Benchmarking the Stability of Variable Selection Methods in the Cox Model

Mathilde Sautreuil
Laboratory of Mathematics and Informatics (MICS)
CentraleSupélec, Université Paris-Saclay
91190 Gif-sur-Yvette, France
`mathilde.sautreuil@centralesupelec.fr`

Sarah Lemler
Laboratory of Mathematics and Informatics (MICS)
CentraleSupélec, Université Paris-Saclay
91190 Gif-sur-Yvette, France
`sarah.lemler@centralesupelec.fr`

Paul-Henry Cournède
Laboratory of Mathematics and Informatics (MICS)
CentraleSupélec, Université Paris-Saclay
91190 Gif-sur-Yvette, France
`paul-henry.cournede@centralesupelec.fr`

March 4, 2021

Abstract

This paper benchmarks the stability and quality of feature selection methods in the Cox model with high dimensional covariates. For this purpose, we consider different classical regularization procedures and screening methods, and we use several indexes to measure the stability and the quality in variable selection of each method. We first study the stability of these methods from simulated data and then the stability of gene selection on a real dataset. The simulation study confirmed the low stability of regularization methods and showed the selection quality was lacking. The screening methods seem to solve the stability problem in high-dimension from our study, but the selection quality is not always correct. We observe a similar behavior of these methods on the real dataset. Finally, the paper highlighted the potential of a screening method using biological knowledge.

1 Introduction

Precision medicine is often seen as the future of medicine. Its objective is to personalize treatments, diagnostic, or prognostic, according to each patient's characteristics. With the advent of high-throughput sequencing, the data enabling to characterize patients can be extremely voluminous, they provide their molecular portraits. However, it can be difficult to extract from this rich mass of data the most relevant information, the key features of interest. These key features are often referred to as "markers". For prognostic purposes, survival analysis models are extremely useful tools to predict patients relapse or death. However, the large number of covariates in view of the usual cohort size hinders model identification as well as model interpretation. In this high-dimensional setting, methods have been developed to reduce the dimension by selecting the most relevant covariates for the model. The aim of this paper is to study the stability and quality of classical feature selection methods for the Cox model.

Survival analysis is the study of the time elapsed until the occurrence of an event of interest. We will call it *death*, but it may as well be relapse or remission. Consider a simple survival regression model of the form:

$$Y_i \sim \mathbb{P}(y|\beta^T X_{i.}), \tag{1}$$

with Y_i the survival time of individual i, $X_{i.} = (X_{i1}, \dots, X_{ip})^T$ the set of variables of individual i that we suppose standardized and $\beta = (\beta_1, \dots, \beta_p)^T$ the regression parameters to be estimated. In this paper, we focus on the Cox model [1] to link the survival time to covariates. The model is written in terms of the conditional hazard and the regression parameter β of this model reflects the effect of variables (genes) on survival duration. It will be detailed in

Section 2.1. The classical procedure for estimating the parameter β consists of minimizing the opposite of the partial log-likelihood $\mathcal{L}$:

$$\widehat{\beta} = \arg\min_{\beta} \{-\mathcal{L}(\beta)\}. \tag{2}$$

Variable selection in survival analysis consists of automatically setting to zero the coefficients of variables having the least impact on survival duration, while conserving the variables with the greatest coefficients among the estimated parameters. The most classical solution is the well-known Lasso, which consists in minimizing the opposite of the partial log-likelihood $\mathcal{L}$ to which an L^1 penalty term is added:

$$\widehat{\beta} = \arg\min \{-\mathcal{L}(\beta) + \lambda||\beta||_1\}. \tag{3}$$

where $\lambda \in \mathbb{R}^+$ is a regularization hyperparameter controlling the compromise between the model fit to the data and its complexity. Many authors have been interested in regularization methods. These methods were first implemented in a linear framework [2, 3, 4, 5] and later adapted to the survival framework [6, 7, 8, 9, 10, 11]. The idea of regularization methods, such as the Lasso method and its derivatives, is to penalize the likelihood so that irrelevant variables are set to zero.

However, when the number of covariates is much larger than the number of patients, regularization procedures haw proves unstable [12, 13]. In the regularization procedures, the regularization hyperparameter λ is obtained by cross-validation, which implies a random choice of sub-samples. Thus, if we run the procedure several times and particularly in high dimension, we can observe that we do not obtain at each time the same set of selected variables. This point is critical since we are unsure if the selected variables are relevant (false positives) or if we have missed some important ones (false negatives). To address this problem, [12] have proposed screening methods. The idea is to coarsely reduce the number of variables using a score before applying finer regularisation methods on a high but smaller number of pre-selected variables. These methods of screening differ in their pre-selection procedure, we present in Section 2.3 the methods SIS [9] and ISIS [9], CoxCS [10] and PSIS [11].

This paper aims to study the regularization and screening methods to examine the quality of their selection and their stability in selection for the Cox model. To evaluate the stability of such methods, we consider similarity indices such as generalizations of the Sørensen and the Jaccard indexes [14, 15, 16]. These indices are used in ecology to measure the variability of species composition in different sites, and we propose a new index. We offer to adapt them to our framework to measure the regularization and screening methods' stability. We also introduce a new index inspired by the F-Score in classification and which is a function of the potential (and unknown) number of significant variables.

The paper is organized as follows. In Section 2, we recall some basics and notation on the Cox model and present some regularization and screening methods developed for this model. To compare the stability of the different methods, we propose to use similarity indexes presented in Section 3. We first assess the methods on simulated datasets. We present the results of these simulations in terms of selection quality in Section 4. Finally, we apply and study the stability of the different methods on a real dataset of Clear Cell Renal Carcinoma, in Section 5, and discuss some genes of interest highlighted by the selection methods.

2 Different selection methods and their application to the Cox model

2.1 Cox model in high dimension

The Cox model [1] is a classical model in the field of survival analysis predicting the survival time from covariates. It allows us to study the time elapsed until an event of interest.

The Cox model is defined for an individual i from the instantaneous risk λ which is a function of time conditional on the explanatory variables given for the individual i $X_{i.} = (X_{i1}, \dots, X_{ip})^T \in \mathbb{R}^p$:

$$\lambda(t|X_{i.}) = \alpha_0(t)\exp(\beta^T X_{i.}), \tag{4}$$

with $\alpha_0(t)$ baseline risk and $\beta = (\beta_1, ..., \beta_p)^T \in \mathbb{R}^p$ the vector of regression coefficients. The baseline risk is the instantaneous risk of death when all variables are zero. This function $\lambda(t|X_{i.})$ corresponds to the instantaneous risk of death at time t knowing that the individual i is alive before time t. We can separate the instantaneous risk into two parts because the term $\alpha_0(t)$ depends only on time and will be the same for all individuals at a given time. The second term in (4) depends only on the variables specific to each individual. The instantaneous risk between two individuals depends only on the factors to which they are subjected. This characteristic is useful when we are interested in prognosis, i.e., when we only want to know the factors influencing survival.

The Cox model [1] is semi-parametric because the estimation involves the estimation of a vector of parameters of $\mathbb{R}^p$ and a function $\alpha_0(t)$. But it is possible to estimate β without needing to know the baseline risk function $\alpha_0(t)$ thanks to

Cox's partial likelihood [1]. The partial likelihood of Cox [17] is based on the probability that an individual i dies at an observed time knowing that a death occurs. Let $t_1, \dots, t_n$ be the set of observed times ordered for n individuals and $R(t_i)$ is the set of individuals at risk at time t_i, the probability that the individual i dies knowing that an event occurs at time t_i is:

$$\frac{\exp(\beta^T X_{i.})}{\sum_{l \in R(t_i)} \exp(\beta^T X_{l.})}. \tag{5}$$

The partial likelihood of Cox [1] is part of the total likelihood that does not dependent on the baseline risk function $\alpha_0(t)$. It is written:

$$L(\beta) = \prod_{i=1}^{n} \left[\frac{\exp(\beta^T X_{i.})}{\sum_{l \in R(t_i)} \exp(\beta^T X_{l.})} \right], \tag{6}$$

with $R(t_i)$ the individuals at risk at time t_i. Maximizing $\mathcal{L}(\beta)$ allows in reasonable size to estimate correctly the parameter β of Cox's model. In high dimension (*i.e.* when the number of variables is greater than the sample size), the classical estimation procedure consisting of maximizing the Cox's partial log-likelihood no longer works. The solution, therefore, consists of minimizing the opposite of the Cox partial log-likelihood [17] to which we add a penalty term as in (3) with the partial log-likelihood L instead of the log-likelihood $\mathcal{L}$. The addition of this penalty term solves the optimization problem by encouraging a smaller and more easily interpretable model for the large dimension. Many penalties exist with different interpretability properties. We refer to [18] for more details on the penalty concept. These penalized functions are also called regularization methods, we will use this term throughout the paper, and we present in Section 2.2 two of them: the Lasso [2, 6] and the Adaptive-Lasso [3, 7]. However, these regularization methods may be unstable in variable selection [13]. Other methods have therefore appeared, called screening methods. Their general idea is to make a pre-selection before applying a regularization procedure such as a Lasso [6] for example. These methods of screening differ by the pre-selection used, we present in Section 2.3 the methods SIS [9] and ISIS [9], coxCS [10] and PSIS [11].

2.2 Regularization methods

2.2.1 The Lasso

The Lasso procedure was first introduced in the framework of a linear regression model by [2] and then in the survival analysis field by [6]. This procedure is classical in large dimensions and is the most known and most used. Its penalization is in the form of:

$$\begin{aligned} pen(\beta) &= \Gamma||\beta||_1 \\ &= \Gamma\left(\sum_{j=1}^{p} |\beta_j|\right). \end{aligned} \tag{7}$$

The Lasso estimator of the parameter β is obtained by considering the following problem:

$$\widehat{\beta}^{l_1} = \arg\min_{\beta} \left\{ -L(\beta) + \Gamma \sum_{j=1}^{p} |\beta_j| \right\}, \tag{8}$$

where $L(\beta)$ is the Cox's partial log-likelihood defined by (6). This optimization problem is convex in β and thus allows the use of convex optimization algorithms for the estimation of β. The optimization problem is thus equivalent to minimize the log partial likelihood of Cox [1] by adding a constraint of the type:

$$\sum_{j=1}^{p} |\beta_j| \leq s,$$

with $s \in \mathbb{R}^+$. This amounts to constraining β to be in a ball of standard l_1 radius s in $\mathbb{R}^p$. The obtained $\widehat{\beta}^{l_1}$ estimator is then sparse, that is to say, that a certain number of coefficients of $\widehat{\beta}^{l_1}$ are zero. This gives it interpretability in the variable selection which is our objective in this paper.

2.2.2 The Adaptive-Lasso

During the presentation of the Lasso [6] regularization method, we recalled that it is unstable in selection. Indeed, the Lasso will select them randomly among two variables strongly correlated if they both have an effect on the variable to be explained. Therefore, [3] proposed an adaptive version of the Lasso, called Adaptive-Lasso. This procedure was later extended to the Cox model by [7]. This regularization penalizes large coefficients less than smaller ones. This is achieved thanks to the penalization used, which is a weighted Lasso penalty (standard l_1):

$$pen(\beta) = \Gamma \sum_{j=1}^{p} w_j |\beta_j|, \tag{9}$$

with $w_j = \frac{1}{|\widehat{\beta}_j|^\gamma}$ where $\gamma > 0$ and the $\widehat{\beta}_j$ are the coordinates obtained by an estimator in a preliminary step. In the work presented in this paper, we have used as a preliminary estimator $\widehat{\beta}$ the Lasso estimator $\widehat{\beta}^{l_1}$. The penalty used is therefore of the form:

$$pen(\beta) = \Gamma \sum_{j=1}^{p} \frac{|\beta_j|}{|\widehat{\beta}_j^{l_1}| + \epsilon},$$

where ϵ is the minimum of the non-zero $\widehat{\beta}_j^{l_1}$. This constant ϵ is added to the denominator to avoid dividing by zero in practice. The Adaptive-Lasso [7] procedure thus corresponds to the minimization of the Cox partial log-likelihood with the addition of a weighted Lasso penalty:

$$\widehat{\beta}^{l_{ada}} = \arg\min_{\beta} \left\{ -\sum_{i=1}^{n} \left(\beta^T X_{i.}\right) - \sum_{i=1}^{n} \delta_i \log \left(\sum_{l \in R_{i.}} \exp\left(\beta^T X_{l.}\right) \right) + \Gamma \sum_{j=1}^{p} \frac{|\beta_j|}{|\widehat{\beta}_j^{l_1}| + \epsilon} \right\},$$

with $R_{i.}$ the individuals at risk at time $t_{i.}$, δ_i the censoring indicator and the regularization parameter $\Gamma \in \mathbb{R}^+$ chosen by cross-validation. The Adaptive-Lasso estimator $\widehat{\beta}^{l_{ada}}$ is better in variable selection but is more biased than Lasso. The solution consists of re-running the unpenalized classical estimation procedure with only the selected variables by adaptive-lasso to reduce the bias.

2.3 Screening methods

The regularization methods are considered unstable in selection [13]. Moreover, this instability phenomenon reinforces when the number of variables increases significantly, which is the case with molecular data. There are several screening methods whose general principle summarizes in two steps. The first one consists of reducing the number of variables by keeping only those with a score obtained from the Cox model (and specific to each method) superior to a certain threshold. The second step consists of executing a Lasso procedure to select the most significant variables among those chosen in the first step.

2.3.1 (I)SIS methods

[8] introduce the SIS and ISIS methods, and a *package* `R` has been realized in which we can find the different variants of SIS and ISIS.

SIS

For each variable, a score is calculated, called *marginal utility*, which corresponds to the estimation of the regression coefficients by maximum of the log partial likelihood of the marginal model containing only the m^e variable:

$$u_m = \arg\max_{\beta_m} \left(\sum_{i=1}^{n} (\delta_i \beta_m X_{im}) - \sum_{i=1}^{n} \delta_i \log \left(\sum_{j \in R_{i.}} \exp(\beta_m X_{jm}) \right) \right).$$

The variables are ranked according to the value of the score in descending order. Then, the first d variables with the highest score are selected and their indices will correspond to the set $\mathcal{I}$. The set of these variables constitutes the first selected model. However, we cannot know the order of magnitude of this set and by choosing a value too large for d (the choice of the d value is discussed at the end of this section), the model could contain non-important variables. Reducing the model to the size d allows us to apply the Lasso penalty on the Cox model's partial log-likelihood optimization problem:

$$\arg\max_{\beta_\mathcal{I}} \left(\sum_{i=1}^{n} \delta_i \beta_\mathcal{I}^T X_\mathcal{I} + \sum_{i=1}^{n} \delta_i \log \left(\sum \exp(\beta_\mathcal{I}^T X_\mathcal{I}) \right) - \Gamma \sum_{m \in \mathcal{I}} |\beta_m| \right),$$

where $\beta_{\mathcal{I}}$ is the vector of regression parameters that correspond to the variables whose indices belong to $\mathcal{I}$. The Lasso procedure is used to set the non-informative variables to zero. The final model will be composed of the variables whose parameters are non-zero, the set of selected variables of the model is noted $\widehat{\mathcal{M}}$ and the estimated coefficients are noted $\beta_{\widehat{\mathcal{M}}}$.

ISIS

The SIS method may not perform well when some significant variables are uncorrelated with the variable to explain. Two uncorrelation situations exist. The first one is when variables are correlated with each other but do not individually impact strongly on the variable to explain. The second is when variables are not necessarily linked with each other but will separately have a more relevant impact on the variable to explain than some significant variables. An iterative version of SIS solves this problem by comparing the selected at each step with the variables of the model chosen by SIS. ISIS, therefore, tries to make more use of the joint information of the variables. The procedure is as follows:

1. The SIS method is initially applied to all the variables of the initial model. The k_1 variables with the highest score are selected. The obtained model will be $\widehat{\mathcal{M}}_1$ of dimension $|\widehat{\mathcal{M}}_1|$ and the set of indices of the variables belonging to the model $\widehat{\mathcal{M}}_1$ is $\mathcal{I}_1$.
2. The set of indices of the unselected variables is noted $\mathcal{I}_1^C$. For each $m \in \mathcal{I}_1^C$, a new score is computed, called *conditional utility*:

$$u_{m|\widehat{\mathcal{M}}_1} = \underset{\beta_m, \beta_{\widehat{\mathcal{M}}_1}}{\arg\max} \left[\sum_{i=1}^{n} \delta_i (\beta_m X_{im} + \beta_{\widehat{\mathcal{M}}_1}^T X_{i\widehat{\mathcal{M}}_1}) - \sum_{i=1}^{n} \delta_i \left\{ \log \sum_{j \in R_i.} \exp(\beta_m X_{jm} + \beta_{\widehat{\mathcal{M}}_1}^T X_{i\widehat{\mathcal{M}}_1}) \right\} \right].$$

   We compute the score for unselected variables of the first step the score by taking into account the $\widehat{\mathcal{M}}_1$ model. We rank each variable according to this score. The k_2 variables with the highest score are selected. The set $\mathcal{I}_2$ contains the indices of the selected k_2 variables and is called relative set.
3. Next, we maximize the partial log-likelihood of the Cox model with a Lasso penalty term for the two sets of variables selected in steps 1 and 2 ($\mathcal{I}_1 \cap \mathcal{I}_2$). Variables with non-zero coefficients will be selected and will constitute the final model noted $\widehat{\mathcal{M}}_2$.
4. Finally, step 2 (with k_i) and 3 are repeated until the cardinality of the final model reaches the value d defined upstream or until $\widehat{\mathcal{M}}_i = \widehat{\mathcal{M}}_{i+1}$.

For the SIS and ISIS methods, we must define a threshold value d corresponding to the maximum number of variables selected in the model. But this d value is difficult to choose. [19] suggested in the paper accompanying their *package* R SIS to set $d = \lfloor \frac{n}{4\log(n)} \rfloor$ as part of the censored survival data. This parameter base on experiments and its choice is not justified. Other screening procedures have emerged as PSIS avoiding choosing the value of d.

2.3.2 PSIS

The PSIS method is a screening method developed by [11]. This method has similarities with the SIS method, but the score calculation is different between the two methods. The originality consists of computing a threshold to select the number of variables in the intermediate model. The choice of the threshold is justified by [11] to control false positives. The steps of the PSIS method are:

1. The regression coefficients are estimated individually for each variable by maximizing the Cox partial log-likelihood. The estimate of the regression coefficients β_j and the estimate of the variance of the estimated coefficients $\hat{\beta}$ (calculated from the inverse of the Fisher information matrix I) $I_j(\widehat{\beta}_j)^{-1}$ are retrieved.
2. The rate of false positives is fixed, we introduce: $q_n = f/p_n$, where f is the number of tolerated false positives and p_n is the dimension of the variables and thus a threshold γ is calculated as follows: $\gamma = \phi^{-1}(1 - q_n/2)$, where ϕ is the distribution function of the normal distribution.
3. The variables are then classified according to their score value. This score is calculated from the estimated regression coefficients and the variance of these coefficients: it is equal to $I_j(\widehat{\beta}_j)^{1/2}|\widehat{\beta}_j|$. Variables whose score is higher than the γ threshold are selected. These variables, therefore, belong to the intermediate model.

4. Finally, a maximum estimate of the partial log-likelihood associated with a Lasso penalty term is performed on the variables belonging to the intermediate model. Variables with non-zero coefficients are selected and correspond to the final model.

The advantage of this procedure is the property of false-positive control, i.e., it ensures that the false positive rate will be lower than the allowed rate (f/p_n). Although this method base on false-positive control, it does not take into account the false-negative rate. Controlling the latter would avoid omitting some essential variables and thus avoid obtaining an uninformative model. Finally, the non-iterative nature of PSIS leads to the same problems as those generated by SIS in the case of variables that are not correlated to the variable to explain but still influence the variable to explain through joint correlation with other variables.

2.3.3 CoxCS

The CoxCS method [10] allows adding in the screening procedure some covariates that are known to influence the variable of interest. These pre-selected covariates are then always selected by the screening procedure and can help find other relevant covariates. In practice, the CoxCS method uses biological knowledge to make the pre-selection. After the addition of biological knowledge, the procedure is the same as for PSIS.

3 Evaluation of the selection methods

Regularization methods are known as unstable in high-dimension, and screening methods try to answer this problem. In this paper, we try to quantify the stability of these methods from two similarity indexes, such as the Sørensen and Jaccard index [14, 15, 16]. Initially used in ecology to measure the variability of species composition in different sites, we propose to use them in the framework of variable selection. In the context of our study, we want to measure the variation in selection by the different methods run several times. Indeed, the choice of genes by the other methods differs for different *seeds*. A *seed* is an integer used to initialize a random number generator. However, regularization methods perform cross-validation to choose the Γ hyperparameter of the penalty criterion. Cross-validation is made by dividing the sample into k sub-samples, $k-1$ sub-samples will constitute the training set, and the last sub-sample will be the testing set). The filling of these k sub-parts is done randomly, according to the chosen *seed*.

The approach we have followed is to run the methods on 100 different *seeds*, and we have created a matrix within a row representing a *seed*, and a column representing a gene. If the gene j is selected for the i *seed*, then the coefficient (i, j) of the matrix will be worth 1 otherwise it is worth 0.

3.1 Sørensen Index

We use the Sørensen index to measure the similarity of the selected genes between the different *seeds*. The Sørensen index allows us to compare the *seeds* by considering the presence or absence of genes. It corresponds to the ratio between the overlap of selected genes by the different *seeds* and the average number of selected genes.

Let N be the number of genes selected by at least one *seed* and S the number of *seeds*. Let E_i be the set of genes selected by the *seed* i, let $|E_i| = n_i$ be its cardinal. Conversely, we note s_j the number of *seeds* for which the gene j is selected.

If there are only two *seeds*, the Sørensen index interprets in a set term as the ratio between the size of the intersection of the selected gene sets and the average size of the sets. The Sørensen index is thus given by:

$$S_2 = \frac{|E_1 \cap E_2|}{\frac{1}{2}\left(|E_1| + |E_2|\right)}$$

A gene being in $E_1 \cap E_2$ is said to belong to a collection. A gene not belonging to $E_1 \cap E_2$ is not in any collection. The belonging of a gene is generalized to a larger number of *seeds*. Therefore, the number of overlaps of selection sets to which a j gene belongs is simply $s_j - 1$. In the optimal case, the size of this overlay would be $S - 1$, and the overlay rate for the j gene is $(s_j - 1)/(S - 1)$.

Finally, the recovery measure is the sum of all genes of this recovery rate. In the case where $S = 2$, this recovery is directly the size of the intersection

$|E_1 \cap E_2| = \sum_{j=1}^{N}(s_j - 1)/(S - 1)$

Then, S_2 rewrites:

$$S_2 = \frac{\sum_{j=1}^{N}(s_j - 1)}{\frac{1}{2}\left(n_1 + n_2\right)}$$

To generalize to a larger number of *seeds*, divide the overlap measure by the average size overall selection sets:

$$S_S = \frac{\frac{1}{S-1}\sum_{j=1}^{N}(s_j-1)}{\frac{1}{S}\sum_{i=1}^{S} n_i}.$$

In the case where all the *seeds* select the same N genes, the denominator (i.e., the overlap) is worth N, which is also the average size of the sets, so the index is worth 1. Conversely, if each gene is selected only once, $s_j = 1$ for all j, and the index is worth 0.

3.2 Jaccard index

The Jaccard index also enables to compute the similarity of gene selection between the different *seeds*. The Sørensen index tends to indicate nested selection scenarios as more stable, even if the number of variables varies, whereas the Jaccard index penalizes this type of scenario. The Jaccard index divides the number of genes shared by all two seeds samples with the total number of genes present in all samples seeds:

$$J_2 = \frac{a}{a+b+c}. \tag{10}$$

Let N be the number of genes selected by at least one *seed* and S be the number of *seeds*. We now present these two indexes in multiple cases. We recall that S is the number of seeds and N is the number of genes observed in at least one site. We call E_i the set of genes observed in the seed site i, we note $|E_i| = n_i$ its cardinal. Conversely, one notes s_j the number of seeds where the gene j is present.

By considering two seeds, the Jaccard index interprets as the ratio between the intersection size of the observed gene sets and the union size on of the sets of observed genes:

$$J_2 = \frac{|E_1 \cap E_2|}{(|E_1| + |E_2| - |E_1 \cap E_2|)}.$$

If a gene belongs to $E_1 \cap E_2$, it is said to belong to recovery. If it does not belong to $E_1 \cap E_2$, it does not belong to any recovery. The gene belonging generalizes to a larger number of seeds. Therefore, the number of overlaps of observation sets to which a genes j belongs is simply $s_j - 1$. In the optimal case, the size of this recovery would be $S - 1$, and it is called the recovery rate for the gene j: $(s_j - 1)/(S - 1)$. Finally, the recovery measure is the sum of all cash of this recovery rate. In the case where $S = 2$, this recovery is directly the size of the intersection

$|E_1 \cap E_2| = \sum_{j=1}^{N}(s_j - 1)/(S - 1)$.

and J_2 is rewritten:

$$J_2 = \frac{\sum_{j=1}^{N}(s_j-1)}{N}.$$

By generalizing to a larger number of seeds, the overlap measure divides by the union size of the sets for the Jaccard index:

$$J_S = \frac{\frac{1}{S-1}\sum_{j=1}^{N}(s_j-1)}{N}.$$

If we observe the same number N at all sites, the denominator (i.e., the recovery) is worth N, and this is also the average size of the sets, so the index is worth 1. Conversely, if we observe each species only once, $s_j = 1$ for all j, and the index is worth 0.

3.3 F_{score} metrics

The Sørensen and Jaccard indexes enable checking the stability of the regularization and screening methods. But the values of these indexes depend on the number of selected covariates. We are interested in two other metrics: F_{score} and $F_{score}(n^\star)$. The F_{score}, detailed in Section 3.3.1, enables to check the quality of the selection by combining the recall and the precision. If the value of F_{score} is close to 1, better is the quality of the variable selection. The $F_{score}(n^\star)$, introduced in Section 3.3.2, enables the quantification of the stability of selection by taking into account the number of true pertinent covariates selected. If this value is close to 1, better is the stability of the selection. For our study of stability, it is so important to look at the F_{score} and the $F_{score}(n^\star)$ together.

3.3.1 The classical F_{score}

The classical F_{score} [20] is a metric combining the precision and the recall. The precision is the ratio between the number of true positives and the number of considered positives (true positives and false positives). The recall is the ratio between the number of true positives and the number of positives (false negatives and true positives).

3.3.2 New index: F_{score} based on the number of hypothetical true covariates

We propose a new index, called $F_{score}(n^*)$, based on the number of the "true" variables that influence the survival time to compare on simulations the similarity indices on the different methods. We suppose that the selection sets are nested (assuming a proper index rearrangement):

$n_1 \leq n_2 \leq \cdots \leq n_S = N$

Let's denote by n^* the real number of significant features, and we also suppose that the total N genes selected contain these features.

s_i is considered a proportion (the number of times gene i is selected divided by the number of experiments)

$num(s)$ is the number of genes that is selected at least s times, $num(0^+) = N$.

We suppose that the methods are "consistent": the bigger s_i, the more probable the i^{th} gene is a truly important one, such that the genes can be ranked according to s. With the proper index rearrangement, we can suppose that:

$$s_1 \geq s_2 \geq \cdots \geq s_N, \text{ and } s_i = 0, \forall i > N$$

Let n^* be the number of true positive genes.

Let us denote $s^* = \begin{cases} \arg\max\{s \,|\, num(s) \geq n^*\} = s_{n^*} \text{ if } N \geq n^* \\ 0 \text{ otherwise} \end{cases}$

As a consequence, we also extrapolate by saying that, if $N < n^*$, then all genes such that $s_i > 0$ are true positive genes, while if $N \geq n^*$, then all genes such that $s_i \geq s^*$ are true positive genes.

Based on the assumptions that these genes are positive, we would expect that for all of them, $s_i = 1$, that is to say, that they are selected all the time. Therefore, $(1 - s_i)$ is the proportion of times that the gene is a false negative, and $\sum_{1\leq i\leq n^*}(1 - s_i)$ is the average number of a false negative.

Note that the formula remains valid when $n^* > N$, since the $n^* - N$ genes for which $s_i = 0$ are missed.

We then deduce the True Positive as the positive minus the false negative, and the average predicted positive as $\sum_{i=1}^{N} s_i$.

We can finally write Precision and Recall:

$$Precision(n^*) = \frac{n^* - \sum_{i=1}^{n^*}(1 - s_i)}{\sum_{i=1}^{N} s_i} = \frac{\sum_{i=1}^{n^*} s_i}{\sum_{i=1}^{N} s_i}$$

$$Recall(n^*) = \frac{n^* - \sum_{1\leq i\leq n^*}(1 - s_i)}{n^*} = \frac{\sum_{i=1}^{n^*} s_i}{n^*}$$

A traditional measure to measure the compromise between precision and recall is their harmonic mean (F-Score), which would read:

$$F_{score}(n^*) = 2\frac{Precision(n^*)Recall(n^*)}{Precision(n^*) + Recall(n^*)}$$

This new index depends on the "true" variables that influence the survival time. How can this index be used in practice? First, the idea in this paper is to compare the selection methods on simulations. In this case, we know which variables are relevant. We can compare the strategies from this new index with n^* known. This new index gives us an indication of the stability of the different methods in different configurations. However, when we work with real datasets, we do not know the number of "true" variables that influence the survival time. In this case, we can vary our index according to the number n^* of "true" relevant variables.

Moreover, we are interested in other metrics as AIC and BIC, and we also look at the number of selected genes presented in Section 3.4.

3.4 Supplementary criteria

We also calculated the Akaike Information Criterion (called AIC for *Akaike Information Criterion*) of the model obtained for each *seed*. The AIC criterion is a metric developed by [21] and enables the evaluation of the quality of a model. It allows us to manage both the quality of the fit and the complexity of the model by penalizing models with a large number of parameters. The best model will be the one with the lowest value of the AIC criterion. Therefore, this criterion is based on a compromise between the fit quality and the model complexity by penalizing models with a large number of parameters, limiting the effects of over-fitting (increasing the number of parameters necessarily improves the quality of the fit). We calculate the mean and standard deviation of the AIC criterion for each of the methods over the 100 *seeds*. This calculation also allows us to judge the quality of the selection.

For the simulated datasets, we also compute the classical F-score [20], introduced in Section 3.3.1, as we know the true positives.

From these different criteria, we can measure the various selection methods' performances in terms of stability and quality of the selection.

4 Tests on simulated data

First, we want to compare the regularization and screening method on simulated data when we know exactly which variables influence survival time.

4.1 Simulations

We simulated two datasets from the R package. We generate the survival times of these datasets from a Cox model where the baseline hazard function is modeled by a Weibull distribution $\mathcal{W}(a, b)$. The number of parameters is different between the two datasets: 1000 covariates for the first one and 25 000 covariates for the second one. The number of relevant covariates is equal in both cases to 20.

The simulation of data from the Cox model in the R package base on [22]. We have chosen to carry out this simulation to generate survival data that respects the proportional risk hypothesis. The generation of survival data from a Cox model base on:

$$T = H_0^{-1}\left[\frac{-\log(1-U)}{\exp(\beta^T X_{i.})}\right], \tag{11}$$

where $U \sim \mathcal{U}[0,1]$ and $X_{i.} \sim \mathcal{U}[-1,1]$. For this simulation, we consider that survival times follow a Weibull distribution $\mathcal{W}(a, b)$. In this case, we have the cumulative risk function expressed by:

$$H_0(t) = bt^a \tag{12}$$

and survival times can therefore be simulated from:

$$T = \frac{1}{b^{1/a}}\left(\frac{-\log(1-U)}{\exp(\beta^T X_{i.})}\right)^{1/a}. \tag{13}$$

We take for these simulations $a = 1.969765$ and $b = 7.586963e-07$ to have a mean equal to 1134 and a median equal to 1134. The idea is to simulate data that are not so far from real data.

4.2 Stability analysis

We study the stability of regularization and screening methods, but also the validity of the selection. The Sorensen, Jaccard, and F-score(n*) indexes enable the evaluation of the stability, while the AIC criterion and the classical F-score allow us to judge the quality of the selection. Table 1 gives the results of these different indexes on the two simulated datasets described above.

For the CoxCS method, we have considered two cases. The first one, denoted CoxCS1, considers 5 among the 20 relevant covariates as pre-selected covariates to add knowledge in the screening procedure. The other one, called CoxCS2, considers 5 among the 20 relevant covariates and 5 covariates known to be non-relevant as pre-selected covariates. The idea for this second case is to see if the procedure gives good results even if we are wrong about the pre-selection.

First, we can see that regularization methods have similar results about stability. The values of Sørensen and Jaccard indexes are close between the Lasso and Adaptive-Lasso methods. Moreover, the value of $F_{score}(n^\star)$ is also similar for these methods. Indeed, the value of $F_{score}(n^{\star})$ is equal to 0.8506 for the Adaptive-Lasso and to 0.8304 for the Lasso for 1000 covariates. We can observe that the value of the $F_{score}(n^{\star})$ of methods is very good on datasets with 1000 covariates. But when the number of covariates increases, the regularization methods have bad results in terms of stability. Indeed, in the high dimensional case, the Adaptive-Lasso is the method with the highest value of the $F_{score}(n^{\star})$ equal to 0.3897, and this value is meager, concluding to a bad level of stability for this method. It is necessary to look at the $F_{score}(n^{\star})$ by comparing it with the F_{score} to quantify both the stability of selection and the quality of selection. We can see that the quality of selection is rather bad for the regularization methods. Indeed, the value of the F_{score} is around 0.38 for all regularization methods. We can also observe that the quality of selection decreases with the dimensionality of the datasets. The F_{score} for all regularization methods is around 0.38 for the dataset with 1000 covariates, and this score is null on the datasets containing 25000 covariates. The results confirm the problem of stability of regularization methods in high-dimension and show us that the quality of the selection of these methods is bad in high-dimension.

For the screening methods, the Sørensen and Jaccard indexes are not the most adapted. Indeed, we can see that SIS and ISIS's screening methods have perfect values of these indexes due to the number of selected covariates correct and do not change according to seeds. These indexes must be completed with other criteria to confirm the stability and the quality of selection. First, we can see by looking at the $F_{score}(n^{\star})$ that SIS and ISIS (with d = 20) have a perfect score, explained by the fact that we impose to these methods to choose not more than 2O covariates. But their values of F_{score} are low (0.35 for 1000 covariates and 0.05 for 25000 covariates), which reflects the fact that the 20 selected covariates are not the right ones. The values of F_{score} for SIS and ISIS are close to those of the regularization methods. Moreover, these values of the F_{score} decrease with the dimensionality of the datasets, despite the values of $F_{score}(n^{\star})$ do not decrease. The methods SIS and ISIS, initially developed to solve the stability problem of regularization methods, seem to maintain selection instability. The PSIS method does not give satisfactory results in terms of variable selection either. This study has shown us that the method coxCS seems to be a good method considering the stability and selection quality. As introduced above, we considered two cases for the coxCS method. The first one, denoted CoxCS1, considers 5 among the 20 relevant covariates as pre-selected covariates to add knowledge in the screening procedure. The other one, denoted CoxCS2 considered 5 among the 20 relevant covariates and 5 covariates known to be non-relevant as pre-selected covariates. We can see that the values of Sørensen and Jaccard indexes are very good for both cases of the coxCS method. Their values are lower than these of SIS and ISIS methods. But the values of the F_{score} and the $F_{score}(n^{\star})$ of the coxCS methods are high, contrary to SIS and ISIS methods. The value of $F_{score}(n^{\star})$ for coxCS1 (5 pertinent covariates in the pre-selection) is equal to 0.919 for 1000 covariates and the one for coxCS2 (5 pertinent covariates and 5 non-pertinent covariates in the pre-selection) is equal to 0.8307. This value slows a few when we add the non-relevant covariates to the pre-selection. This observation shows that knowledge can improve the stability of selection. Besides, this knowledge seems to improve also the quality of selection. The F_{score} of all methods except the coxCS method is lower than 0.5. The Fsocre of coxCS1 is 0.8456 for 1000 covariates and 0.7571 for 25000 covariates, which means that most of the selected covariates are true covariates. We observe the same behavior with coxCS2, where the F_{score} for 1000 covariates is 0.7526, and the one for 25000 covariates is 0.778. We can see that the values of F_{score} and $F_{score}(n^{\star})$ do not decrease or decrease slowly. CoxCS also seems to handle the high-dimensional data. The stability study shows that coxCS is a good method to make the variable selection in high-dimension when some knowledge is available. Finally, the study confirms that it is crucial to detect potential markers in survival analysis to discuss the domain's experts. Indeed, coxCS seems to be the best method in our stability study. However, the users must use the included knowledge in this method correctly. Moreover, for the choice of d in SIS and ISIS, it would seem relevant to consider a higher value of d than one is looking for because one loses little stability but one gains in quality of selection. We, therefore, recommend discussing with professionals in the domain to find out what strategy to consider in practice.

5 Tests on Real dataset

We apply the methods presented above to a real data set. This dataset concerns clear cell renal carcinoma cancer (ccRCC) from the TCGA database (*The Cancer Genome Atlas*). The data are available at `https://www.cancer.gov/tcga`.

From this database, we consider three sets of covariates for our study. First, we consider as covariates only the *Immune-Checkpoints* (IC), *i.e.* genes involved in the immune response process. There are 48 covariates in this first set. Then, we choose to work with the genes considered as differentially expressed in the study published in [23]. The number of genes found differentially expressed by the DESeq2 method [24] is 11 289 genes. Finally, we consider all the coding genes present in humans. After filtering to remove the null genes, we obtain 17 789 genes.

We applied both regularization methods and screening methods on these three sets of covariates. For the pre-selection of the CoxCS method for the second and third datasets, we had the idea to use the latest publications about markers capable of predicting patient survival. For this, we carried out a PubMed search using the following keywords and MeSH terms (for *Medical Subject Headings*): `ccRCC [tiab] prognosis [tiab] survival [tiab] genes NOT RNA`. MeSH is

Methods	#Covariates	Sorensen	Jaccard	#Genes selected	$F_{score}(n^\star)$ $(n^\star = 20)$	F_{score}
Lasso	1000	0.9939	0.6184	28 (6.9515)	0.8304	0.38
	25000	0.9869	0.4298	4.79 (2.5556)	0.3864	0
Adaptive-Lasso	1000	0.9937	0.6122	25.26 (7.0534)	0.8506	0.384
	25000	0.9892	0.4788	4.84 (2.2862)	0.3897	0
SIS (d=10)	1000	1	1	10 (0)	0.6667	0.4
	25000	1	1	10 (0)	0.6667	0.0667
SIS (d=20)	1000	1	1	20 (0)	1	0.35
	25000	1	1	20 (0)	1	0.05
SIS (d = 50)	1000	0.9996	0.9592	47.02 (1.8694)	0.5968	0.3581
	25000	1	1	48 (0)	0.5882	0.1471
SIS (d=100)	1000	0.9944	0.6388	53.96 (25.4899)	0.4911	0.2977
	25000	0.9998	0.978	92.93 (0.9018)	0.3542	0.124
ISIS (d =10)	1000	0.999	0.9082	10 (0)	0.6667	0.3333
	25000	1	1	10 (0)	0.6667	0
ISIS (d =20)	1000	0.9995	0.9519	20 (0)	0.999	0.35
	25000	1	1	20 (0)	1	0.05
ISIS (d = 50)	1000	0.999	0.9082	50 (0)	0.5714	0.2857
	25000	1	1	50 (0)	0.5714	0.1143
ISIS (d =100)	1000	0.9898	0.4924	100 (0)	0.3333	0.2005
	25000	0.999	0.9082	100 (0)	0.3333	0.1
PSIS	1000	1	1	6 (0)	0.4615	0.4615
	25000	0.9987	0.8869	4.44 (0.4989)	0.3633	0
coxCS1	1000	0.9988	0.8963	23.33 (1.9749)	0.919	0.8456
	25000	0.998	0.8352	20.92 (6.0963)	0.8876	0.7571
coxCS2	1000	0.9993	0.9377	28.15 (1.158)	0.8307	0.7526
	25000	0.9996	0.9641	27.97 (3.6803)	0.8197	0.778

Table 1: Summary of selection stability and quality by considering differents metics (Sørensen index, Jaccard index, F_{score}, $F_{score}(n^\star)$, the mean and the sd of selected covariate number) for the simulation study.

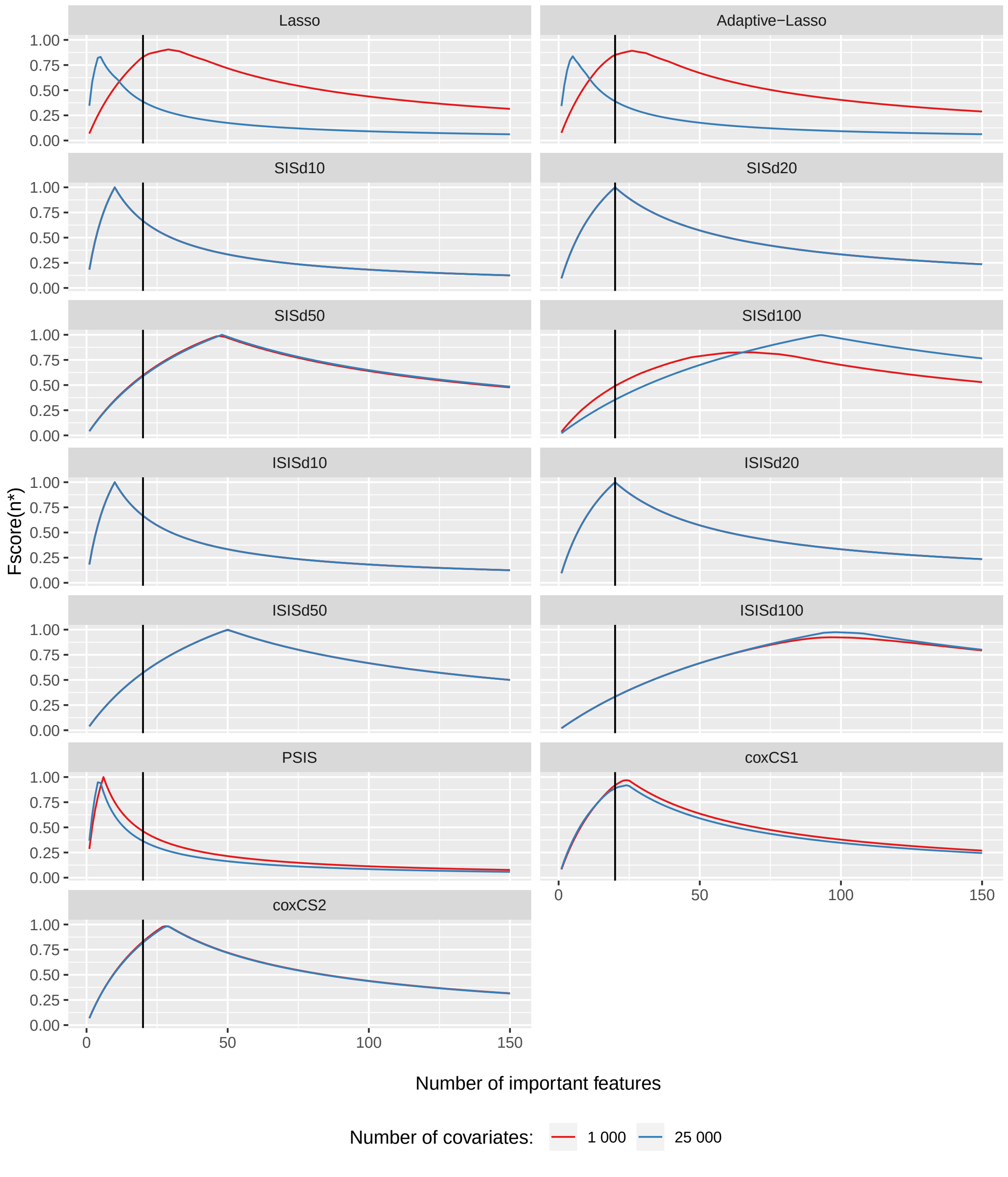


Figure 1: The $F_{score}(n^{\star})$ are plotted as a function of $n*$ for the different methods

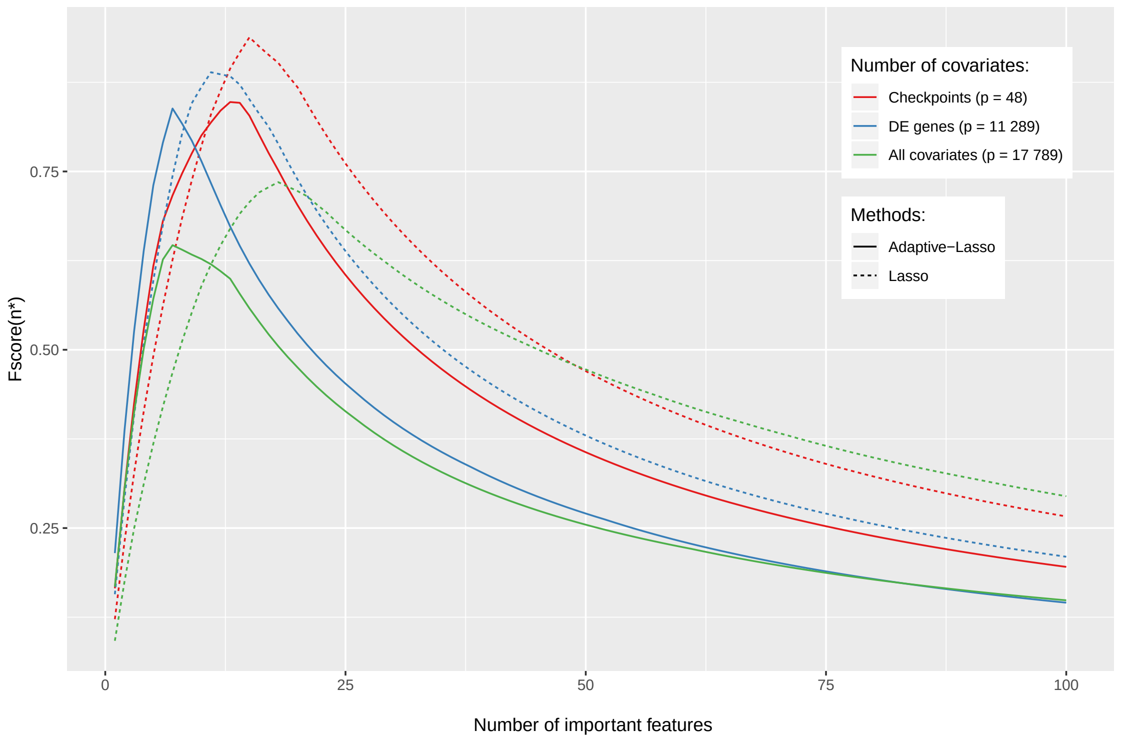


Figure 2: The $F_{score}(n^{\star})$ are plotted as a function of $n*$ for the different methods for the real datasets

a hierarchically organized and controlled vocabulary produced by the *National Library of Medicine*. We thus obtained 32 results corresponding to our search, and we decided to keep the first 10 *abstracts*. These 10 *abstracts* were saved and later loaded into the web application `beca annotate` nunesbecas2013. This web application aims to enable the identification and annotation of medical concepts in a text. The identification and annotation of genes and proteins are performed using *machine learning* methods present in Gimli camposgimli2013. Gimli camposgimli2013 is a *open-source* tool for the automatic recognition of biomedical terms. The list of genes obtained is: $KDM2B$, $HMGCS2$, $HSD11B1$, $IL10$, $KDM5B$, $KDM5A$, $KDM5C$, $KDM5D$, $KDM1B$, $OGDHL$, $SSBP2$, $VSIG4$ and $XCR1$. We decided to use this list as biological knowledge for pre-selection in coxCS for the real dataset.

We applied the methods on the 100 different *seeds* (seed is an integer used to initialize a random number generator) to study their stability and better interpret the selection. In addition to the calculation of similarity indices, we have also calculated a selection percentage for each gene (see Table 4 and Table 5 in the Supplementary material). This selection percentage allows us to avoid the conclusion that a gene that would have been selected only a few times, by chance, is important. We carried out the biological analysis of the selected genes from the site `GeneCards` [25] accessible at `https://www.genecards.org/` and from the site `COSMIC` [26] (for *Catalogue Of Somatic Mutations In Cancer*) available at `https://cancer.sanger.ac.uk/cosmic`. `GeneCards` is a database that provides complete information on all predicted and annotated human genes. `COSMIC` is also a database, but this one exclusively dedicates to cancer. For `COSMIC` we have particularly relied on the *Cancer Gene Census* catalog, which lists genes with mutations involved in cancer.

5.1 Stability analysis and marker discovery

5.1.1 Regularization methods

TABLE 2 presents the results of the Sørensen and Jaccard indexes, the mean number of selected genes (and the standard deviation), the $F_{score}(n^{\star})$ and the AIC for the Lasso and the Adaptive-Lasso for the three sets of covariates: the Immune-Checkpoints, the differential expressed covariates and all genes. As expected, the Sørensen index is high for the regularization methods when the number of covariates is small. When the number of covariates increases, the Adaptive-Lasso seems to be less stable than the Lasso. Indeed the more covariates we consider, the worse the Sørensen and Jaccard indexes are for the Adaptive-Lasso. The Lasso also has more difficulties when the number of covariates

		Lasso	Adaptive Lasso
Immune-Checkpoints	Sørensen index	0.9960	0.9933
	Jaccard index	0.73	0.60
	$F_{score}(n^{\star})$ (n*=20)	0.8682	0.703
	Number of selected genes	15.36 (2.83)	10.84 (3.53)
	AIC	1915.50 (4.33)	1917.71 (11.06)
Differential expressed genes	Sørensen index	0.9946	0.9436
	Jaccard index	0.65	0.14
	$F_{score}(n^{\star})$ (n*=20)	0.739	0.523
	Number of selected genes	11.72 (2.34)	7.84 (3.01)
	AIC	1867.35 (1.95)	1862.47 (23.04)
All genes	Sørensen index	0.9332	0.8284
	Jaccard index	0.12	0.05
	$F_{score}(n^{\star})$ (n*=20)	0.7225	0.4754
	Number of selected genes	17.70 (3.57)	8.65 (3.64)
	AIC	1873.43 (24.95)	1870.42 (40.97)

Table 2: Results of similarity indexes (Sørensen and Jaccard index), of the average AIC and of the average of selected genes on the whole seeds for the studied regularization methods according to the set of genes given as input. The standard deviations are precised in the brackets.

becomes large. However, it seems to be still stable when we work with the differential expressed covariates, whereas the Adaptive-Lasso has rather bad similarity indexes in this case. However, these two indexes are not the most adapted because they depend strongly on the number of selected covariates. The index $F_{score}(n^{\star})$ we have proposed enables us to vary the number of pertinent covariates and to see the effects of this number on the stability of selection. By considering $F_{score}(n^{\star})$, the Lasso method seems to be the most stable. Figure 2 shows that the dotted curves are higher than the solid curves, confirming this observation. That means that the values of the $F_{score}(n^{\star})$ for the Adaptive-Lasso are the highest and the most precise when the number of relevant covariates is larger than 10. However, the selection quality seems better for the Lasso method; these values are lower than those of the Adaptive-Lasso method. The values of AIC stay close between these two methods. We can also observe that the values of the indexes (Sørensen, Jaccard, $F_{score}(n^{\star})$) decrease with the dimensionality of data, which confirms that the regularization methods are not stable in high-dimension.

However, we can see from TABLE 4 in the Supplementary material that many genes are selected in only one *seed* for the Lasso and Adaptive-Lasso methods. We can see that the Sørensen and Jaccard indexes do not highlight this phenomenon, which is not a good point for the stability of these methods.

5.1.2 Screening methods

We can see on TABLE 3 that the different screening methods give different results as far as stability is concerned. Indeed, if the SIS method seems very stable even when the number of variables increases, the other methods have more difficulties in high-dimension. The behavior of stability appears to be confirmed with the $F_{score}(n^{\star})$ for the SIS and ISIS methods. Contrary to the values of Sørensen and Jaccard indexes, the values of $F_{score}(n^{\star})$ for the coxCS and PSIS methods increase with the dimensionality of data. This observation confirms the drawback of Sørensen and Jaccard indexes due to their too strict or soft behavior. In Figure 3, we can see that SIS and ISIS always have best $F_{score}(n^{\star})$ for a small $n^{\star}$ even when the number of covariates in the study increases, these methods are very sparse, but we know that they do not select the relevant covariates. The screening methods select very few variables, notably SIS and ISIS. For example, for the Immune-Checkpoints, the average of the variables chosen for SIS is 2.57. The Lasso, on the contrary, selects an average of 15.36 genes. However, the number of variables in this dataset is small, and we chose these variables early because they had a potential impact on clear cell Renal Cell Carcinoma (ccRCC). Many variables could therefore correlate with each other, and this could explain this lower selection stability result. PSIS and CoxCS select more variables when the number of covariates increases. We can nevertheless note that for the sets constitute with the 48 Immune-Checkpoints, the Lasso remains the most stable method. The ISIS method seems to be the worst stable method, and the CoxCS method based on the biological knowledge gives good stability results in terms of $F_{score}(n^{\star})$. However, the knowledge used for CoxCS is maybe not the best biological knowledge that we can add to the method. The performances of this method could be even better with better biological knowledge. From the AIC, we can also conclude that the variables selected by PSIS and coxCS seem to explain less well the survival duration than the other methods. This observation could also explain by a wrong chosen knowledge added to the method.

		SIS	ISIS	PSIS	coxCS
Immune-Checkpoints	Sørensen index	0.9708	0.6089	0.9983	0.9974
	Jaccard index	0.2495	0.798	0.8566	0.796
	$F_{score}(n^{\star})$ (n*=20)	0.2277	0.3532	0.6004	0.5704
	Number of selected genes	2.57 (2.23)	4.29 (0.69)	8.58 (0.57)	7.98 (2.01)
	AIC	1953.37 (21.36)	1935.35 (3.66)	1961.22 (4.33)	1946.50 (9.92)
Differential expressed genes	Sørensen index	0.9905	0.382	0.9662	0.8885
	Jaccard index	0.5101	0.5841	0.2207	0.8127
	$F_{score}(n^{\star})$ (n*=20)	0.3416	0.3519	0.8824	0.6188
	Number of selected genes	4.12 (1.57)	4.27 (1.02)	18.51 (5.11)	8.96 (1.47)
	AIC	1903.63 (7.78)	1895.25 (5.92)	1944.20 (5.27)	1960.39 (3.50)
All genes	Sørensen index	0.9962	0.8956	0.9610	0.9341
	Jaccard index	0.7222	0.6212	0.2492	0.1231
	$F_{score}(n^{\star})$ (n*=20)	0.4496	0.424	0.7494	0.7814
	Number of selected genes	5.80 (1.04)	5.39 (1.50)	27.21 (9.18)	25.85 (14.44)
	AIC	1873.80 (0.71)	1880.01 (24.69)	1931.38 (12.55)	1937.71 (13.69)

Table 3: Results of similarity indexes (Sørensen and Jaccard index), of the average AIC and of the average of selected genes on the whole seeds for the studied screening methods according to the set of genes given as input. The standard deviations are precised in the brackets.

5.2 Potential markers explaining the survival duration

Concerning the gene selection from the regularization methods, we refer the reader to Table 4 in the supplementary material. Several genes are often selected, and some have biological interpretability. First, we have the $FBXL5$ gene that is involved in the immune system. We report several phenotypes for this gene, one of which corresponds to chronic kidney disease. The location of this gene is on the same segment as the $BST1$ and $CD38$ *Immune-Checkpoints* on chromosome 4. The $FBXL5$ gene, therefore, appears to be an essential gene to explain survival for clear cell renal cell carcinoma. Then, the $CKAP4$ gene occurs in the selection of the different methods and is involved in the immune system. A disease associated with the $CKAP4$ gene is cystitis, an infection that affects the bladder, a part of the urinary system like the kidneys. These observations may reinforce the idea that the $CKAP4$ gene would be a useful marker for the survival of patients with kidney cancer. Although they are not methodically selected and/or selected with a small percentage, two other genes that we find interesting are the $CHEK2$ and $C10orf90$ genes, which are tumor suppressors. The function of a tumor suppressor gene is to prevent the runaway of cell division. If it is present in a cancer cell, it tends to slow down cell proliferation. These genes are, therefore, good indicators of a patient's survival to the disease in question.

We can also observe that the selected genes are most of the time the same considering only the differentially expressed genes and all the genes (cf. TABLE 4) for the Lasso and Adaptive-Lasso. For both methods, we have the minimum set $\{GDF5$, $CKAP4$,$CUBN$,$OTOF$,$SORBS2\}$. In this set, we have the genes $CKAP4$ that already appeared to be promising biomarkers of patient survival. The $FBXL5$ gene mentioned above as the right biomarker candidate is a selected gene by both methods. Finally, the $CUBN$ gene also appears to be a good biomarker to explain patient survival. It is a receptor located on the epithelial tissue of the intestine and kidney. The $CUBN$ may be a prognostic marker for clear cell renal cell carcinoma from the study of [27].

Concerning the gene selection from the screening methods, we can see that SIS and ISIS select the $CHEK2$ gene from Table 5. In Section 5.1.1, we already mentioned that it could be interesting as markers for the prognosis of patients

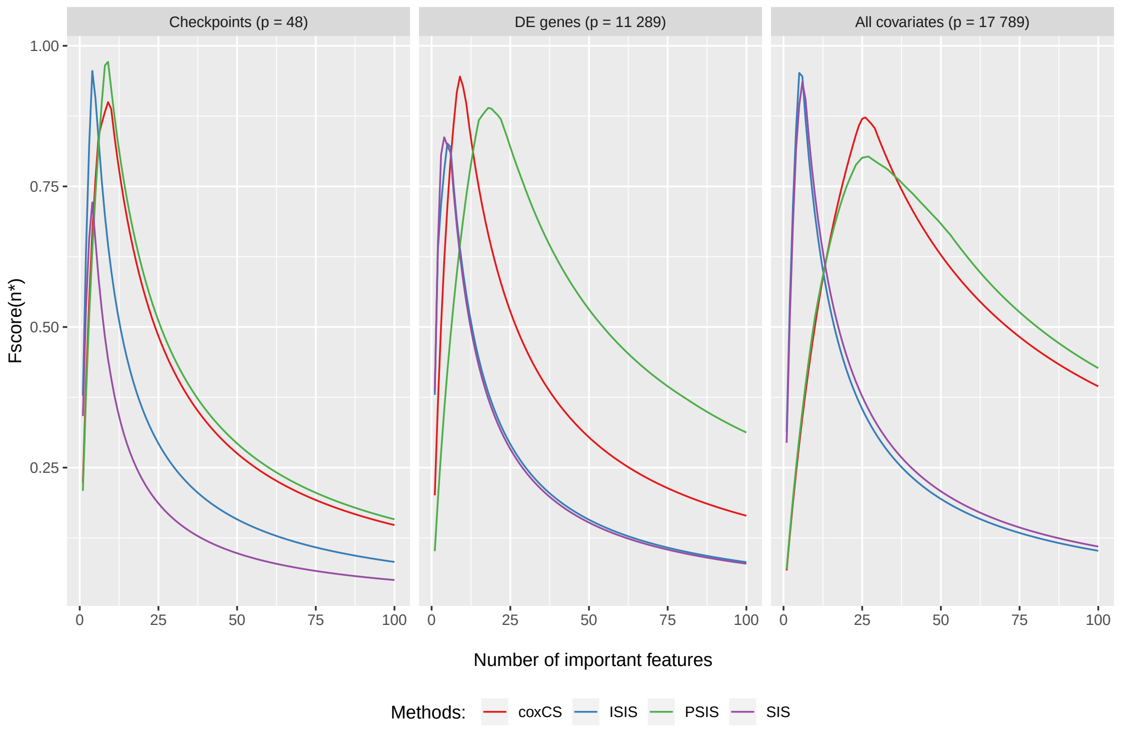


Figure 3: The $F_{score}(n^\star)$ are plotted as a function of $n*$ for the different screening methods for the real datasets.

with ccRCC. Finally, we can notice that the screening method's biological knowledge has changed the gene selection compared to SIS, ISIS, and PSIS methods that do not use knowledge. Indeed, none of the genes selected in coxCS are found in SIS, ISIS or PSIS (cf. TABLE 5 in the supplementary material). It is more difficult to conclude on these results because none of the selected genes is by all methods simultaneously. One explanation of these bad results is that we introduce knowledge from a preliminary study on abstracts. Maybe this study lacks informed advice from a biologist or doctor.

6 Discussion

In this study, we have compared several feature selection methods in the Cox model, with a focus on high-dimension: Lasso, Adaptive-Lasso, SIS, ISIS, PSIS, CoxCS. These comparisons were achieved in a simulation study and on a real dataset. In the simulation study and for a low number of covariates, we showed that Lasso and adpative-Lasso are stable in selection from the values of the similarity indexes. However, the selection quality remains poor, the situation being better for adaptive-Lasso. But in high-dimension, Lasso and adpative-Lasso are both unstable and the quality of selection is also low. The screening methods seem to solve, partially, the problem of stability noticed for the regularization methods. Indeed, SIS and ISIS methods have rather good stability results, but the selection is incorrect. By choosing a pertinent value of the selection threshold d, we can improve the selection quality. It thus needs to be chosen carefully. CoxCS seems to be a very good method when some biological knowledge is available: in the simulation study, the selection stability of this method is good, and the selection quality is also correct. In this study, screening methods allow a more straightforward interpretation of the results. The number of variables selected by the screening methods is lower than that of the regularization methods.

On the real dataset, screening methods (PSIS and coxCS) appear to be more stable, especially for the highest dimension, when all genes are kept in the regression model. The $F_{score}(n^\star)$ values for the screening and regularization methods are in agreement with this. The $F_{score}(n^\star)$ values of the screening methods are higher in high dimensions than those of the regularization methods. Moreover, we can observe that the value of the $F_{score}(n^\star)$ of regularization methods decreases with increasing dimension, which is not the case with screening methods. As mentioned in the simulation study, CoxCS is an attractive method because it also has good stability results by considering $F_{score}(n^\star)$ on the real dataset. If there is no prior available biological information, then PSIS seems like the best compromise in terms of both stability and quality. Finally, we noticed that the Sørensen index tended to indicate nested selection scenarios as more stable, even if the number of variables varies. In contrast, the Jaccard index penalizes this type of design. That is why we proposed the $F_{score}(n^{\star})$, inspired by the F_{score}, allowing to consider a variable number of pertinent covariates, since it is generally unknown. A critical point to consider in future studies is to further explore why higher regularity indexes do not necessarily translate into a better behavior of the model in terms of AIC.

The study of the selection of variables to explain patient survival from *Immune-Checkpoints* shows that some of them would be interesting to study further. Indeed, the $B7\text{-}H3(CD276)$ gene seems to be a good biomarker to explain patient survival. It is also the case with $HLA.G$. We have seen in the study and in [23] that $HLA.G$ shares information with other genes. [23] raises the possibility that this $HLA.G/ILT2$ pair may be an alternative to $PD1/PDL1$ treatments when patients do not respond to the latter. A perspective from a biological point of view is to further study this couple.

By extending the variable selection study to differentially expressed genes and the set of genes as a whole, we were able to observe genes that could be of potential interest. It would be relevant to deepen their study. The $CHEK2$,$CKAP4$ genes appear to be important markers to explain survival. The $CHEK2$ gene is already known to have an impact on breast cancer and the $CKAP4$ gene is involved in the immune system. We recall that the clear cell renal carcinoma is an immunogenic cancer, which means that it can stop the immune system. Another gene that appears in the selection to explain the survival of patients is the $CUBN$, gene. In [27], this gene was said to play a key role in the development of clear cell renal carcinoma. Finally, the $FBXL5$ gene also appears to be a potential marker to explain patient survival as it has a role in the immune system, is involved in chronic kidney disease, and is thought to be related to two *Immune-Checkpoints* listed in the study by [23].

References

[1] D. R. Cox. Regression Models and Life-Tables. *Journal of the Royal Statistical Society. Series B (Methodological)*, 34(2):187–220, 1972.

[2] Robert Tibshirani. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996.

[3] Hui Zou. The Adaptive Lasso and Its Oracle Properties. *Journal of the American Statistical Association*, 101(476):1418–1429, 2006.

[4] Pierre J. M. Verweij and Hans C. Van Houwelingen. Penalized likelihood in Cox regression. *Statistics in Medicine*, 13(23-24):2427–2436, 1994.

[5] Jianqing Fan and Runze Li. Variable Selection via Nonconcave Penalized Likelihood and Its Oracle Properties. *Journal of the American Statistical Association*, 96(456):1348–1360, 2001.

[6] Robert Tibshirani. The Lasso Method for Variable Selection in the Cox Model. *Statistics in Medicine*, 16(4):385–395, 1997.

[7] H. H. Zhang and W. Lu. Adaptive Lasso for Cox's proportional hazards model. *Biometrika*, 94(3):691–703, 2007.

[8] Jianqing Fan, Yang Feng, and Yichao Wu. *High-dimensional variable selection for Cox's proportional hazards model*. Institute of Mathematical Statistics, 2010.

[9] Jianqing Fan and Rui Song. Sure independence screening in generalized linear models with NP-dimensionality. *The Annals of Statistics*, 38(6):3567–3604, 2010.

[10] Hyokyoung G. Hong, Jian Kang, and Yi Li. Conditional screening for ultra-high dimensional covariates with survival outcomes. *Lifetime Data Analysis*, 24(1):45–71, 2018.

[11] Sihai Dave Zhao and Yi Li. Principled sure independence screening for Cox models with ultra-high-dimensional covariates. *Journal of Multivariate Analysis*, 105(1):397–411, 2012.

[12] Jianqing Fan and Jinchi Lv. Sure independence screening for ultrahigh dimensional feature space. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(5):849–911, 2008.

[13] Stefan Michiels, Serge Koscielny, and Catherine Hill. Prediction of cancer outcome with microarrays: a multiple random validation strategy. *Lancet (London, England)*, 365(9458):488–492, 2005.

[14] Héctor T. Arita. Multisite and multispecies measures of overlap, co-occurrence, and co-diversity. *Ecography*, 40(6):709–718, 2017.

[15] Andrés Baselga. Multiple site dissimilarity quantifies compositional heterogeneity among several sites, while average pairwise dissimilarity may be misleading. *Ecography*, 36(2):124–128, 2013.

[16] Andrés Baselga. Partitioning the turnover and nestedness components of beta diversity. *Global Ecology and Biogeography*, 19(1):134–143, 2010.

[17] D. R. Cox. Partial likelihood. *Biometrika*, 62(2):269–276, 1975.

[18] Peter J. Bickel, Bo Li, Alexandre B. Tsybakov, Sara A. van de Geer, Bin Yu, Teófilo Valdés, Carlos Rivero, Jianqing Fan, and Aad van der Vaart. Regularization in statistics. *Test*, 15(2):271–344, 2006.

[19] Diego Franco Saldana and Yang Feng. SIS : An R Package for Sure Independence Screening in Ultrahigh-Dimensional Statistical Models. *Journal of Statistical Software*, 83(2), 2018.

[20] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to information retrieval*. Cambridge University Press, New York, 2008. OCLC: ocn190786122.

[21] Hirotogu Akaike. *Information Theory and an Extension of the Maximum Likelihood Principle*, pages 199–213. Springer Science+Business Media, New York, 1998.

[22] Ralf Bender, Thomas Augustin, and Maria Blettner. Generating survival times to simulate Cox proportional hazards models. *Statistics in Medicine*, 24(11):1713–1723, 2005.

[23] Diana Tronik-Le Roux, Mathilde Sautreuil, Mahmoud Bentriou, Jérôme Vérine, Maria Belén Palma, Marina Daouya, Fatiha Bouhidel, Sarah Lemler, Joel LeMaoult, François Desgrandchamps, Paul-Henry Cournède, and Edgardo D. Carosella. Comprehensive landscape of immune-checkpoints uncovered in clear cell renal cell carcinoma reveals new and emerging therapeutic targets. *Cancer Immunology, Immunotherapy*, 2020.

[24] Michael I. Love, Wolfgang Huber, and Simon Anders. Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. *Genome Biology*, 15(12):550, 2014.

[25] Gil Stelzer, Naomi Rosen, Inbar Plaschkes, Shahar Zimmerman, Michal Twik, Simon Fishilevich, Tsippi Iny Stein, Ron Nudel, Iris Lieder, Yaron Mazor, Sergey Kaplan, Dvir Dahary, David Warshawsky, Yaron Guan-Golan, Asher Kohn, Noa Rappaport, Marilyn Safran, and Doron Lancet. The GeneCards Suite: From Gene Data Mining to Disease Genome Sequence Analyses. *Current Protocols in Bioinformatics*, 54(1):1.30.1–1.30.33, 2016.

[26] Zbyslaw Sondka, Sally Bamford, Charlotte G. Cole, Sari A. Ward, Ian Dunham, and Simon A. Forbes. The COSMIC Cancer Gene Census: describing genetic dysfunction across all human cancers. *Nature Reviews Cancer*, 18(11):696–705, 2018.

[27] Gabriela Gremel, Dijana Djureinovic, Marjut Niinivirta, Alexander Laird, Oscar Ljungqvist, Henrik Johannesson, Julia Bergman, Per-Henrik Edqvist, Sanjay Navani, Naila Khan, Tushar Patil, Åsa Sivertsson, Mathias Uhlén, David J. Harrison, Gustav J. Ullenhag, Grant D. Stewart, and Fredrik Pontén. A systematic search strategy identifies cubilin as independent prognostic marker for renal cell carcinoma. *BMC cancer*, 17(1):9, 2017.

A Supplementary material

A.1 Méthodes de régularisation

A.1.1 Sur l'ensemble des gènes

	gènes sélectionnés			
Lasso	*ABCB5*(1%)	*ACRC*(2%)	*AMZ2*(1%)	*ANAPC7*(6%)
	APOBEC2(2%)	*ARHGEF4*(2%)	**B3GNTL1**(87%)	*BIN1*(1%)
(AIC =	*BIRC6*(7%)	*C10orf90*(25%)	*C14orf165*(1%)	**C19orf76**(80%)
1873.43	*C1R*(1%)	*C20orf112*(1%)	*CACNA1S*(2%)	*CAMK2N2*(1%)
±24.95)	*CAMSAP1*(1%)	*CCDC19*(25%)	*CCDC51*(1%)	*CDC7*(1%)
	CDCA3(62%)	*CENPBD1*(3%)	**CHEK2**(81%)	**CKAP4**(87%)
	CNN2(1%)	*CTAGE9*(7%)	**CUBN**(87%)	*CYB5D2*(1%)
	DAG1(2%)	*DDX24*(3%)	*DHRS12*(8%)	*EHBP1L1*(3%)
	EIF4EBP2(59%)	*EMILIN3*(1%)	*FAM133A*(7%)	*FAM63A*(3%)
	FAM64A(1%)	*FAM66A*(7%)	*FAM95B1*(1%)	**FBXL5**(81%)
	FLI1(1%)	*FUT3*(7%)	*GABBR1*(1%)	*GABPB2*(1%)
	GDF5(74%)	*GTPBP2*(1%)	*GZMA*(1%)	**HBP1**(68%)
	HEG1(7%)	*HPCAL1*(5%)	*HPCA*(7%)	*IGF2*(7%)
	IKBIP(5%)	*IKBKG*(5%)	*ILDR1*(7%)	*ITFG2*(2%)
	ITPRIPL1(1%)	*JAGN1*(1%)	*KIAA1109*(1%)	*KIAA1524*(1%)
	KIF18B(13%)	*KIF21B*(7%)	*KLHL14*(7%)	*LEO1*(2%)
	LOC284233(8%)	*LPIN1*(2%)	*LRRC23*(4%)	*LRRC8E*(13%)
	LRRN4(7%)	*MAST4*(22%)	**MBOAT7**(72%)	*MDM4*(2%)
	MGST1(2%)	*MIA*(1%)	*MKRN3*(3%)	*MOGAT2*(7%)
	MRAP(2%)	*NAA30*(2%)	*NDUFA8*(7%)	*NEK2*(8%)
	NUMBL(1%)	*OSTC*(5%)	*OTOF*(81%)	*P2RX7*(2%)
	PABPC3(8%)	*PDCD1*(5%)	*PDCD2*(1%)	*PHTF2*(2%)
	PIGK(5%)	*PIGO*(5%)	*PKD1L3*(2%)	*PRDM7*(1%)
	PRKAA1(1%)	*PROCA1*(2%)	*PROS1*(2%)	*PRUNE*(8%)
	PTPLA(81%)	**RANGAP1**(69%)	*RGS17*(50%)	*RGS20*(50%)
	RP9P(6%)	*RPL36AL*(8%)	*SCAP*(1%)	*SCO1*(4%)
	SEC61A2(80%)	*SERPINB8*(1%)	*SH2D4A*(6%)	*SHQ1*(1%)
	SLC12A8(6%)	*SLC45A4*(4%)	*SNCA*(6%)	**SORBS2**(81%)
	SYVN1(4%)	*TACC2*(1%)	*TAF9*(1%)	*TARP*(1%)
	THEM4(1%)	*THEMIS*(4%)	*TMEM17*(4%)	*TMEM203*(1%)
	TMEM207(1%)	*TMEM71*(4%)	*TTLL11*(6%)	*TTLL1*(1%)
	TUBGCP5(2%)	*TUSC1*(4%)	*UBE2D2*(1%)	*UMODL1*(2%)
	VWA5B1(1%)	*ZNF148*(1%)	*ZNF232*(1%)	*ZNF252*(1%)
	ZNF626(4%)	*ZNF676*(7%)	*ZNF766*(1%)	*ZNF90*(2%)
Adaptive-	*ABCA8*(1%)	*ABCF3*(1%)	*ADCY10*(1%)	*ANAPC7*(2%)
Lasso	*ARHGEF4*(1%)	*ATP2A3*(1%)	**B3GNTL1**(82%)	*BIRC6*(2%)
	BMP8B(2%)	*BRSK1*(1%)	*C10orf12*(1%)	*C10orf90*(21%)
(AIC =	*C12orf40*(3%)	*C19orf76*(6%)	*C21orf45*(1%)	*C2CD4C*(2%)
1870.42	*C4orf39*(1%)	*C5orf32*(1%)	*CAMSAP1*(2%)	*CASKIN1*(1%)
±40.97)	*CBWD2*(1%)	*CCDC14*(1%)	*CCDC19*(19%)	*CCNG1*(1%)
	CENPBD1(5%)	*CHCHD10*(1%)	**CKAP4**(70%)	*CLNK*(2%)
	COL6A2(1%)	**CUBN**(82%)	*CUEDC1*(2%)	*DACH1*(1%)
	DAG1(1%)	*DBF4B*(2%)	*DDX24*(1%)	*DHRS12*(1%)
	DQX1(1%)	*DUSP2*(1%)	*EHBP1L1*(1%)	*EIF4EBP2*(1%)
	ELFN2(1%)	*ENGASE*(1%)	*ENTPD6*(1%)	*FAM133A*(4%)
	FAM63A(1%)	*FAM66A*(3%)	*FAM7A2*(1%)	*FAM95B1*(1%)
	FBXL5(48%)	*FBXO39*(2%)	*FCGR1C*(1%)	*GABPB2*(1%)
	GAS2L3(1%)	**GDF5**(74%)	*GFM2*(1%)	*GJB1*(1%)
	GNAI2(1%)	*GSTO1*(2%)	*GZMA*(1%)	*HBP1*(27%)
	HDAC7(1%)	*HDGF*(1%)	*HEG1*(1%)	*HIST1H2BK*(2%)
	IKBKG(1%)	*ILDR1*(3%)	*JAGN1*(1%)	*JPH3*(1%)

	$KIF21B$(4%)	$KIF22$(2%)	$KLHL14$(1%)	$LAD1$(1%)
	$LAMB3$(2%)	$LCA5L$(1%)	$LILRB2$(1%)	$LOC284233$(2%)
	$LOC646471$(1%)	$LPIN1$(1%)	$LRRC8E$(4%)	$LRRN4$(2%)
	$LST1$(1%)	$MAST4$(1%)	$MDM4$(2%)	$MED20$(1%)
	$MFSD6$(1%)	$MIOX$(2%)	$MKI67IP$(1%)	MME(1%)
	$MOGAT2$(2%)	$MOXD1$(2%)	$MRGPRX3$(2%)	$NBPF3$(1%)
	$NCRNA00081$(1%)	$NDUFA8$(2%)	$NDUFS6$(2%)	$NFAM1$(1%)
	$NKRF$(2%)	$NXT1$(2%)	$OR9Q1$(1%)	$OSTC$(1%)
	OTOF(79%)	$OXCT1$(1%)	$PARP1$(1%)	$PKD1L3$(1%)
	$POLD2$(1%)	$POLI$(2%)	$PROS1$(2%)	$PROSC$(1%)
	$PRR12$(1%)	$PTGER3$(1%)	$PTPLA$(27%)	$RAB40AL$(5%)
	$RANGAP1$(2%)	$RGS20$(24%)	$RNF216L$(1%)	$RPL36AL$(3%)
	$SCO1$(3%)	$SEC13$(1%)	**SEC61A2**(26%)	$SERPINA7$(1%)
	$SERPINB8$(1%)	$SETD5$(2%)	$SFTA1P$(1%)	$SH2D4A$(1%)
	$SHQ1$(1%)	$SKA2$(1%)	$SLCO1A2$(1%)	$SLITRK3$(1%)
	$SMARCA2$(1%)	$SNCA$(1%)	$SNORA39$(1%)	**SORBS2**(78%)
	$SPIC$(2%)	$STX1A$(1%)	$SYNC$(1%)	$TAF1L$(1%)
	$TAF5L$(2%)	$TAF9$(1%)	$TBC1D15$(1%)	$TMCC1$(1%)
	$TMEM31$(1%)	$TMEM71$(1%)	$TNFSF12$(2%)	$TRAF7$(1%)
	$TUSC1$(3%)	$UAP1$(1%)	$UBE2R2$(1%)	$UMODL1$(1%)
	$USP42$(2%)	$WNT1$(4%)	$ZAP70$(1%)	$ZGPAT$(2%)
	$ZNF252$(1%)	$ZNF283$(2%)	$ZNF766$(2%)	$ZNF833$(1%)
	$ZNF90$(1%)			

Table 4: Résultats des méthodes de régularisation sur l'ensemble de gènes

A.2 Méthodes de *Screening*

A.2.1 Sur l'ensemble des gènes

	gènes sélectionnés			
SIS (AIC = 1873.80 ± 0.71)	**C5orf23**(100%)	**CAD**(83%)	**CHEK2**(100%)	**CUBN**(100%)
	DHRS12(27%)	**GDF5**(100%)	*SPC24*(1%)	*TOP2A*(69%)
ISIS (AIC = 1880.01 ±24.69)	*APOBEC2*(1%)	**C5orf23** (94%)	**CAD**(88%)	**CHEK2**(94%)
	COMMD5(1%)	*CTSO*(1%)	**CUBN**(94%)	*DHRS12*(45%)
	DUOX1(1%)	*EHBP1L1*(1%)	*FAM155B*(1%)	**GDF5** (93%)
	GNA11(2%)	*GRM7*(2%)	*IGF2*(2%)	*KIF21B*(2%)
	LPIN1(1%)	*NDUFA8*(2%)	*NKD1*(2%)	*NUP153*(2%)
	PHTF2(2%)	*PROS1*(2%)	*SNCA*(2%)	*SPAG5*(2%)
	STK40(2%)	*TMEM17*(2%)	*TUSC1*(2%)	*ZGPAT*(1%)
PSIS (AIC = 1931.38 ±12.55)	**ABTB2**(70%)	*ACER2*(2%)	*ACSBG1*(1%)	**ADAM17**(97%)
	ADCK1(85%)	*ADCY7*(1%)	**AFAP1L2**(96%)	*AGPAT5*(29%)
	AHNAK(2%)	*ALG12*(85%)	*ALKBH6*(10%)	*ALPK2*(22%)
	AMH(24%)	*AMIGO2*(29%)	*ANKH*(29%)	*ANKRD12*(13%)
	ANKRD22(77%)	**AP2A1**(99%)	*AP3B2*(15%)	*APH1B*(19%)
	APOL4(78%)	*ARHGEF6*(14%)	*ASB12*(16%)	*ASF1B*(29%)
	ASNA1(84%)	*ATG9A*(29%)	*ATP10B*(43%)	**ATP2C2**(97%)
	ATP8B3(8%)	*AUH*(43%)	**AXIN2**(96%)	*BAGE2*(19%)
	BCL2L10(1%)	*BCL9*(2%)	*BCR*(1%)	*BIN1*(2%)
	BMP8A(22%)	**BTN3A1**(78%)	*C10orf11*(22%)	**C11orf41**(82%)
	C12orf43(85%)	*C12orf52*(20%)	*C12orf62*(15%)	*C14orf129*(56%)
	C14orf37(2%)	*C14orf79*(19%)	*C16orf93*(8%)	*C17orf44*(56%)
	C17orf46(1%)	*C18orf8*(1%)	*C19orf36*(1%)	**C19orf55**(99%)
	C22orf26(84%)	**C2orf62**(97%)	*C3orf47*(76%)	**C3orf50**(95%)
	C4orf44(1%)	**C6orf163**(98%)	*C6orf59*(1%)	*C7orf55*(70%)
	C7orf59(1%)	*C8orf84*(15%)	*C9orf167*(13%)	**C9orf93**(99%)
	CAPRIN2(13%)	*CCDC102A*(1%)	*CCDC160*(8%)	*CCDC3*(70%)
	CCL13(29%)	*LILRA6*(1%)	*LMAN2*(1%)	*LOC100130691*(1%)
	LOC284232(1%)	*LOC441177*(1%)	*LOC613037*(1%)	*LOC81691*(1%)
	MAK(1%)	*MAPKAPK5*(1%)	*MCM3AP*(1%)	*MRPS10*(1%)
	MRPS18C(1%)	*MYT1*(1%)	*NDUFAF3*(1%)	*NEK2*(1%)
	NFKBIE(1%)	*NFKBIL1*(1%)	*NNT*(1%)	*NTN1*(1%)
	RAB34(1%)	*RAP1GAP2*(1%)	*RFK*(1%)	*RGS2*(1%)
	RHEBL1(1%)	*RNF121*(1%)	*RPL36*(1%)	*RSPO4*(1%)
	SALL4(1%)	*SCGN*(1%)	*SLA2*(1%)	*SLC41A1*(1%)
	SNF8(1%)	*SNRPA1*(1%)	*SOAT2*(1%)	*SOBP*(1%)
	SPHK2(1%)	*STAG3*(1%)		
coxCS (AIC = 1937.71 ±13.69)	**A1CF**(91%)	**A2BP1**(91%)	**A2ML1**(91%)	**A2M**(91%)
	A4GALT(91%)	**A4GNT**(91%)	**AAAS**(91%)	**AACSL**(89%)
	AACS(91%)	**AADAC**(89%)	**AADAT**(89%)	**AAGAB**(89%)
	AAK1(90%)	*AAMP*(73%)	**AARS2**(91%)	**AARSD1**(89%)
	AASDH(28%)	**AASS**(91%)	**AATF**(91%)	**AATK**(86%)
	ABCA10(91%)	**ABCA11P**(89%)	*ABCA12*(27%)	*ABCA13*(23%)
	ABCA1(50%)	**ABCA2**(91%)	**ABCA3**(89%)	**ABCA4**(91%)
	ABCA5(91%)	*CCDC109B*(1%)	*CCDC110*(1%)	*CCDC112*(1%)
	CCDC113(1%)	*CCDC114*(1%)	*CCDC115*(1%)	*CCDC116*(1%)
	CCDC117(1%)	*CCDC11*(1%)	*CCDC120*(1%)	*CCDC121*(1%)
	CCDC122(1%)	*CCDC123*(1%)	*CCDC125*(1%)	*CCDC126*(1%)
	CCDC12(1%)	*CCDC130*(1%)	*CCDC132*(1%)	*CCDC134*(1%)
	CCDC135(1%)	*CCDC137*(1%)	*CCDC138*(1%)	*CCDC13*(1%)
	CCDC144A(1%)	*CCDC144B*(1%)	*CCDC144C*(1%)	*CCDC146*(1%)
	CCDC147(1%)	*CYP4A22*(2%)	*CYP4B1*(2%)	*CYP4F12*(2%)
	CYP4F22(2%)	*CYP4F2*(2%)	*CYP4F3*(2%)	*CYP4V2*(2%)
	CYP4X1(2%)	*CYP4Z2P*(2%)	*CYP51A1*(2%)	*CYP7A1*(2%)
	CYP7B1(2%)	*CYP8B1*(1%)	*CYR61*(1%)	*CYS1*(2%)

	$CYSLTR1(2\%)$	$CYTH2(2\%)$	$CYTH3(2\%)$	$CYTH4(1\%)$
	$CYTIP(1\%)$	$CYTSA(1\%)$	$CYTSB(1\%)$	$CYYR1(1\%)$
	$D2HGDH(1\%)$	$D4S234E(1\%)$	$DAAM1(1\%)$	$DAAM2(1\%)$
	$DAB1(1\%)$	$HAVCR1(1\%)$	$HAVCR2(1\%)$	$HBA1(1\%)$
	$HBA2(1\%)$	$HBB(1\%)$	$HBD(1\%)$	$HBE1(1\%)$
	$HBEGF(1\%)$	$HBG1(1\%)$	$HBG2(1\%)$	$HBP1(1\%)$
	$HBS1L(1\%)$	$HBXIP(1\%)$	$HCFC1R1(1\%)$	$HCFC1(1\%)$
	$HCG18(1\%)$	$HCG22(1\%)$	$HCG26(1\%)$	$HCG27(1\%)$
	$HCG4P6(1\%)$	$HCG4(1\%)$	$HCG9(1\%)$	$HCN2(1\%)$
	$HCN3(1\%)$	$HCN4(1\%)$	$HCP5(1\%)$	$HCST(1\%)$
	$KNTC1(1\%)$	$KPNA1(1\%)$	$KPNA3(1\%)$	$KPNA4(1\%)$
	$KPNA5(1\%)$	$KPNA6(1\%)$	$KPNB1(1\%)$	$KPTN(1\%)$
	$KRAS(1\%)$	$KRBA1(1\%)$	$KRBA2(1\%)$	$KRCC1(1\%)$
	$KREMEN1(1\%)$	$KRI1(1\%)$	$KRIT1(1\%)$	$KRT13(1\%)$
	$KRT14(1\%)$	$KRT15(1\%)$	$KRT16(1\%)$	$KRT18(2\%)$
	$KRT19(2\%)$	$KRT1(2\%)$	$KRT23(2\%)$	$KRT24(2\%)$
	$KRT25(2\%)$	$KRT2(2\%)$	$KRT34(2\%)$	$PSMD8(2\%)$
	$PSMD9(2\%)$	$PSME2(2\%)$	$PSME3(2\%)$	$PSME4(2\%)$
	$PSMF1(1\%)$	$PSMG1(2\%)$	$PSMG2(2\%)$	$PSMG3(1\%)$
	$PSMG4(2\%)$	$PSORS1C1(2\%)$	$PSORS1C2(2\%)$	$PSORS1C3(2\%)$
	$PSPC1(2\%)$	$PSPH(1\%)$	$PSPN(1\%)$	$PSTPIP1(2\%)$
	$PSTPIP2(2\%)$	$PTAFR(2\%)$	$PTAR1(2\%)$	$PTBP2(2\%)$
	$PTCD1(2\%)$	$PTCD2(2\%)$	$PTCH2(2\%)$	$PTCHD1(2\%)$
	$PTCHD2(2\%)$	$PTCRA(2\%)$	$PTDSS1(2\%)$	$UBR5(2\%)$
	$UBR7(2\%)$	$UBTD2(2\%)$	$UBTFL1(2\%)$	$UBTF(2\%)$
	$UBXN10(2\%)$	$UBXN11(2\%)$	$UBXN1(2\%)$	$UBXN2A(2\%)$
	$UBXN2B(2\%)$	$UBXN4(2\%)$	$UBXN6(2\%)$	$UBXN7(2\%)$
	$UBXN8(2\%)$	$UCA1(2\%)$	$UCHL1(2\%)$	$UCK1(2\%)$
	$UCK2(2\%)$	$UCKL1AS(2\%)$	$UCKL1(2\%)$	$UCN3(2\%)$
	$UCN(2\%)$	$UCP2(2\%)$	$UCP3(2\%)$	$UFC1(2\%)$
	$UFD1L(2\%)$	$UFM1(2\%)$	$UFSP1(2\%)$	$UFSP2(2\%)$

Table 5: Résultats des méthodes de *Screening* sur l'ensemble de gènes

A.3 Getting biological knowledge for coxCS method

The CoxCS method uses biological knowledge to make a pre-selection. For this pre-selection, The proposed idea to use the latest publications referencing predictive biomarkers about patient survival. For this, we performed a PubMed search using the following keywords and MeSH terms (for *Medical Subject Headings*): `ccRCC [tiab] prognosis [tiab] survival [tiab] genes NOT RNA` MeSH is a hierarchically organized and controlled vocabulary produced by the *National Library of Medicine*. We thus obtained 32 results corresponding to our search, and we decided to keep the first 10 *abstracts*. These 10 *abstracts* were saved and later loaded into the web application `beca annotate` nunesbecas2013. The purpose of this web application is the identification and annotation of medical concepts in a text. The identification and annotation of genes and proteins performed by using *machine learning* methods present in Gimli camposgimli2013. Gimli camposgimli2013 is a *open-source* tool for automatic recognition of biomedical terms. The list of genes obtained is: $KDM2B$, $HMGCS2$, $HSD11B1$, $IL10$, $KDM5B$, $KDM5A$, $KDM5C$, $KDM5D$, $KDM1B$, $OGDHL$, $SSBP2$, $VSIG4$ and $XCR1$. We decided to use this list as biological knowledge for pre-selection in coxCS.

Cancer Immunology, Immunotherapy (2020) 69:1237–1252
https://doi.org/10.1007/s00262-020-02530-x

ORIGINAL ARTICLE

Comprehensive landscape of immune-checkpoints uncovered in clear cell renal cell carcinoma reveals new and emerging therapeutic targets

Diana Tronik-Le Roux[1,2,7] · Mathilde Sautreuil[3] · Mahmoud Bentriou[3] · Jérôme Vérine[1,4] · Maria Belén Palma[5] · Marina Daouya[1,2] · Fatiha Bouhidel[4] · Sarah Lemler[3] · Joel LeMaoult[1,2] · François Desgrandchamps[1,6] · Paul-Henry Cournède[3] · Edgardo D. Carosella[1,2]

Received: 26 June 2019 / Accepted: 18 February 2020 / Published online: 12 March 2020

Abstract
Clear cell renal cell carcinoma (ccRCC) constitutes the most common renal cell carcinoma subtype and has long been recognized as an immunogenic cancer. As such, significant attention has been directed toward optimizing immune-checkpoints (IC)-based therapies. Despite proven benefits, a substantial number of patients remain unresponsive to treatment, suggesting that yet unreported, immunosuppressive mechanisms coexist within tumors and their microenvironment. Here, we comprehensively analyzed and ranked forty-four immune-checkpoints expressed in ccRCC on the basis of in-depth analysis of RNAseq data collected from the TCGA database and advanced statistical methods designed to obtain the group of checkpoints that best discriminates tumor from healthy tissues. Immunohistochemistry and flow cytometry confirmed and enlarged the bioinformatics results. In particular, by using the recursive feature elimination method, we show that HLA-G, B7H3, PDL-1 and ILT2 are the most relevant genes that characterize ccRCC. Notably, ILT2 expression was detected for the first time on tumor cells. The levels of other ligand-receptor pairs such as CD70:CD27; 4-1BB:4-1BBL; CD40:CD40L; CD86:CTLA4; MHC-II:Lag3; CD200:CD200R; CD244:CD48 were also found highly expressed in tumors compared to adjacent non-tumor tissues. Collectively, our approach provides a comprehensible classification of forty-four IC expressed in ccRCC, some of which were never reported before to be co-expressed in ccRCC. In addition, the algorithms used allowed identifying the most relevant group that best discriminates tumor from healthy tissues. The data can potentially assist on the choice of valuable immune-therapy targets which hold potential for the development of more effective anti-tumor treatments.

Keywords HLA-G · Transcriptome · Immune-checkpoints · RNAseq · TCGA · ccRCC

Abbreviations

ccRCC	Clear cell renal cell carcinoma
HLA-G	Human leukocyte antigen-G
IC	Immune-checkpoints
ILT2	Immunoglobulin-like transcript 2 (LILRB1)
ILT4	Immunoglobulin-like transcript 4 (LILRB2)

Mathilde Sautreuil and Mahmoud Bentriou have contributed equally to this work.

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s00262-020-02530-x) contains supplementary material, which is available to authorized users.

✉ Diana Tronik-Le Roux
diana.le-roux@cea.fr

1 Commissariat à L'Energie Atomique Et Aux Energies Alternatives (CEA), Direction de La Recherche Fondamentale (DRF), Service de Recherche en Hémato-Immunologie (SRHI), Hôpital Saint-Louis, Paris, France

2 Université de paris, U976 HIPI Unit, Institut de Recherche Saint-Louis, 75010 Paris, France

3 Laboratory of Mathematics and Informatics (MICS), CentraleSupélec, Université Paris-Saclay, 91190 Gif-sur-Yvette, France

4 Service D'Anatomo-Pathologie, AP-HP, Hôpital Saint-Louis, Paris, France

5 Cátedra de Citología, Histología Y Embriología A, Facultad de Ciencias Médicas, UNLP, Buenos Aires, Argentina

6 Service D'Urologie, AP-HP, Hôpital Saint-Louis, Paris, France

7 CEA, Direction de La Recherche Fondamentale, Service de Recherche en Hémato-Immunologie, Hôpital Saint-Louis, IUH, 1, avenue Claude Vellefaux, 75010 Paris, France

RFEM	Recursive feature elimination method
SVM	Support vector machine
TCGA	The Cancer Genome Atlas

Introduction

Clear cell renal cell carcinoma (ccRCC) is a renal tumor typically characterized by malignant epithelial cells with clear cytoplasm and a compact alveolar or acinar growth pattern interspersed with intricate, arborizing vasculature. Among kidney cancers, 70% are estimated to be ccRCC [1] constituting the most common renal cell carcinoma subtype. Patients cannot be treated by radiation or chemotherapy. Metastases at the diagnosis time are observed for 25–30% of the patients and less than 10% of these survive more than five years. Recurrence occurs in 20–30% of patients even after complete nephrectomy of primary tumors. Genetic evidence revealed that ccRCC originates from sequential losses of multiple tumor suppressor genes such as *von Hippel-Lindau (VHL)* gene. Other genes such as *PBRM1* (Polybromo 1) and *BAP1* (BRCA1-associated protein) have also been identified as renal cancer driver genes, mainly underexpressed in ccRCC [2]. These three genes are closely located on chromosome 3p. Remarkably, chromosome 3p loss occurs in more than 90% of sporadic ccRCCs [3].

ccRCC has long been recognized as an immunogenic cancer. As such, significant attention has been directed toward optimizing immune-checkpoints (IC)-based therapies. IC are defined as cell surface molecules that transduce positive or negative signals, into effector cells. When the same ligand interacts with an activating or an inhibitory receptor, inhibition is dominant and allows tumor cells to escape destruction by the immune system [4]. Therefore, blockade of inhibitory IC by antibody-based therapeutics constitutes a suitable strategy to restore an effective anti-tumor response [5, 6].

Clinical phase I/II trials have provided evidence that treatments with anti-CTLA-4 or PD-L1:PD-1 antibodies are capable of restoring suitable anti-tumor responses and that therapies using combinations of anti-CTLA-4 and PD-L1:PD-1 are more efficient than using each one individually [7–9]. Although these trials have helped to give a step forward in the treatment of some patients, others still remain unresponsive. Potential causes might be the complexity of IC pathways since several yet undescribed checkpoints, besides CTLA-4 and PD-1:PD-L1, might be expressed simultaneously and act concomitantly through non-overlapping, sometimes opposite, mechanisms. In addition, other causes of the inefficiency of the treatments might be the heterogeneity of expression within tumor cells or the toxicity due to their action on non-tumor tissues. Consequently, to achieve optimal specificity and limit potential side effects, IC targeted by immune-therapies must have restricted expression in normal tissues and high expression upon cellular transformation.

In this context, the search for novel IC targets has been intensified. Ongoing trials include antibody-based protocols against LAG-3, TIM-3 and VISTA [10] or agonist antibody-based protocols for co-stimulatory molecules ICOS, OX40 and 4-1BB [11]. Some of these molecules are evaluated in mono or combined therapies. However, the uncomplete list of their side effects and the lack of knowledge about their mechanisms of action prevent their immediate clinical utilization.

The advent of next-generation sequencing technologies, particularly applied to RNAseq, has demonstrated its usefulness in classifying patients with similar pathologies, screening genes involved in carcinogenesis and in identifying prognostic factors in cancer [12–14]. This technique also revealed inter- and intra-tumor heterogeneity which challenges the precise diagnosis and selection of reliable therapeutic targets [15]. The full exploration of large and growing public databases such as The Cancer Genome Atlas (TCGA) data portal (https://tcga-data.nci.nih.gov/tcga/tcgaHome2.jspa) which covers RNAseq transcriptome data for more than 10,000 samples of human cancers including over 30 different cancer types, holds the potential to help identify suitable therapeutic targets.

Many statistical tools were developed to analyze RNA-seq data [16]. Comparison of these different tools[16] singled out DESeq2 as the most conservative [17]. However, several statistical challenges must still be faced in order to unravel molecular processes and interactions at stake for each cancer type.

Here, we provide a novel and comprehensible view on forty-four IC expressed in ccRCC and highlight the most relevant group of checkpoints that characterize ccRCC and the ones that best discriminate tumor from healthy tissues. The data can potentially assist on the choice of valuable immune-therapy targets which hold potential for the development and effectiveness of anti-tumor treatments.

Materials and methods

TCGA data collection and processing

RNA-sequencing profiles of ccRCC samples and their related clinical data obtained from The Cancer Genome Atlas (TCGA) data portal (https://tcga-data.nci.nih.gov/tcga/) were used to identify expressed IC. A set of 25 283 RNA-seq derived from the 539 patients with ccRCC was downloaded. These were normalized by the FPKM-UQ method that scales the number of reads of each gene compared to the total number of reads and the length of genes.

Statistical methods used for the differential analysis

The differential analysis was conducted on the RNA-seq data using DESeq2 [18] which is a statistical method implemented in an R software package. DESeq2 allows identifying differentially expressed genes using the standard comparison mode between two experimental conditions. A high variability between patients exists in RNA-seq data. Therefore, to take into account this variability and accurately analyze these data, it is mandatory to use a distribution that incorporates over-dispersion as does the negative binomial distribution. DESeq2 takes into account this variability by using Wald test based on a negative binomial distribution to test whether each model, tumor or healthy cells, differs significantly from zero. It models counts by a negative binomial distribution because counts are over-dispersed (the variance is higher than the mean), and the negative binomial distribution has two parameters called mean and dispersion which enable to model the mean–variance relationship. A scaling factor normalization procedure is performed to take into account the varying sequencing depths of the different samples. The generalized linear models (GLMs) with a logarithmic link are used to estimate coefficients that indicate the overall expression strength of the gene. The p-value adjustment for multiple tests was performed by the Benjamini–Höchberg procedure [19], which consists in controlling the false discovery rate (FDR). We considered differentially expressed genes those with an adjusted p value ≤ to 0.05.

Feature selection method

Supervised learning is a powerful tool for mathematical analysis of biological data. Here, we consider the following classification task: "how well can I predict if a tissue is tumoral or not with gene expression data?" In this section, our goal is to rank IC genes according to their powers of prediction: We want to find the smallest and the best subset of genes that realizes this classification task. To achieve this, we consider the recursive feature elimination (RFE) algorithm [20] with a linear-support vector machine (linear-SVM). This algorithm selects features by recursively considering smaller and smaller sets of features and assigns weights to the features directly linked to the coefficients of the linear model. A subset of the ranked genes is selected according to the SVM-IC [21]. The details of the method are explained below.

Dataset and tools

Python and scikit-learn were used [22]. The features are normalized (FPKM-UQ normalization) gene expressions of the differentially expressed genes ($d = 41$) in order to use well-known gene selection techniques for micro-array-based data [23]. The data are scaled with zero mean and variance one for better performance with the linear-SVM (see below for details of this algorithm).

Linear Support Vector Machine training

We focused on a binary classification model. We want to build a classifier f:

$$\begin{aligned} f : \mathbb{R}^d &\rightarrow \{-1, 1\} \\ (x_1, \dots, x_d) &\rightarrow y \end{aligned}$$

where $\left(x_1, \dots, x_d\right)$ is the gene expression data of a tissue sample and the target value is the type of the tissue: non-tumor adjacent or tumor.

Support vector machine with a linear kernel was chosen for the classification learning algorithm since the data are almost linearly separable. Also, linear decision functions capture very well the underlying distributions in micro-array classification tasks [23–25]. The goal is to find the optimal parameters of a classifier function that as the form:

$$\begin{aligned} f : \mathbb{R}^d &\rightarrow \{-1, 1\} \\ x &\rightarrow \operatorname{sign}(w^t x + b) \end{aligned}$$

We define the dataset as $\left\{\left(x_i, y_i\right)\right\}_{\{1 \le i \le n\}}$ where $x_i = \left(x_{i1}, \dots, x_{id}\right)$ is the gene expression data of the sample **i** and y_i its label (non-tumor adjacent tissue or tumor tissue). For each gene $j \in \{1, \dots, d\}$, a weight w_j is involved in the classification task. This weight is used for the ranking of genes. The linear-SVM learning algorithm can be defined as an optimization problem:

$$\begin{aligned} \underset{w,b,\xi}{\text{minimize}} \quad & \frac{1}{2}\|w\|^2 + C \sum_{i=1}^{n} \xi_i \\ \text{subject to} \quad & \forall i \in \{1, \dots, n\}, y_i(w^t x_i + b) \ge 1 - \xi_i \\ & \forall i \in \{1, \dots, n\}, \xi_i \ge 0. \end{aligned}$$

Recursive feature elimination (RFE) algorithm

The RFE method recursively deletes the genes that are the less important in the classification task. At an iteration $k \in \{1, \dots, d\}$, d minus k genes are left to rank. At each iteration, we train the linear SVM classifier used in the RFE method over the whole dataset. Each gene has a weight associated to this classifier in the decision making whether to distinguish if a tissue is tumoral or not. The square of this weight is the importance of a gene: It shows how the value of the gene expression is important in the decision making. The linear-SVM is trained over these genes, and the gene that has the least importance cj is deleted in the classification task $(cj = Wj^2)$.Let X be the matrix of the dataset

$\{(x_i, y_i)\}_{\{1 \le i \le n\}}$ (where **i** is the patient) so that $X[i, :] = x_i$ (the vector of gene expression data of patient **i**) and $X[:, j]$ the vector of expression data of gene **j** among all the patients. Let $y = (y_i)_i$ be the vector of the labels. The RFE algorithm is detailed below.

```
Algorithm 1 RFE algorithm
Require: $X$, $y$
Ensure: A set of ranked genes $G = [\text{gene}_1, \dots, \text{gene}_d]$
  Initialization : $S \leftarrow [1, \dots, d]$, $G \leftarrow \emptyset$
  while $S \neq \emptyset$ do
    $X_{\text{current}} \leftarrow X[:, S]$
    Get $w$ from linear-SVM training on the dataset $X_{\text{current}}, y$
    $\forall j \in [1, \dots, d], c_j \leftarrow w_j^2$
    $j_{\min} \leftarrow \arg\min_j c_j$
    $S \leftarrow S \backslash \{j_{\min}\}$
    $G \leftarrow \text{concat}(j_{\min}, G)$
  end while
```

The set of ranked genes G is then obtained. We want to compare the classifiers trained on a subset of features G_k in order to get the best subset of genes:

$$G_k = [\text{gene}_1, \dots, \text{gene}_k], \quad \forall k \in \{1, \dots, d\}$$

where gene_1 is the most important gene and gene_d the least important one. To compare these classifiers, an information criterion is performed [21] which can be seen as an equivalent of Akaike's information criterion for linear-SVM. In the framework of the linear-SVM learning algorithm, one can define the SVM information criteria as: $\text{SVMIC}(G_k) = \sum_{i=1}^{n} \xi_i + 2\text{Card}(G_k)$ where the ξ_i are the coefficients learned by the linear-SVM algorithm over $X[:, G_k]$ and $\text{Card}(G_k)$ is the length of the subset. In this case, $\text{Card}(G_k) = \text{k}$. The smaller this criterion is, the better the classifier is. So, we chose the best subset of features with: $G_{k_{\text{best}}} = \min_{G_k} \text{SVMIC}(G_k)$. This subset is the best compromise between classification performance and smallness of the subset in order to get the most significant genes.

Recovery of tumors from ccRCC patients

Tumors were obtained from patients who underwent a radical nephrectomy for ccRCC as first therapeutic intervention in the Urology Department of Saint-Louis Hospital (Paris, France). Renal tumors were classified as ccRCC by an experienced uropathologist according to the World Health Organization (WHO) classification of tumors of the kidney [26]. Adjacent non-tumor renal parenchyma was removed simultaneously. All patients participating in this study gave their informed written consent. The study was approved by the institutional review boards of Saint-Louis Hospital, Paris.

Immunohistochemistry (IHC)

Immunohistochemistry was performed for each pair of tumor and normal renal parenchyma on 4-µm-thick, formalin-fixed and paraffin-embedded or on snap-frozen tissue sections according to technical specificities of each antibody. The following murine antibodies were used: CD70, B7H4, HVEM, B7H5 (VISTA), CD40, CD163, HLA-G (clone 4H84) and PD-L1. Staining was performed on automated slide stainers from Roche (BenchMark ULTRA system, Tucson, AZ) using the OptiView DAB IHC Detection Kit (Roche), Cell Conditioning 1 (CC1) standard antigen retrieval, an antibody incubation time of 32 min at 37 °C, ultraWash procedure, counterstaining with Hematoxylin II for 4 min, and bluing reagent for 8 min. Isotype-matched immunoglobulins were used for negative controls simultaneously. The immunohistochemical analyses were finally realized by an uropathologist using a BX51 microscope (Olympus France S.A.S, Rungis).

Primary tumor cell culture

Tumor tissues were cultured in RPMI1640 supplemented with 20% fetal bovine serum, 20 µg/mL insulin, 10 µg/mL transferrin, 25 nM sodium selenite, 50 nM hydrocortisone, 1 ng/mL epidermal growth factor, 10 µM ethanolamine, 10 µM phosphorylethanolamine, 100 pM triiodothyronine, 2 mg/mL bovine serum albumin, 2 mM glutamine, 0.5 mM sodium pyruvate. After 21 days, medium was removed, tumor cells detached with EDTA 2 mM solution and analyzed by flow cytometry.

Flow cytometry analysis

Flow cytometry analyses were performed on tumor and tumor-infiltrating cells. Tumor-infiltrating cells were phenotyped immediately after isolation, whereas tumor cells were phenotyped after 3-week culture. This culture step was necessary to ensure sufficient cell numbers for flow cytometry analysis, even though some phenotype modifications by the in vitro culture might arise. Acquisition was performed on a MACSQuant 10 flow cytometer (Miltenyi Biotec), and analysis was performed using MACSQuantify (Miltenyi Biotec) and Flowjo softwares. Antibodies were CD45 VioGreen, CD3 PerCP, CD4 VioBright FITC, CD8 APC-Vio770, HLA-DR PerCP, CD28 PE-Vio770, CD137 PE-Vio770, B7-H3 VioBlue, ILT4 APC, CD70 APC, PD-L1 APC, CD137L PE from Miltenyi Biotec; HLA-G PE (clone MEM-G/09), ILT2 APC and ILT2PE (clone HP-F1) from eBioscience. PD-1 BV421 from Biolegend and CD27 PE, CD40 PE, CD86 FITC from Beckman Coulter.

Results

Landscape of IC expressed in ccRCC

To provide a global landscape of IC expressed in ccRCC patients, we analyzed the TCGA database which contains RNAseq information for 539 patients with ccRCC. To avoid selecting biased expression thresholds, we further focused on patients for whom data included tumor and adjacent non-tumor tissues. This information was available for 72 patients and is depicted in Table 1. A total of forty-four IC were chosen on the basis of a previously report [27].To test our methodology, three tumor suppressor genes expressed at very low levels in most ccRCC were added as controls: the von Hippel-Lindau (*VHL*), the Polybromo 1 (*PBRM1*) and the BRCA1-associated protein (*BAP1*).

Expression ratios for tumor versus healthy adjacent tissues were first analyzed for each of the forty-four IC by bi-clustering genes for each of the 72 patients individually. The results are represented on a heatmap, a colored representation of a matrix of numbers that helps the visualization of gene expression data (Fig. 1). In this representation, each column represents a gene, each row a patient. The standardized expression levels are depicted by the color gradients. The results show perceptible differences between patients even though a global pattern emerges. Notably, the ratio of expression of tumor to adjacent non-tumors for B7H4 was found extremely low except for 2 out of 72 samples. The three control genes *PBRM1, VHL* and *BAP1* that are mostly down-regulated in ccRCC were grouped together, and their low expressions in ccRCC samples were confirmed (Fig. 1).

Statistically modulated IC in ccRCC

To establish statistically robust results, we chose to conduct a differential analysis to compare the expression levels of IC in ccRCC samples versus the adjacent non-tumor tissues using the DESeq2. A set of 1264 genes were found up-regulated and 1194 genes down-regulated, by considering a *p*-value of 0.05. The similar numbers of induced and repressed genes is consistent with an absence of methodological biases. From these, we extracted expression information for the forty-four IC simultaneously. The results obtained are represented by bar plots (Fig. 2). The x-axis represents the value of the log fold change and the y-axis the genes. A gradient of color is used in order to show the significance of genes in the study. The redder the bar, the more differentially expressed the gene.

Moreover, the genes on the *y*-axis are ranked according to their adjusted *p*-values. The lower the adjusted *p*-value, the lower the position of the associated gene on the axis, and the more significant the differential expression. If the value of the log fold change is negative, the gene is underexpressed, and if it is positive, the gene is overexpressed. Values representing the mean of normalized counts for all samples, the log2 FoldChange, the *p* values and the adjusted *p*-values of the test are represented in supplementary Table 1.

Notably, among the most significantly differentially overexpressed genes in tumor cells we found those encoding ligand-receptor pairs such as: CD70:CD27; HLA-G:ILTs; 4-1BB:4:1BBL, CD40:CD40L; CD86:CTLA4; MHC-II:Lag3; CD200:CD200R; CD244:CD48. In contrast, adjacent non-tumor cells express significantly higher levels of B7-H4 compared to tumor cells. As expected, the expression levels of the three control genes: *PBRM1, VHL* and *BAP1,* were lower in tumors than in adjacent non-tumor tissues, which further confirms the robustness of our statistical analysis.

Localization of representative IC within tumor compartments.

To deepen the results of the TCGA analysis, we aimed at identifying more precisely the nature of the IC-expressing cells. To this end, we performed IHC analysis on different samples derived from a new cohort of ccRCC patients that underwent a radical nephrectomy at the Urology Department of Saint-Louis Hospital (Paris, France).

Tumor and normal renal parenchyma of ten patients were assessed simultaneously. We focused on B7-H4, the most significantly down-regulated gene and three up-regulated genes in tumor: CD70, CD40 and B7-H5 (VISTA). When samples were analyzed with the antibody directed against B7H4, strong labeling of the normal renal parenchyma of all renal tubular epithelial cells was observed in all cases. On

Table 1 ID and clinical information obtained from The Cancer Genome Atlas (TCGA) for the 72 patients for whom data included tumor and adjacent non-tumor tissues

Patient ID	Case ID	Sex	Year of birth	Year of death	Tumor stage	Age at diagnosis (days)	Days to death	Days to last follow up
TCGA-CW-5581	2d0f6d4f-acb9-4b45-a69d-c9a3f68c3732	Male	1959	–	Stage I	16,195	–	2799
TCGA-CZ-4865	305eaef4-4644-46e3-a696-d2e4a972f691	Female	1936	2006	Stage I	25,709	166	–
TCGA-B0-5691	31a0dd95-8199-4d25-b40d-4b353644af46	Female	1936	–	Stage I	24,120	–	3431
TCGA-CW-6088	4100b960-7963-4a1e-ba24-0a6526387e06	Male	1943	–	Stage I	22,020	–	3222
TCGA-CW-6090	4dd51edf-05d1-445a-b43c-6fce29eb21d4	Male	1935	–	Stage I	24,909	–	2552
TCGA-B0-5703	576ea0ef-3abb-479d-adb7-ba646cee344e	Male	1936	–	Stage I	26,897	–	2246
TCGA-CW-5589	58ef1a13-a549-4043-b66c-5327bfcbd2e6	Male	1952	–	Stage I	19,312	–	2378
TCGA-B0-5705	62cf9546-b932-4a5e-bc9b-4103506d296d	Female	1937	–	Stage I	23,746	–	4537
TCGA-CJ-6030	6ab00314-5228-43ba-9880-9d5177b64c61	Male	1938	2009	Stage I	24,096	2299	–
TCGA-B8-4619	78ec8bc9-e502-4f5b-b0f6-893,718,650,352	Male	1952	–	Stage I	21,206	–	523
TCGA-B0-5690	794aeb92-205f-4c75-bb1a-0b2c6db99fea	Female	1951	–	Stage I	19,685	–	3392
TCGA-CZ-5986	883c37e2-c723-45d7-9e25-082d1bec34e2	Male	1945	–	Stage I	22,456	–	373
TCGA-A3-3358	8a575e00-5dc5-416d-a8f5-2cdfb8e62c31	Female	1948	–	Stage I	21,087	–	1307
TCGA-B8-5549	98dea82b-46f9-4b77-be8a-06669bcf731b	Male	1957	–	Stage I	19,561	–	194
TCGA-A3-3387	9dc7812b-c7a2-4de4-bf6d-4c7261384a62	Male	1957	–	Stage I	17,920	–	617
TCGA-CZ-5982	a43401ae-14cd-4f57-946b-9a9c073f2a47	Female	1946	–	Stage I	21,592	–	2439
TCGA-B2-5641	a682a2a3-dc53-4b7e-8929-07215737ec5e	Male	1931	–	Stage I	28,953	–	656
TCGA-B0-5699	c814c26c-ee8e-4fd8-a3d3-441b302ead3b	Male	1950	–	Stage I	19,647	–	3841
TCGA-B0-5697	dfd2c288-5054-4fc1-9cae-f437779dc2de	Male	1957	–	Stage I	18,486	–	2630
TCGA-B8-5552	e58f2c7b-d1cc-48f3-a0c9-dcd2b27c37f6	Female	1969	–	Stage I	15,219	–	1046
TCGA-CJ-5689	e865d40a-9989-436c-8426-88cc84c863e8	Male	1915	2009	Stage I	32,872	1620	–
TCGA-CZ-5984	ea26d89f-2843-489b-8627-3d5ed0cabc2a	Male	1954	–	Stage I	18,775	–	2067
TCGA-B2-5636	ed2e9354-5ee1-4fcf-92ec-a1b507818b91	Male	1931	–	Stage I	28,901	–	919
TCGA-CJ-5672	eee108d6-bb11-4369-b776-0f524afef6f3	Male	1920	2010	Stage I	30,900	2190	–
TCGA-CZ-5988	f2806652-1d7b-4c46-bb4e-ac7a2c96c68d	Male	1968	–	Stage I	14,131	–	693
TCGA-CZ-5989	13e25128-9be1-4f67-a43f-a8744a619203	Male	1946	–	Stage II	22,170	–	1905

Table 1 (continued)

Patient ID	Case ID	Sex	Year of birth	Year of death	Tumor stage	Age at diagnosis (days)	Days to death	Days to last follow up
TCGA-B0-5706	3b2b492b-94af-4540-b283-3b9ef98d5b2f	Male	1959	–	Stage II	16,442	–	3205
TCGA-CZ-5456	467bf226-a646-4217-bce6-8d0f11c756eb	Male	1949	–	Stage II	21,164	–	2422
TCGA-CZ-5985	4c474c70-1d72-49e8-a8cc-a4145c5ab607	Male	1948	–	Stage II	21,242	–	1997
TCGA-CZ-5453	577847b9-f9b6-4954-868f-3d5ab2b4f694	Male	1939	–	Stage II	24,640	2419	25
TCGA-CZ-5469	6205c2d8-d431-4c4c-8ac6-ee407170e833	Male	1966	2009	Stage II	15,242	946	–
TCGA-CZ-4864	7fc6b44d-ae66-409c-98e1-c1a518c33e95	Male	1916	2009	Stage II	31,557	2830	2830
TCGA-CZ-5463	8e9e684c-20e5-48b1-9e40-970037fe959f	Male	1931	–	Stage II	27,911	–	662
TCGA-CZ-5470	99f59583-2728-4c1d-b98c-8fbc3bb0a819	Female	1936	–	Stage II	26,494	–	386
TCGA-CZ-5451	9cdda9fa-d492-4902-afc1-eb2244e70c1b	Male	1932	–	Stage II	27,312	–	1929
TCGA-CZ-5452	e9faa588-d45a-450a-81a2-949eb2834bc0	Male	1937	–	Stage II	25,486	–	1789
TCGA-CW-5584	09c4ea05-928d-49b7-b7fb-30cff3481b14	Male	1929	2003	Stage III	27,352	164	–
TCGA-B0-5694	11111b58-c7df-4291-ad8a-4baec9ff7d1f	Male	1937	2009	Stage III	26,060	480	–
TCGA-B0-5696	1d176c53-6cbb-4a39-9a6f-669b4f1cb575	Male	1938	–	Stage III	25,552	–	2609
TCGA-CJ-5676	310c31bf-91ce-40b4-99e4-a10242296de9	Male	1957	–	Stage III	17,170	–	4067
TCGA-CJ-5679	3fa6c93e-e7fe-402c-9526-c81411aa0920	Male	1930	2004	Stage III	26,858	679	–
TCGA-CZ-5466	5722df9f-5631-476d-a11b-b3c1e9a40fbf	Male	1940	–	Stage III	24,669	–	685
TCGA-CZ-5465	5e248b21-e69d-4dee-a1e8-029ade80d0eb	Female	1931	–	Stage III	27,838	2564	1446
TCGA-CW-5587	73fc6ae6-7c5e-44de-a9a0-17292cbb01cc	Female	1941	–	Stage III	22,693	–	2226
TCGA-B0-5711	88c91a7b-5c41-4361-85d0-da759ab94204	Male	1954	–	Stage III	18,444	–	3989
TCGA-CZ-4863	88e7ce26-5b3f-4e4e-89b7-f706063fc467	Female	1955	–	Stage III	18,730	–	1928
TCGA-B0-5701	8f8a632d-7fbd-4c86-b909-afb7edb9ad28	Male	1942	–	Stage III	23,915	–	2461
TCGA-CZ-5467	b602a73b-809c-44c6-a787-d496705e7ae8	Female	1921	2007	Stage III	31,473	73	–
TCGA-B0-5709	bf768635-b809-4df4-a4e6-7495e66aa227	Female	1941	–	Stage III	22,843	–	3974
TCGA-CZ-5458	c4247ccb-8201-428e-a2e7-b5104f095588	Male	1963	–	Stage III	15,790	–	2789
TCGA-B8-4620	eda2b871-8a18-4854-92d8-d8d66fb127d8	Female	1940	–	Stage III	25,786	–	777
TCGA-CZ-5457	f00b7956-73a9-4e4b-85d2-60b3ba13f4a4	Male	1944	–	Stage III	22,763	–	2754

Table 1 (continued)

Patient ID	Case ID	Sex	Year of birth	Year of death	Tumor stage	Age at diagnosis (days)	Days to death	Days to last follow up
TCGA-CJ-5681	2939c03a-6f3f-4f7b-b246-34361baadeb9	Female	1959	2004	Stage IV	16,121	552	–
TCGA-CZ-5461	393abcf7-1155-4b32-85ca-cd44d259e4b4	Male	1954	2006	Stage IV	19,030	330	330
TCGA-B0-4712	3f72d63f-ad48-4500-baf6-897b1d6dda7d	Male	1930	2009	Stage IV	28,095	1337	–
TCGA-CJ-6033	4edff57f-4b0e-4770-beac-590da7d7232c	Female	1950	2004	Stage IV	19,919	224	–
TCGA-B0-5712	514af471-31d2-43fa-88dc-8639a5e97181	Female	1936	–	Stage IV	24,988	–	2722
TCGA-CJ-5680	5db69ebe-38db-4ef3-b825-674b4a6ddaee	Female	1938	2005	Stage IV	23,778	768	–
TCGA-CZ-5455	74749fe8-f5d0-4d0d-8c7a-e56eba6503b3	Male	1943	2007	Stage IV	23,150	561	–
TCGA-CJ-5678	822cf6c1-dd65-4814-94b1-0c335208ad9b	Male	1941	2004	Stage IV	22,944	574	–
TCGA-CZ-5468	878d1caa-c4d8-4864-888c-310a0c1ee898	Male	1923	2007	Stage IV	30,729	59	59
TCGA-CW-5580	88fc4bc4-32cf-4d92-8c29-20d920b8f719	Female	1930	2008	Stage IV	26,696	1964	–
TCGA-B8-4622	b125ff14-4fb4-4f90-8622-fed49bdfe954	Male	1953	–	Stage IV	21,135	–	1525
TCGA-CW-6087	bbdaa931-e922-49be-bfbf-fa0c2ae27d7a	Male	1942	2003	Stage IV	22,438	41	–
TCGA-CJ-5677	d2664fde-ce3b-45e6-9a23-4a07980f7bac	Female	1950	2006	Stage IV	19,849	782	–
TCGA-B0-5402	d7ab7ec0-3de7-4ffd-a5ac-f75579355b2a	Male	1946	–	Stage IV	23,477	–	1290
TCGA-CZ-5462	e0127e51-43ba-4536-bc9d-004591f9c627	Male	1924	2007	Stage IV	30,659	311	–
TCGA-B0-4700	e33dff22-5bf1-4dd8-bed7-063cb555677c	Male	1944	2009	Stage IV	22,034	1980	–
TCGA-CW-5591	f1ae0181-74f8-47fd-83be-83a7d01101cc	Male	1948	–	Stage IV	20,712	–	2271
TCGA-CZ-5454	f2801b21-5444-4cc3-a642-c60c6d82cd3d	Male	1943	2007	Stage IV	23,083	722	–
TCGA-CZ-5987	f5759059-c0e3-4f1a-af96-5c7197d3c33c	Male	1946	2007	Stage IV	21,928	445	–
TCGA-CW-5585	0487fc41-386c-4d76-9084-daf959bf5e98	Male	1952	–	Stage IV	18,898	–	2609

the other hand, a very heterogeneous labeling was observed within and between ccRCC samples. In the majority of ccRCC tumors, no labeling was noted in both tumor and tumor-infiltrating inflammatory cells. In the other cases, a slight labeling was noted in tumor cells and/or tumor-infiltrating inflammatory cells. This confirms further the down-regulation of B7H4 highlighted in the RNAseq analysis. A representative example is illustrated in Fig. 3. When antibodies directed against the three up-regulated genes CD70, CD40 and B7-H5 (VISTA) were used, tumors were strongly stained when compared to normal renal parenchyma which further confirms the transcriptome results. Nevertheless, the supplementary information given by the morphologic analysis concerns the nature of cells that highly express these checkpoints. Indeed, IHC results showed that tumor cells themselves expressed high levels of CD70 whereas only rare parietal cells of Bowman's capsule, some interstitial fibrous areas and glomerular capillary vessels were labeled with anti-CD70 in normal renal parenchyma. In sharp contrast, CD40 and B7-H5 (VISTA) antibodies intensely labeled numerous inflammatory cells in contact with tumor cells

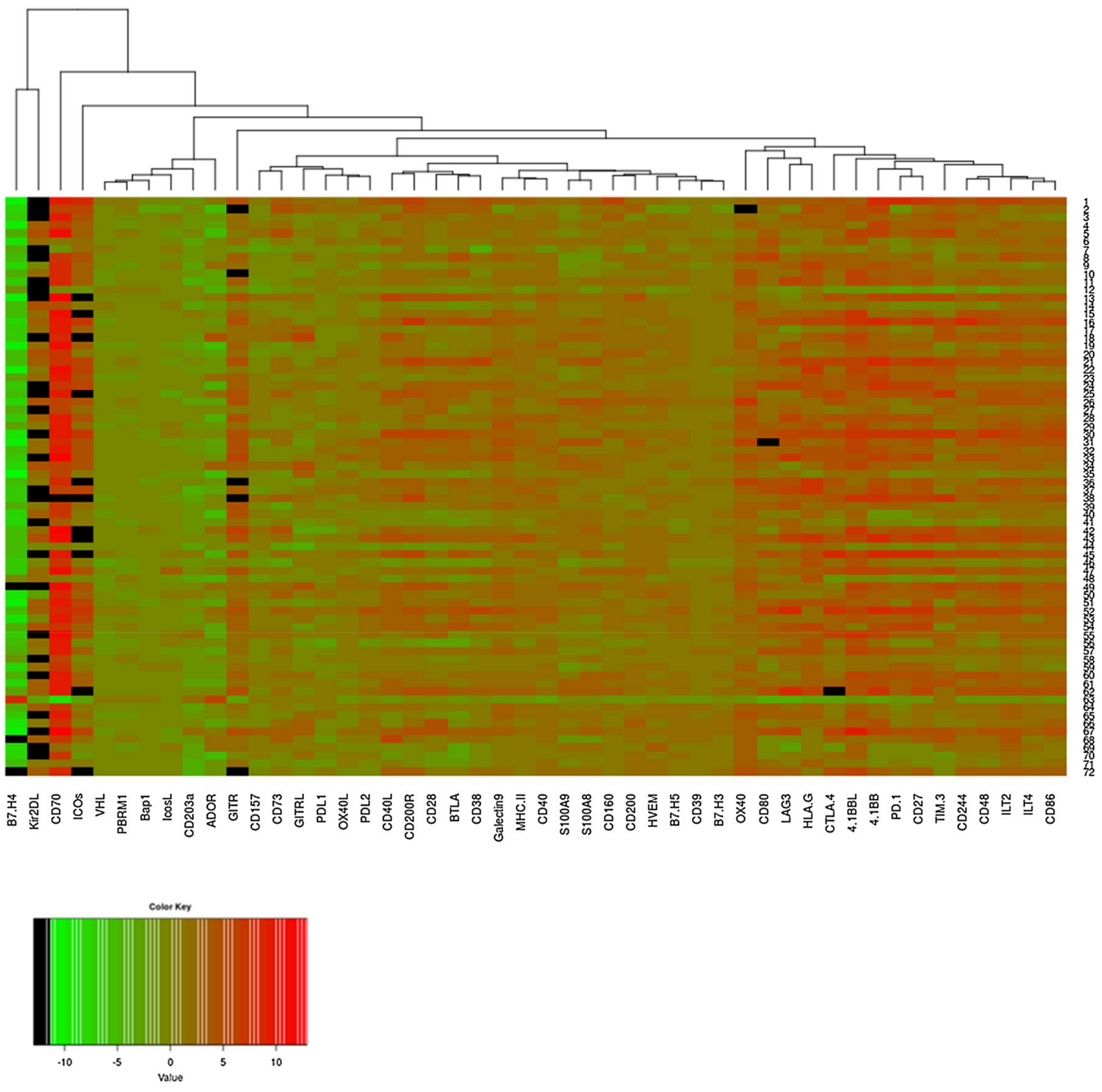


Fig. 1 Heatmap representation of the logarithm of the ratio of gene expression between tumor and non-tumor adjacent tissues. In this representation, each column represents a gene, and each row a patient. The standardized expression levels are depicted by the color gradient: green, low expression ratio in ccRCC samples; red, high expression ratio. Values in black represent a ratio equals to 0 (whose log is $-\infty$)

such as CD163-labeled macrophages representative stains are illustrated in Fig. 3.

Altogether, IHC results confirmed and extended those obtained by the analysis of RNAseq data showing for the first time the down-regulation of B7H4 and overexpression of CD70, CD40 and B7-H5 (VISTA) in ccRCC.

The ID of each patient included in the TCGA database (Table 1) allows to gain further insight into IHC results which are digitalized and made available on https://cancer.digitalslidearchive.org.

Most relevant IC expressed in tumors of ccRCC patients

Our survey of IC expressed in ccRCC reveals the overexpression of 38 IC at different levels. To find out which of them are the most important representatives of this type of cancer, we considered the recursive feature elimination (RFE) algorithm with a linear-SVM. The model eliminates the least important features, one by one iteratively until all features in the dataset are exhausted. Features are then ranked according to when they were eliminated. As such, it is a robust method for finding the best performing subset of features and owns the advantage that it can be applied to the whole dataset without fixing a threshold beforehand.

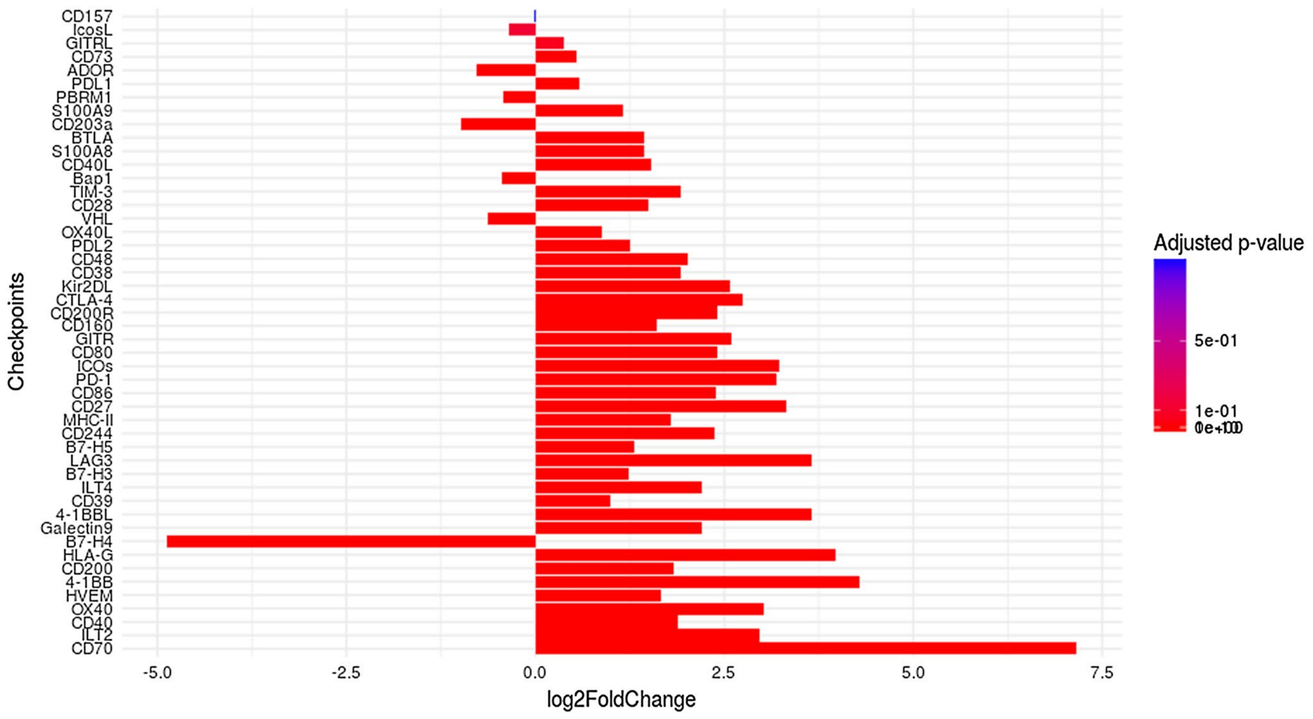


Fig. 2 Categorization of forty-four differentially expressed IC. Numbers on the *x*-axis represent the value of the logarithm fold change. If this value is negative, then the gene is underexpressed, and if this one is positive, then the gene is overexpressed. The higher the value is, the more the gene is affected by the condition (tumor). To represent differentially expressed genes, we used a color gradient. More red the bar is, more the IC is considered as differentially expressed

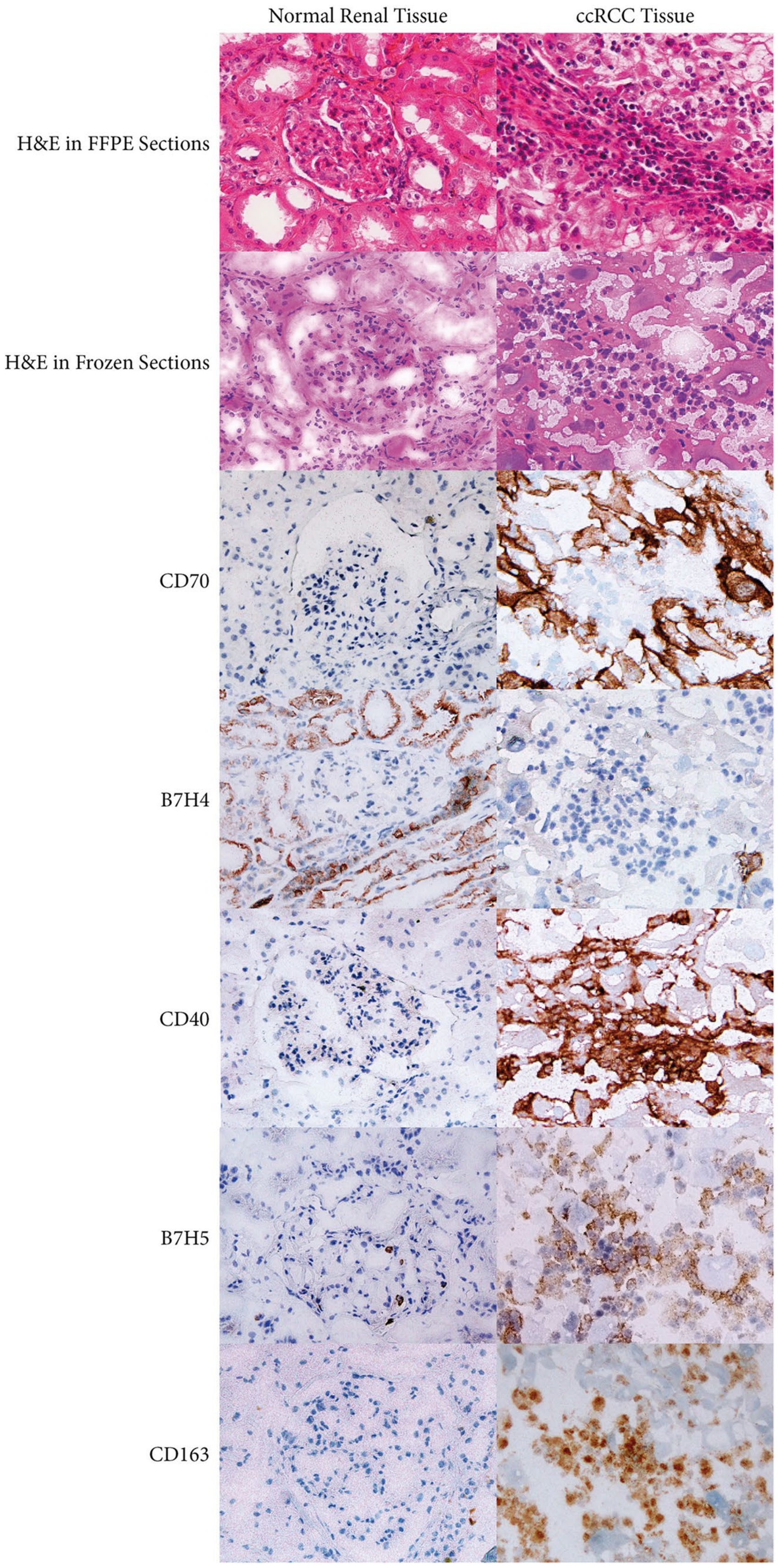
The RFE algorithm was therefore applied to the 539 patients included in the TCGA database followed by the linear-SVM model. The results, named SVMIC, revealed that the optimal number of features is 7. The accuracy is visible on the plot represented in Fig. 4a and was estimated by tenfold cross-validation over the whole dataset. The dataset was separated in ten parts, for each part we trained the model on the nine others and computed the accuracy on the singled-out part. Finally, we took the mean of the ten computed accuracies. The output of the whole algorithm is the ranked subset of the following genes: ’HLA-G’ ’HVEM’ ’PD-L1′ ’B7-H3′ ’ILT2′ ’CD40′ ’B7-H5′. The importance of each gene, defined as the square of its weight (see Materials and Methods for more details), shows how the value of the gene expression is important in the decision making. It is depicted by a graphical representation in Fig. 4b. The *x*-axis represents the selected genes, and the y-axis the importance of the genes in the linear-SVM model. Although HLA-G has been identified as the most significant gene, the graph shows that its bar plot is smaller. This would mean that HLA-G shares information with one or several of the other selected genes. This interesting feature should be the aim of future work.

Checkpoints expression in tumor cells

To validate the RFE algorithm results, we analyzed ten additional tumors derived from ccRCC patients. The results obtained by immunohistochemistry for six representative samples are illustrated in Fig. 5. A constant labeling of tumor cells and more rarely tumor-infiltrating inflammatory cells was noted in all ccRCC tumors with the antibody directed against HVEM. Nevertheless, this antibody also labeled renal tubular epithelial cells of the normal renal parenchyma. Immunostaining with HLA-G and PD-L1 antibodies revealed high and heterogeneous inter- and intra-tumor labeling. In sharp contrast, no staining was observed in normal renal parenchyma.

To better quantify the cell populations that express the IC, we analyzed tumor cells individually by flow cytometry. We focus on the most significant checkpoints revealed by the RFE algorithm: HLA-G:ILT2 and HLA-G:ILT4, 4-1BBL:4-1BB (CD137L:CD137), PD-L1:PD-1, and B7-H3, to which we added CD70, one of the most overexpressed IC in ccRCC. Thus, ten additional ccRCC tumors were more thoroughly analyzed (Fig. 6a) and a representative example, expressing the majority of the IC is shown in Fig. 6b. Consistent with the results of the RFE model, PD-L1, HLA-G and CD137L were expressed in all tumors studied. CD70

Fig. 3 IHC illustration of the localization of differentially expressed IC within tumor compartments of a representative ccRCC patient. Protein products of the selected genes CD70, B7H4, CD40, HVEM, B7H5 and CD163 were assessed in normal renal tissue and tumor tissue (H&E and immunoperoxidase stains are also shown)



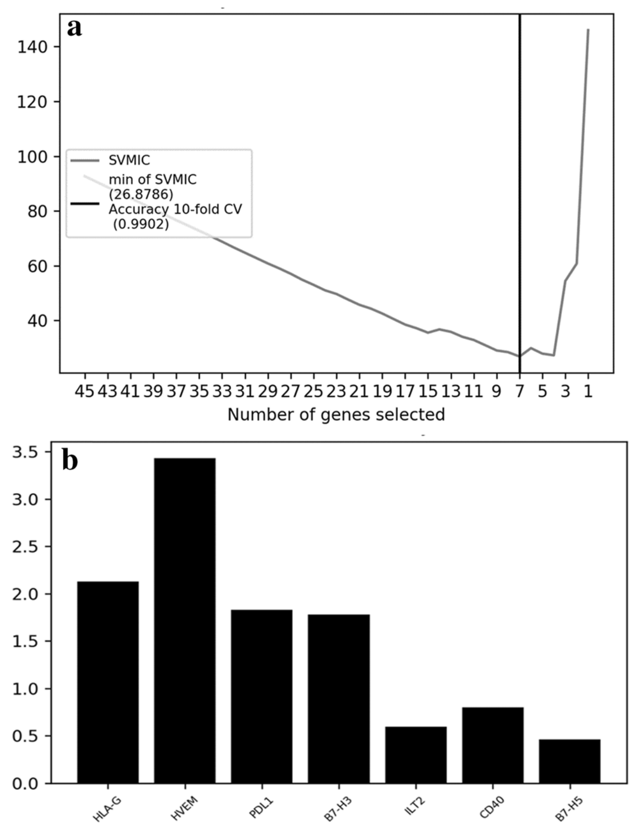


Fig. 4 A. Plot of the SVM information criteria as a function of the subsets generated by the RFE algorithm. *x*-axis: number of selected genes; *y*-axis: cross-validations score (number of correct classifications). B. Barplot showing the importance of each of the seven features selected. The *x*-axis represents the selected genes and the y-axis the importance of the genes in the linear-SVM model

and B7-H3 were found in 6/7 and 5/8 tumors studied, respectively (Fig. 6c). There was a high inter-individual variability with respect to the proportion of tumor cells expressing a given checkpoint. For instance, the proportion of tumor cells expressing PD-L1 varied from 27% in patient #1 to 100% in patient #3. Similarly, this range was 17–74% for 4-1BB (CD137L) and 5–97% for HLA-G (Fig. 6c). Notably, ILT2 expression was observed in tumor cells from six out of ten patients, and particularly significant in three of them (16%, 20% and 30%). Similarly, ILT4 expression was also observed in tumor cells from six out of ten patients, and particularly significant (from 28 to 79%). This is noteworthy since ILT2 and ILT4 are usually considered as being expressed only by leukocytes [28]. All the values are represented in Fig. 6c. These results are thus consistent with those derived from the RNAseq analysis and the RFE model and reveal for the first time the expression of ILT2 on tumor cells. In addition, a very important conclusion that follows from culturing individual tumors is that a particular tumor expresses several IC simultaneously, and this has to be taken into account for optimizing future IC-based future therapies, since the treatment with a single checkpoint does not seem to be efficient enough.

Discussion

Immune-checkpoint blockade by antibodies that target specific ligand–receptor interactions has emerged as one of the most promising therapeutic options for patients with ccRCC. Despite the success of these therapies, a considerable proportion of patients remain unresponsive to treatment suggesting that multiple non-redundant immunosuppressive mechanisms coexist within tumor and its microenvironment.

In this study, we comprehensibly analyzed forty-four IC expressed in tumors derived from patients with ccRCC in terms of gene and protein expression differences between tumor and non-tumor-adjacent tissues and ranked them according to their relevances to characterize ccRCC. Among the ligand-receptor pairs identified in this study, some are involved in inhibitory pathways, whereas others are implicated in stimulatory pathways and might be expressed in the same tumor cells or the microenvironment.

One of the most highly differentially expressed RNA in ccRCC is CD70, a type II integral membrane protein that belongs to the tumor necrosis factor superfamily. This result was confirmed at the protein level by IHC analysis and is consistent with previous report [29] adding arguments in favor of the robustness of our analysis.

Our TCGA survey also revealed high expression of CD40 in ccRCC samples compared to adjacent non-tumor tissues. Using IHC, we have confirmed this and further show that CD40 is expressed in the abundant CD163-labeled macrophage population surrounding tumor cells. Clinical trials using CD40 agonists are still ongoing even though its administration is associated with particular toxicities [30]. Another ligand-receptor pair found highly overexpressed in ccRCC samples is 4-1BB (CD137) and 4-1BBL (CD137L). These are co-stimulatory molecules that are involved in multiple steps during the progression of inflammation and hematopoiesis. High expression of 4-1BB has also been noted on a number of cancer cell lines such as liver or leukemia cell lines [31]. Clinical trials with two agonist antibodies, urelumab and utomilumab, are ongoing. Despite initial signs of efficacy, clinical development of urelumab has been hampered by inflammatory liver toxicity [32]. To our knowledge, the presence of high levels of 4-1BBL in ccRCC is reported here for the first time. The presence of co-stimulatory molecules would provide *a "second"* signal that activates T-cell and promote anti-tumor response following the blockage of the negative IC [33]. In agreement with this, blockade of CTLA-4 (inhibitory IC) together with

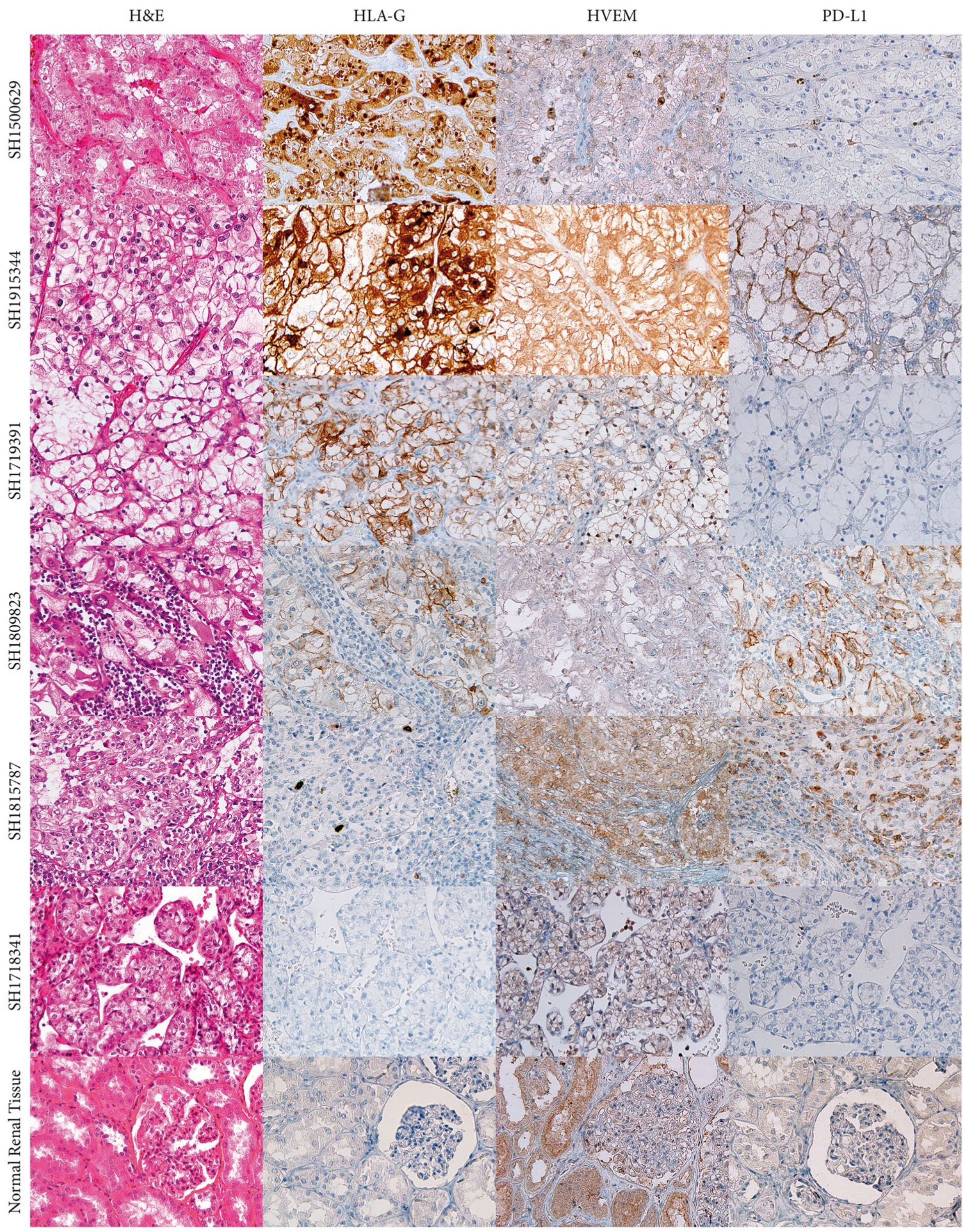


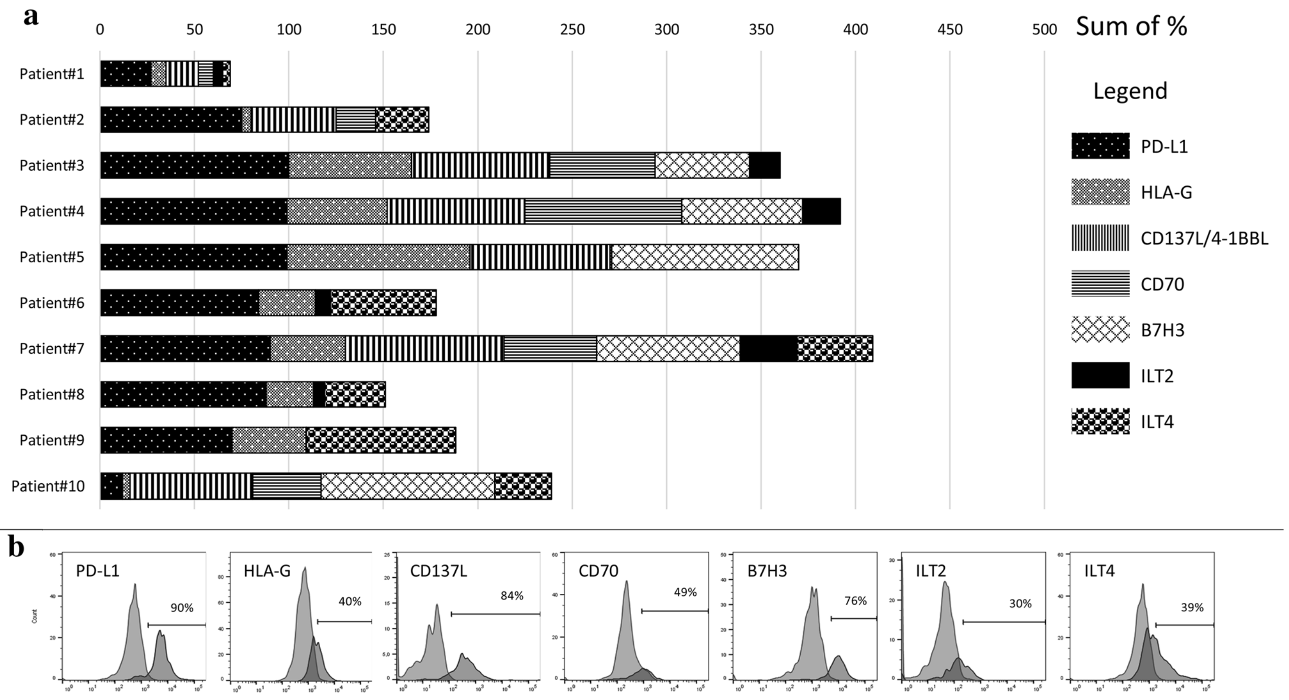
Fig. 5 IHC illustration of the localization of the three more important selected IC obtained by the linear-SVM model

engagement of ICOS (stimulatory IC) enhanced anti-tumor responses and significantly improved tumor rejection [34].

Several members of the B7 family were also found modulated in ccRCC. A hallmark of the B7 family is the capability to co-stimulate or co-inhibit T cell responses in the presence of peptide/MHC complex-mediated TCR signaling [35]. The B7 co-stimulatory ligands are important for full activation of naïve T cells in the lymphoid organs. In contrast, B7 co-inhibitory ligands are crucial for the termination of over activated T cell response and maintenance of self-tolerance [36]. The analysis of the TCGA data highlighted members of the B7 family that are differentially modulated in ccRCC, mainly B7H3, B7-H4 and B7H5/VISTA.

The levels of B7H4 were highly decreased in ccRCC. This correlates with results of our protein expression analysis performed by IHC on our cohort of ccRCC patients but sharply contrast with studies performed in other cancer types such as pancreatic or hepatocellular carcinoma, in which the overexpression of the B7-H4 protein constitute a negative prognostic marker for patient's survival [37]. This emphasizes the need to assess the expression of each IC in the different cancer types.

Another member of the B7 family, CD276/B7-H3, was found overexpressed at mRNA and protein levels. Given its immunomodulatory capacities and its overexpression on both, cancer cells and tumor-infiltrating blood vessels [38, 39], CD276/B7-H3 may be considered a more general



c

	Patient#1	Patient#2	Patient#3	Patient#4	Patient#5	Patient#6	Patient#7	Patient#8	Patient#9	Patient#10
PD-L1	27	75	100	99	99	84	90	88	70	12
HLA-G	8	5	65	53	97	30	40	25	39	4
CD137L/4-1BBL	17	45	73	73	74	ND	84	ND	ND	65
CD70	8	21	56	83	1	ND	49	ND	ND	36
B7-H3	0	0	50	64	99	ND	76	ND	0,1	92
ILT2	5	0	16	20	0	8	30	6	0,3	0
ILT4	4	28	0	0	0	56	40	32	79	30

Fig. 6 Expression of selected IC by tumor cells. **a** Cell surface expression of the indicated IC by tumor cells from ten patients. **b** Illustration of cell surface profiles of all the IC expressed in tumor of patient 7. **c** Percentage of cell surface IC expression for the ten ccRCC patients studied

target in cancer's treatment. However, its therapeutic use awaits more information due to its dual activity, either as stimulatory effect on the proliferation of both CD4+ and CD8+-T cells or as co-inhibitory molecule. Moreover, since B7-H3 is broadly expressed, especially in peripheral healthy tissues, B7-H3′s blockade by specific antibodies might be associated with severe adverse effects. In addition, B7H3 seems to share a yet unidentified receptor with B7H4. Considering that levels of B7H3 and B7H4 are highly increased and decreased respectively in ccRCC, it would be necessary to determine which of the two will work and in what circumstances.

This may also be the case of B7H5/VISTA, which is a molecule with dual activity. It behaves as a stimulatory ligand for antigen presenting cells (APCs) causing immune activation and as a negative ligand for T-cells, suppressing activation and proliferation [40]. In this study, the mRNA and protein levels of this molecule were found increased in ccRCC samples. In particular, IHC clearly showed that B7H5/VISTA is expressed on abundant macrophage surrounding tumor cells. It is likely that given the multiple binding partners of these molecules and their bidirectionality with regard to signaling, their molecular mechanisms of action have to be clarified more thoroughly before therapeutic application.

The recursive method pointed out HLA-G and PD-L1 as highly relevant IC expressed in tumor cells of ccRCC patients. These confirmed recent reports show high expression of HLA-G in ccRCC which may coexist with PD-L1 [41, 42]. This was also demonstrated here by in vitro culture of tumor cells derived from ccRCC patients. If PD-L1 is already used for therapeutic purposes, HLA-G still is not. We have previously made the proof of concept that anti-HLA-G antibodies can be used in immunotherapy protocols [43]. In addition, HLA-G offers a significant therapeutic window for targeted therapy since: (1) HLA-G has a restricted expression in normal tissues, (2) HLA-G has broad immune-inhibitory functions through binding to its receptors ILT2 and ILT4 present on immune cells. Indeed, HLA-G/ILTs can block all stages of an immune response, from APC activation and effector priming to the function of fully activated CTLs, NK or B cells [44]. Therefore, developing strategies to block this pathway would offer the advantage of exerting extended effects with less prominent systemic toxicities. We propose that simultaneously engaging PD-L1 and HLA-G blockade would constitute a promising new therapeutic opportunity. Moreover, the demonstration that ILT4 [41, 45] and for the first time in this study, ILT2, are expressed not only by tumor infiltrating cells but also by CD45-negative tumor cells themselves, adds supplementary interest of this checkpoint as a therapeutic target.

In conclusion, the approach adopted in this study, which couples transcriptome stringent data analyses and protein measurement by IHC and flow cytometry, highlighted key IC, some of which were never reported before to be expressed in ccRCC. In particular, the ranking of these checkpoints and the demonstration that each tumor express several IC simultaneously emphasize the importance that exists to expand the potential therapeutic targets to treat ccRCC. Targeting HLA-G/ILT in HLA-G-positive tumors, either concomitantly or in case of non-responsiveness to anti-PD1/PD-L1, would be one strategy to be tested. The remaining challenges include the understanding of fundamental mechanisms of IC action in pathological conditions and the identification of optimal combinations of IC for ccRCC patient's benefit.

Acknowledgements The authors are particularly grateful to Dr. Nathalie Rouas-Freiss for very instructive discussions and critical reading of the manuscript. We are also thankful to Dr. Marcela Garcia and Santiago Miriuka for their enlightened suggestions on tumor culture procedures. We kindly appreciate the experimental assistance of Alix Jacquier on cytometric assays and Jerome Delmotte on immunochemistry. The results shown are based upon data generated by the TCGA Research Network: https://www.cancer.gov/tcha.

Authors contributions DTLR was involved in conceptualization, formal analysis, writing original draft, supervision; MS helped in methodology, software, formal analysis, writing original draft; MB contributed to methodology, software, formal analysis, writing original draft; JV performed experiments, formal analysis and diagnosis; MBP contributed to methodology, performed experiments; MD performed experiments and analysis; FB performed experiments; JLM contributed to methodology, formal analysis, writing original draft; FD provided samples; S.L. contributed to formal analysis, writing original draft; PHC helped in formal analysis, methodological supervision, writing-original draft, funding; EDC was involved in research design, funding.

Funding This work was funded by the Commissariat à l'Energie Atomique et aux Energies Alternatives (CEA) and Ecole CentraleSupélec, Université Paris-Saclay.

Compliance with ethical standards

Conflict of interest The authors declare no potential conflicts of interest.

Ethical approval The study was approved by the institutional ethics committee of Saint-Louis Hospital, Paris.

Informed consent Patients provided written informed consent before sampling.

References

1. Znaor A, Lortet-Tieulent J, Laversanne M, Jemal A, Bray F (2015) International variations and trends in renal cell carcinoma incidence and mortality. Eur Urol 67:519–530
2. Brugarolas J (2013) PBRM1 and BAP1 as novel targets for renal cell carcinoma. Cancer J 19:324–332
3. Hsieh JJ, Le VH, Oyama T, Ricketts CJ, Ho TH, Cheng EH (2018) Chromosome 3p loss-orchestrated VHL, HIF, and epigenetic deregulation in clear cell renal cell carcinoma. J Clin Oncol 36:JCO2018792549
4. Levi-Schaffer F, Mandelboim O (2018) Inhibitory and coactivating receptors recognising the same ligand: immune homeostasis exploited by pathogens and tumours. Trends Immunol 39:112–122
5. Topalian SL, Drake CG, Pardoll DM (2015) Immune checkpoint blockade: a common denominator approach to cancer therapy. Cancer cell 27:450–461
6. Kruger S, Ilmer M, Kobold S, Cadilha BL, Endres S, Ormanns S et al (2019) Advances in cancer immunotherapy 2019—latest trends. J Exp Clin Cancer Res 38:268–278
7. Rotte A (2019) Combination of CTLA-4 and PD-1 blockers for treatment of cancer. J Exp Clin Cancer Res 1(381):255
8. Lalani AA, McGregor BA, Albiges L, Choueiri TK, Motzer R, Powles T et al (2019) Systemic treatment of metastatic clear cell renal cell carcinoma in 2018: current paradigms, use of immunotherapy, and future directions. Eur Urol 1:100–110
9. George S, Rini BI, Hammers HJ (2019) Emerging role of combination immunotherapy in the first-line treatment of advanced renal cell carcinoma: a review. JAMA Oncol 5:411–421
10. Sharma P, Allison JP (2015) The future of immune checkpoint therapy. Science 348:56–61
11. Marin-Acevedo JA, Soyano AE, Dholaria B, Knutson KL, Lou Y (2018) Cancer immunotherapy beyond immune checkpoint inhibitors. J Hematol Oncol 11:8
12. Guinney J, Dienstmann R, Wang X, de Reynies A, Schlicker A, Soneson C et al (2015) The consensus molecular subtypes of colorectal cancer. Nat Med 21:1350–1356
13. Thorsson V, Gibbs DL, Brown SD, Wolf D, Bortone DS, Ou Yang TH et al (2018) The immune landscape of cancer. Immunity 48(812–30):e14
14. Rover LK, Gevensleben H, Dietrich J, Bootz F, Landsberg J, Goltz D et al (2018) PD-1 (PDCD1) promoter methylation is a prognostic factor in patients with diffuse lower-grade gliomas harboring isocitrate dehydrogenase (IDH) mutations. EBioMedicine 28:97–104
15. Lopez JI, Angulo JC (2018) Pathological bases and clinical impact of intratumor heterogeneity in clear cell renal cell carcinoma. Curr Urol Rep 19:3
16. Soneson C, Delorenzi M (2013) A comparison of methods for differential expression analysis of RNA-seq data. BMC Bioinformatics 14:91
17. Anders S, Huber W (2010) Differential expression analysis for sequence count data. Genome Biol 11:R106
18. Love MI, Huber W, Anders S (2014) Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. Genome Biol 15:550

19. Reiner A, Yekutieli D, Benjamini Y (2003) Identifying differentially expressed genes using false discovery rate controlling procedures. Bioinformatics 19:368–375
20. Guyon I, Weston J, Barnhill S, Vapnik V (2002) Gene selection for cancer classification using support vector machines. Mach Learn 46(1):389–422
21. Claeskens G, Croux C, Van Kerckhoven J (2008) An information criterion for variable selection in support vector machines. J Mach Learn Res 9:541–558
22. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E (2011) Scikit-learn: Machine learning in Python. J Mach Learn Res 12:2825–2830
23. Statnikov A, Wang L, Aliferis CF (2008) A comprehensive comparison of random forests and support vector machines for microarray-based cancer classification. BMC Bioinformatics 9:319
24. Dudoit S, Fridlyand J, Speed TP (2002) Comparison of discrimination methods for the classification of tumors using gene expression data. J Am Stat Assoc 97(457):77–87
25. Dupuy A, Simon RM (2007) Critical review of published microarray studies for cancer outcome and guidelines on statistical analysis and reporting. J Natl Cancer Inst 99:147–157
26. Moch H, Cubilla AL, Humphrey PA, Reuter VE, Ulbright TM (2016) The 2016 WHO classification of tumours of the urinary system and male genital organs-part A: renal, penile, and testicular tumours. Eur Urol 70:93–105
27. Pardoll DM (2012) The blockade of immune checkpoints in cancer immunotherapy. Nat Rev Cancer 12:252–264
28. Desgrandchamps F, LeMaoult J, Goujon A, Riviere A, Rivero-Juarez A, Djouadou M et al (2018) Prediction of non-muscle-invasive bladder cancer recurrence by measurement of checkpoint HLAG's receptor ILT2 on peripheral CD8(+) T cells. Oncotarget 9:33160–33169
29. Junker K, Hindermann W, von Eggeling F, Diegmann J, Haessler K, Schubert J (2005) CD70: a new tumor specific biomarker for renal cell carcinoma. The Journal of urology 173:2150–2153
30. Hassan SB, Sorensen JF, Olsen BN, Pedersen AE (2014) Anti-CD40-mediated cancer immunotherapy: an update of recent and ongoing clinical trials. Immunopharmacol Immunotoxicol 36:96–104
31. Vinay DS, Kwon BS (2012) Immunotherapy of cancer with 4-1BB. Mol Cancer Therapeutics 11:1062–1070
32. Chester C, Sanmamed MF, Wang J, Melero I (2018) Immunotherapy targeting 4-1BB: mechanistic rationale, clinical results, and future strategies. Blood 131:49–57
33. Chen L, Flies DB (2013) Molecular mechanisms of T cell co-stimulation and co-inhibition. Nat Rev Immunol 13:227–242
34. Fan X, Quezada SA, Sepulveda MA, Sharma P, Allison JP (2014) Engagement of the ICOS pathway markedly enhances efficacy of CTLA-4 blockade in cancer immunotherapy. J Exp Med 211:715–725
35. Chen YP, Zhang J, Wang YQ, Liu N, He QM, Yang XJ et al (2017) The immune molecular landscape of the B7 and TNFR immunoregulatory ligand-receptor families in head and neck cancer: a comprehensive overview and the immunotherapeutic implications. Oncoimmunology 6:e1288329
36. Jung K, Choi I (2013) Emerging co-signaling networks in T cell immune regulation. Immune Netw 13:184–193
37. Podojil JR, Miller SD (2017) Potential targeting of B7-H4 for the treatment of cancer. Immunol Rev 276:40–51
38. Castellanos JR, Purvis IJ, Labak CM, Guda MR, Tsung AJ, Velpula KK et al (2017) B7-H3 role in the immune landscape of cancer. Am J Clin Exp Immunol 6:66–75
39. Seaman S, Zhu Z, Saha S, Zhang XM, Yang MY, Hilton MB et al (2017) Eradication of tumors through simultaneous ablation of CD276/B7-H3-positive tumor cells and tumor vasculature. Cancer Cell 31:501–515
40. Lines JL, Pantazi E, Mak J, Sempere LF, Wang L, O'Connell S et al (2014) VISTA is an immune checkpoint molecule for human T cells. Cancer research 74:1924–1932
41. Rouas-Freiss N, LeMaoult J, Verine J, Tronik-Le Roux D, Culine S, Hennequin C et al (2017) Intratumor heterogeneity of immune checkpoints in primary renal cell cancer: Focus on HLA-G/ILT2/ILT4. Oncoimmunology 6:e1342023
42. Tronik-Le Roux D, Renard J, Verine J, Renault V, Tubacher E, LeMaoult J et al (2017) Novel landscape of HLA-G isoforms expressed in clear cell renal cell carcinoma patients. Mol Oncol 11:1561–1578
43. Agaugue S, Carosella ED, Rouas-Freiss N (2011) Role of HLA-G in tumor escape through expansion of myeloid-derived suppressor cells and cytokinic balance in favor of Th2 versus Th1/Th17. Blood 117:7021–7031
44. Carosella ED, Rouas-Freiss N, Tronik-Le Roux D, Moreau P, LeMaoult J (2015) HLA-G: an immune checkpoint molecule. Adv Immunol 127:33–144
45. Zhang P, Guo X, Li J, Yu S, Wang L, Jiang G et al (2015) Immunoglobulin-like transcript 4 promotes tumor progression and metastasis and up-regulates VEGF-C expression via ERK signaling pathway in non-small cell lung cancer. Oncotarget 6:13550–13563

SMM: An R Package for Estimation and Simulation of Discrete-time semi-Markov Models

by Vlad Stefan Barbu, Caroline Bérard, Dominique Cellier, Mathilde Sautreuil and Nicolas Vergne

Abstract Semi-Markov models, independently introduced by Lévy (1954), Smith (1955) and Takacs (1954), are a generalization of the well-known Markov models. For semi-Markov models, sojourn times can be arbitrarily distributed, while sojourn times of Markov models are constrained to be exponentially distributed (in continuous time) or geometrically distributed (in discrete time). The aim of this paper is to present the R package **SMM**, devoted to the simulation and estimation of discrete-time multi-state semi-Markov and Markov models. For the semi-Markov case we have considered: parametric and non-parametric estimation; with and without censoring at the beginning and/or at the end of sample paths; one or several independent sample paths. Several discrete-time distributions are considered for the parametric estimation of sojourn time distributions of semi-Markov chains: Uniform, Geometric, Poisson, Discrete Weibull and Binomial Negative.

Introduction

Semi-Markov models, independently introduced by Lévy (1954), Smith (1955) and Takacs (1954), are a generalization of the well-known Markov models. For semi-Markov models, sojourn times can be arbitrarily distributed, while sojourn times of Markov models are constrained to be exponentially distributed (in continuous time) or geometrically distributed (in discrete time). For this reason, semi-Markov processes are more general and more adapted for applications than the Markov processes.

Semi-Markov processes have become important tools in probability and statistical modeling with applications in various domains like survival analysis, biology, reliability, DNA analysis, insurance and finance, earthquake modeling, meteorology studies, etc.; see, e.g., Heutte and Huber-Carol (2002), Ouhbi and Limnios (2003), Chryssaphinou et al. (2008), Janssen and Manca (2006), Votsi et al. (2012), D'Amico et al. (2013), Votsi et al. (2014), D'Amico et al. (2016b), Barbu et al. (2016), D'Amico et al. (2016a) for semi-Markov processes in continuous or discrete time with various applications and Sansom and Thomson (2001), Bulla and Bulla (2006), Barbu and Limnios (2008), for hidden semi-Markov models and applications in climatology, finance and DNA analysis.

Note that the semi-Markov theory is developed mainly in a continuous-time setting, while utterly less works address the discrete-time case. We refer the reader to Limnios and Oprisan (2001) for continuous-time framework and to Barbu and Limnios (2008) and references therein for discrete-time framework. The R package **SMM** that we present in this paper is developed in discrete time. Note that undertaking works also in discrete time (modeling stochastic tools, associated estimation procedures, corresponding software, etc.) is an important issue for several reasons. In our opinion, the most relevant of these reasons is that the time scale is intrinsically discrete in several applications. For instance, in DNA studies, when modeling a nucleotide or protein sequence by means of a stochastic process, the "time" of that process is in fact the position along the sequence, so it is discrete. Other examples can be found in some reliability/survival analysis applications where the time represents the number of cycles of a system or the counting of days/hours/etc. We can argue further for the importance of developing works also in discrete time, in parallel to their analogous ones developed in continuous time. For instance, we can mention the simplicity of computations in discrete time, the fact that a discrete-time stochastic process does not explode, the potential use of discrete processes after the discretization of continuous ones, etc.

Few R packages have been developed to handle semi-Markov models or hidden semi-Markov models. For semi-Markov models we have the recent **semiMarkov** R package (Król and Saint-Pierre, 2015) that performs maximum likelihood estimation for parametric continuous-time semi-Markov processes, where the distribution can be chosen between Exponential, Weibull or exponentiated Weibull. That package computes associated hazard rates; covariates can also be taken into account through the Cox proportional hazard model. Two R packages are also dedicated to hidden semi-Markov models, implementing estimation and prediction methods: the **hsmm** R package (Bulla et al., 2010) and the **mhsmm** R package (O'Connell and Højsgaard, 2011).

Note that there is no R package developed for discrete-time multi-state semi-Markov models. Thus the purpose of this paper is to present an R package that we have developed, called **SMM**, which performs parametric and non-parametric estimation and simulation for multi-state discrete-time semi-Markov processes. For the parametric estimation, several discrete distributions are considered for the sojourn times: Uniform, Geometric, Poisson, Discrete Weibull and Negative Binomial. The non-parametric estimation concerns the sojourn time distributions, where no assumptions are made on the shape of distributions. Moreover, the estimation can be done on the basis of one or several trajectories, with or without censoring. The aim of this paper is to describe the different possibilities of this package. To summarize, the package **SMM** that we present deals with different problems:

- Parametric estimation for sojourn time distributions (Uniform, Geometric, Poisson, Discrete Weibull and Negative Binomial) or non-parametric estimation;
- One or several sample paths;
- Four different types of sojourn times: a general one depending on the current state and on the next state to be visited, one depending only on the next state, one depending only on the current state, and one depending neither on the current state nor on the next state;
- Four different types of censoring: censoring at the beginning of sample paths, censoring at the end of sample paths, censoring at the beginning and at the end of sample paths or no censoring at all.

Let us make some remarks about these points.

First, concerning the censoring, the simplest situation is the one when all the sojourn times are completely observed (non censored). A more complicated and realistic framework is when the last sojourn time is not completely observed, thus being right censored; in most practical situations this case occurs. An analogous situation is when the first sojourn time is not completely observed, thus being also right censored. In practice, this last case occurs when one does not know the beginning of a phenomenon modeled by a semi-Markov chain. Note that this censoring at the beginning of the sample path is a right censoring (not a left censoring); indeed, when the first sojourn time is censored as in our paper, the available information is that the real sojourn time (that is not observed) is greater than this censored observed time. Thus we are clearly in a right censoring framework, although this happens at the beginning (i.e., left) of the trajectory, which could seem confusing. The most complete framework is when both the first and the last sojourn times are right censored.

Second, when considering estimation starting from several independent sample paths of a semi-Markov chain, it is assumed that all the trajectories are censored in the same way; note that this is not a real constraint, but we imposed this condition only in order to avoid useless technical notations that would make the comprehension more difficult.

Third, note that it is important for the four types of models (of sojourn times) to be considered separately because: (i) in practical situations, one model could be more adapted than some other; (ii) different models will yield specific parameter estimators, so it is important to study them separately.

The paper is organized as follows. The next section **Semi-Markov models** describes the semi-Markov models used in this package. Section **The SMM package** illustrates the different functions of the **SMM** package. We end the paper by presenting some concluding remarks on this R package.

Semi-Markov models

Let us consider a random system with finite state space $E = \{1, \dots, s\}$, $s < \infty$. Let $(\Omega, \mathcal{A}, \mathbb{P})$ be a probability space and assume that the time evolution of the system is governed by a stochastic process $Y = (Y_k)_{k \in \mathbb{N}^*}$, defined on $(\Omega, \mathcal{A}, \mathbb{P})$ with values in E; that is to say that Y_k gives the state of the system at time k. Let $T = (T_m)_{m \in \mathbb{N}^*}$, defined on $(\Omega, \mathcal{A}, \mathbb{P})$ with values in $\mathbb{N}$, be the successive time points when state changes in $(Y_k)_{k \in \mathbb{N}^*}$ occur (the jump times) and let also $J = (J_m)_{m \in \mathbb{N}^*}$, defined on $(\Omega, \mathcal{A}, \mathbb{P})$ with values in E, be the successively visited states at these time points. The relation between the process Y and the process J of the successively visited states is given by $Y_k = J_{N(k)}$, or, equivalently, $J_m = Y_{T_m}, m, k \in \mathbb{N}$, where $N(k) := \max\{m \in \mathbb{N} \mid T_m \leq k\}$ is the discrete-time counting process of the number of jumps in $[1, k] \subset \mathbb{N}$.

In this paper we consider four different semi-Markov models corresponding to the following four types of sojourn times.

- Sojourn times depending on the current state and on the next state:

$$f_{ij}(k) = \mathbb{P}(T_{m+1} - T_m = k | J_m = i, J_{m+1} = j);$$

- Sojourn times depending only on the current state:

$$f_{i\bullet}(k) = \mathbb{P}(T_{m+1} - T_m = k | J_m = i);$$

- Sojourn times depending only on the next state to be visited:

$$f_{\bullet j}(k) = \mathbb{P}(T_{m+1} - T_m = k | J_{m+1} = j);$$

- Sojourn times depending neither on the current state nor on the next state:

$$f(k) = \mathbb{P}(T_{m+1} - T_m = k).$$

Note that the sojourn times of the type $f_{i\bullet}(\cdot)$, $f_{\bullet j}(\cdot)$ or $f(\cdot)$ are particular cases of the general type $f_{ij}(\cdot)$. Nonetheless, in some specific applications, particular cases can be important because they are adapted to the phenomenon under study; that is the reason why we investigate these cases separately.

General case: sojourn times of the type $f_{ij}(.)$

Definition 1 (semi-Markov chain SMC and Markov renewal chain MRC)**.** *If we have*

$$\mathbb{P}(J_{m+1} = j, T_{m+1} - T_m = k | J_m = i, J_{m-1}, \ldots, J_1, T_m, \ldots, T_1) = \mathbb{P}(J_{m+1} = j, T_{m+1} - T_m = k | J_m = i), \tag{1}$$

then $Y = (Y_k)_k$ is called a semi-Markov chain *(SMC) and $(J, T) = (J_m, T_m)_m$ is called a* Markov renewal chain *(MRC).*

All along this paper we assume that the MRC or SMC are homogeneous with respect to the time in the sense that Equation (1) is independent of m.

Note that if (J, T) is a MRC, then it can be proved that $J = (J_m)_{m \in \mathbb{N}^*}$ is a Markov chain with state space E, called the *embedded Markov chain* of the MRC (J, T) (or of the SMC Y).

Definition 2. *For a semi-Markov chain we define:*

- *the* semi-Markov kernel $(q_{ij}(k))_{i,j \in E, k \in \mathbb{N}}$,

$$q_{ij}(k) = \mathbb{P}(J_{m+1} = j, T_{m+1} - T_m = k | J_m = i);$$

- *the* initial distribution $(\mu_i)_{i \in E}$,

$$\mu_i = \mathbb{P}(J_1 = i) = \mathbb{P}(Y_1 = i);$$

- *the* transition matrix $(p_{ij})_{i,j \in E}$ *of the embedded Markov chain* $J = (J_m)_m$,

$$p_{ij} = \mathbb{P}(J_{m+1} = j | J_m = i);$$

- *the* conditional sojourn time distributions $(f_{ij}(k))_{i,j \in E, k \in \mathbb{N}}$,

$$f_{ij}(k) = \mathbb{P}(T_{m+1} - T_m = k | J_m = i, J_{m+1} = j).$$

Note that

$$q_{ij}(k) = p_{ij} f_{ij}(k). \tag{2}$$

Clearly, a semi-Markov chain is uniquely determined a.s. by an initial distribution $(\mu_i)_{i \in E}$ and a semi-Markov kernel $(q_{ij}(k))_{i,j \in E, k \in \mathbb{N}}$ or, equivalently, by an initial distribution $(\mu_i)_{i \in E}$, a Markov transition matrix $(p_{ij})_{i,j \in E}$ and conditional sojourn time distributions $(f_{ij}(k))_{i,j \in E, k \in \mathbb{N}}$.

Other assumptions we make are: (i) We do not allow transitions to the same state, i.e., $p_{ii} = 0$ for all $i \in E$, or equivalently $q_{ii}(k) = 0$, for all $i \in E$, $k \in \mathbb{N}$; (ii) We assume that there are not instantaneous transitions, that is $q_{ij}(0) \equiv 0$ or equivalently $f_{ij}(0) \equiv 0$ for all $i, j \in E$; note that this implies that T is a strictly increasing sequence.

For the conditional sojourn time distributions, one can consider the associated cumulative distribution function defined by

$$F_{ij}(k) := \mathbb{P}(T_{m+1} - T_m \leq k | J_m = i, J_{m+1} = j) = \sum_{t=1}^{k} f_{ij}(t).$$

For any distribution function $F(\cdot)$ we can consider the associated survival/reliability function defined by

$$\overline{F}(k) := 1 - F(t).$$

Consequently we have

$$\overline{F}_{ij}(k) := \mathbb{P}(T_{m+1} - T_m > k | J_m = i, J_{m+1} = j) = 1 - \sum_{t=1}^{k} f_{ij}(t) = \sum_{t=k+1}^{\infty} f_{ij}(t).$$

Particular cases: sojourn times of the type $f_{i.}(.)$, $f_{.j}(.)$ and $f(.)$

We have considered up to here general semi-Markov models with the semi-Markov kernel of the type given in (2). Particular types of this semi-Markov model can be taken into account, by considering particular cases of holding time distributions $f_{ij}(k)$, where these distributions depend only on the current state i, or only on the next state j, or neither on i nor on j. For each case, let us define the semi-Markov kernel and the distribution function associated to the sojourn time distribution.

- Sojourn times depending only on the current state:

$$\begin{aligned} q_{ij}(k) &:= p_{ij} f_{i\bullet}(k), \text{ where} \\ f_{i\bullet}(k) &= \mathbb{P}(T_{m+1} - T_m = k | J_m = i) = \sum_{v \in E} p_{iv} f_{iv}(k), \\ F_{i\bullet}(k) &:= \mathbb{P}(T_{m+1} - T_m \leq k | J_m = i) = \sum_{t=1}^{k} f_{i\bullet}(t) = \sum_{t=1}^{k} \sum_{v \in E} p_{iv} f_{iv}(t). \end{aligned} \tag{3}$$

- Sojourn times depending only on the next state:

$$\begin{aligned} q_{ij}(k) &:= p_{ij} f_{\bullet j}(k), \text{ where} \\ f_{\bullet j}(k) &= \mathbb{P}(T_{m+1} - T_m = k | J_{m+1} = j), \\ F_{\bullet j}(k) &:= \mathbb{P}(T_{m+1} - T_m \leq k | J_{m+1} = j) = \sum_{t=1}^{k} f_{\bullet j}(t). \end{aligned} \tag{4}$$

- Sojourn times depending neither on the current state nor on the next state:

$$\begin{aligned} q_{ij}(k) &:= p_{ij} f(k), \text{ where} \\ f(k) &= \mathbb{P}(T_{m+1} - T_m = k). \\ F(k) &:= \mathbb{P}(T_{m+1} - T_m \leq k) = \sum_{t=1}^{k} f(t). \end{aligned} \tag{5}$$

We also denote the associated survival/reliability functions respectively by $\overline{F}_{i\bullet}(k)$, $\overline{F}_{\bullet j}(k)$, $\overline{F}(k)$.

The SMM package

The **SMM** R package is mainly devoted to the simulation and estimation of discrete-time semi-Markov models in different cases by the two following functions:

- `simulSM()` for the simulation of sequences from a semi-Markov model:
  - One or several trajectories
  - According to classical distributions for the sojourn times (Uniform, Geometric, Poisson, Discrete Weibull and Negative Binomial) or according to distributions given by the user
  - Four different types of censoring mechanisms: censoring at the beginning of sample paths, censoring at the end, censoring at the beginning and at the end, no censoring
  - Four different types of sojourn times: depending on the current state and on the next state, depending only on the current state, depending only on the next state, depending neither on the current state nor on the next state
- `estimSM()` for the estimation of model parameters:
  - One or several trajectories
  - Parametric (Uniform, Geometric, Poisson, Discrete Weibull and Negative Binomial) or non-parametric distributions for the sojourn times
  - Four different types of censoring mechanisms: censoring at the beginning of sample paths, censoring at the end, censoring at the beginning and at the end, no censoring

  – Four different types of sojourn times: depending on the current state and on the next state, depending only on the current state, depending only on the next state, depending neither on the current state nor on the next state

As the Negative Binomial distribution and the Discrete Weibull distribution can have several different parameterizations, note that we have considered the following ones in our package:

  – Discrete Weibull of type I with the density $f(x) = q^{(x-1)^\beta} - q^{x^\beta}$ for $x = 1, 2, \ldots, n$ with $n > 0$, q is the first parameter, $0 < q < 1$, and β is the second parameter, $\beta > 0$;
  – Negative Binomial with the density $f(x) = \dfrac{\Gamma(x+\alpha)}{\Gamma(\alpha)x!}\left(\dfrac{\alpha}{\alpha+\mu}\right)^\alpha\left(\dfrac{\mu}{\alpha+\mu}\right)^x$ for $x = 0, 1, \ldots, n$ with $n > 0$, Γ is the Gamma function, α is the parameter of overdispersion, $\alpha > 0$, and μ is the mean, $\mu \in \mathbb{R}$.

In order to avoid any confusion, note also that the expressions of the different densities considered in this package are also provided in the corresponding manual.

The **SMM** R package is also devoted to the simulation and estimation of discrete-time Markov models by the two following functions:

- `simulMk()` for the simulation of sequences from a kth order Markov model;
- `estimMk()` for the estimation of the parameters of the model.

All the different possibilities of the package are illustrated in Figure 1.

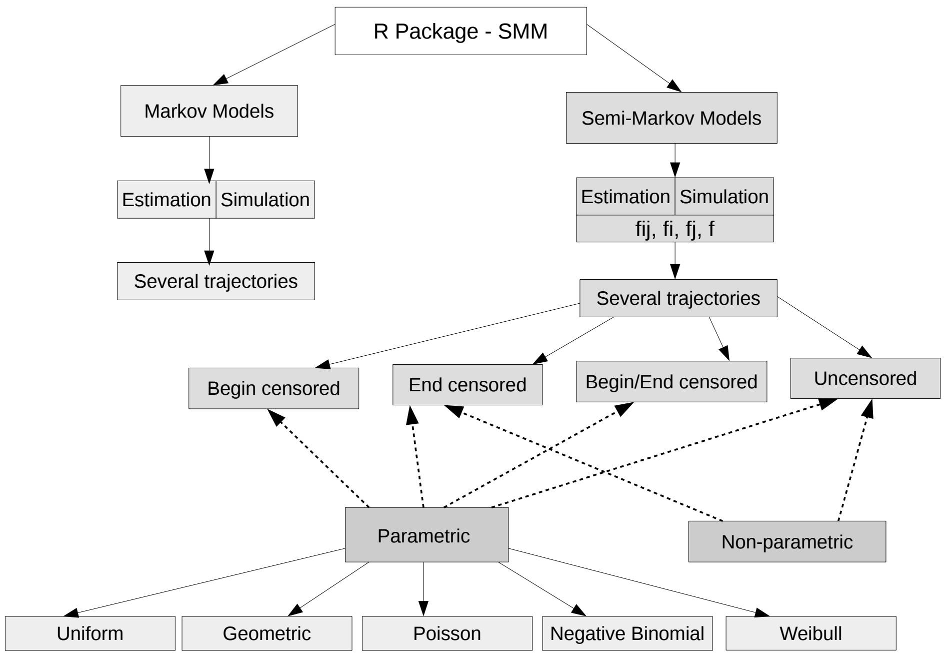


Figure 1: Schema of the **SMM** package.

Simulation of semi-Markov models

Parametric simulation: according to classical distributions

In this part, we will consider the simulation according to classical distributions.

Parameters: This simulation is carried out by the function `simulSM()`. The different parameters of the function are:

- `E`: Vector of state space of length S
- `NbSeq`: Number of simulated sequences

- `lengthSeq`: Vector containing the lengths of each simulated sequence
- `TypeSojournTime`: Type of sojourn time; it can be `"fij"`, `"fi"`, `"fj"` or `"f"` according to the four cases previously discussed
- `init`: Vector of initial distribution of length S
- `Ptrans`: Matrix of transition probabilities of the embedded Markov chain $J = (J_m)_m$ of size $S \times S$
- `distr`: Sojourn time distributions:
  - is a matrix of distributions of size $S \times S$ if `TypeSojournTime` is equal to `"fij"`,
  - is a vector of distributions of size S if `TypeSojournTime` is equal to `"fi"` or `"fj"`,
  - is a distribution if `TypeSojournTime` is equal to `"f"`,

  where the distributions to be used can be one of `"uniform"`, `"geom"`, `"pois"`, `"weibull"` or `"nbinom"`.
- `param`: Parameters of sojourn time distributions:
  - is an array of parameters of size $S \times S \times 2$ if `TypeSojournTime` is equal to `"fij"`
  - is a matrix of parameters of size $S \times 2$ if `TypeSojournTime` is equal to `"fi"` or `"fj"`
  - is a vector of parameters if `TypeSojournTime` is equal to `"f"`
- `cens.beg`: Type of censoring at the beginning of sample paths; 1 (if the first sojourn time is censored) or 0 (if not). All the sequences must be censored in the same way.
- `cens.end`: Type of censoring at the end of sample paths; 1 (if the last sojourn time is censored) or 0 (if not). All the sequences must be censored in the same way.
- `File.out`: Name of fasta file for saving the sequences; if `File.out` = NULL, no file is created. A fasta file is a simple text file containing sequences only described by one description line beginning by a ">": an example is given in Figure 2.

```
> sequence 1
aaacgtacgagtcgatcgatcgactcgatcgtacgtacggt
> sequence 2
acgtattacgatgctagctaggttggggactgcatgcatgaatgagcgatc
```

Figure 2: Example of a fasta file.

The R commands below generate three sequences of size 1000, 10000 and 2000 respectively with the finite state space $E = \{a, c, g, t\}$, where the sojourn times depend on the current state and on the next state.

```
## state space
E = c("a","c","g","t")
S = length(E)
## sequence sizes
lengthSeq3 = c(1000, 10000, 2000)
## creation of the initial distribution
vect.init = c(1/4,1/4,1/4,1/4)
## creation of transition matrix
Pij = matrix(c(0,0.2,0.3,0.4,0.2,0,0.5,0.2,0.5,0.3,0,0.4,0.3,0.5,0.2,0),
             ncol=4)
## creation of the distribution matrix
distr.matrix = matrix(c("dweibull", "pois", "geom", "nbinom",
                        "geom", "nbinom", "pois", "dweibull",
                        "pois", "pois", "dweibull", "geom",
                        "pois","geom", "geom", "nbinom"),
                      nrow = S, ncol = S, byrow = TRUE)
## creation of an array containing the parameters
param1.matrix = matrix(c(0.6,2,0.4,4,0.7,2,5,0.6,
                                         2,3,0.6,0.6,4,0.3,0.4,4),
                                       nrow = S, ncol = S, byrow = TRUE)
param2.matrix = matrix(c(0.8,0,0,2,0,5,0,0.8,
                         0,0,0.8,0,4,0,0,4),
                                          nrow = S, ncol = S, byrow = TRUE)
param.array = array(c(param1.matrix, param2.matrix), c(S,S,2))
## for the reproducibility of the results
set.seed(1)
## simulation of 3 sequences
seq3 = simulSM(E = E, NbSeq = 3, lengthSeq = lengthSeq3,
                         TypeSojournTime = "fij", init = vect.init,
               Ptrans = Pij, distr = distr.matrix, param = param.array,
               File.out = "seq3.txt")
```

First, note that in this simulation, the parameters `cens.beg` and `cens.end` are equal to 0, that is to say the simulated sequences are not censored.

Second, note also that the parameters of the distributions are given in the following way: for example, $f_{13}(\cdot)$ is Geometric distribution with parameter 0.4, while $f_{14}(\cdot)$ is Negative Binomial with parameters 4 and 2. In other words, the parameters of $f_{13}(\cdot)$ are given in the vector `param.array[1,3,]` that is equal to `(0.4, 0)` and the parameters of $f_{14}(\cdot)$ are given in the vector `param.array[1,4,]` that is equal to `(4, 2)`; that means that if a distribution has only 1 parameter, the corresponding vector of parameters will have 0 on the second position.

Values: The function `simulSM()` returns a list of simulated sequences. These sequences can be saved in a fasta file by using the parameter `File.out`.

```
seq3[[1]][1:15]
[1] "g" "g" "g" "g" "c" "c" "c" "a" "a" "a" "c" "c" "c" "g" "g"
```

Non-parametric simulation: according to distributions given by the user

Now we will consider the simulation according to distributions given by the user.

Parameters: This simulation is carried out by the function `simulSM()`. The different parameters of the function are:

- `E`: Vector of state space of length S
- `NbSeq`: Number of simulated sequences
- `lengthSeq`: Vector containing the lengths of each simulated sequence
- `TypeSojournTime`: Type of sojourn time; it can be `"fij"`, `"fi"`, `"fj"` or `"f"` according to the four cases previously discussed
- `init`: Vector of initial distribution of length S
- `Ptrans`: Matrix of transition probabilities of the embedded Markov chain $J = (J_m)_m$ of size $S \times S$
- `laws`: Sojourn time distributions introduced by the user:
  - is an array of size $S \times S \times Kmax$ if `TypeSojournTime` is equal to `"fij"`,
  - is a matrix of size $S \times Kmax$ if `TypeSojournTime` is equal to `"fi"` or `"fj"`,
  - is a vector of length $Kmax$ if `TypeSojournTime` is equal to `"f"`,

  where $Kmax$ is the maximum length for the sojourn times.
- `cens.beg`: Type of censoring at the beginning of sample paths; 1 (if the first sojourn time is censored) or 0 (if not). All the sequences must be censored in the same way.
- `cens.end`: Type of censoring at the end of sample paths; 1 (if the last sojourn time is censored) or 0 (if not). All the sequences must be censored in the same way.
- `File.out`: Name of fasta file for saving the sequences; if `File.out` = NULL, no file is created.

The R commands below generate three sequences of size 1000, 10000 and 2000 respectively with the finite state space $E = \{a, c, g, t\}$, where the sojourn times depend only on the next state.

```
## state space
E = c("a","c","g","t")
S = length(E)
## sequence sizes
lengthSeq3 = c(1000, 10000, 2000)
```

```
## creation of the initial distribution
vect.init = c(1/4,1/4,1/4,1/4)
## creation of transition matrix
Pij = matrix(c(0,0.2,0.3,0.4,0.2,0,0.5,0.2,0.5,0.3,0,0.4,0.3,0.5,0.2,0),
             ncol=4)
## creation of a matrix corresponding to the conditional
## sojourn time distributions
Kmax = 6
nparam.matrix = matrix(c(0.2,0.1,0.3,0.2,0.2,0,0.4,0.2,0.1,
                                            0,0.2,0.1,0.5,0.3,0.15,0.05,0,0,
                                            0.3,0.2,0.1,0.2,0.2,0),
                        nrow = S, ncol = Kmax, byrow = TRUE)
## for the reproducibility of the results
set.seed(2)
## simulation of 3 sequences with censoring at the beginning
seqNP3_begin = simulSM(E = E, NbSeq = 3, lengthSeq = lengthSeq3,
TypeSojournTime = "fj", init = vect.init, Ptrans = Pij,
                                 laws = nparam.matrix, File.out = "seqNP3_begin.txt",
                                 cens.beg = 1, cens.end = 0)
## simulation of 3 sequences with censoring at the end
seqNP3_end = simulSM(E = E, NbSeq = 3, lengthSeq = lengthSeq3,
                                 TypeSojournTime = "fj", init = vect.init, Ptrans = Pij,
                                 laws = nparam.matrix, File.out = "seqNP3_end.txt",
                                 cens.beg = 0, cens.end = 1)
## simulation of 3 sequences censored at the beginning and at the end
seqNP3_begin_end = simulSM(E = E, NbSeq = 3, lengthSeq = lengthSeq3,
                                 TypeSojournTime = "fj", init = vect.init, Ptrans = Pij,
                                 laws = nparam.matrix, File.out = "seqNP3_begin_end.txt",
                                 cens.beg = 1, cens.end = 1)
## simulation of 3 sequences without censoring
seqNP3_no = simulSM(E = E, NbSeq = 3, lengthSeq = lengthSeq3,
                                 TypeSojournTime = "fj", init = vect.init, Ptrans = Pij,
                                 laws = nparam.matrix, File.out = "seqNP3_no.txt")
## for the reproducibility of the results
seqNP3_begin = read.fasta("seqNP3_begin.txt")
seqNP3_end = read.fasta("seqNP3_end.txt")
seqNP3_begin_end = read.fasta("seqNP3_begin_end.txt")
seqNP3_no = read.fasta("seqNP3_no.txt")
seqNP3_begin[[1]][1:15]
```

Note that in this simulation all the different censoring mechanisms are considered.

Values: The function `simulSM()` returns a list of simulated sequences. These sequences can be saved in a fasta file by using the parameter `File.out`.

```
seqNP3_begin[[1]][1:15]
 [1] "c" "g" "g" "g" "g" "g" "g" "c" "c" "c" "a" "a" "a" "a" "a"
```

Estimation of semi-Markov models

In this subsection we explain and illustrate the estimation of a semi-Markov model in the parametric and non-parametric cases.

Parametric estimation of semi-Markov models

We will consider the distributions $f_{ij}(k) = f_{ij}(k;\theta_{ij})$ depending on unknown parameters $\theta_{ij} \in \mathbb{R}^{m_{ij}}$, where the dimension of parameters set m_{ij} is known; no assumptions are made on $\left(p_{ij}\right)_{ij}$. From the data, we want to estimate p_{ij} and θ_{ij}, $i, j \in E$. Let us assume that we have at our disposal several independent sample paths of a semi-Markov chain, say L, each of them of length n_l, $l = 1, \ldots, L$, censored at the beginning and at the end of the trajectory, i.e.,

$$y_1^l, y_2^l, \ldots, y_{n_l}^l,$$

or, equivalently,

$$j_0^l, k_0^l, j_1^l, k_1^l, j_2^l, k_2^l, \dots, j_{t^l}^l, k_{t^l}^l, j_{t^l+1}^l, k_{t^l+1}^l$$

with $\sum_{i=0}^{t^l+1} k_i^l = n_l$, where $j_0^l, \dots, j_{t^l+1}^l$ are the successive distinct visited states, k_0^l is the first sojourn time, assumed to be right censored, $k_{t^l+1}^l$ is the last sojourn time, assumed also to be right censored, while $k_1^l, \dots, k_{t^l}^l$ are the other successive sojourn times, assumed to be complete (observed, non censored).

To estimate the parameters of model, we use the maximum likelihood estimation:

$$\begin{aligned}
&\underset{p_{uv},\theta_{uv};u,v\in E}{\operatorname{argmax}} \quad (l(p_{uv},\theta_{uv};u,v\in E)) \qquad (6)\\
&= \Bigg(\underset{p_{uv},\theta_{uv};v\in E}{\operatorname{argmax}} \Bigg(\sum_{v\in E} N_{uv}(L,n_{1:L}) \log(p_{uv}) + \sum_{v\in E} \sum_{k=1}^{\max_l(n_l)} N_{uv}(k;L,n_{1:L}) \log(f_{uv}(k;\theta_{uv}))\\
&+ \sum_{v\in E} \sum_{k=1}^{\max_l(n_l)} \overline{N}_{uv}^b(k;L) \log(\overline{F}_{uv}(k;\theta_{uv}))\\
&+ \sum_{k=1}^{\max_l(n_l)} \overline{N}_{u\bullet}^e(k;L) \log\Bigg(1 - \sum_{m=1}^{k} \sum_{v\in E} p_{uv} f_{uv}(m;\theta_{uv})\Bigg)\Bigg)\Bigg)_{u\in E},
\end{aligned}$$

where we introduced the following counting processes

$$\begin{aligned}
N_{ij}(L,n_{1:L}) &= \sum_{l=1}^{L} \sum_{m=1}^{N^l(n_l)-1} \mathbb{1}_{\{J_m^l=i;J_{m+1}^l=j\}},\\
N_{ij}(k;L,n_{1:L}) &= \sum_{m=1}^{N^l(n_l)-1} \mathbb{1}_{\{J_m^l=i;J_{m+1}^l=j;T_{m+1}^l-T_m^l=k\}},\\
\overline{N}_{ij}^b(k;L) &= \sum_{l=1}^{L} \mathbb{1}_{\{J_0^l=i;J_1^l=j;T_1^l-T_0^l>k\}},\\
\overline{N}_{i\bullet}^e(k;L) &= \sum_{l=1}^{L} \mathbb{1}_{\{J_{T^l_{N^l(n_l)}}^l=i,X_{T^l_{N^l(n_l)+1}}^l>k\}},
\end{aligned}$$

where

$$N^l(n_l) = \max\{m \in \mathbb{N} \mid T_m^l \le n_l\}$$

is the counting process of jump number in $[1;n_l]$ of the trajectory l.

Note that:

- $N_{ij}(L,n_{1:L})$ represents the number of transitions from state i to state j along the L sample paths;

- $N_{ij}(k;L,n_{1:L})$ represents the number of transitions from state i to state j along the L sample paths, with sojourn time of length k in state i;

- $\overline{N}_{ij}^b(k;L)$ represents the number of trajectories starting in state i, with a next transition to state j and censored sojourn time in state i greater than k;

- $\overline{N}_{i\bullet}^e(k;L)$ represents the number of trajectories ending in state i with a censored sojourn time in state i greater than k.

Note also that in the expression (6) of the log-likelihood, the first two terms correspond to the transition probabilities and the observed (non censored) sojourn times, the third term is the contribution to the likelihood of the first sojourn times, assumed to be right censored, while the last term is the contribution to the likelihood of the last sojourn times, assumed to be right censored.

Up to here we presented the estimation for the general case, taking into account the censoring at the beginning and at the end and the sojourn times depending on the current state and on the next state. Thus the maximization problem (6) is written with the sojourn times depending on the current state and on the next state (the general model of the type $q_{ij}(k) = p_{ij}f_{ij}(k)$ given in (2)), but the different cases of sojourn times are written and coded in the package. Note also that different types of censoring are also written and coded in the package.

Parameters: The estimation is carried out by the function `estimSM()`. The different parameters of the function are:

- `file`: Path of the fasta file which contains the sequences from which to estimate
- `seq`: List of the sequence(s) from which to estimate
- `E`: Vector of state space of length S
- `TypeSojournTime`: Type of sojourn time; it can be `"fij"`, `"fi"`, `"fj"` or `"f"` according to the four cases previously discussed
- `distr`: Sojourn time distributions:
  - is a matrix of distributions of size $S \times S$ if `TypeSojournTime` is equal to `"fij"`,
  - is a vector of distributions of size S if `TypeSojournTime` is equal to `"fi"` or `"fj"`,
  - is a distribution if `TypeSojournTime` is equal to `"f"`,

  where the distributions to be used can be one of `"uniform"`, `"geom"`, `"pois"`, `"weibull"` or `"nbinom"`.
- `cens.beg`: Type of censoring at the beginning of sample paths; 1 (if the first sojourn time is censored) or 0 (if not). All the sequences must be censored in the same way.
- `cens.end`: Type of censoring at the end of sample paths; 1 (if the last sojourn time is censored) or 0 (if not). All the sequences must be censored in the same way.

Note that the sequences from which we estimate can be given either as an `R` list (`seq` argument) or as a file in fasta format (`file` argument).

```
## data
seq3 = read.fasta("seq3.txt")
E = c("a","c","g","t")
## creation of the distribution matrix
distr.matrix = matrix(c("dweibull", "pois", "geom", "nbinom",
                        "geom", "nbinom", "pois", "dweibull",
                        "pois", "pois", "dweibull", "geom",
                        "pois","geom", "geom", "nbinom"),
                      nrow = S, ncol = S, byrow = TRUE)
## estimation of simulated sequences
estSeq3 = estimSM(seq = seq3, E = E, TypeSojournTime = "fij",
distr = distr.matrix, cens.end = 0, cens.beg = 0)
```

Here, we estimate simulated sequences with no censoring. The estimation performed will correspond to the likelihood given in (6), without the third and forth terms.

Values: The function `estimSM()` returns a list containing:

- `init`: Vector of size S with estimated initial probabilities of the semi-Markov chain

  ```
  estSeq3$init
  [1] 0.0000000 0.0000000 0.3333333 0.6666667
  ```

- `Ptrans`: Matrix of size $S \times S$ with estimated transition probabilities of the embedded Markov chain $J = (J_m)_m$

  ```
  estSeq3$Ptrans
            [,1]      [,2]      [,3]      [,4]
  [1,] 0.0000000 0.2263948 0.4957082 0.2778970
  [2,] 0.2134472 0.0000000 0.2892209 0.4973319
  [3,] 0.2900627 0.4709042 0.0000000 0.2390331
  [4,] 0.4108761 0.2024169 0.3867069 0.0000000
  ```

- `param`: Array with estimated parameters of the sojourn time distributions

```
estSeq3$param
, , 1

          [,1]      [,2]     [,3]      [,4]
[1,] 0.0000000 1.7867299 0.411398 4.1358797
[2,] 0.6920415 0.0000000 4.745387 0.6131806
[3,] 2.0123457 3.0722433 0.000000 0.5729614
[4,] 3.8088235 0.3068702 0.392237 0.0000000

, , 2

     [,1] [,2] [,3]      [,4]
[1,]    0    0    0 2.1043710
[2,]    0    0    0 0.8425902
[3,]    0    0    0 0.0000000
[4,]    0    0    0 0.0000000
```

Note that, for example, `estSeq3$param[1,3,]` is the vector containing the parameters of the distribution $f_{13}(\cdot)$.

- q: Array of size $S \times S \times Kmax$ with estimated semi-Markov kernel

```
estSeq3$q[,,1:3]
, , 1

           [,1]       [,2]       [,3]       [,4]
[1,] 0.00000000 0.03792273 0.20393336 0.05070777
[2,] 0.14771431 0.00000000 0.00251382 0.19237763
[3,] 0.03877405 0.02181093 0.00000000 0.13695675
[4,] 0.00911087 0.06211573 0.15168076 0.00000000

, , 2

           [,1]       [,2]       [,3]       [,4]
[1,] 0.00000000 0.06775768 0.12003558 0.07072333
[2,] 0.04548987 0.00000000 0.01192905 0.09806611
[3,] 0.07802680 0.06700849 0.00000000 0.05848582
[4,] 0.03470170 0.04305426 0.09218596 0.00000000

, , 3

           [,1]       [,2]       [,3]       [,4]
[1,] 0.00000000 0.06053233 0.07065318 0.06124460
[2,] 0.01400899 0.00000000 0.02830398 0.06214172
[3,] 0.07850845 0.10293320 0.00000000 0.02497570
[4,] 0.06608632 0.02984219 0.05602722 0.00000000
```

Note that, for example, $q_{13}(2) = \mathbb{P}(J_{m+1} = 3, T_{m+1} - T_m = 2 | J_m = 1) = 0.12192187$.

Non-parametric estimation of semi-Markov models
Here we will consider two types of estimation for semi-Markov chains: a direct estimation, obtaining thus empirical estimators (in fact, approached MLEs), cf. Barbu and Limnios (2006), Barbu and Limnios (2008) and an estimation based on a couple Markov chain associated to the semi-Markov chain (see Trevezas and Limnios, 2011).

No censoring: direct estimation
Let $\{Y_1, Y_2, \ldots, Y_n\}$ be a trajectory of a semi-Markov chain $Y = (Y_n)_{n\in\mathbb{N}}$, censored at an arbitrary fixed time n.

- The case $q_{ij}(k) = p_{ij} f_{ij}(k)$: The maximum likelihood estimators are

$$\widehat{p_{ij}}(n) = \frac{N_{ij}(n)}{N_{i\bullet}(n)}, \quad \widehat{f_{ij}}(k;n) = \frac{N_{ij}(k;n)}{N_{ij}(n)}, \quad \widehat{q_{ij}}(k;n) = \frac{N_{ij}(k;n)}{N_{i\bullet}(n)}.$$

where $N_{ij}(n) = \sum_{m=1}^{N(n)-1} \mathbb{1}_{\{J_m=i;J_{m+1}=j\}}$, $N_{i\bullet}(n) = \sum_{m=1}^{N(n)-1} \mathbb{1}_{\{J_m=i\}}$,
$N_{ij}(k;n) = \sum_{m=1}^{N(n)-1} \mathbb{1}_{\{J_m=i;J_{m+1}=j;T_{m+1}-T_m=k\}}$.

- The case $q_{ij}(k) = p_{ij} f_{i\bullet}(k)$: The maximum likelihood estimators are

$$\widehat{p_{ij}}(n) = \frac{N_{ij}(n)}{N_{i\bullet}(n)}, \quad \widehat{f_{i\bullet}}(k;n) = \frac{N_{i\bullet}(k;n)}{N_{ij}(n)}, \quad \widehat{q_{ij}}(k;n) = \frac{N_{i\bullet}(k;n)}{N_{i\bullet}(n)},$$

where $N_{i\bullet}(k;n) = \sum_{m=1}^{N(n)-1} \mathbb{1}_{\{J_m=i;T_{m+1}-T_m=k\}}$.

- The case $q_{ij}(k) = p_{ij} f_{\bullet j}(k)$: The maximum likelihood estimators are

$$\widehat{p_{ij}}(n) = \frac{N_{ij}(n)}{N_{i\bullet}(n)}, \quad \widehat{f_{\bullet j}}(k;n) = \frac{N_{\bullet j}(k;n)}{N_{ij}(n)}, \quad \widehat{q_{ij}}(k;n) = \frac{N_{\bullet j}(k;n)}{N_{i\bullet}(n)},$$

where $N_{\bullet j}(k;n) = \sum_{m=1}^{N(n)-1} \mathbb{1}_{\{J_{m+1}=j;T_{m+1}-T_m=k\}}$.

- The case $q_{ij}(k) = p_{ij} f(k)$: The maximum likelihood estimators are

$$\widehat{p_{ij}}(n) = \frac{N_{ij}(n)}{N_{i\bullet}(n)}, \quad \widehat{f}(k;n) = \frac{N(k;n)}{N_{ij}(n)}, \quad \widehat{q_{ij}}(k;n) = \frac{N(k;n)}{N_{i\bullet}(n)},$$

where $N(k;n) = \sum_{m=1}^{N(n)-1} \mathbb{1}_{\{T_{m+1}-T_m=k\}}$.

Censoring: couple Markov chain For a semi-Markov chain $Y = (Y_n)_{n\in\mathbb{N}}$, let $U = (U_n)_{n\in\mathbb{N}}$ be the backward recurrence time of the SMC, defined by

$$U_n := n - T_{N(n)}. \tag{7}$$

We can show (cf. Limnios and Oprisan, 2001) that the chain $(Y, U) = (Y_n, U_n)_{n\in\mathbb{N}}$ is a Markov chain with state space $E \times \mathbb{N}$. We will denote its transition matrix by $\tilde{\mathbf{p}} := (p_{(i,t_1)(j,t_2)})_{i,j\in E, t_1,t_2\in\mathbb{N}}$.

The maximum likelihood estimators of $q_{ij}(k)$ (Trevezas and Limnios, 2011) are given by

$$\widehat{q_{ij}}(k;n) = \widehat{p}_{(i,k-1)(j,0)}(n) \prod_{t=0}^{k-2} \widehat{p}_{(i,t)(i,t+1)}(n), \tag{8}$$

where $\widehat{p}_{(i,t_1)(j,t_2)}(n)$ represents the classical MLE of the transition probability $p_{(i,t_1)(j,t_2)}$. Thus we obtain the corresponding estimator of p_{ij}

$$\widehat{p}_{ij} = \sum_{k=0}^{\infty} \widehat{q}_{ij}(k). \tag{9}$$

In order to compute the estimators of the sojourn times, we consider the four different types of semi-Markov kernels defined in Equations (2), (3), (4) and (5).

Parameters: The estimation is carried out by the function `estimSM()` and several parameters must be given.

- `file`: Path of the fasta file which contains the sequences from which to estimate
- `seq`: List of the sequence(s) from which to estimate
- `E`: Vector of state space of length S
- `TypeSojournTime`: Type of sojourn time; always equal to "NP" for the non-parametric estimation
- `cens.beg`: Type of censoring at the beginning of sample paths; 1 (if the first sojourn time is censored) or 0 (if not). All the sequences must be censored in the same way.
- `cens.end`: Type of censoring at the end of sample paths; 1 (if the last sojourn time is censored) or 0 (if not). All the sequences must be censored in the same way.

Note that the sequences from which we estimate can be given either as an R list (`seq` argument) or as a file in fasta format (`file` argument). The parameter `distr` is always equal to "NP".

```
 ## data
seqNP3_no = read.fasta("seqNP3_no.txt")
```

```
E = c("a","c","g","t")
## estimation of simulated sequences
estSeqNP3= estimSM(seq = seqNP3_no, E = E, TypeSojournTime = "fj",
                   distr = "NP", cens.end = 0, cens.beg = 0)
```

Here, we estimate simulated sequences with no censoring.

Values: The function `estimSM()` returns a list containing:

- `init`: Vector of size S with estimated initial probabilities of the semi-Markov chain

  ```
  estSeqNP3$init
  [1] 0.0000000 0.6666667 0.3333333 0.0000000
  ```

- `Ptrans`: Matrix of size $S \times S$ with estimated transition probabilities of the embedded Markov chain $J = (J_m)_m$

  ```
  estSeqNP3$Ptrans
            [,1]      [,2]      [,3]      [,4]
  [1,] 0.0000000 0.2051948 0.5090909 0.2857143
  [2,] 0.1938179 0.0000000 0.3107769 0.4954052
  [3,] 0.3010169 0.4874576 0.0000000 0.2115254
  [4,] 0.3881686 0.1936791 0.4181524 0.0000000
  ```

- `laws`: Array of size $S \times S \times Kmax$ with estimated values of the sojourn time distributions

  ```
  estSeqNP3$laws[,,1:2]
  , , 1

            [,1]      [,2]      [,3]      [,4]
  [1,] 0.0000000 0.3941423 0.4728997 0.2939271
  [2,] 0.1896104 0.0000000 0.4728997 0.2939271
  [3,] 0.1896104 0.3941423 0.0000000 0.2939271
  [4,] 0.1896104 0.3941423 0.4728997 0.0000000

  , , 2

            [,1]      [,2]      [,3]      [,4]
  [1,] 0.0000000 0.1949791 0.3089431 0.1959514
  [2,] 0.1073593 0.0000000 0.3089431 0.1959514
  [3,] 0.1073593 0.1949791 0.0000000 0.1959514
  [4,] 0.1073593 0.1949791 0.3089431 0.0000000
  ```

- q: Array of size $S \times S \times Kmax$ with estimated semi-Markov kernel

  ```
  estSeqNP3$q[,,1:3]
  , , 1

             [,1]       [,2]      [,3]       [,4]
  [1,] 0.00000000 0.07792208 0.2562771 0.06753247
  [2,] 0.03508772 0.00000000 0.1378446 0.15956558
  [3,] 0.05559322 0.18576271 0.0000000 0.06372881
  [4,] 0.07698541 0.08670989 0.1920583 0.00000000

  , , 2

             [,1]       [,2]      [,3]       [,4]
  [1,] 0.00000000 0.04329004 0.1411255 0.05627706
  [2,] 0.01670844 0.00000000 0.1019215 0.10025063
  [3,] 0.03796610 0.09762712 0.0000000 0.03864407
  [4,] 0.03889789 0.03160454 0.1385737 0.00000000

  , , 3

             [,1]       [,2]       [,3]       [,4]
  [1,] 0.00000000 0.02770563 0.07965368 0.04069264
  [2,] 0.07101086 0.00000000 0.05179616 0.03926483
  [3,] 0.09152542 0.05423729 0.00000000 0.02440678
  [4,] 0.10940032 0.01782820 0.05591572 0.00000000
  ```

Supplementary functions

In this package, others functions are available. These functions enable to compute the initial distribution for a semi-Markov model, the log-likelihood of a semi-Markov model and the AIC and BIC of a semi-Markov model.

- `InitialLawSM()`: Estimation of initial distribution for a semi-Markov model

Parameters:

- `q`: Array of size $S \times S \times Kmax$ with estimated semi-Markov kernel

```
seq = list(c("a","c","c","g","t","a","a","a","a",
                           "g","c","t","t","t","g"))
res = estimSM(seq = seq, E = c("a","c","g","t"), distr = "NP")
Warning message:
In .comptage(J, L, S, Kmax): Warning: missing transitions
q = res$q
p = res$Ptrans

InitialLawSM(E = c("a","c","g","t"), seq = seq, q = q)
$init
[1] 0.2205882 0.2205882 0.2058824 0.3529412
```

Values: The function `InitialLawSM()` returns a list containing a vector of the initial distribution.

- `LoglikelihoodSM ()`: Computation of the log-likelihood

Parameters:

- `E`: Vector of state space of length S
- `seq`: List of the sequence(s) from which to estimate
- `mu`: Vector of initial distribution of length S
- `Ptrans`: Matrix of transition probabilities of the embedded Markov chain $J = (J_m)_m$ of size $S \times S$
- `TypeSojournTime`: Type of sojourn time; it can be `"fij"`, `"fi"`, `"fj"` or `"f"` according to the four cases previously discussed
- `distr`: Sojourn time distributions:
  - is a matrix of distributions of size $S \times S$ if `TypeSojournTime` is equal to `"fij"`,
  - is a vector of distributions of size S if `TypeSojournTime` is equal to `"fi"` or `"fj"`,
  - is a distribution if `TypeSojournTime` is equal to `"f"`,

  where the distributions to be used can be one of `"uniform"`, `"geom"`, `"pois"`, `"weibull"` or `"nbinom"`.
- `param`: Parameters of sojourn time distributions:
  - is an array of parameters of size $S \times S \times 2$ if `TypeSojournTime` is equal to `"fij"`
  - is a matrix of parameters of size $S \times 2$ if `TypeSojournTime` is equal to `"fi"` or `"fj"`
  - is a vector of parameters if `TypeSojournTime` is equal to `"f"`
- `laws`: Sojourn time distributions introduced by the user:
  - is an array of size $S \times S \times Kmax$ if `TypeSojournTime` is equal to `"fij"`,
  - is a matrix of size $S \times Kmax$ if `TypeSojournTime` is equal to `"fi"` or `"fj"`,
  - is a vector of length $Kmax$ if `TypeSojournTime` is equal to `"f"`,

  where $Kmax$ is the maximum length for the sojourn times.

```
## state space
E = c("a","c","g","t")
S = length(E)
## creation of transition matrix
Pij = matrix( c(0,0.2,0.3,0.4,0.2,0,0.5,0.2,0.5,0.3,0,0.4,0.3,0.5,0.2,0),ncol=4)
## for the reproducibility of the results
set.seed(3)
## simulation
seq5000 = simulSM(E = E, NbSeq = 1, lengthSeq = 5000,
                                TypeSojournTime = "f", init = c(1/4,1/4,1/4,1/4),
                                Ptrans = Pij, distr = "pois", param = 2, File.out =
                                      "seq5000.txt")
## computation of the log-likelihood
LoglikelihoodSM(seq = seq5000, E = E, mu = rep(1/4,4),
                                Ptrans = Pij, distr = "pois", param = 2,
                                TypeSojournTime = "f")
$L
$L[[1]]
[1] -1748.431

$Kmax
[1] 10
```

Values: The function `likelihoodSM()` returns a list containing:

- `L`: List with the value of the likelihood for each sequence
- `Kmax`: Maximal sojourn time

We also consider model selection criteria in order to evaluate and choose among candidate models; the Akaike information criterion (AIC) and the Bayesian information criterion (BIC) are considered.

■ `AIC_SM()`: computation of the AIC

$$AIC(M) = -2\log\mathcal{L} + 2M,$$

where $\mathcal{L}$ is the log-likelihood, M is the number of parameters involved in the model

Parameters:

- `E`: Vector of state space of length S
- `seq`: List of the sequence(s) from which to estimate
- `mu`: Vector of initial distribution of length S
- `Ptrans`: Matrix of transition probabilities of the embedded Markov chain $J = (J_m)_m$ of size $S \times S$
- `TypeSojournTime`: Type of sojourn time; it can be `"fij"`, `"fi"`, `"fj"` or `"f"` according to the four cases previously discussed
- `distr`: Sojourn time distributions:
  - is a matrix of distributions of size $S \times S$ if `TypeSojournTime` is equal to `"fij"`,
  - is a vector of distributions of size S if `TypeSojournTime` is equal to `"fi"` or `"fj"`,
  - is a distribution if `TypeSojournTime` is equal to `"f"`,

  where the distributions to be used can be one of `"uniform"`, `"geom"`, `"pois"`, `"weibull"` or `"nbinom"`.
- `param`: Parameters of sojourn time distributions:
  - is an array of parameters of size $S \times S \times 2$ if `TypeSojournTime` is equal to `"fij"`
  - is a matrix of parameters of size $S \times 2$ if `TypeSojournTime` is equal to `"fi"` or `"fj"`
  - is a vector of parameters if `TypeSojournTime` is equal to `"f"`
- `laws`: Sojourn time distributions introduced by the user:
  - is an array of size $S \times S \times Kmax$ if `TypeSojournTime` is equal to `"fij"`,
  - is a matrix of size $S \times Kmax$ if `TypeSojournTime` is equal to `"fi"` or `"fj"`,

  – is a vector of length *Kmax* if `TypeSojournTime` is equal to `"f"`,

 where *Kmax* is the maximum length for the sojourn times.

```
## state space
E = c("a","c","g","t")
S = length(E)
lengthSeq3 = c(1000, 10000, 2000)
## creation of the initial distribution
vect.init = c(1/4,1/4,1/4,1/4)
## creation of transition matrix
Pij = matrix( c(0,0.2,0.3,0.4,0.2,0,0.5,0.2,0.5,0.3,0,0.4,0.3,0.5,0.2,0),ncol=4)
## creation of the distribution matrix
distr.matrix = matrix(c("dweibull", "pois", "geom", "nbinom",
                                                  "geom", "nbinom", "pois", "dweibull",
                                                  "pois", "pois", "dweibull", "geom",
                                                  "pois","geom", "geom", "nbinom"),
                                                  nrow = S, ncol = S, byrow = TRUE)
## creation of an array containing the parameters
param1.matrix = matrix(c(0.6,2,0.4,4,0.7,2,5,0.6,2,3,0.6,
                                                   0.6,4,0.3,0.4,4), nrow = S,
                                                 ncol = S, byrow = TRUE)
param2.matrix = matrix(c(0.8,0,0,2,0,5,0,0.8,0,0,0.8,
                                                  0,4,0,0,4), nrow = S, ncol = S,
                                                 byrow = TRUE)
param.array = array(c(param1.matrix, param2.matrix), c(S,S,2))
## for the reproducibility of the results
set.seed(4)
## simulation of 3 sequences
seq.crit = simulSM(E = E, NbSeq = 3, lengthSeq = lengthSeq3,
                                  TypeSojournTime = "fij", init = vect.init,
                                  Ptrans = Pij, distr = distr.matrix,
                                  param = param.array, File.out = "seq.crit.txt")
## computation of the AIC
AIC_SM(seq = seq.crit, E = E, mu = rep(1/4,4), Ptrans = Pij,
                 distr = distr.matrix, param = param.array,
                 TypeSojournTime = "fij")
[[1]]
[1] 1692.061

[[2]]
[1] 16826.28

[[3]]
[1] 3334.314
```

Values: The function `AIC_SM()` returns a list with the value of AIC for each sequence.

■ `BIC_SM()`: computation of the BIC

$$BIC(M) = -2\log\mathcal{L} + log(n)M$$

where $\mathcal{L}$ is the log-likelihood, M is the number of parameters involved in the model and n is the sample size.

Parameters:

- `E`: Vector of state space of length S
- `seq`: List of the sequence(s) from which to estimate
- `mu`: Vector of initial distribution of length S
- `Ptrans`: Matrix of transition probabilities of the embedded Markov chain $J = (J_m)_m$ of size $S \times S$
- `TypeSojournTime`: Type of sojourn time; it can be `"fij"`, `"fi"`, `"fj"` or `"f"` according to the four cases previously discussed

- `distr`: Sojourn time distributions:
  - is a matrix of distributions of size $S \times S$ if `TypeSojournTime` is equal to `"fij"`,
  - is a vector of distributions of size S if `TypeSojournTime` is equal to `"fi"` or `"fj"`,
  - is a distribution if `TypeSojournTime` is equal to `"f"`,

  where the distributions to be used can be one of `"uniform"`, `"geom"`, `"pois"`, `"weibull"` or `"nbinom"`.
- `param`: Parameters of sojourn time distributions:
  - is an array of parameters of size $S \times S \times 2$ if `TypeSojournTime` is equal to `"fij"`
  - is a matrix of parameters of size $S \times 2$ if `TypeSojournTime` is equal to `"fi"` or `"fj"`
  - is a vector of parameters if `TypeSojournTime` is equal to `"f"`
- `laws`: Sojourn time distributions introduced by the user:
  - is an array of size $S \times S \times Kmax$ if `TypeSojournTime` is equal to `"fij"`,
  - is a matrix of size $S \times Kmax$ if `TypeSojournTime` is equal to `"fi"` or `"fj"`,
  - is a vector of length $Kmax$ if `TypeSojournTime` is equal to `"f"`,

  where $Kmax$ is the maximum length for the sojourn times.

```
## computation of the BIC
BIC_SM(seq = seq3, E = E, mu = rep(1/4,4), Ptrans = Pij,
                distr = distr.matrix, param = param.array,
                TypeSojournTime = "fij")
[[1]]
[1] 1760.811

[[2]]
[1] 16927.22

[[3]]
[1] 3412.733
```

Values: The function `BIC_SM()` returns a list with the value of BIC for each sequence.

The Markov case

In the `SMM` R package, we have also implemented the estimation and the simulation of discrete-time multi-state Markov models. As in the semi-Markov case, other functions are available, enabling to estimate the initial distribution, to compute the log-likelihood and also the AIC and BIC of a Markov model.

■ `simulMk()`: Simulation of a Markov chain of order k

```
## state space
E <- c("a","c","g","t")
S = length(E)
vect.init <- c(1/4,1/4,1/4,1/4)
k<-2
p <- matrix(0.25, nrow = S^k, ncol = S)
## for the reproducibility of the results
set.seed(5)
## simulation of 3 sequences with the simulMk function
seq.markov = simulMk(E = E, nbSeq = 3, lengthSeq = c(1000, 10000, 2000),
          Ptrans = p, init = vect.init, k = 2, File.out= "seq.markov.txt")

seq.markov[[1]][1:25]
[1] "g" "g" "g" "g" "c" "c" "c" "a" "a" "a" "c" "c" "c" "g" "g" "c" "c" "c" "c"
 "c" "g" "g" "g" "a" "a"
```

■ `estimMk()`: Estimation of a Markov chain of order k

```
## state space
E <- c("a","c","g","t")
## for the reproducibility of the results
seq.markov = read.fasta("seq.markov.txt")
## estimation of simulated sequences
res.markov = estimMk(seq = seq.markov, E = E, k = 2)
```

Values: The function `estimMk()` returns a list containing:

- `init`: Vector of initial probabilities of the Markov chain

```
res.markov$init
[1] 0.2513810 0.2491905 0.2519048 0.2475238
```

- `Ptrans`: Matrix of transition probabilities of the Markov chain

```
res.markov$Ptrans
           [,1]      [,2]      [,3]      [,4]
 [1,] 0.2845283 0.2498113 0.2188679 0.2467925
 [2,] 0.2479645 0.2361214 0.2709104 0.2450037
 [3,] 0.2640625 0.2507813 0.2515625 0.2335937
 [4,] 0.2577475 0.2448980 0.2433862 0.2539683
 [5,] 0.2371988 0.2454819 0.2402108 0.2771084
 [6,] 0.2458629 0.2584712 0.2450749 0.2513790
 [7,] 0.2435897 0.2533937 0.2624434 0.2420814
 [8,] 0.2457887 0.2258806 0.2710567 0.2572741
 [9,] 0.2300296 0.2692308 0.2684911 0.2322485
[10,] 0.2767584 0.2194190 0.2484709 0.2553517
[11,] 0.2542248 0.2578986 0.2549596 0.2329170
[12,] 0.2377567 0.2535545 0.2440758 0.2646130
[13,] 0.2527473 0.2590267 0.2417582 0.2464678
[14,] 0.2448196 0.2578665 0.2509593 0.2463546
[15,] 0.2617602 0.2276176 0.2594841 0.2511381
[16,] 0.2390469 0.2790161 0.2559570 0.2259800
```

■ `InitialLawMk()`: Estimation of the initial distribution of a Markov chain of order *k*

```
seq = list(c("a","c","c","g","t","a","a","a","a","g","c","t","t","t","g"))
res = estimMk(seq = seq, E = c("a","c","g","t"), k = 1)
Warning message:
In estimMk(seq = seq, E = c("a", "c", "g", "t"), k = 1):
  missing transitions
p = res$Ptrans

InitialLawMk(E = c("a","c","g","t"), seq = seq, Ptrans = p, k = 1)
$init
[1] 0.2205882 0.2205882 0.2058824 0.3529412
```

■ `LoglikelihoodMk()`: Computation of the log-likelihood

Parameters:

- `mu`: Initial distribution
- `Ptrans`: Probability transition matrix
- `k`: Order of the Markov chain

```
## state space
E = c("a","c","g","t")
S = length(E)
## creation of transition matrix
p = matrix(rep(1/4,S*S),ncol=4)
## for the reproducibility of the results
set.seed(6)
## simulation of two sequences of length 20 and 50 respectively
seq.markov2 = simulMk(E = E, nbSeq = 2, lengthSeq = c(20,50),
                       Ptrans = p, init = rep(1/4,4), k = 1,
                       File.out = "seq.markov2.txt")
## computation of the log-likelihood
LoglikelihoodMk(seq = seq.markov2, E = E, mu = rep(1/4,4), Ptrans = p, k = 1)
$L
$L[[1]]
[1] -27.72589

$L[[2]]
[1] -69.31472
```

Values: The function `likelihoodSM()` returns a list containing the value of the likelihood for each sequence.

■ `AIC_Mk()`: Computation of the AIC for a Markov chain of order *k*

Parameters:

- `mu`: Initial distribution
- `Ptrans`: Probability transition matrix
- `k`: Order of the Markov chain

```
## for the reproducibility of the results
seq.markov2 = read.fasta("seq.markov2.txt")

## computation of the AIC
AIC_Mk(seq = seq.markov2, E = E, mu = rep(1/4,4), Ptrans = p, k = 1)
[[1]]
[1] 79.45177

[[2]]
[1] 162.6294
```

Values: The function `AIC_Mk()` returns a list containing the value of the AIC for each sequence.

■ `BIC_Mk()`: Computation of the BIC for a Markov chain of order *k*

Parameters:

- `mu`: Initial distribution
- `Ptrans`: Probability transition matrix
- `k`: Order of the Markov chain

```
## for the reproducibility of the results
seq.markov2 = read.fasta("seq.markov2.txt")

## computation of the AIC
BIC_Mk(seq = seq.markov2, E = E, mu = rep(1/4,4), Ptrans = p, k = 1)
[[1]]
[1] 91.40056

[[2]]
[1] 185.5737
```

Values: The function `BIC_Mk()` returns a list containing the value of the BIC for each sequence.

Concluding remarks

In this paper we have presented the **SMM**, an R package for the simulation and the estimation of discrete-time multi-state semi-Markov models.The conditional sojourn time can be modeled by an arbitrary distribution for a semi-Markov model, which enables a generalization with respect to Markov models, where the sojourn time is only modelled by a Geometric distribution (in discrete time) or an Exponential distribution (in continuous time). The **SMM** package offers a variety of conditional sojourn time distributions (Poisson, Uniform, Negative Binomial, Geometric and Discrete Weibull). This package provides also non-parametric estimation and simulation and takes into account censored data of several types.

To summarize, the importance and interest of the R package **SMM** that we have developed come from:

- considering versatile tools, namely discrete-time multi-state semi-Markov processes, that are of use in a variety of applied fields, like survival analysis, biology, reliability, DNA analysis, insurance and finance, earthquake modeling, meteorology studies, etc. An example of application can be the description of DNA sequences by using a 2-state process to distinguish between coding and non-coding regions in DNA sequences. It is well known that the length of such regions generally does not follow a Geometric distribution. Thus semi-Markov chains can represent adapted tools for this type of modeling.

- implementing parametric and non-parametric estimation/simulation;

- considering several censoring schemes, that important in various applications;

- taking into account one or several independent sample paths;

- considering different types of semi-Markov kernels: either of the general type $q_{ij}(k) = p_{ij}f_{ij}(k)$ with the holding time distributions $f_{ij}(k)$ depending on the current state and on the next state to be visited, or with the holding time distributions depending only on the current state, $q_{ij}(k) := p_{ij}f_{i\bullet}(k)$, or with the holding time distributions depending only on the next state to be visited, $q_{ij}(k) := p_{ij}f_{\bullet j}(k)$, or with the holding time distributions depending neither on the current, nor on the future state, $q_{ij}(k) := p_{ij}f(k)$. As already mentioned, it is important that these four types of models be considered separately.

In conclusion, the R package **SMM** that we have developed deals with an important and versatile tool, useful for researchers, practitioners and engineers in various fields.

Bibliography

V. S. Barbu and N. Limnios. Empirical estimation for discrete time semi-Markov processes with applications in reliability. *Journal of Nonparametric Statistics*, 18(4):483–498, 2006. URL https://doi.org/10.1080/10485250701261913. [p236]

V. S. Barbu and N. Limnios. *Semi-Markov Chains and Hidden Semi-Markov Models toward Applications*, volume 191 of *Lecture Notes in Statistics*. Springer-Verlag, New York, NY, 2008. ISBN 978-0-387-73171-1 978-0-387-73173-5. URL http://link.springer.com/10.1007/978-0-387-73173-5. [p226, 236]

V. S. Barbu, A. Karagrigoriou, and A. Makrides. Semi-Markov modelling for multi-state systems. *Methodology and Computing in Applied Probability*, 2016. ISSN 1573-7713. URL https://doi.org/10.1007/s11009-016-9510-y. [p226]

J. Bulla and I. Bulla. Stylized facts of financial time series and hidden semi-Markov models. *Comput. Statist. Data Anal.*, 51:2192–2209, 2006. URL https://doi.org/10.1016/j.csda.2006.07.021. [p226]

J. Bulla, I. Bulla, and O. Nenadić. Hsmm - An R package for analyzing hidden semi-Markov models. *Computational Statistics & Data Analysis*, 54(3):611–619, 2010. ISSN 0167-9473. URL https://doi.org/10.1016/j.csda.2008.08.025. [p226]

O. Chryssaphinou, M. Karaliopoulou, and N. Limnios. On discrete time semi-Markov chains and applications in words occurrences. *Comm. Statist. Theory Methods*, 37:1306–1322, 2008. URL https://doi.org/10.1080/03610920701713328. [p226]

G. D'Amico, F. Petroni, and F. Prattico. Wind speed modeled as an indexed semi-Markov process. *Environmetrics*, 24(6):367–376, 2013. ISSN 1099-095X. URL https://doi.org/10.1002/env.2215. [p226]

G. D'Amico, J. Janssen, and R. Manca. Downward migration credit risk problem: a non-homogeneous backward semi-Markov reliability approach. *Journal of the Operational Research Society*, 67(3):393–401, 2016a. ISSN 1476-9360. URL https://doi.org/10.1057/jors.2015.35. [p226]

G. D'Amico, R. Manca, C. Corini, F. Petroni, and F. Prattico. Tornadoes and related damage costs: Statistical modelling with a semi-Markov approach. *Geomatics, Natural Hazards and Risk*, 7(5): 1600–1609, 2016b. URL https://doi.org/10.1080/19475705.2015.1124462. [p226]

N. Heutte and C. Huber-Carol. Semi-Markov models for quality of life data with censoring. In M. Mesbah, M.-L. T. Lee, and B. F. Cole, editors, *Statistical Methods for Quality of Life Studies*, pages 207–218. Kluwer, Dordrecht, 2002. URL https://doi.org/10.1007/978-1-4757-3625-0_16. [p226]

J. Janssen and R. Manca. *Applied Semi-Markov Processes*. Springer-Verlag, 2006. URL https://doi.org/10.1007/0-387-29548-8. [p226]

A. Król and P. Saint-Pierre. **SemiMarkov** : An *R* Package for Parametric Estimation in Multi-State Semi-Markov Models. *Journal of Statistical Software*, 66(6), 2015. ISSN 1548-7660. URL https://doi.org/10.18637/jss.v066.i06. [p226]

P. Lévy. Processus semi-markoviens. In *Proc. of International Congress of Mathematics, Amsterdam*, 1954. [p226]

N. Limnios and G. Oprisan. *Semi-Markov Processes and Reliability*. Birkhäuser, Boston, 2001. URL https://doi.org/10.1007/978-1-4612-0161-8. [p226, 237]

J. O'Connell and S. Højsgaard. Hidden Semi Markov Models for Multiple Observation Sequences: The **mhsmm** Package for *R*. *Journal of Statistical Software*, 39(4), 2011. ISSN 1548-7660. URL https://doi.org/10.18637/jss.v039.i04. [p226]

B. Ouhbi and N. Limnios. Nonparametric reliability estimation for semi-Markov processes. *J. Statist. Plann. Inference*, 109(1-2):155–165, 2003. URL https://doi.org/10.1016/S0378-3758(02)00308-7. [p226]

J. Sansom and P. J. Thomson. Fitting hidden semi-Markov models to breakpoint rainfall data. *J. Appl. Probab.*, 38A:142–157, 2001. URL https://doi.org/10.1239/jap/1085496598. [p226]

W. L. Smith. Regenerative stochastic processes. *Proc. R. Soc. Lond. Ser. A Math. Phys. Eng.*, 232:6–31, 1955. URL https://doi.org/10.1098/rspa.1955.0198. [p226]

L. Takacs. Some investigations concerning recurrent stochastic processes of a certain type. *Magyar Tud. Akad. Mat. Kutato Int. Kzl.*, 3:115–128, 1954. [p226]

S. Trevezas and N. Limnios. Exact MLE and asymptotic properties for nonparametric semi-Markov models. *Journal of Nonparametric Statistics*, 23(3):719–739, 2011. URL https://doi.org/10.1080/10485252.2011.555543. [p236, 237]

I. Votsi, N. Limnios, G. Tsaklidis, and E. Papadimitriou. Estimation of the expected number of earthquake occurrences based on semi-Markov models. *Methodology and Computing in Applied Probability*, 14(3):685–703, 2012. ISSN 1573-7713. URL https://doi.org/10.1007/s11009-011-9257-4. [p226]

I. Votsi, N. Limnios, G. Tsaklidis, and E. Papadimitriou. Hidden semi-Markov modeling for the estimation of earthquake occurrence rates. *Communications in Statistics: Theory and Methods*, 43: 1484–1502, 2014. URL https://doi.org/10.1080/03610926.2013.857414. [p226]

Vlad Stefan Barbu
Université de Rouen-Normandie
Laboratoire de Mathématiques Raphaël Salem
UFR des Sciences et Techniques
Avenue de l'Université, BP. 12
76801 Saint-Étienne-du-Rouvray, France
E-mail: barbu@univ-rouen.fr
URL: http://lmrs.univ-rouen.fr/persopage/barbu

Caroline Bérard
Université de Rouen-Normandie

LITIS EA 4108
E-mail: caroline.berard@univ-rouen.fr

Dominique Cellier
Université de Rouen-Normandie
LITIS EA 4108
E-mail: dominique.cellier@laposte.net

Mathilde Sautreuil
Université de Rouen-Normandie
Laboratoire de Mathématiques Raphaël Salem
UFR des Sciences et Techniques
Avenue de l'Université, BP. 12
76801 Saint-Étienne-du-Rouvray, France
E-mail: mathilde.sautreuil@gmail.com

Nicolas Vergne
Université de Rouen-Normandie
Laboratoire de Mathématiques Raphaël Salem
UFR des Sciences et Techniques
Avenue de l'Université, BP. 12
76801 Saint-Étienne-du-Rouvray, France
E-mail: nicolas.vergne@univ-rouen.fr
URL: http://lmrs.univ-rouen.fr/persopage/nicolas-vergne