Augmenting ScalpelSig

James Gao

April 2022

Abstract

A common area of cancer research is determining actionable biomarkers in the genome to treat cancer patients. Whole genome sequencing is expensive, to the point where it's just not clinically viable to perform, much less decipher actionable treatments from. In an attempt to solve this problem, ScalpelSig [2] designs genomic panels to detect the presence of mutational signatures in tumor samples taken from cancer patients. The main idea of the ScalpelSig algorithm is to retrieve some subset of the genome so that clinicians don't need to perform expensive whole genome sequencing to detect mutational signatures in a cancer patient. The goal is to "learn" the genomic panel so that clinicians can sequence just these subsets of the genome. The genomic panels retrieved from ScalpelSig was demonstrated to outperform genomic panels such as MSK-Impact [1], and most notably, whole exome sequencing. In this paper, we aim to develop an improvement upon the original ScalpelSig algorithm by adjusting the window sizes in the genome that are used to compute mutation counts, so that we can further reduce the total amount of data that needs to be sequenced in the genome, while maintaining similar accuracy levels.

Introduction

Cancer is a disease characterized by the accumulation of random mutations. The most common type of mutation is single nucleotide variation, or SNV for short. In SNV, a base pair in the genome is randomly converted into a different type of base pair. These mutations are denoted by the conversion of a single nucleotide in the base pair. There are six canonical SNV mutation categories: C >A, C >G, C >T, T >A, T >C, T >G. We also include the 5' and 3' flanking bases where the mutation occurs to represent the mutation (i.e A[C >T]G). With 6 possible types of SNV mutation categories, and 4 possible nucleotides in each of the flanking bases, we have 6 * 4 * 4 = 96 total SNV mutation categories. For this paper, we will represent mutational exposures as 96 dimensional vectors that encode the counts of each SNV mutation category. The ground truth mutational signatures are defined as a 96 dimensional probability distribution representing the distribution of mutation categories. Nearly

all methods for mutational signature analysis studies performs whole genome sequencing in order to retrieve the total counts of all these mutations, which is used to estimate signature exposures.

Mutational signatures have been demonstrated to be patterned [3]. For instance, mutations developed from smoking tend to produce a different pattern than mutations developed from UV radiation. As a result, many studies use the mutational exposures taken from whole genome sequencing to estimate the presence of certain mutational signatures in the tumor sample. In doing so, it is possible to make an estimation of whether a particular mutational signature is "active" within a genome. This provides us with information that could potentially be used to develop treatments tailored to a specific patient.

However, whole genome sequencing is extremely expensive, especially in the context of this mutational signature analysis problem where we are trying to perform WGS on each patient in order to identify the active mutational signatures. To solve this problem, Franzese et al. develops Scalpelsig [2], a novel procedure to acquire a genomic panel (subset of the whole genome) that allows us to estimate the mutational exposures and determine whether a particular signature is active in a given tumor sample.

The high level procedure of ScalpelSig is:

1. Divide the genome into windows of fixed size
2. Using a window scoring function, get a heuristic measure of mutational signature activity in each window
3. Select the top scoring windows and use these windows as our genomic panel

ScalpelSig has demonstrated great performance, surpassing benchmark accuracies of other genomic panels such as MSK-Impact [1] and coming quite close to the performance of whole exome sequencing (WES). The closeness of performance to WES is of particular interest, because the genomic panel designed by ScalpelSig used less than 10% of the overall sequencing data that WES used.

In the original ScalpelSig method, the windows that the genome are divided into are of fixed size. In this paper, I aim to augment ScalpelSig by varying the window sizes in order to further reduce the data we need to sequence from the whole genome, while attempting to maintain similar performance of the original ScalpelSig method.

Materials and Methods

Data

I used the breast cancer dataset taken from the ScalpelSig github repository (https://github.com/lrgr/scalpelsig). The mutation counts were found in the "COMBINED_sbs_array_winsize1e+05.rds" file. This file was approximately 6.7 Gb, so it was quite a large dataset to work with. I used rpy2 (https://rpy2.github.io/) to convert this R dataset into a numpy array in Python so I could work with it. After processing the mutation count data from this file, I retrieved a numpy array of shape (num samples, num windows, num mutation categories) = (560, 30977, 96). Each window is of size 10 kb, and encodes the total mutation counts over all bases in that 10 kb window.

The ground truth mutational signature distributions were found in the "cosmic_signatures.tsv" file. This file contained the probability distribution of mutations that defined each of the 13 signatures commonly found in breast cancer samples.

The whole exome sequencing signature estimations were found in the "gencode_exon_panel_sig_est.tsv" file. This file contains the results of signature estimation, which is used to directly determine whether a particular sample is considered active for a given signature. I will elaborate on this signature estimation as I discuss the methods. The activity predictions computed from the signature estimations from this file are used as our benchmark for whole exome sequencing that we will compare our ScalpelSig methods on.

There is also a file that contains the signature estimations for MSK-Impact panels. However, it seems that the samples in the MSK-Impact file don't match the samples in the "COMBINED_sbs_array_winsize1e+05.rds" file, which is the file that we are performing our ScalpelSig method on. As a result, we exclude the MSK-Impact panel from our benchmark comparisons.

Finally, there is a file named "nz_signature_estimation.tsv", which I'm not quite sure what exactly it represents. However, when I compute the signature estimations from this file to predict the activity of tumor samples, I obtain an .8 AUPR score between the predictions computed from this file, and the predictions computed by using WGS data. Therefore, I suspect that this file contains some type of ground truth for WGS predictions, although it is not an exact match. As a result, I retrieve the ground truth WGS predictions in two ways:

1. I use all the counts from "COMBINED_sbs_array_winsize1e+05.rds" to manually compute WGS counts and I use these counts to get a signature estimation which will ultimately be our input used to predict the activity of each tumor sample

2. I use the signature estimation from "nz_signature_estimation.tsv" as the

input to predict the activity of each tumor sample

Signature Estimation

To compute mutational exposures, we use the SignatureEstimation framework (https://github.com/lrgr/signature-estimation-py), which takes as input the mutation count matrix (num samples x num mutational categories), and the mutational signature matrix (num signatures x num mutational categories). SignatureEstimation outputs a matrix (num samples x num signatures) that encodes the mutational exposure for each signature. We can read the output for each sample as a set of weights representing the weight of the exposure for each particular signature in that sample.

Computing Signature Activity

In the ScalpelSig paper, a sample is defined to be "active" for a particular signature if the mutational exposure of that signature is greater 5% of the total exposures of all signatures in that sample. In our procedure, we use SignatureEstimation to compute the mutational exposures from panel or WGS counts, then we define samples with >5% of the mutations attributed to each signature as active for that particular signature. We use the weights retrieved from SignatureEstimation to compute this exposure percentage.

ScalpelSig Original Methods

As discussed in the introduction, ScalpelSig at the high level is comprised of three steps:

1. Divide the genome into windows of fixed size
2. Using a window scoring function, get a heuristic measure of mutational signature activity in each window
3. Select the top scoring windows and use these windows as our genomic panel

The "COMBINED sbs array winsize1e+05.rds" file already contains the genome split into windows of size 10 kb (kilo base), which handles the window splitting section of the ScalpelSig method.

ScalpelSig uses scalar projection as a heuristic measure to score the goodness of fit between a mutation count vector c and a signature vector q. We write this as:

$$proj_q(c) = \frac{1}{|q|} < q, c >$$

We want to maximize this projection score for samples that are considered active and minimize this projection score for samples that are considered inactive. As

a result, we define a contrastive heuristic function to optimize over these criteria. Let us define P to be the set of windows $\{w_1, w_2, ...\}$ that define the ScalpelSig genomic panel. Let us define S as the set of all tumor samples, and A as the set of all active samples. Let us also define c_w^i to be the mutational count vector in sample i taken from window w. Finally, let us define q to be the mutational signature vector (probability distribution). We define our contrastive heuristic function as follows:

$$maximize_P \sum_{w \epsilon P} (\sum_{i \epsilon A} proj_q(c_w^i) - \sum_{j \epsilon S \setminus A} proj_q(c_w^j))$$

By maximizing this heuristic function, we essentially choose windows that do a good job of discriminating between active and inactive samples for a particular signature, which allows us to construct a panel that can optimally detect the presence of a particular signature in any given tumor sample.

All we need to do to obtain a panel that maximizes this heuristic function is take the N highest scoring windows out of all the windows we split the genome into, where N is the number of windows in our genomic panel. We define our window scoring function in a very similar manner:

$$h(w) = \sum_{i \epsilon A} proj_q(c_w^i) - \sum_{j \epsilon S \setminus A} proj_q(c_w^j)$$

However, we know that mutation rates differ between tumor samples, so there is a completely reasonable possibility that our heuristic window scoring function will simply prioritize windows that contain an exceptionally large mutation count, without any regard to the actual distribution of the mutation itself. To resolve this issue, ScalpelSig introduces a downweighting parameter $\alpha \epsilon (0, 1]$, and rewrites our window scoring function as:

$$h(w) = \sum_{i \epsilon A} proj_q(c_w^i)^\alpha - \sum_{j \epsilon S \setminus A} proj_q(c_w^j)^\alpha$$

Since $\alpha \epsilon (0, 1]$, larger projected values will be downweighted more than smaller projected values, which will help normalize the mutation counts such that abnormally large mutational count vectors won't be prioritized over normally sized mutational count vectors.

In my experiments, I use 90% of the tumors samples from the mutational count data taken from "COMBINED_sbs_array_winsize1e+05.rds" for training and selecting the windows to be used in our ScalpelSig panel. I take the total mutation counts over the entire genome of the training samples and input these mutation counts into the SignatureEstimation framework to retrieve mutational exposures. I then use the 5% threshold to determine which samples are active and

inactive for each signature. Now that we have the set of active and inactive samples, we can apply the window scoring function to each window in order to select the top N windows for our genomic panel. I select 250 windows for my ScalpelSig panel and I also use $\alpha = .5$, the same setting that the original ScalpelSig paper uses for its most relevant results. For training our genomic panel, I perform the ScalpelSig method on each of the 13 mutational signatures commonly found in breast cancer samples, and which are defined in the "cosmic_signatures.tsv" file, which contains our mutational signature vectors. We end our training with 13 genomic panels optimized for each signature.
After obtaining the genomic panels from the ScalpelSig training procedure we can now attempt to predict which tumor samples in our held out test set are active for a particular signature. We can simply accumulate the mutational counts from our genomic panels, rather than accumulate the mutational counts over the entire genome as inputs to our SignatureEstimation function, which will output the mutational exposures for each signature. We then apply the same 5% activity threshold rule to determine which samples are active for a particular signature.

We then compute the ground truth predictions in two ways:

1. Use WGS mutational counts from the test samples to compute signature exposures from SignatureEstimation and use the exposures to determine which samples are active and inactive (same 5% threshold)

2. Use the SignatureEstimation results from "nz_signature_estimation.tsv" to determine which samples are active and inactive (same 5% threshold)

   After all of this, we use AUPR between our ground truth predictions and our ScalpelSig panel predictions as the measure of performance for our algorithm. We run this experiment for 15 iterations, and average out the AUPR for all experiments to get a more unbiased measure of performance.

ScalpelSig V2 Methods

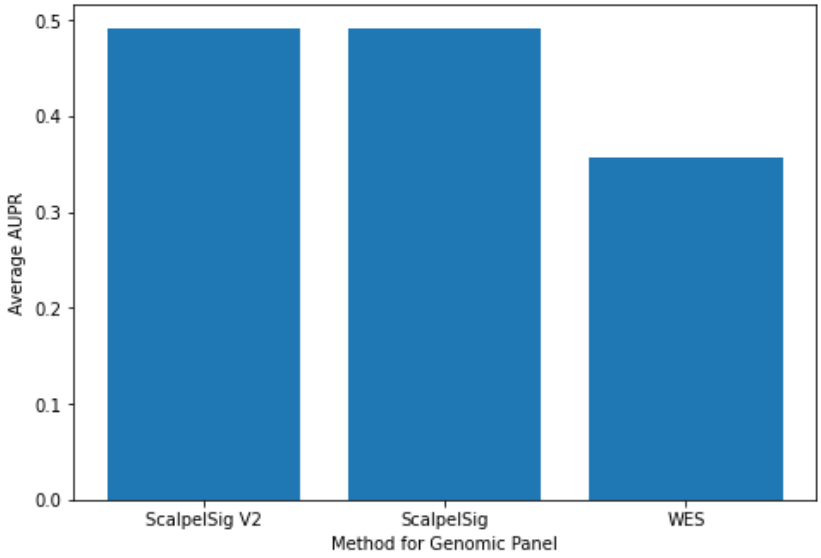
In my augmented ScalpelSig algorithm, I make a straightforward modification. In each window that I iterate through to score, I split the window into some number of mini-windows (4 mini-windows in my experiments). I then scatter mutations to get my simulated data for each of these mini-windows. Afterwards, I compile the first $i \epsilon [1, \text{num mini windows}]$ consecutive mini windows as a single window, and score it using our window scoring function. I then select the best compilation of consecutive mini-windows. I do this for each possible value of i starting from 1 and select the best scoring compiled window out of all of these. Since I start by setting the 1st mini-window as our best scoring window, each successive iteration that increases the total size of the window needs to outscore the previous, smaller windows to be worth selecting as our new best window. This allows the Scalpel algorithm to prioritize smaller window sizes, and only selecting a larger window size if there is an improvement in the score. This does exactly what we are trying to accomplish in this paper.

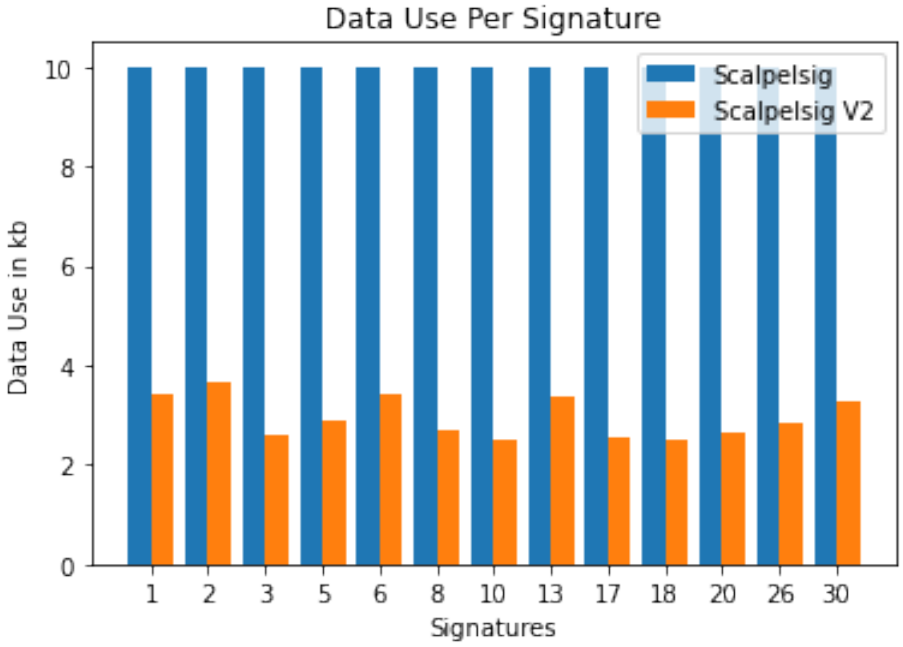
ScalpelSig V2 Testing Procedure

As soon as I begin working on adjusting the window sizes, I quickly came to an issue. The WGS data available from the ScalpelSig dataset that we retrieve from "COMBINED_sbs_array_winsize1e+05.rds" only has counts in 10 kb windows. As a result, we don't have real data that we can freely modify window sizes for.

To get around this issue, I decided to "scatter" the mutational counts from the 10 kb windows across a set of "mini-windows", which are just smaller windows that make up a 10 kb window. For instance, if I have 4 mini-windows, then I will scatter the mutational counts from the 10 kb window into 4 2.5 kb windows. I use a probabilistic method for this, constructing a probability vector of length "num mini windows". For each entry in the probability vector, I will keep that proportion of counts from the original 10 kb window in the corresponding mini window in the probability vector. As a result, I can simulate what the data would look like if I actually had smaller window sizes available. Since our dataset is so large, I need to apply this method anytime I am using mutation counts for the window, rather than being able to scatter the mutation counts for the entire dataset at once. I attempted to do this, since it makes the data easier to work with, but as expected, my computer simply ran out of memory so I settled on scattering mutations at runtime.

Experimental Results Using WGS Mutation Counts Signature Estimations

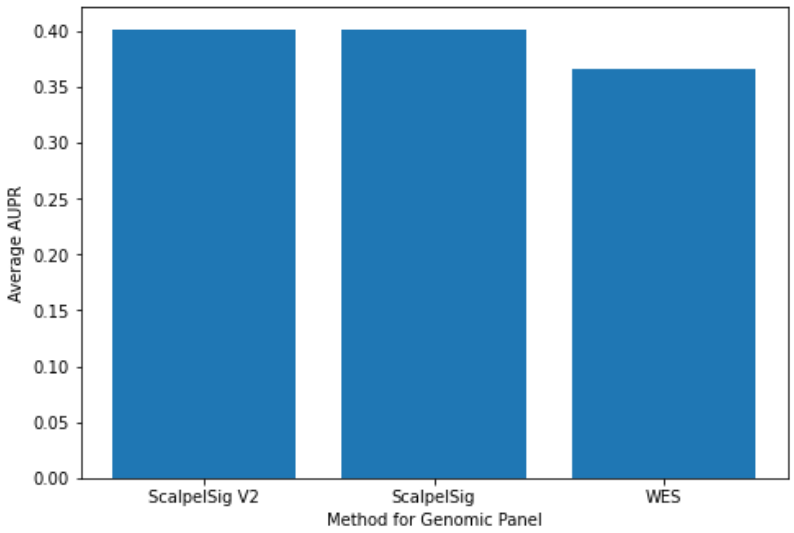


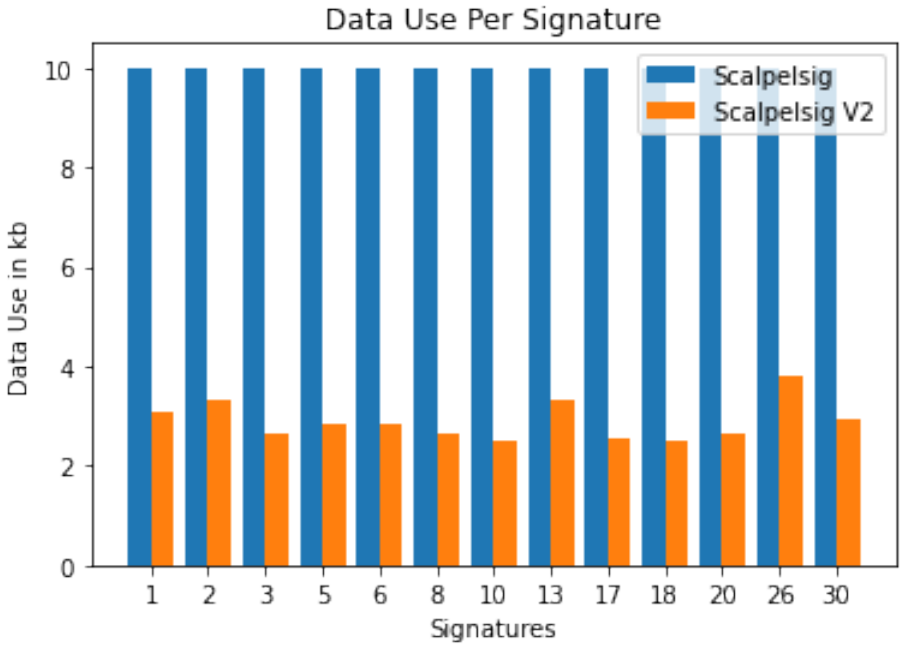


As we can see, ScalpelSig and ScalpelSig V2 (my augmented version) outperformed the whole exome sequencing baseline. While these results are excellent,

I am skeptical of the validity of the results, since WES uses more than 10 times the amount of data as our ScalpelSig panels. The ground truth results from WGS here were taken by manual procedure of taking the total mutation counts from the test set data, running it through SignatureEstimation, and extracting predictions. While I am hesitant to accept these performance results, I am pleased to see that ScalpelSig V2 has nearly identical performance to the original ScalpelSig, while using less data. This makes sense to me, since ScalpelSig V2 will select the best scoring window possible, and if the best score is obtained from a smaller compilation of mini-windows, then ScalpelSig V2 will choose that smaller compiled window set. Furthermore, SignatureEstimation is, essentially, just an estimation. So, the variance introduced when we apply our probabilistic scatter mutation function most likely isn't significant enough with the parameters I selected from this experiment to cause a significant difference in the overall activity predictions. In terms of running time, ScalpelSig V2 took about four times as long as the original ScalpelSig method to run the same experiments on. This is exactly in line with expectations, since I am essentially running ScalpelSig over a dataset that is four times as large since we evaluate four mini-windows per 10 kb window from our initial dataset.

Experimental Results Using Signature Estimations from nz_signature_estimation.tsv





These results using "nz_signature_estimation.tsv" as our ground truth mutational exposures seem much more reasonable. However, I am still skeptical about the outperformance on WES. For sanity, I have also included the AUPR

for each signature in my original ScalpelSig implementation and the WES benchmark AUPRS, alongside the AUPRs in the original ScalpelSig paper.

Signature	Original ScalpelSig AUPRS	WES AUPRS
1	0.9074	0.8618
2	0.6269	0.4445
3	0.6593	0.549
5	0.9693	0.944
6	0.0967	0.053
8	0.905	0.891
10	0.0036	0.0036
13	0.6565	0.4975
17	0.0107	0.0107
18	0.1155	0.1396
20	0.0083	0.0393
26	0.0223	0.025
30	0.2411	0.29

As we can see, the results from my ScalpelSig implementation is quite consistent with the results from the original ScalpelSig implementation, which is very encouraging. However, the WES results do not seem to match at all, indicating that perhaps the dataset provided by the ScalpelSig authors may not have been a great representation of the WES mutation exposures, or that the authors may have done something different than I did to compute the WES AUPRS.

Discussion

ScalpelSig V2 has yielded promising results in this study. However, the results we obtained were performed on simulated data using my scatter method to randomly distribute mutation counts over the mini windows. Further research is necessary on real data to see whether this truly is a viable approach to further cut down on the data usage of the ScalpelSig algorithm.

One drawback to the ScalpelSig V2 algorithm is that it takes significantly longer than the original ScalpelSig algorithm to train. This is expected, since we are taking multiple mini-windows and scoring each of them. However, for the use case of the ScalpelSig algorithm, this doesn't really matter too much since the goal of the algorithm is simply to design a genomic panel for clinical use. In practice, clinicians won't need to run the training algorithm, they simply need to take the designed genomic panel to output a prediction for activity levels.

One other direction for potential future studies is exploring other cancer types besides breast cancer. Right now, experimental data is limited to breast cancer samples, so we have no idea how ScalpelSig and ScalpelSig V2 will perform on

other cancer sets.

A final and very interesting direction for future research is exploring the ways we select mini-windows. One idea is to experiment with different parameter settings for the number of mini-windows we evaluate per window. In my experiments, I selected a window size of 4 just as a default setting and was able to achieve great results. Larger window sizes would require a steep runtime commitment to train, but it will likely result in ever further data reduction in the final genomic panel, since the mutation count matrix is very sparse which allows us to retrieve very specific areas of the genome to evaluate. However, this could also risk overfitting on the training dataset, so further explorations on this subject would be necessary.

Another potential idea in this direction is to evaluate all subset combinations of the mini-windows, rather than the first n contiguous mini-windows in each window group. This would allow us to extract even more specific regions of the genome, but again, this is prone to overfitting so further experimentation would be necessary to get the right parameter settings.

ScalpelSig V2 Repository

The ScalpelSig V2 implementation, as well as the experiments I ran can be found at:
https://github.com/Jamesjg2/ScalpelSigV2

References

[1] Donavan T. Cheng, Talia N. Mitchell, Ahmet Zehir, Ronak H. Shah, Ryma Benayed, Aijazuddin Syed, Raghu Chandramohan, Zhen Yu Liu, Helen H. Won, Sasinya N. Scott, A. Rose Brannon, Catherine O'Reilly, Justyna Sadowska, Jacklyn Casanova, Angela Yannes, Jaclyn F. Hectman, Jinjuan Yao, Wei Song, Dara S. Rong, Alifya Oultache, Snjezana Dogan, Laetitia Borsu, Meera Hameed, Khedoudja Nafa, Maria E. Arcila, Marc Ladanyi, and Michael F. Berger. Memorial sloan kettering-integrated mutation profiling of actionable cancer targets (msk-impact).

[2] Nicholas Franzese, Jason Fan, Roded Sharan, and Mark D.M. Leiserson. Scalpelsig designs targeted genomic panels from data to detect activity of mutational signatures.

[3] Thomas Helleday, Saeed Eshtad, and Serena Nik-Zainal. Mechanisms underlying mutational signatures in human cancers.