POLITECNICO DI TORINO
Department of Mechanical and Aerospace Engineering
Master of Science in Aerospace Engineering

Politecnico di Torino

Master's Degree Thesis

Spacecraft Docking Optimization
Integrating Machine Learning Approaches

Supervisors:
Prof. Paolo Maggiore
Dott. Raoul Vetere

Candidate:
Leonardo Filippini (S290517)

Academic Year 2023/2024

Acknowledgments

My profound gratitude goes to my supervisor Raoul, whose direct guidance and active involvement have shaped the development of this thesis.

I am deeply grateful to my whole family for their steady support throughout my academic journey to graduation. Your belief in me has truly been the cornerstone of my achievements.

To my dear mom Silvia, your boundless love and encouragement have been invaluable in helping me navigate through the toughest of times. All of my accomplishments, including this one, can be traced back to you.

To my beloved girlfriend Arianna, who has become an integral part of my family and consistently guided me towards the right path, I owe immense gratitude. Your support and wisdom have been my guiding light.

My heartfelt thanks extend to my dear friends at Politecnico di Torino, with whom I have forged bonds that are unbreakable. Without your camaraderie and assistance, traversing through the challenges of academia would have been impossible. Federico, Francesca, Francesco, Aleksandr – I truly owe this accomplishment to each one of you.

A special mention goes to my housemate Francesco, with whom I've shared countless memories over the past 7 years. Your presence has always ensured a warm and positive environment to return to, no matter the difficulties faced. You are my brother.

Finally, a big thank you to all of those who I have not mentioned but have played a big part in shaping my journey, from my lifelong group of friends in Rome to the precious relationships

fostered at the ONAOSI Foundation in Turin (recurring guests included!).

Thank you all from the bottom of my heart!

Abstract

This thesis endeavors to explore and substantiate the utility of a novel approach to Spacecraft Docking Optimization, which integrates Machine Learning methodologies. The research commences by validating the proposed methodology through comprehensive mathematical analysis, grounding the model in well-established mathematical theorems and principles, particularly employing Optimal Control techniques to delineate the underlying framework. Subsequent to establishing the mathematical underpinnings, the study introduces and elucidates the integration of Neural Networks within the proposed framework, highlighting their potential efficacy in addressing the complexities inherent in solving Partial Differential Equations for optimization purposes.

Furthermore, the thesis progresses to conduct empirical evaluations utilizing computational simulations implemented in a Python programming environment. Through experimentation, comparisons are drawn between the conventional approach reliant on Optimal Control techniques and the Neural Network-based approach. These empirical assessments serve to ascertain the comparative effectiveness and efficiency of the two methodologies, thereby providing empirical validation of the viability of integrating Machine Learning approaches in this case, as well as many others.

By demonstrating the efficacy of Neural Networks in addressing complex optimization problems, this research hopes to contribute to the broader field of optimization by offering a different approach to integrating Artificial Intelligence, specifically Machine Learning, into the optimization of dynamic models.

Contents

List of Figures

Chapter 1

Theory Fundamentals

1.1 Problem Statement

Spacecraft docking, the process of bringing two separate vehicles together and securely attaching them in space, is a critical aspect of space exploration and satellite servicing missions. Achieving successful docking requires precise control and coordination of various parameters such as relative position, orientation, and velocity between the approaching vehicles. Traditionally, docking maneuvers have been performed manually or with limited automation, requiring significant human intervention and posing challenges in terms of efficiency, safety, and reliability. The problem of soft landing can be modeled through many approaches, one of which is tackled in this thesis and has its roots in the mathematical field of *Optimal Control*, deriving from the study of the *Hamilton-Jacobi-Bellman equation*. By formulating the docking problem within the framework of Optimal Control, input strategies can be designed to optimize stability and total energy, ensuring safe and precise alignment of docking ports. Modern attempts at improving spacecraft docking often involve the integration of machine learning algorithms with conventional Optimal Control techniques. In this paper, we will explore these approaches to seek to exploit the capabilities of neural networks in learning complex patterns from data, optimizing trajectory planning, and improving the overall performance of docking maneuvers. Through theoretical analysis and simulation studies, we seek to develop robust

and adaptive algorithms for enabling reliable and efficient docking operations.

Hereafter, we will denote the state of a given system as $y(t)$ and the controls acting upon that system as $\alpha(t)$. Initially, the objective will be to find the set of controls which minimises a particular functional, called the *cost functional*. The following chapter will introduce the problem and derive the aforementioned equations.

1.2 Derivation of the Hamilton-Jacobi-Bellman Equation

A generic Optimal Control problem can be described as follows: let us define a sufficiently smooth function f, such that the following system of differential equations has a unique solution:

$$\begin{cases} \dot{y}(t) = f(y(t), \alpha(t)) \\ y(0) = x \end{cases}$$

where:

- $f : \mathbb{R}^n \times \mathrm{A} \to \mathbb{R}^n$ is uniformly continuous with respect to the variable y and continuous with respect to the time variable t;
- $y : [0, T] \to \mathbb{R}^n$ is continuous at least;
- $x \in \mathbb{R}^n$;
- $\mathrm{A} \subset \mathbb{R}^m$ defined as $\mathrm{A} = \{\alpha : [0, t] \to \mathbb{R}^m \mid \alpha \text{ is Lebesgue measurable}\}$, represents the set containing the totality of valid controls.

We will then introduce a J functional which approximates the cost of the action which is being performed

$$\lambda > 0, \quad J(x, \alpha) = \int_0^{+\infty} \mathscr{L}(y(t), \alpha(t)) \cdot e^{-\lambda t}\, dt$$

where:

- $\mathscr{L} : \mathbb{R}^n \times \mathrm{A} \to \mathbb{R}^n$ is differentiable;
- x represents the initial position in the previous system of equations.

We can notice how even though the integral is calculated to infinity, its value still approaches a finite number thanks to the decreasing exponential function inside of it.

To proceed to study the problem we need to introduce more precise hypotheses on the regularity of $f(y, \alpha)$ and $\mathscr{L}(y, \alpha)$. We will start by establishing the definition of a *module*:

Definition 1. *A function* $\omega : \mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}$ *is a module if:*

1. $\omega(\,\cdot\,, \mathrm{R})$ *is continuous;*
2. $\omega(\,\cdot\,, \mathrm{R})$ *is non-decreasing;*
3. $\omega(0, \mathrm{R}) = 0$, *for all* $\mathrm{R} \geq 0$.

We will now introduce the *continuity hypotheses*:

A1.
- $\mathscr{L}$ is continuous;
- There exists $\omega_{\mathscr{L}} : \mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}$, such that

$$|\mathscr{L}(x, \alpha) - \mathscr{L}(y, \alpha)| \leq \omega_{\mathscr{L}}(|x - y|, \mathrm{R})$$

  for all $\alpha \in \mathrm{A}$, for all $R > 0$ and for all $x, y \in \mathbb{R}^n$;
- There exists $M_{\mathscr{L}} > 0$, such that

$$|\mathscr{L}(x, \alpha)| \leq M_{\mathscr{L}}.$$

A2.
- $f : \mathbb{R}^n \times \mathrm{A} \to \mathbb{R}^n$ is continuous;
- f is bounded in $\mathrm{B}(0, R)$ for all $R > 0$;
- f is *Lipschitz continuous*, that is to say there exists $L_f \geq 0$, such that

$$|f(x, \alpha) - f(y, \alpha)| \leq L_f |x - y|$$

  for all $\alpha \in \mathrm{A}$ and for all $x, y \in \mathbb{R}^n$.

1.2.1 Principle of Dynamic Programming

Having written down the conditions, we can introduce and prove the first important principle of this research: the *Principle of Dynamic Programming.*

Definition 2. *v is the value function which represents, by definition, the least possible cost of action starting from the initial position:*

$$v(x) = \inf_{\alpha \in \mathrm{A}} \{J(x, \alpha)\}$$

for all $x \in \mathbb{R}^n$.

We can prove that the previous equation can be written down as

$$v(x) = \inf_{\alpha \in \mathrm{A}} \left\{ \int_0^t \mathscr{L}(y(s), \alpha(s)) \cdot e^{-\lambda s}\, ds + e^{-\lambda t} \cdot v(y(t)) \right\}, \quad \forall t > 0 \tag{1.1}$$

for all $x \in \mathbb{R}^n$. In control theory, this is known as the *Principle of Dynamic Programming.*

Proof

Let us suppose that there is a value α^*, such that

$$v(x) = J(x, \alpha^*) = \int_0^{+\infty} \mathscr{L}(y(t), \alpha^*(t)) \cdot e^{-\lambda t}\, dt \tag{1.2}$$

with $\lambda > 0$, meaning that α^* is the control that guarantees the least possible cost of action for a given trajectory $y(t)$.

For every $t > 0$, we can split the integral as follows:

$$\begin{aligned} J(x, \alpha^*) &= \int_0^t \mathscr{L}(y(s), \alpha^*(s)) \cdot e^{-\lambda s}\, ds + \int_t^{+\infty} \mathscr{L}(y(s), \alpha^*(s)) \cdot e^{-\lambda s}\, ds \\ &= I_1 + \int_t^{+\infty} \mathscr{L}(y(s), \alpha^*(s)) \cdot e^{-\lambda t}\, ds \end{aligned} \tag{1.3}$$

renaming the first integral as I_1 for visual clarity.

We can now make a change of variable in the second integral from s to $s+t$ and rearrange it introducing v calculated in $y(t)$ using the *semigroup property*:

$$\begin{aligned} J(x,\alpha^*) &= I_1 + \int_0^{+\infty} \mathscr{L}(y(s+t),\alpha^*(s+t)) \cdot e^{-\lambda(s+t)}\, ds \\ &= I_1 + e^{-\lambda t} \int_0^{+\infty} \mathscr{L}(y(s+t),\alpha^*(s+t)) \cdot e^{-\lambda s}\, ds \\ &= I_1 + e^{-\lambda t} \cdot v(y(t)) \end{aligned} \tag{1.4}$$

Looking back at Equation 1.1 we can now write

$$\begin{aligned} v(x) &= I_1 + e^{-\lambda t} \cdot v(y(t)) \\ &= \int_0^t \mathscr{L}(y(s),\alpha^*(s)) \cdot e^{-\lambda s}\, ds + e^{-\lambda t} \cdot v(y(t)) \end{aligned} \tag{1.5}$$

Thus, we have proven the *Principle of Dynamic Programming* in the case where an optimal control exists and is unique. If α is not optimal (i.e. $\alpha \neq \alpha^*$), we can invoke the inf function and rewrite the equation as such:

$$v(x) = \inf_{\alpha \in \mathrm{A}} \left\{ \int_0^t \mathscr{L}(y(s),\alpha(s)) \cdot e^{-\lambda s}\, ds + e^{-\lambda t} \cdot v(y(t)) \right\}, \quad \forall t > 0$$

for all $x \in \mathbb{R}^n$. In order to show the validity of the equation, we will now rename the right term of the previous equation:

$$v(x) = \underbrace{\inf_{\alpha \in \mathrm{A}} \left\{ \int_0^t \mathscr{L}(y(s),\alpha(s)) \cdot e^{-\lambda s}\, ds + e^{-\lambda t} \cdot v(y(t)) \right\}}_{w(x)}, \quad \forall t > 0$$

for all $x \in \mathbb{R}^n$, and proceed to prove that $v(x) \geq w(x)$.

Using a similar process to the one used for equations 1.2 and 1.3, we can now generalize those

same relations to compute for all values of α:

$$\begin{aligned} J(x,\alpha) &= \int_0^t \mathscr{L}(y(s),\alpha(s)) \cdot e^{-\lambda s}\, ds + \int_t^{+\infty} \mathscr{L}(y(s),\alpha(s)) \cdot e^{-\lambda s}\, ds \\ &= I_1 + \int_0^{+\infty} \mathscr{L}(y(s+t),\alpha(s+t)) \cdot e^{-\lambda(s+t)}\, ds \\ &= I_1 + e^{-\lambda t} \int_0^{+\infty} \mathscr{L}(y(s+t),\alpha(s+t)) \cdot e^{-\lambda s}\, ds \end{aligned}$$

We introduce a new variable $\tilde{\alpha}(s) = \alpha(s+t)$. By the same logic used in Equation 1.4, it follows that

$$\begin{aligned} v(x) &= I_1 + e^{-\lambda t} \cdot J(y,\tilde{\alpha}) \\ &\geq I_1 + e^{-\lambda t} \cdot v(y) \\ &\geq \omega(x) \end{aligned}$$

therefore

$$v(x) \geq \omega(x).$$

We will now proceed to prove the opposite claim:

$$v(x) \leq \omega(x).$$

Let z be a generic state of y:

$$z = y(t,\alpha)$$

with $\alpha \in \mathrm{A}$. Using the definition of infimum, we can write the generic relation between v and J as

$$v(z) \geq J(z,\alpha_1) - \epsilon \tag{1.6}$$

with $\epsilon > 0$ and $\alpha_1 \in \mathrm{A}$. We will now define a function $\bar{\alpha}(s)$, such that

$$\bar{\alpha}(s) = \begin{cases} \alpha(t) & \text{if } s \leq t \\ \alpha_1(s-t) & \text{if } s > t. \end{cases}$$

Let $\bar{y}, y_1$ be the solutions associated with $\bar{\alpha}, \alpha_1$. It follows that

$$\begin{aligned} v(x) \leq J(x, \bar{\alpha}) &= \int_0^t \mathscr{L}(\bar{y}(s), \bar{\alpha}(s)) \cdot e^{-\lambda s}\, ds + \int_t^{+\infty} \mathscr{L}(\bar{y}(s), \bar{\alpha}(s)) \cdot e^{-\lambda s}\, ds \\ &= I_1 + e^{-\lambda t} \int_0^{+\infty} \mathscr{L}(y(s+t), \alpha(s+t)) \cdot e^{-\lambda s}\, ds \end{aligned}$$

Using the relation in Equation 1.6, we can rewrite it as

$$I_1 + e^{-\lambda t} \cdot J(y_1, \alpha_1) \leq I_1 + e^{-\lambda t} \cdot v(y(t)) + \epsilon.$$

It follows that

$$v(x) \leq \omega(x) + \epsilon.$$

We know that ϵ is arbitrarily small so we can rewrite the previous equation as

$$v(x) \leq \omega(x).$$

The last two demonstrations proved that the *Principle of Dynamic Programming* is correct. The basic intuition for it is that we can associate the value function $v(x)$ with the least cost trajectory and equate it to the total cost up to a moment t, plus the cost associated with the rest of the trajectory. This might seem a trivial point to make but it will allow us to benefit from it when we differentiate it in the following section.

Differentiation

The *Principle of Dynamic Programming* provides a condition that v must meet, but is impractical. For this reason, assuming that v is differentiable, we can deduce a differential version of

the Principle which will turn the problem of determining v into a system of partial differential equations, making it possible to solve them computationally. To achieve this, we will assume that $\alpha = \alpha^*$ and proceed to rearrange Equation 1.5 and divide it by t:

$$\frac{v(x) - e^{-\lambda t} \cdot v(y(t))}{t} - \frac{1}{t} \int_0^t \mathscr{L}(y(s), \alpha^*(s)) \cdot e^{-\lambda s} \, ds = 0.$$

If we introduce a new function $h(t) = e^{-\lambda t} \cdot v(y(t))$, we can rewrite the previous equation as

$$\frac{h(0) - h(t)}{t} - \frac{1}{t} \int_0^t \mathscr{L}(y(s), \alpha^*(s)) \cdot e^{-\lambda s} \, ds = 0.$$

We can now assume t to be approaching zero with positive values ($t \to 0^+$) and rewrite the first addend as a difference quotient. The integral vanishes, leaving only the integrand function, thanks to the *Fundamental Theorem of Calculus*, as the function $\mathscr{L}$ was assumed to be continuous:

$$-\frac{dh(t)}{dt} - \mathscr{L}(y(t), \alpha^*(t)) \cdot e^{-\lambda t} = 0$$

Knowing that $h(t)$ is a product of two functions and assuming that $v \in \mathrm{C}^1([0, +\infty), \mathbb{R})$, we can calculate its derivative accordingly:

$$\lambda e^{-\lambda t} \cdot v(y(t)) - e^{-\lambda t} \cdot \nabla v(y(t)) \cdot f(y(t), \alpha^*(t)) - \mathscr{L}(y(t), \alpha^*(t)) = 0.$$

We can now proceed to compute this equation for $t = 0$ (and therefore $y(0) = x$):

$$\lambda v(x) - \nabla v(x) \cdot f(x, \alpha^*(t)) - \mathscr{L}(x, \alpha^*(t)) = 0. \tag{1.7}$$

This is known as the *Hamilton-Jacobi-Bellman equation for infinite-horizon problems.* Since this relation was derived considering α^* as the optimal control, it can be easily generalized by eliminating the dependence on α^* and introducing an upper bound on the part of the

equation involving the controls:

$$\lambda v(x) + \sup_{\alpha \in \mathrm{A}} \{-\nabla v(x) \cdot f(x, \alpha(t)) - \mathscr{L}(x, \alpha(t))\} = 0.$$

This equation allows us to find the optimal trajectory for reducing cost, as long as the value function is differentiable. It includes a supremum function with negative signs before the terms inside of it, while other textbooks might contain an equivalent formulation with an infimum function and positive signs before the inner terms. But how can we find the control function starting with $v(x)$? For this purpose, we can introduce the *Verification Theorem*:

Theorem 1. *Let $\mathscr{L} \in \mathrm{C}^1$ be a convex function with respect to the controls, let $f(y, u)$ be a linear function with respect to the controls and let $\Phi(y)$ be the classical solution to*

$$\lambda \Phi(y) + H(\nabla_y \Phi(y), y) = 0.$$

It follows that

$$\Phi(y) = v(x).$$

This theorem implies that $v(x)$ is the only valid solution to the *Hamilton-Jacobi-Bellman equation.*

Proof

If we define $\alpha(s)$ to be a control function for all $s \in [0, t]$, we can choose an $x(s)$ function such that

$$\begin{cases} \dot{y}(s) = f(y(s), \alpha(s)) \\ y(0) = x. \end{cases}$$

If $\Phi(y) = \phi(y)e^{-\lambda s}$, it follows that

$$\begin{aligned} -\Phi(y) &= \int_0^{+\infty} \frac{d}{ds}[\Phi(y(s), s)]\, ds \\ &= \int_0^{+\infty} [\nabla\phi(y(s))\dot{y}(s)e^{-\lambda s} - \lambda\phi(y(s))e^{-\lambda s}]\, ds \\ &= \int_0^{+\infty} e^{-\lambda s}[\nabla\phi(y(s))f(y(s), \alpha(s)) - \lambda\phi(y(s))]\, ds. \end{aligned}$$

If we then add $\int_0^{+\infty} e^{-\lambda s}\mathscr{L}(y(s), \alpha(s))\, ds + \Phi(y)$ to both sides, it follows that

$$\begin{aligned} \int_0^{+\infty} e^{-\lambda s}\mathscr{L}(y(s), \alpha(s))\, ds = &\int_0^{+\infty} e^{-\lambda s}[\nabla\phi(y(s))f(y(s), \alpha(s)) - \lambda\phi(y(s))]\, ds \\ &+ \int_0^{+\infty} e^{-\lambda s}\mathscr{L}(y(s), \alpha(s))\, ds + \Phi(y). \end{aligned}$$

We have already defined $H(y, p) = \sup_\alpha\{-p \cdot f(y, \alpha) - \mathscr{L}(y, \alpha)\}$, from which we can derive

$$-H(y, p) \leq -p \cdot f(y, \alpha) + \mathscr{L}(y, \alpha)$$

for all $p \in \mathbb{R}^n$ and for all $\alpha \in$ A. It follows that

$$\int_0^{+\infty} e^{-\lambda s}\mathscr{L}(y(s), \alpha(s))\, ds \leq -\int_0^{+\infty} e^{-\lambda s}[\lambda\Phi(y(s)) + H(\nabla\Phi(y(s)), y)]\, ds + \Phi(y).$$

We know that $\Phi(y)$ is the solution for the *Hamilton-Jacobi-Bellman equations*, therefore we can cancel the integral such that

$$\int_0^{+\infty} e^{-\lambda s}\mathscr{L}(y(s), \alpha(s))\, ds \leq \Phi(y).$$

Introducing the supremum function with respect to the controls, we get

$$v(x) \leq \Phi(y).$$

We will now proceed to prove the opposite statement. Let $r(y)$ be a function defined as

$$r(y) \in \operatorname{argmin}_\alpha \{\mathscr{L}(y,\alpha) + \nabla\Phi(y) f(y,\alpha)\}.$$

Taking advantage of the convexity of $\mathscr{L}(x,\alpha)$ with respect to the controls and considering the linearity of $f(y,\alpha)$ with respect to the controls, the previous equation is well-defined. Let $y(s)$ be the solution to

$$\begin{cases} \dot{y}(s) = f(y(s), r(y(s))) \\ y(0) = x. \end{cases}$$

We proceed as previously done:

$$\int_0^{+\infty} e^{-\lambda s}\mathscr{L}(y(s),\alpha(s))\,ds = \int_0^{+\infty} e^{-\lambda s}[\nabla\phi(y(s)) f(y(s),\alpha(s)) + \mathscr{L}(y(s),\alpha(s)) - \lambda\phi(y(s))]\,ds + \Phi(y).$$

Substituting $\alpha(s)$ with $r(y(s))$, we can minimize the inside of the integral such that

$$\int_0^{+\infty} e^{-\lambda s}\mathscr{L}(y(s),\alpha(s))\,ds \geq -\int_0^{+\infty} e^{-\lambda s}[\lambda\Phi(y(s)) + H(\nabla\Phi(y(s)))]\,ds + \Phi(y).$$

The inside of the integral on the right can be cancelled since $\Phi(y(s))$ is the solution to the *Hamilton-Jacobi-Bellman equation*. It follows that

$$v(x) \geq \Phi(y)$$

which proves our initial statement.

1.3 Viscosity Solutions

We have proven that, in Optimal Control problems, the *Hamilton-Jacobi-Bellman equation* holds when v is a continuous and differentiable function. These equations represent a fundamental and necessary condition to determine such a function, as they allow us to apply the

classical tools of Mathematical Analysis to our problem. Typically, however, the function v is not differentiable but only continuous, as in the case where the controls used are piecewise continuous (e.g. *bang-bang controls*), which are quite common in the field of Control Engineering.

In this case, we need to generalize the concept of derivative in order to define an analogue of the *Hamilton-Jacobi-Bellman equation* for when the value function is continuous but not differentiable. We therefore provide the following definitions:

Definition 3. *The superderivative of the function v is*

$$D^+v(x) = \{p \in \mathbb{R}^n : \limsup_{y \to x} \frac{v(y) - v(x) - p(x-y)}{|x-y|} \leq 0\}.$$

Definition 4. *The subderivative of the function v is*

$$D^-v(x) = \{p \in \mathbb{R}^n : \liminf_{y \to x} \frac{v(y) - v(x) - p(x-y)}{|x-y|} \geq 0\}.$$

We can now ask if splitting the regular derivative with a subderivative and a superderivative is a useful thing to do. To answer this we can evoke the following lemma:

Lemma 1. *If $D^-v(x), D^+v(x)$ are both non-empty sets, then*

$$D^-v(x) = D^+v(x)$$

and v is differentiable in x (and $p = \nabla v$).

Proof

Let us suppose that $D^-v(x), D^+v(x) \neq 0$. Let $p^+ \in D^+v(x), p^- \in D^-v(x)$, then

$$\liminf_{y\to x} \frac{v(y) - v(x) - p^-(x-y)}{|x-y|} \geq 0$$
$$\limsup_{y\to x} \frac{v(y) - v(x) - p^+(x-y)}{|x-y|} \leq 0.$$

If we subtract the second statement from the first we get

$$\limsup_{y\to x} \frac{(p^+ - p^-)(x-y)}{|x-y|} \geq 0$$

which, implying that $x - y = v$, can be rewritten as

$$\limsup_{|v|\to 0} \frac{(p^+ - p^-)v}{|v|} \geq 0.$$

It follows that if $|v| < \epsilon$, then

$$\frac{(p^+ - p^-)v}{|v|} \geq 0.$$

We will now choose v to be described as $v = -\epsilon\frac{p^+ - p^-}{|p^+ - p^-|}$ and its absolute value to be described as $|v| = \epsilon\frac{|p^+ - p^-|}{|p^+ - p^-|} = \epsilon$. Substituting these values in the previous equation we get

$$-\epsilon\frac{|p^+ - p^-|^2}{|p^+ - p^-|} \geq 0.$$

It follows that

$$|p^+ - p^-| \geq 0$$

and therefore that

$$p^+ = p^-$$

and therefore that

$$D^-v(x) = D^+v(x) = \{p\}.$$

The statements 1 and 2 must be true for the same value of p, therefore

$$\lim_{y \to x} \frac{v(y) - v(x) - p(x-y)}{|x-y|} = 0$$

is the definition for the differential $p = \nabla v$. We will now proceed to define a *viscosity solution* as follows:

Definition 5. *Let u be a viscosity solution for the partial differential equation*

$$F(x, u(x), \nabla u(x)) = 0$$

if:

- $F(x, u(x), p) \leq 0$ *for all* $x \in \mathbb{R}^n$ *and for all* $p \in D^+u(x)$*;*
- $F(x, u(x), p) \geq 0$ *for all* $x \in \mathbb{R}^n$ *and for all* $p \in D^-u(x)$*.*

We will now make some propositions in order to prove that the definition of viscosity solution is consistent and equivalent to a classical solution when physical properties are given to the function $u(t)$.

Lemma 2. *If $u \in \mathrm{C}^0(\Omega)$, $\Omega \subset \mathbb{R}^n$ is a viscosity solution, it will be a viscosity solution for all $\Omega' \subset \Omega$.*

Proof

If there exists a function $\phi \in \mathrm{C}^1(\Omega)$ such that the function $u - \phi$ has a local maximum x_0, it follows that it will also be the local maximum of a function $u - \tilde{\phi}$, for all $\tilde{\phi} \equiv \phi$ in the open ball centered at x_0 with a radius R. We can therefore say that

$$\lambda u(x_0) + \sup_{\alpha \in \mathrm{A}} \{-\nabla \phi(x_0) \cdot f(x_0, \alpha(t)) - \mathscr{L}(x_0, \alpha(t))\} \leq 0.$$

It follows that

$$\lambda u(x_0) + \sup_{\alpha \in \mathrm{A}} \{-\nabla \tilde{\phi}(x_0) \cdot f(x_0, \alpha(t)) - \mathscr{L}(x_0, \alpha(t))\} \leq 0.$$

Therefore, u is a viscosity solution for all $\Omega' \subset \Omega$.

Lemma 3. *If $u \in \mathrm{C}^1(\mathbb{R}^n)$ is a viscosity solution, it will be both a subsolution and a supersolution.*

Proof

If $u \in \mathrm{C}^1(\mathbb{R}^n)$ is a viscosity solution, such that

$$\lambda u(x_0) + \sup_{\alpha \in \mathrm{A}} \{-\nabla \phi(x_0) \cdot f(x_0, \alpha(t)) - \mathscr{L}(x_0, \alpha(t))\} \leq 0$$

with $\phi \in \mathrm{C}^1(\mathbb{R}^n)$ such that $u - \phi$ has a local maximum in x_0, it follows that

$$\nabla u = \nabla \phi$$

only if $u \in \mathrm{C}^1$, and therefore

$$\lambda u(x_0) + \sup_{\alpha \in \mathrm{A}} \{-\nabla u(x_0) \cdot f(x_0, \alpha(t)) - \mathscr{L}(x_0, \alpha(t))\} \leq 0$$

for all $x \in \mathbb{R}^n$.

Lemma 4. *If $u \in \mathrm{C}^0(\mathbb{R}^n)$ is a regular solution (differentiable on almost all of its domain), it will be a viscosity solution.*

Proof

If $u \in \mathrm{C}^0(\mathbb{R}^n)$ is a function which is differentiable on almost all of its domain, it follows that $\phi \in \mathrm{C}^1(\mathbb{R}^n)$. The combined statement of these propositions allows us to consider the viscosity solutions as regular solutions, meaning that, from now on, we won't need to operate under

the assumption that u might not be a differentiable function. The function u can finally be derived.

The previous definition for a viscosity solution is equivalent to the following lemma:

Lemma 5. *If $v \in \mathrm{C}^0(\Omega)$ is limited, and $\Omega \subset \mathbb{R}^n$, we can derive that:*

1. *v is a viscosity subsolution and $p \in D^+v(x)$ if and only if there exists a function $\phi \in \mathrm{C}^1(\Omega)$, such that $\nabla\phi(x) = p$ and $(v - \phi)$ has a local maximum;*
2. *v is a viscosity supersolution and $p \in D^-v(x)$ if and only if there exists a function $\phi \in \mathrm{C}^1(\Omega)$, such that $\nabla\phi(x) = p$ and $(v - \phi)$ has a local minimum.*

Proof

To derive the previous lemma, we can assume that $p \in D^+v(x)$. It follows that for

$$\limsup_{y \to x} \frac{v(x) - v(y) - p(x - y)}{|x - y|} \leq 0$$

there exists a value $\delta > 0$ such that $v(x) \leq v(y) + p(x - y) + \sigma(|x - y|) \cdot |x - y|$, with:

- σ being an increasing and continuous function;
- $\sigma(0) = 0$.

We will now proceed to introduce a novel function the use of which will become apparent later in the demonstration:

$$\rho(r) = \int_0^r \sigma(t)\, dt.$$

We can observe that:

1. $\rho(0) = \int_0^0 \sigma(t)\, dt = 0$;
2. $\rho'(r) = \sigma(r) - \sigma(0) = \sigma(r)$, from which it follows that $\rho'(0) = \sigma(0) = 0$;
3. $\rho(2r) = \int_0^r \sigma(t)\, dt + \int_r^{2r} \sigma(t)\, dt \geq \int_0^r \sigma(t)\, dt + \sigma(r) \int_0^r 1\, dt = \rho(r) + \sigma(r) \cdot r \geq \sigma(r) \cdot r$, from which it follows that $\rho(2r) \geq r \cdot \sigma(r)$.

We proceed to introduce another novel funtion ϕ, such that

$$\phi(y) = u(x) + p(x-y) + \rho(2|x-y|)$$

with:

- $\phi \in \mathrm{C}^1$, from which it follows that $\nabla\phi(y) = p$.

We can find that the analytical relation for the local maximum of the function $(u-\phi)$ is

$$\begin{aligned}(u-\phi)(y) &\leq u(x) + p(x-y) + \sigma(|x-y|)\cdot|x-y| - u(x) - p(x-y) - \rho(2|x-y|)\\ &= \sigma(|x-y|)\cdot|x-y| - \rho(2|x-y|)\\ &\leq \sigma(|x-y|)\cdot|x-y| - \sigma(|x-y|)\cdot|x-y|\\ &= 0\\ &= (u-\phi)(x)\end{aligned}$$

which implies that

$$(u-\phi)(y) \leq (u-\phi)(x).$$

It follows that x is a local maximum for all $y \in \mathrm{B}(x,\delta)$. We can prove the second property of Lemma 1 by observing that $D^- u(x) = -D^+\{-u(x)\}$ and repeating the previous calculations.

We can now deal with the scenario in which v might not be differentiable ($v \notin \mathrm{C}^1$) and introduce the following theorem, based on the previous lemma:

Theorem 2. *Let $H : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ be the Hamiltonian given by the formula*

$$H(x,p) = \sup_{\alpha\in\mathrm{A}}\{-f(x,\alpha(t))\cdot p - \mathscr{L}(x,\alpha(t))\}.$$

Considering hypotheses A1 and A2 to be true and $v \in \mathrm{C}^0(\mathbb{R}^n)$, it follows that

$$\lambda v(x) + H(x, \nabla_x\phi(x)) = 0$$

for all ϕ described as in the previous lemma, is a viscosity solution.

We are now going to use these properties to prove the following theorem:

Theorem 3. *If $v \in \mathrm{C}^0(\Omega)$, $\Omega \subset \mathbb{R}^n$ is limited, it follows that*

$$\lambda v(x) + \sup_{\alpha \in \mathrm{A}} \{-p \cdot f(x, \alpha) - \mathscr{L}(x, \alpha(t))\} \leq 0$$

for all $p \in D^+ v(x)$ and for all $x \in \mathbb{R}^n$.

Let us add and subtract $v(y(t))$ from the *Principle of Dynamic Programming*:

$$v(x) = \int_0^t \mathscr{L}(y(t), \alpha^*(t)) \cdot e^{-\lambda t}\, dt + e^{-\lambda t} \cdot v(y(t)) + v(y(t)) - v(y(t)).$$

We can rewrite it introducing a variable $\alpha \in \mathrm{A}$:

$$v(y(t)) - v(x) + \int_0^t \mathscr{L}(y(t), \alpha) \cdot e^{-\lambda t}\, dt + (e^{-\lambda T} - 1) \cdot v(y(t)) \geq 0. \tag{1.8}$$

We know that for all $p \in D^+ v(x)$ and for $t > 0$, it follows that

$$v(y(t)) - v(x) \leq p(y(t) - x) + o(|y(t) - x|).$$

We can now substitute the first two addends of Equation 1.8 with the right-hand side of the previous equation and divide by t:

$$\frac{p(y(t) - x)}{t} + \frac{o(|y(t) - x|)}{t} + \frac{1}{t} \int_0^T \mathscr{L}(y(t), \alpha) \cdot e^{-\lambda t}\, dt + \frac{(e^{-\lambda t} - 1)}{t} \cdot v(y(t)) \geq 0.$$

We can now assume t to be approaching positive zero (calculating the derivatives in $y(0) = x$):

$$p \cdot \dot{y}(0) + \mathscr{L}(x, \alpha) - \lambda v(x) \geq 0.$$

Assuming that $p \in D^+v(x)$ we can rewrite the previous equation as

$$p \cdot f(x, \alpha) + \mathscr{L}(x, \alpha) - \lambda v(x) \geq 0$$

and therefore

$$F(x, v(x), p) \geq 0$$

for all $p \in D^+v(x)$ and for all $x \in \mathbb{R}^n$. We can finally define F as the viscosity solution. We will dedicate the following paragraph to the proof of this non-differentiable version of the *Hamilton-Jacobi-Bellman equation* for the sake of clarity, as it requires the demonstration of some auxiliary results.

Proof for Theorem 2

We will start by proving that if v is continuous, than it will also be a viscosity solution of the *Hamilton-Jacobi-Bellman equation*. Let ϕ be a function such that $\phi \in \mathrm{C}^1(\mathbb{R}^n)$ and x be the local maximum of the function $v - \phi$. It follows that

$$v(x) - \phi(x) \geq v(y) - \phi(y)$$

for all $y \in \mathrm{B}(x, R)$. We can rearrange this relation to be

$$v(x) - v(y) \geq \phi(x) - \phi(y)$$

for all $y \in \mathrm{B}(x, R)$. Let $\alpha(t)$ be constant and $y(t)$ be its associated solution. It follows that for small values of t, $y(t)$ is an element of the open ball centered at x with a radius R. Therefore

$$v(x) - v(y(t)) \geq \phi(x) - \phi(y(t)) \tag{1.9}$$

for all $y \in \mathrm{B}(x, R)$ and for small values of t. We will now subtract $v(y(t))$ from equation 1.5 and rearrange it to be:

$$v(x) - v(y(t)) \leq \int_0^t \mathscr{L}(y(t), \alpha(t)) \cdot e^{-\lambda s}\, ds + v(y(t)) \cdot (e^{-\lambda t} - 1).$$

Looking back at Equation 1.8 we can now say that:

$$\phi(x) - \phi(y(t)) \leq v(x) - v(y(t)) \leq \int_0^t \mathscr{L}(y(t), \alpha(t)) \cdot e^{-\lambda s}\, ds + v(y(t)) \cdot (e^{-\lambda t} - 1).$$

It follows that:

$$\phi(x) - \phi(y(t)) \leq \int_0^t \mathscr{L}(y(t), \alpha(t)) \cdot e^{-\lambda s}\, ds + v(y(t)) \cdot (e^{-\lambda t} - 1).$$

We can now divide by t and assume that it approaches 0 with positive values:

$$\frac{\phi(x) - \phi(y(t))}{t} \leq \frac{1}{t} \int_0^t \mathscr{L}(y(t), \alpha(t)) \cdot e^{-\lambda s}\, ds + v(y(t)) \cdot \frac{(e^{-\lambda t} - 1)}{t}$$

for all $\alpha \in \mathrm{A}$. It follows that:

$$-\frac{d\phi(x)}{dt} \leq \mathscr{L}(x, \alpha(t)) - \lambda v(x)$$

for all $\alpha \in \mathrm{A}$. It follows that:

$$-\nabla\phi(x) \cdot f(x, \alpha(t)) - \mathscr{L}(x, \alpha(t)) + \lambda v(x) \leq 0$$

for all $\alpha \in \mathrm{A}$. We now transform it into an equivalence using the supremum function:

$$\lambda v(x) + \sup_{\alpha \in \mathrm{A}} \{-\nabla\phi(x) \cdot f(x, \alpha(t)) - \mathscr{L}(x, \alpha(t))\} = 0.$$

It follows that:

$$\lambda v(x) + H(x, \nabla\phi) \leq 0.$$

v is therefore a *viscosity subsolution*. We now want to prove that it is also a *viscosity supersolution*. Using Lemma 1 with an opposite logic, we can think of x as the local minimum of $v - \phi$. It follows that:

$$v(x) - v(y) \leq \phi(x) - \phi(y)$$

for all $y \in \mathrm{B}(x, R)$. I can now use the *Principle of Dynamic Programming* ($\geq$) to set $\bar{\alpha} \in \mathrm{A}$ as a generic variable and $\bar{y}$ as its solution and say:

$$v(x) \geq \int_0^t \mathscr{L}(\bar{y}(s), \bar{\alpha}(s)) \cdot e^{-\lambda s}\, ds + v(\bar{y}(t)) \cdot e^{-\lambda t} - t\epsilon \tag{1.10}$$

with $\epsilon > 0$. Following hypotheses A1 and A2, I can say:

$$|\mathscr{L}(\bar{y}(s), \bar{\alpha}(s)) - \mathscr{L}(x, \bar{\alpha}(s))| \leq \omega_{\mathscr{L}}(|\bar{y}(s) - x|, R).$$

Knowing that f is continuous and bounded, we can say that $|\bar{y}(s) - x| \leq Ms$ for small values of s, therefore:

$$|\mathscr{L}(\bar{y}(s), \bar{\alpha}(s)) - \mathscr{L}(x, \bar{\alpha}(s))| \leq \omega_{\mathscr{L}}(Ms, R). \tag{1.11}$$

We know that f is a Lipschitz continuous function from hypotheses A1 and A2, therefore we can derive that:

$$|f(\bar{y}(s), \bar{\alpha}(s)) - f(x, \bar{\alpha}(s))| \leq L(|\bar{y}(s) - x|) \leq LMs.$$

We can proceed to integrate the left side of Equation 1.11 and elaborate it using the triangle inequality:

$$\begin{aligned}\left| \int_0^t \mathscr{L}(\bar{y}(s), \bar{\alpha}(s)) \cdot e^{-\lambda s}\, ds - \int_0^t \mathscr{L}(x, \bar{\alpha}(s)) \cdot e^{-\lambda s}\, ds \right| & \\ \leq \int_0^t |\mathscr{L}(\bar{y}(s), \bar{\alpha}(s)) - \mathscr{L}(x, \bar{\alpha}(s))| \cdot e^{-\lambda s}\, ds & \\ \leq \int_0^t \omega_{\mathscr{L}}(Ms, R) \cdot e^{-\lambda s}\, ds &\end{aligned}$$

We know that:

$$\int_0^t \omega_{\mathscr{L}}(Ms, R) \cdot e^{-\lambda s}\, ds = o(t)$$

therefore:

$$\int_0^t \mathscr{L}(\bar{y}(s), \bar{\alpha}(s)) \cdot e^{-\lambda s}\, ds = \int_0^t \mathscr{L}(x, \bar{\alpha}(s)) \cdot e^{-\lambda s}\, ds + o(t)$$

therefore, substituting in Equation 1.10,

$$v(x) \geq \int_0^t \mathscr{L}(x, \bar{\alpha}(s)) \cdot e^{-\lambda s}\, ds + v(y(t)) \cdot e^{-\lambda t} + o(t).$$

We will now subtract $v(y(t))$ from both sides:

$$v(x) - v(y(t)) \geq \int_0^t \mathscr{L}(x, \bar{\alpha}(s)) \cdot e^{-\lambda s}\, ds + v(y(t)) \cdot (e^{-\lambda t} - 1) + o(t) - \epsilon t.$$

We can once again use Lemma 1 to go from v to ϕ to make sure we are dealing with a differentiable function:

$$\phi(x) - \phi(y(t)) \geq \int_0^t \mathscr{L}(x, \bar{\alpha}(s)) \cdot e^{-\lambda s}\, ds + v(y(t)) \cdot (e^{-\lambda t} - 1) + o(t) - \epsilon t.$$

Let us now divide by t and make it approach zero with positive values:

$$\begin{aligned} -\frac{d\phi(y(t))}{dt} &\geq \mathscr{L}(x, \bar{\alpha}(t)) - \lambda v(y(t)) + \epsilon + o(1) \\ -\nabla\phi(y(t)) \cdot \dot{y}(t) - \mathscr{L}(x, \bar{\alpha}(t)) + \lambda v(y(t)) &\geq \epsilon + o(1) \\ -\nabla\phi(x) \cdot f(x, \bar{\alpha}(t)) - \mathscr{L}(x, \bar{\alpha}(t)) + \lambda v(y(t)) &\geq \epsilon \end{aligned}$$

for all $\bar{\alpha} \in$ A and for all $\epsilon > 0$. If we introduce the supremum function and consider the arbitrariness of ϵ, we are left with

$$\lambda v(x) + H(x, \nabla\phi(x)) = 0.$$

This result finally proves that v is also a *viscosity supersolution* and, therefore, a viscosity solution. This result allows us to derive the *Hamilton-Jacobi-Bellman equation* in its *weak form*. It is called a weak form because the gradient of the function $v(x)$ is substituted with the gradient of a differentiable function $\phi(x)$ which shares a stationary point with it.

1.4 Adding Constraints

The value function $v(x)$ is differentiable, but if we decide to impose constraints it might become not differentiable. We will now consider an equation with constraints:

$$\begin{cases} \dot{y} = f(x, \alpha) \\ y(0) = x. \end{cases}$$

The set of possible controls can be described as

$$A = \{\alpha : [0, +\infty) \to U \mid y(t) \in \bar{\theta}\}$$

with $y(t)$ being the corresponding solution, $U \in \mathbb{R}^m$ and $\bar{\theta}$ being a closed subset of $\mathbb{R}^n$. Therefore we can write our value function as such

$$v(x) = \inf_{\alpha \in A} J(\alpha).$$

We will explain in depth the following statements since their proof is beyond the scope of this thesis. Let us suppose that:

1. There exist $h, r > 0$ and $\eta : \mathbb{R}^n \to \mathbb{R}$ is a continuous and closed function in $\bar{\theta} \subset \mathbb{R}$ such that

   $$B(y + t\eta(y), rt) \subset \theta$$

   for all $y \in \bar{\theta}$ and for all $t \in (0, h]$;

2. $\alpha : [0, +\infty) \to U \subset \mathbb{R}^m$ with U being a compact system;

3. All the previous hypotheses on the f and $\mathscr{L}$ functions are valid.

Definition 6. *$v(x)$ is a closed and continuous function in $\bar{\theta}$. It is a viscosity solution for the constrained problem*

$$\lambda v(x) + H(x, \nabla v(x)) = 0$$

for all $x \in \bar{\theta}$ if $v(x)$ is a subsolution in $\mathring{\theta}$ and a supersolution in $\bar{\theta}$

Theorem 4. *If the conditions 1-3 are true, the closed and continuous value function $v(x)$ is the only constrained viscosity solution for*

$$\lambda v(x) + H(x, \nabla v(x)) = 0.$$

The proof can be found in the referenced papers.

Theorem 5. *Let H_1, H_2 be the Hamiltonians associated with the dynamics f_1, f_2. Let v_1, v_2 be the constrained viscosity solutions for*

$$\lambda v(x) + H_1(x, \nabla v(x)) = 0$$
$$\lambda v(x) + H_2(x, \nabla v(x)) = 0.$$

It follows that

$$\sup_{x \in \theta} |v_1(x) - v_2(x)| \leq \sup_{x \in \bar{\theta}, \alpha \in U} [f_1(x, \alpha) - f_2(x, \alpha)].$$

The previous theorems are used to prove the following theorem:

Theorem 6. *Let us suppose that:*

1. *There exists $\beta > 0$ such that for all $x \in \partial\theta$ and there exists $\alpha \in U$ such that*

$$f(x, \alpha) \cdot \nu(x) \leq -\beta < 0;$$

2. *$\partial\theta \in \mathrm{C}^2$.*

It follows that $v(x)$ is a closed and continuous function in $\bar{\theta}$.

This last theorem proves that there exists a condition for which the value function $v(x)$ is not only a viscosity solution but a constrained viscosity solution. The proof for this statement requires introducing two more lemmas which are beyond the scope of this thesis. Basically, in intuitive terms, adding a constraint of this type is equivalent to tracing an analytical curve around an obstacle and adding a rule which imposes that the trajectory must not cross that boundary. This is possible by making sure that the thrust vector is always pointing in the same direction as the outward-pointing normal vector of the curve, which is guaranteed by the inequality $f(x, \alpha) \cdot \nu(x) \leq -\beta < 0$.

Chapter 2

Classical Numerical Methods

There are several classical methods for solving the *Hamilton-Jacobi-Bellman equation*; in this particular case, two have been chosen based on prior knowledge and the intrinsic elegance of the techniques employed. The first technique we will discuss has its roots in functional analysis, while the second technique is grounded in the field of perturbative methods. Even though we will only use the Semi-Lagrangian method, it is useful to delineate other possible alternatives for illustrative purposes.

2.1 From Viscosity Solutions to Real Solutions

In this section we are going to use a finite difference method (such as *Euler's method*) to define an *operator*, which we will later prove to be a *contraption*. Since we will be dealing with a contraption, we will be able to employ all known theorems in Functional Analysis to define a convergent succession, thereby deriving a solution by iteration based on a finite difference method (*contraption method*) and ask the question: is this viscosity solution v an effective solution?

$$\lambda v(x) + \sup_{a \in \mathrm{A}} \{-\nabla v(x) \cdot f(x, a(t)) - \mathscr{L}(x, a(t))\} = 0$$

Knowing that both f and $\mathscr{L}$ are Lipschitz continuous as in A1 and A2, it follows that:

$$|f(x,a) - f(y,a)| \leq L_f(|x-y| + |a_0 - a_1|) \tag{2.1}$$
$$|\mathscr{L}(x,a) - \mathscr{L}(y,a)| \leq L_{\mathscr{L}}(|x-y| + |a_0 - a_1|).$$

Fixing $h > 0$, we replace the problem described by the differential equations with the following single-step approximation:

$$\begin{cases} y_{n+1} = y_n + h\phi(y_n, A_n, h) \\ y_0 = x \end{cases}$$

where:

- $A_n = (a_n^0, a_n^1, \cdots, a_n^q) \in \mathbb{R}^{M\times(q+1)}$ is the set containing all the admissible controls after the n^{th} time;
- ϕ satisfies the following consistency condition:

  $$\lim_{h\to 0} \phi(x, (a, a, \cdots, a), h) = f(x, a).$$

  This condition becomes necessary for the convergence of the approximation of the system of differential equations;
- ϕ satisfies the following Lipschitz condition

  $$|\phi(x, A, h) - \phi(y, A, h)| \leq L_\phi(|x-y|)$$

  for all A matrices of admissible controls and for all $0 < h < h_0 < \infty$.

Similarly, the Lipschitz condition becomes a sufficient condition for the aforementioned convergence, therefore we can write down the discrete version of the functional J

$$J_h(\{A_n\}) = \sum_{n=0}^{\infty} \sum_{i \in \mathrm{I}} h\omega_i \cdot \mathscr{L}(y_{n+\tau_i}, a_n^i) \cdot \beta^{n+\tau_i}$$

where[1]:

- $\beta = e^{-\lambda h}$;
- $\omega_i \in (0,1)$, $\sum_{i=0} \omega_i = 1$ are the weights associated with the trapezoid formula;
- $\mathrm{I} = \{0, \cdots, q\}$;
- $0 \leq \tau_i \leq 1$ are the knots associated with the trapezoid formula.

The function v will then be:

$$v_h(x) = \inf_{\{A_n\}} J_h(\{A_n\})$$

Lemma 6. *Considering B1, B2 and B3 hypotheses to be correct, it follows that*

$$v_h(x) = \inf_{\{A_n\}} \left\{ \sum_{n=0}^{p-1} \sum_{i \in \mathrm{I}} h\omega_i \cdot \mathscr{L}(y_{n+\tau_i}, a_n^i) \cdot \beta^{n+\tau_i} + v_h(y_p) \cdot \beta^p \right\}$$

for all $p \geq 1$, for all $x \in \mathbb{R}^n$ and with $\beta < 1$. The points $y_{n+\tau_i}$ are intermediate points between y_n and y_{n+1} based on τ_i in order to approximate the differential problem.

[1] This approximation derives from the trapezoid, which states that

$$\left| \int_a^b g(s)\, ds - \sum_{n=0}^{N} \sum_{i \in \mathrm{I}} h\omega_i \cdot g((n+\tau_i) \cdot h) \right| \leq c \left(\frac{b-a}{N} \right)^p$$

Proof

By the definition of $v_h(x)$, for any $\epsilon > 0$, there exists an admissible sequence $\{A_n\}$ which depends on ϵ, such that

$$v_h(x) + \epsilon \geq \sum_{n=0}^{p-1} \sum_{i \in \mathrm{I}} h\omega_i \cdot \mathscr{L}(y_{n+\tau_i}, a_n^i) \cdot \beta^{n+\tau_i} + \sum_{n=p}^{\infty} \sum_{i \in \mathrm{I}} h\omega_i \cdot \mathscr{L}(y_{n+\tau_i}, a_n^i) \cdot \beta^{n+\tau_i} \tag{2.2}$$

where we have denoted by $\bar{y}_n$ the sequence generated by the discrete control $\{A_n\}$. The second term in the right-hand side of the previous equation may be rewritten as such:

$$v_h(x) + \epsilon \geq \sum_{n=0}^{p-1} \sum_{i \in \mathrm{I}} h\omega_i \cdot \mathscr{L}(y_{n+\tau_i}, a_n^i) \cdot \beta^{n+\tau_i} + \beta^p \sum_{n=0}^{\infty} \sum_{i \in \mathrm{I}} h\omega_i \cdot \mathscr{L}(y_{p+n+\tau_i}, a_{p+n}^i) \cdot \beta^{n+\tau_i}.$$

Since ϵ is arbitrary, observing that $\bar{y}_{p+n}$ is the solution of Equation 2.1 related to the control $\{A_{p+n}\}$ and with initial state $\bar{y}_p$, we obtain from Equation 2.2

$$v_h(x) \geq \sum_{n=0}^{p-1} \sum_{i \in \mathrm{I}} h\omega_i \cdot \mathscr{L}(\bar{y}_{n+\tau_i}, \bar{a}_n^i) \cdot \beta^{n+\tau_i} + \beta^p v_h(\bar{y}_p).$$

To prove the reverse inequality, it suffices to observe that, due to the definition of $v_h(x)$, for any sequence $\left\{\hat{A}_n\right\}_{0 \leq n \leq p-1}$ we have

$$v_h(x) \geq \sum_{n=0}^{p-1} \sum_{i \in \mathrm{I}} h\omega_i \cdot \mathscr{L}(\hat{y}_{n+\tau_i}, \hat{a}_n^i) \cdot \beta^{n+\tau_i} + \beta^p \inf_{\{A_n\}} J_{\hat{y}_p}^h(\{A_n\})$$

and hence

$$v_h(x) \geq \sum_{n=0}^{p-1} \sum_{i \in \mathrm{I}} h\omega_i \cdot \mathscr{L}(\hat{y}_{n+\tau_i}, \hat{a}_n^i) \cdot \beta^{n+\tau_i} + \beta^p v_h(\hat{y}_p)$$

which completes the proof. In our specific case, we will assume $p = 1$, $\tau_i = \tau = 0$, $\omega_i = \omega = 1$ and that $\phi(x, A, h) = f(x, a)$. Therefore, Lemma 6 will become

$$v_h(x) = \inf_{a \in \mathrm{A}} \{h\mathscr{L}(x, a) + \beta v_h(x + hf(x, a))\}$$

We will call this a 1^{st} order scheme. This is not the only option, other viable schemes could be the *Runge-Kutta* or *Heun* schemes.

2.1.1 Semi-Lagrangian Method

We will now try to prove the following theorem:

Theorem 7. *If the functions $\mathscr{L}$ and f satisfy the B1 and B2 hypotheses, the discretized Hamilton-Jacobi-Bellman equation has a unique solution $v_h \in \mathrm{L}^\infty(\mathbb{R}^n)$, such that:*

$$\|v_h(x)\| \leq \frac{M_{\mathscr{L}}}{\lambda}$$

Proof

We start by defining $\mathrm{B}(x, R)$ as

$$\mathrm{B}(x, R) = \{v \in \mathscr{L}^\infty(\mathbb{R}^n),\ \|v_h(x)\|_\infty \leq R\}.$$

We will then define the functional T_h applied to the analytical function v as

$$T_h v(x) = \min_A \left[\sum_{i \in \mathrm{I}} h\omega_i \cdot f(x_{\tau_i}, a^i) \cdot \beta^{\tau_i} + \beta \cdot v(x + h \cdot \phi(x, A)) \right].$$

With this definition, the discretized Hamilton-Jacobi-Bellman equation transforms into the following fixed-point equation:

$$v_h(x) = T_h v(x).$$

The first notion that we are going to derive is that

$$T : \mathrm{B}\left(x, \frac{M_{\mathscr{L}}}{\lambda}\right) \to \mathrm{B}\left(x, \frac{M_{\mathscr{L}}}{\lambda}\right)$$

for all $\lambda \in \mathbb{R}^+$.

Proof

If $v \in \mathrm{B}(x, R)$ for all $R > 0$, it follows that

$$\|T_h v(x)\|_\infty = \left\| \sum_{i \in \mathrm{I}} h\omega_i \cdot \mathscr{L}(x_{\tau_i}, a^i) \cdot \beta^{\tau_i} + \beta \cdot v(x + h \cdot \phi(x, A)) \right\|_\infty .$$

By applying the triangle inequality rule, we obtain

$$\|T_h v(x)\|_\infty \leq \sum_{i \in \mathrm{I}} h\omega_i \cdot \left\| \mathscr{L}(x_{\tau_i}, a^i) \right\|_\infty \cdot \beta^{\tau_i} + \beta \cdot \|v(x + h \cdot \phi(x, A))\|_\infty .$$

Knowing that $\mathscr{L}$ is bounded ($\|\mathscr{L}\| \leq M_{\mathscr{L}}$) and $v(x) \in \mathrm{B}(0, R)$ ($\|v(x)\|_\infty \leq R$), we can simplify the equation further as such:

$$\|T_h v(x)\|_\infty \leq hM_{\mathscr{L}} \sum_i \omega_i + \beta R.$$

Therefore,

$$\|T_h v(x)\|_\infty \leq hM_{\mathscr{L}} + \beta R.$$

We will now arbitrarily assign the value $R = \frac{hM_{\mathscr{L}}}{1-\beta}$. The reasoning is that if we substitute this value for R in the previous equation we will get

$$\|T_h v(x)\|_\infty \leq hM_{\mathscr{L}} + \beta \frac{hM_{\mathscr{L}}}{1-\beta}.$$

It follows that

$$\|T_h v(x)\|_\infty \leq \frac{(1-\beta)hM_{\mathscr{L}} + \beta hM_{\mathscr{L}}}{1-\beta}.$$

It follows that

$$\|T_h v(x)\|_\infty \leq \frac{hM_{\mathscr{L}}}{1-\beta}.$$

That is to say

$$\|T_h v(x)\|_\infty \leq R$$

and, therefore,

$$T : \mathrm{B}(x, R) \to \mathrm{B}(x, R)$$

We will now want to prove that the functional T_h is a contraction[2] for some matrix $A = \{a^0, a^1, \cdots, a^n\}$, therefore

$$|T_h v_1(x) - T_h v_2(x)| \leq \left| \sum_{i \in \mathrm{I}} h\omega_i \cdot \mathscr{L}(x_{\tau_i}, a^i) \cdot \beta^{\tau_i} + \beta \cdot v_1(x + h \cdot \phi(x, A)) \right. \\ \left. - \sum_{i \in \mathrm{I}} h\omega_i \cdot \mathscr{L}(x_{\tau_i}, a^i) \cdot \beta^{\tau_i} - \beta \cdot v_2(x + h \cdot \phi(x, A)) \right|$$

which can be simplified as

$$|T_h v_1(x) - T_h v_2(x)| \leq \beta |v_1(x + h \cdot \phi(x, A)) - v_2(x + h \cdot \phi(x, A))|.$$

Taking the supremum of each side, we can rewrite it as

$$\|T_h v_1 - T_h v_2(x)\|_\infty \leq \beta \|v_1 - v_2\|_\infty.$$

Since we know that β is positive and less than 1, we have finally proved that the functional T_h is a contraction. We now know for sure that $T_h v(x) = v_h(x)$ has a unique solution because of the *Banach-Caccioppoli theorem.*[3]

We can now develop an algorithm to find the solution v_h by converging numerically:

$$v_h^{(n+1)} = T_h(v_h^{(n)}) \tag{2.3}$$

with $v_h^{(0)}$ chosen arbitrarily. We know that this method will converge to the right solutions

[2] A functional $F : x \to x$ is defined as a *contraction* when there exists $L_f \leq 1$, such that

$$|F(u) - F(v)| \leq L|u - v|, \quad \forall u, v \in x$$

[3] **Theorem (Banach-Caccioppoli)**: If (X, d) is a metric space and $f : X \to X$ is a contraction, f will have a unique solution to the $f(x) = x$ fixed-point problem.

thanks to *Picard's theorem.*[4] We will now proceed to prove the following theorem:

Theorem 8. *If the hypotheses B1, B2 and B3 are valid, $\mathscr{L}$ is limited and $\lambda \geq L_\phi$, it follows that for all $h \leq h_0$*

$$\|v - v_h\|_\infty \leq C_0 \cdot h^p$$

where p is the order of approximation for the trapezoid method.

We will define the following conditions:

C1. For all $z \in \mathbb{R}^n$ that are in initial conditions and for all $\alpha \in [0, h] \to U$ that are controls there will exist a matrix $A \in \mathbb{R}^{m \times (q+1)}$ and a constant $k_1 > 0$ such that:

$$|y(h, \alpha) - z - h \cdot \phi(z, A)| \leq k_1 \cdot h^{p+1}$$

C2. There exists a constant $k_2 > 0$, such that:

$$\left| \int_0^h \mathscr{L}(y(s), \alpha(s)) \cdot e^{-\lambda s} - \sum_{i \in \mathrm{I}} h\omega_i \cdot \mathscr{L}(z_{\tau_i}, a^i) \cdot \beta^{\tau_i} \right| \leq k_2 \cdot h^{p+1}$$

We know from the *Principle of Dynamic Programming* that

$$v(x) = \inf_{\alpha \in \mathrm{A}} \left\{ \int_0^h \mathscr{L}(y(s), \alpha(s)) \cdot e^{-\lambda s}\, ds + e^{-\lambda h} \cdot v(y(h)) \right\}.$$

If we consider the matrix $A = (a_0, a_1, \cdots, a_p)$ as the minimum of controls given by the discretized *Hamilton-Jacobi-Bellman equation* and $\alpha(s)$ as the first control of the hypotheses C1 and C2, it follows that

$$\begin{aligned} |v(x) - v_h(x)| \leq & \left| \int_0^h \mathscr{L}(y(s), \alpha(s)) \cdot e^{-\lambda s}\, ds - \sum_{i \in \mathrm{I}} h\omega_i \cdot \mathscr{L}(x_{\tau_i}, a^i) \cdot \beta^{\tau_i} \right| \\ & + \beta |v(y(h)) - v_h(x + h \cdot \phi(x, A))|. \end{aligned} \tag{2.4}$$

[4] A method

$$v_h^{(n+1)} = T_h(v_h^{(n)})$$

will converge only if T_h is a contraction.

We can notice how the modules in the right side of the inequality can be both rewritten as

$$\left| \int_0^h \mathscr{L}(y(s), \alpha(s)) \cdot e^{-\lambda s} - \sum_{i \in \mathrm{I}} h\omega_i \cdot \mathscr{L}(x_{\tau_i}, a^i) \cdot \beta^{\tau_i} \right| \leq k_2 \cdot h^{p+1}$$

$$\begin{aligned} |v(y(h)) - v_h(x + h \cdot \phi(x, A))| &\leq |v(y(h)) - v(x + h \cdot \phi(x, A)| \\ &\quad + |(v_h(x + h \cdot \phi(x, A)))| \\ &\quad - |(v_h(x + h \cdot \phi(x, A)))|. \end{aligned}$$

Since we know that the v function is Lipschitz continuous, the right side of the second equation can be written in the following inequality:

$$\begin{aligned} |v(y(h)) - v(x + h \cdot \phi(x, A))| + |(v_h(x + h \cdot \phi(x, A)))| - |(v_h(x + h \cdot \phi(x, A)))| \\ \leq L_v |y(h) - x - h \cdot \phi(x, A)| + \sup_x |v_h(x) - v(x)|. \end{aligned}$$

which can in turn be rewritten as

$$L_v |y(h) - x - h \cdot \phi(x, A)| + \sup_x |v_h(x) - v(x)| \leq L_v k_1 \cdot h^{p+1} + \|v_h - v\|_\infty$$

which means that Equation 2.4 can be rewritten as

$$|v(x) - v_h(x)| \leq k_2 \cdot h^{p+1} + \beta(L_v k_1 \cdot h^{p+1} + \|v_h - v\|_\infty)$$

for all $x \in \mathbb{R}^n$. It follows that

$$\|v - v_h\| \leq k_2 \cdot h^{p+1} + \beta L_v k_1 \cdot h^{p+1} + \beta \|v_h - v\|_\infty$$

for all $x \in \mathbb{R}^n$. It follows that

$$(1 - \beta) \cdot \|v - v_h\| \leq k_2 \cdot h^{p+1} + \beta L_v k_1 \cdot h^{p+1}$$

for all $x \in \mathbb{R}^n$. It follows that

$$\|v - v_h\| \leq C\frac{h^{p+1}}{1-\beta}$$

for all $x \in \mathbb{R}^n$, where $C = k_2 + \beta L_v k_1$. We know that $\beta = e^{-\lambda h}$, therefore $1 - \beta = 1 - e^{-\lambda h}$. If h approaches zero with positive values, we can derive that $1 - \beta = \lambda h$ through a notable limit. Therefore,

$$\|v - v_h\| \leq C\frac{h^{p+1}}{\lambda h}.$$

It follows that

$$\|v - v_h\| \leq C_0\frac{h^{p+1}}{h}.$$

It follows that

$$\|v - v_h\| \leq C_0 \cdot h^p, \quad C_0 = \frac{C}{\lambda}.$$

2.1.2 Linear Case Example

We now know that we can use Equation 2.3 to find a solution v_h. We can then make h approach 0 with positive values and derive the solution v to arbitrary accuracy.

We have shown that the problem of finding solutions to the *Principle of Dynamic Programming* can be reduced to a fixed-point problem, which can be easily solved using the Banach-Caccioppoli theorem. Subsequently, we demonstrated that, under suitable assumptions, there is a convergence comparable to the error incurred when choosing to approximate the value function using the trapezoidal method. Clearly, this is not the only way to approximate such an integral, and an entire family of results on the convergence rate and quality of solutions can be generated. We can employ the contraction method by assuming that the function $f(x, a)$ is linear with respect to position and controls:

$$f(x, a) = Ax + Ba + b$$

with $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$ and $b \in \mathbb{R}^n$. We will also assume that the control vector v_h is

linear with respect to position

$$v_h(x) = Mx + c$$

with $M \in \mathbb{R}^{n\times n}$ and $c \in \mathbb{R}^n$. We can then combine the previous equations and obtain

$$Mx + c = \inf_a \{h\mathscr{L}(x, a) + \beta[M(x + h(Ax + Ba + b)) + c]\}.$$

We can now assume that $a = \alpha^*(x)$, which implies that the inf function is no longer needed since we already have a value which minimizes its content:

$$Mx + c = h\mathscr{L}(x, \alpha^*(x)) + \beta[Mx + MhAx + MhB\alpha^*(x) + Mhb + c].$$

We can recognise this to be an equation in the form

$$v_h^{(n+1)} = T_h\left(v_h^{(n)}\right)$$

with T_h being a contraction and $v_h^{(0)}$ being chosen arbitrarily. We can rearrange it for visual clarity:

$$M^{(n+1)}x^{(n+1)} + c^{(n+1)} = h\mathscr{L}(x, \alpha^*(x)) + \left[\beta M^{(n)} + \beta M^{(n)}hA\right]x + M^{(n)}\beta hB\alpha^*(x) + M^{(n)}hb + c^{(n)}.$$

This recursive equation for the Semi-Lagrangian method will now depend on how the $\alpha^*(x)$ function is behaved, since it has to be derived for any specific case. The more it is continuous in x, the easiest it will be to find a good recursive equation. In each specific case, we will have to assume the values for $\mathscr{L}(x, \alpha^*(x))$ and $f(x, a)$ in order to find the relative $\alpha^*(x)$ function.

The same logic can be used with a control function which is piece-wise continuous, such as

$$v_h(x) = \sum_{i=1}^{N} c_i\eta_i(x)$$

with $\eta_i(x) = M_i x + c_i$. We can then obtain

$$\sum_{i=1}^{N} c_i(M_i x + c_i) = \inf_a \left\{ h\mathscr{L}(x,a) + \beta \left[\sum_{i=1}^{N} c_i(M_i(x + h(Ax + Ba + b)) + c_i) \right] \right\}.$$

We can now assume that $a = \alpha^*(x)$

$$\sum_{i=1}^{N} c_i(M_i x + c_i) = h\mathscr{L}(x, \alpha^*(x)) + \beta \left[\sum_{i=1}^{N} (c_i M_i x + c_i M_i hAx + c_i M_i hB\alpha^*(x) + c_i M_i hb + c_i) \right].$$

This is in the form

$$v_h^{(n+1)} = T_h\left(v_h^{(n)}\right)$$

with T_h being a contraction and $v_h^{(0)}$ being chosen arbitrarily. We can once again rearrange it for visual clarity:

$$\sum_{i=1}^{N} \left(c_i^{(n+1)} M_i^{(n+1)} x^{(n+1)} + \left(c_i^{(n+1)}\right)^2 \right) = h\mathscr{L}(x, \alpha^*(x)) + \left[\sum_{i=1}^{N} \left(\beta c_i^{(n)} M_i^{(n)} + \beta c_i^{(n)} M_i^{(n)} hA \right) \right] x^{(n)} + \sum_{i=1}^{N} \left(\beta c_i^{(n)} M_i^{(n)} hB\alpha^*(x) + \beta c_i^{(n)} M_i^{(n)} hb + \beta c_i^{(n)} \right).$$

As in the previous case, the recursive equation will depend on how we choose $\alpha^*(x)$. This last chapter highlighted that Semi-Lagrangian is not the only method to rely on for solving PDE systems and that there are many ways to apply neural networks if desired.

2.2 Perturbation Method

After exploring a resolution method which utilizes functional analysis, we now adopt a *perturbative method*. This approach belongs to a family of methods that consider the solution to a specific problem as a series of appropriately determined terms. We begin to delve into it by

solving the following equation:

$$A(v) - f(x) = 0$$
$$B(v) = 0$$

with $x \in \mathbb{R}^n$, $A : \mathrm{C}^1 \to \mathbb{R}$ being a differential operator and $B : \mathrm{C}^0 \to \mathbb{R}$ being a linear operator which determines the boundary conditions. We will now proceed to split the differential operator into linear and non-linear components as such:

$$A = L + N.$$

In this way, the previous equation can be rewritten as such:

$$L(v) + N(v) - f(x) = 0. \tag{2.5}$$

We will now define a family of solutions

$$v_\lambda : \mathbb{R}^n \to \mathbb{R},\ \lambda \in [0, 1]$$

for the following equation, which is the convex transform of the previous equation:

$$k(v_\lambda, \lambda) = (1 - \lambda)[L(v_\lambda) - L(v_s)] + \lambda[A(v_\lambda) - f(x)] = 0. \tag{2.6}$$

Let us notice that if $k = 1$, the equation becomes

$$A(v_1) - f(x) = 0$$

while if $k = 0$ the equation becomes

$$L(v_0) - L(v_s) = 0.$$

In this latter case, we can notice how the equation is linear (since L is the linear part of A). We can now proceed to rewrite Equation 2.6 as follows:

$$\begin{aligned} k(v_\lambda, \lambda) &= L(v_\lambda) - L(v_s) - \lambda L(v_\lambda) + \lambda L(v_s) + \lambda L(v_\lambda) + \lambda N(v_\lambda) - \lambda f(x) \\ &= L(v_\lambda) - (1-\lambda) \cdot L(v_s) + \lambda \cdot [N(v_\lambda) - f(x)]. \end{aligned} \tag{2.7}$$

This is known as *homotopy method* since it relies on having two addends in the equation rely on a combination of factors depending on the parameter λ. We will now assume λ to be a small value and suppose that the solution v can be expressed as a power series in terms of it:

$$v = v_0 + \lambda v_1 + \lambda^2 v_2 + \cdots = \sum_{n=0}^{+\infty} \lambda^n v_n. \tag{2.8}$$

If we suppose $\lambda = 1$, then

$$v = v_0 + v_1 + v_2 + \cdots = \sum_{n=0}^{+\infty} v_n. \tag{2.9}$$

Let us now suppose that:

- The 2^{nd} derivative of $N(v)$ is small in value;
- $\|L^{-1}\frac{\partial N}{\partial v}\| < 1$.

We will proceed to prove the following theorem:

Theorem 9. *If $N(v)$ is a non-linear function and $v = \sum_{k=0}^{+\infty} \lambda^k v_k$, then*

$$\frac{\partial^n}{\partial \lambda^n} N(v)_{\lambda=0} = \frac{\partial^n}{\partial \lambda^n} N\left(\sum_{k=0}^{n} \lambda^k v_k\right)_{\lambda=0}.$$

Proof

We know that $v = \sum_{k=0}^{+\infty} \lambda^k v_k$ can be split as follows:

$$\begin{aligned} v &= \sum_{k=0}^{+\infty} \lambda^k v_k \\ &= \sum_{k=0}^{n} \lambda^k v_k + \sum_{k=n+1}^{+\infty} \lambda^k v_k. \end{aligned}$$

We can finally derive that

$$\begin{aligned} \frac{\partial^n}{\partial \lambda^n} N(v)_{\lambda=0} &= \frac{\partial^n}{\partial \lambda^n} N \left(\sum_{k=0}^{n} \lambda^k v_k + \sum_{k=n+1}^{+\infty} \lambda^k v_k \right)_{\lambda=0} \\ &= \frac{\partial^n}{\partial \lambda^n} N \left(\sum_{k=0}^{n} \lambda^k v_k \right)_{\lambda=0}. \end{aligned}$$

The second summation vanishes since a polynomial of at least degree $n+1$ is differentiated n times, always leaving at least one lambda term of degree 1, which is then evaluated at $\lambda = 0$ and disappears. We now introduce the following relations:

$$\begin{aligned} L(v) &= v(x) - f(x) - N(v) = 0 \\ H(v, \lambda) &= (1 - \lambda) \cdot F(v) + \lambda L(v) = 0. \end{aligned}$$

If we substitute the first equation into the second one and define $F(v) = v(x) - f(x)$, we can obtain:

$$H(v, \lambda) = v(x) - f(x) - \lambda N(v) = 0. \tag{2.10}$$

We will now do a Maclaurin expansion of $N(v)$ with respect to λ

$$N(v) = N(v)_{\lambda=0} + \left(\frac{\partial}{\partial \lambda} N(v)_{\lambda=0} \right) \cdot \lambda + \left(\frac{1}{2!} \frac{\partial^2}{\partial \lambda^2} N(v)_{\lambda=0} \right) \cdot \lambda^2 + \cdots + \left(\frac{1}{n!} \frac{\partial^n}{\partial \lambda^n} N(v)_{\lambda=0} \right) \cdot \lambda^n.$$

If we substitute Equation 2.8 into the previous equation we obtain

$$N(v) = N\left(\sum_{k=0}^{+\infty}\lambda^k v_k\right)_{\lambda=0} + \left(\frac{\partial}{\partial\lambda}N\left(\sum_{k=0}^{+\infty}\lambda^k v_k\right)_{\lambda=0}\right)\cdot\lambda + \left(\frac{1}{2!}\frac{\partial^2}{\partial\lambda^2}N\left(\sum_{k=0}^{+\infty}\lambda^k v_k\right)_{\lambda=0}\right)\cdot\lambda^2 + \cdots + \left(\frac{1}{n!}\frac{\partial^n}{\partial\lambda^n}N\left(\sum_{k=0}^{+\infty}\lambda^k v_k\right)_{\lambda=0}\right)\cdot\lambda^n.$$

According to Theorem 4 we derive

$$N(v) = N(v_0) + \left(\frac{\partial}{\partial\lambda}N\left(\sum_{k=0}^{1}\lambda^k v_k\right)_{\lambda=0}\right)\cdot\lambda + \left(\frac{1}{2!}\frac{\partial^2}{\partial\lambda^2}N\left(\sum_{k=0}^{2}\lambda^k v_k\right)_{\lambda=0}\right)\cdot\lambda^2 + \cdots + \left(\frac{1}{n!}\frac{\partial^n}{\partial\lambda^n}N\left(\sum_{k=0}^{n}\lambda^k v_k\right)_{\lambda=0}\right)\cdot\lambda^n. \tag{2.11}$$

Substituting 2.8 and 2.11 into 2.10, and equating the terms with the same order of λ, we obtain

$$\begin{aligned}
&\lambda^0 : v_0(x) - f(x) = 0 \quad \Rightarrow v_0(x) = f(x)\\
&\lambda^1 : v_1(x) - N(v_0) = 0 \quad \Rightarrow v_1(x) = N(v_0)\\
&\lambda^2 : v_2(x) - \frac{\partial}{\partial\lambda}N\left(\sum_{k=0}^{1}\lambda^k v_k\right)_{\lambda=0} = 0 \quad \Rightarrow v_2(x) = \frac{\partial}{\partial\lambda}N\left(\sum_{k=0}^{1}\lambda^k v_k\right)_{\lambda=0}\\
&\quad\vdots\\
&\lambda^{n+1} : v_{n+1}(x) - \frac{1}{n!}\frac{\partial^n}{\partial\lambda^n}N\left(\sum_{k=0}^{n}\lambda^k v_k\right)_{\lambda=0} = 0 \quad \Rightarrow v_{n+1}(x) = \frac{1}{n!}\frac{\partial}{\partial\lambda^n}N\left(\sum_{k=0}^{n}\lambda^k v_k\right)_{\lambda=0}.
\end{aligned}$$

We will now proceed to prove the following theorem:

Theorem 10. *The solution to problem A is the following:*

$$v(x) = f(x) + H_0(v_0) + H_1(v_0, v_1) + \cdots + H_n(v_0, \cdots, v_n)$$

where:

$$H_r(v_0, \cdots, v_r) = -L^{-1}\left(\frac{1}{r!}\frac{\partial^r}{\partial\lambda^r}N\left(\sum_{k=0}^{r}\lambda^k v_k\right)\right), \quad r = 0, \cdots, n$$

are the He operators.

To conclude the chapter, we will present some simple considerations and a small illustrative example. The first thing to note is that this method is non-iterative, so there is no need to evaluate the convergence rate since the solution depends simply on a finite set of non-linear equations that can be solved in various ways. Secondly, it is observed that the He polynomials are well-defined, making it straightforward to construct an approximate solution for the *Hamilton-Jacobi-Bellman equation* in cases where the operators can be easily separated into linear and non-linear parts. What will follow is a one-dimensional example to provide an idea of the method's application.

The last two theorems allow us to formalize our objective:

$$\dot{x} = x + u, \quad x : \mathbb{R}^+ \to \mathbb{R}.$$

We want to find the best control u which minimizes energy as written in the following formula

$$J = \frac{1}{2}\int_0^{+\infty}(x^2(t) + u^2(t)) \cdot e^{-\lambda t}\, dt.$$

We can define the Hamiltonian as follows:

$$\begin{aligned} H(x, p, u) &= \mathscr{L}(x, u) + p \cdot f(x, u) \\ &= \frac{1}{2}(x^2 + u^2) + p \cdot (x + u). \end{aligned}$$

We want to solve the following differential equation

$$\partial_t v + \max_{u \in \mathrm{U}}\{\mathscr{L}(x, u) + \nabla v(x) \cdot f(x, u)\} = 0$$

which can be interpreted as

$$\partial_t v + \max_{u \in \mathrm{U}} \underbrace{\left\{ \frac{1}{2}(x^2 + u^2) + v'(x) \cdot (x + u) \right\}}_{H(x,u)} = 0. \tag{2.12}$$

We will now find the maximum by finding the gradient of its content and find the optimal control u^*:

$$\nabla_u H(x, u) = u + v'(x) = 0$$

from which it follows that

$$u^* = -v'(x). \tag{2.13}$$

Substituting the optimal control in Equation 2.12 we arrive to the following partial differential equation:

$$\partial_t v(t, x) + \frac{1}{2}(x^2 + (v'(x))^2 + v'(x) \cdot (x - v'(x)) = 0.$$

We can elaborate further:

$$\partial_t v(t, x) + \frac{1}{2}x^2 + \frac{1}{2}v'^2(x) + v'(x) \cdot x - v'^2(x) = 0$$

from which it follows that

$$\partial_t v(t, x) + \frac{1}{2}x^2 + v'(x) \cdot x - \frac{1}{2}v'^2(x) = 0. \tag{2.14}$$

If we include the boundary conditions

$$\begin{cases} \partial_t v(t, x) + \frac{1}{2}x^2 + v'(x) \cdot x - \frac{1}{2}v'^2(x) = 0 \\ v(0, x) = 0. \end{cases}$$

we can see how Equation 2.14 can be mapped onto Equation 2.5 as such:

$$\underbrace{\partial_t v(t,x)}_{L} + \underbrace{\frac{1}{2}x^2}_{f(x)} + \underbrace{v'(x)\cdot x - \frac{1}{2}v'^2(x)}_{N} = 0.$$

We will now find the solution to the previous equation by substituting its terms in Equation 2.7:

$$\partial_t v_\lambda - (1-\lambda)\cdot\partial_t v_s + \lambda\cdot[v'_\lambda(x)\cdot x - \frac{1}{2}v'^2(x) - f(x)] = 0$$

which can be rearranged as follows:

$$\partial_t v_\lambda - \partial_t v_s + \lambda\cdot[v'_\lambda(x)\cdot x - \frac{1}{2}v'^2(x) - f(x) + \partial_t v_s] = 0.$$

We will now substitute the terms $v_\lambda, v'_\lambda, v'$ with their relative Taylor series expansion:

$$\sum_{n=0}^{+\infty}\lambda^n\partial_t v_\lambda - \partial_t v_s + \lambda\cdot\left[x\cdot\sum_{n=0}^{+\infty}\lambda^n v'_n(x) - \frac{1}{2}\left(\sum_{n=0}^{+\infty}\lambda^n v'_n(x)\right)^2 - f(x) + \partial_t v_s\right] = 0.$$

We can finally expand the previous sums into a series of equations:

$$\begin{aligned}
&\lambda^0 : \partial_t v_0 - \partial_t v_s = 0\\
&\lambda^1 : \partial_t v_1 + x\cdot v'_1(x) + \frac{1}{2}x^2 + \partial_t v_s - \frac{1}{2}v'^2_0(x) = 0\\
&\quad\vdots\\
&\lambda^n : \partial_t v_n - \frac{1}{2}\sum_{k=0}^{n-1}\partial_x v_{n-k-1} + x\cdot\partial_x v_{n-1} = 0.
\end{aligned}$$

We can see how at every step we find a value v_n. At the end, we can sum all of those values as shown in Equation 2.9 and finally find the value function, and therefore the optimal control as shown in Equation 2.13.

Chapter 3

Introduction to Neural Networks

The International Business Machine (IBM) describes Artificial Neural Networks (ANN) as a mathematical tool which seeks to replicate human brain behaviour, imitating the brain structure and its neural connections to allow the identification of patterns in order to solve common problems in a wide range of fields. To define exactly what an ANN is we have to deal with the concepts of "architecture" and "activation function".

Definition 7. *Let $\mathcal{A}_{n,m}$ be the set of functions going from $\mathbb{R}^n$ to $\mathbb{R}^m$. For all $f \in \mathcal{A}_{n,m}$ there exists a matrix $W \in \mathbb{R}^{n \times m}$ and a vector $b \in \mathbb{R}^m$ such that*

$$f(x) = Wx + b.$$

Despite being very versatile, the following definition shows how the mathematical basis for Simple Neural Networks is remarkably simple.

Definition 8. *A Simple Neural Network is a function $ANN_\Theta : \mathbb{R}^n \to \mathbb{R}^k$*

$$ANN_\Theta(x) = \sigma(a(x)) = \sigma(Wx + b)$$

where $a \in \mathcal{A}_{n,k}$, $\{W, b\} = \Theta$ are the parameters associated with the function (respectively called

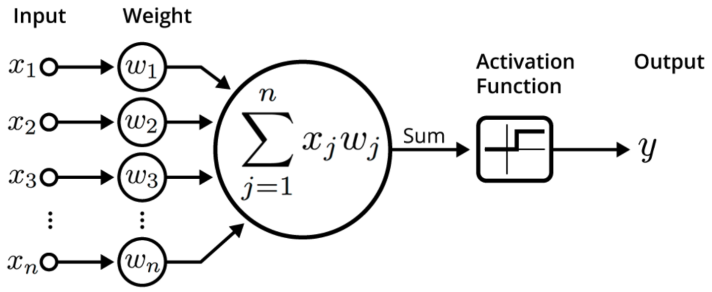

Figure 3.1: Example of an ANN elaborating n inputs and generating a single output through a single neuron.

weights and biases), and $\sigma : \mathbb{R}^k \to \mathbb{R}^k$ *is a non-linear function called activation function.*

Each "activation function-function" couple is called a *neuron* (or *node*). All of the activation functions and weights form the *architecture of the Simple Neural Network*. To enhance our understanding of Simple Neural Network architectures we will now provide the definition of *Rectified Linear Unit* (*ReLU*):

$$ReLU : x \to \max(0, x) = \begin{pmatrix} \max(0, x_1) \\ \max(0, x_2) \\ \vdots \\ \max(0, x_n) \end{pmatrix}.$$

The most widely used ANN is given by the following $(ReLU, \Theta)$ architecture:

$$ANN_\Theta : \mathbb{R}^n \to \mathbb{R}^k, \quad ANN(x) = ReLU(Wx + b) = \begin{pmatrix} \max(0, \sum_{i=1}^n w_{1,i} x_i + b_1) \\ \max(0, \sum_{i=1}^n w_{2,i} x_i + b_2) \\ \vdots \\ \max(0, \sum_{i=1}^n w_{k,i} x_i + b_k) \end{pmatrix}.$$

The neural mechanism is simple: an input signal is processed and each neuron is activated if the processed signal reaches a given amplitude, generating a corresponding output. If the threshold value is not reached, the neuron will not generate an output. In order to generalize

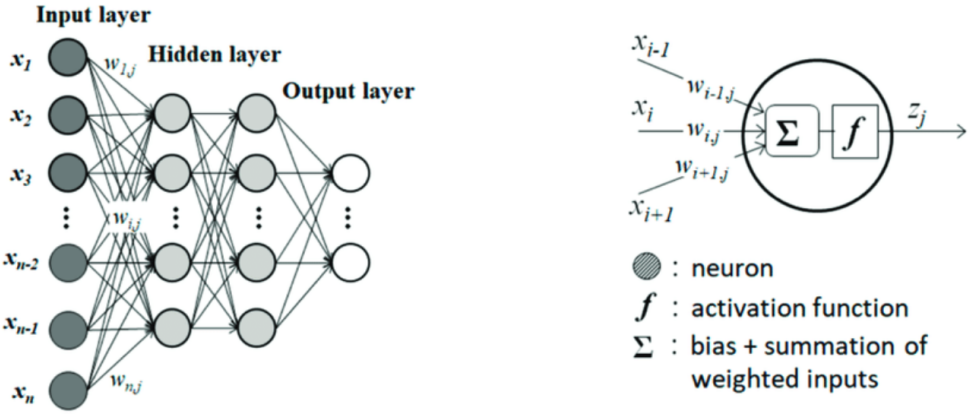

Figure 3.2: On the left, a Deep Neural Network (DNN) processing n inputs and, through a succession of hidden layers, generating two outputs in the output layer. On the right, a schematic representation of the inner workings of a single neuron.

the idea further to make it useful in a wider range of situations, let us introduce the concept of *Deep Neural Network* (*DNN*):

Definition 9. *A* $DNN_\Theta : \mathbb{R}^n \to \mathbb{R}^k$ *function is a Deep Neural Network (DNN) if there exists a succession of integers* $\{n_i\}_{i=1,\cdots,s}$ *where* $n_0 = n$ *and* $n_s = k$*, a succession of activation functions* $\sigma_{i=1,\cdots,s} : \mathbb{R}^{n_i} \to \mathbb{R}^{n_{i+1}}$*, a succession of weight matrices* $\{W^{(i)}\}_{i=1,\cdots,s}$ *and a succession of bias vectors* $\{b^{(i)}\}_{i=1,\cdots,s}$ *both associated with* $\{f_i\}_{i=1,\cdots,s}$ *functions, such that*

$$DNN_\Theta(x) = \prod_{i=1}^{s} \sigma_i(f_i(x)) = \prod_{i=1}^{s} \sigma_i(W^{(i)}x + b^{(i)})$$

where the sequence product is intended as an operation for composing functions.

The weights, as in the previous definition, are indicated with $\Theta = \left(\{W^{(i)}\}_{i=1,\cdots,s}, \{b^{(i)}\}_{i=1,\cdots,s}\right)$. The "activation function-function" couples at the i^{th} step are called *nodes* and their totality makes up the i^{th} layer. The first layer is called the *input layer* while the last layer is called the *output layer*. Some of the fields in which ANNs and DNNs can be applied include linear regressions, classifications, identification of images and patterns, etc. The characteristics of both architectures can be fine-tuned to suit a desired purpose. There are also various families of neural networks, including Sequential Networks,

Recursive Networks, Bidirectional Networks, and GAN Networks. The ones introduced so far belong to the family of Sequential Networks, where the output from each layer serves as the input for the next layer. Now, we will present a theorem highlighting one of the main attractions of neural networks in general: their ability for the universal approximation of data.

Theorem 11. *Let $D \subset \mathbb{R}^n$ be an open set and $\Psi : D \to \mathbb{R}^n$ a differentiable function. Let us suppose that there exists a compact set $K \subset D$ such that any solution $x(t)$ of the following Cauchy problem*

$$\begin{cases} \dot{x}(t) = \Psi(x(t)), \quad t \in [t_0, t_f] \\ x(t_0) = x_0 \in K \end{cases}$$

is still contained in K for all $t \in [t_0, t_f]$, therefore for all $\epsilon > 0$ there exists an integer N and a $DNN_\Theta(t)$ with n outputs and N hidden layers, such that

$$\max_{t \in [t_0, t_f]} |x(t) - DNN_\Theta(t)| < \epsilon$$

The proof of this theorem is beyond the scope of this thesis.

3.1 Optimization

It is abundantly clear that neural networks can be employed in solving problems of Optimal Control such as the *Hamilton-Jacobi-Bellman equation*. Despite having shown the suitability of neural networks for the purpose of this thesis, there are still some questions to be answered:

- What role do weights and biases play in the approximation of functions?
- What are the criteria for choosing the activation functions to obtain a high-performance network? What constitutes as "high-performance"?
- Is there a particular way to tune the parameters to obtain better performances?

Let us suppose to have a succession of data $\{y_j\}_{j=1,\cdots,N}$ collected from any experimental test. We can assert that for any input x_j there is an output y_j, yet we don't know what the relation

between each input and output is, neither which parameters are relevant for constituting that relation. In this case, it is convenient to use neural networks to try and catch the subtle connections among the data. Given an architecture $(\{\sigma_i\}, \Theta)$, we define an associated $DNN_\Theta : \mathbb{R}^n \to \mathbb{R}^k$ as in Definition 8, where $\Theta = (\{W^{(i)}\}, \{b^{(i)}\})$. Let $L : \mathbb{R}^k \times \mathbb{R}^k \to \mathbb{R}$ be a differentiable function called *Loss Function*. The approximation problem for deriving the relation between inputs $X = \{x_j\}_j$ and outputs $Y = \{y_j\}_j$ can be formulated as follows:

$$\min_\Theta \frac{1}{N} \sum_{j=1}^{N} L(DNN_\Theta(x_j), y_j) = \min_\Theta F(X, Y, \Theta). \tag{3.1}$$

The Loss Function measures the quality of the approximation given by the DNN_Θ. The problem is therefore reduced to finding the appropriate Θ parameters to make sure that the Loss Function is minimized. The most widely used method to achieve this is the *gradient descent algorithm*: if $F(X, Y, \Theta)$ is a differentiable function, its point of minimum is obtained by moving in the direction opposite to its gradient. That is to say: fixing an initial condition Θ_0, the algorithm will follow an iterative succession defined by

$$\Theta_{n+1} = \Theta_n - \alpha_n \partial_\Theta F(X, Y, \Theta_n) \tag{3.2}$$

where α_n is the *Learning Rate*, which can be constant or updated during each cycle. In order to demonstrate the efficacy of this method we need to introduce some regularity results.

Definition 10. *A differentiable function $f(\Theta)$ is L-smooth if its gradient satisfies the Lipschitz condition, that is to say*

$$|\partial_\Theta f(\Theta_1) - \partial_\Theta f(\Theta_2)| \leq L|\Theta_1 - \Theta_2|$$

for all $\Theta_1, \Theta_2 \in \mathbb{R}^n$.

A consequence of the previous definition is the following lemma:

Lemma 7. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a two-times differentiable function. If the function f is*

L-smooth, then

$$f(\Theta_1) \leq f(\Theta_0) + \partial_\Theta f(\Theta_0)(\Theta_1 - \Theta_0) + \frac{L}{2}|\Theta_1 - \Theta_0|^2 \tag{3.3}$$

for all $\Theta_0, \Theta_1 \in \mathbb{R}^n$.

Proof

We can derive that

$$\begin{aligned} f(\Theta_1) &= f(\Theta_0) + \int_0^1 \partial_\Theta f(\Theta_0 + \nu(\Theta_1 - \Theta_0))(\Theta_1 - \Theta_0)\, d\nu \\ &= f(\Theta_0) + \partial_\Theta f(\Theta_0)(\Theta_1 - \Theta_0) \\ &\quad + \int_0^1 [\partial_\Theta f(\Theta_0 + \nu(\Theta_1 - \Theta_0)) - \partial_\Theta f(\Theta_0)](\Theta_1 - \Theta_0)\, d\nu \\ &\leq f(\Theta_0) + \partial_\Theta f(\Theta_0)(\Theta_1 - \Theta_0) + L|\Theta_1 - \Theta_0|^2 \int_0^1 \nu\, d\nu. \end{aligned}$$

Solving the last integral brings us to the statement we wanted to prove in the first place.

Lemma 8. *If* $f(\Theta)$ *is L-smooth, then*

$$f\left(\Theta - \frac{1}{L}\partial_\Theta f(\Theta)\right) - f(\Theta) \leq -\frac{1}{2L}|\partial_\Theta f(\Theta)|^2 \tag{3.4}$$

for all $\Theta \in \mathbb{R}^n$.

Proof

Fixing Θ, we will use Equation (3.3) with $\Theta_0 = \Theta$ and $\Theta_1 = \Theta - \frac{1}{L}\partial_\Theta f(\Theta)$ to derive

$$f\left(\Theta - \frac{1}{L}\partial_\Theta f(\Theta)\right) \leq f(\Theta) - \frac{1}{L}|\partial_\Theta f(\Theta)|^2 + \frac{1}{2L}|\partial_\Theta f(\Theta)|^2$$

which is the statement we wanted to prove in the first place.

Lemma 9. *Let* $f(\Theta)$ *be a convex and L-smooth function, then*

$$(\partial_\Theta f(\Theta_1) - \partial_\Theta f(\Theta_0))(\Theta_1 - \Theta_0) \geq \frac{1}{L}|\partial_\Theta f(\Theta_1) - \partial_\Theta f(\Theta_0)|^2. \tag{3.5}$$

Proof

We can derive that, for all $\xi \in \mathbb{R}^n$,

$$\begin{aligned} f(\Theta_1) - f(\Theta_0) &= f(\Theta_1) - f(\xi) + f(\xi) - f(\Theta_0) \\ &\leq \partial_\Theta f(\Theta_1)(\Theta_1 - \xi) + \partial_\Theta f(\Theta_0)(\xi - \Theta_0) + \frac{L}{2}|\xi - \Theta_0|^2 \end{aligned}$$

where the term $f(\Theta_1) - f(\xi)$ is used through the definition of convexity[1] while the term $f(\xi) - f(\Theta_0)$ is used as in Equation 3.3. Given that this approximation is valid for all $\xi \in \mathbb{R}^n$, then

$$\xi = \Theta_0 - \frac{1}{L}(\partial_\Theta f(\Theta_0) - \partial_\Theta f(\Theta_1)).$$

This way we can derive

$$\begin{aligned} f(\Theta_1) - f(\Theta_0) \leq \partial_\Theta f(\Theta_1) \left(\Theta_1 - \Theta_0 + \frac{1}{L}(\partial_\Theta f(\Theta_0) - \partial_\Theta f(\Theta_1)) \right) \\ - \frac{1}{L}\partial_\Theta f(\Theta_0)(\partial_\Theta f(\Theta_0) - \partial_\Theta f(\Theta_1)) \\ + \frac{1}{2L}|\partial_\Theta f(\Theta_0) - \partial_\Theta f(\Theta_1)|^2 \end{aligned}$$

that is to say

$$f(\Theta_1) - f(\Theta_0) \leq \partial_\Theta f(\Theta_1)(\Theta_1 - \Theta_0) - \frac{1}{2L}|\partial_\Theta f(\Theta_0) - \partial_\Theta f(\Theta_1)|^2.$$

The thesis follows by applying what has just been demonstrated by reversing the role of Θ_0 and Θ_1. By adding the estimates together, we obtain

$$0 \leq (\partial_\Theta f(\Theta_1) - \partial_\Theta f(\Theta_0))(\Theta_1 - \Theta_0) - \frac{1}{L}|\partial_\Theta f(\Theta_1) - \partial_\Theta f(\Theta_0)|^2.$$

We can now proceed to prove the following convergence theorem.

[1] By definition, for all $t \in [0, 1]$ and for all $\Theta_0, \Theta_1 \in \mathbb{R}^n$ there exists

$$f(t\Theta_0 + (1-t)\Theta_1) \leq tf(\Theta_0) + (1-t)f(\Theta_1)$$

Theorem 12. *Let $f(\Theta) = F(X, Y, \Theta)$ be a convex and L-smooth function and $\{\Theta_i\}_i$ be defined as in Equation 3.2, then*

$$f(\Theta_n) - f(\Theta^*) \leq \frac{2L|\Theta_0 - \Theta^*|^2}{n+1}$$

where Θ^* is the solution to Equation 3.1.

Proof

We can derive from Equation 3.2 that

$$\begin{aligned} |\Theta_{n+1} - \Theta^*|^2 &= |\Theta_n - \Theta^* - \alpha_n \partial_\Theta f(\Theta_n)|^2 \\ &= |\Theta_n - \Theta*|^2 + \alpha_n^2 |\partial_\Theta f(\Theta_n)|^2 - 2\alpha_n (\Theta_n - \Theta^*) \partial_\Theta f(\Theta_n). \end{aligned} \tag{3.6}$$

Knowing that $\partial_\Theta f(\Theta^*) = 0$, as per Equation 3.5, we can derive that

$$\begin{aligned} (\Theta_n - \Theta^*) \partial_\Theta f(\Theta_n) &= (\Theta_n - \Theta^*)(\partial_\Theta f(\Theta_n) - \partial_\Theta f(\Theta^*)) \\ &\geq \frac{1}{L} |\partial_\Theta f(\Theta_n)|^2 \end{aligned}$$

and therefore Equation 3.6 becomes

$$|\Theta_{n+1} - \Theta^*|^2 \leq |\Theta_n - \Theta^*|^2 + \left(\alpha_n^2 - \frac{2\alpha_n}{L}\right) |\partial_\Theta f(\Theta_n)|^2.$$

Imposing that $\alpha_n - \frac{2}{L} \leq 0$ for all n, we obtain that the succession $|\Theta_n - \Theta^*|^2$ is decreasing. Let us suppose that $\alpha_n = \frac{1}{L}$ and apply Equation 3.4:

$$\begin{aligned} f(\Theta_{n+1}) - f(\Theta_n) &= f\left(\Theta_n - \frac{1}{L} \partial_\Theta f(\Theta_n)\right) - f(\Theta_0) \\ &\leq -\frac{1}{2L} |\partial_\Theta f(\Theta_n)|^2. \end{aligned}$$

We can now reorganize the terms and subtract $f(\Theta^*)$, obtaining

$$f(\Theta_{n+1}) - f(\Theta^*) \leq f(\Theta_n) - f(\Theta^*) - \frac{1}{2L} |\partial_\Theta f(\Theta_n)|^2. \tag{3.7}$$

Contextually, we can use the convexity property

$$\begin{aligned} f(\Theta_n) - f(\Theta^*) &\leq \partial_\Theta f(\Theta_n)(\Theta_n - \Theta^*) \\ &\leq |\partial_\Theta f(\Theta_n)||\Theta_n - \Theta^*| \\ &\leq |\partial_\Theta f(\Theta_n)||\Theta_0 - \Theta^*| \end{aligned}$$

where we employed the notion that the succession $|\Theta_n - \Theta^*|$ is decreasing. Therefore,

$$|\partial_\Theta f(\Theta_n)| \geq \frac{f(\Theta_n) - f(\Theta^*)}{|\Theta_0 - \Theta^*|}$$

which, when substituted in Equation 3.7, brings us to the following estimate

$$f(\Theta_{n+1}) - f(\Theta^*) \leq f(\Theta_n) - f(\Theta^*) - \frac{1}{2L}\frac{(f(\Theta_n) - f(\Theta^*))^2}{|\Theta_0 - \Theta^*|^2}.$$

Let us define $\delta_n = f(\Theta_n) - f(\Theta^*)$. It follows that $\delta_{n+1} \leq \delta_n$ and, therefore,

$$\delta_{n+1} \leq \delta_n - \frac{1}{2L}\frac{\delta_n^2}{|\Theta_0 - \Theta^*|^2}.$$

We can now divide by $\delta_n\delta_{n+1}$ to obtain

$$\begin{aligned} \frac{1}{\delta_n} &\leq \frac{1}{\delta_{n+1}} - \frac{1}{2L}\frac{1}{|\Theta_0 - \Theta^*|^2}\frac{\delta_n}{\delta_{n+1}} \\ &\leq \frac{1}{\delta_{n+1}} - \frac{1}{2L}\frac{1}{|\Theta_0 - \Theta^*|^2} \end{aligned}$$

given that $\frac{\delta_n}{\delta_{n+1}} \geq 1$. To complete the proof, let us sum all the integers $k \leq n$ to obtain

$$\begin{aligned} \frac{1}{\delta_n} &\leq \frac{1}{\delta_n} - \frac{1}{\delta_0} \\ &= \sum_{k=0}^{n} \frac{1}{\delta_k} - \frac{1}{\delta_{k+1}} \\ &\leq -\frac{n+1}{2L|\Theta_0 - \Theta^*|^2}. \end{aligned}$$

That is to say

$$\delta_n \leq \frac{2L|\Theta_0 - \Theta^*|^2}{n+1}.$$

This result can also be applied to non-convex functions after employing certain convexification techniques. However, for the purposes of our discussion, we will restrict ourselves to considering only convex functions, as all the loss functions we will encounter are expected to be convex.

3.2 Neural Methods

After discussing the main characteristics of neural networks we can now define the algorithms that employ them. Let us see what our problem looks like:

$$F(x, v(x), \nabla v(x)) = 0$$

where $F : \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}$ is a smooth function. Our specific scenario can be modelled as such:

$$\lambda v(x) + H(x, \nabla v(x)) = 0$$

for all $x \in \Omega \subset \mathbb{R}^n$, and

$$v(x) = g(x)$$

for all $x \in \partial\Omega$. In order to solve this kind of problem we can utilize the Physics Informed Neural Networks (PINN), a type of neural network that is being used in many physical applications, especially in partial differential equations. In the following two paragraphs we will delve into the specific functioning of PINNs for two kinds of approaches: a *local approach* and a *global approach*.

3.2.1 Local Approach

This approach focuses on a subset $\Omega \subset \mathbb{R}^n$ that is closed and compact. We then consider a mesh of $\{x_i\}_{i\in I}$ evenly distributed points in $\mathring{\Omega}$, a mesh of $\{y_j\}_{j\in J}$ evenly distributed points in

$\partial\Omega$ and a differentiable function $g : \mathbb{R}^n \to \mathbb{R}$. We can now define a candidate solution function $v_\Theta = DNN_\Theta : \mathbb{R}^n \to \mathbb{R}$. Our objective will be to minimize a *loss function*

$$L(\theta) = L_{\text{border}}(\theta) + L_{\text{eq}}(\theta)$$

with

$$L_{\text{border}}(\theta) = \frac{1}{|J|} \sum_{j \in J} \|v_\Theta(x_j) - g(x_j)\|^2$$

and

$$L_{\text{eq}}(\theta) = \frac{1}{|I \cup J|} \sum_{i \in I \cup J} \|\lambda v_\theta(x_i) + H(x_i, \nabla v_\theta(x_i))\|^2$$

therefore

$$L(\theta) = \frac{1}{|J|} \sum_{j \in J} \|v_\Theta(x_j) - g(x_j)\|^2 + \frac{1}{|I \cup J|} \sum_{i \in I \cup J} \|\lambda v_\theta(x_i) + H(x_i, \nabla v_\theta(x_i))\|^2.$$

This is called a local approach since in order to be useful we need to fix a closed and compact Ω set and a g function. This is the approach we will adopt in this research.

3.2.2 Global Approach

In this approach, we let Ω be a closed and compact set and $v_\theta : \mathbb{R}^n \to \mathbb{R}$. If we introduce a set of functions $g(x, \epsilon) : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$, it follows that $v_\theta : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$. We can then take once again $\{x_i\} \subset \mathring{\Omega}$ and $\{x_j\} \subset \partial\Omega$, with $\{\epsilon_k\}_k \subset \mathbb{R}$. In this case, the first addend of our loss function becomes

$$L_{\text{border}}(\theta) = \frac{1}{|J||K|} \sum_{j \in J} \sum_{k \in K} \|v_\Theta(x_j, \epsilon_k) - g(x_j, \epsilon_k)\|^2.$$

The presence of the ϵ variable tells us that we are not taking into consideration just one function for the values at the border but a set of different possible functions. In other words, we can derive a solution which depends on the boundary conditions and therefore can be

easily computed for each possible case. The second addend of our loss function becomes

$$L_{\text{eq}}(\theta) = \frac{1}{|I \cup J||K|} \sum_{i \in I \cup J} \sum_{k \in K} \|\lambda v_\theta(x_i, \epsilon_k) + H(x_i, \nabla v_\theta(x_i, \epsilon_k))\|^2$$

therefore

$$\begin{aligned} L(\theta) = &\frac{1}{|J||K|} \sum_{j \in J} \sum_{k \in K} \|v_\Theta(x_j, \epsilon_k) - g(x_j, \epsilon_k)\|^2 \\ &+ \frac{1}{|I \cup J||K|} \sum_{i \in I \cup J} \sum_{k \in K} \|\lambda v_\theta(x_i, \epsilon_k) + H(x_i, \nabla v_\theta(x_i, \epsilon_k))\|^2. \end{aligned}$$

This can be called a global approach since the term ϵ allows us to find solutions for a whole set of boundary conditions instead of just a specific instance. While the local approach aims to find a singular point in the space of solutions for the Partial Differential Equations, the global approach aims to find a whole subset of the space of solutions.

Chapter 4

Model Definition

Let us imagine a spacecraft trying to dock to a satellite orbiting a planetary object in a circular orbit. To define a docking trajectory, the spacecraft needs to be moved from a known starting position and velocity state, through a given constraint frame, to a final position and velocity state coinciding with the satellite. Our objective will be to do so in an optimal manner, i.e. minimizing a cost functional. Given that this will be an approximate model, we will begin by making some assumptions.

4.1 Assumptions

Assumption 1. *The system is described by the two-body problem approximation.*

Both the spacecraft and the satellite have a negligible mass and their motion is solely influenced by the gravitational pull of the planetary object and is not disturbed by the gravitational fields of other celestial bodies.

Assumption 2. *The planetary object is perfectly spherical.*

Therefore standard gravitational laws don't need to be corrected to account for irregularities in the planetary object's shape.

Assumption 3. *The distance between the spacecraft and the satellite is very small compared*

to the distance between the satellite and the planetary object.

Therefore the spacecraft and the satellite can be considered as following the same circular trajectory.

Assumption 4. *The spacecraft is travelling through a perfect vacuum.*

The radius of the circular orbit is sufficiently large, therefore aerodynamic drag forces can be neglected and the atmospheric pressure considered null. That is to say that we can model the whole environment by just dealing with energetic considerations.

Assumption 5. *The thrust direction can be changed instantaneously in any direction.*

For the sake of simplicity, the mechanical limitations for directional steering of the thrust will be ignored in this model. Of course, this is not a valid assumption to make when building a real GN&C system. Regardless, it is considered a reasonable assumption to make while in the early stages of designing an Optimal Control algorithm.

4.2 Equations of Motion

All vectorial quantities from now on will be displayed in a bold typeface.

We will start by defining an auxiliary reference system with the origin placed in the center of mass of a satellite following a circular orbit of radius $\mathbf{r_1}$ around a planetary object. We will call the distance between the spacecraft and the planetary object $\mathbf{r_2}$ and the distance between the satellite and the spacecraft $\mathbf{r} = \mathbf{r_2} - \mathbf{r_1}$ (as shown in Figure 4.1). Both reference systems share a third axis which completes the right-hand triad which we will call $\hat{\mathbf{e}}_\mathbf{3}$. As already stated in the Assumptions paragraph, we will consider $\mathbf{r} \ll \mathbf{r_1}$. We will start by introducing Newton's Law of Motion for point-like objects in positions $\mathbf{r_1}$ and $\mathbf{r_2}$.

$$\ddot{\mathbf{r}}_\mathbf{1} + \frac{\mu \mathbf{r_1}}{\|\mathbf{r_1}\|^3} = 0$$
$$\ddot{\mathbf{r}}_\mathbf{2} + \frac{\mu \mathbf{r_2}}{\|\mathbf{r_2}\|^3} = \mathbf{F}$$

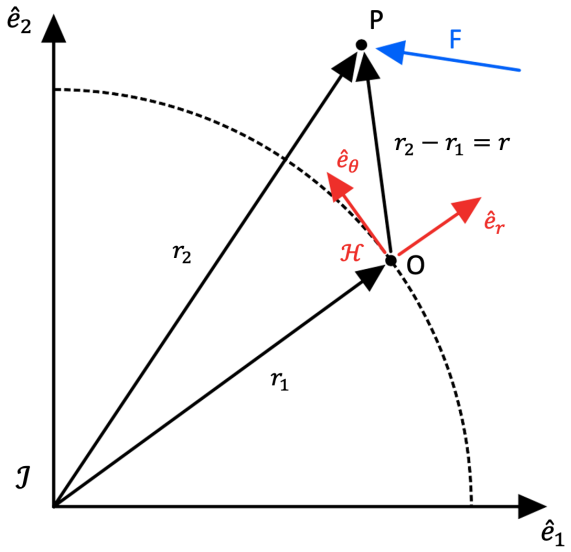

Figure 4.1: Schematic representation of the auxiliary reference system.

The satellite, in position $\mathbf{r_1}$, is not subjected to any perturbation force, while the spacecraft, in position $\mathbf{r_2}$, is subject to a force $\mathbf{F}$ which will be supplied by the thruster controls. By subtracting these two equations we can write

$$\ddot{\mathbf{r}}_\mathbf{2} - \ddot{\mathbf{r}}_\mathbf{1} + \frac{\mu \mathbf{r_2}}{\|\mathbf{r_2}\|^3} - \frac{\mu \mathbf{r_1}}{\|\mathbf{r_1}\|^3} = \mathbf{F}.$$

If we rearrange the equation and substitute $\mathbf{r} = \mathbf{r_2} - \mathbf{r_1}$, we get

$$\ddot{\mathbf{r}} + \frac{\mu}{\|\mathbf{r_1}\|^3}\left(\mathbf{r_2}\|\mathbf{r_1}\|^3\|\mathbf{r_2}\|^{-3} - \mathbf{r_1}\right) = \mathbf{F}.$$

Since the satellite is describing a circular orbit, we know that $\|\mathbf{r_1}\|$ is constant. Therefore, we can substitute $\frac{\mu}{\|\mathbf{r_1}\|^3}$ with a constant n^2 describing the body's mean motion

$$\ddot{\mathbf{r}} + n^2\left(\mathbf{r_2}\|\mathbf{r_1}\|^3\|\mathbf{r_2}\|^{-3} - \mathbf{r_1}\right) = \mathbf{F}. \tag{4.1}$$

Our objective is to have the equation not be dependent on $\mathbf{r_2}$, therefore we can use the

modulus formula to describe it in terms of the other vectors

$$\begin{aligned}\|\mathbf{r_2}\| &= \sqrt{\mathbf{r_2}\cdot\mathbf{r_2}} \\ &= \sqrt{(\mathbf{r}+\mathbf{r_1})\cdot(\mathbf{r}+\mathbf{r_1})}.\end{aligned}$$

It follows that

$$\begin{aligned}\|\mathbf{r_2}\|^{-3} &= [(\mathbf{r}+\mathbf{r_1})\cdot(\mathbf{r}+\mathbf{r_1})]^{-\frac{3}{2}} \\ &= (\|\mathbf{r_1}\|^2 + 2\mathbf{r}\cdot\mathbf{r_1} + \|\mathbf{r}\|^2)^{-\frac{3}{2}}.\end{aligned}$$

We can now find the binomial extension for this expression in order to get a second order approximation for it

$$(x+y)^r = x^r + rx^{r-1}y + \mathcal{O}(r^2).$$

We can now find that the values of the variables are respectively $x = \|\mathbf{r_1}\|^2$, $y = 2\mathbf{r}\cdot\mathbf{r_1} + \|\mathbf{r}\|^2$ and $r = -\frac{3}{2}$. Substituting the values we get

$$\begin{aligned}\|\mathbf{r_2}\|^{-3} &= (\|\mathbf{r_1}\|^2 + 2\mathbf{r}\cdot\mathbf{r_1} + \|\mathbf{r}\|^2)^{-\frac{3}{2}} \\ &= \|\mathbf{r_1}\|^{-3}\left(1 - \frac{3}{2}\left(\frac{2\mathbf{r}\cdot\mathbf{r_1}}{\|\mathbf{r_1}\|^2}\right) + \mathcal{O}(r^2)\right).\end{aligned}$$

We can now ignore the higher order terms and substitute the previous equation into Equation 4.1

$$\ddot{\mathbf{r}} = n^2\left(-\mathbf{r} + 3\frac{\mathbf{r}\cdot\mathbf{r_1}}{\|\mathbf{r_1}\|^2}\mathbf{r_1}\right) + \mathbf{F}$$

Looking back at Figure 4.1, we can notice how all this calculations are done for the inertial reference frame $\mathcal{I}$. We are now interested in finding the same equations of motion for the rotating reference frame $\mathcal{H}$. To do this, we will have to do a change of reference frame. From now on, the notation ${}^{\mathcal{I}}(\cdot)$ and ${}^{\mathcal{H}}(\cdot)$ will be used to distinguish between vectors in different reference frames. We can now write the vectorial equations to change reference frame for

velocity and acceleration

$$
\begin{aligned}
{}^{\mathcal{I}}(\dot{\mathbf{r}}) &= {}^{\mathcal{H}}(\dot{\mathbf{r}}) + \omega \times \mathbf{r} \\
{}^{\mathcal{I}}(\ddot{\mathbf{r}}) &= {}^{\mathcal{H}}\frac{d}{dt}\left({}^{\mathcal{H}}(\dot{\mathbf{r}}) + \omega \times \mathbf{r}\right) + \omega \times \left({}^{\mathcal{H}}(\dot{\mathbf{r}}) + \omega \times \mathbf{r}\right)
\end{aligned}
$$

where ω is the angular velocity between the $\mathcal{I}$ and $\mathcal{H}$ reference systems. The second equation can be expanded to find the usual components of acceleration

$$
{}^{\mathcal{I}}(\ddot{\mathbf{r}}) = {}^{\mathcal{H}}(\ddot{\mathbf{r}}) + {}^{\mathcal{H}}(\dot{\omega}) \times \mathbf{r} + 2\left(\omega \times {}^{\mathcal{H}}(\dot{\mathbf{r}})\right) + \omega \times (\omega \times \mathbf{r})\,.
$$

We know that ${}^{\mathcal{H}}(\dot{\omega}) \times \mathbf{r} = 0$ since the angular rate must be constant in time, given that the reference frame $\mathcal{H}$ rotates with a constant angular velocity equal to the mean motion n^2. We can now solve for ${}^{\mathcal{H}}(\ddot{\mathbf{r}})$ and substitute in the velocity equation to obtain

$$
{}^{\mathcal{H}}(\ddot{\mathbf{r}}) = -2n\hat{\mathbf{e}}_{\mathbf{3}} \times {}^{\mathcal{H}}(\dot{\mathbf{r}}) - n^2(\hat{\mathbf{e}}_{\mathbf{3}} \times (\hat{\mathbf{e}}_{\mathbf{3}} \times \mathbf{r})) - n^2(\mathbf{r} - 3(\hat{\mathbf{e}}_{\mathbf{r}} \cdot \mathbf{r})\hat{\mathbf{e}}_{\mathbf{r}}) + \mathbf{F}
$$

which can be written in vectorial form as

$$
\begin{pmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{pmatrix}_{\mathcal{H}} = \underbrace{\begin{pmatrix} 2n\dot{y} \\ -2n\dot{x} \\ 0 \end{pmatrix}_{\mathcal{H}} + \begin{pmatrix} n^2x \\ n^2y \\ 0 \end{pmatrix}_{\mathcal{H}}}_{\text{Rotating Frame}} - \underbrace{\begin{pmatrix} n^2x \\ n^2y \\ n^2z \end{pmatrix}_{\mathcal{H}} + \begin{pmatrix} 3n^2x \\ 0 \\ 0 \end{pmatrix}_{\mathcal{H}}}_{\text{Gravity Perturbations}} + \underbrace{\begin{pmatrix} F_x \\ F_y \\ F_z \end{pmatrix}_{\mathcal{H}}}_{\text{Other Perturbations}}
$$

which, in turn, can be written as a system of equations inside of the $\mathcal{H}$ reference frame

$$
\begin{cases}
\ddot{x} - 2n\dot{y} - 3n^2x = F_x \\
\ddot{y} + 2n\dot{x} = F_y \\
\ddot{z} + n^2z = F_z
\end{cases}
$$

with

$$F_x = \mathbf{F} \cdot \hat{\mathbf{e}}_\mathbf{r}$$

$$F_y = \mathbf{F} \cdot \hat{\mathbf{e}}_\theta$$

$$F_z = \mathbf{F} \cdot \hat{\mathbf{e}}_\mathbf{3}.$$

These equations are known as *Clohessy-Wiltshire Equations*. We can observe that the Z-axis equation is decoupled from the system and resembles a harmonic oscillator, meaning that any force applied on the Z-axis will cause an oscillation about that same axis. This allows us to neglect this variable as long as we don't generate a thrust with a z component. We can also notice that the position on the Y-axis does not appear in the formulas and therefore does not influence the motion of the system. That is because the y coordinate corresponds to the position along the orbit which, being circular, is perfectly simmetric and therefore is the same everywhere. We can now create an analogous system to solve these 2^{nd} order differential equations in a Python script by keeping track of each differential operator separately

$$\begin{cases} \dot{x} = V_x \\ \dot{y} = V_y \\ \dot{V_x} = 3n^2 x + 2nV_y + F_x \\ \dot{V_y} = -2nV_x + F_y \end{cases}$$

which translates to the following algebraic system

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{V_x} \\ \dot{V_y} \end{pmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 3n^2 & 0 & 0 & 2n \\ 0 & 0 & -2n & 0 \end{bmatrix} \begin{pmatrix} x \\ y \\ V_x \\ V_y \end{pmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{pmatrix} F_x \\ F_y \end{pmatrix}$$

which can be in turn be written as

$$\dot{x} = Ax + Ba$$

where $x \in \mathbb{R}^4$ is the state vector, which gives us information on the position and velocity on the X- and Y-axes; and $a \in \mathbb{R}^2$ is the control vector which gives us information on the magnitude and direction of thrust in the X and Y directions. We can now proceed to define the functional J for our specific purpose, which we will want to minimize in order to optimize the docking trajectory

$$J = \int_0^{+\infty} (x^T Q x + a^T R a) e^{-\lambda t}\, dt$$

where

$$Q = \begin{bmatrix} q_1 & 0 & 0 & 0 \\ 0 & q_2 & 0 & 0 \\ 0 & 0 & q_3 & 0 \\ 0 & 0 & 0 & q_4 \end{bmatrix}, \quad R = \begin{bmatrix} r_1 & r_2 \\ r_3 & r_4 \end{bmatrix}.$$

This implies that

$$\mathscr{L}(y(t), a(t)) = x^T Q x + a^T R a.$$

It follows that

$$v(x) = \inf_{a \in \mathrm{A}} \left\{ \int_0^t (x^T Q x + a^T R a) \cdot e^{-\lambda s}\, ds + e^{-\lambda t} \cdot v(y(t)) \right\}, \quad \forall t > 0.$$

It follows that

$$\lambda v(x) + \inf_{a \in \mathrm{A}} \{ \nabla v(x) \cdot \dot{x} + x^T Q x - a^T R a \} = 0.$$

4.3 Classical Approach

We can now apply the Semi-Lagrangian method and write the discretized version of the control function

$$v_h(x) = \inf_{a \in A} \{ h\mathscr{L}(x, a) + \beta v_h(x + hf(x, a)) \}$$

where $h \ll 1$, $\beta = e^{-\lambda}$ and $\lambda > 0$. Knowing that $\mathscr{L} = x^T Q x + a^T R a$ and $f(x, a) = Ax + Ba$, we can write

$$v_h(x) = \inf_{a \in A} \{h(x^T Q x + a^T R a) + \beta v_h(x + h(Ax + Ba))\}.$$

We can assume the control function to be optimal and remove the infimum function. Since both addends in the right side of the equation are quadratic with respect to x, we can assume $v_h(x)$ to be quadratic with respect to x. It follows that there exist $M \in \mathbb{R}^{n \times n}$, $p \in \mathbb{R}^n$ and $c \in \mathbb{R}$ such that

$$v_h(x) = x^T M x + p^T x + c.$$

It follows that

$$\begin{aligned} x^T M^{(k+1)} x + (p^{(k+1)})^T x + c^{(k+1)} = \inf_{a \in A} \{ & h(x^T Q x + a^T R a) \\ & + \beta [[x + h(Ax + Ba)]^T M^{(k)} [x + h(Ax + Ba)] \\ & + (p^{(k)})^T [x + h(Ax + Ba)] + c^{(k)}]\}. \end{aligned}$$

It follows that the iterative algorithm would become

$$\begin{aligned} x^T M^{(k+1)} x + (p^{(k+1)})^T x + c^{(k+1)} = \inf_{a \in A} \{ & h(x^T Q x + a^T R a) \\ & + \beta [[x + h(Ax + Ba)]^T M^{(k)} [x + h(Ax + Ba)] \\ & + (p^{(k)})^T [x + h(Ax + Ba)] + c^{(k)}]\}. \end{aligned}$$

We can now rearrange the terms inside of the infimum function and take out the terms that are not dependent on the controls function:

$$\begin{aligned} x^T M^{(k+1)} x + (p^{(k+1)})^T x + c^{(k+1)} = h x^T Q x + \beta [& x^T M^{(k)} x \\ & + h x^T M^{(k)} A x + h x^T M^{(k)} B a + h x^T A^T M^{(k)} x \\ & + h^2 x^T A^T M^{(k)} A x + h^2 x^T A^T M^{(k)} B a \\ & + h a^T B^T M^{(k)} x + h^2 a^T B^T M^{(k)} A x \\ & + (p^{(k)})^T x + h (p^{(k)})^T A x + c^{(k)}] \\ & + \inf_{a \in A} \{ h a^T R a + \beta [h^2 a^T B^T M^{(k)} B a + h p^{(k)})^T B a] \}. \end{aligned}$$

Since $h^2 \ll 1$, we can eliminate all the terms that include it:

$$\begin{aligned} x^T M^{(k+1)} x + (p^{(k+1)})^T x + c^{(k+1)} = h x^T Q x + \beta [& x^T M^{(k)} x + h x^T M^{(k)} A x + h x^T M^{(k)} B a \\ & + h x^T A^T M^{(k)} x + h a^T B^T M^{(k)} x \\ & + (p^{(k)})^T x + h (p^{(k)})^T A x + c^{(k)}] \\ & + h \inf_{a \in A} \{ a^T R a + \beta (p^{(k)})^T B a \}. \end{aligned} \tag{4.2}$$

We can now find the optimal control value a^* by taking the gradient of the terms inside of the infimum function and equating it to zero:

$$\begin{aligned} G(a) &= a^T R a + \beta (p^{(k)})^T B a \\ \nabla_a G(a) &= (R + R^T) a + \beta B^T p = 0. \end{aligned}$$

It follows that

$$a^* = -\beta (R + R^T)^{-1} B^T p$$

which is also known as *Linear-Quadratic Control*. We will then be able to remove the infimum function by substituting the optimal value inside of it.

Given that we want to introduce constraints in the differential equation system, in order to

generalize this optimal control we can introduce a matrix function $T(\theta) \in \mathbb{R}^{2\times 2}$

$$T(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

which will allow us to rotate the control thrust vector by changing the θ value. Utilizing the results found in the Section 4 of Chapter 1, we define Θ to be the constraint that needs to be satisfied. We know that the following has to be true

$$\nu(x) \cdot f(x, a^*) < 0 \tag{4.3}$$

for all $x \in \partial\Theta$. In order to do it we will multiply $T(\theta)$ with the control vector and obtain

$$a^* = -\beta T(\theta)(R + R^T)^{-1}B^T p.$$

In our case the Equation 4.3 will become

$$\nu(x) \cdot (Ax - B(\beta T(\theta)(R + R^T)^{-1}B^T p) < 0.$$

We want to chose θ in order to make sure that the previous equation is always valid outside of the constraint Θ. By including a^* in Equation 4.2 and rearranging the terms in order to highlight the variables M, p and c we get

$$\begin{aligned} x^T M^{(k+1)}x + (p^{(k+1)})^T x + c^{(k+1)} \\ &= x^T[hQ + \beta(M^{(k)} + hM^{(k)}A + hA^T M^{(k)})]x \\ &\quad + [\beta((p^{(k)}) + hp^{(k)}A - \beta h((M^{(k)}BT(\theta)(R + R^T)^{-1}B^T p^{(k)})^T \\ &\quad + (T(\theta)(R + R^T)^{-1}B^T p^{(k)})^T B^T M^{(k)})]x \\ &\quad + [\beta(c^{(k)} + \beta h((T(\theta)(R + R^T)^{-1}B^T p^{(k)})^T RT(\theta)(R + R^T)^{-1} \\ &\quad - (p^{(k)})^T BT(\theta)(R + R^T)^{-1})B^T p^{(k)})]. \end{aligned}$$

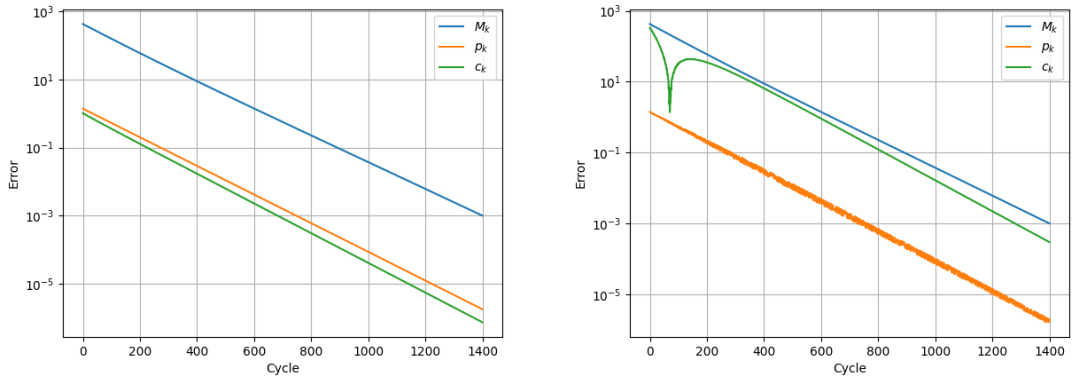

Figure 4.2: Left: Logarithmic plot showing the convergence of the M_k, p_k and c_k errors for $\theta = 0$. Right: Logarithmic plot showing the convergence of the M_k, p_k and c_k errors for random values of θ.

It follows that

$$\begin{aligned}
M^{(k+1)} &= hQ + \beta(M^{(k)} + hM^{(k)}A + hA^T M^{(k)}) \\
p^{(k+1)} &= \beta(p^{(k)} + hp^{(k)}A - \beta h((M^{(k)}BT(\theta)(R + R^T)^{-1}B^T p^{(k)})^T \\
&\quad + (T(\theta)(R + R^T)^{-1}B^T p^{(k)})^T B^T M^{(k)})) \\
c^{(k+1)} &= \beta(c^{(k)} + \beta h((T(\theta)(R + R^T)^{-1}B^T p^{(k)})^T RT(\theta)(R + R^T)^{-1} \\
&\quad - (p^{(k)})^T BT(\theta)(R + R^T)^{-1})B^T p^{(k)}).
\end{aligned}$$

This means that M, p and c can be found iteratively. By importing these functions into a Python script, we can run the iteration and confirm that the values for M_k, p_k and c_k, as well as their associated errors, will converge if $\theta = 0$ as shown in Figure 4.2 (meaning that there is no constraint and $T(\theta)$ will not affect the outcome). In this case the p_k and c_k coefficients converge to null values (as shown in Figure 4.3), meaning that the final equation for the value function $v(x)$ will be

$$v(x) = x^T Mx$$

which is associated to an elliptic paraboloid (as shown in Figure 4.4) and confirms the Banach-Caccioppoli Theorem. It is also important to note that in the case with $\theta = 0$ the rotation

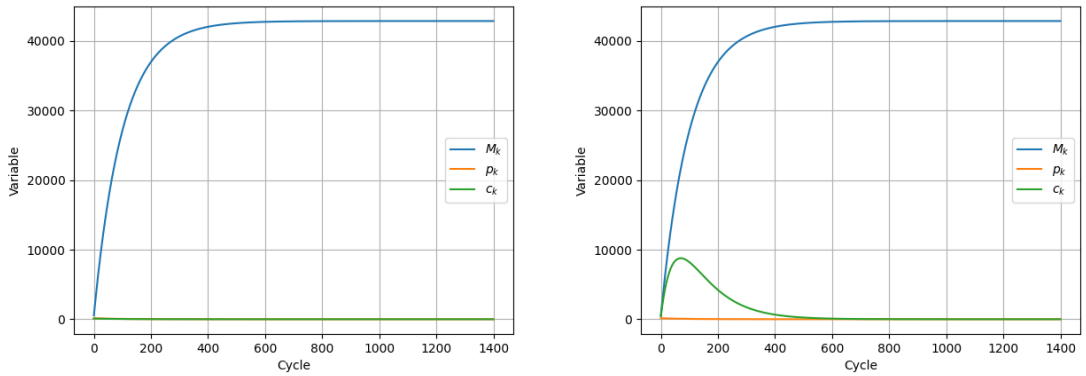

Figure 4.3: Left: Plot showing the convergence of the M_k, p_k and c_k variables for $\theta = 0$. We can also observe how the terms p_k and c_k converge to zero, which is compatible with the Banach-Caccioppoli Theorem introduced in Chapter 2. The M_k matrix also converges to a value which is expected from the theory. Right: Plot showing the convergence of the M_k, p_k and c_k variables for random values of θ. We can also observe how the terms p_k and c_k converge to zero, which is compatible with the Banach-Caccioppoli Theorem introduced in Chapter 2. The M_k matrix also converges to a value which is expected from the theory.

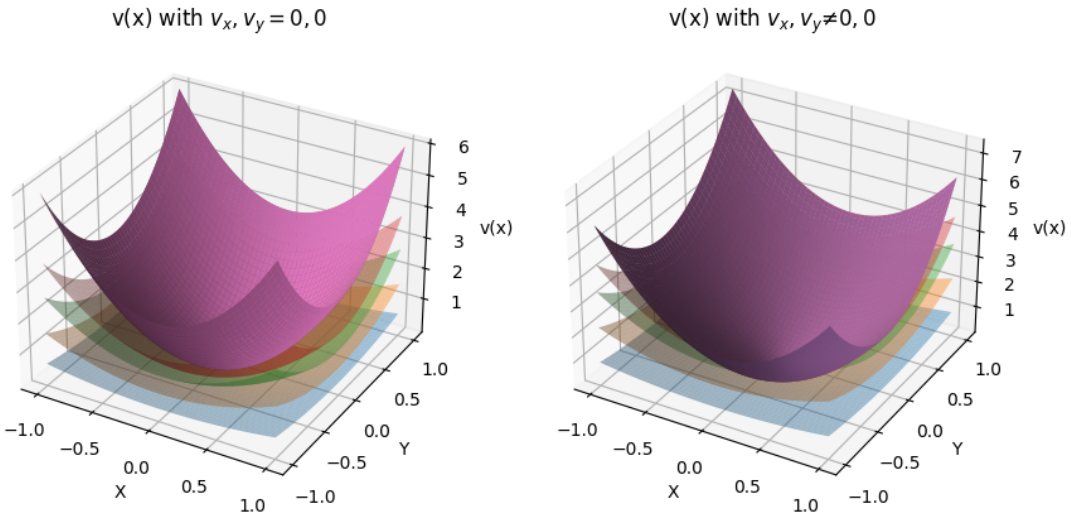

Figure 4.4: Left: 3D plot showing the convergence of the value function $v(x)$ for $\theta = 0$. Right: 3D plot showing the convergence of the value function $v(x)$ for random values of θ.

matrix $T(\theta)$ does not survive the iteration process, which is not a problem since we are not imposing any constraints at this point. But what happens if we try to impose constraints, therefore try to assign values to θ in order to adjust the direction of the thrust vector? In order to verify if the $T(\theta)$ function can affect in a meaningful way the shape of the value function, we will run an iteration where at each step the variable θ is assigned a random value between 0 and 2π. If we run the iteration this way we can see how in the long term the variables p_k and c_k still converge to zero (despite an early influence, as shown in Figure 4.3), leaving only the M_k variable which is not dependent on θ, meaning that in this kind of model there is no way to get a value function which reflects a constraint such as imposing the trajectory to remain constrained to a certain area. The value function will therefore remain the same as in the previous case (as shown in Figure 4.4). At this point, if we wanted to take this model further, we could have the value function be described by a sum of different analytical functions which reflect the constraints we want to apply, such as

$$v_h(x) = \sum_{i=1}^{N} l_i(x)$$

with $l : \Omega_i \to \mathbb{R}$, $\Omega_i \subset \mathbb{R}^n$ and $\Omega_i \cap \Omega_j = \varnothing$, $i \neq j$. We could delve into such a proposition but the calculations would become a lot more complicated. This is why Deep Neural Networks (DNNs) are typically introduced in such cases.

4.4 Neural Approach

We start with the usual formulation for the problem we want to solve

$$\dot{x} = Ax + Ba$$

minimizing

$$J_x(a) = \int_0^{+\infty} (x^T Q x + a^T R a) e^{-\lambda t}\, dt$$

with $\lambda > 0$. In this case the approach is different: the *Principle of Dynamic Programming* is used in its differential form in order to more easily include and differentiate neural networks in the process. It follows that the *Hamilton-Jacobi-Bellman equation* must be valid in the following form:

$$\lambda v_\theta(x) + \inf_{a \in A}\{(x^T Q x + a^T R a) + (\nabla v_\theta(x))^T (Ax + Ba)\} = 0$$

with A being the set of all admissible controls. If we take the derivative of the content of the inf function and equate it to zero we can derive that the optimal value for a is $a^* = -(R^T + R)^{-1} B^T \nabla v_\theta(x)$ and substitute it in the previous equation

$$\lambda v_\theta(x) - x^T Q x - (a^*)^T R a^* - (\nabla v_\theta(x))^T (Ax + Ba^*) = 0.$$

which can be rewritten as

$$\begin{aligned}\lambda v_\theta(x) - x^T Q x \\ - (-\beta T(\theta)(R + R^T)^{-1} B^T p)^T R (-\beta T(\theta)(R + R^T)^{-1} B^T p) \\ - (\nabla v_\theta(x))^T (Ax + B(-\beta T(\theta)(R + R^T)^{-1} B^T p)) = 0.\end{aligned}$$

which can, in turn, be rewritten as

$$\lambda v_\theta(x) + H(x, \nabla v_\theta(x)) = 0 \tag{4.4}$$

where $v_\theta(x) : \mathbb{R}^n \to \mathbb{R}$ is the only unknown variable which we can now isolate.

The most important step in building a neural network is to identify a *loss function*, which in this case will help solve our system of differential equations. As seen in Chapter 3, Section 2, the loss function for our particular case can be obtained directly from Equation 4.4 as such:

$$L_{dyn}(\Theta) = \frac{1}{N} \sum_{i=1}^{N} |\lambda v_\theta(x_i) - H(\nabla v_\theta(x_i))|^2.$$

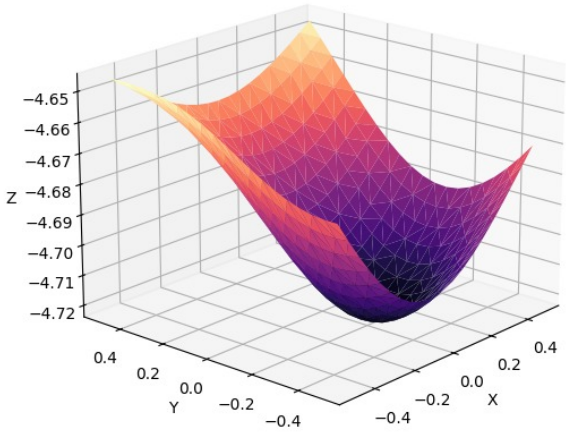

Figure 4.5: Plot showing the value function $v(x)$ without constraints obtained using a Physics Informed Neural Network (PINN). The result is very similar to the paraboloid found in the classic approach.

If we run the PINN with no constraints we can obtain a very similar value function to the one found through the classical approach (Figure 4.5), albeit more generalized and not analytically defined.

Imposing constraints

Let us now try to define $\Theta \subset \mathbb{R}^n$ as a topological constraint and $\{x_i\}_{i=1,\cdots,N} \subset \Theta$ which, in our example, will be

$$\Theta = \{x \in \mathbb{R}^n \,|\, \|x\| < 1\}$$

which is a circumference around the target point. The definition of each position x_i can be written in the polar coordinates

$$\alpha_i \in \{\alpha_i \,|\, \alpha_i \in [0, 2\pi]\}$$

$$\rho_i \in \{\rho_i \,|\, \rho_i \in [0, 1]\}$$

as

$$x_i = (\rho_i \cos \alpha_i, \rho_i \sin \alpha_i).$$

We utilized these definitions to create a loss function which imposed constraints based on the position relative to the unitary circle previously defined:

$$L_{dyn} = k_1 L_{diff} + (1 - k_1 - k_2) L_{BC} + k_2 L_{dots}$$

with $k_1 = 4 \cdot 10^{-1}, k_2 = 1 \cdot 10^{-4}$, where:

- $L_{diff}(\Theta) = \frac{1}{N} \sum_{i=1}^{N} |\lambda v_\theta(x_i) - H(\nabla v_\theta(x_i))|^2$, as shown in Chapter 3, Section 2;
- L_{BC} imposes the boundary conditions such that the value function $v(x)$ is equal to zero when on the edge of the circumference;
- L_{dots} which defines the way θ (contained in the *Hamilton-Jacobi-Bellman equation* thanks to the inclusion of the rotation matrix) is constrained.

The gradient $\nabla v_\theta(x_i)$ can be calculated exactly thanks to the *back-propagation* algorithm of neural networks: since the DNN is a composite function made up of known analytical functions, there is no need to approximate its gradient by using a numerical method such as the Finite Difference method, which would introduce an error in the result. Having defined the loss function, we can finally define the weights and biases Θ_0 associated with the nodes of our neural network, which will be small and random. The iterative process through which we will converge to the optimal values for the weight and biases will be

$$\Theta_{k+1} = \Theta_k - \alpha \nabla_\Theta L_{dyn}(\Theta).$$

As we specified earlier the gradient of $L_{dyn}(\Theta)$ does not introduce an error. We can now finally proceed to show the results for the simulation without constraints in order to compare it to the Classical Approach. As seen in Figure 4.6, the total loss function converges to zero, meaning that our trained Neural Network was successful in optimizing the value of $v_\theta(x)$ as

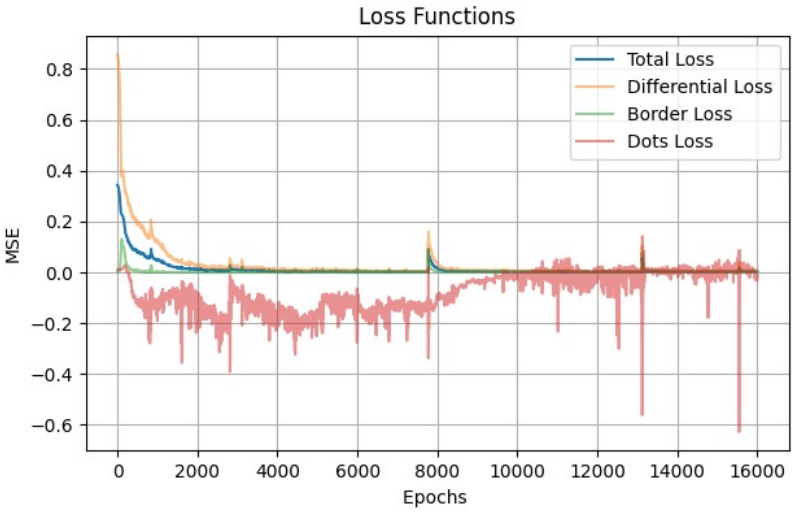

Figure 4.6: Plot showing the convergence of the loss function and its components.

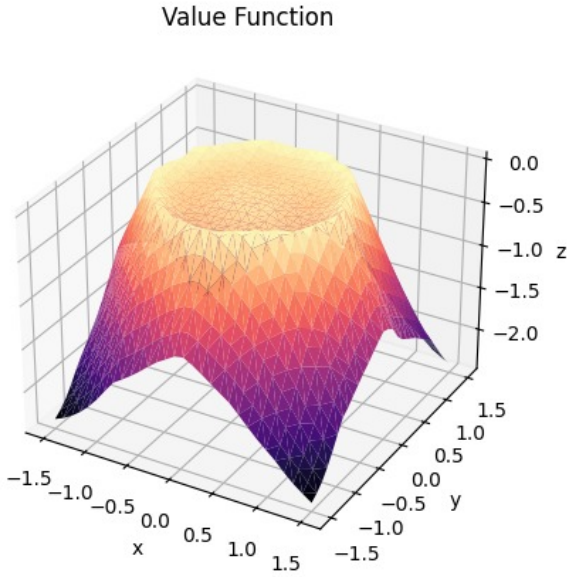

Figure 4.7: Plot showing the value function $v(x)$ with a constraint $\|x\| < 1$ (a circumference around the target point) obtained using a Physics Informed Neural Network (PINN).

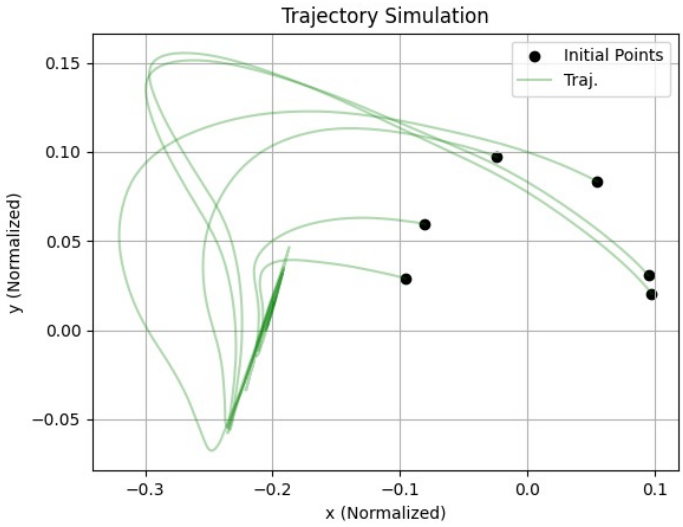

Figure 4.8: Plot showing the convergence of trajectories from different starting positions using Gradient Descent.

we expected. From the graph of the value function $v_\theta(x)$ (shown in Figure 4.7) we can see how the constraints are effectively inforced: if the spacecraft is inside of the circumference, it will tend to fall into the origin of the graph (where the target sits). If, on the other hand, the spacecraft is outside of the circumference, it will tend to fall in the opposite direction of the target. Therefore, in this case, we will want our initial position to be inside of the circumference. Finally, if we look at the computation for the trajectories (shown in Figure 4.8) we can see how starting from different initial points results in the trajectories converging into a single location. The location is slightly translated for the origins from a failure in accounting for the asymmetry which arises from the motion equations.

Conclusions

The findings of this study underscore the effectiveness of Physics Informed Neural Networks (PINNs) in modeling dynamic systems requiring adaptability such as spacecraft docking in space. Specifically, our analysis reveals that PINNs excel in scenarios where incorporating constraints is essential, as they streamline the integration of such constraints into the modeling process. Unlike traditional methods that may necessitate the complex task of approximating value functions through a sum of analytical functions, PINNs offer a more straightforward approach by directly incorporating the appropriate constraints.

Moving forward, there is ample room for further development in the application of PINNs to docking maneuvers. While our demonstration in this thesis introduced a constraint for illustrative purposes, it lacked practicality. A more realistic constraint, such as a cone-shaped region centered around the target point to prevent spacecraft deviation, could significantly enhance the utility of PINNs in docking maneuvers.

Additionally, an oversight in our approach is evident from the trajectories' graphs, where the final points exhibit slight translation from the origin. This oversight stems from the failure to account for the asymmetry of the dynamic model, which arises from centering the reference frame on the target point. Future research efforts should prioritize addressing such nuances to improve the accuracy and applicability of PINN-based models.

Bibliography

[1] S. Cai, Z. Mao, Z. Wang, M. Yin, and G. E. M. Karniadakis. Physics-informed neural networks (PINNs) for fluid mechanics: a review. *Acta Mechanica Sinica*, 37:1727–1738, 2021.

[2] S. Di Cairano, H. Park, and I. Kolmanovsky. Model Predictive Control Approach for Guidance of Spacecraft Rendezvous and Proximity Maneuvering. *International Journal of Robust and Nonlinear Control*, 22(12):1398–1427, 2012.

[3] P. Cannarsa and C. Sinestrari. *Semiconcave Functions, Hamilton-Jacobi Equations, and Optimal Control*. Birkhäuser, Basel, 2004.

[4] M. Falcone. A Numerical Approach to the Infinite Horizon Problem of Deterministic Control Theory. *Applied Mathematics and Optimization*, 15:1–13, 1987.

[5] M. Falcone and R. Ferretti. Discrete time high-order schemes for viscosity solutions of Hamilton-Jacobi-Bellman equations. *Numerische Mathematik*, 67:315–344, 1994.

[6] H. Frankowska and M. Mazzola. Discontinuous solutions of Hamilton-Jacobi-Bellman equation under state constraints. *Calculus of Variations and Partial Differential Equations*, 46:725–747, 2013.

[7] K. Funahashi and Y. Nakamura. Approximation of Dynamical Systems by Continuous Time Recurrent Neural. *Neural Networks*, 6(6):801–806, 1993.

[8] E. Kharazmi, Z. Zhang, and G. E. M. Karniadakis. VPINNs: Variational physics-informed neural networks with domain decomposition. *Computer Methods in Applied Mechanics and Engineering*, 374, 2021.

[9] D. E. Kirk. *Optimal Control Theory: An Introduction*. Dover Publications, New York, 1970.

[10] G. S. Misyris, A. Venzke, and S. Chatzivasileiadis. Physics-Informed Neural Networks for Power Systems. *IEEE Power & Energy Society General Meeting*, 2020.

[11] H. Mete Soner. Optimal Control with State-Space Constraint I. *SIAM Journal on Control and Optimization*, 24(3), 1986.

[12] S. Wang, Y. Teng, and P. Perdikaris. Understanding and Mitigating Gradient Flow Pathologies in Physics-Informed Neural Networks. *SIAM Journal on Scientific Computing*, 43(5), 2021.