Optimal Control of Inverted Pendulum

Degree Project in Engineering Physics, First Cycle, SA104X
Department of Mathematics, section of Numerical Analysis
Royal Institute of Technology

Author: Oskar Stattin, ostattin@kth.se
Supervisor: Mattias Sandberg
Examiner: Mårten Olsson
Abstract

This report investigates two methods of finding the optimal control of an inverted pendulum with a quadratic cost functional.

In the first method a discretisation of a Hamiltonian system is taken as a symplectic Euler-scheme for Newton’s method which is used to find the optimal control from an initial guess. According to the Pontryagin principle this gives the optimal control, since the solution to a Hamiltonian system gives the optimum to a control problem. The second method uses the matrix Riccati differential equation to find the optimal control for a linearised model of the pendulum.

The result was two programs that find the optimal control. The first method’s program demands clever initial guesses in order to converge. The linearised model’s solutions are only valid for a limited area, which turned out to be surprisingly large.
Sammanfattning

I den här rapporten implementeras två metoder för att finna den optimala styrningen av en inverterad pendel med kvadratisk kostnadsfunktional.


Resultatet var två program som finner den optimala styrningen. Den första metodens program kräver smarta startgissningar för att konvergera. Den lineariserade modellens lösningar har ett begränsat giltighetsområde som visade sig vara överraskande stort.
# Contents

1  Introduction ........................................ 2  
   1.1 Problem formulation ............................. 2  
2  Background ........................................ 3  
   2.1 Introduction to optimal control .................. 3  
   2.2 Pontryagin’s principle ............................ 4  
   2.3 The Riccati equation ............................... 5  
   2.4 Newton’s method .................................... 6  
   2.5 Model of the inverted pendulum ................... 7  
3  Method and implementation .......................... 9  
   3.1 Construction of the non-linear model ............... 9  
   3.2 Construction of the linear model ................. 10  
   3.3 Implementations ................................... 11  
4  Discussion and results ............................... 14
1 Introduction

As automated control technologies such as autonomous vehicles, robotics and process control are spreading, the need for optimal control grows. Whether it is to optimise the energy used, the time elapsed, or another resource, using optimal control to enhance performance has become essential to stay competitive.

Optimal control has been a field of study since its formulation mid 20th century, and has been much aided by the increase of computing power and development of efficient algorithms. Its different areas of application are diverse, and ranges from simple dynamic systems such as the inverted pendulum, to more complex ones such as aircrafts.

The inverted pendulum is one of the most studied examples of a dynamical system with an unstable equilibrium. Its variations are as many as the different control laws used to stabilise it.

1.1 Problem formulation

The main concern of this report will be finding the optimal control of an inverted pendulum. One way of doing this is by linearising the differential equation for the pendulum, and using the matrix Riccati equations to find an optimal control. Another way is by using the symplectic Euler method, as described in [3].

The goals are thus:

- Implement a program finding the optimal control of an inverted pendulum, using
  1. symplectic Euler method
  2. the matrix Riccati differential equation.
2 Background

In this section the necessary theoretical groundwork for the report is done, consisting of a brief introduction to optimal control theory, the Riccati equation, Newton’s method, and our model of the inverted pendulum.

2.1 Introduction to optimal control

Optimal control is a branch of control theory aiming to optimise the control of a system. The goal is to steer the system’s dynamics, modelled by differential equations, using a control function called $u$. When this control is the optimal control, it will be denoted $u^\ast$. Below follows the mathematical framework of an optimal control problem.

Given a system with state trajectory $X(s)$ modelled by some differential equation with initial values $X_0$, the problem can be written

$$X'(s) = f(X(s), u(s)), \quad X(0) = X_0.$$  \hspace{1cm} (1)

The dimension of the control and the state space is given by

$$u : [0, T] \to B \subset \mathbb{R}^m, \quad x : [0, T] \to \mathbb{R}^d.$$  \hspace{1cm} (2)

$B$ is the set of accepted controls, and $T$ is the end time. The objective is to minimise the functional

$$g(X(T)) + \int_0^T h(X(s), u(s))ds,$$  \hspace{1cm} (3)

where $g$ is called a terminal cost, the positional goal of our state trajectory, and $h$ is called a running cost. This running cost is the "resource" we want to minimise. Minimising this functional is done by adjusting the control function and this is how the control is made optimal.

The value function, $v(x, t)$ is introduced,

$$v(x, t) := \inf_{X(t) = x, u \in B} \left[ g(X(T)) + \int_t^T h(X(s), u(s))ds \right].$$  \hspace{1cm} (4)

This function is the viscosity solution of the Hamilton-Jacobi-Bellman equation [2].
\[ v_t(x, t) + H(v_x(x, t), x) = 0, \quad (x, t) \in \mathbb{R}^d \times (0, T), \]
\[ v(x, T) = g(x), \quad x \in \mathbb{R}^d. \quad (5) \]

The gradient with respect to the spatial variable \( x \) is denoted \( v_x \), and similarly \( v_t \) is the partial derivative with respect to \( t \). The function \( H \) is the Hamiltonian, defined as

\[ H(\lambda, x) = \min_{u \in B} \left( \lambda \cdot f(x, u) + h(x, u) \right), \]
\[ \lambda = v_x(X(t), t). \quad (6) \]

A solution to the HJB-equation gives us the optimal control. A thorough proof of the optimality granted by the value function solving the HJB equation can be found in Theorem 38 in *Mathematical Control Theory* by Sonnag [5].

The strength of this approach is that the HJB equation turns the problem into an optimisation at each stage. For every pair of \( X \) and \( t \) a control \( u \) is found that minimises the Hamiltonian and solves the HJB equation.

### 2.2 Pontryagin’s principle

To arrive at the symplectic Euler method mentioned in the problem formulation, something more is needed, and it is the Pontryagin principle that will give us this something. This principle states that if we have the optimal control, \( u^* \) (which strictly should be denoted \( u^*(s) \)), and the optimal trajectory, \( X(s) \), of a problem, and find a \( \lambda \) solving the equation

\[-\lambda'(s) = f_x(X(s), u^*) \cdot \lambda + h_x(s, X(s), u^*),\]
\[\lambda(T) = g_x(X(T)),\]

then this \( \lambda \) satisfies the equation

\[-f(X(s), u^*) \cdot \lambda(s) - h(X(s), u^*) \geq -f(X(s), u) \cdot \lambda(s) - h(X(s), u), \quad (7) \]

where \( u \) is any other control from the optimal control \( u^* \). Our \( \lambda \) also fulfills

\[ \lambda(s) = v_x(s, X(s)) \]
when \( v \) is differentiable at \((s, X(s))\). If the Hamiltonian is differentiable and if \( v_x \) is smooth then the pair \( \lambda(s) \) and \( X(s) \) will be a solution to the bi-characteristic Hamiltonian system

\[
\begin{align*}
X'(s) &= H_\lambda(\lambda(s), X(s)), \quad X(0) = X_0, \quad 0 < s < T, \\
-\lambda'(s) &= H_x(\lambda(s), X(s)), \quad \lambda(T) = g_x(X(T)), \quad 0 < s < T.
\end{align*}
\]

(8)

The gradients are denoted in the same way as before, so \( H_x \) and \( H_\lambda \) means the gradients with respect to \( x, \lambda \). For a more rigorous derivation see Theorem 7.4.17 and its Corollary 7.4.18 with proofs in \[4\].

This Hamiltonian system can be discretised to give the following equations

\[
\begin{align*}
\bar{X}_{n+1} - \bar{X}_n - \Delta t H_\lambda(\bar{\lambda}_{n+1}, \bar{X}_n) &= 0, \quad \bar{X}_0 = X_0, \\
\bar{\lambda}_n - \bar{\lambda}_{n+1} - \Delta t H_x(\bar{\lambda}_{n+1}, \bar{X}_n) &= 0, \quad \bar{\lambda}_N = g_x(\bar{X}_N),
\end{align*}
\]

(9)

which is called a symplectic Euler scheme.

2.3 The Riccati equation

If a system is linear and has a quadratic cost, the optimal control can be found using the Riccati equations. The system can then be described in a compact form with matrices,

\[
\dot{x}(t) = Ax(t) + Bu(t).
\]

(10)

The cost functional can be formed using matrices \( R, Q \) and \( S \), where \( R \) is a symmetric \( m \times m \) matrix, \( Q \) is a symmetric \( n \times n \) matrix and \( S \) is a constant \( n \times n \) matrix,

\[
x(T)'Sx(T) + \int_0^T (x(t)'Q(t)x(t) + u(t)'R(t)u(t))dt.
\]

(11)

This translates the HJB equation into an equation which can be solved more easily than the non-linear one discussed in the previous section.

\[
0 = \min_{u \in B} \left[ x'Qx + u'Ru + v_t(t, x) + v_x(t, x)'(Ax + Bu) \right], \quad v(T, x) = x'Sx.
\]

(12)
An ansatz for the value function is done with the introduction of the function $P$ which is an $n \times n$ symmetric matrix of functions,

$$v(t, x) = x' P(t) x.$$  \hfill(13)

This function, together with the terminal-value condition, $v(T, x)$, gives us the Riccati differential equations for $P$ after some calculations, see Example 3.2.2 from [1] or chapter 8.2, Linear Systems with Quadratic Control, in [5],

$$\dot{P}(t) = -P(t)A + A'P(t) - P(t)BR^{-1}(t)B'P(t) - Q,$$

$$P(T) = S. \hfill(14)$$

The optimal control is found by using the solution to the Riccati equations,

$$u^*(t) = -R^{-1}(t)B'P(t)x(t). \hfill(15)$$

\section*{2.4 Newton’s method}

The scalar version of Newton’s method is a method for solving systems on the form $f(y) = 0$. With an initial guess of $y$, the equation is solved using iterations $y_{n+1} = y_n + \delta y_n$ where $\delta y_n = -f(y_n)/f'(y_n)$, updating the derivative for every new position $y$. This scalar version can be generalised by exchanging $y$ with a system of variables, $y$, and the function $f$ with a vector-valued function $F$. Here $f'(y_n)$ will then be replaced by a Jacobian, defined as $J_{ij} = \partial F_i/\partial y_j$, turning the equation for updating the state into $dy_i = -J_{yi}^{-1}F(y_i)$. The equation can be solved by iterating

$$y_{i+1} = y_i - J_{yi}^{-1}F(y_i), \hfill(16)$$

for $i = 1, 2 \ldots M$. The asymptotic order of convergence of Newton’s method is quadratic, although it can be difficult finding an initial guess in the region of convergence.

Other methods that rely on approximation of derivatives to produce a Jacobian are called quasi-newton methods and are typically of a lower order of convergence. An example of these methods is the one-dimensional secant method with an order of convergence around 1.618, recognisable as the golden ratio. MATLABs integrated solver \texttt{fsolve} is an example of a method creating a Jacobian using numerical approximations of the derivatives.
2.5 Model of the inverted pendulum

To derive the model of the pendulum, the system is first observed as a conservative system. External and frictional forces are neglected. The quantities are regarded as dimensionless. This may be somewhat unphysical, but in this report this general equation serves our purpose.

Viewing the pendulum as a conservative system allows us to use the Euler-Lagrange equations in our derivation of the equations of motion,

\[
\frac{d}{dt} \left( \frac{\partial L}{\partial \dot{q}_j} \right) = \frac{\partial L}{\partial q_j},
\]

\[
L = E_k - E_p.
\]

Here \( q \) is a generalised coordinate, namely the \( x \)-coordinate, or the angle \( \theta \) between the pendulum and base, and \( L \) is the Lagrangian defined as the difference between kinetic and potential energy, \( E_k \) and \( E_p \), of the system.

The model of the inverted pendulum is depicted in Figure 1. The pendulum is considered a point mass, \( m_2 \), at a distance \( l \) from the base, which has the mass \( m_1 \). Their velocities are \( v_1 \) for the base, and \( v_2 \) for the pendulum. Both the base and the pendulum have one degree of freedom each: translational and rotational respectively.

Forming the Lagrangian we get the expression

\[
L = \frac{1}{2} m_1 v_1^2 + \frac{1}{2} m_2 v_2^2 - m_1 gl \sin \theta,
\]
where the constant $g = 9.82$ is the gravitational constant. The velocities are derived from the change of position, $v = \sqrt{\dot{x}^2 + \dot{y}^2}$,

\[
v_2^2 = \dot{x}^2,
\]
\[
v_1^2 = \left(\frac{d}{dt}(x - l \sin \theta)\right)^2 + \left(\frac{d}{dt}(l \cos \theta)\right)^2 = (\dot{x} - l \dot{\theta} \sin \theta)^2 + (l \dot{\theta} \cos \theta)^2 = \dot{x}^2 - 2l \dot{\theta} \dot{x} \sin \theta + l^2 \dot{\theta}^2 (\sin^2 \theta + \cos^2 \theta) = \dot{x}^2 - 2l \dot{\theta} \dot{x} \sin \theta + l^2 \dot{\theta}^2. \tag{19}\]

Inserting these expressions into the Lagrangian gives us

\[
L = \frac{1}{2} m_1 (\dot{x}^2 - 2l \dot{\theta} \dot{x} \sin \theta + l^2 \dot{\theta}^2) + \frac{1}{2} m_2 \dot{x}^2 - m_1 gl \sin \theta. \tag{20}\]

Now the Euler-Lagrange equations can be used to determine the equations of motion of the pendulum. The expressions

\[
\frac{d}{dt} \left( \frac{\partial L}{\partial \dot{\theta}} \right) = \frac{\partial L}{\partial \theta},
\]
\[
\frac{d}{dt} \left( \frac{\partial L}{\partial \dot{x}} \right) = \frac{\partial L}{\partial x}, \tag{21}\]

gives us the equations of motions after some simplifications. These are determined as being

\[
0 = g \cos \theta - \ddot{x} \sin \theta + l \ddot{\theta}, \tag{22}\]
\[
0 = (m_1 + m_2) \ddot{x} - m_1 l \ddot{\theta} \sin \theta + m_1 l \dot{\theta}^2 \cos \theta. \tag{23}\]

Setting all masses and lengths to one, $l = m_1 = m_2 = 1$, neglecting (23), and considering the acceleration of the base as a control, $\ddot{x} = u$, gives a controlled ODE in (22).
3 Method and implementation

An overview of how the optimal control problem is approached is found in this section. Both the design and the implementation of the algorithm described in the problem formulation are presented.

3.1 Construction of the non-linear model

The goal of the inverted pendulum is to reach the unstable equilibrium at $\theta = \pi / 2$, using as little force as possible. The model described in section 2.5 is used, and the equation (22) as the differential equation for $\theta$, with $\ddot{x} = u$.

The state space will be of dimension two since its differential equation is of order two and the control is scalar making its dimension one. This makes $d = 2$ and $m = 1$ in (2). A new variable is introduced to reduce the order of the differential equation, $\dot{\theta}(t) = \psi(t)$, making the two variables the angle from the vertical plane and the angular velocity. Counter-clockwise is defined as positive.

Our problem is then

$$X(s) = \begin{pmatrix} \theta(s) \\ \psi(s) \end{pmatrix}, \quad X'(s) = f(X(s), u(s)) = \begin{pmatrix} \psi \\ u \sin \theta - g \cos \theta \end{pmatrix}.$$

(24)

The two parts of the value function are both defined as quadratic costs,

$$h(X(s), u(s)) = \frac{u^2}{2},$$

$$g(X(T)) = K \left( \theta(T) - \frac{\pi}{2} \right)^2.$$

(25)

Here $K$ is a constant which can be adjusted so that the penalty of accelerating the base is in proportion to the penalty of not reaching the goal.

These two functions are motivated by our wish to minimise the force used on the base, force being proportional to acceleration. This choice also simplifies our comparison between the two methods, since the second one demands a quadratic control.

The Hamiltonian can then be calculated as

$$H(\lambda, X) = \min_{u \in B} [\lambda \cdot f + h] = \min_{u \in B} \left[ \lambda_1 \psi + \lambda_2 (u \sin \theta - g \cos \theta) + \frac{u^2}{2} \right].$$

(26)
To minimise this function with respect to $u$, the expression is differentiated and set equal to zero,

$$
\frac{\partial}{\partial u} \left( \lambda \cdot f + h \right) = \lambda_2 \sin \theta + u = 0,
$$

$$
u = -\lambda_2 \sin \theta.
$$

(27)

Plugging this back into the Hamiltonian, it can be written as

$$
H(\lambda, X) = \lambda_1 \psi - \frac{(\lambda_2 \sin \theta)^2}{2} - g \lambda_2 \cos \theta
$$

(28)

Now the gradients of the Hamiltonian can be calculated as

$$
H_\lambda = \begin{pmatrix}
\psi \\
-\lambda_2 \sin^2 \theta - g \cos \theta
\end{pmatrix},
$$

(29)

and

$$
H_X = \begin{pmatrix}
\lambda_2 \sin \theta(g - \lambda_2 \cos \theta) \\
\lambda_1
\end{pmatrix}.
$$

(30)

These will be used in the Hamiltonian system’s discrete counterpart, (9). Lastly the gradient of the function $g$ is calculated,

$$
g_x(X(t)) = \begin{pmatrix}
2K(\dot{\theta}(t) - \frac{\pi}{2}) \\
0
\end{pmatrix}.
$$

(31)

This gradient is needed to determine the end value of the dual function, $\lambda(T) = g_x(X(T))$.

### 3.2 Construction of the linear model

To simplify the construction of the linear model, a new angle is introduced, $\hat{\theta} = \theta + \pi/2$. Using this angle in the expression (24), the second term becomes $u \cos \hat{\theta} + g \sin \hat{\theta}$. With a Taylor expansion of the two terms around $\hat{\theta} = 0$ a new expression is found:

$$
f(X(s), u(s)) = \begin{pmatrix}
\psi \\
u + g \hat{\theta} + O(\hat{\theta}^2)
\end{pmatrix}.
$$

(32)
The higher terms of \(\dot{\theta}\) can be omitted since the expression is evaluated around \(\dot{\theta} = 0\), where they become very small. This equation can be expressed as a linear system on the form (10) with the matrices

\[
A = \begin{pmatrix} 0 & 1 \\ g & 0 \end{pmatrix}, \quad X(s) = \begin{pmatrix} \dot{\theta}(s) \\ \dot{\psi}(s) \end{pmatrix}, \quad B = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \quad u(s) = u(s). \tag{33}
\]

One can also define a cost functional in the new linearised model,

\[
g(x, t) = K\dot{\theta}(T)^2 \\
h(x, t) = \int_0^T \frac{u^2}{2} ds \tag{34}
\]

which, using the notation found in (11), gives the two following matrices

\[
R = \begin{pmatrix} 1 \\ 2 \end{pmatrix}, \quad S = \begin{pmatrix} K & 0 \\ 0 & 0 \end{pmatrix}. \tag{35}
\]

### 3.3 Implementations

A discretisation of the state trajectory is made. We introduce \(N\) discretisation points for the numerical approximations over a time \([0, T]\).

**Method using symplectic Euler**

To investigate how to solve (9) with the Hamiltonian (28), an algorithm described in [3] will be implemented in MATLAB. The solution to \(F(Y) = 0\) is wanted, and to determine the solution Newton’s method is implemented, \(Y_{i+1} = Y_i - J^{-1} F(Y_i)\). A discrete version of the Hamiltonian boundary system called a symplectic Euler scheme is used as the function \(F(Y)\), see (9).

Together with the initial values and the end conditions on the dual vari-
The following function is acquired,

\[
\mathbf{F} = \begin{pmatrix}
\theta_0 - y_1 \\
\psi_0 - y_2 \\
\bar{X}_2 - \bar{X}_1 - \Delta t H_x(\bar{\lambda}_2, \bar{X}_1) \\
\bar{\lambda}_1 - \bar{\lambda}_2 - \Delta t H_x(\bar{\lambda}_2, \bar{X}_1) \\
\vdots \\
\bar{X}_N - \bar{X}_{N-1} - \Delta t H_x(\bar{\lambda}_N, \bar{X}_{N-1}) \\
\bar{\lambda}_{N-1} - \bar{\lambda}_N - \Delta t H_x(\bar{\lambda}_N, \bar{X}_{N-1}) \\
\lambda_{N,1} - g_\theta(\theta_N, \psi_N) \\
\lambda_{N,2} - g_\psi(\theta_N, \psi_N)
\end{pmatrix}.
\] (36)

The solution vector \( \mathbf{Y} \) is ordered in the following way,

\[
\mathbf{Y} = \begin{pmatrix}
y_1 \\
y_2 \\
y_3 \\
\vdots \\
y_{4N-1} \\
y_{4N}
\end{pmatrix} = \begin{pmatrix}
x_0 \\
\lambda_0 \\
x_1 \\
\vdots \\
x_N \\
\lambda_N
\end{pmatrix} = \begin{pmatrix}
\theta_0 \\
\psi_0 \\
\lambda_{1,0} \\
\lambda_{2,0} \\
\vdots \\
\theta_N \\
\psi_N \\
\lambda_{1,N} \\
\lambda_{2,N}
\end{pmatrix}.
\] (37)

When the function \( \mathbf{F} \) is determined, the Jacobian can be calculated. As mentioned before, there are ways of approximating the Jacobian with quasi-newton methods, but we opt to calculate it analytically to achieve a higher order of convergence.

The Jacobian is determined by taking partial derivatives with respect to the discretisation variables. Each row is a gradient of a function \( F_1, F_2, \ldots, F_{4N} \).
with respect to the variables \( y_1, y_2, \ldots, y_{4N} \).

\[
J = \begin{pmatrix}
\frac{\partial F_1}{\partial y_1} & \frac{\partial F_1}{\partial y_2} & \cdots & \frac{\partial F_1}{\partial y_{4N}} \\
\frac{\partial F_2}{\partial y_1} & \ddots & & \\
\vdots & & \ddots & \\
\frac{\partial F_{4N}}{\partial y_1} & & & \frac{\partial F_{4N}}{\partial y_{4N}}
\end{pmatrix} = \begin{pmatrix}
\frac{\partial F_1}{\partial Y} \\
\vdots \\
\frac{\partial F_{4N}}{\partial Y}
\end{pmatrix}
\]

(38)

The Jacobians will be sparse and consequently MATLAB:s function \texttt{sparse} is used to make calculations faster.

The initial guesses of \( Y \) used are linear progression from the initial value of the pendulum to the goal of the state trajectory \( \theta = \pi/2 \). From this initial guess of the angle, the angular velocities are calculated numerically. The dual variables \( \lambda_1 \) and \( \lambda_2 \) are calculated via forward Euler from the end value and the differential equation given in (9).

**Method using the Riccati equation**

To determine the optimal control of the linearised model, (32), the matrix Riccati differential equation, (14), is used. From the time evolution of the matrix \( P \) the optimal control is found.

Determining the elements of \( P \) for every time \( s \) allows us to calculate the optimal control using (15). \( P \) has four elements, but only three need to be solved because it is symmetric. A simple forward Euler method solves this,

\[
P_{i+1} = P_i + \Delta t \dot{P}_i, \quad i = 1, 2, \ldots, N.
\]

(39)

Once \( P \) is known, the state trajectory can be found using once again forward Euler, but on the differential equation

\[
X'(s) = (A - BR^{-1}B'P(s))X(s).
\]

(40)

The optimal control \( u^* \) is found using (15).
4 Discussion and results

Results

The two programs were implemented successfully. Some solutions for the two methods are presented in Figure 6 and 7. Some comparisons between them can be found in Figures 2, 3, and 4.

Figure 2: Comparison of the solutions of the two methods for a small starting deviation from the target angle.

The program for solving the linear model behaved as expected. Its solutions’ deviation from the other method’s solutions turned out to be surprisingly small, and became evident only for rather big angles, as seen in the three figures with comparisons. Only the last one clearly differs. When the starting angle exceeded a certain deviation from the equilibrium, slightly short of $\theta = \pi/2$, the linearised model and its solutions became unrealistic.

The second program also behaved as expected. Given realistic initial guesses, the solutions converged, although the program did have problems achieving the expected quadratic convergence associated with Newton’s method. This is probably due to some error among the analytically derived functions generating the elements of the Jacobian.

The importance of the initial guesses was known from theory, and quickly
Figure 3: Comparison of the solutions of the two methods for a medium starting deviation from the target angle.

Figure 4: Comparison of the solutions of the two methods for a large starting deviation from the target angle.
became central when analysing the program’s solutions. The standard initial guess used, a linear progression from the starting angle to the equilibrium, gave convergent results for angles close to the equilibrium, but gave divergent results for all starting angles \( 0 \geq \theta \geq -\pi \). For these starting angles the initial guess needs more finess.

One example studied was the starting angle \( \theta = 0 \). With the standard initial guess the results diverged. This divergence seems intuitive: accelerating the base can’t make the pendulum swing up if it is already parallel to the movement of the base. The region of convergence for the method could not be reached. When the initial guess was changed to a linear guess from \( \theta = 0 \) to \( \theta = -3\pi/2 \), which is equivalent to \( \theta = \pi/2 \), a solution could be found, see Figure 5.

![Figure 5: Reaching the equilibrium through the means of a different initial guess for the non-linear model.](image)

Using this solution as an initial guess for angles close to \( \theta = 0 \) gave convergent solutions. Although not rigorously tested, this technique of diluting old solutions seems a good way of extending our region of solvable starting angles.

Worth mentioning is that when using this technique for starting angles slightly bigger than \( \theta = 0 \), a different solution to the old one was found. It becomes obvious that the old solution is but a local minimum. This is an apparent weakness of our method: we cannot know if our solutions are global or local.
Figure 6: Solutions for the linearised method with several starting values.

Figure 7: Solutions for the non-linearised method with several starting values.
Discussion

This short investigation of optimal control of an inverted pendulum is in many ways incomplete. Many different aspects have been left unexplored and the two implementations presented remain a drop in the ocean.

Many extensions to this project can be imagined. Firstly the simple model of the pendulum could be reworked to a more refined one, or switched to that of another type of pendulum, such as a double pendulum. Secondly different kinds of Hamiltonians could be investigated. One could for instance introduce functions penalising the translational and angular velocity of the base and the pendulum, as well as a terminal demand on the angular velocity.

On a final note one could also question the usefulness of this approach to optimal control and its relatability to real-life implementations. Since we assume perfect state knowledge, no friction, as well as zero disturbances to the control and the system, the model is obviously idealised. One could argue that optimising a feedback law might have been more useful in this sense.
Bibliography


