Semi-LASSO : un weighted LASSO pour l'intégration de régresseurs connus dans un modèle linéaire

Anouk Rago [1] & Nicolas Champagnat [2] & Anne Gégout-Petit [3]

[1] *Université de Lorraine, CNRS, Inria, IECL, France, anouk.rago@univ-lorraine.fr*
[2] *Université de Lorraine, CNRS, Inria, IECL, France, nicolas.champagnat@inria.fr*
[3] *Université de Lorraine, CNRS, Inria, IECL, France, anne.gegout-petit@univ-lorraine.fr*

Résumé. Le LASSO est une technique très largement utilisée lorqu'il s'agit à la fois d'estimer les paramètres d'un modèle et d'effectuer une sélection de variables. Il est particulièrement utile pour étudier de grands jeux de données, comme cela peut être le cas en biologie des systèmes par exemple, ce qui le rend très utilisé dans le domaine de l'inférence de réseaux de gènes. Cette méthode peut par ailleurs être enrichie et améliorée par des connaissances préalables sur les régresseurs potentiels, afin de guider la sélection de variables. Dans ce cas, on peut employer un weighted LASSO, dérivé du LASSO original, dans lequel l'ajout de poids spécifiques à chaque variable permet d'encoder des a priori. Le package R 'glmnet' permet à l'utilisateur de spécifier ses propres poids via un paramètre. Dans ce papier, nous introduisons une nouvelle méthode appelée semi-LASSO qui résout un cas spécifique de weighted LASSO. Son implémentation repose sur l'utilisation du package 'glmnet', mais inclut une première étape de réduction de dimension pour une meilleure optimisation de la fonction de coût du LASSO. Des simulations numériques sont effectuées sur des données synthétiques afin de comparer les résultats obtenus avec le weighted LASSO de 'glmnet' et notre méthode semi-LASSO.

Mots-clés. LASSO, régression, connaissance a priori, R, réseaux de gènes

Abstract. The LASSO is a widely used technique when it comes to perform both estimation of parameters and variable selection. It is particularly convenient when one has to deal with large datasets like genomic data, and is thus very used in the field of gene networks inference. Nevertheless, the LASSO method can be improved thanks to prior knowledge on the potential regessors of the model. In this case, a weighted LASSO is used, the weights role is to encode the prior information. The 'glmnet' package proposes to the user to tune the 'penalty.factor' parameter to specify his own weights. We introduce in this paper a new method called semi-LASSO, which solves a specific case of weighted LASSO. Even if its implementation relies on the 'glmnet' package, it first reduces the space dimension for the LASSO optimization. Numerical experiments are performed on synthetic data to compare the performances of these two methods.

Keywords. LASSO, regression, prior knowledge, R, gene network inference

1 Introduction

Le LASSO, (*Tibshirani* 1996), est une technique très largement utilisée lorqu'il s'agit à la fois d'estimer les paramètres d'un modèle et d'effectuer une sélection de variables. Il est particulièrement utile pour étudier de grands jeux de données, comme cela peut être le cas en biologie par exemple. C'est pour cette raison qu'un grand nombre de méthodes utilisant le LASSO et ses dérivés, (*Zou* 2006), ont été développées pour inférer des réseaux de régulations de gènes,(*Sanguinetti et al* 2019), qui décrivent les relations existant entre les gènes. Cependant, reconstruire ce réseau en utilisant seulement des données transcriptomiques est une tâche complexe, notamment à cause du très grand nombre de gènes en présence ($p \approx 20\,000$), relativement au faible nombre d'échantillons n recueillis : le LASSO est alors un bon outil pour sélectionner les régresseurs les plus pertinents parmi les gènes. Il n'est pas rare que des connaissances biologiques sur la structure du réseaux ou certains liens entre les protéines ou les gènes soient connus des biologistes. Lorsqu'on utilise un LASSO, une façon d'inclure ces connaissances est de spécifier différents poids pour chaque variable dans le terme de pénalisation, comme cela a été fait par exemple par *Bergensen et al* (2019). Il a de plus été montré qu'optimiser la quantité de connaissances à ajouter dans le modèle donne des résultats significativement meilleurs (*Cassan et al* 2023). D'un point de vue plus large, spécifier des pénalisations différentes peut être utile dans n'importe quel problème de régression pour lequel de l'information sur les régresseurs potentiels est disponible. Le package R 'glmnet' offre déjà une implémentation du weighted LASSO où l'utilisateur peut spécifier ses propres poids via le paramètre 'penalty.factor'.

Nous nous intéressons ici à un modèle de régression linéaire pénalisée pour lequel des connaissances a priori sont disponibles. Nous supposons que ces connaissances prennent la forme d'une certitude que certains régresseurs doivent appartenir au modèle. Dans ce cas particulier, leur pénalisation est mise à 0. La pénalisation des autres régresseurs potentiels reste inchangée et la même pour tous. Au lieu d'optimiser directement la fonction objectif du LASSO par un algorithme adapté, nous proposons tout d'abord de transformer le problème afin de diviser en deux parties la procédure d'optimisation. La première est une méthode des moindres carrés ordinaires, suivie d'une procédure LASSO classique en plus petite dimension. Ceci permet notamment d'améliorer l'estimation des paramètres ainsi que de réduire l'erreur de prédiction.
Dans la Section 2, nous présentons brièvement le LASSO et son extension. La Section 3 présente notre méthode appelée semi-LASSO et la Section 4 propose des simulations numériques afin de comparer les performances des méthodes 'glmnet' et semi-LASSO. Enfin, la Section 5 analyse les résultats obtenus.

2 LASSO et weighted LASSO

Supposons que nous avons un ensemble de p variables explicatives $X_1, \ldots, X_p \in \mathbb{R}^p$ et une variable réponse $Y \in \mathbb{R}$ telles que :

$$Y = \beta_0 + \sum_{j=1}^{p} \beta_j X_j + \epsilon$$

avec $\epsilon \sim \mathcal{N}(0, \sigma^2)$ un bruit centré et $\boldsymbol{\beta} = (\beta_0, \beta_1, ..., \beta_p) \in \mathbb{R}^{p+1}$ les coefficients du modèle, potentiellement nuls. Supposons également avoir n réalisations indépendantes de $X_1, \ldots, X_p$ et Y : on note $\mathbf{x} = (x_{ij}) \in \mathcal{M}_{n\times(p+1)}(\mathbb{R})$ la matrice des observations dont la première colonne est une colonne de 1, et $\mathbf{y} \in \mathbb{R}^n$ les observations de Y. Le problème de régression pénalisée LASSO s'écrit alors :

$$\min_{(\beta_0,\beta_1,\ldots,\beta_p)} \quad \frac{1}{2n}||\boldsymbol{y} - \boldsymbol{x}\boldsymbol{\beta}||_2^2 + \lambda \mathrm{P}(\boldsymbol{\beta}) \tag{1}$$

avec λ positif le paramètre de régularisation et $\mathrm{P}(\boldsymbol{\beta})$ le terme de pénalisation LASSO défini par :

$$\mathrm{P}(\boldsymbol{\beta}) = \sum_{j=1}^{p} |\beta_j| \tag{2}$$

Plus λ est grand, plus le nombre de coefficients β_j mis à 0 est élevé, créant ainsi un modèle parcimonieux. Pour résoudre le problème (1), une descente de gradient par coordonnées est généralement utilisée directement sur la fonction objectif à minimiser : c'est ainsi que fonctionne le package 'glmnet' de R. Dans l'Equation (2), tous les coefficients β_j sont considérés de la même manière et pénalisés de façon égale. Une extension possible du LASSO consiste donc à ajouter un poids différent pour chaque coefficient dans le terme de pénalisation. L'objectif derrière cet ajout peut être double :

— On cherche à obtenir des propriété théorique sur l'estimateur $\hat{\boldsymbol{\beta}}$: il s'agit de l'adaptive LASSO, proposé par *Zou* (2006). Dans ce cas, les poids sont liés aux données et modifiés de façon itérative dans le but d'obtenir certaines propriétés théoriques pouvant faire défaut à l'estimateur du LASSO original.
— On cherche à inclure des informations sur les régresseurs du modèle : il s'agit du weighted LASSO, utilisé par exemple par *Bergensen et al* (2011).

Dans ces cas-là, la fonction de pénalisation devient :

$$\mathrm{P}_{\boldsymbol{w}}(\boldsymbol{\beta}) = \sum_{j=1}^{p} w_j \, |\beta_j| \tag{3}$$

avec $\boldsymbol{w} = (w_1, \ldots, w_p) \in \mathbb{R}^{+p}$ le vecteur des poids. Ce nouveau problème peut être résolu avec le même algorithme de descente de gradient que pour le LASSO classique, notamment en spécifiant ses propres poids avec le paramètre 'penalty.factor' dans la fonction 'glmnet'.

3 Semi-LASSO, un weighted LASSO pour inclure de l'information a priori

3.1 Ajouter des a priori dans le modèle

Les modèles de régression avec pénalisation sont très utilisés pour l'inférence de réseaux de gènes (*Sanguinetti et al* 2019), notamment grâce à leur propension à gérer de gros jeux de données où $n \ll p$, mais également car ils peuvent sélectionner quelques gènes intéressants en tant que régresseurs, opérant ainsi une sélection de variables parmi des milliers. Néanmoins, de plus en plus d'informations concernant les interactions entre les gènes sont aujourd'hui disponibles, par le biais d'expériences biologiques réalisées en laboratoire. Celles-ci peuvent être prises en compte dans le modèle grâce au weighted LASSO décrit dans la Section 2, avec des poids correctement choisis.
Nous nous focalisons ici sur un cas particulier de weighted LASSO, où les poids sont soit 0, c'est-à-dire pas de pénalisation, soit 1, comme dans le LASSO original. Plus concrètement, un poids de 0 signifie que la variable associée à ce poids sera automatiquement incluse dans le modèle et le coefficient β_j associé sera génériquement non nul. Cela correspond donc aux variables pour lesquelles nous avons la certitude qu'elles sont liées à Y.
Les variables $\mathbf{X}$ sont ainsi divisées en 2 groupes :

- G_K —K pour "Known"— représente l'ensemble des variables pour lesquelles nous sommes certains de leur appartenance au modèle. Le poids de chaque variable dans G_K est mis à 0.
- G_U —U pour "Unknown"— représente l'ensemble des variables pour lesquelles nous n'avons aucune information a priori sur leur appartenance au modèle. Leurs poids sont laissés à 1, chacune étant pénalisée de la même manière par le paramètre λ.

Il nous apparaît raisonnable de supposer que la majorité des modélisations faites à partir de données réelles comportent un terme constant (β_0). Ainsi nous décidons de placer la variable correspondant à ce paramètre dans le groupe G_K. Nous supposons également dans ce qui suit que le nombre de variables placées dans G_K ne dépasse pas n le nombre d'observations, sans quoi la méthode que nous proposons devient inapplicable.

3.2 Formulation mathématique

En utilisant les notations introduites précédemment, le problème que nous voulons résoudre se réécrit :

$$\min_{(\beta_0,\beta_1,\dots,\beta_p)} \quad \frac{1}{2n}||\boldsymbol{y} - \boldsymbol{x}\boldsymbol{\beta}||_2^2 + \lambda \mathrm{P}_{\mathbf{1}}(\boldsymbol{\beta}) \tag{4}$$

avec

$$\begin{aligned} \mathrm{P}_{\mathbf{1}}(\boldsymbol{\beta}) &= \sum_{j=1}^{p} 1_{X_j \in G_U} |\beta_j| \\ &= \sum_{j,\, X_j \in G_U} |\beta_j| \end{aligned}$$

On note $\boldsymbol{\beta}_K$ (respectivement $\boldsymbol{\beta}_U$), le vecteur tel que $\boldsymbol{\beta}_K = (\beta_j,\, X_j \in G_K)$ (respectivement $\boldsymbol{\beta}_U = (\beta_j,\, X_j \in G_U)$). De la même manière, on note $\mathbf{x}_K$ (respectivement $\mathbf{x}_U$) la matrice contenant les n observations des variables dans G_K (respectivement G_U). L'équation (4) se réécrit alors :

$$\min_{\boldsymbol{\beta}_U} \big(\min_{\boldsymbol{\beta}_K} \big(\frac{1}{2n} ||\boldsymbol{y} - \boldsymbol{x}\boldsymbol{\beta}||_2^2 \big) + \lambda \sum_{j,\, X_j \in G_U} |\beta_j| \big) \tag{5}$$

$$= \min_{\boldsymbol{\beta}_U} \big(\frac{1}{2n} ||\mathbf{u} - \mathbf{v}\boldsymbol{\beta}_U||_2^2, + \lambda \sum_{j,\, X_j \in G_U} |\beta_j| \big) \tag{6}$$

avec $\mathbf{u}$ et $\mathbf{v}$ bien choisis. On reconnaît ici la fonction objectif du LASSO, dépendant uniquement de $\boldsymbol{\beta}_U$. La solution de l'équation (6) peut être trouvée numériquement, par exemple en utilisant la version classique du LASSO implémentée dans 'glmnet'. Une fois que $\hat{\boldsymbol{\beta}}_U$ est connu, nous pouvons en déduire $\hat{\boldsymbol{\beta}}_K$ en utilisant les valeurs obtenues pour $\hat{\boldsymbol{\beta}}_U$. Le pseudocode correspondant à cette procédure désignée dans la suite par semi-LASSO est donnée dans l'Algorithme 1.

Une solution alternative serait d'utiliser la fonction 'glmnet' directement en spécifiant des poids 0 et 1 dans l'argument 'penalty.factor'. Cependant, la méthode que nous proposons est plus efficace dans certaines situations pour les raisons suivantes.
En séparant les variables en deux groupes distincts avant d'optimiser numériquement la fonction objectif donnée Equation (4), nous avons deux problèmes de minimisation à résoudre. Le premier est un problème des moindres carrés pour une régression linéaire classique possédant une solution analytique bien connue, et le second est une régression avec pénalisation LASSO classique. Le deuxième problème est résolu numériquement, mais sur un espace de plus petite dimension, car il y a moins de variables, comparé à la méthode consistant à utiliser le weigthed LASSO de 'glmnet' directement. Nous verrons ultérieurement que le semi-LASSO permet d'obtenir une meilleure estimation des paramètres et de réduire l'erreur de prédiction.

4 Simulations numériques

4.1 Simulation des données

Afin de comparer notre méthode d'estimation de paramètres avec 'glmnet', nous effectuons des simulations numériques. Nous nous focalisons ici sur des simulations où $n = 100 \ll p = 500$. Nous avons également choisi de construire des variables $(X_1, \ldots, X_p)$ corrélées par

Algorithm 1 Semi-LASSO

Inputs :
G_K and G_U the groups of variables
$\mathbf{x} \in \mathcal{M}_{n\times p}(\mathbb{R})$ the data matrix
$\mathbf{y} \in \mathbb{R}^n$ the vector of observed values for Y

Output :
$\boldsymbol{\beta}$ the vector of coefficients

if $G_K = \emptyset$ **then** ▷ No prior information is known
$\boldsymbol{\beta}$ obtained with classical LASSO

else if $G_U = \emptyset$ **then** ▷ All regressors are known
$\boldsymbol{\beta}$ obtained with classical regression

else
compute $\mathbf{x}_K$ and $\mathbf{x}_U$ from $\mathbf{x}$
add a column of 1 in $\mathbf{x}_K$ for the intercept estimation
compute $\mathbf{u}$ and $\mathbf{v}$
solve $\min_{\boldsymbol{\beta}_U} \left(\frac{1}{2n}||\mathbf{u} - \boldsymbol{\beta}_U\mathbf{v}||_2^2, +\lambda \sum_{j,\, X_j \in G_U} |\beta_j|\right)$ with glmnet to obtain $\hat{\boldsymbol{\beta}}_U$
deduce $\hat{\boldsymbol{\beta}}_K$
end if

blocs, en nous appuyant sur le travail de thèse de *Friguet* (2010). Enfin, seules $p_u = 40$ variables parmi les 500 sont réellement utilisées pour construire Y, ce qui signifie que seules 40 coordonnées de $\boldsymbol{\beta} \in \mathbb{R}^p$ sont non nulles.
50 jeux de données sont simulés, de façon à évaluer la variabilité provenant du bruit ϵ et de l'utilisation de la cross-validation pour trouver le paramètre de pénalisation optimal λ.

4.2 Introduire de l'a priori dans le modèle

Pour voir comment l'algorithme du semi-LASSO se comporte avec différents niveaux de connaissance a priori, nous ajoutons progressivement des variables du groupe G_U au groupe G_K. Ainsi, nous testons 11 scenarii différents en commençant par le cas où $G_K = \emptyset$, aucun régresseur n'étant connu au préalable. Nous ajoutant ensuite 4 variables, correspondant à 10% des vrais régresseurs, dans G_K plusieurs fois, pour atteindre finalement $G_K = (X_{j_1}, \ldots, X_{j_{p_u}})$. Pour chaque scenario, nous estimons $\boldsymbol{\beta}$, à la fois avec le semi-LASSO et le weighted LASSO de 'glmnet'.

4.3 Critères de comparaison utilisés

Afin de comparer les résultats obtenus avec le semi-LASSO et 'glmnet', nous utilisons les mesures suivantes :

— La sensibilité, définie par se $= \frac{TP}{TP+FN} = \frac{|\{\beta_j \neq 0\} \bigcap \{\widehat{\beta_j} \neq 0\}|}{|\{\beta_j \neq 0\}|}$, avec TP les vrais positifs, autrement dit le nombre de vrais régresseurs correctement sélectionnés par la méthode, et FN les faux négatifs, autrement dit le nombre de vrais régresseurs qui n'ont pas été retrouvé par la méthode. Ce taux mesure la proportion de régresseurs trouvés par la méthode parmi tous ceux étant réellement dans le modèle.

— La spécificité, définie par sp $= \frac{TN}{TN+FP} = \frac{|\{\beta_j = 0\} \bigcap \{\widehat{\beta_j} = 0\}|}{|\{\beta_j = 0\}|}$, avec TN les vrais négatifs, c'est-à-dire le nombre de régresseurs ne faisant pas partie du modèle et qui ne sont effectivement pas sélectionnés en tant que tels par la méthode, et FP les faux positifs, c'est-à-dire les régresseurs ne faisant pas partie du modèle mais malgré tout sélectionnés par la méthode. Ce taux mesure la proportion de variables ne faisant pas partie du modèle et qui sont correctement exclues du modèle par la méthode considérée.

D'autres critères permettant d'évaluer la performance de prédiction des deux méthodes seront présentées lors de l'exposé.

5 Résultats

La Figure 1 présente la sensibilité et la spécificité obtenues pour chacune des méthodes selon le niveau de connaissances lors de l'estimation des paramètres. Chaque boxplot correspond aux 50 estimations de paramètres, une pour chaque jeu de données. Nous pouvons tout d'abord remarquer que la sensibilité augmente avec la connaissance a priori, ce qui semble cohérent avec le fait que l'on force des variables pertinentes à être incluses dans le modèle. Lorsque le niveau de connaissance a priori ne dépasse pas 80%, le semi-LASSO semble plus efficace que 'glmnet' pour trouver les vrais régresseurs. Avec cette méthode, nous enlevons tout d'abord l'influence des variables de G_K sur Y avant d'utiliser un LASSO classique sur les variables restantes de G_U. Nous pouvons raisonnablement supposer que cette étape préalable permet au LASSO d'être ensuite plus efficace pour trouver les vrais régresseurs parmi les variables restantes.
Si nous nous intéressons à la spécificité, nous pouvons observer que celle-ci est proche de 1 pour les deux méthodes considérées. La méthode 'glmnet' donne de meilleurs résultats que le semi-LASSO, en particulier quand la connaissance a priori augmente. Encore une fois, cela est probablement lié à la première étape de l'algorithme du semi-LASSO : en enlevant l'influence des variables de G_K lorsque la connaissance a priori est grande, nous forçons le LASSO à nous fournir des régresseurs parmi les variables restantes, alors même qu'il y en a de moins en moins à trouver. De plus en plus de variables non pertinentes sont donc rajoutées dans le modèle, faisant baisser la spécificité du semi-LASSO. Dans 'glmnet', l'optimisation de la fonction objectif est faite sur l'espace entier des p variables, ce qui limite le nombre de faux positifs.

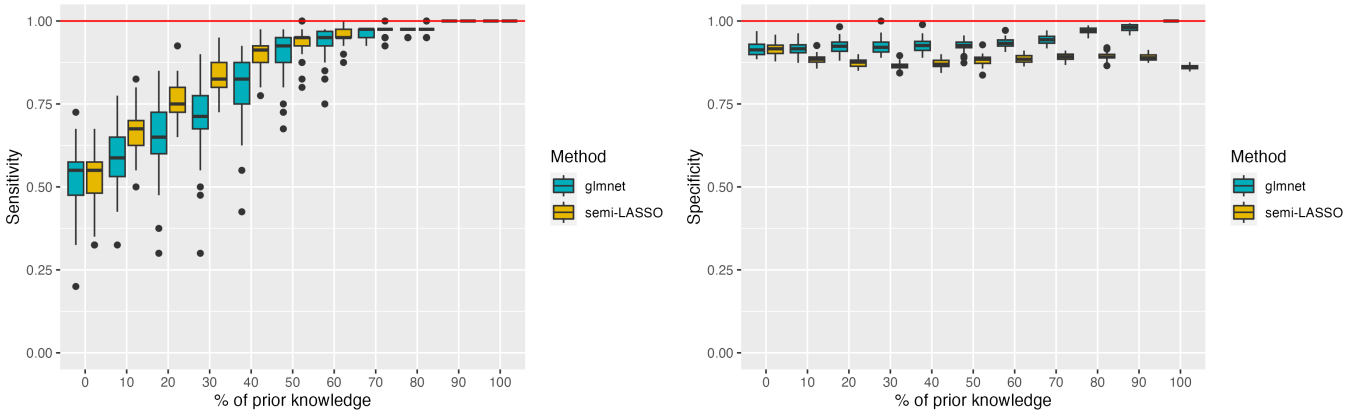

FIGURE 1 – Sensibilité et spécificité en fonction du niveau de connaissances lors de l'estimation des paramètres, pour les méthodes semi-LASSO et 'glmnet'.

6 Conclusion

Nous avons présenté une nouvelle méthode mettant en oeuvre un weighted LASSO, conçue spécifiquement pour intégrer des régresseurs connus au préalable dans le modèle. Il s'appuie sur le package R 'glmnet', mais n'utilise pas l'argument 'penalty.factor' permettant de spécifier les poids. Au lieu de cela, nous transformons d'abord le problème afin de diviser le problème d'optimisation en deux parties : une méthode des moindres carrés et un LASSO en plus petite dimension. Cela permet d'obtenir de meilleurs résultats notamment sur le nombre de vrais régresseurs détectés. Il faut malgré tout souligner que notre méthode ne peut s'appliquer que si le niveau d'a priori n'est pas trop élévé : $|G_K| \leq n$. Enfin, le semi-LASSO augmente le nombre de faux positifs par rapport à 'glmnet'.
Suivant le problème que l'utilisateur cherche à résoudre, il est laissé à sa discrétion d'utiliser le semi-LASSO ou 'glmnet' en ayant conscience des limites et avantages de chacune des deux méthodes.

Bibliographie

Sanguinetti, G. , Huynh-Thu, V. *et al* (2019), Gene Regulatory Networks : Methods and Protocols, *Springer New York*.

Tibshirani, R. (1996),Regression shrinkage and selection via the lasso, *Journal of the Royal Statistical Society Series B : Statistical Methodology*

Zou, H. (2006). The adaptive lasso and its oracle properties. *Journal of the American statistical association*, 101(476), 1418-1429.

Friedman, J., Hastie, T., Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*.

Bergersen, L. C., Glad, I. K., Lyng, H. (2011). Weighted lasso with data integration. *Statistical*

applications in genetics and molecular biology, 10(1). Cassan, O., Lecellier, C. H., Martin, A., Brehelin, L., Lèbre, S. (2023). Optimizing data integration improves Gene Regulatory Network inference in Arabidopsis thaliana. bioRxiv, 2023-09.

Friguet, C. (2010). Impact de la dépendance dans les procédures de tests multiples en grande dimension (Doctoral dissertation, Agrocampus-Ecole nationale supérieure d'agronomie de rennes).