SelEdit. . . – a collection of R packages to implement some optimization-based selective editing techniques

Use of R in Official Statistics 2018

D. Salgado, M.E. Esteban, S. Saldaña

Statistics Spain (INE)

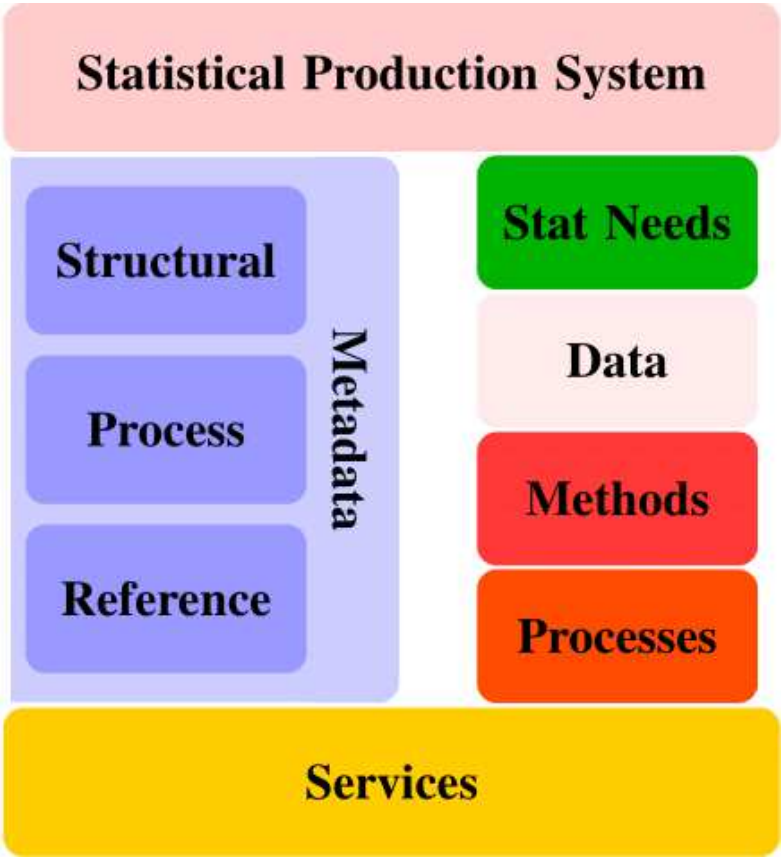
CBS, The Hague, 12-14 September, 2018

Overview

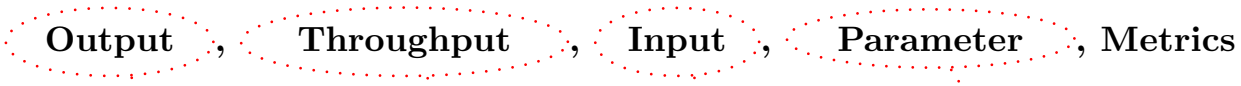
INSTITUTO NACIONAL DE ESTADISTICA

1. The approach
2. Methodology Layer: optimization-based selective editing technique
3. OOP/Functional Layer: classes and methods (R packages)
4. Scripting Layer: scripts
5. Interactive Layer

The approach



The approach



User interface layer:

Dashboard

Scripting layer:

newDataObject = Action(DataObject, ParameterObject)

OO+Functional Programming layer:

DataObject $\rightsquigarrow$ Standardised data model class

ParameterObject $\rightsquigarrow$ Standardised parameter class

Action $\rightsquigarrow$ Statistical method implementation

Statistical method layer:

$$\hat{Y} = f(s, \mathbf{y})$$

The methodology

- Editing must minimize the amount of resources deployed to recontacts, follow-ups and interactive tasks, in general.
- Data quality must be ensured.

$$[P_0] \quad \max \ \mathbb{E}_m \left[\mathbf{1}^T \mathbf{R} \middle| \mathbf{Z} \right]$$

$$\text{s.t.} \quad \mathbb{E}_m \left[L \left(\widehat{Y}^{(*,q)}(\mathbf{r}), Y^{(0,q)} \right) \middle| \mathbf{Z} \right] \leq \eta_q, \qquad q = 1, \ldots, Q$$

$$\mathbf{R}(\omega) \in \Omega_0 \subset \{0,1\}^{\times n}$$

$$[P_{co}(\mathbf{M}, \boldsymbol{\eta}, \Omega_0)] \quad \max \ \mathbf{1}^T \mathbf{r}$$

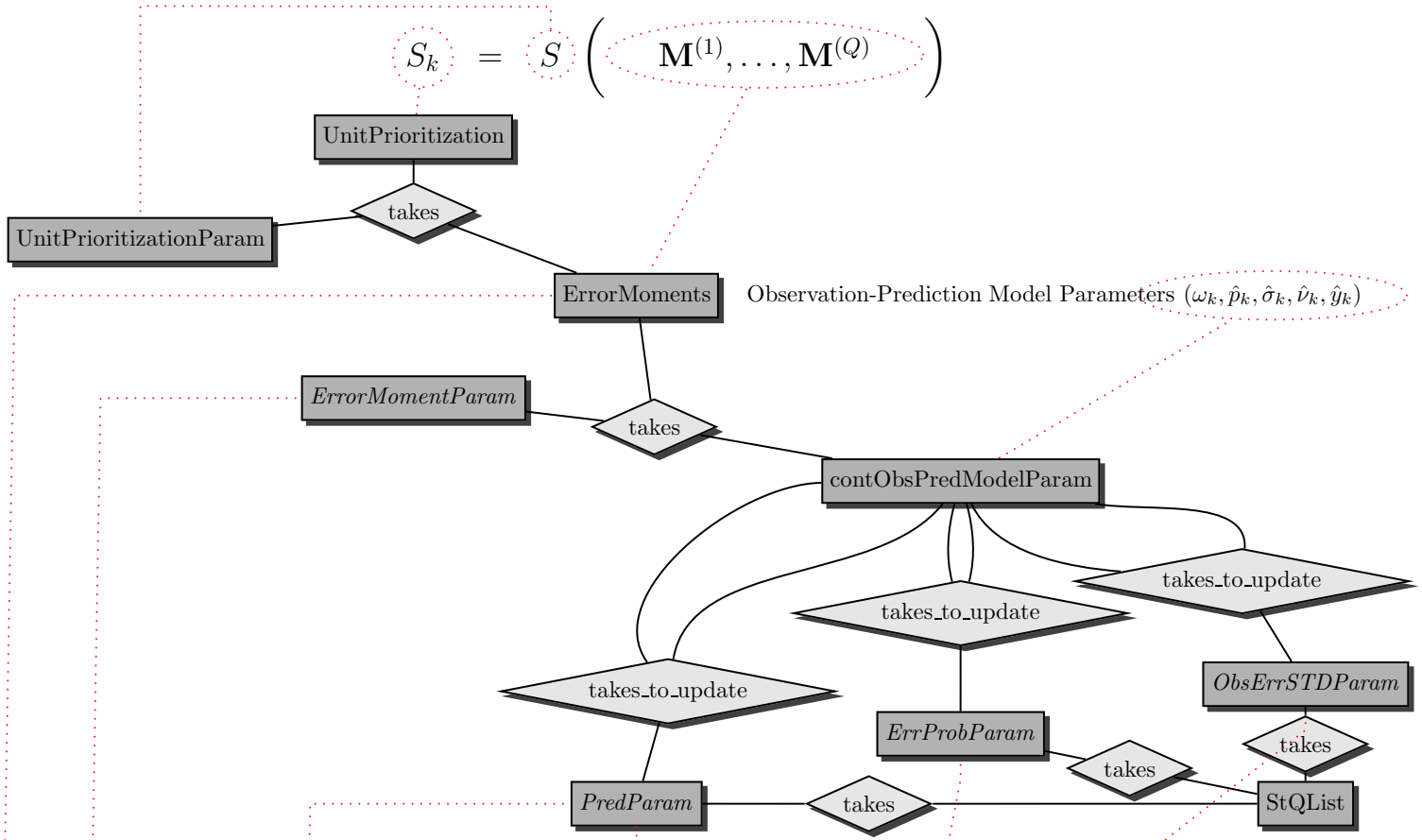
$$\text{s.t.} \quad \mathbf{r}^T M^{(q)} \mathbf{r} \leq \eta_q, \qquad q = 1, \ldots, Q$$

$$\mathbf{r} \in \Omega_0 \subset \{0,1\}^{\times n}$$

$$M_{kk} = \sqrt{\tfrac{2}{\pi}} \cdot \omega_k \cdot \nu_k \cdot {}_1F_1 \left(-\tfrac{1}{2}; \tfrac{1}{2}; -\tfrac{(y_k - \hat{y}_k)^2}{2\nu_k^2} \right) \cdot \zeta_k \left(\tfrac{y_k - \hat{y}_k}{\nu_k} \right)$$

$$S_k = S(M_{kk}^{(1)}, \ldots, M_{kk}^{(Q)})$$

OOP and functional layers



The packages

- To manage the internal data repository:
  - `StQ`
  - `RepoTime`
  - `RepoUtils`
  - `RepoReadWrite`
- To implement the selective editing technique:
  - `contObsPredModelParam`
  - `SelEditErrorMoment`
  - `SelEditUnitPriorit`
  - `SelEditUnitAllocation`
  - `SelEditFunctions`
- Statistical infrastructure:
  - `StQPrediction`
  - `StQImputation`
  - `TSPred`
  - `BestTSPred`

Scripting layer: setting up the process. I

```
# The data dictionary
DD <- getDD(raw)

# Qualifiers identifying statistical units
IDQual <- getIDQual(DD, 'MicroData')

## Variables under analysis (with variable names according to repo)
# 'b1': turnover
# 'c11', 'c121', 'c122': number of different types of employees
VarAnalysis <- UnitToIDDDNames(c('b1', 'c11', 'c121', 'c122'), DD)

# Variables for the domains ('rama': code for economic activity)
VarDomains <- UnitToIDDDNames('rama', DD)


# Missing values imputed using the mean or median by domain
MeanImpParam <- new(Class = 'MeanImputationParam',
                    VarNames = VarAnalysis,
                    DomainNames = VarDomains)
MedianImpParam <- new(Class = 'MedianImputationParam',
                      VarNames = VarAnalysis,
                      DomainNames = VarDomains)

# Observation error STD by MLE
ObsErrVarMLEParam <- new(Class = 'ObsErrorSTDMLEParam',
                         RawData = raw.StQList,
                         EdData = edited.StQList,
                         VarNames = VarAnalysis,
                         Imputation = MeanImpParam)
```

Scripting layer: setting up the process. II

```
# Measurement error probability by MLE
ErrorProbMLEParam <- new(Class = 'ErrorProbMLEParam',
                         RawData = raw.StQList,
                         EdData = edited.StQList,
                         VarNames = VarAnalysis,
                         Imputation = MeanImpParam)

# Prediction by random walk under monthly frequency (function RegDiffTSPred)
TS.list <- list(Reg = list('RegDiffTSPred', frequency = 12, forward = 1))
BestTSPredParam <- new(Class = 'BestTSPredParam',
                       TSPred.list = TS.list,
                       VarNames = VarAnalysis)
PredTSParam <- new(Class = 'PredTSParam',
                   TS = edited.StQList,
                   Param = BestTSPredParam)
PredValueTSParam <- new(Class = 'PredValueTSParam',
                        PredictionParam = PredTSParam,
                        ImputationParam = MeanImpParam)

# Design weights ('factor') taken from edited values
edited <- readRDS(file.path(path, 'FF.rds'))
weights.DT <- getValues(edited, UnitToIDDDNames('factor', DD))
setnames(weights.DT, wName, VarAnalysis[1])
weights.DT[, (VarAnalysis[-1]) := get(VarAnalysis[1])]

# Loss function (and some other details)
AbsLossPar <- new(Class = 'AbsLossErrorMomentParam',
                  VarNames = VarAnalysis,
                  Homoskedastic = rep(FALSE, length(VarAnalysis)),
                  Imputation = MedianImpParam)
```

Scripting layer: computation

```
# Unit (global) score function
UnitPriorParam <- new(Class = 'UnitPrioritizationParam',
                      UnitScFunction = 'MinkUnitSc',
                      ScFunctionParam =  list(alpha = 2, Weights = 1))
```

```
## Compute the measurement error model parameters
# Create an empty object for the error model parameters
ObsPredPar <- new(Class = 'contObsPredModelParam',
                  Data = raw,
                  VarRoles = list(Units = IDQual,
                                  ObjVariables = VarAnalysis,
                                  Domains = VarDomains))
# Compute observation error STD
ObsPredPar <- ComputeObsErrorSTD(ObsPredPar, ObsErrVarMLEParam)
# Compute error probability
ObsPredPar <- ComputeErrorProb(ObsPredPar, ErrorProbMLEParam)
# Compute predictions and prediction error STD
ObsPredPar <- ComputePred(ObsPredPar, PredValueTSParam)
# Include design weights
setDesignW(ObsPredPar) <- weights.DT

# Compute the measurement error moments (local scores)
ErrorMoments <- ComputeErrorMoment(ObsPredPar, AbsLossPar)

# Prioritse units
PriorUnits <- PrioritizeUnits(ErrorMoments, UnitPriorParam)
```

Interactive layer

- Strongly based upon the scripting layer
- Several options under consideration:
  - SAS macros executing R scripts in batch
  - Shell scripts based on R scripts
  - Dashboards using Shiny
    ($\rightsquigarrow$ turn R scripts into higher-order functions?)