2.2 UML diagrams

The following section presents the logic underlying the sub-methods called during the simulation. The order that each method is called is depicted in the 'Main' diagram (Figure 2B).

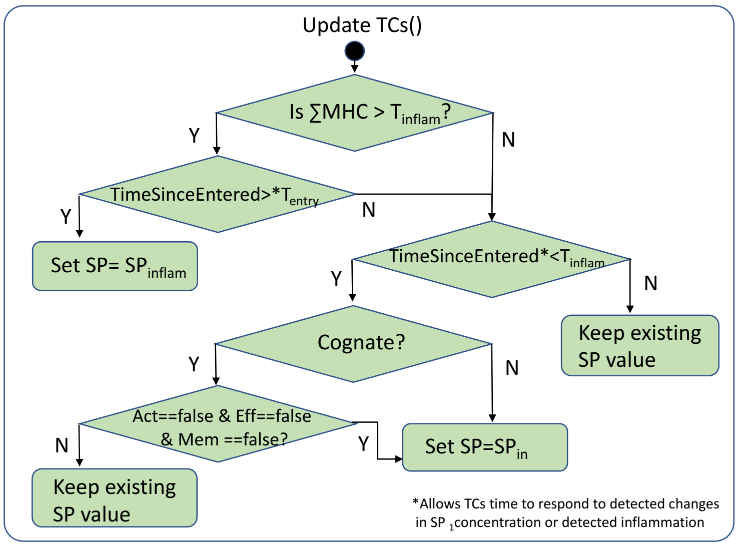
Update TCs()
Is ∑MHC > T_inflam?
Y
N
TimeSinceEntered>*T_entry
Y
N
Set SP= SP_inflam
TimeSinceEntered*<T_inflam
Y
N
Cognate?
Keep existing SP value
Y
N
Act==false & Eff==false & Mem ==false?
N
Y
Set SP=SP_in
Keep existing SP value
*Allows TCs time to respond to detected changes in SP 1 concentration or detected inflammation

Figure S9. UML diagram for 'UpdateTCs'. By checking the time since the TC entered, a 'lag time' representing time for detection of signals and receptor internalisation to occur.

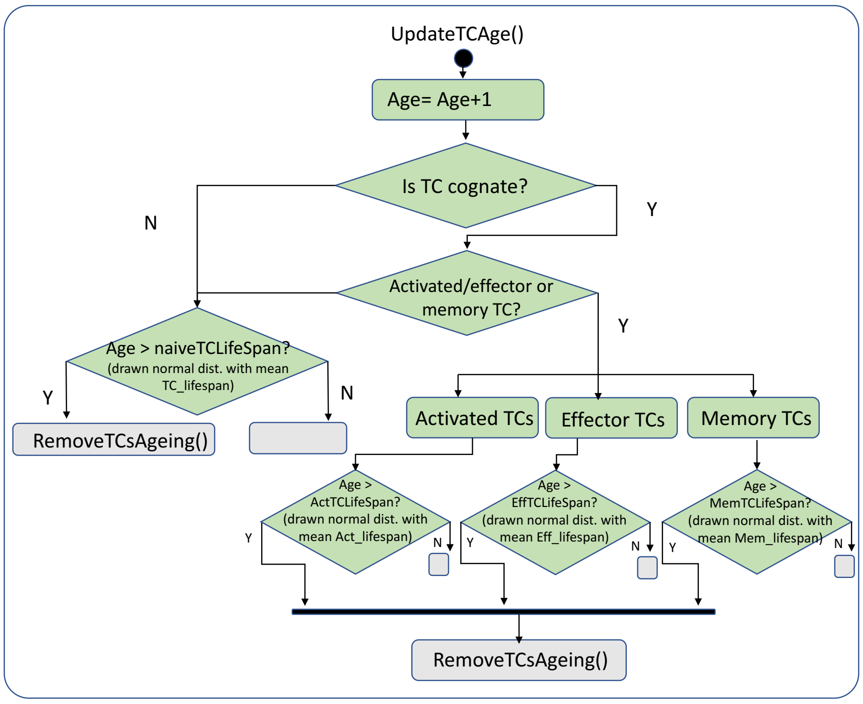
UpdateTCAge()
Age= Age+1
Is TC cognate?
N
Y
Activated/effector or memory TC?
Y
Age > naiveTCLifeSpan? (drawn normal dist. with mean TC_lifespan)
Y
N
RemoveTCsAgeing()
Activated TCs
Effector TCs
Memory TCs
Age > ActTCLifeSpan? (drawn normal dist. with mean Act_lifespan)
Age > EffTCLifeSpan? (drawn normal dist. with mean Eff_lifespan)
Age > MemTCLifeSpan? (drawn normal dist. with mean Mem_lifespan)
Y
N
Y
N
Y
N
RemoveTCsAgeing()

Figure S10. UML diagram for 'UpdateTCsAgeing'.

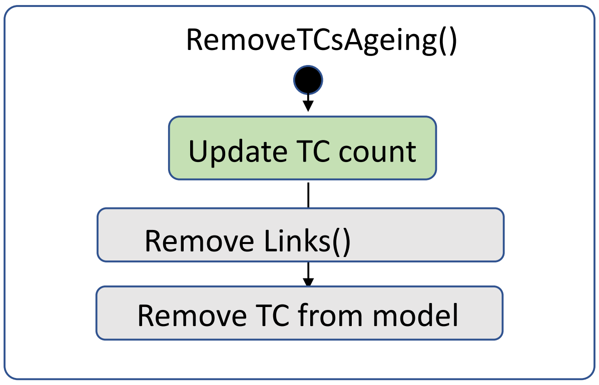

Figure S11. UML diagram for 'RemoveTCsAgeing'. The different types of TCs draw ages from gaussian distributions with a different mean value.

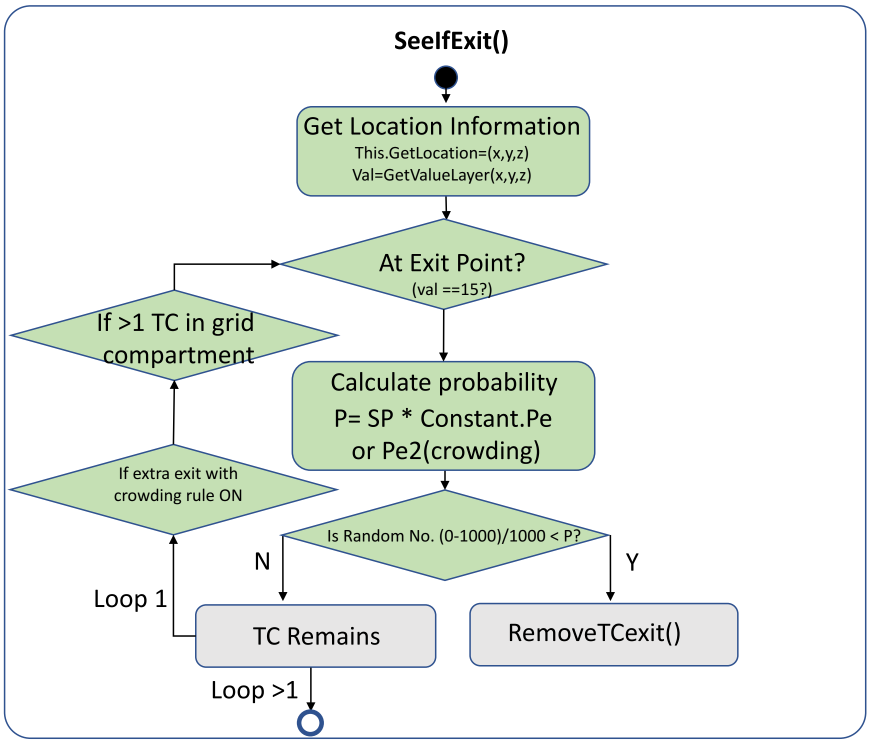

Figure S12. UML diagram for 'SeeIfExit'. The original simulations did not incorporate the second check of TC crowding (crowding rule).

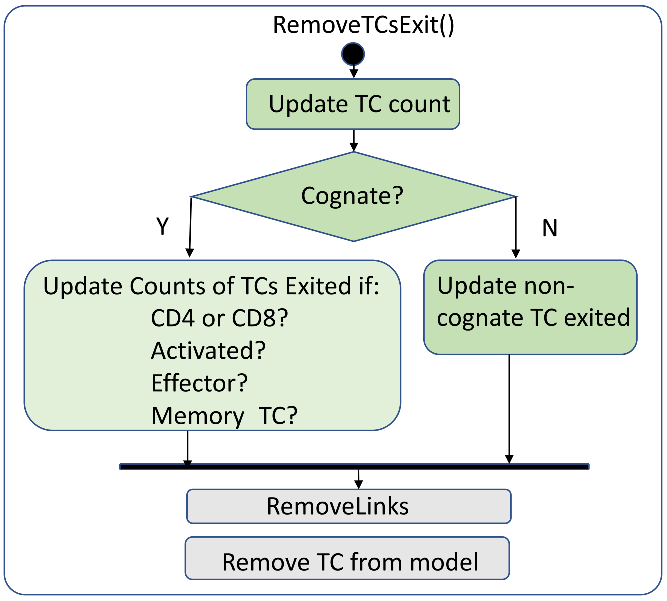

Figure S13. UML diagram for 'RemoveTCExit'. Ageing and TC exit are treated differently due to different counters being updated

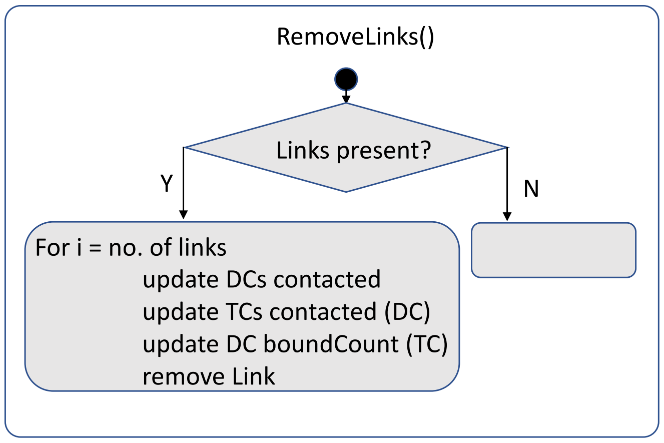

Figure S14. UML diagram for 'RemoveLinks'. Links between TCs and DCs are formed using a projection network, a feature of RepastSimphony, and must be removed when the connection is over.

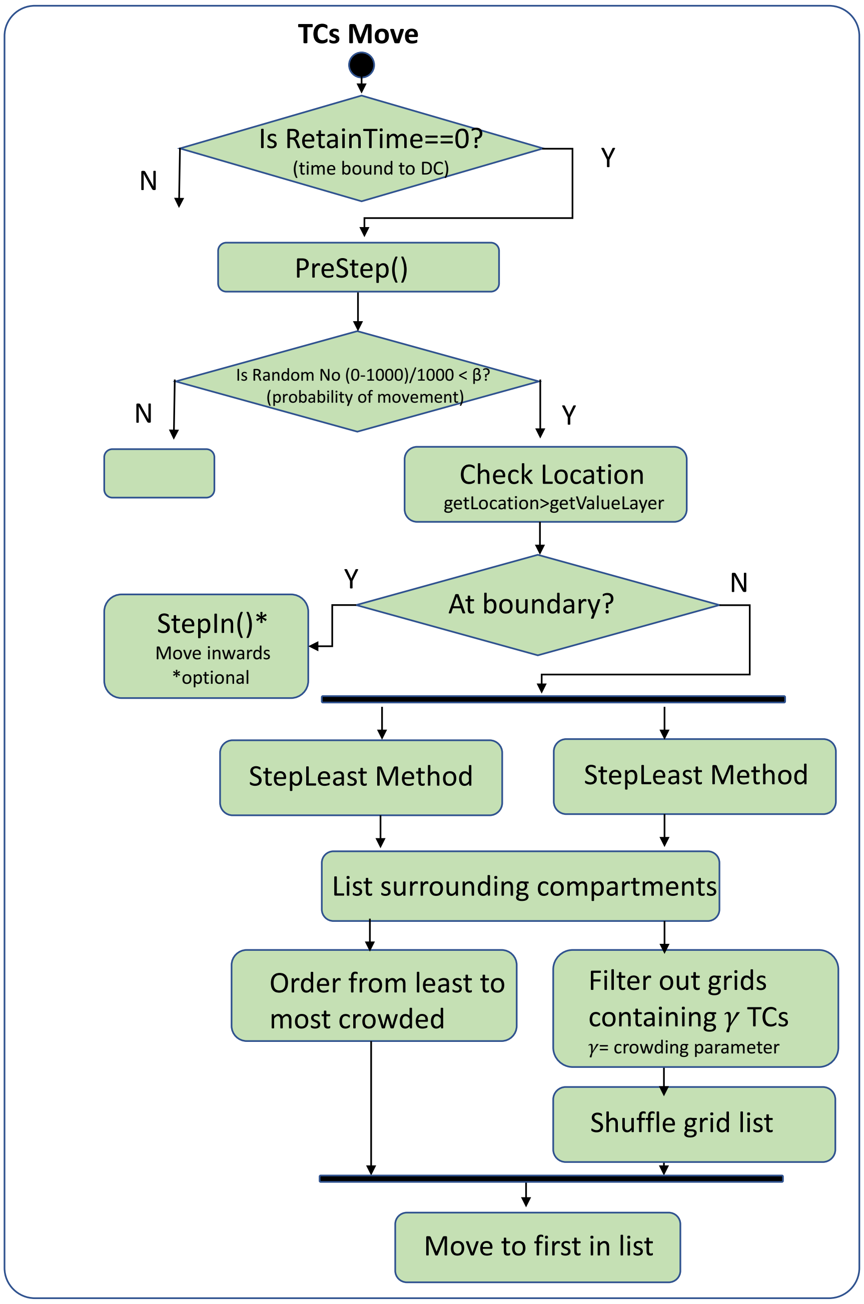

Figure S15. UML diagram for 'TCsMove'. The model has the capacity for different movement options, and maximum TCs per grid is set with crowding parameter γ

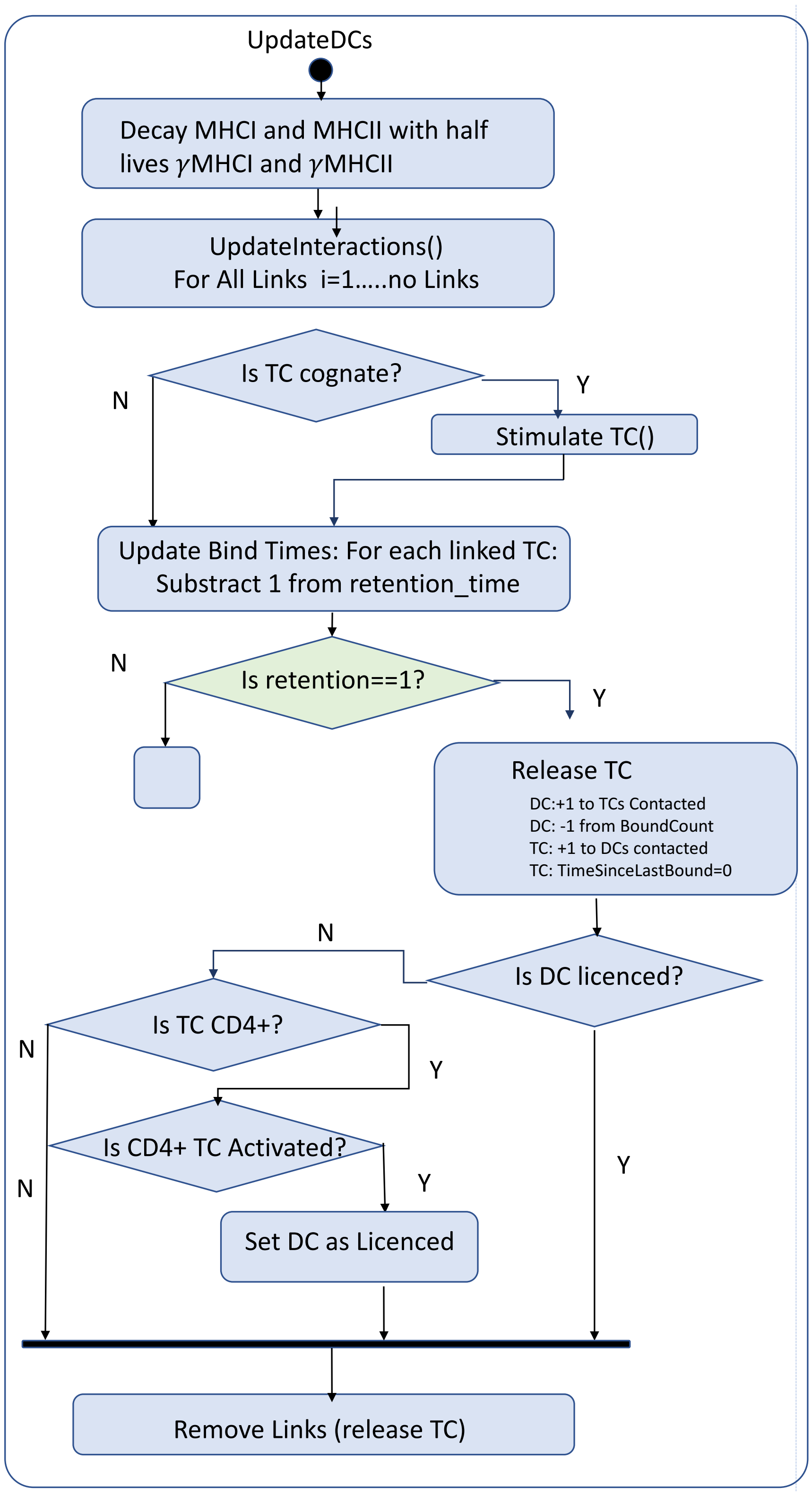

Figure S16. UML diagram for 'UpdateDCs'. DCs enter as mature migrating DCs but may become licenced through interaction with activated CD4+ TCs

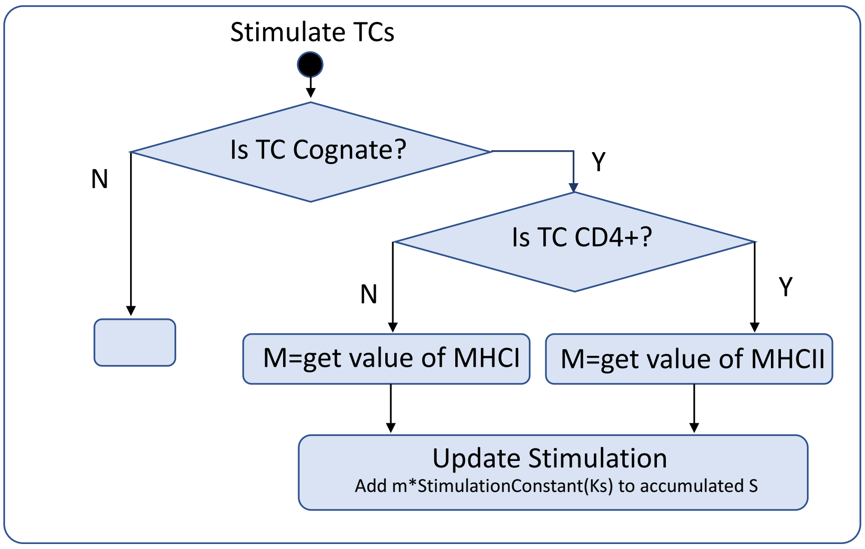

Figure S17. UML diagram for 'StimulateTCs'. DCs can access the variables of the agent attatched via a projection link and thus alter their state based on the DCs MHCI/II presentation and the Stimulation constant, Ks.

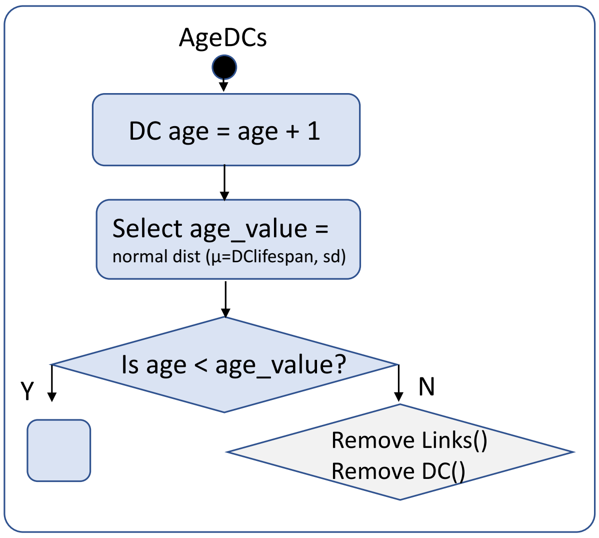

Figure S18. UML diagram for 'AgeDCs'. Mature DCs have a lifespan of around 2.5 days.

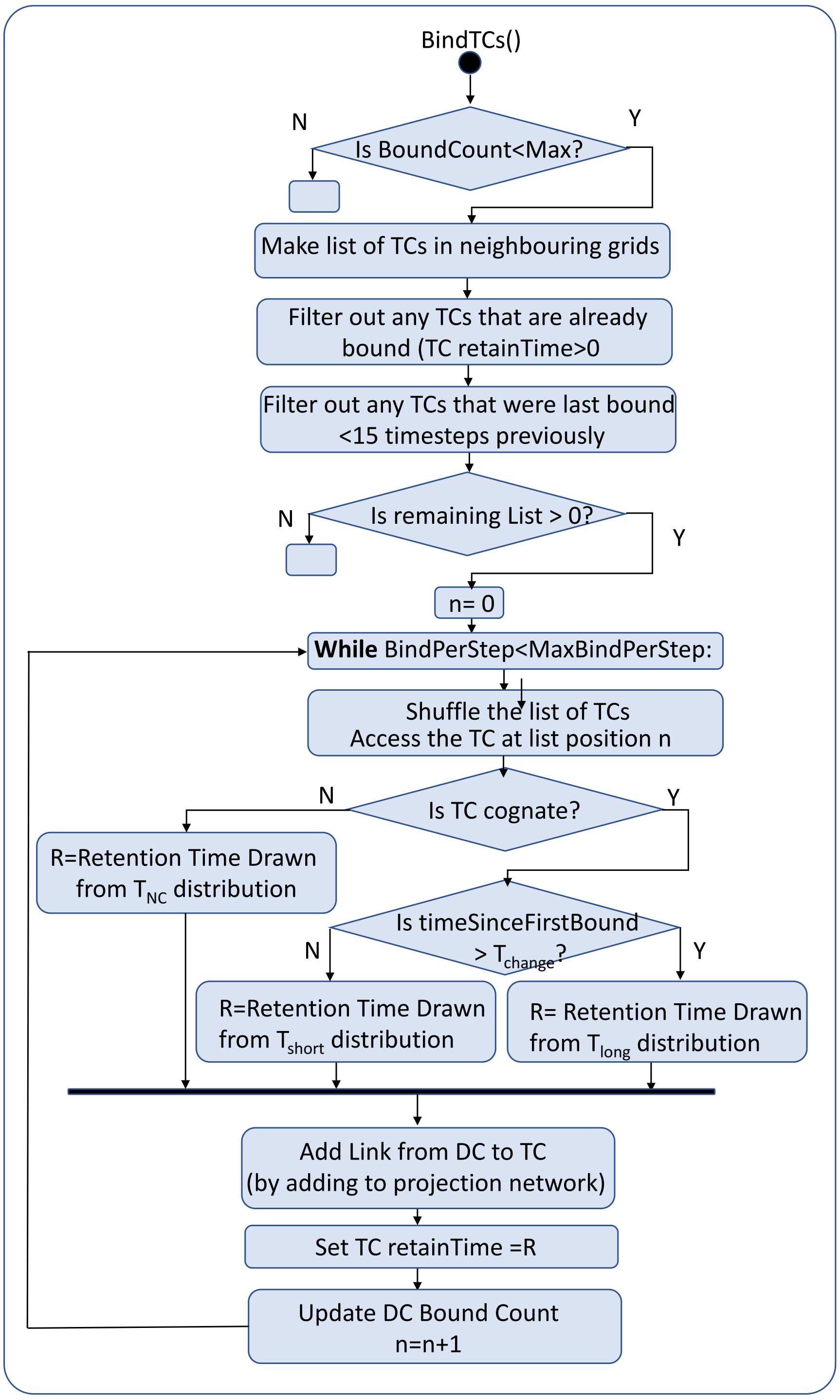
BindTCs()
N
Y
Is BoundCount<Max?
Make list of TCs in neighbouring grids
Filter out any TCs that are already bound (TC retainTime>0
Filter out any TCs that were last bound <15 timesteps previously
N
Is remaining List > 0?
Y
n= 0
While BindPerStep<MaxBindPerStep:
Shuffle the list of TCs
Access the TC at list position n
N
Is TC cognate?
Y
R=Retention Time Drawn from T_NC distribution
Is timeSinceFirstBound > T_change?
N
Y
R=Retention Time Drawn from T_short distribution
R= Retention Time Drawn from T_long distribution
Add Link from DC to TC (by adding to projection network)
Set TC retainTime =R
Update DC Bound Count
n=n+1

Figure S19. UML diagram for ’BindTCs’. TCs receive a ’interaction time’ drawn from a relevant probability density function, based on their type and history.

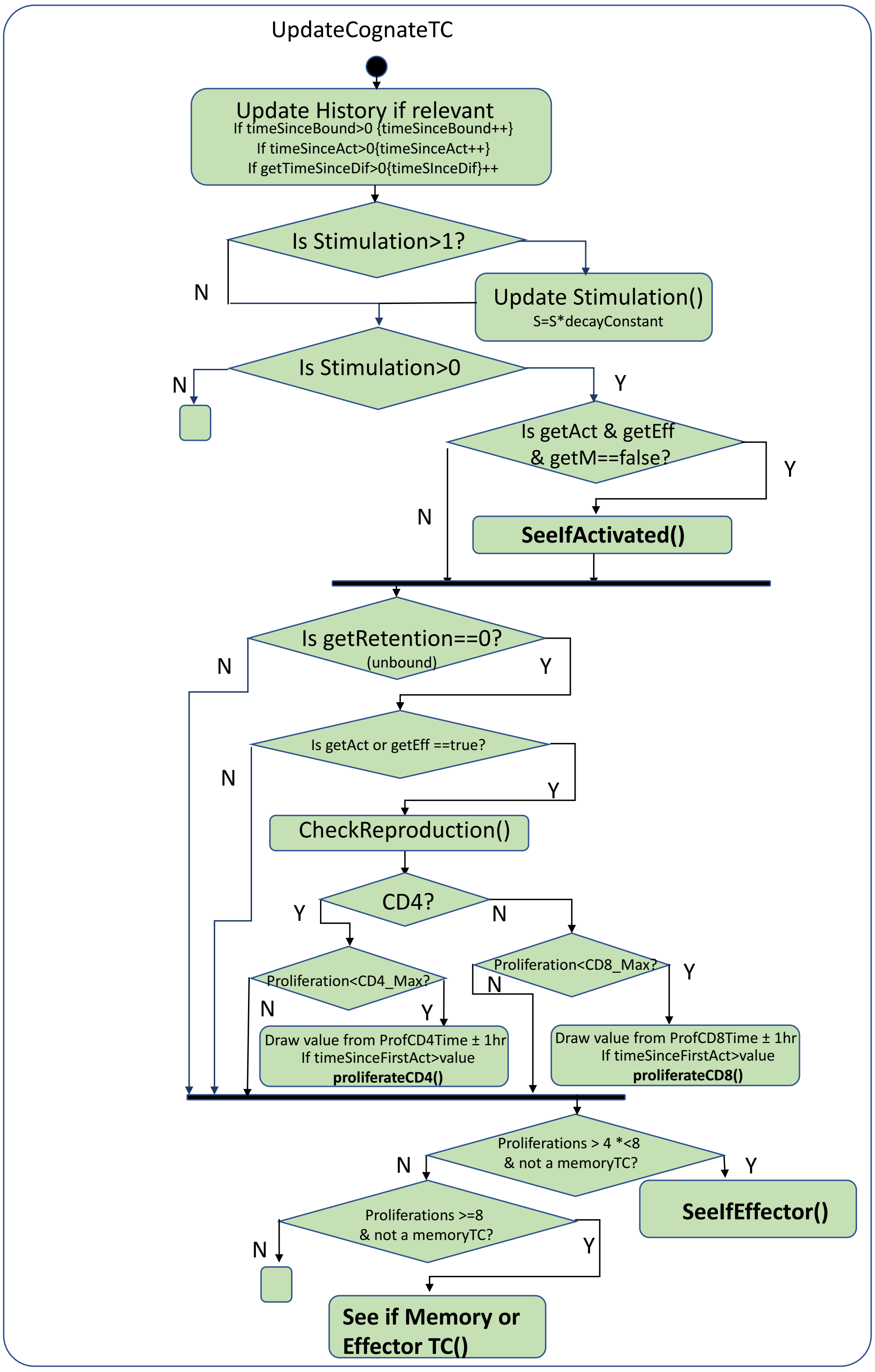
UpdateCognateTC
Update History if relevant
If timeSinceBound>0 {timeSinceBound++}
If timeSinceAct>0{timeSinceAct++}
If getTimeSinceDif>0{timeSInceDif}++
Is Stimulation>1?
N
Update Stimulation()
S=S*decayConstant
Is Stimulation>0
N
Y
Is getAct & getEff & getM==false?
Y
N
SeeIfActivated()
Is getRetention==0? (unbound)
N
Y
Is getAct or getEff ==true?
N
Y
CheckReproduction()
CD4?
Y
N
Proliferation<CD4_Max?
N
Y
Proliferation<CD8_Max?
N
Y
Draw value from ProfCD4Time ± 1hr If timeSinceFirstAct>value proliferateCD4()
Draw value from ProfCD8Time ± 1hr If timeSinceFirstAct>value proliferateCD8()
Proliferations > 4 *<8 & not a memoryTC?
N
Y
SeeIfEffector()
Proliferations >=8 & not a memoryTC?
N
Y
See if Memory or Effector TC()

Figure S20. UML diagram for 'UpdateCognateTCs'.

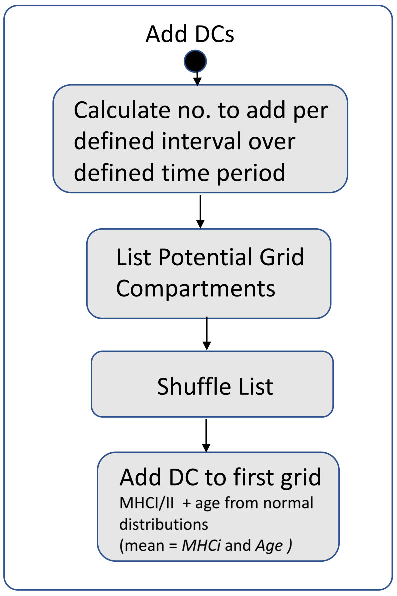
Add DCs
Calculate no. to add per defined interval over defined time period
List Potential Grid Compartments
Shuffle List
Add DC to first grid
MHCI/II + age from normal distributions
(mean = *MHCi* and *Age*)

Figure S21. UML diagram for 'AddDCs'.

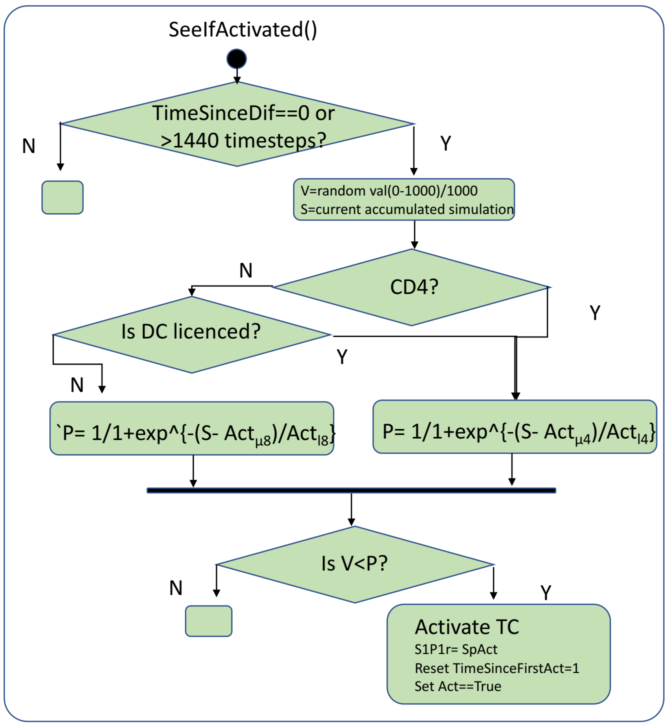
SeeIfActivated()
TimeSinceDif==0 or >1440 timesteps?
N
Y
V=random val(0-1000)/1000
S=current accumulated simulation
CD4?
N
Y
Is DC licenced?
Y
N
`P= 1/1+exp^{-(S- Act$_{\mu8}$)/Act$_{l8}$}
P= 1/1+exp^{-(S- Act$_{\mu4}$)/Act$_{l4}$}
Is V<P?
N
Y
Activate TC
S1P1r= SpAct
Reset TimeSinceFirstAct=1
Set Act==True

Figure S22. UML diagram for 'SeeIfActivated'.

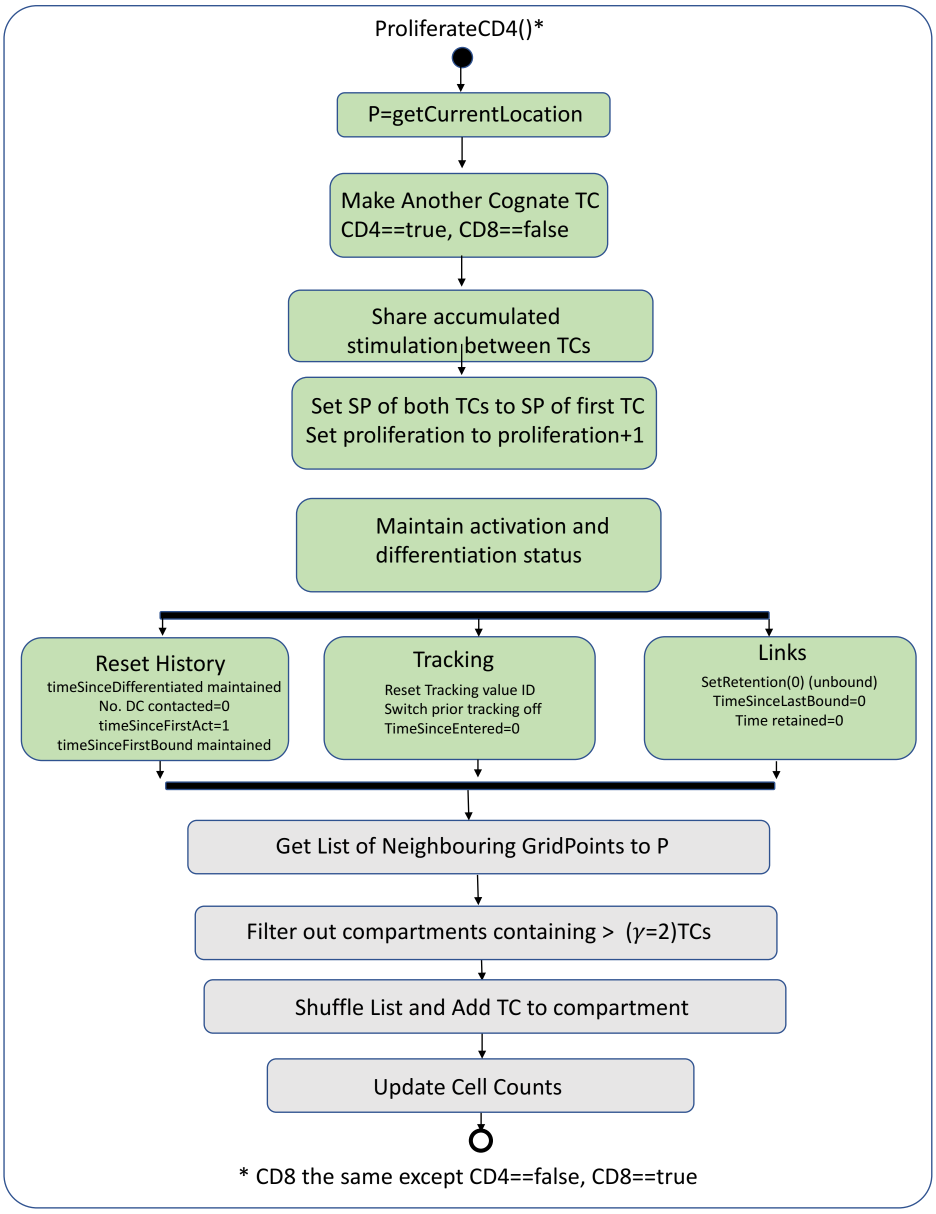
ProliferateCD4()*
P=getCurrentLocation
Make Another Cognate TC CD4==true, CD8==false
Share accumulated stimulation between TCs
Set SP of both TCs to SP of first TC Set proliferation to proliferation+1
Maintain activation and differentiation status
Reset History
timeSinceDifferentiated maintained
No. DC contacted=0
timeSinceFirstAct=1
timeSinceFirstBound maintained
Tracking
Reset Tracking value ID
Switch prior tracking off
TimeSinceEntered=0
Links
SetRetention(0) (unbound)
TimeSinceLastBound=0
Time retained=0
Get List of Neighbouring GridPoints to P
Filter out compartments containing > (γ=2)TCs
Shuffle List and Add TC to compartment
Update Cell Counts
* CD8 the same except CD4==false, CD8==true

Figure S23. UML diagram for 'Proliferation'.

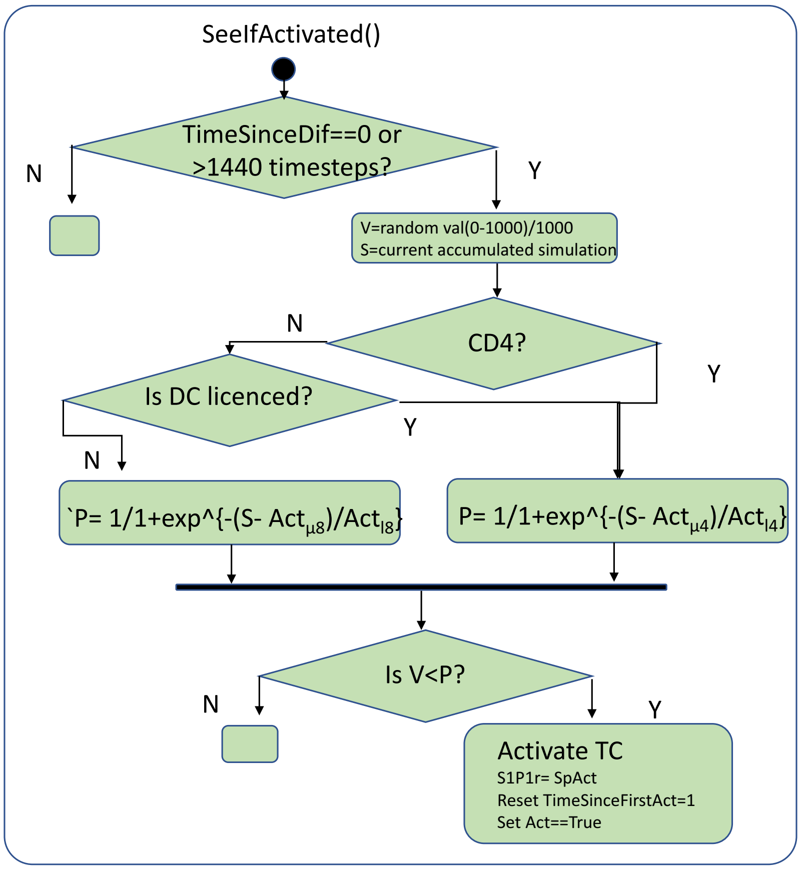
SeeIfActivated()
TimeSinceDif==0 or >1440 timesteps?
N
Y
V=random val(0-1000)/1000
S=current accumulated simulation
N
CD4?
Y
Is DC licenced?
Y
N
`P= 1/1+exp^{-(S- Act$_{\mu 8}$)/Act$_{l8}$}
P= 1/1+exp^{-(S- Act$_{\mu 4}$)/Act$_{l4}$}
Is V<P?
N
Y
Activate TC
S1P1r= SpAct
Reset TimeSinceFirstAct=1
Set Act==True

Figure S24. UML diagram for 'SeeIfEffector' or 'SeeIfEffectorMemory. The two behaviours are similar but for seeIfEffectorMemory, the time since differentation cannot be zero, as the TC must have previously differentiated, and more memory TCs are made.

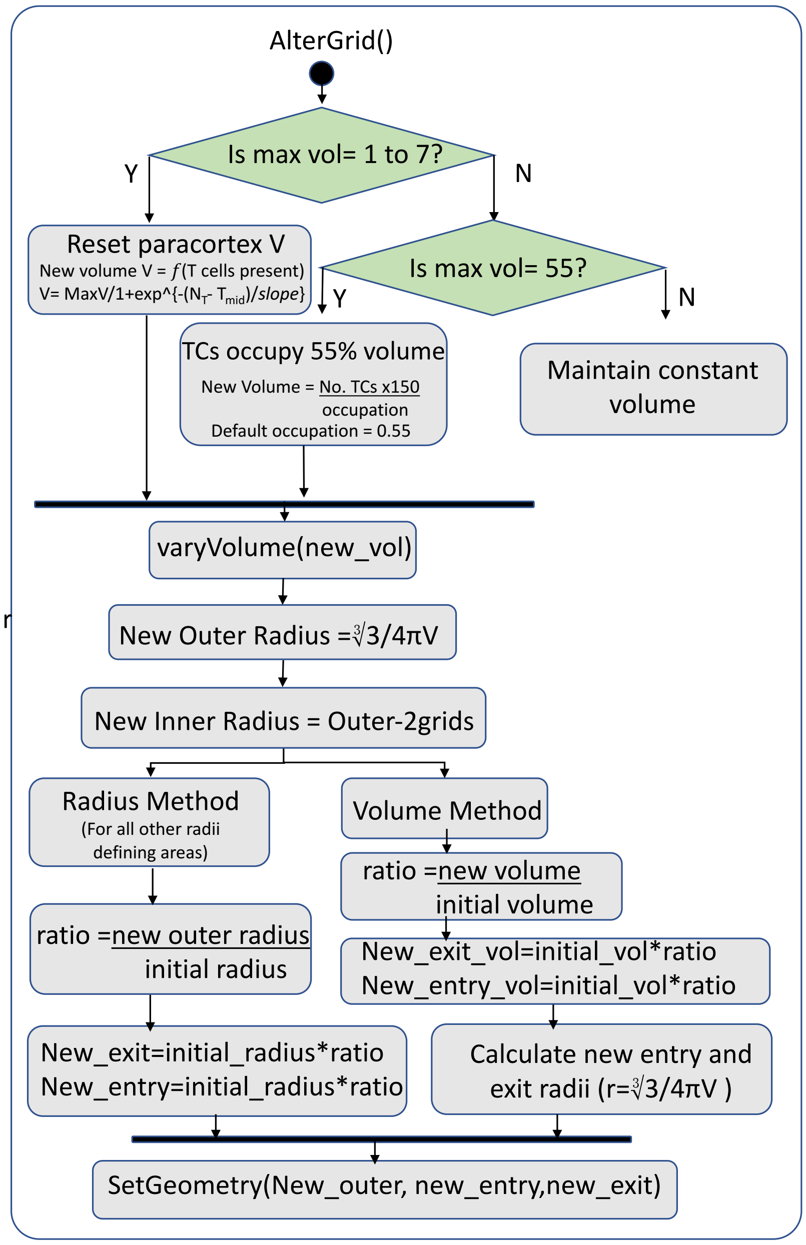
AlterGrid()
Is max vol= 1 to 7?
Y
N
Reset paracortex V
New volume V = ƒ(T cells present)
V= MaxV/1+exp^{-(N_T- T_mid)/slope}
Is max vol= 55?
Y
N
TCs occupy 55% volume
New Volume = No. TCs x150 / occupation
Default occupation = 0.55
Maintain constant volume
varyVolume(new_vol)
New Outer Radius =∛3/4πV
New Inner Radius = Outer-2grids
Radius Method
(For all other radii defining areas)
Volume Method
ratio =new volume / initial volume
ratio =new outer radius / initial radius
New_exit_vol=initial_vol*ratio
New_entry_vol=initial_vol*ratio
New_exit=initial_radius*ratio
New_entry=initial_radius*ratio
Calculate new entry and exit radii (r=∛3/4πV)
SetGeometry(New_outer, new_entry,new_exit)

Figure S25. UML diagram for 'AlterGrid'. This methods carries out the calculation of the new paracortex volume then passes the calculated radii to 'Set Geometry'.

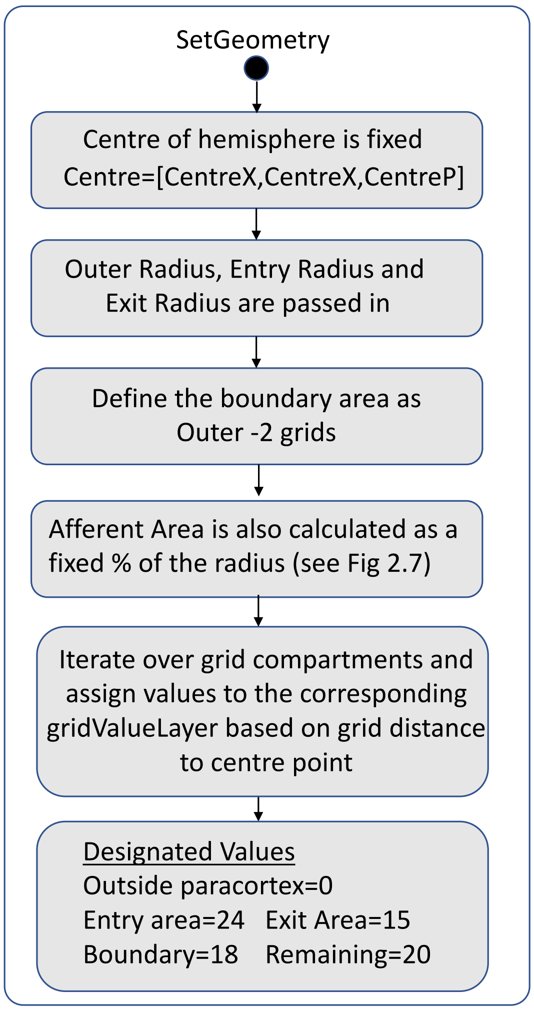
SetGeometry
Centre of hemisphere is fixed Centre=[CentreX,CentreX,CentreP]
Outer Radius, Entry Radius and Exit Radius are passed in
Define the boundary area as Outer -2 grids
Afferent Area is also calculated as a fixed % of the radius (see Fig 2.7)
Iterate over grid compartments and assign values to the corresponding gridValueLayer based on grid distance to centre point
Designated Values
Outside paracortex=0
Entry area=24 Exit Area=15
Boundary=18 Remaining=20

Figure S26. UML diagram for 'SetGeometry'. Only the radii are required as inputs to determine the geometry of the LN