Science Advances

AAAS

advances.sciencemag.org/cgi/content/full/3/4/e1601880/DC1

Supplementary Materials for

The economic value of grassland species for carbon storage

Bruce A. Hungate, Edward B. Barbier, Amy W. Ando, Samuel P. Marks, Peter B. Reich, Natasja van Gestel, David Tilman, Johannes M. H. Knops, David U. Hooper, Bradley J. Butterfield, Bradley J. Cardinale

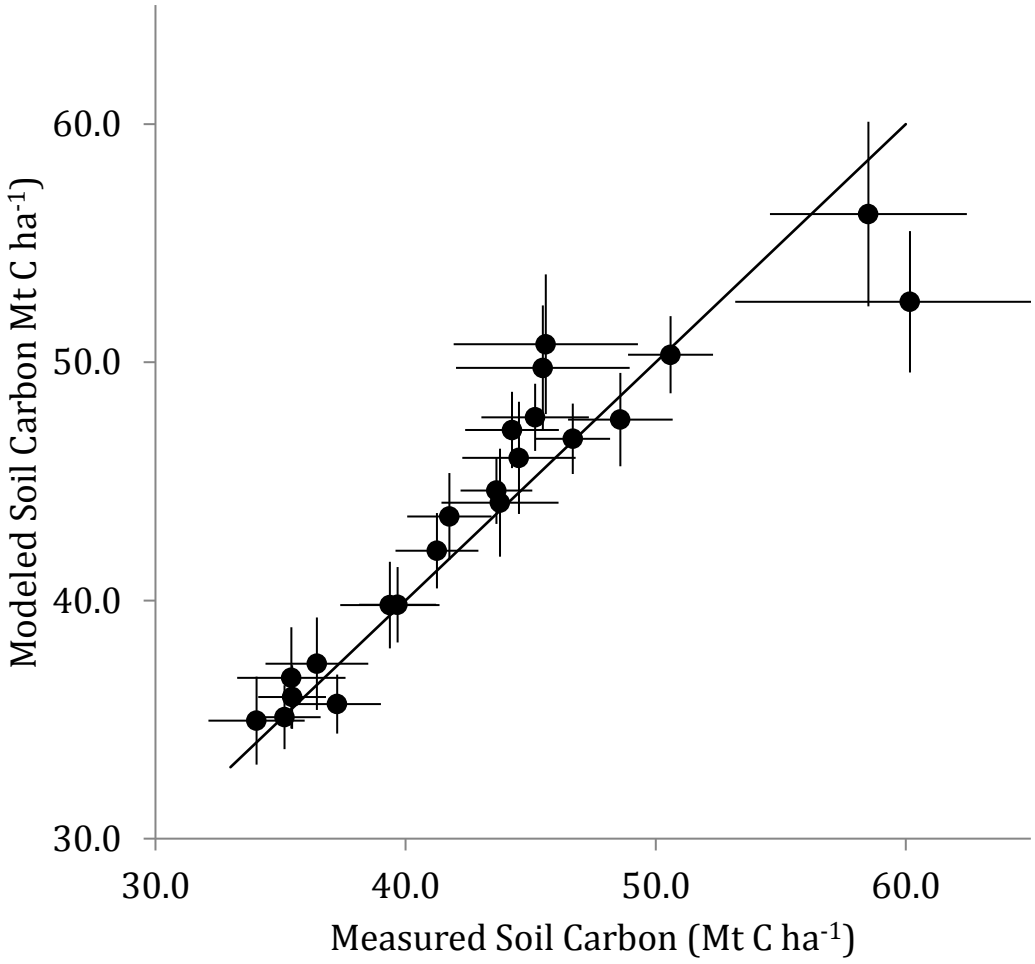
Published 5 April 2017, *Sci. Adv.* **3**, e1601880 (2017)
DOI: 10.1126/sciadv.1601880

This PDF file includes:

Supplementary Materials

S1. Measured soil carbon in the BioCON and BigBio experiments compared to soil carbon modeled using data assimilation and the *I-k* model

Points show mean soil carbon content (for the depth, 0-100 cm) for each level of species richness at each time point when soil carbon was measured directly.



S2. Michaelis-Menten parameters for plant, soil (at year 50), and total C (at year 50) as a function of species richness

Parameters a and b were estimated for the model, C = (aS)/(b+S), where C is the carbon stock, S is species richness, and a and b are the Michaelis-Menten scaling parameters. Parameters were estimated using bootstrapping and optimization to minimize the residual sums of squares. 5000 bootstrap iterations, each bootstrap preserved the original experimental design. Confidence limits are 2.5% and 97.5%. Units are metric tons C per hectare.

		a	b
E120 (BigBio)	Plant	9.01 (7.92 to 10.32)	1.805 (1.186 to 2.636)
	Soil	50.04 (46.29 to 54.09)	0.297 (0.139 to 0.481)
	Ecosystem	58.4 (54.28 to 62.64)	0.403 (0.245 to 0.585)
E141 (BioCON)	Plant	6.58 (5.55 to 8.81)	1.353 (0.571 to 3.757)
	Soil	77.79 (62.62 to 99.83)	0.612 (0.116 to 1.39)
	Ecosystem	83.89 (68.27 to 106.34)	0.632 (0.151 to 1.379)

S3. List of species (by binomial name, common name, and functional group) included in the BioCON and BigBio experiments at the Cedar Creek Long-Term Ecological Research (LTER)

Range (hectares) shows current geographic distribution as included in the Global Biodiversity Information Facility (*50*)

Name	Common Name	Functional Group	Experiment	Range
Achillea millefolium	Yarrow	Non-Leguminous Forb	BioCON	9,594,800
Agropyron repens	Quack grass	Cool Season (C3) Grass	BioCON	40,800
Agropyron smithii	Western Wheatgrass	Cool Season (C3) Grass	BigBio	51,100
Amorpha canescens	Lead plant	Legume	Both	45,400
Andropogon gerardii	Big bluestem	Warm Season (C4) Grass	Both	75,200
Anemone cylindrica	Candle anemone	Non-Leguminous Forb	BioCON	31,800
Asclepias tuberosa	Butterfly milkweed	Non-Leguminous Forb	Both	64,300
Bouteloua gracilis	Blue grama	Warm Season (C4) Grass	BioCON	100,200
Bromus inermis	Smooth brome	Cool Season (C3) Grass	BioCON	1,056,700
Elymus canadensis	Canada wild rye	Cool Season (C3) Grass	BigBio	91,700
Koeleria cristata	Junegrass	Cool Season (C3) Grass	Both	62,200
Lespedeza capitata	Roundhead bush clover	Legume	Both	36,500
Liatris aspera	Blazing star	Non-Leguminous Forb	BigBio	20,900
Lupinus perennis	Lupine	Legume	Both	19,000

Monarda fistulosa	Bee balm	Non-Leguminous Forb	BigBio	59,900
Panicum virgatum	Switch grass	Warm Season (C4) Grass	BigBio	90,200
Petalostemum candidum	White prairie clover	Legume	BigBio	68,800
Petalostemum purpureum	Purple prairie clover	Legume	BigBio	61,400
Petalostemum villosum	Silky prairie clover	Legume	Both	8,400
Poa pratensis	Kentucky bluegrass	Cool Season (C3) Grass	Both	8,459,900
Schizachyrium scoparium	Little bluestem	Warm Season (C4) Grass	Both	79,700
Solidago rigida	Rigid goldenrod	Non-Leguminous Forb	BioCON	30,700
Sorghastrum nutans	Indian grass	Warm Season (C4) Grass	Both	56,900

S4. R-code for estimating marginal carbon and value for the BioCON experiment

```
# Set working directory as appropriate

library(plyr)
library(reshape2)
source("Optimize_Michaelis-Menten_weighted_rev4.R")


#_____________________________________________________________________________
#_____________________________ C POOLS, FLUXES, & VALUE_____________________________

d = read.csv("/BigBio_BioCON_total.csv")                # Import the data set (53 columns, 3 for ID, then 50,
                                                        # one for each year), by 259 rows, one header, the rest for data
biocon <- subset(d, site=="biocon")                     # subset the data from the biocon experiment


results_biocon <- matrix(data=NA,nrow=0,ncol=5)         # creates an empty matrix to hold bootstrapped results


for (j in 1:1000){                                      #beginning of bootstrap procedure. 1000 iterations



    d1 <- matrix(NA, nrow = 0, ncol = 53)               # define d1 as empty matrix into which bootstrap
                                                        # results will go

    for (i in 1:16){
            e <- biocon[sample( which( biocon$S==i),            #re-sample data with replacement from biocon, such that the sample
                length(which(biocon$S==i)) ,replace=TRUE),]     #size is defined by the number of actual replicates
            d1<- rbind(d1,e)                                    #for each level of species richness, 1-16
    }

        m<-melt(d1, id.vars = c("site","plot","S"))             #reformat the bootstrapped output so the
                                                                    #data is in one column
        results_biocon.tmp = ddply(m, .(variable), function(x) MM.optim.weighted(x[,3], x[,5], df=T))
                                                            #apply the function, MM.optim.weighted,
                                                            # which estimates a and b in the
                                                            # Michaelis-Menten model for carbon
                                                            # content, C = (S x a)/(S + b)
                                                            #using "variable" as the grouping factor
                                                            #and using columns 3 (species richess)
                                                            # and 5 (carbon content) as the x- and
                                                            #y-variables for the relationship
    for (i in 1:16){
            C_stock.tmp <- (i*results_biocon.tmp[,3])/(i+results_biocon.tmp[,4])
            results_biocon.tmp <- cbind(results_biocon.tmp,C_stock.tmp)
            colnames(results_biocon.tmp)[i+5] <- paste("Cstock", i, sep = "")
    }                                                       #use the M-M parameters to calculate
                                                            #predicted C stocks for each level of
                                                            #species richness, 1-16
                                                            #put results in a data table
                                                            #called "results_biocon.tmp"
    for (i in 1:15){
            Marg_C.tmp <- (results_biocon.tmp[,i+6] - results_biocon.tmp[,i+5])
                                                            #calculate cumulative marginal C content, as Cstock for
                                                            #S=i minus C stock for S=i-1

            results_biocon.tmp <- cbind(results_biocon.tmp,Marg_C.tmp)          #put the results in extra columns
            colnames(results_biocon.tmp)[i+21] <- paste("CMC", i+1, sep = "")   #give the columns names
    }
annual <- results_biocon.tmp[1,22:36]
    for (i in 1:49){
          tmp <- results_biocon.tmp[i+1,22:36]-results_biocon.tmp[i,22:36]      #calculate annual marginal C content, as
            annual <- rbind(annual,tmp)                                         #cumulative marginal C for year i+1 minus
            }                                                                   #cumulative marginal C for year i
                                                                                #do this for years 2-50 and put the results
                                                                                #in rows
                                                                                #note: annual marginal C for year 1 is the
                                                                                #same as cumulative marginal C for year 1


results_biocon.tmp <- cbind(results_biocon.tmp,annual)                          #add the annual marginal C to the other data
    for (i in 1:15){                                                            #as new columns
        colnames(results_biocon.tmp)[i+36] <- paste("AMC", i+1, sep = "")       #name the new columns
    }
results_biocon <- rbind(results_biocon, results_biocon.tmp)                     #add all the rows of data to the original dataframe
                                                                                #this step will repeat for each bootstrap, adding
                                                                                #50 new rows of data (for the 50 years)
```

```
                                                                    #so, the output file, "results_biocon", will have
                                                                    #50,000 rows (50 years x 1000 boostraps)

                                                                    # Keep track of progress by printing the count of
 every
                                                                    # 50th bootstrap iteration
print.every.n.steps = 50
  if((j/print.every.n.steps) %% 1 == 0 ) {                          # if j can be divided by 50 (produces integer)
    print(j)
  }
}                                                                   # end bootstrap procedure



names(results_biocon)[1] <- "year"                                  # rename year column

results_biocon$year = as.numeric(gsub("[^\\d]+", "", results_biocon$year, perl=TRUE))
                                                                    # Extract digits, i.e. year (converts e.g.
                                                                    # X1 to 1 and X50 to 50)
results_biocon <- results_biocon[,-2]                               # remove extra column



for (i in 1:15) {                                                   #this calculates marginal value (per species)
    s1 <- 137.26*exp(-0.04*results_biocon[,1])*results_biocon[,i+35] #of C stored for each year, using the annual
 marginal
    results_biocon <- cbind(results_biocon,s1)                      #rate of C uptake (begins in column 36, or 35+i)
    colnames(results_biocon)[i+50] <- paste("MedAnnValue", i+1, sep = "")  #times the price of carbon (here, medium, using the
    }                                                               #3% average price from the Interagency Working
 Group,
                                                                    #adjusted to 2010 dollars, $137.26), discounted at
                                                                    #4% per year. Append values to dataframe as new
 columns
                                                                    #Give the columns names

for (i in 1:15) {                                                   #same as above, but now for the low social cost of
    s2 <- 41.94*exp(-0.04*results_biocon[,1])*results_biocon[,i+35] #carbon estimate ($41.94)
    results_biocon <- cbind(results_biocon,s2)
    colnames(results_biocon)[i+65] <- paste("LowAnnValue", i+1, sep = "")
    }

for (i in 1:15) {                                                   #same as above, but now for the high social cost of
    s3 <- 400.33*exp(-0.04*results_biocon[,1])*results_biocon[,i+35] #carbon estimate ($400.33)
    results_biocon <- cbind(results_biocon,s3)
    colnames(results_biocon)[i+80] <- paste("HighAnnValue", i+1, sep = "")
    }



# CALCULATE SUMMARY STATISTICS FOR results_biocon (C stocks, cumulative marginal C, annual marginal C)
# SUMMARIZE ALL VALUES BY YEAR: MEAN, MEDIAN, 2.5% CL, 97.5% CL
# WRITE TO CSV FILE

melted <- melt(results_biocon, id.vars=c("year"))                   #reorganizes the data to a format amenable to ddply
 command
names(melted)[2:3] <- c("response","value")                         #names colums
z <- ddply(melted, c("year", "response"), summarise,                #calculates mean and median
      mean = mean(value), median = median(value))
z1 <- ddply(melted, c("year","response"), summarise, LCI = quantile(value,  #calculates 2.5% and 97.5% percentiles as lower and
 upper
       probs = 0.025), UCI = quantile(value,0.975))                 #confidence limits
z2 <- cbind(z[1:4],z1[3:4])                                         #collects the output into one file with mean,
                                                                    #median and CLs
write.csv(z2,file = "biocon annual means & CIs")                    #writes a csv file containing the output

#this sums values over the 50 year period. creates new data frame called "value"
index <- rep(1:1000,50)                                             #this creates 50 repetitions of the integer
                                                                    #series 1 to 1000
index <- index[order(index)]                                        #this reorders the series just created, so it gives
 fifty
valuebc <- matrix(data=NA,nrow=1000,ncol=0)                         #values of 1, followed by fifty 2s, etc.
                                                                    #up to 1000

for (i in 1:15)     {                                               #sums values of accumulated carbon over 50 year
```

```
period
            testb <- aggregate(x=results_biocon[,i+50], by = list(index), FUN=sum)  #for medium SCC value ($137.26/ton)
            valuebc <- cbind(valuebc,testb[,2])                                     #for each marginal change in species (1 to 2
            colnames(valuebc)[i] <- paste("CumValueMed_BC", i+1, sep = "")          #2 to 3, ... 15 to 16); gives the columns names
                }
for (i in 1:15)     {
            testb <- aggregate(x=results_biocon[,i+65], by = list(index), FUN=sum)  #same as above, for low SCC value ($41.94/ton)
            valuebc <- cbind(valuebc,testb[,2])
            colnames(valuebc)[i+15] <- paste("CumValueLow_BC", i+1, sep = "")
                }
for (i in 1:15)     {
            testb <- aggregate(x=results_biocon[,i+80], by = list(index), FUN=sum)  #same as above, for high SCC value ($400.33/ton)
            valuebc <- cbind(valuebc,testb[,2])
            colnames(valuebc)[i+30] <- paste("CumValueHigh_BC", i+1, sep = "")
                }


meltedvaluebc <- melt(valuebc)                                                      #reorganizes the data to be amenable to ddply
names(meltedvaluebc)[2] <- "responsevar"                                            #names the identifier column
v <- ddply(meltedvaluebc, c("responsevar"), summarise,                              #calculates mean, median, 2.5% CL, and 97.5% CL
      mean = mean(value), median = median(value))
v1 <- ddply(meltedvaluebc, c("responsevar"), summarise, LCI = quantile(value, probs = 0.025), UCI = quantile(value,0.975))
                                                                                    #calculates 2.5% CL, and 97.5% CL
v2 <- cbind(v,v1[2:3])                                                              #combines the columns into a single dataframe
write.csv(v2,file = "biocon cumulative value & CIs")                                #writes the output to csv file
```

S5. R-code function for estimating Michaelis-Menten parameters relating species richness to carbon stocks

```
## Function to optimize parameters for Michaelis-Menten (MM) relationship using
#       observations weighted by the inverse of their variance
#
#   Usage:  parList = MM.optim.weighted(x,y,w)
#           x = vector of x-values
#           y = vector of y-values
#           w = vector of weights; default: w=1 (if unweighted)
#-----------------------------------
#       parList = output list (see further below for equation):
#           a = parameter 1 (in numerator)
#           b = parameter 2 (denominator)
#           Michaelis Menten: y = (a*x) / (b+x)
#   Natasja van Gestel 4/14/14


MM.optim.weighted=function(x,y,w=1,df = F) {

# cbind x with y and remove NAs if present
 data=cbind(x,y,w)
     if(any(is.na(data))) {
        pos=which(is.na(x) | is.na(y))
        data=data[-pos,]
        x=data[,1]
        y=data[,2]
        w=data[,3]
     }

 # if all w's are one (i.e. unweighted, then skip this portion)

 if (all(w!=1)) { # only if weights differ
    # Data need to be present in proportional abundance to 1/w
    # Identify value of highest weight, because this has lowest variance or best fit
    w.max=max(w)
    new.w = w/w.max
    n = 1000*new.w  # value of lowest variance will be valued at 100, everything else is
                    # less than that, in proportion to 1/variance

    # Expand data frame to reflect new "sample sizes"
    expanded = data.frame(x = rep(x, n), y = rep(y, n))

    # New x and y
    x=expanded$x
    y=expanded$y
}

# Create function to find the two parameters (a and b) that minimize sums-of-squares
 sumsOfSquares=function(parm, x, y) {
 pred.y = (parm[1]*x) / (parm[2]+x)
 sum((y-pred.y)^2)
 }

# Estimate starting values for a and b, respectively
 start.values=c(min(y), min(x))

# Optimize by minimizing sums of squares
 output = optim(par=start.values, fn=sumsOfSquares, x=x, y=y, method="Nelder-Mead", control=list(maxit=10000))

 # Use Nelder-Mead output values as input to BFGS
 start.values= output$par
 output = optim(par=start.values, fn=sumsOfSquares, x=x, y=y, method="BFGS", control=list(maxit=10000))

 # If output is negative, then put bounds on the start.values
 if (output$par[1]<0 | output$par[2]<0) {
 start.values=output$par # from BFGS
 output = optim(par=start.values, fn=sumsOfSquares, x=x, y=y, lower=rep(0,2), method="L-BFGS-B", control=list(maxit=10000))
 }

# Return convergence, a, b and sums-sof-squares

 if(df) {
 return(data.frame(convergence=output$convergence, a = output$par[1], b=output$par[2], sums.of.squares=output$value))
 } else {
 return(list(convergence=output$convergence, a = output$par[1], b=output$par[2], sums.of.squares=output$value))
 }
} # end function
```