RISK-AVERSE DISTRIBUTIONAL REINFORCEMENT LEARNING

A CVAR OPTIMIZATION APPROACH

SILVESTR STANKO

Department of Computer Science
Faculty of Electrical Engineering
Czech Technical University

February, 2018

ABSTRACT

Short summary of the contents in English. . . a great guide by Kent Beck how to write good abstracts can be found here:

https://plg.uwaterloo.ca/~migod/research/beck00PSLA.html

ABSTRAKT

Český abstrakt

We have seen that computer programming is an art,
because it applies accumulated knowledge to the world,
because it requires skill and ingenuity, and especially
because it produces objects of beauty.

— **knuth:1974** [**knuth:1974**]

ACKNOWLEDGMENTS

Put your acknowledgments here.

Many thanks to everybody who already sent me a postcard!

Regarding the typography and other help, many thanks go to Marco Kuhlmann, Philipp Lehman, Lothar Schlesier, Jim Young, Lorenzo Pantieri and Enrico Gregorio[1], Jörg Sommer, Joachim Köstler, Daniel Gottschlag, Denis Aydin, Paride Legovini, Steffen Prochnow, Nicolas Repp, Hinrich Harms, Roland Winkler, Jörg Weber, Henri Menke, Claus Lahiri, Clemens Niederberger, Stefano Bragaglia, Jörn Hees, Scott Lowe, Dave Howcroft, and the whole LaTeX-community for support, ideas and some great software.

Regarding LyX: The LyX port was intially done by *Nicholas Mariette* in March 2009 and continued by *Ivo Pletikosić* in 2011. Thank you very much for your work and for the contributions to the original style.

[1] Members of GuIT (Gruppo Italiano Utilizzatori di TeX e LaTeX)

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LISTINGS

ACRONYMS

1

INTRODUCTION

A staple of an intelligent agent is the ability to reason and act in an environment over time, while working towards a desirable goal. This is the setting explored in reinforcement learning (RL), a branch of machine learning that focuses on dynamic decision making in an unknown environment.

Recent advances in artificial intelligence (AI) are encouraging goverments and corporations to deploy AI in high-stakes settings including driving cars autonomously, managing the power grid, trading on stock exchanges, and controling autonomous weapon systems. As the industry steps away from specialized AI systems towards more general solutions, the demand for safe approaches to artificial intelligence increases.

In this thesis, we tackle one aspect of safe reinforcement learning, robustness, by considering the risk involved in acting in a nondeterministic, noisy environment.

1.1 MOTIVATION

Lately, there has been a surge of successes in machine learning research and applications, ranging from visual object detection [15] to machine translation [3]. Reinforcement learning has also been a part of this success, with excelent results regarding human-level control in computer games [18] or beating the best human players in the game of Go [25]. While these successes are certainly respactable and of great importance, reinforcement learning still has a long way to go before being applied on critical real-world decision-making tasks. This is partially caused by concerns of safety, as mistakes caused by can be costly in the real world.

One of the problems encountered when training a reinforcement learning agent is sample efficiency, or the large amount of training time needed for the agent to succesfully learn new and correct behaviors. The solution used by many is to train the agent in simulation - it is indeed faster (as the simulation can run in parallel or faster than real-time), safer (we do not face any real danger in simulations) and cheaper than to train the agent in the real world. This approach then raises the question if an agent trained in simulation would perform well outside of the simulation.

Robustness, or distributional shift, is one of the identified issues of AI safety [16][1] directly tied to the discrepancies between the environment the agent trains on and is tested on. Chow et al. [9] have shown that risk, a measure of uncertainty of the potential loss/reward, can be seen as equal to robustness, taking into account the differences during train- and test-time. This point is discussed in more detail in chapter 2.

While the term risk is a general one, we will focus on a concrete notion of risk - a particular risk metric called Conditional Value-at-Risk (CVaR). Due to it's favorable computational properties, CVaR has been recognized as the industry standard for measuring risk in finance, as in 2014 the Basel Committee on Banking Supervision changed its guidelines for banks to replace VaR (a previously used metric) with CVaR for assessing market risk Committee [10]. The metric has also been identified as one of the metrics satisfying axioms of risk in robotics [17].

Another motivational point, aside from robustness, might be one of general decision-making. Commonly encountered in finance, decision makers face the problem of maximizing profits while keeping the risks to a minimum. The solutions to problems encountered in this thesis can therefore be seen as ones of general time-dependent risk-averse decision making.

The aim of this thesis is to consider reinforcement learning agents that maximize Conditional Value-at-Risk instead of the usual expected value, hereby learning a robust, risk-averse policy. The word *distributional* in the title emphasizes that our approach takes inspirations from, or directly uses, recent advances in distributional reinforcement learning [5].

1.2 THESIS OUTLINE

1.3 CONTRIBUTIONS

2

PRELIMINARIES

The goal of this chapter is to provide a formal background on the covered material, together with a unified notation (which differs quite a lot from publication to publication). After we establish some basic theoretical foundations in Section 2.1, we remind the reader of the basic notions of reinforcement learning in Section 2.2 and of the recently explored and useful distributional reinforcement learning in Section 2.3. We follow up with the basics of risk together with the crucial CVaR measure in Section 2.4.

The interested reader is welcome to explore the books and publications referenced throughout this chapter and in Section 2.6. An informed reader may choose to skip to Section 2.5 where we formalize the problems tackled in this thesis.

2.1 PROBABILITY ESSENTIALS

random variable

exp value

todo: operator contraction

2.2 REINFORCEMENT LEARNING

The idea that we learn by interacting with our environment is probably the first to occur to us when we think about the nature of learning. Reinforcement learning [26] is a sub-field of machine learning that deals with time-dependent decision making in an unknown environment. The learner (often called agent) is not told which actions to take, but instead must discover which actions yield the most reward by trying them. In the most interesting and challenging cases, actions may affect not only the immediate reward but also subsequent situations and rewards. These two characteristics, trial-and-error search and delayed reward are the most important distinguishing features of reinforcement learning.

The general interaction between the agent and an environment can be seen in Figure 2.1. In each time-step t, the agent receives an observation x_t and a reward r_t and picks an action a_t and the process repeats. Below we formalize all the necessary notions of states and actions as a Markov Decision Process.

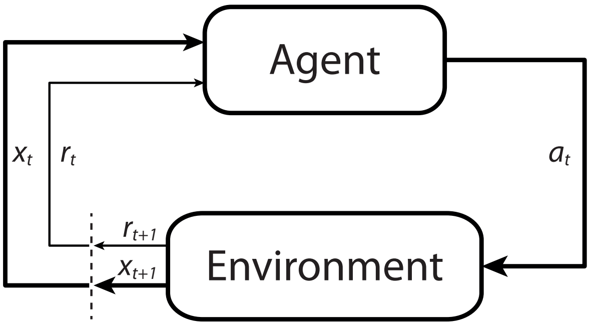

Figure 2.1: The Reinforcement learning cycle

2.2.1 *Markov Decision Processes*

Markov Decision Process (MDP, Bellman [6]) is a classical formalization of sequential decision making, where actions influence not just immediate rewards, but also subsequent situations, or states, and through those future rewards. They are a mathematically idealized form of the reinforcement learning problem for which precise theoretical statements can be made.

The word Markov points to the fact that we assume that the state transitions of an MDP satisfy the Markov property [**???**]. This means that the conditional probability distribution of future states of the process depends only upon the present state and not the whole history of events that preceded it.

Definition 1. *MDP is a 5-tuple $\mathcal{M} = (\mathcal{X}, \mathcal{A}, R, P, \gamma)$, where*
$\mathcal{X}$ *is the finite state space*
$\mathcal{A}$ *is the finite action space*
$R(x, a) \in [R_{\min}, R_{\max}]$ *is a random variable representing the reward generated by being in state x and selecting action a*
$P(\cdot|x, a)$ *is the transition probability distribution*
$\gamma \in [0, 1)$ *is a discount factor*

Definition 2. *A stationary (or markovian) policy is a mapping from states to actions $\pi : \mathcal{X} \to \mathcal{A}$.*

When solving MDPs with the usual discounted expected value criterion, it is common to limit the policy space to stationary policies, where the decision to take an action depends only on the current state. Unfortunately, when one considers other more general criteria, it is necessary to consider the whole action-state history that led up to the last state. This fact leads to the definition of a history-dependent policies.

todo: make the following readable

Definition 3. *Let the space of admissible histories up to time t be $H_t = H_{t-1} \times \mathcal{A} \times \mathcal{X}$ for $t \geq 1$, and $H_0 = \mathcal{X}$. A generic element $h_t \in H_t$ is of the form $h_t = (x_0, a_0, ..., x_{t-1}, a_{t-1})$. Let $\Pi_{H,t}$ be the set of all history-dependent policies with the property that at each time t the randomized control action is a function of h_t. In other words, $\Pi_{H,t} = \mathbb{R}aces\pi_0 : H_0 \to \mathbb{P}(\mathcal{A}), ..., \pi_t : H_t \to \mathbb{P}(\mathcal{A})|\pi_i(h_i) \in \mathbb{P}(\mathcal{A})\forall h_i \in H_i, 1 \leq i \leq t$. We also let $\Pi_H = \lim_{t\to\infty} \Pi_{H,t}$ be the set of all history-dependent policies.*

todo: as will become clear later, we can limit ourselves to an extended state spaces, not necessarily noting the whole history

2.2.2 *Return*

The reinforcement learning framework, generally considers either the natural discounted infinite-horizon return **todo:** few sentences on discounted/undiscounted returns

We define the return $Z^\pi(x)$ as a random variable representing the discounted reward along a trajectory generated by the MDP by following the policy π, starting at state x

$$\begin{gathered} Z^\pi(x) = \sum_{t=0}^{\infty} \gamma^t R(x_t, a_t) \\ x_t \sim p(\cdot|x_{t-1}, a_{t-1}), a_t \sim \pi, x_0 = x \end{gathered} \tag{2.1}$$

As a useful notation, we denote $Z^\pi(x, a)$ as the random variable representing the discounted reward along a trajectory generated by first selecting action a and then following policy π.

$$\begin{gathered} Z^\pi(x, a) = \sum_{t=0}^{\infty} \gamma^t R(x_t, a_t) \\ x_t \sim p(\cdot|x_{t-1}, a_{t-1}), a_t \sim \pi, x_0 = x, a_0 = a \end{gathered} \tag{2.2}$$

We will sometimes omit the superscript π when the policy is clear from the context.

2.2.3 *Bellman equation*

The *value function* V^π of a policy π describes the expected return received from state $x \in \mathcal{X}$ and acting according to π:

$$\begin{gathered} V^\pi(x) = \mathbb{E}\, Z^\pi(x) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(x_t, a_t)\right] \\ x_t \sim p(\cdot|x_{t-1}, a_{t-1}), a_t \sim \pi, x_0 = x \end{gathered} \tag{2.3}$$

The *action-value* function Q^π of a policy π describes the expected return from taking action $a \in \mathcal{A}$ from state $x \in \mathcal{X}$, then acting according to π:

$$\begin{gathered} Q^\pi(x, a) = \mathbb{E}\, Z^\pi(x, a) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(x_t, a_t)\right] \\ x_t \sim p(\cdot|x_{t-1}, a_{t-1}), a_t \sim \pi, x_0 = x, a_0 = a \end{gathered} \tag{2.4}$$

Fundamental to reinforcement learning is the use of Bellman's equation [6] to describe the value and action-value functions by a recursive relationship:

$$V^\pi(x) = \mathbb{E}\, R(x, \pi(x)) + \gamma \mathop{\mathbb{E}}_{p,\pi} V^\pi(x') \tag{2.5}$$

$$Q^{\pi}(x,a) = \mathbb{E}\, R(x,a) + \gamma \mathop{\mathbb{E}}_{p,\pi} V^{\pi}(x') \tag{2.6}$$

In reinforcement learning we are typically interested in acting so as to maximize the expected return. The most common approach for doing so involves the optimality equation

$$Q^{*}(x,a) = \mathbb{E}\, R(x,a) + \gamma\, \mathbb{E}_{p} \max_{a' \in \mathcal{A}} Q^{*}(x',a').$$

This equation has a unique fixed point Q^*, the optimal value function, corresponding to the set of optimal policies Π^* (π^* is optimal if $\mathbb{E}_{a\sim\pi^*} Q^*(x,a) = \max_a Q^*(x,a)$).

We view value functions as vectors in $\mathbb{R}^{\mathcal{X}\times\mathcal{A}}$, and the expected reward function as one such vector. In this context, the *Bellman operator* $\mathcal{T}^{\pi}$ and *optimality operator* $\mathcal{T}$ are

$$\mathcal{T}^{\pi} Q(x,a) := \mathbb{E}\, R(x,a) + \gamma \mathop{\mathbb{E}}_{P,\pi} Q(x',a') \tag{2.7}$$

$$\mathcal{T} Q(x,a) := \mathbb{E}\, R(x,a) + \gamma \mathop{\mathbb{E}}_{P} \max_{a' \in \mathcal{A}} Q(x',a') \tag{2.8}$$

These operators are useful as they describe the expected behaviour of popular learning algorithms such as SARSA and Q-Learning [26]. In particular they are both contraction mappings, and their repeated application to some initial Q_0 converges exponentially to Q^{π} or Q^*, respectively [7].

2.3 DISTRIBUTIONAL REINFORCEMENT LEARNING

In contrast to standard reinforcement learning, where we model the expected value of the return, in distributional reinforcement learning [**many**] we aim to model the full distribution of return. This is advantageous in cases where we want to e.g. model parametric uncertainty [**...**] or design risk-sensitive algorithms [20][19]. Bellemare, Dabney, and Munos [5] also argue, that the distributional approach is beneficial even in the case we are optimizing the expected value, as the distribution gives us more information which helps the now commonly used approximate algorithms (such as DQN [18]).

At the core of the distributional approach lies the recursive equation of the return distribution:

$$\begin{gathered} Z(x,a) \overset{D}{=} R(x,a) + \gamma Z(x',a') \\ x_t \sim p(\cdot|x_{t-1}, a_{t-1}), a_t \sim \pi, x_0 = x, a_0 = a \end{gathered} \tag{2.9}$$

where $\overset{D}{=}$ denotes that random variables on both sides of the equation share the same probability distribution.

In the *policy evaluation* setting [**sutton98reinforcement**] we are interested in the value function V^{π} associated with a given policy π. The analogue here is the value distribution Z^{π}. In this section we characterize Z^{π} and study the behaviour of the policy evaluation operator $\mathcal{T}^{\pi}$. Note that Z^{π} describes the intrinsic randomness of the agent's interactions with its environment, rather than some measure of uncertainty about the environment itself.

We view the reward function as a random vector $R \in \mathcal{Z}$, and define the transition operator $P^\pi : \mathcal{Z} \to \mathcal{Z}$ where we use capital letters to emphasize the random nature of the next state-action pair (X', A'). We define the distributional Bellman operator $\mathcal{T}^\pi : \mathcal{Z} \to \mathcal{Z}$ as

$$\mathcal{T}^\pi Z(x,a) \overset{D}{=} R(x,a) + \gamma P^\pi Z(x,a). \tag{2.10}$$

We emphasize that this is a distributional equation and the distributional bellman operator is therefore fundamentally different from the standard bellman operator.

Bellemare, Dabney, and Munos [5] have shown, that the distributional bellman operator $\mathcal{T}^\pi$ is not a contraction in the commonly used KL divergence [**...**], but is a contraction in the infinity wasserstein metric which we describe bellow, as it will become useful as a tool for evaluating algorithms in the rest of the thesis. Another important fact is, that the bellman optimality operator $\mathcal{T}$ is not a contraction in any metric **unclear:** formally state this? would require more definitions . The distribution does not converge to a fixed point, but rather to a sequence of optimal (in terms of expected value) policies.

2.3.1 *The Wasserstein Metric*

One of the tools for analysis of distributional approaches to reinforcement learning is the Wasserstein metric d_p between cumulative distribution functions [**bickel81asymptotic**]. For F, G two c.d.fs over the reals, it is defined as

$$d_p(F,G) := \inf_{U,V} \|U - V\|_p,$$

where the infimum is taken over all pairs of random variables (U, V) with respective cumulative distributions F and G. The infimum is attained by the inverse c.d.f. transform of a random variable $\mathcal{U}$ uniformly distributed on $[0,1]$:

$$d_p(F,G) = \|F^{-1}(\mathcal{U}) - G^{-1}(\mathcal{U})\|_p.$$

For $p < \infty$ this is more explicitly written as

$$d_p(F,G) = \left(\int_0^1 \left| F^{-1}(u) - G^{-1}(u) \right|^p du \right)^{1/p}. \tag{2.11}$$

todo: explain in words, also mention why it's better than KL **todo:** maybe visuals, if it will become important

2.4 RISK-SENSITIVITY

2.4.1 *Value-at-Risk*

todo: introduction, importance, flaws, ... Let Z be a bounded-mean random variable, i.e. $\mathbb{E}[|Z|] < \infty$, with cummulative distribution function (c.d.f.) $F(z) = \mathbb{P}(Z \leq z)$.

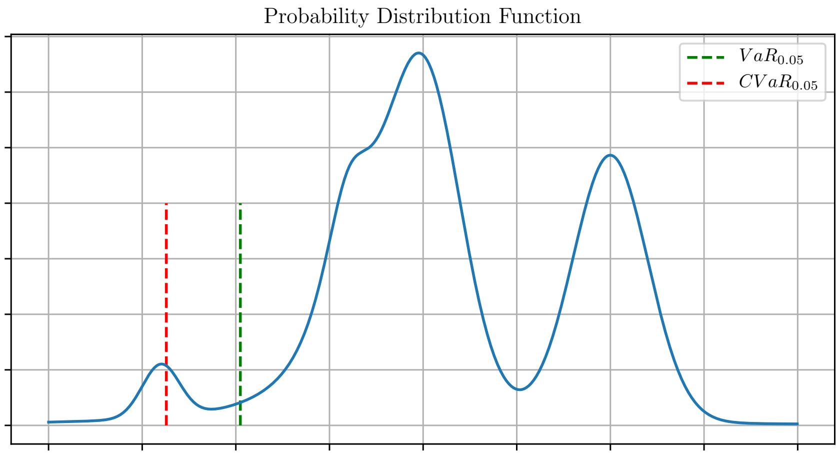

Figure 2.2: Value-at-Risk and Conditional Value-at-Risk of a general probability distribution with $\alpha = 0.05$. The main flaw of the VaR metric is clearly visible here, as we could shift the leftmost

In this paper we interpret Z as a reward[1]. The value-at-risk (VaR) at confidence level $\alpha \in (0, 1)$ is the α quantile of Z, i.e.

$$\mathrm{VaR}_\alpha(Z) = F^{-1}(\alpha) = \max\{z | F(z) \leq \alpha\} \tag{2.12}$$

We will use the notation $\mathrm{VaR}_\alpha(Z)$, $F^{-1}(\alpha)$ interchangebly, often explictly denoting the random variable of inverse c.d.f. as $F_Z^{-1}(\alpha)$.

2.4.2 *Conditional Value-at-Risk*

todo: introduction, robustness, ... The conditional value-at-risk (CVaR) at confidence level $\alpha \in (0, 1)$ is defined as:

$$\mathrm{CVaR}_\alpha(Z) = \frac{1}{\alpha}\int_0^\alpha F_Z^{-1}(\beta)\mathrm{d}\beta = \frac{1}{\alpha}\int_0^\alpha \mathrm{VaR}_\beta(Z)\mathrm{d}\beta \tag{2.13}$$

We will also use the following equivalent formulation from [22]:

$$\mathrm{CVaR}_\alpha(Z) = \max_s \left\{\frac{1}{\alpha}\mathbb{E}\left[(Z-s)^-\right] + s\right\} \tag{2.14}$$

where $(x)^- = \min(x, 0)$ represents the negative part of x.

2.5 PROBLEM FORMULATION

todo: combine rl and risk, remind motivations

[1] This is in accordance with reinforcement learning literature and opposed to risk-related literature.

The risk-sensitive problem we wish to adress for a given confidence level α is as follows:

$$\max_{\pi} \mathrm{CVaR}_{\alpha}(Z^{\pi}(x_0)) \tag{2.15}$$

2.5.1 *Time-consistency*

An important property of CVaR MDPs is that of time-consistency. ***Choose one definition, describe*** The notion of time consistency varies from author to author; in Shapiro [XXX] it is shown ***.

*** Policy gradient literature ignores the time consistency-issue, leading to locally optimal policies *** show that they can be worse than EXP ***.

2.6 LITERATURE SURVEY

3

VALUE ITERATION WITH CVAR

Value iteration is a standard algorithm for maximizing expected discounted reward used in reinforcement learning. In this chapter we extend the results of Chow et al. [9], who have recently proposed an approximate value iteration algorithm for CVaR MDPs.

The original algorithm requires the computation of a linear program in each step of the value iteration procedure. Utilizing a connection between the used αCVaR_α function and the quantile function, we sidestep the need for this computation and propose a linear-time version of the algorithm, making CVaR value iteration feasible for much larger MDPs.

We first present the original algorithm in section 3.1. The improved algorithm is presented in section 3.2. In section 3.3, we test the algorithm on selected environments.

3.1 CVAR VALUE ITERATION

Chow et al. [9] present a dynamic programming formulation for the CVaR MDP problem (see section 2.5). **todo:** more We repeat their key ideas and results bellow, as they form a basis for our contributions presented in later sections. The results are presented with our notation introduced in chapter 2, which differs slightly from the paper, but the core ideas remain the same.

3.1.1 *Bellman Equation for CVaR*

The results of Chow et al. [9] heavily rely on the CVaR decomposition theorem [**decomp**]:

unclear: repeat the original theorems in full?

$$CVaR_\alpha\left(Z^\pi(x,a)\right) = \min_{\xi \in \mathcal{U}_{\text{CVaR}}(\alpha, P(\cdot|x,a))} \sum_{x'} p(x'|x,a)\xi(x')CVaR_{\xi(x')\alpha}\left(Z^\pi(x')\right) \tag{3.1}$$

where the risk envelope $\mathcal{U}_{\text{CVaR}}(\alpha, P(\cdot|x,a))$ coincides with the dual definition of CVaR **??**.

The theorem states that we can compute the $CVaR_\alpha\left(Z^\pi(x,a)\right)$ as the minimal (or worst-case) weighted combination of $CVaR_\alpha\left(Z^\pi(x')\right)$ under a probability distribution perturbed by $\xi(x')$.

Note that the decomposition requires only the representation of CVaR at different (or all) confidence levels and no the whole distribution. **todo:** make the distinction clear, maybe in prelim? .

Chow et al. [9] extend these results by defining the *CVaR value-function* $V(x,y)$ with an augmented state-space $\mathcal{X} \times \mathcal{Y}$ where $\mathcal{Y} = (0,1]$ is an additional continuous state.

$$V(x,y) = \max_{\pi \in \Pi} \text{CVaR}_y \left(Z^\pi(x) \right) \tag{3.2}$$

Similar to standard DP, it is convenient to work with with operators defined on the space of value functions. This leads to the following definition of the CVaR Bellman operator $\mathbf{T} : \mathcal{X} \times \mathcal{Y} \to \mathcal{X} \times \mathcal{Y}$:

$$\mathbf{T}V(x,y) = \max_a \left[R(x,a) + \gamma \min_{\xi \in \mathcal{U}_{\text{CVaR}}(\alpha, P(\cdot|x,a))} \sum_{x'} p(x'|x,a)\xi(x')V\left(x', y\xi(x')\right) \right] \tag{3.3}$$

or in our simplified notation:

$$\mathbf{T}CVaR_y(Z(x)) = \max_a \left[R(x,a) + \gamma CVaR_y(Z(x,a)) \right] \tag{3.4}$$

[9](lemma 3) showed that the operator $\mathbf{T}$ is a contraction and also perserves the convexity of $yCVaR_t$. The maximization problem 3.1 is a convex one and therefore has a unique solution. Additionally, the fixed point of this contraction is the optimal $V^*(x,y) = \max_{\pi \in \Pi} CVaR_y(Z^\pi(x,y))$ (Theorem 4).

The value-function V^* can then be used to extract the optimal policy π^* of the original problem **??**, using the following theorem

Theorem 1 (Optimal Policies, Theorem 5 in [9])**.** *Let* $\pi^*_H = \{\mu_0, \mu_1, \dots\} \in \Pi_H$ *be a history-dependent policy recursively defined as:*

$$\mu_k(h_k) = u^*(x_k, y_k), \ \forall k \geq 0, \tag{3.5}$$

with initial conditions x_0 *and* $y_0 = \alpha$*, and state transitions*

$$x_k \sim P(\cdot \mid x_{k-1}, u^*(x_{k-1}, y_{k-1})), \quad y_k = y_{k-1}\xi^*_{x_{k-1}, y_{k-1}, u^*}(x_k), \forall k \geq 1, \tag{3.6}$$

where the stationary Markovian policy $u^*(x,y)$ *and risk factor* $\xi^*_{x,y,u^*}(\cdot)$ *are solution to the min-max optimization problem in the CVaR Bellman operator* $\mathbf{T}[V^*](x,y)$*. Then,* π^*_H *is an optimal policy for problem* (**??**) *with initial state* x_0 *and CVaR confidence level* α*.*

This algorithm is unfortunately unusable in practice, as the state-space is continuous in y. The solution proposed in [9] is then to represent the convex $yCVaR_y$ as a piecwise linear function.

3.1.2 *Value Iteration with Linear Interpolation*

Given a set of $N(x)$ interpolation points $\mathbf{Y}(x) = \left\{ y_1, \dots, y_{N(x)} \right\}$, we can interpolate the $yV(x, y)$ function on these points, i.e.

$$\mathcal{I}_x[V](y) = y_i V(x, y_i) + \frac{y_{i+1} V(x, y_{i+1}) - y_i V(x, y_i)}{y_{i+1} - y_i}(y - y_i),$$

where $y_i = \max\{y' \in \mathbf{Y}(x) : y' \leq y\}$ The interpolated Bellman operator is then also a contraction and has a bounded error(Theorem 7).

todo: bounded -> linear in θ

$$\mathbf{T}_{\mathcal{I}} V(x, y) = \max_a \left[R(x, a) + \gamma \min_{\xi \in \mathcal{U}_{\text{CVaR}}(\alpha, P(\cdot|x,a))} \sum_{x'} p(x'|x, a) \frac{\mathcal{I}_{x'}[V](y\xi(x'))}{y} \right] \quad (3.7)$$

The full value iteration procedure is presented in algorithm 1

Algorithm 1 `CVaR Value Iteration with Linear Interpolation`

```
1: Given:
    • $N(x)$ interpolation points $\mathbf{Y}(x) = \left\{ y_1, \dots, y_{N(x)} \right\} \in [0, 1]^{N(x)}$ for every $x \in \mathcal{X}$
      with $y_i < y_{i+1}$, $y_1 = 0$ and $y_{N(x)} = 1$.
    • Initial value function $V_0(x, y)$ that satisfies Assumption ??.
2: For $t = 1, 2, \dots$
    • For each $x \in \mathcal{X}$ and each $y_i \in \mathbf{Y}(x)$, update the value function estimate as
      follows:
          $V_t(x, y_i) = \mathbf{T}_{\mathcal{I}}[V_{t-1}](x, y_i),$
3: Set the converged value iteration estimate as $\widehat{V}^*(x, y_i)$, for any $x \in \mathcal{X}$, and $y_i \in \mathbf{Y}(x)$.
```

This algorithm can be used to find an approximate global optimum in any MDP. There is however the issue of computational complexity. As the algorithm stands, the straightforward approach is to solve each iteration of 3.7 as a linear program, since the problem is convex and piecewise linear. This however is not practical, as the LP computation can be demanding and is therefore not suitable for large state-spaces.

unclear: maybe formulate the LP exactly?

In the next section we aim to find a different way of computing the optimization problem presented in 3.1.

3.2 EFFICIENT COMPUTATION USING QUANTILE REPRESENTATION

We note several important facts regarding the CVaR computation. Firstly, it is unimportant *how* we arrive at the $\text{CVaR}_y(Z(x, a))$ present in the CVaR Bellman operator

3.4. Secondly, note that any discrete distribution has a piecewise linear $y\mathrm{CVaR}_y$ function [23], and this statement is twosided: any a piecewise linear $y\mathrm{CVaR}_y$ function can be seen as representing a certain discrete distribution.

The relation between $y\mathrm{CVaR}_y$ and the underlying distribution is best described in the following equation:

$$\frac{\partial}{\partial\alpha}\alpha\mathrm{CVaR}_\alpha(Z) = \frac{\partial}{\partial\alpha}\int_0^\alpha VaR_\beta(Z)d\beta = VaR_\alpha(Z) \tag{3.8}$$

We see that the quantile function of Z is extractable from the $y\mathrm{CVaR}_y$ function by differentiating over y **todo:** unify y, α . The opposite direction also holds:

$$\alpha\mathrm{CVaR}_\alpha(Z) = \int_0^\alpha VaR_\beta(Z)\mathrm{d}\beta + c \tag{3.9}$$

where we have $c = 0$ because of a starting condition $y\mathrm{CVaR}_y\Big|_{y=0} = 0$.

3.2.1 *CVaR Computation via Quantile Representation*

We propose the following procedure: instead of using linear programming for the CVaR computation, we use the underlying distributions represented by the $\alpha CVaR_\alpha$ function to compute CVaR.

The computation of CVaR of a discrete probability mixture is a linear-time process as we show bellow. The general steps of the computation are as follows

1. transform $y\mathrm{CVaR}_y$ of each possible state transition to a discrete probability distribution function using 3.8
2. combine these to to a distribution representing the full state-action distribution
3. compute $y\mathrm{CVaR}_y$ for all atoms using 3.9

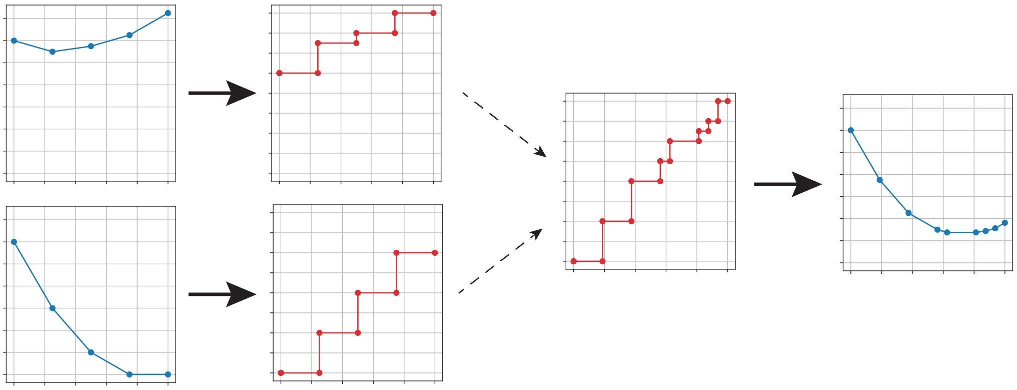

Figure 3.1: Visualization of the CVaR Value Iteration for a single state and action with two transition states. Thick arrows represent the conversion between $\alpha\mathrm{CVaR}_\alpha$ and the quantile function.

unclear: proof necessary? also, maybe it is already in the ξ proof

3.2.2 *ξ-computation*

Similarly to theorem 1, we need a way to compute the $y_{k+1} = y_k \xi^*(x_k)$ to extract the optimal policy. Again, we can skip the LP computation by using the following intuition: y_{k+1} is the portion of $Z(x_{k+1})$ that is present in $\mathrm{CVaR}_{y_k}(Z(x_k))$. In the continuous case, it is the probability in $Z(x_{k+1})$ before the $\mathrm{VaR}_{y_k}(Z(x_k))$ as we show bellow.

todo: proof for discrete distributions

Theorem 2. *Solution to minimization problem 3.1 can be computed without optimization by setting*

$$\xi(x') = \frac{F_{x'}(F_x^{-1}(\alpha))}{\alpha} \tag{3.10}$$

Proof. For simplification, we work only with two states: x' the actual sampled state and $\bar{x}'$ representing the other states. The equation then simplifies to

$$\begin{aligned} CVaR_\alpha(x,a) &= \min_\xi p\xi CVaR_{\xi\alpha}(x') + (1-p)\frac{1-p\xi}{1-p} CVaR_{\frac{1-p\xi}{1-p}\alpha}(\bar{x}') \\ &= \min_\xi p\xi CVaR_{\xi\alpha}(x') + (1-p\xi) CVaR_{\frac{1-p\xi}{1-p}\alpha}(\bar{x}') \end{aligned} \tag{3.11}$$

To find the min we first find the first derivative[1] w.r.t. ξ

$$\begin{aligned} \frac{\partial CVaR_\alpha}{\partial \xi} &= pCVaR_{\xi\alpha} + p\xi \frac{\partial CVaR_{\alpha\xi}}{\partial \xi} - pCVaR_{\frac{1-p\xi}{1-p}\alpha} + (1-p\xi)\frac{\partial CVaR_{\frac{1-p\xi}{1-p}\alpha}}{\partial \xi} \\ &= pCVaR_{\xi\alpha} + p\xi\left[\frac{1}{\xi}VaR_{\xi\alpha} - \frac{1}{\xi}CVaR_{\xi\alpha}\right] - pCVaR_{\frac{1-p\xi}{1-p}\alpha} \\ &\qquad + (1-p\xi)\left[\frac{p}{1-p\xi}CVaR_{\frac{1-p\xi}{1-p}\alpha} - \frac{p}{1-p\xi}VaR_{\frac{1-p\xi}{1-p}\alpha}\right] \\ &= pCVaR_{\xi\alpha} + pVaR_{\xi\alpha} - pCVaR_{\xi\alpha} - pCVaR_{\xi\alpha} - pCVaR_{\frac{1-p\xi}{1-p}\alpha} \\ &\qquad + CVaR_{\frac{1-p\xi}{1-p}\alpha} - pVaR_{\frac{1-p\xi}{1-p}\alpha} \\ &= pVaR_{\xi\alpha} - pVaR_{\frac{1-p\xi}{1-p}\alpha} \end{aligned} \tag{3.12}$$

By setting the derivative to 0 (to find the min), we get

$$VaR_{\xi\alpha}(x') = VaR_{\frac{1-p\xi}{1-p}\alpha}(\bar{x}') \tag{3.13}$$

By inserting claim 3.10 into **??** we get the symmetrical claim

$$\frac{1-p\xi}{1-p} = \xi(\bar{x}') = \frac{F_{\bar{x}'}(F_x^{-1}(\alpha))}{\alpha} \tag{3.14}$$

[1] We used the following identities:

$$\frac{\partial CVaR_{\alpha\xi}}{\partial \xi} = \frac{1}{\xi}VaR_{\xi\alpha} - \frac{1}{\xi}CVaR_{\xi\alpha} \qquad \frac{\partial CVaR_{\frac{1-p\xi}{1-p}\alpha}}{\partial \xi} = \frac{p}{1-p\xi}CVaR_{\frac{1-p\xi}{1-p}\alpha} - \frac{p}{1-p\xi}VaR_{\frac{1-p\xi}{1-p}\alpha}$$

We rewrite 3.11 as (assuming ξ is the minimum point)

$$\begin{aligned}\frac{1}{\alpha}\int_0^{\alpha} F_x^{-1}(t)dt &= p\xi\frac{1}{\xi\alpha}\int_0^{\xi\alpha} F_{x'}^{-1}(t)dt + (1-p\xi)\frac{1-p}{(1-p\xi)\alpha}\int_0^{\frac{1-p\xi}{1-p}\alpha} F_{\bar{x}'}^{-1}(t) \\ &= p\frac{1}{\alpha}\int_0^{\xi\alpha} F_{x'}^{-1}(t)dt + (1-p)\frac{1}{\alpha}\int_0^{\frac{1-p\xi}{1-p}\alpha} F_{\bar{x}'}^{-1}(t)\end{aligned} \tag{3.15}$$

This must also hold if we multiply both sides by α

$$\int_0^{\alpha} F_x^{-1}(t)dt = p\int_0^{\xi\alpha} F_{x'}^{-1}(t)dt + (1-p)\int_0^{\frac{1-p\xi}{1-p}\alpha} F_{\bar{x}'}^{-1}(t) \tag{3.16}$$

And we take derivations w.r.t. α of both sides

$$F_x^{-1}(\alpha) = p\xi F_{x'}^{-1}(\xi\alpha) + (1-p\xi)F_{\bar{x}'}^{-1}(\frac{1-p\xi}{1-p}\alpha) \tag{3.17}$$

By inserting 3.10 we get

$$\begin{aligned}p\xi F_{x'}^{-1}(\xi\alpha) + (1-p)\xi_2 F_{\bar{x}'}^{-1}(\xi_2\alpha) &= p\xi F_{x'}^{-1}(F_{x'}(F_x^{-1}(\alpha))) + (1-p\xi)F_{\bar{x}'}^{-1}\left(F_{\bar{x}'}(F_x^{-1}(\alpha))\right) \\ &= p\xi F_x^{-1}(\alpha) + (1-p\xi)F_x^{-1}(\alpha) = F_x^{-1}(\alpha)\end{aligned} \tag{3.18}$$

We've shown that the proposed solution 3.10 satisfies the minimization constraint 3.13 (= is a minimal point) and satisfies the dual decomposition 3.1. (This has been shown only in the differentiated form)

□

3.3 EXPERIMENTS

todo: $|\mathcal{X}| \sim 1M$ tabular environment

todo: get matlab code from tamar

3.3.1 *Cliffworld*

3.4 SUMMARY

4

Q-LEARNING WITH CVAR

4.1 VAR-BASED POLICY IMPROVEMENT

4.1.1 *Policy Improvement*

Recall the primal definition of CVaR (***). Our goal can then be rewritten as

$$\max_{\pi} CVaR_{\alpha}^{\pi}(Z) = \max_{\pi} \max_{s} \frac{1}{\alpha} \mathbb{E}\left[(Z^{\pi} - s)^{-}\right] + s \tag{4.1}$$

It also holds [XXX] that for the maximum, it holds $s^* = \mathrm{VaR}_{\alpha}$

$$\mathrm{CVaR}_{\alpha}(Z) = \max_{s} \left\{ \frac{1}{\alpha} \mathbb{E}\left[(Z - s)^{-}\right] + s \right\} = \frac{1}{\alpha} \mathbb{E}\left[(Z - \mathrm{VaR}_{\alpha}(Z))^{-}\right] + \mathrm{VaR}_{\alpha}(Z) \tag{4.2}$$

The main idea of the algorithm 2, partially explored in [4], is as follows: If we knew the value s^* in the solution to equation (***), we could simplify the problem to maximize only

$$\max_{\pi} CVaR_{\alpha}(Z) = \max_{\pi} \frac{1}{\alpha} \mathbb{E}\left[(Z^{\pi} - s^*)^{-}\right] + s^* \tag{4.3}$$

Given that we have access to the return distributions, we can improve the policy by simply choosing an action that maximizes CVaR in the first state $a_0 = \arg\max_{\pi} \mathrm{CVaR}_{\alpha}(Z^{\pi}(x_0))$. We can then, as an approximation, set $s = VaR_{\alpha}(Z(x_0))$ and then only maximize the simpler criterion

$$\max_{\pi} CVaR_{\alpha}(Z) = \max_{\pi} \frac{1}{\alpha} \mathbb{E}\left[(Z^{\pi} - s^*)^{-}\right] + s^* \tag{4.4}$$

The algorithm can be seen as coordinate ascent; in the first phase (when we compute CVaR) we maximize w.r.t. s while keeping π fixed; in the second phase we fix s and maximize w.r.t. π.

In the following theorem, we show that this indeed leads to a monotonic improvement over the previous policy.

Theorem 3. *Let π be a fixed policy, $\alpha \in (0, 1]$. By following policy π' from algorithm 2, we will improve $CVaR_{\alpha}(Z)$ in expectation:*

$$CVaR_{\alpha}(Z^{\pi}) \leq CVaR_{\alpha}(Z^{\pi'})$$

Proof. Let s^* be a solution to eq 2.14. Then by optimizing $\max_\pi \frac{1}{\alpha}\mathbb{E}\left[(Z-s^*)^-\right]$, we will monotonely improve the optimization criterion 4.3.

$$CVaR_\alpha(Z^\pi) = \frac{1}{\alpha}\mathbb{E}\left[(Z^\pi - s^*)^-\right] + s^* \leq \max_{\pi'} \frac{1}{\alpha}\mathbb{E}\left[(Z^{\pi'} - s^*)^-\right] + s^*$$

Note the following facts:

$$Z_t = R_t + \gamma Z_{t+1} \tag{4.5}$$

$$\mathbb{E}\left[(Z_t - s)^-\right] = \mathbb{E}\left[(Z_t - s)\mathbb{1}(Z_t \leq s)\right] \tag{4.6}$$

$$\mathbb{E}[H(Z)] = \sum_i p_i \mathbb{E}[H(Z_i)] \tag{4.7}$$

The last equation holds if $Z \sim p_i$ is a probability mixture for any function H. We can rewrite the criterion as

$$\begin{aligned}
\mathbb{E}\left[(Z_t - s)^-\right] = \mathbb{E}\left[(Z_t - s)\mathbb{1}(Z_t \leq s)\right] &= \mathbb{E}\left[(R_t + \gamma Z_{t+1} - s)\mathbb{1}(Z_{t+1} \leq \frac{s - R_t}{\gamma})\right] \\
&= \sum_{x_{t+1}, r_t} P(x_{t+1}, r_t \,|\, x_t, a)\mathbb{E}\left[(r_t + \gamma Z(x_{t+1}) - s)\mathbb{1}(Z(x_{t+1}) \leq \frac{s - r_t}{\gamma})\right] \\
&= \sum_{x_{t+1}, r_t} P(x_{t+1}, r_t \,|\, x_t, a)\mathbb{E}\left[\gamma\left(Z(x_{t+1}) - \frac{s - r_t}{\gamma}\right)\mathbb{1}(Z(x_{t+1}) \leq \frac{s - r_t}{\gamma})\right] \\
&= \gamma \sum_{x_{t+1}, r_t} P(x_{t+1}, r_t \,|\, x_t, a)\mathbb{E}\left[\left(Z(x_{t+1}) - \frac{s - r_t}{\gamma}\right)\mathbb{1}(Z(x_{t+1}) \leq \frac{s - r_t}{\gamma})\right] \\
&= \gamma \sum_{x_{t+1}, r_t} P(x_{t+1}, r_t \,|\, x_t, a)\mathbb{E}\left[\left(Z(x_{t+1}) - \frac{s - r_t}{\gamma}\right)^-\right]
\end{aligned} \tag{4.8}$$

Now let's say we sampled reward $\hat{r}_t$ and state $\hat{x}_{t+1}$, we are still trying to find a policy π^* that maximizes

$$\begin{aligned}
\pi^* &= \arg\max_\pi \mathbb{E}\left[(Z_t - s)^- \,\middle|\, \hat{x}_{t+1}, \hat{r}\right] \\
&= \arg\max_\pi \mathbb{E}\left[\left(Z(\hat{x}_{t+1}) - \frac{s - \hat{r}_t}{\gamma}\right)^-\right]
\end{aligned} \tag{4.9}$$

Where we ignored the unsampled states (since these are not a function of $\hat{x}_{t+1}$) and the multiplicative constant γ that will not affect the maximum argument.

At the starting state, we set $s = s^*$. At each following state we select an action according to equation 4.9. By induction we maximize the criterion **??** in each step. □

Note that the resulting policy is nonstationary, however we do not need an extended state-space to follow this policy, it is only necessary to remember our previous s.

Algorithm 2 VaR-based policy improvement

```
input $\alpha, x_0, \gamma$
$a = \arg\max_a CVaR_\alpha(Z(x_0, a))$
$s = VaR_\alpha(Z(x_0, a))$
$x_t, r_t = \text{envTransition}(x_0, a)$
while $x_t$ is not terminal do
    $s = \frac{s - r_t}{\gamma}$
    $a = \arg\max_a \mathbb{E}\left[(Z(x_t, a) - s)^-\right]$
    $x_t, r_t = \text{envTransition}(x_t, a)$
end while
```

4.1.2 *Repeated policy improvement*

This policy then could be evaluated again by the distributional Q-learning procedures, however we

4.2 TODO

4.3 TODO

4.4 TODO

4.5 EXPERIMENTS

todo: ai safety gridworld

5

APPROXIMATE Q-LEARNING WITH CVAR

5.1 TODO

5.2 TODO

5.3 TODO

5.4 EXPERIMENTS

6

CONCLUSION

Bäuerle and Ott [4] Bellemare, Dabney, and Munos [5] Chow et al. [9] Dabney et al. [11] Garcıa and Fernández [12] Majumdar and Pavone [17] Morimura et al. [19] Morimura et al. [20] Pflug and Pichler [21] Rockafellar and Uryasev [22] Rockafellar and Uryasev [23] Majumdar and Pavone [17] Leike et al. [16] Amodei et al. [1] Shapiro [24] Artzner et al. [2] Tamar et al. [27] Sutton and Barto [26] Watkins and Dayan [29] Bellman [6] Tsitsiklis [28] Boyd and Vandenberghe [8] Kreyszig [14] Koenker and Hallock [13] Committee [10] Mnih et al. [18] Silver et al. [25] Bahdanau, Cho, and Bengio [3] Krizhevsky, Sutskever, and Hinton [15]

A

INTRODUCTION

This bundle for LaTeX has two goals:

1. Provide students with an easy-to-use template for their Master's or PhD thesis. (Though it might also be used by other types of authors for reports, books, etc.)

2. Provide a classic, high-quality typographic style that is inspired by **bringhurst:2002**'s "*The Elements of Typographic Style*" [**bringhurst:2002**].

Risk-Averse Distributional Reinforcement Learning version 0.0

The bundle is configured to run with a *full* MiKTeX or TeXLive[1] installation right away and, therefore, it uses only freely available fonts. (Minion fans can easily adjust the style to their needs.)

People interested only in the nice style and not the whole bundle can now use the style stand-alone via the file `classicthesis.sty`. This works now also with "plain" LaTeX.

As of version 3.0, `classicthesis` can also be easily used with LyX[2] thanks to Nicholas Mariette and Ivo Pletikosić. The LyX version of this manual will contain more information on the details.

This should enable anyone with a basic knowledge of LaTeX 2_ε or LyX to produce beautiful documents without too much effort. In the end, this is my overall goal: more beautiful documents, especially theses, as I am tired of seeing so many ugly ones.

The whole template and the used style is released under the GNU General Public License.

If you like the style then I would appreciate a postcard:

André Miede
Detmolder Straße 32
31737 Rinteln
Germany

The postcards I received so far are available at:

http://postcards.miede.de

So far, many theses, some books, and several other publications have been typeset successfully with it. If you are interested in some typographic details behind it, enjoy Robert Bringhurst's wonderful book.

A well-balanced line width improves the legibility of the text. That's what typography is all about, right?

[1] See the file `LISTOFFILES` for needed packages. Furthermore, `classicthesis` works with most other distributions and, thus, with most systems LaTeX is available for.

[2] http://www.lyx.org

IMPORTANT NOTE: Some things of this style might look unusual at first glance, many people feel so in the beginning. However, all things are intentionally designed to be as they are, especially these:

- No bold fonts are used. Italics or spaced small caps do the job quite well.
- The size of the text body is intentionally shaped like it is. It supports both legibility and allows a reasonable amount of information to be on a page. And, no: the lines are not too short.
- The tables intentionally do not use vertical or double rules. See the documentation for the `booktabs` package for a nice discussion of this topic.[3]
- And last but not least, to provide the reader with a way easier access to page numbers in the table of contents, the page numbers are right behind the titles. Yes, they are *not* neatly aligned at the right side and they are *not* connected with dots that help the eye to bridge a distance that is not necessary. If you are still not convinced: is your reader interested in the page number or does she want to sum the numbers up?

Therefore, please do not break the beauty of the style by changing these things unless you really know what you are doing! Please.

YET ANOTHER IMPORTANT NOTE: Since `classicthesis`' first release in 2006, many things have changed in the LaTeX world. Trying to keep up-to-date, `classicthesis` grew and evolved into many directions, trying to stay (some kind of) stable and be compatible with its port to LyX. However, there are still many remains from older times in the code, many dirty workarounds here and there, and several other things I am absolutely not proud of (for example my unwise combination of KOMA and `titlesec` etc.).

An outlook into the future of `classicthesis`.

Currently, I am looking into how to completely re-design and re-implement `classicthesis` making it easier to maintain and to use. As a general idea, `classicthesis.sty` should be developed and distributed separately from the template bundle itself. Excellent spin-offs such as `arsclassica` could also be integrated (with permission by their authors) as format configurations. Also, current trends of `microtype`, `fontspec`, etc. should be included as well. As I am not really into deep LaTeX programming, I will reach out to the LaTeX community for their expertise and help.

A.1 ORGANIZATION

A very important factor for successful thesis writing is the organization of the material. This template suggests a structure as the following:

You can use these margins for summaries of the text body. . .

- `Chapters/` is where all the "real" content goes in separate files such as `Chapter01.tex` etc.
- `FrontBackMatter/` is where all the stuff goes that surrounds the "real" content, such as the acknowledgments, dedication, etc.

[3] To be found online at http://mirror.ctan.org/macros/latex/contrib/booktabs/.

- `gfx/` is where you put all the graphics you use in the thesis. Maybe they should be organized into subfolders depending on the chapter they are used in, if you have a lot of graphics.

- `Bibliography.bib`: the BibTEX database to organize all the references you might want to cite.

- `classicthesis.sty`: the style definition to get this awesome look and feel. Does not only work with this thesis template but also on its own (see folder `Examples`). Bonus: works with both LATEX and PDFLATEX... and LYX. Great tool and it's free!

- `ClassicThesis.tex`: the main file of your thesis where all gets bundled together.

- `classicthesis-config.tex`: a central place to load all nifty packages that are used.

  Make your changes and adjustments here. This means that you specify here the options you want to load `classicthesis.sty` with. You also adjust the title of your thesis, your name, and all similar information here. Refer to Section A.3 for more information.

  This had to change as of version 3.0 in order to enable an easy transition from the "basic" style to LYX.

In total, this should get you started in no time.

A.2 STYLE OPTIONS

There are a couple of options for `classicthesis.sty` that allow for a bit of freedom concerning the layout:

. . . or your supervisor might use the margins for some comments of her own while reading.

- General:
  - `drafting`: prints the date and time at the bottom of each page, so you always know which version you are dealing with. Might come in handy not to give your Prof. that old draft.
- Parts and Chapters:
  - `parts`: if you use Part divisions for your document, you should choose this option. (Cannot be used together with `nochapters`.)
  - `linedheaders`: changes the look of the chapter headings a bit by adding a horizontal line above the chapter title. The chapter number will also be moved to the top of the page, above the chapter title.
- Typography:
  - `eulerchapternumbers`: use figures from Hermann Zapf's Euler math font for the chapter numbers. By default, old style figures from the Palatino font are used.
  - `beramono`: loads Bera Mono as typewriter font. (Default setting is using the standard CM typewriter font.)
  - `eulermath`: loads the awesome Euler fonts for math. Palatino is used as default font.

Options are enabled via `option=true`

- Table of Contents:
  - `tocaligned`: aligns the whole table of contents on the left side. Some people like that, some don't.
  - `dottedtoc`: sets pagenumbers flushed right in the table of contents.
  - `manychapters`: if you need more than nine chapters for your document, you might not be happy with the spacing between the chapter number and the chapter title in the Table of Contents. This option allows for additional space in this context. However, it does not look as "perfect" if you use `\parts` for structuring your document.
- Floats:
  - `listings`: loads the `listings` package (if not already done) and configures the List of Listings accordingly.
  - `floatperchapter`: activates numbering per chapter for all floats such as figures, tables, and listings (if used).

Furthermore, pre-defined margins for different paper sizes are available, e. g., `a4paper`, `a5paper`, and `letterpaper`. These are based on your chosen option of `\documentclass`.

The best way to figure these options out is to try the different possibilities and see what you and your supervisor like best.

In order to make things easier, `classicthesis-config.tex` contains some useful commands that might help you.

A.3 CUSTOMIZATION

This section will show you some hints how to adapt classicthesis to your needs.

The file classicthesis.sty contains the core functionality of the style and in most cases will be left intact, whereas the file classicthesis-config.tex is used for some common user customizations.

The first customization you are about to make is to alter the document title, author name, and other thesis details. In order to do this, replace the data in the following lines of classicthesis-config.tex:

Modifications in classicthesis-config.tex

```
% ****************************************************
% 2. Personal data and user ad-hoc commands
% ****************************************************
\newcommand{\myTitle}{A Classic Thesis Style\xspace}
\newcommand{\mySubtitle}{An Homage to...\xspace}
```

Further customization can be made in classicthesis-config.tex by choosing the options to classicthesis.sty (see Section A.2) in a line that looks like this:

```
\PassOptionsToPackage{
  drafting=true,    % print version information on the bottom
       of the pages
  tocaligned=false, % the left column of the toc will be
     aligned (no indentation)
  dottedtoc=false,  % page numbers in ToC flushed right
  parts=true,       % use part division
  eulerchapternumbers=true, % use AMS Euler for chapter font
     (otherwise Palatino)
  linedheaders=false,       % chaper headers will have line
     above and beneath
  floatperchapter=true,     % numbering per chapter for all
     floats (i.e., Figure 1.1)
  listings=true,    % load listings package and setup LoL
  subfig=true,      % setup for preloaded subfig package
  eulermath=false,  % use awesome Euler fonts for
     mathematical formulae (only with pdfLaTeX)
  beramono=true,    % toggle a nice monospaced font (w/ bold)
  minionpro=false   % setup for minion pro font; use minion
     pro small caps as well (only with pdfLaTeX)
}{classicthesis}
```

Many other customizations in classicthesis-config.tex are possible, but you should be careful making changes there, since some changes could cause errors.

A.4 ISSUES

This section will list some information about problems using classicthesis in general or using it with other packages.

Beta versions of classicthesis can be found at Bitbucket:

https://bitbucket.org/amiede/classicthesis/

There, you can also post serious bugs and problems you encounter.

A.5 FUTURE WORK

So far, this is a quite stable version that served a couple of people well during their thesis time. However, some things are still not as they should be. Proper documentation in the standard format is still missing. In the long run, the style should probably be published separately, with the template bundle being only an application of the style. Alas, there is no time for that at the moment... it could be a nice task for a small group of LaTeXnicians.

Please do not send me email with questions concerning LaTeX or the template, as I do not have time for an answer. But if you have comments, suggestions, or improvements for the style or the template in general, do not hesitate to write them on that postcard of yours.

A.6 BEYOND A THESIS

The layout of `classicthesis.sty` can be easily used without the framework of this template. A few examples where it was used to typeset an article, a book or a curriculum vitae can be found in the folder `Examples`. The examples have been tested with `latex` and `pdflatex` and are easy to compile. To encourage you even more, PDFs built from the sources can be found in the same folder.

A.7 LICENSE

GNU GENERAL PUBLIC LICENSE: This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but *without any warranty*; without even the implied warranty of *merchantability* or *fitness for a particular purpose*. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; see the file `COPYING`. If not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.

CLASSICTHESIS AUTHORS' NOTE: There have been some discussions about the GPL's implications on using `classicthesis` for theses etc. Details can be found here:

https://bitbucket.org/amiede/classicthesis/issues/123/

We chose (and currently stick with) the GPL because we would not like to compete with proprietary modified versions of our own work. However, the whole template is free as free beer and free speech. We will not demand the sources for theses, books, CVs, etc. that were created using `classicthesis`.

Postcards are still highly appreciated.

BIBLIOGRAPHY

[1] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. "Concrete problems in AI safety." In: *arXiv preprint arXiv:1606.06565* (2016).

[2] Philippe Artzner, Freddy Delbaen, Jean-Marc Eber, and David Heath. "Coherent measures of risk." In: *Mathematical finance* 9.3 (1999), pp. 203–228.

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." In: *arXiv preprint arXiv:1409.0473* (2014).

[4] Nicole Bäuerle and Jonathan Ott. "Markov decision processes with average-value-at-risk criteria." In: *Mathematical Methods of Operations Research* 74.3 (2011), pp. 361–379.

[5] Marc G Bellemare, Will Dabney, and Rémi Munos. "A distributional perspective on reinforcement learning." In: *arXiv preprint arXiv:1707.06887* (2017).

[6] Richard Bellman. "A Markovian decision process." In: *Journal of Mathematics and Mechanics* (1957), pp. 679–684.

[7] Dimitri P Bertsekas and John N Tsitsiklis. "Neuro-dynamic programming: an overview." In: *Decision and Control, 1995., Proceedings of the 34th IEEE Conference on.* Vol. 1. IEEE. 1995, pp. 560–564.

[8] Stephen Boyd and Lieven Vandenberghe. *Convex optimization.* Cambridge university press, 2004.

[9] Yinlam Chow, Aviv Tamar, Shie Mannor, and Marco Pavone. "Risk-sensitive and robust decision-making: a CVaR optimization approach." In: *Advances in Neural Information Processing Systems.* 2015, pp. 1522–1530.

[10] Basel Committee et al. "Fundamental review of the trading book: A revised market risk framework." In: *Consultative Document, October* (2013).

[11] Will Dabney, Mark Rowland, Marc G Bellemare, and Rémi Munos. "Distributional Reinforcement Learning with Quantile Regression." In: *arXiv preprint arXiv:1710.10044* (2017).

[12] Javier Garcıa and Fernando Fernández. "A comprehensive survey on safe reinforcement learning." In: *Journal of Machine Learning Research* 16.1 (2015), pp. 1437–1480.

[13] Roger Koenker and Kevin F Hallock. "Quantile regression." In: *Journal of economic perspectives* 15.4 (2001), pp. 143–156.

[14] Erwin Kreyszig. *Introductory functional analysis with applications.* Vol. 1. wiley New York, 1989.

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks." In: *Advances in neural information processing systems.* 2012, pp. 1097–1105.

[16] Jan Leike, Miljan Martic, Victoria Krakovna, Pedro A Ortega, Tom Everitt, Andrew Lefrancq, Laurent Orseau, and Shane Legg. "AI Safety Gridworlds." In: *arXiv preprint arXiv:1711.09883* (2017).

[17] Anirudha Majumdar and Marco Pavone. "How Should a Robot Assess Risk? Towards an Axiomatic Theory of Risk in Robotics." In: *arXiv preprint arXiv:1710.11040* (2017).

[18] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. "Human-level control through deep reinforcement learning." In: *Nature* 518.7540 (2015), p. 529.

[19] Tetsuro Morimura, Masashi Sugiyama, Hisashi Kashima, Hirotaka Hachiya, and Toshiyuki Tanaka. "Nonparametric return distribution approximation for reinforcement learning." In: *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*. 2010, pp. 799–806.

[20] Tetsuro Morimura, Masashi Sugiyama, Hisashi Kashima, Hirotaka Hachiya, and Toshiyuki Tanaka. "Parametric return density estimation for reinforcement learning." In: *arXiv preprint arXiv:1203.3497* (2012).

[21] Georg Ch Pflug and Alois Pichler. "Time-consistent decisions and temporal decomposition of coherent risk functionals." In: *Mathematics of Operations Research* 41.2 (2016), pp. 682–699.

[22] R Tyrrell Rockafellar and Stanislav Uryasev. "Optimization of conditional value-at-risk." In: *Journal of risk* 2 (2000), pp. 21–42.

[23] R Tyrrell Rockafellar and Stanislav Uryasev. "Conditional value-at-risk for general loss distributions." In: *Journal of banking & finance* 26.7 (2002), pp. 1443–1471.

[24] Alexander Shapiro. "On Kusuoka representation of law invariant risk measures." In: *Mathematics of Operations Research* 38.1 (2013), pp. 142–152.

[25] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. "Mastering the game of go without human knowledge." In: *Nature* 550.7676 (2017), p. 354.

[26] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. Vol. 1. 1. MIT press Cambridge, 1998.

[27] Aviv Tamar, Yinlam Chow, Mohammad Ghavamzadeh, and Shie Mannor. "Policy gradient for coherent risk measures." In: *Advances in Neural Information Processing Systems*. 2015, pp. 1468–1476.

[28] John N Tsitsiklis. "Asynchronous stochastic approximation and Q-learning." In: *Machine learning* 16.3 (1994), pp. 185–202.

[29] Christopher JCH Watkins and Peter Dayan. "Q-learning." In: *Machine learning* 8.3-4 (1992), pp. 279–292.

DECLARATION

Put your declaration here.

Prague, February, 2018

Silvestr Stanko

COLOPHON

This document was typeset using the typographical look-and-feel `classicthesis` developed by André Miede and Ivo Pletikosić. The style was inspired by Robert Bringhurst's seminal book on typography "*The Elements of Typographic Style*". `classicthesis` is available for both LaTeX and LyX:

https://bitbucket.org/amiede/classicthesis/

Happy users of `classicthesis` usually send a real postcard to the author, a collection of postcards received so far is featured here:

http://postcards.miede.de/

Thank you very much for your feedback and contribution.