The Nonlinear Systems Toolbox is a package of MATLAB .m files for the analysis and synthesis of nonlinear control systems described by polynomials.
Introduction

The Nonlinear Systems Toolbox is a package of MATLAB .m files for the analysis and synthesis of nonlinear control systems described by polynomials.

If a system is not polynomial but it is described by elementary functions, there is a routine, tay_poly.m, that computes the Taylor polynomials at an equilibrium point. Then the analysis and synthesis are done on the polynomial approximation and the results are applied to the nonlinear system around the equilibrium point.
The Nonlinear Systems Toolbox is a package of MATLAB .m files for the analysis and synthesis of nonlinear control systems described by polynomials.

If a system is not polynomial but it is described by elementary functions, there is a routine, tay_poly.m, that computes the Taylor polynomials at an equilibrium point. Then the analysis and synthesis are done on the polynomial approximation and the results are applied to the nonlinear system around the equilibrium point.

This approach is an extension of the standard engineering technique of replacing a nonlinear system by its Jacobian linearization at an equilibrium point.
The Nonlinear Systems Toolbox is a package of MATLAB .m files for the analysis and synthesis of nonlinear control systems described by polynomials.

If a system is not polynomial but it is described by elementary functions, there is a routine, tay_poly.m, that computes the Taylor polynomials at an equilibrium point. Then the analysis and synthesis are done on the polynomial approximation and the results are applied to the nonlinear system around the equilibrium point.

This approach is an extension of the standard engineering technique of replacing a nonlinear system by its Jacobian linearization at an equilibrium point.
Stabilization to an Equilibrium

System

\[ \dot{x} = f(x, u) \]
\[ x \in \mathbb{R}^{n \times 1}, \quad u \in \mathbb{R}^{m \times 1} \]
Stabilization to an Equilibrium

System

\[ \dot{x} = f(x, u) \]
\[ x \in \mathbb{R}^{n\times1}, \quad u \in \mathbb{R}^{m\times1} \]

Equilibrium Point

\[ 0 = f(x^e, u^e) \]

WLOG: \( x^e = 0, \ u^e = 0 \)
Stabilization to an Equilibrium

System

\[ \dot{x} = f(x, u) \]
\[ x \in \mathbb{R}^{n \times 1}, \quad u \in \mathbb{R}^{m \times 1} \]

Equilibrium Point

\[ 0 = f(x^e, u^e) \]

WLOG: \( x^e = 0, \ u^e = 0 \)

Fundamental Problem: Find a state feedback law \( u = \kappa(x) \) that make this equilibrium asymptotically stable at least locally.
Standard Approach

Find the Linear Approximating System at the Equilibrium

\[ \dot{x} = Fx + Gu \]
Standard Approach

Find the Linear Approximating System at the Equilibrium

$$\dot{x} = Fx + Gu$$

where

$$F = \frac{\partial f}{\partial x}(0, 0) \quad G = \frac{\partial f}{\partial u}(0, 0)$$
Standard Approach

Find the Linear Approximating System at the Equilibrium

\[ \dot{x} = Fx + Gu \]

where

\[ F = \frac{\partial f}{\partial x}(0, 0) \quad G = \frac{\partial f}{\partial u}(0, 0) \]

Then find a linear feedback \( u = Kx \) that puts the eigenvalues of \( F + GK \) in the open left half plane.
Standard Approach

Find the Linear Approximating System at the Equilibrium

\[ \dot{x} = Fx + Gu \]

where

\[ F = \frac{\partial f}{\partial x}(0, 0) \quad G = \frac{\partial f}{\partial u}(0, 0) \]

Then find a linear feedback \( u = Kx \) that puts the eigenvalues of \( F + GK \) in the open left half plane.

If \( F, G \) is a stabilizable pair then this can be done by pole placement (e.g. Ackermann’s Formula) or by optimization techniques (LQR).
Standard Approach

Find the Linear Approximating System at the Equilibrium

\[ \dot{x} = Fx + Gu \]

where

\[ F = \left. \frac{\partial f}{\partial x} \right|_{(0,0)} \quad G = \left. \frac{\partial f}{\partial u} \right|_{(0,0)} \]

Then find a linear feedback \( u = Kx \) that puts the eigenvalues of \( F + GK \) in the open left half plane.

If \( F, G \) is a stabilizable pair then this can be done by pole placement (e.g. Ackermann’s Formula) or by optimization techniques (LQR).

The latter is preferable because the optimal cost is a candidate Lyapunov function for the closed loop nonlinear system.
Linear Quadratic Regulation (LQR)

Minimize

\[ \frac{1}{2} \int_{0}^{\infty} x' Q x + u' R u \, dt \]

subject to

\[ \dot{x} = F x + G u \]

\[ x(0) = x^0 \]
Linear Quadratic Regulation (LQR)

Minimize

$$\frac{1}{2} \int_0^\infty x' Q x + u' R u \, dt$$

subject to

$$\dot{x} = Fx + Gu$$

$$x(0) = x^0$$

The standard assumptions are

- $Q \geq 0$
Linear Quadratic Regulation (LQR)

Minimize

\[ \frac{1}{2} \int_0^\infty x'Qx + u'Ru \, dt \]

subject to

\[ \dot{x} = Fx + Gu \]

\[ x(0) = x^0 \]

The standard assumptions are

- \( Q \geq 0 \)
- \( R > 0 \)
Linear Quadratic Regulation (LQR)

Minimize

\[ \frac{1}{2} \int_{0}^{\infty} x'Qx + u'Ru \, dt \]

subject to

\[ \dot{x} = Fx + Gu \]
\[ x(0) = x^0 \]

The standard assumptions are

- \( Q \geq 0 \)
- \( R > 0 \)
- \( F, G \) are a stabilizable pair
Linear Quadratic Regulation (LQR)

Minimize

\[ \frac{1}{2} \int_{0}^{\infty} x' Q x + u' R u \, dt \]

subject to

\[ \dot{x} = Fx + Gu \]
\[ x(0) = x^0 \]

The standard assumptions are

- \( Q \geq 0 \)
- \( R > 0 \)
- \( F, G \) are a stabilizable pair
- \( Q^{1/2}, F \) are a detectable pair
Under the standard assumptions there exists a unique $P \geq 0$ that satisfies the algebraic Riccati equation

\[
0 = F'P + PF + Q - PGR^{-1}G'P \\
K = -R^{-1}G'P
\]

and the optimal cost of starting at $x$ is $\pi(x) = \frac{1}{2}x'Px$.  

Under the standard assumptions there exists a unique $P \geq 0$ that satisfies the algebraic Riccati equation

\[ 0 = F'P + PF + Q - PGR^{-1}G'P \]
\[ K = -R^{-1}G'P \]

and the optimal cost of starting at $x$ is $\pi(x) = \frac{1}{2}x'Px$.

The optimal feedback is $u = \kappa(x) = Kx$. 
Under the standard assumptions there exists a unique $P \geq 0$ that satisfies the algebraic Riccati equation

$$0 = F'P + PF + Q - PGR^{-1}G'P$$

$$K = -R^{-1}G'P$$

and the optimal cost of starting at $x$ is $\pi(x) = \frac{1}{2}x'Px$.

The optimal feedback is $u = \kappa(x) = Kx$.

The eigenvalues of the closed loop matrix $F + GK$ are all in the open left half plane.
Al’brekht’s Method generalizes the standard LQR approach by computing higher order Taylor polynomial approximations to the optimal cost $\pi(x)$ and optimal feedback $\kappa(x)$ for the nonlinear optimal control problem of minimizing

$$\int_{0}^{\infty} l(x, u) \, dt$$

subject to

$$\dot{x} = f(x, u)$$
$$x(0) = x^0$$
Hamilton-Jacobi-Bellman Equations

If $\pi(x)$ and $\kappa(x)$ are smooth then they satisfy the HJB PDEs

$$0 = \mathcal{H} \left( \frac{\partial \pi}{\partial x} (x), x, \kappa(x) \right)$$

$$\kappa(x) = \arg\min_u \mathcal{H} \left( \frac{\partial \pi}{\partial x} (x), x, u \right)$$

where the Hamiltonian for $p \in \mathbb{R}^{1 \times n}$ is

$$\mathcal{H} (p, x, u) = pf(x, u) + l(x, u)$$
Hamilton-Jacobi-Bellman Equations

If \( \pi(x) \) and \( \kappa(x) \) are smooth then they satisfy the HJB PDEs

\[
0 = \mathcal{H}
\left(
\frac{\partial \pi}{\partial x}(x), x, \kappa(x)
\right)
\]

\[
\kappa(x) = \arg\min_u \mathcal{H}
\left(
\frac{\partial \pi}{\partial x}(x), x, u
\right)
\]

where the Hamiltonian for \( p \in \mathbb{R}^{1 \times n} \) is

\[
\mathcal{H}(p, x, u) = p f(x, u) + l(x, u)
\]

If \( \mathcal{H} \) is strictly convex in \( u \) for every \( p, x \) then HJB equations can be rewritten as

\[
0 = \frac{\partial \pi}{\partial x}(x) f(x, \kappa(x)) + l(x, \kappa(x))
\]

\[
0 = \frac{\partial \pi}{\partial x}(x) \frac{\partial f}{\partial u}(x, \kappa(x)) + \frac{\partial l}{\partial u}(x, \kappa(x))
\]
Al’brekht’s Method

Al’brecht developed the power series method for solving the HJB equations for smooth systems that have Taylor series expansions.

\[ f(x, u) = Fx + Gu + f^{[2]}(x, u) + f^{[3]}(x, u) + \ldots \]

\[ l(x, u) = \frac{1}{2} (x'QX + u'Ru) + l^{[3]}(x, u) + l^{[4]}(x, u) + \ldots \]
Al’brekht’s Method

Al’brecht developed the power series method for solving the HJB equations for smooth systems that have Taylor series expansions.

\[ f(x, u) = Fx + Gu + f^2(x, u) + f^3(x, u) + \ldots \]
\[ l(x, u) = \frac{1}{2} (x'QX + u'Ru) + l^3(x, u) + l^4(x, u) + \ldots \]

He assumed that the optimal cost and optimal feedback had similar expansions

\[ \pi(x) = \frac{1}{2} x'Px + \pi^3(x) + \pi^4(x) + \ldots \]
\[ \kappa(x) = Kx + \kappa^2(x) + \kappa^3(x) + \ldots \]
Al’brekht’s Method

Al’brecht developed the power series method for solving the HJB equations for smooth systems that have Taylor series expansions.

\[ f(x, u) = Fx + Gu + f^{[2]}(x, u) + f^{[3]}(x, u) + \ldots \]
\[ l(x, u) = \frac{1}{2} (x'QX + u'Ru) + l^{[3]}(x, u) + l^{[4]}(x, u) + \ldots \]

He assumed that the optimal cost and optimal feedback had similar expansions

\[ \pi(x) = \frac{1}{2} x'Px + \pi^{[3]}(x) + \pi^{[4]}(x) + \ldots \]
\[ \kappa(x) = Kx + \kappa^{[2]}(x) + \kappa^{[3]}(x) + \ldots \]

He plugged these expansions into HJB. At the lowest degrees he got the familiar LQR equations

\[ 0 = F'P + PF + Q - PGR^{-1}G'P \]
\[ K = -R^{-1}G'P \]
Al’brekht’s Method

Next the unknown degree three terms $\pi^{[3]}(x)$ of the cost and the unknown degree two terms $\kappa^{[2]}(x)$ of the feedback satisfy

$$0 = \frac{\partial \pi^{[3]}}{\partial x}(x)(F + GK)x + x'Pf^{[2]}(x, Kx) + l^{[3]}(x, Kx)$$

$$0 = \frac{\partial \pi^{[3]}}{\partial x}(x)G + x'P \frac{\partial f^{[2]}}{\partial u}(x, Kx) + \frac{\partial \pi^{[3]}}{\partial u}(x, Kx) + (\kappa^{[2]}(x)'R$$
Al’brekht’s Method

Next the unknown degree three terms $\pi^3(x)$ of the cost and the unknown degree two terms $\kappa^2(x)$ of the feedback satisfy

$$0 = \frac{\partial \pi^3}{\partial x}(x)(F + GK)x + x'Pf^2(x, Kx) + l^3(x, Kx)$$

$$0 = \frac{\partial \pi^3}{\partial x}(x)G + x'P \frac{\partial f^2}{\partial u}(x, Kx) + \frac{\partial^3}{\partial u}(x, Kx) + (\kappa^2(x)'R$$

Notice the linear triangular structure. Under the standard LQR assumptions the first linear equation is always solvable for $\pi^3(x)$ because the eigenvalues of the map

$$\pi^3(x) \mapsto \frac{\partial \pi^3}{\partial x}(x)(F + GK)x$$

are sums of three eigenvalues of $F + GK$, $\sigma(F + GK) < 0$. 
Al’brekht’s Method

Next the unknown degree three terms $\pi^3(x)$ of the cost and the unknown degree two terms $\kappa^2(x)$ of the feedback satisfy

$$0 = \frac{\partial \pi^3}{\partial x}(x)(F + GK)x + x'Pf^2(x, Kx) + l^3(x, Kx)$$

$$0 = \frac{\partial \pi^3}{\partial x}(x)G + x'P\frac{\partial f^2}{\partial u}(x, Kx) + \frac{\partial^3}{\partial u}(x, Kx) + (\kappa^2(x)'R$$

Notice the linear triangular structure. Under the standard LQR assumptions the first linear equation is always solvable for $\pi^3(x)$ because the eigenvalues of the map $\pi^3(x) \mapsto \frac{\partial \pi^3}{\partial x}(x)(F + GK)x$ are sums of three eigenvalues of $F + GK$, $\sigma(F + GK) < 0$.

Then the second linear equation is always solvable for $\kappa^2(x)$ because $R$ is assumed to be invertible.
Al’brekht’s Method

The higher degree terms are found in a similar fashion.
Al’brekht’s Method

The higher degree terms are found in a similar fashion.

This method has been implemented in the MATLAB based Nonlinear Systems Toolbox to arbitrary degree and dimensions.
Al’brekht’s Method

The higher degree terms are found in a similar fashion.

This method has been implemented in the MATLAB based Nonlinear Systems Toolbox to arbitrary degree and dimensions.

Al’brekht’s method works in reasonable dimensions. For example, the HJB equations can be solved to degree $4$ in $\pi(x)$ and degree $3$ in $\kappa(x)$ for systems with state dimension $n = 25$ and control dimension $m = 8$ on this four year old laptop.
Al’brekht’s Method

The higher degree terms are found in a similar fashion.

This method has been implemented in the MATLAB based Nonlinear Systems Toolbox to arbitrary degree and dimensions.

Al’brekht’s method works in reasonable dimensions. For example, the HJB equations can be solved to degree $4$ in $\pi(x)$ and degree $3$ in $\kappa(x)$ for systems with state dimension $n = 25$ and control dimension $m = 8$ on this four year old laptop.

Al’brekht’s method is fast. This laptop took $0.082334$ seconds to solve the HJB equations for the satellite attitude problem, $(n = 6, \ m = 3)$, to degree $4$ in $\pi(x)$ and degree $3$ in $\kappa(x)$. 
Pros and Cons of Al’brekht’s Method

- Al’brekht’s Method can be extended to discrete time and time varying problems.
Pros and Cons of Al’brekht’s Method

- Al’brekht’s Method can be extended to discrete time and time varying problems.
- Except for the Riccati equation it only involves solving linear equations and Matlab software is available.
Pros and Cons of Al’brekht’s Method

- Al’brekht’s Method can be extended to discrete time and time varying problems.
- Except for the Riccati equation it only involves solving linear equations and Matlab software is available.
- Al’brekht’s Method is restricted to smooth systems with no state or control constraints.

The LQR part must yield a Hurwitz $F + GK$.

The software is fast and it can used for systems of moderately large state dimension, e.g., $n=25$, $m=8$.

Going to higher degree approximations to $\pi(x)$ and $\kappa(x)$ increases their accuracy near $x=0$.

Going to higher degree approximations can enlarge the basin of stability of the closed loop system but it is not guaranteed to do so. It can also decrease it.

Going to higher degree approximations requires more memory. There are $n+d-1 \choose d$ monomials of degree $d$ in $n$ variables, approximately $n^d/d!$. 
Pros and Cons of Al’brekht’s Method

- Al’brekht’s Method can be extended to discrete time and time varying problems.
- Except for the Riccati equation it only involves solving linear equations and Matlab software is available.
- Al’brekht’s Method is restricted to smooth systems with no state or control constraints.
- The LQR part must yield a Hurwitz $F + GK$.

Going to higher degree approximations to $\pi(x)$ and $\kappa(x)$ increases their accuracy near $x = 0$. Going to higher degree approximations can enlarge the basin of stability of the closed loop system but it is not guaranteed to do so. It can also decrease it. Going to higher degree approximations requires more memory. There are $n + d - 1$ choose $d$ monomials of degree $d$ in $n$ variables, approximately $n^d/d!$. 
Pros and Cons of Al’brekht’s Method

- Al’brekht’s Method can be extended to discrete time and time varying problems.
- Except for the Riccati equation it only involves solving linear equations and Matlab software is available.
- Al’brekht’s Method is restricted to smooth systems with no state or control constraints.
- The LQR part must yield a Hurwitz $F + GK$.
- The software is fast and it can be used for systems of moderately large state dimension, e.g., $n = 25$, $m = 8$. 

Going to higher degree approximations to $\pi(x)$ and $\kappa(x)$ increases their accuracy near $x = 0$.

Going to higher degree approximations can enlarge the basin of stability of the closed loop system but it is not guaranteed to do so. It can also decrease it.

Going to higher degree approximations requires more memory. There are $n + d - 1$ choose $d$ monomials of degree $d$ in $n$ variables, approximately $n^d / d!$. 
Pros and Cons of Al’brekht’s Method

- Al’brekht’s Method can be extended to discrete time and time varying problems.
- Except for the Riccati equation it only involves solving linear equations and Matlab software is available.
- Al’brekht’s Method is restricted to smooth systems with no state or control constraints.
- The LQR part must yield a Hurwitz $F + GK$.
- The software is fast and it can used for systems of moderately large state dimension, e.g., $n = 25, m = 8$.
- Going to higher degree approximations to $\pi(x)$ and $\kappa(x)$ increases their accuracy near $x = 0$. 
Pros and Cons of Al’brekht’s Method

- Al’brekht’s Method can be extended to discrete time and time varying problems.
- Except for the Riccati equation it only involves solving linear equations and Matlab software is available.
- Al’brekht’s Method is restricted to smooth systems with no state or control constraints.
- The LQR part must yield a Hurwitz $F + GK$.
- The software is fast and it can used for systems of moderately large state dimension, e.g., $n = 25$, $m = 8$.
- Going to higher degree approximations to $\pi(x)$ and $\kappa(x)$ increases their accuracy near $x = 0$.
- Going to higher degree approximations can enlarge the basin of stability of the closed loop system but it is not guaranteed to do so. It can also decrease it.
Pros and Cons of Al’brekht’s Method

- Al’brekht’s Method can be extended to discrete time and time varying problems.
- Except for the Riccati equation it only involves solving linear equations and Matlab software is available.
- Al’brekht’s Method is restricted to smooth systems with no state or control constraints.
- The LQR part must yield a Hurwitz $F + GK$.
- The software is fast and it can used for systems of moderately large state dimension, e.g., $n = 25$, $m = 8$.
- Going to higher degree approximations to $\pi(x)$ and $\kappa(x)$ increases their accuracy near $x = 0$.
- Going to higher degree approximations can enlarge the basin of stability of the closed loop system but it is not guaranteed to do so. It can also decrease it.
- Going to higher degree approximations requires more memory. There are $n + d - 1$ choose $d$ monomials of degree $d$ in $n$ variables, approximately $n^d/d!$.
Pros and Cons of Al’brekht’s Method

- Besides supplying a stabilizing feedback, Al’brekht’s Method supplies a candidate Lyapunov function so the basin of attraction can be estimated. (This can take much longer than Al’brekht’s method itself.)

- Because the feedback is a higher degree polynomial it can easily lead to finite escape time in the model and catastrophe in the actual plant.

- Because the feedback is a higher degree polynomial it can easily violate state and/or control constraints.

- On the other hand higher degree penalty terms can be added to the Lagrangian to enforce such constraints.

- Even though it is higher degree, Al’brekht’s Method is a local method but patchy extensions are possible.

- Al’brekht’s Method can be used to speed up Model Predictive Control (MPC)!
Pros and Cons of Al’brekht’s Method

• Besides supplying a stabilizing feedback, Al’brekht’s Method supplies a candidate Lyapunov function so the basin of attraction can be estimated. (This can take much longer than Al’brekht’s method itself.)

• Because the feedback is a higher degree polynomial it can easily lead to finite escape time in the model and catastrophe in the actual plant.
Pros and Cons of Al’brekht’s Method

• Besides supplying a stabilizing feedback, Al’brekht’s Method supplies a candidate Lyapunov function so the basin of attraction can be estimated. (This can take much longer than Al’brekht’s method itself.)

• Because the feedback is a higher degree polynomial it can easily lead to finite escape time in the model and catastrophe in the actual plant.

• Because the feedback is a higher degree polynomial it can easily violate state and/or control constraints.
Pros and Cons of Al’brekht’s Method

• Besides supplying a stabilizing feedback, Al’brekht’s Method supplies a candidate Lyapunov function so the basin of attraction can be estimated. (This can take much longer than Al’brekht’s method itself.)

• Because the feedback is a higher degree polynomial it can easily lead to finite escape time in the model and catastrophe in the actual plant.

• Because the feedback is a higher degree polynomial it can easily violate state and/or control constraints.

• On the other hand higher degree penalty terms can be added to the Lagrangian to enforce such constraints.
Pros and Cons of Al’brekht’s Method

- Besides supplying a stabilizing feedback, Al’brekht’s Method supplies a candidate Lyapunov function so the basin of attraction can be estimated. (This can take much longer than Al’brekht’s method itself.)

- Because the feedback is a higher degree polynomial it can easily lead to finite escape time in the model and catastrophe in the actual plant.

- Because the feedback is a higher degree polynomial it can easily violate state and/or control constraints.

- On the other hand higher degree penalty terms can be added to the Lagrangian to enforce such constraints.

- Even though it is higher degree, Al’brekht’s Method is a local method but patchy extensions are possible.
Pros and Cons of Al’brekht’s Method

• Besides supplying a stabilizing feedback, Al’brekht’s Method supplies a candidate Lyapunov function so the basin of attraction can be estimated. (This can take much longer than Al’brekht’s method itself.)

• Because the feedback is a higher degree polynomial it can easily lead to finite escape time in the model and catastrophe in the actual plant.

• Because the feedback is a higher degree polynomial it can easily violate state and/or control constraints.

• On the other hand higher degree penalty terms can be added to the Lagrangian to enforce such constraints.

• Even though it is higher degree, Al’brekht’s Method is a local method but patchy extensions are possible.

• Al’brekht’s Method can be used to speed up Model Predictive Control (MPC)!
Suppose we have the constraint

\[ 0 \geq \beta(x, u) \]

which we assume is not active at the origin \( \beta(0, 0) < 0 \).
Suppose we have the constraint

\[ 0 \geq \beta(x, u) \]

which we assume is not active at the origin \( \beta(0, 0) < 0 \).

Frequently such constraints can be handle by adding penalty terms to the Lagrangian \( l(x, u) \).
Suppose we have the constraint

$$0 \geq \beta(x, u)$$

which we assume is not active at the origin $\beta(0, 0) < 0$.

Frequently such constraints can be handled by adding penalty terms to the Lagrangian $l(x, u)$.

Here are two simple examples.
Al’brekht with an Inequality State Constraint

Unstable linear system

\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2
\end{bmatrix}
= \begin{bmatrix}
0 & 1 \\
1 & 0
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix}
+ \begin{bmatrix}
0 \\
1
\end{bmatrix} u
\]
Al’brekht with an Inequality State Constraint

Unstable linear system

\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2
\end{bmatrix} =
\begin{bmatrix}
0 & 1 \\
1 & 0
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix} +
\begin{bmatrix}
0 \\
1
\end{bmatrix} u
\]

Lagrangian

\[
l(x, u) = \frac{1}{2} (x_1^2 + x_2^2 + u^2)
\]

State Constraint

\[x_1 \leq 0.5\]

Initial Condition

\[
\begin{bmatrix}
x_1(0) \\
x_2(0)
\end{bmatrix} =
\begin{bmatrix}
0.4 \\
0.7
\end{bmatrix}
\]
Al’brekht with an Inequality State Constraint

**Unstable linear system**

\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2
\end{bmatrix} =
\begin{bmatrix}
0 & 1 \\
1 & 0
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix} +
\begin{bmatrix}
0 \\
1
\end{bmatrix} u
\]

**Lagrangian**

\[l(x, u) = \frac{1}{2} (x_1^2 + x_2^2 + u^2)\]

**State Constraint**

\[x_1 \leq 0.5\]
Al’brekht with an Inequality State Constraint

Unstable linear system

\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2
\end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u
\]

Lagrangian

\[
l(x, u) = \frac{1}{2} (x_1^2 + x_2^2 + u^2)
\]

State Constraint

\[
x_1 \leq 0.5
\]

Initial Condition

\[
\begin{bmatrix} x_1(0) \\ x_2(0) \end{bmatrix} = \begin{bmatrix} 0.4 \\ 0.7 \end{bmatrix}
\]
Linear Feedback

\[
l(x, u) = \frac{1}{2} (x_1^2 + x_2^2 + u^2)
\]

\[
u = -2.4142x_1 - 2.4142x_2
\]
Al’brekht with a State Inequality Constraint

Linear Feedback

\[
l(x, u) = \frac{1}{2} \left( x_1^2 + x_2^2 + u^2 \right)
\]

\[
u = -2.4142x_1 - 2.4142x_2
\]
Al’brekht with a State Inequality Constraint

Quintic Feedback

\[ l(x, u) = \frac{1}{2} (x_1^2 + x_2^2 + u^2) + 32x_1^5 + 64x_1^6 \]

\[ u = -2.41x_1 - 2.41x_2 \\
-22.62x_1^4 - 28.49x_1^3x_2 - 15.36x_1^2x_2^2 - 4.00x_1x_2^3 - 0.41x_2^4 \\
-45.25x_1^5 - 67.01x_1^4x_2 - 45.60x_1^3x_2^2 - 16.93x_1^2x_2^3 - 3.34x_1x_2^4 - 0.27x_2^5 \]
Al’brekht with a State Inequality Constraint

Quintic Feedback

\[ l(x, u) = \frac{1}{2} (x_1^2 + x_2^2 + u^2) + 32x_1^5 + 64x_1^6 \]
\[ u = -2.41x_1 - 2.41x_2 - 22.62x_1^4 - 28.49x_1^3x_2 - 15.36x_1^2x_2^2 - 4.00x_1x_2^3 - 0.41x_2^4 - 45.25x_1^5 - 67.01x_1^4x_2 - 45.60x_1^3x_2^2 - 16.93x_1^2x_2^3 - 3.34x_1x_2^4 - 0.27x_2^5 \]
Al’brekht with a Control Inequality Constraint

\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2
\end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u
\]

\[
l(x, u) = \frac{1}{2} (x_1^2 + x_2^2 + u^2)
\]
Al’brekht with a Control Inequality Constraint

\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2
\end{bmatrix} = \begin{bmatrix}
0 & 1 \\
1 & 0
\end{bmatrix} \begin{bmatrix}
x_1 \\
x_2
\end{bmatrix} + \begin{bmatrix}
0 \\
1
\end{bmatrix} u
\]

\[l(x, u) = \frac{1}{2} (x_1^2 + x_2^2 + u^2)\]

Control Constraint

\[|u| \leq 1\]
Al’brekht with a Control Inequality Constraint

\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2
\end{bmatrix} =
\begin{bmatrix}
0 & 1 \\
1 & 0
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix} +
\begin{bmatrix}
0 \\
1
\end{bmatrix}
\]

\[
l(x, u) = \frac{1}{2} (x_1^2 + x_2^2 + u^2)
\]

Control Constraint

\[|u| \leq 1\]

Linear Feedback

\[u = -2.4142(x_1 + x_2)\]
Feasible Region of Linear Feedback

\[ u = -2.4142(x_1 + x_2) \]
Feasible Region of Cubic Feedback

\[ l(x, u) = \frac{1}{2} (x_1^2 + x_2^2 + u^2) + \frac{1}{10} u^4 \]
\[ u = -2.4142(x_1 + x_2) - 3.2263(x_1 + x_2)^3 \]
Double Pendulum Example

% Define system
n=4; % state dimension
m=1; % control dimension
d=3; % degree of optimal feedback
x=sym('x',[n,1]); % state variables
u=sym('u',[m,1]); % control variable

l1=1; % length of first massless link
l2=2; % length of second massless link
m1=2; % mass at end of first link
m2=1; % mass at end of second link
b1=0.5; % damping coefficient at first joint
b2=0.5; % damping coefficient at second joint
g=9.8; % gravitational constant

x0=[pi;pi;0;0]; % equilibrium state
u0=0; % equilibrium control
\% inertia matrix

\[
M = \begin{bmatrix}
m_1l_1^2 + m_2l_2^2, & m_2l_1l_2\cos(x(1,1)-x(2,1)) \\
m_2l_1l_2\cos(x(1,1)-x(2,1)), & m_2l_2^2
\end{bmatrix};
\]

\% Coriolis and centripetal matrix

\[
C = \text{jacobian}(\text{reshape}(M, 4, 1), x(1:2, 1));
\]

\[
C = \text{reshape}(C x(3:4, 1), 2, 2)/2;
\]

\% Kinetic energy

\[
T = x(3:4, 1)' \cdot M \cdot x(3:4, 1)/2;
\]

\% Potential energy

\[
V = g \cdot (m_1l_1(1-\cos(x(1,1))) + m_2(l_1(1-\cos(x(1,1)))+l_2(1-\cos(x(2,1))))
\]

\% Lagrangian

\[
L = T - V;
\]
% dynamics

f12=x(3:4,1);
f34=inv(M)*(jacobian(L,x(1:2,1))).'-C*x(3:4,1) +[u;0]-[b1*x(3,1);b2*(x(4,1)-x(3,1))]);
f=[f12;f34];

% control Lagrangian

l=((x(1,1)-x0(1,1))^2+(x(2,1)-x0(2,1))^2+u^2)
% call hjb_set_up.m to convert the
% symbolic f and l into the
% matrices ff and ll of coefficients
% of their Taylor polynomials at x0, u0.

[ff,ll]=hjb_set_up(f,l,x,u,x0,u0,n,m,d);

% This takes 2.0280 sec. when d=3.

% This takes 40.7541 sec. when d=5.
% call hjb.m to find the Taylor polynomial py
% of the optimal cost to degree d+1
% and the Taylor polynomial ka of the
% optimal feedback to degree d.

[ka,fk,py,lk]= hjb(ff,ll,n,m,d);

% This takes 0.0142 sec. when d=3.

% This takes 0.1211 sec. when d=5.
% verify that py and kappa satisfy the
% first HJB equation

HJB_residue=norm(dd(py,[1,n],[2,d+1],fk,[n,n])

% The HJB_residue is 9.5176e-07 when d=3.

% The HJB_residue is 3.5544e-04 when d=5.
Rigid Body with 6 DOF.

% The rigid body is an ellipsoid
% $$ x^2/9+y^2+z^2 \leq 1 $$
% of uniform density 1.

% The state and control dimensions are
n=12;
m=6;
% The degree of the approximation is
d=3;
% The state variables in order are
% the x, y, z coordinates of the center
% of mass relative to the inertial frame,
% the Euler angles phi, theta, psi
% of the body frame
% relative to the inertial frame,
% the u, v, w linear velocities
% relative to the body frame,
% the p, q, r angular velocities
% relative to the body frame.

x=sym('x',[n,1]);

% The control variables in order are
% the forces X,Y,Z
% relative to the body frame,
% the moments K, M, N
% relative to the body frame.

u=sym('u',[6,1]);
% The equilibrium point is
x0=zeros(n,1);
u0=zeros(m,1);

% rotation matrix between inertial
% and body frames
J1=[cos(x(6,1)),-sin(x(6,1)),0;....
    sin(x(6,1)),cos(x(6,1)),0;......
0,0,1];
J1=J1*[cos(x(5,1)),0,sin(x(5,1));....
0,1,0;......
-sin(x(5,1)),0,cos(x(5,1))];
J1=J1*[1,0,0;0, ....
cos(x(4,1)),-sin(x(4,1));......
0,sin(x(4,1)),cos(x(4,1))];
% matrix to convert Euler angular velocities
% to body angular velocities
J2=[1,sin(x(6,1))*tan(x(5,1)), cos(x(6,1))*tan(x(5,1));
   0,cos(x(6,1)),sin(x(6,1));
   0,sin(x(6,1))/cos(x(5,1)),cos(x(6,1))/cos(x(5,1));

f=[J1*x(7:9,1);J2*x(10:12)];
% inertia matrix

M_RB_inv=diag([1/(4*pi),1/(4*pi),1/(4*pi),5/(8*pi),1/(8*pi),1/(8*pi)]);

% coriolis and centripetal matrix
C_RB=[0,0,0,0,4*pi*x(9,1),-4*pi*x(8,1);....
    0,0,0,-4*pi*x(9,1),0,4*pi*x(7,1);....
    0,0,0,4*pi*x(8,1),-4*pi*x(7,1),0;....
    0,4*pi*x(9,1),-4*pi*x(8,1),0,8*pi*x(12,1),-4*pi*x(9,1),0,4*pi*x(7,1),-8*pi*x(12,1),
    4*pi*x(8,1),-4*pi*x(7,1),0,8*pi*x(11,1),8

    temp=-C_RB*x(7:12)+u;

% dynamics
f=[f;M_RB_inv*temp];
% control Lagrangian

l=x.'*eye(12)*x+u.'*eye(6)*u;
% call hjb_set_up.m to convert the symbolic f
% matrices ff and ll of coefficients of their
% at x0, u0.

 tic
 [ff,ll]=hjb_set_up(f,l,x,u,x0,u0,n,m,d);
 set_up_time=toc
call hjb.m to find the Taylor polynomial py
% to degree d+1 and the Taylor polynomial ka
% to degree d.

tic
[ka,fk,py,lk]= hjb(ff,ll,n,m,d);
computation_time =toc

% verify that py and kappa satisfy the first
HJB_Residual =norm(dd(py,[1,n],[2,d+1],fk,[n,
set_up_time =

1.6362e+03

computation_time =

8.0931

HJB_Residual =

9.2981e-12