WildfireRisk example session with ACT data

knitr::opts\_chunk$set(echo = TRUE)  
  
library(here)

## here() starts at C:/michael/Rworkspaces/owen/WildfireRisk/package\_testing

library(dplyr)

library(ggplot2)  
library(raster)

library(WildfireRisk)

theme\_set(theme\_bw())  
  
READ\_ALL <- TRUE

## Data

Data are rasters of time since fire and forest cover (0/1) in the ACT. Note that we use raster::readAll to load the rasters into memory rather than accessing them from disk when reading data.

tsf <- raster( here("data/act\_rasters/actplus\_tsf17/hdr.adf") )  
  
forest <- raster( here("data/act\_rasters/actfor\_comb/hdr.adf") )  
  
if (READ\_ALL) {  
 tsf <- readAll(tsf)  
 forest <- readAll(forest)  
}

Re-project the forest raster to have the same CRS and bounds as the time since fire raster. Note that we use method = "ngb" (nearest neighbour resampling) to retain the binary cell values.

Note: the updated layers from Owen are in the same projection so we can skip this step.

forest <- forest %>%  
 projectRaster(tsf, method = "ngb") %>%  
 raster::trim()

## Test raster sampling

Read the set of census block centroids for the ACT.

locations <- read.csv( here("data/ACT\_Test\_Data/act\_cens11\_centres.CSV") )  
   
# check that first three columns are ID, X, Y (if no identifiers, then  
# first two columns should be X, Y)  
head(locations)

## mb\_code11 x y  
## 1 80000010000 689355 6092607  
## 2 80000020000 690852 6094767  
## 3 80000030000 691131 6092941  
## 4 80000040000 692719 6092477  
## 5 80000051000 692554 6093852  
## 6 80000052000 692471 6092915

# make identifiers alpha-numeric  
# the convoluted 'sprintf' bit is to avoid R turning values like 8000050000  
# into "800005+E4"  
locations[, 1] <- stringr::str\_trim( sprintf("%12.0f", locations[, 1]) )  
  
# SUBSET OF LOCATIONS FOR TESTING  
ii <- sample(1:nrow(locations), 500)  
locations <- locations[ii, ]

For each location, generate 80 radial scan lines. Line lengths are drawn from an exponential distribution truncated at 20km maximum length and shifted so that no lines are shorter than 100m.

To set the coordinate reference system of the scan lines we can just pass in one of the rasters for the crs argument. Alternatively we could specify the EPSG code for MGA Zone 55 (28355) or a proj4 format text string for the projection.

length\_fun <- function(n) 100 + rexp\_truncated(n, 1/5000, 20000 - 100)  
lines <- make\_scan\_lines(locations, 80, lengths = length\_fun, crs = tsf)  
  
head(lines)

## Simple feature collection with 6 features and 2 fields  
## geometry type: LINESTRING  
## dimension: XY  
## bbox: xmin: 688991 ymin: 6077819 xmax: 706552.9 ymax: 6079201  
## epsg (SRID): NA  
## proj4string: +proj=utm +zone=55 +south +ellps=GRS80 +towgs84=0,0,0,0,0,0,0 +units=m +no\_defs  
## locationid lineid geometry  
## 1 80023410000 1 LINESTRING (688991 6077819,...  
## 2 80023410000 2 LINESTRING (688991 6077819,...  
## 3 80023410000 3 LINESTRING (688991 6077819,...  
## 4 80023410000 4 LINESTRING (688991 6077819,...  
## 5 80023410000 5 LINESTRING (688991 6077819,...  
## 6 80023410000 6 LINESTRING (688991 6077819,...

## Fire risk calculation

Calculate wildfire travel probability for each scan line using the calculate\_risk function. We pass it the scan lines, the rasters for time since fire and forest cover, and the desired spacing between sample points on each scan line. The function does the raster sampling in batches (to avoid excessive memory use) and then applies the risk equation of Price et al 2015.

The function returns a spatial data frame (an sf object). With the scan lines, the variables for risk prediction, and the calculated probability of wildfire.

system.time(  
 risk.lines <- calculate\_risk(lines, tsf, forest, sample.spacing = 100)  
)

## Warning in calculate\_risk(lines, tsf, forest, sample.spacing = 100): Overwriting existing database: pointdata.db

## user system elapsed   
## 47.33 0.84 50.12

Summarize line probabilities (mean and quantiles) for each location:

risk.loc <- summarize\_location\_risk(risk.lines)  
  
head(risk.loc)

## Simple feature collection with 6 features and 7 fields  
## geometry type: POINT  
## dimension: XY  
## bbox: xmin: 652531 ymin: 6021575 xmax: 723644 ymax: 6102681  
## epsg (SRID): NA  
## proj4string: +proj=utm +zone=55 +south +ellps=GRS80 +towgs84=0,0,0,0,0,0,0 +units=m +no\_defs  
## # A tibble: 6 x 8  
## locationid pobs\_mean pobs\_25 pobs\_75 pmax\_mean pmax\_25 pmax\_75  
## <fct> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 102328400~ 0.554 0.126 0.973 0.687 0.486 0.990  
## 2 102341710~ 0.651 0.170 1.000 0.675 0.226 1.000  
## 3 102341900~ 0.899 0.977 1.000 0.913 0.991 1.000  
## 4 105742500~ 0.507 0.00241 0.999 0.529 0.00379 1.000  
## 5 105743660~ 0.501 0.0219 0.990 0.530 0.0383 0.994  
## 6 105744420~ 0.689 0.173 1.000 0.710 0.229 1.000  
## # ... with 1 more variable: geometry <POINT [m]>

## Ignition probabilities in location neighbourhoods

ignitionprob <- raster(here("data/act\_rasters/ignition/act\_igden\_n2/hdr.adf"))  
  
ignition.loc <- summarize\_location\_nbrhood(lines, ignitionprob)

## Burn blocks

burnblocks <- st\_read(here("data/ACT\_Test\_Data/ACTBurnBlocks\_gt1.shp") )

## Reading layer `ACTBurnBlocks\_gt1' from data source `C:\michael\Rworkspaces\owen\WildfireRisk\package\_testing\data\ACT\_Test\_Data\ACTBurnBlocks\_gt1.shp' using driver `ESRI Shapefile'  
## Simple feature collection with 861 features and 18 fields  
## geometry type: POLYGON  
## dimension: XY  
## bbox: xmin: 638160 ymin: 6018102 xmax: 721208.6 ymax: 6111042  
## epsg (SRID): NA  
## proj4string: +proj=utm +zone=55 +south +ellps=GRS80 +units=m +no\_defs

burnblocks <- st\_transform(burnblocks, st\_crs(lines))

For each burn block, summarize the risk values of scan lines that intersect it wholly or partially. Note that this doesn’t take into account the length or proportion of the scan line crossing the block.

system.time(  
 risk.block <- summarize\_block\_risk(risk.lines, burnblocks)  
)

## user system elapsed   
## 5.56 0.00 5.56

head(risk.block)

## Simple feature collection with 6 features and 26 fields  
## geometry type: POLYGON  
## dimension: XY  
## bbox: xmin: 656460.6 ymin: 6041681 xmax: 666473.2 ymax: 6079492  
## epsg (SRID): NA  
## proj4string: +proj=utm +zone=55 +south +ellps=GRS80 +units=m +no\_defs  
## OBJECTID SHAPE\_Leng SHAPE\_Area IGNITION SOURCE MAPPED DIGITISED COMMENTS  
## 1 1 31956.21 25476284 <NA> <NA> <NA> <NA> <NA>  
## 2 2 14498.31 5420733 <NA> <NA> <NA> <NA> <NA>  
## 3 3 35227.83 41624859 <NA> <NA> <NA> <NA> <NA>  
## 4 4 33125.73 27267206 <NA> <NA> <NA> <NA> <NA>  
## 5 5 39999.45 42605667 <NA> <NA> <NA> <NA> <NA>  
## 6 6 22826.31 9075019 <NA> <NA> <NA> <NA> <NA>  
## YEARNUM DATE\_ TEXT\_ID BURN\_DATE CREATED\_US CREATED\_DA LAST\_EDITE  
## 1 0 <NA> <NA> 0 <NA> <NA> <NA>  
## 2 0 <NA> <NA> 0 <NA> <NA> <NA>  
## 3 0 <NA> <NA> 0 <NA> <NA> <NA>  
## 4 0 <NA> <NA> 0 <NA> <NA> <NA>  
## 5 0 <NA> <NA> 0 <NA> <NA> <NA>  
## 6 0 <NA> <NA> 0 <NA> <NA> <NA>  
## LAST\_EDI\_1 ORIG\_FID hectares nlines nlocations pobs\_mean pobs\_25  
## 1 <NA> 0 2547.630 11 1 0.9360023 0.93396754  
## 2 <NA> 1 542.073 3 2 0.6650014 0.53001100  
## 3 <NA> 2 4162.490 5 2 0.4171065 0.06097424  
## 4 <NA> 3 2726.720 2 1 0.1562645 0.10861939  
## 5 <NA> 4 4260.570 80 1 0.9806951 0.98841541  
## 6 <NA> 5 907.502 2 1 0.9340250 0.91627356  
## pobs\_75 pmax\_mean pmax\_25 pmax\_75 geometry  
## 1 0.9930367 0.9993195 0.9991255 0.9999163 POLYGON ((666325.6 6066187,...  
## 2 0.9339675 0.8981946 0.8477195 0.9991255 POLYGON ((661507.7 6057730,...  
## 3 0.9329530 0.7135091 0.6558035 0.9991061 POLYGON ((658045.2 6056219,...  
## 4 0.2039097 0.7914943 0.7236489 0.8593397 POLYGON ((666003.4 6049847,...  
## 5 0.9998335 0.9997539 0.9998474 0.9999979 POLYGON ((665874.8 6076181,...  
## 6 0.9517765 0.9990498 0.9987817 0.9993178 POLYGON ((662639.1 6078932,...

## Simulate prescribed burning and update risk

Set time since fire of a selected block to 0 to simulate prescribed burning, then re-calculate risk for lines passing through the block.

# Check if there is enough memory to load all sample point data at once for faster   
# calculations. Here we will set a limit of 2Gb  
  
enough.mem <- pointdata\_memory(risk.lines, "Gb") < 2  
  
cat("Check for memory to load all point data returned", enough.mem, "\n")

## Check for memory to load all point data returned TRUE

system.time(  
 risk.block.treated <- treat\_blocks(risk.block,   
 risk.lines,   
 in.memory = enough.mem)  
)

## user system elapsed   
## 293.39 1.47 294.94

Compare risk values for blocks when treated versus baseline (observed and maximum time since fire).

dat <- risk.block.treated %>%  
 # lose spatial bits  
 as.data.frame() %>%  
   
 dplyr::select(OBJECTID, nlines, pobs\_mean, pmax\_mean, ptreat\_mean) %>%  
   
 dplyr::filter(nlines > 0) %>%  
   
 tidyr::gather(var, baseline, c(pobs\_mean, pmax\_mean)) %>%  
   
 mutate(ptreat\_reduction = baseline - ptreat\_mean) %>%  
   
 # arrange data so that points for blocks with more scan lines  
 # are plotted last  
 arrange(var, nlines)  
  
  
ggplot(data = dat) +  
 geom\_point(aes(x = baseline, y = ptreat\_reduction, colour = nlines),  
 size = 2) +  
   
 scale\_color\_distiller(name = "Number of\nscan lines\nin block",   
 palette = "Reds", direction = 1) +  
   
 # Use square root scale for y-axis to open up the point cloud  
 scale\_y\_sqrt() +  
   
 # geom\_abline(intercept = 0, slope = 1, linetype = "dashed") +  
   
 labs(x = "Baseline risk value",  
 y = "Reduction from treatment") +  
   
 facet\_wrap(~ var,   
 labeller = as\_labeller( c(pmax\_mean = "Baseline long unburnt",  
 pobs\_mean = "Baseline actual TSF")) )

