GRridge: Adaptive group-regularized ridge regression by use of co-data. Course version.

Mark A. van de Wiel and Putri W. Novianti

April 10, 2018

Department of Epidemiology & Biostatistics
VU University Medical Center
Amsterdam, The Netherlands

mark.vdwiel@vumc.nl

PREAMBLE FOR THE COURSE. This document is based on the `GRridge` vignette. We made some updates and added more explanation when needed. In addition, we added *Exercises in italics*. The solutions will be provided later on.

Contents

1 Overview

Predicting binary or continuous response from high-dimensional data is a well-addressed problem nowadays. Many existing methods have been adapted to cope with high-dimensional data, in particular by means of regularization. Adaptive group regularized ridge regression was introduced to improve the predictive performance of logistic ridge regression by incorporating external and/or internal auxiliary information on the features: the co-data. More formally, co-data can be described as any nominal or quantitative feature-specific information, obtained independently of the response variable. Three types of co-data are distinguished:

1. Response-independent summaries in the primary data (e.g. standard deviation).
2. Feature-specific summaries from an independent study (e.g. p-values).
3. Feature groupings from public data bases (e.g. pathways).

The `GRridge` package implements adaptive group-regularized (survival, linear, logistic) ridge regression by use of co-data. The package includes convenience functions to convert such co-data to the correct input format. In addition, it includes functions for evaluating the predictive performance.

`GRridge` package is based on these following publications:

Mark van de Wiel, Tonje Lien, Wina Verlaat, Wessel van Wieringen, Saskia Wilting. (2016). Better prediction by use of co-data: adaptive group-regularized ridge regression. Statistics in Medicine, 35(3), 368-81. [Wiel et al., 2016]

Putri W. Novianti, Barbara C. Snoek, Saskia Wilting, Mark van de Wiel. (2017). Better diagnostic signatures from RNAseq data through use of auxiliary co-data. Bioinformatics. doi: 10.1093/bioinformatics/btw837. [Novianti et al., 2017]

1.1 Elements of the `GRridge` package

Key elements of the GRridge package are:

1. An automatic function to create a partition of features, namely `CreatePartition` and `matchGeneSets` function for non-overlapping and overlapping groups, respectively. The package also provides a function to regroup a considerably number of overlapping groups by `mergeGroups` function.
2. Comparison of the performance of GRridge model with, ordinary ridge regression, lasso and non-penalized regression, using the `grridge` function:
    - ridge regression: the function automatically estimates the ordinary ridge regression model.

- lasso: set `comparelasso=TRUE`.
- non-penalized regression: set `compareunpenal=TRUE`.

3. The `grridgeCV` function provides predicted classes and predicted probabilities that are estimated by cross-validation.

4. Post-hoc feature selection of X-relevant features. Features selection via an additional L1-penalty("`selectionEN=TRUE`", in the `grridge` function). It uses the argument "`maxsel=X`" in the `grridge` function to set the maximum number of selected features.

5. Evaluation of the performance of classification models is visualized by receiver operating characteristics (ROC) curve ("`roc`" function) and is quantified by area under the curve ("`auc`" function).

1.2 Getting started

The `GRridge` package is freely available online. We show `GRridge` installation from github

```
library(devtools)
install_github("markvdwiel/GRridge")
```

OR from Bioconductor:

```
install.packages("GRridge",
                 repos="http://bioconductor.org/packages/devel/bioc/")
```

The GRridge package depends on these following R packages: `penalized` [Goeman, 2010], `survival` (Therneau, 2015) and `Iso` (Turner, 2015). Check that you are using version (>=1.3.4), e.g. by `sessionInfo()`.

You may want to clean your memory first:

```
rm(list = ls())
```

Desired working directory:

```
setwd("C:/Synchr/Onderwijs/CodataCourse/")
```

2 Example 1: DNA-Methylation data

A cervical cancer study measures DNA methylation level on normal healthy controls (control, n=20) and high-grade precursor lesions (precursor, n=17) tissue biopsies [Farkas et al., 2013]. A popular platform for measuring methylation is the Infinium HumanMethylation450 BeadChip (Illumina,San Diego, CA, USA), which contains 450,000 probes per individual, where each probe renders a so-called beta-value. The preprocessing process rendered 40,000 methylation probes [Wiel et al., 2016].

Load the GRridge library and its dependencies

```
library(GRridge)
```

Load the primary data set

```
data(dataFarkas)
```

It contains these following objects

- datcenFarkas: methylation data for cervix samples (arcsine-transformed beta values).
- respFarkas: binary response (Normal and Precursor).
- CpGannFarkas: annotation of probes according to location (CpG-Island, North-Shelf, South-Shelf, North- Shore, South-Shore, Distant).

We first create a partition based on location of the probes (`CpGannFarkas`). For nominal input (factor), `CreatePartition(vec)` creates a partition of features (probes) with groups according to the levels of `vec`.

```
firstPartition <- CreatePartition(CpGannFarkas)
```

```
[1] "Summary of group sizes:"
Distant  Island N_Shelf N_Shore S_Shelf S_Shore
  14047   12858    2006    5262    1765    4062
```

EXERCISES: Methylation is thought to be most relevant on CpG-islands and much less on Distant probes. So which group do you expect to receive the largest/smallest penalty? Why do the sizes of the group matter, in terms of the impact of the penalty weights?

A practical issue when applying penalized regression is the need or 'no need' for standardization of the features. A potential of GRridge method is that it can let the data decide how the variance of features should impact the penalties. More discussion about this issue can be found in [Wiel et al., 2016]. A partition based on standard deviation (sd) of each feature is then created. For numeric

input, it creates a partition according to ranking, here into uniformly-sized groups based on sds. The argument `decreasing=FALSE` implies here that groups of probes with smaller sds may potentially be penalized less when using the monotone argument below in the `grridge` function (which implicitly also happens when standardizing the data; however this solution is more non-parametric).

```
sdsF <- apply(datcenFarkas,1,sd)
secondPartition <- CreatePartition(sdsF,decreasing=FALSE,
                                   uniform=TRUE,ngroup=4)
```

```
[1] "Group size set to: 10000"
[1] "Sorting vec in increasing order, assuming small values are MORE relevant"
[1] "Summary of group sizes:"
group1 group2 group3 group4
 10000  10000  10000  10000
```

Concatenate two partitions into one list

```
partitionsFarkas <- list(cpg=firstPartition,sds=secondPartition)
```

A list of monotone functions from the corresponding partition.

```
monotoneFarkas <- c(FALSE,TRUE)
```

`monotoneFarkas` indicates that monotone increasing group-penalties are desired for the 2nd partition (sd-based), and not for the first one.

EXERCISE: Why could it be useful to impose monotony?

`grridge()` function applies group-regularized ridge to data `datcenFarkas`, response `respFarkas` and probe grouping `partitionFarkas`. It recognizes automatically whether survival, linear or logistic (here) regression should be performed. Here, it saves the prediction objects from ordinary and group-regularized ridge. Includes non-penalized intercept by default. Below, we set `standardizeX = FALSE`, because we use the sds as co-data. `grridge()` starts with CV for the global lambda, followed by the empirical bayes estimation of the multipliers. We set `niter=3` to reduce computing time somewhat. Still, the fitting may take a few minutes.

EXERCISE: Why is standardization of the covariates usually recommended?

```
grFarkas <- grridge(datcenFarkas, respFarkas, partitionsFarkas,
                    optl=NULL, monotone= monotoneFarkas, niter=3, standardizeX = FALSE)
```

The printed output contains per iteration: the current cross-validated likelihood (CVL, usually negative, used to monitor convergence, see [Wiel et al.,

2016]); relative error: how much do the penalty estimates change (on average) when perturbing the input. The smaller, the less shrinkage of the penalty estimation matrix is required; Shrink Factor (>=0, <=1): how much is the penalty estimation matrix shrunken to a diagonal, see [Novianti et al., 2017]; method used for estimation (e.g. exactstable); penalty multipliers per group. After convergence of all partitions (CVL does not improve anymore): Final penalty multipliers plus computing time.

Names of the slots of the `grFarkas` output:

```
names(grFarkas)
```

The group-penalty multipliers from the GRridge model (`grFarkas`) are:

```
grFarkas$lambdamults
```

EXERCISE: Based on the GRridge model object, make a prediction for the first two patients. Use predict.grridge for this. Is this a fair prediction?

`grridgeCV()` function performs K-fold cross-validation to assess predictive performances of the predictors saved in the `grFarkas` object. It invokes `grridge()` using the same arguments as used by the above call to `grridge()` to create `grFarkas`. The result is a matrix with 3 columns containing the true response, and the predictions by ordinary and group-regularized logistic ridge. Have a coffee while you run this, because it takes a few minutes...

```
grFarkasCV <- grridgeCV(grFarkas, datcenFarkas, respFarkas, outerfold=5)
```

Print the predictions for each method that was used (default: at least ordinary ridge and grridge)

```
print(grFarkasCV)
```

EXERCISE: For how many samples does GRridge improve the predictions w.r.t. ordinary ridge?

The performance of probabilistic classifiers is visualized by ROC curves and is measured by AUC.

```
cutoffs <- rev(seq(0,1,by=0.01))
rocridgeF <- roc(probs=grFarkasCV[,2],
                 true=grFarkasCV[,1],cutoffs=cutoffs)
rocgrridgeF <- roc(probs=grFarkasCV[,3],
                   true=grFarkasCV[,1],cutoffs=cutoffs)
plot(rocridgeF[1,],rocridgeF[2,],type="l",lty=1,ann=FALSE,col="grey")
points(rocgrridgeF[1,],rocgrridgeF[2,],type="l",lty=1,col="black")
legend(0.6,0.3, legend=c("ridge","GRridge"),
       lty=c(1,1), lwd=c(1,1),col=c("grey","black"))
```

EXERCISE: Compute the AUCs for these roc-curves. How much does the co-data improve the results?

3 Example 2: mRNA sequencing data

Blood platelets extracted from patients with breast cancer (breast, n=40) and colorectal cancer (CRC, n=41) were used to profile their RNA markers for the purpose of early cancer detection [Best et al., 2015]. The raw sequencing data set is publicly available in GEO database (GEO: GSE68086). The raw data was preprocessed, rendering 18,410 transcripts (or features).

Load the GRridge library and its dependencies

```
library(GRridge)
```

Load the primary data set

```
data(dataWurdinger)
```

The object contains

- datWurdinger_BC : A matrix containing preprocessed mRNA sequencing data (quasi-gausian scale,normalized). Columns are samples (81 samples with Breast Cancer and Colorectal Cancer) and rows are features (18410 features).
- respWurdinger : A factor containing responses for samples with Breast cancer (n=40) and colorectal cancer (n=41)
- annotationWurdinger : A list containing ensembleID, geneSymbol, entrezID and chromosome location.
- coDataWurdinger : A list containing co-data sets from external sources, namely (i) a list of genes that are expressed in platelets; (ii) immunologic signature pathways and; (iii) transcription factor based pathways.

In this second example, we focus on the application of GRridge method by using multiple external co-data, namely (1) immunologic signature pathways (2) transcription factor based pathways (3) platelet expressed genes and (4) genomic annotation based on chromosomal location. The first two co-data are based on the gene set enrichment analysis (GSEA) from the Molecular Signatures Database (MSigDB). We created a list of platelet-expressed genes, which is based on the joint lists of two studies, i.e. [Gnatenko et al., 2009] and [Bugert et al., 2003]. The last partition was based on chromosomal location taken from biomaRt databases [Durinck et al., 2009].

Here, we focus on the binary classification case between breast cancer (breast) and colorectal cancer (CRC).

First, the preprocessed primary data set were transformed and standardized:

```
# Transform the data set to the square root scale
dataSqrtWurdinger <- sqrt(datWurdinger_BC)
#Standardize the features
dataStdWurdinger <- t(apply(dataSqrtWurdinger,1,
                            function(x){(x-mean(x))/sd(x)}))
```

EXERCISE: Why is the sqrt transformation useful for these mRNAseq count data?

```
# A list of gene names in the primary RNAseq data
genesWurdinger <- annotationWurdinger$geneSymbol
```

3.1 A partition containing overlapping groups

We first show an example of GRridge classification model by using overlapping groups, i.e. pathway-based grouping. Transcription factor based pathway was extracted from the MSigDB (Section C3: motif gene sets; subsection: transcription factor targets; file's name: "`c3.tft.v5.0.symbols.gmt`"). The gene sets are based on TRANSFAC version 7.5 database (http://www.gene-regulation.com/).

A partition based on the GSEA object (`TFsym`) is then created. Some features may belong to more than one group. The argument `minlen=25` implies the minimum number of features in a gene set. If `remain=TRUE`, gene sets with less than 25 members are grouped to the "`remainder`" group. "`genesWurdinger`" is an object containing gene names from the mRNA sequencing data set. See `help(matchGeneSets)` for more detail information. The `TFsym` can be downloaded from: https://github.com/markvdwiel/GRridgeCodata/tree/master/Transcription-factor-binding-site-pathway.

```
setwd("C:\\Synchr\\Onderwijs\\CodataCourse\\CoDataPlatelets")
load("TFsym.RData")
```

```
gseTF <- matchGeneSets(genesWurdinger,TFsym,minlen=25,remain=TRUE)
```

Print the indices of the data features (genes) that are member of the first TF pathway.

```
print(gseTF)[[1]]
```

The output value of the `matchGeneSets` function can be used directly as an input in the `grridge` function (`partitions=gseTF`). There is no need to create a partition via the `CreatePartition` function. A similar approach can be taken for other pathways-based partitions.

3.2 Merge groups in a partition

A pathway-based partition often contains a considerable number of gene sets (or groups). There are 604 and 4871 groups in the transcription factor and immunological based pathway, respectively (per July 4, 2016). Overfitting may be an issue in the GRridge predictive modeling on such a large number of groups.

EXERCISE: make a plot that visualizes the TF-based group sizes

As a solution, a data driven re-grouping based on hierarchical clustering analysis can be applied. The `GRridge` package provides a function to merge groups in a partition, i.e. `mergeGroups`. In this example the initial gene sets will be re-grouped into 5 groups (`maxGroups=5`) [takes a minute or 2!].

```
gseTF_newGroups <- mergeGroups(highdimdata= dataStdWurdinger,
                               initGroups=gseTF, maxGroups=5)
```

To extract indices of new groups,

```
gseTF2 <- gseTF_newGroups$newGroups
```

EXERCISE: What are the new group sizes?

The `gseTF2` object is a list of the components of which contain the indices of the features belonging to each group. This object is in the same format as the output from the "`CreatePartition`" function. Hence, the result can be used directly as an input in the "`grridge`" function.

To observe which original groups are part of which new groups:

```
newClustMembers <- gseTF_newGroups$newGroupMembers
```

3.3 Create all partitions

We now create four partitions based on theavailable co-data, described as follows:

1. co-data 1: a partition based on immunologic signature pathway. This following object was obtained by the same approach as transcription factor based pathway mentioned in the previous sections. We merged the initial gene sets (groups) into five new groups following the procedure mentioned in section 3.2.

   ```
   immunPathway <- coDataWurdinger$immunologicPathway
   parImmun <- immunPathway$newGroups
   ```

   EXERCISE: `parImmun` *is a list object the members of which contain the indices of all genes part of the corresponding pathway group. Are these groups overlapping or not?*

2. co-data 2: a partition based on transcription factor based pathway. This object is the same as the just created `gseTF2` object, described in section 3.2.

```
transcriptionFactor <- coDataWurdinger$transcriptionFactor
parTranscriptFactor <- transcriptionFactor$newGroups
```

3. co-data 3: a partition based on a list of platelets expressed genes. Here, `vec` is a character vector, e.g. a known list of genes, and `varnamesdata` contains the names of the genes in the data. Then, a partition with 2 groups is created.

```
plateletsExprGenes <- coDataWurdinger$plateletgenes
parPlateletGenes <- CreatePartition(vec=plateletsExprGenes,
                    varnamesdata=as.character(genesWurdinger))
```

4. co-data 4: a partition based on chromosomal location. A list of chromosomal location based on `biomaRt` data bases.

```
ChromosomeWur0 <- as.vector(annotationWurdinger$chromosome)
ChromosomeWur <- ChromosomeWur0
idC <- which(ChromosomeWur0=="MT" | ChromosomeWur0=="notBiomart" |
               ChromosomeWur0=="Un")
ChromosomeWur[idC] <- "notMapped"
parChromosome <- CreatePartition(as.factor(ChromosomeWur))
```

EXERCISES: Why might `Chromosome` *be a very strong co-data source, in this particular application (Breast cancer vs Colon cancer classification)? What would be an alternative, and possibly better, way of modelling this information?*

Concatenate all partitions into one list.

```
partitionsWurdinger <- list(immunPathway=parImmun,
                        transcriptionFactor=parTranscriptFactor,
                        plateletsGenes=parPlateletGenes,
                        chromosome=parChromosome)
```

A list of monotone functions from the corresponding partitions,

```
monotoneWurdinger <- c(FALSE,FALSE,FALSE,FALSE)
```

Now grridge is ready to run. It start with CV for the global lambda, followed by the empirical bayes estimation of the multipliers. As a comparison, a lasso model is also built (by setting `comparelasso=TRUE`) and post-hoc feature selection via additional L1 penalization method is performed for the `GRridge` model (by setting `selectionEN=TRUE`). The maximum number of selected markers is fixed, and context dependent (here, we use `maxsel=10`). The procedure takes a few minutes.

```
grWurdinger <- grridge(dataStdWurdinger,respWurdinger,
                        partitions=partitionsWurdinger,
                        monotone= monotoneWurdinger,
```

```
                    innfold = 3, comparelasso=TRUE,
                    optl=NULL, selectionEN=TRUE,
                    maxsel=10)
```

Names of the slots of the `grWurdinger` output:

```
names(grWurdinger)
```

EXERCISE: Check the penalty multipliers. Do these confirm your expectations, in particular for chromosome?

`grWurdinger$resEN$whichEN` contains indexes (and the features' name) of the selected features based on the group- weighted elastic net and `grWurdinger$resEN$betaEN` has the information of the beta value of the corresponding selected feature.

EXERCISE: How many features are in the overlap of the lasso model and the GRridge+EN model?

Now we asses the performance of the GRridge model by performing 3-fold CV. Takes a few minutes. Note that K=3 is used to minimize computing time; we recommend larger folds, e.g. K=10, to obtain more accurate estimates of performance.

```
grWurdingerCV <- grridgeCV(grWurdinger, dataStdWurdinger,
                           respWurdinger, outerfold=3)
```

First 6 predictions for all models:

```
head(grWurdingerCV)
```

The performance of probabilistic classifiers is visualized by ROC curves and is measured by AUCs

```
cutoffs <- rev(seq(0,1,by=0.01))
rocridge <- roc(probs= grWurdingerCV[,2],
                true= grWurdingerCV[,1],cutoffs)
rocGRridge <- roc(probs= grWurdingerCV[,3],
                  true= grWurdingerCV[,1],cutoffs)
rocLasso <- roc(probs= grWurdingerCV[,4],
                true= grWurdingerCV[,1],cutoffs)
rocGRridgeEN <- roc(probs= grWurdingerCV[,5],
                    true= grWurdingerCV[,1],cutoffs)
plot(rocridge[1,],rocridge[2,],type="l",lty=2,ann=FALSE,col="grey")
points(rocGRridge[1,],rocGRridge[2,],type="l",lty=1,col="black")
points(rocLasso[1,],rocLasso[2,],type="l",lty=1,col="blue")
points(rocGRridgeEN[1,],rocGRridgeEN[2,],type="l",lty=1,col="green")
legend(0.6,0.35, legend=c("ridge","GRridge", "lasso","GRridge+varsel"),
       lty=c(1,1), lwd=c(1,1),col=c("grey","black","blue","green"))
```

EXERCISE: Compute the AUCs for these roc-curves. How much does the co-data improve the results?

3.4 Additional material: Partition ordering and selection

Although there is usually no harm including multiple co-data sets, more co-data does not guarantee the better predictive performance of a GRridge model. The function `PartitionsSelection` implements a procedure to optimize the use of co-data in a GRridge model. A co-data set will be included in the predictive model, if it gives a significant improvement to the model. The contribution of a co-data is evaluated by cross-validation likelihood (cvl) value. The partitions selection procedure is similar with the forward selection in a classical regression model.This selection reassures each and every set gives additive positive affect to the predictive model. It also optimizes the order in which partitions are used. `PartitionsSelection` takes considerable time, so is not illustrated here.

References

Best, M. G., Sol, N., Kooi, I., Tannous, J., Westerman, B. A., Rustenburg, F., Schellen, P., Verschueren, H., Post, E., Koster, J., et al. (2015). Rna-seq of tumor-educated platelets enables blood-based pan-cancer, multiclass, and molecular pathway cancer diagnostics. *Cancer Cell*, 28(5):666–676.

Bugert, P., Dugrillon, A., Günaydin, A., Eichler, H., and Klüter, H. (2003). Messenger rna profiling of human platelets by microarray hybridization. *Thrombosis and haemostasis*, 90(4):738–748.

Durinck, S., Spellman, P. T., Birney, E., and Huber, W. (2009). Mapping identifiers for the integration of genomic datasets with the r/bioconductor package biomart. *Nature protocols*, 4(8):1184–1191.

Farkas, S. A., Milutin-Gašperov, N., Grce, M., and Nilsson, T. K. (2013). Genome-wide dna methylation assay reveals novel candidate biomarker genes in cervical cancer. *Epigenetics*, 8(11):1213–1225.

Gnatenko, D. V., Dunn, J. J., Schwedes, J., and Bahou, W. F. (2009). Transcript profiling of human platelets using microarray and serial analysis of gene expression (sage). *DNA and RNA Profiling in Human Blood: Methods and Protocols*, pages 245–272.

Goeman, J. J. (2010). L1 penalized estimation in the cox proportional hazards model. *Biometrical journal*, 52(1):70–84.

Novianti, P., Snoek, B., Wilting, S., and van de Wiel, M. (2017). Better diagnostic signatures from rnaseq data through use of auxiliary co-data. *Bioinformatics*.

Wiel, M. A., Lien, T. G., Verlaat, W., Wieringen, W. N., and Wilting, S. M. (2016). Better prediction by use of co-data: adaptive group-regularized ridge regression. *Statistics in Medicine*, 35(3):368–381.