The challenge of feeding the world while conserving half the planet

Zia Mehrabi, Erle C. Ellis and Navin Ramankutty

zia.mehrabi@ubc.ca

August 8, 2018

Supplementary Information

1 Supplementary Notes

1.1 Reproducibility

We use the **R** packages `knitr` and `checkpoint` [25, 8]. The package `knitr` facilitates producing a dynamic document that contains all the steps required to analyze the data. `checkpoint()` will install all packages versions for you that we used in our analysis to avoid result discrepencies that may arise from software differences. Thus the reader is provided with all the code to fully reproduce the analysis, and adapt or repurpose it for another analyses.

```
library("checkpoint")
checkpoint(snapshotDate = "2018-03-01")
```

For the analysis in this document we will be using the `raster` [13], `rgdal` [5], `ggplot2` [22], `plyr` [20], `sp` [14], `R.matlab` [3], `rworldmap` [17], `car` [11], `maptools`[6] , `reshape2`[21] ,`gridExtra`[1],amd ,`cowplot`[23], and, `data.table`[10] packages.

```
packages<-c("raster", "rgdal", "ggplot2", "plyr", "sp",
            "R.matlab", "rworldmap", "car", "maptools",
            "knitr", "reshape2","gridExtra","cowplot", "data.table")
lapply(packages, require, character=TRUE)
gpclibPermit()
```

1.2 Aims of this document

The aim of this document is to provide the code to reproduce the analysis presented in Mehrabi, Ellis and Ramankutty 2018 entitled "The challenge of feeding the world while conserving half the planet". Half-Earth is a recent proposal to devote half the surface of the Earth to nature. (http://www.half-earthproject.org/; http://natureneedshalf.org/). The underlying rationale for Half-Earth derives from scaling relationships between habitat area and species numbers which imply that conserving half of Earth's habitat area would save 85% of existing species on the planet [24]. The practical costs of Half-Earth incurred through trade-offs with other land uses, such as agriculture, and its impacts on already disadvantaged populations around the world still remain poorly understood [2]. This analysis aims to help fill this research gap.

The nuts and bolts of this analysis is a simple thought experiment constructed to better understand some of the broad trade-offs between agriculture and Half-Earth proposal. The document is structured into two sections, in section 2 we process the data, in section 3 we conduct the thought experiment, create the plots used in the figure in the main text of the manuscript, and conduct some additional analysis, and finally in section 4 we plot the Supplementary Figures. For brevity the code in section 2 is not evaluated in this document.

2 Supplementary Methods A

2.1 Data preparation overview

To assess the trade-offs between Half-Earth and agriculture we compiled 8 spatial datasets. Details of each and the currently live urls and points of contact for data set acquisition are listed below. In some cases these data are not available for us to redistribute, and so those wishing to reproduce this analysis are recommended to approach the persons responsible, and back-date to the dates of acquisition listed below.

1. Global Cropland and Pasture Area [16], retrieved online from www.earthstat.org , on December 13th 2017.
2. MODIS land cover [12], retrieved online from http://glcf.umd.edu/data/lc/, on December 17th 2017.
3. Calorie production for the worlds 41 major crop plants,[7] retrieved directly from the original authors, available from Minnesota Insitute on the Environment (jsgerber@umn.edu)
4. The World Database of Protected Areas, [19], retrieved online from https://www.protectedplanet.net/, on December 13th 2017.
5. Key Biodiversity Areas 2017 [4] retrieved via FTP link from Gill Bunting at Bird Life International (Gill.Bunting@birdlife.org) on Nov 6th 2017.
6. Ecoregion boundaries,[9], retrieved online from https://storage.googleapis.com/teow2016/Ecoregions2017.zip, on April 2017
7. Country boundaries [17], retrieved internally in R.
8. Potential Natural Vegetation, [15], retrieved online from on April 30th 2018, from www.earthstat.org.

All datasets were rasterized, projected to Eckert IVs equal area at a spatial resolution of 8.4km x 8.4 km (the most accurate scale for reproduction of global sums for data set [7], and set to a common extent). This section explains that processing.

2.2 Rasterizing

2.2.1 Ecoregions

First we read in the ecoregion data, rasterize it, and write to file.

```
Dinn<-shapefile("halfearthdata/Ecoregion2017_Erle/Ecoregions2017.shp")
raster.Dinn<-raster(resolution = 0.0833282)
extent(raster.Dinn) <- extent(Dinn)
Dinn$ECO_NAME<-as.factor(Dinn$ECO_NAME)
Dinn.raster <- rasterize(as(Dinn, "SpatialPolygons"), raster.Dinn, field = Dinn@data[,"ECO_NAME"],fun="first")
getwd()
Dinn.rast <- writeRaster(Dinn.raster, "halfearthdata/processed/Dinnrast.tif", format="GTiff", overwrite=TRUE)
```

2.2.2 KBA

Then we read in the KBA data, rasterize it, and write to file.

```
KBA<-shapefile("KBAsGlobal_2017_1a/Global_KBA_poly.shp")
raster.KBA<-raster(res = c(0.0833282,0.0833282))
extent(raster.KBA) <- extent(KBA)
KBA.raster <- rasterize(as(KBA, "SpatialPolygons"), raster.KBA, field = KBA@data[,"SITRECID"],fun="first")
kbavals<-values(KBA) #set non KBA values to NA
kbavals2<-ifelse(kbavals>0,1,NA)
KBA2<-KBA
values(KBA2)<-kbavals2
KBA.rast<- writeRaster(KBA2, "halfearthdata/processed/KBArast.tif", format="GTiff", overwrite=TRUE)
```

2.2.3 Crop calories

Then we read in the global crop calorie data, and write to file. These data are supplied as .mat files.

```
Cass.total<-readMat("halfearthdata/Emily_data/Cassidy Crop Allocation and Delivery/GlobalTotalkcal.mat")
rasterD <-raster(ncol=4320, nrow=2160)
values(rasterD)<-c(Cass.total$GlobalTotalkcal)
kcal.41<-rasterD
plot(kcal.41)
kcal.rast<- writeRaster(kcal.41, "halfearthdata/processed/kcalrast.tif", format="GTiff", overwrite=TRUE)
```

We also retrieve the calorie delivery fraction data from Cassidy et al. A unit of 1 identifies that all kcal produced are delivered as food, wheras a unit of <1 identifies that some proportion of calories are fed to animals from that pixel.

```
Cass.food<-readMat("halfearthdata/Emily_data/Cassidy Crop Allocation and Delivery/GlobalFoodkcal.mat")
rasterD <-raster(ncol=4320, nrow=2160)
# Cass.food£GlobalFoodkcal[is.na(Cass.food£GlobalFoodkcal)] <- 0 #replace NA with zero
values(rasterD)<-c(Cass.food$GlobalFoodkcal)
kcal.food<-rasterD
kcal.food<- writeRaster(kcal.food, "halfearthdata/processed/kcal.food.tif", format="GTiff", overwrite=TRUE)
```

2.2.4 WDPA

Here we read in the world protected area data. This data was first imported into QGIS [18], to simplify the geometry with the default tolerance. It was then rasterized it in QGIS, to reduce processing time in R. We set non-WDPA values in this raster to NA, and write the result to file.

```
WDPA<-raster("halfearthdata/WDPA_Oct2017/WDPA.tif")
str(WDPA)
wdpavals<-values(WDPA)#set non WDPA values to NA
wdpavals2<-ifelse(wdpavals>0,1,NA)
WDPA2<-WDPA
values(WDPA2)<-wdpavals2
WDPA.rast<- writeRaster(WDPA2, "halfearthdata/processed/WDPArast.tif", format="GTiff", overwrite=TRUE)
```

2.2.5 Countries

Finally we make a raster of the world country data from the `rworldmap` package.

```
world<-getMap(resolution='low')
raster.world<-raster(res = c(0.0833282,0.0833282))
extent(raster.world) <- extent(world)
world.raster <- rasterize(as(world, "SpatialPolygons"), raster.world, field = world@data[,"ADMIN"],fun="first")
world.rast<- writeRaster(world.raster, "halfearthdata/processed/worldrast.tif", format="GTiff", overwrite=TRUE)
```

2.3 Additional processing

2.3.1 Read data

Here we read in the data for additional processing. We read in the processed data sets outlined above, as well as two others. These include the MODIS LC data set as an indication of land cover, and potential natural vegetation.

```
KBA<-raster("halfearthdata/processed/KBArast.tif")
CA<-raster("halfearthdata/CroplandPastureArea2000_Geotiff/cropland2000_area.tif")
PA<-raster("halfearthdata/CroplandPastureArea2000_Geotiff/pasture2000_area.tif")
Dinn<-raster("halfearthdata/processed/Dinnrast.tif")
kcal<-raster("halfearthdata/processed/kcalrast.tif")
kcal.food<-raster("halfearthdata/processed/kcal.food.tif")
WDPA<-raster("halfearthdata/processed/WDPArast.tif")
world<-raster("halfearthdata/processed/worldrast.tif")
LC<-raster("halfearthdata/MODIS_GLCF/LC_hd_global_2012.tif")
PNV<-raster("halfearthdata/potveg_geotiff/potentialvegetation_geotiff/potentialvegetation.tif")
```

2.3.2 Convert to equal area

Here we project everything to be equal area (set to 8439m x 8439m, as this is with the best match for the kcal data, it is within 0.001%). We use two methods of interpolation, bilinear for continuous data and nearest neighbour for categorical data.

```
rast.list.ngb<-list(Dinn,KBA, WDPA, LC, world, PNV)
rast.list.bil<-list(kcal,CA,PA, kcal.food)

rast.list.ngb.eq<-lapply(rast.list.ngb, function(x)
  projectRaster(x, res=c(8439, 8439),
                crs="+proj=eck4 +datum=WGS84 +ellps=WGS84 +towgs84=0,0,0",method="ngb", over = T))

rast.list.bil.eq<-lapply(rast.list.bil, function(x)
  projectRaster(x, res=c(8439, 8439),
                crs="+proj=eck4 +datum=WGS84 +ellps=WGS84 +towgs84=0,0,0",method="bilinear", over = TRUE))

rast.list<-c(rast.list.ngb.eq, rast.list.bil.eq) #combine data
names(rast.list)<-c("Dinn","KBA", "WDPA", "LC", "world", "PNV","kcal","CA", "PA","kcal.food")
```

Now we double check that the sum of the original crop calorie data is equivalent to the equal area representation. We find these are very close to each other, and so the equal area conversion is reasonable in aggregate for this variable.

```
sum(values(rast.list.bil.eq[[1]]), na.rm = T) #check kcal values
sum(values(kcal), na.rm = T)
```

2.3.3 Back corrections

No reprojection of geospatial data is perfect. In some cases it can lead to nonsensical outputs. A good example of this is shown here with cropland and pasture data, where the sum of the original lat-long data on proportions of cropland and pasture in a cell is never greater than zero, but where for the derived products the summed area can exceed one. This is logically impossible in reality, but is an artifact created through the reprojection.

```
plot(CA+PA) #original lat long
plot(rast.list$PA+rast.list$CA) #eck 4 version
```

The details of the net differences between the aggregate sums of equal area projected and lat-long version are given below. As we can see reprojection reduced the global area of cropland by 2% and increased it for pasture by 1%.

```
cell.area=8439*8439 #area of grid cell
sum(values(rast.list$CA)*cell.area, na.rm=T)/sum(values(CA)*cell.area, na.rm=T)
sum(values(rast.list$PA)*cell.area, na.rm=T)/sum(values(PA)*cell.area, na.rm=T)
```

It is difficult to deal with these two issues easily in one fail swoop. It is possible to calibrate using the original data sources pretty easily. This is done in the chunk of code below for continuous data. But this does not deal with the nonsensical output of having proportional coverage in pixels greater than one.

```
values(rast.list$CA)<-(values(rast.list$CA)/sum(values(rast.list$CA),na.rm=T))*sum(values(CA), na.rm=T)#CA
values(rast.list$PA)<-(values(rast.list$PA)/sum(values(rast.list$PA),na.rm=T))*sum(values(PA), na.rm=T) #PA
values(rast.list$kcal)<-(values(rast.list$kcal)/sum(values(rast.list$kcal),na.rm=T))*sum(values(kcal), na.rm=T) #food kcal
values(rast.list$kcal.food)<-(values(rast.list$kcal.food)/
 sum(values(rast.list$kcal.food),na.rm=T))*sum(values(kcal.food), na.rm=T) #kcal
```

One work around, is to ensure that no sum of proportional areas in pixels exceeds one. We do this below, and preferentially reduce pasture areas within each pixel. A more exact approach, would be to write a spatial reallocation algorithm that deals with both of these issues together. However, for the purposes of this thought experiment, and for the specific results we provide, the approach taken here seems reasonable. Indeed, as we can see from below, not only have we now achieved sensible plot values, but we still have maintained our pasture aggregate sums to a high degree – with our reprojected pasture areas being only 0.02% less than the original lat long file. By focusing only on pasture in correcting for proportional coverage, the aggregate sums of all other continuous variables is maintained.

```
values(rast.list$PA)<-ifelse(values(rast.list$PA)+values(rast.list$CA)>1,1-values(rast.list$CA), values(rast.list$PA))
plot(rast.list$PA+rast.list$CA)
sum(values(rast.list$PA)*cell.area, na.rm=T)/sum(values(PA)*cell.area, na.rm=T)
```

2.3.4 Set extents

Next we need to trim the extents. We can see from the plots below, and in the README for the MODIS LC data, that the LC data does not include Antarctica. In fact it only covers southern latitudes to -64 degrees.

```
par(mfrow = c(1,1))
lapply(rast.list, function(x) plot(x, main=names(x))) #check
```

Therefore, we need to expand the LC data to include the area that will house Antarctica. We draw from the country data which includes a boundary for Antarctica. As explained by Prof. Rob Hijmans in his manual for the package (http://rspatial.org/spatial/rst/8-rastermanip.html), the extend function fills NA's in rows that are added. We couple this with the crop function to maintain equal extents across all datasets, then apply the setExtent function to ensure that rounding is exactly the same across the files.

```
ex1<-extent(rast.list$Dinn)
rast.list.ex<-lapply(rast.list, function(x) extend(x, ex1)) #extend
rast.list.c<-lapply(rast.list.ex, function(x) crop(x, ex1)) #then crop
rast.list.et<-lapply(rast.list.c, function(x) setExtent(x, ex1, keepres=TRUE)) #then force extent to be the same
```

2.3.5 Create stack

With the data all with the same resolution, extent and equal area CRS, we can now create a stack of all the layers.

```
rast.all<-stack(rast.list.et)
```

Now let's check the plots again.

```
plot(rast.all) #check
```

2.4 Land class variable

2.4.1 Cropland and pasture

We now need to make a common land class variable that merges cropland and pasture area data and MODIS LC data. We will use the Ramankutty et al. identifiers for cropland and pasture data.

We create a numeric code for each cropland, pasture and cropland-pasture mix and add this to the land class variable. For cropland we assign 18, for the pasture-cropland mix we assign 17 , and for pasture we assign 19.

```
newclass<-ifelse(values(rast.all$CA) >0 & values(rast.all$PA) >0,17,
                 ifelse(values(rast.all$CA) > 0 & values(rast.all$PA) %in% c(0,NA)  ,18,
                        ifelse(values(rast.all$PA) >0 & values(rast.all$CA)   %in% c(0,NA) , 19,
                               values(rast.all$LC))))
summary(as.factor(newclass)) #check
```

2.4.2 Antarctica

We need to specify Antarctica in the LC data .Antarctica is identified under an id of 9 in the country data. We therefore designate Antarctica an id of 15, to correspond to the id for "Rock and Ice" in the README given to accompany the LC file. We then check this by plotting, it looks good. We then add this to the newclass variable. [We add this in here because doing it before defining cropland, mixed and pasture pixels, would lead to it being lost due to NAs in CA and PA data sets].

```
values(rast.all$LC)<-ifelse(values(rast.all$world)==9, 15, values(rast.all$LC))
plot(rast.all$LC) #check Antarctica has been added
newclass<-ifelse(values(rast.all$LC)==15, 15, newclass) #add to newclass
```

2.4.3 WDPA and KBA

We then assign numeric values for the WDPA and KBA areas: 20, and 21 respectively. These may include agricultural production, but we will give these areas top priority in any Half-Earth Scenario for conservation. We keep the ocean as zero.

```
newclass<-ifelse(is.na(values(rast.all$KBA))|values(rast.all$LC)==0, newclass, 21)
newclass<-ifelse(is.na(values(rast.all$WDPA))|values(rast.all$LC)==0, newclass, 20)
```

2.4.4 Urban

We also maintain urban areas with agricultural production as urban.

```
newclass<-ifelse(values(rast.all$LC)==13, 13, newclass)
```

2.4.5 Add new LC

We then create a new layer in the stack for the new land class variable

```
rast.all$newclass<-rast.all$LC
values(rast.all$newclass)<-as.numeric(newclass)
summary(values(rast.all$newclass)) #check
```

And then join the data into a single data frame.

```
data.all<-data.frame(coordinates(rast.all), values(rast.all))
summary(as.factor(data.all$newclass))
```

2.4.6 Rank land classes

Finally we recode and rank the land cover variables into a more limited number of classes. The rank represents the order in which we will give back individual pixels ("nature only landscape approach"), or area within pixels ("Shared landscape approach"). All agricultural areas are given back in order of least productivity. We define agricultural productivity based on calories gained from 41 major arable crops. The ranking for this hypothetical deal with nature is given below. The order of this ranking is not supposed to be definitive, but simply to highlight broad scale the trade-offs between Half-Earth and agricultural production. We assign water to be NA.

1. World protected areas
2. Key Biodiversity Areas
3. Non-agricultural lands (Forests, Wetlands, Shrublands and savannas, Grasslands, Snow, Ice, Barren)
4. Agricultural lands (Pasture)
5. Agricultural lands (Mixed)
6. Agricultural lands (Cropland)
7. Urban

```
data.all$rank<-recode(data.all$newclass,
 " c('0')= NA;
   c('1','2','3','4','5','6', '7','8','9' ,'10', '11','12', '14','15', '16') = '3';
   c('13')= '7';
   c('17')= '6';
   c('18')= '5';
    c('19')= '4';
   c('20')= '1';
   c('21')= '2'
  ")
summary(as.factor(data.all$rank)) #check
```

2.4.7 Rank by calories

We then need to compute the calories in each pixel. We compute this in a similar step-wise process to the areas, for non crop productive locations (i.e. a value of zero), and for crop areas [split into non-food calories, i.e. feed and bio fuel and other, and food calories].

```
data.all$kcal.noag<-(0) #zero kcal for all non-ag areas
data.all$kcal.nonfood<- data.all$kcal-data.all$kcal.food
data.all$kcal.prop<-data.all$kcal.food/data.all$kcal
head(subset(data.all, kcal.prop>0))
```

We then join the calories column with the rank column. This orders the agricultural productivity, and other variables by the calories, but keeps the ranking of the land use classes. We generate three different calorie rankings, one for food, and one for non-food calories, and one for all sources of calories; marking them by different orders of magnitude.

```
data.all$rank.all<-(data.all$kcal/max(data.all$kcal, na.rm=T)+data.all$rank)
data.all$rank.nonfood<-(data.all$kcal.nonfood/max(data.all$kcal.nonfood, na.rm=T)+data.all$rank)+10
data.all$rank.food<-(data.all$kcal.food/max(data.all$kcal.food, na.rm=T)+data.all$rank)+100
head(data.all[which(data.all$kcal>0),])
```

2.4.8 Compute areal coverage

Next we compute the non-agricultural area in each grid cell, and the area allocated to non food calories (feed, bio fuels, other) vs. food calorie production. First we check to see if there are any cases with calorie data where our cropland area product is 0. There are a few cases here, but they only represent a very small amount of global kcal. This is an extremely minor part of the data, and likely results from processing error.

```
sub<-subset(data.all, kcal>0 & (CA==0))
 #cells with kcal but no crop f. area data
sum(sub$kcal, na.rm=T)/sum(data.all$kcal, na.rm=T) #
```

Then we define the variables that we will need for later analysis, and sanity check these against those reported in Ramankutty et al. 2008. We also compute the non-crop area, feed/bio fuel/other area, and food area within each pixel in each ranked class. We disaggregate non-food vs food areas using the calorie fraction data, on the assumption that areal allocation is equal per unit of calories for a given pixel.

```
area<-cell.area*nrow(subset(data.all, !is.na(rank))) #terrestrial area
ice.free<- area- (cell.area*nrow(subset(data.all, LC==15))) #ice -free terrestrial area
crop.area<-data.all$CA*cell.area #crop area
pasture.area<-data.all$PA*cell.area #crop area
crop.area[is.na(crop.area)] <- 0  #remove NAs
pasture.area[is.na(pasture.area)] <- 0 #remove NAs
non.ag<-ifelse(!is.na(data.all$rank), cell.area-crop.area-pasture.area, NA) #non.ag
food.area<-crop.area*data.all$kcal.prop #food area
feed.area<-crop.area-food.area #feed/biofuel area
food.area[is.na(food.area)] <- 0 #remove NAs
feed.area[is.na(feed.area)] <- 0  #remove NAs
non.ag[is.na(other.ca.area)] <- 0  #remove NAs
sum(crop.area, na.rm=T)/ice.free #sanity checks, should be ~12\%,
```

```
sum(pasture.area, na.rm=T)/ice.free #sanity checks, should be ~22\% ,
sum(crop.area,pasture.area, na.rm=T)/ice.free #sanity checks, should be ~34\%,
sum(non.ag,crop.area,pasture.area, na.rm=T)/area
```

We then assign the relevant area calculations to the data frame. For all land classes except urban, first we give back non-agricultural area, then non-food (feed, bio fuel, and other) crop area alongside pasture area, then finally agricultural areas producing food calories. For urban locations we reverse the grouping – first we give back non-food (feed, bio fuel, other) agricultural area, then food crop areas and finally, non-agricultural areas. Notably, we simply embed this reversion for urban pixels in the data, to account for logic of high priority for avoiding encroaching on urban settlements, but it is not reflected in the variable names.

```
data.all$food.area<- ifelse(data.all$rank==5,food.area+non.ag, food.area)
data.all$feed.area<-feed.area+pasture.area
data.all$non.ag<-ifelse(data.all$rank==5,0, non.ag)
sum(crop.area, pasture.area, na.rm=T)/ice.free #sanity check
sum(data.all$non.ag,data.all$feed.area,data.all$food.area,  na.rm=T)/area #sanity check
sum(non.ag, crop.area,pasture.area, na.rm=T)/area #sanity check
data.all$past.area<-pasture.area #add in pasture area for later
```

2.4.9 Add in Country ids

During processing we lost the country identifiers, so here we add that back in now, both in ADMIN names, and ISO3's.

```
world<-getMap(resolution='low')
lookup<-as.data.frame(cbind(as.character(world@data[,'ISO3']),world@data[,'ADMIN'],as.character(world@data[,'ADMIN'])))
matched<-lookup[match(data.all$world, lookup$V2), ]
data.all$country<-matched[[1]]
data.all$ISO3<-matched[[3]]
```

Finally we save the processed data as an RDS file.

```
saveRDS(data.all, "halfearthdata/processed/data.all.new.rds")
```

3 Supplementary Methods B

3.1 Analysis overview

In this section we conduct the analysis underlying the results presented in the manuscript. To simulate crop calorie losses under Half-Earth, we gave back pixels (i.e. "nature only landscapes"), or area in pixels ("shared landscapes"), on the planet in sequence of the rank and increasing calorie production, at global (no boundary), country (boundaries set by countries), and ecoregion (boundaries set by Ecoregions) scales. We assess losses at 50% land re-allocation under each scenario.

3.2 Read data

First we read in data.

```
data.all<-readRDS("halfearthdata/processed/data.all.new.rds")
```

Let's double check the number of countries and ecoregions.

```
geocount<-c(length(unique(data.all$Dinn)),length(unique(data.all$ISO3)))
geocount
```

```
[1] 821 231
```

Because we want consistency in the number of cells, and area, given back under each scenario, and because each of the polygon boundaries for the ecoregions, countries and the land cover datasets are not exactly the same, we ensure NAs across the datasets are consistent. In addition we remove the NAs in the land cover data which represent water bodies, so we are working only on the terrestrial land surface area from here on in.

```
data.all<-subset(data.all, !is.na(Dinn) & !is.na(country) & !is.na(rank))
```

Now let's check how many countries and ecoregions we are working with now.

```
subgeocount<-c(length(unique(data.all$Dinn)),length(unique(data.all$ISO3)))
subgeocount
```

```
[1] 775 182
```

3.3 Run scenarios

We then write out a loop to compute the half-earth scenario for each geographical scale of analysis. All the processing is included as one chunk of code here, but is hopefully laid out in a logical way that can be refactored if needed in future. It takes the following inputs:

1. geog = column name with the geographic identifier (i.e. indicating scale of analysis)
2. data= dataset (i.e. data.all)
3. scen = character label for the scenario being run

And then creates a list output of length 4. The first and second elements are the output for the "nature only landscapes" approach to Half-Earth. The third and fourth are the outputs for the "shared landscapes" approach. The first, and third, elements include all original variables in the data.all data.frame, alongside a range of new ones. As outlined below, these list elements and the new variables in them can be used to identify the exact pixels given back under any Half-Earth scenario (it includes a variable called below.mid with 1's to identify these cells), and for summarizing kcal losses by ecoregion and country. The second and fourth elements of this list are dataframes with the following three columns summarizing losses under each scenario:

1. perc.world = cumulative sum of terrestrial land area given back
2. kcal = kcal cumulative of the kcal lost in cells given back
3. perc.kcal = cumulative sum of the percent of kcal given back

Each of the lists for each scale of analysis are saved to their own RDS file. Before we run this analysis we first call a function for computing cumulative sums while ignoring NAs. We also re-call the cell area.

```
cum.na <- function(x) { #cumulative sum, ignore NA function
x[which(is.na(x))] <- 0
return(cumsum(x))}
cell.area <-8439*8439 #define cell area
```

And now we run the scenarios.

```
##############################SETUP#####################################
geog.n<-list(1,as.factor(data.all$Dinn), as.factor(data.all$ISO3))#geographic ids
scen<-c("global", "ecoregion", "country") #scenarios

geog.n<-list(1)#geographic ids
scen<-c("global") #scenarios


for(i in 1:length(geog.n)){

data<-data.all
geog<-geog.n[[i]]
dat.split<-split(data,geog) #split data

##############################NATURE ONLY APPROACH#########################
#ORDER CELLS BY RANK#
dat.split.s<-lapply(dat.split, function(x) x[order(x$rank.all),])

#IDENTIFY GRID CELLS TO GIVE BACK#
dat.split.s<-dat.split.s[lapply(dat.split.s,nrow)>0] #remove locations with no data
dat.split.s<-lapply(dat.split.s, function(x) cbind(x, 1/nrow(x))) #assign numbers
dat.split.s<-lapply(dat.split.s, function(x) cbind(x, round(cumsum(x$`1/nrow(x)`), digits=2)))  #add in the cumulative percentag
names<-c(colnames(dat.split.s[[1]])[1:26], "id", "perc.area") #reset names
dat.split.s<-lapply(dat.split.s, setNames, names)
dat.split.s<-lapply(dat.split.s, function(x) cbind(x, ifelse(x$perc.area<0.51, 1,0)))
dat.split.all.short<-rbindlist(dat.split.s) #use rbindList, quicker

#COMPUTE THE PERCENT LOSSES BY AREA ACROSS GEOG UNITS#
dat.all.sum.short<-ddply(dat.split.all.short,"perc.area", summarize,
                      kcal.sum=sum(kcal, na.rm=T),
                      PA.sum=sum(PA*cell.area, na.rm=T),
                      CA.sum=sum(CA*cell.area, na.rm=T),
                      kcal.food.sum=sum(kcal.food, na.rm=T),
                      kcal.feed.sum=sum(kcal.nonfood, na.rm=T)
                      ) #summarize the kcal by percent area, take time
dat.all.sum.short$perc.kcal<-cumsum(dat.all.sum.short$kcal.sum)/max(cumsum(dat.all.sum.short$kcal.sum))
dat.all.sum.short$perc.CA<-cumsum(dat.all.sum.short$CA.sum)/max(cumsum(dat.all.sum.short$CA.sum))
dat.all.sum.short$perc.PA<-cumsum(dat.all.sum.short$PA.sum)/max(cumsum(dat.all.sum.short$PA.sum))
dat.all.sum.short$perc.kcal.non<-cumsum(dat.all.sum.short$kcal.feed.sum)/max(cumsum(dat.all.sum.short$kcal.feed.sum))
dat.all.sum.short$perc.kcal.food<-cumsum(dat.all.sum.short$kcal.food.sum)/max(cumsum(dat.all.sum.short$kcal.food.sum))
```

```
##############################SHARED APPROACH#################################
#MAKE DATA LONG#
vars<-setdiff(colnames((dat.split[[1]])), c("feed.area", "food.area", "non.ag"))
dat.split.l <-lapply(dat.split, function(x) melt(x, id.vars = c(vars), value.name="area", variable.name="Area.type"))
head(dat.split.l[[1]])
#DEFINE NEW RANK AND OUTCOME VARIABLES#
dat.split.l<-lapply(dat.split.l , function(x) cbind(x, ifelse(x$Area.type=="non.ag",
x$rank.all,ifelse(x$Area.type=="feed.area", x$rank.nonfood, x$rank.food)), #new rank column
ifelse(x$Area.type=="non.ag", 0,
ifelse(x$Area.type=="feed.area", x$kcal.nonfood, x$kcal.food)), #new kcal
ifelse(x$Area.type=="feed.area", x$kcal.nonfood,0), #new kcal.nonfood
ifelse(x$Area.type=="food.area", x$kcal.food,0), #new kcal. food
ifelse(x$Area.type=="non.ag", 0, x$CA*cell.area), #new CA column
ifelse(x$Area.type=="non.ag", 0,x$past.area))) #new PA column

names<-c(colnames(dat.split.l[[1]])[1:25], "rank.new",
         "kcal.new", "kcal.new.feed", "kcal.new.food","CA.n", "PA.n") #reset names
dat.split.l<-lapply(dat.split.l, setNames, names)

#ORDER CELLS BY NEW RANK#
dat.split.l<-lapply(dat.split.l, function(x) x[order(x$rank.new),])

#IDENTIFY GRID CELLS TO GIVE BACK#
dat.split.l<-lapply(dat.split.l , function(x) cbind(x,cum.na(x$area), #cumsum area
       rep(nrow(x)*cell.area*(1/3), nrow(x)))) #total area
dat.split.l<-lapply(dat.split.l , function(x)
        cbind(x, round(x$`cum.na(x$area)`/(x$`rep(nrow(x) * cell.area * (1/3), nrow(x))`), digits=2),
               ifelse(x$`cum.na(x$area)`<= (x$`rep(nrow(x) * cell.area * (1/3), nrow(x))`)/2, 1, 0))) #area percent
        names<-c(colnames(dat.split.l[[1]])[1:31], "cum.sum.area",
         "tot.area",  "perc.area", "below.med") #set names

dat.split.l<-lapply(dat.split.l, setNames, names)

dat.split.all<-rbindlist(dat.split.l) #use rbindList, quicker

#COMPUTE THE PERCENT LOSSES BY AREA ACROSS GEOG UNITS#
dat.all.sum<-ddply(dat.split.all,"perc.area", summarize,
                    kcal.sum=sum(kcal.new, na.rm=T),
                    PA.sum=sum(PA.n, na.rm=T),
                    CA.sum=sum(CA.n, na.rm=T),
                    kcal.food.sum=sum(kcal.new.food, na.rm=T),
                    kcal.feed.sum=sum(kcal.new.feed, na.rm=T)
                    ) #summarize the kcal by percent area, take time
dat.all.sum$perc.kcal<-cumsum(dat.all.sum$kcal.sum)/max(cumsum(dat.all.sum$kcal.sum))
dat.all.sum$perc.CA<-cumsum(dat.all.sum$CA.sum)/max(cumsum(dat.all.sum$CA.sum))
dat.all.sum$perc.PA<-cumsum(dat.all.sum$PA.sum)/max(cumsum(dat.all.sum$PA.sum))
dat.all.sum$perc.kcal.non<-cumsum(dat.all.sum$kcal.feed.sum)/max(cumsum(dat.all.sum$kcal.feed.sum))
dat.all.sum$perc.kcal.food<-cumsum(dat.all.sum$kcal.food.sum)/max(cumsum(dat.all.sum$kcal.food.sum))

##############################SAVE RESULT#####################################
return<- list(dat.split.all.short, dat.all.sum.short, dat.split.all, dat.all.sum)
path<-paste("halfearthdata/processed/",scen[i],"processed.rds", sep="")
saveRDS(return, path)
}
```

3.4 Main Text Figures

First we read in the data processed in the previous subsection.

```
global<-readRDS("halfearthdata/processed/globalprocessed.rds")
ecoregion<-readRDS("halfearthdata/processed/ecoregionprocessed.rds")
country<-readRDS("halfearthdata/processed/countryprocessed.rds")
```

3.4.1 Figure 1

We then join all of the data together for plotting. Here we are interested in the results for both nature only landscapes and shared landscapes approaches, and comparing the summary results for each of these. To do this we, we need to call the second, and fourth elements of each of the lists read in above.

```
to.plot<-rbind(global[[2]], country[[2]], ecoregion[[2]],
               global[[4]], country[[4]], ecoregion[[4]])

to.plot$Scenario<-c(rep("Global.NatOnly", nrow(global[[2]])),
                    rep("Country.NatOnly", nrow(country[[2]])),
                    rep("Ecoregion.NatOnly", nrow( ecoregion[[2]])),
                    rep("Global.Shared", nrow(global[[4]])),
                    rep("Country.Shared", nrow(country[[4]])),
                    rep("Ecoregion.Shared", nrow( ecoregion[[4]])))

to.plot$Scenario1<-c(rep("Global", nrow(global[[2]])),
                     rep("Country", nrow(country[[2]])),
                     rep("Ecoregion", nrow( ecoregion[[2]])),
                     rep("Global", nrow(global[[4]])), rep("Country", nrow(country[[4]])),
                     rep("Ecoregion", nrow( ecoregion[[4]])))

to.plot$Scenario2<-c(rep("Nature Only", nrow(global[[2]])),
                     rep("Nature Only", nrow(country[[2]])),
                     rep("Nature Only", nrow( ecoregion[[2]])),
                     rep("Shared", nrow(global[[4]])),
                     rep("Shared", nrow(country[[4]])),
                     rep("Shared", nrow( ecoregion[[4]])))
```

Before we plot we print the aggregated losses for each scenario below. These are the main results reported in the manuscript.

```
to.plot.sub<-subset(to.plot,perc.area<0.501)
ddply(to.plot.sub, c("Scenario1" ,"Scenario2"), summarize,
      kcal.loss.feed=max(perc.kcal.non)*100,
      kcal.loss.food=max(perc.kcal.food)*100,
      CA.loss=max(perc.CA )*100,
      PA.loss=max(perc.PA )*100
      )
```

	Scenario1	Scenario2	kcal.loss.feed	kcal.loss.food	CA.loss	PA.loss
1	Country	Nature Only	15.7	22.09	21	37.4
2	Country	Shared	11.0	0.39	10	4.1
3	Ecoregion	Nature Only	24.5	28.80	31	45.4
4	Ecoregion	Shared	22.7	2.88	15	10.3
5	Global	Nature Only	9.7	11.24	12	21.1
6	Global	Shared	0.0	0.00	0	0.0

And then we make this summary data long for plotting, and make the variable names intelligible.

```
vars<-setdiff(colnames(to.plot), c("perc.kcal.food", "perc.kcal.non", "perc.PA","perc.CA"))
to.plot.l <- melt(to.plot, id.vars = c(vars), value.name="loss", variable.name="Loss.type") #make data long

to.plot.l$Loss.type<-recode(to.plot.l$Loss.type,  #make var names intelligable
  " c('perc.kcal.food')= 'D. Food kcal';
   c('perc.kcal.non') = 'C. Feed, biofuel, and other kcal';
   c('perc.PA')= 'B. Pasture land';
   c('perc.CA')= 'A. Crop land'" )
```

We then plot out the different scenarios. These plots form the basis for figure 1 given in the main text of our manuscript.

```r
ggplot(to.plot.l, aes(perc.area*100,loss*100, color=Scenario1))+
  geom_vline(xintercept=50, lty=2, colour="black")+
  geom_line()+
  scale_colour_manual(values = c("#56B4E9","#E75E00", "#009E73", "gray"))+
  theme_bw()+
  theme(plot.background = element_blank(),
         panel.grid.major = element_blank(),
         panel.grid.minor = element_blank(),
           # strip.text.x = element_blank(),
        strip.background = element_rect(colour = "white", fill = "white"),
        legend.key = element_rect(fill = "white", colour = "white"))+
  ylab("Percent agricultural loss")+
  xlab("Percent of earth given back")+
    # ylab("")+
    # xlab("")+
  facet_wrap(~Loss.type*Scenario2, nrow=4, ncol=2) + theme(legend.position="none")
```

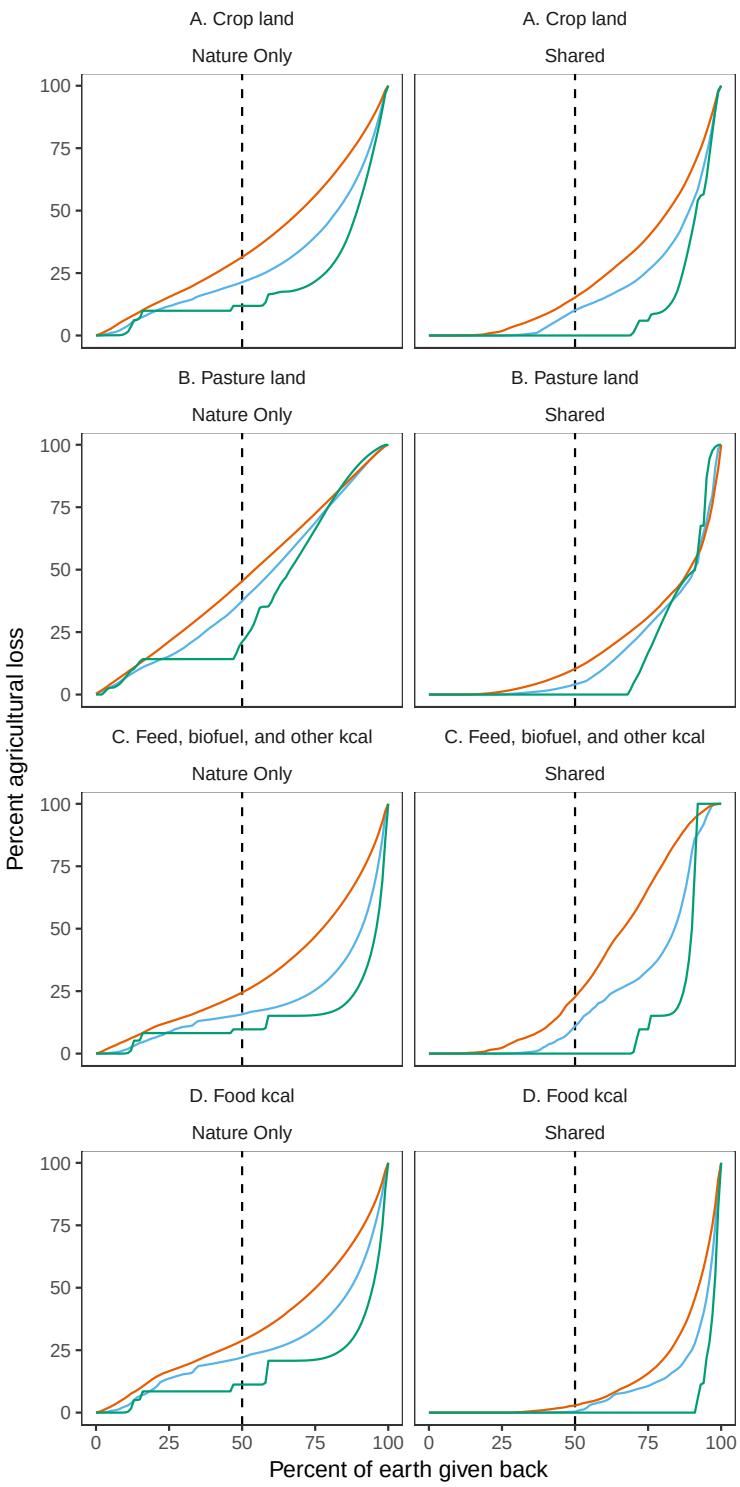


Figure 1: Feeding the world under a global deal for nature. The horizontal dashed line shows the Half-Earth line. Left side: Nature only landscapes, Right side: Shared landscapes. green= global, blue= country, red= ecoregion.

3.4.2 Figure 2

Here we map out the scenarios. First we need to identify which pixels were given back under each scenario, and the calories lost in each. We do this to visualize where the calorie trade-offs are locally. We isolate the data needed for plotting.

```
global.NatOnly<-subset(global[[1]], `ifelse(x$perc.area < 0.51, 1, 0)`==1)
country.NatOnly<- subset(country[[1]], `ifelse(x$perc.area < 0.51, 1, 0)`==1)
ecoregion.NatOnly<-subset(ecoregion[[1]], `ifelse(x$perc.area < 0.51, 1, 0)`==1)
global.Sharing<-subset(global[[3]], below.med==1)
country.Sharing<- subset(country[[3]], below.med==1)
ecoregion.Sharing<-subset(ecoregion[[3]], below.med==1)

kcal.max<-round(max(log10((global.NatOnly$kcal/1000)+1), na.rm=T), digits=1)
kcal.min<-round(min(log10((global.NatOnly$kcal/1000)+1), na.rm=T), digits=1)
```

Then we write a function to map the kcal lost for each scenario in each location given back. We choose a blue colour gradient to aid discernment of logged steps in the scale.

```
breaks=c(0:9)
map.scenarios<-function(data, kcal){
df<-data.frame(data$x, data$y,kcal)
dfr<- rasterFromXYZ(df)  #Convert first two columns as lon-lat and third as value
crs(dfr)<-"+proj=eck4 +datum=WGS84 +ellps=WGS84 +towgs84=0,0,0"
values(dfr)<-log10((values(dfr)/1000)+1)
colfunc <- colorRampPalette(c("gray95", "#edf8b1", "#c7e9b4","#7fcdbb", "#41b6c4","#1d91c0",
                              "#225ea8", "#253494","#081d58", "black")) #set the color
data(wrld_simpl) #get world map data
wrld <- spTransform(wrld_simpl, crs(dfr))
plot(wrld, col='light gray', border="transparent",lwd=0.2)
plot(dfr,  col=colfunc(10),add=TRUE, legend.width = 1, legend.shrink=0.5,
     legend.args=list(text=' loss (log10 kcal)' ,side=4,cex=0.8,line=2), breaks=breaks)
}
```

And then we plot all of these together. The result forms the basis of figure 2 in the main text of the manuscript.

```
par(mfrow = c(3,2),
      oma = c(0,0,0,0) ,
          mar = c(0,0,1,0))
map.scenarios(global.NatOnly,  global.NatOnly$kcal)
map.scenarios(global.Sharing,global.Sharing$kcal.new)
map.scenarios(country.NatOnly,country.NatOnly$kcal)
map.scenarios(country.Sharing,country.Sharing$kcal.new)
map.scenarios(ecoregion.NatOnly,ecoregion.NatOnly$kcal)
map.scenarios(ecoregion.Sharing,ecoregion.Sharing$kcal.new)
```

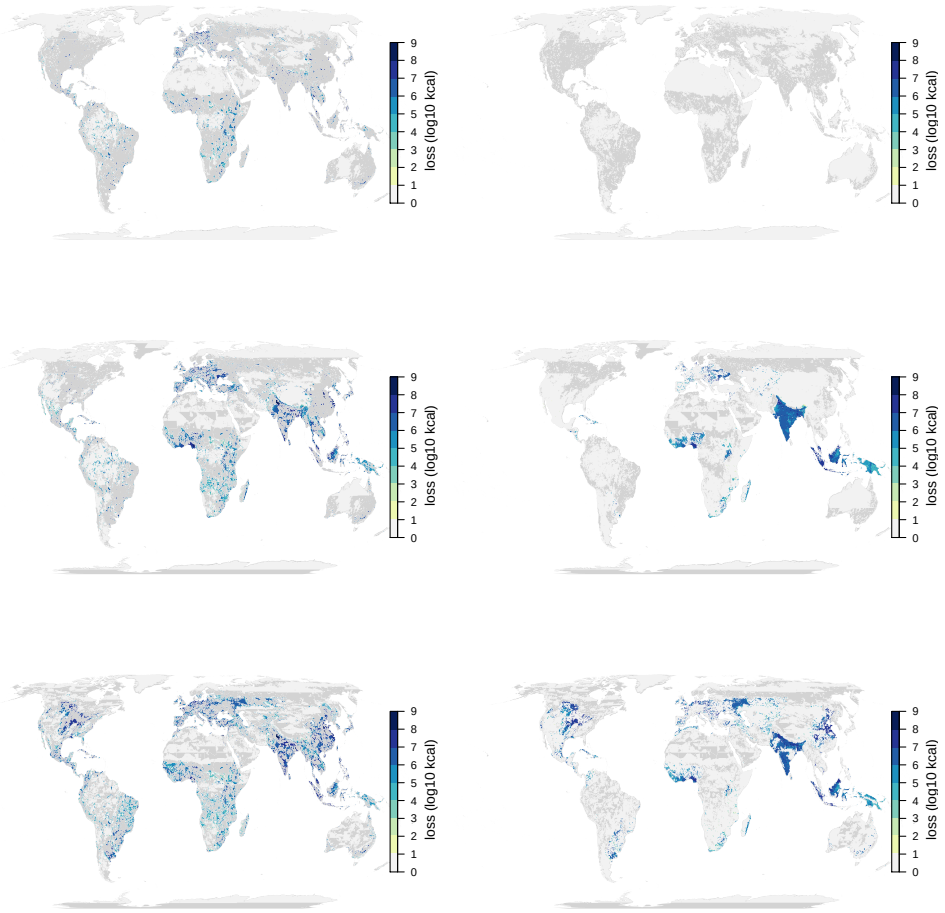


Figure 2: Maps of calorie losses under each Half-Earth scenario. From top to bottom: global, country and ecoregion. Legends show kcal losses in log10 units. Left side: Nature only landscapes, Right side: Shared landscapes.

3.5 Country calorie losses

Here we report the countries with calorie losses under the two different conservation approaches at the ecoregion scale. We report countries with the top losses under the natures only landscapes approach below. Notably China and India show substantial calorie losses.

```
country.loss.hard<-ddply(ecoregion.NatOnly, c("country"), summarize,  #global
                         kcal.loss=sum(kcal, na.rm=T))
head(country.loss.hard[order(-country.loss.hard$kcal.loss),])
```

```
    country kcal.loss
31      CHN   4.9e+14
76      IND   4.5e+14
172     USA   3.1e+14
75      IDN   1.6e+14
115     MYS   7.5e+13
119     NGA   6.6e+13
```

We then compute the losses under a shared landscapes approach. While these reduce the losses for China and India by at least a half, the losses still remain high.

```
country.loss.soft<-ddply(ecoregion.Sharing, c("country"), summarize,  #global
                         kcal.loss=sum(kcal.new, na.rm=T))
head(country.loss.soft[order(-country.loss.soft$kcal.loss),])
```

```
    country kcal.loss
76      IND   2.1e+14
31      CHN   2.0e+14
172     USA   1.7e+14
75      IDN   9.3e+13
115     MYS   6.4e+13
119     NGA   5.1e+13
```

Next we compute the total calories produced in different countries. Below we show the top calorie producing nations.

```
country.kcal<-ddply(data.all, c("country"), summarize,  #global
                         kcal=sum(kcal, na.rm=T))
head(country.kcal[order(-country.kcal$kcal),])
```

```
    country    kcal
32      CHN 1.7e+15
174     USA 1.4e+15
78      IND 1.0e+15
77      IDN 5.0e+14
24      BRA 4.8e+14
117     MYS 3.6e+14
```

And finally we join all of these data, to estimate the percent losses. We are particularly interested in the losses under the shared approach, which we report in the main text of the paper. We show the top losses in percent terms below.

```
kcal.count<-country.kcal[match(country.loss.soft$country,country.kcal$country), ]
country.loss.soft$perc.loss<-country.loss.soft$kcal.loss/ kcal.count$kcal
(country.loss.soft[order(-country.loss.soft$perc.loss),])[1:10,]
```

```
    country kcal.loss perc.loss
24      BRN   1.4e+10      0.30
61      GHA   8.8e+12      0.29
119     NGA   5.1e+13      0.27
130     PNG   1.4e+12      0.25
14      BEL   8.5e+11      0.24
96      LBR   4.3e+11      0.23
76      IND   2.1e+14      0.21
107     MKD   4.0e+11      0.20
154     SVN   3.0e+11      0.20
75      IDN   9.3e+13      0.19
```

3.5.1 Ecoregion coverage

In an earlier publication Dinerstein et al. [9] reported that Half-Earth has been already attained, or been deemed achievable, in 49% of the Earth's 846 terrestrial Ecoregions. Dinerstein et al. identified 98/846 (or 12%) had achieved more than half protected, and that 313/846 Ecoregions (or 37%) to have not achieved half protected, but to have enough "unaltered habitat" available to achieve it. They defined this unaltered habitat (also synonymously labelled as both "natural" in the main text, and "natural/semi-natural" in the Supplementary Methods) by a two-step procedure, first using satellite classified tree cover data to identify forests, and then defining remaining habitats according to anthropogenic biomes. However, in their analysis they did not quantitatively identify the trade-offs with agricultural production. Here we identify how many of the world's Ecoregions could achieve the 50% figure without a loss of calories, under both the nature only landscapes, and shared landscapes approaches. We find that for a nature only landscapes approach, Half-Earth can be achieved in 101 or the 775, or 14%, of Ecoregions used in this analysis without any loss of calories.

```
lossbyeco<-ddply(ecoregion.NatOnly, 'Dinn', summarize,
      loss=sum(kcal, ra.rm=T))
nrow(subset(lossbyeco, loss<10)) #number of ecoregions with no calorie loss
```

```
[1] 111
```

```
(nrow(lossbyeco[which(lossbyeco$loss<10),]))/nrow(lossbyeco) #percent of ecoregions
```

```
[1] 0.14
```

Interestingly, under a shared landscapes approach, we find that Half-Earth can be achieved in nearly five times more ecoregions without calorie losses (or 65%).

```
lossbyeco.s<-ddply(ecoregion.Sharing, 'Dinn', summarize,
      loss=sum(kcal.new, ra.rm=T))
nrow(subset(lossbyeco.s, loss<10)) #number of ecoregions with no calorie loss
```

```
[1] 501
```

```
(nrow(lossbyeco.s[which(lossbyeco.s$loss<10),]))/nrow(lossbyeco) #percent of ecoregions.
```

```
[1] 0.65
```

3.6 Co-benefits

To understand the co-benefits of a Half-Earth scenario, (e.g. such as climate mitigation potential) it is important to understand the gross change in vegetation that might result from such a plan. Here we show the how agriculturally productive land given back under each scale of analysis (country, ecoregion, and global), influences change in the coverage of potential vegetation. To do this we use the Potential Natural Vegetation (PNV) data set [15].

First we need to calculate PNV coverage in area without agricultural production – this represents a baseline. We will then calculate the PNV area inside agriculturally productive pixels given back under each Half-Earth scenario. The ratio of these two numbers is used to indicate the potential vegetation benefits of giving back agricultural land. It should be noted that the baseline estimates are consistent across scenarios, but are likely to overestimate PNV in current in non-agricultural lands because these lands are unlikely to be completely unaltered by human activity (and hence our estimates of PNV gains are likely to be conservative).

3.6.1 Nature only landscapes

First we get the number of cells for the baseline PNV, in non-agriculturally productive and non-urban land globally.

```
cell.area <-8439*8439 #define cell area
data.all$PNV.base<-ifelse(data.all$rank >=7, NA,data.all$PNV)

baseline<-ddply(data.all, 'PNV.base',
```

```
      summarize,
      n.cells= length(x),
      ag.area= sum(CA*cell.area, PA*cell.area, na.rm=T),
      area=length(x)*cell.area)

baseline$PNV.base<-baseline$area-baseline$ag.area #this is the baseline vegetation on the planet
```

Then we obtain the PNV in recovered agricultural land in each scenario.

```
world.no<-ddply(global.NatOnly, 'PNV',
      summarize,
      area= sum(CA*cell.area,PA*cell.area, na.rm = T),
      n.cells= length(x))

country.no<-ddply(country.NatOnly, 'PNV',
      summarize,
      area= sum(CA*cell.area,PA*cell.area, na.rm = T),
      n.cells= length(x))

ecoregion.no<- ddply(ecoregion.NatOnly, 'PNV',
      summarize,
      area= sum(CA*cell.area,PA*cell.area, na.rm = T),
      n.cells= length(x))
```

Finally we join all data together into a common data.frame. The Gain.prop column represents the proportional gain in cells for each PNV class for agriculturally productive land.

```
gain<-c(world.no$area/baseline$PNV.base,
        country.no$area/baseline$PNV.base,
        ecoregion.no$area/baseline$PNV.base)
scenario<-c(rep("Global", 16), rep("Country", 16), rep("Ecoregion", 16))
class<-rep(world.no$PNV, 3)
gain.df.no<-data.frame(gain, scenario, class)
colnames(gain.df.no)<-c("Gain.prop", "Scenario", "PNV.class")
gain.df.no<-(subset(gain.df.no, !is.na(PNV.class))) # remove NAs
```

We then write the code to plot the results.

```
NO<-ggplot(gain.df.no, aes( PNV.class,Gain.prop*100, colour=Scenario))+
# geom_point()+
  geom_hline(yintercept=c(25,50,75,100), lty=2, colour="black", lwd=0.1)+
  geom_line()+
   scale_colour_manual(values = c("#56B4E9","#E75E00", "#009E73"))+
  theme_bw()+
   theme(plot.background = element_blank(),
         panel.grid.major = element_blank(),
         panel.grid.minor = element_blank(),
        strip.background = element_rect(colour = "white", fill = "white"))+
    theme(legend.position="none")     +
  scale_x_continuous(breaks= 1:15, labels= c("1" ="Tropical Forest (E)", "2" ="Tropical Forest (D)",
                                             "3" ="Temp.Forest (BE)","4" ="Temp.Forest (NE)",
                                             "5" ="Temp.Forest (D)","6" ="Boreal.Forest (E)",
                                             "7" ="Boreal.Forest (D)","8" ="Mixed Forest (E/D)",
                                             "9" ="Savanna","10" ="Grassland/Steppe","11" = "Dense Shrubland",
                                             "12" = "Open Shrubland", "13" ="Tundra", "14" ="Desert",
                                             "15" ="Polar"))+
   ylab("Change in area (%)")+
  xlab("")+
  coord_flip()
```

3.6.2 Shared lanscapes

Now we repeat the analysis except for the sharing landscapes approach.

```
world.s<-ddply(global.Sharing, 'PNV',
      summarize,
      n.cells= length(x),
      area= sum(CA.n,PA.n, na.rm = T)
```

```
      )
country.s<-ddply(country.Sharing, 'PNV',
      summarize,
      n.cells= length(x),
      area= sum(CA.n,PA.n, na.rm = T))

ecoregion.s<- ddply(ecoregion.Sharing, 'PNV',
      summarize,
      n.cells= length(x),
      area= sum(CA.n,PA.n, na.rm = T))
```

And join all data together into a common data.frame.

```
gain.s<-c(world.s$area/baseline$PNV.base,
        country.s$area/baseline$PNV.base,
        ecoregion.s$area/baseline$PNV.base)
scenario<-c(rep("Global", 16), rep("Country", 16), rep("Ecoregion", 16))
class<-rep(world.s$PNV, 3)
gain.df<-data.frame(gain.s, scenario, class)
colnames(gain.df)<-c("Gain.prop", "Scenario", "PNV.class")
gain.df.s<-(subset(gain.df, !is.na(PNV.class))) # remove NAs
```

Then we create the plot.

```
s<-ggplot(gain.df.s, aes( PNV.class,Gain.prop*100, colour=Scenario))+
  geom_hline(yintercept=c(25,50,75,100), lty=2, colour="black", lwd=0.1)+
  geom_line()+
  scale_colour_manual(values = c("#56B4E9","#E75E00", "#009E73"))+
  theme_bw()+
   theme(plot.background = element_blank(),
         panel.grid.major = element_blank(),
         panel.grid.minor = element_blank(),
        strip.background = element_rect(colour = "white", fill = "white"))+
    theme(legend.position="none")+
  scale_x_continuous(breaks= 1:15, labels= c("1" ="", "2" ="",
                                             "3" ="","4" ="","5" ="","6" ="",
                                             "7" ="","8" ="","9" ="","10" ="",
                                             "11" = "","12" = "", "13" ="", "14" ="",
                                             "15" =""))+
   ylab("Change in area (%)")+
  xlab("")+
  coord_flip()
```

We also print a table of the gains below which show the percent gain figures under each scenario.

```
all.gain<-data.frame(gain.df.no$Scenario, gain.df.no$PNV.class, gain.df.no$Gain.prop, gain.df.s$Gain.prop)
colnames(all.gain)<-c("Scenario", "PNV.class", "Nat.Only", "Shared")
all.gain$PNV.class<-recode(all.gain$PNV.class,
 " c('1')= 'Tropical Forest (E)';
   c('2') = 'Tropical Forest (D)';
   c('3')= 'Temp.Forest (BE)';
   c('4')= 'Temp.Forest (NE)';
   c('5')= 'Temp.Forest (D)';
   c('6')= 'Boreal.Forest (E)';
   c('7')= 'Boreal.Forest (D)';
   c('8')= 'Mixed Forest (E/D)';
   c('9')= 'Savanna';
   c('10')= 'Grassland/Steppe';
   c('11')= 'Dense Shrubland';
   c('12')= 'Open Shrubland';
   c('13')= 'Tundra';
   c('14')= 'Desert';
   c('15')= 'Polar'
  ")
```

```
all.gain
```

	Scenario	PNV.class	Nat.Only	Shared
1	Global	Tropical Forest (E)	0.0398	0.00000
2	Global	Tropical Forest (D)	0.1214	0.00000
3	Global	Temp.Forest (BE)	0.0717	0.00000
4	Global	Temp.Forest (NE)	0.0444	0.00000
5	Global	Temp.Forest (D)	0.1272	0.00000
6	Global	Boreal.Forest (E)	0.0068	0.00000
7	Global	Boreal.Forest (D)	0.0069	0.00000
8	Global	Mixed Forest (E/D)	0.0173	0.00000
9	Global	Savanna	0.1360	0.00000
10	Global	Grassland/Steppe	0.4150	0.00000
11	Global	Dense Shrubland	0.1262	0.00000
12	Global	Open Shrubland	0.2300	0.00000
13	Global	Tundra	0.0601	0.00000
14	Global	Desert	0.0295	0.00000
15	Global	Polar	0.0111	0.00000
16	Country	Tropical Forest (E)	0.0779	0.07106
17	Country	Tropical Forest (D)	0.2601	0.39317
18	Country	Temp.Forest (BE)	0.1016	0.01117
19	Country	Temp.Forest (NE)	0.0994	0.02191
20	Country	Temp.Forest (D)	0.2256	0.05298
21	Country	Boreal.Forest (E)	0.0098	0.00038
22	Country	Boreal.Forest (D)	0.0104	0.00217
23	Country	Mixed Forest (E/D)	0.0321	0.00625
24	Country	Savanna	0.2298	0.08504
25	Country	Grassland/Steppe	0.7275	0.22649
26	Country	Dense Shrubland	0.1989	0.06394
27	Country	Open Shrubland	0.4318	0.08320
28	Country	Tundra	0.1172	0.00374
29	Country	Desert	0.0388	0.00727
30	Country	Polar	0.0184	0.00319
31	Ecoregion	Tropical Forest (E)	0.0909	0.07585
32	Ecoregion	Tropical Forest (D)	0.4022	0.37771
33	Ecoregion	Temp.Forest (BE)	0.2689	0.11883
34	Ecoregion	Temp.Forest (NE)	0.1533	0.06645
35	Ecoregion	Temp.Forest (D)	0.3630	0.18358
36	Ecoregion	Boreal.Forest (E)	0.0103	0.00298
37	Ecoregion	Boreal.Forest (D)	0.0110	0.00420
38	Ecoregion	Mixed Forest (E/D)	0.0525	0.02820
39	Ecoregion	Savanna	0.3498	0.14244
40	Ecoregion	Grassland/Steppe	0.9416	0.65670
41	Ecoregion	Dense Shrubland	0.3280	0.19671
42	Ecoregion	Open Shrubland	0.4264	0.22300
43	Ecoregion	Tundra	0.0730	0.03195
44	Ecoregion	Desert	0.0359	0.00811
45	Ecoregion	Polar	0.0131	0.00297

4 Supplementary Figures

In this section we plot the supplementary figures that we do not present in the main text of the manuscript.

4.0.1 Supplementary Figure 1

First we plot the maps of binarized calorie losses by country under Half-Earth. Each plot shows Ecoregions in which Half-Earth can be achieved without loss of calories. Note that Antarctica and Greenland are listed as locations under a "Nature only landscapes" approach where loss of calories occurs. This is due to the fact these locations are listed under a common "Rock and ice" Ecoregion, along with other locations on the Earth,such as a few pixels in Northern India and Nepal, which are subject to calorie loss. Removing this "Rock and ice" Ecoregion from this analysis makes no difference to the percentage of losses for this approach (e.g. 101/775 vs 100/775).

```
map.ecoregion.loss<-function(data, lossby){
data.all$Dinn.no<-ifelse(data.all$Dinn  %in% c(subset(lossby, loss<10)$Dinn), 1,0 )
dfloss<-data.frame(data.all$x, data.all$y,data.all$Dinn.no)
dfrloss<- rasterFromXYZ(dfloss)  #Convert first two columns as lon-lat and third as value
crs(dfrloss)<-"+proj=eck4 +datum=WGS84 +ellps=WGS84 +towgs84=0,0,0"
 colfunc <- colorRampPalette(c("red", "light gray")) #set the color ramp
data(wrld_simpl) #get world map data
wrld <- spTransform(wrld_simpl, crs(dfrloss))
plot(wrld, col='light gray', border="transparent",lwd=0.2)
plot(dfrloss, col=colfunc(10), add=TRUE, legend=F)
}

par(mfrow = c(2,1),
      oma = c(0,0,0,0) ,
          mar = c(0,0,1,0))
map.ecoregion.loss(data.all, lossbyeco)
map.ecoregion.loss(data.all, lossbyeco.s)
```

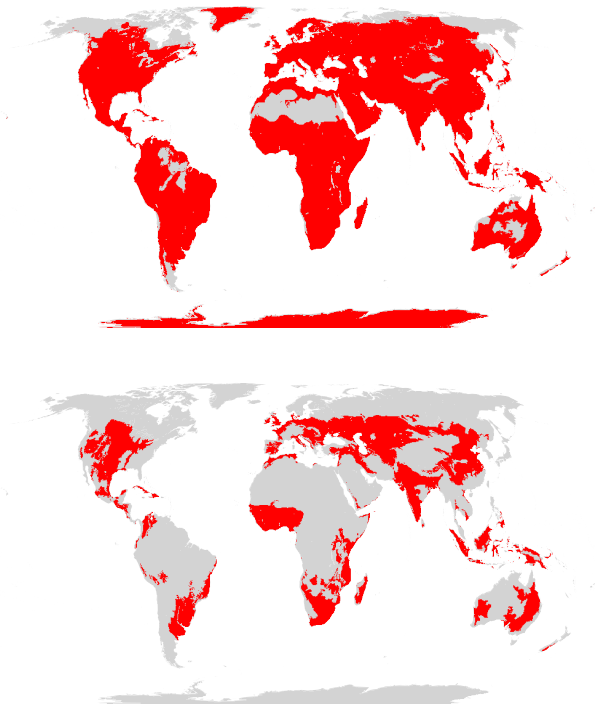


Figure 3: Ecoregions where Haf-Earth be achieved without a loss of calories. Gray= Ecoregions with no loss of calories, Red= Ecoregions with loss of calories. Top = Nature only landscsapes. Bottom = Shared landscapes.

4.0.2 Supplementary Figure 2

Then we plot out the co-benefits of Half-Earth for recover of natural vegetation. As we see from below, the ecoregion based Half-Earth plan leads to large gains in grasslands and modest gains in tropical and temperate deciduous forests, and in savanna and shrublands.

```
plot_grid( NO,s, align =c("v", "h"), ncol = 2)
```

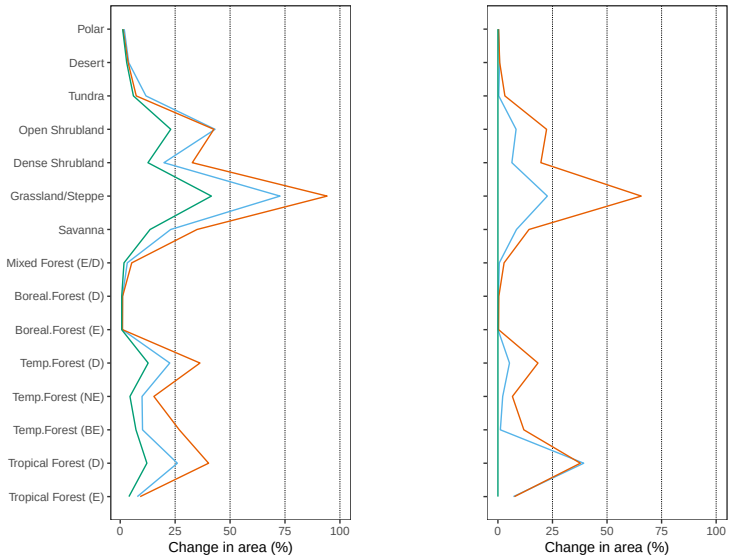
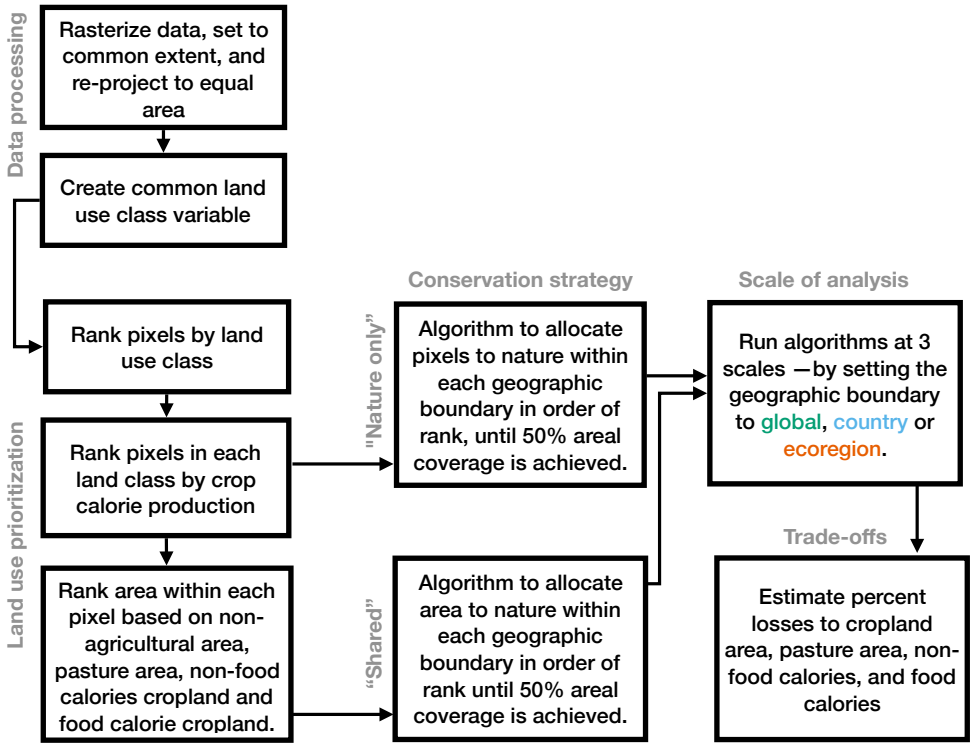


Figure 4: Recovery of natural vegetation from agriculture under each Half-Earth Scenario. The x axis shows the percent gains in areal coverage in potential natural vegetation attained by giving back least agriculturally productive lands. green= global, blue= country, red= ecoregion. E=evergreen, D=deciduous, N=needle, B=broadleaf. Left = Nature only landscsapes. Right = Shared landscapes.

4.0.3 Supplementary Figure 3

And finally we plot out a flow diagram for the analysis, which we reference in the methods section of our paper.



Session information

```
sessionInfo()
```

```
R version 3.4.2 (2017-09-28)
Platform: x86_64-apple-darwin15.6.0 (64-bit)
Running under: macOS High Sierra 10.13.5

Matrix products: default
BLAS: /System/Library/Frameworks/Accelerate.framework/Versions/A/Frameworks/vecLib.framework/Versions/A/libBLAS.dylib
LAPACK: /System/Library/Frameworks/Accelerate.framework/Versions/A/Frameworks/vecLib.framework/Versions/A/libLAPACK.dylib

locale:
[1] en_CA.UTF-8/en_CA.UTF-8/en_CA.UTF-8/C/en_CA.UTF-8/en_CA.UTF-8

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
 [1] data.table_1.10.4-2 cowplot_0.8.0       gridExtra_2.3
 [4] reshape2_1.4.3      maptools_0.9-2      car_2.1-6
 [7] rworldmap_1.3-6     R.matlab_3.6.1      plyr_1.8.4
[10] ggplot2_2.2.1       rgdal_1.1-10        raster_2.6-7
[13] sp_1.2-7            checkpoint_0.4.1    knitr_1.17
[16] RevoUtils_10.0.6

loaded via a namespace (and not attached):
 [1] Rcpp_0.12.13       nloptr_1.0.4       compiler_3.4.2
 [4] highr_0.6          R.methodsS3_1.7.1  R.utils_2.5.0
 [7] tools_3.4.2        lme4_1.1-14        dotCall64_0.9-04
[10] nlme_3.1-131       evaluate_0.10.1    tibble_1.3.4
[13] gtable_0.2.0       lattice_0.20-35    mgcv_1.8-22
[16] rlang_0.2.1        Matrix_1.2-11      parallel_3.4.2
[19] SparseM_1.77       spam_2.1-1         stringr_1.2.0
[22] MatrixModels_0.4-1 fields_9.0         maps_3.2.0
[25] grid_3.4.2         nnet_7.3-12        foreign_0.8-69
[28] minqa_1.2.4        magrittr_1.5       scales_0.5.0
[31] MASS_7.3-47        splines_3.4.2      pbkrtest_0.4-7
[34] colorspace_1.3-2   labeling_0.3       quantreg_5.33
[37] stringi_1.1.5      lazyeval_0.2.0     munsell_0.4.3
[40] R.oo_1.21.0
```

Supplementary References

[1] Baptiste Auguie. *gridExtra: Miscellaneous Functions for "Grid" Graphics*. R package version 2.3. 2017. URL: https://CRAN.R-project.org/package=gridExtra.

[2] Andrew Balmford and Rhys Green. "How to spare half a planet". In: *Nature* 552 (2017), p. 175. ISSN: 14764687. DOI: 10.1038/d41586-017-08578-7.

[3] Henrik Bengtsson. *R.matlab: Read and Write MAT Files and Call MATLAB from Within R*. R package version 3.6.1. 2016. URL: https://CRAN.R-project.org/package=R.matlab.

[4] BirdLife. *World Database of Key Biodiversity Areas*. 2017. URL: www.keybiodiversityareas.org (visited on 11/01/2017).

[5] Roger Bivand, Tim Keitt, and Barry Rowlingson. *rgdal: Bindings for the Geospatial Data Abstraction Library*. R package version 1.1-10. 2016. URL: https://CRAN.R-project.org/package=rgdal.

[6] Roger Bivand and Nicholas Lewin-Koh. *maptools: Tools for Reading and Handling Spatial Objects*. R package version 0.9-2. 2017. URL: https://CRAN.R-project.org/package=maptools.

[7] Emily S Cassidy et al. "Redefining agricultural yields: from tonnes to people nourished per hectare". In: *Environmental Research Letters* 8 (2013), p. 034015. ISSN: 1748-9326. DOI: 10.1088/1748-9326/8/3/034015. URL: http://iopscience.iop.org/1748-9326/8/3/034015/article/.

[8] Microsoft Corporation. *checkpoint: Install Packages from Snapshots on the Checkpoint Server for Reproducibility*. R package version 0.4.1. 2017. URL: https://CRAN.R-project.org/package=checkpoint.

[9] Eric Dinerstein et al. *An Ecoregion-Based Approach to Protecting Half the Terrestrial Realm*. 2017. DOI: 10.1093/biosci/bix014. arXiv: 1611.06654.

[10] Matt Dowle and Arun Srinivasan. *data.table: Extension of 'data.frame'*. R package version 1.10.4-2. 2017. URL: https://CRAN.R-project.org/package=data.table.

[11] John Fox and Sanford Weisberg. *car: Companion to Applied Regression*. R package version 2.1-6. 2017. URL: https://CRAN.R-project.org/package=car.

[12] Mark A. Friedl et al. "MODIS Collection 5 global land cover: Algorithm refinements and characterization of new datasets". In: *Remote Sensing of Environment* 114.1 (2010), pp. 168–182. ISSN: 00344257. DOI: 10.1016/j.rse.2009.08.016. URL: http://dx.doi.org/10.1016/j.rse.2009.08.016.

[13] Robert J. Hijmans. *raster: Geographic Data Analysis and Modeling*. R package version 2.6-7. 2017. URL: https://CRAN.R-project.org/package=raster.

[14] Edzer Pebesma and Roger Bivand. *sp: Classes and Methods for Spatial Data*. R package version 1.2-7. 2018. URL: https://CRAN.R-project.org/package=sp.

[15] Navin Ramankutty and Jonathan A Foley. "Estimating historical changes in global land cover: croplands from 1700 to 1992". In: *Global Biogeochemical Cycles* 13 (1999), pp. 997–1027.

[16] Navin Ramankutty et al. "Farming the planet : 1 . Geographic distribution of global agricultural lands in the year 2000". In: 22.August 2007 (2008), pp. 1–19. DOI: 10.1029/2007GB002952.

[17] Andy South. *rworldmap: Mapping Global Data*. R package version 1.3-6. 2016. URL: https://CRAN.R-project.org/package=rworldmap.

[18] QGIS Development Team. *QGIS Geographic Information System.Open Source Geospatial Foundation Project*. 2018. URL: http://qgis.osgeo.org.

[19] UNEP-WCMC and IUCN. *The World Database on Protected Areas (WDPA)/The Global Database on Protected Areas Management Effectiveness (GD-PAME)]*. 2017. URL: https://www.protectedplanet.net/ (visited on 11/01/2017).

[20] Hadley Wickham. *plyr: Tools for Splitting, Applying and Combining Data*. R package version 1.8.4. 2016. URL: https://CRAN.R-project.org/package=plyr.

[21] Hadley Wickham. *reshape2: Flexibly Reshape Data: A Reboot of the Reshape Package*. R package version 1.4.3. 2017. URL: https://CRAN.R-project.org/package=reshape2.

[22] Hadley Wickham and Winston Chang. *ggplot2: Create Elegant Data Visualisations Using the Grammar of Graphics*. R package version 2.2.1. 2016. URL: https://CRAN.R-project.org/package=ggplot2.

[23] Claus O. Wilke. *cowplot: Streamlined Plot Theme and Plot Annotations for 'ggplot2'*. R package version 0.8.0. 2017. URL: https://CRAN.R-project.org/package=cowplot.

[24] EO Wilson. *Half earth : our planet's fight for life*. New York: Liveright, 2016.

[25] Yihui Xie. *knitr: A General-Purpose Package for Dynamic Report Generation in R*. R package version 1.17. 2017. URL: https://CRAN.R-project.org/package=knitr.