**Mahleran To-do List**

|  |  |  |
| --- | --- | --- |
| **Issue** | **For** | **Re-solved?** |
| Why are variables in interstorm and vegetation dynamics routines real not double precision, and are there any issues with conversion | J, E |  |
| Check which subroutines still use implicit variable types and convert everything to implicit none | J |  |
| Check units and unit conversion | ? |  |
| **read\_parameters\_from\_xml\_file** allows no\_species to be set:   * But code elsewhere assumes no\_species is only 2 * Reads and sets following parameters for vegetation model for each “species”:   + theta\_WP   + root (for two layers only)   + uptake   + r   + mr   + e * Sets dist\_max = maximum distance for shrub dispersal, but doesn’t allow setting of the parameter for grass (which is dealt with differently in the dispersal routine) * start\_season and stop\_season are read in, but can’t be varied according to vegetation type – is this reasonable? * Need to check for other hard-coded parameters in the vegetation submodels | J, E |  |
| **MAHLERAN\_vegdyn\_setting\_xml**:   * dt\_vegetation is hard-coded as 14 (days) – should it be allowed to vary? * Modify the way c\_veg is initialized (see below) | J, E, L |  |
| Vegetation-cover data   * At the moment, the name of the shrub-cover map is read in in **read\_parameters\_from\_xml\_file** and there’s a separate routine called **read\_shb\_cov\_xml** in **MAHLERAN\_interstorm\_setting\_xml**. * There are issues in that the array cover has been misinterpreted to relate to total vegetation cover (it is actually cover *type*) – details in separate emails to Eva. * More generic approach would be to have:   + For each of 1:no\_species:     - A name of the vegetation type (can be used in output filenames and headers)     - Either a filename with the cover proportion (0-1) or a flag to say that the cover should be generated randomly (can use distribution functions in **calculate\_surface\_properties\_from\_types** as long as there are constraints to stop total cover >1)   + Modify **veg\_dyn** so that veg (:, :) is the sum of the c\_veg (2, :, i, j) for the feedback to detachment   + Check wherever the variables grass\_cover and shrub\_cover are used and replace them with appropriate c\_veg values | J, E, L |  |
| **veg\_dyn** is the main subroutine controlling the vegetation model   * It is currently called from **MAHLERAN\_interstorm\_xml** every 14 days (controlled by the value of dt\_vegetation, but it is hard-coded as noted above, as well as the values of start\_season and stop\_season). * The growth routine is called each time * If the day equals the stop\_season value, then dispersal and mortality are called * If start\_season and stop\_season are species (or climate?) dependent, then both the calls in **MAHLERAN\_interstorm\_xml** and **veg\_dyn** would need to be modified * There is currently a hard-coded overgrazing parameter. It is set only to work during the years 1997 and 1998. * Grazing is constrained to be a fixed proportion of the vegetation. * No grazing takes place outside the growing season, because of the way **veg\_dyn** is called * (Over)grazing options need to be more flexible and have input parameters. * Not clear how grass\_cover and shrub\_cover which are then updated are used elsewhere * l.74 has:   + cover (:, :) = (c\_veg (2, 1, :, :) + c\_veg (2, 2, :, :)) \* 100. * It should be:   + veg (:, :) = (c\_veg (2, 1, :, :) + c\_veg (2, 2, :, :)) \* 100., and ultimately updated for no\_species>2 * There is a call to **output\_vegdyn** to output vegetation values (so this only happens during the growing season) and to output the final maps (which wouldn’t work if the last day of simulation were outwith the growing season) * Values of gr (:, :, :), mort (:, :, :), disp (:, :, :) are reset to zero, but these are always set to zero at the start of the relevant routines, so are these lines necessary here? | J, E, L |  |
| **growth**   * lap is hardcoded to 20% (maximum overlap between grass and shrub) – needs to be a parameter and flexible (e.g. a matrix of overlaps between species k=1:no\_species and species l=1:no\_species) * mean\_sm is recalculated each time the loop goes through no\_species – move to a separate loop so it’s only calculated once per call? * What about soil moisture stored at depth and available to shrubs? Should it only be accessed in the summer growing season? * dummy1 seems to be used to calculate two separate things. Rename dummy1 and dummy2 to make the code more readable? * Where dummy1 is used for relative water uptake per unit cover, it needs to be in the loop for when no\_species>2 * dt\_vegetation / 365. at the end could be made into a constant so it’s not recalculated each time (unless dt\_vegetation is made variable?) | J, E |  |
| **mortality**   * need to check that any change to growing season by species type doesn’t affect the current routine * similarly, why is it only called once at the end of the season? Why not at fortnightly intervals like growth? * Similar issue with calculation of seasonal mean soil moisture as in growth routine * Rename dummy1 to something more meaningful * duration\_season / 365. Could be made into a constant so not recalculated each time. And this scaling could be vectorized for speed. * Erosion feedback is currently hardcoded and only affects grass. Why not affect all species, and why use a constant rather than a proportion? * If stop\_season is species-dependent, net\_erosion\_season needs to be reset elsewhere | J, E, L |  |
| **dispersal**   * Same issue with calculation of mean\_sm as elsewhere * Hardcoded routines for grass and shrub dispersal * dummy1 is mean grass in 3x3. counter isn’t used. Potential for edge effects as the mean uses values beyond the plot boundary. * Shrubs only disperse if their cover is >0.3 – why, and why a fixed value? * Parameters d\_0s and cd are hardcoded at the start of the routine * Could be made more generic with appropriate choice of kernel size, dist\_max, cover\_threshold, d\_0s and cd * Why is there no temporal scaling according to the growing-season length? | J, E, L |  |
| **output\_vegdyn**   * Check output should still occur when rday.le.total\_days * **n** is used to count cells then output four variables – change first instance to n\_cell (which could be counted only once at the start of the run) * dummy is used to calculate four different variables (so can be used in implied do loop for output, but naming might be clearer) * dummy is hardcoded to be limited to only allow up to 10 different species * For more species, check fmtstring doesn’t overflow – also unclear why fmtstring is 8\*no\_species, not 4\*no\_species * Maps for final output could be combined into single loop if the species names are input values * Net erosion is output for last year – but will have been reset to zero if **mortality** called for that year. What is the value of this output rather than the general erosion output? | J, E |  |
| **calc\_inf**   * Need to confirm (from Tietjen et al. 2009) what the feedbacks are from the vegetation cover. * And generalize for no\_species > 2 | J, E, L |  |
| **calc\_et**   * Uses veg for feedback so will currently not have a vegetation feedback because of the updating of cover in **veg\_dyn** | J, E, L |  |
| **calc\_sm**   * Should there be a vegetation dependency? * Better accounting for wilting point, field capacity and plant-available moisture * What link is there between interstorm and storm soil moisture? (or is this in **calc\_inf**?) | J, E, L |  |
| **interstorm\_initialize\_xml**   * Issues as above with initialization of grass\_cover * Why is field capacity set as 0.75 \* saturated moisture content? Should it be a free parameter?   + Look at pedotransfer functions (e.g. in Optirrig) | J, E, L |  |
| **raindrop\_detachment**   * Feedback is via veg (i.e. total vegetation cover) – might need to make it ultimately a function of vegetation type? | J, E, L |  |
| **splash\_detach\_markers\_xml**   * Doesn’t seem to have vegetation feedback | J, E, L |  |
| **flow\_detachment**   * No vegetation feedback | J, E, L |  |
| **Drought scenarios**   * Currently just through modifying climate – is this OK? | J, E, L |  |
| **Fire**   * Could add in the Perry and Enright model (but only reasonable for larger scale applications) * Would need to have feedbacks to soil parameters | J, E, L |  |
| **Roots**   * Need to evolve through time? * How is above- and below-ground biomass linked? * What about vegetative (re)growth? | J, E, L |  |
| **Climate data**   * Standardization of formats * Deciding what to calculate and what to read in |  |  |
| **Variables**   * Listing in common format at head of subroutine |  |  |
| **Boundary conditions**   * Inflows at plot edges * Decouple from the rainmask map |  |  |
| **Particle sizes**   * Number of classes * Size limits of different classes * Interactions between sizes during erosion processes |  |  |
| **Tracking particle size with depth**   * And what happens with soil depth as erosion/topographic updating occurs |  |  |
|  |  |  |