Accessory publication

Potentially limited detectability of short-term changes in boreal fire regimes: a simulation study

Juha M. Metsaranta

Natural Resources Canada, Canadian Forest Service, Pacific Forestry Centre, 506 West Burnside Road, Victoria, BC, V8Z 1M5, Canada.

Present address: Natural Resources Canada, Canadian Forest Service, Northern Forestry Centre, 5320 122 Street, Edmonton, AB, T6H 3S5, Canada. Email: jmetsara@nrcan.gc.ca

Introduction

This accessory publication contains the R code (R Development Core Team 2009) used to conduct the simulations in this paper. It may be used by interested readers to confirm, refute, or expand upon the results presented. It consists of two parts. Appendix 1 provides the code for a function *firecycles* which does the simulation calculations. Appendix 2 provides further code that uses *firecycles* to conduct the scenarios described in the manuscript. There is extensive commentary throughout both Appendix 1 and Appendix 2, such that readers familiar with R should be able to easily understand how the simulations were conducted, and how to modify the code to conduct their own simulations or extract additional results.

References

R Development Core Team (2009). 'R: A language and environment for statistical computing.' (R Foundation for Statistical Computing: Vienna) Available at www.r-project.org [Verified 9 December 2010]

Appendix 1.

```
######################################################
# This is a function called "firecycles" that was used to do the
# simulation modeling presnted in this paper.
#
# The arguments required by the function are:
#
# fires = The annual area burned observations (ha/year)
# areamin = Minimum estimate of forest area (ha)
# areamax = Maximum estimate of forest area (ha)
# nsim = Total number of simulations (n)
# max_mult = Multiplier of maximum area burned in "fires" for truncating drawn fires (e.g. a
value of
#       2 will allow the maximum area burned to be twice the maximum in "fires").
# randomtype = what distribution to use for area burned (Must "1" or "2").
#
#      Type 1 is log normal
#      Type 2 is empirical quantile function
#
# ramptype = Which parameter will be ramped up to simulate increased annual area burned.
Only
#      Type 4 ramp up will work when randomtype = 2 (Must be "1", "2", "3", or "4")
#
#      Type 1 is no ramp up,
#      Type 2 is mu,
#      Type 3 is sigma,
#      Type 4 is the drawn AAB.
#
# rampvalue = The value that the parameter (indicated by ramptype), will be ramped up to.
# ramptime = The time period over which the ramp up to the value given in rampvalue will
occur. If ramptime = 1,
#      it will be instantaneous in step 1.  (e.g a value of 50 will give a rampup time of 50
years).)
#
# First, some required libraries
#
library(MASS)
library(zoo)
#
firecycles<-
function(fires,areamin,areamax,nsim,maxmult,randomtype,ramptype,rampvalue,ramptime){

  at_fifty=c()          # Results
  mean_fifty=c()               # Results
  firecycles=c()               # Results
  forestareas=c()              # Results
  f_mult=c()            # Multiplier applied to drawn number
  mu_draw=c()           # Lognormal parameter mu used to draw number
  sd_draw=c()           # Lognormal parameter sd used to draw number
  mu_f=c()              # Lognormal parameter mu estimated from data
  sd_f=c()              # Lognormal parameter sd estimated from data
  slpe_out=c()          # Slope of interpolation line used in ramp up
  intr_out=c()          # Intercept of interpolation line used in ramp up
  runmean_50=c()               # Fire cycle estimated for a fifty year running mean AAB
  runmean_100=c()              # Fire cycle estimated for a 100 year running mean AAB
  runmean_200=c()              # Fire cycle estimated for a 200 year running mean AAB
  runmean_500=c()              # Fire cycle estimated for a 500 year running mean AAB
  drawnf=c()            # Vector of random AAB used in calculating the running mean
  numdraws=0            # Total number of AAB drawn
    #
    # Determine the maximum area burned in "fires"
    #
    maxf= max(fires)
    #
    # Determine what distribution type to use
    #
    if (randomtype==1){
      #
      # Fits the lognormal distribution and gets the parameters
      #
      LogNorm<-fitdistr(fires,"lognormal")
      muf<-as.double(LogNorm$estimate[1])
      sdf<-as.double(LogNorm$estimate[2])
      }
    if (randomtype==2){
      #
```

```
      # This will use the empirical quantile function.  To do this requires adding
      # zero and maxmult times maxf to the input fire time series.
      #
      n=length(fires)
      fires[[n+1]]=0
      fires[[n+2]]=maxmult*maxf
      muf=-999
      sdf=-999
      }
    #
    # The simulations start here
    #
    for (i in 1:nsim){
      #
      # Reset the output and some other variables
      #
      t_cyc=0                              # The years in fire cycle counter
      s_fire=0                             # The cumulative sum of drawn burned areas
      fifty=-999                                  #
      the_av=-999                                 #
      fa_cyc = (runif(1)*(areamax-areamin)) + areamin       # The total burnable area for
this simulation
        #
        # This will loop through until the sum of drawn fires
        # is equal to the total forest area
        #
        while(s_fire<fa_cyc) {
          #
          # Add one year to the years in fire cycle counter
          #
          t_cyc=t_cyc+1
          #
          # Determine how to adjust the parameters and/or the multiplier applied to the
drawn number,
          # depending upon the arguments for randomtype, ramptype, ramptime and rampvalue.
          #
          if(randomtype==1){
           #
            # These are the options for log normal random numbers
            #
            if(ramptype==1){
                #
                # This is the no ramp-up option.
                # Uses the drawn number directly.
                #
                slpe=1
                intr=1
                mudraw=muf
                sddraw=sdf
                fmult=1
              } # end if
            if(ramptype==2){
                #
                # This will change the parameter mu.
                #
                if (ramptime==1){
                  #
                  # This option instantaneously changes mu in the first time step
                  # to the value specified in the argument rampvalue.
                  #
                  slpe=1
                  intr=1
                  mudraw=rampvalue
                  sddraw=sdf
                  fmult=1}
                else{
                  #
                  # This option linear interpolates mu from the first time step to the time
step
                  # specified in the argument ramptime to the value specified in the
argument rampvalue.
                  #
                  slpe=(rampvalue-muf)/(ramptime-1)
                  intr=muf-slpe
                  if(t_cyc<ramptime+1){mudraw=slpe*t_cyc+intr}else{mudraw=rampvalue}
                  sddraw=sdf
                  fmult=1}
```

```
            } # end if
          if(ramptype==3){
              #
              # This will change the parameter sigma.
              #
              if (ramptime==1){
                #
                # This option instantaneously changes sigma in the first time step
                # to the value specified in the argument rampvalue.
                #
                slpe=1
                intr=1
                mudraw=muf
                sddraw=rampvalue
                fmult=1}
              else{
                #
                # This option linear interpolates sigma from the first time step to the
time step
                # specified in the argument ramptime to the value specified in the
argument rampvalue.
                #
                slpe=(rampvalue-sdf)/(ramptime-1)
                intr=sdf-slpe
                mudraw=muf
                if(t_cyc<ramptime+1){sddraw=slpe*t_cyc+intr}else{sddraw=rampvalue}
                fmult=1}
            } # end if
          if(ramptype==4){
              #
              # This will change not change the parameters, but instead the multiplier
applied to the drawn number.
              #
              if (ramptime==1){
                #
                # This option instantaneously changes the multiplier applied to the drawn
number
                # in the first time step to the value specified in the argument rampvalue.
                #
                slpe=1
                intr=1
                mudraw=muf
                sddraw=sdf
                fmult=rampvalue}
              else{
                #
                # This option linear interpolates the multiplier applied to the drawn
number from the first time step
                # to the time step specified in the argument ramptime to the value
specified in the
                # argument rampvalue.
                #
                slpe=(rampvalue-1)/(ramptime-1)
                intr=1-slpe
                mudraw=muf
                sddraw=sdf
                if(t_cyc<ramptime+1){fmult=slpe*t_cyc+intr}else{fmult=rampvalue}}
            }
        #
        # Here is where it actually draws the random number
        #
        r_fire = (rlnorm(1,meanlog=mudraw, sdlog=sddraw)) * fmult
        }
        if (randomtype==2){
          #
          # These are the options for random numbers from the empirical quantile function
          #
          if(ramptype==4){
            #
            # This will change not change the parameters, but instead the multiplier
applied to the drawn number.
            #
            if (ramptime==1){
                #
                # This option instantaneously changes the multiplier applied to the drawn
number
                # in the first time step to the value specified in the argument rampvalue.
```

```
                  #
                  slpe=1
                  intr=1
                  mudraw=muf
                  sddraw=sdf
                  fmult=rampvalue}
                else{
                  #
                  # This option linear interpolates the multiplier applied to the drawn 
number from the first time step
                  # to the time step specified in the argument ramptime to the value 
specified in the
                  # argument rampvalue.
                  #
                  slpe=(rampvalue-1)/(ramptime-1)
                  intr=1-slpe
                  mudraw=muf
                  sddraw=sdf
                  if(t_cyc<ramptime+1){fmult=slpe*t_cyc+intr}else{fmult=rampvalue}}
                }
            else{
            #
            # No other ramp up type is applicable to the quantile function.  Thus, 
everything else is the no ramp-up
            # option that uses the drawn number directly
            #
            slpe=1
            intr=1
            mudraw=-999
            sddraw=-999
            fmult=1
            }
          #
          # Here is where it actually draws the random number
          #
          r_fire = (quantile(fires,runif(1)))*fmult
          }
          #
          # Here is where it will truncate the drawn number (r_fire) if it is too big.  The 
truncation point is
          # determined by the maximum area burned in “fires” (maxf) and the value of the 
argument max_mult.
          # The truncation point is calculates as maxf * max_mult
          #
          if(r_fire > maxf * maxmult){r_fire=maxf * maxmult}else{r_fire = r_fire}
          #
          # Here, the value of the drawn number (r_fire) to the cumulative sum of drawn 
numbers (s_fire)
          #
          s_fire = s_fire + r_fire
          #
          # If this is the 50th number drawn for this simulation (ie the 50th simulation 
year), then
          # it calculates the average annual area burned and the estimated fire cycle for 
that 50 year series
          #
          if(t_cyc == 50){the_av=(s_fire/50)}
          if(t_cyc == 50){fifty=(fa_cyc/the_av)}
          #
          # This part outputs some additional information about the drawn area burned and 
the
          # running mean for the first 10,000 drawn numbers (for all simulations).  This 
also
          # outputs some information about the log normal parameters and the interpolation 
used
          # for ramping up.  These can be used to ensure that rampup was done correctly
          #
          numdraws = numdraws+1                   # Total number of random draws made to 
this point
          if (numdraws < 10001){
            drawnf[[numdraws]]=r_fire          # The number that as drawn
            f_mult[[numdraws]]=fmult              # The multiplier that was applied to the 
drawn number
            mu_f[[numdraws]]=muf                  # Log-normal mu parameter estimated from 
the data
            sd_f[[numdraws]]=sdf                  # Log-normal sigma parameter estimated 
from the data
```

```
            slpe_out[[numdraws]]=slpe             # The slope used during the linear
interpolation of the rampup
            intr_out[[numdraws]]=intr             # The intercept used during the linear
interpolation of the rampup
            mu_draw[[numdraws]]=mudraw            # Log-normal mu parameter used to draw the
number
            sd_draw[[numdraws]]=sddraw            # Log-normal sigma parameter used to draw
the number
              if (numdraws > 49)
                #
                # Fifty (50) year running mean of drawn annual area burned
                #
                {r_m=fa_cyc/rollmean(drawnf,50)[length(rollmean(drawnf,50))]}
                else{r_m=-999}
                runmean_50[[numdraws]]=r_m
              if (numdraws > 99)
                #
                # One hundred (100) year running mean of drawn annual area burned
                #
                {r_m=fa_cyc/rollmean(drawnf,100)[length(rollmean(drawnf,100))]}
                else{r_m=-999}
                runmean_100[[numdraws]]=r_m
              if (numdraws >199)
                #
                # Two hundred (200) year running mean of drawn annual area burned
                #
                {r_m=fa_cyc/rollmean(drawnf,200)[length(rollmean(drawnf,200))]}
                else {r_m=-999}
                runmean_200[[numdraws]]=r_m
            if (numdraws > 499)
                #
                # Five hundred (500) year running mean of drawn annual area burned
                #
             {r_m=fa_cyc/rollmean(drawnf,500)[length(rollmean(drawnf,500))]}
             else {r_m=-999}
             runmean_500[[numdraws]]=r_m
          }
        }
    #
    # Here is where it adds the result of this simulation
    # to the result vectors
    #
    at_fifty[[i]]=fifty                   # The fire cycle estimated from the average of the
first 50 years
    mean_fifty[[i]]=the_av                # Average annual area burned in the first 50 years
    firecycles[[i]]=t_cyc                 # Number of years it took for cumulative sum to
equal the forest area
    forestareas[[i]]=fa_cyc               # The forest area used in this simulation
    }
#
# This is the output.  The first part, a, includes the results for the “nsim” simulations.
# The second part, b, includes the additional information about the first 10,000 drawn
numbers
#
a=data.frame(at_fifty,mean_fifty,firecycles,forestareas)
b=data.frame(drawnf,f_mult,mu_draw,mu_f,sd_draw,sd_f,slpe_out,intr_out,runmean_50,runmean_10
0,runmean_200,runmean_500)
return(list(cycles=a,fires=b))
}
################################################################################
```

Appendix 2.

```
######################################################
# This is the code that conducts the simulations in the paper
#
# First, some data are needed
#
aab<-
c(55913,35972,750301,58317,14324,167947,72322,25318,64920,471358,31247,212349,212799,58849,2
5739,18299,7961,83393,
207602,23430,146355,1177379,2059778,481812,25069,99884,24792,7339,112191,99410,829875,136340
,66844,28881,358140,15560,1202198,14775,1321,940381,239344)        # The annual area burned
data (ha/year)
year<-seq(1959,1999,1)        # Years over which aab was observed
minarea<-47900000                     # Minimum forest area (ha)
maxarea<-49300000                     # Maximum forest area (ha)
nsim<-10000                           # Total number of simulations for a scenario
#
######################################################
# These are the results for the default scenario.  These
# are the main results that are used in the paper, and are
# presented in the bottom panel of Fig. 2c, as well as
# in Fig. 3a.  These results as the basis of the power
# analysis in Fig 3b.
#
    Default<-firecycles(aab,minarea,maxarea,nsim,2,1,1,-999,-999)
    #
    # These are the results for when simulations run to completion, and
    # they are plotted in Fig. 2c, as well as Fig. 3a (Default)
    #

quantile(subset(Default$cycles$firecycles,Default$cycles$firecycles>0),c(0.025,0.5,0.975))
    #
    # These are the results from the average area burned in the first 50 years
    # of each simulation. They are plotted in Fig. 3a (Default at 50)
    #
    quantile(subset(Default$cycles$at_fifty,Default$cycles$at_fifty>0),c(0.025,0.5,0.975))

######################################################
# Sensitivity analysis scenarios for examining
# the effect of larger maximum area burned truncation
# points.  These results are not presented in the
# manuscript, but are referred to in the Results, in the
# section entitled “Sensitivity analysis”, and are described
# extensively in the Methods, also in the section entitled “Sensitivity
# Analysis”
#
    Default_root2<-firecycles(aab,minarea,maxarea,nsim,sqrt(2),1,1,-999,-999)
    Default_4<-firecycles(aab,minarea,maxarea,nsim,4,1,1,-999,-999)
    Default_8<-firecycles(aab,minarea,maxarea,nsim,8,1,1,-999,-999)
    Default_16<-firecycles(aab,minarea,maxarea,nsim,16,1,1,-999,-999)


quantile(subset(Default_root2$cycles$firecycles,Default_root2$cycles$firecycles>0),c(0.025,0
.5,0.975))

quantile(subset(Default$cycles$firecycles,Default$cycles$firecycles>0),c(0.025,0.5,0.975))

quantile(subset(Default_4$cycles$firecycles,Default_4$cycles$firecycles>0),c(0.025,0.5,0.975
))

quantile(subset(Default_8$cycles$firecycles,Default_8$cycles$firecycles>0),c(0.025,0.5,0.975
))

quantile(subset(Default_16$cycles$firecycles,Default_16$cycles$firecycles>0),c(0.025,0.5,0.9
75))

######################################################
# Sensitivity analysis scenarios for examining the
# effect of a larger forest area and different
# distribution for modeling annual area burned
# These results are not presented in the
# manuscript, but are referred to in the Results, in the
# section entitled “Sensitivity analysis”, and are described
# extensively in the Methods, also in the section entitled “Sensitivity
# Analysis”
#
```

```
    LargeFor<-firecycles(aab,59000000,59000001,nsim,2,1,1,-999,-999)
    DiffDist<-firecycles(aab,minarea,maxarea,nsim,2,2,1,-999,-999)
    Both<-firecycles(aab,59000000,59000001,nsim,2,2,1,-999,-999)


quantile(subset(LargeFor$cycles$firecycles,LargeFor$cycles$firecycles>0),c(0.025,0.5,0.975))

quantile(subset(DiffDist$cycles$firecycles,DiffDist$cycles$firecycles>0),c(0.025,0.5,0.975))
    quantile(subset(Both$cycles$firecycles,Both$cycles$firecycles>0),c(0.025,0.5,0.975))

######################################################
# Scenarios with different ways of implementing
# a doubling of annual area burned
#
# These three scenarios (RampMu, RampSigma, and RampDraw
# gradually increase mu, sigma, and the drawn random number over
# the first fifty years of the simulation
#
# The results from the RampMu scenario are presented in the
# manuscript in Fig 3a (Gradual at 50), and are also used in the
# power analysis in Fig 3b.
#
# The other two scenarios (RampSigma and RampDraw) are not
# presented in the manuscript, but are referred to in the Methods, in
# the section entitled “Detecting Changes in the fire cycle”
#
    RampMu<-firecycles(aab,minarea,maxarea,nsim,2,1,2,12.03,50)
    RampSigma<-firecycles(aab,minarea,maxarea,nsim,2,1,3,1.93,50)
    RampDraw<-firecycles(aab,minarea,maxarea,nsim,2,1,4,2,50)

    quantile(subset(RampMu$cycles$firecycles,RampMu$cycles$firecycles>0),c(0.025,0.5,0.975))

quantile(subset(RampSigma$cycles$firecycles,RampSigma$cycles$firecycles>0),c(0.025,0.5,0.975
))

quantile(subset(RampDraw$cycles$firecycles,RampDraw$cycles$firecycles>0),c(0.025,0.5,0.975))
    quantile(subset(RampMu$cycles$at_fifty,RampMu$cycles$at_fifty>0),c(0.025,0.5,0.975))

quantile(subset(RampSigma$cycles$at_fifty,RampSigma$cycles$at_fifty>0),c(0.025,0.5,0.975))
    quantile(subset(RampDraw$cycles$at_fifty,RampDraw$cycles$at_fifty>0),c(0.025,0.5,0.975))

# These three scenarios (SteppMu, StepSigma, and StepDraw
# instantaneously increase mu, sigma, and the drawn random number in the
# first year of the simulation.
#
# The results from the StepMu scenario are presented in the
# manuscript in Fig 3a (Step at 50), and are also used in the
# power analysis in Fig 3b.
#
# The other two scenarios (StempSigma and StepDraw) are not
# presented in the manuscript, but are referred to in the Methods, in
# the section entitled “Detecting Changes in the fire cycle”
#
    StepMu<-firecycles(aab,minarea,maxarea,nsim,2,1,2,12.03,1)
    StepSigma<-firecycles(aab,minarea,maxarea,nsim,2,1,3,1.93,1)
    StepDraw<-firecycles(aab,minarea,maxarea,nsim,2,1,4,2,1)

    quantile(subset(StepMu$cycles$firecycles,StepMu$cycles$firecycles>0),c(0.025,0.5,0.975))

quantile(subset(StepSigma$cycles$firecycles,StepSigma$cycles$firecycles>0),c(0.025,0.5,0.975
))

quantile(subset(StepDraw$cycles$firecycles,StepDraw$cycles$firecycles>0),c(0.025,0.5,0.975))
    quantile(subset(StepMu$cycles$at_fifty,StepMu$cycles$at_fifty>0),c(0.025,0.5,0.975))

quantile(subset(StepSigma$cycles$at_fifty,StepSigma$cycles$at_fifty>0),c(0.025,0.5,0.975))
    quantile(subset(StepDraw$cycles$at_fifty,StepDraw$cycles$at_fifty>0),c(0.025,0.5,0.975))
#
######################################################
```