

LGCP with PC priors

Patrick Brown

September 30, 2021

```
library("geostatsp")
data('murder')
data('torontoPop')
```

```
if(requireNamespace("rgdal") & requireNamespace("rgeos") ) {
  # theCrs = mapmisc::omerc(murder, angle=-20)
  theCrs = CRS("+proj=omerc +lat_0=43.7117469868935 +lonc=-79.3789787759006 +alpha
murderT = spTransform(murder, theCrs)
borderT = spTransform(torontoBorder, projection(murderT))
borderC = crop(borderT, extent(-12700, 7000, -7500, 3100))
}

## Loading required namespace: rgdal
## Loading required namespace: rgeos

covList = list(
  pop=torontoPdents,
  inc = log(torontoIncome) )

formulaHere = ~ inc + offset(pop, log=TRUE)
```

LGCP with priors given by quantiles

```
if(requireNamespace("rgdal", quietly=TRUE) & requireNamespace("rgeos", quietly=TRUE) & r
  resG=lgcp(
    formula = formulaHere,
    data=murderT,
    grid=squareRaster(borderC, 30),
    covariates=covList,
```

```

border=borderC, buffer=2000,
prior = list(
  sd = c(lower = 0.2, upper = 2),
  range = c(lower = 2, upper=20)*1000),
control.inla=list(strategy='gaussian')
)
} else {
  resG = NULL
}

```

LGCP with penalised complexity prior

$pr(sd > 1) = 0.05$ and $pr(phi < 0.2) = 0.95$

```

if(requireNamespace("rgdal", quietly=TRUE) & requireNamespace("rgeos", quietly=TRUE) & r
  resP=lgcp(formulaHere, data=murderT,
    grid=squareRaster(borderC, 30),
    covariates=covList,
    border=borderC, buffer=2000,
    prior = list(
      sd = c(u=0.5, alpha=0.05),
      range = c(u=10*1000, alpha = 0.4)),
    control.inla = list(strategy='gaussian')
  )
} else {
  resP = NULL
}

```

LGCP with table priors

```

sdSeq = seq(0,4,len=501)
rangeSeq = seq(0,15*1000, len=501)
if(requireNamespace("rgdal", quietly=TRUE) & requireNamespace("rgeos", quietly=TRUE) & r
  resT=lgcp(formulaHere,
    data=murderT,
    grid=squareRaster(borderC, 30),
    covariates=covList,
    border=borderC, buffer=2000,
    prior = list(
      sd = cbind(sdSeq, dexp(sdSeq, 2)),

```

```

                                range = cbind(rangeSeq, dexp(rangeSeq, 1/5000))
                                control.inla = list(strategy='gaussian')
                                )
} else {
    resT = NULL
}

```

Parameters

```

if(!is.null(resG$parameters))
    knitr::kable(resG$parameters$summary[,c(1,3,5)], digits=3)

```

```

if(!is.null(resP$parameters))
    knitr::kable(resP$parameters$summary[,c(1,3,5)], digits=3)

```

```

if(!is.null(resT$parameters))
    knitr::kable(resT$parameters$summary[,c(1,3,5)], digits=3)

```

Maps

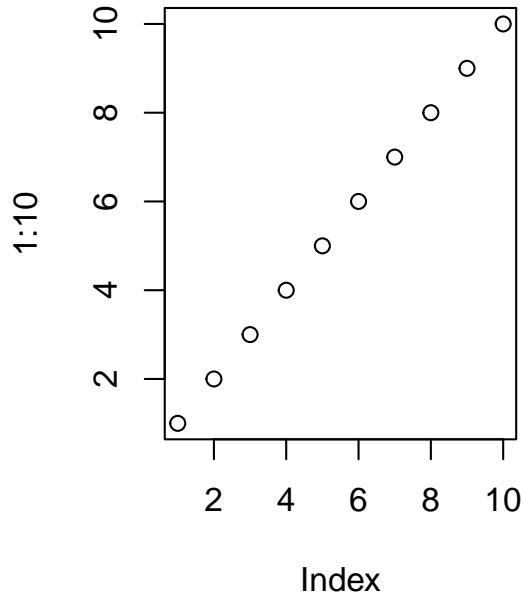


Figure 1: Priors and posteriors

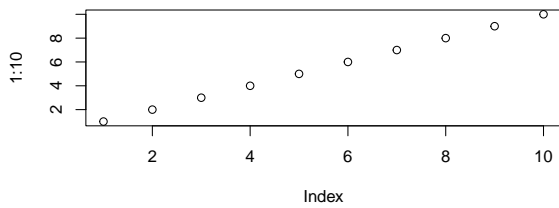


Figure 2: Random effects and fitted values