Various GLGM examples

Patrick Brown

January 3, 2018

```
library('mapmisc')

## Loading required package:  sp
## Loading required package:  raster
## map images will be cached in  /tmp/Rtmp322Bbk/mapmiscCache

library("geostatsp")

## Loading required package:  Matrix

data('swissRain')

havePackages = c(
  'INLA' = requireNamespace('INLA', quietly=TRUE)
)

print(havePackages)

##  INLA
## FALSE

swissRain$lograin = log(swissRain$rain)
swissAltitudeCrop = raster::mask(swissAltitude,swissBorder)
```

number of cells... smaller is faster but less interesting

```
if(Sys.info()['user'] =='patrick'){
  Ncell = 60
} else {
  Ncell = 25
}
```

standard formula

```
if(all(havePackages)) {

  swissFit =  glgm(lograin~ CHE_alt,
    swissRain,
    Ncell,
    covariates=swissAltitudeCrop, family="gaussian", buffer=20000,
    priorCI=list(
      sd=c(2, 0.05),
      range=c(500000, 0.5)),
    control.family=list(hyper=list(prec=
      list(prior="pc.prec",
          param=c(1, 0.1))))
  )
  knitr::kable(swissFit$parameters$summary, digits=3)


  swissExc = excProb(
    x=swissFit,  random=TRUE,
    threshold=0)

  plot(swissExc, breaks = c(0, 0.2, 0.8, 0.95, 1.00001),
    col=c('green','yellow','orange','red'))

  plot(swissBorder, add=TRUE)


  swissExcP = excProb(
    swissFit$inla$marginals.predict, 3,
      template=swissFit$raster)
  plot(swissExcP, breaks = c(0, 0.2, 0.8, 0.95, 1.00001),
    col=c('green','yellow','orange','red'))
  plot(swissBorder, add=TRUE)

  matplot(
    swissFit$parameters$sd$posterior[,'x'],
    swissFit$parameters$sd$posterior[,c('y','prior')],
    lty=1, col=c('black','red'), type='l',
    xlab='sd', ylab='dens', xlim = c(0,3))

  matplot(
    swissFit$parameters$range$posterior[,'x'],
    swissFit$parameters$range$posterior[,c('y','prior')],
    lty=1, col=c('black','red'), type='l',
    xlab='sd', ylab='dens', xlim = c(0,3))
}
```

non-parametric elevation effect

```
altSeq = exp(seq(
  log(100), log(5000),
  by = log(2)/5))

swissAltCut = raster::cut(
 swissAltitudeCrop,
 breaks=altSeq
)
names(swissAltCut) = 'bqrnt'

if(all(havePackages)) {

 swissFitNp = glgm(
  formula = lograin ~ f(bqrnt, model = 'rw2', scale.model=TRUE,
   values = 1:length(altSeq),
   prior = 'pc.prec', param = c(0.1, 0.01)),
  data=swissRain,
  grid = Ncell,
  covariates=swissAltCut,
  family="gaussian", buffer=20000,
  priorCI=list(sd=c(u = 0.5, alpha = 0.1), range=c(50000,500000)),
  control.mode=list(theta=c(2.7, 4.3, 0.46, 1.9),restart=TRUE),
  control.family=list(hyper=list(prec=list(prior="loggamma",
     param=c(.1, .1))))
 )
 knitr::kable(swissFitNp$parameters$summary, digits=3)

 matplot(
  altSeq,
  exp(swissFitNp$inla$summary.random$bqrnt[,
   c('0.025quant', '0.975quant', '0.5quant')]),
  log='xy',
  xlab ='elevation', ylab='rr',
 type='l',
  lty = 1,
  col=c('grey','grey','black')
  )

 swissExcP = excProb(swissFitNp$inla$marginals.predict,
  3, template=swissFitNp$raster)
  plot(swissExcP, breaks = c(0, 0.2, 0.8, 0.95, 1.00001),
    col=c('green','yellow','orange','red'))
  plot(swissBorder, add=TRUE)

}
```

intercept only, named response variable

```
if(all(havePackages)) {
  swissFit =  glgm("lograin", swissRain, Ncell,
    covariates=swissAltitude, family="gaussian", buffer=20000,
    priorCI=list(sd=c(0.2, 2), range=c(50000,500000)),
    control.mode=list(theta=c(1.9,0.15,2.6),restart=TRUE)
  )

  knitr::kable(swissFit$parameters$summary, digits=4)
}
```

intercept only, add a covariate just to confuse glgm.

```
if(all(havePackages)) {

  swissFit =  glgm(
    formula=lograin~1,
    data=swissRain,
    grid=Ncell,
    covariates=swissAltitude,
    family="gaussian", buffer=20000,
    priorCI=list(sd=c(0.2, 2), range=c(50000,500000)),
    control.mode=list(theta=c(1.9,0.15,2.6),restart=TRUE),
    control.family=list(hyper=list(prec=list(prior="loggamma", param=c(.1, .1))))
  )

  knitr::kable(swissFit$parameters$summary, digits=3)

  swissExc = excProb(
    swissFit$inla$marginals.random$space, 0,
    template=swissFit$raster)
  plot(swissExc, breaks = c(0, 0.2, 0.8, 0.95, 1.00001),
    col=c('green','yellow','orange','red'))
  plot(swissBorder, add=TRUE)
}
```

covariates are in data

```
newdat = swissRain
newdat$elev = extract(swissAltitude, swissRain)
if(all(havePackages)) {
  swissFit =  glgm(lograin~ elev + land,
    newdat, Ncell,
    covariates=list(land=swissLandType),
    family="gaussian", buffer=40000,
    priorCI=list(sd=c(0.2, 2), range=c(50000,500000)),
    control.mode=list(theta=c(1.9,0.15,2.6),restart=TRUE),
    control.family=list(hyper=list(prec=list(prior="loggamma",
          param=c(.1, .1))))
  )
  knitr::kable(swissFit$parameters$summary, digits=3)

  plot(swissFit$raster[['predict.mean']])
}
```

formula, named list elements

```
if(all(havePackages)) {

  swissFit =  glgm(lograin~ elev,
    swissRain, Ncell,
    covariates=list(elev=swissAltitude),
    family="gaussian", buffer=20000,
    priorCI=list(sd=c(0.2, 2), range=c(50000,500000)),
    control.mode=list(theta=c(1.9,0.15,2.6),restart=TRUE),
    control.family=list(hyper=list(prec=list(prior="loggamma",
          param=c(.1, .1))))
  )
  swissFit$parameters$summary
}
```

categorical covariates

```
if(all(havePackages)) {
  swissFit =  glgm(
    formula = lograin ~ elev + factor(land),
    data = swissRain, grid = Ncell,
    covariates=list(elev=swissAltitude,land=swissLandType),
    family="gaussian", buffer=20000,
    priorCI=list(sd=c(0.2, 2), range=c(50000,500000)),
    control.mode=list(theta=c(1.9,0.15,2.6),restart=TRUE),
    control.family=list(hyper=list(prec=list(prior="loggamma",
          param=c(.1, .1))))
  )

  knitr::kable(swissFit$parameters$summary, digits=3)


  plot(swissFit$raster[['predict.mean']])

}
```

put some missing values in covaritates also don't put factor() in formula

```
temp = values(swissAltitude)
temp[seq(10000,12000)] = NA
values(swissAltitude) = temp
if(all(havePackages)) {

  swissFitMissing =  glgm(rain ~ elev + land,swissRain,  Ncell,
    covariates=list(elev=swissAltitude,land=swissLandType),
    family="gaussian", buffer=20000,
    priorCI=list(sd=c(0.2, 2), range=c(50000,500000)),
    control.mode=list(theta=c(1.9,0.15,2.6),restart=TRUE),
    control.family=list(hyper=list(prec=list(prior="loggamma",
          param=c(.1, .1))))
  )
  knitr::kable(swissFitMissing$parameters$summary, digits=3)

}
```

these tests are time consuming, so only patrick will do them

```
if(all(havePackages) & Sys.info()['user'] =='patrick') {

  data('loaloa')
  rcl = rbind(
    # wedlands and mixed forests to forest
    c(5,2),c(11,2),
# savannas to woody savannas
    c(9,8),
    # croplands and urban changed to crop/natural mosaid
    c(12,14),c(13,14))
  ltLoaR = reclassify(ltLoa, rcl)
  levels(ltLoaR) = levels(ltLoa)


```

```r
  elevationLoa = elevationLoa - 750
  elevLow = reclassify(elevationLoa, c(0, Inf, 0))
  elevHigh = reclassify(elevationLoa, c(-Inf, 0, 0))

  covList = list(elLow = elevLow, elHigh = elevHigh,
    land = ltLoaR, evi=eviLoa)

  loaFit = glgm(
    y ~ land + evi + elHigh + elLow, #+ f(villageID,model="iid"),
    loaloa,
    Ncell,
    covariates=covList,
    family="binomial", Ntrials = loaloa$N,
    shape=2, buffer=25000,
    priorCI = list(sd=c(0.2, 4), range=c(20000,500000)))

  loaFit$par$summary

  plot(loaFit$raster[['predict.exp']])


# prior for observation standard deviation
  swissFit =  glgm( formula="lograin",data=swissRain, grid=Ncell,
    covariates=swissAltitude, family="gaussian", buffer=20000,
    priorCI=list(sd=c(0.1, 2), range=c(50000,500000),
      sdNugget=c(0.1, 2)),
    control.mode=list(theta=c(1.9,0.15,2.6),restart=TRUE)
  )



# a model with little data, posterior should be same as prior

  data2 = SpatialPointsDataFrame(cbind(c(1,0), c(0,1)),
    data=data.frame(y=c(0,0), offset=c(-50,-50), x=c(-1,1)))

  res = glgm(data=data2, grid=20, formula=y~1 + x+offset(offset),
    covariates=NULL,
    priorCI = list(sd=c(0.3,0.5), range=c(0.25, 0.4)),
    family="poisson",buffer=0.5,
    control.fixed=list(mean.intercept=0, prec.intercept=1,
      mean=0,prec=4),
    control.mode=list(theta=c(2, 2),restart=TRUE)
  )
  par(mfrow=c(3,1))

# intercept
  plot(res$inla$marginals.fixed[['(Intercept)']], col='blue', type='l',
    xlab='intercept',lwd=3)
  xseq = res$inla$marginals.fixed[['(Intercept)']][,'x']
  lines(xseq, dnorm(xseq, 0, 1),col='red',lty=2,lwd=3)
  legend("topright", col=c("blue","red"),lty=1,legend=c("prior","post'r"))

# beta
  plot(res$inla$marginals.fixed[['x']], col='blue', type='l',
    xlab='beta',lwd=3)
  xseq = res$inla$marginals.fixed[['x']][,'x']
  lines(xseq, dnorm(xseq, 0, 1/2),col='red',lty=2,lwd=3)
  legend("topright", col=c("blue","red"),lty=1,legend=c("prior","post'r"))


# sd
  plot(res$parameters$sd$prior,type='l', col='blue',xlab='sd',lwd=3)
  lines(res$parameters$sd$post,col='red',lty=2,lwd=3)
  legend("topright", col=c("blue","red"),lty=1,legend=c("prior","post'r"))


# range
  plot(res$parameters$range$prior,type='l', col='blue', xlab='range',lwd=3)
  lines(res$parameters$range$post,col='red',lty=2,lwd=3)
  legend("topright", col=c("blue","red"),lty=1,legend=c("prior","post'r"))
}
```

covariates are in data, interactions

```
newdat = swissRain
newdat$elev = extract(swissAltitude, swissRain)
if(all(havePackages)) {

  swissFit =  glgm(
    formula = lograin~ elev : land,
    data=newdat,
    grid=squareRaster(swissRain,50),
    covariates=list(land=swissLandType),
    family="gaussian", buffer=0,
    priorCI=list(sd=c(0.2, 2), range=c(50000,500000)),
    control.mode=list(theta=c(1.9,0.15,2.6),restart=TRUE),
    control.family=list(hyper=list(prec=list(prior="loggamma",
          param=c(.1, .1))))
  )

  knitr::kable(swissFit$parameters$summary, digits=3)

  plot(swissFit$raster[['predict.mean']])

}
```

specifying spatial formula

```
swissRain$group = 1+rbinom(length(swissRain), 1, 0.5)
theGrid = squareRaster(swissRain,50, buffer=30*1000)
if(all(havePackages)) {
  swissFit = glgm(
    formula = rain~ 1,
    data=swissRain,
    grid=theGrid,
    family="gaussian",
    spaceFormula = ~ f(space, model='matern2d',
      nrow = nrow(theGrid), ncol = ncol(theGrid),
      nu = 1, replicate = group))

  swissFit$rasterTwo = setValues(
    swissFit$raster[[1:2]],
    as.matrix(swissFit$inla$summary.random$space[
      ncell(theGrid)+values(swissFit$raster[['space']]),
      c('mean','0.5quant')]))

  plot(swissFit$rasterTwo[['mean']])
}
```

pc prior for range, no data, check posterior

```
theGrid = squareRaster(swissRain,20, buffer=300*1000)
if(all(havePackages)) {
  swissFit = glgm(
    formula = rain~ 1,
    data=swissRain[1:3, ],
    grid=theGrid,
    family="gaussian",
    control.mode = list(theta = c(-3.7, -0.75, 2.25), restart=TRUE),
    priorCI = list(
      sd = c(u=1, alpha=0.5),
      range = c(u=400*1000, alpha=0.5)),
    verbose = TRUE
    )


  matplot(swissFit$parameters$range$posterior[,'x'],
    swissFit$parameters$range$posterior[,c('y','prior')],
    xlim = c(0, 800*1000),
    type='l'
    )
}
```