

1 Abstract

2 Introduction

When studying systems and control theory we often focus on finding properties of a given mathematical model. However, in the real world the mathematical model of a system is not always available. In most cases we only have the measurements/data as produced by a system. To combat this, we can use tools from the field of data-driven analysis and control. These tools are able to infer system theoretical properties from the data without explicitly knowing the underlying mathematical model of the system. In this paper we will look at the implementation of the tools provided in [waarde2019data] and [waarde2020noisy].

3 Definitions

In this section we will introduce the notion of informativity. We will also look at the notation that will be used in the paper to indicate data and sets of systems.

3.1 Model classes

We will start by defining model classes. When considering linear systems in state space form we know that they come in different forms. One of the more general forms is the following.

$$\mathbf{x}(t+1) = A\mathbf{x}(t) + B\mathbf{u}(t) + \mathbf{w}(t) \quad (1)$$

$$\mathbf{y}(t+1) = C\mathbf{x}(t) + D\mathbf{u}(t) \quad (2)$$

Let us assume we have a system that has no input (u), output (y) or noise (w) variables and has a state space dimension of 3. Then the system $\Sigma(A, B, C, D)$ can be described using only an A matrix of size 3. Intuitively this is a very different type of system compared to a system that has a state space that is only 1 dimension and does contain an input, output and noise variable. Hence we define a model class in the following way.

Def: Model class

A system Σ_1 is said to be the same model class as Σ_2 if the dimension of the state space, input space and output space are equivalent between both systems.

If we apply this to the previous example we can see that the 2 systems are not of the same model class because their state spaces have different dimensions.

3.2 Informativity

Let Σ be the set of discrete time models of a given model class and let \mathcal{S} be a system from that model class. Then we know that \mathcal{S} is contained in Σ . Lets say we want to infer a property from our system \mathcal{S} . Then if we show that the property holds for all systems in Σ then it would also hold

for \mathcal{S} . However, due to the size of Σ this is not very feasible. Thus we need to reduce Σ to a more manageable set. To do this we will use the data generated by \mathcal{S} . We will call this data \mathcal{D} . We will define the set of all systems of the given model class that are able to generate the data \mathcal{D} by $\Sigma_{\mathcal{D}}$. By construction we know that $\mathcal{S} \in \Sigma_{\mathcal{D}} \subseteq \Sigma$.

Suppose we want to show if the true system is controllable, we might not be able to uniquely identify the true system using the data, i.e. $\#\Sigma_{\mathcal{D}} > 1$. However if we are able to show that every system in the set $\Sigma_{\mathcal{D}}$ is controllable, then we would also know that the true system is controllable. This is the idea behind data informativity. We say the data \mathcal{D} is informative for a property \mathcal{P} if all systems that describe the data $\Sigma_{\mathcal{D}}$ have the property \mathcal{P} . Let $\Sigma_{\mathcal{P}} \subseteq \Sigma$ be the set of all systems that have the property \mathcal{P} . Then we can reformulate the definition of data informativity as follows.

Def: Informativity [waarde2019data]

We say that the data \mathcal{D} is informative for a property \mathcal{P} if $\Sigma_{\mathcal{D}} \subseteq \Sigma_{\mathcal{P}}$.

Suppose that the data describes only the true system, i.e. $\Sigma_{\mathcal{D}} = \{\mathcal{S}\}$ then we know that if a property \mathcal{P} hold for \mathcal{S} then the data is informative for that property. However, in general, just because \mathcal{S} has a property \mathcal{P} does not immediately imply that the data is informative for \mathcal{P} since the data might describe more then one system. Later on in section (8.3) we will see this in an example where the data describes infinity many systems.

We will also look at control problems. Suppose we want to see if the property '*is stable in full state feedback with a controller K* ' holds on the data, then we need to know if all systems are stabilisable by state feedback using the controller K . For this we will define the set of systems that are stabilised using state feedback for the controller K as follows:

$$\Sigma_K = \{(A, B) \mid A + B K \text{ is stable}\}$$

Then we have that the data is informative if $\Sigma_{\mathcal{D}} \subseteq \Sigma_K$. We will generalise this using the following definition.

Def: Informativity for control [waarde2019data]

We say that the data \mathcal{D} is informative for a property $\mathcal{P}(\cdot)$ if there exists a controller K such that $\Sigma_{\mathcal{D}} \subseteq \Sigma_{\mathcal{P}(K)}$.

3.3 Data

We will use the following example [waarde2019data] to give a more precise definition of data.

Lets consider systems from the model class of state space dimension n and input space dimension m without noise or output. Then we know that all systems contained in Σ are of the form:

$$\mathbf{x}(t+1) = A\mathbf{x}(t) + B\mathbf{u}(t) \quad (3)$$

Where $\mathbf{x}(t)$ is the n -dimensional state vector and $\mathbf{u}(t)$ is the m -dimensional input vector evaluated at time t . We will pick a system from Σ and call it our 'true' system. We will denote this system as (A_s, B_s) . We will use our true system to generate/measure the input and state data on q time intervals $\{0, 1, \dots, T_i\}$ where $i \in \{1, 2, \dots, q\}$. We denote the data collected on one of these intervals as follows:

$$\begin{aligned} U_-^i &= \begin{bmatrix} u^i(0) & u^i(1) & \dots & u^i(T_i - 1) \end{bmatrix} \\ X^i &= \begin{bmatrix} x^i(0) & x^i(1) & \dots & x^i(T_i) \end{bmatrix} \end{aligned}$$

We will now 'split' the state data into a 'past' and 'future' segment, these are defined similar to U_-^i .

$$\begin{aligned} X_-^i &= \begin{bmatrix} x^i(0) & \dots & x^i(T_i - 1) \end{bmatrix} \\ X_+^i &= \begin{bmatrix} x^i(1) & \dots & x^i(T_i) \end{bmatrix} \end{aligned}$$

With this representation of our state and input data we have that $X_+^i = A_s X_-^i + B_s U_-^i$. This holds for all measured intervals i of the true system. We will now combine the data of all intervals to get a more general form.

$$\begin{aligned} U_- &= \begin{bmatrix} U_-^1 & \dots & U_-^q \end{bmatrix} & X &= \begin{bmatrix} X^1 & \dots & X^q \end{bmatrix} \\ X_+ &= \begin{bmatrix} X_+^1 & \dots & X_+^q \end{bmatrix} & X_- &= \begin{bmatrix} X_-^1 & \dots & X_-^q \end{bmatrix} \end{aligned}$$

We will define our data $\mathcal{D} := (U_-, X)$. In this example we have that $\Sigma_{\mathcal{D}} = \Sigma_{(U_-, X)} = \Sigma_{i/s}$ and is defined by:

$$\Sigma_{i/s} = \Sigma_{(U_-, X)} = \left\{ (A, B) \mid X_+ = \begin{bmatrix} A & B \end{bmatrix} \begin{bmatrix} X_- \\ U_- \end{bmatrix} \right\} \quad (4)$$

By construction we know that at least the true system (A_s, B_s) is contained in this set.

We can extend this concept to also include the output of a system. Assume we have a system of the form:

$$\mathbf{x}(t+1) = A\mathbf{x}(t) + B\mathbf{u}(t) \quad (5a)$$

$$\mathbf{y}(t+1) = C\mathbf{x}(t) + D\mathbf{u}(t) \quad (5b)$$

Let us define Y_- in the following way:

$$\begin{aligned} Y_-^i &= \begin{bmatrix} y^i(0) & y^i(1) & \dots & y^i(T_i - 1) \end{bmatrix} \\ Y_- &= \begin{bmatrix} Y_-^1 & Y_-^2 & \dots & Y_-^q \end{bmatrix} \end{aligned}$$

Then we can define the set of systems that can describe the data as follows:

$$\Sigma_{i/o/s} = \Sigma_{(U_-, X, Y_-)} = \left\{ (A, B, C, D) \mid \begin{bmatrix} X_+ \\ Y_- \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} X_- \\ U_- \end{bmatrix} \right\} \quad (6)$$

Lastly we will consider systems with bounded noise. These systems will be of the form:

$$\mathbf{x}(t+1) = A_s \mathbf{x}(t) + B_s \mathbf{u}(t) + \mathbf{w}(t) \quad (7)$$

For these systems we will only use a single measurement interval. Hence we will define our noise data W_- as follows.

$$W_- = \begin{bmatrix} w(0) & w(1) & \dots & w(T-1) \end{bmatrix}$$

Note that W_- is unknown, we only have access to our state and input data generated by these systems. Later, in section (13) we will go into more detail on how we define the bound on the noise and how we can use this for control.

Verify if
it really
is one
interval

4 System identification

In this section we will see how we can use the data to identify the true system. We will first consider the mathematics and then we will see how this can be implemented. Lastly we will consider an example that we will solve using the Matlab function based on the previously discussed algorithm.

Def: Informative for system identification [waarde2019data]

We say that the data is informative for system identification if the data only describes 1 system, i.e. $\Sigma_{\mathcal{D}} = \{(A_s, B_s)\}$.

4.1 Mathematics

Consider systems of the form (3) and let (U_-, X) be the data generated by the true system. Assume that we have recorded T data points, i.e. the dimension of U_- is $m \times T$ and the dimension of X is $n \times (T + 1)$. Recall that the set of systems described by this data is given by:

$$\Sigma_{(U_-, X)} = \left\{ (A, B) \mid X_+ = \begin{bmatrix} A & B \end{bmatrix} \begin{bmatrix} X_- \\ U_- \end{bmatrix} \right\} \quad (4)$$

Where $\begin{bmatrix} X_- \\ U_- \end{bmatrix}$ is $(n + m) \times T$. Suppose that the $\text{rank} \left(\begin{bmatrix} X_- \\ U_- \end{bmatrix} \right) = n + m$, then there exists a right inverse $\begin{bmatrix} V_1 & V_2 \end{bmatrix}$ such that $\begin{bmatrix} X_- \\ U_- \end{bmatrix} * \begin{bmatrix} V_1 & V_2 \end{bmatrix} = I_{n+m}$. If we multiply the equation from (4) with this inverse we get the following:

$$X_+ \begin{bmatrix} V_1 & V_2 \end{bmatrix} = \begin{bmatrix} A & B \end{bmatrix} I_{n+m}$$

Thus if the data is full rank we are able to retrieve the (A, B) pair directly. This results in the following proposition.

4.2 Algorithm

Prop: [waarde2019data]

The data (U_-, X) is informative for system identification if and only if

$$\text{rank} \left(\begin{bmatrix} X_- \\ U_- \end{bmatrix} \right) = n + m$$

Furthermore, if the data is full rank, there exists a right inverse $\begin{bmatrix} V_1 & V_2 \end{bmatrix}$ (as defined above), and for any such right inverse $A_s = X_+ V_1$ and $B_s = X_+ V_2$.

Suppose we are considering a system without any inputs, then the proposition reduces to checking if X_- has full row rank and finding a right inverse X_-^\dagger of X_- . Then we retrieve the system as follows $A_s = X_+ X_-^\dagger$.

Lets assume we are considering an input, state, output system of the form:

Recall the set of systems described by the data is given by (6):

$$\Sigma_{(U_-, X, Y_-)} = \left\{ (A, B, C, D) \mid \begin{bmatrix} X_+ \\ Y_- \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} X_- \\ U_- \end{bmatrix} \right\} \quad (6)$$

In this case if the proposition holds, we can also retrieve the C and D matrix by computing $C = Y_- V_1$ and $D = Y_- V_2$.

4.3 Implementation

Syntax

```
[bool, A] = isInformIdentification(X)
[bool, A, B] = isInformIdentification(X, U)
[bool, A, B, C, D] = isInformIdentification(X, U, Y)
```

Description

[bool, A] = isInformIdentification(X): returns if the state data is informative for system identification, if it is then A contains the A is the system matrix in state space representation.

[bool, A, B] = isInformIdentification(X, U): returns if the state and input data is informative for system identification, if it is then A and B are the system matrices in state space representation.

[bool, A, B, C, D] = isInformIdentification(X, U, Y): returns if the state, input and output data is informative for system identification, if it is then A, B, C and D are the system matrices in state space representation.

Input arguments

X: State data matrix of dimension $n \times T + 1$.

U: Input data matrix of dimension $m \times T$.

Y: Output data matrix of dimension $p \times T$.

Output arguments

bool: (boolean) True if the data is informative for system identification, false otherwise

A: (matrix) If the data is informative, it contains the systems A matrix, empty otherwise.

B: (matrix) If the data is informative, it contains the systems B matrix, empty otherwise.

C: (matrix) If the data is informative, it contains the systems C matrix, empty otherwise.

D: (matrix) If the data is informative, it contains the systems D matrix, empty otherwise.

4.4 Examples

Lets consider the following state and input data:

$$X = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad U = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

As we can see the data is not sufficient for system identification:

$$\text{rank} \left(\begin{bmatrix} X_- \\ U_- \end{bmatrix} \right) = \text{rank} \left(\begin{bmatrix} 0 & 1 \\ 0 & 0 \\ 1 & 0 \end{bmatrix} \right) = 2 \neq 3$$

This is because the data can be generated by systems of the following form:

$$\Sigma_{i/s} = \left\{ \left(\begin{bmatrix} 0 & a_1 \\ 1 & a_2 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right) \mid a_1, a_2 \in \mathbb{R} \right\}$$

However, if we were to consider the same data with 1 additional data point then the data would be informative for system identification:

$$X_- = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad U = \begin{bmatrix} 1 & 0 & \alpha \end{bmatrix} \quad \text{rank} \left(\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & \alpha \end{bmatrix} \right) = 3$$

We can also find the same results using the MatLab functions:

```
1 X = [0 1 0 ; 0 0 1]; U = [1 0];  
2 [bool, A, B] = isInformIdentification(X, U)
```

Which will return: [**false**, [], []].

5 Controllability

In this section we will see how we can use the data to conclude if a set of systems are controllable or not. We will first consider the mathematics and then we will see how this can be implemented. Lastly we will consider an example that we will solve using the Matlab function based on the previously discussed algorithm.

Def: Informative for controllability [waarde2019data]

We say that the data is informative for controllability if all systems that describe the data are controllable. I.e. $\Sigma_{i/o} \subseteq \{(A, B) \mid (A, B) \text{ is controllable}\}$.

We will base our algorithm on theorem 8 and remark 9 from [waarde2019data]. These give necessary and sufficient conditions for the informativity.

Thr: Informative for controllability

The data (U_-, X) is informative for controllability if and only if $\text{rank}(X_+ - \lambda X_-) = n \ \forall \lambda \in \mathbb{C}$.

This theorem can be reduced to check a finite amount of complex numbers similar to the Hautus test. We have that the above theorem is equivalent to $\text{rank}(X_+) = n$ and $\text{rank}(X_+ - \lambda X_-) = n$ for all $\lambda \neq 0$ with $\lambda^{-1} \in \sigma(X_- X_+^\dagger)$ with X_+^\dagger being the right inverse of X_+ and $\sigma(\cdot)$ denotes the set of eigenvalues of the matrix.

5.1 Implementation

The algorithm above is implemented in the following functions:

Syntax

```
[bool] = isInformControllable(X)
```

Description

`[bool] = isInformControllable(X)`: Returns if the state data from an input-state dataset is informative for controllability.

Input arguments

X: State data matrix of dimension $n \times T + 1$ from an input-state dataset.

Output arguments

bool: (boolean) True if the data is informative for controllability, false otherwise

5.2 Examples

We will consider the same data as we did in the example of system identification. Recall the provided data was:

$$X = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad U = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

For the data to be informative for controllability we need that $X_+ - \lambda X_-$ is full row rank for all $\lambda \in \mathbb{C}$.

$$\text{rank}(X_+ - \lambda X_-) = \text{rank} \left(\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} 0 & \lambda \\ 0 & 0 \end{bmatrix} \right) = 2$$

Hence the data is informative for controllability.

We can also find the same result by using the Matlab function:

```
1 X = [0 1 0 ; 0 0 1]; U = [1 0];
2 [bool] = isInformControllable(X)
```

Which will return: [1].

We can verify the result by considering the systems that generate this data. Recall that the data is generated by systems of the form:

$$\Sigma_{i/s} = \left\{ \left(\begin{bmatrix} 0 & a_1 \\ 1 & a_2 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right) \mid a_1, a_2 \in \mathbb{R} \right\}$$

Thus the controllability matrix is given by:

$$\begin{bmatrix} B & AB \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Since the controllability matrix has full rank we can conclude that systems of this form are indeed controllable.

6 Stabilisability

In this section we will see how we can use the data to conclude if a set of systems are stabilisable or not. We will first consider the mathematics and then we will see how this can be implemented. Lastly we will consider an example that we will solve using the Matlab function based on the previously discussed algorithm.

Def: Informative for stabilisability [waarde2019data]

We say that the data is informative for stabilisability if all systems that describe the data are stabilisable. I.e. $\Sigma_{i/o} \subseteq \{(A, B) \mid (A, B) \text{ is stabilisable}\}$.

We will base our algorithm on theorem 8 and remark 9 from [waarde2019data]. These give necessary and sufficient conditions for the informativity.

Thr: Informative for stabilisability

The data (U_-, X) is informative for stabilisability if and only if $\text{rank}(X_+ - \lambda X_-) = n \forall \lambda \in \mathbb{C}$ such that $|\lambda| \geq 1$.

This theorem can be reduced to check a finite amount of complex numbers similar to the Hautus test. We have that the above theorem is equivalent to $\text{rank}(X_+ - X_-) = n$ and $\text{rank}(X_+ - \lambda X_-) = n$ for all $\lambda \neq 1$ with $(\lambda - 1)^{-1} \in \sigma(X_-(X_+ - X_-)^\dagger)$ with $(X_+ - X_-)^\dagger$ being the right inverse of $(X_+ - X_-)$ and $\sigma(\cdot)$ denotes the set of eigenvalues of the matrix.

6.1 Implementation

Syntax

```
[bool] = isInformStabilisable(X)
[bool] = isInformStabilisable(X, tolerance)
```

Description

`[bool] = isInformStabilisable(X)`: Returns if the data is informative for stabilisability. Uses default tolerance of `1e-14`.

`[bool] = isInformStabilisable(X, tolerance)`: Returns if the data is informative for stabilisability given a specific tolerance.

Input arguments

X: State data matrix of dimension $n \times T + 1$ from an input-state dataset.

tolerance: Tolerance used for determining when a value is zero up to machine precision. Default value is `1e-14`.

Output arguments

bool: (boolean) True if the data is informative for stabilisability, false otherwise

6.2 Examples

In this example we will consider the same data that we used in system identification and controllability, namely:

$$X = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad U = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

For the data to be informative for stabilisability we need that $X_+ - \lambda X_-$ is full row rank for all $\lambda \in \mathbb{C}$ such that $|\lambda| \geq 1$.

$$\text{rank}(X_+ - \lambda X_-) = \text{rank} \left(\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} 0 & \lambda \\ 0 & 0 \end{bmatrix} \right) = 2$$

Since the rank condition hold for all λ it will also hold for all $|\lambda| \geq 1$. Hence the data is informative for stabilisability. This is consistent with the non data driven theory that controllability implies stabilisability.

We can also find the same result by using the Matlab function:

```
1 X = [0 1 0 ; 0 0 1]; U = [1 0];
2 [bool] = isInformStabilisable(X)
```

Which will return: `[1]`.

We can verify the result by considering the systems that generate this data. Recall that the data is generated by systems of the form:

$$\Sigma_{i/s} = \left\{ \left(\begin{bmatrix} 0 & a_1 \\ 1 & a_2 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right) \mid a_1, a_2 \in \mathbb{R} \right\}$$

Recall that a pair (A, B) is stabilisable if and only if $[A - \lambda I \quad B]$ is full row rank for all $\lambda \in \sigma(A)$ such that $\text{Re}(\lambda) \geq 0$.

$$\text{rank} \left([A - \lambda I \quad B] \right) = \text{rank} \left(\begin{bmatrix} -\lambda & a_1 & 1 \\ 1 & a_2 - \lambda & 0 \end{bmatrix} \right) = 2$$

Thus the systems that describe the data are stabilisable.

7 Stability

In this section we will consider data informativity for stability of unforced systems. We will first consider the mathematics and then we will see how it can be implemented. Lastly we will consider an example that we will solve using the Matlab function based on the previously discussed algorithm.

In this section we will consider our data to be obtained from an unforced system i.e. there was no input. Because of this we will define our data and set of systems as follows:

$$\mathcal{D} = (X) \qquad \Sigma_{\mathcal{D}} = \{A|X_+ = AX_-\}$$

Using these definitions we can define data informativity for stability as follows.

Def: Informative for stability

We say the data is informative for stability if all systems describing the data are stable.

7.1 Mathematics

However, using our definition of informativity for stabilisability we are able to reduce the condition to showing that the data is informative for identification and that the identified system is stable.

Cor: [Cor 11]waarde2019data

The data X is informative for stability if and only if X_- has full row rank and $X_+X_-^\dagger$ is stable for any right inverse X_-^\dagger . This is equivalent to $\Sigma_{\mathcal{D}} = \{A_s\}$ and $A_s = X_+X_-^\dagger$ being stable.

7.2 Implementation

The algorithm above is implemented in the following functions:

Syntax

```
[bool] = isInformStable(X)
```

Description

`[bool] = isInformStable(X)`: Returns if the data of a unforced system is informative for stability.

Input arguments

X: State data matrix of dimension $n \times T + 1$ from an input-less dataset.

Output arguments

bool: (boolean) True if the data is informative for stability, false otherwise

7.3 Examples

For this example we will consider the following state data generated by a unforced system:

$$X = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{4} \\ 0 & \frac{1}{2} & \frac{1}{2} \end{bmatrix}$$

For the data to be informative for stability we need that the data is informative for system identification and that the identified system is stable. The data is informative for system identification if and only if there exists a right inverse X_-^\dagger of X_- . Then the identified system is given by $A = X_+ X_-^\dagger$.

$$A = \begin{bmatrix} \frac{1}{2} & \frac{1}{4} \\ \frac{1}{2} & \frac{1}{2} \end{bmatrix} \begin{bmatrix} 1 & \frac{1}{2} \\ 0 & \frac{1}{2} \end{bmatrix}^{-1} = \begin{bmatrix} \frac{1}{2} & 0 \\ \frac{1}{2} & \frac{1}{2} \end{bmatrix}$$

As we can see from the A matrix its eigenvalues are $\sigma(A) = \{\frac{1}{2}, \frac{1}{2}\}$. Thus the system is stable and hence the data is informative for stability.

We can also find the same result by using the Matlab function:

```
1 X = [1 0.5 0.25; 0 0.5 0.5];  
2 [bool] = isInformStable(X)
```

Which will return: `[1]`.

8 State feedback

In this section we will see when and how we can construct a state feedback controller directly from the state-input data. We will first consider the mathematics after which we will look at how we can implement this. Lastly we will look at an example and how we can apply the provided functions.

In this section we will consider closed loop state feedback. This means that we want to substitute our input with an input that is based on the state of the system, i.e. $u = Kx$ where $K \in \mathbb{R}^{m \times n}$. We will consider the system to be in a stable closed loop form if all eigenvalues of $A + BK$ are stable (inside the unit circle). We can extend this notion to data-driven control as follows.

Def: Informative for state feedback [waarde2019data]

We say that the data (U_-, X_-) is informative for stabilization by state feedback if there exists a feedback gain K such that all systems described by the data are part of the set of systems that are stabilised using the gain K . I.e. $\Sigma_{i/s} \subseteq \Sigma_K$.

We will start by noting that if the data is informative for controllability or stabilisability then there is no guarantee that there also exists a state feedback controller that stabilises all of the systems at once. However, if we find such a controller that we do know that all the systems described by the data are stabilisable. We will see an example of this later on in this section.

8.1 Mathematics

We will start by looking at the following theorem that gives us necessary and sufficient conditions for informativity for stabilisation by state feedback.

Thr: [waarde2019data]

The data (U_-, X_-) is informative for stabilisation by state feedback if and only if the matrix X_- has full row rank and there exists a right inverse X_-^\dagger of X_- such that $X_+X_-^\dagger$ is stable. Moreover, K is such that $\Sigma_{i/s} \subseteq \Sigma_K$ if and only if $K = U_-X_-^\dagger$, where X_-^\dagger is as described above.

From this theorem we can see that the problem can be reduced to finding a specific right inverse such that the systems are stable. From this right inverse we are able to construct the corresponding controller for stabilisation by state feedback.

However, in its current form it is not straightforward to compute a right inverse such that $X_+X_-^\dagger$ is stable. Hence we will use a different version of this theorem that has been rewritten in terms of linear matrix inequalities.

Thr: [waarde2019data]

The data (U_-, X_-) is informative for stabilisation by state feedback if and only if there exists a matrix $\Theta \in \mathbb{R}^{T \times n}$ satisfying

$$X_- \Theta = (X_- \Theta)^\top \quad \begin{bmatrix} X_- \Theta & X_+ \Theta \\ \Theta^\top X_+^\top & X_- \Theta \end{bmatrix} > 0$$

Moreover, K satisfies $\Sigma_{i/s} \subseteq \Sigma_K$ if and only if $K = U_- \Theta (X_- \Theta)^{-1}$ for some matrix Θ satisfying the above conditions.

This version of the theorem can be implemented using linear matrix inequality solvers such as Yalmip or CVX. The functions provided in the toolbox are implemented using Yalmip.

8.2 Implementation

Syntax

```
[bool, K, diagnostics, info] = isInformStateFeedback(X, U)
[bool, K, diagnostics, info] = isInformStateFeedback(X, U, tolerance)
[bool, K, diagnostics, info] = isInformStateFeedback(X, U, tolerance, options)
```

Description

`[bool, K, diagnostics, info] = isInformStateFeedback(X, U)`: Returns if the data is informative for stabilisation by state feedback. If so, it also returns a corresponding controller K for closed loop feedback control of the form $A+BK$.

`[bool, K, diagnostics, info] = isInformStateFeedback(X, U, tolerance)`: Returns if the data is informative for stabilisation by state feedback given a specific tolerance. If so, it also returns a corresponding controller K for closed loop feedback control of the form $A+BK$.

`[bool, K, diagnostics, info] = isInformStateFeedback(X, U, tolerance, options)`: Returns if the data is informative for stabilisation by state feedback given a specific tolerance and a `sdpssettings` object. If so, it also returns a corresponding controller K for closed loop feedback control of the form $A+BK$.

Input arguments

X: State data matrix of dimension $n \times T + 1$ from an input-state dataset..

U: Input data matrix of dimension $m \times T$ from an input-state dataset..

tolerance: Tolerance used for determining when a value is zero up to machine precision. Default value is `1e-14`.

options: `sdpssettings` used with the Yalmip solver.

Output arguments

bool: (boolean) True if the data is informative for stabilisation by state feedback, false otherwise. If false then the `info` variable can be checked to find which condition failed.

K: (matrix) If the data is informative, it contains a stabilising controller K for closed loop control $A+BK$, empty otherwise.

diagnostics: (struct) Diagnostics from the Yalmip `optimize()` function.

info: (int) Diagnostic variable used to identify which conditions (if any) failed. The verification is done on the solution obtained from Yalmip. For information about the type of error use the `help` command in Matlab

Limitation

Due to computational limitation in the solver or conditioning of this problem it might be the case that a valid K is found even though `bool` is false. This is because we are able to find a solution close

enough to the real solution that the results are still satisfactory, even though not all conditions are met.

8.3 Examples

We will consider 2 examples in which the data is not informative for system identification. In both of the examples the data is informative for controllability. However in only one of the example will the data be informative for stabilisation by state feedback.

Example 1 Let us consider the data from the controllability example.

$$X = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad U = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

From that example we know that the data is informative for controllability and that the general form of systems that can produce this data is given by:

$$\Sigma_{i/s} = \left\{ \left(\begin{bmatrix} 0 & a_1 \\ 1 & a_2 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right) \mid a_1, a_2 \in \mathbb{R} \right\}$$

Using this we can write the general closed loop form.

$$x(t+1) = \begin{bmatrix} -k_1 & a_1 - k_2 \\ 1 & a_2 \end{bmatrix} x(t)$$

Where k_1 and k_2 are fixed and $a_1, a_2 \in \mathbb{R}$. Using the first theorem we conclude that the data is not informative for stabilisation by state feedback since X_- does not have full row rank.

We can also find the same result by using the Matlab function:

```
1 X = [0 1 0; 0 0 1]; U = [1 0];
2 [bool] = isInformStateFeedback(X, U)
```

Which will return: `[0]`.

Example 2 [waarde2019data] In the next example we will consider the following data.

$$X = \begin{bmatrix} 1 & 0.5 & -0.25 \\ 0 & 1 & 1 \end{bmatrix} \quad U = \begin{bmatrix} -1 & -1 \end{bmatrix}$$

Since X_- is square and non-singular, we are able to use the first theorem to check if $X_+X_-^{-1}$ is stable.

$$X_+X_-^{-1} = \begin{bmatrix} 0.5 & -0.5 \\ 1 & 0.5 \end{bmatrix}$$

Since its eigenvalues are $\frac{1}{2}(1 \pm \sqrt{2}i)$ we can conclude that $X_+X_-^{-1}$ is stable. Hence the data is informative for stabilisation by state feedback and the corresponding controller is given by:

$$K = U_-X_-^{-1} = \begin{bmatrix} -1 & -0.5 \end{bmatrix}$$

We will note that the general form for systems that can produce this data is given by:

$$\Sigma_{i/s} = \left\{ \left(\begin{bmatrix} 1.5 + a_1 & 0.5 a_1 \\ 1 + a_2 & 0.5 + 0.5 a_2 \end{bmatrix}, \begin{bmatrix} 1 + a_1 \\ a_2 \end{bmatrix} \right) \mid a_1, a_2 \in \mathbb{R} \right\}$$

Using this we are able to verify that the closed loop system is indeed stable for all values a_1 and a_2 .

$$A + BK = \begin{bmatrix} 1.5 + a_1 & 0.5 a_1 \\ 1 + a_2 & 0.5 + 0.5 a_2 \end{bmatrix} + \begin{bmatrix} -1 - a_1 & -0.5 - 0.5 a_1 \\ -a_2 & -0.5 a_2 \end{bmatrix} = \begin{bmatrix} 0.5 & -0.5 \\ 1 & 0.5 \end{bmatrix}$$

Which, as we saw earlier, is indeed stable.

We can also find the same result by using the Matlab function:

```
1 X = [1 0.5 0.25; 0 1 1]; U = [-1 -1];  
2 [bool, K] = isInformStateFeedback(X, U)
```

Which will return: [0, [-1 -0.5]]. See the limitation section for more details on why bool is false.

9 Deadbeat control

In this section we will see how we can find a deadbeat controller using only our data. We will first discuss the underlying mathematics after which we will focus on how we can implement this into code. We will also look at a few examples on how to use the provided function.

Def: Informative for deadbeat control [waarde2019data]

We say the data (U_-, X_-) is informative for deadbeat control if there exists a feedback gain K such that all closed loop systems are nilpotent. I.e. $\Sigma_{i/s} \subseteq \Sigma_K^{nil}$.

In other words, the data is informative if we can stabilise all systems described by the data with a given K such that all closed loop systems are stable in a single step, i.e. only have the zero eigenvalue.

9.1 Mathematics deadbeat control

We will begin by considering the following theorem that gives necessary and sufficient conditions for informativity for deadbeat control.

Thr: [waarde20]

The data (U_-, X_-) is informative for deadbeat control if and only if the matrix X_- has full row rank and there exists a right inverse X_-^\dagger of X_- such that $X_+X_-^\dagger$ is nilpotent. Moreover, if this condition is satisfied then the feedback gain $K := U_-X_-^\dagger$ yields a deadbeat controller.

Using this theorem the problem boils down to finding a suitable right inverse such that $X_-X_-^\dagger$ is nilpotent given that X_- has full row rank. We will consider the following 2 cases, first we will look at the case where X_- is a full rank square matrix. Second we will look at the case where X_- is a full row rank wide matrix ($T > n$).

First let's assume X_- is a square matrix and has full row rank. Then we know that the only right inverse of X_- is its inverse X_-^{-1} . Hence the data is informative for deadbeat control if and only if $X_+X_-^{-1}$ is nilpotent.

Now let's assume that X_- is a wide matrix and has full row rank. Then there exists an $F \in \mathbb{R}^{T \times n}$ spanning the row space of X_- and an $G \in \mathbb{R}^{T \times (T-n)}$ spanning the null space of X_- such that $\begin{bmatrix} F & G \end{bmatrix}$ is non singular and $X_- \begin{bmatrix} F & G \end{bmatrix} = \begin{bmatrix} I_n & 0_{n \times (T-n)} \end{bmatrix}$. Now let $X_-^\dagger = F + GH$ where $H \in \mathbb{R}^{(T-n) \times n}$ if and only if X_-^\dagger is a right inverse of X_- .

Proof ' \Rightarrow ':

Let $X_-^\dagger = F + GH$ as described above. Then $X_-X_-^\dagger = X_-F + X_-GH = I_n + 0_{n \times (T-n)} * H = I_n$.

Proof ' \Leftarrow ':

Thus finding a right inverse such that $X_+X_-^\dagger$ is nilpotent is equivalent to finding an H such that $X_+F + X_+GH$ is nilpotent. Finding this H can be done using pole placement methods for the pair (X_+F, X_+G) and placing all eigenvalues at zero.

However, due to limitation in Matlab's pole placement function we are not able to implement the algorithm into Matlab directly. This is because Matlab has 2 functions for pole placement, the `place()` function which does not support poles with high multiplicity, which is hence not useable.

add
proof

The other function `acker()` does support pole placement of poles with high multiplicity, but it is limited to a single dimensional input space. Since X_+G is an $n \times (T - n)$ matrix, we have that our input space is of dimension $T - n$. However, we are able to circumvent this issue by extending the `acker()` function to support higher dimensional input. To do this we will use the proof of the sufficiency in theorem 3.29 as well as lemma 3.31 from [bookTrentelman].

9.2 Mathematics extending pole placement

Lem: [bookTrentelman]

If (A, B) is controllable there exist vectors u_0, \dots, u_{n-1} such that the vectors defined by

$$x_0 := 0, \quad x_{k+1} := Ax_k + Bu_k \quad (k = 0, \dots, n-1)$$

are independent.

Assume that the pair (A, B) is controllable, we will use [bookTrentelman] to construct u_0, \dots, u_{n-1} and x_0, x_1, \dots, x_n . We pick u_n to be arbitrary. Then there exists a unique map F_0 such that $F_0x_k = u_k$ for $k = 1, \dots, n$. We can substitute this to get:

$$x_{k+1} = Ax_k + BF_0x_k = (A + BF_0)x_k$$

for $k \in [1, \dots, n]$. From this we can see that $x_k = (A + BF_0)^{k-1}x_1$. By construction of lemma 3.31 we have that $x_0 = 0$, thus we have $x_1 = A * 0 + B * u_0 = Bu_0$. We will call this $b = Bu_0 = x_1$. By construction x_1, \dots, x_n are independent. Thus we know that the controllability matrix $\begin{bmatrix} b & (A + BF_0)b & \dots & (A + BF_0)^{n-1}b \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & \dots & x_n \end{bmatrix}$ is full rank. Hence the pair $(A + BF_0, b)$ is controllable. Since the matrix b is a column vector we can use the `acker()` algorithm to compute a suitable controller f such that $A + BF_0 + bf = A + B(F_0 + u_0f)$ has appropriate eigenvalues. From this we construct the controller $F = F_0 + u_0f$ that places the eigenvalues in the chosen positions.

9.3 Implementation

The algorithm above is implemented in the following functions:

Syntax

```
[bool, K] = isInformDeadbeatControl(X, U)
[K] = MultiInputAcker(A, B, poles)
```

Description

`[bool, K] = isInformDeadbeatControl(X, U)`: Returns if the data is informative for deadbeat control. If so, it also returns a corresponding controller K for closed loop feedback control of the form $A+BK$.

`[K] = MultiInputAcker(A, B, poles)`: Returns a controller of the form $A+BK$ such that the eigenvalues are placed at `poles`.

Input arguments

X: State data matrix of dimension $n \times T + 1$.
U: Input data matrix of dimension $m \times T$.
A: A matrix of state space representation.
B: B matrix of state space representation.
poles: Desired poles for pole placement.

Output arguments

bool: (boolean) True if the data is informative for deadbeat control, false otherwise
K: (matrix) Contains a stabilising controller K for closed loop control $\mathbf{A} + \mathbf{BK}$, empty otherwise.

9.4 Examples

In this example we will consider the following data:

$$X = \begin{bmatrix} 0 & 2 & 1 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix} \quad U = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$$

Before we look if the data is informative for deadbeat control, we will first consider if the data is informative for system identification and controllability. Recall that for system identification we need that $\begin{bmatrix} X_- \\ U_- \end{bmatrix}$ is full row rank and for controllability we need that $X_+ - \lambda X_-$ is full row rank for all $\lambda \in \mathbb{C}$.

$$\begin{aligned} \text{rank}\left(\begin{bmatrix} X_- \\ U_- \end{bmatrix}\right) &= n + m & \text{rank}(X_+ - \lambda X_-) &= n \\ \text{rank}\left(\begin{bmatrix} 0 & 2 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}\right) &= 3 & \text{rank}\left(\begin{bmatrix} 2 & 1 - 2\lambda & -\lambda \\ 1 - \lambda & -\lambda & 0 \end{bmatrix}\right) &= 2 \end{aligned}$$

Hence the data is informative for system identification and controllability. Due to the being able to identify the system and the system being controllable we expect that we are able to find a deadbeat controller. Before we actually identify the system we will first attempt to construct one from the data directly. For this we will first need to find an F and G such that $X_- \begin{bmatrix} F & G \end{bmatrix} = \begin{bmatrix} I_2 & 0_{2 \times 1} \end{bmatrix}$. We find the following F and G :

$$F = \begin{bmatrix} -\frac{1}{3} & \frac{5}{6} \\ \frac{1}{3} & \frac{1}{6} \\ \frac{1}{3} & -\frac{1}{3} \end{bmatrix} \quad G = \begin{bmatrix} \frac{1}{\sqrt{6}} \\ -\frac{1}{\sqrt{6}} \\ \frac{\sqrt{2}}{\sqrt{3}} \end{bmatrix}$$

Any right inverse of X_- can be expressed as $X_-^\dagger = F + GH$ where $H \in \mathbb{R}^{1 \times 2}$. Recall that we need to find a right inverse such that $X_+ X_-^\dagger$ is nilpotent. Hence we need to find an H such that $X_+ F + X_+ GH$ only has zero eigenvalues.

$$X_+ F + X_+ GH = \begin{bmatrix} -\frac{1}{3} & \frac{11}{6} \\ -\frac{1}{3} & \frac{5}{6} \end{bmatrix} + \begin{bmatrix} \frac{1}{\sqrt{6}} \\ \frac{1}{\sqrt{6}} \end{bmatrix} \begin{bmatrix} h_1 & h_2 \end{bmatrix}$$

By applying pole placement methods we can find that $H = \begin{bmatrix} \frac{\sqrt{2}}{\sqrt{3}} & \frac{-5}{\sqrt{6}} \end{bmatrix}$ does indeed provide a right inverse such that $X_+ X_-^\dagger$ is nilpotent. Using this right inverse we can construct the deadbeat controller K as follows.

$$K = U_- * X_-^\dagger = U_-(F + GH) = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 2 \end{bmatrix} = \begin{bmatrix} 0 & 0 \end{bmatrix}$$

At this point we will recall that we were able to identify the system using the data. If we do this we find that the system is given by:

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

As we can see the system already has only zero eigenvalues, hence it will follow the properties of deadbeat control if we provide no input.

We can also find the same results by using the Matlab function:

```
1 X = [0 2 1 0 ; 1 1 0 0]; U = [1 0 0];  
2 [bool, K] = isInformDeadbeatControl(X,U)
```

Which will return: `[1, [0.5329e-14 -0.6217e-14]]`.

10 Linear quadratic regulation

In this section we will see how we can construct a linear quadratic regulator (LQR) based on a given cost function from the input-state data. We will first define what LQR control entails for normal control theory after which we will consider its data-driven dual. We will also see how this data-driven version can be applied to retrieve the controller from the data. Lastly we will see how to solve such a problem both by doing it by hand as well as using the provided Matlab functions.

10.1 Non data-driven LQR

We will first revisit the topic of non data-driven LQR. To give an intuitive idea, we want to not only find a solution to the stabilisation problem, we also want to find the best solution given some cost function. Let us consider the discrete-time linear systems of the form $x(t+1) = Ax(t) + Bu(t)$. We will denote the state sequence generated by this system based on a given initial condition x_0 and input function u as $x_{x_0, u}(\cdot)$. We will also define a quadratic cost function associated to the system.

$$J(x_0, u) = \sum_{t=0}^{\infty} x^{\top}(t)Qx(t) + u^{\top}(t)Ru(t)$$

Where Q is symmetric positive semi definite and R is symmetric positive definite. Using this, we want to find for every initial condition x_0 an input \bar{u} such that the system stabilises and the cost function is minimised as well as finite, In other words, the $\lim_{t \rightarrow \infty} x_{x_0, \bar{u}}(t) = 0$ given that the cost function $J(x_0, \bar{u}) \leq J(x_0, u)$ for all inputs u that cause the system to stabilise. Such an input \bar{u} is called optimal for the given initial condition. As we might expect, an optimal solution does not always exist for every initial condition. Hence we say that an LQR problem is solvable for given (A, B, Q, R) if for every initial condition x_0 there exists an optimal solution.

Using the following theorem we can find the solution to the LQR problem. Recall from [bookTrentelman] that an eigenvalue is called (Q, A) -observable if

$$\text{rank} \begin{pmatrix} A - \lambda I \\ Q \end{pmatrix} = n$$

where n is the dimension of the state space.

Thr: [waarde2019data]

let Q be symmetric positive semi definite and R be symmetric positive definite. Then the following statements hold:

1. If (A, B) is stabilisable, there exists a unique largest real symmetric solution P^+ to the discrete-time algebraic Riccati equation

$$P = A^{\top}PA - A^{\top}PB(R + B^{\top}PB)^{-1}B^{\top}PA + Q \quad (8)$$

in the sense that $P^+ \geq P$ for every real symmetric P satisfying the discrete-time algebraic Riccati equation. The matrix P^+ is positive semi definite.

2. If, in addition to stabilisability of (A, B) , every eigenvalue of A on the unit circle is (Q, A) -observable then for every x_0 a unique optimal input \bar{u} exists. Furthermore, this input sequence

is generated by the feedback law $u = Kx$, where K is given by:

$$K := -(R + B^\top P^+ B)^{-1} B^\top P^+ A \quad (9)$$

Moreover, the matrix $A + BK$ is stable.

3. In fact, the linear quadratic regulator problem is solvable for (A, B, Q, R) if and only if (A, B) is stabilisable and every eigenvalue of A on the unit circle is (Q, A) -observable.

If a LQR problem is solvable then we call K the optimal feedback gain for (A, B, Q, R) .

10.2 Data-driven LQR

Now that we have familiarised ourselves with the concept of LQR control we will see how to extend it to data-driven control. We will start by defining the set of systems for which a solution is valid given a feedback gain K and cost matrices Q and R .

$$\Sigma_K^{Q,R} := \{(A, B) : K \text{ is the optimal gain for } (A, B, Q, R)\}$$

Using this we can define informativity for linear quadratic regulation as follows.

Def: Informative for linear quadratic regulation [waarde2019data]

Given matrices Q and R , we say that the data (U_-, X) is informative for linear quadratic regulation if there exists feedback gain K such that the optimal gain for all systems is K , i.e. $\Sigma_{i/s} \subseteq \Sigma_K^{Q,R}$.

We will now consider the following theorem that gives us necessary and sufficient conditions for finding the solution of a data-driven LQR problem.

Thr: [waarde2019data]

Let Q be symmetric positive semi definite and R be symmetric positive definite. Then the data (U_-, X) is informative for linear quadratic regulation if and only if at least one of the following two conditions hold:

1. The data (U_-, X) is informative for system identification, that is $\Sigma_{i/s} = \{(A_s, B_s)\}$, and the linear quadratic regulator problem is solvable for (A_s, B_s, Q, R) . In this case, the optimal feedback gain K is of the form (9) where P^+ is of the form (8).
2. For all $(A, B) \in \Sigma_{i/s}$ we have $A = A_s$. Moreover, A_s is stable, $QA_s = 0$, and the optimal feedback gain is given by $K = 0$.

As we can see from the theorem if we are able to identify the system then the data-driven LQR problem reduces to a normal LQR problem for which we have already seen how to solve them. However, there is still the case in which all systems have the same $A = A_s$ matrix, the A_s matrix is inherently stable and $QA_s = 0$. We will now look at why this is a valid solution to the LQR problem in the first place.

Assume we have a A_s which is stable and a Q such that $QA_s = 0$. From this point we assume B to be an arbitrary input matrix of dimension $n \times m$. Since $x(t) \in \text{im}(A)$ for all $t > 0$ we have that $Qx(t) = 0$ for all $t > 0$ if we pick our input function to be zero. Note that if we pick our input to be zero then the input cost of the cost function will also be zero. Hence the cost will be equal to $J(x_0, u) = x_0^\top Qx_0$, which is finite. Since the system is inherently stable we know that the system

will stabilise without any input. Lastly we note that the cost associated with the initial condition is present in all cost sequences generated for different inputs, hence the zero input case is the minimal solution to the cost function. Hence this case will provide a valid solution to our LQR problem.

However, even though it is a valid solution it is not easily computable in its current form. Hence we will consider the following theorem which rewrites the condition to one of linear matrix inequalities which can be solved by computational means.

Thr: [waarde2019data]

Let Q be symmetric positive semi definite and R be symmetric positive definite. Then the data (U_-, X) is informative for linear quadratic regulation if and only if at least one of the following two conditions hold:

1. *The data (U_-, X) is informative for system identification, that is $\Sigma_{i/s} = \{(A_s, B_s)\}$, and the linear quadratic regulator problem is solvable for (A_s, B_s, Q, R) . In this case, the optimal feedback gain K is of the form (9) where P^+ is of the form (8).*
2. *There exists $\Theta \in \mathbb{R}^{T \times n}$ such that $X_- \Theta = (X_- \Theta)^\top$, $U_- \Theta = 0$, $Q X_+ \Theta = 0$ and*

$$\begin{bmatrix} X_- \Theta & X_+ \Theta \\ \Theta^\top X_+^\top & X_- \Theta \end{bmatrix} > 0$$

In this case, the optimal feedback gain $K = 0$.

10.3 Examples using implementation

The algorithm above is implemented in the following functions:

Syntax

```
[bool, K, diagnostics] = isInformLQR(X, U, Q, R)
[bool, K, diagnostics] = isInformLQR(X, U, Q, R, tolerance)
[bool, K, diagnostics] = isInformLQR(X, U, Q, R, tolerance, options)
```

Description

`[bool, K, diagnostics] = isInformLQR(X, U, Q, R)`: Returns if the data is informative for LQR control. If so, it also returns a corresponding controller K for the closed loop feedback control of the form $A+BK$.

`[bool, K, diagnostics] = isInformLQR(X, U, Q, R, tolerance)`: Returns if the data is informative for LQR control given a specific tolerance. If so, it also returns a corresponding controller K for the closed loop feedback control of the form $A+BK$.

`[bool, K, diagnostics] = isInformLQR(X, U, Q, R, tolerance, options)`: Returns if the data is informative for LQR control given a specific tolerance and `sdpsettings`. If so, it also returns a corresponding controller K for the closed loop feedback control of the form $A+BK$.

Input arguments

X: State data matrix of dimension $n \times T + 1$ from an input-state dataset.

U: Input data matrix of dimension $m \times T$ from an input-state dataset.

Q: State cost matrix.

R: Input cost matrix.

tolerance: Tolerance used for determining when a value is zero up to machine precision. Default value is $1e-14$.

options: sdpsettings used with the Yalmip solver.

Output arguments

bool: (boolean) True if the data is informative for system identification, false otherwise

K: (matrix) If the data is informative, it contains a stabilising controller K for closed loop control $A+BK$, empty otherwise.

diagnostics: (struct) Diagnostics from the Yalmip `optimize()` function.

10.3.1 Examples

We will consider an example in which all systems have the same A matrix, A is stable and $QA = 0$. We will pick the input data in such a way that it is not full rank. For this we will use the following data

$$X = \begin{bmatrix} 1 & 1.5 & 1.75 \end{bmatrix} \quad U = \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix} \quad Q = \begin{bmatrix} 0 \end{bmatrix} \quad R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Note that the choice of R is arbitrary since we have picked the example such that the optimal input will be no input. We will start by checking $U_- \Theta = 0$.

$$U_- \Theta = 0$$

$$\begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} a+b \\ 0 \end{bmatrix}$$

Hence $\Theta = \begin{bmatrix} a & -a \end{bmatrix}^\top$. Since $Q = 0$ we know that $QX_+ \Theta = 0$ is satisfied. We also know that $X_- \Theta = (X_- \Theta)^\top$ will hold since $X_- \Theta$ will be a scalar. Hence we only need to check the matrix inequality.

$$\begin{bmatrix} X_- \Theta & X_+ \Theta \\ \Theta^\top X_+^\top & X_- \Theta \end{bmatrix} = \begin{bmatrix} \frac{1}{2}a & \frac{1}{4}a \\ \frac{1}{4}a & \frac{1}{2}a \end{bmatrix} > 0$$

If we pick $a = 1$ then the eigenvalues of the matrix will be $\{\frac{1}{4}, \frac{3}{4}\}$ and hence the matrix inequality is satisfied.

We can also find the same result by using the Matlab function:

```

1 X = [1 1.5 1.75];
2 U = [1 1 ; 0 0];
3 Q = [0];
4 R = [1 0 ; 0 1];
5 [bool, K] = isInformLQR(X, U, Q, R)

```

Which will return: `[1, [0 ; 0]]`.

11 Dynamic measurement feedback

In this section we will consider input-state-output systems and how we can use their data to construct a dynamic measurement feedback controller that stabilises the closed loop system. We will start by recalling what a dynamic measurement feedback controller entails. After which we will consider how we can extend this to data-driven control and how we can define the set of systems as well as informativity. We will then consider necessary and sufficient conditions for finding a dynamic measurement feedback controller for input-state-output data. In the next section we will introduce the concept of state estimation which we will use to extend our theory about dynamic measurement feedback to work using only input and output data.

11.1 Non-data driven dynamic feedback control

In this section we will consider discrete time systems that are of the following form.

$$x(t+1) = A_s x(t) + B_s u(t) \quad (10a)$$

$$y(t) = C_s x(t) + D_s u(t) \quad (10b)$$

Instead of only using state feedback we will also add an observer to the loop. By combining the observer and the state feedback we get a dynamic measurement controller (see [bookTrentelman]).

$$w(t+1) = K w(t) + L y(t)$$

$$u(t) = M w(t)$$

Using this controller we can write the closed loop system as:

$$\begin{bmatrix} x(t+1) \\ w(t+1) \end{bmatrix} = \begin{bmatrix} A_s & B_s M \\ L C_s & K + L D_s M \end{bmatrix} \begin{bmatrix} x(t) \\ w(t) \end{bmatrix}$$

For this controller to be a stabilising controller we need that the closed loop system is stable.

11.2 Data-driven stabilisation by dynamic measurement feedback

Recall that the $\Sigma_{i/o/s}$ is given by:

$$\Sigma_{i/o/s} = \left\{ (A, B, C, D) \mid \begin{bmatrix} X_+ \\ Y_- \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} X_- \\ U_- \end{bmatrix} \right\} \quad (6)$$

We will now define the set of systems to be stabilised using the dynamic feedback controller (K, L, M) as follows.

$$\Sigma_{K,L,M} = \left\{ (A, B, C, D) \mid \begin{bmatrix} A & B M \\ L C & K + L D M \end{bmatrix} \text{ is stable} \right\}$$

Using these sets we can construct a definition for informativity for dynamic measurement feedback.

Def: Informative for stabilisation by dynamic measurement feedback [waarde2019data]
We say that the data (U_-, X, Y_-) is informative for stabilisation by dynamic measurement feedback

if there exists matrices K , L and M such that all systems described by the data are stabilised using the dynamic measurement controller (K, L, M) . I.e. $\Sigma_{i/o/s} \subseteq \Sigma_{K,L,M}$.

We will start by assuming that our input data is of full rank. The following theorem provides us with a condition on when the data is informative for dynamic measurement feedback.

Thr: [waarde2019data]

Consider the data (U_-, X, Y_-) and assume that U_- has full row rank. Then the data is informative for stabilisation by dynamic measurement feedback if and only if the following conditions are satisfied:

1. We have that the data is informative for system identification.

$$\Sigma_{i/o/s} = \{(A_s, B_s, C_s, D_s)\}$$

2. The pair (A_s, B_s) is stabilisable and the pair (C_s, A_s) is detectable.

Moreover, if the above conditions are satisfied, a stabilising controller (K, L, M) can be constructed as follows:

1. Take M such that $A_s + B_s M$ is stable.
2. Take L such that $A_s - L C_s$ is stable.
3. Define $K = A_s + B_s M - L C_s - L D_s M$.

Although the results of this theorem is useful it is quite limited in applicability. Hence we will consider the following lemma that we will use to remove the full input rank restriction from the previous theorem.

Lem: [waarde2019data]

Consider the data (U_-, X, Y_-) and the corresponding set $\Sigma_{i/o/s}$. Let the rank U_- be k where $k \leq m$. Then we can find a matrix S of size $m \times k$ such that S has full column rank and $U_- = S \bar{U}_-$ where \bar{U}_- has full row rank of dimension k . Then the data (U_-, X, Y_-) is informative for stabilisation by dynamic measurement feedback if and only if the data (\bar{U}_-, X, Y_-) is informative for stabilisation by dynamic measurement feedback. Moreover, if we define $\bar{\Sigma}_{i/o/s}$ to be the set of systems that are described by the reduced data (\bar{U}_-, X, Y_-) , $(\bar{K}, \bar{L}, \bar{M})$ is the corresponding dynamic measurement controller and S^\dagger is the left inverse of S , then

$$\begin{aligned} \Sigma_{i/o/s} \subseteq \Sigma_{K,L,M} &\implies \bar{\Sigma}_{i/o/s} \subseteq \Sigma_{K,L,S^\dagger M} \\ \bar{\Sigma}_{i/o/s} \subseteq \Sigma_{\bar{K},\bar{L},\bar{M}} &\implies \Sigma_{i/o/s} \subseteq \Sigma_{\bar{K},\bar{L},S\bar{M}} \end{aligned}$$

Using this duality we are able to extend the previous theorem to work for cases in which the input data is not full rank.

Cor: [waarde2019data]

Let S be any full column rank matrix such that $U_- = S \bar{U}_-$ with \bar{U}_- full row rank k . The data (U_-, X, Y_-) is informative for stabilisation by dynamic measurement feedback if and only if the following two conditions are satisfied:

1. (\bar{U}_-, X, Y_-) is informative for system identification.

$$\Sigma_{i/o/s} = \{(\bar{A}_s, \bar{B}_s, \bar{C}_s, \bar{D}_s)\}$$

2. The pair (\bar{A}_s, \bar{B}_s) is stabilisable and the pair (\bar{C}_s, \bar{A}_s) is detectable.

Moreover, if the above conditions are satisfied, a stabilising controller (K, L, M) is constructed as follows:

1. Let \bar{M} be such that $\bar{A}_s + \bar{B}_s \bar{M}$ is stable.
2. Take $M = S \bar{M}$.
3. Take L such that $\bar{A}_s - L \bar{C}_s$ is stable.
4. Define $K = \bar{A}_s + \bar{B}_s \bar{M} - L \bar{C}_s - L \bar{D}_s \bar{M}$.

Note that if we were to substitute $U_- = S \bar{U}_-$ in the state space equations we get the following.

$$\begin{bmatrix} X_+ \\ Y_- \end{bmatrix} = \begin{bmatrix} A_s & B_s \\ C_s & D_s \end{bmatrix} \begin{bmatrix} X_- \\ U_- \end{bmatrix} = \begin{bmatrix} A_s & B_s S \\ C_s & D_s S \end{bmatrix} \begin{bmatrix} X_- \\ \bar{U}_- \end{bmatrix}$$

If we assume that the state and reduced input data have full rank then we can see that we are able to identify the systems A_s and C_s matrices uniquely. However we are only able to identify the systems B_s and D_s matrices up to similarity transformation S .

11.3 Implementation

The algorithm above is implemented in the following functions:

Syntax

```
[bool, K, L, M] = isInformDynamicMeasurementFeedback(X, U, Y)
[bool, K, L, M] = isInformDynamicMeasurementFeedback(X, U, Y, polesM, polesL)
```

Description

`[bool, K, L, M] = isInformDynamicMeasurementFeedback(X, U, Y)`: Returns if the data is informative for dynamic measurement feedback. If the data is informative the function also returns a controller (K, L, M) where the poles of M and L are placed at $\{0/n, \dots, (n-1)/n\}$ where n is the dimension of the state space.

`[bool, K, L, M] = isInformDynamicMeasurementFeedback(X, U, Y, polesM, polesL)`: Returns if the data is informative for dynamic measurement feedback. If the data is informative the function also returns a controller (K, L, M) where the poles of M and L are placed at the provided locations.

Input arguments

X: State data matrix of dimension $n \times T + 1$ from a input/state/output dataset.

U: Input data matrix of dimension $m \times T$ from a input/state/output dataset.

Y: Output data matrix of dimension $p \times T$ from a input/state/output dataset.

polesM: An array containing the desired pole locations for **A** + **BM**.

polesL: An array containing the desired pole locations for **A** - **LC**.

Output arguments

bool: (boolean) True if the data is informative for dynamic measurement feedback, false otherwise

K: (matrix) The K matrix of the dynamic measurement feedback controller.

L: (matrix) The L matrix of the dynamic measurement feedback controller.

M: (matrix) The M matrix of the dynamic measurement feedback controller.

11.4 Examples

We will consider an example in which the input data is not full rank. We will consider the following data.

$$U_- = \begin{bmatrix} 1 & -1 & 2 & 0 \\ -1 & 1 & -2 & 0 \end{bmatrix}$$

$$X = \begin{bmatrix} 1 & 3 & 2 & 1 & -4 \\ 0 & 3 & -5 & -6 & -2 \end{bmatrix}$$

$$Y_- = \begin{bmatrix} 2 & 6 & 4 & 2 \end{bmatrix}$$

We will start by finding a matrix S such that $U_- = S\bar{U}_-$ where S has full column rank and \bar{U}_- has full row rank.

$$U_- = \begin{bmatrix} 1 \\ -1 \end{bmatrix} \begin{bmatrix} 1 & -1 & 2 & 0 \end{bmatrix} = S\bar{U}_-$$

We will now use the reduced input sequence to identify the system. Note that this system will be different from the true systems, since those were not identifiable.

$$\bar{A} = \begin{bmatrix} 2 & 1 \\ -2 & 0 \end{bmatrix} \quad \bar{B} = \begin{bmatrix} 1 \\ -1 \end{bmatrix} \quad \bar{C} = \begin{bmatrix} 2 & 0 \end{bmatrix} \quad \bar{D} = \begin{bmatrix} 0 \end{bmatrix}$$

Lastly, we have to check if the pair (\bar{A}, \bar{B}) is stabilisable and the pair (\bar{C}, \bar{A}) is detectable. Before we check these conditions, we will first check if the system is controllable and observable, since this directly implies that it is also stabilisable and detectable.

$$\text{rank} \begin{bmatrix} \bar{B} & \bar{A}\bar{B} \end{bmatrix} = \text{rank} \begin{bmatrix} 1 & 1 \\ -1 & -2 \end{bmatrix} = 2 \quad \text{rank} \begin{bmatrix} \bar{C} \\ \bar{C}\bar{A} \end{bmatrix} = \text{rank} \begin{bmatrix} 2 & 4 \\ 0 & 2 \end{bmatrix} = 2$$

From this we can conclude that the data is indeed informative for stabilisation by dynamic measurement feedback. At this point we can use non data-driven techniques to find a stabilising dynamic

feedback controller (K, L, \bar{M}) for our system $(\bar{A}, \bar{B}, \bar{C}, \bar{D})$. After we have found such a controller we can construct the controller for our original system by calculating $(K, L, S\bar{M})$.

We can also find the same result by using the provided Matlab function.

```
1 U = [1 -1 2 0 ; -1 1 -2 0];  
2 Y = [2 6 4 2];  
3 X = [1 3 2 1 -4 ; 0 3 -5 -6 -2];  
4  
5 [bool] = isInformDynamicMeasurementFeedback(X, U, Y);
```

Which will return [1].

12 State estimation

In this section we will consider if and how we can reconstruct the state data only based on the input and output data. We will use this reconstructed state data to construct a controller for stabilisability by dynamic measurement feedback. We will also show how this theory can be applied in an example.

In this section we will be considering discrete time systems of the following form.

$$x(t+1) = A_s x(t) + B_s u(t) \quad (10a)$$

$$y(t) = C_s x(t) + D_s u(t) \quad (10b)$$

We will assume that our data does not contain any state space data, i.e. $\mathcal{D} = \{U_-, Y_-\}$. We will also assume our data to be sampled from a single interval as well as that the dimension of the state space is known. If the dimension of the state space is not known beforehand then it might be found using subspace identification methods. However, we will not be going into details about subspace identification methods in this paper.

Using these assumptions we will define the set of systems corresponding to the input output data as all systems (A, B, C, D) such that there exists a state sequence X for which the system equations hold.

$$\Sigma_{i/o} = \left\{ (A, B, C, D) \mid \exists X \in \mathbb{R}^{n \times (T+1)} \text{ s.t. } \begin{bmatrix} X_+ \\ Y_- \end{bmatrix} = \begin{bmatrix} A_s & B_s \\ C_s & D_s \end{bmatrix} \begin{bmatrix} X_- \\ U_- \end{bmatrix} \text{ holds} \right\}$$

Using this set we can define informativity for stabilisation by dynamic measurement feedback for input output data as follows.

Def: [waarde2019data]

We say that the data (U_-, Y_-) is informative for stabilisation by dynamic measurement feedback if there exist matrices K , L and M such that all systems described by the data are stabilised using the given controller. I.e. $\Sigma_{i/o} \subseteq \Sigma_{K,L,M}$.

We will start by noting that analogous to the input state output case we are able to reduce our input data to be of full rank. In other words, using the following lemma we are able to assume that \bar{U}_- has full row rank of dimension k .

Lem: [waarde2019data]

Consider the data (U_-, Y_-) and the corresponding set $\Sigma_{i/o}$. Let S be a matrix of full column rank such that $U_- = S\bar{U}_-$ with \bar{U}_- a matrix of full row rank.

Then the data (U_-, Y_-) is informative for stabilisation by dynamic measurement feedback if and only if the data (\bar{U}_-, Y_-) is informative for stabilisation by dynamic measurement feedback.

Before we look at the state sequence reconstruction we will first need to introduce some notation. We will start by defining a *Hankel matrix of depth l* . A Hankel matrix is defined on a signal $f(0), \dots, f(T-1)$, in our case this would be the column entries of the U_- matrix or the Y_- matrix. Using this signal we can define the Hankel matrix of depth l given a signal f as follows.

$$\mathcal{H}_l(f) = \begin{bmatrix} f(0) & f(1) & \dots & f(T-l) \\ f(1) & f(2) & \dots & f(T-l+1) \\ \vdots & \vdots & & \vdots \\ f(l-1) & f(l) & \dots & f(T-1) \end{bmatrix}$$

Provided our input and output data, take k such that $2k < T$. We can construct the following block Hankel matrices, $\mathcal{H}_{2k}(u)$ and $\mathcal{H}_{2k}(y)$. We can partition these matrices in 'past' and 'future' data.

$$\mathcal{H}_{2k}(u) = \begin{bmatrix} U_p \\ U_f \end{bmatrix} \quad \mathcal{H}_{2k}(y) = \begin{bmatrix} Y_p \\ Y_f \end{bmatrix}$$

Where each of the partitions have k block rows. Now we will define $x(0), \dots, x(T)$ to be a state sequence compatible with the data (U_-, Y_-) . We will partition the state sequence as follows.

$$X_p = \begin{bmatrix} x(0) \dots x(T-2k) \end{bmatrix}$$

$$X_f = \begin{bmatrix} x(k) \dots x(T-k) \end{bmatrix}$$

Finally we will define $rs(M)$ to denote the row space of a given matrix M .

Thr: [waarde2019data]

Consider a system of the form (10) and assume it is minimal. Let n be the dimension of the state space and the input/output data (U_-, Y_-) be sampled from a single interval. Assume that k is such that $n < k < \frac{1}{2}T$. If

$$\text{rank} \begin{bmatrix} \mathcal{H}_{2k}(u) \\ \mathcal{H}_{2k}(y) \end{bmatrix} = 2km + n \quad (12)$$

then

$$rs(X_f) = rs \left(\begin{bmatrix} U_p \\ Y_p \end{bmatrix} \right) \cap rs \left(\begin{bmatrix} U_f \\ Y_f \end{bmatrix} \right)$$

and the row space of X_f is of dimension n .

Using this theorem we are able to construct a valid state sequence up to similarity transform S . I.e. we can find $\hat{X} = SX_f$ for some unknown invertible matrix S . Using this state sequence we can construct the following input/state/output sequences.

$$\hat{U}_- = \begin{bmatrix} u(k) & u(k+1) & \dots & u(T-k-1) \end{bmatrix}$$

$$\hat{Y}_- = \begin{bmatrix} y(k) & y(k+1) & \dots & y(T-k-1) \end{bmatrix}$$

$$\hat{X} = S \begin{bmatrix} x(k) & x(k+1) & \dots & x(T-k) \end{bmatrix}$$

Using this newly constructed input/state/output data we are able to extend our previous theorem on stabilisation by dynamic measurement feedback.

Cor: [waarde2019data]

Consider a system of the form (10) and assume it is minimal. Let the input/output data be given by (U_-, Y_-) and measured on a single interval. Assume that k is such that $n < k < \frac{1}{2}T$. Then the data (U_-, Y_-) is informative for stabilisation by dynamic measurement feedback if the following 2 conditions hold:

1. The rank condition (12) holds.
2. The data $(\hat{U}_-, \hat{X}, \hat{Y}_-)$ as defined in (13) is informative for stabilisation by dynamic measurement feedback.

Moreover, if these conditions are satisfied, a stabilising controller (K, L, M) such that $\Sigma_{i/o} \subseteq \Sigma_{K,L,M}$ can be found by applying the methods for stabilisation by dynamic measurement feedback on the data $(\hat{U}_-, \hat{X}, \hat{Y}_-)$. Note that the conditions provided in the corollary above are sufficient but not necessary for informativity for stabilisation by dynamic measurement feedback.

Before we look at an example we first make some notes on how to actually compute the state sequence from the intersection. For this we will use the following theorem.

Thr: [bookMoonen]

We take the following matrices based on the previous theorems.

$$H_1 = \begin{bmatrix} U_p \\ Y_p \end{bmatrix} \quad H_2 = \begin{bmatrix} U_f \\ Y_f \end{bmatrix} \quad H = \begin{bmatrix} H_1 \\ H_2 \end{bmatrix}$$

We take the following singular value decomposition:

$$H = \begin{bmatrix} U_{11} & U_{12} \\ U_{21} & U_{22} \end{bmatrix} \begin{bmatrix} S_{11} & 0 \\ 0 & 0 \end{bmatrix} v^t$$

Then the state sequence X_f can be calculated as:

$$X_f = U_q^\top U_{12}^\top H_1$$

where U_q is defined through the singular value decomposition of $U_{12}^\top U_{11} S_{11}$.

$$U_{12}^\top U_{11} S_{11} = \begin{bmatrix} U_q & U_q^\perp \end{bmatrix} \begin{bmatrix} S_q & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_q^\top \\ V_q^{\top\perp} \end{bmatrix}$$

This theorem gives us a relatively easy to compute method for finding the state sequence of the data. The above described algorithm has been implemented in the following function.

12.1 Implementation

Syntax

```
[bool, X_hat, U_hat, Y_hat] = isInformStateIdentification(U, Y, n)
```

Description

`[bool, X_hat, U_hat, Y_hat] = isInformStateIdentification(U, Y, n)`: Returns if the input/output data is informative for state identification, if the data is informative, then the function will return a state sequence up to similarity that can be used for finding a dynamic measurement controller using `isInformDynamicMeasurementFeedback(X_hat, U_hat, Y_hat)`.

Input arguments

U: Input data matrix of dimension $m \times T$ from a input/output data set.
Y: Output data matrix of dimension $p \times T$ from a input/output data set.
n: Dimension of the state space.

Output arguments

bool: (boolean) True if the data is informative for state space identification, false otherwise

X_hat: (matrix) A state sequence that can be used to find a dynamic measurement feedback controller.

U_hat: (matrix) A input sequence that can be used to find a dynamic measurement feedback controller.

Y_hat: (matrix) A output sequence that can be used to find a dynamic measurement feedback controller.

12.2 Examples

In this example we will consider the following system:

$$A = \begin{bmatrix} 0 & 1 \\ -1 & -1 \end{bmatrix} \quad B = \begin{bmatrix} 1 \\ -1 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix} \quad D = \begin{bmatrix} 0 \end{bmatrix}$$

We will use this system to generate the following data:

$$X = \begin{bmatrix} 0 & 2 & 2 & -2 & 2 & 0 & -4 & 3 & 1 & -3 & 4 & -1 & 7 & -11 \\ 0 & -1 & -2 & 1 & 0 & -2 & 4 & 1 & -4 & 2 & -1 & -3 & -6 & 4 \end{bmatrix}$$

$$U_- = \begin{bmatrix} 1 & 2 & 0 & 1 & 0 & -2 & -1 & 0 & 1 & 2 & 0 & 10 & -5 \end{bmatrix}$$

$$Y_- = \begin{bmatrix} 0 & 1 & 1 & -2 & 2 & 0 & -4 & 3 & 1 & -3 & 4 & -1 & 7 \end{bmatrix}$$

We will start by defining the following values:

$$\begin{aligned} n &= 2 && \text{Dimension of state space.} \\ m &= 1 && \text{Dimension of input space.} \\ l &= 2 && \text{Dimension of output space.} \\ T &= 13 && \text{Number of samples.} \end{aligned}$$

We will start by picking a value for k such that $n < k < \frac{1}{2}T$. We will pick $k = 3$. Now we will define the block Hankel matrices

$$\mathcal{H}_{2k}(u) = \begin{bmatrix} 1 & 2 & 0 & 1 & 0 & -2 & -1 & 0 \\ 2 & 0 & 1 & 0 & -2 & -1 & 0 & 1 \\ 0 & 1 & 0 & -2 & -1 & 0 & 1 & 2 \\ - & - & - & - & - & - & - & - \\ 1 & 0 & -2 & -1 & 0 & 1 & 2 & 0 \\ 0 & -2 & -1 & 0 & 1 & 2 & 0 & 10 \\ -2 & -1 & 0 & 1 & 2 & 0 & 10 & -5 \end{bmatrix} = \begin{bmatrix} U_p \\ U_f \end{bmatrix}$$

$$\mathcal{H}_{2k}(y) = \begin{bmatrix} 0 & 1 & 1 & -2 & 2 & 0 & -4 & 3 \\ 1 & 1 & -2 & 2 & 0 & -4 & 3 & 1 \\ 1 & -2 & 2 & 0 & -4 & 3 & 1 & -3 \\ - & - & - & - & - & - & - & - \\ -2 & 2 & 0 & -4 & 3 & 1 & -3 & 4 \\ 2 & 0 & -4 & 3 & 1 & -3 & 4 & -1 \\ 0 & -4 & 3 & 1 & -3 & 4 & -1 & 7 \end{bmatrix} = \begin{bmatrix} Y_p \\ Y_f \end{bmatrix}$$

Using MatLab we can verify that the rank condition is satisfied.

$$\text{rank} \begin{bmatrix} \mathcal{H}_{2k}(u) \\ \mathcal{H}_{2k}(y) \end{bmatrix} = 8$$

Now that all preliminary conditions are checked we can start reconstructing the state sequence. We will start by calculating the singular value decomposition of H .

$$H = \begin{bmatrix} Y_p \\ U_p \\ Y_f \\ U_f \end{bmatrix} = \begin{bmatrix} U_{11} & U_{12} \\ U_{21} & U_{22} \end{bmatrix} \begin{bmatrix} S_{11} & 0 \\ 0 & 0 \end{bmatrix} v^t$$

Where U_{11} has size 6×8 and U_{12} has size 6×4 .

Using these matrices we can compute the singular value decomposition of the following matrix to find U_q .

$$U_{12}^\top U_{11} S_{11} = \begin{bmatrix} U_q & U_q^\perp \end{bmatrix} \begin{bmatrix} S_q & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_q^\top \\ V_q^{\top\perp} \end{bmatrix}$$

Note that U_q is of size 4×2 .

Now that we have found all the components to reconstruct a state sequence we get the following.

$$\hat{X}_f = U_q^\top U_{12}^\top H_1 = \begin{bmatrix} 1.2000 & -1.4047 & 0.4096 & 1.9903 & -2.3119 & 0.1168 & 1.6975 & -2.6047 \\ 0.2141 & 0.4000 & -1.2282 & 1.6563 & 1.2141 & -2.2564 & 0.6282 & 0.1859 \end{bmatrix}$$

Recall that the sequence X_f was defined as $X_f = [x(k) \dots x(T-k)]$. As we can see this state sequence differs from our original state sequence.

$$X_f = \begin{bmatrix} -2 & 2 & 0 & -4 & 3 & 1 & -3 & 4 \\ 1 & 0 & -2 & 4 & 1 & -4 & 2 & -1 \end{bmatrix}$$

To conclude that the 'real' X_f is indeed the same up to similarity transformation as \hat{X}_f we will calculate the rank of the following matrix to conclude that they are linearly dependent.

$$\text{rank} \left(\begin{bmatrix} \hat{X}_f \\ X_f \end{bmatrix} \right) = \text{rank} (\hat{X}_f) = \text{rank} (X_f) = 2$$

We will now finish by defining a correspond \hat{U}_- and \hat{Y}_- for the found state sequence.

$$\begin{aligned} \hat{U}_- &= \begin{bmatrix} 1 & 0 & -2 & -1 & 0 & 1 & 2 & 0 \end{bmatrix} = \begin{bmatrix} u(k) & u(k+1) & \dots & u(T-k-1) \end{bmatrix} \\ \hat{Y}_- &= \begin{bmatrix} -2 & 2 & 0 & -4 & 3 & 1 & -3 & 4 \end{bmatrix} = \begin{bmatrix} y(k) & y(k+1) & \dots & y(T-k-1) \end{bmatrix} \end{aligned}$$

Using the provided matlab function `[bool, X_hat, U_hat, Y_hat] = isInformStateIdentification(U, Y, n)` we can get the same result.

```

1 U = [1  2  0  1  0 -2 -1  0  1  2  0 10 -5];
2 Y = [0  1  1 -2  2  0 -4  3  1 -3  4 -1  7];
3 [bool, X_hat, U_hat, Y_hat] = isInformStateIdentification(U, Y, 2)

```

We can now use this newly defined input/state/output data to find a controller for dynamic feedback stabilisation as described by the theorem. For this we will use the provided Matlab function discussed in the previous section.

```
1 U = [1  2  0  1  0 -2 -1  0  1  2  0 10 -5];  
2 Y = [0  1  1 -2  2  0 -4  3  1 -3  4 -1  7];  
3 [bool, X_hat, U_hat, Y_hat] = isInformStateIdentification(U, Y, 2);  
4  
5 [bool, K, L, M] = isInformDynamicMeasurementFeedback(X_bar, U_bar, Y_bar);
```

This will return the following controllers:

$$K = \begin{bmatrix} 1.5655 & 0.5221 \\ 1.3028 & 0.4145 \end{bmatrix} \quad L = \begin{bmatrix} 0.9512 \\ 0.0070 \end{bmatrix} \quad M = \begin{bmatrix} -1.2989 & -2.0616 \end{bmatrix}$$

Since we have access to the original system we can also verify that this is indeed a stabilising controller.

$$\text{eig} \left(\begin{bmatrix} A & BM \\ LC & K + LDM \end{bmatrix} \right) = \{0, 0, 0.5, 0.5\}$$

13 Noise and control

In this section we will discuss how we can define our data in the case where there is an unknown noise term acting on the system. We will formulate an assumption on our noise that we will use to represent the control problem as one of quadratic matrix inequalities. We will consider the classical S-lemma for quadratic function and look at a generalised matrix S-lemma that we can use to solve the control problem. Lastly we will consider how we can implement the necessary condition in a function for use in the actual stabilisation problems coming in the next sections.

13.1 Noise and its set of systems

Recall that we defined systems with bounded noise in the following way.

$$\mathbf{x}(t+1) = A_s \mathbf{x}(t) + B_s \mathbf{u}(t) + \mathbf{w}(t) \quad (7)$$

Where W_- was defined in the same way as U_- and Y_- , i.e. it is a matrix where each column represents a sample from the given variable. Using this we can rewrite (7) using the data matrices in the following way.

$$X_+ = A_s X_- + B_s U_- + W_- \quad (14)$$

Before we consider the set of systems with noise, we will first look at how we can reformulate our boundedness assumption on the noise as a quadratic matrix inequality.

[**waarde2020noisy**] Let the noise samples $w(0), w(1), \dots, w(T-1)$ be collected in the matrix W_- , satisfy the bound

$$\begin{bmatrix} I \\ W_-^\top \end{bmatrix}^\top \begin{bmatrix} \Phi_{11} & \Phi_{12} \\ \Phi_{12}^\top & \Phi_{22} \end{bmatrix} \begin{bmatrix} I \\ W_-^\top \end{bmatrix} \geq 0 \quad (15)$$

for known matrices $\Phi_{11} = \Phi_{11}^\top$, Φ_{12} and $\Phi_{22} = \Phi_{22}^\top < 0$.

This assumption gives us a . We can also look at the special case in which $\Phi_{12} = 0$ and $\Phi_{22} = -I$. In this case we are able to rewrite (15) to get the following.

$$W_- W_-^\top = \sum_{t=0}^{T-1} w(t)w(t)^\top \leq \Phi_{11} \quad (16)$$

In this case we can see that the noise needs to have a finite bound to conform to our assumption.

Now that we have introduced our notation, we are able to define the set of systems that are described by some input and state data given that there is noise.

$$\Sigma_{i/s/n} = \{(A, B) \mid (14) \text{ holds for some } W_- \text{ satisfying (15)}\} \quad (17)$$

We will now substitute (14) in (15) to get a single quadratic matrix inequality that can be used to define Σ .

$$\begin{bmatrix} I \\ A^\top \\ B^\top \end{bmatrix}^\top \begin{bmatrix} I & X_+ \\ 0 & -X_- \\ 0 & -U_- \end{bmatrix} \begin{bmatrix} \Phi_{11} & \Phi_{12} \\ \Phi_{12}^\top & \Phi_{22} \end{bmatrix} \begin{bmatrix} I & X_+ \\ 0 & -X_- \\ 0 & -U_- \end{bmatrix}^\top \begin{bmatrix} I \\ A^\top \\ B^\top \end{bmatrix} \geq 0 \quad (18)$$

Using this inequality we can redefine Σ as follows.

Lem: [**waarde2020noisy**]

We have that $\Sigma = \{(A, B) \mid (18) \text{ is satisfied}\}$.

bound
on
noise,
extend

13.2 Problem formulation

We will start by defining when data is informative for quadratic stabilisation. We will go into more detail about this type of control in the later section 14. For now we will use this definition to give an intuitive idea on how we end up at the final theory.

Def: Informative for quadratic stabilisation [waarde2020noisy]

The data (U_-, X) is called informative for quadratic stabilisation if there exists a feedback gain K and a Lyapunov matrix $P = P^\top > 0$ such that $P - (A + BK)P(A + BK)^\top > 0$ for all $(A, B) \in \Sigma_{i/s/n}$

We can expand the matrix inequality to get the following.

$$P - (A + BK)P(A + BK)^\top = \begin{bmatrix} I \\ A^\top \\ B^\top \end{bmatrix}^\top \begin{bmatrix} P & 0 & 0 \\ 0 & -P & -PK^\top \\ 0 & -KP & -KPK^\top \end{bmatrix} \begin{bmatrix} I \\ A^\top \\ B^\top \end{bmatrix} > 0 \quad (19)$$

For the sake of argument, let's pick P and K to be fixed. Then our control problem reduces to knowing when there is sufficient overlap in both solution sets of the quadratic matrix inequalities (19) and (18), or in other words, when does one quadratic matrix inequality imply another quadratic matrix inequality.

13.3 Classical S-lemma

We will start by recalling the S-lemma for quadratic functions. A function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is called quadratic if it can be written in the form

$$f(x) = \begin{bmatrix} 1 \\ x \end{bmatrix}^\top \begin{bmatrix} M_{11} & M_{12} \\ M_{12}^\top & M_{22} \end{bmatrix} \begin{bmatrix} 1 \\ x \end{bmatrix}$$

for some $M_{11} \in \mathbb{R}$, $M_{12} \in \mathbb{R}^{1 \times n}$ and $M_{22} \in \mathbb{R}^{n \times n}$.

Thr: S-lemma

Let $f, g : \mathbb{R}^n \rightarrow \mathbb{R}$ be quadratic functions. Suppose that there exists $\bar{x} \in \mathbb{R}^n$ such that $g(\bar{x}) > 0$. Then $f(x) \geq 0$ for all $x \in \mathbb{R}^n$ such that $g(x) \geq 0$ if and only if there exists a scalar $\alpha \geq 0$ such that

$$f(x) - \alpha g(x) \geq 0 \quad \forall x \in \mathbb{R}^n$$

However, using the previously mentioned definition of quadratic functions we are able to rewrite the S-lemma to matrix inequalities.

$$\begin{bmatrix} M_{11} & M_{12} \\ M_{12}^\top & M_{22} \end{bmatrix} - \alpha \begin{bmatrix} N_{11} & N_{12} \\ N_{12}^\top & N_{22} \end{bmatrix} \geq 0$$

In which $\alpha > 0$ is a scalar and N_{11}, N_{12} and N_{22} define the quadratic function g . Note that the assumption that there exists a $\bar{x} \in \mathbb{R}^n$ such that $g(\bar{x}) > 0$ is called the Slater condition.

13.4 Matrix S-lemma

Now that we have familiarised ourselves with the classical S-lemma, we will move on to the generalised matrix S-lemma as described in [waarde2020noisy].

Thr: Matrix S-lemma [waarde2020noisy]

Let $M, N \in \mathbb{R}^{(n+k) \times (n+k)}$ be symmetric matrices and assume that there exists some matrix $\bar{Z} \in \mathbb{R}^{n \times k}$ such that

$$\begin{bmatrix} I \\ \bar{Z} \end{bmatrix}^\top N \begin{bmatrix} I \\ \bar{Z} \end{bmatrix} > 0. \quad (20)$$

Then the following statements are equivalent:

1. $\begin{bmatrix} I \\ Z \end{bmatrix}^\top M \begin{bmatrix} I \\ Z \end{bmatrix} \geq 0$ for all $Z \in \mathbb{R}^{n \times k}$ with $\begin{bmatrix} I \\ Z \end{bmatrix}^\top N \begin{bmatrix} I \\ Z \end{bmatrix} \geq 0$.
2. $\begin{bmatrix} I \\ Z \end{bmatrix}^\top M \begin{bmatrix} I \\ Z \end{bmatrix} \geq 0$ for all $Z \in \mathbb{R}^{n \times k}$ with $\begin{bmatrix} I \\ Z \end{bmatrix}^\top N \begin{bmatrix} I \\ Z \end{bmatrix} > 0$.
3. There exists a scalar $\alpha \geq 0$ such that $M - \alpha N \geq 0$.

Similar to the classical S-lemma, the assumption (20) is called the generalised Slater condition.

Using this matrix S-lemma we are able to get find a solution to our quadratic matrix inequality problem. We will go into more detail about the details for each type of control in their respective sections. But before we look at the specific cases we will first consider how we can verify that the generalised Slater condition holds for a given matrix N . For this we will consider the following theorem which gives us an easy to compute condition for verifying the Slater condition.

Thr:

Let $N \in \mathbb{R}^{(k+n) \times (k+n)}$ be symmetric. Then the following are equivalent.

1. There exists a $Z \in \mathbb{R}^{n \times k}$ such that $\begin{bmatrix} I_k \\ \bar{Z} \end{bmatrix}^\top N \begin{bmatrix} I_k \\ \bar{Z} \end{bmatrix} > 0$.
2. N has at least k positive eigenvalues.

Proof '1 \Rightarrow 2':

fdsa

Proof '1 \Leftarrow 2':

fdsa

13.5 Implementation

Syntax

```
[bool] = testSlater(X, U, Phi)
```

Description

`[bool] = testSlater(X, U, Phi)`: Returns if the Slater condition holds for a matrix N constructed from the data (X, U) and the noise matrix Φ .

Input arguments

X: State data matrix of dimension $n \times T + 1$ from a input/state data set.

U: Input data matrix of dimension $m \times T$ from a input/state data set.

Phi: Noise matrix as in (15).

Output arguments

bool: (boolean) True if the Slater condition holds, false otherwise.

14 Quadratic stabilisation

Recall from the previous section that we defined the data to be informative for quadratic stabilisation in the following way.

Def: Informative for quadratic stabilisation [waarde2020noisy]

The data (U_-, X) is called informative for quadratic stabilisation if there exists a feedback gain K and a Lyapunov matrix $P = P^\top > 0$ such that $P - (A + BK)P(A + BK)^\top > 0$ for all $(A, B) \in \Sigma_{i/s/n}$

We will also recall that this matrix inequality can be rewritten as:

$$P - (A + BK)P(A + BK)^\top = \begin{bmatrix} I \\ A^\top \\ B^\top \end{bmatrix}^\top \begin{bmatrix} P & 0 & 0 \\ 0 & -P & -PK^\top \\ 0 & -KP & -KPK^\top \end{bmatrix} \begin{bmatrix} I \\ A^\top \\ B^\top \end{bmatrix} > 0 \quad (19)$$

As alluded to earlier, we want to find a condition in which case (19) holds as well as:

$$\begin{bmatrix} I \\ A^\top \\ B^\top \end{bmatrix}^\top \begin{bmatrix} I & X_+ \\ 0 & -X_- \\ 0 & -U_- \end{bmatrix} \begin{bmatrix} \Phi_{11} & \Phi_{12} \\ \Phi_{12}^\top & \Phi_{22} \end{bmatrix} \begin{bmatrix} I & X_+ \\ 0 & -X_- \\ 0 & -U_- \end{bmatrix}^\top \begin{bmatrix} I \\ A^\top \\ B^\top \end{bmatrix} \geq 0 \quad (18)$$

We will once again note that in order to find a solution, we want to check if (19) hold for all systems such that (18) holds. To do this, we will partition our matrices in the following way.

$$M = \left[\begin{array}{c|c} M_{11} & M_{12} \\ \hline M_{12}^\top & M_{22} \end{array} \right] := \left[\begin{array}{c|cc} P & 0 & 0 \\ \hline 0 & -P & -PK^\top \\ 0 & -KP & -KPK^\top \end{array} \right]$$

$$N = \left[\begin{array}{c|c} N_{11} & N_{12} \\ \hline N_{12}^\top & N_{22} \end{array} \right] := \begin{bmatrix} I & X_+ \\ 0 & -X_- \\ 0 & -U_- \end{bmatrix} \begin{bmatrix} \Phi_{11} & \Phi_{12} \\ \Phi_{12}^\top & \Phi_{22} \end{bmatrix} \begin{bmatrix} I & X_+ \\ 0 & -X_- \\ 0 & -U_- \end{bmatrix}^\top \quad (21)$$

Thr: [waarde2020noisy]

Assume that the generalised Slater condition (20) holds for N in (21) and some $\bar{Z} \in \mathbb{R}^{(n+m \times n)}$. Then the data (U_-, X) is informative for quadratic stabilisation if and only if there exists an $n \times n$ matrix $P = P^\top > 0$, an $L \in \mathbb{R}^{m \times n}$ and scalars $\alpha \geq 0$ and $\beta > 0$ satisfying:

$$\begin{bmatrix} P - \beta I & 0 & 0 & 0 \\ 0 & -P & -L^\top & 0 \\ 0 & -L & 0 & L \\ 0 & 0 & L^\top & P \end{bmatrix} - \alpha \begin{bmatrix} I & X_+ \\ 0 & -X_- \\ 0 & -U_- \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \Phi_{11} & \Phi_{12} \\ \Phi_{12}^\top & \Phi_{22} \end{bmatrix} \begin{bmatrix} I & X_+ \\ 0 & -X_- \\ 0 & -U_- \\ 0 & 0 \end{bmatrix}^\top \geq 0 \quad (22)$$

Moreover, if P and L satisfy this matrix inequality, then $K = LP^{-1}$ is a stabilising feedback gain for all $(A, B) \in \Sigma$.

Figure
out
how to
present
this

14.1 Implementation

Syntax

```
[bool, K, diagnostics, info] = isInformQuadraticStabilisation(X, U, Phi)
[bool, K, diagnostics, info] = isInformQuadraticStabilisation(X, U, Phi, tolerance)
[bool, K, diagnostics, info] = isInformQuadraticStabilisation(X, U, Phi, tolerance,
options)
```

Description

```
[bool, K, diagnostics, info] = isInformQuadraticStabilisation(X, U, Phi): .
[bool, K, diagnostics, info] = isInformQuadraticStabilisation(X, U, Phi, tolerance):
.
[bool, K, diagnostics, info] = isInformQuadraticStabilisation(X, U, Phi, tolerance,
options): .
```

Input arguments

X: State data matrix of dimension $n \times T + 1$ from a input/state data set.

U: Input data matrix of dimension $m \times T$ from a input/state data set.

Phi: Noise matrix as in (15).

tolerance: Tolerance used for determining when a value is zero up to machine precision. Default value is $1e-14$.

options: sdpsettings used with the Yalmip solver.

Output arguments

bool: (boolean) True if the data is informative for quadratic stabilisation, false otherwise. If false then the **info** variable can be check to find which condition failed.

K: (matrix) If the data is informative, it contains a stabilising controller **K** for closed loop control $A+BK$, empty otherwise.

diagnostics: (struct) Diagnostics from the Yalmip `optimize()` function.

info: (int) Diagnostic variable use to identify which conditions (if any) failed. The verification is done on the solution obtained from Yalmip. For information about the type of error use the `help` command in Matlab

14.2 Example

In this example we will consider the following unstable system.

$$x(t+1) = 2x(t) - \frac{1}{2}u(t) + w(t)$$

We will assume our noise to be picked uniformly from the interval $[-0.1, 0.1]$. We will start by generating a random noise sequence and a random input sequence which we will use to construct

the following data.

$$\begin{aligned} W_- &= \begin{bmatrix} 0.0706 & 0.0244 & -0.0298 & 0.0026 & -0.0196 \end{bmatrix} \\ U_- &= \begin{bmatrix} -0.1655 & -0.9007 & 0.8054 & 0.8896 & -0.0183 \end{bmatrix} \\ X &= \begin{bmatrix} 0 & 0.1533 & 0.6281 & 0.1956 & -0.2466 & -0.2571 \end{bmatrix} \end{aligned}$$

For our noise matrix Φ we will use that if we pick $\Phi_{12} = \bar{0} = [0 \dots 0]$ and $\Phi_{22} = -I$ then we know that $W_- W_-^\top \leq \Phi_{11}$. Since we know that our data is sampled from a uniform distribution with a maximum value of 0.1 we know that $W_- W_-^\top \leq 0.1^2 T$ where T is the total number of samples considered. Since we are considering 5 sample we will pick our noise matrix to be:

$$\Phi = \begin{bmatrix} 0.05 & \bar{0} \\ \bar{0}^\top & -I_9 \end{bmatrix}$$

Using some computation we can verify that this choice of Φ is indeed a valid one for the data.

$$\begin{bmatrix} I \\ W_-^\top \end{bmatrix}^\top \begin{bmatrix} \Phi_{11} & \Phi_{12} \\ \Phi_{12}^\top & \Phi_{22} \end{bmatrix} \begin{bmatrix} I \\ W_-^\top \end{bmatrix} = 0.0431 \geq 0 \quad (23)$$

Before we attempt to solve the matrix inequality, we need to check if the generalised Slater condition holds. As we have seen in the previous section, we only have to verify that N has at least 1 positive eigenvalue.

$$\begin{aligned} N &= \begin{bmatrix} I & X_+ \\ 0 & -X_- \\ 0 & -U_- \\ 0 & \bar{0} \end{bmatrix} \begin{bmatrix} 0.09 & \bar{0} \\ \bar{0}^\top & -I_9 \end{bmatrix} \begin{bmatrix} I & X_+ \\ 0 & -X_- \\ 0 & -U_- \\ 0 & \bar{0} \end{bmatrix}^\top = \begin{bmatrix} -0.5332 & 0.2343 & -0.6482 & 0 \\ 0.2343 & -0.5171 & -0.5463 & 0 \\ -0.6482 & -0.5463 & -2.2790 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \\ \sigma(N) &= \{ -2.5921 \quad -0.7571 \quad 0 \quad 0.02 \} \end{aligned}$$

Now that we have our data prepared, we will attempt to find a $P > 0$, L , $\alpha \geq 0$ and $\beta > 0$ such that (22) holds.

$$\begin{bmatrix} P - \beta I & 0 & 0 & 0 \\ 0 & -P & -L^\top & 0 \\ 0 & -L & 0 & L \\ 0 & 0 & L^\top & P \end{bmatrix} - \alpha \begin{bmatrix} I & X_+ \\ 0 & -X_- \\ 0 & -U_- \\ 0 & \bar{0} \end{bmatrix} \begin{bmatrix} 0.09 & \bar{0} \\ \bar{0}^\top & -I_9 \end{bmatrix} \begin{bmatrix} I & X_+ \\ 0 & -X_- \\ 0 & -U_- \\ 0 & \bar{0} \end{bmatrix}^\top \geq 0$$

Using computational tools such as Yalmip we can find that the following values form a valid solution to the matrix inequality.

$$P = \frac{3}{10} \quad L = \frac{1}{2} \quad \alpha = \frac{9}{10} \quad \beta = \frac{1}{10}$$

Using these values we are able to construct a stabilising controller $K = LP^{-1} = 1\frac{2}{3}$. As we can see the closed loop system is indeed stable.

$$A + BK = \frac{1}{6}$$

We are also able to find a stabilising feedback gain using the provided Matlab function.

```
1 U = [-0.1655 -0.9007 0.8054 0.8896 -0.0183];  
2 X = [ 0          0.1533 0.6281 0.1956 -0.2466 -0.2571];  
3 Phi = [0.05 zeros(1,5) ; zeros(5,1) -eye(5)];  
4 [bool, K] = isInformQuadraticStabilisation(X, U, Phi)
```

Which will return: [1, 1.6977]. Note that this is a different but still valid controller.

15 \mathcal{H}_2 Control

Def: Informative for \mathcal{H}_2 control [waarde2020noisy]

the data (U_-, X) is informative for \mathcal{H}_2 control with performance γ if there exists matrices $p = p^\top > 0$ and K such that:

$$P > (A + BK)^\top P(A + BK) + (C + DK)^\top (C + DK) \quad \text{trace}(P) < \gamma^2$$

holds for all $(A, B) \in \Sigma_{i/s/n}$.

Thr: [waarde2020noisy]

Assume that the generalised Slater condition (20) holds for N in (21) and some $\bar{Z} \in \mathbb{R}^{(n+m \times n)}$. Then the data (U_-, X) is informative for \mathcal{H}_2 control with performance γ if and only if there exists matrices $Y = Y^\top > 0$, $Z = Z^\top$ and L , and scalars $\alpha \geq 0$ and $\beta > 0$ satisfying:

$$\begin{bmatrix} Y - \beta I & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & Y & 0 \\ 0 & 0 & 0 & L & 0 \\ 0 & Y & L^\top & Y & (CY + DL)^\top \\ 0 & 0 & 0 & CY + DL & I \end{bmatrix} - \alpha \begin{bmatrix} I & X_+ \\ 0 & -X_- \\ 0 & -U_- \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \Phi_{11} & \Phi_{12} \\ \Phi_{12}^\top & \Phi_{22} \end{bmatrix} \begin{bmatrix} I & X_+ \\ 0 & -X_- \\ 0 & -U_- \\ 0 & 0 \\ 0 & 0 \end{bmatrix}^\top \geq 0$$

$$\begin{bmatrix} Y & (CY + DL)^\top \\ CY + DL & I \end{bmatrix} > 0 \quad \begin{bmatrix} Z & I \\ I & Y \end{bmatrix} \geq 0 \quad \text{trace}(Z) < \gamma^2$$

Moreover, if Y and L satisfy the above conditions, then $K = LY^{-1}$ is such that $A + BK$ is stable and $\|G(z)\|_{\mathcal{H}_2} < \gamma$ for all $(A, B) \in \Sigma$.

Explain
 $G(z)$

15.1 Implementation

Syntax

```
[bool, K, diagnostics, gamma, info] = isInformH2(X, U, Phi, C, D)
[bool, K, diagnostics, gamma, info] = isInformH2(X, U, Phi, C, D, tolerance)
[bool, K, diagnostics, gamma, info] = isInformH2(X, U, Phi, C, D, tolerance, options)
```

Description

```
[bool, K, diagnostics, gamma, info] = isInformH2(X, U, Phi, C, D): .
[bool, K, diagnostics, gamma, info] = isInformH2(X, U, Phi, C, D, tolerance): .
[bool, K, diagnostics, gamma, info] = isInformH2(X, U, Phi, C, D, tolerance, options): .
```

Input arguments

X: State data matrix of dimension $n \times T + 1$ from a input/state data set.

U: Input data matrix of dimension $m \times T$ from a input/state data set.

Phi: Noise matrix as in (15).

C: State performance matrix.

D: State performance matrix.

tolerance: Tolerance used for determining when a value is zero up to machine precision. Default value is `1e-14`.

options: `sdpsettings` used with the Yalmip solver.

Output arguments

bool: (boolean) True if the data is informative for quadratic stabilisation, false otherwise. If false then the **info** variable can be checked to find which condition failed.

K: (matrix) If the data is informative, it contains a stabilising controller **K** for closed loop control **A+BK**, empty otherwise.

diagnostics: (struct) Diagnostics from the Yalmip `optimize()` function.

gamma: (double) Value of gamma as found by the solver.

info: (int) Diagnostic variable used to identify which conditions (if any) failed. The verification is done on the solution obtained from Yalmip. For information about the type of error use the **help** command in Matlab

15.2 Example

In this example we will consider the same data that we used in the quadratic stabilisation example.

$$x(t+1) = 2x(t) - \frac{1}{2}u(t) + w(t)W_- = \begin{bmatrix} 0.0706 & 0.0244 & -0.0298 & 0.0026 & -0.0196 \\ 0.0706 & 0.0244 & -0.0298 & 0.0026 & -0.0196 \\ 0.0706 & 0.0244 & -0.0298 & 0.0026 & -0.0196 \\ 0.0706 & 0.0244 & -0.0298 & 0.0026 & -0.0196 \\ 0.0706 & 0.0244 & -0.0298 & 0.0026 & -0.0196 \end{bmatrix}$$
$$U_- = \begin{bmatrix} -0.1655 & -0.9007 & 0.8054 & 0.8896 & -0.0183 \end{bmatrix}$$
$$X = \begin{bmatrix} 0 & 0.1533 & 0.6281 & 0.1956 & -0.2466 & -0.2571 \end{bmatrix}$$

We are also able to find a stabilising feedback gain using the provided Matlab function.

```
1 C = 1;
2 D = 0;
3 U = [-0.1655 -0.9007 0.8054 0.8896 -0.0183];
4 X = [ 0          0.1533 0.6281 0.1956 -0.2466 -0.2571];
5 Phi = [0.05 zeros(1,5) ; zeros(5,1) -eye(5)];
6 [bool, K] = isInformH2(X, U, Phi, C, D)
```

16 H_∞ Control

Def: Informative for \mathcal{H}_∞ control [waarde2020noisy]

The data (U_-, X) is informative for \mathcal{H}_∞ control with performance γ if there exists matrices $P = P^\top > 0$ and K such that:

$$P - (A + BK)^\top (P^{-1} - \frac{1}{\gamma^2} I)^{-1} (A + BK) - (C + DK)^\top (C + DK) > 0$$

$$P^{-1} - \frac{1}{\gamma^2} I > 0$$

Holds for all $(A, B) \in \Sigma$.

Thr: [waarde2020noisy]

Assume that the generalised Slater condition (20) holds for N in (21) and some $\bar{Z} \in \mathbb{R}^{(n+m \times n)}$. Then the data (U_-, X) is informative for \mathcal{H}_∞ control with performance γ if and only if there exists matrices $Y = Y^\top > 0$ and L , and scalars $\alpha \geq 0$ and $\beta > 0$ satisfying:

$$\begin{bmatrix} Y - \beta I & 0 & 0 & 0 & (CY + DL)^\top \\ 0 & 0 & 0 & Y & 0 \\ 0 & 0 & 0 & L & 0 \\ 0 & Y & L^\top & Y - \frac{1}{\gamma^2} I & 0 \\ CY + DL & 0 & 0 & 0 & I \end{bmatrix} - \alpha \begin{bmatrix} I & X_+ \\ 0 & -X_- \\ 0 & -U_- \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \Phi_{11} & \Phi_{12} \\ \Phi_{12}^\top & \Phi_{22} \end{bmatrix} \begin{bmatrix} I & X_+ \\ 0 & -X_- \\ 0 & -U_- \\ 0 & 0 \\ 0 & 0 \end{bmatrix}^\top \geq 0$$

$$Y - \frac{1}{\gamma^2} I > 0$$

Moreover, if Y and L satisfy the above inequalities, then $K = LY^{-1}$ is such that $A + BK$ is stable and $\|G(z)\|_{\mathcal{H}_\infty} < \gamma$ for all $(A, B) \in \Sigma$.

Explain
G(z)

16.1 Implementation

Syntax

```
[bool, K, diagnostics, gamma, info] = isInformHInf(X, U, Phi, C, D)
[bool, K, diagnostics, gamma, info] = isInformHInf(X, U, Phi, C, D, tolerance)
[bool, K, diagnostics, gamma, info] = isInformHInf(X, U, Phi, C, D, tolerance, options)
```

Description

a

```
[[bool, K, diagnostics, gamma, info] = isInformHInf(X, U, Phi, C, D): .
[bool, K, diagnostics, gamma, info] = isInformHInf(X, U, Phi, C, D, tolerance): .
[bool, K, diagnostics, gamma, info] = isInformHInf(X, U, Phi, C, D, tolerance, options):
.
```

Input arguments

X: State data matrix of dimension $n \times T + 1$ from a input/state data set.

U: Input data matrix of dimension $m \times T$ from a input/state data set.

Phi: Noise matrix as in (15).

C: State performance matrix.

D: State performance matrix.

tolerance: Tolerance used for determining when a value is zero up to machine precision. Default value is $1e-14$.

options: sdpsettings used with the Yalmip solver.

Output arguments

bool: (boolean) True if the data is informative for quadratic stabilisation, false otherwise. If false then the **info** variable can be checked to find which condition failed.

K: (matrix) If the data is informative, it contains a stabilising controller K for closed loop control $A+BK$, empty otherwise.

diagnostics: (struct) Diagnostics from the Yalmip `optimize()` function.

gamma: (double) Value of γ as found by the solver.

info: (int) Diagnostic variable used to identify which conditions (if any) failed. The verification is done on the solution obtained from Yalmip. For information about the type of error use the `help` command in Matlab

16.2 Example

17 How to use the toolbox with continuous time systems

18 How to use the toolbox for linearising models

19 Improving controller performance using H_2

20 Reference section title

Methods
work
only
some-
times
due to
limita-
tion in
Yalmip
/ other
solvers

Further
re-
search:
Condi-
tions
for pole
place-
ment
(similar
to dead-
beat
control),

Fix S in
dynamic
mea-
sure-
ment
feedback