networks with selected structural properties for simulations: A review and tutorial for neuroscientists. Front. Comput. Neurosci. *5*.

20. Mesoudi, A. (2011). Cultural Evolution (University of Chicago Press).

21. Broeke, G.T., Voorn, G.V., and Ligtenberg, A. (2016). Which sensitivity analysis method should I use for my agent-based model? J. Artif. Soc. Soc. Simul. *19*.

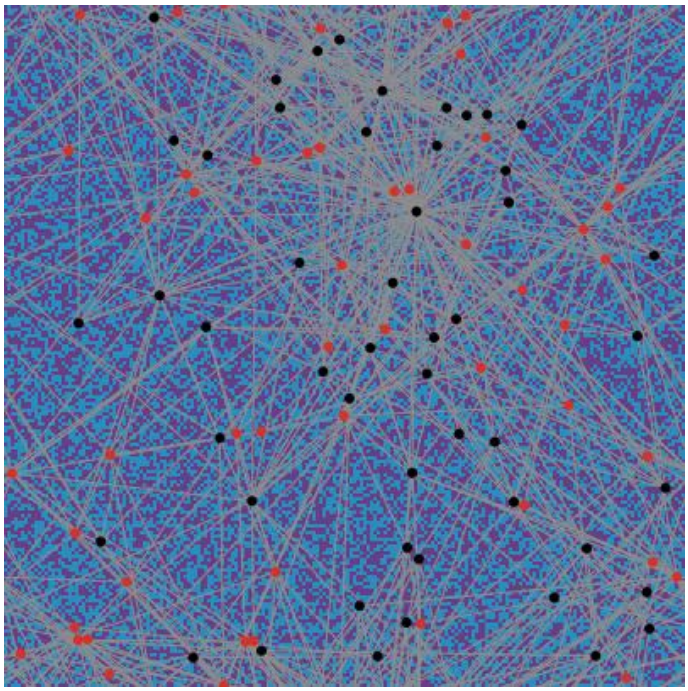
22. Allen, M., Poggiali, D., Whitaker, K., Marshall, T.R., and Kievit, R. (2018). Raincloud plots: A multi-platform tool for robust data visualization. PeerJ Prepr. *6*.

23. McKay, M.D., Beckman, R.J., and Conover, W.J. (1979). Comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics *21*, 239–245.

24. Alden, K., and Coles, M. (2014). Easing parameter sensitivity analysis of NetLogo simulations using SPARTAN. In Artificial Life 14: Proceedings of the Fourteenth International Conference on the Synthesis and Simulation of Living Systems (The MIT Press), pp. 622–628.

Supplemental Figures

Figure S17. A screenshot of the spatially explicit NetLogo model. Here, 100 agents (circle-shapes) are connected to each other in a Klemm-Eguíluz network. Agents coloured in black are more pro-environmentally than non-environmentally disposed, and vice versa for agents coloured in red. The network is represented with grey links connecting the agents. Notice how some agents are much more connected than others. The environment consists of two kinds of patches, pro-environmental affordances (violet) and non-environmental affordances (sky-blue). Agents move around the grid in a random walk. The torus-shaped world wraps around horizontally and vertically.



Supplemental Tables

Table S2. Parameters. The model's parameters, descriptions of parameters, and ranges of possible parameter values.

Model parameter	Description	Possible range
number-of-agents	Total number of agents.	[1, 1000]
social-learning	Rate of social transmission of behaviour.	[0, 1]
asocial-learning	Rate of individual learning and habituation.	[0, 1]
pro-amount	Initial proportion of pro-environmental affordances in the landscape of affordances.	[0, 1]
initial-pro	Defines the initial pro-environmental personal state, pro-env, which is the probability of interacting with pro-environmental affordances when encountered.	[0, 1]
initial-non	Defines the initial non-environmental personal state, non-env, which is the probability of interacting with non-environmental affordances when encountered.	[0, 1]
construct-non	Probability of constructing a non-environmental affordance.	[0, number-of-agents]
construct-pro	Probability of constructing a pro-environmental affordance.	[0, number-of-agents]
network-param	*m0* in the Klemm-Eguíluz model [17]. Defines the initial complete graph in the network generating algorithm.	[1, number-of-agents]
mu	μ in the Klemm-Eguíluz model [17]. Probability of connecting with low degree nodes. Alters the clustering coefficient of the network. [18]	[0, 1]

Table S3. Parameter values for the abstract model run.

Model parameter	**Value**
number-of-agents	300
social-learning	0.00007
asocial-learning	0.00005
pro-amount	[0, 1]
initial-pro	0.5
initial-non	0.5
construct-non	0 or 10
construct-pro	0 or 10
network-param	5
mu	0.9

Table S4. Parameter values for the Copenhagen simulation.

Model parameter	**Value**
number-of-agents	300
social-learning	0.00007
asocial-learning	0.00005
pro-amount	0.4
initial-pro	0.2
initial-non	0.8
construct-non	0
construct-pro	5
network-param	5
mu	0.9