NETLOGO CODE FOR THE SOCIAL-ECOLOGICAL PLANT INVASION AGENT-BASED MODEL (SEPIM)

SEPIM is a model for exploring how landowners' social factors can influence the success of invasive species control, in terms of landowners' collective ability to contain the invasive plant species spread or eradicate it at landscape level.

SEPIM is associated with the publication "Yletyinen, J., Perry, G.L.W., Burge, O., Mason, N., Stahlmann-Brown, P. 2021. Invasion landscapes as social-ecological systems: role of social factors in invasive plant species control. People and Nature (*in press*)."

The model was developed by Johanna Yletyinen and George L.W. Perry with financial support from New Zealand Ministry of Business, Innovation and Employment through the Winning Against Wildings research programme.

Uploaded in April 2021.

```
extensions [nw csv rnd matrix rnd]


breed [farmers farmer ]

undirected-link-breed [white_links white_link]
directed-link-breed [blue_links blue_link]

globals
[
  run_seed
  values-weight2
  social-network-weight2
  advice-weight2
  view-finance-weight2
  view-env-weight2
  view-easy-weight2

;landscape:
  farm_list
  abund_farm
  world-size
  farmer_spl
  sample_abund_farm
  count_farms
  perimeter_set
  to_fill
  LC_list

;social networks
  linked_farmers ;farmers linked to others in the same region
  linked_nfarmers ;neighbour links

;baseline
  bl_invasion
  bl_dense
  bl_moderate
  bl_sparse
  baseline_dense
  baseline_moderate
  baseline_sparse
  baseline_sources
  count_plantations
  plantations
  clean
  success_t
  eradication

;invasion
  inv_area
  forestry_area
  inv_change
  max_invasion
  min_invasion
  invasion_VAR_list
  sparse
  moderate
  dense
  farm_invasions
  matrix
  source_age
  invaded
  uninvaded
  available
  unavailable
  matrix_U
  matrix_S
  matrix_M
  old_invasion
  year1_spread
  years_increase
  years_decrease
  fast_spread
  fast_loss
  plantation_area

;control
  max_control
  low_control
  no_control
  ctr_farmers
  control_VAR_list
  control_t_list
  control_times
  decision_makers
  target_group
  non_target_group

;outcome variables
  inv_increase
  never_ctr
  participation
  unaffected_farmers
  median_inv_n
  control_VAR
  invasion_VAR
  dense_change
  moderate_change
  sparse_change
  most_invaded_farms
  longest_control
  persistent
  max_inv_n
```

```
  source_count
  resource_barrier
  median_resource_barrier
  max_resource_barrier
]

farmers-own [
;farm and invasion, control
  my_farm
  has_wildings
  baseline_control
  nhb_farm_patches
  nhb_farm_ids
  my_neighbors
  control_when
  when_own_since
  high_country ;invasion / land cover
  hill_country
  pastures
  tussock
  bush
  river_coast
  time_lag
  control ;controls wildings: 1 = yes, 0 = no
  has_plantation
  control_t
  control_yrs
  total_ctr_c
  n_inv
  inv_t
  detection
  no_resources
  control_diff2_temp

;network
  degree_adjoin
  degree_region

;behavioural factors and related actor attributes
  my_nfb
  effect_ngh_inc
  effect_ngh_dec
  ;effect_ngh_forest
  nw_influence
  control_diff2
  control_prob
  advice
  affect_env
  affect_finance
  affect_ease
  area
  view_f
  view_env
  view_ease
  control_diff
  help
  risk
  patience
  values
  target_g
  survey_ID
]


patches-own [
;baseline
  land_cover
  owner_id
  contains_wildings?
  baseline_wildings?
  baseline_control_patch?
  density
  other_land_use?
  when_own_since_p
  plantation
  farm_size

;invasion
  spread-cone1S
  spread-cone2S
  spread-cone3S
  spread-cone1M
  spread-cone2M
  spread-cone3M
  spread-cone1D
  spread-cone2D
  spread-cone3D
  d_class
  age
  earliest_arrival
  after_control
  counter_D
  counter_M
  counter_S
  source_distance
  free
  n_invasions
  eradicated
]


;;;;;;;;;;;;;;;;;;;;;;;;;;;; SET-UP ;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

```
to setup
  clear-all

 set run_seed new-seed
 random-seed run_seed

;;FOR THE 2019 WILDINGS STUDY, WEIGHTS FOR SOME OF THE FACTORS CHANGE SIMULTANOUSLY:
;  ; 1) BOTH NEIGHBOUR FEEDBACK AND SOCIAL NETWORK INFLUENCE USE "SOCIAL-NETWORK-WEIGHT"
;  ; 2) BOTH ADVICE AND FUNDING USE "ADVICE-WEIGHT"
;; FUNDING IS INCLUDED IN THE COUNT_PROB, I.E. INTO DECISION-MAKING SIMILARLY TO OTHER FACTORS
 let view_sum view-finance-weight + view-easy-weight + view-env-weight
 let wgt_sum view_sum + social-network-weight + advice-weight +  values-weight ;;;
 set social-network-weight social-network-weight / wgt_sum
 set advice-weight advice-weight / wgt_sum
 set values-weight values-weight / wgt_sum
 set view-finance-weight view-finance-weight  / view_sum
 set view-env-weight view-env-weight /  view_sum
 set view-easy-weight view-easy-weight /  view_sum

;Lists for calculating variance for participation in control and invasion area
  set control_VAR_list []
  set invasion_VAR_list []
  set LC_list []

  setup_farmers
  setup_patches farmer_spl
  fill_landscape
  setup_baseline_wildings
  define_neighbors
  links_to_elsewhere
reset-ticks
end


to setup_farmers
;Farmers are created in relation to the size of world (landscape area), one patch is either 2 ha. Actor attributes are allocated from the Surv
  ;;;print "create farmers"
  file-close-all
  if not file-exists? "Actor_attributes_CONIFERS.csv"[
    user-message "No such file!"
    stop ]
  let data csv:from-file "Actor_attributes_CONIFERS.csv"
  set farmer_spl n-of number-of-farmers n-values length data [i -> i + 1]
  let i 0
  while [i < number-of-farmers]
  [
    let farmer_data (item ((item i farmer_spl) - 1) data)

  create-farmers 1 [
    set survey_ID item 0 farmer_data ;ID
    set has_wildings item 2 farmer_data ;has wildings on own land
    set control item 3 farmer_data ;controls wildings
  set degree_adjoin item 4 farmer_data ;social network links to neighbors
  set degree_region item 5 farmer_data ;social network links to other farmers in the area
  set high_country item 8 farmer_data ;whether the farmer has wildings on this land cover, density of invasion
  set hill_country item 7 farmer_data ;whether the farmer has wildings on this land cover, density of invasion
  set pastures item 9 farmer_data ;whether the farmer has wildings on this land cover, density of invasion
  set tussock item 10 farmer_data ;whether the farmer has wildings on this land cover, density of invasion
  set bush item 6 farmer_data ;whether the farmer has wildings on this land cover, density of invasion
  set river_coast item 11 farmer_data ;whether the farmer has wildings on this land cover, density of invasion
  set risk item 18 farmer_data ;willingness to take risks
  set patience item 17 farmer_data ;degree of patience
  set my_nfb 0 ;preparation for calculating neighbour feedback
  set effect_ngh_inc item 14 farmer_data ; if neighbours increase control: 1 = I will decrease mine, 2 = no change, 3 = increase. Note that
  set effect_ngh_dec item 12 farmer_data ; if neighbours decrease control: 1 = I will decrease mine, 2 = no change, 3 = increase. Note that
  set advice item 15 farmer_data ;would additional information or advice be helpful. 1 = yes, 2 = no
 ; set effect_ngh_forest item 13 farmer_data ;what does if neighbours plant forest - not in use
  set has_plantation item 16 farmer_data ;has forest plantation
  set affect_env item 20 farmer_data ;views on wildings: environment
  set affect_finance item 21 farmer_data ;views on wildings: finances
  set affect_ease item 19 farmer_data ;views on wildings: farming
  set control_when item 22 farmer_data ;since when has been controlling
  set when_own_since item 23 farmer_data ;since when noticed wildings
  set area item 24 farmer_data ;farm size
  set control_diff item 25 farmer_data ;perceived difficutly of controlling wildings
   ]
   set i i + 1
]
  file-close

;Prepare control-related variables
 ask farmers [
    set label control_diff
    set total_ctr_c 0
    ask farmers with [control = 1] [set control_t control_yrs]
    set detection 0

; Convert Survey Data on views on conifers and personal values to values fit for the model. These variables don't change during the model run.
; All behavioral factors produce a value 0 - 1, in which higher value indicate higher probability for controlling wildings.
    set view_f (affect_finance / 10) ;to convert from 0 - 10 (strongly negative impact to strongly positive impact) to 0 - 1
    set view_env (affect_env / 10)
    set view_ease (affect_ease / 10)
    set risk (risk / 10) ;this variable is 0 - 10, in which 10 is fully prepared to take risks
    if patience = 2 [set patience 1] ;two in data means highest patience, 50 lowest. So 2,10,20,30,40,50. Converted here: 1 is the highest pat
    if patience = 10 [set patience 0.8]
    if patience = 20 [set patience 0.6]
    if patience = 30 [set patience 0.4]
    if patience = 40 [set patience 0.2]
    if patience = 50 [set patience 0]
    set values (risk + patience) / 2 ;divide with max (risk + patience = 2) to get the percentage. Values are in the 2019 wildings manuscript
    ;The higher Values is, the less the farmer wants to take risks and the more patient he is, i.e. more keen to control.
     ]

;If help and advice is > 0, it has to be set in the setup, or it will add "help" to farmers on every tick.
```

```
;Control_diff is 0 - 1, in which 0 represents extremely difficult and 1 easy
;Selective funds is funding targeted to farmers with wildings in reproductive age (i.e. prone to spread).

  if control-action? [
    ;;;print "help and advice"

      if not selective-funds? [
     ask farmers [
        set control_diff2 control_diff
    ]]

    ask farmers [ifelse advice = 1
           [set help 1]
      [      set help 0]]
  ]

ask farmers [
    set size 2
    set color white
    set shape "dot"
  ]
end


;Farms are clusters of patches, allocated to farmers in relation to the standardized sizes of their farms (Survey Data: farm area) until all p
to setup_patches [spl]
  ;;;print "create farms"
  set farm_list n-values number-of-farmers [i -> i]
  set abund_farm reduce sentence csv:from-file "farm_size_conifers.csv" ;data on farm sizes
  set sample_abund_farm []
  ifelse length abund_farm > number-of-farmers [
    foreach spl
    [
      x ->
        set sample_abund_farm lput (item (x - 1) abund_farm) sample_abund_farm
    ]]
  [ set sample_abund_farm abund_farm ]
  let abund_sum sum sample_abund_farm
  set sample_abund_farm (map [ [a] -> a /  abund_sum ] sample_abund_farm)
  set count_farms n-values number-of-farmers [1]
  ask patches [set land_cover -1]
    foreach farm_list
    [ x -> ask one-of patches with [land_cover = -1]
      [ set land_cover x]
    ]
  set perimeter_set (patch-set [neighbors4 with [land_cover = -1]] of patches with [land_cover != -1 ])

  set world-size world-width * world-height
end


to fill_landscape
  let habitat_weights (map list farm_list sample_abund_farm)
  set to_fill count patches with[land_cover = -1]
  while [ any? patches with [land_cover = -1]]
  [ let h first rnd:weighted-one-of-list habitat_weights [[p] -> last p]
      let target one-of perimeter_set with [count neighbors with [land_cover = h] >= -1]
       if target != nobody
       [ ask target
         [ set land_cover [land_cover] of one-of neighbors4 with [land_cover != -1]
           let new_edge_cells neighbors4 with [land_cover = -1]
           set perimeter_set (patch-set (other perimeter_set) new_edge_cells)
         ]
      ]
  ]

  ask farmers
  [ let my_id who
    set my_farm patches with [land_cover = my_id]
    move-to one-of my_farm
  ]

  ask farmers [
    let id survey_id
    ask my_farm [set owner_id id
      set farm_size count patches with [owner_id = [owner_id] of myself ] ]]
end


;Survey Data on wildings invasion is reported as density of invasion / land cover. A baseline invasion in the landscape is created by using pr
;Canterbury New Zealand region (2012 values https://www.lawa.org.nz/explore-data/land-cover/ accessed in November 2018, stored in the code as
;covers on each farm, based on : 1) for each land cover, get a number from the random binomial. 2) Rescale list to sum to number of patches in
;land cover type using that list.

to-report random-binomial [n p]
  report length filter [a -> a] n-values n [random-float 1 < p]
end


to setup_baseline_wildings

  ;;;print "set baseline"
  ask patches [set contains_wildings? false]
  ask patches [set baseline_wildings? false]
  ask patches [set other_land_use? false]

  ;a-g are not needed but kept as a record of values
  let a 0.15 ;tussock Canterbury
  let b 0.45 ;improved pastures CAN
  let c 0.15 ;bush and scrub/shrubland CAN
  let d 0.06 ;water bodies CAN (probably excludes coast)
  let f 0.12 ;represents high country CAN
  let g 0.11 ;hill country (the remaining)
```

```
while [length LC_list <= 6] [
  let LU_p random-binomial 6 0.15 ;0.15 is a mean of a-g
  set LC_list lput LU_p LC_list
  if length LC_list = 6 [
    ;;;print "LC_list made"
  ]
]

if sum LC_list = 0 [setup_baseline_wildings]

ask farmers with [has_wildings = 1 ] [
  let t tussock
  let p pastures
  let hc high_country
  let hi hill_country
  let bu bush
  let r river_coast

  ask my_farm
   [let x farm_size / sum LC_list
    let multiplied_list map [i -> i * x] LC_list

      ask n-of item 1 multiplied_list patches [set density t]
      ask n-of item 2 multiplied_list patches [set density p]
      ask n-of item 3 multiplied_list patches [set density hc]
      ask n-of item 4 multiplied_list patches [set density hi]
      ask n-of item 5 multiplied_list patches [set density bu]
      ask n-of item 6 multiplied_list patches [set density r]]]

     ;;;print "densities set"

;Then mark invasion-specific land uses: baseline invasion, forestry plantation, densities, times when invasion and control began. The baseli
;based on Survey Data. Define variables describing baseline system.
ask patches [
     if density > 0 [
     set contains_wildings? true
      set baseline_wildings? true]]

let for_farmers farmers with [has_plantation = 1]
set count_plantations count for_farmers
ask for_farmers [
  ask my_farm [set plantation 1]]
set plantations patches with [plantation = 1]


 ask farmers [
  let arrival when_own_since
  ask my_farm [set when_own_since_p arrival ;this represents maturity of trees
    if when_own_since_p > 60 [set when_own_since_p 60 - 1 ] ;time period of the model minus 1 since matrix begins from 0
]]

;Forestry planations can be considered either a source of invasion or a landscape barrier to invasion, i.e. does not have invasive conifers.
  if any? for_farmers [
 ask for_farmers
    [set has_wildings 0
      ask my_farm [
       set contains_wildings? false
       set baseline_wildings? false
       set pcolor brown
        ifelse forestry-barrier?
           [set density 0]
           [set density 2]       ;if the forestry species is invasive (i.e. the one modeled), set density to moderate
 ]]]

 ask farmers with [density = 0 and has_plantation = 0] [
      ask my_farm [
        set contains_wildings? false
        set baseline_wildings? false
        set pcolor black]]

 ask farmers with [has_wildings = 1 and control = 1] [
 set color green
 set baseline_control 1
 ;set control 1
 ask my_farm [ set baseline_control_patch? true]
 set total_ctr_c 1 ]

 ask farmers with [has_wildings = 0 and control = 1] [ ;couple of strange cases in Survey Data.
   set after_control random-float 3 ] ;a number assigned randomly since we do not have data for that

 set never_ctr count farmers with [total_ctr_c = 0]

 ask patches with [density > 0 and pcolor != brown]  [
         if density = 3 [set pcolor red]
         if density = 2 [set pcolor orange]
   if density = 1 [set pcolor yellow]]


;Of the landscape (see LAWA ref(, 25 % is in other land use, i.e., can not be invaded or are immediately controlled. These patches represent
;cropping/horticulture, urban/bare/lightly vegetated, water bodies. They have to be patches where there is no turtle. An invasion-farm can't
;become completely gray.
  ask n-of (0.25 * count patches) patches with [not any? turtles-here] [
    set other_land_use? true
    set contains_wildings? false
    set baseline_wildings? false
    set pcolor gray
    set density 0]  ;

 set unavailable patches with [other_land_use?]
 set available patches with [pcolor != brown and pcolor != gray]
```

```
 set bl_invasion count patches with [density > 0 and plantation = 0]
 if bl_invasion = 0 [setup] ;to avoid starting a simulation with no invasion

  set bl_dense count patches with [density = 3] ;these are needed in calculating changes
  set bl_moderate count patches with [density = 2 and plantation = 0]
  set bl_sparse count patches with [density = 1]
  set forestry_area count patches with [plantation = 1]

  set max_invasion count patches with [density > 0 and plantation = 0 ] ;) / count patches ) * 100
  set min_invasion count patches with [density > 0 and plantation = 0] ;) / count patches ) * 100
  set max_control count farmers with [control = 1]
  set baseline_sources count farmers with [has_wildings = 1]


;For invasion ("Spread" procedure), upload invasion model matrices. Invasion matrices are available for two species, one with reproduction age
;However, simulations are performed with one species at a time.

  ifelse repro-age = 7
      [ let matrix_probs (csv:from-file "TransitionYears_repro7_uninvaded.csv")
        set matrix_U matrix:from-row-list matrix_probs
        let matrix_probsS (csv:from-file "TransitionYears_repro7_sparse.csv")
        set matrix_S matrix:from-row-list matrix_probsS
        let matrix_probsM (csv:from-file "TransitionYears_repro7_moderate.csv")
        set matrix_M matrix:from-row-list matrix_probsM]

      [ let matrix_probs1 (csv:from-file "TransitionYears_repro16_uninvaded.csv")
        set matrix_U matrix:from-row-list matrix_probs1
        let matrix_probs2 (csv:from-file "TransitionYears_repro16_sparse.csv")
        set matrix_S matrix:from-row-list matrix_probs2
        let matrix_probs3 (csv:from-file "TransitionYears_repro16_moderate.csv")
        set matrix_M matrix:from-row-list matrix_probs3 ]
 end

;For calculating neighbour feedback, neighboring farms are defined as surrounding farms (patch sets) that the farmer in question does not own.
to define_neighbors
  ;;;print "neighbors"
  ask farmers[
    set nhb_farm_patches no-patches
    let f_id survey_id
    let nhb_farms (patch-set [neighbors] of my_farm) with [owner_id != f_id]
    set nhb_farm_ids remove-duplicates [owner_id] of nhb_farms
    foreach nhb_farm_ids
    [
      i ->
      set nhb_farm_patches (patch-set nhb_farm_patches patches with [owner_id = i])
      set my_neighbors turtles-on nhb_farm_patches]
    ]

  ask farmers with [degree_adjoin > count my_neighbors] [set degree_adjoin count my_neighbors]
  while [any? farmers with [degree_adjoin > count my-links ]] [
      ask farmers with [degree_adjoin > count my-links] [
           create-blue_link-from one-of my_neighbors]
  ]
  ask links [set color blue set thickness 0.1]
end

;Create a binary social network including 1) directed links to neighbours and 2) reciprocal links to any farmers.
;The networks are random graphs with degree sequences based on Survey Data.
to links_to_elsewhere
;This code follows: https://stackoverflow.com/questions/33000956/netlogo-efficiently-create-network-with-arbitrary-degree-distribution
;and https://stackoverflow.com/questions/32967388/netlogo-efficient-way-to-create-fixed-number-of-links (both sites accessed in November 2018)

  set linked_farmers farmers with [degree_region != 0]
  ask linked_farmers [
    if degree_region >= count linked_farmers [set degree_region count linked_farmers]
    let new_links degree_region - count my-white_links
    set linked_farmers other linked_farmers
    if new_links > 0 [
      let chosen n-of min (list new_links count linked_farmers) linked_farmers
      create-white_links-with chosen
      ask chosen [if count my-white_links = degree_region [set linked_farmers other linked_farmers]]]]
  ask white_links [set color white set thickness 0.1]

  ;to prepare for tick 0:
  count_sn_influence
  count_nghb_fb
end


;;;;;;;;;;;;;;;;;;;;;;;;;;;; GO ;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;The model goes through the following stages on each tick: 1) invasion, i.e. wildings spread, 2) control of wildings: detect wildings, calcula
;eradicate wildings on own farm (i.e. control), 3) update variables 4) next time step (tick). The model stops after 60 years (60 ticks). If co
;stage 2) is excluded.


to go
if ticks >= 30 [
 calculate_outcomes
 stop
]

spread

if control-action? [
    detect
    count_prob
    take_action
    ]

  if not control-action? [
    ask patches with [density = 0] [set pcolor black]
    ask patches with [other_land_use?] [set pcolor gray
```

```
                                    set density 0]]
tick
end

to update_timer
  if seedlings-unnoticed? [ask patches with [eradicated = 1] [set after_control (after_control + 1)]]
end


;Wildings invasion is based on Burge & Mason's NZ wildings invasion model.
to spread

  if ticks >= 1 [ ask farmers [set control 0]] ;Farmers eradicate all wildings at once, and make new decision each year. Hence, in the beginni
 ;have returned to "no control" state. Social network influence and neighbour feedback influence are calculated in the end of the year, i.e. c
 ;at this state does not affect social diffusion.

;Starting status for patches is recorded to be able to detect change.
  ask available [
    ifelse density <= 0
    [set free  1]
    [set free  0 ]]

  ;;;print "invasion starts"

;When farmers control invasion, they may not notice the smallest seedlings (i.e. failure in early detection). Thus, despite control, widlings
;and grow, i.e. wildings invasion emerges without new output from invasion source patches (coloniser). When these seedlings are over 3 years,
  if seedlings-unnoticed? [
    ask available with [ after_control > 0 ] [ ;Invasion originating from failure in early detection applies only to land that has been under
    if after_control >= 3 [                    ;  conifers are clearly visible at the age of three
 ;DOC: https://www.doc.govt.nz/about-us/science-publications/conservation-publications/threats-and-impacts/weeds/south-island-wilding-conifer-
 ;Seed remains viable in the soil for about three years, though can remain viable in cones for much longer.
      if random-float 1 < random-normal seedlings-probability 0.1 [ ;set in parameter settings (slider), standard error arbitary
            set density 1  ;This invasion is marked as sparse invasion, we assume that control/eradication removes most of the wildings.
            set age after_control
            set eradicated 0 ;invasion is now considered a "normal" invasion, no longer as unnoticed seedlings
  ] ]]]

  set invaded available with [density > 0 ]
  set uninvaded available with [ density <= 0]

;On the first time step, invasion is based on survey data
    if ticks = 0 [
    set old_invasion invaded with [when_own_since_p > 0 ]
    ask old_invasion [
        set earliest_arrival when_own_since_p
     ]
    ask patches with [density > 0] [if when_own_since_p > 0 [set age earliest_arrival]]
  ]


;Only those invasion source patches that have trees in reproduction age can spread seeds.
  set sparse invaded with [density = 1 and age >= repro-age ]
  set moderate invaded with [density = 2 and age >= repro-age ]
  set dense invaded with [density = 3 and age >= repro-age]

;The invasion is dependent on distance downwind and density in source patches and target patches (i.e. colonised land).
;Invasion spreads from source patches into their invasion cones. Years of target cells as invaded from those source densities must be counted
;for enabling transitions to higher density classes. This part uses matrices originating from Mason & Burge model.

;The code first assigns the distances to where invasion spreads and gives the target cells in those distances a

;Counter_x counts transition years, which are stored in input matrices, by setting the arrival to ticks. When ticks - arrival = transition yea
;In transition matrices, rows are starting densities and columns are densities to which a patch shifts when the transition year (values in Exc

  ;for dense invasion in source cells, i.e. where invasion spreads from
  ask dense [
   let cone_set (patch-set patch-at -1 0 patch-at -2 0 )
    ask cone_set [ ; 4 is the highest distance that dense invasion spreads to (see spread-cones), 10 keeps the viewing angle narrow
      if earliest_arrival = 0 [set earliest_arrival ticks]
      if counter_D = 0 [set counter_D ticks ]]
      ask patch-at-heading-and-distance 1 90 [set source_distance 0 ] ;source_distances are simply pointing which row to use in the matrix. ze
      ask patch-at-heading-and-distance 1 90  [set source_distance 1 ]
      ask patch-at-heading-and-distance 2 90  [set source_distance 2 ]]

  ask moderate [
    let cone_set (patch-set patch-at -1 0 patch-at -2 0 )
    ask cone_set [
      if earliest_arrival = 0 [set earliest_arrival ticks]
      if counter_M = 0 [set counter_M ticks]]
    ask patch-at-heading-and-distance 1 90  [set source_distance 0 ] ;source_distances are simply pointing which row to use in the matrix.
    ask patch-at-heading-and-distance 1 90  [set source_distance 1 ]
    ask patch-at-heading-and-distance 1 90  [set source_distance 2 ]]

ask sparse [
 let cone_set (patch-set patch-at -3 0 patch-at -4 0 )
 ask cone_set [
     if earliest_arrival = 0 [set earliest_arrival ticks]
         if counter_S = 0  [set counter_S ticks]]
  ask patch-at-heading-and-distance 1 90  [set source_distance 1 ]
  ask patch-at-heading-and-distance 1 90  [set source_distance 2 ]]

 ;;;print "source_densities set"

  ask moderate [
   let transition matrix:get matrix_M source_distance 3  ;matrix:get matrix row-i col-j, 4th column is 'dense' density class, starts with 0 i
   if (ticks - counter_D) = transition [set density 3 ]]

  ask sparse [
     if (ticks - counter_M) = matrix:get matrix_S source_distance  2 [set density 2 ]
     if (ticks - counter_D) = matrix:get matrix_S source_distance   3 [set density 3 ]]

 ask uninvaded with [source_distance > 0] [
     if (ticks - counter_S) = matrix:get matrix_U source_distance  1 [set density 1 ]
     if (ticks - counter_M) = matrix:get matrix_U source_distance  2 [set density 2 ]
```

```
      if (ticks - counter_D) = matrix:get matrix_U source_distance  3  [set density 2 ]]

;;;print "invasion over"

;Just in case:
ask unavailable [set density 0 ]

;To calculate number of invasions for output variables
ask patches with [free = 1] [
    if density > 0 [
      set n_invasions (n_invasions + 1)
      ask farmers with [who = [land_cover] of myself]
           [set n_inv (n_inv + 1)]
  ]]

;Update of invasion variables.
;Only some patches of the farm may be invaded, but farmers eradicate wildings from the entire farm if they take action. Farm (and farmer) stat
;Only affected farmers make decision on participating in wildings control.

ask available [
    if density > 0
      [set age age + 1
       set contains_wildings? true
        let it earliest_arrival
        let farm_density density
        ask farmers with [who = [land_cover] of myself]
          [set inv_t it
            set has_wildings 1
  ]]]



  set decision_makers farmers with [has_wildings = 1 and has_plantation = 0 ]

  selective_help ;Selective financial help is calculated after updating the age of wildings
end


;Not all farmers detect invasion immediately.
to detect
  ;;;print "detect"
  ask farmers with [has_wildings = 1 and plantations = 0] [
    set time_lag precision (random-normal detection-time 1) 1  ;random-normal reports a normally distributed random floating point number, fir
    if inv_t > ticks ;inv_t is the earliest arrival, this is for baseline invasion farms for which earliest arrival is age of invasion
       [if inv_t > time_lag [
           set detection ticks + 1] ] ;to avoid zero
    if inv_t < ticks - time_lag [  ;for farmers who did not have wildings in the beginning of simulation, earliest arrival is ticks
      set detection ticks + 1]
    set decision_makers farmers with [detection = ticks + 1]]
 if ticks <= 1 [set decision_makers farmers with [control = 1]] ;To avoid not having anyone controlling in the start, we know from Survey Data

end

;For decision-makers, probability to control is calculated as the weighted sum of behavioural factors. While values and views remain the same,
;and neighbour feedback influence are updated on each time step.
to count_prob ;
  ;;;print "count prob"

  ask decision_makers  [
    let avg_d (1 - (mean [density] of my_farm / 3)) ;for this value, higher density should lead to lower probability, hence "1 minus...", thre
    set control_diff2_temp (control_diff2 + avg_d) / 2
    set control_prob (((social-network-weight * nw_influence) + (social-network-weight * my_nfb) + (view-finance-weight * view_f) + (view-env-
    (social-network-weight + social-network-weight + view-finance-weight + view-env-weight + view-easy-weight +  advice-weight  +  values-weig
    ]
end

;Decision-makers' participation in controlling wildings depends on their probability to control. The model first excludes those who don't have
;to control wildings. If a farmer participates in control action, the whole farm becomes wildings free (but see failure in early detection). I
;control widlings, the invasion on their land remains as is. Ecological and social variables are updated.
to take_action

set source_count count farmers with [has_wildings = 1 ]

if control-action? [
;first check if a farmer can afford control. "Too time consuming / expensive" was a reason most frequently given for not controlling wildings.
  ;;;print "can afford?"
  ask decision_makers with [control_diff2_temp < difficulty_TH] [ ;if they have reported it's extremely difficult (0 or 0.1), will not control
   if has_wildings = 1 [
    ; print "no resources"
 set control 0
 set no_resources (no_resources + 1)  ; Counts how many times farmers are affected but have no resources
 set control_t 0] ]                 ;Stops counting control years


;;;print "make decisions"
ask decision_makers with [control_diff2_temp > difficulty_TH] [
        ifelse random-float 1 < control_prob * (random-normal 1 noise-weight) ;If a farmer's probability is in favor of control:

            [set control 1                  ;Start control action.
             set has_wildings 0
             set control_t (control_t + 1)  ; Adds this year to control time counter.
             if seedlings-unnoticed? [
              ask my_farm with [density > 0] [  ;only these patches are marked as eradicated for failure of early detection process
                 set eradicated 1 ]]
             ask my_farm [
             set density 0 ;Wildings eradicated!
             set after_control 0
             set counter_D 0 ;eradication sets counter_d to zero
             set counter_M 0
             set counter_S 0 ]]
            [set control 0 ;If a farmer's probability is not in favor of control, there are no changes in invasion. "Update timer" prodecure
  ]]
]
```

```
    count_sn_influence ;social network influence and neighbour feedback is calculated in the end of invasion-control cycle, when control statu
    count_nghb_fb ;social network influence and neighbour feedback is calculated in the end of invasion-control cycle, when control status has

   ;Here ends the section that only occurs if control action is included.

 ;Update the patch and farmer states for uninvaded farms
  ask patches [if density = 0 [
    set contains_wildings? false
    set pcolor black
    set earliest_arrival 0
  ]]


;Calculate variables and trends
  if ticks > 0 [
 ;;;print "calculate trends"
set inv_area count patches with [density > 0 and plantation = 0] ;
if inv_area > max_invasion [set max_invasion inv_area]
if inv_area < min_invasion [set min_invasion inv_area]
set ctr_farmers count farmers with [control = 1]

ask farmers with [control = 1] [
              set total_ctr_c ( total_ctr_c + 1)
              set participation count farmers - count farmers with [total_ctr_c = 0]
              set never_ctr count farmers with [total_ctr_c = 0]
              set unaffected_farmers count farmers with [n_inv = 0]]

if ctr_farmers > max_control [set max_control ctr_farmers]


set invasion_VAR_list lput inv_area invasion_VAR_list
set control_VAR_list lput ctr_farmers control_VAR_list

let last_year_spread year1_spread  ;Uses the last time tick's spread as past spread
let invasion count available with [density > 0]
    if any? available [set year1_spread (invasion / count available ) * 100
let difference  (year1_spread - last_year_spread)
 ;print "difference" print difference
  if difference > 0 [set years_increase years_increase + 1]
  if difference < 0 [set years_decrease years_decrease + 1]
  if difference > 5 [set fast_spread fast_spread + 1]
      if difference < -5 [set fast_loss fast_loss + 1] ]
if invasion = 0 [ ;if control has succeeded
    set clean clean + 1
    if last_year_spread = 0 and invasion = 0 [ set eradication eradication + 1]
    if eradication > repro-age [if success_t = 0 [set success_t ticks
      ;;;print "WILDINGS-FREE LANDSCAPE!"]
      ]]

    let participants count farmers with [control = 1]
    if participants < 0.2 * number-of-farmers [set low_control low_control + 1]
    if participants = 0 [set no_control no_control + 1]
  ]
;;;print "year ends"
    ]
end


;;;;;;;;;;;;;;;;;;;;;;;;;;;; COUNT BEHAVIORAL FACTORS ;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;Values and Views have been calculated in the setup, in association with actor attributes. This section calculates social network influence an
;that are updated during on each tick. Each behavioral factor results in a value between 0 and 1. The higher the value is, the higher the prob
;wildings control.

;If a farmer has links to adjoining farms, calculate farmer's neighbour feedback by taking a proportion of
;neighboring farms with native forest from the total count of all neighboring farms, and assigning farmer's response from Survey Data.
to count_nghb_fb
   ;;;print "nb fb"
  ask farmers [
 ifelse count my_neighbors with [control =  1] > count my_neighbors with [control = 0] [
      if effect_ngh_inc = 3 [set my_nfb 1] ;increase my control efforts
      if effect_ngh_inc = 1 [set my_nfb 0  ] ;decrease my control efforts
    ]
 [
      if effect_ngh_dec = 3 [set my_nfb 1 ] ;increase my control efforts
      if effect_ngh_dec = 1 [set my_nfb 0  ] ;decrease my control efforts
    ]
]
end

;If a farmer is connected to other farmers, social influence from fellow farmers is calculated. If a number of connected farmers participating
;than number of connected farmers who do not participate, the probability for a farmer to participate increases.
to count_sn_influence
  ;;;print "s nw influence"
  ask farmers [
  ifelse count my-in-links > 0
  [
  let control_friends in-link-neighbors with [control = 1] ;By using in-link here, we exclude farmers who have outdegree but no indegree. This
  ;based on indegree data. Regional links are reciprocal (undirected). Note that in-link-neighbours and my-in-links: "Reports the agentset of
      ;them to the caller as well as all turtles that have an undirected link connecting them with the caller."
   set nw_influence count control_friends / count my-in-links ]
    [ set nw_influence 0 ]  ;If a farmer is not connected to others, set network influence to 0 ]

  ]
end


to selective_help
  if selective-funds? [ ;funding available to farmers with mature wildings. Funding is usually available also for follow up eradication 5-6 ye
    ;that is, however, captured by "unselective" financial help, since it continuous non-stop during simulation.
    ask farmers [set target_g 0] ;to not carry over values from previous time tick
    ask patches [
        if age >= repro-age  [
```

```
              ask farmers with [who = [land_cover] of myself] [
           set target_g 1]]
    set target_group farmers with [target_g = 1]
    set non_target_group decision_makers with [target_g != 1]
  ]

if any? target_group
    [ask target_group [
        set control_diff2 1  ;Change their difficulty category to the category of "extremely easy" + funding weight
        ]]
if any? non_target_group [
              ask non_target_group [
                  set control_diff2 control_diff ] ;Non_target_group keeps the value they had reported in the survey, we are not intervening w
  ]]
end


;Calculate variables for measuring outcomes
to calculate_outcomes
 set values-weight2 values-weight ;needed for NLRX
 set social-network-weight2 social-network-weight
 set advice-weight2 advice-weight
 set view-finance-weight2  view-finance-weight
 set view-env-weight2 view-env-weight
 set view-easy-weight2 view-easy-weight
;end of 2019 change
set inv_area count patches with [density > 0 and plantation = 0] ;
set inv_change (inv_area - bl_invasion)
if control-action?  [set control_VAR variance control_VAR_list ] ;needs a list
if control-action? [set invasion_VAR variance invasion_VAR_list ]
set dense_change count patches with [density = 3]
set moderate_change count patches with [density = 2 and plantation = 0]
set sparse_change count patches with [density = 1]
set farm_invasions [n_inv] of farmers ;invasion frequency
set median_inv_n median farm_invasions ;median invasion frequency
set max_inv_n max farm_invasions ;max invasion frequency
set resource_barrier [no_resources] of farmers
if length resource_barrier > 3 [
set median_resource_barrier median [no_resources] of farmers
    set max_resource_barrier max [no_resources] of farmers ]
set most_invaded_farms count farmers with [n_inv > 10 ] ;farms with over 10 invasion
set control_times median [control_t] of farmers ;number of years farmers control
set longest_control max [control_t] of farmers
set persistent count farmers with [control_t > 10] ;note that if the farmer never stopped control, this includes baseline control years
end
```