SamplingStrata: : CHEAT SHEET

To install last available release:

```
library(devtools)
install_github("barcaroli/SamplingStrata")
```

Optimal stratification

Given a sampling frame, SamplingStrata allows to optimize its stratification when designing a sampling survey, given precision constraints on target estimates.

Three different methods

The optimization can be run by indicating three different methods, on the basis of the following:

A. if stratification variables are categorical (or reduced to) then the method is the "atomic";
B. if stratification variables are continuous, then the method is the "continuous";
C. if stratification variables are continuous, and there is spatial correlation among units in the sampling frame, then the required method is the "spatial".

A. Method "atomic"

Different steps:

1. define the sampling frame;
2. set precision constraints;
3. build atomic strata;
4. run optimization;
5. perform evaluation;
6. select the sample.

Sampling frame

Data on 2896 Swiss municipalities

```
library(SamplingStrata)
data("swissmunicipalities")
swissmunicipalities$id <-
      c(1:nrow(swissmunicipalities))
frame <- buildFrameDF(
        df = swissmunicipalities,
        id = "id",
        domainvalue = "REG",
        X = c("POPTOT","HApoly"),
        Y =c("Surfacesbois", "Airind"))
```

Stratification variables

Target variables

Precision constraints

```
ndom <-
length(unique(frame$domainvalue))
cv <- as.data.frame(list(
           DOM = rep("DOM1",ndom),
           CV1 = rep(0.10,ndom),
           CV2 = rep(0.10,ndom),
           domainvalue = c(1:ndom)))
```

10% of maximum expected CV

Atomic strata

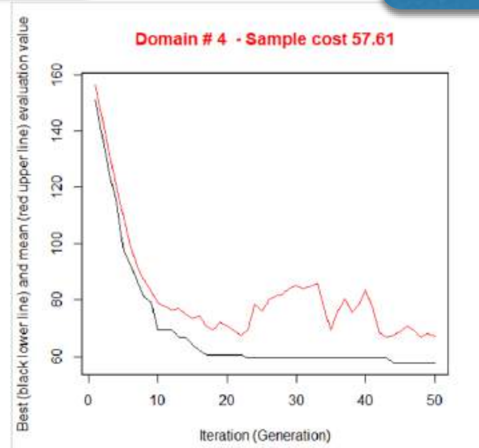
```
strata <- buildStrataDF(frame)
```

Optimization

```
solution <-
optimStrata(method="atomic",
            framesamp = frame,
            errors = cv,
            iter = 50,
            pops = 10)
```

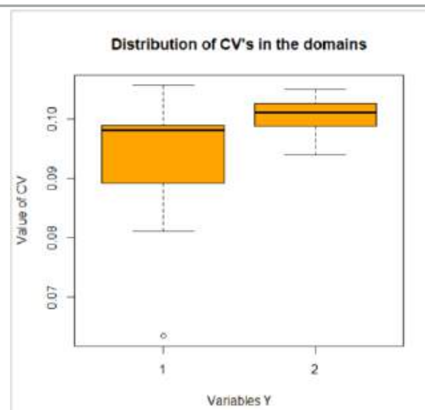
Number of iterations

Number of solutions per iteration



Evaluation

```
outstrata <- solution$aggr_strata
framenew <- solution$framenew
eval <- evalSolution(framenew,outstrata)
eval$coeff_var
```



Sample selection

```
s <- selectSample(framenew,outstrata)
head(s)
```

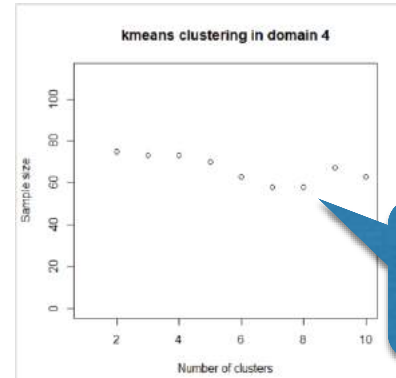
	DOMAINVALUE	STRATO	ID	X1	X2	Y1	Y2	LABEL	WEIGHTS
1	1	1	2398	241	294	101	0	1	21.38462
2	1	1	2331	267	449	215	1	1	21.38462
3	1	1	2410	237	935	471	0	1	21.38462
4	1	1	2112	370	330	98	0	1	21.38462
5	1	1	2563	173	178	16	0	1	21.38462
6	1	1	2091	382	594	338	0	1	21.38462

B. Method "continuous"

Same steps with the exception of strata building, not necessary.
Frame definition and precision constraints settings are done in the same way than in method "atomic".
One more step is in determination of the most promising number of strata with kmeans clustering.

Kmeans clustering

```
kmean <- KmeansSolution2(frame=frame,
                 errors=cv,
                 maxclusters = 10)
nstrat <- tapply(kmean$suggestions,
                 kmean$domainvalue,
                 FUN=function(x)
                     length(unique(x)))
sugg <- prepareSuggestion(
                 kmean = kmean,
                 frame = frame,
                 nstrat = nstrat)
```

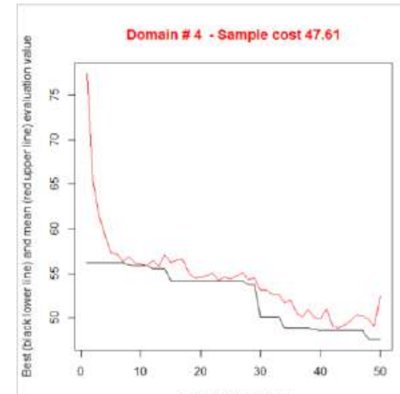


Suggested number of strata (8) for domain 4

Optimization

```
solution <- optimStrata (
                 method = "continuous",
                 framesamp = frame,
                 errors = cv,
                 nStrata = nstrat,
                 iter = 50,
                 pops = 10,
                 suggestions = sugg)
```

Suggestion prepared by kmeans clustering



Evaluation

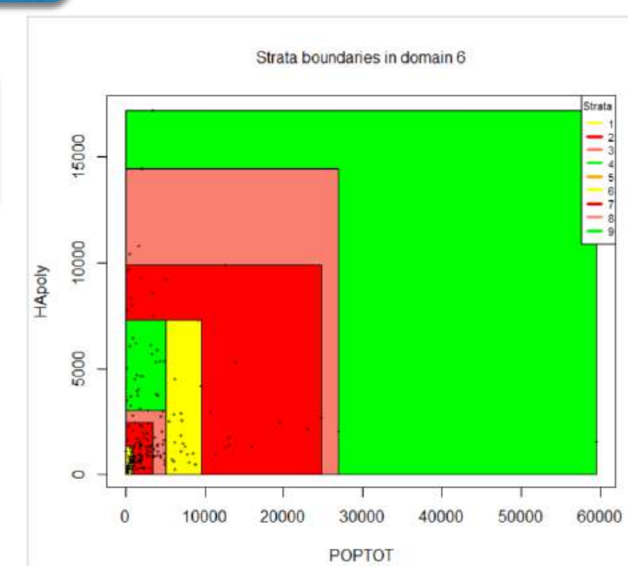
```
framenew <- solution$framenew
outstrata <- solution$aggr_strata
ss <-summaryStrata(framenew,outstrata)
head(ss)
```

	Domain	Stratum	Population	Allocation	SamplingRate
1	1	1	278	13	0.047370
2	1	2	49	6	0.113769
3	1	3	70	5	0.070377
4	1	4	69	10	0.145075
5	1	5	33	6	0.173466
6	1	6	22	9	0.424133

	Lower_X1	Upper_X1	Lower_X2	Upper_X2
1	27	780	32	986
2	65	1422	48	1018
3	95	1562	198	2288
4	78	1963	159	11907
5	1992	2711	107	8925
6	2759	3567	185	11378

```
plotStrata2d(framenew,
             outstrata,
             domain = 6,
             vars = c("X1","X2"),
             labels = c("POPTOT", "HApoly"))
```

Visualization of strata by couples of X's




```
eval <-
evalSolution(framenew,outstrata)
eval$coeff_var
```

Sample selection

```
s <- selectSample(framenew,outstrata)
head(s)
```

	DOMAINVALUE	STRATO	ID	X1	X2	Y1	Y2	LABEL	WEIGHTS
1	1	1	2398	241	294	101	0	1	21.38462
2	1	1	2331	267	449	215	1	1	21.38462
3	1	1	2410	237	935	471	0	1	21.38462
4	1	1	2112	370	330	98	0	1	21.38462
5	1	1	2563	173	178	16	0	1	21.38462
6	1	1	2091	382	594	338	0	1	21.38462

C. Method "spatial"

In cases where units in the sampling frame are geo-referenced and there is spatial correlation among them, it is possible to apply the "spatial" method in the optimization of the frame stratification.

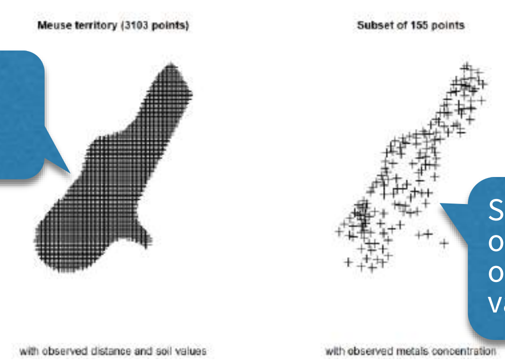
Different steps:

1. perform a preliminary spatial analysis and fit spatial models on target variables
2. define the sampling frame and add predicted values, prediction errors and coordinates;
3. set precision constraints;
4. run optimization;
5. select the sample.

Spatial analysis

We make use of the «Meuse river» datasets, reporting measures of 4 metals concentration.

```
library(sp)
# locations (155 observed points)
data("meuse")
# grid of points (3103)
data("meuse.grid")
meuse.grid$id <- c(1:nrow(meuse.grid))
coordinates(meuse)<-c('x','y')
coordinates(meuse.grid)<-c('x','y')
```

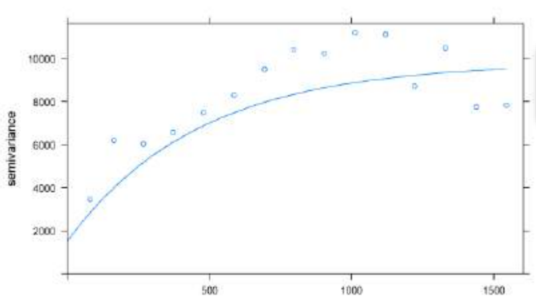


Grid of Meuse river

Sample of observed values

```
library(gstat)
library(automap)
v <- variogram(lead~dist+soil,data=meuse)
fit.vgm.lead <- autofitVariogram(
       lead ~dist+soil,meuse,model="Exp")
plot(v, fit.vgm.lead$var_model)
```

Analysis and fitting



prediction

```
lead.kr <- krige(lead~dist+soil,
          meuse, meuse.grid,
          model=fit.vgm.lead$var_model)
lead.pred <- ifelse(lead.kr[1]$var1.pred<0,
                    0,lead.kr[1]$var1.pred)
lead.var <- ifelse(lead.kr[2]$var1.var < 0,
                    0,lead.kr[2]$var1.var)
```

Sampling frame

```
df <- as.data.frame(list(
       dom=rep(1,nrow(meuse.grid)),
       lead.pred=lead.pred,
       lead.var=lead.var,
       lon=meuse.grid$x,
       lat=meuse.grid$y,
       id=c(1:nrow(meuse.grid))))
frame <- buildFrameSpatial(df=df,
           id="id",
           X=c("lead.pred"),
           Y=c("lead.pred"),
           variance=c ("lead.var"),
           lon="lon",
           lat="lat",
           domainvalue = "dom")
```

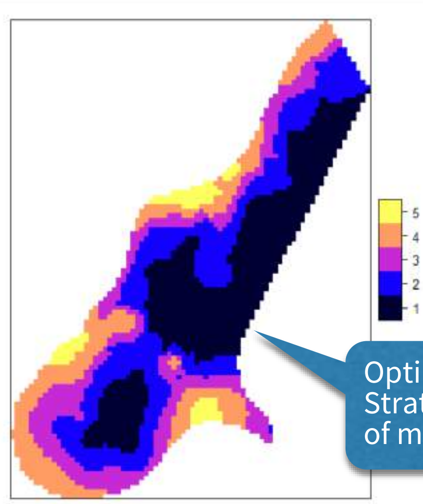
Precision constraints

```
cv2 <- as.data.frame(list(
        DOM=rep("DOM1",1),
        CV1=rep(0.05,1),
        domainvalue=c(1:1) ))
```

Optimization

```
solution <- optimStrata(method="spatial",
   errors=cv2, framesamp=frame, iter=25,
   nStrata=5, fitting=1, kappa=1,
   range=fit.vgm.lead$var_model$range[2])
```

```
framenew <- solution$framenew
outstrata <- solution$aggr_strata
frameres <- SpatialPixelsDataFrame(
     points=framenew[c("LON","LAT")],
     data=framenew)
frameres$LABEL <-
     as.factor(frameres$LABEL)
spplot(frameres,c("LABEL"),
       col.regions=bpy.colors(5))
```



Optimal Stratification of meuse.grid

Use of models

Usually, values of target variables are not available in sampling frames, but only of co-variates. In order to calculate correctly the variance of target variables in strata, we can make use of models. When applying methods 'atomic' and 'continuous', it possible to declare linear or log-linear models linking each target variable to one co-variate available in the sampling frame.

Consider the case with 'swissmunicipalities' dataset. Suppose that for all units we only have values for POPTOT and HApoly, while only on a subset (500) of it the values for Surfacesbois and Airbat are also available.
We fit the following models:

```
k <- sample(c(1:2896),500)
s <- swissmunicipalities[k,]
Airind_POPTOT <-
   lm(Airind~POPTOT, data=s)
Bois_HApoly <-
   lm(Surfacesbois~HApoly,data=s)
```

For both models we calculate heteroscedasticity indexes and variance:

```
airind <-
computeGamma(Airind_POPTOT$residuals,
                 s$POPTOT,nbins = 14)
airind
# gamma      sigma   r.square
# 0.59235109 0.06794055 0.87070106
bois <-
computeGamma(Bois_HApoly$residuals,
                 s$HApoly,nbins = 14)
bois
# gamma      sigma   r.square
# 0.8547931 0.4483606 0.9732122 )
```

We can now instantiate the values in the 'model' dataframe:

```
model <- NULL
model$beta[1] <-
      Airind_POPTOT$coefficients[2]
model$sig2[1] <- airind[2]^2
model$type[1] <- "linear"
model$gamma[1] <- airind[1]
model$beta[2] <-
      Bois_HApoly$coefficients[2]
model$sig2[2] <- bois[2]^2
model$type[2] <- "linear"
model$gamma[2] <- bois[1]
model <- as.data.frame(model)
model
# beta       sig2      type   gamma
# 0.01109583 0.1708807 linear 0.4703953
# 0.26068155 0.2010272 linear 0.8547931
```

Sampling frame

```
swissmunicipalities$dom <- 1
frame <- buildFrameDF(
                 df=swissmunicipalities,
                 id="id",
                 X=c("POPTOT","HApoly"),
                 Y=c("POPTOT","HApoly"),
                 domainvalue = "dom")

frame$airind <-
  swissmunicipalities$Airind
frame$surfacesbois <-
  swissmunicipalities$Surfacesbois
```

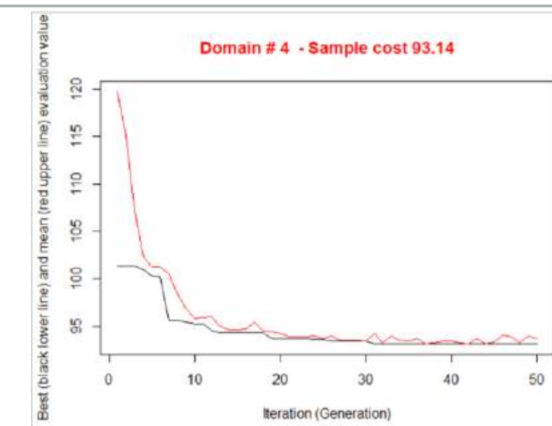
Co-variates as both X's and Y's

Optimization

With the same precision constraints of 10% for both target variables we run the optimization step:

```
solution <-
  optimStrata(
    method = "continuous",
    errors = cv,
    framesamp = frame,
    model = model,
    nStrata = 5,
    iter = 50,
    pops = 10)
```

'model' dataframe previously defined



Evaluation

```
framenew <- solution$framenew
outstrata <- solution$aggr_strata
framenew$Y3 <- framenew$AIRIND
framenew$Y4 <- framenew$SURFACESBOIS
val <- evalSolution(framenew,outstrata)
val$coeff_var
# CV1    CV2    CV3    CV4   dom
# 0.0107 0.0706 0.0316 0.0603 DOM1
# 0.0073 0.0364 0.0220 0.0426 DOM2
# 0.0062 0.0252 0.0253 0.0332 DOM3
# 0.0071 0.0328 0.0303 0.0572 DOM4
# 0.0055 0.0646 0.0171 0.0541 DOM5
# 0.0037 0.0745 0.0173 0.0606 DOM6
# 0.0036 0.0753 0.0145 0.0541 DOM7
```

Notice that both the CV's of the co-variates (CV1 and CV2) andthe CV's of the real target variables (CV3 and CV4) are compliant to the 10% precision constraints.

 Giulio Barcaroli • gbarcaroli@gmail.com