Metrics for comparing response time distributions

Bachelor Thesis of

Georg Hinkel

At the Departments of Informatics
Institute for Program Structures and Data Organization (IPD)

Reviewer: Prof. Dr. Ralf H. Reussner
Second reviewer: Prof. Dr. Nobert C. Henze
Advisor: Dr. Jens Happe
Second advisor: Dr. Bruno Ebner

Duration: 20 June 2011 – 30 September 2011
Contents

1. Introduction 1
   1.1. Analyzing performance in modern software systems 1
   1.2. Aim of this Bachelor thesis 2
   1.3. Former results 2

2. Foundations 3
   2.1. \( L^p \)-spaces 3
   2.2. Fourier transform 4
   2.3. Laplace transform 5
   2.4. Moment-generating function 5
   2.5. Numerical integration 6
      2.5.1. Trapezoidal rule 6
      2.5.2. Simpson’s Rule 6
   2.6. Performance Predictions using the Palladio Component Model 7
      2.6.1. Palladio Component Model 7
      2.6.2. PCM-Bench 8

3. Distance measures 13
   3.1. \( L^p \)-norms and \( m_p \)-measures 13
   3.2. Laplace Transform induced measures 16
      3.2.1. Injectivity of the bilateral Laplace Transform on \( C^n(\mathbb{R}) \) 17
      3.2.2. The variance distance measure 20
      3.2.3. The translation measure 21
      3.2.4. Front and tail measure 23

4. Numeric computation of the distance measures 29
   4.1. Numeric computation of \( L^p \)-norms and \( m_p \)-measures 29
   4.2. Numeric computation of the Laplace Transform induced measures 29

5. Evaluation 31
   5.1. \( L^p \)-norms 32
   5.2. Laplace Transform 34

6. Implementation 39
   6.1. Overview 39
   6.2. \( L^p \) spaces 41
   6.3. Laplace Transform induced measures 42

7. Conclusion 45
   7.1. Results 45
   7.2. Related Work 45
   7.3. Future Work 46
Bibliography

Appendix

A. Matlab Implementation
   A.1. Measurements.m
   A.2. laplace.m
   A.3. lpnorm.m
   A.4. theta.m
1. Introduction

1.1. Analyzing performance in modern software systems

Nowadays there are still environments where the software performance is very important. By the term performance we mean that a system is able to fulfill certain runtime constraints, e.g., reaction time or throughput [SW01]. If a system is dimensioned too large, it might require an unnecessary large amount of resources. If a system is dimensioned too small or does not scale properly when faced with more users than it was designed for, the performance requirements may be violated. This mostly causes a direct impact on business. The negative economic consequences mostly include a loss of customers and therefore usually tend to be worse than just inefficiently used resources. For example, if a webshop server crashes under heavy load, the customers not only go to another vendor, but also might honor the crash with bad ratings for the webshop - a marketing desaster. Thus, it is very important to evaluate, whether a system can fulfill its performance requirements even in case of heavy load. For systems that serve many users, e.g. web services, it is important to know, whether certain software architectures support scalability. Doing the design phase, the software architecture can be modified easily and cheaply. Meanwhile, it can become much more long lasting and expensive, if the scalability is only tested at the end of the development cycle and it turns out the system is not scalable.

In practice, performance analysis is rarely done at all, especially in the most important earlier development cycles. Therefore, many performance issues lying in the software architecture remain unveiled until tests or even after the product release. One of the reasons for this is the rare spread of models and tools to analyze the performance of a system or parts of it before its actual implementation.

However, indeed there are models and tools to support performance analysis. One example is the Palladio Component Model (PCM) that will be discussed later in chapter 2.6. This model also supports reverse engineering, i.e. to generate the model out of already written code. The PCM is subject of research, it still has to be evaluated.
1.2. Aim of this Bachelor thesis

In order to evaluate the goodness-of-fit of performance predictions, the results have to be compared with the actual measurements. These predictions and also the measurements are represented as probability distributions. This leads to the main problem to find metrics for comparison of these performance prediction induced probability distributions. If prediction and results (or maybe two different predictions for different setups) are not equal, it is important to understand why. If there were measures that characterized the difference of probability functions automatically, there would be new ways to evaluate and eventually improve the prediction model. To find some of these metrics will be the main goal of this Bachelor Thesis.

But a comparison of distribution functions might also be useful for software architects working with a performance prediction model. It is for example useful to figure out, whether the predictions of two different setups are only shifted or dispersed. It might help to understand the effect of a change in the software architecture before the actual implementation.

Additionally, these metrics are to be implemented within the PCM-Bench. The PCM-Bench is a toolset for performance predictions with the Palladio Component Model (PCM, see Chapter 2.6). The PCM-Bench integration together with its design and test documents are further artifacts of this Bachelor Thesis.

1.3. Former results

In an internal worksheet, the problem has been addressed using stochastic tests, namely Kolmogorov-Smirnov-Test, Anderson-Darling-Test, Cramer-von-Mises-Test, Wilcoxon-Signed-Rank-Test and Dunn Test. All these tests make statements on equality. They consider samples and test whether they could be from the same basic set. Already in case of quite small differences they assume that prediction and measurement are most likely not from the same basic set. This is caused by the fine grained both prediction and measurement probability densities that make the tests very sensible. Trying to apply these tests for comparing performance predictions with the measured results, the result is mostly that the prediction does not coincide with the measurement. There is no distinction whether e.g. the distributions are just shifted or completely different, there is no information about a “distance” between these distributions. This is the issue to be solved with this Bachelor Thesis.
2. Foundations

In this section, we introduce some foundations, this Bachelor thesis is based on. First of all, we introduce the theory of $L^p$-spaces with their induced norms in Section 2.1 as there already are some well known metrics to compare arbitrary functions. Afterwards, we describe the Fourier transform from Functional Analysis in section 2.2 and the close relative, the Laplace transform in section 2.3. Later in this Bachelor thesis, we will use the Laplace transform to detect shifted functions.

2.1. $L^p$-spaces

In Functional Analysis, $L^p$-Spaces ($1 \leq p < \infty$) over a Borel-set $A \in \mathcal{B}^d$ are spaces of functions with finite $L^p$-Norm, which is defined as follows:

**Definition 2.1 ($L^p$-norm)**

Let $f : A \to \mathbb{R}$ be bounded and right continuous, $A \in \mathcal{B}^d$ a bounded borel set and let $1 \leq p \leq \infty$, then we can define

$$\|f\|_p := \left( \int_A |f(x)|^p dx \right)^{\frac{1}{p}}$$

(2.1)

in case $p < \infty$ and furthermore

$$\|f\|_\infty := \text{esssup}_{x \in A} |f(x)|$$

(2.2)

in case $p = \infty$ as the $L^p$-norm of $f$ over $A$.

These norms induce normed vector spaces:

**Definition 2.2 ($L^p(A)$-space)**

Let $1 \leq p \leq \infty$, $A \in \mathcal{B}^d$, then we can define the $L^p$-space over $A$

$$L^p(A) := \{f : A \to \mathbb{R} \mid \|f\|_p < \infty\}$$

(2.3)

Usually, the set $A \subset \mathbb{R}^d$ can be of arbitrary dimension $d \in \mathbb{N}$. However, in this Bachelor thesis we focus on the case $d = 1$. 

3
Furthermore, the elements of any $L^p$-Space usually are considered as spaces of classes of functions that coincide almost everywhere (that is everywhere except for at most countably many points). This is usually done in order to engage definiteness of the $L^p$-norms as defined above. As all elements of such a class coincide almost everywhere, one usually identifies the class of functions with any representative. However, we only consider distribution functions that are right-continuous, so we generally identify such a class of functions with its unique right-continuous representative.

With the element-wise addition, these $L^p$-spaces become vector spaces. Using the $L^p$-norm from (2.1), the $L^p$-Spaces become normed vector spaces which are complete [Sch09]. Since a norm always induces a metric, these spaces and their connected norms may be interesting for finding metrics for the distributions, especially because every bounded continuous function with compact support always lies in $L^p(A)$ for any interval $A$ and $1 \leq p \leq \infty$. So if both performance predictions and results are considered as functions with bounded domain $A$, they are indeed bounded and with compact support and therefore elements in $L^p(A)$.

As we consider probability density functions, it is also known from stochastics ([Hen10], p. 83) that the $L^1$-norm is also exactly twice the variation distance, i.e., let $\mathbb{P}_f, \mathbb{P}_g$ be the probability measures concerning to the probability density functions $f$ and $g$, then we have that

$$d(\mathbb{P}_f, \mathbb{P}_g) := \sup_{S \subset A} |\mathbb{P}_f(S) - \mathbb{P}_g(S)| = \frac{1}{2} \|f - g\|_1. \quad (2.4)$$

### 2.2. Fourier transform

The Fourier transform is an integral transform known from functional analysis and usually applied as Fourier sequences in the context of splitting up any signal into frequencies. We will use the Fourier Transform in a completely different way to detect shifted functions.

**Definition 2.3 (Fourier transform)** The Fourier Transform of a function $f \in L^1(\mathbb{R})$ is given by

$$\hat{f}(\xi) := (\mathcal{F}(f))(\xi) := \frac{1}{\sqrt{2\pi}} \int_{\mathbb{R}} e^{-i\xi x} f(x) dx. \quad (2.5)$$

Here, the integral in (2.5) is a complex integral with $i^2 = -1$ as the imaginary unit.

**Proposition 2.4** The Fourier Transform has the nice property that translations are transformed into factors ([Sch09] p. 71), i.e., for any $t \in \mathbb{R}$ we have

$$(\mathcal{F}(T_t f))(\xi) := e^{i\xi t} \hat{f}(\xi) \quad (2.6)$$

Here, $T_t$ is the translation operator that shifts a function $t$ to the right, i.e.,

$$(T_t f)(\xi) = f(\xi + t). \quad (2.7)$$

Because the Fourier transform is bijective (see for example [Sch09], p. 74), there is no information about the original function $f$ lost in transformation.

---

1Jean Baptiste Joseph Fourier, 1768-1830
2.3. Laplace transform

A close relative to the Fourier Transform is the Laplace transform that is used for instance in probability theory in the context of characteristic functions of probability distributions (Hen10 p. 160).

**Definition 2.5 (Laplace transform)** The Laplace transform of a function $f : \mathbb{R} \rightarrow \mathbb{R}$ is given by

$$(\mathcal{L}(f))(s) := \int_0^\infty e^{-st} f(t) dt$$

(2.8)

The Laplace transform does not necessarily exist for all $s \in \mathbb{R}$.

Like the Fourier Transform, the Laplace transform also has the nice property that translations are transformed into factors.

There is also a bilateral Laplace transform:

**Definition 2.6** Let $f : \mathbb{R} \rightarrow \mathbb{R}$ be a function, then the bilateral Laplace transform of $f$ is defined as

$$\tilde{f}(\xi) := (\mathcal{B}(f))(\xi) := \int_{\mathbb{R}} e^{-\xi t} f(t) dt.$$  

(2.9)

Also the bilateral Laplace transform does not necessarily exist for all points $\xi \in \mathbb{R}$.

The bilateral form is known much less, since its domain is much smaller. Anyway, we will use the bilateral form below. Since we only consider bounded functions with compact support, the integrand in (2.9) also has compact support for any value $\xi \in \mathbb{R}$ and thus, the bilateral Laplace transform exists for any bounded function with compact support on whole $\mathbb{R}$. On the other hand, things are simpler using the bilateral Laplace transform as it does not involve the Heaviside-function.

2.4. Moment-generating function

In this section, we cover some foundations from stochastics, this Bachelor thesis will be based on. In this section, we will always have the probability space $(\Omega, \mathcal{A}, P)$.

**Definition 2.7** The Moment-generating function of a random variable $X$ is defined as

$$M_X(t) := E[e^{tX}] \quad (t \in \mathbb{R})$$

(2.10)

Also the Moment-generating function does not necessarily exist everywhere.

If the random variable $X$ further has a probability density function $f$ regarding to the Lebesque-measure, then this simplifies to

$$M_X(t) = \int_{\mathbb{R}} e^{tx} f(x) dx = \mathcal{B}(-t)$$

(2.11)

The Moment-generating function is called this way, because it also can be rewritten in the following way (see MS04, p. 101)

$$M_X(t) = \sum_{k=0}^{\infty} \frac{t^k}{k!} E[X^k].$$

(2.12)

We will use this in chapter 3.2.3.
2.5. Numerical integration

As Fourier Transform and Laplace transform of a function are integral transforms, working with them involves computing this infinite integral. Since the prediction functions have already been assumed to have compact support, the integral in (2.5), (2.8) and (2.9) can assumed to be finite. However, it is not always possible to solve such an integral in an analytic manner. In addition, measured results usually are just a set of points, so the complete function is unknown. Therefore, the Fourier Transform has to be computed using numeric methods. For more detail of the featured methods, see for example [QSS02].

2.5.1. Trapezoidal rule

The simplest way to approximate any integral is the trapezoidal rule. Given \( n+1 \) equidistant points \( t_k \ (k = 1, \ldots, n; t_0 := a, t_n := b, h := \frac{b-a}{n}) \) of any function \( g \), the trapezoidal rule is the quadrature approximation

\[
\int_a^b g(t) \, dt \approx \frac{h}{2} \sum_{k=1}^{n} \left[ g(t_{k-1}) + g(t_k) \right] =: T(g),
\]

where \( a \) and \( b \) are the lower and upper bounds where the integral is to be computed. In the case of performance predicting functions, we have \( g(t) = e^{-i\xi t} f(t) \) for any point \( \xi \) where one wants to compute the value of the Fourier transform, respectively \( g(t) = e^{-\xi t} f(t) \) if the Laplace transform has to be computed.

There is a known error estimate for the Trapezoidal Rule. The error is bounded by

\[
| \int_a^b g(t) \, dt - T(g) | \leq \frac{h^2}{12} (b-a) \max_{\xi \in [a,b]} |g''(\xi)|.
\]

The inequality yields if \( g \) is twice continuously differentiable.

2.5.2. Simpson’s Rule

Another method for numerical integration is Simpson’s rule. Given \( 2n+1 \) equidistant points \( t_k \ (t_0 := a, t_{2n} := b, h := \frac{b-a}{2n}) \) of any function \( g \), the Simpson Rule yields the following approximation

\[
\int_a^b g(t) \, dt \approx \frac{h}{6} \sum_{k=1}^{n} \left[ g(t_{2k-2}) + 4g(t_{2k-1}) + g(t_{2k}) \right] =: S(g)
\]

where again \( a \) and \( b \) are the lower and upper bounds where the integral should be computed. Simpson’s rule is an easy way to integrate functions with given equidistant points.

There is also an estimate for the error of Simpson’s rule. The error is bounded by

\[
| \int_a^b g(t) \, dt - S(g) | \leq \frac{h^4}{180} (b-a) \max_{\xi \in [a,b]} |g^{(4)}(\xi)|.
\]

The inequality yields if \( g \) is four times continuously differentiable.

Since the error in Simpson’s rule is much less than the error in the Trapezoidal Rule, Simpson’s rule might be interesting, if the error in the Trapezoidal rule gets too large.
2.6. Performance Predictions using the Palladio Component Model

One of the existing models for performance analysis is the Palladio Component Model (PCM) [BKR09]. The PCM is currently developed at Karlsruhe Institute of Technology, FZI Research Center for Information Technology and Paderborn University (see [Pal11]). It simulates the performance of a component-based software system and determines its response times, throughput and resource utilisation based on an abstract architecture model.

In order to compute these predictions, the simulator needs an abstract system design plan, enriched with runtime details and control flow of the components. Software architects and component developers can specify the required information using a rich tool support, the PCM-Bench.

Predictions are based on certain boundary conditions like dependencies to external services (e.g. libraries), resources and the usage profile.

2.6.1. Palladio Component Model

The Palladio Component Model describes software systems as a composition of components which can be described in more detail. For this purpose, components are in general categorized as basic or composite. A basic component is an independent piece of the system, that is entirely new developed, meanwhile a composite component is a composition of one or more other components. Thus, components can be described in deep by describing their structure. In this way, the development of a software system can be considered as development (or reuse) of components and wiring them appropriately.

To allow the wiring, each component must have interfaces, which can here be considered as possible extension points, except for the fact that an interface can exist without any component requiring or providing it. A component can provide or require several interfaces. In order to work correctly, the required interfaces have to be linked to a component providing this interface.

Of course, a component can provide its services to many others requiring any interface the component provides. In this case, one experiences the problem whether such a multiple times used component would be linked as literally the same component (the same instance on the same allocation context) or just a second component instance with the same type.

Because a component can behave different when deployed in different systems, the PCM distinguishes between an abstract component type, its implementation and a deployed instance. The abstract component type is further split up in the Provides Component Type, which basically only contains the provided interfaces, and the Full Component Type, including also i.e. required interfaces, but no information how the services are implemented. The deployed components split up in Assembly Contexts, where the references to external services, e.g. other components or third party libraries, are fixed, and the Allocation Context, where all deployment parameters are fixed, including the information what or how many machines will run it.

Since the Palladio Component Model is designed for performance prediction, it also provides opportunities to describe a components behaviour. In the Service Effect Specification (SEFF), the resource usage of a service can be specified in detail. This information is then evaluated by the Palladio Simulator for performance prediction.
2.6.2. PCM-Bench

PCM-Bench is the tool support for the Palladio Component Model. It is implemented as Add-In for Eclipse. Eclipse is an Open Source project for an extensible integrated development environment. It includes a set of diagrams to enable a user to specify both system design plans, component service performance specifications regarding CPU or HDD usage and also deployment parameters like available CPU, HDD or RAM space on the device where the system parts are going to be deployed.

The performance specification of a service within a component is described by a Service Effect Specification (SEFF), which is represented by a diagram quite similar to the UML activity diagram:

![Diagram](image)

Figure 2.1.: Examples for SEFF diagrams

In the Palladio Component Model, the performance of a software system depends on how the system is used.
The intended way to describe the usage profile is the usage model diagram.

Figure 2.2.: Usage model diagram
The already mentioned performance parameters regarding the available resources, e.g. the CPU and HDD capabilities of the various machines within the system, are described by the resource environment diagram.

Figure 2.3.: Resource environment diagram in PCM-Bench
As all required parameters and specifications are made, it’s possible to simulate the performance of the software system and therefore retrieve performance predictions for the resource usage.

![Pie chart example](image)

Figure 2.4.: Example pie diagram for App server CPU usage

Similar diagrams are available to view the results for i.e. HDD usage or LAN throughput.
3. Distance measures

The following section contains the considerations made on the distance measures. The resulting distance measures can be computed numerically belong to the chapter [3]. In this Bachelor Thesis, the performance prediction functions are always assumed to be piecewise continuous, nonnegative functions with compact support, i.e. \( \{x \in \mathbb{R} | f(x) \neq 0\} \) is bounded. The compact support comes from the assumption that any performance predicting system would never predict an unlimited resource usage, since any infinite loop would rather cause an exception. The piecewise continuity means that any function in this space has at most finitely many discontinuities. At first, we do not use the condition that the functions under consideration are densities of probability distributions, but this is used later on.

The space of these functions under consideration will further be denoted by \( C^c(\mathbb{R}) \), i.e.

\[
C^c(\mathbb{R}) = \left\{ f \in L^\infty(\mathbb{R}) : \{x \in \mathbb{R} : f(x) \neq 0\} \text{ is compact} \right\}.
\]

It is clear that \( C^c(\mathbb{R}) \) is a linear subspace of \( L^\infty(\mathbb{R}) \).

3.1. \( L^p \)-norms and \( m_p \) measures

Most considerations on \( L^p \) spaces are long known in functional analysis and therefore, the \( L^p \)-norms for \( 1 \leq p \leq \infty \) are only considered in the evaluation chapter [5]. What remains about \( L^p \)-norms, is the generalization to values \( 0 < p < 1 \), so let \( m_p \) be defined similar as in (2.1):

**Definition 3.1** Let \( f \in C^c(\mathbb{R}) \), \( 0 < p < 1 \), then we define

\[
m_p(f) := \left( \int_{\mathbb{R}} |f(x)|^p dx \right)^{\frac{1}{p}}.
\]

**Proposition 3.2** For all \( f \in C^c(\mathbb{R}) \) it holds that \( m_p(f) < \infty \).
Proof: Let $f \in C^c(\mathbb{R})$ be arbitrary. Thus we retrieve

$$m_p(f) = \left( \int_{\mathbb{R}} |f(x)|^p dx \right)^{\frac{1}{p}}$$

$$\leq \left( \int_{\mathbb{R}} \|f\|^p_\infty dx \right)^{\frac{1}{p}}$$

$$= \|f\|_\infty \left( \int_{\mathbb{R}} 1_{\{f \neq 0\}} dx \right)^{\frac{1}{p}} < \infty. \quad \blacksquare$$

Remark 3.3 It is clear that $m_p$ also fulfills the homogeneity condition, i.e. for any $\lambda \in \mathbb{R}$ and for any $f \in C^c(\mathbb{R})$, we have

$$m_p(\lambda f) = \left( \int_{\mathbb{R}} |\lambda f(x)|^p dx \right)^{\frac{1}{p}} = |\lambda|m_p(f).$$

But it does not fulfill the triangular inequality and is therefore not a norm. As a counterexample, consider the functions $f$ and $g$ as follows:

$$f(x) := (1 - |x|)^+ = \begin{cases} 1 - |x|, & x \in [-1, 1], \\ 0, & \text{else}, \end{cases}$$

and

$$g(x) := (1 - |x - 3|)^+ = \begin{cases} 1 - |x - 3|, & x \in [2, 4], \\ 0, & \text{else}. \end{cases}$$

We get

$$m_p(f) = \left( \int_{-1}^{1} (1 - |x|)^p dx \right)^{\frac{1}{p}}$$

$$= 2^{\frac{1}{p}} \left( \int_{0}^{1} (1 - |x|)^p dx \right)^{\frac{1}{p}}$$

$$= 2^{\frac{1}{p}} \left[ -\frac{1}{p+1} (1 - x)^{p+1} \right]_0^1$$

$$= \frac{2^\frac{1}{p}}{p+1},$$

where we used the symmetry of $f$ and a linear transform to solve the integral. For $g(x) = f(x - 3)$, we also have $m_p(g) = m_p(f)$ since $m_p$ is invariant under translation and furthermore for the disjoint support of $f$ and $g$, we can also deduct that

$$m_p(f + g) = \left( \int_{-1}^{1} |f(x)|^p dx + \int_{2}^{4} |g(x)|^p dx \right)^{\frac{1}{p}}$$

$$= \left( 2 \int_{-1}^{1} |f(x)|^p dx \right)^{\frac{1}{p}}$$

$$= 2^{\frac{1}{p}} m_p(f)$$

$$< 2m_p(f) = m_p(f) + m_p(g)$$
and thus, $m_p$ does not fulfill the triangle inequality. But it is obvious that $m_p$ still fulfills the definiteness.

**Example 3.4** Consider the sequence of functions $f_n$ defined by

$$f_n(x) := \begin{cases} \frac{1}{n} & x \in [0,n) \\ 0 & \text{else.} \end{cases}$$

It is clear that $f_n \in C_c(\mathbb{R})$ and furthermore $\|f_n\|_1 = 1$ for every $n \in \mathbb{N}$. But we retrieve for $p < 1$

$$m_p(f_n) = \left( \int_{\mathbb{R}} |f_n(x)|^p dx \right)^{\frac{1}{p}}$$

$$= \left( \int_0^n \frac{1}{n^p} dx \right)^{\frac{1}{p}}$$

$$= \frac{n}{n^p}$$

$$= n^{1-p} \xrightarrow{n \to \infty} \infty.$$ 

For $p > 1$, we have $m_p(f_n) \to 0$ as $n \to \infty$. Whereas, if we consider the sequence $g_n$ defined by

$$g_n(x) := n1_{[0,\frac{1}{n}]}(x),$$

we again clearly have $\|g_n\|_1 = 1$ for every $n \in \mathbb{N}$, but we retrieve again for $p < 1$

$$m_p(g_n) = \left( \int_{\mathbb{R}} |g_n(x)|^p dx \right)^{\frac{1}{p}}$$

$$= \left( \int_0^n \frac{1}{n^p} dx \right)^{\frac{1}{p}}$$

$$= \frac{n}{n^p}$$

$$= n^{1-p} \xrightarrow{n \to \infty} 0.$$ 

Similar, we get $m_p(g_n) \to 0$ as $n \to \infty$.

From this result, we can deduct that the $m_p$ measure for $p < 1$ takes into account the Lebesque-measure of the support rather than the supremum. As we intend to use the $m_p$-measure as a distance measure in the context of performance prediction functions, this can be important since the support of the difference function of any two given functions is just the set where the functions differ. A (in the sense of the Lesbeque-measure) small support of the difference function could be a dispersion or a shift of a small part of the function, whereas a large support of the difference function mean that the functions differ on a large domain, e.g. if one of the functions in question is heavy-tailed or they are completely different. The difference between a heavy tail and the case of the functions being anything but similar can be distinguished by other norms, i.e. the $L^p$-norms for large values of $p$ that mind rather the supremum.
3.2. Laplace Transform induced measures

In 2.4, we already had that the Fourier transform translations to factors. We want to use this proposition to get the value for the translation. Thus, let $f, g \in L^1(\mathbb{R})$ the functions to compare, consider the mapping

$$
\theta_{f,g}(\xi) := \ln \left( \frac{\hat{f}(\xi)}{\hat{g}(\xi)} \right) i\xi.
$$

Unlike $\mathbb{R}$, there is no unique inverse mapping of the exponential function and thus, (3.1) is not unambiguous. If it was, $\theta_{f,g}$ mapped translations to constant functions, i.e., for any $f \in C^c(\mathbb{R})$ and $t \in \mathbb{R}$, we had

$$
\theta_{f,T_t f}(\xi) = \ln \left( \frac{f(\xi)}{T_t f(\xi)} \right) i\xi = t
$$

for every $\xi \in \mathbb{R}$ and checking whether $g$ is just a shift of $f$ would be as easy as checking, whether $\theta_{f,g}$ was constant.

To get around the problem of finding an inverse to the complex exponential function, we use the Laplace Transform as in (2.9).

**Proposition 3.5** Let $f \in C^c(\mathbb{R})$. Then $(B(f))(\xi)$ exists for all $\xi \in \mathbb{R}$.

**Proof:** For any $\xi \in \mathbb{R}$ we have

$$
(B(f))(\xi) = \int_{\mathbb{R}} e^{-\xi x} f(x) dx \leq \sup_{x \in \text{supp}(f)} e^{-\xi x} \cdot \lambda(\text{supp}(f)) < \infty.
$$

The last inequality yields as $f$ was assumed to have a compact support. ■

**Corollary 3.6** For any $t \in \mathbb{R}$, we have

$$
B(T_t f) = e^{\xi t} B f,
$$

where $T_t$ is the translation operator as in (2.7).

**Proof:** For any $\xi \in \mathbb{R}$ we have

$$
(B(T_t f))(\xi) = \int_{\mathbb{R}} e^{-\xi x} f(x + t) dx
= \int_{\mathbb{R}} e^{-\xi(z-t)} f(z) dz
= e^{\xi t} (B(f))(\xi),
$$

where we just applied a translation transformation. ■

Given that the ability of $\theta$ to detect shifted functions is mainly based on the translation property and corollary 3.6 yields a quite similar result, we can define
3.2. Laplace Transform induced measures

Definition 3.7 Let \( f, g \in C^c(\mathbb{R}) \), then the **Compare Characteristic** \( \tilde{\theta}_{f,g} \) is defined as

\[
\tilde{\theta}_{f,g}(\xi) := \frac{\ln \left( \frac{\mathcal{B}(f)(\xi)}{\mathcal{B}(g)(\xi)} \right)}{\xi}
\]

\( \tilde{\theta}_{f,g} : \mathbb{R} \setminus \{0\} \to \mathbb{R} \)

Remark 3.8 In case that \( f \) and \( g \) are nonnegative, their bilateral Laplace Transform is strictly positive. Thus, their Compare Characteristic exists on whole \( \mathbb{R} \setminus \{0\} \). In this case, \( \tilde{\theta}_{f,g} \) is continuous.

Same as the formerly defined \( \theta \), the Compare Characteristic also maps a function with its translation to a constant mapping, as we will show in the next proposition.

Proposition 3.9 Let \( f \in C^c(\mathbb{R}) \) and \( t \in \mathbb{R} \). Then it holds that

\[
\tilde{\theta}_{f,T_t f} = -t,
\]

where \( T_t \) is the translation operator from \((2.7)\).

Proof:

\[
\tilde{\theta}_{f,T_t f}(\xi) = \frac{\ln \left( \frac{\mathcal{B}(f)(\xi)}{\mathcal{B}(T_t f)(\xi)} \right)}{\xi}
\]

\[
= \frac{\ln \left( \frac{\mathcal{B}(f)(\xi)}{\mathcal{B}(f)(\xi-e^{-\xi t})} \right)}{\xi}
\]

\[
= \frac{\ln(e^{-\xi t})}{\xi} = -t.
\]

\[
\blacksquare
\]

Thus, the Compare Characteristic of translations is constant. For the inverse proposition, we need the injectivity of the bilateral Laplace Transform.

3.2.1. Injectivity of the bilateral Laplace Transform on \( C^c(\mathbb{R}) \)

In [DB06], p. 134, an inverse formula for the Laplace Transform is given. However, since the domain of the Laplace transform is still unclear, especially if we require the transform function to exist on whole \( \mathbb{R} \), I want to give a proof for the injectivity of the Laplace Transform on \( C^c(\mathbb{R}) \).

Proposition 3.10 For any \( f, g \in C^c(\mathbb{R}) \), we have \( \int_{\mathbb{R}} \tilde{f}(x)g(x)dx = \int_{\mathbb{R}} f(x)\tilde{g}(x)dx \).

Proof: Since all \( f, g \in C^c(\mathbb{R}) \) have a compact support, \( (x, y) \mapsto e^{-xy}f(x)g(y) \) is integrable and thus Fubini’s theorem yields

\[
\int_{\mathbb{R}} \tilde{f}(x)g(x)dx = \int_{\mathbb{R}} \int_{\mathbb{R}} e^{-xy}f(y)g(x)dy dx
\]

\[
= \int_{\mathbb{R}} \int_{\mathbb{R}} e^{-xy}f(y)g(x)dx dy
\]

\[
= \int_{\mathbb{R}} f(y) \int_{\mathbb{R}} e^{-xy}g(x)dx dy
\]

\[
= \int_{\mathbb{R}} f(y)\tilde{g}(y)dy
\]

\[
\blacksquare
\]
Remark 3.11 Unfortunately, for any $f \in C^c(\mathbb{R})$, $\mathcal{B}(f)$ is not necessarily integrable. Consider for instance the indicator function $1_{[0,1]}$. It holds that

$$
(\mathcal{B}1_{[0,1]})(\xi) = \int_{\mathbb{R}} e^{-\xi x} 1_{[0,1]}(x) \, dx \\
= \int_0^1 e^{-\xi x} \, dx \\
= \begin{cases} 
\frac{e^{-\xi}}{-\xi} - \frac{1}{\xi} & (\xi \neq 0) \\
\int_0^1 dx = 1 & (\xi = 0)
\end{cases}
$$

which is not integrable on $\mathbb{R}$ for $\frac{1-e^{-\xi}}{-\xi} \to \infty$ as $\xi \to -\infty$. Furthermore, obviously we have that $(\mathcal{B}1_{[0,1]})(\xi) \neq 0$ for all $\xi \in \mathbb{R}$, so that in general $\mathcal{B}(C^c(\mathbb{R})) \not\subseteq C^c(\mathbb{R})$.

The last remark implies that a bijectivity proof would require more considerations on $\mathcal{B}(C^c(\mathbb{R}))$ as $\mathcal{B}(C^c(\mathbb{R})) \neq C^c(\mathbb{R})$. However, we only need the injectivity to get the inverse of proposition 3.9.

Theorem 3.12 $\mathcal{B}$ is injective on $C^c(\mathbb{R})$.

Proof: Let $f, g \in C^c(\mathbb{R})$ be arbitrary. We want to show the implication

$$\hat{f} = \hat{g} \Rightarrow f = g$$

Since $f, g \in C^c(\mathbb{R})$, there exists an $R > 0$, such that

$$\text{supp}(f) \subset [-R, R], \text{supp}(g) \subset [-R, R].$$

Thus, we have

$$\int_{\mathbb{R}} e^{-xy} f(y) \, dy = \int_{\mathbb{R}} e^{-xy} g(y) \, dy \quad (x \in \mathbb{R})$$

and hence

$$h(\lambda) := \int_{\mathbb{R}} e^{-\lambda x} (f(x) - g(x)) \, dx \equiv 0.$$ 

With the above defined $R$, we retrieve

$$h(\lambda) = \int_{-R}^R e^{-\lambda x} (f(x) - g(x)) \, dx \equiv 0.$$ 

This implies that the above defined function $h$ is constant. Since $h$ is continuously differentiable, its derivative must also be constant zero, so we retrieve
3.2. Laplace Transform induced measures

\[
\frac{\partial}{\partial \lambda} \int_{-R}^{R} e^{-\lambda x} (f(x) - g(x)) \, dx = \int_{-R}^{R} \frac{\partial}{\partial \lambda} e^{-\lambda x} (f(x) - g(x)) \, dx \\
= \int_{-R}^{R} (-x) e^{-\lambda x} (f(x) - g(x)) \, dx \\
= - \int_{-R}^{R} xe^{-\lambda x} (f(x) - g(x)) \, dx \equiv 0.
\]

Inductively and using the linearity of the integral, we obtain that for all polynomials \( p \) it holds that

\[
\int_{\mathbb{R}} p(x) (f(x) - g(x)) \, dx \equiv 0,
\]
where we set \( \lambda = 0 \).

Now fix any \( \alpha > 0, y \in \mathbb{R} \) and consider the function

\[
\gamma_\alpha(x) := \frac{1}{\alpha} e^{-\frac{(x-y)^2}{\alpha}}.
\]

As a composition of continuous functions, \( \gamma_\alpha \) is continuous. According to Weierstraß approximation theorem, we obtain that there is a sequence \( (p^{(n)}_\alpha)_{n \in \mathbb{N}} \) of polynomials, such that

\[
|\gamma_\alpha(x) - p^{(n)}_\alpha(x)| \overset{\rightarrow}{n \to \infty} 0 \quad \text{for all } x \in [-R,R].
\]

With the theorem of dominated convergence and the majorant \( \frac{2}{\alpha} (|f| + |g|) \) we obtain that

\[
0 = \lim_{n \to \infty} \int_{\mathbb{R}} p^{(n)}_\alpha(x) (f(x) - g(x)) \, dx \\
= \int_{\mathbb{R}} \lim_{n \to \infty} p^{(n)}_\alpha(x) (f(x) - g(x)) \, dx \\
= \int_{\mathbb{R}} \gamma_\alpha(x) (f(x) - g(x)) \, dx
\]
for every \( \alpha > 0 \). Thus, taking \( \alpha_n = \frac{1}{n} \) yields
0 = \lim_{n \to \infty} \int_{\mathbb{R}} \gamma_n (x)(f(x) - g(x))dx
= \lim_{n \to \infty} \int_{\mathbb{R}} ne^{-n^2(x-y)^2} (f(x) - g(x))dx
= \lim_{n \to \infty} \int_{\mathbb{R}} e^{-z^2} (f(\frac{1}{n}z + y) - g(\frac{1}{n}z + y))dz \tag{3.2}
\geq \int_{\mathbb{R}} \lim_{n \to \infty} e^{-z^2} (f(\frac{1}{n}z + y) - g(\frac{1}{n}z + y))dz \tag{3.3}
= \int_{\mathbb{R}} e^{-z^2} (f(y) - g(y))dz
= (f(y) - g(y)) \int_{\mathbb{R}} e^{-z^2}dz \tag{3.4}
= (f(y) - g(y)) \sqrt{\pi}, \tag{3.5}
where we used the substitution \( z = n(x - y) \) in (3.2) and applied Fatou’s lemma in (3.3). The integral \( \int_{\mathbb{R}} e^{-z^2}dz \) in (3.4) can be computed using polar coordinates, but I leave out the proof here, since this is a well known integral.

An immediate consequence of (3.5) is that \( f(y) - g(y) \geq 0 \). Since \( f \) and \( g \) can be swapped, we can also assume that \( g(y) - f(y) \geq 0 \) and therefore \( f(y) = g(y) \). Since \( y \) was arbitrary, we obtain, that \( f \equiv g \) and thus, \( B \) is injective. \( \blacksquare \)

### 3.2.2. The variance distance measure

Reconsider proposition 3.9 if \( \bar{\theta}_{f,g} \equiv c \) for a constant \( c \in \mathbb{R} \), then we can assume that

\[
\ln \left( \frac{\bar{f}(\xi)}{\bar{g}(\xi)} \right) = c \xi
\]

for any \( \xi \in \mathbb{R} \) and therefore

\[
(Bg)(\xi) = \bar{g}(\xi) = e^{-c\xi}\bar{f}(\xi) = e^{-c\xi}(Bf)(\xi).
\]

For a fixed \( f \in C^c(\mathbb{R}) \), with the above considerations \( g = T_c f \) solves this equation. Now for the injectivity of \( B \), this is the only solution. This result is the basement on the measure we will introduce in this section.

**Definition 3.13** For \( f, g \in C^c(\mathbb{R}) \) non-negative, the theta variance distance measure over \( \mathbb{R} \) is defined as follows:

\[
m_{\text{Var}(\bar{\theta})} : C^c(\mathbb{R}) \times C^c(\mathbb{R}) \to [0, \infty],
\]

\[
m_{\text{Var}(\bar{\theta})}(f, g) := \text{Var}(\bar{\theta}) = \inf_{\mu \in \mathbb{R}} \int_{\mathbb{R}} (\bar{\theta}_{f,g}(\xi) - \mu)^2 d\xi.
\]

**Corollary 3.14** Let \( f, g \in C^c(\mathbb{R}) \). If \( m_{\text{Var}(\bar{\theta})}(f, g) = 0 \), then there is a \( \mu \in \mathbb{R} \), such that

\[
g(x) = f(x + \mu) \text{ for all } x \in \mathbb{R}.
\]
3.2. Laplace Transform induced measures

Proof: \((\hat{\theta}_{f,g}(\xi) - \mu)^2\) is non-negative for all \(\xi, \mu \in \mathbb{R}\). As the mapping

\[
I_{f,g}(\mu) := \int_{\mathbb{R}} (\hat{\theta}_{f,g}(\xi) - \mu)^2 d\xi
\]

is continuous, there is a \(\mu_0\), such that \(I_{f,g}(\mu_0) = 0\). With the non-negativeness, we retrieve that \((\hat{\theta}_{f,g}(\xi) - \mu_0)^2 = 0\) for all \(\xi \in \mathbb{R}\). □

**Proposition 3.15** The variance distance measure as defined in 3.13 is invariant under translation of both arguments, i.e. for any \(f, g \in C^c(\mathbb{R})\) and \(s, t \in \mathbb{R}\), we have

\[
m_{\text{Var}}(\hat{\theta}(T_s f, T_t g)) = m_{\text{Var}}(\hat{\theta}(f, g))
\]

Proof: Let \(f, g \in C^c(\mathbb{R})\) and \(s, t \in \mathbb{R}\) arbitrary. Using the translation operator from (2.7), we can compute

\[
\hat{\theta}_{T_s f, T_t g}(\xi) = \frac{\ln\left(\frac{\mathcal{B}(T_s f)(\xi)}{\mathcal{B}(T_t g)(\xi)}\right)}{\xi} = \frac{\ln(e^{(s-t)} f(\xi))}{\xi} = \frac{\xi(s-t) + \ln(f(\xi))}{\xi} = (s-t) + \hat{\theta}_{f,g}(\xi).
\]

So \(\hat{\theta}_{T_s f, T_t g}\) and \(\hat{\theta}_{f,g}\) only differ by a constant additive term. Since the infimum over a subtraction is taken, this will be ignored.

3.2.3. The translation measure

If \(g\) is just a shift of \(f\), the compare characteristic \(\hat{\theta}_{f,g}\) is constant and equals the negative translation. Now if \(g\) is not just a shift of \(f\), it is unclear, where to evaluate the characteristic. This section takes this into account.

In this section, we work on a probability space \((\Omega, \mathcal{A}, P) = (\mathbb{R}, \mathcal{B}, \lambda^1)\). We assume two random variables \(X\) and \(Y\) with probability densities \(f\) (for \(X\)) and \(g\) (for \(Y\)). We further want to claim that both \(f, g \in C^c(\mathbb{R})\).

**Definition 3.16** Let \(f, g \in C^c(\mathbb{R})\). We then define the translation measure as the following

\[
m_{\text{mw}}(f, g) := \lim_{\xi \to 0} \hat{\theta}_{f,g}(\xi).
\]

One might expect a measure to be non-negative, but taking the absolute value would actually lose the information on the "direction" of the translation measure. However, the first thing we will need to show is the existence of the translation measure for all \(f, g \in C^c(\mathbb{R})\).
Proposition 3.17 Let $X$ be a random variable with probability density $f \in C_c(\mathbb{R})$, then it holds that
\[
\lim_{n \to \infty} \left( (B(f))(\frac{1}{n}) \right)^n = e^{-E[X]}.
\]
Particularly, the mean value of $X$ exists.

Proof: As the probability density $f$ has a compact domain and is bounded, all moments $E[X^k]$ ($k \in \mathbb{N}$) exist. Using (2.11) and (2.12), we obtain that
\[
h(\xi) := (B(f))(\xi) \frac{1}{\xi}
\]
\[
= (M_X(-\xi))^{\frac{1}{\xi}}
\]
\[
= \left( \sum_{k=0}^{\infty} \frac{(-\xi)^k}{k!} E[X^k] \right)^{\frac{1}{\xi}}
\]
Using (9), we show the claim in two parts. Let $t_n = \frac{1}{n}$. Thus we retrieve
\[
h(t_n) = \left( 1 - \frac{E[X]}{n} + \frac{1}{n^2} \sum_{k=2}^{\infty} \frac{(-1)^n}{k! n^{k-2}} E[X^k] \right)^n
\]
As $\frac{1}{n^2} \sum_{k=2}^{\infty} \frac{(-1)^n}{k! n^{k-2}} E[X^k] = O(\frac{1}{n^2})$ as $n \to \infty$, the claimed limit is well known (for $(t_n)_{n \in \mathbb{N}}$). The proof for $t_n = -\frac{1}{n}$ is analogue.

Theorem 3.18 Let $X$ and $Y$ be random variables with Lebesque-densities $f$ and $g$ with $f, g \in C_c(\mathbb{R})$. Then it holds that
\[
\lim_{\xi \to 0} \tilde{\theta}_{f,g}(\xi) = E[Y] - E[X].
\]
Particularly, $\tilde{\theta}_{f,g}$ is continuous.

Proof: As a integral transform with kernel function that is differentiable, $B$ is differentiable. As $(B(z))(0) = 1$ for any $z \in C_c(\mathbb{R})$, there is a neighbourhood of 0 where both $B(f)$ and $B(g)$ are positive. Thus, we have for any $\xi$ in this neighbourhood that
\[
h(\xi) := \ln \left( \frac{(B(f))(\xi)}{(B(g))(\xi)} \right)
\]
is differentiable with derivative
\[
h'(\xi) = \frac{(B(g))(\xi)(B'(f))(\xi)(B(g))(\xi) - (B(f))(\xi)(B'(g))(\xi)}{(B(f))(\xi)(B(g))(\xi)}
\]
\[
= \frac{(B'(f))(\xi)(B(g))(\xi) - (B(f))(\xi)(B'(g))(\xi)}{(B(f))(\xi)(B(g))(\xi)}.
\]
Now we have that
\[
(B'(f))(0) = \int_{\mathbb{R}} -te^{-t\xi} f(t)dt = -E[X]
\]
and equally \((B'(g))(0) = -E[Y]\). Now, using \((B(f))(0) = (B(g))(0) = 1\) we obtain
\[
h'(0) = E[Y] - E[X].
\]
L'Hôpital’s rule yields the claim.

### 3.2.4. Front and tail measure

In this section, we make deeper considerations on the limits of the compare characteristic for \(\xi \to \pm \infty\).

**Definition 3.19** Let \(f \in C^c(\mathbb{R})\), we then define

(i) \(\underline{x}(f) := \inf \text{supp} f\),

(ii) \(\overline{x}(f) := \sup \text{supp} f\).

**Theorem 3.20** Let \(f, g \in C^c(\mathbb{R})\) be probability density functions, then we have the limit
\[
\lim_{\xi \to \infty} \hat{\theta}_{f,g}(\xi) = \overline{x}(g) - \underline{x}(f).
\]

**Proof:** We split the proof in two parts. We will first show "\(\geq\)". Let \(\xi > 0, \epsilon > 0\), then we obtain
\[
\frac{\hat{f}(\xi)}{\hat{g}(\xi)} = \int_{\mathbb{R}} e^{-\xi x} f(x) dx \int_{\mathbb{R}} e^{-\xi x} g(x) dx \geq \int_{\mathbb{R}} e^{-\xi x} f(x) dx \int_{\mathbb{R}} e^{-\xi \underline{x}(g)} g(x) dx = e^{\xi \underline{x}(g)} \int_{\mathbb{R}} e^{-\xi x} f(x) dx
\]
\[
= e^{\xi \underline{x}(g)} \left( \int_{-\infty}^{\underline{x}(f)+\epsilon} e^{-\xi x} f(x) dx + \int_{\underline{x}(f)+\epsilon}^{\infty} e^{-\xi x} f(x) dx \right) \quad (3.6)
\]
where we applied the fact that \(\|g\|_1 = 1\) (since \(g\) is a probability density function). Using the extended mean value theorem we further retrieve
\[
\int_{-\infty}^{\underline{x}(f)+\epsilon} e^{-\xi x} f(x) dx + \int_{\underline{x}(f)+\epsilon}^{\infty} e^{-\xi x} f(x) dx = e^{-\xi \eta_1} \int_{-\infty}^{\underline{x}(f)+\epsilon} f(x) dx + e^{-\xi \eta_2} \int_{\underline{x}(f)+\epsilon}^{\infty} f(x) dx \quad (3.7)
\]
for some \(\eta_1 \in (-\infty, \underline{x}(f) + \epsilon)\) and \(\eta_2 \in (\underline{x}(f) + \epsilon, \infty)\). Further we have
\[
\int_{-\infty}^{\underline{x}(f)+\epsilon} f(x) dx = \int_{\underline{x}(f)}^{\underline{x}(f)+\epsilon} f(x) dx > 0
\]
since \(f\) is non-negative and \(\underline{x}(f) = \inf \text{supp}(f)\). Furthermore, because \(f\) is a probability density function, we have
\[
\int_{\tilde{z}(f)+\epsilon}^{\infty} f(x)dx < 1 < \infty.
\]

Now as especially \(\eta_1 < \tilde{z}(f) + \epsilon\) and \(\eta_2 > \tilde{z}(f) + \epsilon\), we can get the limit

\[
\lim_{\xi \to \infty} \left( e^{\xi(\tilde{z}(f)+\epsilon)} \left( e^{-\xi \eta_1} \int_{-\infty}^{\tilde{z}(f)+\epsilon} f(x)dx + e^{-\xi \eta_2} \int_{\tilde{z}(f)+\epsilon}^{\infty} f(x)dx \right) \right)^{\xi^{-1}} = e^{\tilde{z}(f)+\epsilon-\eta_1},
\]

and deduce that

\[
\lim_{\xi \to \infty} \left( e^{-\xi \eta_1} \int_{-\infty}^{\tilde{z}(f)+\epsilon} f(x)dx + e^{-\xi \eta_2} \int_{\tilde{z}(f)+\epsilon}^{\infty} f(x)dx \right)^{\xi^{-1}} = e^{-\eta_1}.
\]

Hence using (3.6), (3.7) and (3.8)

\[
\lim_{\xi \to \infty} \left( \frac{\tilde{f}(\xi)}{\tilde{g}(\xi)} \right)^{\xi} \geq e^{\tilde{z}(g)} \left( \int_{-\infty}^{\tilde{z}(f)+\epsilon} e^{-\xi x} f(x)dx + \int_{\tilde{z}(f)+\epsilon}^{\infty} e^{-\xi x} f(x)dx \right)^{\xi^{-1}} = e^{\tilde{z}(g)-\tilde{z}(f)-\epsilon}.
\]

Taking the logarithm leads to

\[
\lim_{\xi \to \infty} \tilde{\theta}_{f,g}(\xi) \geq \tilde{z}(g) - \tilde{z}(f) - \epsilon.
\]

As \(\epsilon\) was arbitrary, this is the first part.

For "\(\leq\)", we use the first part and obtain

\[
\lim_{\xi \to \infty} \tilde{\theta}_{f,g}(\xi) = \lim_{\xi \to \infty} -\tilde{\theta}_{g,f}(\xi)
\leq -(\tilde{z}(f) - \tilde{z}(g)) = \tilde{z}(g) - \tilde{z}(f)
\]

\(\blacksquare\)
Theorem 3.21 Let \( f, g \in C_c(\mathbb{R}) \) be probability density functions, then we have the limit
\[
\lim_{\xi \to -\infty} \hat{\theta}_{f,g}(\xi) = \bar{\pi}(f) - \bar{\pi}(g).
\]

Proof: Again we split the proof in two parts showing the inequality "\( \geq \)" first. Let \( \xi > 0, \epsilon > 0 \), then we obtain
\[
\frac{\hat{f}(\xi)}{\hat{g}(\xi)} = \frac{\int_{\mathbb{R}} e^{-\xi x} f(x) dx}{\int_{\mathbb{R}} e^{-\xi x} g(x) dx} \geq \frac{\int_{\mathbb{R}} e^{-\xi x} f(x) dx}{\int_{\mathbb{R}} e^{-\xi \bar{\pi}(g)} g(x) dx} = e^{\xi \bar{\pi}(g)} \int_{\mathbb{R}} e^{-\xi x} f(x) dx
\]
\[
= e^{\xi \bar{\pi}(g)} \left( \int_{-\infty}^{\bar{\pi}(f) - \epsilon} e^{-\xi x} f(x) dx + \int_{\bar{\pi}(f) - \epsilon}^{\infty} e^{-\xi x} f(x) dx \right) \tag{3.9}
\]
where we applied the fact that \( \|g\|_1 = 1 \). Using the extended mean value theorem we further retrieve
\[
\int_{-\infty}^{\bar{\pi}(f) - \epsilon} e^{-\xi x} f(x) dx + \int_{\bar{\pi}(f) - \epsilon}^{\infty} e^{-\xi x} f(x) dx = e^{-\xi \eta_1} \int_{-\infty}^{\bar{\pi}(f) - \epsilon} f(x) dx + e^{-\xi \eta_2} \int_{\bar{\pi}(f) - \epsilon}^{\infty} f(x) dx \tag{3.10}
\]
for some \( \eta_1 \in (-\infty, \bar{\pi}(f) - \epsilon) \) and \( \eta_2 \in (\bar{\pi}(f) - \epsilon, \infty) \). Further we have
\[
\int_{\bar{\pi}(f) - \epsilon}^{\infty} f(x) dx = 
\]
\[
\int_{\bar{\pi}(f) - \epsilon}^{\infty} f(x) dx > 0
\]
since \( f \) is non-negative and \( \bar{\pi}(f) = \sup \text{supp}(f) \). Furthermore, because \( f \) is a probability density function, we have
\[
\int_{-\infty}^{\bar{\pi}(f) - \epsilon} f(x) dx < 1 < \infty.
\]
Now as especially \( \eta_1 < \bar{\pi}(f) - \epsilon \) and \( \eta_2 > \bar{\pi}(f) - \epsilon \), we can get the limit
\[
\lim_{\xi \to -\infty} \left( e^{\xi(\pi(f)-\epsilon)} \left( e^{-\xi \eta_1} \int_{-\infty}^{-\epsilon} f(x)dx + e^{-\xi \eta_2} \int_{-\epsilon}^{\infty} f(x)dx \right) \right)^{\frac{1}{\xi}} = \\
= \lim_{\xi \to -\infty} \left( e^{\xi(\pi(f)-\epsilon - \eta_1)} \int_{-\infty}^{-\epsilon} f(x)dx + e^{\xi(\pi(f)-\epsilon - \eta_2)} \int_{-\epsilon}^{\infty} f(x)dx \right)^{\frac{1}{\xi}} = \\
= \lim_{\xi \to -\infty} e^{\pi(f)-\epsilon - \eta_2} \left( \int_{-\infty}^{-\epsilon} f(x)dx \right)^{\frac{1}{\xi}} = e^{\pi(f)-\epsilon - \eta_2},
\]

and deduce that
\[
\lim_{\xi \to -\infty} \left( e^{-\xi \eta_1} \int_{-\infty}^{-\epsilon} f(x)dx + e^{-\xi \eta_2} \int_{-\epsilon}^{\infty} f(x)dx \right)^{\frac{1}{\xi}} = e^{-\eta_2} \tag{3.11}
\]

Hence using (3.9), (3.10) and (3.11)
\[
\lim_{\xi \to -\infty} \left( \frac{\hat{f}(\xi)}{\hat{g}(\xi)} \right)^{\frac{1}{\xi}} \geq e^{\pi(g)} \left( \int_{-\infty}^{-\epsilon} e^{-\xi x} f(x)dx + \int_{-\epsilon}^{\infty} e^{-\xi x} f(x)dx \right)^{\frac{1}{\xi}} = \\
e^{\pi(g)} \left( e^{-\xi \eta_1} \int_{-\infty}^{-\epsilon} f(x)dx + e^{-\xi \eta_2} \int_{-\epsilon}^{\infty} f(x)dx \right)^{\frac{1}{\xi}} = \\
e^{\pi(g)-\eta_2} \geq e^{\pi(g)-\pi(f)+\epsilon}.
\]

Taking the logarithm leads to
\[
\lim_{\xi \to -\infty} \hat{\theta}_{f,g}(\xi) \geq \pi(g) - \pi(f) + \epsilon
\]

As \(\epsilon\) was arbitrary, this is the first part.

Again, we retrieve the second inequality by using the first part and
\[
\lim_{\xi \to -\infty} \hat{\theta}_{f,g}(\xi) = \lim_{\xi \to -\infty} -\hat{\theta}_{g,f}(\xi) \leq -(\pi(f) - \pi(g)) = \pi(g) - \pi(f)
\]

From the proofs of the last theorems 3.20 and 3.21, the behaviour of \(f\) and \(g\) at the rim of their support is of great importance to the limit of \(\hat{\theta}_{f,g}\) and the speed of convergence. Thus, the intuition behind the following definitions is to catch this behaviour of \(f\) and \(g\) at the rims by analyzing their speed of convergence. We approximate this by the derivative.
Definition 3.22 Let $f, g \in C_c^c(\mathbb{R})$ be probability density functions. For any "large" $\xi \in \mathbb{R}$ we define

(i) the front measure at $\xi$ of $f$ and $g$ as

$$m^{(\xi)}_F(f, g) := \tilde{\theta}'_{f,g}(\xi),$$

(ii) the tail measure at $\xi$ of $f$ and $g$ as

$$m^{(\xi)}_T(f, g) := \tilde{\theta}'_{f,g}(-\xi).$$

Remark 3.23 Note that we did not proof yet, that the Compare Characteristic is always differentiable. This might be subject to future work. It is actually hard to get an idea what value of $\xi$ to choose. A smaller value of $\xi$ will mostly ignore the limit process, but on the other hand side we have $\lim_{\xi \to \pm \infty} \tilde{\theta}'_{f,g}(\xi) = 0$ as $\tilde{\theta}_{f,g}$ converges.

As the previous considerations show, the compare characteristics pay attention to the probability density’s tail as $\xi \to -\infty$ and to the front as $\xi \to \infty$. Using the derivative, one can see the direction, where the compare characteristic is going. Again, the consideration of two probability densities only shifted is an advantage. In the case of only shifted probability densities, their compare characteristic is constant and describes the value of the translation. Now as $f$ and $g$ might not be just translated, the value of the compare characteristics is somehow the translation if you emphasize on a particular area of the distribution (as mentioned, as $\xi \to \infty$ the compare characteristic pays attention to the front, etc.).

If $m^{(\xi)}_F(f, g)$ was positive for a large value of $\xi$, the value of the compare characteristic rises as it pays attention to the very front of the distributions. As for [3.18] and considering the bilateral Laplace Transform as just a weighted integral, we can assume that $f$ has a larger front. This can also be seen by considering the effect on putting more emphasis on the tail of $f$. This would clearly make the expected value to rise and thus for [3.18] the compare characteristic would get smaller. The same effect applies to $g$ with the difference that this would lower the difference $E[Y] - E[X]$ (where $X$ is distributed with density $f$ and $Y$ is distributed with density $g$). The intuition is that the density with "heavier front" wins this battle. As we considered $m^{(\xi)}_F(f, g) > 0$, $f$ obviously won and therefore must have had a larger front. As $\tilde{\theta}_{f,g} = -\tilde{\theta}_{g,f}$, we can further deduct that in case of $m^{(\xi)}_T(f, g) < 0$, $g$ has a larger front than $f$.

Similar, we can deduce an interpretation of the tail distance measure. If $m^{(\xi)}_T(f, g) > 0$, the value of the compare characteristics gets smaller as it pays more attention to the tail of the distributions (beware that we let $\xi \to -\infty$). As a greater emphasis on a distributions tail would clearly make the expected value rise, we again have this kind of a battle of $f$ and $g$. As the value of the compare characteristics gets smaller, again, $f$ seems to have won this battle because of a "heavier tail".

Remark 3.24 The theorems [3.20] and [3.21] also lead to a problem with the variance distance measure: As the differences $\tilde{\pi}(g) - \tilde{\pi}(f)$ and $\pi(g) - \pi(f)$ do not necessarily equal each other, the variance might not exist and thus $m^{\text{Var}}_{\tilde{\pi}}(f, g) = \infty$. To solve this issue, we finally define
**Definition 3.25** For $f, g \in C^c(\mathbb{R})$ non-negative, the theta variance distance measure over a bounded subset $A \subset \mathbb{R}$ is defined as follows:

$$m_{\text{Var}(\theta)}|A : C^c(\mathbb{R}) \times C^c(\mathbb{R}) \rightarrow [0, \infty),$$

$$m_{\text{Var}(\theta)}|A(f, g) := \inf_{\mu \in \mathbb{R}} \int_{A} (\hat{\theta}_{f,g}(\xi) - \mu)^2 d\xi.$$ 

**Corollary 3.26** Let $(\Omega, A, P) = (A, A \cap \mathcal{B}, \frac{1}{\lambda(A)} \lambda)$ be a probability space, $P$ be the uniform distribution over the bounded set $A \subset \mathbb{R}$ and $X$ be a random variable with $X(\omega) = \hat{\theta}_{f,g}(\omega)$ for all $\omega \in \Omega$, then we have that

$$m_{\text{Var}(\theta)}|A(f, g) = \text{Var} X.$$ 

This justifies the name "variance distance measure".
4. Numeric computation of the distance measures

In this chapter we consider the issues occured whilst computing the metrics. The order of the distance measures considered is the same as in chapter 3, so we first consider the numerical computation of the $L^p$-measures followed by the numerical computation of the Laplace Transform based distance measures.

4.1. Numeric computation of $L^p$-norms and $m_p$-measures

For its simplicity, at first just the plain Trapezoidal Rule has been used to estimate the integral in (2.1). Experiments using MATLAB showed that the error of the Trapezoidal Rule is acceptable taken to account that the computed distances are only estimates whether two probability density functions are some kind of similar.

The reason for choosing the Trapezoidal Rule also lies in the fact, that the performance prediction functions in question usually come from histograms. Now probability density functions created by a histogram are exactly integrated by the Trapezoidal Rule by construction, since the probability density is normed in such a way that the addition is exactly one.

Furthermore, the performance prediction functions are only known at some points that we can hardly influence. Basically, this means that adaptive integration algorithms cannot be applied to this case, since additional points cannot be generated.

4.2. Numeric computation of the Laplace Transform induced measures

Computing the Laplace Transform induced measures, a numerical integration to compute the Laplace Transform is only one of the issues evaluating the measures. For the variance distance measure, it is also necessary to approximate the variance, which is usually also an integral. Furthermore, the limit necessary for the front and tail distance measures cannot exactly be evaluated. The remark 3.24 also implies the necessity of a bounded domain for theta.
We also face the problem that for any $f \in C^c(\mathbb{R})$ with positive support
\begin{equation}
\mathcal{B}(f)(\xi) = \int_{\mathbb{R}} e^{-\xi x} f(x) dx \leq e^{-\xi \mathcal{E}(f)} \to 0 \quad \text{as} \quad \xi \to \infty
\end{equation}
and
\begin{equation}
\mathcal{B}(f)(\xi) = \int_{\mathbb{R}} e^{-\xi x} f(x) dx \geq e^{-\xi \mathcal{E}(f)} \to \infty \quad \text{as} \quad \xi \to -\infty.
\end{equation}
As the convergence of the exponential function to either infinity or zero is very quickly, we will get a problem trying to evaluate $\tilde{\theta}_{f,g}$ for large values of $\xi$. On the other hand, the computation of $\tilde{\theta}_{f,g}(\xi)$ involve the division by $\xi$. Since the numerical condition of the division gets worse as the divisor goes to zero, we have to spare out an area around zero for the domain of $\tilde{\theta}_{f,g}$. In addition to that, both $\mathcal{B}_f(\xi)$ and $\mathcal{B}_g(\xi)$ converge to 1 as $\xi \to 0$ so that also the numerator goes to zero. In experiments, the domain $[-50, -1] \cup [1, 50]$ showed good results.

For the "large numbers" for both front and tail measure, we also choose the rims of this domain, 50. As we need the derivative of $\tilde{\theta}_{f,g}$, we simply take the difference quotient. As we are not bound to a specific value of $\xi$ and (using the mean value theorem) the difference quotient exactly equals the derivative at some intermediate point, this will do.

A bit more complicated is the limit $\lim_{\xi \to 0} \tilde{\theta}_{f,g}(\xi)$ as for the sake of the bad numerical condition of the computation of $\tilde{\theta}_{f,g}$ near zero it is not reasonable to get values of $\tilde{\theta}_{f,g}$ close to zero. Thus, we will need to use polynomial interpolation. For the sake of simplicity, I simply used the linear interpolation here, so we will approximate
\begin{equation}
\tilde{\theta}_{f,g}(0) \approx \frac{\tilde{\theta}_{f,g}(1) + \tilde{\theta}_{f,g}(-1)}{2}
\end{equation}
The numerical analysis of the goodness of this way to evaluate the distance measures remains subject of future work.
5. Evaluation

In this section, we discuss the metrics resulting from chapter 3. At first, we evaluate the metrics induced by the various $L^p$-norms and afterwards, we discuss the goodness of fit of the Laplace-transform-induced metrics. We use the distributions listed in 5.1 for reference. These distributions are mostly the same as used in the internal worksheet mentioned in section 1.3. The second normal distribution is slightly different to have two distributions that are only shifted.

![Figure 5.1: Distributions](image)

All results in this chapter have been computed using MATLAB 7.11.0 (R2010b). The probability density functions have been approximated using MATLAB's own functions to
5. Evaluation

approximate the probability densities for any of the distributions. As all of the sample
probability density functions have infinite support, the support has been cut off to the
interval $[0, 8]$. In this interval, I used 1000 intermediate points $x_1 = 0, \ldots, x_{1000} = 8$.

I make a short rating of the goodness of each distance measure, but these ratings are only
estimations. At the end, the usefulness of the computed results of any metric does depend
on the use case and therefore has to be reevaluated when using them in any other context.
Furthermore, these ratings of the usefulness are quite subjective and may be different
from different points of view. This is mainly because an objective view on the goodness
of fit of any of these distance measures would require a formal definition of similarity of
different distribution functions. Taken into account that different scenarios might also
require different definitions of a similarity of probability distributions, it is not possible to
evaluate the goodness of any distance measure without loosing generality.

5.1. $L^p$-norms

In this subsection, the $L^p$ norms from chapter 2.1 are covered. As every value for $p$ induces
a distance measure, we obviously have to limit ourselves to just a few of them. So in this
Bachelor thesis, only the values 1, 2 and $\frac{1}{2}$ are subject of consideration.

For the computations, the appearing integrals have been computed using the Trapezoidal
rule. From the common distributions, 101 equidistant points have been used to approxi-
mate the probability density function.

The results for the $L^1$-norm, $L^2$-norm and $L^{\frac{1}{2}}$-measure are shown in the tables 5.2, 5.1
and 5.3

<table>
<thead>
<tr>
<th></th>
<th>$L^1$</th>
<th>$L^2$</th>
<th>$L^{\frac{1}{2}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.2320</td>
<td>0.4576</td>
<td>1.1392</td>
</tr>
<tr>
<td>0.2320</td>
<td>0</td>
<td>0.6217</td>
<td>1.0511</td>
</tr>
<tr>
<td>0.4576</td>
<td>0.6217</td>
<td>0</td>
<td>1.3713</td>
</tr>
<tr>
<td>1.1392</td>
<td>1.0511</td>
<td>1.3713</td>
<td>0</td>
</tr>
<tr>
<td>1.2013</td>
<td>1.2678</td>
<td>1.0762</td>
<td>1.7905</td>
</tr>
<tr>
<td>0.3295</td>
<td>0.2941</td>
<td>0.5727</td>
<td>1.2188</td>
</tr>
<tr>
<td>1.2022</td>
<td>1.0421</td>
<td>1.4044</td>
<td>1.4044</td>
</tr>
<tr>
<td>0.9058</td>
<td>0.7829</td>
<td>1.1233</td>
<td>1.4370</td>
</tr>
</tbody>
</table>

Table 5.1.: $L^2$-norm results

Setting $p = 2$ turned out to be worst of these three values for $p$, since mainly the points
of extreme differences are considered. This leads to the problem that the norm cannot
distinguish between a dispersion and a completely different function. For example, the first f-noncentral distribution (solid red) can approximately be considered as being a dispersed version of the first log-normal distribution (solid blue). These distributions seem to fit much better than again the solid red f-noncentral distribution compared with the dashed green normal distribution. However, the $L^2$-distance is even smaller. This is a great example of how the $L^2$-norm does not coincide with intuition and thus, it does not fit the requirements.

The result table shows that a $L^2$ distance of 0.5 or less indicate that the two functions in question are quite similar, whereas a distance of above 1 indicate that the functions do not coincide. The distance measure does only check whether the distribution match exactly, e.g. it does not recognize the similarity of the two normal distributions that are only shifted.

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>0.3785</th>
<th>0.6248</th>
<th>1.7960</th>
<th>1.2731</th>
<th>0.4594</th>
<th>1.2715</th>
<th>0.9602</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.3785</td>
<td>0</td>
<td>0.9137</td>
<td>1.6994</td>
<td>1.4055</td>
<td>0.4689</td>
<td>1.1250</td>
<td>0.8774</td>
<td></td>
</tr>
<tr>
<td>0.6248</td>
<td>0.9137</td>
<td>0</td>
<td>1.9946</td>
<td>0.9864</td>
<td>0.7135</td>
<td>1.4087</td>
<td>1.1518</td>
<td></td>
</tr>
<tr>
<td>1.7960</td>
<td>1.6994</td>
<td>1.9946</td>
<td>0</td>
<td>2.0000</td>
<td>1.8893</td>
<td>1.9796</td>
<td>1.9325</td>
<td></td>
</tr>
<tr>
<td>1.2731</td>
<td>1.4055</td>
<td>0.9864</td>
<td>2.0000</td>
<td>0</td>
<td>1.1634</td>
<td>1.5835</td>
<td>1.4079</td>
<td></td>
</tr>
<tr>
<td>0.4594</td>
<td>0.4689</td>
<td>0.7135</td>
<td>1.8893</td>
<td>1.1634</td>
<td>0</td>
<td>0.9688</td>
<td>0.6229</td>
<td></td>
</tr>
<tr>
<td>1.2715</td>
<td>1.1250</td>
<td>1.4087</td>
<td>1.9796</td>
<td>1.5835</td>
<td>0.9688</td>
<td>0</td>
<td>0.5501</td>
<td></td>
</tr>
<tr>
<td>0.9602</td>
<td>0.8774</td>
<td>1.1518</td>
<td>1.9325</td>
<td>1.4079</td>
<td>0.6229</td>
<td>0.5501</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.2.: $L^1$-norm results

The situation for the $L^1$-norm is slightly better. However, also in this case, there are some examples that doubt the usability of this norm. For instance, again the first f-noncentral distribution and the first log-normal distribution have a quite high distance measure, although this distance is less than the distance between the f-noncentral and the second normal distribution. Sadly, there are also examples how the upper bound of the $L^1$-norm is broken by experiments, as for instance the distance between the second normal distribution and any of the two featured pareto distributions.

Since 2 is a theoretically (that means without errors) upper bound for the $L^1$-distance comparing two probability distributions, naturally values near 2 mean that the functions do not have much in common. Again, the $L^1$-distance is only a measure of how the two functions in question match to each other exactly. With a value of 1.9946, the distance of the two normal distributions are almost at the theoretically upper bound.

So far, the $L^2$-distance measure appears to be the best distance measure for comparing distribution functions, although it is the only one yet evaluated that is not a metric. Un-
like the $L^1$- or $L^2$-norm, the $L^\frac{1}{2}$ distance measure clearly separates the second normal distribution from all the other example distributions.

Interestingly, the $L^\frac{1}{2}$-distance measure also takes into account the tail of a distribution. Both of the two heavy tailed pareto distributions do have a quite high distance to any of the other distributions (except for the distance of the second pareto distribution and the second log-normal-distribution). This is mainly because the $L^\frac{1}{2}$ measure takes much more into account the measure of the support rather than the supremum of the difference function.

### 5.2. Laplace Transform

In this section, we cover the distance measures defined in $3.2$.

Computing any of the Laplace Transform based distance measures, the domain $A$ of $m_{\text{Val}(\hat{\theta})}(f,g)$ (see $3.25$) has been to $[-50,-1] \cup [1,50]$ according to $1.2$. To compute the Laplace Transforms, I used the Trapezoidal Rule with 1000 intermediate points as this showed good results. To compute the variance, I have taken 200 values out, 100 equidistant values out of both $[-50,-1]$ and $[1,50]$, $t_1 = -50, \ldots, t_{100} = -1, t_{101} = 1, \ldots, t_{200} = 50$. Using these numbers, I approximated the distance measures as follows:

Table 5.3: $L^\frac{1}{2}$-measure results

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1.8354</th>
<th>2.0074</th>
<th>6.0443</th>
<th>3.4094</th>
<th>1.4268</th>
<th>3.3927</th>
<th>2.6711</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.8354</td>
<td>0</td>
<td>3.9218</td>
<td>7.0502</td>
<td>5.1274</td>
<td>2.2619</td>
<td>4.2800</td>
<td>3.3865</td>
<td></td>
</tr>
<tr>
<td>2.0074</td>
<td>3.9218</td>
<td>0</td>
<td>5.7547</td>
<td>1.3830</td>
<td>2.0431</td>
<td>2.4231</td>
<td>2.6726</td>
<td></td>
</tr>
<tr>
<td>6.0443</td>
<td>7.0502</td>
<td>5.7547</td>
<td>0</td>
<td>4.0716</td>
<td>6.2906</td>
<td>5.1747</td>
<td>5.7855</td>
<td></td>
</tr>
<tr>
<td>3.4094</td>
<td>5.1274</td>
<td>1.3830</td>
<td>4.0716</td>
<td>0</td>
<td>2.8978</td>
<td>2.1969</td>
<td>2.7941</td>
<td></td>
</tr>
<tr>
<td>1.4268</td>
<td>2.2619</td>
<td>2.0431</td>
<td>6.2906</td>
<td>2.8978</td>
<td>0</td>
<td>2.4078</td>
<td>1.5148</td>
<td></td>
</tr>
<tr>
<td>3.3927</td>
<td>4.2800</td>
<td>2.4231</td>
<td>5.1747</td>
<td>2.1969</td>
<td>2.4078</td>
<td>0</td>
<td>1.3156</td>
<td></td>
</tr>
<tr>
<td>2.6711</td>
<td>3.3865</td>
<td>2.6726</td>
<td>5.7855</td>
<td>2.7941</td>
<td>1.5148</td>
<td>1.3156</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
\[
\mu = \frac{1}{200} \sum_{n=1}^{200} \bar{\theta}_{f,g}(t_n)
\]

\[
m_{\text{Var}(\theta)}(A(f, g)) \approx \frac{1}{200} \sum_{n=1}^{200} (\bar{\theta}_{f,g}(t_n) - \mu)^2
\]

\[
m_{\text{mw}}(f, g) \approx \frac{\bar{\theta}_{f,g}(t_{100}) + \bar{\theta}_{f,g}(t_{101})}{2}
\]

\[
m_{\mathcal{F}}^{(50)}(f, g) \approx \frac{\bar{\theta}_{f,g}(t_{200}) - \bar{\theta}_{f,g}(t_{199})}{2}
\]

\[
m_{\mathcal{T}}^{(50)}(f, g) \approx \frac{\bar{\theta}_{f,g}(t_2) - \bar{\theta}_{f,g}(t_1)}{2}
\]

In the MATLAB implementation, the variance distance measure equals the sample variance of the sample set \(t_1, \ldots, t_{200}\). Thus, the MATLAB operator \texttt{var} is used here.

| \[0\] | \[0.1761\] | \[5.4024\] | \[4.6174\] | \[5.9055\] | \[0.0286\] | \[0.0333\] | \[0.0962\] |
| \[5.4024\] | \[0.1761\] | \[6.6062\] | \[5.7143\] | \[7.3166\] | \[0.0791\] | \[0.1735\] | \[0.0367\] |
| \[4.6174\] | \[5.7143\] | \[0.0983\] | \[0\] | \[0.6116\] | \[5.0420\] | \[5.0738\] | \[5.2747\] |
| \[5.9055\] | \[7.3166\] | \[0.4514\] | \[0.6116\] | \[0\] | \[6.4283\] | \[6.3207\] | \[6.7169\] |
| \[0.0286\] | \[0.0791\] | \[5.8518\] | \[5.0420\] | \[6.4283\] | \[0\] | \[0.0235\] | \[0.0232\] |
| \[0.0333\] | \[0.1735\] | \[5.8368\] | \[5.0738\] | \[6.3207\] | \[0.0235\] | \[0\] | \[0.0689\] |
| \[0.0962\] | \[0.0367\] | \[6.0802\] | \[5.2747\] | \[6.7169\] | \[0.0232\] | \[0.0689\] | \[0\] |

Table 5.4.: variance measure results

In the variance distance measure of the two normal distributions, the results show how the effect that this distance measure is invariant under translation. In fact, the experiments with 1000 points in the domain of the Laplace Transform and 1001 points to compute the Laplace Transform showed differences for \(\bar{\theta}_{f,g}\) of only 0.01 as maximum. Thus, the results for the variance measure clearly show that the solid green distribution is just the dashed green distribution, only shifted by (see \(5.5\)) 1.800 to the left.

The exactness of this distance measure may be indicated by the distance of the two normal distributions. MATLAB computes it to 1.799950325540008, which is pretty near the correct value of 1.8. As the normal distributions are not exactly shifted (as they are both cut off at the ends), this is very good. Interestingly, “plainly” computing the difference of
Table 5.5.: mean value results

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>0.3363</th>
<th>-0.2782</th>
<th>1.5217</th>
<th>-0.4703</th>
<th>-0.2602</th>
<th>-0.8115</th>
<th>-0.3487</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.3363</td>
<td>0</td>
<td>-0.6145</td>
<td>1.1855</td>
<td>-0.8066</td>
<td>-0.5965</td>
<td>-1.1477</td>
<td>-0.6850</td>
<td></td>
</tr>
<tr>
<td>0.2782</td>
<td>0.6145</td>
<td>0</td>
<td>1.8000</td>
<td>-0.1921</td>
<td>0.0180</td>
<td>-0.5332</td>
<td>-0.0705</td>
<td></td>
</tr>
<tr>
<td>-1.5217</td>
<td>-1.1855</td>
<td>-1.8000</td>
<td>0</td>
<td>-1.9920</td>
<td>-1.7819</td>
<td>-2.3332</td>
<td>-1.8704</td>
<td></td>
</tr>
<tr>
<td>0.4703</td>
<td>0.8066</td>
<td>0.1921</td>
<td>1.9920</td>
<td>0</td>
<td>0.2101</td>
<td>-0.3412</td>
<td>0.1216</td>
<td></td>
</tr>
<tr>
<td>0.2602</td>
<td>0.5965</td>
<td>-0.0180</td>
<td>1.7819</td>
<td>-0.2101</td>
<td>0</td>
<td>-0.5513</td>
<td>-0.0885</td>
<td></td>
</tr>
<tr>
<td>0.8115</td>
<td>1.1477</td>
<td>0.5332</td>
<td>2.3332</td>
<td>0.3412</td>
<td>0.5513</td>
<td>0</td>
<td>0.4628</td>
<td></td>
</tr>
<tr>
<td>0.3487</td>
<td>0.6850</td>
<td>0.0705</td>
<td>1.8704</td>
<td>-0.1216</td>
<td>0.0885</td>
<td>-0.4628</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

the mean values of the normal distributions by computing the sample mean values

\[
\mu_g - \mu_f = \frac{8 - 0}{1000} \sum_{n=1}^{1000} x_n g(x_n) - \frac{8 - 0}{1000} \sum_{n=1}^{1000} x_n f(x_n)
\]

ended up in the worse result 1.798199659141265. The difference comes from the fact that the two normal distributions are cut at different positions in relation to their center (mean value). However, at least in this example, the result of \( m_{m_w}(f, g) \) is even better than the "plain" one, in the sense that it decreases the influence of the inexactness made by cutting the distribution.

The fact that the highest absolute values for the front measure appear in comparison with the second normal distribution does confirm that the front measure measures the probability mass away from the center (mean value). Unfortunately, it seems like the front measure mainly measures the length of the supports rim to the center rather than anything else. Thus, it does not seem to be suitable to measure a dispersion at the front of the distribution. However, the performance prediction induced probability distributions do not have an infinite front with such a low probability density that one would like to approximate to zero. The effect also might come from the fact that the fronts of the featured distributions all are quite short in comparison to their tails.

Unlike the front measure, the tail measure results confirm the intuition that they would measure heavy tails. Both of the normal distributions and the first lognormal distribution do not have a heavy tail (at least in comparison with the other featured ones). The tail measure of one of them compared with the rest always has an absolute value of about 20 and the sign indicates which distribution is the heavy tailed one: A positive value of the distribution \( f \) in the row and \( g \) in the column means that \( m_T^{(50)}(f, g) \) is positive and thus, \( g \) is more heavy-tailed.
Interestingly, the front and tail measure seem additive, i.e. $m^{(50)}_F(f, g) + m^{(50)}_F(g, h) = m^{(50)}_F(f, h)$ respectively $m^{(50)}_T(f, g) + m^{(50)}_T(g, h) = m^{(50)}_T(f, h)$ for all featured probability distributions $f, g, h$. This might be an aspect of future work.
<p>| | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.3024</td>
<td>-18.8790</td>
<td>-18.6009</td>
<td>-24.8306</td>
<td>0.6311</td>
<td>0.2637</td>
<td>1.1172</td>
<td></td>
</tr>
<tr>
<td>-1.3024</td>
<td>0</td>
<td>-20.1814</td>
<td>-19.9033</td>
<td>-26.1330</td>
<td>-0.6713</td>
<td>-1.0387</td>
<td>-0.1852</td>
<td></td>
</tr>
<tr>
<td>18.8790</td>
<td>20.1814</td>
<td>0</td>
<td>0.2781</td>
<td>-5.9516</td>
<td>19.5101</td>
<td>19.1427</td>
<td>19.9962</td>
<td></td>
</tr>
<tr>
<td>18.6009</td>
<td>19.9033</td>
<td>-0.2781</td>
<td>0</td>
<td>-6.2297</td>
<td>19.2320</td>
<td>18.8646</td>
<td>19.7182</td>
<td></td>
</tr>
<tr>
<td>24.8306</td>
<td>26.1330</td>
<td>5.9516</td>
<td>6.2297</td>
<td>0</td>
<td>25.4616</td>
<td>25.0942</td>
<td>25.9478</td>
<td></td>
</tr>
<tr>
<td>-0.6311</td>
<td>0.6713</td>
<td>-19.5101</td>
<td>-19.2320</td>
<td>-25.4616</td>
<td>0</td>
<td>-0.3674</td>
<td>0.4862</td>
<td></td>
</tr>
<tr>
<td>-0.2637</td>
<td>1.0387</td>
<td>-19.1427</td>
<td>-18.8646</td>
<td>-25.0942</td>
<td>0.3674</td>
<td>0</td>
<td>0.8536</td>
<td></td>
</tr>
<tr>
<td>-1.1172</td>
<td>0.1852</td>
<td>-19.9962</td>
<td>-19.7182</td>
<td>-25.9478</td>
<td>-0.4862</td>
<td>-0.8536</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.7.: tail measure results
6. Implementation

In this chapter, we discuss some of the relevant issues implementing the featured distance measures. The whole chapter only takes into account the implementation in Java, as the MATLAB implementation is for demonstration and testing purposes only. Thus, I do not discuss it from the point of software engineering, as the quality of this code is, as intended, not the best.

6.1. Overview

The implementation is made upon two important interfaces. The first one is the interface for a distance measure, `IMetric`, that takes two `IProbabilityDensityFunctions` and computes a set of `IMetricResults`. The `IProbabilityDensityFunction` interface originally comes from the ProbFunction framework and defines a probability density function. These probability density functions either can be represented by samples, as known continuous density functions or by boxes. A factory class inside the ProbFunction model
can transform the representations into each other. The reason why a whole collection of IMetricResult is created, is for performance reasons. Several single distance measures might share intermediate results which might be hard to calculate. Designing the compute method as returning a collection of IMetricResults, it is easy to share intermediate results. In addition to that, an IMetric implementation instance could also just leave out a distance measure without throwing an exception, e.g. if it somehow did not make sense. The compute method remains the only operation within the IMetric interface. The second interface, IMetricResult, just contains getter functions for the actual distance value and for a description that is meant to be used like a key in order to distinguish the different distance measures computed by an IMetric instance. The fact that any metric result can be accessed only by iterating through a collection or an index access, is a weakness of this approach, but returning a single metric result had restricted the implementations too strictly. There is a default implementation of IMetricResult, namely MetricResult, since this class is basically for simply holding the result values. The more important interface IMetric has two implementations within this Bachelor thesis, LpMetric and LaplaceMetric. Both of them will be explained in their own sections, LpMetric in section 6.2 and LaplaceMetric in section 6.1.

Figure 6.2.: ProbFunction model changes

To get the implementation done easily, there are also modifications that have been made to the ProbFunction-model itself. There have been introduced two new interfaces to edit any given IProbabilityDensityFunction, IUnaryOperation and IBinaryOperation that represent unary and binary point-wise operations on functions. So basically, what they do is that they take the values of one (unary) or two (binary) functions and the point where this value comes from and produce a new value for this point. This is a clear breach of the name, since any operation point-wise applied to a probability density function does not necessarily produce a probability density function as the resulting function does not necessarily fulfill the nonnegativeness or the property that the integral is exactly one. But, this breach has been made actually before this Bachelor thesis, since the IProbabilityDensityFunction already supported add, subtract, multiply and division operations. The approach using the new interfaces IUnaryOperation and IBinaryOperation just makes this breach more extensible, since now any operation is supported without further changes in the code of the classes implementing IProbabilityDensityFunction. For convenience, but mainly to avoid the need of having to change existing code, the add, sub, mult and div oper-
ation of the IProbabilityDensityFunction are still in the interface, but for the sake of avoidance of code redundancy, their implementations have been changed to use the IBinaryOperation-implementations AddOperation, SubtractOperation, MultiplyOperation and DivisionOperation. The implementations of all these classes simply ignore the third parameter xi. There is a static pointer of instances of each of these classes, so it is not necessary to create new instances every time one wants to use them. This is valid since all of these classes are implemented stateless and thus threadsafe.

6.2. $L^p$ spaces

![Image of Lp implementation]

Figure 6.3.: Lp implementation

The $L^p$ measures were implemented quite easily, as the class diagram and the sequence diagram 6.4 can show. What the LpMetric compute method simply does, is to apply a power operation onto the two functions that should be compared, and integrates the resulting function. The power operation is designed to be both a unary and binary operation. The unary version just returns the absolute value taken to the power of a specified parameter p. The problem with this solution lies in its larger memory leak. The implementation of the apply function is to create an entirely new SamplePDF instance. Thus, the binary compute method just applies the unary compute method to the difference of the function values. But still, there is an instance of a newly created SamplePDF that is created as an intermediate value and dropped after calculation (of course, the exact time of destruction is subject to the Garbage Collector, but anyway). The issue here is that there is a large amount of memory taken, all the data has to be copied and released afterwards. That might be a performance leak.

From the view of software engineering, though, this solution is quite good for its simplicity and the possible reusability of the PowerOperation. However, the implementation was finally made as drawn in the UML diagrams, using the binary version, but I left the PowerOperation to implement also the IUnaryOperation for a possible reuse.

Since the numerical aspect of approximating the exactness of the integral in the $L^p$ measures has not been reviewed, I extrapolated the integration into a new component, accessible through the interface INumericalIntegrationRule. This also can be seen as a continued perversion of the name ”IProbabilityDensityFunction”, since the integration over whole $\mathbb{R}$ of a true probability density function would be as simple as writing the
constant 1, but since a \texttt{IProbabilityDensityFunction} in the context of this Bachelor thesis rather stands for a generic function represented by samples, it is valid to approximate its integral.

![Diagram](image)

Figure 6.4.: Lp compute sequence diagram

6.3. Laplace Transform induced measures

The implementation of the Laplace Transform induced measures was a slightly more complicated, as you can see on the sequence diagram 6.5. As all of the Laplace Transform induced measures somehow involve the $\theta_{f,g}$ function, it is a central issue of implementing these measures to compute and somehow represent this function.

Since the variance distance measure is approximated by the approximation of a limited sample, we store the computed $\theta_{f,g}$ in a list data structure. This decision was made since we do not need any functionality of any of the \texttt{IProbabilityDensityFunction} interfaces \texttt{ISamplePDF} or \texttt{IBoxedPDF} and implementing \texttt{IContinuousPDF} would induce a large effort in implementation.

The computation of $\theta_{f,g}$ splits up into two methods. One that computes $\theta_{f,g}(\xi)$ for a given value of $\xi$ and one that combines these values to a whole function. The first mentioned function, \texttt{computeTheta(IProbabilityDensityFunction f, IProbabilityDensityFunction g, double xi)} computes the value for $\theta_{f,g}(xi)$. The sequence diagram for this function is part of figure 6.6 (in figure 6.6 any loops have been ignored as this would blow up the diagram). The computeTheta-method uses a \texttt{LaplaceOperation}, which is an implementation of the interface \texttt{IUnaryOperation}, as well as a numerical integration rule to compute the bilateral Laplace transformation of $f$ and $g$.

The overloaded function \texttt{computeTheta(IProbabilityDensityFunction f, IProbabilityDensityFunction g)} creates a list out of these values. This is done with help of
6.3. Laplace Transform induced measures

ThetaDomain, which is a class implementing `Iterable<Double>` that iterates through the intended domain (in the sense of 3.25) for \( \hat{\theta}_{f,g} \). As the interface implies, ThetaDomain iterates the intermediate values from \(-\max\) to \(-\min\) with steps `step` and afterwards from \(\min\) to \(\max\). If \(\max - \min\) is not a multiple of `step`, ThetaDomain iterates to

\[
\theta_- = \max\{-\max + k \cdot \text{step} : k \in \mathbb{N}_0, -\max + k \cdot \text{step} \leq -\min\}
\]

and then moves next to \(-\theta_- \geq \min\), so that the resulting domain is symmetrical. This makes the implementation of the computeTheta\((f,g)\)-method as easy as converting the ThetaDomain by the converter computeTheta\((f,g,xi)\).

The other methods of LaplaceMetric (apart from the compute-method) are helper functions to compute the sample mean value or sample variance of a list of points. Obviously, `getVariance` uses `getMeanValue`. The compute-method implementing the IMetric interface computes the \( \hat{\theta}_{g,g} \) function as a list of `double` values and deduct the distance measures from that list.
Figure 6.6.: Laplace Transform compute sequence diagram
7. Conclusion

In this section, we review of what has been achieved. The first section 7.1 covers the results from this Bachelor thesis. In the following section 7.3 we discuss what remains to work on.

7.1. Results

The result of this Bachelor thesis mainly is a set of six distance measures as featured in chapter 3. The $L^p$-metrics provide some kind of an absolute distance. They pretty much ignore the shape of the function, they just recognize the absolute difference function. A value for $p$ near infinity means the metric concentrates on the biggest differences, whereas a value near zero rather concentrate on the support of the difference function. The distance measures based on the Laplace Transform are (with the exception of the mean value measure) independent under translation, so that they really focus on the shape. The front and tail measures especially concentrate on the behaviour on the rim of the support of the function, meanwhile the variance distance measure focusses on the shape at all.

In addition to the results in theory, both a MATLAB and a Java implementation of these metrics are results of this work.

7.2. Related Work

There are also other tools to predict response times of a system. Another example of a performance modelling formalism with a component-driven approach is PEPA. Also this model is designed to support a variety of different software systems and it also uses stochastic processes via Fluid Analysis to retrieve response times (e.g. continuous-time Markov chains) \cite{BHK08}.

A popular other way of getting a distance of two distributions that also can be multidimensional is the Earth mover’s distance (EMD), but it is very slow and it is usually applied to discrete probability spaces, only. A possible solution lies in the usage of the kernel norm

$$D_K(P, Q) = \sqrt{\kappa(P, P) + \kappa(Q, Q) - 2\kappa(P, Q)}$$
where
\[ \kappa(\mathcal{P}, \mathcal{Q}) = \sum_{p \in \mathcal{P}} \sum_{q \in \mathcal{Q}} K(p, q) \]

for any subsets \( \mathcal{P}, \mathcal{Q} \subset \mathbb{R}^d \) for any symmetric kernel function \( K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R} \). As these distances are intended to be used with points, one would most likely apply these distances for a set of points. As the result of a performance prediction system as Palladio is most likely a probability distribution that is approximated by a set of points, this is no problem. However, one has to check which kernel functions to choose.

The approach of using kernel metrics is more general than the one that is proposed here. However, the distance measures introduced in this thesis work on probability distributions with a probability density with respect to the Lebesgue-measure and thus with a cumulative distribution function that is continuously differentiable. As the kernel metrics work on sets of points, there are designed for usage in discrete probability spaces, i.e. having a non-continuous cumulative distribution function.

For more detail, see [JKPV11]. This article also focusses on the algorithmic analysis of computing these kernel distances.

### 7.3. Future Work

Like already mentioned earlier in this work, I did not spend time to prove the distance measures of chapter 3.2 to be metrics in a mathematical way. Obviously, they are not definite, as mean value distance measure, front distance measure and tail distance measure may attend negative values, but it might be useful to know whether taking the absolute value would turn them to (in the sense of metric spaces) true metrics or semi-metrics. The main part here is the triangular inequality and to characterize the equivalence classes induced by the distance measure as semi-metric.

There is some evaluation done in this work yet, but the main evaluation of the distance measures featured in this work will remain to the experience when using them. The featured measures will also not provide one with all the information there is about two probability functions, but it is something to work with. But also within these metrics, one might also want to do a more detailed analysis on the numerical computation of these metrics.

An example of a further metric deducted from the compare characteristics here could be the integration of the compare characteristics over a given interval. As the compare characteristics always states some kind of a difference of expected values, a positive value of \( \tilde{\theta}_{f,g} \) in the context of response times would always be preferable and thus, the integral over an interval would be a chance of getting a distance measure that could be used to make a decision of what component (for example) to use, including some considerations on outliers on the tail (to make the outliers in the tail more important than the outliers at the tail, one could simply use an asymmetric interval.

But this thesis might also be basement to future work in stochastics: Most of the propositions and theorems in chapter 3.2 can be generalized to tear down the requirement of
the two distributions in question to be probability measures that are absolute continuous regarding the Lebesgue-measure. What we used as bilateral Laplace Transform here could easily be substituted by obtaining the expected value for some random variables with any real-valued distribution. By real-valued distribution, I explicitly also mean discrete distributions with natural values, such as a Poisson distribution. The compare characteristic also makes sense, if the distributions in question do not work on the same base set, i.e., if one of them is discrete while the other one is continuous. Thus, the compare characteristic may yield a metric between probability spaces.
Bibliography


Appendix

A. Matlab Implementation

In this section, I will provide the MATLAB implementation used to create the measurements.

A.1. Measurements.m

```matlab
thetaDomain = zeros(1, 200);
thetaDomain(1,1:100) = linspace(-50, -1, 100);
thetaDomain(1,101:200) = linspace(1,50,100);
X = linspace(0, 8, 1001);

fnoncentr1 = ncfpdf(X, 10, 100, 4);
fnoncentr2 = ncfpdf(X, 5, 30, 2);

normal1 = normpdf(X, 1.2, 0.3);
normal2 = normpdf(X, 3, 0.3);

lognormal1 = lognpdf(X, 0, 1/8);
lognormal2 = lognpdf(X, 0, 1/2);

pareto1 = gppdf(X, 0.1, 0.3, 0.3);
pareto2 = gppdf(X, 0.2, 0.4, 0.4);

A = zeros(8, length(X));
A(1,:) = fnoncentr1;
A(2,:) = fnoncentr2;
A(3,:) = normal1;
A(4,:) = normal2;
A(5,:) = lognormal1;
A(6,:) = lognormal2;
A(7,:) = pareto1;
A(8,:) = pareto2;

Var = zeros(8,8);
Mw = zeros(8,8);
L1 = zeros(8,8);
L2 = zeros(8,8);
L05 = zeros(8,8);
Fronts = zeros(8,8);
Tails = zeros(8,8);
```
thetaN = length(thetaDomain);
for i = 1:8
  for j = i:8
    th = theta(X, thetaDomain, A(i,:),A(j,:));
    th2 = theta(X, thetaDomain, A(j,:),A(i,:));
    Var(i,j) = var(th);
    Var(j,i) = var(th2);
    Mw(i,j) = (th(thetaN / 2) + th(thetaN / 2 + 1)) / 2;
    Mw(j,i) = (th2(thetaN / 2) + th2(thetaN / 2 + 1)) / 2;
    L1(i,j) = lpnorm(X, A(i,:), A(j,:), 1);
    L1(j,i) = lpnorm(X, A(j,:), A(i,:), 1);
    L2(i,j) = lpnorm(X, A(i,:), A(j,:), 2);
    L2(j,i) = lpnorm(X, A(j,:), A(i,:), 2);
    L05(i,j) = lpnorm(X, A(i,:), A(j,:), 0.5);
    L05(j,i) = lpnorm(X, A(j,:), A(i,:), 0.5);
    Tails(i,j) = 1000*(th(2) - th(1));
    Tails(j,i) = 1000*(th2(2) - th2(1));
    Fronts(i,j) = 1000*(th(200) - th(199));
    Fronts(j,i) = 1000*(th2(200) - th2(199));
  end
end
clc

A.2. laplace.m

function lxi = laplace( Xdata, f, xi)
  lxi = 0;
  N = length(Xdata)-1;
  xk = Xdata(1);
  for j = 1:(N-1)
    xnext = Xdata(j+1);
    lxi = lxi + (exp(-xi*xk) * f(j) + exp (-xi*(xnext)) * f(j+1)) / 2;
    xk = xnext;
  end
  lxi = lxi * (Xdata(end)-Xdata(1)) / N;
end

A.3. lpnorm.m

function norm = lpnorm( Xdata, f, g, p )
  norm = 0;
  N = length(Xdata)-1;
  for j = 1:(N-1)
    norm = norm + (abs(f(j)-g(j))^p + abs(f(j+1)-g(j+1))^p) / 2;
  end
A. Matlab Implementation

end
norm = norm * (Xdata(end)-Xdata(1)) / N;
norm = norm ^ (1/p);
end

A.4. theta.m

function Ydata = theta(Xdata, domain, f, g)
    Ydata = zeros(1, length(domain));
    for k=1:length(domain)
        xk = domain(k);
        Ydata(k) = log(laplace(Xdata, f, xk) / laplace(Xdata, g, xk)) / xk;
    end
end