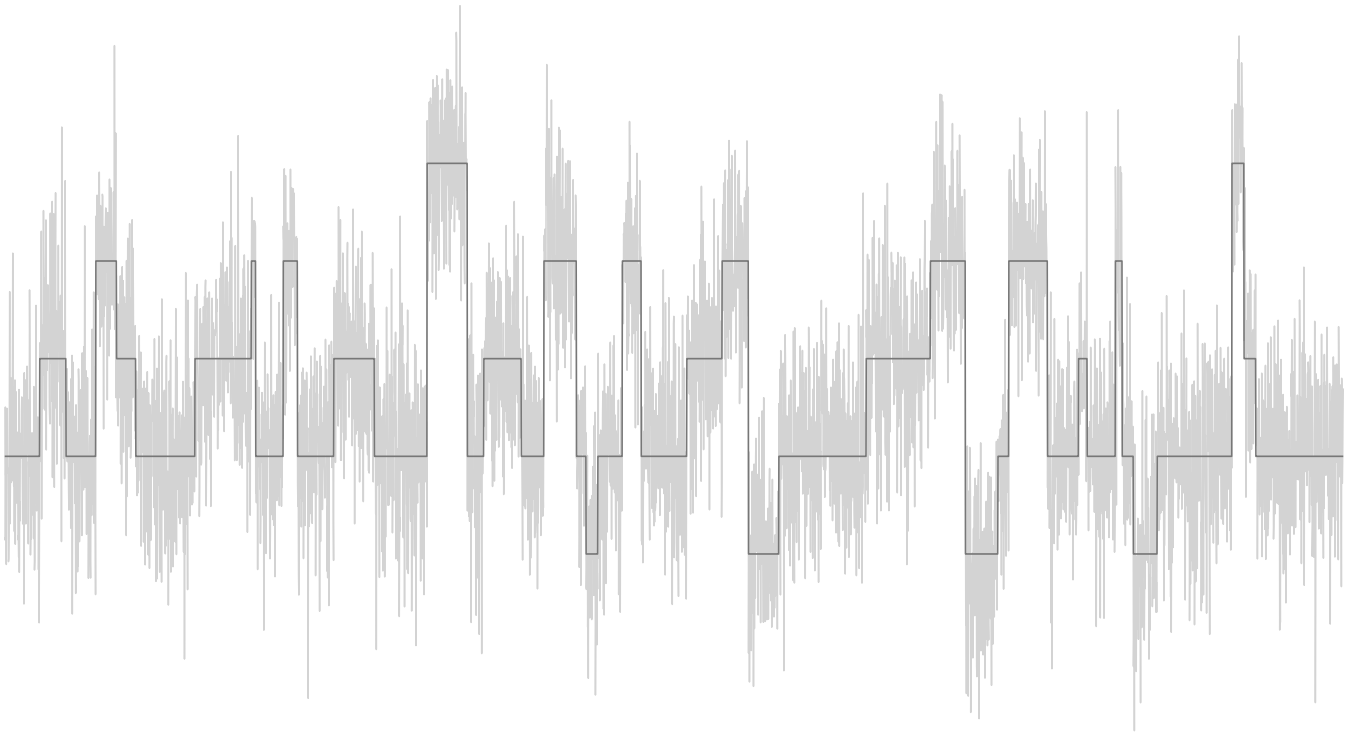


Deconvolution methods for quantification of copy number variations in liquid biopsy sequencing

Master's thesis in Engineering Mathematics and Computational Science

LOTTA ERIKSSON
LINNEA HALLIN

DEPARTMENT OF MATHEMATICAL SCIENCES
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2024
www.chalmers.se

Master's thesis 2024

Deconvolution methods for quantification of copy number variations in liquid biopsy sequencing

Lotta Eriksson
Linnea Hallin

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mathematical Sciences
Chalmers University of Technology
Gothenburg, Sweden 2024

Deconvolution methods for quantification of copy number variations in liquid biopsy sequencing
Lotta Eriksson
Linnea Hallin

Supervisor: Eszter Lakatos
Examiner: Erik Kristiansson

Master's Thesis 2024
Department of Mathematical Sciences
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: A simulated liquid biopsy measurement of a copy number profile.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2024

Deconvolution methods for quantification of copy number variations in liquid biopsy sequencing

Lotta Eriksson
Linnea Hallin
Department of Mathematical Sciences
Chalmers University of Technology

Abstract

Copy number variations, prevalent in cancers, are genomic alterations that result in losses or gains of entire genomic regions. Such alterations can be evaluated using cheap low-pass whole genome sequencing using liquid biopsies. These methods are promising for tracking the evolution of the cancer in real-time, due to their low cost and non-invasive nature which enable frequent sampling. The DNA sequenced from liquid biopsies is a mixture of cancer-specific DNA and DNA from healthy cells, the latter without these alterations. Therefore liquid biopsies can, for example, help monitor the proportion of an emerging cancer subtype in the tumor. Lakatos et al. [10] introduced methods for estimating the cancer proportion and the proportion of the most dominant cancer subtype in the sample, termed purity estimation and subclonal tracking. However, our ability to track the cancer evolution is hindered by the low signal in such samples, due to the contamination of healthy DNA, and measurement noise. We thus aim to develop methods for denoising and deconvolution of the underlying copy number profile of the tumor, to enhance the signal in liquid biopsy sequencing measurements.

In this work, we evaluate two frameworks for deconvoluting such samples: a denoising autoencoder and Bayesian change point detection. We compare these methods to rolling median-based segmentation, using the mean squared error of the reconstructed copy number profile and the F1-score. We demonstrate that both deconvolution methods work better than the rolling median in low-purity and noisy regions. We then implement our methods for purity estimation and subclonal tracking, based on the methods by Lakatos et al. and using the denoised data obtained from the previous step. In general, we find that Bayesian change point detection outperforms the other methods, is suitable for denoising liquid biopsy samples, and can be used for subclonal tracking. Using our full updated pipeline, we can improve the estimation of purity and subclonal ratio values, especially in low-purity and low-quality samples.

Keywords: Denoising autoencoder, Bayesian change point detection, cumulative segmented regression, genomics, copy-number variations, liquid biopsy sequencing

Acknowledgements

First and foremost, we wish to express our gratitude to our supervisor Eszter Lakatos. Thank you for our many interesting discussions, both about the thesis and life in general, and for your enthusiasm and positive outlook. Without you, this work would not have been possible.

Thank you to everyone in the Chalmers CODEc group, the Cvijovic Lab, and the Polster Lab for welcoming us into your little community with open arms. The Thursday meetings were always a great distraction from our daily work.

We want to thank our friends and families for their continuous support. Thank you for the pingis Mondays, the chats in the office, and the Tuesday Laplace fikas. A special thank you to Ruben for your exceptional support and for always having the solutions to our problems.

Finally, thank you, the reader, for showing interest in our work. Have a great read!

Lotta Eriksson and Linnea Hallin, Gothenburg, May 23, 2024

Contents

1 Introduction **1**
1.1 Aims and objective . . . 2

2 Theory **3**
2.1 Next generation sequencing . . . 3
2.1.1 Liquid biopsies . . . 3
2.2 Copy number variations . . . 4
2.2.1 Detecting copy number variations . . . 4
2.2.2 ENCODE blacklist . . . 5
2.2.3 Copy number profiles from liquid biopsies . . . 5
2.3 Analyzing copy number variations . . . 6
2.3.1 Purity estimation . . . 6
2.3.2 Subclonal ratio estimation . . . 7
2.4 Denoising autoencoders . . . 8
2.4.1 Autoencoders . . . 9
2.4.2 Loss function . . . 9
2.4.3 Convolutional neural networks . . . 9
2.4.4 Activation function . . . 10
2.4.5 Pooling layers . . . 11
2.4.6 Convolutional denoising autoencoders . . . 11
2.5 Change point detection . . . 11
2.5.1 Bayesian change point detection . . . 12
2.5.2 Cumulative segmented regression . . . 13
2.5.3 Model selection . . . 13
2.5.4 Diagnostics of segmentation . . . 14

3 Implementation **15**
3.1 Datasets . . . 15
3.1.1 Extension of the ENCODE-blacklist . . . 15
3.1.2 Simulation of synthetic data . . . 16
3.1.3 Simulation of longitudinal data . . . 16
3.2 Convolutional denoising autoencoder . . . 17
3.2.1 Activation function . . . 17
3.2.2 Loss function . . . 17
3.2.3 Parameter tuning . . . 18
3.2.4 Segmentation of denoised signals . . . 19

3.3 Bayesian change point detection . . . 19
3.4 Purity estimation . . . 20
3.5 Estimation of subclonal ratio . . . 20
3.5.1 Choice of threshold θ . . . 20
3.6 Model evaluation . . . 21
3.6.1 Rolling median denoising . . . 21
3.6.2 Evaluation metrics . . . 22

4 Results **23**
4.1 Deconvolution . . . 23
4.2 Purity estimation . . . 26
4.2.1 Synthetic dataset . . . 26
4.2.2 In silico dataset . . . 28
4.3 Subclonal ratio estimation . . . 28
4.3.1 Synthetic dataset . . . 29
4.3.2 In silico data . . . 29

5 Discussion **33**
5.1 Deconvolution models . . . 33
5.2 Estimation of purity and subclonal ratio . . . 34
5.3 Future research . . . 35

6 Conclusion **37**

Bibliography **39**

1 | Introduction

Cancer is a class of human pathologies, that claims millions of lives across the globe every year. Hence, it is of great interest to make identification, and tracking of cancer progression, more accessible and accurate. Sequencing of cell-free DNA (cfDNA) from liquid biopsies, in particular blood samples, is a promising tool for cancer detection and monitoring, as they are cheap, and less invasive than traditional tissue-based methods.

The driving force behind cancer is somatic mutations. One somatic mutation is *structural variations* that affect large genomic regions. Copy number variations (CNVs) are a type of structural variation that results in gains or losses of entire genomic regions [24]. CNVs are prevalent in cancer and are often exclusive to the tumor cell. Tumor cells frequently undergo necrosis and then release tumor-specific cell-free DNA (ctDNA) in the process, i.e. degraded DNA fragments, into bodily fluids. Therefore, CNVs can be evaluated using cheap low-pass whole genome sequencing (lpWGS) using liquid biopsies, which makes them a promising tool for tracking cancer progression due to their non-invasive nature.

A challenge when treating cancer is *resistance*. Some cells are resistant at the start of the treatment or become resistant after starting treatment through further mutations. We refer to an emerging, putatively resistant, cancer population as an emerging *subclone*. We are interested in studying the progression of such a subclone, as an increasing proportion of the subclone might indicate that the current treatment is insufficient. Lakatos et. al. developed methods for tracking the most pervasive subclone, using longitudinal lpWGS measurements of somatic CNVs [10].

Our ability to track the tumor progression is today hindered by the low signal in liquid biopsy sequencing data. Often, only a small proportion of the total cfDNA sequenced comes from the tumor. If the tumor consists of different types, such as the ancestral tumor and a resistant subclone, it will contribute to an even smaller proportion of the total DNA pool. The DNA reduces the signal from healthy cells, as they do not contain CNVs. Furthermore, lpWGS measurements from liquid biopsies are contaminated by measurement noise. Current methods for tracking cancer proportions from cfDNA samples require at least 5-10% of the sequenced cfDNA to originate from the tumor, which is often not the case [10]. It is difficult to distinguish true variations from random noise in the sequencing process at lower cancer proportions, see Figure 1.1. Hence, it is of interest to denoise liquid biopsy sequencing samples to be able to distinguish true CNVs at even lower cancer proportions than what is currently possible.

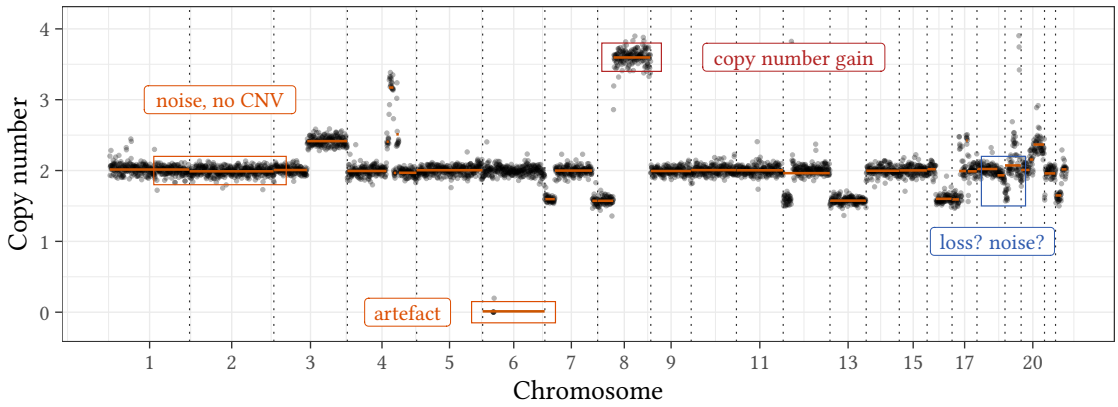


Figure 1.1: An example of a whole genome DNA copy number profile. The blue part in chromosomes 18 and 19 is not clearly distinguished; it could be either CNV or noise.

1.1 Aims and objective

Due to the high noise levels and generally low quality of liquid biopsy samples, this thesis aims to develop methods for noise removal, thereby increasing the quality of such samples. Using the denoised data, we aim to deconvolve the copy number profile of the genome, i.e. divide the genome into segments of equal copy number. Furthermore, we aim to use the processed data to achieve a higher level of accuracy in estimating the sample purity and subclonal proportion over time, especially when dealing with lower cancer proportions.

For this purpose, we employ two methods for noise reduction of liquid biopsy measurements; a neural network trained to identify and remove measurement noise, and a Bayesian change point detection to discern different segments, i.e., copy numbers, within the data. Following noise reduction, segmentation methods based on statistical modeling are applied to partition the genome into contiguous segments of homologous copy number state. We evaluate the method's suitability using the reconstruction error of the copy number profile and the segmentation quality using the F_1-score. We further assess the quality of the purity and the subclonal ratio estimation using both simulated and experimental data.

2 | Theory

In this section, we begin by briefly presenting the background of copy number variations, and how to detect them using liquid biopsy sequencing and their current limitations. Furthermore, we present current methods for estimating the cancer proportion and subclonal ratio in the tumor. The background of the denoising methods is then presented: a denoising autoencoder, a type of neural network trained to remove noise, as well as a Bayesian model for dividing the genome into segments of equal copy number.

2.1 Next generation sequencing

DNA sequencing is a laboratory technique for determining the sequence of nucleotides in a DNA molecule. So-called next-generation sequencing (NGS) has improved our ability to detect genomic variations [4]. NGS techniques sequence large amounts of DNA strands, so-called *reads*, that come from randomly fragmented copies of the genome. The reads are assumed to be random representations of the genome or a targeted genomic region [12]. The reads do not come with a genomic position, and must therefore be mapped to a reference genome to assign its position. The *coverage* refers to the number of reads that cover a nucleotide. We often consider the average coverage over the genome.

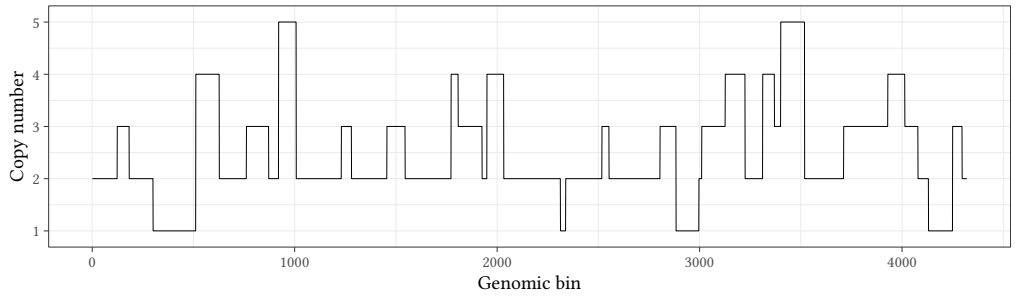
2.1.1 Liquid biopsies

Via apoptosis and necrosis of the cell, DNA fragments are shed into the bloodstream, called *cell-free DNA* (cfDNA) since the fragments are freely circulating. The same process occurs for tumor cells, resulting in cell-free circulating tumor DNA (ctDNA), which is detectable in patients for several different types of cancer and correlated to the stage of the disease [18]. The ctDNA contains tumor-specific abnormalities that are not present in the healthy DNA. A *liquid biopsy* is a test done on a sample taken from blood or other bodily fluid; in the field of cancer research, this is done to search for tumor content or other biomarkers [5]. NGS techniques can be used to sequence the cfDNA in liquid biopsies, referred to as *liquid biopsy sequencing*. Classical methods for cancer detection include tissue-based biopsies, which are often invasive and time-consuming. Liquid biopsies are more suited for frequent sampling, due to their non-invasive nature and easy procedure [11], making them a great alternative for tracking the development of cancer in patients, especially during treatment where this sort of information is desirable. We will subsequently by *liquid biopsy* refer only to testing done on blood samples.

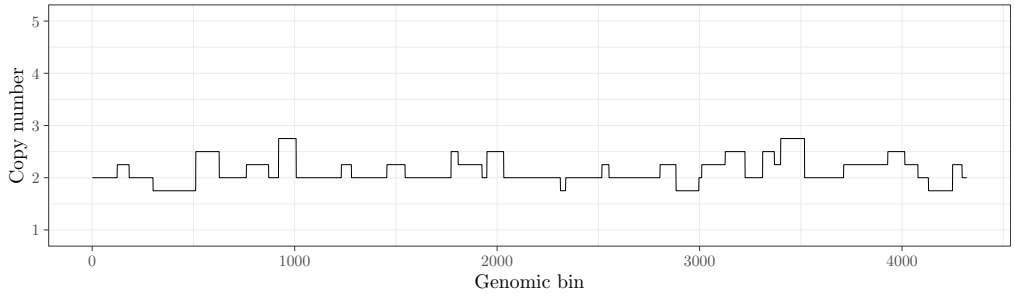
2.2 Copy number variations

The *copy number state* of a genome is the number of copies of each region of the genome. The copy number state of a healthy human should be 2 throughout the genome since the chromosomes come in pairs. A *copy number variation* (CNV) is the deviation from this diploid copy number state, which is caused by duplications or deletions of genomic regions, see Figure 2.1a. Such genomic variations are prevalent in cancer, with some variations ranging over entire chromosomes. CNVs are a good tool for tracking cancer, as these variations are generally exclusive to the tumor.

In liquid biopsies, the DNA found in the sample is a mixture of cancer DNA and DNA from healthy cells, where the proportion of cancer DNA is thought to correspond to the amount of cancer in the body. The proportion of cancer DNA in a sample is referred to as the sample *purity*. The copy number profile produced from liquid biopsy sequencing will not show the copy number profile of the tumor, but instead the average copy number profile of the cancer and healthy cells. We refer to the contribution of the healthy DNA as *normal contamination*. In practice, since the copy number of healthy cells is 2 across the genome, the resulting copy number profile is the one of the cancer DNA, with an amplitude decrease proportional to the purity, see Figure 2.1b.



(a) Underlying copy number profile



(b) Copy number profile with purity 25%

Figure 2.1: Copy number profiles with and without normal contamination

2.2.1 Detecting copy number variations

In the analysis of CNVs, we often use NGS to perform so-called whole genome sequencing (WGS), where the entire genome is sequenced. A subset of WGS is low-pass whole genome sequencing (lpWGS), where the average coverage is 0.001-1x We assume that

the coverage of a genomic region is linearly correlated with its copy number state, after correction of systematic biases. Instead of considering each nucleotide, we often divide the genome into discrete and non-overlapping genomic bins of fixed length, e.g. 50 or 500 kilobases (kb), and aggregate the reads mapped to each genomic region. Hence, we consider the copy number state of each genomic bin. After processing the data, by removing systematic biases, we perform segmentation to group genomic bins of equal copy number states into longer segments, and divide regions of unequal copy number states [12]. In this work, the processing of the raw sequencing data is done through the R package `QDNAseq` [20, 21], a method that can handle data down to 0.1x genomic coverage.

2.2.2 ENCODE blacklist

Apart from systematic biases in NGS data, there exists a set of regions with unstructured or high signals, independently of the experiment or cell line, called the ENCODE blacklist. To use NGS for reading out the genomic signal of an individual, we require high-accuracy mapping of the DNA to the reference genome, in our case the human genome, and accurate annotation of the reference. Some regions in the genome contain inconsistencies in the annotation, due to difficulty in the genome assembly, such as repetitive regions. Including such regions in the analysis can lead to inaccurate and biased results. Hence, the ENCODE blacklist was created to flag regions that appeared to have artifact signals [1].

2.2.3 Copy number profiles from liquid biopsies

The mathematical formulation of the copy number profile in this section follows the formulation in [10]. We assume that a tumor consists of two types of cells, both ancestral (A) and an emerging, putatively resistant, subclone (S). In addition, normal cells (N) contribute to the DNA pool. The different cells continuously shed cfDNA into the blood, with a proportion that varies over time. The purity of a sample at timepoint i is denoted p_i and the proportion of the DNA coming from the subclone is called the *subclonal ratio*, denoted r_i. The proportions of the cell types at a time point i are given by

$$N_i = 1 - p_i, \quad A_i = p_i \cdot (1 - r_i), \quad S_i = p_i \cdot r_i. \tag{2.1}$$

The copy number profile can be divided into distinct segments, i.e. genomic regions with homologous copy number states, with the majority of segments in a population remaining constant over time. The j'th segment of the copy number profile therefore consists of three time-independent copy number states $C(A)^j, C(S)^j$ and $C(N)^j$, corresponding to the copy number state of the ancestral, subclonal, and normal cells respectively. Hence, the measured copy number of a segment C_i^j in a sample i is a combination of these absolute copy number states, weighted by their respective proportion (N_i, A_i, S_i). We know that $C(N)^j = 2$, due to normal cells being in a diploid state. Using Equation (2.1) the measured copy number state of a segment j is

$$C_i^j = 2 + p_i \cdot \Big[(1 - r_i)C(A)^j + r_i C(S)^j - 2\Big] + \sigma_{ij}, \tag{2.2}$$

where σ_{ij} is the measurement noise in sample i at segment j. We note that the underlying absolute copy number states $C(A)^j, C(S)^j$, and $C(N)^j$ are integers, while the measurements are continuous.

2.3 Analyzing copy number variations

In this section, we describe the liquidCNA-algorithm introduced by Lakatos et. al. [10]: an algorithm for estimating the purity of a sample and one for estimating the subclonal ratios of a set of samples, obtained from the same patient at different times points.

2.3.1 Purity estimation

The purity estimation relies on the fact that copy numbers are *always* integers. Hence, the observed copy number distribution should have distinct peaks with a distance of p_i, see Figure 2.2. The noise broadens the set of copy numbers from a discrete set to a continuous scale. The modes are identified using a peak-finding algorithm.

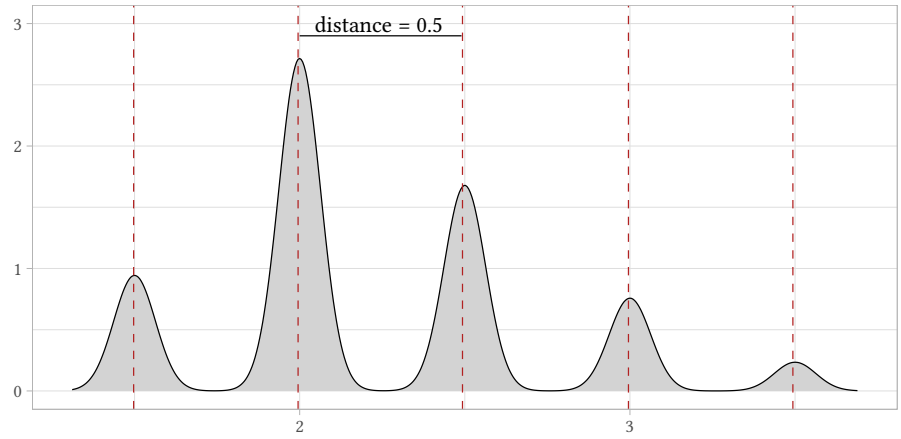


Figure 2.2: Smoothed distribution of the copy numbers in a sample of 50% purity. The distance between consecutive peaks is the sample purity.

To make the estimation robust, density smoothing is performed for a range of smoothing parameters. For each smoothing parameter s the distribution is adjusted so that the largest peak is centered at a copy number of 2. The observed modes of the distribution are compared to the expected modes $\{2 - p_i, 2, 2 + p_i, \dots\}$ for a range of purities $p_i \in [0.03, 1]$. The purity $\hat{p}_i$ that best describes the purity of the sample is the one that minimizes the summed squared distance between each mode and the closest observed mode

$$\hat{p}_i^{(s)} = \min_{p_i \in [0.03,1]} \sum_{C(A)} \min((2 + p_i(C(A) - 2)) - \text{modes})^2), \tag{2.3}$$

where $C(A)$ are the theoretically possible integer copy number states. The purity estimate $\hat{p}_i$ is taken as the mean or median of the best purities $\{\hat{p}_i^{(s)}\}$ across the smoothing parameters. The copy number profile can then be purity-corrected to remove normal

contaminations according to

$$C_{\text{corr}} = \frac{2}{\hat{p}_i} \cdot \left[\frac{C_{\text{seg}}}{M_{\text{max}}} - 1 \right] + 2, \tag{2.4}$$

where C_{corr} is the corrected profile, C_{seg} is the segmented profile and M_{max} is the maximum mode of the copy number distribution of C_{seg}. A simpler version would be to use the median instead of M_{max}, but for some samples, the median will be centered around a copy number level of 3 instead of 2, which will offset the correction.

2.3.2 Subclonal ratio estimation

The estimation of the subclonal ratio uses longitudinal data, where several samples are taken from the same patient at regular intervals during treatment. The idea behind the subclonal tracking is that the cancer subclone, resistant to the treatment, has subclone-specific variations that change over time in the subclone. The first step is to estimate the purity and remove normal contamination for each sample, as described in Section 2.3.1. The effect of an emerging subclone is visualized in Figure 2.3.

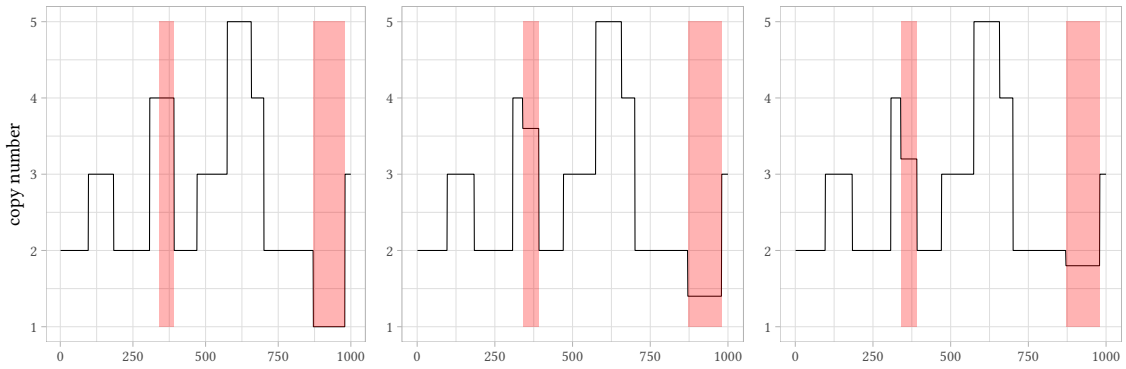


Figure 2.3: Progression of a cancer copy number profile over time. The red areas mark the variations of the subclone, which increase in relative size as the treatment progresses.

Ideally, we have a series of purity-corrected samples as in Figure 2.3. However, the segments do not align perfectly in real-world data, and we, therefore, calculate the union of all breakpoints and discard too short segments. In the next step, we categorize the segments into three types: *clonal* segments, which do not change over time given a purity correction; *subclonal* segments, which are segments where the subclone differs from the original cancer type; and *unstable* segments, which change over time, but are not related to the subclone. Unstable segments are a consequence of random mutations or measurement noise. Lakatos et. al. identify subclonal segments by first computing the difference to the *baseline sample*, often chosen as the sample retrieved at the start of treatment, as

$$\Delta C(T)_i^j = C(T)_i^j - C(T)_1^j = r_i(C(S)^j - C(A)^j), \tag{2.5}$$

where $C(T)_i^j$ is the tumor specific segment j in sample i. The subclonal segments are selected as a set of segments with a monotone pattern across *an ordering of samples* that

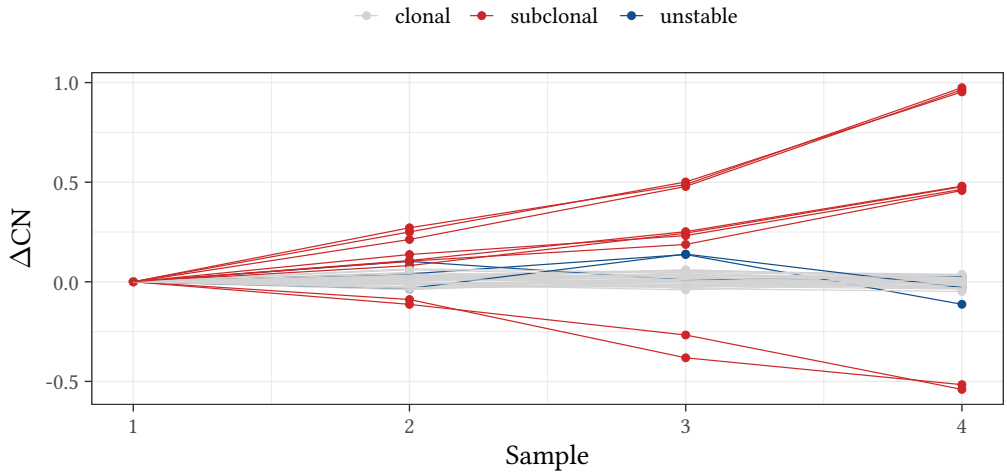


Figure 2.4: Differences between the samples 1–4 and the baseline sample 1. Each line represents a segment and the color represents the segment's classification. The samples are simulated to have a subclonal ratio of 0, 0.125, 0.25, and 0.5, respectively.

maximizes the number of subclonal segments, see Figure 2.4. For further reference, see the original paper. The subclonal ratio r_i is derived using Equation (2.5) where

$$\Delta C(T)_i^j \in \{\dots, -2r_i, -r_i, r_i, 2r_i, \dots\}, \quad \forall\, j \in \mathcal{S}, \tag{2.6}$$

where $\mathcal{S}$ is the set of subclonal segments. The values are fitted using a mixture of Gaussian distributions, with the mean of the Gaussians following Equation (2.6). The subclonal ratio is taken as the constrained mean parameter r_i of the Gaussian mixture that optimizes the fit. This procedure is performed to account for the measurement noise.

2.4 Denoising autoencoders

Autoencoders are algorithms that aim to learn an informative representation of the input data, by learning to reconstruct a set of input observations. Autoencoders are often used for unsupervised learning tasks such as dimensionality reduction and denoising. The typical autoencoder architecture consists of three parts: an encoder, a latent feature representation, and a decoder. The autoencoder should reconstruct the input sufficiently well, and at the same time, create a latent representation of the input that is meaningful and useful. The encoder and decoder parts of the structure are often neural networks [14]. The encoder acts as a feature extractor, and the decoder reconstructs the original signal from the latent feature representation. The dimension of the latent feature representation is often much lower than the input dimension, called a bottleneck. A *denoising autoencoder* is a special type of autoencoder that learns to remove noise from input observations from noisy data by learning to reconstruct the underlying signal from its noisy counterpart.

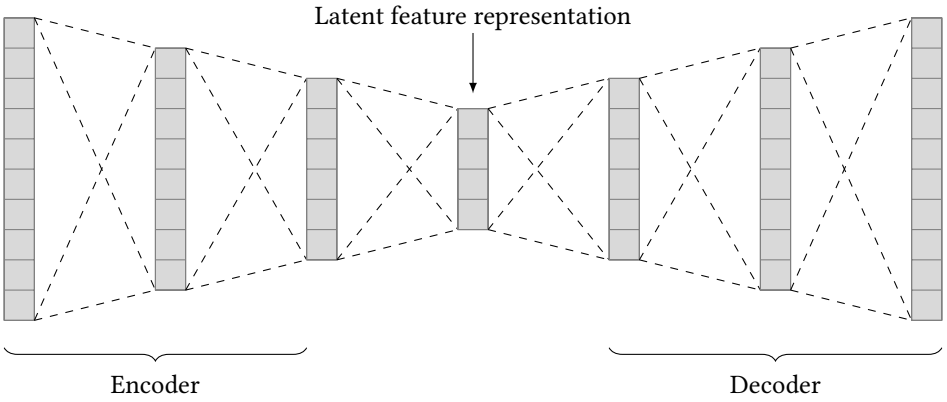


Figure 2.5: Schematic overview of the autoencoder architecture.

2.4.1 Autoencoders

In our setting, we have a training dataset consisting of M unlabeled observations $\boldsymbol{x}^{(i)} \in \mathbb{R}^N$ for $i = 1, 2, \ldots, M$ where N is the number of genomic bins. The latent feature representation $\boldsymbol{h}_i \in \mathbb{R}^q$ is the output of the encoder part of the network

$$\boldsymbol{h}_i = g(\boldsymbol{x}^{(i)}), \tag{2.7}$$

where $g : \mathbb{R}^N \to \mathbb{R}^q$ is a function dependent on some set of parameters. The output of the decoder $\tilde{\boldsymbol{x}}^{(i)} \in \mathbb{R}^N$ is a function of the latent feature representation

$$\tilde{\boldsymbol{x}}^{(i)} = f(\boldsymbol{h}_i) = f(g(\boldsymbol{x}^{(i)})). \tag{2.8}$$

The autoencoder aims to find the functions f, g that minimize the average discrepancy between the input signals $\boldsymbol{x}^{(i)}$ and its reconstruction $\tilde{\boldsymbol{x}}^{(i)}$ for $i = 1, 2, \ldots, M$.

2.4.2 Loss function

During training, the *loss function* is the metric aimed to be minimized. The loss function is a measure of the difference in the output and input signal:

$$\mathbb{E}[\Delta(\boldsymbol{x}^{(i)}, \tilde{\boldsymbol{x}}^{(i)})] = \mathbb{E}[\Delta(\boldsymbol{x}^{(i)}, f(g(\boldsymbol{x}^{(i)}))],$$

where $\Delta(\cdot, \cdot)$ is a function describing the difference between its inputs. We aim to find the weights in the network that minimize the difference between $\boldsymbol{x}^{(i)}$ and $\tilde{\boldsymbol{x}}^{(i)}$ according to this metric [14]. The most common loss function is the mean squared error (MSE).

2.4.3 Convolutional neural networks

Convolutional neural networks (CNNs) are a class of artificial neural networks designed to adaptively learn spatial hierarchies in the data. In this section, we present the major building blocks of CNNs. This section follows the presentation in [26].

The core building blocks of a CNN are *convolutional layers*, which perform feature extraction of the data and typically consist of both linear and non-linear components. The

linear components are convolution operations, that extract features by applying a *kernel*, i.e. an array of weights, across the data. Element-wise multiplication is performed between the kernel and each position in the data and is then summed up, to obtain the output value of the corresponding position in the output, called a *feature map*, see Figure 2.6. In general, this procedure is repeated multiple times to produce multiple feature maps, each representing different characteristics of the input vector. Hence, we can see the kernels as a form of feature extractors. The distance between two consecutive kernel positions is referred to as the *stride* and is usually set to 1, even though larger strides can be used to achieve down-sampling of the data. The kernels are shared across all sequence positions, referred to as weight sharing. One advantage of weight sharing is that local features are translation invariant as the kernel will move across the sequences and detect learned local patterns [19]. The kernel weights, vital for feature extraction, undergo learning throughout the network's training process.

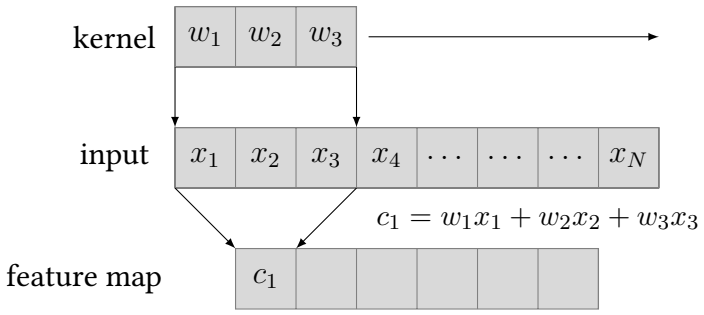


Figure 2.6: Schematic overview of convolution operation in a CNN. The kernel slides over the input and performs convolutions, to obtain the values in the feature map.

It is possible to use convolutional layers in an autoencoder-like architecture. Then, convolutional layers are used in the encoder, and so-called *transposed convolutional layers* are used in the decoder. Such layers perform up-sampling of the feature maps, by employing convolution-like operations. The deconvolution process contains trainable parameters, that are learned during the training of the network [27].

2.4.4 Activation function

Non-linearities can be introduced to neural networks through an appropriate choice of activation function, which allows the network to learn more complex and non-linear mappings between input and output. Let $\{x_i\}_{i=1}^{p}$ be the input received by a neuron in the neural network, coming from the system of connected neurons. Let $\{w_i\}_{i=1}^{p}$ be the weights that modify the received information, by either amplifying or reducing the values. The bias b modifies the weights of the connections between the neurons. The received values are, in turn, modified by the weights, and are summed to produce the *net-input*, which is passed on to the *activation function*, which determines if the neuron is activated or not [13], see Figure 2.7. A commonly used activation function is the ReLU-function, which is more computationally effective than other commonly used activation functions [19], where

$$\mathsf{ReLU}(x) = \max(x, 0). \tag{2.9}$$

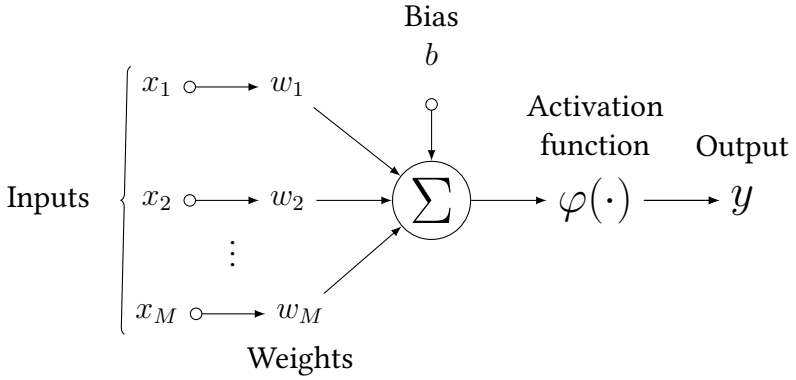


Figure 2.7: Schematic overview of a simple neural network.

2.4.5 Pooling layers

Another common method for achieving downsampling of the data, apart from using a larger stride, is to use *pooling layers*. Pooling layers are used to reduce the dimensionality of the feature maps to reduce the effect of small distortions and to decrease the number of parameters. The most common pooling method is *max pooling*, which extracts the largest value in a patch. Another commonly used pooling method is *average pooling* which instead extracts the mean of each patch.

2.4.6 Convolutional denoising autoencoders

In a convolutional denoising autoencoder, the encoder consists of convolutional layers that perform feature extraction, and pooling layers that downsample the input signal. The decoder requires upsampling to transform the downsampled signal into its original dimension. The so-called *transposed convolutional layer* generates an output feature map with a dimension larger than the input feature map. In between the transposed convolutional layers, we have upsampling layers for upsampling the dimension. The upsampling factor must be compatible with the downsampling performed in the pooling layers in the encoder part of the network.

2.5 Change point detection

Change points are points in a time series or other contiguous data where the underlying model or parameters change. The problem of detecting change points has applications in bioinformatics and genomics. An important application is multiple breakpoints detection, used to find the positions of CNVs in the genome. In this section, we present two statistical methods for detecting the number and positions of multiple breakpoints: *Bayesian change point detection*, a method based on Bayesian inference, which assumes i.i.d. Gaussian noise; and *cumulative segmented regression*, which is useful when the distribution of the noise is not necessarily i.i.d. Gaussian.

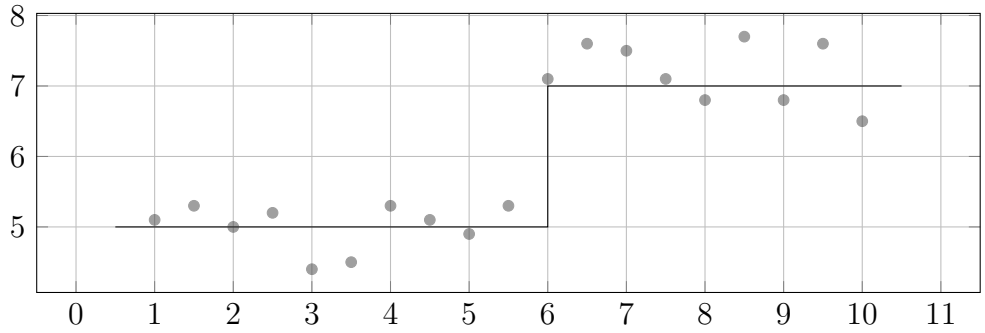


Figure 2.8: Normally distributed data with a change point in $x = 6$, where the mean changes from $\mu = 5$ to $\mu = 7$.

2.5.1 Bayesian change point detection

Bayesian change point detection was developed by Barry and Hartigan [2] and later implemented as an R-package by Erdman and Emerson [7]. The method combines Bayesian inference with the so-called product partition model (PPM) [3], which assumes a dataset can be partitioned into different blocks separated by the change points. Each partition is described by a multivariate probability distribution, whose parameters can be found by Bayesian analysis, see [2, 22] for further reference.

The Bayesian change point (BCP) model [2] is specified for the mean μ of a normal distribution, where it is assumed that the variance σ^2 is constant over time. The prior of the mean is $\mathcal{N}(\mu_0, \sigma_0^2)$, where μ_0 and σ_0^2 are unknown hyperparameters. The base assumptions are that (i) the probability of a change point at a position i is p, independently at each i, that (ii) the observations are $\mathcal{N}(\mu_i, \sigma)$ and that (iii) given the partition and the parameters, observations in different segments are mutually independent. The prior distribution of μ_{ij} (the segment beginning at $i + 1$ and ending at j) is $\mathcal{N}(\mu_0, \sigma_0^2/(j - 1))$, chosen such that long segments are favored, only including short segments given that there is sufficient data to estimate them. In the original algorithm, the calculations are $\mathcal{O}(N^3)$, and while an exact implementation is possible, it would be too slow for big datasets, thus the implementation is a Markov chain Monte Carlo (MCMC) approximation that is $\mathcal{O}(N^2)$ instead, where N is the number of genomic bins.

The partition of the data is denoted by $\rho = (U_1, U_2, \ldots, U_N)$, where $U_i \in \{0, 1\}$ and $U_i = 1$ indicates a change point at position $i+1$. Random partition samples are generated through a Markov chain. The partition is initialized as $U_i = 0$ for all $i < N$ and $U_N \equiv 1$. We generate a new partition by iterating through the old partition and at each position i we assign U_i with the conditional probability p_i where

$$\frac{p_i}{1 - p_i} = \frac{P(U_i = 1|\mathbf{X}, U_j, j \neq i)}{P(U_i = 0|\mathbf{X}, U_j, j \neq i)} \tag{2.10}$$

whose exact analytical form can be found in the article by Barry and Hartigan [2]. This form contains integrands which are numerically unstable for long sequences, and they are thus simplified by Erdman and Emerson [7] as incomplete beta integrals. After each iteration, the posterior means are updated conditional on the current partition.

This MCMC implementation of the algorithm in Barry and Hartigan [2] estimates the

posterior distribution of the change points and the means μ_{ij}. It thus not only produces a segmentation but also indicates the uncertainty of the segmentation. Furthermore, it is possible to perform change point detection with multiple samples simultaneously, to detect change points that are present in all samples, subsequently referred to as *multi-sample BCP*.

2.5.2 Cumulative segmented regression

Let $\{(x_i, y_i)\}_{i=1}^N$ be the observed data at hand, which represents the genomic bin and corresponding measured copy number. The observed copy number profile is a piecewise constant function with $K + 1$ segments

$$y_i = \beta_0 + \sum_{k=1}^{K} \beta_k \mathbf{1}\{x_i > \tau_k\} + \varepsilon_i, \tag{2.11}$$

where ε_i is the noise and $\{\tau_i\}_{i=1}^K \subseteq \{x_i\}_{i=1}^N$ are the breakpoints. Many methods for segmentation assume i.i.d. Gaussian errors, but the following method does not require independent and homoscedastic errors, only $\mathbb{E}[\varepsilon_i] = 0$. The associated statistical problem is to identify the number of breakpoints K in the sequence, and their positions [17].

The cumulative segmented algorithm is based on transforming the piecewise constant function in Equation (2.11) into a piecewise linear function. We transform Equation (2.11) by taking the cumulative sum on both sides to obtain

$$z_i = \beta_0 + \sum_{k=1}^{K} \beta_k (x_i - \tau_k)_+ + \eta_i, \tag{2.12}$$

where for each i and k

$$z_i = \sum_{j=1}^{i} y_j, \quad x_i = i = \sum_{j=1}^{i} 1, \quad \eta_i = \sum_{j=1}^{i} \varepsilon_i, \quad (x_i - \tau_k)_+ = \sum_{j=1}^{i} \mathbf{1}\{x_j > \tau_k\}. \tag{2.13}$$

Note that Equation (2.13) has the same parameters as Equation (2.11), but different errors, covariates, and responses. Most importantly, Equation (2.13) has a piecewise linear relationship, and a computationally efficient change points detection algorithm that does not work for the piecewise constant models can be employed, see [15, 16] for further reference. Such algorithms are based on, given a set of starting values for the change points, iteratively fitting a linear model, and updating the proposed changepoints until convergence. As the number of starting points is unknown in practice, the algorithm is initiated by overestimating the number of starting points K^*. Estimations of such models are performed using least squares, to ensure unbiased estimates, by requiring only zero mean errors. If the original data fulfills this assumption, so does the transformed data [17].

2.5.3 Model selection

The number of breakpoints selected using the method briefly presented in Section 2.5.2 is likely larger than the true number of breakpoints, so the next step is to select the

number of breakpoints, which is a model selection problem. The LARS algorithm, see [6] for reference, will return the entire path, from the null model to the model with K^* breakpoints included, at the cost of a single least-squares computation. The best model, having $\hat{K} \leq K^*$ breakpoints, can be selected using a suitable model selection criterion. A commonly used model selection criterion is the Bayesian Information Criterion (BIC) which penalizes too large models. We use the modified BIC value

$$\mathsf{BIC}_{C_N} = \log(\hat{\sigma}) + k \cdot \frac{\log(N)}{N} \cdot C_N \tag{2.14}$$

where $\hat{\sigma}$ is the residual variance estimate, $k = 1 + 2 \cdot \#\text{change points}$ is the number of model parameters, and C_N is a known constant [17]. We use model selection to reduce the risk of overfitting, which in this case corresponds to introducing too many segments.

2.5.4 Diagnostics of segmentation

Assigning a position in the genome as a breakpoint can be seen as a binary classification problem. To measure the performance of the segmentation method, we can therefore use the metrics *precision* and *recall*. Precision measures the percentage of relevant instances among all retrieved instances, i.e. the proportion of correct positive classifications. Recall is the percentage of identified instances among the relevant instances, i.e. what proportions of positives were identified correctly. Let TP, FP, and FN denote true positives, false positives, and false negatives, respectively. Then

$$\mathsf{Recall} = \frac{\mathsf{TP}}{\mathsf{TP} + \mathsf{FP}}, \quad \mathsf{Precision} = \frac{\mathsf{TP}}{\mathsf{TP} + \mathsf{FN}}. \tag{2.15}$$

For a perfect classification, precision and recall will both be 1, as there are no false positives or false negatives. Usually, that is not the case, and instead, there is a trade-off between precision and recall, where a decrease in false positives implies an increase in false negatives and vice versa. A relevant measure that accounts for both the precision and recall is the F_1*-score*, given by

$$\mathsf{F}_1\text{−}\mathsf{score} = 2 \cdot \frac{\mathsf{Precision} \cdot \mathsf{Recall}}{\mathsf{Precision} + \mathsf{Recall}}. \tag{2.16}$$

3 | Implementation

In this section, we present our implementation of the methods developed for the denoising and deconvolution of copy number variations in liquid biopsy data. We first present the implementation of the two pipelines used; a denoising autoencoder followed by cumulative segmented regression and BCP and how to validate the model performance. Then, we describe an improved implementation of liquidCNA used for purity- and subclonal ratio estimation. For details on the code, see our GitHub repository [8].

3.1 Datasets

We have four types of experimental data at hand, each with the genome divided into 500kb bins, with unknown noise distribution:

1. ovarian cell line data;
2. an in silico subsampled version of the dataset (1), based on $\{5, 10, 20, 50\}$ million reads;
3. lung cancer data, collected as part of the FIGARO study [23];
4. blood samples from healthy individuals, used as a negative control.

We refer to [10] for further details on collecting and processing the experimental data. The samples in dataset (1) were created by mixing cancer DNA and DNA from healthy blood, at proportions (weights) corresponding to the purity. We are expected to underestimate the purity due to the definition of theoretical purity of mixed samples made by the authors; a highly duplicated genome will have a larger weight than a diploid genome. Hence, when mixing the different samples, we will have a larger proportion of the healthy genome than expected. See the original paper for further reference.

3.1.1 Extension of the ENCODE-blacklist

We perform outlier detection using the experimental data. Problematic regions were detected using the negative control data, where a bin was considered an outlier if it deviated more than 3 standard deviations from the mean. Such data should in theory have no outliers, and the outliers were therefore blacklisted and removed. In our case, we found extreme outliers resulting from a read count close to zero. We further validated these outliers by analyzing cell line and FIGARO data, where they were also present. With this evidence, we extended the ENCODE blacklist to include these newly found problematic regions.

3.1.2 Simulation of synthetic data

To train the autoencoder, and evaluate the methods, we simulate representative data of highly mutated genomes. We note that experimental data is preferred, but not available. We simulate a genome of $N = 4320$ genomic bins, which roughly corresponds to dividing the genome into 500kb bins, after removing the extended ENCODE blacklist, see Section 3.1. We simulate segments with copy numbers between 1 and 5, with 2 being the most probable state, and duplications being more probable than deletions. The segment length is drawn uniformly between 12 and 130. The simulation parameters are chosen to be comparable with the original liquidCNA algorithm. We focus our simulation on purities of 5-50%, as current methods generally work well for high-purity samples. After simulating the underlying copy-number profile, we contaminate it with Gaussian noise to make the learning more robust.

The noise is assumed to be dependent on the copy number. The noise function is created so that our noise levels are directly comparable to those used in Lakatos et al. [10]. For a segment j with copy number C^j the noise standard deviation σ is defined as

$$\sigma = 0.1 \cdot \sigma_0(1 + 0.075 \cdot C^j), \tag{3.1}$$

where

$$\sigma_0 = \max(0.01, X), \quad X \sim \mathcal{N}(\sigma_{\text{mean}}, \sigma_{\text{sd}}^2), \tag{3.2}$$

and σ_{mean} and σ_{sd} is the pre-defined mean and standard deviation at a specific noise level. In this work, we use $\sigma_{\text{mean}} \in \{1, 2, 4, 8\}$. Noise level 1 approximately reflects the expected noise in practice, noise level 2 twice the expected noise, and so on, hence the scaling factor 0.1 in Equation (3.1). We are using a 10 times larger bin size than in [10], and therefore perform an extra downsampling step: for each position in the segment, 10 values are simulated with noise $\mathcal{N}(0, \sigma^2)$, and the value of the downsampled observation is their mean.

3.1.3 Simulation of longitudinal data

The simulation is done similarly as described in Section 3.1.2. We simulate a baseline copy number profile, which will be used as the first sample and as the basis for the other sample profiles. The baseline can be seen as the profile of the ancestral tumor. For each segment in the baseline profile, we sample a variation $\{-1, 0, 1, 2\}$, with a probability of 0.75 to draw a zero. For the remaining variations, we assign the single duplication the highest probability, followed by a single deletion with the next highest probability (values 1 and -1 respectively), and assign the smallest probability to a value of 2. The total profile in sample i is computed as the weighted mean of the ancestral and subclonal profile, with weights $(1 - r_i)$ and r_i respectively. The procedure is performed for each timepoint $i = 2, 3, \ldots, n$ where n is the number of longitudinal samples (including the baseline).

Furthermore, there is randomness in the assigned purity p_i, and subclonal ratio r_i of a sample i. In practice, we do not use baseline samples of purity less than 15%. Therefore, we only simulate baseline samples of at least 15% purity. In our simulations, the subclonal ratio of the baseline sample is always zero. The purities and subclonal ratios

of the remaining samples are sampled uniformly from $[0.12, 0.46]$ and $[0.05, 0.8]$ respectively. However, samples with an estimated purity of 10% or less are disregarded in the subclonal ratio estimation algorithm to ensure sufficient sample quality.

3.2 Convolutional denoising autoencoder

The first method used to denoise liquid biopsy samples is a convolutional denoising autoencoder. The autoencoder is implemented in `Tensorflow` v2.16.1 [25]. The encoder part of the autoencoder consists of convolutional layers, followed by pooling layers, to perform feature extraction and down-sampling of the input sequence. The decoder instead uses transpose convolutional layers and upsampling to reconstruct the underlying signal from the latent feature representation. The architecture is symmetric around the bottleneck. The number of layers, the feature size, the pooling method, and the kernel size are all tuning parameters. To select an optimal model for our task, we implement a tuning scheme where we tune said parameters, see Section 3.2.3.

3.2.1 Activation function

As an activation function, we use the ReLU function, shown in Equation (2.9). While commonly used, it suffers from the *dying ReLU problem*, which refers to neurons becoming inactive, and only outputting zero, independently of the input. This is considered as a dead state of the neuron, and results in no weights being updated, due to the derivative being zero, causing the gradient to fail in the backpropagation. An alternative to the ReLU function is

$$\mathsf{LeakyReLU}(x) = \max(x, \alpha x) \tag{3.3}$$

that does not assign zero to negative input values which solves the problem, at the expense of reduced performance. We explored using Equation (3.3) with $\alpha = 0.02$ to avoid the dying ReLU problem, but noticed a performance decrease. Another approach to avoiding this problem is to reduce the learning rate. We found that reducing the learning rate from 10^{-3} to 10^{-4} helped in avoiding the problem, which was then used in the final implementation of the model instead of LeakyReLU.

3.2.2 Loss function

As a training objective to be minimized, we use the sum of the *mean squared error* (MSE) and the *total variation loss*. Let $\boldsymbol{y} = (y_i)_{i=1}^N$ and $\hat{\boldsymbol{y}} = (\hat{y}_i)_{i=1}^N$ be the true and predicted values respectively. Then we define the loss function as

$$\mathcal{L}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 + \frac{\lambda}{N-1}\sum_{i=1}^{N-1}|\hat{y}_{i+1} - \hat{y}_i|, \tag{3.4}$$

where $\lambda \in [0, \infty)$ is a regularization parameter, that needs to be tuned. The second term is used to preserve sharp edges in the underlying signal. This is suitable since the function we want to reconstruct is piecewise constant.

3.2.3 Parameter tuning

To obtain the best model, given our simulated copy number data, we tune the number of layers to use, the pooling method, the initial feature size, and the kernel size. The values used for each of the parameters are presented in Table 3.1. Here, the number of layers refers to the number of hidden layers in the network. After each added hidden layer, we add a pooling layer, that will down-sample the feature maps by a factor of 2. Furthermore, the feature size refers to the number of feature maps used in the first layer. The feature size is divided by 2 for each added hidden layer. For each added layer in the encoder, an up-sampling layer followed by a transposed convolutional layer is added in the decoder.

In earlier trials of the tuning, two pooling methods were used: max pooling and average pooling. As the results of the two were quite similar, with max pooling performing slightly better, and as the tuning is very time-consuming, we decided to only include max pooling in the tuning. Moreover, we also included up to 4 layers in the initial stage of tuning, but we found that the performance was reduced, likely due to overfitting. Moreover, it is time-consuming to train such a complex model, and we decided to exclude it in the final tuning to reduce the time spent on tuning.

Table 3.1: Range of values used in tuning of autoencoder

Parameter	# layers	feature size	kernel size
Values	2, 3	32, 64, 128	5, 7, 9

The autoencoder was tuned using 2000 samples simulated according to Section 3.1.2, with $\sigma_{\text{mean}} = 0.1$ and $\sigma_{\text{sd}} = 0.01$. All training data is normalized into $[0, 1]$ using min-max normalization to help with convergence. The tuning itself was done exhaustively, with a 10-fold cross-validation for each set of parameters. Finally, the model was trained for at most 10 epochs. To avoid overfitting of the model, we implemented *early stopping*, which stops the training early, if the validation loss is non-decreasing in two consecutive epochs. During training, we used an 80/20 split of the training data to monitor the validation loss for this purpose.

The tuning was repeated for a sequence of penalty values $\lambda \in \{0, 0.025, \ldots, 0.1\}$. This range of relatively low penalty values was used as we found that they outperformed large penalty values in earlier stages of the tuning. The downside to using Equation (3.4) as the loss function, is that models trained using different λ are not directly comparable. We would likely favor the smallest λ, as the total variation term is automatically smaller. We select the best model for each λ and refit the models on the dataset. To select the best λ, we want to find the model that performs best, in terms of purity-corrected reconstructions of the copy number profiles. We measure this by the MSE between the original and the reconstructed profile, as described in Section 2.5.4. The final model chosen from the tuning process uses an initial feature size of 128, kernel size of 7, 2 hidden layers, and a penalty value of 0.025. We note that the model performance was not sensitive to the hyperparameters used.

After additional evaluation, we decided to retrain the model on more data and with varying noise levels, as overall performance increased with more diverse training data. The model was retrained on trice as much data, and Gaussian noise with varying means $\{0.05, 0.1, 0.15\}$ and standard deviations $\{0.005, 0.01, 0.015\}$, with 200 samples per purity.

3.2.4 Segmentation of denoised signals

To retrieve the underlying piecewise constant function from the denoised data, we used the cumulative segmentation algorithm described in Section 2.5.2. The method will likely overestimate the number of breakpoints, so we perform model selection using the proposed Equation (2.14) in [16] with constant $C_n = \log(\log(n))$ as selection criteria. The idea is then to identify an "elbow" in the BIC-curve plot, which is often a subjective procedure. We use an automatic cutoff rule based on the percentage decrease in the BIC between consecutive values. We select the smallest number of segments where the percental decrease between two consecutive values is 1%. It is generally better to overestimate the number of breakpoints. However, too short segments need to be merged into longer segments, or filtered away before downstream analyses. We therefore merge segments shorter than 12 bins into the closest surrounding segment, and the value of the segment is recomputed as the weighted mean of the two merged segments, where the weight is the length of the segment. Segments are merged iteratively until no segments are shorter than 12 bins long.

3.3 Bayesian change point detection

As an alternative to the denoising autoencoder, we use the Bayesian approach for signal denoising, as described in Section 2.5.1. We use the MCMC implementation described in [7] for the Bayesian change point detection (BCP). Contrary to the previous section, we do not use any denoising of the data before segmentation, as we found BCP to work well even with noisy measurements and low purities. Conversely, as BCP assumes Gaussian distributed noise, pre-processing of the noisy data might worsen the performance of the BCP segmentation.

The output of the Bayesian change point detection is not only the posterior means and variances over the segments but also the posterior probability of a position $i \in \{1, 2, \ldots, N\}$ in the binned genome being a change point. The posterior probability is used to segment the data. We remove changepoints with a low posterior probability of being a change point and merge short segments as described in Section 3.2.4. We use a cutoff of $\varepsilon = 0.1$ for the posterior probabilities, as this worked well in practice in combination with merging or filtering of short segments. Finally, the means of the segments are recomputed, as the mean of the observations in each segment, to produce more reliable segment levels.

3.4 Purity estimation

The purity in the samples is estimated following the same principle as described in Section 2.3.1. We implemented a peak-finding algorithm that identifies local maxima by analyzing the second derivative of the smoothed density function, instead of employing the peak-finding algorithm used in liquidCNA. Peaks identified further away than 2.5 standard deviations from the mean were discarded, as these often were unreliable. The density smoothing was performed for $s \in \{0.6, 0.8, \ldots, 1.8\}$, and $\hat{p}_i$ taken as the median across the smoothing parameters. For each s, we use a purity interval of $[0.01, 1]$ with 200 equidistant values.

3.5 Estimation of subclonal ratio

To track emerging subclones, we implement an alternative version of the liquidCNA approach to subclonal ratio estimation presented in Section 2.3.2. Firstly, the purity is evaluated as in Section 3.4, using multi-sample Bayesian change point detection, and the profiles of the original data are corrected according to Equation (2.4). We then use the BCP segmentation to find the common breakpoints in the samples, where for each sample a breakpoint is identified if the posterior probability is below the cutoff $\varepsilon = 0.1$. We take the union of the breakpoints and then discard segments shorter than 12 bins, i.e. no merging of segments is performed. For each sample, we compute the segment values as the mean of the purity-corrected original data. In the following part, only one value per sample and segment is used, i.e., segment length is no longer considered.

We choose the first sample, taken at the beginning of the treatment, as the baseline sample and compute the differences $\Delta C(T)_i^j$ to the baseline. The baseline sample is assumed to contain a negligible amount of subclonal cancer. A segment is classified as clonal if

$$|\Delta C(T)_i^j| < \theta\,, \tag{3.5}$$

for all samples, where θ is a threshold parameter. The non-clonal segments are classified as either subclonal or unstable, based on whether they follow a monotone pattern after a sample reordering as in Section 2.3.2. A segment j is classified as unstable if

$$\Delta C(T)_i^j - \Delta C(T)_{i+1}^j > -\delta \quad \text{or} \quad \Delta C(T)_i^j - \Delta C(T)_{i+1}^j < \delta,$$

where δ is a threshold parameter for the monotonicity. Finally, the estimation of the subclonal ratio is based on peak distances, inspired by the purity estimation. The modes are identified as in Section 3.4, and the optimal ratio minimizes the squared difference. An illustration of the results can be seen in Figure 3.1 for a sample with subclonal ratio 12.5%. To make the estimation more robust we use a range $s \in \{0.5, 0.75, \ldots, 2.5\}$ of different smoothing factors, and select the optimal ratio $\hat{r}_i$ as the median of the estimations $\{\hat{r}_i^{(s)}\}$ across the different smoothing parameters. This is repeated for all non-baseline samples.

3.5.1 Choice of threshold θ

In manual analysis, where only one patient is examined, the threshold θ should be chosen such that it separates the clonal from the subclonal segments, often visible by ocular

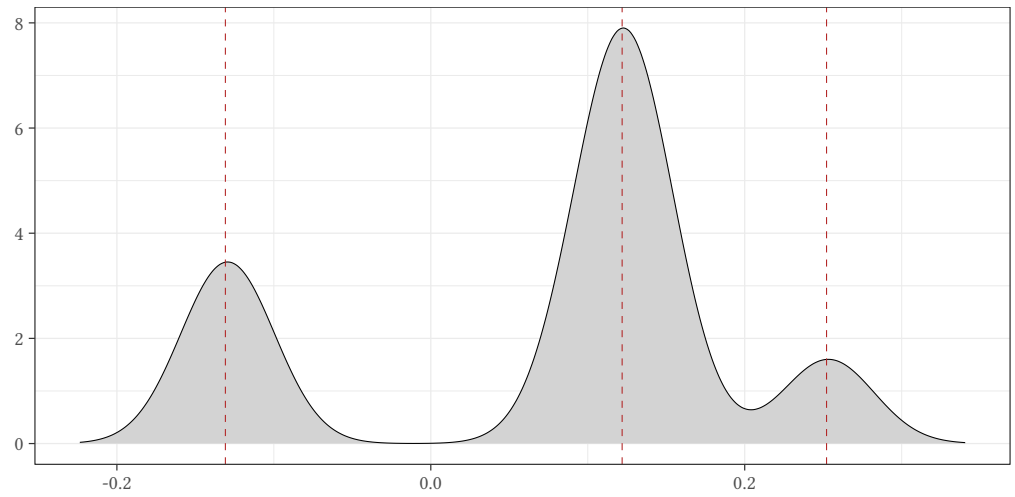


Figure 3.1: Smoothed distribution of subclonal segments with subclonal ratio 12.5%.

inspection. However, in the automated process, we found that the choice of threshold can be tricky. Inspired by liquidCNA [10], we choose the threshold in the following way: For each $\theta \in \Theta$, where Θ is a set of possible threshold values, and for each permutation of the samples, we classify the segments into clonal, subclonal, and unstable as above. We then choose the permutation with the highest ratio of subclonal to unstable segments. Out of these θ with a given ordering, we choose the θ with the highest subclonal to unstable ratio, given that at least 5 and at most 20 subclonal segments are above the threshold.

A naive choice of Θ would be $\Theta = [0, 1]$, including all possible ratios, however, as previously stated, in the case of manual analysis, the optimal threshold is usually clearly visible to the human eye. In theory, the optimal choice of threshold, following Equation (3.5), is the maximal true ratio of the samples in a set, r_{max}. To both simulate the range of human error and save computational time, we use $\Theta = [r_{\text{max}} - 0.2, r_{\text{max}} + 0.1]$. This will in some cases improve the results and in many cases provide the same results but faster, and the improvement in results could easily be done by manual tuning according to the diagnostics plots.

3.6 Model evaluation

In this section, we describe the workflow used for evaluating the denoising methods. We use the rolling median as a base model for comparison. We also describe the experimental data at hand, and how the models are evaluated on the experimental data.

3.6.1 Rolling median denoising

A baseline denoising method is the rolling median. The size of the sliding window k was tuned on simulated data with different noise levels $\sigma \in \{1, 2, 4, 8\}$ and purities in $[0.05, 0.5]$. We tried a range $k \in \{11, 21, \ldots, 51\}$ and selected $k = 21$. We note that the optimal choice of sliding window is heavily dependent on the noise level in the sample,

and the purity. We selected $k = 21$ as this performed well, in terms of purity-corrected segmented signals, in the most realistic cases, and for purities high enough for use in downstream analyses.

3.6.2 Evaluation metrics

The models are compared using two metrics: purity-corrected copy-number profiles, and the F1-score. The corrected profiles indicate how well we are reconstructing the magnitude of the copy numbers, while the F1-score measures segmentation quality. The denoised signal is evaluated by first applying the denoising method, then estimating the purity of the sample, performing the segmentation, and finally correcting the segmented profile according to Equation (2.4). Then, the MSE of the purity-corrected profile is computed. This metric is suitable since we need an accurate and detailed reconstruction to estimate the subclonal ratio r_i in further analyses. We also compute the predicted change points from the segmented profiles and the F1-score, according to Equation (2.16). We define the predicted change point as a true positive if it is $+/-$ 5 bins from the true breakpoint.

4 | Results

This section outlines three main findings: model evaluation, purity estimation, and subclonal ratio estimation. We begin by comparing the performance of our denoising methods against the baseline method, rolling median denoising, in what we term model evaluation. Following this, we analyze purity estimation across synthetic and in silico datasets, each with varying noise levels and numbers of reads, for all three methods. Finally, we present the results of subclonal ratio estimation using the BCP method across synthetic and in silico datasets.

4.1 Deconvolution

The method comparison between the denoising methods, in terms of the absolute error of the purity-corrected CNVs and the F_1-score, is presented in Figure 4.1 and Figure 4.2 respectively. Recall that the performance is evaluated at purities $\{0.05, 0.1, \ldots, 0.5\}$. For ease of visualization, we group the purities into intervals of range 0.1; each purity interval contains two distinct purity values. In terms of reconstruction error, the denoising autoencoder performs better than the base model, i.e. the rolling median denoiser, in the low purity and high noise scenarios. Nevertheless, both denoising methods are outperformed by BCP at noise levels $\{2, 4\}$. We note the lack of significant increase in performance at noise level 1 and purities above 30% using the developed methods compared to the rolling median. However, the rolling median is not sufficient in noisier scenarios.

When instead considering the F_1-score, BCP consistently outperforms the other denoising methods, see Figure 4.2. This is even though the cutoff was not optimized for each purity level, something which does not affect the reconstruction error much but might affect the F_1-score since e.g. setting a too-high cutoff will underestimate the number of segments and produce a suboptimal segmentation and consequently reduce the F_1-score. A suboptimal cutoff can explain the BCP results in Figure 4.2 where the F1-score tends to be higher at noise level 2 or 4 than at noise level 1 for some purity intervals, e.g. $(0.3, 0.4]$ and $(0.4, 0.5]$. This indicates that we need another cutoff ε when working with low-noise samples.

As mentioned, the base model excels at high purity and low noise; to explain this, an example of the method prediction and the ground truth in this scenario is presented in Figure 4.3. The change points are identifiable by the eye, and no advanced models are needed. The predictions in this case are close to indistinguishable. Furthermore, the purity estimation in this scenario is reliable, and the difference in performance is

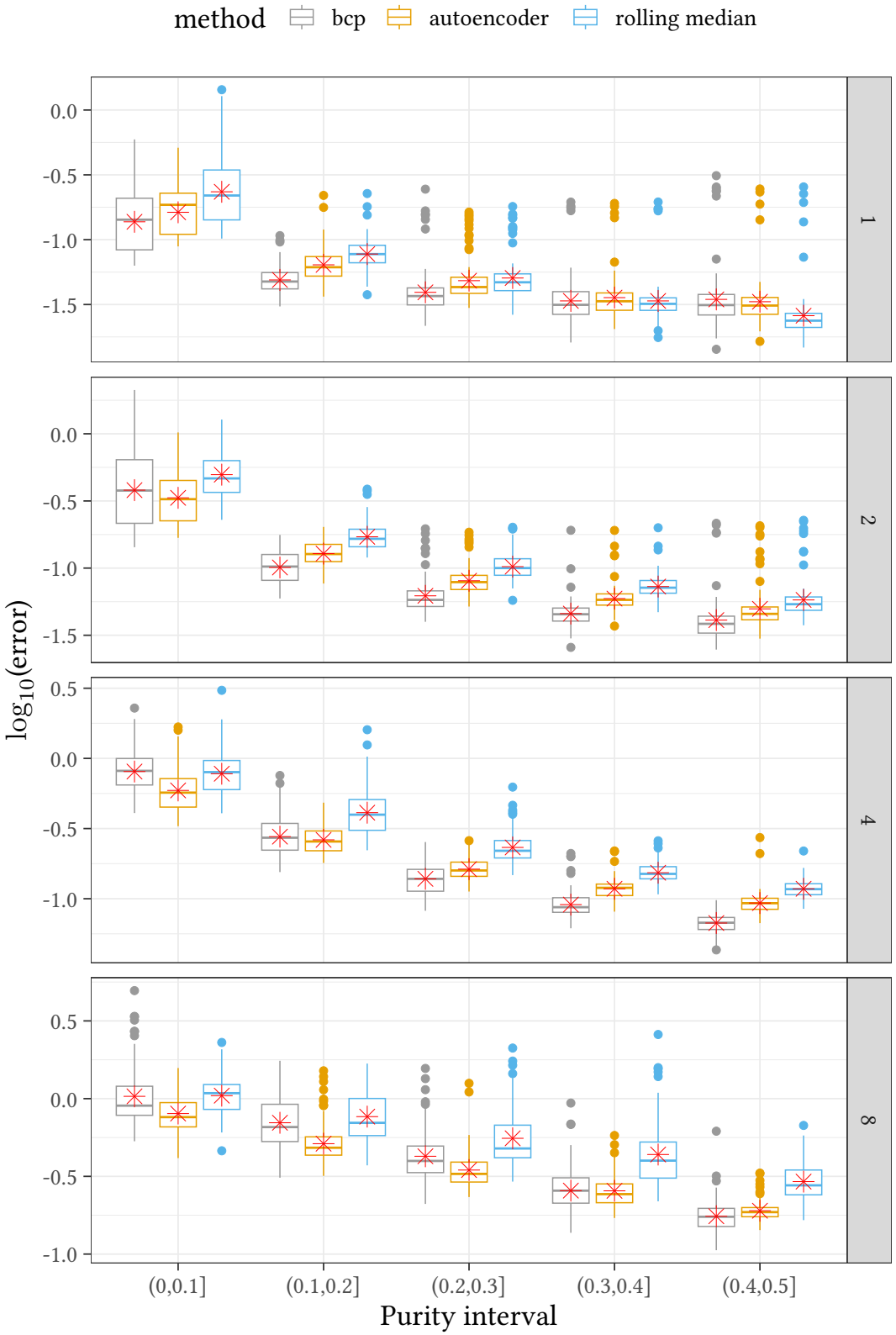


Figure 4.1: Error of purity estimated copy number profiles, for each method at different purity intervals and noise levels, indicated by the gray panels. The error refers to the absolute error between the ground truth and the model prediction.

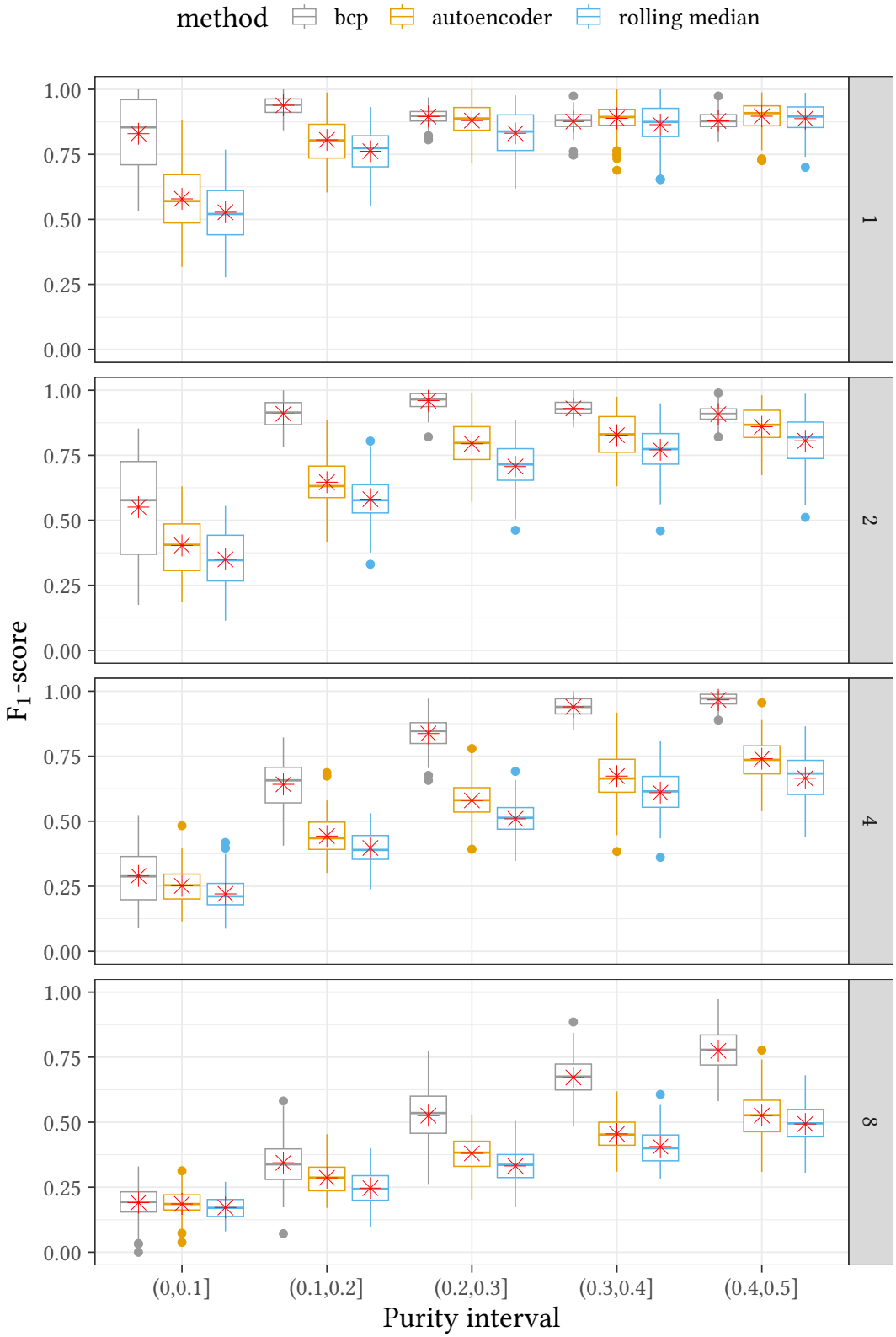


Figure 4.2: F_1-scores for the methods at different purity intervals and noise levels, indicated by the gray panels. The red stars indicate the mean error over the 200 samples

caused by randomness in the segmentation methods, such as the selection procedure of the number of segments, which in our case has been more thoroughly optimized for the autoencoder and rolling median than for BCP.

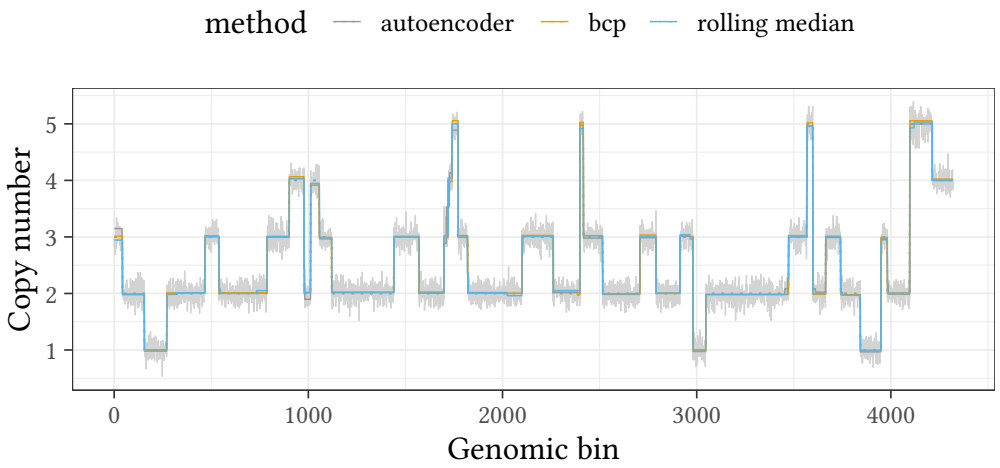


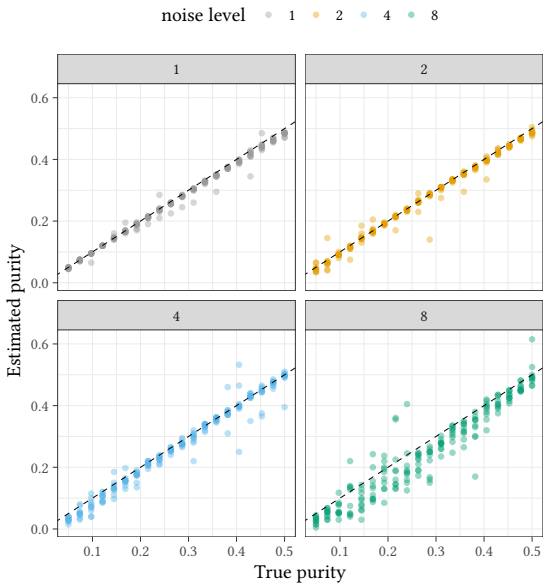
Figure 4.3: Simulated copy number profile and prediction, after purity correction. The original profile was of 50% purity at noise level 1. Note how the copy number levels are now at integer values, corresponding to the ground truth.

4.2 Purity estimation

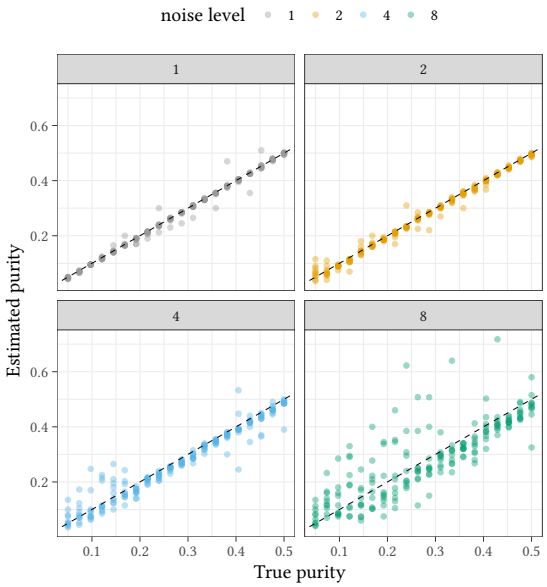
In this section, we study how well we can estimate the sample purity, using both synthetic and in silico datasets. The synthetic data consists of 20 samples per purity in $\{0.05, 0.075, \ldots, 0.5\}$ at four different noise levels $\{1, 2, 4, 8\}$. The synthetic dataset is used to study how sensitive the estimation is to the noise levels, while the in silico data is used to validate that the methods generalize and apply to real-world data with unknown noise distribution. The in silico data is the subsampled dataset described in Section 3.1.

4.2.1 Synthetic dataset

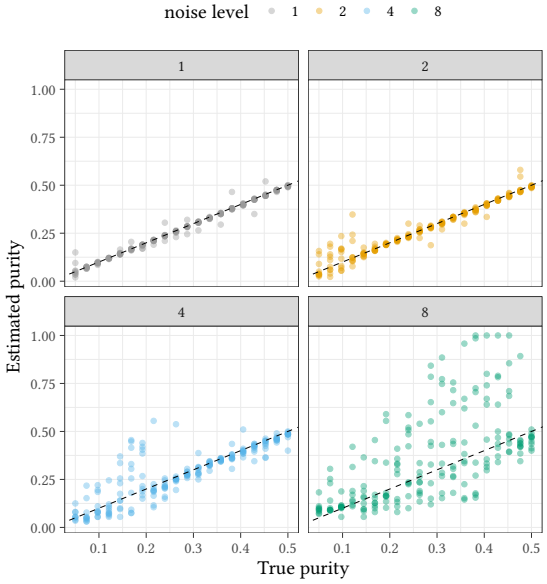
We naturally find that increasing the noise will increase the error and variance of the purity estimation. Furthermore, the purity estimation is more reliable at larger purities. The results are aligned with Figure 4.4 where BCP in general performs the best, and the denoising autoencoder comes second. All methods perform well on noise level 1, even at low purities. At noise level 2, we find that the denoising autoencoder starts breaking down at around 5%, and the rolling median at around 12.5% purity. At noise level 4 BCP tends to underestimate the purity below 20%. Also, the denoising autoencoder starts breaking down below 20% with increased variance in the estimates. Furthermore, the rolling median stops producing reliable estimates around 25% purity. At noise level 8, we are no longer able to produce low variance and reliable purity estimates. We note that we have a smaller variance and fewer outliers when using BCP at this noise level, but instead, we observe a trend of underestimating the purity.



(a) BCP



(b) Denoising autoencoder



(c) Rolling median

Figure 4.4: Purity estimation for four different noise levels $\{1, 2, 4, 8\}$, using the different methods. The results are based on 20 samples per purity in $\{0.05, 0.075, \ldots, 0.5\}$. The dashed gray line is the $y = x$ line.

4.2.2 In silico dataset

We further evaluate how well our method can predict the purity of in silico samples, using the subsampled data described in Section 3.1. The results are presented in Figure 4.5. Note the underestimation of the purity, caused by the mixing procedure. Reducing the number of reads is expected to increase the level of noise and hence make it harder to accurately estimate the purity. We find that all methods, especially the denoising autoencoder and BCP, produced low variance estimates. We find only two samples whose purity has been overestimated, both at 5 million reads at purity below 10%. We find a smaller error in the estimates from the denoising autoencoder on these observations. For the rolling median, however, we find several incorrect estimates. Four of them have 5 million reads and a purity below 10% and are expected to be difficult. Interestingly, we have an outlier at a purity of 12.5% and 10 million reads. The other two outliers have a purity above 10% but only 5 million reads.

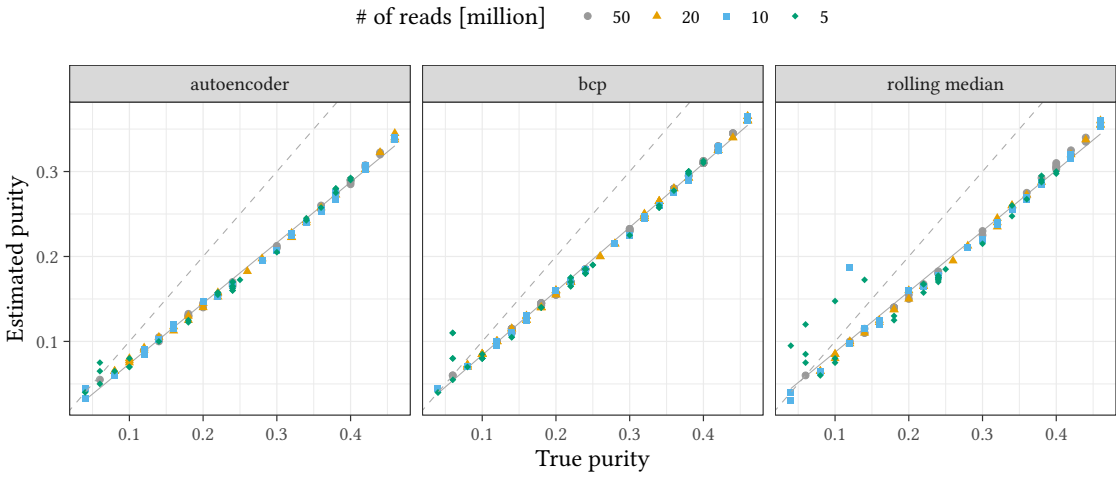


Figure 4.5: Purity estimation of in silico data. The data is subsampled to contain a varying number of reads. A lower number of reads represents a higher noise level. The dashed gray line is the $y = x$ line.

4.3 Subclonal ratio estimation

In this section, we evaluate how well we can estimate the subclonal ratio. We limit the analysis to BCP as the denoising method, as previous results have shown that this method is suitable for this task since it provides the most reliable estimates of the copy numbers and the best segmentation results. Additionally, BCP allows for multi-sample breakpoint detection, where common breakpoints are detected. For the synthetic data, this is done on all samples in a set, while for the in silico data, the multi-sample BCP was done on pairs of samples, since not enough breakpoints were detected if three or more samples were included simultaneously. However, we noticed an improvement from single-sample to paired BCP, thus pairs were deemed appropriate.

4.3.1 Synthetic dataset

To estimate the subclonal ratio, we simulate $M = 200$ datasets with $N = 5$ samples each, corresponding to samples taken at different time points. Except for the first sample in each set, the purities and ratios are sampled uniformly from $\{0.12, 0.14, \ldots, 0.46\}$ and $\{0.05, 0.1, \ldots, 0.8\}$, respectively. The subclonal ratio of the first sample, i.e., the baseline sample, is set to zero, as the estimated subclonal ratio of the other samples would otherwise be underestimated. Furthermore, the purity of the first sample is at least 15%. For this study, we discard the highest noise level, i.e. noise level 8, as the results in Section 3.4 were worse than for the other noise levels, and we do not expect the subclonal tracking to be more accurate than the purity estimation. Furthermore, this noise level is very high and in real patient samples is expected to contain noise similar to noise level 1 or 2.

The results are presented in Figure 4.6, for noise levels $\{1, 2, 4\}$. The results for noise levels 1 and 2 are good, but the algorithm seems to break down for noise level 4, where we find a trend of overestimating the subclonal ratio. This is expected, as at this level of noise the noise outweighs any subtle subclone-associated fluctuations in copy number. As for the purity estimation, the ratios are estimated automatically, and some of the outliers would have been estimated more correctly had it been possible to choose the parameters based on the diagnostics plots.

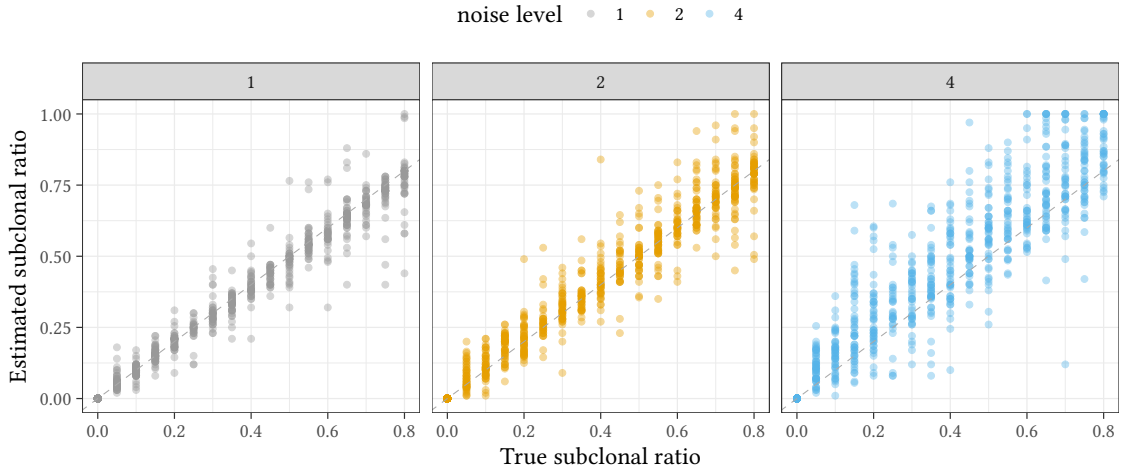


Figure 4.6: Estimated subclonal-ratio compared to the true subclonal-ratio in synthetic samples with varying noise levels. The dashed gray line is the $y = x$ line.

4.3.2 In silico data

The subsampled data described in Section 3.1 was also used to validate how well we can estimate the subclonal ratio in the non-baseline samples. The validation procedure is the same as the liquidCNA. The estimation is performed with $\{50, 20, 10, 5\}$ as the minimal read number of the sample. The estimation is based on $M = 200$ datasets of $N = 5$ uniformly sampled longitudinal samples, and a baseline sample. A sample is discarded if its estimated purity is below 10%, and an entire dataset is discarded if four

or fewer samples, including the baseline, remain after discarding low-purity samples. This is done to introduce some randomness in the number of samples included in the estimation. Note that this brings some differences to the synthetic data, where only five samples, including baseline, are used, but no samples are discarded due to low purity.

The results are similar to those for the synthetic datasets but with more robustness against noise and fewer outliers. The former may be explained by the fact that, for the highest noise level in the synthetic data, all samples have the same noise, while for the highest noise level in the in silico data, lower-noise samples are also included. Another notable difference from the synthetic data is that the baseline sample is always the same, 18.5% purity sample, using our estimate, while the synthetic data baseline samples vary between 15% and 46%.

The impact of adding the data with 5 million reads seems almost negligible, with almost no difference between the two lower plots in Figure 4.7. This is promising, as a low-reads sample can be included in the analysis given that there are enough high-reads samples to compensate. The reduced number of outliers is explained by all samples being of the same cell lines and having alterations in the same segments, meaning the automated process can be fine-tuned more easily. The synthetic data had more variance over the sets and was subsequently more prone to outliers.

In Figure 4.8 we present the comparison of our subclonal estimation algorithm and the original implementation of liquidCNA, using samples of at least 50 million reads, using the same segments and segment values as input. To make the comparison fairer, we discard results where any of the methods returned `NA`, e.g. due to the smoothing of distributions failing. No optimal cutoff interval is used in this comparison. We find that the original implementation works better at the lowest subclonal ratio, but our method produces slightly more consistent estimations at larger subclonal ratios, especially for $r_i = 0.7$. The differences are expected to be minor since the data is processed the same, so the differences boil down to the different segment classification methods, and the density smoothing instead of the Gaussian mixture fitting step.

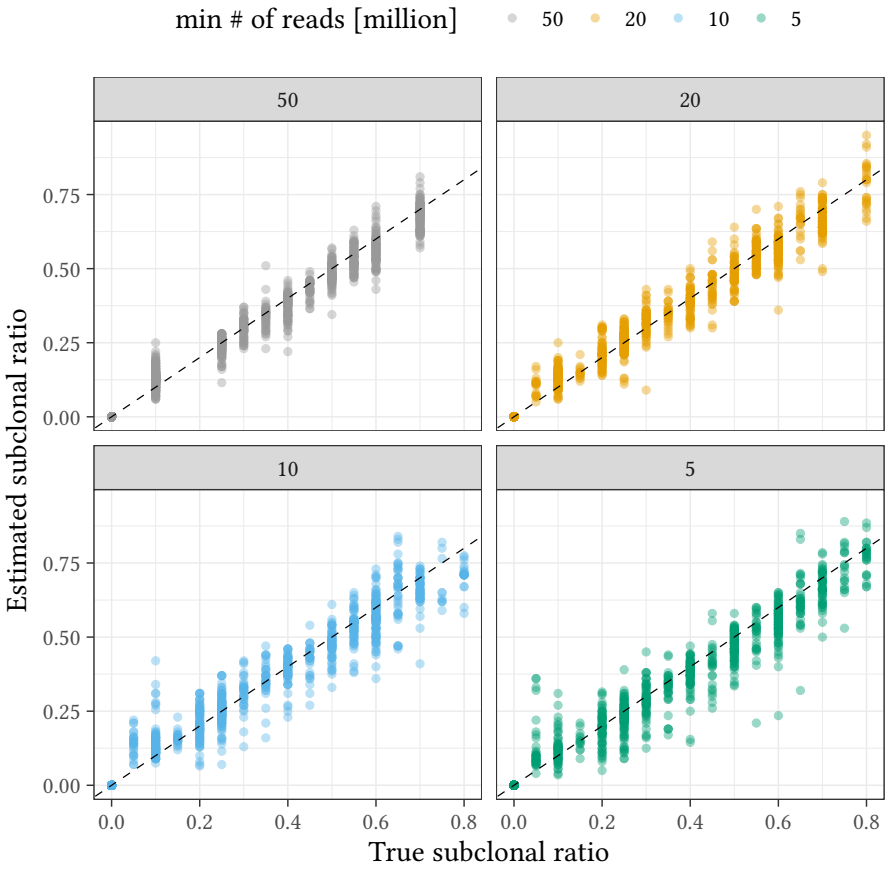


Figure 4.7: Estimated subclonal-ratio compared to the true subclonal-ratio in silico samples with varying read numbers. The dashed gray line is the $y = x$ line.

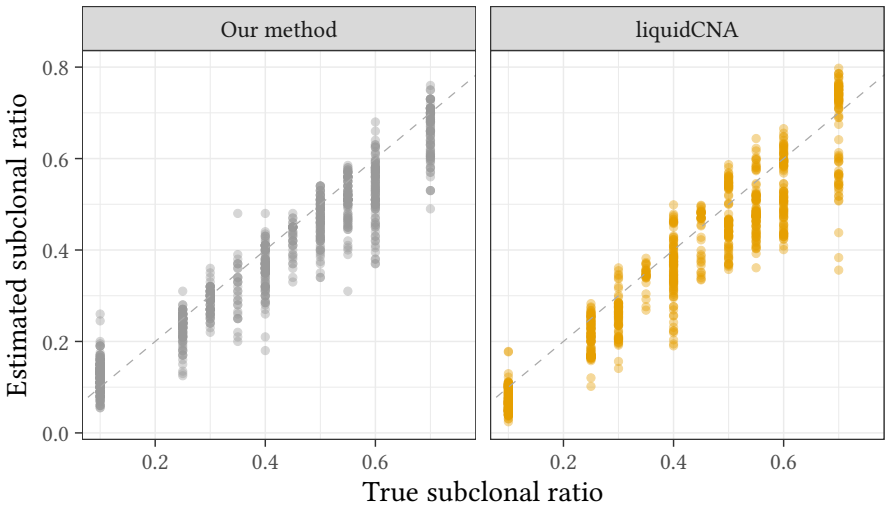


Figure 4.8: Comparison of subclonal estimation using our method and the original liquidCNA method, at minimum 50 million reads, using BCP for the segmentation. The dashed gray line is the $y = x$ line.

5 | Discussion

In this section, we extend the discussion regarding the results in Chapter 4. We discuss the general performance of the denoising models and their ability to generalize to unseen observations, mainly in terms of different cancer types or noise distributions. Furthermore, we address some potential issues in the developed evaluation framework and the denoising models. We then discuss the purity and subclonal ratio estimations, their limitations and their performance. Finally, we discuss how our methods can be used in further studies.

5.1 Deconvolution models

In general, we found that BCP worked best for denoising the samples. One aspect of our methods that has not been brought up in the previous sections is the computational cost, which is an important consideration when working with genomes of higher resolutions, such as a bin size of 50kb. The rolling median denoising is almost instantaneous and the time spent on denoising is negligible compared to the purity and subclonal ratio estimations. On the other hand, BCP has a time complexity of $\mathcal{O}(N^2)$, making processing a large number of high-resolution samples infeasible. While the denoising autoencoder is reasonably fast for predictions, there is an initial cost of training the network.

Another disadvantage of the denoising autoencoder, which is not present for BCP or rolling median denoising, is that it only works for fixed-size inputs. To denoise data of 50kb bins we would therefore have to construct a new model and retrain it, potentially with the added effort of re-finetuning the hyperparameters, which is heavily computationally demanding. It is important to note that while using smaller bin sizes is possible, we recommend using larger bin sizes, such as our choice of 500kb. As we are more interested in the big picture, rather than finding each small genomic region, such high resolution is often unnecessary. Furthermore, binning into larger bin sizes also works as an initial denoising step.

Furthermore, a critical aspect of the model is its ability to generalize to new observations, particularly considering diverse distributions of varying lengths across different types of cancer. In this regard, the BCP model is advantageous as it does not assume anything about the segment lengths. On the other hand, the effectiveness of both the rolling median and the denoising autoencoder is dependent on the size of the sliding window or kernel size respectively, which must be tailored to the expected distribution of the segment lengths. While smaller windows can potentially handle longer segments, they may yield noisier reconstructions. Conversely, larger windows risk overlooking smaller

genomic variations. Therefore, when working with cancer datasets beyond ovarian cancer that this particular model is based on, it's advisable to assess the suitability of the denoiser for the specific context.

We further note that the estimates obtained from automatic segmentation are less accurate than those where the segments are determined by studying the diagnostic plot. In practice, we are often only studying a handful of clinical samples at a time, making the manual examination of diagnostic plots feasible. This allows for the detection and correction of suboptimal parameter choices, such as a too-high cutoff for the posterior probability using BCP, or underestimating the number of breakpoints using the cumulative segmented regression. However, when testing hundreds of samples, we must rely on the automatic cutoffs and parameters, likely resulting in suboptimal method performance. The same discussion applies to the estimation of the sample purity and subclonal ratio, where we rely on multiple parameters and automatic selections. In practice, it is quite easy to determine whether the smoothed density obtained seems reasonable or not. Sadly, we cannot detect and remove such faulty smoothings in the automated process. Therefore, in a real setting, the pipeline is expected to provide more accurate deconvolution and subclonal tracking.

5.2 Estimation of purity and subclonal ratio

During this work, we have aimed to improve the accuracy of the purity and subclonal ratio estimation. We want to highlight that this has been done in two steps: firstly, we developed deconvolution methods to denoise the data to improve sample quality and secondly, we slightly improved the existing methods in [10]. As we can see in Figure 4.8, where the input to both methods is the same segment values, our implementation of the subclonal ratio estimation is slightly better. We want to emphasize that the biggest improvement in this work lies in the sample denoising. Using the developed methods, we have been able to accurately estimate the purity and subclonal ratio in both synthetic datasets and more importantly experimental data. The purity was estimated well up to noise level 4, at sufficiently high purities. The subclonal estimation on the other hand suffers more from the added noise in our simulations and it is advisable to use more reads if the purpose of the collection of the sample is to perform subclonal tracking.

Even though the results of the subclonal tracking are promising, it still has the potential to improve. The emphasis of future work should be on developing an alternative selection rule for selecting the subclonal segments since this is a step that is hard to automate. Another suggestion that might increase method performance is tuning the model parameters to select more appropriate default values. In general, the current purity estimation algorithm provides satisfactory results, but we suggest researching methods for estimating the number of modes of the distribution and optimal smoothing parameters. This is applicable also in the subclonal estimation where it is important to accurately determine the number of modes.

5.3 Future research

We have thus far shown that BCP works very well for denoising liquid biopsy measurements of copy number profiles and, to improve the performance further, we suggest additional extensions to the method. A notable limitation regarding the evaluation processes is that we have focused this work on data with Gaussian noise, even though this may not be true in practice. This is important to note, as one of the assumptions of BCP is that the noise is Gaussian. The method may therefore not work as well with data with more complex noise distributions. However, we found that the method still worked well on the insilico datasets, indicating that the model is generalizable to more complex noise distribution. Hence, an interesting extension to the BCP model would be to include other continuous noise distributions, such as Laplace-distributed measurement noise. Furthermore, it would be interesting to study not only the posterior means over the segment but also the variance, as this can give us further knowledge regarding the noise distribution of the liquid biopsy sequencing. For example, it could provide insight into whether the noise is multiplicative or additive. Another interesting extension is to compare the noise distribution of liquid biopsies to regular tissue-based biopsies and see how well our methods generalize.

Due to data availability, our work is focused on ovarian cancer, however, cancer is not a monotone disease and there are different ways that cancer will be expressed in the genome. A study on copy number alterations in different cancer types [9] shows that ovarian cancer has one of the highest copy number variation rates of all 32 cancer types in the study. Additionally, the copy number variations in ovarian cancer are on average longer than in other cancers. Altogether this means that the methods we have developed for ovarian cancer might not be directly transferable to other cancers, thus a possible future expansion would be to research other cancer types and see which changes, if any, are needed to adapt, if possible, to the different characteristics of those cancers.

Lastly, it would have been informative to perform the method evaluation and comparison on experimental data, to see how well the segmentation generalizes. Unfortunately, the underlying copy number profiles of the available clinical samples are unknown, making it impossible to do the same model evaluation as for the synthetic data. Hence, we relied a lot on simulated data, which is not optimal since we are making assumptions about e.g. variation lengths and noise distributions that might differ from real data. An alternative would be to use copy number simulators to generate data. As these simulators are computationally demanding, we deemed it infeasible to produce a dataset of the size we required within the time constraints of the thesis. However, in future work, such methods could be employed to create improved training and evaluation data for the models.

6 | Conclusion

Copy number variations serve as relevant biomarkers for cancer detection, as these genomic variations are often exclusive to the tumor cells. Furthermore, it is possible to capture such variations using low-pass whole genome sequencing of liquid biopsies, which are promising for tracking cancer evolution since they are cheap and non-invasive, enabling frequent sampling. Due to the noisy nature and the general low cancer proportion in such samples, we evaluate denoising and deconvolution methods, to remove healthy contamination and obtain the underlying copy number state of the genome. We show that it is possible to do this accurately, even in noisy scenarios.

In this work, the emphasis has been on implementing a denoising autoencoder for noise removal, as well as employing a Bayesian model for change point detection. Both models performed better than a baseline denoising model, rolling median denoising, in low purity and high noise scenarios. Furthermore, we conclude that the Bayesian change point detection is a more suitable model for copy number quantification in liquid biopsy sequencing, as the method provides better segmentation results, reliable purity estimates, and more resistance to noise. This method is also more generalizable to other types of cancer, as no assumptions are made on the segment length distributions.

Considering the thesis aims, we believe that the work is successful, since both denoising methods performed better than the baseline, in reconstructing the tumor copy number profile and detecting change points. Furthermore, we can more accurately estimate the sample purity compared to the baseline, at lower purities and higher noise. Using BCP as the most promising denoising method, we were able to estimate the subclonal ratio well in experimental data. This is promising for future application on existing clinical data, and on additional clinical data that has previously been of too poor quality for providing reliable estimates.

Bibliography

[1] Haley M. Amemiya, Anshul Kundaje, and Alan P. Boyle. "The ENCODE Blacklist: Identification of Problematic Regions of the Genome". en. In: *Scientific Reports* **9(1)**.1 (June 2019), p. 9354.

[2] Daniel Barry and J. A. Hartigan. "A Bayesian Analysis for Change Point Problems". In: *Journal of the American Statistical Association* **88(421)**.421 (Mar. 1993), p. 309.

[3] Daniel Barry and J. A. Hartigan. "Product Partition Models for Change Point Problems". In: *The Annals of Statistics* **20(1)**.1 (1992), pp. 260–279.

[4] *Copy Number Variation (CNV) Analysis: A Guide for Clinical Researchers*. en.

[5] *Definition of liquid biopsy - NCI Dictionary of Cancer Terms - NCI*. en. nciAppModulePage. Feb. 2011.

[6] Bradley Efron et al. "Least angle regression". In: *The Annals of Statistics* **32(2)**.2 (Apr. 2004).

[7] Chandra Erdman and John W. Emerson. "**bcp** : An *R* Package for Performing a Bayesian Analysis of Change Point Problems". en. In: *Journal of Statistical Software* **23(3)**.3 (2007).

[8] Lotta Eriksson and Linnea Hallin. *Deconvolution methods for quantification of copy number variations in liquid biopsy sequencing*. Version 1.0.0. May 2024. DOI: 10.5281/zenodo.11366699.

[9] Luuk Harbers et al. "Somatic Copy Number Alterations in Human Cancers: An Analysis of Publicly Available Data From The Cancer Genome Atlas". In: *Frontiers in Oncology* **11** (July 2021), p. 700568.

[10] Eszter Lakatos et al. "LiquidCNA: Tracking subclonal evolution from longitudinal liquid biopsies using somatic copy number alterations". en. In: *iScience* **24(8)**.8 (Aug. 2021), p. 102889.

[11] Chen Lin et al. "Liquid Biopsy, ctDNA Diagnosis through NGS". In: *Life* **11(9)**.9 (Aug. 2021), p. 890.

[12] Biao Liu et al. "Computational methods for detecting copy number variations in cancer genome using next generation sequencing: principles and challenges". en. In: *Oncotarget* **4(11)**.11 (Nov. 2013), pp. 1868–1881.

[13] Osval Antonio Montesinos López, Abelardo Montesinos López, and Dr Jose Crossa. Fundamentals of Artificial Neural Networks and Deep Learning. en. In: *Multivariate Statistical Machine Learning Methods for Genomic Prediction [Internet]*. Springer, Jan. 2022.

[14] Umberto Michelucci. *An Introduction to Autoencoders*. Jan. 2022.

[15] Vito M. R. Muggeo. "Estimating regression models with unknown break-points". en. In: *Statistics in Medicine* **22(19)**.19 (Oct. 2003), pp. 3055–3071.

[16] Vito M. R. Muggeo. "Segmented: An R package to fit regression models with broken-line relationships". In: *R News* **8(1)**.1 (May 2008), pp. 20–25.

[17] Vito M. R. Muggeo and Giada Adelfio. "Efficient change point detection for genomic sequences of continuous measurements". en. In: *Bioinformatics* **27(2)**.2 (Jan. 2011), pp. 161–166.

[18] Aaron M Newman et al. "An ultrasensitive method for quantitating circulating tumor DNA with broad patient coverage". en. In: *Nature Medicine* **20(5)**.5 (May 2014), pp. 548–554.

[19] Nishant Ravikumar et al. Chapter 16 - Deep learning fundamentals. In: *Medical Image Analysis*. Ed. by Alejandro F. Frangi, Jerry L. Prince, and Milan Sonka. The MICCAI Society book Series. Academic Press, Jan. 2024, pp. 415–450.

[20] Ilari Scheinin et al. "DNA copy number analysis of fresh and formalin-fixed specimens by shallow whole-genome sequencing with identification and exclusion of problematic regions in the genome assembly". In: *Genome Research* **24(12)**.12 (Dec. 2014), pp. 2022–2032.

[21] Ilari Scheinin [Aut], Daoud Sie [Aut Cre], and Henrik Bengtsson [Aut]. *QDNAseq*. en-US. 2017.

[22] Tobias Setz. "Stable Portfolio Design Using Bayesian Change Point Models and Geometric Shape Factors". en. PhD thesis. [object Object], 2017.

[23] Jean-Charles Soria et al. "A phase IB dose-escalation study of the safety and pharmacokinetics of pictilisib in combination with either paclitaxel and carboplatin (with or without bevacizumab) or pemetrexed and cisplatin (with or without bevacizumab) in patients with advanced non–small cell lung cancer". en. In: *European Journal of Cancer* **86** (Nov. 2017), pp. 186–196.

[24] Ziyu Tao et al. "The repertoire of copy number alteration signatures in human cancer". en. In: *Briefings in Bioinformatics* **24(2)**.2 (Mar. 2023), bbad053.

[25] TensorFlow Developers. *TensorFlow*. Mar. 2024.

[26] Rikiya Yamashita et al. "Convolutional neural networks: an overview and application in radiology". en. In: *Insights into Imaging* **9(4)**.4 (Aug. 2018), pp. 611–629.

[27] Yimin Yang et al. *Deconvolution-and-convolution Networks*. Mar. 2021.

DEPARTMENT OF MATHEMATICAL SCIENCES
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY