Documented ICBM recalibration equations

In this document I describe the JAGS model file used in the ICBM recalibration.

As a doubleck, the code is shown here. This is exactly the file currently used by the model (the following code chunk is produced with a relative link). **Please consider this just for documentation, and skip to the end of it, where the code is described in detail with mathematical notation.**

```
model{

  ####Loop for treatments
  for(j in 1:J_Ultuna)
  {

    Y_R_Ultuna[j,1]<-(SOC_init_Ultuna[j]*(1-Init_ratio_Ultuna))*0.5
    Y_S_Ultuna[j,1]<-(SOC_init_Ultuna[j]*(1-Init_ratio_Ultuna))*0.5
    Y_FYM_Ultuna[j,1]<-0
    Y_GM_Ultuna[j,1]<-0
    Y_PEA_Ultuna[j,1]<-0
    Y_SAW_Ultuna[j,1]<-0
    Y_SLU_Ultuna[j,1]<-0
    Y_STR_Ultuna[j,1]<-0
    O_Ultuna[j,1]  <-SOC_init_Ultuna[j]*Init_ratio_Ultuna

    # loop for years
    for (i in 1:(N_Ultuna)){

      alpha_1[j,i]=ifelse(j==1, 0, alpha) #ifelse to have the intercept zero in case of bare fallow
      alpha_maize_1[j,i]=ifelse(j==1, 0, alpha_maize) #ifelse to have the intercept zero in case of bare fallow


      #Inputs R (roots), with different allometric functions for crops
      #depth coefficients from: Fan, J., McConkey, B., Wang, H., & Janzen, H. (2016). Root distribution by depth for
      #temperate agricultural crops.
```

```r
    #Field Crops Research, 189, 68-74. https://doi.org/10.1016/j.fcr.2016.02.013
    I_R_cereals_Ultuna[j,i]      <-  (1+exudates_coeff)*e_depth_cer*0.5560437*C_percent*
                                          ((alpha_1[j,i]+Yields_cereals_Ultuna[j,i])*(1/(SR_cereals_ult)))
    I_R_root_crops_Ultuna[j,i]   <-  (1+exudates_coeff)*e_depth_root*0.5902446*0.32*C_percent*
                                          ((alpha_1[j,i]+Yields_root_crops_Ultuna[j,i])*(1/(SR_root_crops_ult)))#SR_root_crops_ult))
    I_R_oilseeds_Ultuna[j,i]     <-  (1+exudates_coeff)*e_depth_oil*0.6367459*C_percent*
                                          ((alpha_1[j,i]+Yields_oilseeds_Ultuna[j,i])*(1/(SR_oilseeds_ult)))#SR_oilseeds_ult))
    I_R_maize_Ultuna[j,i]        <-  (1+exudates_coeff)*e_depth_maize*0.5981638*C_percent*
                                          ((alpha_maize_1[j,i]+Yields_maize_Ultuna[j,i])*(1/(SR_maize_ult)))#SR_maize_ult))
    I_R_Ultuna[j,i]              <-  I_R_cereals_Ultuna[j,i]+I_R_root_crops_Ultuna[j,i]+I_R_oilseeds_Ultuna[j,i]+I_R_maize_Ultuna[j,i]
    #Inputs S
    I_S_Ultuna[j,i]              <- ((Yields_cereals_Ultuna[j,i]+Yields_root_crops_Ultuna[j,i]+Yields_oilseeds_Ultuna[j,i])*
                                          stubbles_ratio_Ultuna+Yields_maize_Ultuna[j,i]*stubbles_ratio_Ultuna_maize)*C_percent

    #Young R
    Y_R_Ultuna[j,i+1]         <-  (I_R_Ultuna[j,i]+Y_R_Ultuna[j,i])*exp(-k1_ult*re_Ultuna[j,i])
    Y_S_Ultuna[j,i+1]         <-  (I_S_Ultuna[j,i]+Y_S_Ultuna[j,i])*exp(-k1_ult*re_Ultuna[j,i])

    Y_FYM_Ultuna[j,i+1]       <-  (I_FYM_Ultuna[j,i]+Y_FYM_Ultuna[j,i])*exp(-k1_ult*re_Ultuna[j,i])
    Y_GM_Ultuna[j,i+1]          <-  (I_GM_Ultuna[j,i]+Y_GM_Ultuna[j,i])*exp(-k1_ult*re_Ultuna[j,i])
    Y_PEA_Ultuna[j,i+1]       <-  (I_PEA_Ultuna[j,i]+Y_PEA_Ultuna[j,i])*exp(-k1_ult*re_Ultuna[j,i])
    Y_SAW_Ultuna[j,i+1]       <-  (I_SAW_Ultuna[j,i]+Y_SAW_Ultuna[j,i])*exp(-k1_ult*re_Ultuna[j,i])
    Y_SLU_Ultuna[j,i+1]       <-  (I_SLU_Ultuna[j,i]+Y_SLU_Ultuna[j,i])*exp(-k1_ult*re_Ultuna[j,i])
    Y_STR_Ultuna[j,i+1]       <-  (I_STR_Ultuna[j,i]+Y_STR_Ultuna[j,i])*exp(-k1_ult*re_Ultuna[j,i])

    #Old

    #old flux
    fluxR_Ultuna[j,i]           <-  h_R_ult*((k1_ult*(Y_R_Ultuna[j,i]+I_R_Ultuna[j,i]))/(k2_ult-k1_ult))
    fluxS_Ultuna[j,i]           <-  h_S_ult*((k1_ult*(Y_S_Ultuna[j,i]+I_S_Ultuna[j,i]))/(k2_ult-k1_ult))

    flux_FYM_Ultuna[j,i]        <-  h_FYM_ult *((k1_ult*(Y_FYM_Ultuna[j,i]+I_FYM_Ultuna[j,i]))/(k2_ult-k1_ult))
    flux_GM_Ultuna[j,i]         <-  h_S_ult   *((k1_ult*(Y_GM_Ultuna[j,i] +I_GM_Ultuna[j,i])) /(k2_ult-k1_ult))
    flux_PEA_Ultuna[j,i]        <-  h_PEA_ult *((k1_ult*(Y_PEA_Ultuna[j,i]+I_PEA_Ultuna[j,i]))/(k2_ult-k1_ult))
    flux_SAW_Ultuna[j,i]        <-  h_SAW_ult *((k1_ult*(Y_SAW_Ultuna[j,i]+I_SAW_Ultuna[j,i]))/(k2_ult-k1_ult))
    flux_SLU_Ultuna[j,i]        <-  h_SLU_ult *((k1_ult*(Y_SLU_Ultuna[j,i]+I_SLU_Ultuna[j,i]))/(k2_ult-k1_ult))
    flux_STR_Ultuna[j,i]        <-  h_S_ult   *((k1_ult*(Y_STR_Ultuna[j,i]+I_STR_Ultuna[j,i]))/(k2_ult-k1_ult))
```

```
    flux_sum_Ultuna[j,i]<-(fluxR_Ultuna[j,i]+
                           fluxS_Ultuna[j,i]+
                           flux_FYM_Ultuna[j,i]+
                           flux_GM_Ultuna[j,i]+
                           flux_PEA_Ultuna[j,i]+
                           flux_SAW_Ultuna[j,i]+
                           flux_SLU_Ultuna[j,i]+
                           flux_STR_Ultuna[j,i])


    O_Ultuna[j,i+1]       <-  (O_Ultuna[j,i]-flux_sum_Ultuna[j,i])*exp(-k2_ult*re_Ultuna[j,i]) +
                                flux_sum_Ultuna[j,i]*exp(-k1_ult*re_Ultuna[j,i])


    #Total C
    Y_tot_Ultuna[j,i] <- Y_R_Ultuna[j,i] +
                        Y_S_Ultuna[j,i] +
                        Y_FYM_Ultuna[j,i] +
                        Y_GM_Ultuna[j,i] +
                        Y_PEA_Ultuna[j,i] +
                        Y_SAW_Ultuna[j,i] +
                        Y_SLU_Ultuna[j,i] +
                        Y_STR_Ultuna[j,i]

    Tot_Ultuna[j,i] <- Y_tot_Ultuna[j,i] + O_Ultuna[j,i]

    #Error of the measurement (assumed proportional to the measurement)
    SOC_Ultuna[j,i]  ~ dnorm(Tot_Ultuna[j,i],1/(error_SOC_Ultuna[j]*1.5))
    #SOC_Ultuna[j,i]  ~ dunif(Tot_Ultuna[j,i]-Tot_Ultuna[j,i]*error_SOC_Ultuna[j],Tot_Ultuna[j,i]+Tot_Ultuna[j,i]*error_SOC_Ultuna[j])

  }

}
```

```
##Parameters Ultuna
# Xie, Yajun. 2020. "A Meta-Analysis of Critique of Litterbag Method Used in Examining Decomposition of Leaf Litters.
#" Journal of Soils and Sediments 20 (4): 1881-86. https://doi.org/10.1007/s11368-020-02572-9.

k1_ult    ~ dunif(0.4, 1)
k2_ult    ~ dunif(0,0.03)


h_S_ult     ~ dunif(0.125-0.125*limits_h,0.125+0.125*limits_h)
h_R_ult     ~ dunif(0.35-0.35*limits_h, 0.35+0.35*limits_h)
h_FYM_ult   ~ dunif(0.27-0.27*limits_h,0.27+0.27*limits_h)
h_PEA_ult   ~ dunif(0.59-0.59*limits_h, 0.59+0.59*limits_h)
h_SAW_ult   ~ dunif(0.25-0.25*limits_h,0.25+0.25*limits_h)
h_SLU_ult   ~ dunif(0.41-0.41*limits_h,0.41+0.41*limits_h)

#root/shoot ratios priors
SR_cereals_ult    ~ dunif(3.6,27.9) #range from martin data
#SR_cereals_ult    ~ dnorm(11,1/2)
SR_root_crops_ult ~ dunif(29.49853-29.49853*limit_SR,29.49853+29.49853*limit_SR)
SR_oilseeds_ult   ~ dunif(8-8*limit_SR,8+8*limit_SR)
SR_maize_ult      ~ dunif(6.25-6.25*limit_SR,6.25+6.25*limit_SR)

#alpha ~dunif(0,1)
#alpha_maize ~dunif(0,1)
alpha = 0
alpha_maize= 0


exudates_coeff ~ dnorm(1.65, 1/0.4125) #10% error
e_depth_cer ~ dnorm(1, 1/0.25) #10% error
e_depth_root ~ dnorm(1, 1/0.25) #10% error
e_depth_oil ~ dnorm(1, 1/0.25) #10% error
e_depth_maize ~ dnorm(1, 1/0.25) #10% error

#Init_ratio_Ultuna ~ dnorm(0.9291667,1/(0.9291667*0.2)) T(0.8,0.98)
Init_ratio_Ultuna ~ dunif(0.8,0.98)

```

```
138   stubbles_ratio_Ultuna_maize ~ dunif(0.01,0.08)
139   stubbles_ratio_Ultuna ~ dunif(0.01,0.08)
140 
141   C_percent ~ dunif(0.40, 0.51)
142 
143   limits_h<-1
144   limits_k<-1
145   limit_SR<-1
146 
147 }
```

The model starts with a loop for each treatment

```
for(j in 1:J_Ultuna){
```

Where the index "j" is updated for each of the treatments considered in that calibration. Inside this loop there is another loop, nested, that runs for each simulation year "n"

```
  for (i in 1:(N_Ultuna)){
```

Inputs (lines 24 to 38)

The two ifelses

```
    alpha_1[j,i]=ifelse(j==1, 0, alpha)
    alpha_maize_1[j,i]=ifelse(j==1, 0, alpha_maize)
```

are for not using the intercept α when considering the bare fallow. Please note that α is in some calibrations anyway set to zero.

Root inputs are calculated for each year and for each crop type (line 28 to 36). Since the input matrix has zero for each crop that was not that year crop, the sum correspond to the crop of that year.

$$I_{R(c,t)} = (1+\rho) \cdot \epsilon_d \cdot z \cdot \theta \cdot (\alpha \cdot A_{cr} \cdot \frac{1}{S:R_s}) \tag{1}$$

Where: ρ=exudates coefficient ϵ_d= error term ("e_depth_" in the code) z_{cr}= the fraction of roots at 20 cm depth (Fan et al., 2016) θ= C concentration α= the intercept (in some calibrations it is set to zero) A_{cr}= aboveground biomass $S:R$= Shoot to root ratio cr=crop t= year We then sum all the crops for each year

$$I_{R(t)} = \sum_{1...s} I_{R(s,t)} \tag{2}$$

With the code:

```
    I_R_Ultuna[j,i]              <-  I_R_cereals_Ultuna[j,i]+I_R_root_crops_Ultuna[j,i]+I_R_oilseeds_Ultuna[j,i]+I_R_maize_Ultuna[j,i]
```

Shoot inputs are calculated with the stubbles ratio ξ:

$$I_{S(s,t)} = A_s \cdot \xi \cdot \theta \tag{3}$$

With the code:

```
((Yields_cereals_Ultuna[j,i]+Yields_root_crops_Ultuna[j,i]+Yields_oilseeds_Ultuna[j,i])*
                                                stubbles_ratio_Ultuna+Yields_maize_Ultuna[j,i]*stubbles_ratio_Ultuna_maize)*C_percent
```

Amdendment inputs ($I_{n(t)}$) are taken directly as mass of C. Each input is then brought forward in a specific Young pool

Young pool

Each material (roots, shoots, different amendments, all denoted by m) decays into more fine organic matter during the year:

$$Y_{m(t+1)} = \left(I_{m(t)} + Y_{n(t)}\right) \cdot e^{-k \cdot r_e} \tag{4}$$

This is repeated for roots, shoots and each of the amendments. In each of the treatments we calculate all these pools, which are just equal to zero in treatments with no inputs in that class:

```
    Y_R_Ultuna[j,i+1]         <-  (I_R_Ultuna[j,i]+Y_R_Ultuna[j,i])*exp(-k1_ult*re_Ultuna[j,i])
    Y_S_Ultuna[j,i+1]         <-  (I_S_Ultuna[j,i]+Y_S_Ultuna[j,i])*exp(-k1_ult*re_Ultuna[j,i])

    Y_FYM_Ultuna[j,i+1]       <-  (I_FYM_Ultuna[j,i]+Y_FYM_Ultuna[j,i])*exp(-k1_ult*re_Ultuna[j,i])
    Y_GM_Ultuna[j,i+1]          <-  (I_GM_Ultuna[j,i]+Y_GM_Ultuna[j,i])*exp(-k1_ult*re_Ultuna[j,i])
    Y_PEA_Ultuna[j,i+1]       <-  (I_PEA_Ultuna[j,i]+Y_PEA_Ultuna[j,i])*exp(-k1_ult*re_Ultuna[j,i])
    Y_SAW_Ultuna[j,i+1]       <-  (I_SAW_Ultuna[j,i]+Y_SAW_Ultuna[j,i])*exp(-k1_ult*re_Ultuna[j,i])
    Y_SLU_Ultuna[j,i+1]       <-  (I_SLU_Ultuna[j,i]+Y_SLU_Ultuna[j,i])*exp(-k1_ult*re_Ultuna[j,i])
    Y_STR_Ultuna[j,i+1]       <-  (I_STR_Ultuna[j,i]+Y_STR_Ultuna[j,i])*exp(-k1_ult*re_Ultuna[j,i])
```

Fluxes

Each year we also have a flux from each Young pool to the Old (humified) pool:

$$\Phi_{n(t)} = h_m \cdot \frac{k_1 \cdot \left(I_{m(t)} + Y_{m(t)}\right)}{k_2 - k_1} \tag{5}$$

This is also repeated for roots, shoots and each of the amendments. In each of the treatments we calculate also all these fluxes, which are just equal to zero in treatments with no inputs in that class:

```
fluxR_Ultuna[j,i]          <-  h_R_ult*((k1_ult*(Y_R_Ultuna[j,i]+I_R_Ultuna[j,i]))/(k2_ult-k1_ult))
fluxS_Ultuna[j,i]          <-  h_S_ult*((k1_ult*(Y_S_Ultuna[j,i]+I_S_Ultuna[j,i]))/(k2_ult-k1_ult))

flux_FYM_Ultuna[j,i]       <-  h_FYM_ult *((k1_ult*(Y_FYM_Ultuna[j,i]+I_FYM_Ultuna[j,i]))/(k2_ult-k1_ult))
flux_GM_Ultuna[j,i]        <-  h_S_ult   *((k1_ult*(Y_GM_Ultuna[j,i] +I_GM_Ultuna[j,i])) /(k2_ult-k1_ult))
flux_PEA_Ultuna[j,i]       <-  h_PEA_ult *((k1_ult*(Y_PEA_Ultuna[j,i]+I_PEA_Ultuna[j,i]))/(k2_ult-k1_ult))
flux_SAW_Ultuna[j,i]       <-  h_SAW_ult *((k1_ult*(Y_SAW_Ultuna[j,i]+I_SAW_Ultuna[j,i]))/(k2_ult-k1_ult))
flux_SLU_Ultuna[j,i]       <-  h_SLU_ult *((k1_ult*(Y_SLU_Ultuna[j,i]+I_SLU_Ultuna[j,i]))/(k2_ult-k1_ult))
flux_STR_Ultuna[j,i]       <-  h_S_ult   *((k1_ult*(Y_STR_Ultuna[j,i]+I_STR_Ultuna[j,i]))/(k2_ult-k1_ult))
```

Note that, while the Y pool is calculated each year for the following year, the fluxes ϕ are calculated for the current year.

Fluxes are summed all together as

$$\Phi = \sum_{1 \ldots m} \left(\phi_{m(t)}\right) \tag{6}$$

Which translates in the code as:

```
flux_sum_Ultuna[j,i]<-(fluxR_Ultuna[j,i]+
                         fluxS_Ultuna[j,i]+
                         flux_FYM_Ultuna[j,i]+
                         flux_GM_Ultuna[j,i]+
                         flux_PEA_Ultuna[j,i]+
                         flux_SAW_Ultuna[j,i]+
                         flux_SLU_Ultuna[j,i]+
                         flux_STR_Ultuna[j,i])
```

Humified pool

All the fluxes are used to calculate the Old pool:

$$O_{(t+1)} = \left(O_{(t)} - \Phi\right) \cdot e^{-k_2 r_e} + \Phi \cdot e^{-k_1 r_e} \tag{7}$$

Which corresponds int he code to:

```
O_Ultuna[j,i+1]       <-   (O_Ultuna[j,i]-flux_sum_Ultuna[j,i])*exp(-k2_ult*re_Ultuna[j,i]) +
                               flux_sum_Ultuna[j,i]*exp(-k1_ult*re_Ultuna[j,i])
```

Total C

Total C is calculated by summing up together all the Young pools with the Old pool at each time step:

$$SOC_{(t)} = \sum_{1...n} (Y_{n(t)}) + O_{(t)} \tag{8}$$

In the code this is achieved by first summing up together the Young pools and then adding the old pool

```
#Total C
Y_tot_Ultuna[j,i] <- Y_R_Ultuna[j,i] +
                      Y_S_Ultuna[j,i] +
                      Y_FYM_Ultuna[j,i] +
                      Y_GM_Ultuna[j,i] +
                      Y_PEA_Ultuna[j,i] +
                      Y_SAW_Ultuna[j,i] +
                      Y_SLU_Ultuna[j,i] +
                      Y_STR_Ultuna[j,i]

Tot_Ultuna[j,i] <- Y_tot_Ultuna[j,i] + O_Ultuna[j,i]
```

Comparing with the results

The simulation ("Tot_Ultuna") is then compared with the measured values ("SOC_Ultuna"). Of course this is done each loop step for the ith (year) and jth (treatment) positions. The error in each point is estimated as the error of each measured time series (so the error of a linear regression on that specific time series), increasef by 50% for additional safety.

$$SOC_{measured} \sim \mathcal{N}(SOC_{simulated}, 1.5\sigma_{SOC_{measured}}) \tag{9}$$

In the code this is rendered as:

```
SOC_Ultuna[j,i]  ~ dnorm(Tot_Ultuna[j,i],1/(error_SOC_Ultuna[j]*1.5))
```

Please notice that JAGS uses precision and not error for the normal distribution, hence the error is expressed as $\frac{1}{\sigma}$. The distribution is here assumed as gaussian (maybe uniform would be more conservative)

The priors

Outside the two loops, after line 97, all priors are specified. The JAGS syntax is pretty intuitive in this sense. The uniform distribution has as parameters just minimum and maximum:

```
param  ~ dunif(min, max)
```

While the normal distribution:

```
param  ~ dnorm(mean, precision)
```