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 introduce ourselves with the notation that will be used in the paper to indicate data and systems.

3.1 Informativity

Let Σ be the set of discrete time models with a given state space dimension n and a given input space dimension m. Then if we have state/input data $\mathcal{D}$ of this form, then we know that the system $\mathcal{S}$ that produced this data is contained in the set Σ. However, using Σ is not very useful unless we reduce the set to be more manageable, we can do this by using our knowledge of the true system, namely the data. We will only consider systems that are able to produce the data, we will call this set $\Sigma_{\mathcal{D}} \subseteq \Sigma$.

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. $\Sigma_{(U_-,X)} = \{\mathcal{S}\}$ 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 we will see this in an example where the data describes infinity many systems.

add reference to example with no identification and state feedback

We will also focus on 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:

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

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 $\mathcal{K}$ such that $\Sigma_{\mathcal{D}} \subseteq \Sigma_{\mathcal{P}(\mathcal{K})}$.

3.2 Data

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

Let n be the dimension of the state space and m be the dimension of the input space. Assume that both are known. Then we know that all systems contained in Σ are of the form:

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

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 assume our data is generated by the 'true' system, we will denote this system as (A_s, B_s). We will now measure the input and state data on q time intervals $\{0, 1, \ldots, T_i\}$ where $i \in \{1, 2, \ldots, q\}$. We denote the data collected on one of these intervals as follows:

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

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

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

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.

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

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:

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

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:

$$\mathbf{x}(t+1) = A\mathbf{x}(t) + B\mathbf{u}(t) \tag{2}$$

$$\mathbf{y}(t+1) = C\mathbf{x}(t) + D\mathbf{u}(t) \tag{3}$$

Let us define Y_- in the following way:

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

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

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

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 (1) 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:

$$\Sigma_{(U_-,X)} = \left\{ (A,B) \mid X_+ = \begin{bmatrix} A & B \end{bmatrix} \begin{bmatrix} X_- \\ U_- \end{bmatrix} \right\} \tag{5}$$

Where $\begin{bmatrix} X_- \\ U_- \end{bmatrix}$ is $(n+m) \times T$. Suppose that the *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 (5) 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

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

Furthermore, if the data is full rank, there exists an 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 (2):

Recall the set of systems described by the data is given by (4):

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

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 Examples using implementation

The algorithm above is implemented in the following functions:

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.

4.3.1 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:

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

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 where to consider the same data with 1 additional data point then the data would be informative for system identification:

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

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

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

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

5 Deadbeat control

In this section we will see how we can find a deadbeat controller using only our data. We will first discusse 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 $\Sigma_{i/s} \subseteq \Sigma_K^{nil}$.

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 only have the zero eigenvalue.

5.1 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.

First lets 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 lets 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 an 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 '$\Leftarrow$':

add proof

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.

5.2 Algorithm deadbeat control

Using the mathematics above we can construct an algorithm for checking if the data is informative for deadbeat control and what the corresponding controller is.

```
provide X_, X+ and U
if X_ has full row rank
    if X_ is square
        if X+ * inv(X_) is nilpotent
            The data is informative for deadbeat control
            K = U_ * inv(X_)
    else
        F : spans rowspace of X_ (pseudo inverse of X_ -> pinv(X_))
        G : spans the nullspace of X_ (null(X_))
        if (X+ F, X+ G) is controllable
            H = poleplacement(X+ F, X+ G, [0 ... 0])
            The data is informative for deadbeat control
            K = U_ * (F + G H)
The data is not informative for deadbeat control
```

However, due to limitation in Matlabs 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. 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 dinmensional input. To do this we will use the proof of the sufficientcy in theroem 3.29 as well as lemma 3.31 from [**bookTrentelman**].

5.3 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

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

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 controlability 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 controlable. 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 apropriate eigenvalues. From this we construct the controller $F = F_0 + u_0f$ that places the eigenvalues in the chosen positions.

5.4 Algorithm extending pole placement

We will now rewrite this into pseudo code:

```
Provide A, B and poles
n : dimension of state space
m : dimension of input space
construct linear independent x_1, ... ,x_n from u_0, ... , u_n-1
```

```
pick an arbitrary u_n
F0 := [u1 u2 ... un] * inv([x1 x2 ... xn])
b  := B * u0
f  := acker(A + B F0, b, poles)
K  := F0 + u_0 f
```

5.5 Examples using implementation

The algorithm above is implemented in the following functions:

Syntax

`[bool, K] = isInformDeadbeatControl(X, U)`

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`.

Input arguments

`X`: State data matrix of dimension $n \times T + 1$.
`U`: Input data matrix of dimension $m \times T$.

Output arguments

`bool`: (boolean) True if the data is informative for deadbeat control, false otherwise
`K`: (matrix) If the data is informative, it contains a stabilising controller `K` for closed loop control `A+BK`, empty otherwise.

5.5.1 Examples