Soil Carbon Pool Models and the Equilibrium Assumption

Thomas Wutzler
Max-Planck-Institute for Biogeochemistry Jena
– Review of Assumptions of the Yasso Soil Carbon Model
– Weakening the Equilibrium Assumption
– Severe consequences for spinup-runs and estimation of carbon stocks
– Minor consequences for century-term simulations
Motivation for Yasso

Yasso model
- Simulates changes and carbon stock of dead wood, organic layer and mineral soil and carbon release
- Simulates on an annual basis
- Requires only
  - Estimates of litter production
  - Basic data on climate (MAT, precip., ETP)
- Used to estimate carbon stocks by wood and litter inputs
Assumptions (1)

- Litter consists of compounds with
  - Exponential decay
  - Typical decomposition rates, independent of origin
- Decomposed mass
  - Is respired in part.
  - Forms more recalcitrant compounds in other part

Liski et al. 2005
Assumptions (2)

- Availability for Decomposition depends on exposition to microbial biomass
- Microbial activity and decomposition rates depend on temperature and moisture
Parameterization

• Standard climate (middle Sweden)
  – Rate of microbial exposure: mass loss experiments
  – Chemical composition: literature
  – Fast pools decomposition rates and respiration proportion: litter bag studies
  – Slow pools respiration proportions: prescribed
  – Slow pools decomposition rates: adjusted to measured equilibrium pools

• Climate dependence
  – Mass loss experiments at 34 sites across Europe
Approach of a Weaker Assumption

- Take slowest pool out of equilibrium assumption.
- Determine decomposition rate by current change in carbon stocks.

\[ \Delta q = u - y \]
\[ \Delta q = u - r \cdot q \]
\[ r = \frac{u - \Delta q}{q} \]

- \( q \): stock
- \( \Delta q \): stock change
- \( u \): input
- \( r \): decay rate
- \( y \): output

Wutzler, not published
What if soils are still accumulating?

Equilibrium experiment performed with standard parameters and litter inputs for Norway Spruce.

Assume different observed stock changes and calculate

- Decay rates $r$
- Time for accumulation near equilibrium (t95)
- Theoretical equilibrium stocks.

\[ \Delta q = 0 = u - r \cdot q_e \]
\[ q_e = \frac{u}{r} = f(u, q, \Delta q) \]
Conclusions (1)

• Soils carbon stocks may be far apart from equilibrium, despite the observation of no significant changes in stocks.

• The Yasso model can not be used by its own to estimate soil carbon stocks of disturbed soils just by litter inputs.

• The parameterization of the decay rate of the slowest pool by observations is very uncertain due to the long time scales. This problem of time scales is shared by other pool models too. (Roth-C, Century, Biome-BGC)

• For assessing long term consequences, we need mechanistically model the decay rates. (e.g. Davidson et al. 2006, Fontaine et al. 2005)
Medium term consequences?

• Hypothesis: Due to the long time scales, uncertainty in slow-pool decay rate will have only a small effect on century-term simulations of
  – Changed litter input
  – Temperature change

• Corroboration: Simulation experiments
  – Beech Chronosequence in central Europe
  – Managed as a shelterwood
  – Measured carbon litter fall: = 3.54 tC/ha/yr; organic layer carbon stocks = 3.7 tC/ha; soil carbon stocks = 41.8tC/ha, input due to harvest residues and root = 0.42 tC/ha/yr
  – Average annual temperature = 6.8° centigrade, precipitation – potential evapotranspiration vegetation period = -89 mm
Simulation experiment: Scenarios

- Five different decay rates (0.0012 /5 /5 /5 /5)
- Two temperature sensitivities s2 (0.36, 1)

<table>
<thead>
<tr>
<th>No Increase of Temperature</th>
<th>Shelterwood</th>
<th>Conservation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>base</td>
<td>litter</td>
</tr>
</tbody>
</table>

| 3K Increase of Temperature over 100yr | t3          | t3 + litter  |
Development of Soil Pools (1)

Temperature Sensitivity $s_2 = 0.36$
Development of Soil Pools (2)

Temperature Sensitivity $s_2 = 1$

![Graph showing carbon stock over time for different soil pools with temperature sensitivity $s_2 = 1$]
Effect of Decay Rate

Difference of soil carbon stocks caused by different decay rates of slowest pool after 100 years.

Soil carbon stocks decrease as the decay rate increases. The graphs show the difference in soil carbon stocks caused by different climate sensitivities (0.36 and 1) over a range of decay rates.
Conclusions (2)

- Simulation experiments corroborate that the large uncertainty in slow-pool decay rate has only a small effect on century-term simulations.
- Increased temperature sensitivity increases this effect. (However, still very small)
- Precondition: Estimation of soil carbon stocks by other method besides spinup run.

Long term critical
Medium term OK
Thank you

Tempe Arizona, December 2005
Calculate the resulting theoretical equilibrium stock.
(after thousands of years with same input and conditions)

\[ \Delta q = 0 = u - r \cdot q_e \]

\[ q_e = \frac{u}{r} \]
Part IV

Forest Utilities
Java Implementation of the non-linear least squares problem

Usability via CORBA hard to deploy
Yield Table Interpolation

- **Interpolation** by between data points of yield tables
- Estimation of **timber volume increment** and correction by stocking density
- Conversion between absolute and relative **site index**
- Estimation of **stand age** by stand properties
- Conversion between timber volume and **basal area**
- Easily **extendable** by inserting data of further yield tables into a database
Tree Biomass Carbon Stocks

**single-Tree**

<table>
<thead>
<tr>
<th>method</th>
<th>age</th>
<th>d/b</th>
<th>height</th>
<th>nha</th>
<th>stem</th>
<th>branches</th>
<th>foliage</th>
<th>fine root</th>
<th>root</th>
<th>vol</th>
<th>one t</th>
<th>total t/ha</th>
</tr>
</thead>
<tbody>
<tr>
<td>BeechWirth05</td>
<td>81</td>
<td>30</td>
<td>32</td>
<td>101</td>
<td>366</td>
<td>40.5</td>
<td>3.2</td>
<td>3.2</td>
<td>52.9</td>
<td>465.2</td>
<td>47.0</td>
<td></td>
</tr>
<tr>
<td>SpruceWirth05</td>
<td>81</td>
<td>30</td>
<td>32</td>
<td>100</td>
<td>241.3</td>
<td>20.9</td>
<td>16.1</td>
<td>32.2</td>
<td>52.2</td>
<td>370.0</td>
<td>37.1</td>
<td></td>
</tr>
</tbody>
</table>

**stand**

<table>
<thead>
<tr>
<th>method</th>
<th>volume</th>
<th>age</th>
<th>site index</th>
<th>stem</th>
<th>branches</th>
<th>foliage</th>
<th>fine root</th>
<th>root</th>
<th>total</th>
</tr>
</thead>
<tbody>
<tr>
<td>BeechWirth05</td>
<td>151</td>
<td>38</td>
<td>14.6</td>
<td>51.1</td>
<td>11.3</td>
<td>1.3</td>
<td>2.6</td>
<td>8.1</td>
<td>74.5</td>
</tr>
<tr>
<td>Pine Lehtonen04</td>
<td>180</td>
<td>39</td>
<td>14.6</td>
<td>33.2</td>
<td>7.9</td>
<td>4.1</td>
<td>16.5</td>
<td>15.9</td>
<td>77.6</td>
</tr>
</tbody>
</table>

KBEF = carbon mass / timber vol.  
= [ (tC/ha) / (m³/ha) ]  
KBEF = b0 + b1 * exp( -b2 * Age )  

By Species, and site index

```xml
<submodel scenario="Beech Wirth05">  
<siteIndexClasses descr="minimal absolute site indices (height at age 100) for relative classes" class1="28.0" class2="20.0"/>
<cont descr="carbonContent (mass Carbon)/(mass dry wood)">
<stem value="0.486" cv="0.01"/>
<foliage value="0.5" cv="0.04"/>
</cont>
<turnover descr="turnover rate 1/yr"/>
</submodel>

<submodel scenario="Spruce Wirth05" inherit="Beech Wirth05"/>

<submodel scenario="Pine Lehtonen04" />
<submodel scenario="Oak Beech Wirth05"/>
</submodel>
```
1. Define an abstract DEVS-interface in a language independent way
2. Provide a model adapter that controls the model a specific in the DEVS simulation engine and provides the DEVS-interface
3. Provide a model proxy that controls a DEVS-interface and acts as a local model within a DEVS-simulation engine
4. Employ a middleware to mediate between model proxy and model adapter
The DEVS-Approach
Towards a More General Model Coupling

Thomas Wutzler
Max-Planck-Institut für Biogeochemie Jena
System Specification Hierarchy

1. Observation frame
2. I/O behaviour
3. I/O function
4. State transition
5. Coupled component
Model Validation

DTSS: fixed times. e.g. computer clock ticks
DEVS: times of events: e.g. clerk
DESS: differential equations, continuous in time, e.g. position of moon
Simulation of DESS

\[ \frac{dx}{dt} \Delta t = \Delta x \]

- Discretization by time
  - Calculate change of state during time step
    \( \Delta x = \frac{dx}{dt} \times \Delta t \)
- Discretization by values (Quantization)
  - Calculate change of time till next boundary crossing
    \( \Delta t = \frac{dx}{dt} \times \frac{1}{\Delta x} \)
  - Precise during rapid changes
  - Only few Transition and Messages within nearly stable phases

Pde-shockwave application: "time to solution is significantly reduced when discrete event integration scheme is employed compared to a representative conventional approach"

Choice of a quantization interval at the value scale -> arbitrary small error

Nutaro et al., 2003. Discrete event solution of gas dynamics within the DEVS framework. In, Computational Science - Iccs 2003
Be Aware of Numerical Errors!

Example: Liski et al. (2005): soil carbon model YASSO

Explicitly runs at a yearly time step.

Uncertainty of stock changes mostly due to rate $a_{fwl}$, with largest effect after harvest.

Rate is at the same magnitude as the time step -> 57% rel. error.

Be careful when interpreting short term changes
Use Simulators that adjust time step or use quantization
Towards Component Models

Monolithic models

- Lacking flexibility
- Hard to Understand, Critique, Debug and Modify

Efficient flexible structure of Independent Components
- A good design lets components
  1. Relate directly to real world components
  2. Have measurable IO-variables
  3. Communicate solely via specified input and output variables

Specialization / Generalization

• **Performance**
• Usage of specialized services like parallelization
• Example: PRISM and the OASIS-coupler 😊
• Disadvantages:
  – Requires adaptation of the component models for specific coupling scenarios. Can you exchange JS-Bach with another Vegetaion model by a coupling specification?

• **Flexibility**
• Genericness: Wider Range of applications
• Easy integration into yet unknown coupling scenarios
• Example: DEVS-Bus
• Disadvantages:
  – How to integrate component models with specialized services and environments?
  – Possible Consumption of performance
The DEVS-Bus

- DEVS: Discrete Event System Specification
  - Set of input- and output ports
  - Variable time step
  - State Transitions
    - Without inputs during time step
    - With inputs during time step
    - With inputs at the end of the time step

- reproduces
  - DTSS (time stepped): time advance equals time step inputs at beginning of steps
  - DESS (differential equation, continuous time): by DTSS or quantized approach

- Simulator/coordinator takes care of message passing and scheduling

- Closure under coupling (coupled model is itself a DEVS model)
The general DEVS-Interface

```java
interface Devs { // OMG-idl (Corba)
    void doInitialize();
    // begin of simulation
    double timeAdvance();
    // time of next output and internal transition without input events
    Message outputFunction();
    // produce outputs for current time
    void internalTransition();
    // state transition
    void externalTransition( in double elapsedTime, in Message msg); // transition with inputs event before next internal transition
    void confluentTransition( in Message msg );
    // input event at time of internal transition
};
```

**Message**: bag { inputPort -> value }
Example: Soil Model Yasso

YassoLitter

YassoSoil

// climatic parameters (external coupling)
couple( this, this->I_MAT, yassoSoil, yassoSoil->I_MAT );
couple( this, this->I_MAT, yassoLitter1, yassoLitter1->I_MAT );

// non woody litter (internal coupling)
couple( yassoLitter1, yassoLitter1->O_EXT, sumExt, 1 );
couple( yassoLitter2, yassoLitter2->O_EXT, sumExt, 2 );
couple( sumExt, 1, yassoSoil, yassoSoil->I_EXT );
• Trade-off between Specialization vs. Generalization
• Excelent general interface: DEVS (events)
  – Reproduces DESS (continuous) and DTSS (time-stepped)
  – Simulators available which guarantee correct message passing and avoid deadlocks
  – General model coupling even between different languages
  – Very efficient for parallel and distributed execution of component models
• Theory and example of application:
Thank you

Model