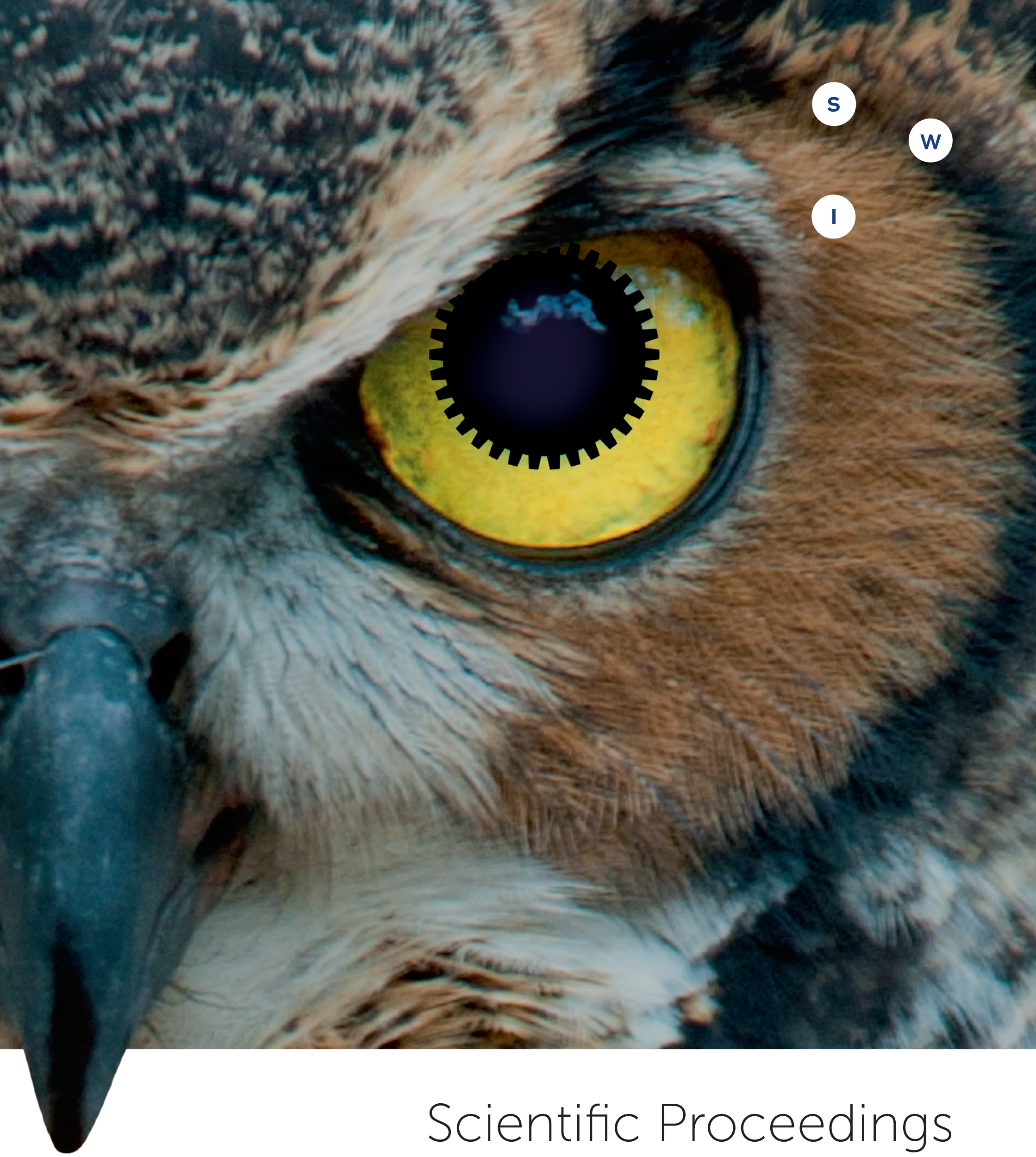


Scientific Proceedings of the Study Group Mathematics with Industry

SWI 2022

Scientific Proceedings
166th European Study Group with Industry

SWI 2022

Enschede, January 24 – January 28, 2022

Editor: Bernard Geurts
Cover design: Kitty Molenaar

Contents

Preface

These are the scientific proceedings of the 166th Study Group Mathematics with Industry (Studiegroup Wiskunde met de Industrie or SWI 2022), held virtually and organized by the University of Twente, January 24–28, 2022. They constitute a collection of five reports from the teams of mathematicians that participated at SWI 2022 and worked on the problems provided by industry partners.

These reports are written in English and aimed at a scientific audience. Besides descriptions of the problems they contain detailed descriptions of investigated solution methods and obtained results. Accompanying the present volume are the popular proceedings, written by science journalist Gieljan de Vries. These provide an account of the work meant for a general audience, written in Dutch.

The organization would like to express its sincere thanks to NWO and 4TU.AMI for sponsoring and PWN (Platform Wiskundig Nederland) for their support in organizing this edition of the study group.

The SWI 2022 organizing committee

Fleurianne Bertrand
Bernard Geurts (Chair)
Jasper Goseling
Ruben Hoeksma
Katharina Proksch
Matthias Schlottbom
Felix Schwenninger
Marielle Slotboom-Plekenpol
Matthias Walter
Jelmer Wolterink

Personalised Health Monitoring for Early Disease Detection

Katharina Proksch[1], Alessandro Di Bucchianico[2], Sandra Keizer[3], Maike de Jongh[4], Marta Regis[5], Roberto Rocchetta[6], Chenyan Huang[7], Larisa Gomaz[8], Aljosa Marjanovic[9], Daphne Nesenberend[10], Marc Paul Noordman[11], Kayode Oshinubi[12], Zohra Rezgui[13] and Oluwatosin Babasola[14]

[1]University of Twente, The Netherlands
[2]Eindhoven University of Technology, The Netherlands
[3]Eindhoven University of Technology, The Netherlands
[4]University of Twente, The Netherlands
[5]Eindhoven University of Technology, The Netherlands
[6]Eindhoven University of Technology, The Netherlands
[7]Eindhoven University of Technology, The Netherlands
[8]University Grenoble Alpes, France
[9]University of Twente, The Netherlands
[10]Leiden University, The Netherlands
[11]University of Groningen
[12]University Grenoble Alpes, France
[13]University of Twente, The Netherlands
[14]University of Bath, United Kingdom

Abstract

Late diagnosis of cancer and cardiovascular diseases often leads to poor chances of cure at high costs. An approach which has the potential to improve the status quo by helping to detect diseases early on, and thereby increase the chances of cure and reduce the costs for treatment, are longitudinal biomarker measurements of microRNA. In this report, we investigate the concept of a personalized baseline based on analysis of variance as well as hierarchical clustering for healthy/sick groups and individual patients in real data. Furthermore, we discuss mathematical models for the detection of illnesses from longitudinal miRNA data. For validation and verification of the proposed methods we develop a data augmentation strategy to generate a large volume of longitudinal miRNA data that can be used and continuously updated.

KEYWORDS: miRNA, sequential detection, biomarkers, clustering

1.1 Introduction

Even with advancing medical treatment, cancer and cardiovascular diseases remain a leading cause of death throughout the world. In Europe alone, cardiovascular diseases claim more than 60 million potential years of life each year Townsend, Kazakiewicz, Lucy Wright, et al. (2022). Early detection of such diseases is very important to improve chances of survival and to decrease medical costs. The company *You2Yourself*[15] (Y2Y) is working on a method to enable early detection of such life-threatening diseases. For this method, urine and blood samples are periodically taken from a large group of initially healthy people over two years in a big study. Based on historical incidence, approximately 7% of the participants of this study are expected to develop a form of cancer, a cardiovascular disease, or a disease of the central nervous system during the two-year duration of the study. The samples are screened for a specific type of biomarker called micro-RNA (miRNA). MiRNAs are small RNAs that play a key role in post-transcriptional gene regulation Lu and Rothenberg (2018).

[15] `https://you2yourself.com/`

Even though different cell types produce the same type of miRNAs, expression profiles vary by tissue type Ludwig et al. (2016). Changes in organs (tumor/inflammation/damage) lead to changes in miRNA profiles, which can be detected in the samples. For example, it has been observed that miRNA patterns change upon tumor formation, suggesting that they might be useful biomarkers for detecting cancer Galvão-Lima, Morais, Valentim, et al. (2021). Taking multiple samples of the same person over time makes it possible to establish a screening procedure based on a personal baseline for the miRNA profile of the blood and urine of the participants. By tracking deviations from that baseline, one could discover the formation of a disease before the onset of clear symptoms. Using a personal baseline instead of the current population based diagnostics is expected to allow for a more sensitive detection, since the biomarker profiles are unique per individual. Figure 1.1 shows a graphical representation of these steps of the study.

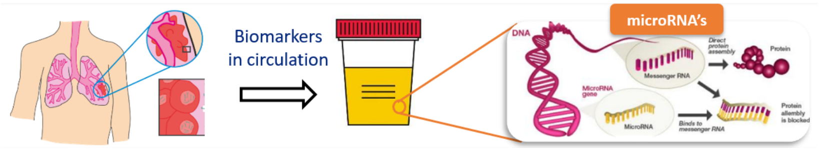


Figure 1.1: **Graphical representation on how deviating miRNA patterns (the biomarkers) end up in a sample.** This patient has a tumor forming in their lung. The miRNA concentrations are different in the tumor microenvironment compared to other parts of the lungs. Some of these biomarkers will end up in circulation, changing the miRNA concentrations in the blood and urine samples as the tumor develops.

The use of miRNAs from urine as biomarkers has the benefit of being patient-friendly and non-invasive when compared to other periodic screening methods or examinations. There are other biomarker options for blood and urine samples, however using miRNA has certain benefits. Figure 1.2 illustrates how deviating biomarker patterns detect formation of illnesses. We stress that this figure is a simplified and idealized representation. In practice, due to the complexity of the biomarker data from the samples, there are several challenges attached to the (statistical) analysis, which we will describe below.

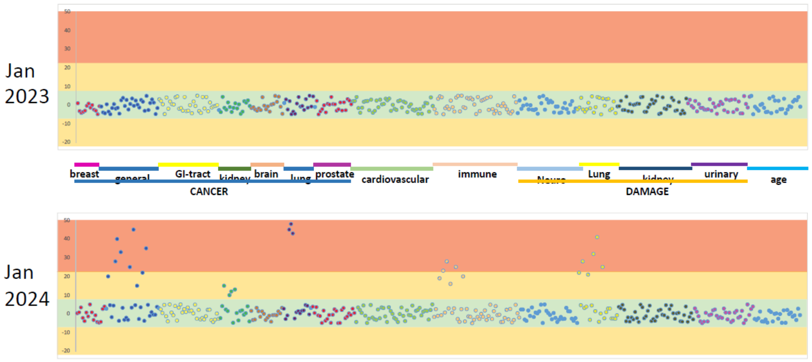


Figure 1.2: **Simplified illustration of how biomarker data could develop over time.** The two figures show the biomarker data of an imaginary patient with one year in between. In January 2023, the biomarker profile is acceptable in reference to the personal baseline. In January 2024 however, a few different miRNAs concentrations are deviating from the personal baseline, indicating that something could be wrong.

1.1.1 Problem statement and organization of the report

One main goal of the research of Y2Y is the extraction of disease detection signatures from complex longitudinal measurements of miRNA profiles. This is difficult because it involves the analysis of longitudinal data in very high dimensions for which no tailor-made methodology exists to the best of our knowledge (see the discussion in Section 1.2.3). A first analysis of the overall project goal led us to define the following three sub-problems, which will be addressed in this report.

1. For a monitoring approach to work, the (stochastic) behavior of the miRNA profile of an average healthy person (a baseline) needs to be explored and put to work as a reference profile. In

Section 1.4.1 we take a closer look at the notion of a baseline and specifically address the question whether a personalized baseline is more suitable than a group or population baseline.

2. Statistical models to detect illnesses and disease progression are needed once a clear notion of a baseline exists. In Section 1.4.3, we discuss the potential of Markov models and mixed effects models in this context.

3. Validation and verification of the proposed analysis tools is important. Since to date only very few longitudinal measurements are currently available, we present a data augmentation strategy in Section 1.4.2, which allows to generate larger samples of synthetic data. The approach is based on the existing data and can be further optimized as soon as more data will become available. Such synthetic data allow to systematically study the performance of any method in the current context in a controlled, yet realistic setting.

This report is organized as follows. We start with a literature review in Section 1.2, followed by a description of the available data and related challenges in Section 1.3. Section 1.4 introduces the approach proposed to tackle some of the challenges. Section 1.5 presents the numerical results and Section 1.6 closes the report with preliminary conclusions and recommendation for future research.

1.2 Literature review

1.2.1 Related work (Medical potential of miRNA for diagnosis)

Since the earliest evidence of miRNA involvement in human cancer was presented in Calin (2002), the topic has been investigated in various studies. A recent review and meta analysis regarding the applicability in the medical field can be found in Condrat et al. (2020), where it is anticipated that monitoring miRNAs will become a routine approach in the development of personalized patient profiles, thus permitting more specific therapeutic interventions as compared to existing, traditional approaches.

One of the two datasets that are analyzed in this report is a cross-sectional data set of 14 controls and 16 patients with stage III and stage IV lung cancer (see Section 1.3 for more details). Related to this, in a meta analysis combining the results of 10 studies, J.-H. Li et al. (2017) investigate the role of miRNAs for the diagnostic and prognostic of lung cancer and the results indicate an excellent overall diagnostic accuracy. Barger and Nana-Sinkam (2015) study miRNAs implicated in lung cancer in general and discuss their usefulness in clinical applications, e.g., as tools for diagnosis, prognosis, and emerging targeted therapeutics.

1.2.2 Related work (Classification and prediction approaches)

Rincon et al. (2019) propose an ensemble feature selection strategy for miRNA signatures for robust cancer classification and detection tasks. They show that a 100-miRNA signature is sufficiently stable to provide nearly the same classification accuracy as the complete Cancer Genome Atlas data set (TCGA, Weinstein, Collisson, and al (2013)). Lopez-Rincon et al. (2020) study a dimensionality reduction and ensemble classification approach for tumor classification from circulating miRNA. Or and Veksler-Lublinsky (2021) recently examined the evolution of miRNA interaction rules and investigate whether these rules are transferable between species using classification methods. Sapre et al. (2016) investigate whether the microRNA (miRNA) profiling of urine could be used to detect urothelial carcinoma of the bladder. Support Vector Machine classifiers with a Student's t-test feature selection procedure is adopted for the detection and the results compared to well-established method (cystoscopy). The authors conclude that miRNA profiling of urine shows promise for the detection of tumour recurrence.

1.2.3 Related work (Longitudinal data and mixed models)

The literature on the analysis of high-dimensional longitudinal data is rather scarce as pointed out recently by Zhong, J. Li, and Kokoszka (2021), who consider analysis of variance and change-point detection in such a setting. While the question of detecting changes over time in high-dimensional data is clearly related to our situation, the view-point is an asymptotic one with respect to time (and sample size). The data in our study were measured at only three time points, using a large sample approximation therefore seems unreasonable. The view-point in the latter reference is related to the view-point in time series analysis, where change point detection in high-dimensions has gained attention in recent years (see, e.g., Jirak (2015) or Cho and Fryzlewicz (2015)). However, in the time-series context many measurements over time are considered and often asymptotic results with respect to time are employed. This perspective is in contrast not only to the available measurements to date but also to future monitoring approaches, where the number of time points will be negligible compared to the data dimension or number of subjects.

Mixed models have been successfully utilized in the analysis of longitudinal biometric data and early disease detection. For example, the predictiveness of ovarian cancer (as a bivalent response variable in dependence of a single biomarker, i.e., a one-dimensional measurement per time point) of two linear mixed models and a pattern mixture model based on the linear mixed model have been compared Han et al. (2020). These models could be extended to deal with the data containing multiple biomarkers and outcomes. However, these methods depend heavily on normality assumptions, which are questionable in our context. S. Li, Cai, and H. Li (2021) consider statistical inference for high-dimensional linear mixed-effects models via a quasi-likelihood approach. The approach does not rely on strict normality assumptions, only sub-Gaussian random components are assumed. The method is based on the Lasso under sparsity conditions on the fixed effects. While this seems suitable at the first glance, the setting and viewpoint considered is that of genome-wide association studies, where effects of genetic variants on a measured phenotype is investigated, i.e., additional measurements on the subjects are regressed on the high dimensional measurements.

Furthermore, it is intuitive that the predictiveness of models can be improved by separation of relevant features and their outcomes, one example is given in Blackwell et al. (2020). It is therefore important to distinguish relevant biomarkers and reduce the model dimensions early on, which is a difficult task in high dimensional data analysis, where standard methodology such as principle component analysis (PCA) fails to be valid (see, e.g., Birnbaum et al. (2013), where estimation of the leading eigenvectors of the covariance matrix is studied under additional structural assumptions on the covariance matrix).

1.2.4 Conclusion on the literature review

There is a clear gap in the literature concerning readily usable statistical methodology to analyze longitudinal miRNA data. Derivation of a fully functioning method is clearly beyond the scope of this workshop. However, we discuss the potential of Markov models and mixed effects models in Section 1.4.3 and fit a mixed effects model to a down-scaled data set (see Section 1.5.5 for the results). The discussion of the existing literature shows that statistical learning has been successfully applied in the analysis of miRNA data. To this end, we apply hierarchical clustering methods to investigate the properties of the data in more detail. Since a serious limitation to date is the availability of large scale longitudinal data sets. Therefore, we develop a data augmentation startegy.

1.3 Description of the data

Two datasets were provided for this study and comprise measurements of miRNA concentrations in urine samples for two independent experiments. The tow data set comprise:

(1) Longitudinal miRNA samples from healthy patients.

(2) Cross-sectional miRNA samples from both healthy and unhealthy patients.

The two datasets will be presented in the next sections.

1.3.1 Longitudinal data

The first data set contains *longitudinal data* of 1941 miRNA concentrations for 7 healthy subjects over 3 distinct time points. In the following, these measurements will be denoted by

$$Y^L_{i,t} \in \mathbb{R}^{1941}, \quad i \in \{1, \dots, 7\}, \quad t \in \{1, 2, 3\}, \tag{1.1}$$

where the index i denotes the individual and t denotes the time at which a measurement was made. Since the exact point in time is irrelevant for this study, t is set to l for the l-th measurement in time for each individual. To provide a first idea of our data, Table 1.1 provides a small excerpt of the concentrations of four exemplary miRNAs in two subjects. The unit is ppm, i.e., parts per million.

$Y^L_{1,1}$	$Y^L_{1,2}$	$Y^L_{1,3}$	$Y^L_{2,1}$	$Y^L_{2,2}$	$Y^L_{2,3}$
131.94	21.47	68.35	0.00	53.03	6.26
27.16	8.05	17.09	0.00	13.26	0.00
38.80	17.45	187.97	190.76	92.80	12.53
7.76	0.00	3.42	0.00	0.00	0.00

Table 1.1: Small excerpt of the longitudinal data of four exemplary miRNAs.

The data shown in the table already clearly suggest that we are dealing with a difficult problem. The variation between the concentrations is quite high and miRNAs with low concentrations might not be measured at all for some individuals, resulting in many zeros. For this data set, a healthy state is assumed for all the patients because regular checks did not diagnose major illness, nonetheless, a disease could be (at least in principle) be progressing without being undetected. Note that more time points are to be added during the course of the study.

1.3.2 Cross-sectional data

The second data set comprises *cross-sectional data* of miRNA concentrations from 30 subjects, 16 of whom had been diagnosed with stage

III or IV lung cancer prior to the study, and 14 are healthy. A cut-off removing all zero-measurements (i.e. zero concentrations over all subjects) is introduced in the second dataset, leaving 1400 miRNAs, corresponding to observations

$$(Y_i^{CS}, k_i) \in \mathbb{R}^{1400} \times \{0,1\}, \quad i \in \{1, \ldots, 30\}. \tag{1.2}$$

In the above model, the index i denotes the individual, whereas the variable k_i denotes the state of the i-th individual (0 for healthy, 1 for sick). Note that the cross-sectional data set only has one measurement per subject, making it unsuitable to investigate subject-specific miRNA concentrations over time. Nevertheless, this data set has the advantage of containing both healthy and sick labels and can thus be used to provide a first idea about the kind of changes in the profiles to expect as the result of the onset of a severe disease and which miRNAs are relevant for disease detection in this case.

1.3.3 Difficulties

The use and analysis of miRNA biomarker data comes with manifold challenges. For instance, one main goal of the research of Y2Y is the extraction of disease detection signatures from complex longitudinal data. However, statistical methods to analyse such longitudinal studies are not standard, as our discussion of the related literature in Section 1.2 shows. Moreover, disease signatures are likely to overlap, making the detection and prognostic tasks even harder to tackle. From the description of the data, is apparent that the data dimension is extremely high compared to the sample sizes and therefore, methodology from *classical statistics* may not be applicable and the viewpoint of *high-dimensional statistics* should be assumed (see, e.g., Wainwright (2019) for a comprehensive monograph on high-dimensional statistics). Furthermore, both data sets only contain a small number of samples (7 and 30 subjects, respectively), which makes the results of data exploration analysis only preliminary, requiring further validation when more samples are available.

1.4 The proposed approach

It is a long way to go until a fully developed health monitoring procedure, for which all fundamental and practical issues will be resolved, can be put to use. This work seeks the first step in this direction by investigating a generalized framework for health monitoring from miRNA biomarker samples. Our main contributions to the existing literature can be summarized as follows:

- A preliminary analysis regarding the concept and feasibility of a personal vs. a population or group baseline in Section 1.4.1
- We present a data augmentation strategy to generate artificial data (based on the already existing samples) and test models and methods on larger data sets in Section 1.4.2.
- We examine and discuss modeling options for disease prognostic and health monitoring from longitudinal bio-markers data in Section 1.4.3.

We used various techniques (such as hierarchical clustering, classification, variance decomposition, statistical tests, and more) to study personal and population-based prognostic models applicable to the longitudinal and cross-sectional miRNA data sets. We present the outcomes of the analyses in Section 1.5. Our findings and recommendations for future research are summarized in Section 1.6.

1.4.1 The concept of a baseline

In mathematical terms, health monitoring can be seen as (sequentially) testing for deviations of measured miRNA profiles from a suitable reference profile. In this report, such a reference profile will be referred to as a *baseline* and it corresponds to the “typical miRNA pattern of an average healthy person”.

In this section, we will discuss the concept of a baseline from a statistical perspective to shed more light into the question what a reasonable notion of a baseline could look like. Furthermore, we will provide an exploratory analysis of the data in respect to the question whether or not to use a personal baseline (in contrast to, e.g., a group baseline).

Mathematical concept of personal and group baseline

The concept of a baseline is essential in order to establish a relative rather than absolute meaning of the miRNA data. Measurements of a healthy person will contain measurements of the miRNA concentrations which are typical for this individual in a healthy state. Multiple measurements of the same person over time will show sampling variability due to the measurement process and also due to the constitution of the patient. A baseline needs to take into account both an average profile (i.e., some measure of centrality of each miRNA) and a measure of expected variability, as both are needed to judge whether an observed deviation from the baseline is significant or not.

Intuitively, it seems to be obvious that a personal baseline is preferable over a group baseline. However, an important disadvantage of the use of a personal baseline is that several measurements of the miRNA profile of an individual in a healthy state are needed in order to properly capture the individual profile including the natural, personal variations in the measured values. In contrast, a group baseline could profit from many prior measurements, so that already a person's first measured miRNA profile could be used for the detection of a disease. The question is which of these two approaches is more feasible in practice. Also, a hybrid approach, combining both personal information and pooled information across individuals, could be a reasonable approach. Based on the longitudinal data set described in Section 1.3 we will look into these question in more detail.

Classification

Longitudinal data allow for the assessment of within individual variation of the miRNA samples over time. Intuitively, one may think that two profiles of the same person should be closer to each other than two profiles of different individuals. To investigate whether this is the case, an attempt has been made to recognize different groups of subjects using hierarchical (linkage) clustering. The hierarchical clustering algorithm starts with a point-cloud, $\{Y^L_{i,t}\}_{i,t}$, say, where every single measured vector of miRNAs starts as a cluster for itself. In each step of the algorithm the distances between the clusters are being calculated and two clusters with the smallest distance are then merged together. This is done until only one cluster remains. The implementation of the algorithm depends, of course, on the notion of distance between the points (i.e. a metric or some dissimilarity function d between the points in $\mathbb{R}^{1941}$) and a notion of a distance between the clusters, also called the linkage function D. Some common linkage functions include arithmetic, geometric and harmonic averages of distances between singular points in the two clusters and minimum and maximum of all the distances. The last two linkage functions give rise to so-called single and complete linkage methods, respectively. The results of such clustering algorithms are dendrograms representing the merging of the clusters, from which the relevant distances can be read. Besides the Euclidean (l_2) distance or the Manhattan (l_1) distance, there are a variety of other metrics and dissimilarity functions available which may capture the separation of the clusters along selected features better. A notion of distance suitable for this task should not put much emphasis on absolute sizes of the components but rather consider the difference of components relative to their sizes, that is, suitably re-scaled versions of common norms might be more appropriate in our context. One such example is the *Canberra distance.* For vectors $u, v \in \mathbb{R}^p$, the Canberra distance is given as

$$d_{\text{Cb}}(u,v) = \sum_{i=1}^{p} \frac{|u_i - v_i|}{|u_i| + |v_i|}.$$

This distance equalizes the contributions of the smaller and larger components and is upper bounded by the dimension p of the space, i.e., $\|d_{\text{Cb}}(\cdot,\cdot)\|_\infty \leq p$. The Canberra distance between two vectors is large if a sparse vector is compared to a non-sparse vector, regardless of the total size of the components of the non-zero vector. In contrast to the Euclidean distance it does not result in extremely large values if the components of one vector are much larger than the components of the other vector. Table 1.2 and Table 1.3 show distance matrices of the data vectors shown in Table 1.1, based on the Euclidean distance and the Canberra distance, respectively. From this it is already obvious that the notions of nearness are very different between these two distances. The clustering results are presented in Section 1.5.1.

	$Y^L_{1,1}$	$Y^L_{1,2}$	$Y^L_{1,3}$	$Y^L_{2,1}$	$Y^L_{2,2}$	$Y^L_{2,3}$
$Y^L_{1,1}$	0	114.39	162.53	177.11	203.22	174.82
$Y^L_{1,2}$	114.39	0	177.11	174.82	81.86	17.90
$Y^L_{1,3}$	162.53	177.11	0	177.11	96.53	186.92
$Y^L_{2,1}$	177.11	174.82	70.59	0	112.18	178.34
$Y^L_{2,2}$	203.22	81.86	96.53	112.18	0	93.84
$Y^L_{2,3}$	174.82	17.90	186.92	178.34	93.84	0

Table 1.2: Distance matrix corresponding to the data presented in Table 1.1, based on the Euclidean distance.

	$Y^L_{1,1}$	$Y^L_{1,2}$	$Y^L_{1,3}$	$Y^L_{2,1}$	$Y^L_{2,2}$	$Y^L_{2,3}$
$Y^L_{1,1}$	0	2.64	1.59	3.66	2.18	3.42
$Y^L_{1,2}$	2.64	0	2.71	3.78	1.80	2.28
$Y^L_{1,3}$	1.59	2.71	0	3.01	1.59	3.71
$Y^L_{2,1}$	3.66	3.78	3.01	0	3.13	3.75
$Y^L_{2,2}$	2.18	1.80	1.59	3.13	0	3.40
$Y^L_{2,3}$	3.42	2.28	3.71	3.75	3.40	0

Table 1.3: Distance matrix corresponding to the data presented in Table 1.1, based on the Canberra distance.

Analysis of variance

From a statistical viewpoint, a personal baseline is preferable over a group baseline if the within person variation of the profiles is smaller than the between person variation. Figure 1.3 shows the sample mean $\pm$ one sample standard deviation for two exemplary miRNAs for each of the seven individuals of the longitudinal data set. While the first miRNA seems to have a high variation between the different individuals, the second seems to be dominated by the between groups variation. A statistical methodology that explores exactly this, is analysis of variance (ANOVA), which deals with the problem of testing whether the means of several populations agree. More precisely, a statistical test

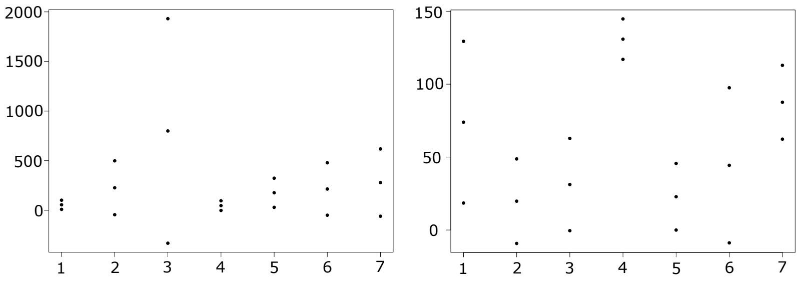


Figure 1.3: sample mean $\pm$ one sample standard deviation for two exemplary miRNAs for each of the seven individuals

problem with the following hypotheses is considered:

$$H_0 : \mu_1 = \ldots = \mu_k \quad \text{vs.} \quad H_1 : \mu_i \neq \mu_j \quad \text{for at least one pair } i \neq j.$$

In the most basic, one-dimensional setting, observations $Y_{i,j}$, where $i = 1, \ldots, k$ and $j = 1, \ldots, n_i$ are considered and it is assumed that the $Y_{i,j}$ are independent and follow a normal distribution with mean μ_i and variance σ^2. The F statistic is the ratio of the MST and the MSE:

$$F = \frac{MST}{MSE} = \frac{\frac{1}{k-1}\sum_{i=1}^{k} n_i(Y_{i\cdot} - Y_{\cdot\cdot})^2}{\frac{1}{n-k}\sum_{i=1}^{k}\sum_{j=1}^{n_i}(Y_{i,j} - Y_{i\cdot})^2}.$$

Here, $Y_{i\cdot}$ and $Y_{\cdot\cdot}$ denote the group means and the overall mean respectively. The numerator and the denominator can be interpreted as components of the total variance, the residual sum of squares RSS:

$$\begin{aligned}\mathrm{RSS} &= \underbrace{\frac{1}{n}\sum_{i=1}^{k}\sum_{j=1}^{n_i}(Y_{i,j} - Y_{\cdot\cdot})^2}_{\text{total variance}} \\ &= \underbrace{\frac{1}{n}\sum_{i=1}^{k}\sum_{j=1}^{n_i}(Y_{i,j} - Y_{i\cdot})^2}_{\text{within groups variance}} + \underbrace{\frac{1}{n}\sum_{i=1}^{k} n_i \sum_{j=1}^{n_i}(Y_{i,\cdot} - Y_{\cdot\cdot})^2}_{\text{between groups variance}}\,.\end{aligned}$$

This means that the F-test compares the within groups variance to the variance between groups. Under the null hypothesis and the given a assumptions, the F statistic follows an F distribution with $k-1$ and $\sum_{i=1}^{k}(n_i - 1)$ degrees of freedom. We computed the values of the F statistic for all miRNAs. The results of this analysis can be found in Section 1.5.2. Clearly, the the validity of the assumptions is questionable in this context, but all F values in relation to each other can nonetheless be seen an indicator for stability.

1.4.2 Synthetic data generation mechanism

This section presents a data augmentation strategy to simulate a large-scale longitudinal study with several volunteers. Data simulators can support the validation and verification of algorithms and speed up the development of data analysis pipelines. Furthermore, the analysis of synthetic data can support decision-making, future data collection and experiments. Once new empirical evidence is collected, it can tune and improve the simulator's accuracy and adherence to reality.

The proposed simulator generates miRNA concentrations from the empirical marginal distributions conditional to the health state of the subjects (healthy/sick). Mathematically, this corresponds to sampling from $\hat{F}_j(\mathbf{x}|y=0)$ for healthy patients and $\hat{F}_j(\mathbf{x}|y=1)$ for sick patients. For notation convenience, we referred to miRNA profiles as $\mathbf{x}$ and to the health labels as y, where $y=0$ indicates a healthy patient.

For a given label y, a miRNA concentration x_j is sampled from the empirical marginal distribution $\hat{F}_{X_j}(x) = \frac{1}{n}\sum_{i=1}^{n}\mathbf{1}_{\{x_{i,j}\leq x\}}$ and realizations in-between samples are obtained by linear interpolation. The procedure work as described next. Consider a vector of miRNA densities $\mathbf{x} = (x_1, x_2, \ldots, x_d)$, where d is the number of miRNA concentrations. A probability value for each entry can be obtained as follows:

$$(U_1, U_2, \ldots, U_d) = (F_1(x_1), F_2(x_2), \ldots, F_d(x_d)) \tag{1.3}$$

where the probability values are uniformly distributed in the unit hypercube $[0, 1]^d$. Our simulator uses a copula model, which defines the dependency between the components of the vector $(U_1, U_2, \ldots, U_d)$:

$$C_\Sigma(u_1, u_2, \ldots, u_d) = \mathbb{P}[U_1 \leq u_1, U_2 \leq u_2, , ..., U_d \leq u_d]. \tag{1.4}$$

A Gaussian copula is used in this work,

$$C_\Sigma = F_G(\Phi^{-1}(u_1), \Phi^{-1}(u_2), ..., \Phi^{-1}(u_d);\ \Sigma), \tag{1.5}$$

where F_G is the joint Gaussian distribution parameterized by the correlation matrix Σ and $\Phi^{-1}(\cdot)$ is the inverse cumulative distribution function of a standard normal random variable. We use an empirical $\hat{\Sigma}$ estimated from data. Once a copula structure is defined, pseudo-random samples are obtained sampling correlated uniform vectors

$$(u_1, ..., u_d) \sim C_{\hat{\Sigma}},$$

and then mapping these realization to the space of miRNA densities. This last step is done by inverting of the empirical distributions evaluated at $(u_1, ..., u_d)$:

$$(x_1, \ldots, x_d) = \left(\hat{F}_1^{-1}(u_1), \ldots, \hat{F}_d^{-1}(u_d)\right). \tag{1.6}$$

An example of the procedure is presented in Figure 1.4, where the copula structure is assumed to be independent of the health state and therefore shared among the two groups. Any distribution family can replace the empirical marginals $\hat{F}_j$ and any copula can replace the heuristic copula for healthy and sick patients. Selecting an appropriate distribution family may be a patient-specific and disease-specific issue that is not further considered in this work. Furthermore, a Gaussian copula family requires a large $d \times d$ correlation matrix as an input and this can complicate numerical tractability given the high dimensionality of the sample space. The selection of a subset of highly correlated miRNAs may be advisable for future developments.

Disease progression model and generation of longitudinal data

Our simulation model samples miRNA concentrations of n_p patients at time fixed time steps $t_1, t_2, ..., t_{n_t}$. A health state index $k_i(t_j), i = 1, .., n_p$ is assigned to to the patients at each time t_j and patients are assumed healthy at the beginning of the longitudinal study, i.e., $k_i(t_1) = 0$ for all i. Several patients will likely develop a sickness during the study. Thus, we introduce a probabilistic transition model to simulate this change in population health over time. The following discrete-time Markov chain defines the transition probabilities:

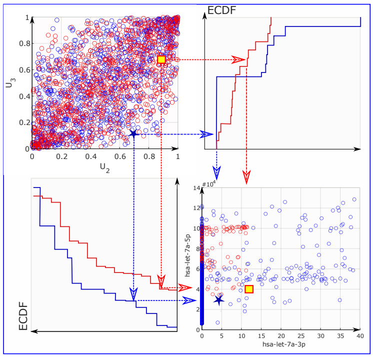
$$\mathbb{P}[k(t_{j+1}) = 1 | k(t_j) = 0] = P_{H2S}$$

where P_{H2S} is the probability that an healthy patient will develop a sickness in the interval $[t_j, t_{j+1}]$. We assume a sick patient to be unable to recover during the course of the simulation, i.e., $\mathbb{P}[k(t_{j+1}) = 1 | (t_j) = 1] = 1$.

Because a sickness fully develops over some time, we propose a disease progression model that combines the empirical distribution of healthy and sick patients. The proposed mixture distribution model is defined as follows:

$$x_j(t) \sim \rho(t)\hat{F}_j(x|k = 0) + (1 - \rho(t)) \cdot \hat{F}_j(x|k = 1)$$

where $\rho(t) \in [0, 1]$ is a real-valued time-dependent sickness factor quantifying the progression of the disease. A value of $\rho(t) = 0$ indicates an healthy patient at time t whilst $\rho(t) = 1$ indicates a fully developed disease. We assume $\rho(t)$ to be a linearly increasing function in the interval t_s and $t_s + t_{trn}$, where y changes from 0 to 1 at t_s and $\rho(t_s + t_{trn}) = 1$ when the sickness is fully developed. Generally speaking, time t_{trn} is a random time which depends on the individual characteristics of the subject and disease. However, for the sake of simplicity, the transition time t_s is a constant input in our model.



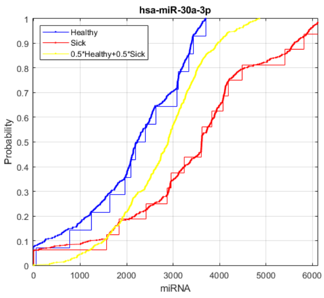


Figure 1.4: Left panel: Correlated samples and inverse empirical CDF transformation for healthy and sick patients. Right Panel: Transition from healthy distribution (blue) to a sick distribution (red).

1.4.3 Detection

This section introduces a mathematical framework for cancer disease predictions that best captures the longitudinal biomarker data. The proposed approach employs mixed effects models and Partially Observable Markov Decision Processes and, due to a lack of time, the latter is only mathematically introduced and not directly applied to the disease prediction problem.

Markov models and POMDPs

In this subsection, we describe a Markovian approach to decision-making problems under uncertainty. Chapter 4 of Poor and Hadjiliadis (2008) gives a detailed description of Markov decision processes applications to sequential detection. Similar ideas have also been explored in the context of maintenance, see e.g., Linderman, McKone-Sweet, and Anderson (2005), Mehrafrooz and Noorossana (2011), and Panagiotidou and Tagaras (2010).

Partially observable Markov decision processes, or POMDPs, provide a formal framework for the interaction of a decision maker (an agent) with a stochastic, partially observable environment. That is, it provides an agent with the capabilities to reason about both action uncertainty, as well as state uncertainty. A POMDP is a discrete time model, in which the agent selects an action at every time step or stage. It extends the regular Markov decision process (MDP) to settings in which the state of the environment cannot be observed. It can be formally defined as a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \Omega, \mathcal{T}, \mathcal{O}, \mathcal{C}, h)$, where $\mathcal{S}$ is a (finite) state space, $\mathcal{A}$ is a (finite) set of action, Ω is the space of observations, $\mathcal{T}(s, a, s') = \mathbb{P}(s'|s, a)$ is a transaction probability function that specifies the probability of a next state s' given a current state s and action a, $\mathcal{O}$ is an observation function, $\mathcal{C}(s, a, s')$ is an immediate cost function for a particular transition s, a, s' and h is the horizon of the problem.

The model $\mathcal{M}$ can help a decision-maker by recommending good actions that maximize the long-run revenue of the monitoring systems or, similarly, that minimize the expected cumulative sum of costs over the horizon h. The rule that dictates which action the agent must take in each state is known as the policy, a map $\pi : \mathcal{S} \to \mathcal{A}$ from the state space to the space of actions. In this work, we wish to maximize the reward generated by a correct prediction of disease from miRNA readings (and minimize costs due to wrong predictions and missed alerts).

States and actions

A state vector $\mathbf{s} \in \mathcal{S}$ consists of three parts, $\mathbf{s} = (\mathbf{x}, y, z)$, where $\mathbf{x}$ is the actual profile of miRNA's, y denotes the actual health status (stage of cancer), and z is a binary variable indicating whether or not cancer has been detected via a traditional diagnostic methods, i.e., an indicator function defined as follows:

$$z = \begin{cases} 0 \text{ if no cancer has been detected.} \\ 1 \text{ if cancer has been detected.} \end{cases} \tag{1.7}$$

We also define action vectors $a \in \mathcal{A}$ as binary indicator of diagnosis based on the miRNA profile:

$$a = \begin{cases} 0 \text{ if diagnosed as healthy based on miRNA profile} \\ 1 \text{ if diagnosed as ill based on miRNA profile.} \end{cases} \tag{1.8}$$

Observations space and transition probability

The observation function $\mathcal{O} : \mathcal{S} \times \mathcal{A} \times \Omega \to [0,1]$ is a function defining the probabilistic accuracy of the miRNA counts and an observation $\omega \in \Omega$ is a vector containing the miRNA counts. The transition function $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0,1]$ is based on the following probabilities:

- Let $p_1 : \mathcal{X} \times \mathcal{Y} \times \mathcal{X} \to [0,1]$ denote the probabilistic transition function between miRNA profiles, where $\mathcal{X}$ is the space of possible profiles.
- Let $p_2 : \mathcal{Y} \times \mathcal{Y} \to [0,1]$ denote the probabilistic transition function between different stages of cancer, where $\mathcal{Y}$ denotes the set of possible stages.
- Let $p_3 : \mathcal{Y} \to [0,1]$ denote the probability that a patient has symptoms severe enough to see a doctor and that cancer is successfully detected given the cancer stage.

Note that $\mathcal{T}$ combines the probability of moving from the present miRNA $\mathbf{x}$ to a new concentration $\mathbf{x}'$ and from a present cancer stage y to a next stage y'. By definition, this is a map from state-to-state $\mathbb{P}(\mathbf{s}'|\mathbf{s})$, and diagnostic actions a have no effect on it.

Cost function

The Cost function $\mathcal{C} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is a fundamental component of POMPDs and must be carefully defined. The cost function assigns to any transition, e.g. from a state-action pair to new state, a cost/reward function. In this detection problem, a cost. The lower the cost, the higher is the value of the action taken in a specific state. In this cancer prediction problem, costs can arise due to delays in detecting (changes in) the health status of a patients from miRNA counts, costs for false diagnosis (false positives), and from missed diagnosis (false negatives). Hence, the selected cost function include three terms:

$$C(\mathbf{s}, a, \mathbf{s}') = \begin{cases} C_1(y) \text{ if } a = 1 \text{ and } z = 1 \text{ (true positive)} \\ C_2 \text{ if } a = 1 \text{ and } z = 0 \text{ (false positive).} \\ C_3(y) \text{ if } a = 0 \text{ and } z = 1 \text{ (missed detection)} \end{cases} \tag{1.9}$$

where $C_1 : \mathcal{Y} \to \mathbb{R}$ and $C_3 : \mathcal{Y} \to \mathbb{R}$ are increasing functions of the stage of cancer y. The cost C_1, is cost associated to correct prediction of illnesses, this quantity should be negative (a reward) and should have a large in absolute value in the early stages (higher rewards for an early detection). The cost C_2 is associated to false positive events whilst cost C_3 arise when cancer is not predicted from the miRNA profile but by other means, e.g., the cancer has detected due to symptoms, but not detected from the miRNA profile. The cost C_3 should be high, especially for later stages.

Remarks and challenges

Numerical analysis of POMDPs generally assumes a finite horizon for the analysis and computation of optimal decision-making policies. In this work, we assume that a person is tested for cancer if either he develops symptoms severe enough to see a doctor or his/hers miRNA profile indicates the potential presence of cancer and we define the end of the horizon as the moment a person is diagnosed with cancer ($z = 1$). Unfortunately, the proposed cost function does not take into account finite horizon and if the disease is not detected within a certain time frame, it may result too late in magnitude. This consideration is similar to other discussions on the performance of control charts in statistical process control, where it has been argued that instead of looking at average time to detection (called ARL = average run length), it is more relevant to consider as performance the probability of successful detection with a certain time frame (called PSD = probability of successful detection). The interested reader is reminded to e.g. Kenett and Pollak (2012) for a detailed discussion on this topic. Another issue concerns the definition of cancer stage y and its relationship with the diagnostic outcome z of established screening tests. To increasing the predictive power of this framework, it would be advisable to study a suitable quantifier for y. Moreover, it would be useful to study a function $y = \psi(\mathbf{x})$ that maps actual miRNA structure/changes to this health state, i.e, a model for the minimal change in miRNA counts that will cause a person transitioning from a healthy to sick state.

Mixed effect models

Monitoring multiple patients over time on a series of miRNAs leads to a high-dimensional dataset, multivariate and longitudinal. It can be large in the number of patients (n), in the number of outcomes (m miRNA) in the number of time points (T), all at the same time. Multivariate longitudinal data come with the challenge of correlations and heterogeneity: the clustering at individual level, different miRNAs can have different variances, measurements can be correlated at each time point for different markers, and counts from the same miRNA can be correlated in time. Even picturing an idea of such correlations in the whole dataset is challenging due to the high dimensionality. On the other hand, the complexity and multidimensionality of the data offers a wide choice of models and possible methods to detect changes. For example, beside looking at the shifts and changes in trend of single microRNAs, it is possible to study how the multiple markers vary together in the healthy status, and use this to detect changes.

Mixed models offer a flexible framework to capture different forms of correlation in the data, and to choose the most suitable covariance structure. The general linear mixed model equation is given by

$$Y_i = X_i\beta + Z_i u_i + e_i \tag{1.10}$$

Y_i is the matrix of observations for the i^{th} individual, X_i a matrix of covariates of interest and β the corresponding matrix of coefficients to be estimated. This term is defined as *fixed* and captures the trend over the whole population, as opposed to u_i that are called *random effects* and model individual-specific characteristics. Normality is often assumed for the random effects ($u_i \sim N(0, \tau^2)$). The design matrix for the random effects Z_i can be a subset of X_i but does not have to be. Finally there is the error term e_i that captures the correlations within individual. It's often assumed to be normal, but extensions exists. The two random effects u_i and e_i are often assumed to be independent, but through their covariances it is often possible to investigate the complex dependency structure of the data. There exist different methods to accommodate the structure of multivariate longitudinal data. One option is to include a Kronecker product covariance $V \otimes \Sigma$ for the repeated measurements - repeated for each subject on the different miRNAs and for multiple points in time, where V models the inter-miRNAs correlations between multiple markers measured at the same time point, and Σ the intra-marker correlation at different time points (again the same for all miRNAs). One reasonable structure could involve unstructured V and autoregressive Σ

$$\begin{pmatrix} \sigma_1^2 & \sigma_{12} & \dots & \sigma_{1m} \\ \sigma_{21} & \sigma_2^2 & \dots & \sigma_{2m} \\ \dots & \dots & \dots & \dots \\ \sigma_{m1} & \sigma_{m2} & \dots & \sigma_m^2 \end{pmatrix} \otimes \begin{pmatrix} 1 & \rho & \rho^2 & \dots & \rho^{T-1} \\ \rho & 1 & \rho & \dots & \rho^{T-2} \\ \dots & \dots & \dots & \dots & \dots \\ \rho^{T-1} & \rho^{T-2} & \dots & 1\dots & \end{pmatrix}.$$

Such models can easily be fitted to a large number of study participants (n) and/or to a long time period (large T). Most likely the implementation of mixed models is not easily scalable to a large number of outcomes (i.e. to the whole set of microRNAs), but they can already be used to study how a number of biomarkers vary together, for example a selection that is of particular interest. We will illustrate this on a set of biomarkers in Section 1.5.5.

Generalized linear mixed models extend the linear mixed model in (1.10) by introducing a link function $g(x)$ that connects the linear predictor $\eta = X_i\beta + Z_iu_i$ to the observed outcome

$$g(E(Y_i)) = \eta, \tag{1.11}$$

so that also outcomes with distributions other than normal can be modelled (for example binary or count outcomes).

For the current purpose, generalized linear mixed model would be suitable - since the data at hand are count data. However, given the time constraints for this initial investigation we have decided to use the linear mixed model on the log-transformed variable. The main reason concerns the available implementation of multivariate models for continuous outcomes with complex covariance structures, but also the fact that the actual data we are using is *derived from* count data. The major limitation of this approach on the other hand is due to the zeros present in the data (about 30% across the five modelled microRNAs), that are lost with the log-transformation. Future research should focus on adapting existing methods to model the original counts with an appropriate generalized linear mixed model. Our results can however be an indication of *how* these methods could be used and *what* they could provide - to enable informed decisions.

1.5 Results

1.5.1 Clustering based on available data

Due to the high dimensionality of the original data, we propose to apply a dimensionality reduction technique using Kolmogorov-Smirnov (KS) tests, which are based on the marginal empirical cdfs of the healthy vs. diseased cohorts. Specifically, we tested the hypotheses

$$H_{0,j} : F_{\mathrm{miRNA}_j}(\cdot \,|\, \text{healthy}) = F_{\mathrm{miRNA}_j}(\cdot \,|\, \text{lung cancer})$$

and we included all miRNAS with p-values below 0.05 in the lower dimensional space. A subset of 27 miRNAs was selected in this manner:

$$Y_{\mathrm{KS},i,t} \in \mathbb{R}^{27}, \quad i = 1, \ldots, 7,\ t = 1, 2, 3. \tag{1.12}$$

Eight of these are also included in the 100 miRNA signature found by Rincon et al. (2019). Figure 1.5 and Figure 1.6 show two exemplary outcomes of linkage clusterings of the longitudinal data.

Figure 1.5 shows an arithmetic linkage clustering approach to the data (1.12), where the Euclidean distance is used to measure the distance between the vectors $Y_{\mathrm{KS},i,t}$ and $Y_{\mathrm{KS},k,s}$. We clearly see that the nearest neighbors of the measurement $Y_{\mathrm{KS},i,t}$ is typically one of the $Y_{\mathrm{KS},k,s}$ with $k \neq i$, i.e., measurements between individuals may very well be closer to each other than measurements of the same individual at different time points. The outcome of the clustering algorithm suggests forming three clusters, whose elements are listed in Table 1.4.

Only for person 3 and person 4, all measurements are in the same cluster. For person 1 the three measurements over time are even assigned to three different clusters. If seven clusters are formed, ideally,

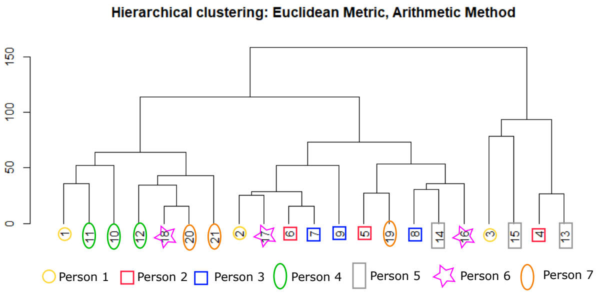


Figure 1.5: Arithmetic linkage clustering approach to the data (1.12), where the Euclidean distance is used to measure the distance between the vectors $Y_{KS,i,j}$ and $Y_{KS,k,l}$.

Cluster 1:	$Y_{\mathrm{KS},1,1}, Y_{\mathrm{KS},4,2}, Y_{\mathrm{KS},4,1}, Y_{\mathrm{KS},4,3}, Y_{\mathrm{KS},6,3}, Y_{\mathrm{KS},7,2}, Y_{\mathrm{KS},7,3}$
Cluster 2:	$Y_{\mathrm{KS},1,2}, Y_{\mathrm{KS},6,2}, Y_{\mathrm{KS},2,3}, Y_{\mathrm{KS},3,1}, Y_{\mathrm{KS},3,3}, Y_{\mathrm{KS},2,2}, Y_{\mathrm{KS},7,1}, Y_{\mathrm{KS},3,2}, Y_{\mathrm{KS},5,2}, Y_{\mathrm{KS},6,1}$
Cluster 3:	$Y_{\mathrm{KS},1,3}, Y_{\mathrm{KS},5,3}, Y_{\mathrm{KS},2,2}, Y_{\mathrm{KS},5,1}$

Table 1.4: Elements of the clusters as formed via average linkage clustering using the Euclidean distance, when three clusters are formed.

one would see that each person forms their own cluster. Instead, we see that the measurements of no person stay in the same cluster (see table 1.5).

Arguably, the Euclidean distance might not be the best distance measure when comparing miRNA profiles. However, while the clustering results using other metrics look different, the general tendency of measurements of the same person over time end up in different clusters, remains. To showcase this, Figure 1.6 shows the outcome of the average linkage clustering based on the Canberra metric. The same can be observed for different types of metrics and different types of linkage functions. Furthermore, we performed the same clustering algorithms for different sub-selections of miRNA, always yielding comparable results. In particular, we used a selection of five miRNA, which had been

Cluster 1:	$Y_{\mathrm{KS},1,1}, Y_{\mathrm{KS},4,2}, Y_{\mathrm{KS},4,1}$
Cluster 2:	$Y_{\mathrm{KS},4,3}, Y_{\mathrm{KS},6,3}, Y_{\mathrm{KS},7,2}, Y_{\mathrm{KS},7,3}$
Cluster 3:	$Y_{\mathrm{KS},1,2}, Y_{\mathrm{KS},6,2}, Y_{\mathrm{KS},2,3}, Y_{\mathrm{KS},3,1}, Y_{\mathrm{KS},3,3}$
Cluster 4:	$Y_{\mathrm{KS},2,2}, Y_{\mathrm{KS},7,1}, Y_{\mathrm{KS},3,2}, Y_{\mathrm{KS},5,2}, Y_{\mathrm{KS},6,1}$
Cluster 5:	$Y_{\mathrm{KS},1,3}$
Cluster 6:	$Y_{\mathrm{KS},5,3}$
Cluster 7:	$Y_{\mathrm{KS},2,2}, Y_{\mathrm{KS},5,1}$

Table 1.5: Elements of the clusters as formed via average linkage clustering using the Euclidean distance, when seven clusters are formed.

previously found in an unpublished data set via differential expression analysis, corresponding to the measurements

$$Y_{\mathrm{DE},i,t} \in \mathbb{R}^5, \quad i = 1, \dots, 7, \ t = 1, 2, 3, \tag{1.13}$$

for the longitudinal data set and

$$Y^{\mathrm{CS}}_{\mathrm{DE},i} \in \mathbb{R}^5, \quad i = 1, \dots, 30, \tag{1.14}$$

for the cross-sectional data.

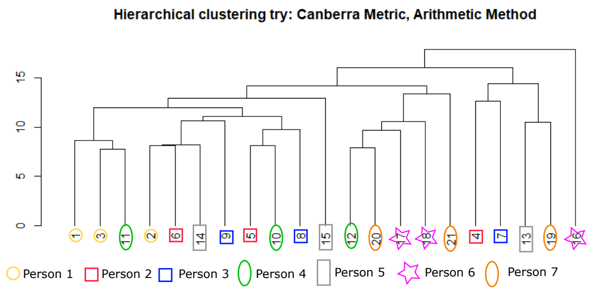


Figure 1.6: Arithmetic linkage clustering approach to the data (1.12), where the Canberra distance is used to measure the distance between the vectors $Y_{\mathrm{KS},i,t}$ and $Y_{\mathrm{KS},k,s}$.

In order to investigate whether such a clustering method can produce useful results in the context of miRNA analysis, we applied it to the cross-sectional data set $\{Y^{\mathrm{CS}}_{\mathrm{DE},i} \mid \quad i = 1, \ldots, 30,\}$ as well. The result (Figure 1.7) clearly shows that the observations of sick patients seem to be comparable and close to each other. In particular, one big cluster of mainly sick individuals and one big cluster of mainly healthy individuals and two smaller clusters are suggested. This shows that, up to fine tuning, such a clustering approach can yield quite reasonable results.

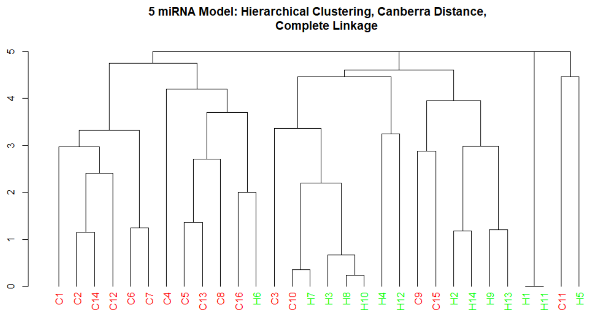


Figure 1.7: Complete linkage clustering approach to the data (1.14), where the Canberra metric is used to measure the distance between the vectors $Y^{\mathrm{CS}}_{\mathrm{DE},i,j}$ and $Y^{\mathrm{CS}}_{\mathrm{DE},k,l}$.

1.5.2 Analysis of variance

Table 1.6 contains values of the F-statistic and corresponding p-values for seven exemplary miRNAs for the longitudinal data of the seven individuals. The two miRNAs from Figure 1.4 are highlighted in blue. While certainly the normal assumption and the independence assumption are highly questionable for our data, these values give a first indication of how difficult the problem is.

A histogram of all p-values is shown in Figure 1.8. Clearly, the region from 0.45 to 0.7 is overpopulated. Most of the miRNAs that have a p-value in this region are zero for most of the measurements. If these are filtered out in a pre-processing step, the histogram would

F-value	1.432	0.870	1.947	3.864	5.661	1.222
p-value	0.271	0.541	0.143	0.017	0.0036	0.352

Table 1.6: Values of the F-statistic and corresponding p-values for seven exemplary miRNAs for the longitudinal data of the seven individuals.

look uniform with a slight elevation in the first bin, indicating that indeed, several miRNAs differ substantially between individuals. In fact, 11, 61 and 108 miRNAs have a p-value of less than $0.01, 0.05$ or 0.1, respectively. Selecting the most significant miRNAs according to

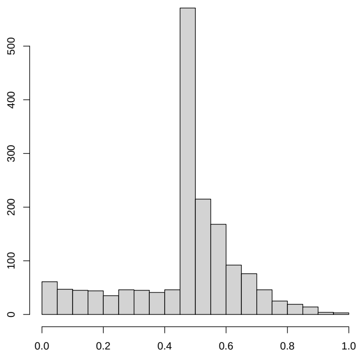


Figure 1.8: Histogram of all p-values from the F-tests.

this criterion, i.e., the most individually different ones, yields a selection of 11 miRNAs. We applied the complete linkage clustering algorithm to this sub-selection of miRNAs as well. The results are shown in Figure 1.9 for the Euclidean distance (right) and the Canberra distance (left).

While the clustering based on the Euclidean distance does not see any strong within person similarities as compared to between person similarities, the Canberra distance clearly does. When 7 clusters are formed, six are person specific. Only one cluster consists of all three

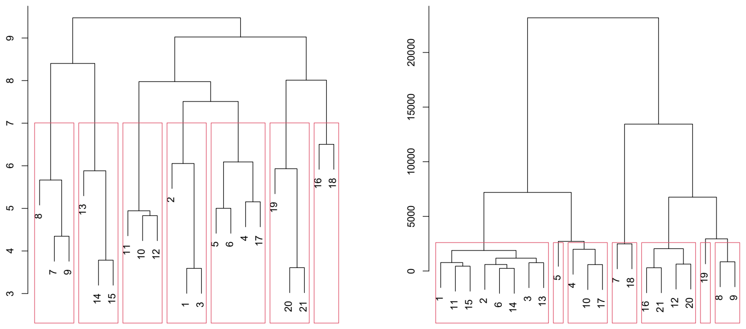


Figure 1.9: complete linkage clustering algorithm to the ANOVA subselection of miRNAs with the Euclidean distance (right) and the Canberra distance (left).

measurements of one subject and one additional measurement of another subject. This indicates once more that when comparing miRNA measurements via their distances, the Canberra distance might be a suitable measure of proximity.

Our first exploratory data analysis clearly suggests that some miRNAs might be better suited for a group baseline, whereas others require a personal baseline. This is certainly an important topic for future research.

1.5.3 Generation of synthetic data

Algorithmic details

The data generating mechanism has been coded within the MATLAB environment and in the *'Simulator_miRNA_LongDataGenMech.m'* function. The DGM takes as input the number of synthetic patients $N_{patients}$, number of time steps N_t (number of longitudinal measurements), and a structure containing options and additional parameters for the transition model. The simulator generates n_t longitudinal samples of $N_{miRNA} = 1421$ miRNA concentrations, health indices (0 health, 1 sick), and disease progression coefficients $\rho(t)$ for each patient. The option input structure contains three fields:

1. *Option.Tran_prob_Healthy2Sick* that represents the healthy to sick transition probability P_{H2S}.
2. *Option.Sick_progression_interval* that defines the number of longitudinal measurements needed for the disease to fully develop.
3. *Option.UseCorrelation* a Boolean index defining weather or the empirical correlation $\hat{\Sigma}$ has to be used when sampling $miRNA$ profiles.

Six are the outputs of the data simulator:

i *miRNA*: cell array $[1 \times N_{patients}]$ with the miRNA samples (grouped by patients), where each elements is a $[N_{miRNA} \times N_t]$ matrix.

ii *Time vector*: Vector of time indices $(1, 2, ..., N_t)$.

iii *Health indicators*: $[N_{patients} \times N_t]$ matrix of Boolean health indicators.

iv *miRNA names*: $[N_{miRNA} \times 1]$ string containing the names of the miRNAs.

v *Sick Percentage*: $[N_{patients} \times N_t]$ matrix. Each element in the matrix defines the sick percentage indicator $\rho(t)$.

vi *Data per miRNAType*: cell array $[1 \times N_{miRNA}]$ with miRNA samples (grouped by miRNA type), where each element is a $[N_{patients} \times N_t]$ matrix.

The data generating mechanism runs very efficiently.
If *Option.UseCorrelation* is set to FALSE, the function took 6.5 seconds to generate data for $N_{patients} = 1000$ over a 2 years longitudinal study ($N_t = 8$). On the other hand, 16 seconds were needed to generate $N_{patients} \times N_t = 1000 \times 8$ applying the correlation structure.

Figure 1.10 presents an example of conditional marginal CDF $F_X(x|y)$ for healthy and sick patients and compare simulated data and real measurements. Note that the simulated marginal CDFs are very similar to the empirical marginal density and, thus, the probabilistic behaviour of

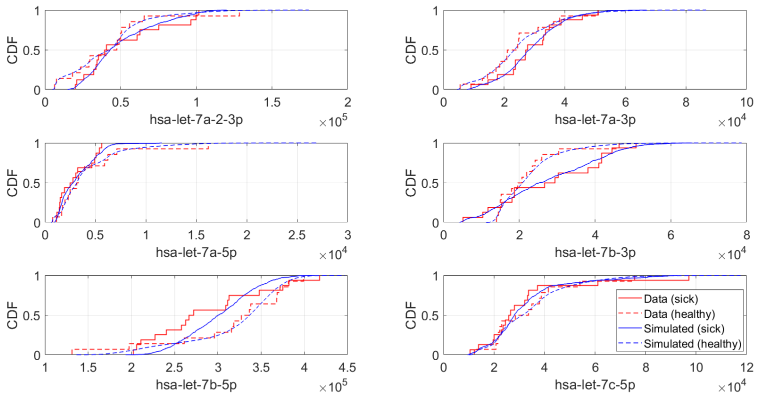


Figure 1.10: A comparison between simulated and experimental F_x for six miRNA types. Solid and dashed red lines display, respectively, the empirical CDFs for sick and healthy patients. The distributions of the simulated data miRNA are presented by blue CDFs.

the real data (at least the behaviour of the marginals) is well-captured by the simulator. Figure 1.11 shows correlated samples for the simulated (blue) vs experimental data (red markers). Qualitatively the simulated samples display overall a reasonable trend, although sub-optimal fitting can be observed for some of the miRNA intances. As example, 'hsa-let-7a-2-3p' shows to be strongly correlated (linearly) with 'hsa-let-7b-3p', see red markers in the top right panel of Figure 1.11. Unfortunately this strong correlation is partially lost in the synthetic data, i.e., the blue markers (synthetic samples) are still positively correlated but with larger dispersion. Despite these limitations, the proposed data simulation tools offer a valuable contribution and can be used to design, test and verify predictive models before expensive data collection is carried out. This can speed up algorithmic developments, inform further data collection, and improve the overall effectiveness of the study.

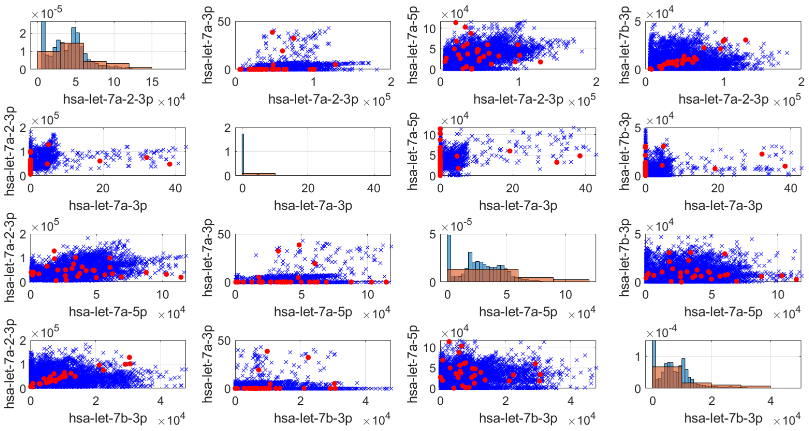


Figure 1.11: An example of correlated synthetic miRNA samples (blue markers) versus the experimental data (red markers). The off-diagonal panels present pairs-wise comparison of four selected miRNA concentrations and the panels on the diagonal compare the marginal distribution of the data (red histograms) and the simulated samples (blue histograms).

1.5.4 Clustering based on synthetic data

The results for the clustering based on the simulated data are comparable to what we obtained for the real data for the longitudinal data sets. However, the separation in healthy versus sick seams is slightly clearer for the real data.

1.5.5 POMDP and mixed effects model

We have fitted a mixed model jointly to a selection of five miRNAs, $Y_{\mathrm{DE},i,t}$, that were indicated as informative by the problem owner, and were also in large part found again in the baseline analysis (cf. Section 1.4.1). Among the fixed effect we included a distinct intercept (microRNA-specific average), and a distinct effect for time (taken as categorical, so that no trend was imposed a priori) for each of the microRNAs. The model has no random effects, and a Kronecker product

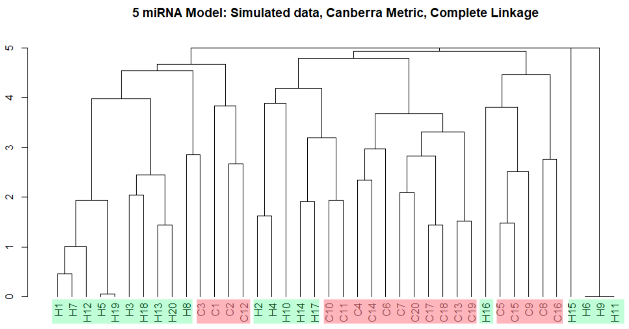


Figure 1.12: Complete linkage clustering of a synthetic data set.

for the covariance matrix as illustrated in Section 1.4.3. The estimates of the fixed effect can be found in Table 1.7. For each of the modelled microRNAs, we report the estimates of the intercept and the estimate at two time points. The third (last) is taken as reference (= 0). Beside the estimate we report the standard error and the corresponding p-value.

More of interest for the current analysis are the estimated variance-covariance parameters between microRNAs (Table 1.8), and the estimated autoregressive coefficient for the correlation in time. These can be found in Table 1.8, together with their standard errors and p-values. None of the covariances between microRNAs is estimated to be significantly different from zero.

1.6 Conclusions and Recommendations

1.6.1 Pros and cons of the proposed approaches

We explored the notion of a baseline using a classification approach and ANOVA. This exploratory data analysis suggests that some miRNAs might be better suited for a group baseline, whereas others might be better suited for a personal baseline. Therefore, a hybrid version might be a good solution and is certainly a direction to think about more in

Intercept				
microRNA		estimate	std. error	pvalue
$Y_{\text{DE}}(1)$		2.85	0.50	<.0001
$Y_{\text{DE}}(2)$		2.95	0.39	<.0001
$Y_{\text{DE}}(3)$		3.04	0.31	<.0001
$Y_{\text{DE}}(4)$		4.01	0.33	<.0001
$Y_{\text{DE}}(5)$		2.63	0.62	0.0178
Effect of time				
microRNA	time	estimate	std. error	pvalue
$Y_{\text{DE}}(1)$	1	1.63	0.70	0.0335
	2	0.44	0.61	0.4877
$Y_{\text{DE}}(2)$	1	0.11	0.60	0.8564
	2	-0.44	0.49	0.3889
$Y_{\text{DE}}(3)$	1	0.35	0.56	0.5469
	2	-0.28	0.38	0.4867
$Y_{\text{DE}}(4)$	1	0.47	0.58	0.4356
	2	-0.20	0.41	0.6239
$Y_{\text{DE}}(5)$	1	0.62	1.10	0.6089
	2	0.22	0.76	0.7870

Table 1.7: Fixed effect estimates

the future. As a general finding it seems that complete linkage clustering based on the Canberra metric seems to suitable to find patterns in our data, whereas other metrics and dissimilarity functions as well as other linkage functions could not provide convincing results.

We developed a numerical simulator to generate large amount of synthetic longitudinal miRNA data. The model was used to efficiently simulate a 2-years long longitudinal study with 10^3 - 10^4 volunteers and only took a few minuets to generate this large amount of labelled longitudinal samples. We captured correlation between miRNA concentrations within the simulated data and the marginal distributions of the synthetic miRNA realizations well-mimic the probabilistic behaviour of the empirical data. Because the simulator provides data for sick patients and for the disease progression (although only artificial), it can be conveniently used for the numerical validation and verifica-

	estimate	std. error	pvalue
VAR($Y_{\text{DE}}(1)$)	1.58	0.56	0.0024
VAR($Y_{\text{DE}}(2)$)	0.74	0.41	0.0374
VAR($Y_{\text{DE}}(3)$)	0.59	0.32	0.0355
VAR($Y_{\text{DE}}(4)$)	0.76	0.30	0.0057
VAR($Y_{\text{DE}}(5)$)	2.15	1.91	0.1310
COV($Y_{\text{DE}}(1), Y_{\text{DE}}(2)$)	0.40	0.40	0.3156
COV($Y_{\text{DE}}(1), Y_{\text{DE}}(3)$)	-0.02	0.25	0.9393
COV($Y_{\text{DE}}(1), Y_{\text{DE}}(4)$)	-0.10	0.29	0.7357
COV($Y_{\text{DE}}(1), Y_{\text{DE}}(5)$)	-0.13	0.52	0.8046
COV($Y_{\text{DE}}(2), Y_{\text{DE}}(3)$)	0.11	0.22	0.6103
COV($Y_{\text{DE}}(2), Y_{\text{DE}}(4)$)	0.33	0.31	0.2827
COV($Y_{\text{DE}}(2), Y_{\text{DE}}(5)$)	-0.46	0.58	0.4305
COV($Y_{\text{DE}}(3), Y_{\text{DE}}(4)$)	0.50	0.27	0.0696
COV($Y_{\text{DE}}(3), Y_{\text{DE}}(5)$)	-0.12	0.45	0.7910
COV($Y_{\text{DE}}(4), Y_{\text{DE}}(5)$)	-0.87	0.78	0.2598
AR(1)	0.24	0.22	0.2741

Table 1.8: Covariance paramter estimates.

tion of the data analysis tools and to speed up the construction of data analysis pipelines. For instance, it could be used to test the accuracy of classification and disease detection methods and models to define a baseline for healthy patients before more data is collected. Another advantage of the proposed method is that it is possible to scale up and tune the simulator with new experimental evidence, e.g., new miRNA samples and discovered relationships between miRNA concentrations and specific diseases and illness progressions. The simulation model has however some limitations, specifically, (i) lack of data and knowledge in literature makes it difficult to define a realistic model; (ii) the model is relatively simple and for the time being only incorporates one illness, neglects time-correlations (auto correlation and correlations between miRNAs) and neglects population heterogeneity; (iii) because synthetic samples are generated from the empirical marginals, this artificially reduces the uncertainty in the distribution of healthy and sick patients'; (iv) transition from healthy assumed linear;(v) censoring and study dropouts neglected; (vi) non-disease related changes not taken into account; (vii) no personalized baseline.

1.6.2 Future research and recommendations

This study shows that it is difficult to prescribe personalized solutions from uncertain low-density mRNA. More samples are needed to get reliable results, e.g. though the URIMON study. In the meantime, our synthetic data can be used to test and validate data analysis and prediction methods. Conceptual approaches for timely detection of disease based on temporal evolution of miRNA counts have been discussed and reviewed (POMDP,mixed effect models). These need to be further developed. Possibly, a combination of POMDP and mixed effect models could be a feasible solution. In order to understand the properties of the data better, quantification of the uncertainty in the measurement process would be very helpful, e.g., repeated measurements on the same urine sample. in order to be able to characterize the variability in the mRNA density per-patient, 1 sample per day (say) for 10 healthy persons for a week or two could be collected and analyzed. Research directions for the future are Statistical Process Control theory and concepts, such as self-starting control charts to automatically obtain personalized baselines of healthy patients. Combinations of SPC and Markov Decision Processes and SPC and mixed effect models exist in other application areas than health. Therefore, we believe that the combination of POMDP, mixed effect models and SPC is a good strategy to profit from the best of the three worlds.

References

Barger, Jennifer F and S Patrick Nana-Sinkam (2015). “MicroRNA as tools and therapeutics in lung cancer”. In: *Respiratory medicine* 109.7, pp. 803–812.

Birnbaum, Aharon et al. (2013). “Minimax bounds for sparse PCA with noisy high-dimensional data”. In: *Ann. Statist.* 41.3, pp. 1055–1084.

Blackwell, Jacob N. et al. (2020). "Early Detection of In-Patient Deterioration: One Prediction Model Does Not Fit All". In: *Critical Care Explorations* 2.

Calin, G.A. et al. (2002). "Frequent deletions and down-regulation of micro-RNA genes miR-15 and miR-16 at 13q14 in chronic lymphocytic leukemia." In: *Proc. Natl. Acad. Sci. U. S. A.* 99, pp. 15524–15529.

Cho, Haeran and Piotr Fryzlewicz (2015). "Multiple-change-point detection for high-time series via sparsified binary segmentation". In: *J. R. Stat. Soc. Ser. B. Stat. Methodol.* 77.2, pp. 475–507.

Condrat, Carmen Elena et al. (2020). "miRNAs as biomarkers in disease: latest findings regarding their role in diagnosis and prognosis". In: *Cells* 9.2, p. 276.

Galvão-Lima, L.J., A.H.F. Morais, R.A.M. Valentim, et al. (2021). "MiRNAs as biomarkers for early cancer detection and their application in the development of new diagnostic tools". In: *BioMed Eng OnLine* 20.21.

Han, Y. et al. (2020). "Statistical approaches using longitudinal biomarkers for disease early detection: A comparison of methodologies." In: *Stat Med.* 39, pp. 4405–4420.

Jirak, Moritz (2015). "Uniform change point tests in high dimension". In: *Ann. Statist.* 43.6, pp. 2451–2483.

Kenett, R.S. and M. Pollak (2012). "On assessing the performance of sequential procedures for detecting a change". In: *Quality and Reliability Engineering International* 28.5, pp. 500–507.

Li, Jing-Hua et al. (2017). "MiR-205 as a promising biomarker in the diagnosis and prognosis of lung cancer". In: *Oncotarget* 8.54, p. 91938.

Li, Sai, T. Tony Cai, and Hongzhe Li (2021). "Inference for High-Dimensional Linear Mixed-Effects Models: A Quasi-Likelihood Approach". In: *Journal of the American Statistical Association* 0.0, pp. 1–12. DOI: 10.1080/01621459.2021.1888740.

Linderman, K., K.E. McKone-Sweet, and J.C. Anderson (2005). "An integrated systems approach to process control and maintenance". In: *European Journal of Operational Research* 164.2, pp. 324–340.

Lopez-Rincon, Alejandro et al. (2020). "Machine Learning-Based Ensemble Recursive Feature Selection of Circulating miRNAs for Cancer Tumor Classification". In: *Cancers* 12.7.

Lu, Thomas X. and Marc E. Rothenberg (2018). “MicroRNA”. In: *Journal of Allergy and Clinical Immunology* 141.4, pp. 1202–1207.

Ludwig, Nicole et al. (Feb. 2016). “Distribution of miRNA expression across human tissues”. In: *Nucleic Acids Research* 44.8, pp. 3865–3877.

Mehrafrooz, Z. and R. Noorossana (2011). “An integrated model based on statistical process control and maintenance”. In: *Computers & Industrial Engineering* 61.4, pp. 1245–1255.

Or, Gilad and Isana Veksler-Lublinsky (Mar. 2021). “Comprehensive machine-learning-based analysis of microRNA-target interactions reveals variable transferability of interaction rules across species”. In: *BMC BioInformatics* 22.

Panagiotidou, S. and G. Tagaras (2010). “Statistical process control and condition-based maintenance: A meaningful relationship through data sharing”. In: *Production and Operations Management* 19.2, pp. 156–171.

Poor, H.V. and O. Hadjiliadis (2008). *Quickest Detection.* Cambridge University Press.

Rincon, Alejandro Lopez et al. (2019). “Automatic discovery of 100-miRNA signature for cancer classification using ensemble feature selection”. In: *BMC Bioinformatics* 20.1.

Sapre, Nikhil et al. (2016). “A urinary microRNA signature can predict the presence of bladder urothelial carcinoma in patients undergoing surveillance”. In: *British Journal of Cancer* 114, pp. 454–462.

Townsend, N., D. Kazakiewicz, F. Lucy Wright, et al. (2022). “Epidemiology of cardiovascular disease in Europe.” In: *Nat Rev Cardiol* 19, pp. 133–143.

Wainwright, Martin J. (2019). *High-dimensional statistics – A non-asymptotic viewpoint.* Vol. 48. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, Cambridge.

Weinstein, J.N., E.A. Collisson, and et al (2013). “The Cancer Genome Atlas Pan-Cancer analysis project”. In: *Nat Genet.* 45.10, pp. 1113–1120.

Zhong, Ping-Shou, Jun Li, and Piotr Kokoszka (2021). “Multivariate analysis of variance and change points estimation for high-dimensional longitudinal data”. In: *Scand. J. Stat.* 48.2, pp. 375–405.

Optimizing Parcel Transportation of PostNL

Ruben Hoeksma[1], Christopher Hojny[2], Jan-Kees van Ommeren[3], Matthias Walter[4]

Abstract

We consider a multi-commodity transportation problem that arises in parcel delivery in the Netherlands. We focus on its deterministic variant and propose a mixed-integer programming model to solve it. We provide an implementation that is based on a multi-stage solution approach in order to overcome computational difficulties. This allows us to solve practical instances to reasonable accuracy.

Keywords: transportation, network design, multi-commodity flow

2.1 Introduction

PostNL is a Netherlands-based mail and parcel delivery company. On a daily basis, the company collects, sorts and delivers mail and parcels throughout, mostly, the Netherlands. This report addresses the optimization of the daily transportation of parcels by trucks between sorting centers in the Netherlands.

[1]University of Twente, The Netherlands
[2]Eindhoven University of Technology, The Netherlands
[3]University of Twente, The Netherlands
[4]University of Twente, The Netherlands

These parcels need to be carried between different *sorting centers*, which are the hub locations where collected parcels are redistributed and from which actual delivery to the recipients starts. To avoid having to send several trucks from every sorting center to every other sorting center, PostNL runs several so-called *cross docks* which are simply hubs in the distribution network. We consider only the transportation of *trolleys*, which are the units of identical size that are loaded with parcels at their origin sorting center and are sent through the network to their destination sorting center.

Deterministic problem. For the deterministic version of the problem we are given a set L of sorting locations, of which some have cross-dock capabilities, which we denote by $L^{\times} \subseteq L$. For every pair $i, j \in L$ $(i \neq j)$ of locations we know the driving times $d(i,j) \in \mathbb{R}_{\geq 0}$ from i to j. There is a set K of commodities to be sent through the network, which correspond to the labels that are attached to each trolley sent through the network. Each such commodity $(i,t) \in K$ consists of a location $i \in L$ and a deadline $t \in \mathbb{R}$ until which all trolleys of this commodity have to be delivered to their destination i. In the idealized deterministic setting, a set G of generated trolleys is given. For each trolley, we are given a tuple $(i,t,i',t') \in G$, where $i \in L$ denotes the location where it appears, a *release time* t at which it appears and a commodity $(i',t') \in K$ that indicates to which destination i' and by which time t' it must be delivered. Note that although identical tuples can occur multiple times, we consider them as unique for ease of notation. Our implementation deals with multiplicities properly. Transportation is done by identical trucks that each have a capacity of $U \in \mathbb{Z}_{\geq 0}$ trolleys. Moreover, loading and unloading a truck at some location takes an amount of time, $t_{\text{load}}(i) \in \mathbb{R}_{\geq 0}$ and $t_{\text{unload}}(i) \in \mathbb{R}_{\geq 0}$ for each location $i \in L$, respectively. At each location $i \in L$, at most $c_{\text{park}}(i) \in \mathbb{Z}_{\geq 0}$ trucks can load or unload at the same time. Finally, at most $c_{\text{out}}(i) \in \mathbb{Z}_{\geq 0}$ generated trolleys can wait for being transported, at most $c_{\text{in}}(i) \in \mathbb{Z}_{\geq 0}$ trolleys can wait at their destination for their deadline and $c_{\times}(i) \in \mathbb{Z}_{\geq 0}$ trolleys can wait for further transportation after they have arrived at some cross dock $i \in L^{\times}$.

Demand forecasts. We were provided with a deterministic instance of the optimization problem we just described. However, knowledge of the exact time at which a trolley is ready to be shipped is quite unrealistic. In practice, PostNL has a demand forecast whose accuracy is unknown to the authors. Moreover, there are further sources of uncertainty, e.g., the actual traveling times or breakdowns in any part of the logistics chain. However, as it will turn out already the deterministic optimization problem is not easy to solve at all. This justifies our focus on the perfect-knowledge version of the real-world problem.

Outline. The paper is structured as follows. We first describe our modeling approach by means of time discretization in Section 2.2. In that section we also derive a mixed-integer programming model and refine it as an attempt to deal with robustness problems. Our solution approach is described in Section 2.3 and the corresponding implementation prototype is explained in Section 2.4. Our results are presented in Section 2.5. We conclude our paper with future recommendations for PostNL in Section 2.6.

2.2 Time discretization and MIP models

In this section we first describe how we discretize the times that are relevant to our problem. Then we explain an auxiliary graph that is useful to derive a mixed-integer programming model for the problem. Finally, a basic and a refined MIP model are described.

2.2.1 Time-expanded network

We first choose a parameter $\Delta > 0$ and then transform all times t to t/Δ, which we call *ticks*. In fact, we only consider integer tick values. To this end, transformed release, loading and unloading times are rounded up to the next integer, while deadlines are rounded down. We denote these rounded values by a Δ superscript, e.g., $t^{\Delta}_{\text{unload}}(i) := \lceil t_{\text{unload}}(i)/\Delta \rceil$. This rounding procedure is conservative in the sense that the available time to route a trolley is never increased, while the time necessary to transport it is never decreased. In particular, if Δ is larger than $t_{\text{load}}(i)$ or $t_{\text{unload}}(i)$, then rounding up several parameters yields values that sometimes overestimate the actual transport time by too much. To resolve this problem, we do not transform the driving times independently, but instead define the *travel ticks* as

$$d^{\Delta}(i,j) := \lceil (d(i,j) + t_{\text{load}}(i) + t_{\text{unload}}(j))/\Delta \rceil .$$

We denote by $t_{\min} \in \mathbb{Z}$ and $t_{\max} \in \mathbb{Z}$ the largest (resp. smallest) tick such that $T := [t_{\min}, t_{\max}] \cap \mathbb{Z}$ contains all transformed release times and deadlines.

Auxiliary network. We now define the auxiliary directed graph $D = (V, A)$ with $V = L \times T$. In particular, we consider multi-commodity flows (with further restrictions) in D, where a flow unit that traverses through node (i,t) represents a trolley that is at i in tick t. Note that the distinction of incoming and outgoing trolleys is easy, since the commodity $k = (j,t)$ contains the destination information and we have $i = j$ if and only if the trolley has i as its destination. The arc set of D consists of arcs $A = A^{\text{move}} \cup A^{\text{stay}}$ with

$$\begin{aligned} A^{\text{move}} &:= \{((i,t),(j,t') : i \in L,\ j \in L,\ t,t' \in T,\ t' = t + d^{\Delta}(i,j)\} , \\ A^{\text{stay}} &:= \{((i,t),(i,t+1) : i \in L,\ t,t+1 \in T\} . \end{aligned}$$

We also introduce the following notation:

$$\begin{aligned} \delta_t^{\text{out}}(i) &:= \{a \in A : a = (i,t,j,t') \text{ for some } j \in L, t' \in T\} , \\ \delta_t^{\text{in}}(i) &:= \{a \in A : a = (j,t',i,t) \text{ for some } j \in L, t' \in T\} . \end{aligned}$$

2.2.2 Basic model

We first describe a base model for the problem. The main decision of the multi-commodity flow interpretation of the trolley routing problem is to decide how many commodities of a specific type traverse an arc $a \in A$ at a certain point of time. For arcs $a \in A^{\text{stay}}$, we need to guarantee that the capacities of the corresponding location are not exceeded. For arcs $a \in A^{\text{move}}$, we must ensure that also sufficiently many trucks are provided for transportation along a. For this reason, we introduce the following variables, modeling the previously mentioned decisions. The *truck variables*

$$x_a \in \mathbb{Z}_{\geq 0} \qquad\qquad \forall a \in A^{\text{move}} \tag{2.1}$$

model how many trucks are available for transportation along arc $a = ((i,t),(j,t'))$. We assume that the loading procedure of these trucks starts at tick t, i.e., the trucks are not necessarily leaving immediately. To make sure that capacities of locations are not exceeded, we introduce *inventory variables*

$$s_{i,k}^{t} \in \mathbb{R}_{\geq 0} \qquad\qquad \forall i \in L,\ \forall k \in K,\ \forall t \in T, \tag{2.2}$$

which keep track of the number of trolleys of commodity k that are at location i at time t. Finally, *flow variables*

$$y_{a,k} \in \mathbb{R}_{\geq 0} \qquad\qquad \forall a \in A,\ \forall k \in K \tag{2.3}$$

model the number of trolleys of commodity k that traverse arc a. Note that we model the flow of commodities using continuous variables. Preliminary computational experiments showed that switching between integer and continuous variables does not make much of a difference for the optima. One potential reason is that actually sending a truck along an arc often yields enough capacity that the full demand of one commodity (at the source node) can be sent without the need to split it across different arcs. However, already for smaller examples the running time increased significantly when requiring integrality of the y-variables. Hence, we decided to make them continuous.

Constraints. To make sure that the previously introduced variables model a solution of the trolley routing problem, we introduce the following constraints. The *truck capacity constraints* guarantee that the total number of trolleys sent via an arc $a \in A^{\text{move}}$ does not exceed the capacity f the available trucks:

$$\sum_{k \in K} y_{a,k} \leq U \cdot x_a, \qquad\qquad \forall a \in A^{\text{move}}. \tag{2.4}$$

Similarly, we need to make sure that the docking capacities at each location are not exceeded. That is, the total number of trucks arriving and departing from a certain location i must not exceed the number of docks D_i:

$$\sum_{\tau=0}^{t^{\Delta}_{\text{unload}}(i)-1} \sum_{a \in \delta^{\text{in}}_{t+\tau}(i)} x_a + \sum_{\tau=0}^{t^{\Delta}_{\text{load}}(i)-1} \sum_{a \in \delta^{\text{in}}_{t-\tau}(i)} x_a \leq D_i \quad \forall i \in L,\ \forall t \in T. \tag{2.5}$$

Note that we need to take the summation over τ into account to also consider the trucks whose unloading (resp. loading) process at location i has not finished yet until tick t.

When routing the trolleys through the directed graph, we need to make sure that the routing adheres to a flow structure, i.e., the following *flow balance constraints* need to be satisfied for all $i \in L$, $k = (j, t') \in K$, and $t \in T \setminus \{0\}$:

$$s^{t-1}_{i,k} + \sum_{a \in \delta^{\text{in}}_t(i)} y_{a,k} - \sum_{a \in \delta^{\text{out}}_t(i)} y_{a,k} + \beta^{\text{in}}_{i,k,t} - \beta^{\text{out}}_{i,k,t} = s^t_{i,k}, \tag{2.6}$$

where $\beta^{\text{in}}_{i,k,t}$ is the number of trolleys of commodity k that are due at time t at location i, and $\beta^{\text{in}}_{i,k,t}$ is the number of trolleys created. Note that these values can be easily computed from the instance data.

Finally, we need to guarantee that the inventory capacities at all locations are not exceeded. Since locations have different capacities for outgoing and incoming trolleys, we introduce

$$\sum_{\substack{(j,t') \in K: \\ j \neq i}} s^t_{i,(j,t')} \leq c_{\text{out}} \qquad \forall i \in L \setminus L^{\times},\ \forall t \in T, \tag{2.7}$$

$$\sum_{\substack{(j,t') \in K: \\ j \neq i}} s^t_{i,(j,t')} \leq c_{\times} \qquad \forall i \in L^{\times},\ \forall t \in T, \tag{2.8}$$

$$\sum_{t' \in T} s^t_{i,(i,t')} \leq c_{\text{in}} \qquad \forall i \in L,\ \forall t \in T. \tag{2.9}$$

Objective function. Since we are interested in small total driving times, we

$$\text{minimize} \sum_{a=((i,t),(j,t')) \in A} d(i,j) x_a. \tag{2.10}$$

Note that while we use the rounded driving times (by means of travel ticks $d^{\Delta}(i,j)$) in order to define the auxiliary graph, we use the actual driving times in the objective function. Hence, the objective values of computed solutions can be related to the real world and do not require a conversion from ticks to actual times.

2.2.3 Refined model

Later, we solve the MIP (2.1)–(2.10) for a certain problem instance with a certain trolley production G. We then evaluate the solution for different other sets $G_1, \dots, G_\ell$ of generated trolleys. The purpose is to investigate the robustness of the computed solution $(x^\star, s^\star, y^\star)$ with respect to modified demands. For this we fix the truck decision variables $x = x^\star$ and try to find vectors y and s that constitute a transportation plan for a particular set G_i of generated trolleys. However, it may happen that the instance for a G_i is infeasible: for instance, after the last truck from a location leaves, another trolley appears, which has no chance of reaching its destination.

A similar problem can appear already for the first optimization with trolley production G if the discretization parameter Δ is too large: in this case, there might not be enough time to go to a destination via a cross dock, and hence much of the transportation would have to arrive exactly at the deadline which may in turn overload the available docks.

To this end, we extended the base model represented in the previous section as follows.

Not delivering trolleys. We introduce additional variables

$$p_k^{\text{in}} \in \mathbb{R}_{\geq 0} \qquad \forall k \in K \tag{2.11}$$

that count the number of trolleys of commodity k that are not delivered. To match this to the total flow balance in the network, we also introduce variables

$$p_{i,k,t}^{\text{out}} \in [0, \beta_{i,k,t}^{\text{in}}] \qquad \forall i \in L,\ \forall k \in K,\ \forall t \in T, \tag{2.12}$$

that indicate the number of trolleys if commodity k that would be produced in i at time t, but that are not released. These variables are incorporated into the basic model by adding, to the left-hand side of (2.6), the term $p^{\text{out}}_{i,k,t}$ as well as subtracting p^{in}_{k} in case $k = (i, t)$ holds. The modified constraint shall be denoted by (2.6'). In order to encourage delivery, we penalize these variables with a certain factor in the objective. In our case, we used a coefficient of 10, which is larger than the extra costs of sending a single truck (say, with the considered trolley) along the longest connection.

Extending the number of depots. For strategic planning, one may want to analyze the effect of certain restrictions. In particular, it could be interesting to judge the benefit of increasing certain capacities such as the sorting capacity of a location or the number of depots. To illustrate this flexibility of our proposed MIP approach we added a corresponding extension regarding the depot numbers. To this end, we introduced variables

$$e_i \in \mathbb{R}_{\geq 0} \qquad\qquad \forall i \in L, \tag{2.13}$$

to indicate extended docking capacity at location i. The modification of the docking constraint (2.5) is straight-forward: we replace its right-hand side with $D_i + e_i$. The modified constraint shall be denoted by (2.5') Again, we add $\sum_{i \in L} 10 e_i$ to the objective function in order to penalize this dock extension. This is solely for demonstration purposes, and for answering an actual strategic question a suitable value would have to be found. The objective function (2.10) augmented with all discussed penalty terms is denoted by (2.10').

Final model. For further reference we denote the final model as

$$\begin{aligned}&\text{minimize (2.10') over } (x, s, y, p^{\text{in}}, p^{\text{out}}, e)\\ &\qquad\qquad \text{subject to (2.1)–(2.4), (2.5'), (2.6'), (2.7)–(2.13).}\end{aligned} \tag{2.14}$$

2.3 Solution approach

Our main challenge is to produce solutions with total travel time as small as possible. For this reason, we have focused in our approach on generating solutions with a small objective value rather than deriving strong lower bounds on the optimal travel time. To find solutions that are as realistic as possible, one is interested solving Model (2.14) for a time discretization of about 15 minutes. Refining the discretization parameter Δ to $\frac{\Delta}{2}$, however, roughly doubles the number of variables and constraints in Model (2.14). This makes it a challenge to solve Model (2.14) or finding good solutions for a very fine time discretization. Our solution approach therefore consists of multiple phases in which solutions for coarse time discretizations are used to initialize the search for good solutions with a finer time discretization.

Phase 1. In Phase 1, we start with a very coarse time discretization $\Delta = 120\,\text{min}$, and our aim is to find a solution in a very limited amount of time. To this end, we define a placeholder solution $(x^\star, s^\star, y^\star)$, and initialize an upper bound on the optimal objective value of $u = \infty$. Then, we iteratively attempt to solve Model (2.14). In each iteration, we specify a time limit of 300 s and provide the model the best known solution $(x^\star, s^\star, y^\star)$ from a previous iteration as start solution. In the first iteration, this solution is a placeholder solution which results in not providing a solution at all. After the time limit is hit, we extract the best solution found during this iteration. If the objective value of the latter is smaller than $0.99u$, we replace $(x^\star, s^\star, y^\star)$ by the newly found solution and update u to its objective value. Otherwise, Phase 1 terminates and returns the best solution found so far.

Our motivation for this strategy is based on the observation that in many cases the solver was not able to improve on a found solution in a reasonable amount of time. Restarting the entire solution process and providing the best incumbent, however, the solver was able to very quickly find an improving solution.

Phase 2. In Phase 2, we refine the time discretization to $\Delta = 60\,\text{min}$. The remaining steps are essentially the same as in Phase 1. The only difference is that we provide also the first iteration a solution, namely the final solution of Phase 1, and that the time limit for each iteration is set to 1800 s.

Phase 3. Phase 3 iteratively attempts to solve Model (2.14) for $\Delta = 30\,\text{min}$. We provide the entire phase a total time limit of 86 400 s, i.e., one day. The remaining structure of Phase 3 is essentially the same as before except for the following differences. Instead of providing each iteration a fixed time limit, we work with a solution time limit. That is, we do not specify an initial time limit for each iteration, but we wait until the solver has found a solution improving on the initially provided one. Afterwards, we allow the solver to continue with the search for better solutions within the solution time limit. Our motivation for this strategy is that we observed that the solver could rather often quickly improve on a found solution, i.e., interrupting the solver right after the first improving solution has been found might have blocked it from providing even better solutions in a reasonable amount of time. For the first iteration, the solution time limit has been set to 300 s. In every succeeding iteration, we double the solution time limit if the newly found solution does not improve on the previously best known solution by at least 1 %. Otherwise, the same solution time limit is used.

2.4 Instance format and software

We implemented the MIP model (2.14) and the solution approach described in the previous section in Python. Our code is available on github[5]. Since we cannot publish the actual instances as they contain some confidential information, we describe the instance format that is used by our code.

The instances are described in 2 files, one *network file* and one *trolley file*. The former describes all properties of the network except for the actual trolleys that are sent through it. This separation allowed us to test a solution for a network with different trolley sets. The network file has the following format:

```
U <NUMBER INDICATING THE CAPACITY OF EACH TRUCK>
i <UNLOADING TIME IN HOURS>
o <LOADING TIME IN HOURS>

# List of locations, one per line.
# <NAME> is a string
```

[5]https://github.com/discopt/postnl

```
# <X> is the longitude
# <Y> is the lattitude
# <OUT-CAPAC.> is the outgoing capacity
# <IN-CAPAC.> is the incoming capacity
# <CROSS-CAPAC.> is the crossdocking capacity.
# <NR. OF DOCKS> is the number of docks.
l <NAME> <X> <Y> <OUT-CAPAC.> <IN-CAPAC.> <CROSS-CAPAC.> <NR. OF DOCKS>
...

# List of distances, one per line.
# <i> and <j> are numbers from 0 up to |L|-1.
d <i> <j> <DISTANCE FROM i to j>
...

# List of commodities, one per line.
# c <TARGET LOCATION> <SHIFT NUMBER> <DEADLINE TIME IN HOURS>
...
```

The *trolleys file* is a CSV file with `;` as a separator character. It contains one header line and otherwise lines of the form

```
<NAME1>;<NAME2>;<SHIFT NUMBER>;<TIME>
```

where first two columns refer to names of some locations $i, j \in L$ from the network file, j together with the shift number constitute a commodity, and the last column specifies the time at which the trolley is spawned.

All our experiments were run on a cluster with 32 Intel Xeon Gold 5217 CPU (3.00 GHz) processors, a total of 64 GB of RAM running an Ubuntu Linux with Gurobi 9.5.1rc2 on 4 threads.

2.5 Results

PostNL provided us one instance on that we could test our solution approach. This instance features 31 locations of which six are classified as cross docks; the remaining 25 locations are regular sorting centers. A regular sorting center has a capacity of 400 outgoing and 1200 ingoing trolleys, for cross docks no limits on the trolley capacities are imposed. Trolleys are assumed to start arriving at sorting centers late afternoon and need to be shipped to their destination until an individually specified time the next morning. Finally, a truck is assumed to have a capacity of 48 trolleys and loading (resp. unloading) a truck takes 10 minutes (resp. 15 minutes).

2.5.1 Solution quality and robustness

One downside of our solution approach is that the produced solution arguably is not robust against changes in the data. In practice, this means that arrival times of trolleys are not deterministic and one needs to find a schedule of trucks that is able to transport as many trolleys on time while still minimizing total mileage. Therefore, we have tried increase robustness of our solution by reducing capacities of sorting centers and/or trucks with which we compute it. If we reduce, for example, the outgoing capacities of sorting centers, we are able to deal with uncertainties of arrival times of trolleys; reducing the truck capacities sends more trucks than strictly needed in the considered scenario. I.e., if more trolleys arrive than expected, we increase the chance that all trolleys can be delivered on time.

In our experiments, we used the method described in Section 2.3 to produce solutions for the original instance (referred to as "original" in the following) provided by PostNL as well as three variations to add aspects of robustness to the solution approach. To this end, we either reduced the outgoing capacity (O) by 25 %, the truck capacity (T) by 8.3 %, or both (OT). Table 2.1 summarizes our results. For each of the three phases, it shows the value of the best known solution (columns 2–4) as well as the number of iterations within this phase (columns 8–10). Moreover, it provides the best known final dual bound (column 5) and the corresponding gap (column 6) in the 30 minutes discretization as well as the final penalty value (column 7) caused by trolleys not delivered on time.

We can see that the formulations with a very coarse discretization of $\Delta = 120\,\text{min}$ are able to provide good solutions in terms of total mileage as the primal bound of Phase 1 is much smaller than the primal bound of Phases 2 and 3. In particular, the primal bounds of Phase 1 almost match the final dual bounds. From a practical point of view, however, these solutions are not useful as the trucks can only leave every two hours. Introducing finer discretizations, Gurobi is only able to find solutions with a relatively large mileage in comparison to the coarse

Table 2.1: Overview of numerical results using the approach of Section 2.3.

	primal bounds/phase			best dual	gap	penalty	#iter./phase		
	1	2	3				1	2	3
original	1333.6	1743.3	1559.6	1330.6	14.7 %	60	4	1	8
O	1371.0	1858.7	1546.1	1326.1	14.2 %	40	6	1	8
T	1439.0	1815.9	1641.9	1417.1	13.7 %	40	5	1	8
OT	1468.6	1933.7	1775.4	1416.8	20.2 %	40	5	1	8

discretization. But note that we cannot conclude that the mileages from Phase 1 are also the right mileages for Phase 3 as also the arrival time of trolleys gets discretized.

Reductions of outgoing- or truck capacities lead, in general, to an increase of value for the best known incumbent solution. The only exception is the reduction of outgoing capacity, where the objective value drops slightly in comparison with the original instance. Reducing the truck capacity (both capacities) leads to increase of the best incumbent's objective value by 5.3 % (13.8 %). Of course, since we could solve neither of the four models to optimality, we cannot conclude that the price of robustness in the sense of the different variations is exactly this value. However, it indicates that the increase in total mileage and penalty values can be rather large. For this reason, it might be interesting to explore different ways to robustify our approach to find robust solutions that have less impact on total mileage.

Table 2.2 depicts the results for different scenarios. A scenario is given by its trolley production G_i and we evaluated our solution (keeping the truck decision that we computed for our instance) for 9 such sets, all of which were provided by PostNL. Interestingly, in terms of robustness the original instance and the one with a reduction of the truck and the outgoing capacities (OT) gives the most robust results. Unfortunately, we could not determine an actual reason for this surprising behavior. In fact, for different solutions that we had produced during the project the robustness of the original solution was much worse. Hence, we conjecture that more computation time (per scenario) would

Table 2.2: Robustness of solution when challenged with different sets of trolley productions. The column base indicates the number of undelivered trolleys and required extra docks for the trolley production that was used as input for our solution approach. The nine further columns indicate these amounts for different productions, and the right-most column shows the average over these 9 instances.

instance	obj. value	undelivered trolleys / extra docks for different scenarios										
		base	1	2	3	4	5	6	7	8	9	average
original	1559.6	3/0	17/1	11/3	11/0	18/0	14/0	15/0	16/0	14/1	17/1	15.1/0.7
O	1546.1	2/0	27/0	80/0	16/0	44/0	18/3	47/1	29/0	20/0	49/4	37.0/0.9
T	1641.9	2/0	65/0	75/0	14/0	39/0	21/0	42/0	20/0	16/0	49/0	38.3/0.0
OT	1775.4	2/0	63/0	19/0	7/0	5/0	7/0	2/0	13/0	12/0	10/0	15.6/0.0

be needed to get a fair assessment of the robustness.

2.5.2 Insights from the solution

We now analyze the properties of the computed solution. Figure 2.1 depicts all connections that are used at all over the day.

It is easy to see that the resulting graph is relatively dense, but also that many of the connections are green or blue, indicating that at most 2 trucks use the connection. However, we believe that this large amount of direct connections is due to the disretization error. If $\Delta \geq 30\,\text{min}$, some indirect connections are infeasible, although in practice they would have been feasible. Since one cannot extract any detailed information from this map, we also made corresponding maps that only depict the direct (resp. indirect connections (see Figure 2.2)).

To untangle the large amount of connections from Figure 2.1 even further, we depict in Figure 2.4 the proposed truck activity on an hourly basis. First, most of the trucks do not depart very early since there are not enough trolleys to be transported. Second, there are only a few late trucks, which is to be expected because the deadlines of the commodities differ significantly. Note that the figures depict truck activity per hour whereas our final time discretization is $\Delta = 30\,\text{min}$, that is, solution values are aggregated. Moreover, our very first time

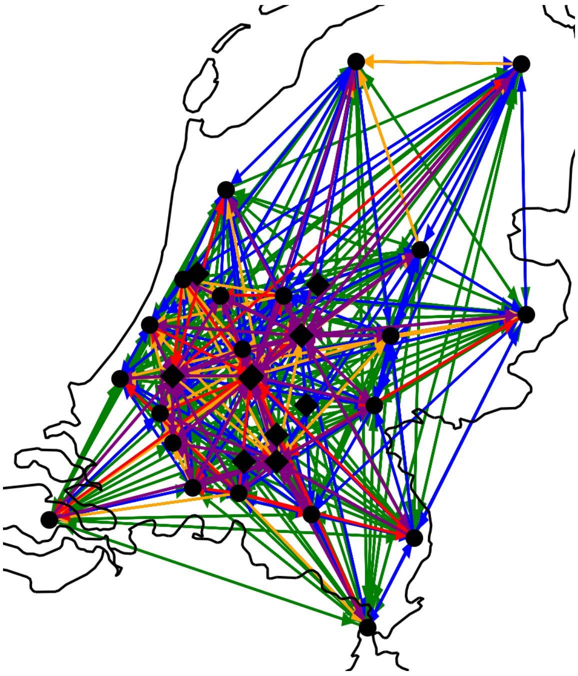
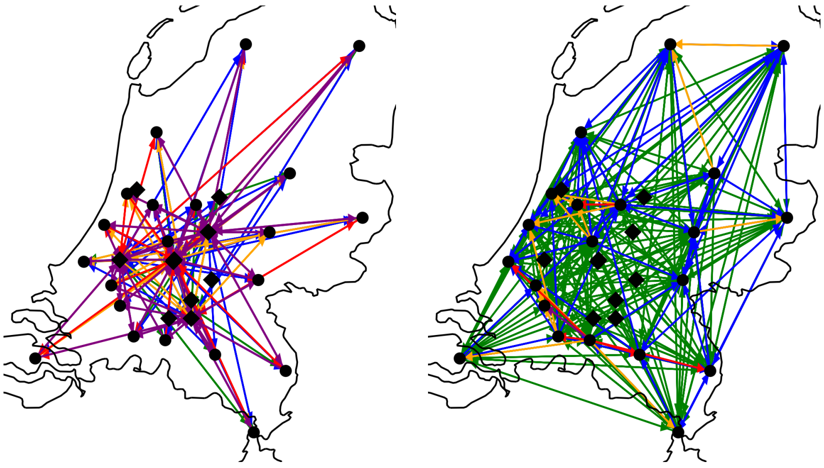


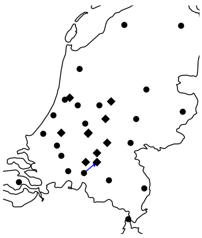
Figure 2.1: Map with all used connections in our computed solution. Colors indicate the number of times a connection is used (green: 1 truck; blue: 2 trucks; orange: 3 trucks; red: 4 trucks; purple: ≥ 5 trucks).

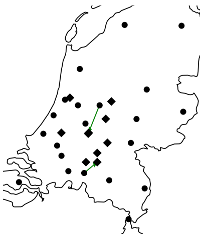


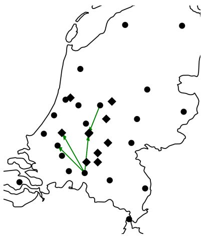
(a) Indirect connections.

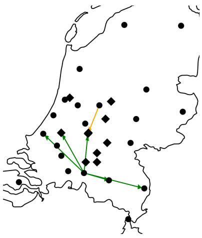
(b) Direct connections.

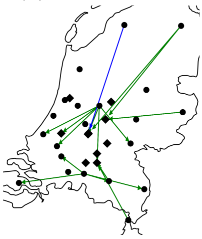
Figure 2.2: Maps with all connections in our computed solution that either involve a cross dock (2.2a) or are direct (2.2b). Colors indicate the number of times a connection is used (green: 1 truck; blue: 2 trucks; orange: 3 trucks; red: 4 trucks; purple: ≥ 5 trucks).

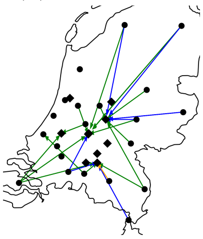

(a) Time slot 1.

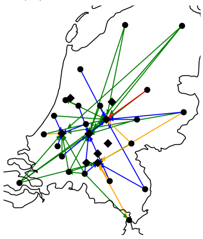

(b) Time slot 2.

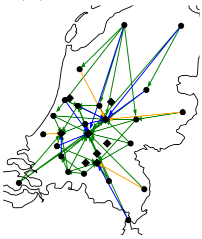

(c) Time slot 3.

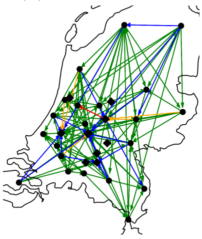

(d) Time slot 4.

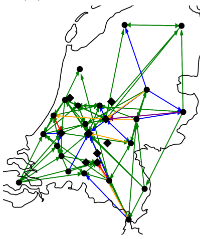

(e) Time slot 5.

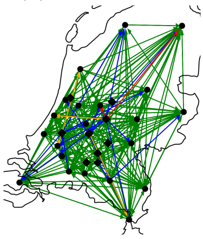

(f) Time slot 6.

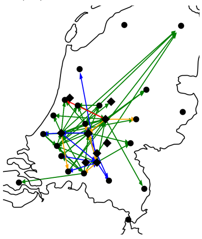

(g) Time slot 7.

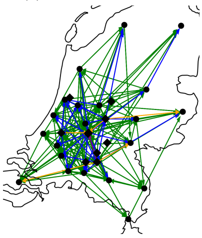

(h) Time slot 8.

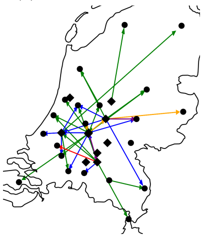

(i) Time slot 9.

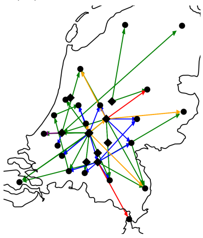

(j) Time slot 10.

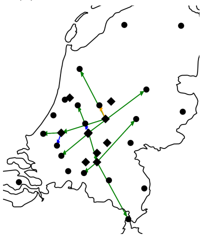

(k) Time slot 11.


(l) Time slot 12.


(m) Time slot 13.


(n) Time slot 14.


(o) Time slot 15.


(p) Time slot 16.

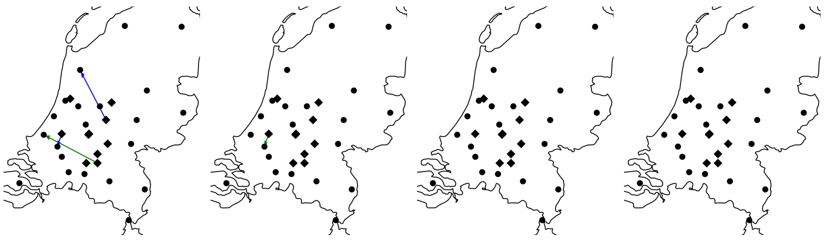
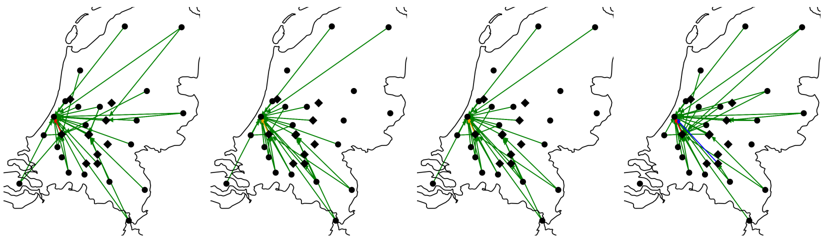

(p) Time slot 17. (q) Time slot 18. (r) Time slot 19. (s) Time slot 20.

Figure 2.4: Maps with connections in our computed solution distributed over 20 subsequent time intervals of length 1 h each. Time slot 1 is the hour of the first released trolley, and time slot 20 is the hour in which the last trucks arrived at their destinations. Colors indicate the number of times a connection is used (green: 1 truck; blue: 2 trucks; orange: 3 trucks; red: 4 trucks; purple: ≥ 5 trucks).

discretization was $\Delta = 120\,\text{min}$, which seems to have an effect on the computed solution: it is apparent that there is much more activity in time slots 9, 11 and 13 than in 8, 10, 12 or 14. We suspect that indeed this is because our computation for $\Delta = 60\,\text{min}$ was warm-started from a solution with $\Delta = 120\,\text{min}$. This implies that there might be potential for improving the computed solution by making use of these less used slots.

Finally, it is worth to have a closer look at how a specific sorting center is connected. In Figures 2.5 and 2.6, we depict the trucks that have a particular sorting center as a destination, where we vary the shifts (and thus the deadlines). Note that while a connection indicates that there is a truck going from a sorting center to another sorting center with trolleys for the specific deadline, this does not mean that all trolleys on such a truck have the same deadline. In particular, if there is only a direct connection from a location i to destination location j for two shifts then it is likely that these connections represent the same truck.

Our first observation is that the overall picture does not change significantly over subsequent shifts, which may indeed be due to sharing of trucks. Of course, some connections appear or disappear, which often



(a) Shift 4. (b) Shift 5. (c) Shift 6. (d) Shift 7.

Figure 2.5: Maps with connections in our computed solution that have a distribution center in the west as its destination, but have different shfits/deadlines. Colors indicate the number of times a connection is used (green: 1 truck; blue: 2 trucks; orange: 3 trucks; red: 4 trucks; purple: ≥ 5 trucks).

happens for single truck connections (colored in green).

In Figure 2.5 one can see that the indirect connections mainly use a cross dock that is very close to the destination, and only make limited use of other cross docks. On the contrary, the destination in the east (Figure 2.6) does not have a cross dock nearby. While there are 3 cross docks of similar distance (the three most eastern ones), only one of them is heavily used. This highlights the optimization potential in our approach as compared to a solution with only direct connections, i.e., that effective aggregation of trolleys can save many truck rides.

We have pointed out a few observations on our computed solution, but we believe that much more insight can be gained when studying it with appropriate background knowledge. In particular, an in-depth comparison to the actual transportation plan is beyond the scope of this paper. Similarly, an evaluation of the computed solution by means of a simulation (that might be based on a more realistic model) would be interesting.

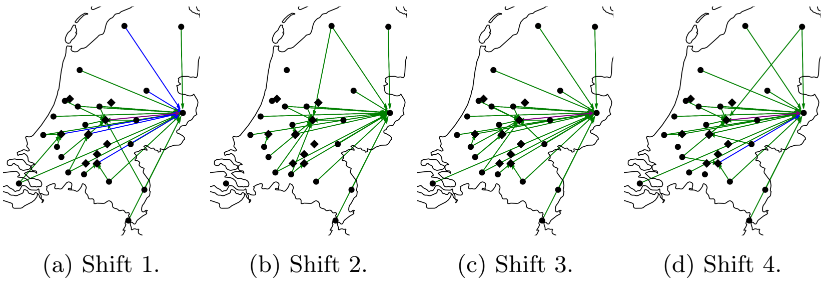


Figure 2.6: Maps with connections in our computed solution that have a distribution center in the east as its destination, but have different shfits/deadlines. Colors indicate the number of times a connection is used (green: 1 truck; blue: 2 trucks; orange: 3 trucks; red: 4 trucks; purple: ≥ 5 trucks).

2.6 Conclusions and recommendations

We believe that our computed solutions indicate that our solution approach is promising in general. However, we also think that the ability to solve instances with $\Delta \leq 15\,\text{min}$ would be crucial to obtain solutions that are close to being practically relevant. For this reason we would like to point out relevant work in the literature as we think that this is most useful for PostNL to finally obtain a method that can be used in practice.

2.6.1 Column generation

One idea to deal with the large number of variables is to apply column generation in order to only explicitly work with a subset of the variables. The other variables are implicitly set to 0, and a so-called *pricing problem* has to be solved in order to generate promising columns that are turned into explicit variables. Computational results are reported in Gendron and Larose (2014). For a general introduction to column generation, we refer to Desrosiers and Lübbecke (2005), Lübbecke and Desrosiers (2005), and Vanderbeck (2005).

2.6.2 Benders' reformulation

One idea to avoid a large number of continuous variables (in our case all but the x-variables) is that of a Benders' reformulation Benders (1962). Instead of working with an LP relaxation $Q \subseteq \mathbb{R}^{n+d}$ (where n indicates the number of x-variables and d the number of other variables), one works with the projection $P \subseteq \mathbb{R}^n$ of Q on the these n variables. The obvious advantage is that the number of variables is decreased, which in our case would be a reduction by a factor larger than 200. The disadvantage is that one has to be able to describe P by means of linear inequalities, which are usually too many to state explicitly. Hence, one needs to generate the inequalities describing P on demand, i.e., be able to find out if a given $\hat{x}$ lies in P or not, and in the latter case, find a violated inequality. This problem is known as the *separation problem.*

One way to do this is to try to *lift* the vector $\hat{x}$ to one in Q, i.e., to solve the LP of finding $(\hat{x}, z) \in Q$ for given $\hat{x}$. If there exists such a z, then $\hat{x} \in P$ follows. Otherwise, one obtains so-called Farkas multipliersSchrijver (1986, Section 7.3) which can be used to derive a (violated) inequality in the x-space. This approach still requires to solve an LP with a huge amount of y-variables and s-variables, which may not be practical. However, certain classes of inequalities (that are part of P's inequality description) are known for which the separation problem can be solved more effectively. A prime example are cut-based inequalities: if one partitions the node set of a network into two sets V_1 and V_2, then the total transportation demand for trolleys that originate somewhere in V_1 but must be carried to a destination in V_2 is known. This implies that the total number of trucks sent along arcs from V_1 to V_2 must be at least this total demand divided by the truck capacity. Such inequalities can be computed using maximum flow algorithms, which is generally much faster than large LPs. More such inequalities and their computational impact can be found in Bienstock and Günlük (1996), Sridhar and Park (2000), Costa, Cordeau, and Gendron (2009), Raack (2012), and Agarwal and Aneja (2017)

2.6.3 Solving multi-commodity flow subproblems

Despite the use of known inequalities in the Benders' approach sketched above, one may need to solve the underlying multi-commodity flow problem for a fixed number of trucks $\hat{x}$. This effectively yields a multi-commodity flow problem in the (time-expanded network) where the $\hat{x}$-vector imposes capacities on certain arcs. While this problem can be phrased as an LP of which several variables can be removed because of 0 capacity, solution times may still be prohibitively large. An alternative is to cast the problem by means of path variables Tomlin (1966). More precisely, for each commodity and suitable path (having a source node with production and a destination node with demand) there is a path variable. These path variables also have to be dealt with by means of column generation. Here, the pricing problem turns out to be a shortest path problem in the network which can be solved very efficiently. This approach is also promising because a commodity is typically sent via very few paths only, even way fewer than one has arcs in the network. Hence, a final LP solution can be represented very compactly in such a model and the hope is that also for intermediate solution steps, not too many such path variables must be considered at the same time.

There also exist other approaches to solve the problem, e.g. based on *Lagrangean relaxation*, but care must be taken that proper dual information can be extracted in order to be able to obtain inequalities for the Benders' reformulation. Computational insights appear, for instance, in Caprara (2015).

References

Agarwal, Yogesh Kumar and Yash P Aneja (2017). "Fixed charge multicommodity network design using p-partition facets". In: *European Journal of Operational Research* 258.1, pp. 124–135. DOI: `10.1016/j.ejor.2016.09.015`.

Benders, Jacques F. (1962). "Partitioning procedures for solving mixed-variables programming problems". In: *Numerische Mathematik* 4.1, pp. 238–252. ISSN: 0945-3245. DOI: `10.1007/BF01386316`.

Bienstock, Daniel and Oktay Günlük (1996). "Capacitated Network Design–Polyhedral Structure and Computation". In: *INFORMS Journal on Computing* 8.3, pp. 243–259. DOI: `10.1287/ijoc.8.3.243`.

Caprara, Alberto (2015). "Timetabling and assignment problems in railway planning and integer multicommodity flow". In: *Networks* 66.1, pp. 1–10. DOI: `10.1002/net.21611`.

Costa, Alysson M., Jean-François Cordeau, and Bernard Gendron (2009). "Benders, metric and cutset inequalities for multicommodity capacitated network design". In: *Computational Optimization and Applications* 42.3, pp. 371–392. DOI: `10.1007/s10589-007-9122-0`.

Desrosiers, Jacques and Marco E. Lübbecke (2005). "A Primer in Column Generation". In: *Column Generation.* Ed. by Guy Desaulniers, Jacques Desrosiers, and Marius M. Solomon. Boston, MA: Springer US, pp. 1–32. ISBN: 978-0-387-25486-9. DOI: `10.1007/0-387-25486-2_1`.

Gendron, Bernard and Mathieu Larose (2014). "Branch-and-price-and-cut for large-scale multicommodity capacitated fixed-charge network design". In: *EURO Journal on Computational Optimization* 2.1, pp. 55–75. ISSN: 2192-4406. DOI: `10.1007/s13675-014-0020-9`.

Lübbecke, Marco E. and Jacques Desrosiers (2005). "Selected Topics in Column Generation". In: *Operations Research* 53.6, pp. 1007–1023. ISSN: 0030-364X. DOI: `10.1287/opre.1050.0234`.

Raack, Christian (2012). "Capacitated Network Design - Multi-Commodity Flow Formulations, Cutting Planes, and Demand Uncertainty". Doctoral Thesis. Berlin: Technische Universität Berlin, Fakultät II - Mathematik und Naturwissenschaften. DOI: `10.14279/depositonce-3291`.

Schrijver, Alexander (1986). *Theory of Linear and Integer Programming.* New York, NY, USA: John Wiley & Sons, Inc.

Sridhar, Varadharajan and June S. Park (2000). "Benders-and-cut algorithm for fixed-charge capacitated network design problem". In: *European Journal of Operational Research* 125.3, pp. 622–632. ISSN: 0377-2217. DOI: `10.1016/S0377-2217(99)00272-6`.

Tomlin, J.A. (1966). "Minimum-cost multicommodity network flows". In: *Operations Research* 14.1, pp. 45–51. DOI: `10.1287/opre.14.1.45`.

Vanderbeck, François (2005). “Implementing Mixed Integer Column Generation”. In: *Column Generation*. Ed. by Guy Desaulniers, Jacques Desrosiers, and Marius M. Solomon. Boston, MA: Springer US, pp. 331–358. ISBN: 978-0-387-25486-9. DOI: 10.1007/0-387-25486-2_12.

Smartscan

Giuseppe Carere[1], Bernard Geurts[2], Bas van 't Hof[3], Fred Holtkamp[4], Erwin Luesink[5], Matthias Schlottbom[6], Julian Suk[7], Michelle Sweering[8] and Jelmer Wolterink[9]

Abstract

This report describes and develops different methods for converting 3D time series data to surface representations. The considered methods contain public domain mesh generation software, as well as linear regression models and surface representations using signed distance functions. We provide a simple code base for the latter two methods such that they can be used for further research in a simple manner. We apply and test the algorithms on a point cloud for a foot model. All methods yield good representations of the underlying foot geometry. Such algorithms can therefore have a big impact on handling foot problems.

[1]TU Eindhoven, The Netherlands
[2]University of Twente, The Netherlands
[3]Vortech BV, The Netherlands
[4]Fontys Paramedische Hogeschool, The Netherlands
[5]University of Twente, The Netherlands
[6]University of Twente, The Netherlands
[7]University of Twente, The Netherlands
[8]CWI, The Netherlands
[9]University of Twente, The Netherlands

KEYWORDS: meshing tools, signed distance function, implicit neural representation, linear regression

3.1 Introduction

More than half (57%) of the Dutch population experiences foot problems that can be treated by a pedorthist, i.e., an orthopedic shoe engineer. Approximately 20% of the foot problems can be treated with orthoses that support the natural shape of the foot. An orthosis is an externally applied device used to straighten or align parts of the neuromuscular system or the skeleton. A smaller group of people with foot problems require complex orthopedic aids, such as ankle-foot orthoses or custom-made shoes. Especially in this group, a careful fitting process is required to avoid injuries, which may lead to complications and in the worst case to amputation.

The main goal of the Smartscan project is to develop a high-tech instrument to renew the fitting/casting process to overcome shortcomings of the current method, such as risk of injuries or fittings that depend on the medicating person. With this new method, the plaster cast of the foot is replaced by a digital representation of the geometry of the foot. Based on this digital twin of the foot, it becomes possible to perform peer consultation. In addition, it provides the additional option to compare foot models digitally and remotely.

To create such a digital twin of the foot, an instrumented glove equipped with a position sensor on each fingertip and pressure sensors in the palm of the hand will be used. The benefit of an instrumented glove is that it is a minor change from the current manual approach, where latex gloves are used by the orthopaedic expert to diagnose the foot problem. The pressure sensors in the palm area of the glove measure the pressure applied by the orthopaedic expert during the casting and correction phase. The acquired pressures will give insight in the amount of occurring pressure points that can influence the desired corrections. The position and pressure data can subsequently be used to develop 3D digital computer-aided design (CAD) models that can be further edited for manufacturing. A digital representation of the geometry of the foot together with the applied pressures will assist the orthopaedic expert in finding the optimal solution.

The main research question is how to clean and control the flow of data from sensors to CAD in real time in an efficient and effective manner, as real time processing would provide the operator of the glove direct feedback.

In the following we report on several approaches to obtain a digital foot model from sensor data. The methods range from using available meshing tools, such as meshlab, a signed distance function approach, and a linear regression model. We show that all approaches yield usable models, while the latter two models can be easily used for further development.

3.2 Procedure

According to NDI[10], i.e., the manufacturer of the sensors, electromagnetic tracking works along the following steps.

1. The transmitter emits a low-intensity, varying electromagnetic field that establishes the measurement volume.

2. Small currents are induced inside the sensors when they enter the EM field.

3. These currents are relayed to the sensor interface unit, where they are amplified and digitized as signals.

4. The signals are transmitted to the system control unit, which calculates each sensor's position and orientation as a transformation.

The result of this procedure is a data set of the form as illustrated on the left in Figure 3.1. This particular dataset constructed contains $n = 3\,963$ samples in time, measured at a frequency of 50Hz. Each sample consists of 9 observed parameters for the three sensors attached to three fingers. The first parameter is time. Next, there are the recorded x, y and z coordinates of each of the sensors, and the orientation of the sensors measured with respect to the reference sensor placed on the foot,

[10]`https://www.ndigital.com/technology/em-overview/`

in the Euler angles α, β, γ. A scalar quality measure is included that indicates the amount of metal interference present in the observations for each finger. Finally, a label is included that should indicate the state of the examination: 'off foot' (0), 'on foot' (1), or 'manipulation' (2). The latter has not been used since only 0-values are reported. The data provide a rough outline of a foot but also contain the movement of the glove towards and away from the foot. To obtain a better estimate of the foot's surface at subsequent steps, we filter this data by removing the initial and final movement of the glove. On the right in Figure 3.1 a filtering procedure has been applied, which has removed the data points indicated in red. The filtering procedure is explained in more detail in Cazacu et al. (2021, Section 2).

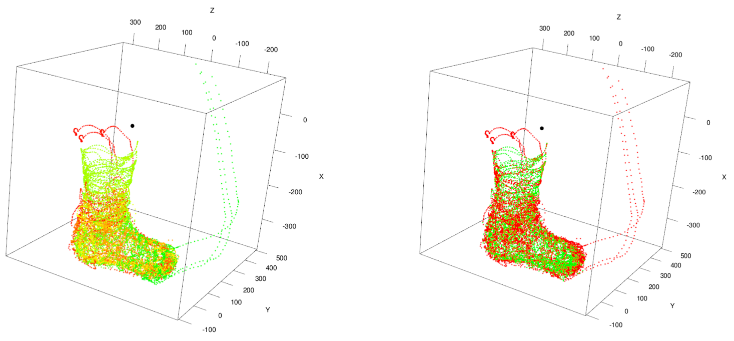


Figure 3.1: Time series position data of a calibration phase of the glove with three location sensors. The black dot indicates the origin. *Left:* time is indicated by color, from green at the initial time to red at the terminal time. *Right:* removed point in pre-applied filtering procedure are indicated in red, retained points in green.

After having established a reference data set by means of the calibration phase, the practitioner performs the diagnostics required to gain the knowledge required to construct a correcting cast. This would be for instance pressure data that would then be projected on a surface created out of the point cloud date from the calibration phase. The calibration data together with the adjustment data should be used to generate a CAD model for a correcting cast.

3.3 Methods and results

The filtered point cloud can be associated with a surface in a number of meaningful ways.

3.3.1 Meshlab

Meshlab[11] is open source software that takes as input a point cloud and outputs a meshed surface. A guideline how to create a surface mesh from point cloud data can be found here[12]. The meshed surface produced in this way by MeshLab associated to the point cloud data of the foot is shown in Figure 3.2.

Since MeshLab is a ready-to-use package, one can quickly produce surface meshes from the given point cloud data. MeshLab can export such meshes in different formats, such as STL, which can then be used for 3D printing. While the availability of different meshing techniques is quite powerful, a drawback is that several parameters can be tuned, which also needs manual interaction by an operator. Moreover, post-processing the obtained mesh needs further tools. Also, the exploitation of the available time-data seems not possible.

[11] `https://www.meshlab.net`

[12] `http://fabacademy.org/2019/docs/FabAcademy-Tutorials/week05_3dscanning_and_printing/point_cloud_mesh.html`

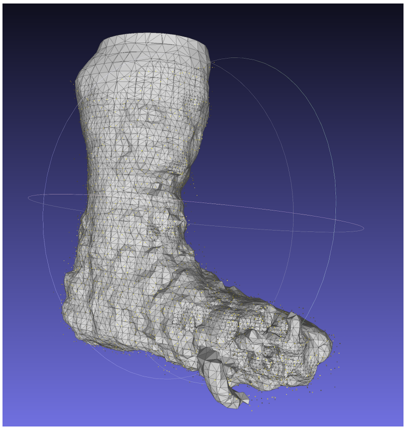


Figure 3.2: A MeshLab mesh from pre-filtered point cloud data.

3.3.2 Signed distance function

As uniform function approximators, deep neural networks can learn a wide class of functions up to arbitrary accuracy under some smoothness conditions. We propose to represent the volume V bounded by the manifold Ω by the zero level-set of its signed distance function (SDF) with arbitrary distance measure d

$$\phi_t(p)\colon p \mapsto \begin{cases} -d(\Omega(t), p) & p \in V(t), \\ d(\Omega(t), p) & p \notin V(t), \end{cases}$$

where $\Omega(t) = \{p \mid \phi_t(p) = 0\}$. We optimize a multi-layer perceptron neural network that takes as input the x, y and z coordinates of a point p, and as output provides its SDF value $d(\Omega, p)$. In machine learning research, using a neural network to represent a shape or function implicitly is known as an implicit neural representation (INR). For downstream applications, ϕ_t can be queried at any point p in V to find the surface representing the polygonal mesh using the marching cubes algorithm (see Figure 3.3).

Any suitable deep neural network can be trained to become such a signed distance function by minimising a certain objective over the

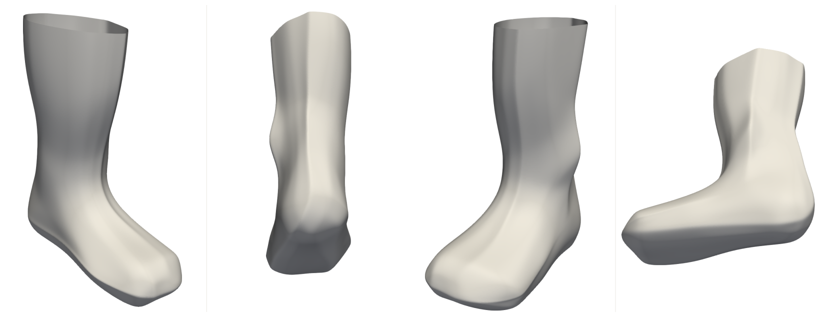


Figure 3.3: Once a SDF is available it can be queried on a voxel grid to recover an arbitrary amount of points on the manifold $p \in \Omega(t)$ at $\phi_t(p) = 0$. Marching cubes algorithm then yields a polygonal surface mesh which is depicted here.

space W of its trainable parameters. To construct a suitable optimisation objective, we sample points $q \in \mathrm{B}_r(p)$ which lie beyond the manifold $p \in \Omega$:

$$\min_{w \in W} \quad \underbrace{\log(1 + \phi_{t,w}(p))^2}_{\text{manifold loss}} + \lambda \underbrace{(\|\nabla_q \phi_{t,w}(q)\| - 1)^2}_{\text{eikonal term}} \qquad p \in \Omega(t), \quad q \in \mathrm{B}_r(p)$$

The manifold loss (compare Figure 3.4) acts as regularisation to deal with noisy measurement data by penalising negative SDF values, following the rationale that during tracing of the foot no measurement can lie inside it. This approach is similar to a (deep-learning-free) regression model without shape prior which fits the surface implicitly and by a sophisticated optimisation objective. It was inspired by Gropp et al. (2020) and uses their open-source code.

Once a SDF $\phi_t(p)$ is computed, normal vectors on the manifold are trivially available as gradient $\nabla_p \phi_t(p)$ for $p \in \Omega(t)$ on the surface. Our implementation is available on GitHub[13].

[13]`https://github.com/sukjulian/swi-fontys-smartscan`

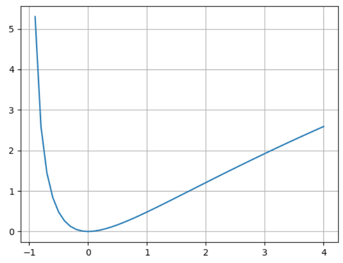


Figure 3.4: Manifold loss $f(x) = \log(1 + x)^2$ to deal with noisy point cloud data conditions the SDF to regard only the innermost points.

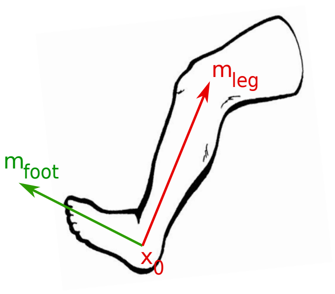


Figure 3.5: A center-line is drawn through the lower leg, and another one through the foot.

3.3.3 Linear regression model

A simple model of a person's lower leg and foot

Figure 3.5 shows a picture of a person's lower leg and foot. We will use this to make a parametric description of the leg in the case where the leg does not move. We shall discuss the situation of a moving foot later.

Center line

A center line can be computed directly from the data set to provide a basis for a linear regression model. This can be achieved by means of univariate splines for each coordinate direction. This method produces a single center line through the point cloud. The spline approach can be applied to any point cloud data set, but is sensitive to the quality of the data. It can be seen in Figure 3.6, in the picture on the right, that one side of the toe area has many more data entries than the other side. This causes the algorithm to produce a center line that is biased towards the number of measurements in a certain area. This issue can be addressed in several ways. A first option is to make sure that the data is evenly distributed, which could be difficult in practice. A second option is to increase the weight of undersampled areas, but this also amplifies outliers. A third option is to apply an additional filter to make the data set more suitable for spline methods.

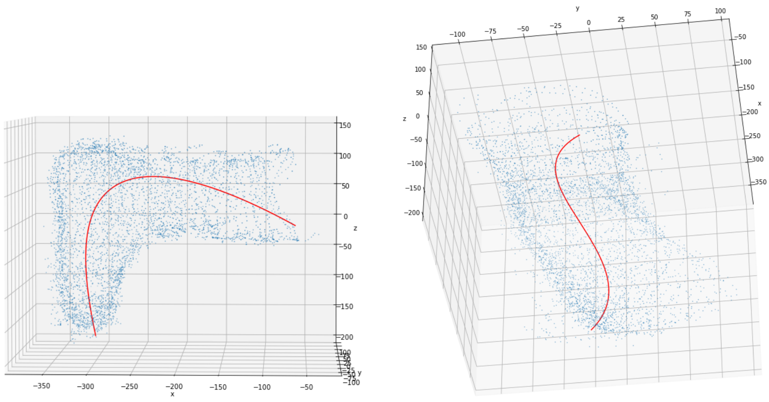


Figure 3.6: Spline fitted through the point cloud data set. The side view in the left figure shows a good fit to the data set, but the figure on the right indicates that the point cloud is undersampled on the left-hand side of the foot.

Alternatively, one can use principal component analysis to find center lines. In Figure 3.1 we can see that the orthopedic expert traces out the foot as time progresses. The idea is to regard the point cloud as time series data, and consider time windows in which the fingers of the expert have traced out a part of the foot. We choose a window of 100 data points, or two seconds, and consider only the filtered data set. If we then consider the average position of the data points in each window, we obtain a cloud of averages which lies inside the foot (see Figure 3.7 *top left*). This moving average lies somewhat around a center line through the foot, which is revealed by a principle component analysis. In particular, the first principle component contains the direction of the most amount of variance of the moving average, and is shown as the yellow line in Figure 3.7 *top right.* This trend allows us to split the leg from the foot, by considering the plane orthogonal to this line at the center. We then repeat the procedure of averaging time windows and principle component analysis for the foot part and leg part separately, to get two candidate center lines (Figure 3.7 *bottom left*). In order to have them intersect, these can be slightly adjusted, producing the desired center lines (Figure 3.7 *bottom right*). This method is fully data driven, since the time window (which we took as 100 data points) can be chosen depending on the number of data points available.

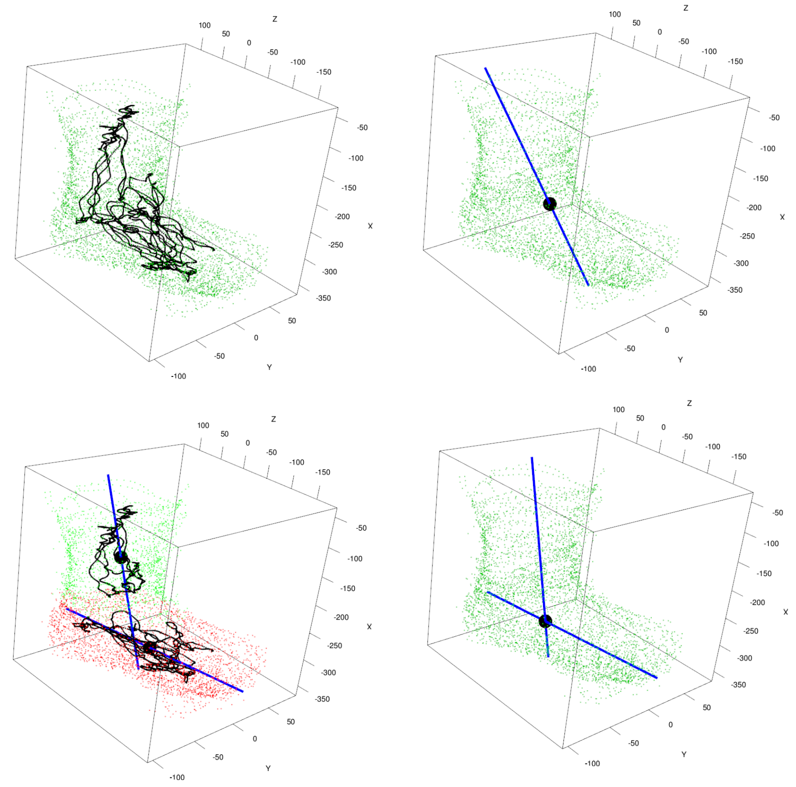


Figure 3.7: Point cloud of filtered data set. *Top left:* moving average in the inside of the foot is shown. *Top right:* line along the first principle component of the moving average is shown, with its center as black dot. *Bottom left:* point cloud is split into foot part and leg part and for each part the line along the first principle component of the corresponding moving average is shown, with centers as black dots. *Bottom right:* The obtained center lines, which intersect at the black dot.

To overcome potential errors in the data-driven centerline extraction approach, a center line can also be introduced manually. The linear regression model that we will now describe is based on two center lines.

The manual procedure uses the point $\vec{x}_0$, which is the location of the ankle joint, and the unit vectors $\vec{m}_{leg}$ and $\vec{m}_{foot}$, which indicate the direction of the leg and foot, respectively. This point and these vectors are used to define center lines, one through the leg, and one through the foot. This is illustrated in Figure 3.5 below.

We introduce the additional unit vectors $\vec{a}_{leg}$ and $\vec{b}_{leg}$, which are both perpendicular to each other and to $\vec{m}_{leg}$ for the leg. Similarly, we introduce $\vec{a}_{foot}$ and $\vec{b}_{foot}$ for the foot. Using this data, we can describe the foot and the leg as two separate shapes, each with the following parametrisation:

$$\vec{x}(l,\phi) = \vec{x}_0 + l\vec{m} + (\cos(\phi)\vec{a} + \sin(\phi)\vec{b})r(l,\phi), \tag{3.1}$$

where the symbols have the following meaning

- The origin $\vec{x}_0$,
- The leg's orientation: $\vec{m}_{leg}$, $\vec{a}_{leg}$, $(\vec{b}_{leg} = \vec{a}_{leg} \times \vec{m}_{leg})$,
- The leg's radius function: $r_{leg}(l,\phi)$
- The foot's orientation: $\vec{m}_{foot}$, $\vec{a}_{foot}$, $(\vec{b}_{foot} = \vec{a}_{foot} \times \vec{m}_{foot})$,
- The foot's radius function: $r_{foot}(l,\phi)$

Now we have two separate problems, which can be coupled because they share the point $\vec{x}_0$.

Finding the radius function

For a given orientation, the radius function can be determined. The samples are converted from $\vec{x} = (x,y,z)$ to the parametrization (l,ϕ,r):

$$\begin{aligned} l &= (\vec{x} - \vec{x}_0) \cdot \vec{m}, \\ \phi &= \text{atan2}((\vec{x} - \vec{x}_0)\vec{b}, (\vec{x} - \vec{x}_0)\vec{a}), \\ r &= |\vec{x} - \vec{x}_0 - l\vec{m}|. \end{aligned} \tag{3.2}$$

For the measurement $\vec{x}_1, ..., \vec{x}_N$ this will give parameter pairs $(l_1,\phi_1), ..., (l_N,\phi_N)$ and measured radii $(r_1, ..., r_N)$.

Next, we introduce a sampling grid for r:

$$\mathtt{r}_{i,j} \approx r(i * \delta l, j * \delta \phi) \tag{3.3}$$

And we determine interpolation weights for all available (l, ϕ) couples:

$$\begin{aligned} r(l,\phi) &= r_{\mathtt{floor}(l/\delta l),\mathtt{floor}(\phi/\delta\phi)}(1-\mathtt{mod}(l,\delta l))(1-\mathtt{mod}(\phi,\delta\phi)) \\ &+ r_{\mathtt{floor}(l/\delta l),\mathtt{ceil}(\phi/\delta\phi)}(1-\mathtt{mod}(l,\delta l))\mathtt{mod}(\phi,\delta\phi) \\ &+ r_{\mathtt{ceil}(l/\delta l),\mathtt{floor}(\phi/\delta\phi)}\mathtt{mod}(l,\delta l)(1-\mathtt{mod}(\phi,\delta\phi)) \\ &+ r_{\mathtt{ceil}(l/\delta l),\mathtt{ceil}(\phi/\delta\phi)}\mathtt{mod}(l,\delta l)\mathtt{mod}(\phi,\delta\phi) \end{aligned} \tag{3.4}$$

Doing this for the available measurements (l_n, ϕ_n), we get an overdetermined system

$$\begin{pmatrix} r_1 \\ \vdots \\ r_N \end{pmatrix} \approx \mathsf{Interp}\ \mathtt{r}, \tag{3.5}$$

for which $\mathtt{r}$ is the least squares solution. We get the result shown in Figure 3.8.

3.4 Filtered inputs

As mentioned before, the input was given to us by Fontys in two forms: there were *raw* and *filtered* data. The reconstruction process itself, however, also goes through some iterative filtering: points which are too far from the reconstructed surface are discarded.

The results are shown in Figure 3.9. When using the filtered data, the algorithm's own filtering has to be disabled, to avoid removing essential parts of the input. With these modifications in the treatment, the results based on the filtered data seem to be slightly smoother. This means that the filtering technique that was employed by Fontys did a slightly better job of removing unreliable data.

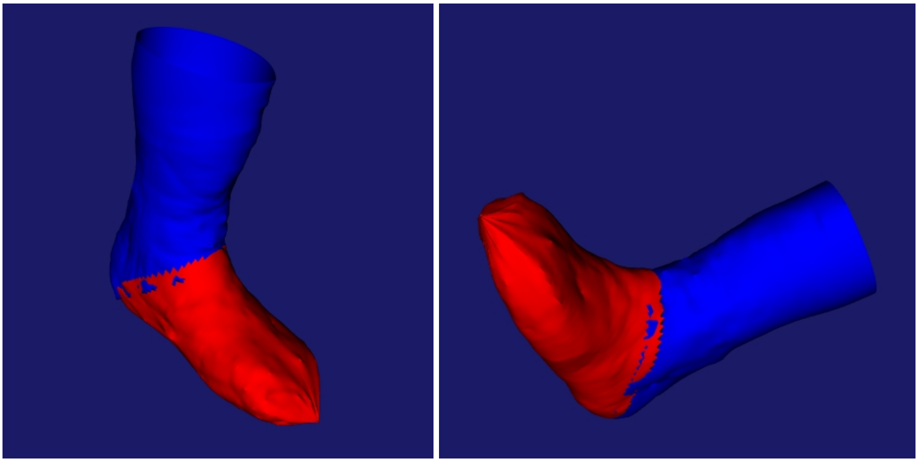


Figure 3.8: 3D pictures of the reconstructed foot using the regression approach.

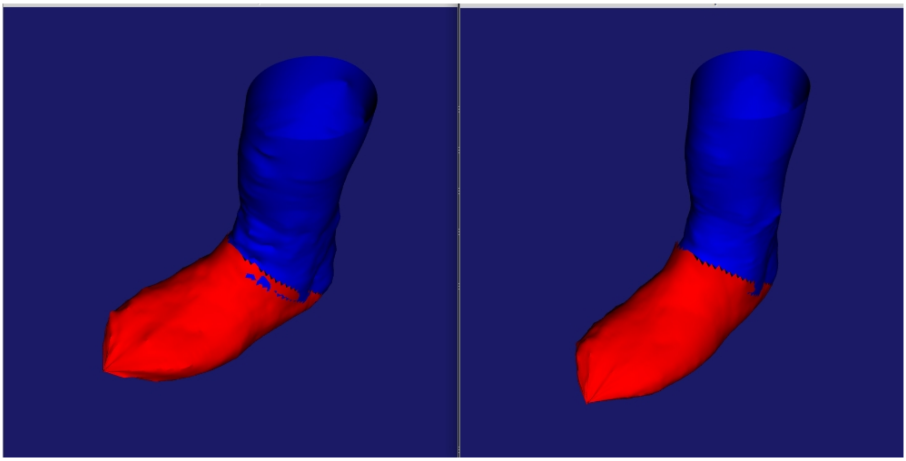


Figure 3.9: Left: reconstructed foot made from *raw* data. Right: reconstructed foot from filtered data

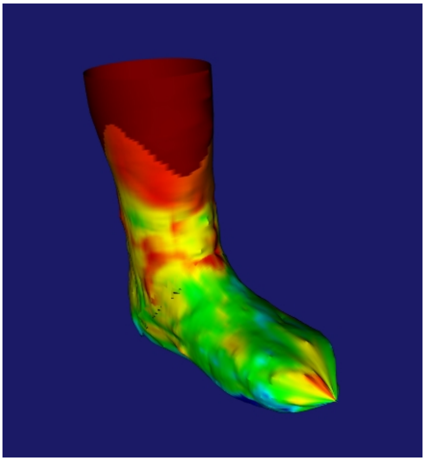


Figure 3.10: A scalar field was mapped onto the surface, and is displayed here in false-color.

3.5 Mapping a scalar field on the surface

Eventually, a goal will be to display pressure fields on the 3D image of the foot. We do not have pressure data, but for now we can use the 'quality' data that were supplied in the input.

These values (like the pressure values) are time-dependent, and we visualize only the contributions of a chosen time interval.

To map the scalar field from the data points to the surface, we first create an array that contains the scalar values in the selected data points (the ones in the chosen time frame), and zeros elsewhere. Next, we create a similar array, containing ones in the selected time frame and zero outside.

Next, we apply the same interpolation to the two arrays as we did to the input radii r. Finally, the interpolated scalar values are scaled with the interpolated ones to get the values which we display, like in Figure 3.10.

3.6 Conclusion and outlook

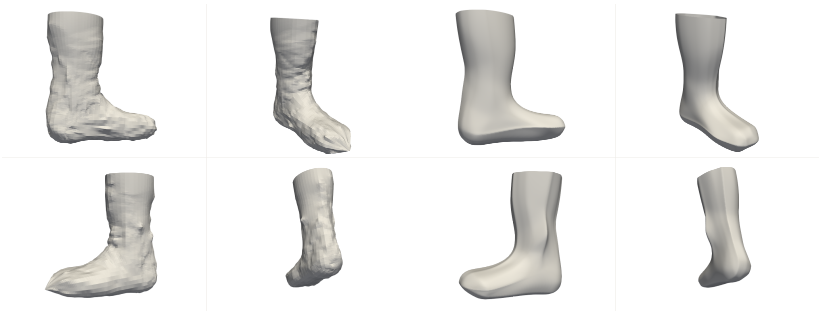


Figure 3.11: **Side-by-side comparison** of reconstructions from the linear regression model **(left)** and the implicit neural representation **(right)**. The resolution of the implicit neural representation reconstruction is chosen extremely fine at the cost of considerable computational overhead.

Figure 3.11 shows a comparison of the reconstructed foot model using the linear regression model and the neural field representation, respectively. We observe that both models represent the overall foot quite well. Both models depend on certain parameters, which influence the smoothness of the reconstructed foot, and balance smoothness against data noise. Both computational models are available in GitHub[14] and can be used for further developments.

A suggested procedure for producing more reliable point clouds using the glove is by introducing a number of calibration stages. The calibration stages depend on the glove, for instance, whether pressure sensors are included or not. If the glove has a combination of location and pressure sensors, the practitioner should touch the pressure sensor with each of the location sensors to determine the distance between the location and the pressure sensors. If the glove touches the foot, a certain minimal pressure should be applied to distinguish sensor data *touching* the foot and *outliers*.

[14]`https://github.com/sukjulian/swi-fontys-smartscan`

In case the glove only consists of location sensors, the practitioner shall first create a virtual cast of the foot in its current position. For that a prescribed number of steps may be performed to allow for simple preprocessing of the data, i.e., filtering outliers and creating easily center lines for the linear regression model.

References

Cazacu, Eduard et al. (2021). “A Position Sensing Glove to Aid Ankle-Foot Orthosis Diagnosis and Treatment”. In: *Sensors* 21.19, p. 6631.

Gropp, Amos et al. (2020). “Implicit Geometric Regularization for Learning Shapes”. In: *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event.* Vol. 119. Proceedings of Machine Learning Research. PMLR, pp. 3789–3799.

Macro Legalization in Chip Design

A. Antoniadis[1], F. Bertrand[2], S. Borst[3], F. Buccoliero[4], E. Donlon[5], W. Fokkema[6], M. Knežević, W. Moltmaker[7] and M. Overmars[8]

Abstract

The design of microchips is a complex process, and is therefore naturally broken down to the design of many smaller components. The largest such components are referred to as 'macros'. In these proceedings we investigate the problem of placing macros on a chip optimally, subject to distance and grid constraints. This problem was formulated by Synopsis®[9]. . As the general problem is known to be NP-hard, we propose several algorithms with various heuristics.

Keywords: Rectangle packing, Chip design, Mixed-integer programming

[1]University of Twente, The Netherlands
[2]University of Twente, The Netherlands
[3]Centrum Wiskunde & Informatica (CWI), The Netherlands
[4]Vrije Universiteit Amsterdam, The Netherlands
[5]Technological University Dublin, Ireland
[6]University of Twente, The Netherlands
[7]Korteweg-de Vries Institute, University of Amsterdam, The Netherlands
[8]University of Twente, The Netherlands
[9]urlhttps://www.synopsys.com/silicon-design.html

4.1 Introduction

Advanced silicon chips power the amazing software we rely on every day. Synopsys® is one of the leader in designing and verifying those complex chips. In order to design chips, several components, such as macros, standard cells and connections need to be placed on the chip. The macros, which can be seen as black-boxes, are the largest blocks. Because of their size, they are the components that must be placed in the most efficient way on a chip. Their placement, though, is constrained by spacing rules and grid alignment.

Macros should keep from each other either a fixed (small) distance or at least a (larger) distance. Moreover, they must not overlap with any other component in the chip.

The macros must align to the grid present on the chip. This constraint transforms the problem of macro-placement into a discrete problem.

The challenge that Synopsys® proposed to SWI 2022 is to design and implement a macro legalization algorithm. Such algorithm must generate legal solutions, i.e. a macro placement where spacing rules and grid alignment are satisfied. Moreover, it must be an efficient algorithm: any instance should not take more than 30 minutes to be solved.

The algorithm should aim at minimizing the macro movement from the initial configuration to the final legal one. The deviation from the initial placement must be minimal, because the position of the macro involves a software problem on top of a hardware one.

The problem proposed is similar to the horizontal rectangle packing problem, which was proved to be NP-complete in E. D. Demaine and M. L. Demaine (2007). The instance of the rectangle packing problem provided by Synopsys® differs from the widely studied one, because it presents a spacing rule for macros instead of a non-overlap condition. Similar problems have been studied before Brenner, Struzyna, and Vygen (2008) and Silvanus (2019). In order to analyse alternative approaches, we present four algorithms. The first one uses a MIP solver to tackle the problem. Two proposed algorithms place the macros in a greedy way and they are therefore called ‘greedy algorithms’. The last is inspired by Brownian motion, a probabilistic algorithm based on random walks.

In Section 4.2, the problem presented by Synopsys® is formally introduced in mathematical terms. In Section 4.3.1, the horizontal rectangle packing problem is formulated as a mixed integer program.

Section 4.3 is devoted to the introduction of the different algorithms used to solve the problem: the MIP solver, the greedy and the flexible greedy algorithm and the Brownian motion approach.

The results obtained by three of the four algorithms are presented in Section 4.4. Due to lack of time, it was not possible to implement the Brownian motion approach.

4.2 Problem Formulation

We begin by introducing our problem formally. To this end, let C denote the chip on which macros and standard cells are placed. We model C to be a square lying in the plane $\mathbb{R}^2$, equipped with a grid G whose dimensions may depend on the specific instance of the problem. (Note that G need not be aligned with C.) We model macros by rectangles in $\mathbb{R}^2$. In the problem at hand we restrict our attention to the placement of the macros and hence neglect the standard cells.

Definition 4.2.1. Let S be a set of macros. A **macro placement** of S is an assignment placing each rectangle representing a macro into C, without scaling, deforming, or rotating it.

Note that in our definition the macros in a macro placement are allowed to overlap. Such macro placements are clearly non-physical. In the problem at hand we are given such a non-physical macro placement, and are tasked with finding a 'legal' macro placement that is 'close' to the given placement. We first define when macro placements are close, reserving the exact definition of a legal placement for afterwards.

Several choices of metric are possible to encode the distance between two macro placements. Letting A be one macro in a placement P, we define $(x_{A,P}, y_{A,P}) \in \mathbb{R}^2$ to be the coordinates of its lower left corner. We also define w_A to be the perimeter of A.

Definition 4.2.2. Let P, Q be macro placements of the same set S of macros. Using the macro parameters (x_A, y_A, w_A) we define the following metrics on the space of macro placements of S:

- The linear $L1$ metric d_1:

$$d_1(P,Q) := \sum_{A \in S} |x_{A,P} - x_{A,Q}| + |y_{A,P} - y_{A,Q}|.$$

- The weighted linear $L1$ metric $d_{1,w}$:

$$d_{1,w}(P,Q) := \sum_{A \in S} w_A \left(|x_{A,P} - x_{A,Q}| + |y_{A,P} - y_{A,Q}|\right).$$

- The weighted squared linear $L1$ metric $d^2_{1,w}$:

$$d^2_{1,w}(P,Q) := \sum_{A \in S} w_A \left(|x_{A,P} - x_{A,Q}| + |y_{A,P} - y_{A,Q}|\right)^2.$$

- The weighted squared $L2$ metric $d_{2,w}$:

$$d_{2,w}(P,Q) := \sum_{A \in S} w_A \left((x_{A,P} - x_{A,Q})^2 + (y_{A,P} - y_{A,Q})^2\right).$$

Remark 1. The weighted squared linear $L1$ metric $d^2_{1,w}$ defined above does not satisfy the definition of a metric.

Indeed, the triangle inequality is in general not satisfied. Take for example a single macro A and consider the following three placements for such a macro: $x_{A,P} = y_{A,P} = y_{A,R} = 0; x_{A,R} = x_{A,Q} = y_{A,Q} = 2$. We obtain that

$$d^2_{1,w}(P,Q) = 16 > 4 + 4 = d^2_{1,w}(P,R) + d^2_{1,w}(R,Q).$$

An advantage of the weighted metrics from Definition 4.2.2 is that they prioritize the proximity of larger macros to their original placement. An advantage of the squared metrics is that they prioritize many small displacements of macros over one large displacement. Both of these priorities are beneficial to retaining the chip design of the original macro placement.

Choosing d to be one of the metrics from Definition 4.2.2, we can formulate the problem of macro legalization as follows: given a macro placement P, we wish to find a macro placement Q that minimizes $d(P, Q)$, subject to the constraint that Q be **legal**. The definition of legality is quite involved, and hence we devote the next section to it.

4.2.1 Legal macro placement

We state the definition of a legal macro placement below. We first introduce some new terminology that will be used to define a legal macro placement.

Definition 4.2.3. A *blockage* is a rectangular component of the chip C where macros must not be placed. In practice, they represent clusters of standard elements of the chip. Therefore, blockages can be viewed as macros which cannot be displaced.

Any macro may have up to four *keep-out margins*. These are distances in each of the four directions. These keep-out margins must not overlap with other macros or keep-out margins.

The chip C is provided with a discrete lattice, which will be called a *grid*.

Definition 4.2.4. Let P be a macro placement of a set S of macros, and let $0 \leq b < c$. We say P is **legal** if the following constraints on the placement of the macros are satisfied:

1. Any two macros $A, B \in S$ must not overlap.
2. Any two macros $A, B \in S$ need to be spaced at exactly a distance of b or at least a distance of c in either the x- or the y-direction. This condition is referred to as a **spacing rule**.
3. Any macro $A \in S$ must not overlap with a blockage.
4. Keep-out margins must not overlap with macros, blockages and other keep-out margins. Note that the spacing rule does not apply for the keep-out margins.
5. The lower left corner $(x_{A,P}, y_{A,P})$ of any macro must lie on a vertex of the grid.

Before we formalize the constraints we define a few more parameters of macros and blockages.

Definition 4.2.5. Let $A \in S$ be any macro. Then we will denote by l_A and h_A the width and height of the macro. Furthermore, we define m_{A_1}, m_{A_2}, m_{A_3} and m_{A_4} to be the keep-out margins for the left, bottom, right and top borders of the macro respectively.

Definition 4.2.6. Let E be the set of blockages. For any blockage $e \in E$, the coordinates of its bottom left corner are given by (x_e, y_e). The width and height are denoted by l_e and h_e respectively.

Overlap and spacing constraints

We will start with formalizing the overlap and spacing constraints of the macros. If the spacing rule is satisfied for any two macros $A, B \in S$, this immediately implies that these macros do not overlap. This is because macros A and B will be separated by at least a distance of $b \geq 0$ in one of the four directions.

To satisfy the spacing rule for macros A and B we distinguish multiple cases. Macro A needs to be to the left, right, below or above macro B. Furthermore, the distance between the macros needs to be exactly b or at least c. This leads to $4 \cdot 2 = 8$ cases, of which at least one needs to hold. For each direction we can set up an equality (distance exactly b) or inequality (distance at least c).

To give an example, if macro A is to the left of macro B exactly at a distance of b, we need that the difference between the right border $x_A + l_A$ of macro A and the left border x_B of macro B is exactly b. In total, this leads to the following eight constraints, of which at least one constraint has to be satisfied.

$$\begin{aligned}
x_B - x_A - l_A &= b \\
x_B - x_A - l_A &\geq c \\
x_A - x_B - l_B &= b \\
x_A - x_B - l_B &\geq c \\
y_B - y_A - h_A &= b \\
y_B - y_A - h_A &\geq c \\
y_A - y_B - h_B &= b \\
y_A - x_B - h_B &\geq c
\end{aligned}$$

Furthermore, we also get 4 inequalities for the keep-out margins, of which at least one has to be satisfied. Because these inequalities are similar to the inequalities for the spacing rules, it is tempting to combine these inequalities and only keep the strongest one. However, this is a simplification. For example, it is possible for two macros to satisfy the spacing rules in the vertical direction and the keep-out margins in the horizontal direction, when the macros are placed diagonally with respect to each other. All in all, it can be considered to combine the spacing rules with the keep-out margins because it reduces the number of inequalities, but it might give suboptimal solutions.

4.3 Solution approaches

We will consider four different techniques for obtaining a solution. One of them consists of solving a *Mixed Integer Programming* (MIP) formulation using a solver. In principle, this is an exact solving method that obtains optimal solutions.

Because solving the MIP to optimality is not always tractable, we also consider three different heuristics that all use a MIP-solver as subroutine: the greedy algorithm, the flexible greedy algorithm and the Brownian motion algorithm.

4.3.1 MIP formulation

Objective

If we want our model to be linear, we need to restrict ourselves to the linear metrics d_1 and $d_{1,w}$. The only problem left to tackle is the absolute values in the objectives, which are nonlinear. This can easily be modeled in the following way: we replace every absolute value $|a|$ by a nonnegative variable b and add the constraints $b \geq a$ and $b \geq -a$. These constraints are equivalent to $b \geq |a|$, and because b is minimized (because it is in the objective), we get $b = |a|$.

Spacing constraints

For each pair of macros, at least one of the spacing constraints given in Section 4.2.1 has to hold. To model this, we introduce a binary variable $d_{AB,p,s}$ for each $p \in \{x, y\}, s \in \{b, c\}$ and each pair (A, B) of macros. If $d_{AB,p,s} = 1$ the corresponding constraint has to hold. On the other hand, if $d_{AB,p,s} = 0$ we can simply add a large number M to the part of the inequality that needs to be larger. To do this for the equalities we will simply split them into two equivalent inequalities. For every pair of macros (A, B) and every $p \in \{x, y\}$ this leads to the following constraints:

$$\begin{aligned} x_B - x_A - l_A + M(1 - d_{AB,x,c}) &\geq c \\ x_B - x_A - l_A - M(1 - d_{AB,x,b}) &\leq b \\ x_B - x_A - l_A + M(1 - d_{AB,x,b}) &\geq b \end{aligned}$$

Note that if one of the binary variables equals 1, the corresponding condition must be satisfied. To ensure that at least one constraint holds, we also need the following inequality.

$$\sum_{p \in \{x,y\}} \sum_{s \in \{b,c\}} (d_{AB,p,s} + d_{BA,p,s}) \geq 1.$$

to ensure that at least one of the eight cases holds.

Blockage constraints

The blockage constraints can be added in a similar manner to the MIP formulation. For each macro-blockage pair $(A, e) \in S \times E$ we have four constraints, of which at least one needs to be active. We again will introduce a binary variable d_{Ae_j} with $j = 1, \ldots, 4$ for each constraint. Again using big-M constraints we get

$$\begin{aligned}
x_e - x_A - l_A - m_{A_3} + M(1 - d_{Ae_1}) &\geq 0, \\
x_A - m_{A_1} - x_e - l_e + M(1 - d_{Ae_2}) &\geq 0, \\
y_e - y_A - h_A - m_{A_4} + M(1 - d_{Ae_3}) &\geq 0, \\
y_A - m_{A_2} - y_e - h_e + M(1 - d_{Ae_4}) &\geq 0.
\end{aligned}$$

We then need the additional constraint that

$$\sum_{j=1}^{4} d_{Ae_j} \geq 1$$

Keepout margin constraints

For the keepout margins the process is similar. We have four constraints for every pair of macros. We introduce binary variables d_{AB_i} with $i = 9, \ldots 12$ and obtain constraints

$$\begin{aligned}
x_B - m_{B_1} - x_A - l_A - m_{A_3} + M(1 - d_{AB_9}) &\geq 0, \\
x_A - m_{A_1} - x_B - l_B - m_{B_3} + M(1 - d_{AB_{10}}) &\geq 0, \\
y_B - m_{B_2} - y_A - h_A - m_{A_4} + M(1 - d_{AB_{11}}) &\geq 0, \\
y_A - m_{A_2} - y_B - h_B - m_{B_4} + M(1 - d_{AB_{12}}) &\geq 0,
\end{aligned}$$

with the additional constraint that

$$\sum_{i=9}^{12} d_{AB_i} \geq 1.$$

Grid alignment

To enforce that the bottom left corners align with grid points in a MIP formulation, we can scale grid spacing such that every grid point is integral. We can then require the variables x_A and y_A to be integers within the MIP, so they will align with the grid points.

4.3.2 Greedy algorithm

The MIP formulation models the problem correctly, but it might become intractable for large instances. This is mainly due to the fact that for every pair of macros, we need to add multiple constraints to prevent overlap. The number of variables will also be quadratic, since we need binary decision variables for each of these constraints. As a result, the amount of constraints in the MIP formulation is quadratic in the number of macros.

Thus, we considered other algorithms such as a greedy algorithm. The basic idea of the greedy algorithm is to place the macros one by one, where each next macro does not overlap with all macros that have already been placed. The algorithm is greedy because each macro is placed as close as possible to its original location. To implement the greedy algorithm, we need to clarify two more aspects: how to determine the order in which the macros are placed and how each macro is placed.

To place the macros, we used an adapted version of the MIP formulation as described in Section 4.3.1. In this case, we only add variables for the current macro that has to be placed. The fixed macros are then treated as fixed parameters. We let A denote the macro to be placed and S_F the set of fixed macros. Then, we only need to add variables x_A and y_A for the placement and binary variables $d_{AB,p,s}$ for all $B \in S_F$ to satisfy the overlap and keepout margin constraints. The number of constraints and variables is then linear in the number of fixed macros. This should lead to better tractability, however in practice this was not always the case as can be seen in Section 4.4.2.

We could also use other approaches to do this placement step. In principle we only need to determine the regions in which the new macro can be placed to not overlap with the fixed macros. Then we simply calculate the smallest distance to the original location among these regions. This also allows us to use other metrics such as the L_2 metric, which was not possible in the MIP due to its linearity requirement.

The ordering of the macros is also very important and can have a large influence on the performance of the algorithm as can be seen in Section 4.4.2. For instance, if the first macros in the order lead to a lot of the area being blocked off, we will run into issues when placing the final macros. However, we also want that large macros are placed earlier since they have a large impact on the objective. As a result, we considered different rules for the initial placement, such as macros with the largest perimeter first, macros closest to the bottom left corner first and keeping the same ordering from the instance. The blockages were always put in front and since these do not overlap with each other they will always be placed on their fixed locations.

To improve on the performance of the algorithm, we also wanted to iteratively try different orderings. After applying the greedy algorithm to an ordering, we consider the impact that each macro had on the solution. We then provide a new ordering of the macros, where macros with large impacts are placed first. Finally, we run the algorithm again with this new ordering. We stop this iterative process if we see no improvement on the solution for some number of iterations.

The final algorithm will be the following.

1. Read the instance P and determine an initial ordering $\tilde{S}$ for the set of macros S. Also read input parameter max_{iter} and initialize $n_{iter} = 0$.
2. Initialize the set of placed macros S_F as the empty set and the value of the solution as 0. Then, for each macro A in $\tilde{S}$ we do:
    (a) Run the MIP solver on the MIP with fixed macros S_F and macro to be placed A to obtain the placement of this macro.
    (b) Add macro A to the set of placed macros S_F.
    (c) Add the objective of the MIP to the total solution value
3. Obtain the value of the solution of the algorithm.
    (a) If the value of the solution is smaller than the current best solution, set n_{iter} to 0 and continue to step 4.

   (b) If the solution value is not better, but $n_{iter} < max_{iter}$, increment n_{iter} by 1 and continue to step 4.

   (c) In the last case where the solution value does not improve and $n_{iter} > max_{iter}$ the algorithm is terminated and we return the best found solution.

4. Reorder the macros according to their impact on the objective to obtain new ordering $\tilde{S}$. Return to step 2 using this new ordering of macros.

A disadvantage of the greedy algorithm is that it does not always lead to optimal solutions and we also do not obtain a bound of the gap to the optimal solution. To see this, consider a macro placement problem where the optimal solution is to move every macro a small distance away from its original location. For any ordering of macros, the greedy algorithm will always place the first macro in this ordering at its original location. Thus, we have no guarantees that the greedy algorithm will find the optimal solution.

4.3.3 Flexible greedy algorithm

The greedy algorithm described in the previous section fixes the exact location of a newly placed macro. This can lead to situations where a macro can not be placed, due to a lack of free space. When this happens, the algorithm does not find a feasible solution, even if enough open space could be obtained by moving the already placed macros.

The 'flexible greedy algorithm' is a slight modification of the original greedy algorithm, that tries to solve this issue. It does so, by not fixing the exact location of a placed macro, but only its position relative to the other macros. Like in the last section, every iteration of the algorithm a new macro is added to the problem and the corresponding MIP is solved. But for an already placed macro A, we no longer require its position (x_A, y_A) to be equal to its previous position. Instead, for every pair of already placed macros, we fix all variables $d_{AB,p,s}$ and $d_{BA,p,s}$ for $p \in \{x, y\}, s \in \{b, c\}$. This leaves us with a larger solution space, while not making the problem much harder to solve. This is because the hardness of solving a MIP comes from the integer constraints. By fixing the integer variables corresponding to the already placed macros, we reduce the problem of placing these macros to an LP.

4.3.4 Brownian motion

The last algorithm we considered is one inspired by Brownian motion, i.e. a probabilistic algorithm based on random walks meant to approximate the optimal solution.

Remark 2. This kind of heuristic is generally also referred to as *simulated annealing.*

The idea behind this algorithm is to let each illegally placed block 'jiggle' away from its position according to Brownian motion, i.e. along a 2-dimensional random walk of its bottom-left corner along the grid. The severity of the jiggle should be inversely proportional to the weight w_A of a given macro. As random walks on a 2-dimensional square grid are expected to stay near the origin on average, it is reasonable to expect this will have a good chance of minimizing the weighted $L1$ metric.

As an additional heuristic, we demand that macros which are already optimally packed should 'stick together', so that this optimal packing is preserved during the Brownian motion even if the cluster of optimally packed macros is illegally placed. To this end we define a graph whose components are these clusters:

Definition 4.3.1. Let V be the set of macros. We let $G_x = (V, E_x)$ be the graph whose edge set E_x consists of pairs (A, B) of macros such that the horizontal distance between A and B is exactly b, and A and B are horizontally adjacent, meaning that the bottom of macro A lies below the top op macro B and vice versa. Similarly we define $G_y = (V, E_y)$ where E_y consists of pairs of macros that are vertically spaced at distance b and are vertically adjacent. We then let $G = (V, E_x \cup E_y)$ and define a **cluster** of macros to be a connected component of G. The weight of a cluster is defined by

$$w(C) := \sum_{A \in C} w_A.$$

If a macro or cluster is illegally placed, the algorithm requires some information about the direction in which this illegality occurs. This is made precise as follows:

Definition 4.3.2. Let A be an illegally placed macro or cluster. We consider the four compass directions $\{\text{up}, \text{down}, \text{left}, \text{right}\}$ in the plane. We say such a direction x is a **direction of illegality** for A if some intersection causing A to be illegally placed occurs in the x-half[10] of A. Note that any intersection defines at least two directions of illegality.

Given these preliminaries, a schematic description of the algorithm is as follows:

1. Generate the graph G and create clusters according to its connected components.
2. Determine which clusters and macros are illegally placed.
    - If there are no such macros or clusters, halt.
3. Let W be the sum of all weight reciprocals $1/w_A$ of illegally placed macros and clusters. Pick an illegally placed macro or cluster A with probability W/w_A.
4. Determine the directions of illegality of A.
5. Randomly move the macro by 1 grid space (or keep it where it is), with probabilities depending on the directions of illegality: see the list of probabilities below.
6. If the new position of A intersects a blockage or lies partially outside of the chip (and the old position did not), revert the movement.
7. Update G.
8. Return to step 2.

[10]By this we mean e.g. the left half or top half.

Below is a list prescribing which way to move a macro or cluster with which probability, depending on its direction of illegality. The probabilities are given in percentages, in the format [up/down/left/right/do not move]. These probabilities can be tweaked, of course.

- If there is illegality in all directions, we move with probabilities [20/20/20/20/20].
- If there is illegality in three directions, without loss of generality left, right, and up, we move with probabilities [5/5/5/65/20].
- If there is illegality in two directions, there are two subcases:
  - These directions are opposite, say left and right. In this case we move with probabilities [35/35/5/5/20].
  - These directions are adjacent, say left and up. In this case we move with probabilities [5/35/5/35/20].

Remark 3. During the SWI we did not get the chance to implement this algorithm due to time constraints. Nevertheless, we included a description of the algorithm here as the Synopsis® representatives thought it might hold potential.

4.4 Experimental results

The three algorithms that we implemented are solving the full problem with a MIP-solver, the greedy algorithm and the flexible greedy algorithm. In our implementations we relaxed the discrete grid constraints, allowing a macro to be placed at every continuous position. We implemented all algorithms in Python, from which we call the MIP-solver. Python is not a very fast programming language and we also did not try to optimize the running time of our implementations. However, in practice we can assume the running time of our algorithm to be dominated by our calls to the MIP-solver.

4.4.1 Using MIP-solver

Using the MIP-solver, we could not directly solve the larger instances. This was mainly due to the grid alignment constraints. In some cases, these grid alignment constraints are incompatible with the small spacing option. This could be solved by preprocessing the instance and investigating which spacing constraints are valid for which pairs of macro's. Using this approach, one could even relax the grid alignment during the solving process and round the given solution to the grid afterwards. This would preserve the b-spacing rules, but it might violate the c-spacing rules by at most 1 grid-spacing. This could be solved by increasing the c-spacing rule by half a grid-spacing, at the cost of a slight loss in the objective function. During the project, we decided to relax the grid alignment rules and otherwise solve to optimality.

Using a commercial solver, we managed to solve four of the larger instances in a reasonable time (instance 101, 102, 104 and 106). For all other large instances, the solver did not produce a reasonably good solution in the given time.

Because our MIP model is quite basic, we conclude that it might be possible to solve larger instances, but the modelling choices would need to be carefully considered. If optimality is required, then heuristic solutions might be provided at the start, and problem-specific cutting planes could be added.

The spacing and margin constraints do not have to be added for all macros because they will never be near each other in a local optimum. This could be implemented by inspecting the solution afterwards for any rule violations, and running the model with additional constraints if any rules were broken. Furthermore, a divide and conquer strategy could be implemented.

4.4.2 Greedy algorithm

Due to time constraints we did not manage to implement the reordering of the macros. We instead ran the greedy algorithm only for different initial orderings of the macros. We applied the greedy algorithms on both the small and big instances. For the small instances the greedy algorithm found optimal or close to optimal solutions very quickly. However, this was also the case for the other algorithms, so these are not very interesting to look at. In this section we will look at instance 101, which is one of the large instances. The initial placement of this instance is given in Figure 4.1.

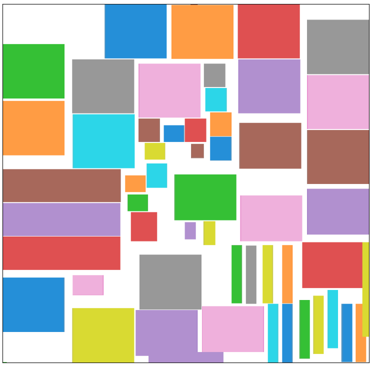
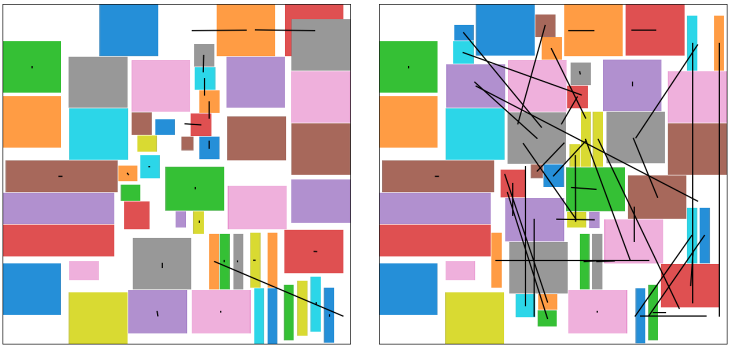


Figure 4.1: Original placement of macros in instance 101.

The greedy algorithm was applied on this instance for two different initial orderings. In the first ordering, the macros were ordered according to their distance from the origin (bottom left corner of the placement region), with small distances coming first. In the second case we put the macros with the largest perimeter first and those with the smallest perimeter last. The resulting placements can be seen in Figure 4.2.

From this figure we can see a large difference in the final placement depending on the ordering. For the ordering according to distance from the origin the macros are quite close to their original placements, with only small deviations except for some individual macros. However, the found solution is not feasible as the macros in the top right corner overlap. The final placement for the ordering where we consider the perimeter has a lot of movement of the macros. However, the final solution is feasible in this case. The likely reason for this difference is that for the distance ordering we place the macros in a structured way from the bottom left to the top right. In each placement we often only need to shift the macros a bit more to the right or top. In the end this can lead to problems, when we reach the top and right borders and do not have any space left, because all these small movements add up.

The perimeter ordering is a lot less structured, because larger macros are not necessarily grouped together. As a result, the available re-



(a) Ordered according to distance from origin.

(b) Ordered according to perimeter of the macro.

Figure 4.2: Final placements by applying the greedy algorithm on instance 101 for different initial orderings of the macros. Black lines denote the distances to the original position of the macros.

gion more quickly becomes irregular, making placement of macros quite hard. But because we placed the large macros first, we do in the end obtain a feasible solution even if it is suboptimal.

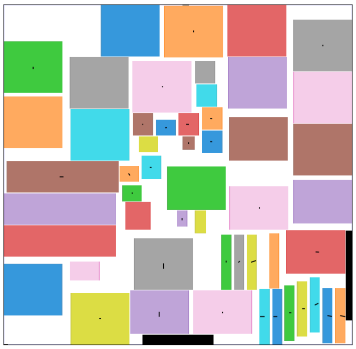
Interestingly, run time was also very different for both orderings. The greedy algorithm was quite fast for the distance ordering, while the perimeter ordering led to way longer run times likely for similar reasons. The distance ordering had a runtime of only a few minutes, while the perimeter ordering took over 20 minutes to place all the macros. This may be improved by using a different routine to place the macros instead of the MIP formulation.

A general problem of the greedy algorithm approach is that when macros become fixed, we cannot move them anymore. Thus, the macros that are placed first do not move and as such also do not take advantage of extra space which we need later when the other macros are placed. Changing the ordering does not always fix this, since then we will simply have other macros that stay in their place. The overall problem is very connected, in the sense that changing the position of one macro can influence macros on the other side of the region. This was apparent in our examples, where we either get a solution that does well for a lot of macros but in the end is infeasible, because we did not use the available space, or we get a solution that is feasible but suboptimal because the final macros have to move a lot. The other solution approaches can fare better in this aspect.

4.4.3 Flexible greedy algorithm

For the flexible greedy algorithm, we see that it generally finds substantially better solutions than the greedy algorithm. For example, we see that in Figure 4.3a the macros are substantially closer to their original position, as compared to Figure 4.2. In particular we see that in Figure 4.2 many macros are at their original position, while there are a few macros that are very far from it. On the other hand, in Figure 4.3a most macros are a small distance away from the original position. We also see that the flexible greedy algorithm finds a feasible solution for most of our test instances, whereas this was not true for the greedy algorithm.

Because the MIPs that we need to solve are more complex than in the greedy algorithm, we observe that the total MIP solving time is larger. Still, the MIPs are not as large as the MIP-formulation of the entire problem. This means that it is still tractable to solve very large instances using the flexible greedy algorithm, even if they could not be solved in the full MIP approach.


(a) Solution for `instance101`, found with the flexible greedy algorithm.

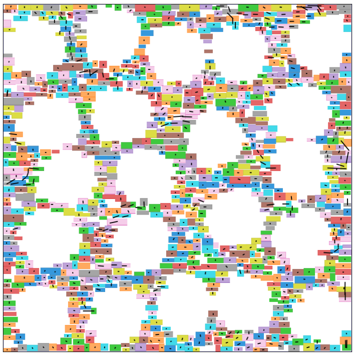

(b) Solution for `instance109`, found with the flexible greedy algorithm.

Figure 4.3: Solutions found by the flexible greedy algorithm. Black lines denote the distances to the original position of the macros.

4.4.4 Comparison

In Table 4.1 we provide a comparison of the results obtained using the different algorithms.

Instance	# Macros	L_1-error of found solution		
		MIP	Greedy	Flexible Greedy
`instance101`	59	134.70	10566.39	165.65
`instance102`	59	302.28	11092.53	364.34
`instance103`	131	-	14263.16	3795.64
`instance104`	100	704.15	2379.79	771.49
`instance105`	313	-	-	1575.85
`instance106`	13	101.66	478.86	113.99
`instance107`	147	-	-	5511.64
`instance108`	571	-	-	14717.14
`instance109`	1171	-	-	26484.18

Table 4.1: Comparison of the quality of the solutions found using the three algorithms. If an algorithm was not able to find a feasible solution, within the time limit, we denote this by a dash.

4.5 Conclusion

Summarizing, we have shown that it is tractable to solve small instances of the macro placement problem to optimality. On the other hand, for larger instances we have provided several heuristics. The solutions found by these heuristics are not optimal, but in some cases they still give solutions which do not differ greatly from the optimal one. However, the quality of these solutions greatly depends on the chosen heuristic.

We expect that the heuristics we designed may still have room for improvement. For example, by changing the order in which the macros are considered. Therefore, we conclude that heuristics may be a promising way to solve the macro placement problem.

References

Brenner, Ulrich, Markus Struzyna, and Jens Vygen (Sept. 2008). “BonnPlace: Placement of Leading-Edge Chips by Advanced Combinatorial Algorithms". In: *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* 27.9, pp. 1607–1620. ISSN: 0278-0070. DOI: 10.1109/TCAD.2008.927674.

Demaine, Erik D. and Martin L. Demaine (2007). "Jigsaw puzzles, edge matching, and polyomino packing: connections and complexity". In: *Graphs Combin.* 23.suppl. 1, pp. 195–208. ISSN: 0911-0119. DOI: 10.1007/s00373-007-0713-4. URL: https://doi.org/10.1007/s00373-007-0713-4.

Silvanus, Jannik (2019). "Improved Cardinality Bounds for Rectangle Packing Representations". PhD thesis. Rheinischen Friedrich-Wilhelms-Universität Bonn.

A Joint Data- and Model-Driven Approach Simulating the Behaviour of a Walkalong Glider

Dieuwertje Alblas[1], Mark van den Bosch[2], Lucas Jansen Klomp[3], Vivi Rottschäfer[4], Len Spek[5], Felix Schwenninger[6], Lotte Weedage[7] and Sjanne Zeijlemaker[8]

Abstract

Properly flying a walkalong glider is a challenging control problem, involving many internal and external influences. We have shed some light on this problem from both a model-driven and data-driven viewpoint. We construct a mathematical model, based on well-established models from literature. This model is used to derive an optimal paddle angle, as well as an optimal distance between the paddle and the glider for steady level flight. Moreover, we derive parameters of the glider during uniform circular flight, as well as their relation to movement of the paddle from our model. As the glider is in practice very sensitive to small

[1]University of Twente, The Netherlands
[2]Leiden University, The Netherlands
[3]University of Twente, The Netherlands
[4]Leiden University, The Netherlands
[5]University of Twente, The Netherlands
[6]University of Twente, The Netherlands
[7]University of Twente, The Netherlands
[8]Eindhoven University, The Netherlands

disturbances in the airflow, we have developed a data-driven model that tracks the status of the glider using video data, recorded by a camera mounted on the paddle. We use image processing and deep learning to automatically detect glider position with respect to the paddle and its yaw. Information from mathematical and data-driven models could be jointly used as an input for a control algorithm on the paddle, allowing for automatic operation of the walkalong glider.

KEYWORDS: walkalong glider, plane, flight dynamics, image analysis, control problem

5.1 Introduction

The artist Zoro Feigl works on a new art project in which a small glider flies on the wake of a moving board, controlled by a robot arm. The project is based on an idea of Joseph E. Grant in 1955, called the *walkalong glider* Grant (1955). This walkalong glider is controlled by a board, called the *paddle* (Figure 5.1). The glider flies on the airwave

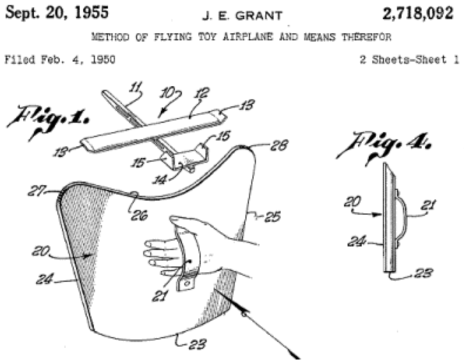


Figure 5.1: Sketch of the walkalong glider idea by Grant (1955).

that is caused by movement of the paddle, and by adjusting the angle of the paddle, the glider's trajectory can be influenced. See Feigl (2021) for a video clip of a walkalong glider demonstration.

5.1.1 Robot arm

The glider in the art project will be controlled by a robot arm with a transparent board mounted on its tip. The model that will be used is the FANUC M-710iC/12L, shown in Figure 5.2. This robot has six axes, which allows for a large range of motion. Figure 5.3 shows its length when fully extended (3.123 meters) and a side view of its reach. The flight path of the glider should always stay within this range. Additionally, we must stay above ground level, as the robot arm will be mounted to the floor. The wrist of the arm can move along three axes: the 'hand' can bend with respect to the arm and the wrist and tip (where the board is mounted) can be rotated 400 and 720 degrees respectively. This allows us to adjust the position and angle of the board relative to the glider.

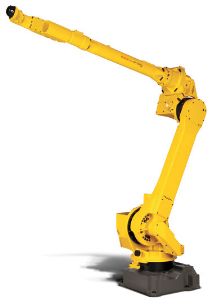


Figure 5.2: The FANUC M-710iC/12L robot arm which is planned to be used to control the glider FANUC Benelux BV (2022).

An important restriction of this robot arm is that it can only rotate up to 360 degrees around its center. Hence, we cannot create a circular flight path by spinning the arm indefinitely.

5.1.2 Goal

The objective of this project is to explore how control on the glider's trajectory can be automated. Instead of a human operating the paddle, it is mounted on a robot arm that should autonomously keep the glider

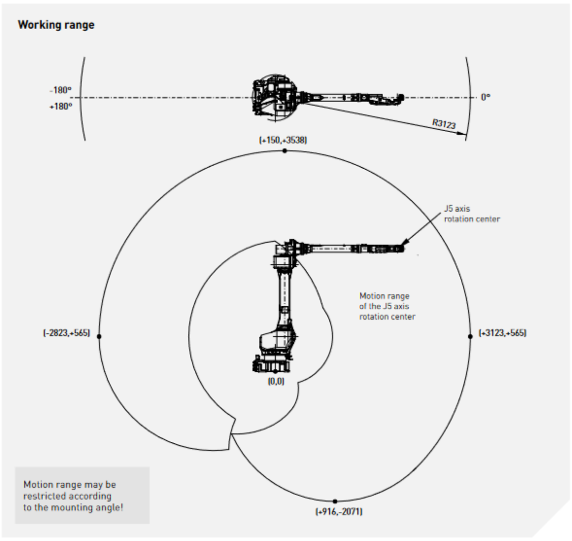

Working range
-180°
+180°
0°
R3123
(+150,+3538)
J5 axis rotation center
Motion range of the J5 axis rotation center
(-2823,+565)
(+3123,+565)
(0,0)
Motion range may be restricted according to the mounting angle!
(+916,-2071)

Figure 5.3: Range of motion of the FANUC M-710iC/12L robot arm FANUC Benelux BV (2022, datasheet).

in the air. This objective requires a control strategy for the robot arm that keeps the glider in the air, even under perturbations in the airflow (such as movement of observers). As a first step towards this objective, we investigated how to model the glider, which data is required to control the glider, and how this data should be processed.

5.1.3 Approach

We have split up this problem into two parts: a model-driven and a data-driven part. These two approaches both focus on a different aspect of the total control strategy: based on the model, we should be able to determine a global flight-path and predict where the glider will be at each moment in time. Additionally, we use a data-driven approach to make small corrections to the flight path and account for disturbances caused by external influences.

This report is set up as follows: first, we give an overview of important models from literature in Section 5.2, and investigate the effect of the paddle on the glider. Subsequently, in Section 5.3, we show the behaviour of the glider based on data analysis and show some key variables that are important to control in order to be able to keep the glider flying. We end this report with some recommendations and an outlook for further research in Section 5.4.

5.2 Model-driven approach

In this section, we consider two different approaches to calculate the flight path of the glider. Firstly, we study a more elaborate model, representing the glider by a point mass. First we introduce this model in a setting with no wind and then extend it to a scenario where wind is present. Subsequently, we simplify this model to approximate the glider angle and turning radius. This is explained in Section 5.2.3.

Both versions of the model rely on characteristics of the glider, as well as known equations from the literature. We begin this section with an overview of all parameters involved.

5.2.1 Definitions and formulas

Our models use several well-known formulas and physical constants from the literature. In this section, we give a brief overview. For a more detailed description, we refer to Stengel (2004, Chapter 1).

Table 5.1 gives a schematic overview of the symbols used in the glider model, their meaning and their value or formula. Some are known constants, such as the density of air at room temperature. However, most parameters depend on the particular glider that we use.

Our glider is based on the Baby Bug model from Harrison Science-toyMaker (2022). It is made out of thin EMS foam with an aluminum weight at the nose. This glider has a total wing span of 20cm and a surface area of 12.2cm^2. Figure 5.4 shows the layout of a single wing; its mass is 0.147g.

Some other parameters from Table 5.1 depend on the characteristics of the glider, and cannot be measured directly; for example the Oswald efficiency number and drag coefficient at zero lift. For these parameters,

Symbol	Description	Value/formula
D	Drag force	$\frac{1}{2}C_D\rho V^2 S$
L	Lift force	$\frac{1}{2}C_L\rho V^2 S$
C_L	Lift coefficient	$C_{L_\alpha}\alpha$
C_{L_α}	Lift-slope derivative	$\frac{\pi AR}{1+\sqrt{1+(AR/2)^2}}$
C_D	Drag coefficient	$C_{D_0}+\varepsilon C_L^2$
C_{D_0}	Drag coefficient at zero lift	0.02
ϵ	Induced-drag factor	$1/(\pi ARe)$
AR	Aspect ratio	b^2/S
e	Oswald efficiency number	0.9
m	Mass of the glider	$1.47\cdot 10^{-4}$kg
g	Gravitational acceleration	9.807 m/s^2
S	Surface area of the glider	0.0122m^2
ρ	Density of air	1.225 kg/m^3
b	Total wing span of the glider	0.2m
V	Relative air speed	See Section 5.2.2
α	Angle of attack	

Table 5.1: A description and its corresponding formula or value of the symbols used in the models.

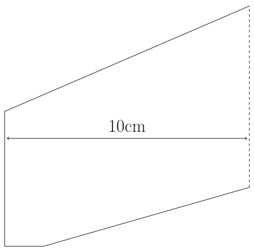


Figure 5.4: A left wing of the Baby Bug glider. The dashed line marks the center of the glider.

we used values for a similar paper airplane in Stengel (2004, Chapter 1).

5.2.2 Glider one point mass model

We first propose to model the glider as a one-point mass in space, similar to Etkin (2005) and Hull (2007). Figure 5.5 illustrates a free-body diagram of the forces acting on the centre of mass; note that the depiction of the glider is for illustrative purposes only.

The velocity of the glider relative to the ground ($\mathbf{U}$) will be written as the sum

$$\mathbf{U} = \mathbf{V} + \mathbf{W}, \tag{5.1}$$

with $\mathbf{V}$ the velocity of the airplane relative to the atmosphere and $\mathbf{W}$ the velocity of the atmosphere relative to the ground. Since the drag and lift are functions of the airspeed $V = \|\mathbf{V}\|$, decomposing the glider velocity $\mathbf{U}$ is more convenient. The lift and drag are respectively defined as:

$$L = \frac{1}{2}\rho V^2 S C_L \quad \text{and} \quad D = \frac{1}{2}\rho V^2 S C_D, \tag{5.2}$$

where the constants $C_L = C_L(\alpha)$ and $C_D = C_D(\alpha)$ depend on the angle of attack α. For our intents and purposes, we may assume there are no side forces acting on the glider. Differently put, the sideslip angle is assumed to be zero. For more information, we refer to Beeler, Moerder, and Cox (2003).

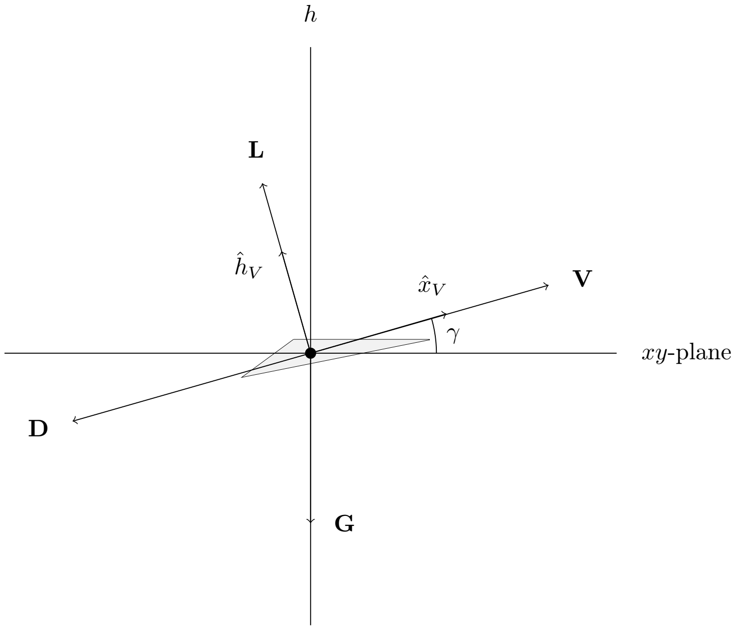


Figure 5.5: A free-body diagram of a glider with no ambient winds. The forces acting on the glider are assumed to be the lift force vector $\mathbf{L}$, the drag force vector $\mathbf{D}$ and the gravity force vector $\mathbf{G} = (0, 0, -mg)^T$. The airspeed vector $\mathbf{V}$ is parallel to the vector $\mathbf{D}$ and perpendicular to the vector $\mathbf{L}$, and $\hat{x}_V$ is defined to be the unit vector in the direction of $\mathbf{V}$.

In this section, we will make use of the following two reference frames. Assuming a flat[9], non-rotating Earth, we obtain an inertial reference frame defined with axes aligned with the unit vector set

$$\hat{U}_I = \left(\begin{array}{c|c|c} \hat{x} & \hat{y} & \hat{h} \end{array} \right) = \left(\begin{array}{c|c|c} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{array} \right). \tag{5.3}$$

[9]The flat earth assumption suffices for our applications, but it is not necessary. See Etkin (2005) and Hull (2007) for flight models over a spherical Earth.

We refer to this inertial system as the "ground axes frame". A sequence of three simple rotations relates the inertial reference frame to a non-inertial, rotating reference frame whose axes are defined such that $\hat{U}_V = \begin{pmatrix} \hat{x}_V & \hat{y}_V & \hat{h}_V \end{pmatrix}$ with $\hat{x}_V = \mathbf{V}/V$. We call this the "wind axes frame"; in the absence of wind, the latter coincides with the "velocity frame". In particular, we define

$$\hat{U}_V = \begin{pmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\gamma & 0 & -\sin\gamma \\ 0 & 1 & 0 \\ \sin\gamma & 0 & \cos\gamma \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{pmatrix} \tag{5.4}$$

$$= \begin{pmatrix} \cos\gamma\cos\psi & -\cos\phi\sin\psi + \sin\phi\sin\gamma\cos\psi & -\sin\phi\sin\psi - \cos\phi\sin\gamma\cos\psi \\ \cos\gamma\sin\psi & \cos\phi\cos\psi + \sin\phi\sin\gamma\sin\psi & \sin\phi\cos\psi - \cos\phi\sin\gamma\sin\psi \\ \sin\gamma & -\sin\phi\cos\gamma & \cos\phi\cos\gamma \end{pmatrix}, \tag{5.5}$$

where ψ is the velocity yaw (heading angle), γ the velocity pitch (flight-path angle), and ϕ the roll (banking angle). Observe that $\hat{x}_V = \hat{U}_V \hat{x}$, $\hat{y}_V = \hat{U}_V \hat{y}$, and $\hat{h}_V = \hat{U}_V \hat{h}$.

Before we continue, it is worth pointing out that the orientation of our coordinate systems deviates from the convention in flight mechanics. Often the x, y, and z-axis are interpreted as north, east, and downwards, respectively, hence one defines the altitude then as $h = -z$, see, e.g., Beeler, Moerder, and Cox (2003) and Hull (2007). We prefer to work with x, y, and h only, which results into a different rotation matrix $\hat{U}_V$.

Flight with the absence of wind

Let us assume our glider experiences no moving atmosphere. In the three dimensional setting, a standard form for the equations of motion

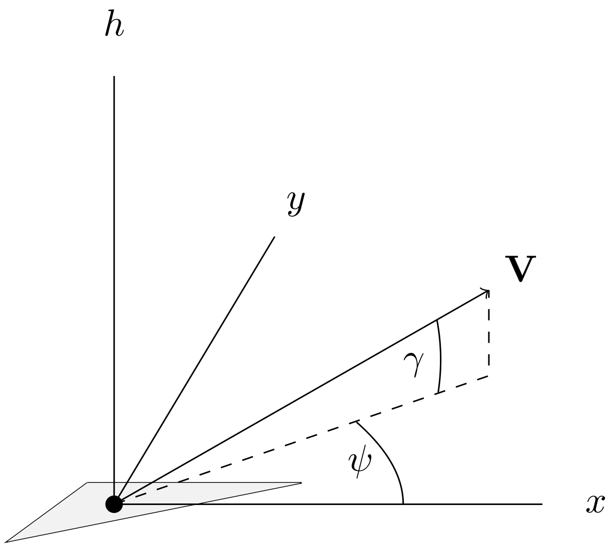


Figure 5.6: A three-dimensional sketch of the ground axes frame. Note that the banking angle ϕ is the only angle not being depicted in the diagram; this particular angle depends on how much the plane is rolling about the airspeed vector $\mathbf{V}$.

for a point mass glider is

$$\dot{V} = -\frac{D}{m} - g\sin\gamma \tag{5.6}$$

$$\dot{\gamma} = \frac{L\cos\phi}{mV} - \frac{g}{V}\cos\gamma \tag{5.7}$$

$$\dot{\psi} = \frac{L\sin\phi}{mV\cos\gamma} \tag{5.8}$$

$$\dot{x} = V\cos\psi\cos\gamma \tag{5.9}$$

$$\dot{y} = V\sin\psi\cos\gamma \tag{5.10}$$

$$\dot{h} = V\sin\gamma. \tag{5.11}$$

Equations (5.6)–(5.8) are the so-called dynamics equations. The kinematic equations for the velocity vector $\mathbf{V} = \begin{pmatrix}\dot{x} & \dot{y} & \dot{h}\end{pmatrix}$ are given by (5.9)–(5.11). Relatively more advanced equations of motion, taking either side or thrust forces into account as well, can be found in Beeler, Moerder, and Cox (2003) and Hull (2007), respectively.

Remark 4. Observe that in the equations above, the banking angle ϕ is presumed to be a control parameter (which may be time dependent). Usually, in the study of flight mechanics, this is a feasible presumption, because a pilot is able to adjust this angle by simply adjusting the wing flaps of the airplane.

It is clear that we cannot control this as easily in our application. Nevertheless, observe that the changes in $\phi = \phi(t)$ are caused by the paddle only and that $\phi = 0$ holds when the paddle is absent. In particular, under additional assumptions, we are able to relate the banking angle ϕ with the angles made by the paddle, see Subsections 5.2.3 and 5.2.3. Alternatively, one is able to obtain an approximation of ϕ by looking at the data (see recommendations and outlook). Different model assumptions, such as saying that ϕ is proportional to $\dot{\psi}$ at an earlier time instant (hence, a delay differential equation), are thought to be possible and numerically tractable.

So, from now we just assume that—indeed—we are able to control the angle ϕ.

A derivation of the equations of motion above can be found in Weitz (2015). Although elementary, we show its derivation for the sake of completeness and because the several observations made are useful for the flight model with wind. By recalling Figure 5.5, we see that the resultant external force becomes

$$\begin{aligned}\mathbf{F} &= \mathbf{L} + \mathbf{D} + \mathbf{G} && (5.12)\\ &= [-D - mg\sin\gamma]\hat{x}_V + mg\cos\gamma\sin\phi\hat{y}_V + [L - mg\cos\gamma\cos\phi]\hat{h}_V. && (5.13)\end{aligned}$$

On the other hand, since $\mathbf{V} = V\hat{x}_V$ holds, we obtain that the acceleration relative to the ground is given by

$$\mathbf{a} = \frac{d\mathbf{V}}{dt} = \dot{V}\hat{x}_V + V\frac{d\hat{x}_V}{dt}. \tag{5.14}$$

Expressing the time derivative of $\hat{x}_V$ within the coordinates of the rotating reference frame, gives us

$$\frac{d\hat{x}_V}{dt} = -[\dot{\gamma}\sin\phi - \dot{\psi}\cos\gamma\cos\phi]\hat{y}_V + [\dot{\gamma}\cos\phi + \dot{\psi}\cos\gamma\sin\phi]\hat{h}_V. \quad (5.15)$$

This relationship is easily verified; a direct calculation uses the angular velocities and invokes the transport theorem, see, e.g., Weitz (2015). Substituting equation (5.15) in equation (5.14) yields

$$\mathbf{a} = \dot{V}\hat{x}_V - V[\dot{\gamma}\sin\phi - \dot{\psi}\cos\gamma\cos\phi]\hat{y}_V + V[\dot{\gamma}\cos\phi + \dot{\psi}\cos\gamma\sin\phi]\hat{h}_V. \quad (5.16)$$

Using the second law of Newton, equation (5.13), and equation (5.16), we get the following dynamics equations:

$$m\dot{V} = -D - mg\sin\gamma, \quad (5.17)$$
$$mV[\dot{\gamma}\sin\phi - \dot{\psi}\cos\gamma\cos\phi] = -mg\cos\gamma\sin\phi, \quad (5.18)$$
$$mV[\dot{\gamma}\cos\phi + \dot{\psi}\cos\gamma\sin\phi] = L - mg\cos\gamma\cos\phi. \quad (5.19)$$

It is obvious (5.17) is equivalent to (5.6). The following observation is extremely useful: we are able to decouple the other two equations in such ways that we will get the first order differential equations (5.7) and (5.8).

When we multiply equation (5.18) by $\sin\phi$ and equation (5.19) by $\cos\phi$ and add the resulting two expressions, we obtain

$$mV\dot{\gamma} = L\cos\phi - mg\cos\gamma. \quad (5.20)$$

Similarly, when we multiply equation (5.18) by $\cos\phi$ and equation (5.19) by $\sin\phi$ and subtract the resulting two expressions from one another, we find

$$mV\cos\gamma\dot{\psi} = L\sin\phi. \quad (5.21)$$

This motivates the equations of motion (5.6)–(5.11).

Flight with the presence of wind

For the two dimensional setting, we refer to Chapter 2 of Hull (2007) for an analogous model to the one below (with thrust) which includes a comprehensive derivation as well. In line with their approach, we derive our three dimensional model. Also, note the similarities with Weitz (2015), yet we will follow slightly different assumptions.

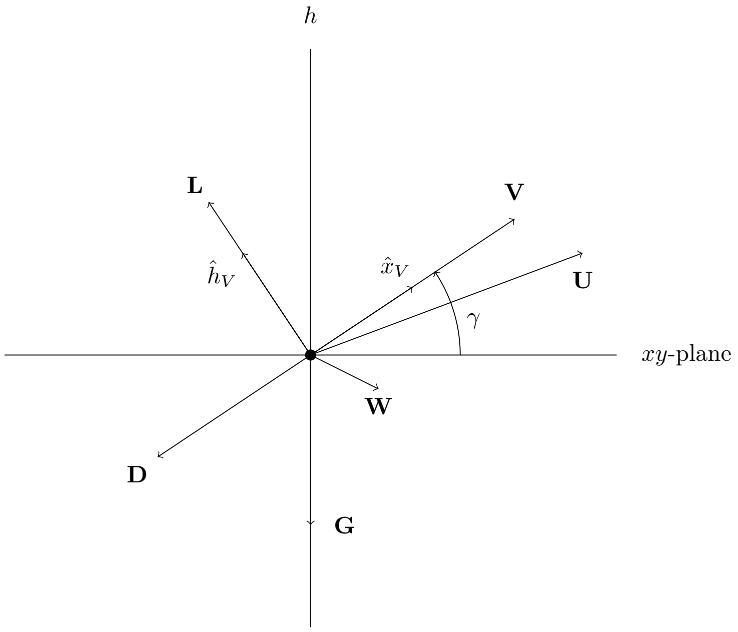


Figure 5.7: A force diagram in the case of a moving atmosphere $\mathbf{W}$.

By writing $\mathbf{W} = (W_x, W_y, W_z)^T$, we notice that the kinematic equations become

$$\dot{x} = V \cos\gamma \cos\psi + W_x \tag{5.22}$$
$$\dot{y} = V \cos\gamma \sin\psi + W_y \tag{5.23}$$
$$\dot{z} = V \sin\gamma + W_z. \tag{5.24}$$

In the scenario where the wind velocity is constant, the dynamic equations (5.6)–(5.8) do not change. In other words, the equations of motion of the glider are then given by the coupled differential equations (5.6)–(5.8) together with (5.22)–(5.24).

Nevertheless, given that the wind field $\mathbf{W}$ changes over time (which is the case for our application), we need to consider the time derivative of $\mathbf{W}$ into the dynamics equations. The acceleration relative to the ground is given by

$$\mathbf{a} = \frac{d\mathbf{U}}{dt} = \frac{d\mathbf{V}}{dt} + \frac{d\mathbf{W}}{dt} = \frac{d\mathbf{V}}{dt} + (\dot{W}_x, \dot{W}_y, \dot{W}_z)^T. \tag{5.25}$$

As before, we would like to express $\mathbf{a}$ in terms of the wind axes frame (now a moving atmosphere is present). Recall that

$$\frac{d\mathbf{V}}{dt} = \dot{V}\hat{x}_V - V[\dot{\gamma}\sin\phi - \dot{\psi}\cos\gamma\cos\phi]\hat{y}_V + V[\dot{\gamma}\cos\phi + \dot{\psi}\cos\gamma\sin\phi]\hat{h}_V. \tag{5.26}$$

Consequently, in the wind axes frame we have

$$\begin{aligned}\mathbf{a} &= \frac{d\mathbf{V}}{dt} + \hat{U}_V(\dot{W}_x, \dot{W}_y, \dot{W}_z)^T & (5.27)\\ &= \big(\dot{V} + \dot{W}_x\cos\gamma\cos\psi + \dot{W}_y\cos\gamma\sin\psi + \dot{W}_z\sin\gamma\big)\hat{x}_V + \\ &\qquad \big(-V[\dot{\gamma}\sin\phi - \dot{\psi}\cos\gamma\cos\phi] + \dot{W}_x\hat{U}_{21} + \dot{W}_y\hat{U}_{22} + \dot{W}_z\hat{U}_{23}\big)\hat{y}_V + & (5.28)\\ &\qquad \big(V[\dot{\gamma}\cos\phi + \dot{\psi}\cos\gamma\sin\phi] + \dot{W}_x\hat{U}_{31} + \dot{W}_y\hat{U}_{32} + \dot{W}_z\hat{U}_{33}\big)\hat{z}_V,\end{aligned}$$

where $\hat{U}_{ij} := (\hat{U}_V)_{ij}$. Once again, using the second law of Newton, equation (5.13), and equation (5.28), we get the following dynamics equations:

$$\begin{aligned} m\dot{V} &= -D - mg\sin\gamma + & (5.29)\\ &\quad - m\left(\dot{W}_x\cos\gamma\cos\psi + \dot{W}_y\cos\gamma\sin\psi + \dot{W}_z\sin\gamma\right), & (5.30)\\ mV[\dot{\gamma}\sin\phi - \dot{\psi}\cos\gamma\cos\phi] &= -mg\cos\gamma\sin\phi + m\left(\dot{W}_x\hat{U}_{21} + \dot{W}_y\hat{U}_{22} + \dot{W}_z\hat{U}_{23}\right), & (5.31)\\ mV[\dot{\gamma}\cos\phi + \dot{\psi}\cos\gamma\sin\phi] &= L - mg\cos\gamma\cos\phi - m\left(\dot{W}_x\hat{U}_{31} + \dot{W}_y\hat{U}_{32} + \dot{W}_z\hat{U}_{33}\right). & (5.32)\end{aligned}$$

Decoupling equations (5.31) and (5.32) as before, results into the dynamics equations:

$$\dot{V} = -\frac{D}{m} - g\sin\gamma - \dot{W}_x\cos\gamma\cos\psi - \dot{W}_y\cos\gamma\sin\psi - \dot{W}_z\sin\gamma \tag{5.33}$$
$$\dot{\gamma} = \frac{L\cos\phi}{mV} - \frac{g}{V}\cos\gamma + \frac{1}{V}\big(\dot{W}_x\sin\gamma\cos\psi + \dot{W}_y\sin\gamma\sin\psi - \dot{W}_z\cos\gamma\big) \tag{5.34}$$
$$\dot{\psi} = \frac{L\sin\phi}{mV\cos\gamma} + \frac{1}{V\cos\gamma}\big(\dot{W}_x\sin\psi - \dot{W}_y\cos\psi\big). \tag{5.35}$$

In conclusion, the equations of motion of the glider with a general wind field is given by

$$\dot{V} = -\frac{D}{m} - g\sin\gamma - \dot{W}_x\cos\gamma\cos\psi - \dot{W}_y\cos\gamma\sin\psi - \dot{W}_z\sin\gamma \tag{5.36}$$
$$\dot{\gamma} = \frac{L\cos\phi}{mV} - \frac{g}{V}\cos\gamma + \frac{1}{V}\big(\dot{W}_x\sin\gamma\cos\psi + \dot{W}_y\sin\gamma\sin\psi - \dot{W}_z\cos\gamma\big) \tag{5.37}$$
$$\dot{\psi} = \frac{L\sin\phi}{mV\cos\gamma} + \frac{1}{V\cos\gamma}\big(\dot{W}_x\sin\psi - \dot{W}_y\cos\psi\big). \tag{5.38}$$
$$\dot{x} = V\cos\psi\cos\gamma + W_x \tag{5.39}$$
$$\dot{y} = V\sin\psi\cos\gamma + W_y \tag{5.40}$$
$$\dot{h} = V\sin\gamma + W_z, \tag{5.41}$$

where we recall

$$L = \frac{1}{2}\rho V^2 SC_L \quad \text{and} \quad D = \frac{1}{2}\rho V^2 SC_D, \tag{5.42}$$

Apart from different angles orientations, a major difference between the model above compared to the model in Weitz (2015) is that we do not assume the wind direction is in the xy-plane. For our intents and purposes, we require W_z and $\dot{W}_z$. Additionally, one usually assumes that W_x, W_y, and W_z are functions of t, x, y, and h, and computes the time derivatives with the chain rule, see also Hull (2007). We will not do this, since W_x, W_y, W_z and their time derivatives are supposed to be obtained from the data or/and by means of computational fluid dynamics simulations (see recommendations and outlook).

5.2.3 A simplified flight model

The model in the previous section is useful for doing detailed simulations on the glider when coupled to a computational fluid dynamics simulator of the airflow generated by the paddle. In this section, however, we take an alternative approach by simplifying the models above. These can then be used to approximate several quantities like the glide angle and the turning radius. This section can be read independently from the section above.

Flying straight ahead

Firstly, we look at the case where the glider flies straight, so that the y-component can be ignored. Like earlier, we let $\mathbf{U} = (U_x, U_z)$ denote the velocity of the glider with respect to the ground and $\mathbf{W} = (W_x, W_z)$ the velocity of the wind generated by the paddle. The relative airspeed of the glider with respect to the wind is given by $\mathbf{V} = \mathbf{U} - \mathbf{W}$. The angle of the vector $\mathbf{V}$ relative to the ground (i.e., with the x-axis), which we denote with γ, is given by

$$\arctan\left(\frac{V_z}{V_x}\right). \tag{5.43}$$

The total aerodynamic force acting on the glider is a combination of the drag D in the opposite direction of $\mathbf{V}$ and the lift L perpendicular to $\mathbf{V}$. By introducing the unit vectors $e_{\mathbf{V}} = (\cos\gamma, \sin\gamma)$ and $e_{\perp\mathbf{V}} = (-\sin\gamma, \cos\gamma)$, we can mathematically express the above as

$$\mathbf{F}_{aero} = Le_{\perp\mathbf{V}} - De_{\mathbf{V}} = \frac{1}{2}\rho V^2 S(C_L e_{\perp\mathbf{V}} - C_D e_{\mathbf{V}}). \tag{5.44}$$

where $V = \|\mathbf{V}\|$. For the lift and drag coefficients, C_L and C_D, we use a standard formulation like in Beeler, Moerder, and Cox (2003):

$$C_D = C_{D_0} + \varepsilon C_L^2, \quad C_L = C_{L_\alpha}\alpha, \tag{5.45}$$

where

$$\varepsilon = \frac{1}{\pi ARe}, \quad C_{L_\alpha} = \frac{\pi AR}{1+\sqrt{1+(AR/2)^2}}. \tag{5.46}$$

Note that these equations are only valid for a small angle of attack $\alpha = \chi - \gamma$, i.e., the angle between the pitch angle of the wing χ and the airspeed vector $\mathbf{V}$. Subsequently, taking the aerodynamic force $\mathbf{F}_{aero}$ and the gravity, $\mathbf{F}_g = -mge_z$, into account results into the following set of equations of motion:

$$\dot{U}_x = -\frac{1}{2m}\rho V^2 S(C_L \sin\gamma + C_D \cos\gamma), \tag{5.47}$$

$$\dot{U}_z = -g + \frac{1}{2m}\rho V^2 S(C_L \cos\gamma - C_D \sin\gamma), \tag{5.48}$$

$$\dot{x} = U_x, \tag{5.49}$$

$$\dot{z} = U_z. \tag{5.50}$$

The equations above can be made more explicit by observing that $V^2 = (U_x - W_x)^2 + (U_z - W_z)^2$ holds, which reduces to $V^2 = U_x^2 + U_z^2$ in absence of the paddle.

For these equations we will now analytically investigate the equilibrium situation $\dot{U}_x = \dot{U}_z = 0$. In words, we solve for a flight trajectory with constant velocity. Notice that setting $\dot{U}_x = 0$ yields the equality

$$C_L \sin\gamma + C_D \cos\gamma = C_{L_\alpha}(\chi-\gamma)\sin\gamma + (C_{D_0} + \varepsilon C_{L_\alpha}^2(\chi-\gamma)^2)\cos\gamma = 0, \tag{5.51}$$

which we can solve for γ. Using $\chi \approx 0$ as the value for the pitch angle and applying the small angle formulae $\sin\gamma \approx \gamma$ and $\cos\gamma \approx 1$ gives us the following approximation for γ at equilibrium:

$$\gamma_{\text{eq}} \approx -\sqrt{\frac{C_{D_0}}{C_{L_\alpha} - C_{L_\alpha}^2 \varepsilon}}. \tag{5.52}$$

The minus sign is due to the fact that $\mathbf{V}$ is pointing downwards. Consequently, setting $\dot{U}_z = 0$ gives us a value for the airspeed at equilibrium, namely

$$V_{\text{eq}} = \sqrt{\frac{2mg}{\rho S(C_L \cos\gamma_{\text{eq}} - C_D \sin\gamma_{\text{eq}})}}. \tag{5.53}$$

Using the values in Table 5.1, we find

$$\gamma_{\text{eq}} \approx -5.5^\circ \quad \text{and} \quad V_{\text{eq}} \approx 0.75\,\text{m/s}. \tag{5.54}$$

Note that these values hold regardless of the wind field $\mathbf{W}$, as long as the wind field is not accelerating. In the scenario where wind is absent, γ is the glide angle and V the velocity of the glider. This implies that without the paddle—and assuming that the glider is not going to accelerate—the glider falls down with an approximate speed of 0.75 m/s and under an angle of 5.5°. In particular, we can decompose V_{eq} as

$$V_{x,\text{eq}} \approx 0.74\,\text{m/s} \quad \text{and} \quad V_{z,\text{eq}} \approx 0.07\,\text{m/s}. \tag{5.55}$$

This equilibrium is, in fact, stable (which is easily verified with standard techniques). If we simulate the equations of motion (5.47)–(5.50) without wind (see Figure 5.8), we see that the glider's angle and velocity converge to the found values in (5.54) and (5.55).

A simplified wind model

A full treatment of the airflow induced by the paddle requires a quite complex computational fluid dynamics (CFD) simulation, like in Figure 5.9 from Sweden International Physicist's Tournament (2017). To circumvent this heavy machinery, we introduce some simplifications, enabling us to do some elementary calculations.

1. We assume that we can perfectly match the speed of the paddle with the horizontal speed of the glider U_x and we keep it at a fixed angle θ and a fixed height above the ground.

2. We assume that we can see the airflow as two dimensional, so it only has a component in the x and z direction.

3. We model only the airflow above the boundary layer, a small region where the friction with the paddle has a large effect.

4. In the co-moving frame of reference of the paddle, there is an incoming horizontal airflow with speed U_x. We assume that the paddle only deflects the airflow and that the speed of the airflow is conserved.

5. We assume that at the edge of the boundary the airflow is parallel with the paddle and that the vertical component decays expo-

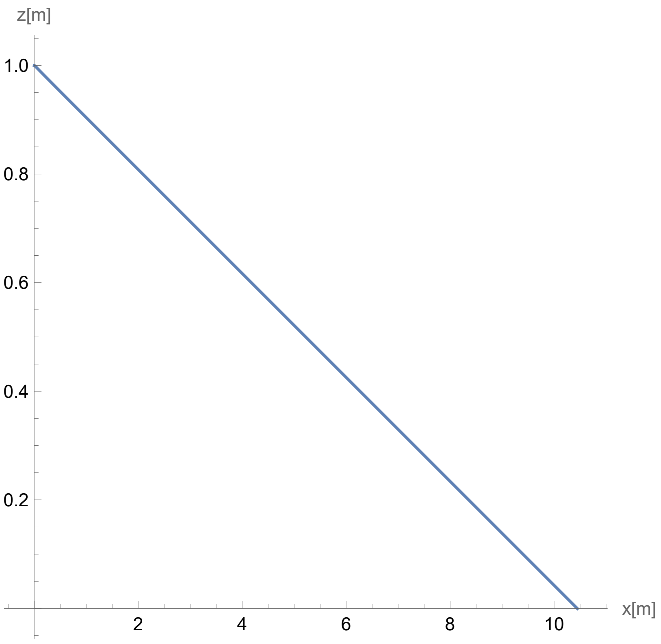


Figure 5.8: A simulation of the equations of motion (5.47)–(5.50) with values of Table 5.1 and initial conditions $x = 0\,\mathrm{m}, z = 1\,\mathrm{m}, U_x = 0.75\,\mathrm{m/s}, U_z = 0\,\mathrm{m/s}$. Observe that the solution quickly converges to the equilibrium state; see equations (5.54) and (5.55).

nentially with a factor c_w and the perpendicular distance to the paddle.

Now, let z denote the height above the boundary layer and consider the perpendicular distance to the paddle given by $\frac{z}{\cos\theta}$. To transform back to the ground reference frame, we are required to add the velocity of the paddle $(U_x, 0)$. With the assumptions above, we use the following equation to model the wind from the ground perspective:

$$\mathbf{W}(U_x, z, \theta) = U_x(1 - \sqrt{1 - e^{-\frac{2c_w z}{\cos\theta}} \sin^2\theta}, e^{-\frac{c_w z}{\cos\theta}} \sin\theta). \tag{5.56}$$

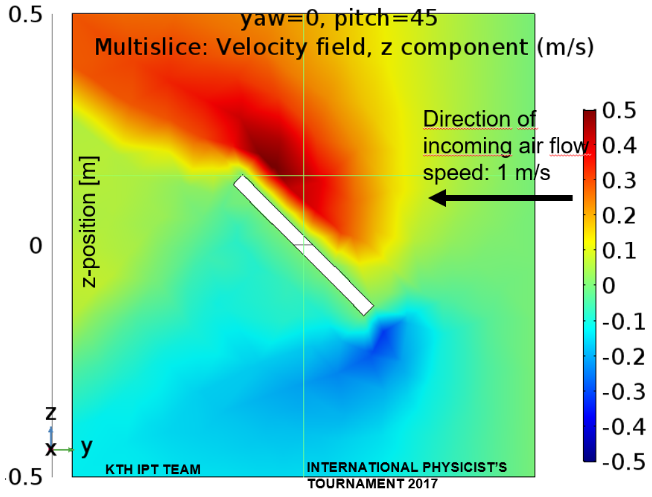


Figure 5.9: A CFD simulation of the airflow induced by the paddle from Sweden International Physicist's Tournament (2017).

Remark 5. The model in (5.56) is an ad hoc description and simplication of the wind generated by the paddle. We have chosen this specific formulation, but we would like to emphasise that we do not exclude other possibilities. This particular choice for the wind field also enabled us to obtain nice closed forms for U_x and z, see (5.58) and (5.59).

Remark 6. Assuming that the wind direction is two dimensional, the paddle length is effectively infinite. For a relatively small paddle, this assumption could be problematic. Moreover, we do not treat the boundary layer, as we assume that the glider is unable to fly here, so it does not need to be modelled. The exponential decay $e^{-\frac{c_w z}{\cos\theta}}$ matches the properties observed from the CFD simulations (Figure 5.9): for $z = 0$ this factor is 1, for $z \to \infty$ it vanishes, and the gradient of the speed difference is proportional with the distance. Ultimately, the choice $\frac{z}{\cos\theta}$, instead of, e.g., $z\cos\theta$, was to better reflect the fact that the wind velocity substantially decreases as θ increases.

We can now use the values of constant flight of the previous subsection, i.e.,

$$\gamma_{\mathrm{eq}} = -5.5^\circ \quad \text{and} \quad V_{\mathrm{eq}} = 0.75\,\mathrm{m/s}, \tag{5.57}$$

to solve for the case where the glider keeps its altitude, i.e., $U_z = 0$. From the equation $\mathbf{V} = \mathbf{U} - \mathbf{W}$ and the wind model description (5.56), we deduce

$$U_x = V_{\mathrm{eq}} \quad \text{and} \quad \sin(-\gamma_{\mathrm{eq}}) = e^{-\frac{c_w z}{\cos\theta}} \sin\theta. \tag{5.58}$$

Using that the height is postive, i.e., $z > 0$, we obtain $\theta > -\gamma_{\mathrm{eq}} = 5.5^\circ$. By analysing Figure 5.9, we estimate $c_w \approx 3\,\mathrm{m}^{-1}$. An estimation for c_w suffices, since it only defines the length scale of z. We can solve for the height of z:

$$z = -\frac{\cos\theta}{c_w} \log\left(\frac{\sin(-\gamma_{\mathrm{eq}})}{\sin\theta}\right). \tag{5.59}$$

We can plot the height z as a function of the paddle angle θ, see Figure 5.10.

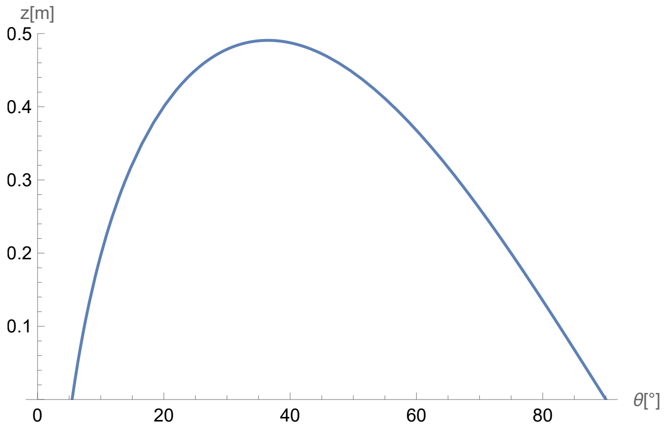


Figure 5.10: The height above the paddle z for a glider with constant speed (5.59) as a function of the paddle angle θ with the parameters $c_w = 3\,\mathrm{m}^{-1}$ and $\gamma_{\mathrm{eq}} = -5.5^\circ$.

The optimal paddle angle is when the stable height above the paddle z is maximal. In this case if there is a disturbance in the airflow, there is more space to recover before the glider enters the turbulent boundary layer and is unable to fly.

Differentiating the function in (5.59) and setting it equal to zero, we find that for $\gamma_{\text{eq}} = -5.5°$ the optimal angle is $\theta = 36°$. For different glider angles see Figure 5.11.

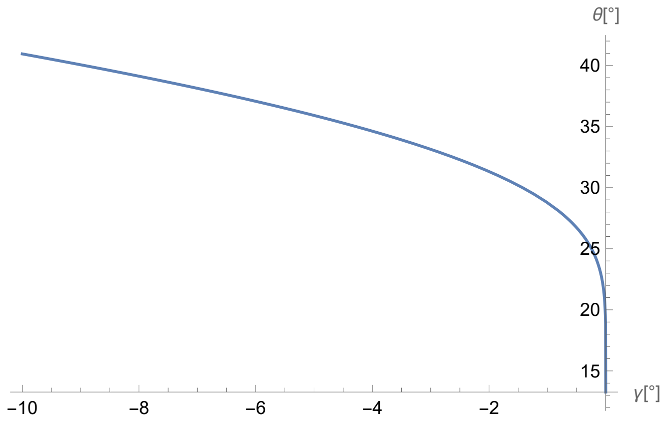


Figure 5.11: For each $\gamma = \gamma_{\text{eq}}$, we plot the optimal paddle angle θ, i.e., the paddle angle for which the function in equation (5.59) is maximal.

Turning radius of the glider

We can make the glider turn by just rotating the paddle. Indeed, rotating the paddle causes a difference in the airflow at each wing, initiating a different lift force on each wing, see Figure 5.12. The imbalance in lift induces a torque on the glider, which makes the glider bank.

To compute the turning radius, we assume that the glider is flying a perfect circle of radius R with constant speed U_x. In this section we consider the vehicle reference frame to be our coordinate system, so we write x for the tangential direction of the circular trajectory—positive in the direction of movement—and write y for the radial direction—positive in the inward direction for a left turn. For uniform circular

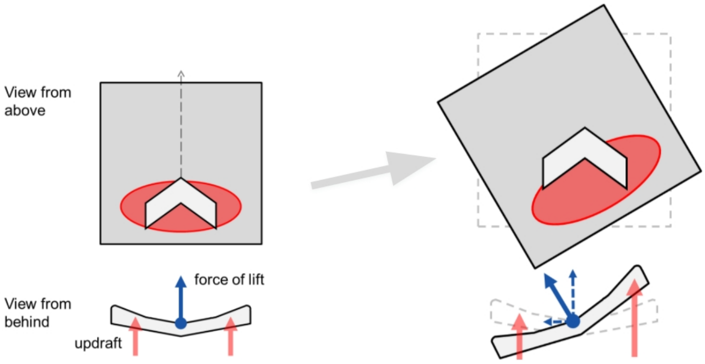


Figure 5.12: The effect of rotating the paddle below the glider. The red spot illustrated on the paddle is often referred to as the sweet spot. This figure is a modified version of a figure in Sweden International Physicist's Tournament (2017).

motion, we need the centripetal force of the glider to be equal to

$$F_c = \frac{mU_x^2}{R}. \tag{5.60}$$

Denote the banking angle by ϕ, which is positive if the right wing tips up. Now, we can decompose the aerodynamic force along the z-direction of the coordinate system used in (5.47)–(5.50), where we have $\phi = 0$, into a upwards, z-direction, and a sidewards, y-direction, pointing vector. For a left turn with constant speed and radius, we can derive the equations of motion by equating the forces in the x, z and y direction, respectively:

$$\tfrac{1}{2}\rho V^2 S(C_L \sin\gamma + C_D \cos\gamma)\cos\phi = 0 \tag{5.61}$$

$$\tfrac{1}{2}\rho V^2 S(C_L \cos\gamma - C_D \sin\gamma)\cos\phi = mg \tag{5.62}$$

$$\tfrac{1}{2}\rho V^2 S(C_L \cos\gamma - C_D \sin\gamma)\sin\phi = \frac{mU_x^2}{R}, \tag{5.63}$$

where $V = \|\mathbf{V}\|$. The first equation gives us (5.51), where we had found $\gamma_{\text{eq}} \approx -5.5°$. Equation (5.62) yields a solution for V, which we will denote by $V_{\text{eq},c}$, where

$$V_{\text{eq,c}} = \sqrt{\frac{2mg}{\rho S(C_L \cos\gamma_{\text{eq}} - C_D \sin\gamma_{\text{eq}})\cos\phi}} = \frac{V_{\text{eq}}}{\sqrt{\cos\phi}}. \tag{5.64}$$

By using the wind model (5.56), we can set $U_x = V_{\text{eq,c}}$ (similar as before), and solving the last equation (5.63) results into the closed form

$$R = \frac{2m}{\rho S(C_L \cos\gamma_{\text{eq}} - C_D \sin\gamma_{\text{eq}})\sin\phi}. \tag{5.65}$$

We have plotted $V_{\text{eq,c}}$ and R against the banking angle ϕ in Figure 5.13. For $\phi = 30°$, we find that $V_{\text{eq,c}} \approx 0.78\,\text{m/s}$ and $R \approx 17\,\text{cm}$ holds.

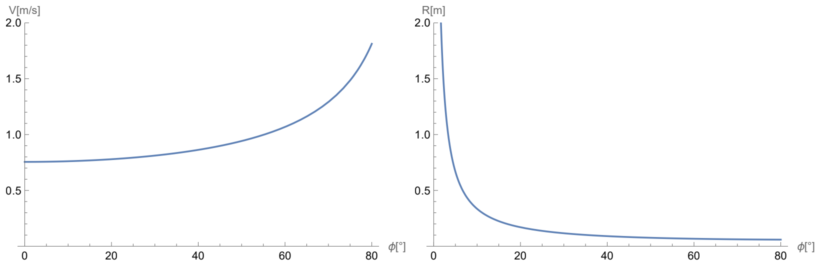


Figure 5.13: The relative speed $V_{\text{eq,c}}$, see (5.64), and turning radius R, see (5.65), plotted against the banking angle ϕ. The glider then moves in a perfect circular trajectory, assuming that the wind generated by the paddle can be approximated with the function in equation (5.56).

Increasing the banking angle with the paddle

As mentioned previously, we can model the aerodynamic force on each on the wings individually. We denote the airflow at the left wing by W_{left} and at the right wing by W_{right}, which consequently induces a force on the left wing $F_{\text{aero,left}}$ and a force on the right wing $F_{\text{aero,right}}$. For this section we only consider the force in the upwards direction with respect to the ground. When this force on the left and right wing is unequal, the induced torque makes the glider bank. This torque is the difference between the forces on the left and right wing multiplied by the arm d. Recall, the arm is the distance between the centerline of the glider and the point on the wing where the force acts, which we approximate as $b/4$. The second derivative of ϕ is then given by the torque divided by the moment of inertia. We approximate the latter as $\frac{1}{12}mb^2$, resulting into the equation

$$\ddot{\phi} = \frac{3}{mb}(F_{\text{left}} - F_{\text{right}}). \tag{5.66}$$

Our goal is to model the change in airflow W due to turning the paddle by an angle ξ. The factor in (5.56) which is the most significant, right now, is the height z. By turning the paddle we increase the height of the left wing and decrease the height of the right wing. With some trigonometric computations, we find that this increase/decrease is equal to $d \sin\xi \sin\theta$. On the other hand, when the banking angle ϕ increases and the plane banks to the right, we find the height decreases on the right wing and increases on the left wing. This increase/decrease is equal to $d \sin\phi$. These two effects cancel out when $\sin\xi \sin\theta = \sin\phi$, which means that $W_{left} = W_{right}$ and the torque is zero. Combining this relation with (5.65) connects the angle turning angle of the glider to the turning radius of the paddle. Mathematically put, we obtain

$$R \sin\xi \sin\theta = \frac{2m}{\rho S(C_L \cos\gamma_{\text{eq}} - C_D \sin\gamma_{\text{eq}})} = constant. \tag{5.67}$$

5.3 Data-driven approach

In parallel to the model-driven approach described in Section 5.2, we used the video data recorded by Zoro Feigl as a starting point of this problem. We begin by discussing the contents of these videos. We then describe our methods for generating data based on the videos and conclude with an analysis of the obtained data.

5.3.1 Available data

Several videos were available of Zoro Feigl flying the walkalong glider in his workplace. In these videos, the camera is mounted behind the paddle. The panel is transparent, allowing for a view of the glider in a consistent perspective. Moreover, the bottom of the glider is marked with a cross consisting of a line from the top to the bottom of the glider in its center and a line from the left wing to the right wing. A characteristic frame from one of the videos is shown in Figure 5.14.

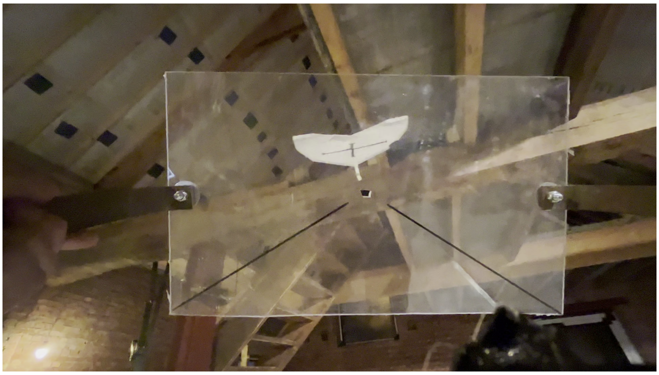


Figure 5.14: Frame of one of the available videos in which Zoro Feigl flies the walkalong glider.

In the videos we observed that the pitch of the glider does not vary much throughout the flight. Hence, we have made the assumption that the pitch of the plane is constant ($\chi = 0$). To extract useful information from the videos, the first step is to detect and locate the glider. This problem is challenging due to a few factors at play.

First, at some points in the videos there is significant background light, which makes it difficult, even for humans, to locate the cross on the glider exactly. Moreover, the perspective of the camera varies between videos and is not completely consistent even within one video. This complicates relating the measurements from these videos to the world coordinate system. Furthermore, the trajectory of the paddle is unknown, which makes relating the data to movements made with the

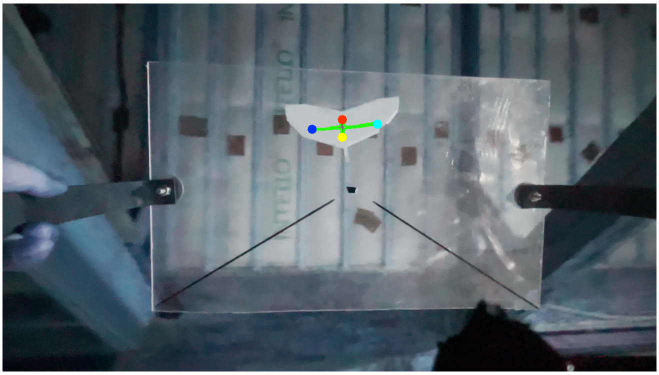


Figure 5.15: Tracking of the cross on the glider by image processing.

paddle impossible. Lastly some relevant factors, as discussed in the modeling section, for example the distance between the glider and the paddle, are challenging to acquire from 2D video data from a single viewpoint.

5.3.2 Detecting the glider

In order to detect the glider in the video footage, we used two methods. First, we implemented an image processing procedure to detect the plane. Subsequently, we implemented a deep learning algorithm in order to make the tracking more robust against variations in lighting and environment.

Classic image processing

Based on the frames as shown in Figure 5.14, we detected the white glider and the cross on the glider, which gave us a location of the 4 points on all ends of the cross (Figure 5.15).

However, this detection is far from perfect, as changes in light/background may result in failures, making the measurements that we obtain with this method very noisy and imperfect.

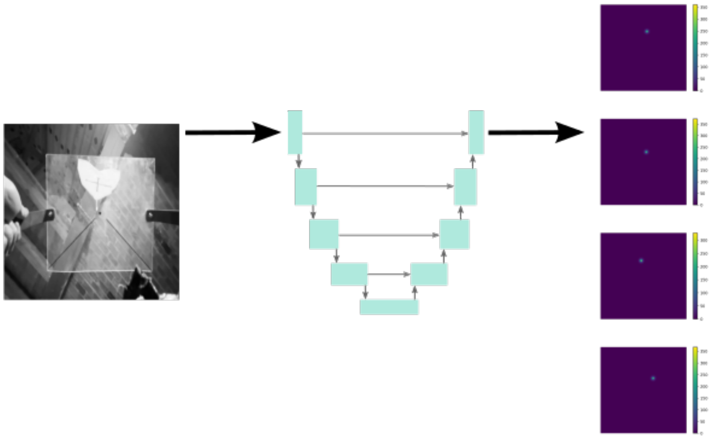


Figure 5.16: Deep learning approach for estimating the location of the glider. Resampled frame is used as input for a CNN, predicting four 2D functions indicating the location of the vertices of the cross drawn on the glider.

Deep learning approach

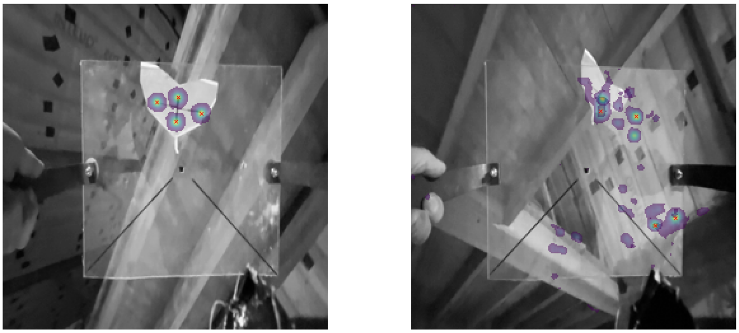
As we identified in the classic imaging approach, the automatic detection of the plane is not robust against changes in lighting and background. Over the last number of years, convolutional neural networks (CNN's) have become very popular in common imaging processing tasks, such as classification, detection and segmentation. CNN's handle image data by extracting distinctive features using consecutive convolution operations. The weights of the convolution kernels and hence the features they pick up are based on the data and follow from an extensive optimization process we refer to as training. It has been well-established that CNN's can be trained to become invariant to these type of conditions, by means of proper data augmentation Shorten and Khoshgoftaar (2019). This means that the automatic detection system can be deployed in a wide variety of surroundings.

To train our network, we first need proper ground-truth data. For this, we manually annotated the four vertices of the cross in a number of frames of one video. The CNN should predict the locations of these four vertices based on image data. To solve this image detection problem, we adapt the method introduced in Sironi et al. (2015), that casts this as a regression problem. For each vertex, we construct a 2D function indicating the location of the vertex with a Gaussian peak, as shown on the right of Figure 5.16. The CNN that we use maps the 2D input image to four 2D output images, predicting the function value for each of their respective vertices, also shown in Figure 5.16. To retrieve the location of each vertex in the image frame, we can simply take the argmax of each of the four images.

We trained a U-Net architecture on a small dataset consisting of 30 samples. The frames from the original video had a resolution of 3840×2160 and had three colour channels. As a preprocessing step, we first resampled the frames to 256×256 pixels, and reduced the colour channels to gray values. These resampled input images are shown on the left of Figure 5.16. We trained the network for 1000 epochs, using an Adam optimizer with a mean squared error loss with learning rate 0.001.

We tested this method on unseen frames, similar to the ones with ground truth annotations. A resulting prediction on one of these frames is shown in Figure 5.17a. The maxima of the predicted locations align with the actual locations of the cross. We only annotated slices at the beginning of the video, due to time constraints in this project [10]. Therefore, for new and unseen situations, the network shows a poor performance (Figure 5.17b). In this case, the glider angle and background lighting differ a lot from the data used for optimization, the network struggles to detect the vertices on the glider. Since detecting the glider on familiar scenes looks promising, we recommend using more annotations with more variety in background and lighting, as well as orientation of the glider as training data for the network.

[10] Annotating the cross on additional video data is not extremely labour intensive.



(a) Unseen, similar frame (b) Unseen, non-similar frame

Figure 5.17: Vertex locations determined using the trained network on different frames of the video.

5.3.3 Analysis of obtained data

We now analyse the data as obtained using the classic image processing approach. As the four points of the cross drawn on the glider are detected, it is possible to find the center point of the cross under the assumption that the paper does not bend. This can then be done by taking the intersection of the line between the leftmost and rightmost point and the line between the top point and bottom point. We then plot the distribution of the measured center points as a heatmap, shown in Figure 5.18. Note that badly detected crosses, identified by a small difference between coordinates of the four points, are not included in the heatmap.

In this heatmap, we see that the glider's position remains close to a specific position relative to the camera. This suggests that it is beneficial for control of the glider to also keep the glider relatively constant in respect to a camera mounted on the robot. Further analysis of videos in which the glider falls could give more insight into the feasibility of certain positions.

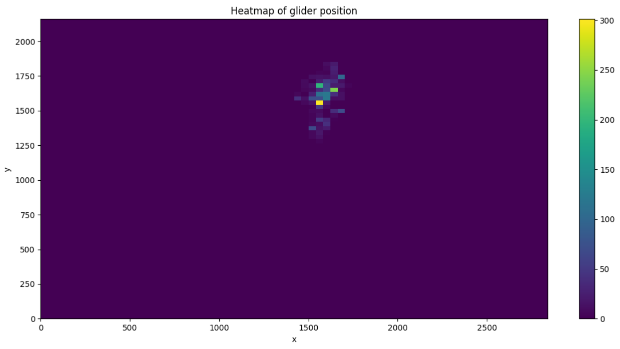


Figure 5.18: Heatmap of the center of the glider relative to the camera. Colours represent the number of frames the center of the glider was in that location in the image frame.

Next, we attempt to find the yaw of the glider. To do this, we assume that the pitch and roll relative to the board are constant, and we thus take the yaw to be the angle of the observed vertical line of the cross. This angle is taken so that an angle of 0 rad corresponds to a vertical line. A positive angle corresponds to the plane turning right and a negative angle corresponds to the plane turning left (Figure 5.19).

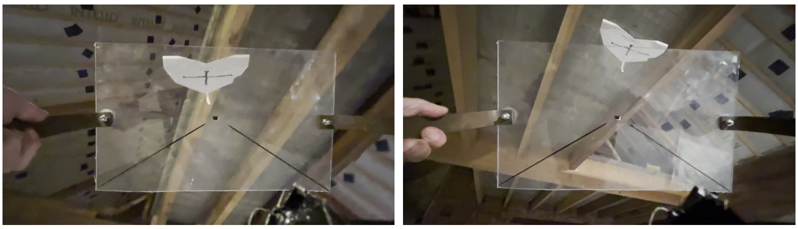


Figure 5.19: Examples of frames for which a small negative yaw (left) and a positive yaw (right) are detected.

Measuring the angle of the plane as observed through the screen for one of the videos provided, we find the result shown in Figure 5.20.

When viewing the video next to this data, large spikes in the observed yaw seemed to correspond to the plane turning in various instances. Therefore, our notion based on these results was that measuring the yaw while flying the glider could help in determining whether controlled actions of the robot have the desired effect. Particularly, it could be used as an indication of whether the glider is turning at a desired rate.

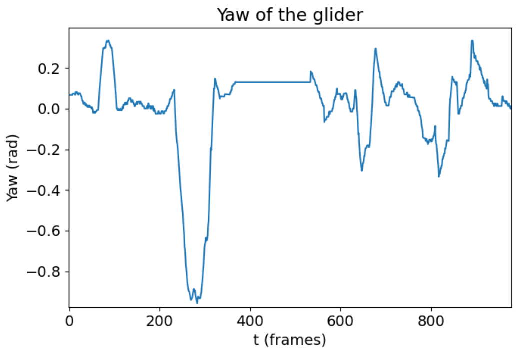


Figure 5.20: Plot of the estimated yaw over time.

5.4 Recommendations and outlook

Based on our findings, we have several recommendations for further development of a control system for the glider project.

From our analysis of the position of the glider, we observed the position of the center of the glider relative to the board is very stable in the correctly scored frames. We expect this could be used as an indication for a control algorithm, as the glider straying too far from the “stable” state could indicate that the control is not working as intended. Moreover, imprecise comparisons between the measured yaw of the glider relative to the board and the corresponding video seem to indicate that a change in yaw correlates with the plane making a turn. This is an observation that can be taken into account when constructing a control strategy. Lastly, the results of our deep learning approach for locating the glider are promising, though more time and data are needed to properly train this network. We expect that such a deep learning approach is appropriate for locating the glider and finetuning control parameters based on processed data on-the-fly.

Large improvements to the data-driven methodologies can be obtained if more details of the data are collected. If the location and distance relative to the camera of all four points of the cross on the glider are known, the metrics discussed in this report could be measured and analysed in much more detail. This leads to more insight in the stable position and rotation of the glider both while turning and flying straight. Finally, we have noticed that the image processing methods applied in this report tend to fail in case of high contrast within the background, or heavy backlight in the video data. This indicates that the room where the installation is placed can influence the effectiveness of control strategies that are based on real-time collected data. Therefore, testing the installation relatively long before the exhibition is opened to visitors is recommended as then changes to either the room or parameters in the control strategy can still be made.

Note that enhancing the data-driven models as proposed above significantly improves the practicality of the model-driven part already. In particular, when we predict the trajectory of the glider, we require a lot of information that should be retrieved from the data. For a proper prediction we need the live location of the glider, as well as its speed and its velocity yaw and pitch. Retrieving the banking angle from the data would be very useful as well, but when the glider is forced to move in a perfect circle, this angle can simply be determined by the angles made by the paddle (see Subsections 5.2.3 and 5.2.3). We think that the banking angle can be retrieved by clever markings on the glider. Also, due to the fact that the location of the paddle is always known—thanks to the technology of the robot arm—we recommend a setup that is able to measure all the angles and distances between the glider and the paddle. This information is just essential for prediction.

Furthermore, we advise those who continue with this project to investigate whether our (simplified) models are realistic. Especially, in the simplified model we approximate the wind generated by the paddle instead of using CFD simulations in order to win computation time, but perhaps this simplification does not coincide with experiments and thus it needs to be verified. Doing more numerical analyses may yield more insight in the flight dynamics of the glider as well. In particular, we expect that several parameters in the models can be quite sensitive. For example, slight changes in the mass parameter m may result into very different results, since the mass of the glider is relatively small (see Table 5.1) and we divide by m in the models. In addition, we do not expect that all the constants in Table 5.1 are correct; these constants can be found by doing enough experiments and subsequently data fitting.

We also recommend—from a modelling perspective—to start with one of the following "simple" trajectories: 1) an almost-circle, i.e., a circular trajectory where the glider needs to fly partly in a straight line without a paddle underneath for a short while (which is due to the fact the robot arm is unable to move in circles indefinitely); 2) an eight-like figure, where the intersection point is above the robot arm. Option 1 is favourable, because then one can use the simplified models. Note that the glider can fly a long distance straight ahead without upward lifting. Option 2 is a favourable trajectory in the sense that it would then not be necessary to let the glider leave the paddle at all. How often one needs to do small corrections and subsequently predict the glider's trajectory again, is something to look into. We actually expect predictions will hold for quite a long time in a proper environment, but there is no problem with regularly applying corrections (if computation time allows it).

Ultimately, a useful continuation of this project may include the construction of a control algorithm. This control algorithm can at least partly be based on the models and observations presented in this report. One idea would be to use funnel control Hackl, Endisch, and Schroder (2008). This particular type of control does not rely on precise knowledge of the underlying system, and aims to limit the solution of the system to a decaying limiting function in time.

References

Beeler, Scott C, Daniel D Moerder, and David E Cox (2003). *A flight dynamics model for a small glider in ambient winds.* Tech. rep. National Aeronautics and Space Administration (NASA).

Etkin, Bernard (2005). *Dynamics of Atmospheric Flight.* Courier Corporation.

FANUC Benelux BV (2022). *Robot arm, M-710iC/12L.* URL `https://www.fanuc.eu/be/en/robots/robot-filter-page/m-710-series/m-710ic-12l`, last accessed on February 17, 2022.

Feigl, Zoro (2021). *Spooky action spatial sketch.* URL `https://www.youtube.com/watch?v=BKchk4GLtQc&ab_channel=ZoroFeigl`, last accessed on March 10, 2022.

Grant, Joseph E (1955). *Method of flying toy airplane and means therefor.* US Patent 2,718,092.

Hackl, Christoph M, Christian Endisch, and D Schroder (2008). "Funnel-control in robotics: An introduction". In: *2008 16th Mediterranean Conference on Control and Automation.* IEEE, pp. 913–919.

Harrison SciencetoyMaker, Slater (2022). *Make the Baby Bug Walkalong Glider.* URL `https://sciencetoymaker.org/walkalong-glider-airsurf-air-surfing/make-your-own-gliders/baby-bug/`, last accessed on February 17, 2022.

Hull, David G (2007). *Fundamentals of airplane flight mechanics.* Vol. 19. Springer.

Shorten, Connor and Taghi M Khoshgoftaar (2019). "A survey on image data augmentation for deep learning". In: *Journal of Big Data* 6.1, pp. 1–48.

Sironi, Amos et al. (2015). "Multiscale centerline detection". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38.7, pp. 1327–1341.

Stengel, Robert F (2004). *Flight Dynamics.* Princeton University Press.

Sweden International Physicist's Tournament, KTH Royal Institute of Technology (2017). *Handy glider.*

Weitz, Lesley A (2015). *Derivation of a point-mass aircraft model used for fast-time simulation.* Tech. rep. MITRE Corporation.

Acknowledgements

The main sponsor for the SWI 2022 was The Netherlands Organisation for Scientific Research (NWO). We gratefully acknowledge their generous support for this event, as well as their continued support for these events in the Netherlands. Additional financial support was provided by 4TU.AMI which we gratefully acknowledge as well. Moreover, the companies who submitted the problems (You2Yourself, PostNL, Fontys, Synopsis) also contributed financially to SWI 2022. Our institution (the University of Twente) as well as PWN (Platform Wiskundig Nederland) provided organizational support. Last but not least we would like to thank all participants for creating an inspiring week of industrial mathematics.

The SWI 2022 organizing committee