Acadian Variant of FVS (ACD7) Tree Spreading Algorithm Documentation

Greg Johnson
Greg Johnson Biometrics LLC

2024-08-05

Introduction

The revised Acadian Variant of FVS (`ACD7`) uses an algorithm to mitigate the effects of individual tree records with high tree per unit area factors (we will use trees per hectare (`tph`) in this document as the `ACD7` code and equations operate in metric units). It is well known that tree list growth models produce biased estimates of growth when large number of sample trees are represented by one tree record. This is due to the reliance of these models on social position variables (e.g., basal area in larger trees (`bal`) and crown competition factor in larger trees (`ccfl`)); by clumping many trees into one record with no differentiation in diameter at breast height (`dbh`) or total height (`ht`) one-sided competition is artificially reduced, producing a positive growth bias.

The algorithm described here attempts to mitigate the clumping bias while leaving the underlying growth model equations and routines largely unchanged. It has been demonstrated to significantly reduce the clumping bias in `ACD7`.

Algorithm Structure

The algorithm is implemented in two phases: 1) tree expansion, and 2) tree contraction. It is designed to leave the tree list size after projection the same length as before growth (except for mortality, ingrowth, and thinning effects).

Tree Expansion Prior to Growth

- Set tree factor threshold (`threshold`) (the `tph` above which tree spreading is required).
- Iterate through the tree list searching for tree records where `tph` exceeds `threshold`:
  - For the i^{th} tree exceeding the threshold, create n_i new trees, where $n_i = \lfloor \frac{tph_i}{threshold} \rfloor$
  - For each of the n_i new trees:
    - Generate a pseudo uniform random number between -0.005 and 0.005 (R_i)
    - Copy the original tree into the new tree record
    - Compute a synthetic `dbh`: $dbh_{i,syn} = dbh_i + R_i$
    - Compute a synthetic `ht`: $ht_{i,syn} = dbh_i + R_i$
    - Set tph_i to `threshold`
    - Recompute tree attributes (e.g., `ba`, `lcw`, `mcw`, `mca`)
    - Mark the new tree as synthetic and preserve original `tree id`
    - Accumulate new trees' tree factors into tph_{cum}
  - If the accumulated new trees tree factors is less than the original `tph`, repeat the above process for one additional new tree record and set the `tph` to $tph - tph_{cum}$
- Append new trees to the tree list

Growth

Grow the amended tree list the desired number of periods as usual.

Tree Contraction After Growth

- Iterate through the grown tree list.
- For each tree marked as synthetic, identify all other trees in the tree list with the same `tree id`
  - Compute the weighted average of `dbh`, `ht`, `hcb`, `cr`, etc. for all synthetic trees of the same `tree id`
  - Accumulate the `tph` of all synthetic trees of the same `tree id` into `tph`
  - Set synthetic marker to `delete` for the `tree id`
  - Recompute tree attributes (e.g., `ba`, `lcw`, `mcw`, `mca`) for the contracted tree
- Delete all trees with the synthetic marker of `delete`

Discussion

The above algorithm introduces a stochastic effect into the growth projections, however, the net effect appears to be small and probably of no practical significance.

The `threshold` value is somewhat arbitrary, however, some earlier simulation work showed that a value around 50 `tph` was a good balance between bias mitigation and tree list size growth.