mathematics

Article

Generalized Data–Driven Predictive Control: Merging Subspace and Hankel Predictors

M. Lazar * and P. C. N. Verheijen

Control Systems Group, Eindhoven University of Technology, 5612 AZ Eindhoven, The Netherlands; p.c.n.verheijen@tue.nl
* Correspondence: m.lazar@tue.nl

Abstract: Data–driven predictive control (DPC) is becoming an attractive alternative to model predictive control as it requires less system knowledge for implementation and reliable data is increasingly available in smart engineering systems. Two main approaches exist within DPC: the subspace approach, which estimates prediction matrices (unbiased for large data) and the behavioral, data-enabled approach, which uses Hankel data matrices for prediction (allows for optimizing the bias/variance trade–off). In this paper we develop a novel, generalized DPC (GDPC) algorithm by merging subspace and Hankel predictors. The predicted input sequence is defined as the sum of a known, baseline input sequence, and an optimized input sequence. The corresponding baseline output sequence is computed using an unbiased, subspace predictor, while the optimized predicted output sequence is computed using a Hankel matrix predictor. By combining these two types of predictors, GDPC can achieve high performance for noisy data even when using a small Hankel matrix, which is computationally more efficient. Simulation results for a benchmark example from the literature show that GDPC with a reduced size Hankel matrix can match the performance of data–enabled predictive control with a larger Hankel matrix in the presence of noisy data.

Keywords: data–driven control; predictive control; constrained control; regularized least squares

MSC: 37N35; 93C40; 93D20

Citation: Lazar, M.; Verheijen, P.C.N. Generalized Data–Driven Predictive Control: Merging Subspace and Hankel Predictors. *Mathematics* **2023**, *11*, 2216. https://doi.org/10.3390/math11092216

Academic Editors: Tiberiu Letia and Alexandru Onea

Received: 31 March 2023
Revised: 28 April 2023
Accepted: 30 April 2023
Published: 8 May 2023

1. Introduction

Reliable data is becoming increasingly available in modern, smart engineering systems, including mechatronics, robotics, power electronics, automotive systems, and smart infrastructures, see, e.g., [1,2] and the references therein. For these application domains, model predictive control (MPC) [3,4] has become the preferred advanced control method for several reasons, including constraints handling, anticipating control actions, and optimal performance. Since obtaining and maintaining accurate models requires effort and reliable data becomes readily available in engineering systems, it is of interest to develop data–driven predictive control (DPC) algorithms that can be implemented in practice. An indirect data–driven approach to predictive control design was already developed in [5] more than 20 years ago, i.e., subspace predictive control (SPC). The SPC approach skips the identification of the prediction model and identifies the complete prediction matrices from input–output data using least squares. This provides an unbiased predictor for sufficiently large data.

More recently, a direct data–driven approach to predictive control design was developed in [6] based on behavioral systems theory and Willems' fundamental lemma [7], i.e., data–enabled predictive control (DeePC). The idea to use (reduced order) Hankel matrices as predictors has been put forward earlier in [8], but the first well–posed constrained data–enabled predictive control algorithm was formulated in [6], to the best of the authors' knowledge. The DeePC approach skips the identification of prediction models or matrices all together and utilizes Hankel matrices built from input–output data to parameterize predicted future inputs and outputs. In the deterministic, noise free case, equivalence of MPC and DeePC was established in [6,9], while equivalence of SPC and DeePC was shown in [10]. Stability guarantees for DeePC were first obtained in [9] by means of terminal equality constraints and input–output–to–state stability Lyapunov functions. Alternatively, stability guarantees for DeePC were provided in [11] using terminal inequality constraints and dissipation inequalities involving storage and supply functions. An important contribution to DeePC is the consistent regularization cost introduced in [12], which enables reliable performance in the presence of noisy data. Indeed, since the DeePC algorithm jointly solves estimation and controller synthesis problems, the regularization derived in [12] allows one to optimize the bias/variance trade–off if data is corrupted by noise. A systematic method for tuning the regularization cost weighting parameter was recently presented in [13].

Computationally, SPC has the same number of optimization variables as MPC, which is equal to the number of control inputs times the prediction horizon. In DeePC, the number of optimization variables is also driven by the data length, which must be in general much larger than the prediction horizon. Especially in the case of noisy data, a large data size is required to attain reliable predictions, see, e.g., [9,12,13]. As this hampers real–time implementation, it is of interest to improve computational efficiency of DeePC. In [14], a computationally efficient formulation of DeePC was provided via LQ factorization of the Hankel data matrix, which yields the same online computational complexity as SPC/MPC. In this approach, DeePC yields an unbiased predictor, similar to SPC. In [15], a singular value decomposition is performed on the original Hankel data matrix and a DeePC algorithm is designed based on the resulting reduced Hankel matrix. Therein, it was shown that this approach can significantly reduce the computational complexity of DeePC, while improving the accuracy of predictions for noisy data. In [16], an efficient numerical method that exploits the structure of Hankel matrices was developed for solving quadratic programs (QPs) specific to DeePC. Regarding real–life applications of DeePC, the minimal data size required for persistence of excitation is typically used, see, e.g., [17,18], or an unconstrained solution of DeePC is used instead of solving a QP, see, e.g., [19]. These approaches however limit the achievable performance in the presence of noisy data and hard constraints, respectively.

In this paper we develop a novel, generalized DPC (GDPC) algorithm by merging subspace and Hankel predictors with the goal of reducing online computational complexity without sacrificing control performance in the presence of noisy data. The predicted input sequence is defined as the sum of a known, baseline input sequence and an optimized input sequence. The corresponding baseline output sequence is computed using an unbiased, subspace predictor based on a large data set, while the optimized predicted output sequence is computed using a Hankel matrix predictor based on a reduced data set. Via the extension of Willems' fundamental lemma to multiple data sets [20], the sum of the two trajectories spanned by two (possibly different) data sets will remain a valid system trajectory, as long as the combined data matrices are collectively persistently exciting and the system is linear. By combining these two types of predictors, GDPC can achieve high performance in the presence of noisy data even when using Hankel matrices of smaller size, which is computationally efficient and preserves the bias–variance tradeoff benefits of DeePC. The performance and computational complexity of GDPC with a minimal (according to DeePC design criteria) size Hankel matrix is evaluated for a benchmark example from the MPC literature and compared to DeePC with a Hankel matrix of varying size.

The remainder of this paper is structured as follows. The necessary notation and the DeePC approach to data–driven predictive control are introduced in Section 2. The GDPC algorithm is presented in Section 3, along with design guidelines, stability analysis and other relevant remarks. Simulation results and a comparison with DeePC are provided in Section 4 for a benchmark example from the literature. Conclusions are summarized in Section 5. Frequently used abbreviations are summarized in Table 1.

Table 1. Table of abbreviations.

MPC	Model Predictive Control
DPC	Data–driven Predictive Control
SPC	Subspace Predictive Control
DeePC	Data–enabled Predictive Control
GDPC	Generalized Data–driven Predictive Control
QP	Quadratic Programming

2. Preliminaries

Consider a discrete–time linear dynamical system subject to zero–mean Gaussian noise $w(k) \sim \mathcal{N}(0, \sigma_w^2 I)$:

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k), \quad k \in \mathbb{N}, \\ y(k) &= Cx(k) + w(k), \end{aligned} \tag{1}$$

where $x \in \mathbb{R}^n$ is the state, $u \in \mathbb{R}^{n_u}$ is the control input, $y \in \mathbb{R}^{n_y}$ is the measured output and (A, B, C) are real matrices of suitable dimensions. We assume that (A, B) is controllable and (A, C) is observable. By applying a persistently exciting input sequence $\{u(k)\}_{k \in \mathbb{N}_{[0,T]}}$ of length T to system (1) we obtain a corresponding output sequence $\{y(k)\}_{k \in \mathbb{N}_{[0,T]}}$.

If one considers an input–output model corresponding to (1), it is necessary to introduce the parameter T_{ini} that limits the window of past input–output data necessary to compute the current output, i.e.,

$$y(k) = \sum_{i=1}^{T_{ini}} a_i y(k-i) + \sum_{i=1}^{T_{ini}} b_i u(k-i), \tag{2}$$

for some real–valued coefficients. For simplicity of exposition we assume the same T_{ini} for inputs and outputs.

Next, we introduce some instrumental notation. For any finite number $q \in \mathbb{N}_{\geq 1}$ of vectors $\{\xi_1, \ldots, \xi_q\} \in \mathbb{R}^{n_1} \times \ldots \times \mathbb{R}^{n_q}$ we will make use of the operator $\mathrm{col}(\xi_1, \ldots, \xi_q) := [\xi_1^\top, \ldots, \xi_q^\top]^\top$. For any $k \geq 0$ (starting time instant in the data vector) and $j \geq 1$ (length of the data vector), define

$$\bar{\mathbf{u}}(k, j) := \mathrm{col}(u(k), \ldots, u(k+j-1)), \quad \bar{\mathbf{y}}(k, j) := \mathrm{col}(y(k), \ldots, y(k+j-1)).$$

Let $N \geq T_{ini}$ denote the prediction horizon. Then we can define the Hankel data matrices:

$$\begin{aligned} \mathbf{U}_p &:= \begin{bmatrix} \bar{\mathbf{u}}(0, T_{ini}) & \ldots & \bar{\mathbf{u}}(T-1, T_{ini}) \end{bmatrix}, \\ \mathbf{Y}_p &:= \begin{bmatrix} \bar{\mathbf{y}}(1, T_{ini}) & \ldots & \bar{\mathbf{y}}(T, T_{ini}) \end{bmatrix}, \\ \mathbf{U}_f &:= \begin{bmatrix} \bar{\mathbf{u}}(T_{ini}, N) & \ldots & \bar{\mathbf{u}}(T_{ini}+T-1, N) \end{bmatrix}, \\ \mathbf{Y}_f &:= \begin{bmatrix} \bar{\mathbf{y}}(T_{ini}+1, N) & \ldots & \bar{\mathbf{y}}(T_{ini}+T, N) \end{bmatrix}. \end{aligned} \tag{3}$$

According to the DeePC design [6], one must choose $T_{ini} \geq n$ and $T \geq (n_u+1)(T_{ini} + N + n) - 1$, which implicitly requires an assumption on the system order (number of states). Given the measured output $y(k)$ at time $k \in \mathbb{N}$ and $T_{ini} \in \mathbb{N}_{\geq 1}$ we define the sequences of known input–output data at time $k \geq T_{ini}$, which are trajectories of system (1):

$$\mathbf{u}_{ini}(k) := \mathrm{col}(u(k - T_{ini}), \ldots, u(k-1)), \quad \mathbf{y}_{ini}(k) := \mathrm{col}(y(k - T_{ini} + 1), \ldots, y(k)).$$

Next, we define the sequences of predicted inputs and outputs at time $k \geq T_{ini}$, which should also be trajectories of system (1):

$$\mathbf{u}(k) := \mathrm{col}(u(0|k), \ldots, u(N-1|k)), \quad \mathbf{y}(k) := \mathrm{col}(y(1|k), \ldots, y(N|k)),$$

where the notation $y(i|k)$ denotes the predicted value of $y(k+i)$ using measured output data available up to $y(k)$.

For a positive definite matrix L let $L^{\frac{1}{2}}$ denote its Cholesky factorization. At time $k \geq T_{ini}$, given $\mathbf{u}_{ini}(k), \mathbf{y}_{ini}(k)$, the regularized DeePC algorithm [12] computes a sequence of predicted inputs and outputs as follows:

$$\min_{\mathbf{g}(k),\mathbf{u}(k),\mathbf{y}(k),\sigma(k)} l_N(y(N|k)) + \sum_{i=0}^{N-1} l(y(i|k), u(i|k)) + \lambda_g l_g(\mathbf{g}(k)) + \lambda_\sigma l_\sigma(\sigma(k)) \tag{4a}$$

subject to constraints:

$$\begin{bmatrix} \mathbf{U}_p \\ \mathbf{Y}_p \\ \mathbf{U}_f \\ \mathbf{Y}_f \end{bmatrix} \mathbf{g}(k) = \begin{bmatrix} \mathbf{u}_{ini}(k) \\ \mathbf{y}_{ini}(k) + \sigma(k) \\ \mathbf{u}(k) \\ \mathbf{y}(k) \end{bmatrix}, \tag{4b}$$

$$(\mathbf{y}(k), \mathbf{u}(k)) \in \mathbb{Y}^N \times \mathbb{U}^N. \tag{4c}$$

Above

$$l(y,u) := \|Q^{\frac{1}{2}}(y - r_y)\|_2^2 + \|R^{\frac{1}{2}}(u - r_u)\|_2^2, \quad l_\sigma(\sigma) := \|\sigma\|_2^2 \tag{5}$$

for some positive definite Q, R matrices. The terminal cost is typically chosen larger than the output stage cost, to enforce convergence to the reference; a common choice is a scaled version of the output stage cost, i.e., $l_N(y) := \alpha l(y,0)$, $\alpha \geq 1$. The references $r_y \in \mathbb{R}^{n_y}$ and $r_u \in \mathbb{R}^{n_u}$ can be constant or time–varying. We assume that the sets $\mathbb{Y}$ and $\mathbb{U}$ contain r_y and r_u in their interior, respectively. The cost

$$l_g(\mathbf{g}) := \|(I - \Pi)\mathbf{g}\|_2^2, \quad \Pi := \begin{bmatrix} \mathbf{U}_p \\ \mathbf{Y}_p \\ \mathbf{U}_f \end{bmatrix}^\dagger \begin{bmatrix} \mathbf{U}_p \\ \mathbf{Y}_p \\ \mathbf{U}_f \end{bmatrix}, \tag{6}$$

is a regularization cost proposed in [12], where $[\cdot]^\dagger$ denotes a generalized pseudo–inverse. Notice that using such a regularization cost requires $T > T_{ini}(n_u + n_y) + Nn_u$ in order to ensure that the matrix $I - \Pi$ has a sufficiently large null–space. If a shorter data length T is desired, alternatively, the regularization cost $l_g(\mathbf{g}) := \|\mathbf{g}\|_2^2$ can be used. However, this regularization is not consistent, as shown in [12].

In the deterministic, noise–free case, the DeePC algorithm [6] does not require the costs l_g, l_σ and the variables σ. We observe that the computational complexity of DeePC is dominated by the vector of variables $\mathbf{g} \in \mathbb{R}^T$, with $T \geq (n_u + 1)(T_{ini} + N + n) - 1$. Hence, ideally one would prefer to work with the minimal value of data length T, but in the presence of noise, typically, a rather large data length T is required for accurate predictions [9,12,13].

3. Generalized Data–Driven Predictive Control

In this section we develop a novel, generalized DPC algorithm by constructing the predicted input sequence $\mathbf{u}(k)$ as the sum of two input sequences, i.e.,

$$\mathbf{u}(k) := \bar{\mathbf{u}}(k) + \mathbf{u}_g(k), \quad k \in \mathbb{N}, \tag{7}$$

where $\bar{\mathbf{u}}$ is a known, base line input sequence typically chosen as the shifted, optimal input sequence from the previous time, i.e.,

$$\bar{\mathbf{u}}(k) := \{u^*(1|k-1), \ldots, u^*(N-1|k-1), \bar{u}(N-1|k)\}, \tag{8}$$

where common choices for the last element $\bar{u}(N-1|k)$ are r_u or $u^*(N-1|k-1)$. At time $k = T_{ini}$, when enough input–output data is available to run the GDPC algorithm, $\bar{\mathbf{u}}(k)$ is initialized using a zero input sequence (or an educated guess). The sequence of inputs $\mathbf{u}_g$ can be freely optimized online by solving a QP, as explained next.

Using a single persistently exciting input sequence split into two parts, or two different persistently exciting input sequences, we can define two Hankel data matrices as in (3), i.e.,

$$\bar{H} := \begin{bmatrix} \bar{\mathbf{U}}_p \\ \bar{\mathbf{Y}}_p \\ \bar{\mathbf{U}}_f \\ \bar{\mathbf{Y}}_f \end{bmatrix} \in \mathbb{R}^{(n_u+n_y)(T_{ini}+N)\times T}, \quad H := \begin{bmatrix} \mathbf{U}_p \\ \mathbf{Y}_p \\ \mathbf{U}_f \\ \mathbf{Y}_f \end{bmatrix} \in \mathbb{R}^{(n_u+n_y)(T_{ini}+N)\times T_g}, \tag{9}$$

where the length T of the first part/sequence can be taken as large as desired and the choice of the length T_g of the second part/sequence is flexible. That is, T_g should be small enough to meet computational requirements, but it should provide enough degrees of freedom to optimize the bias/variance trade off. Offline, compute the matrix

$$\Theta := \bar{\mathbf{Y}}_f \begin{bmatrix} \bar{\mathbf{U}}_p \\ \bar{\mathbf{Y}}_p \\ \bar{\mathbf{U}}_f \end{bmatrix}^{\dagger} \in \mathbb{R}^{(n_y N)\times(T_{ini}(n_u+n_y)+n_u N)}. \tag{10}$$

Online, at time $k \geq T_{ini}$, given $\mathbf{u}_{ini}(k)$, $\mathbf{y}_{ini}(k)$ and $\bar{\mathbf{u}}(k)$, compute

$$\bar{\mathbf{y}}(k) = \Theta \begin{bmatrix} \mathbf{u}_{ini}(k) \\ \mathbf{y}_{ini}(k) \\ \bar{\mathbf{u}}(k) \end{bmatrix} \tag{11}$$

and solve the *GDPC optimization problem*:

$$\min_{\mathbf{g}(k),\mathbf{u}(k),\mathbf{y}(k),\sigma(k)} l_N(y(N|k)) + \sum_{i=0}^{N-1} l(y(i|k), u(i|k)) + \lambda_g l_g(\mathbf{g}(k)) + \lambda_\sigma l_\sigma(\sigma(k)) \tag{12a}$$

subject to constraints:

$$\begin{bmatrix} \mathbf{U}_p \\ \mathbf{Y}_p \\ \mathbf{U}_f \\ \mathbf{Y}_f \end{bmatrix} \mathbf{g}(k) = \begin{bmatrix} 0 \\ \sigma(k) \\ \mathbf{u}(k) - \bar{\mathbf{u}}(k) \\ \mathbf{y}(k) - \bar{\mathbf{y}}(k) \end{bmatrix}, \tag{12b}$$

$$(\mathbf{y}(k), \mathbf{u}(k)) \in \mathbb{Y}^N \times \mathbb{U}^N. \tag{12c}$$

Above, the cost functions $l_N(y)$, $l(y,u)$, $l_g(g)$ and $l_\sigma(\sigma)$ are defined in the same way as in (5) for some $Q, R \succ 0$.

Since the size of the matrix Θ does not depend on the data length T, i.e., the number of columns of the Hankel matrix $\bar{H}$, the computation as in (11) of the predicted output corresponding to $\bar{\mathbf{u}}(k)$ is efficient even for a large T. Thus, GDPC benefits from an unbiased base line output prediction, which allows choosing T_g, i.e., the number of columns of the Hankel matrix H, much smaller than T. In turn, this reduces the online computational complexity of GDPC, without sacrificing performance in the presence of noisy data. It can be argued that the selection of T_g provides a trade–off between computational complexity and available degrees of freedom to optimize the bias/variance trade off.

Remark 1 (Offset–free GDPC design)**.** *In practice it is of interest to achieve offset–free tracking. Following the offset–free design for SPC developed in [21], which was further applied to DeePC in [13], it is possible to design an offset–free GDPC algorithm by defining an incremental input sequence*

$$\Delta\mathbf{u}(k) := \Delta\bar{\mathbf{u}}(k) + \Delta\mathbf{u}_g(k), \quad k \in \mathbb{N},$$

where $\Delta\bar{\mathbf{u}}$ *is chosen as the shifted optimal input sequence from the previous time, i.e.,*

$$\Delta\bar{\mathbf{u}}(k) := \{\Delta u^*(1|k-1), \ldots, \Delta u^*(N-1|k-1), \Delta\bar{u}(N-1|k)\}. \tag{13}$$

The input data blocks in the Hankel matrices $\bar{H}$ and H must be replaced with incremental input data, i.e., $\Delta\bar{\mathbf{U}}_p$, $\Delta\bar{\mathbf{U}}_f$ and $\Delta\mathbf{U}_p$, $\Delta\mathbf{U}_f$, respectively. The input applied to the system is then $u(k) := \Delta u^*(0|k) + u(k-1)$.

In what follows we provide a formal analysis of the GDPC algorithm.

3.1. Well-Posedness and Design of GDPC

In this subsection we show that in the deterministic case GDPC predicted trajectories are trajectories of system (1). In this case, the GDPC optimization problem can be simplified as

$$\min_{\mathbf{g}(k),\mathbf{u}(k),\mathbf{y}(k)} l_N(y(N|k)) + \sum_{i=0}^{N-1} l(y(i|k), u(i|k)) \tag{14a}$$

subject to constraints:

$$\begin{bmatrix} \mathbf{U}_p \\ \mathbf{Y}_p \\ \mathbf{U}_f \\ \mathbf{Y}_f \end{bmatrix} \mathbf{g}(k) = \begin{bmatrix} 0 \\ 0 \\ \mathbf{u}(k) - \bar{\mathbf{u}}(k) \\ \mathbf{y}(k) - \bar{\mathbf{y}}(k) \end{bmatrix}, \tag{14b}$$

$$(\mathbf{y}(k), \mathbf{u}(k)) \in \mathbb{Y}^N \times \mathbb{U}^N. \tag{14c}$$

Lemma 1 (GDPC well–posedness)**.** *Consider one (or two) persistently exciting input sequence(s) of sufficient length(s) and construct two Hankel matrices $\bar{H}$ and H as in (9) with T and T_g columns, respectively, and such that $\bar{H}$ has full row rank. Consider also the corresponding output sequence(s) generated using system (1). For any given input sequence $\bar{\mathbf{u}}(k)$, and initial conditions $\mathbf{u}_{\text{ini}}(k)$ and $\mathbf{y}_{\text{ini}}(k)$, let $\bar{\mathbf{y}}(k)$ be defined as in (11). Then there exists a real vector $\mathbf{g}(k) \in \mathbb{R}^{T_g}$ such that (14b) holds if and only if $\mathbf{u}(k)$ and $\mathbf{y}(k)$ are trajectories of system (1).*

Proof. Define $\bar{\mathbf{g}}(k) := \begin{bmatrix} \bar{\mathbf{U}}_p \\ \bar{\mathbf{Y}}_p \\ \bar{\mathbf{U}}_f \end{bmatrix}^{\dagger} \begin{bmatrix} \mathbf{u}_{ini}(k) \\ \mathbf{y}_{ini}(k) \\ \bar{\mathbf{u}}(k) \end{bmatrix}$. Then it holds that:

$$\begin{aligned} \begin{bmatrix} \bar{H} & H \end{bmatrix} \begin{bmatrix} \bar{\mathbf{g}}(k) \\ \mathbf{g}(k) \end{bmatrix} &= \begin{bmatrix} \bar{\mathbf{U}}_p \\ \bar{\mathbf{Y}}_p \\ \bar{\mathbf{U}}_f \\ \bar{\mathbf{Y}}_f \end{bmatrix} \bar{\mathbf{g}}(k) + \begin{bmatrix} \mathbf{U}_p \\ \mathbf{Y}_p \\ \mathbf{U}_f \\ \mathbf{Y}_f \end{bmatrix} \mathbf{g}(k) \\ &= \begin{bmatrix} \mathbf{u}_{ini}(k) \\ \mathbf{y}_{ini}(k) \\ \bar{\mathbf{u}}(k) \\ \bar{\mathbf{y}}(k) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \mathbf{u}(k) - \bar{\mathbf{u}}(k) \\ \mathbf{y}(k) - \bar{\mathbf{y}}(k) \end{bmatrix} = \begin{bmatrix} \mathbf{u}_{ini}(k) \\ \mathbf{y}_{ini}(k) \\ \mathbf{u}(k) \\ \mathbf{y}(k) \end{bmatrix}. \end{aligned} \tag{15}$$

Since the matrix $\bar{H}$ has full row rank, the concatenated matrix $\begin{bmatrix} \bar{H} & H \end{bmatrix}$ has full row rank and as such, the claim follows from [6] if one input sequence is used and from [20] if two different input sequences are used to build the Hankel matrices. □

The selection of T_g enables a trade off between computational complexity and available degrees of freedom to improve the output sequence generated by the known, base line input sequence. Indeed, a larger T_g results in a larger null space of the data matrix $\begin{bmatrix} \mathbf{U}_p \\ \mathbf{Y}_p \end{bmatrix}$, which confines $\mathbf{g}(k)$ in the deterministic case. However, high performance can be achieved in the case of noisy data even for a smaller T_g, because the base line predicted output is calculated using an unbiased least squares predictor, i.e., as defined in (11).

An alternative way to define the known input sequence $\bar{\mathbf{u}}(k)$ is to use an unconstrained SPC control law [5]. To this end, notice that the matrix Θ can be partitioned, see, e.g., [21], into $\begin{bmatrix} P_1 & P_2 & \Gamma \end{bmatrix}$ such that

$$\bar{\mathbf{y}}(k) = \begin{bmatrix} P_1 & P_2 \end{bmatrix} \begin{bmatrix} \mathbf{u}_{ini}(k) \\ \mathbf{y}_{ini}(k) \end{bmatrix} + \Gamma \bar{\mathbf{u}}(k).$$

Then, by defining $\Psi := \text{diag}\{R, \dots, R\}$, $\Omega := \text{diag}\{Q, \dots, Q, \alpha Q\}$, $G := 2\left(\Psi + \Gamma^T \Omega \Gamma\right)$ and $F := 2\Gamma^T \Omega$, we obtain:

$$\bar{\mathbf{u}}_{\text{spc}}(k) := -G^{-1}\left(F\left(\begin{bmatrix} P_1 & P_2 \end{bmatrix} \begin{bmatrix} \mathbf{u}_{ini}(k) \\ \mathbf{y}_{ini}(k) \end{bmatrix} - \mathbf{r}_y\right) - 2\Psi \mathbf{r}_u\right), \tag{16}$$

where $\mathbf{r}_y := \text{col}(r_y, \dots, r_y)$ and $\mathbf{r}_u := \text{col}(r_u, \dots, r_u)$. Since the inverse of G is computed offline, computing $\bar{\mathbf{u}}_{\text{spc}}(k)$ online is numerically efficient even for a large prediction horizon N. In this case, since the corresponding base line predicted output trajectory is unbiased, the simpler regularization cost $l_g(\mathbf{g}) := \|\mathbf{g}\|_2^2$ can be used in (12a), without loosing consistency. When the base line input sequence is computed as in (16), the optimized input sequence $\mathbf{u}_g(k)$ acts to enforce constraints, when the analytic SPC control law (16) results in violation of input or state constraints, and it can also optimize the bias/variance trade off under appropriate tuning of λ_g.

Remark 2 (Relation with SPC and DeePC)**.** *The main novelty of GDPC with respect to existing data–driven predictive controllers SPC [5] and DeePC [6] is the construction of a hybrid data-driven predictor, i.e., a combination of subspace and Hankel predictors. This comes with the benefit that online computational complexity can be reduced, by reducing the size of the Hankel matrix, while robust control performance is still achieved due to the unbiased subspace predictor. Moreover, GDPC can recover SPC or DeePC under specific settings as follows. First, notice that for a zero baseline input sequence, i.e., $\bar{\mathbf{u}}(k) = 0$ for all $k \in \mathbb{N}$, and a sufficiently large T_g, the optimization problem (14) becomes the DeePC problem (4) and yields the corresponding $\mathbf{g}^*_{DeePC}(k)$. Then, if in GDPC one selects $T_g = T$ and $H = \bar{H}$ in (9), we have that $\mathbf{g}(k) := \mathbf{g}^*_{DeePC}(k) - \bar{\mathbf{g}}(k)$ satisfies (15), and the DeePC solution is recovered for any baseline sequence $\bar{\mathbf{u}}(k)$. Alternatively, if the baseline sequence is chosen as in (16) and the regularization cost $\lambda_g l_g(\mathbf{g}(k)) = \lambda_g \|\mathbf{g}\|_2^2$ is added in (14a), then $\mathbf{u}^*(k) \to \mathbf{u}^*_{SPC}(k)$ as $\lambda_g \to \infty$, where $\mathbf{u}^*_{SPC}(k)$ denotes optimal solution of the corresponding SPC algorithm, see, e.g., [13].*

Remark 3 (GDPC open challenges)**.** *A systematic method for choosing the data length T for the subspace predictor versus the data length T_g for the Hankel predictor is required. This choice represents an additional degree of freedom and depends on the noise level, system order and prediction horizon, while it affects the null space of $\begin{bmatrix} \mathbf{U}_p \\ \mathbf{Y}_p \end{bmatrix}$. The data length T for the subpace predictor can be determined via standard subspace identification criteria, see, e.g., [3,22]. T_g can be chosen via the singular value decomposition method proposed in [15] or as the minimal data length required by DeePC [6]. Another open problem is the tuning of the hyper-parameters λ_g and λ_σ. A systematic method for tuning λ_g was presented for DeePC in [13]. However, the same method does not apply directly to GDPC, due to the presence of the time-varying baseline sequence.*

3.2. Stability of GDPC

In this section we will provide sufficient conditions under which GDPC is asymptotically stabilizing. To this end define $J(\mathbf{y}(k), \mathbf{u}(k)) := l_N(y(N|k)) + \sum_{i=0}^{N-1} l(y(i|k), u(i|k))$, let $l_N(y) := l(y, 0)$ and let $\mathbf{y}^*(k)$ and $\mathbf{u}^*(k)$ denote optimal trajectories at time $k \geq T_{ini}$. Given an optimal input sequence at time $k \geq T_{ini}$, i.e., $\mathbf{u}^*(k) = \bar{\mathbf{u}}(k) + \mathbf{u}^*_g(k)$, define a suboptimal input sequence at time $k+1$ as

$$\mathbf{u}_s(k+1) = \bar{\mathbf{u}}(k+1) + \mathbf{u}_g(k+1)$$
$$= \mathrm{col}(u^*(1|k), \dots, u^*(N-1|k), \bar{u}(N|k)) + \mathrm{col}(0, \dots, 0, 0), \tag{17}$$

and let

$$\mathbf{y}_s(k+1) = \bar{y}(k+1) = \mathrm{col}(y^*(2|k), \dots, y^*(N|k), \bar{y}(N+1|k))$$

denote the corresponding suboptimal output trajectory. Note that the last output in the suboptimal output sequence satisfies:

$$\bar{y}(N+1|k) = \sum_{i=1}^{T_{ini}} a_i y^*(N+1-i|k) + \sum_{i=1}^{T_{ini}} b_i u^*(N+1-i|k).$$

Definition 1 (Class $\mathcal{K}$ functions)**.** *A function $\varphi : \mathbb{R}_+ \to \mathbb{R}_+$ belongs to class $\mathcal{K}$ if it is continuous, strictly increasing and $\varphi(0) = 0$. A function $\varphi : \mathbb{R}_+ \to \mathbb{R}_+$ belongs to class $\mathcal{K}_\infty$ if $\varphi \in \mathcal{K}$ and $\lim_{s \to \infty} \varphi(s) = \infty$. id denotes the identity $\mathcal{K}_\infty$ function, i.e., $id(s) = s$.*

Next, as proposed in [11], we define a non–minimal state:

$$\mathbf{x}_{ini}(k) := \mathrm{col}(y(k-T_{ini}), \dots, y(k-1), u(k-T_{ini}), \dots, u(k-1)),$$

and the function $W(\mathbf{x}_{ini}(k)) := \sum_{i=1}^{T_{ini}} l(y(k-i), u(k-i))$. In what follows we assume that $r_y = 0$ and $r_u = 0$ for simplicity of exposition. However, the same proof applies for any constant references that are compatible with an admissible steady–state.

Assumption 1 (Terminal stabilizing condition)**.** *For any admissible initial state $\mathbf{x}_{\mathrm{ini}}(k)$ there exists a function $\rho \in \mathcal{K}_\infty$, with $\rho < id$, a prediction horizon $N \geq T_{\mathrm{ini}}$ and $\bar{u}(N|k) \in \mathbb{U}$ such that $\bar{y}(N+1|k) \in \mathbb{Y}$ and*

$$l(\bar{y}(N+1|k), \bar{u}(N|k)) - (id - \rho) \circ l(y(k-T_{\mathrm{ini}}), u(k-T_{\mathrm{ini}})) \leq 0. \tag{18}$$

Theorem 1 (Stability of GDPC)**.** *Suppose that there exist $\alpha_{1,l}, \alpha_{2,l}, \alpha_{2,J} \in \mathcal{K}_\infty$ such that*

$$\alpha_{1,l}(\|\mathrm{col}(y,u)\|) \leq l(y,u) \leq \alpha_{2,l}(\|\mathrm{col}(y,u)\|), \quad \forall (y,u) \in \mathbb{Y} \times \mathbb{U} \tag{19a}$$
$$J(\mathbf{y}^*(k), \mathbf{u}^*(k)) \leq \alpha_{2,J}(\|\mathbf{x}_{\mathrm{ini}}(k)\|), \quad \forall \mathbf{x}_{\mathrm{ini}}(k) \in \mathcal{N}, \tag{19b}$$

for some proper set $\mathcal{N}$ with the origin in its interior. Furthermore, let Assumption 1 hold and suppose that problem (14) is feasible for all $k \geq T_{\mathrm{ini}}$. Then system (1) in closed–loop with the GDPC algorithm that solves problem (14) is asymptotically stable.

Proof. As done in [11] for the DeePC algorithm, we consider the following storage function

$$V(\mathbf{x}_{ini}(k)) := J(\mathbf{u}^*(k), \mathbf{y}^*(k)) + W(\mathbf{x}_{ini}(k)),$$

and we will prove that it is positive definite and it satisfies a dissipation inequality. First, from (19a) and by Lemma 14 in [11] we obtain that there exist $\alpha_{1,V}, \alpha_{2,V} \in \mathcal{K}_\infty$ such that

$$\alpha_{1,V}(\|\mathbf{x}_{ini}(k)\|) \leq V(\mathbf{x}_{ini}(k)) \leq \alpha_{2,V}(\|\mathbf{x}_{ini}(k)\|).$$

Then, define the supply function

$$s(y(k-T_{ini}), u(k-T_{ini})) := l(\bar{y}(N+1|k), \bar{u}(N|k)) - l(y(k-T_{ini}), u(k-T_{ini})).$$

By the principle of optimality, it holds that

$$\begin{aligned}
V(\mathbf{x}_{ini}(k+1)) - V(\mathbf{x}_{ini}(k)) &= J(\mathbf{y}^*(k+1), \mathbf{u}^*(k+1)) + W(\mathbf{x}_{ini}(k+1)) \\
&\quad - J(\mathbf{y}^*(k), \mathbf{u}^*(k)) - W(\mathbf{x}_{ini}(k)) \\
&\leq J(\mathbf{y}_s(k+1), \mathbf{u}_s(k+1)) - J(\mathbf{y}^*(k), \mathbf{u}^*(k)) \\
&\quad + W(\mathbf{x}_{ini}(k+1)) - W(\mathbf{x}_{ini}(k)) \\
&= l(\bar{y}(N+1|k), \bar{u}(N|k)) - l(y(0|k), u(0|k)) \\
&\quad + l(y(0|k), u(0|k)) - l(y(k-T_{ini}), u(k-T_{ini})) \\
&= s(y(k-T_{ini}), u(k-T_{ini})).
\end{aligned}$$

Hence, the storage function $V(\mathbf{x}_{ini}(k))$ satisfies a dissipation inequality along closed–loop trajectories. Since by (18) the supply function satisfies

$$s(y(k-T_{ini}), u(k-T_{ini})) \leq -\rho \circ l(y(k-T_{ini}), u(k-T_{ini})),$$

the claim then follows from Corollary 17 in [11]. □

Notice that condition (18) corresponds to a particular case of condition (25) employed in Corollary 17 in [11], i.e., for $M = 1$.

Remark 4 (Terminal stabilizing condition)**.** *The terminal stabilizing condition* (18) *can be regarded as an implicit condition, i.e., by choosing the prediction horizon N sufficiently large, this condition is more likely to hold. Alternatively, it could be implemented as an explicit constraint in problem* (14) *by adding one more input and output at the end of the predicted input and output sequences* $\mathbf{u}(k)$*,* $\mathbf{y}(k)$*, respectively. This also requires including the required additional data in the corresponding* $\bar{H}$ *and H Hankel matrices. This yields a convex quadratically constrained QP, which can still be solved efficiently. The stabilizing condition* (18) *can also be used in the regularized GDPC problem* (12) *to enforce convergence, but in this case a soft constraint implementation is recommended to prevent infeasibility due to noisy data.*

It is worth to point out that the conditions invoked in [9] imply that condition (18) holds, i.e., Assumption 1 is less conservative. Indeed, if a terminal equality constraint is imposed in problem (14), i.e., $y(N|k) = r_y$, then $\bar{u}(N|k) = r_u$ is a feasible choice at time $k+1$, which yields $\bar{y}(N+1|k) = r_y$ and hence, $l(\bar{y}(N+1|k), \bar{u}(N|k)) = 0$. Hence, (18) trivially holds since the stage cost $l(y,u)$ is positive definite and $\rho < \text{id}$. Alternatively, one could employ a data–driven method to compute a suitable invariant terminal set in the space of $\mathbf{x}_{ini}$, as proposed in [23]. Tractable data–driven computation of invariant sets is currently possible for ellipsoidal sets, via linear matrix inequalities, which also yields a convex quadratically constrained QP that has to be solved online.

4. Simulations Results

In this section we consider a benchmark MPC illustrative example from Section 6.4 in [4] based on the flight control of the longitudinal motion of a Boeing 747. After discretization with zero–order–hold for $T_s = 0.1[s]$ we obtain a discrete–time linear model as in (1) with:

$$A = \begin{bmatrix} 0.9997 & 0.0038 & -0.0001 & -0.0322 \\ -0.0056 & 0.9648 & 0.7446 & 0.0001 \\ 0.0020 & -0.0097 & 0.9543 & -0.0000 \\ 0.0001 & -0.0005 & 0.0978 & 1.0000 \end{bmatrix}, \quad B = \begin{bmatrix} 0.0010 & 0.1000 \\ -0.0615 & 0.0183 \\ -0.1133 & 0.0586 \\ -0.0057 & 0.0029 \end{bmatrix}, \qquad (20)$$
$$C = \begin{bmatrix} 1.0000 & 0 & 0 & 0 \\ 0 & -1.0000 & 0 & 7.7400 \end{bmatrix},$$

with two inputs, the throttle u_1 and u_2, the angle of the elevator, and two outputs, the longitudinal velocity and the climb rate, respectively. The inputs and outputs are constrained as follows:

$$\mathbb{U} := \left\{ u \in \mathbb{R}^2 \ : \ \begin{bmatrix} -20 \\ -20 \end{bmatrix} \leq u \leq \begin{bmatrix} 20 \\ 20 \end{bmatrix} \right\}$$
$$\mathbb{Y} := \left\{ y \in \mathbb{R}^2 \ : \ \begin{bmatrix} -25 \\ -15 \end{bmatrix} \leq Y \leq \begin{bmatrix} 25 \\ 15 \end{bmatrix} \right\}.$$

The cost function of the predictive controllers is defined as in (4) and (5) using $\lambda_g = 10^5$, $\lambda_\sigma = 10^7$, $N = 20$, $T_{ini} = 20$, $Q = 10 \cdot I_{n_y}$ and $R = 0.01 \cdot I_{n_u}$. For GDPC, if the suboptimal input sequence is computed as in (8), using shifted optimal sequence from the previous time with $\bar{u}(k+N-1) = u^*(k-2+N)$, the regularization cost $l_g(\mathbf{g}(k))$ is defined as in (6). If the analytic SPC solution (16) is used to compute the suboptimal input sequence, then the regularization cost $l_g(\mathbf{g}(k)) = \lambda_g \|\mathbf{g}(k)\|_2^2$ is used. For DeePC the regularization cost $l_g(\mathbf{g}(k))$ is defined as in (6). In this way, all 3 compared predictive control algorithms utilize consistent output predictors.

The QP problems corresponding to the data–driven predictive controllers are solved using the `quadprog` Matlab QP solver on a laptop with an Intel i7-9750H CPU and 16GB of RAM, with MATLAB version R2021a. The `quadprog` solver is not the fastest QP solver in general, but it has a high accuracy, which is useful to analyze with precision the control performance of different algorithms. The OSQP solver [24] can provide a faster alternative, and it was used in this work only to solve quadratically constrained QPs. Such optimization problems arise if the stabilizing constraint (18) is included in the GDPC problem.

The GDPC algorithm is implemented in Matlab based on Algorithms 1 and 2.

Algorithm 1 Data generation method, prior to controlling the system

```
Input: Measurable system, Desired sampling period $T_s$
 1: Generate persistently exciting input $U = \{u(k)\}_{k \in \mathbb{N}_{[0,T]}}$, see Section 13.3 in [22]
 2: Apply input, measure and collect system response $Y = \{y(k)\}_{k \in \mathbb{N}_{[0,T]}}$
 3: Construct Hankel matrices $\bar{H}$ and $H$ as in (9)
 4: Compute $\Theta$ as in (10) using $\bar{H}$
```

Algorithm 2 Implementation of GDPC

```
Input: $\bar{H}$, $H$, $\Theta$ and the system constraints
 for $k = 0 : T_{ini}$ do
     1: Measure $y(k)$
     2: Apply $u(k) = \gamma \cdot \text{rand}(k)$ to the system and wait for the next sampling interval
 end for
 for $k = T_{ini} + 1 : \text{end}$ do
     1: Measure $y(k)$
     2: Build $\mathbf{y}_{ini}(k)$, $\mathbf{u}_{ini}(k)$ from the last $T_{ini}$ data samples
     3: Compute $\bar{\mathbf{u}}(k)$ using either (8) or (16)
     4: Compute $\bar{\mathbf{y}}(k) = \Theta \begin{bmatrix} \mathbf{u}_{ini}(k)^T & \mathbf{y}_{ini}(k)^T & \bar{\mathbf{u}}(k)^T \end{bmatrix}^T$
     5: Solve QP (12) with any QP solver to obtain $\mathbf{u}(k)$
     6: Apply $u(k) = \mathbf{u}(k)_{[0:n_u]}$ to the system and wait for the next sampling interval
 end for
```

In what follows, the simulation results are structured into 3 subsections, focusing on nominal, noise–free data performance, noisy data performance for low and high noise variance and comparison with DeePC. The controllers will only start when T_{ini} samples have been collected. For the time instants up to $k = T_{ini}$, the system is actuated by a small random input. For the sake of a sound comparison, the random input signal used up to T_{ini} is identical for all simulations/predictive controllers. In the data generation experiment, the system inputs are constructed using two PRBS signals, both constrained between $[-3, 3]$.

4.1. Noise–Free Data GDPC Performance

In this simulation we implement the GDPC algorithm with the suboptimal input sequence computed as in (8). Figure 1 shows the outputs, inputs and optimized inputs $u_g(k)$ over time.

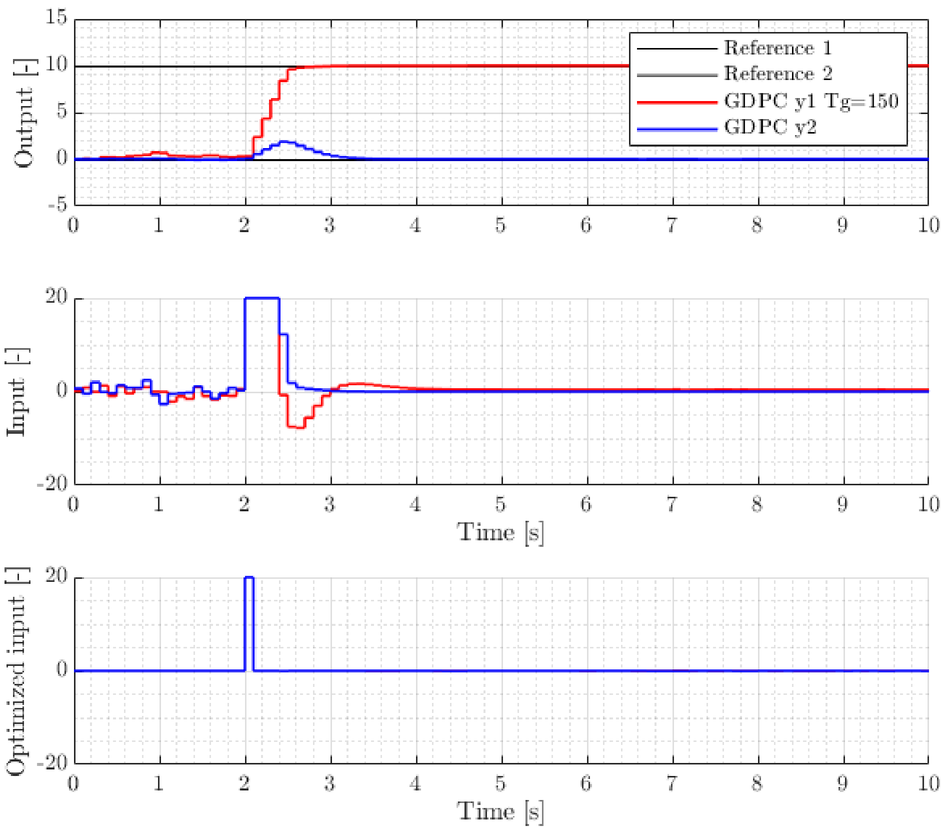

Figure 1. GDPC tracking performance: $T_g = 150$, $T = 1000$, $\sigma_w^2 I = 0I$.

We observe that the GDPC closed–loop trajectories converge to the reference values and that the optimized inputs are active only at the start, after which the suboptimal shifted sequence becomes optimal. The stabilizing condition (18) is implicitly satisfied along trajectories; when imposed online, the GDPC problem is recursively feasible and yields the same trajectories.

4.2. Noisy Data GDPC Performance

Next we illustrate the performance of GDPC for noisy data with a low and high variance. Figures 2 and 3 show the GDPC response using $\bar{u}(k)$ as defined in (8) and (16), respectively, for low variance noise.

Although the two different methods to calculate the base line input sequence $\bar{u}(k)$ show little difference in the resulting total input $u(k)$, the optimized part of the input, $u_g(k)$ shows a notable difference. For the GDPC algorithm that uses the unconstrained SPC we see that the optimized input only acts to enforce constraints, while around steady state the unconstrained SPC becomes optimal.

Next, we show the performance of GDPC for high–variance noise, which also requires a suitable increase of the data size. Figures 4 and 5 show the GDPC response using $\bar{u}(k)$ as defined in (8) and (16), respectively, for high variance noise. We see that both GDPC algorithms achieve robust control performance, while in the case of the SPC baseline input sequence, the optimized input is active also around steady state. This shows that GDPC indeed optimizes the bias variance trade off with respect to the SPC solution.

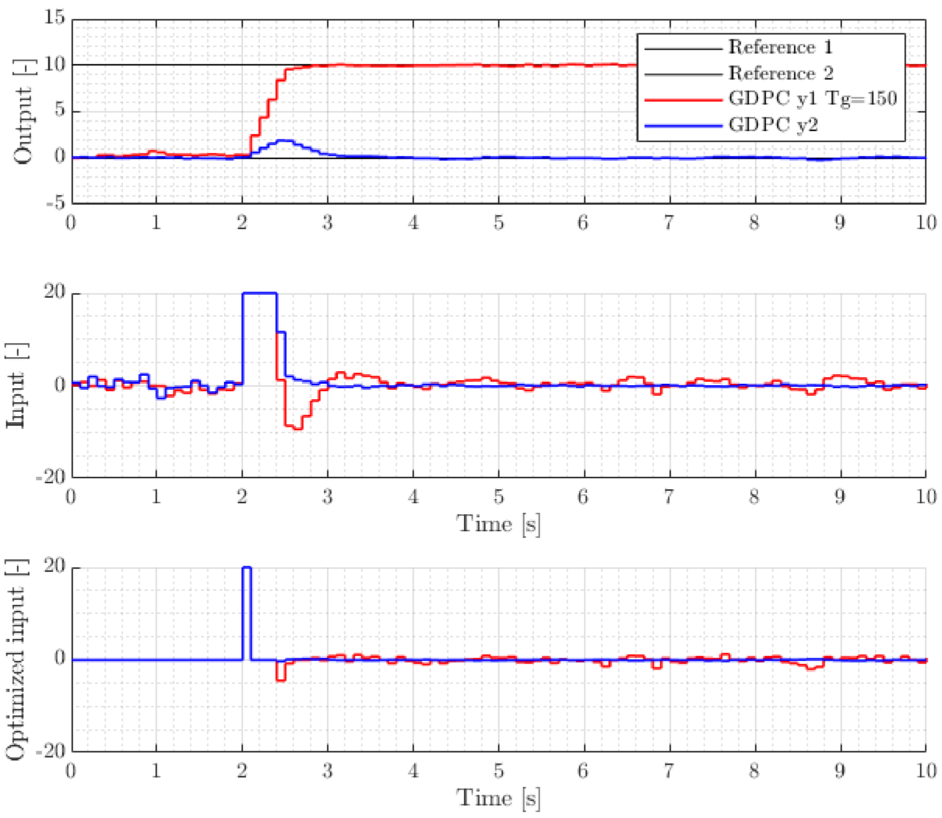

Figure 2. GDPC tracking performance: $T_g = 150$, $T = 1000$, $\sigma_w^2 I = 0.05I$, $\bar{u}$ as in (8).

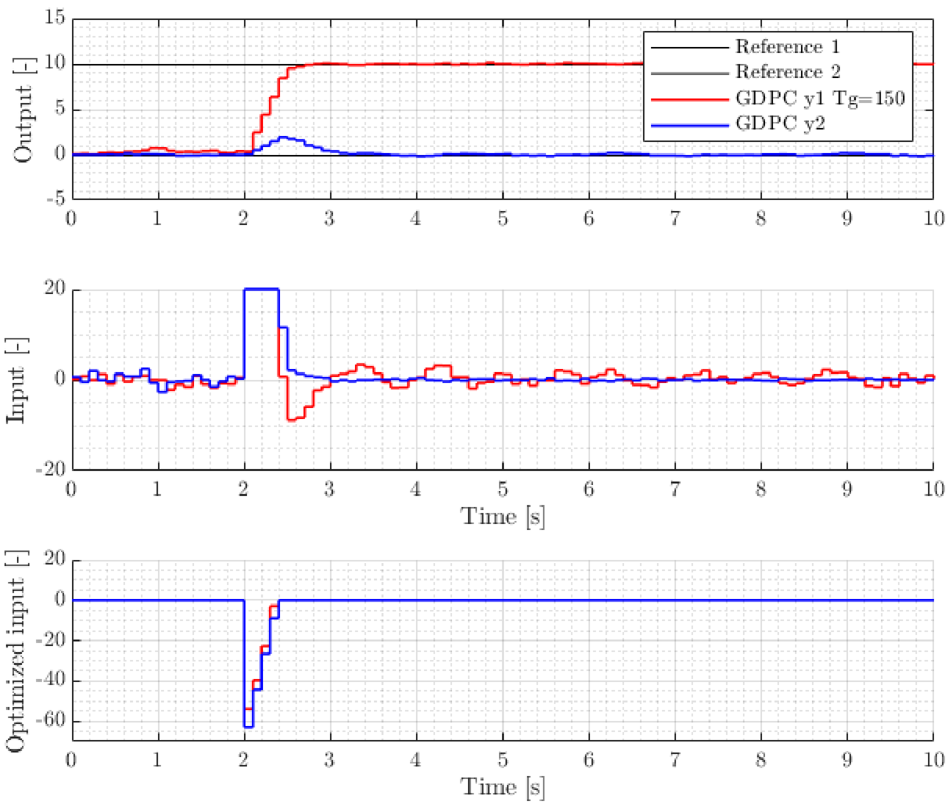

Figure 3. GDPC tracking performance: $T_g = 150$, $T = 1000$, $\sigma_w^2 I = 0.05I$, $\bar{u}$ as in (16).

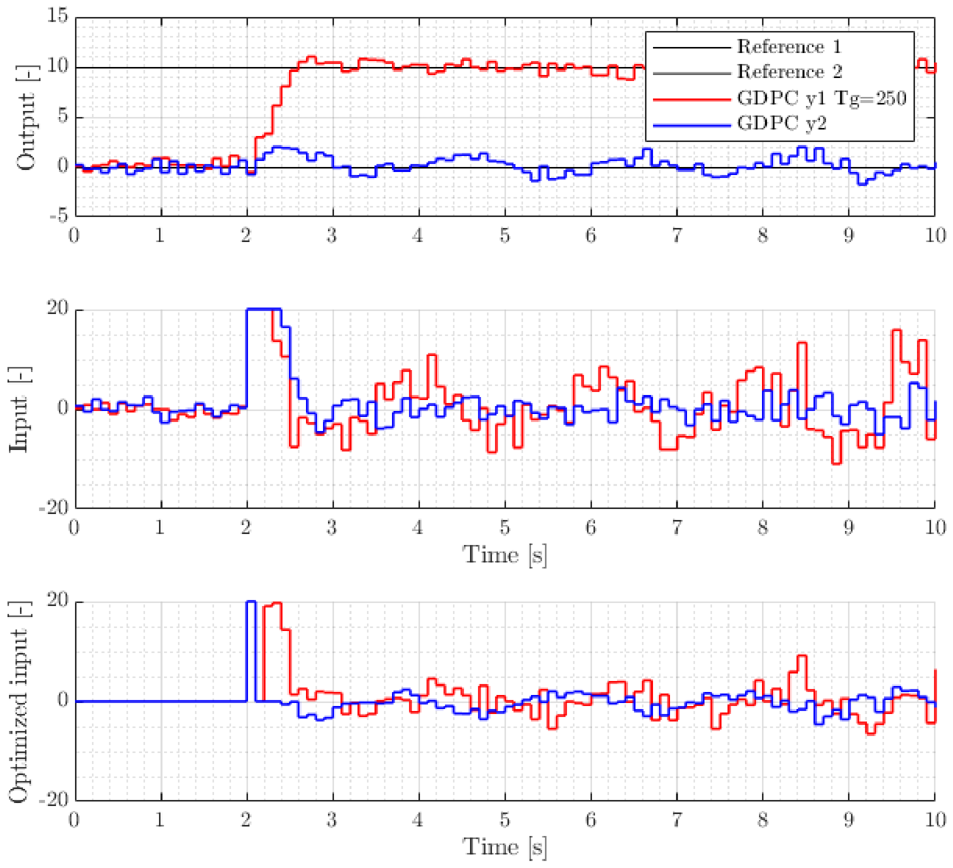

Figure 4. GDPC tracking performance: $T_g = 250$, $T = 5000$, $\sigma_w^2 I = 0.5I$, $\bar{u}$ as in (8).

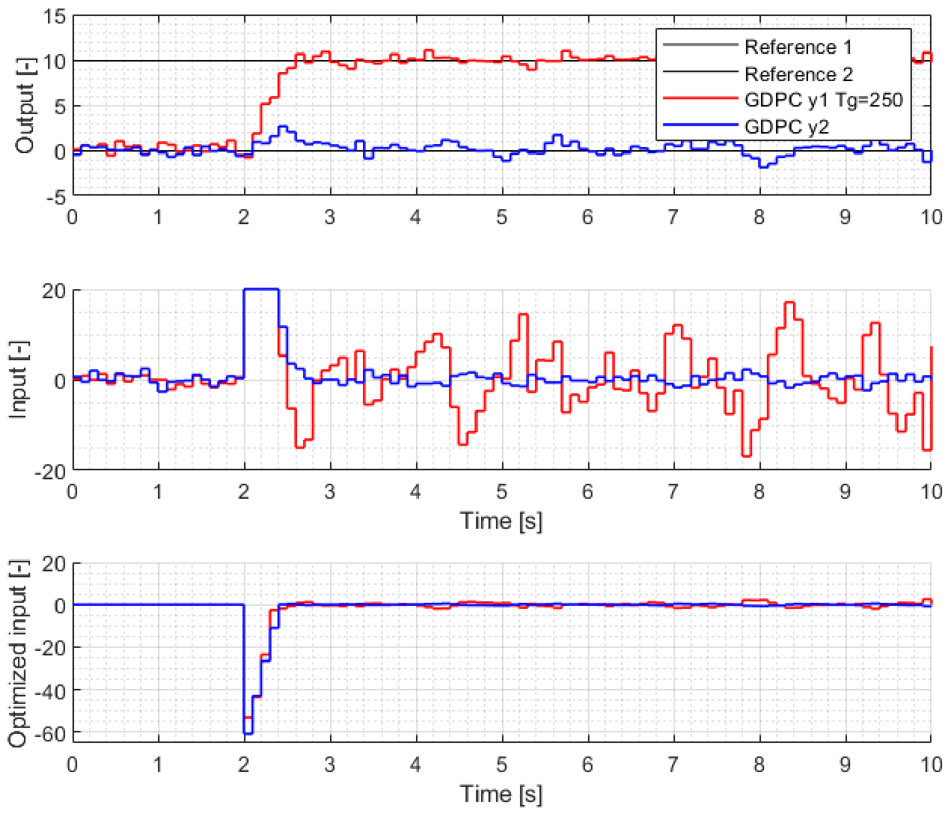

Figure 5. GDPC tracking performance: $T_g = 250$, $T = 5000$, $\sigma_w^2 I = 0.5I$, $\bar{u}$ as in (16).

In Figure 6 it can be observed that DeePC requires a data sequence of length 750 to achieve similar performance with GDPC with data lengths $T_g = 250$ (relevant for online complexity), $T = 5000$.

The three tested predictive controllers yield the following average computational time for the high–variance noise simulation: GDPC with $\bar{u}$ as in (8) -59 ms; GDPC with $\bar{u}$ as in (16) -30 ms; DeePC -371 ms. This suggests that for high noise and large scale systems GDPC provides a computationally efficient alternative to DeePC.

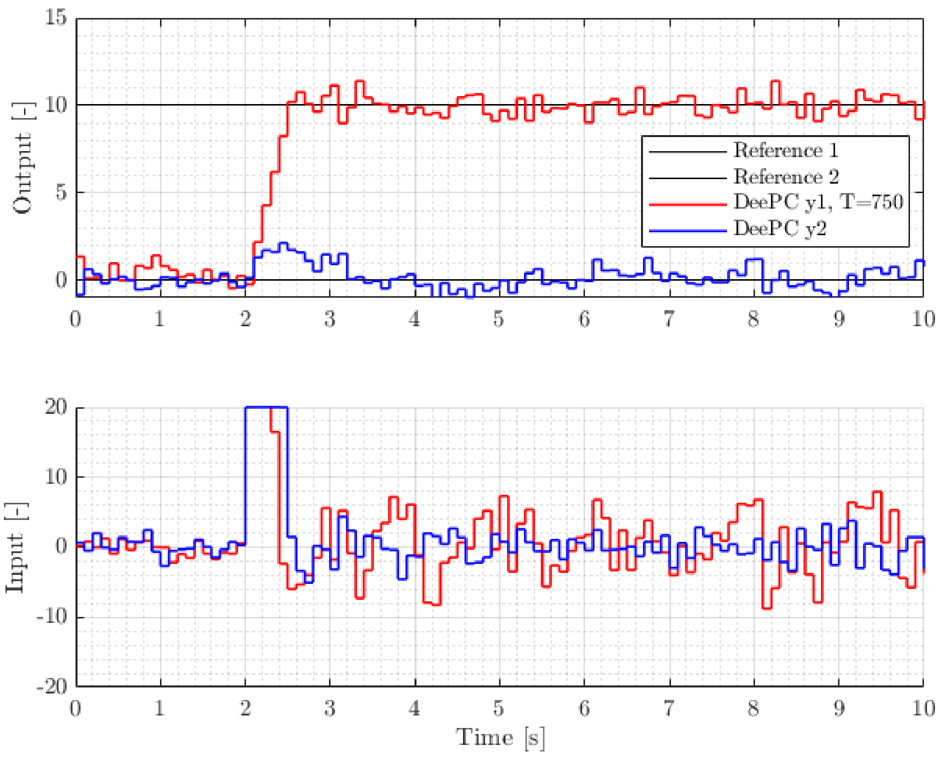

Figure 6. DeePC tracking performance: $T = 750$, $\sigma_w^2 I = 0.5I$.

4.3. Comparison with DeePC over Multiple Runs

In this section, we compare the performance of GDPC with DeePC for different sizes of T over multiple runs. The performance can be expressed using the integral squared error (ISE) or the integral absolute error (IAE) in relation with the input energy (InEn), formally defined as:

$$\text{ISE} = \sum_{k=T_{ini}}^{t_{\max}} \|y(k) - r(k)\|_2^2, \quad \text{IAE} = \sum_{k=T_{ini}}^{t_{\max}} \|y(k) - r(k)\|_1, \quad \text{InEn} = \sum_{k=T_{ini}}^{t_{\max}} \|u(k)\|_2^2,$$

where $t_{\max}$ is the simulation time and note that the performance scores are computed after the simulation ends, thus using simulated data (not predicted data). Furthermore, both the data-collecting experiment and the simulation are influenced by noise with variance $\sigma_w^2 I = 0.05I$. All algorithms are designed based on the same offline collected data set and tested online using the same noise realization.

As shown in Figure 7, we notice that the GDPC predictor with data length $T_g = 150$ (Hankel matrix based predictor) and $T = 1000$ (least squares based predictor) can match the performance of DeePC with data length $T = 500$, while the DeePC performance with $T = 150$ is lower. Also, the input energy is lower for GDPC compared with the the same cost for DeePC with $T = 500$. Two extreme outliers were removed from the results of DeePC with $T = 150$ to make the plots more clear.

From a computational complexity point of view, as shown in Table 2, the average CPU time of GDPC with $T_g = 150$ and $T = 500$ is of the same order as the average CPU time of DeePC with $T = 150$, while the average CPU time of DeePC with $T = 500$ is about 8 times higher.

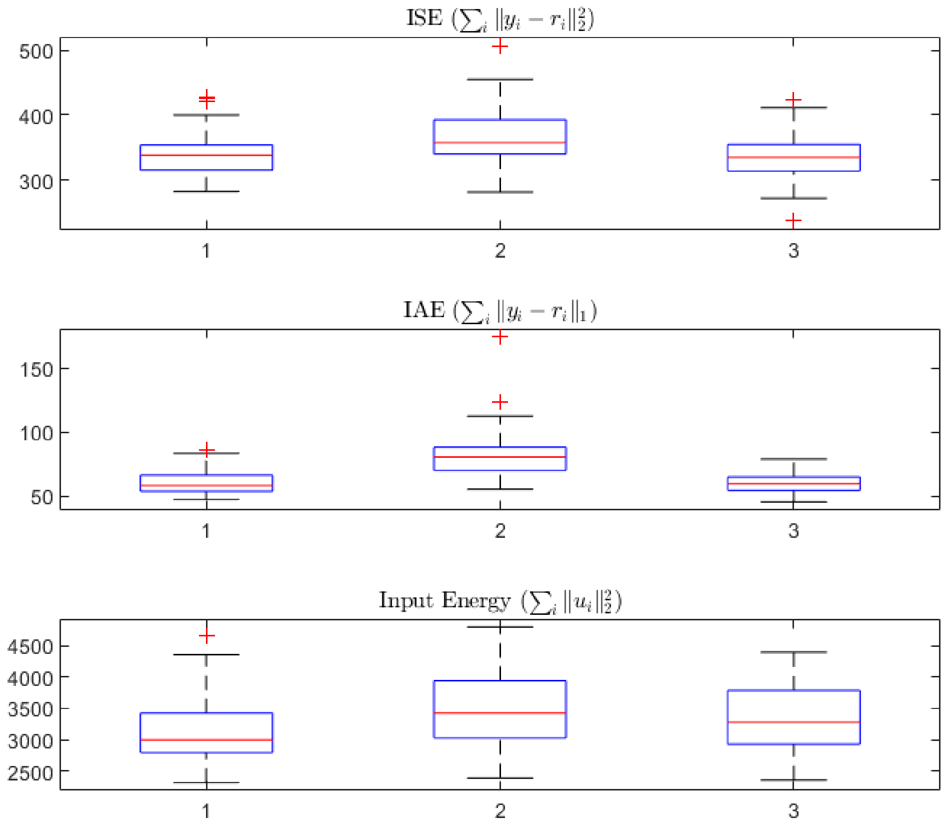

Figure 7. **1**: GDPC with $\bar{u}$ as in (8), $T = 1000$, $T_g = 150$; **2**: DeePC with $T = 150$; **3**: DeePC with $T = 500$. $\sigma_w^2 I = 0.05I$, using 50 Monte-Carlo runs.

Table 2. Average CPU time (Quadprog) over all runs for various data–driven predictive controllers.

	GDPC (8), $T_g = 150$	**DeePC $T = 150$**	**DeePC $T = 500$**
CPU	33 ms	32 ms	260 ms

The obtained results validate the fact that GDPC offers more flexibility to optimize the trade off between control performance in the presence of noisy data and online computational complexity. As such, GDPC provides engineers with a practical and robust data–driven predictive controller suitable for real–time implementation.

5. Conclusions

In this paper we developed a generalized data–driven predictive controller by merging a subspace unbiased predictor with a Hankel matrix based predictor. We have shown that this formulation results in a well–posed data–driven predictive controller with similar properties as DeePC. Sufficient conditions for closed–loop asymptotic stability of GDPC have been establsihed. Also, we have shown that compared to DeePC, the developed GDPC algorithm provides a trade–off between computational complexity and robust control performance.

To demonstrate this, Table 3 summarizes the performance versus computational complexity trade-off of the developed GDPC algorithms and DeePC, respectively, for system (20) and the following settings: $N = 50$, $\sigma_w^2 I = 0.2I$, $T = 5000$ and $T_g = 250$ for GPDC algorithms, and for DeePC we test $T = 250$ and $T = 500$. All algorithms have been designed using the same offline collected data set and tested online using the same noise realization.

Table 3. Average CPU time (Quadprog) and performance indicators for various DPC algorithms.

	GDPC with (8)	**GDPC with (16)**	**DeePC $T = 250$**	**DeePC $T = 500$**
CPU	131 ms	68 ms	106 ms	386 ms
ISE	290	258	397	265
IAE	86	74	159	75
InEn	3400	4400	3900	3400

We observe that the performance of DeePC increases / improves with the increase of the data length, but its computational complexity increases as well. For GDPC with a baseline input sequence as in (8), i.e., the shifted sequence from previous time, and with $T_g = 250$ outperforms DeePC for the same data size $T = 250$, with a slight computational time increase. Moreover, GDPC with a baseline input sequence as in (16), i.e., the unconstrained SPC control law, and with $T_g = 250$ outperforms DeePC even with $T = 500$, while it is computationally faster than DeePC with $T = 250$. The fact that GDPC with baseline (16) uses more input energy than DeePC with $T = 500$ to achieve improved control performance, suggests that the merged predictor in this case is superior, as it is able to correctly exploit the input energy for controlling the system, despite the large noise variance in both offline and online data.

In future work we will research systematic methods for choosing the data lengths T, T_g and tuning the hyper parameters λ_g and λ_σ. Also, we will research methods for providing robust stability guarantees for GDPC using the input-to-state stability framework.

Author Contributions: Methodology, M.L. and P.C.N.V.; Software, M.L. and P.C.N.V.; Formal analysis, M.L.; Writing original draft, M.L.; Review and editing, P.C.N.V. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Hou, Z.S.; Wang, Z. From model-based control to data-driven control: Survey, classification and perspective. *Inf. Sci.* **2013**, *235*, 3–35. [CrossRef]
2. Lamnabhi-Lagarrigue, F.; Annaswamy, A.; Engell, S.; Isaksson, A.; Khargonekar, P.; Murray, R.M.; Nijmeijer, H.; Samad, T.; Tilbury, D.; Van den Hof, P. Systems & Control for the future of humanity, research agenda: Current and future roles, impact and grand challenges. *Annu. Rev. Control* **2017**, *43*, 1–64.
3. Maciejowski, J.M. *Predictive Control with Constraints*; Prentice Hall: Harlow, UK, 2002.
4. Camacho, E.F.; Bordons, C. *Model Predictive Control*; Springer: Berlin/Heidelberg, Germany, 2007.
5. Favoreel, W.; De Moor, B.; Gevers, M. SPC: Subspace predictive control. In Proceedings of the 14th IFAC World Congress, Beijing, China, 5–9 July 1999; Volume 32, pp. 4004–4009.
6. Coulson, J.; Lygeros, J.; Dörfler, F. Data-Enabled Predictive Control: In the Shallows of the DeePC. In Proceedings of the 18th European Control Conference, Naples, Italy, 25–28 June 2019; pp. 307–312.
7. Willems, J.C.; Rapisarda, P.; Markovsky, I.; De Moor, B.L. A note on persistency of excitation. *Syst. Control. Lett.* **2005**, *54*, 325–329. [CrossRef]
8. Yang, H.; Li, S. A data–driven predictive controller design based on reduced Hankel matrix. In Proceedings of the 10th Asian Control Conference (ASCC), Kota Kinabalu, Malaysia, 31 May–3 June 2015; pp. 1–7.
9. Berberich, J.; Köhler, J.; Müller, M.A.; Allgöwer, F. Data-Driven Model Predictive Control With Stability and Robustness Guarantees. *IEEE Trans. Autom. Control* **2021**, *66*, 1702–1717. [CrossRef]
10. Fiedler, F.; Lucia, S. On the relationship between data–enabled predictive control and subspace predictive control. In Proceedings of the 2021 European Control Conference (ECC), Delft, The Netherlands, 29 June–2 July 2021; pp. 222–229.
11. Lazar, M. A Dissipativity-Based Framework for Analyzing Stability of Predictive Controllers. *IFAC PapersOnLine* **2021**, *54*, 159–165. [CrossRef]
12. Dorfler, F.; Coulson, J.; Markovsky, I. Bridging direct & indirect data-driven control formulations via regularizations and relaxations. *IEEE Trans. Autom. Control* **2022**, *68*, 883–897. [CrossRef]

13. Lazar, M.; Verheijen, P.C.N. Offset–free data–driven predictive control. In Proceedings of the 2022 IEEE 61st Conference on Decision and Control (CDC), Cancun, Mexico, 6–9 December 2022; pp. 1099–1104. [CrossRef]
14. Breschi, V.; Chiuso, A.; Formentin, S. Data-driven predictive control in a stochastic setting: A unified framework. *Automatica* **2023**, *152*, 110961. [CrossRef]
15. Zhang, K.; Zheng, Y.; Li, Z. Dimension Reduction for Efficient Data-Enabled Predictive Control. *arXiv* **2022**, arXiv:2211.03697. [CrossRef]
16. Baros, S.; Chang, C.Y.; Colón-Reyes, G.E.; Bernstein, A. Online data-enabled predictive control. *Automatica* **2022**, *138*, 109926. [CrossRef]
17. Elokda, E.; Coulson, J.; Beuchat, P.N.; Lygeros, J.; Dörfler, F. Data-enabled predictive control for quadcopters. *Int. J. Robust Nonlinear Control* **2021**, *31*, 8916–8936. [CrossRef] [PubMed]
18. Berberich, J.; Köhler, J.; Müller, M.A.; Allgöwer, F. Data-driven model predictive control: Closed-loop guarantees and experimental results. *at—Automatisierungstechnik* **2021**, *69*, 608–618. [CrossRef]
19. Carlet, P.G.; Favato, A.; Bolognani, S.; Dörfler, F. Data-Driven Continuous-Set Predictive Current Control for Synchronous Motor Drives. *IEEE Trans. Power Electron.* **2022**, *37*, 6637–6646. [CrossRef]
20. van Waarde, H.J.; De Persis, C.; Camlibel, M.K.; Tesi, P. Willems' Fundamental Lemma for State-Space Systems and Its Extension to Multiple Datasets. *IEEE Control Syst. Lett.* **2020**, *4*, 602–607. [CrossRef]
21. Verheijen, P.C.N.; Gonçalves da Silva, G.R.; Lazar, M. Data–driven rate–based integral predictive control with estimated prediction matrices. In Proceedings of the 25th International Conference on System Theory, Control and Computing (ICSTCC), Iasi, Romania, 20–23 October 2021; pp. 630–636.
22. Ljung, L. *System Identification: Theory for the User*, 2nd ed.; Prentice Hall PTR: Upper Saddle River, NJ, USA, 1999.
23. Berberich, J.; Köhler, J.; Müller, M.A.; Allgöwer, F. On the design of terminal ingredients for data-driven MPC. *IFAC–PapersOnLine* **2021**, *54*, 257–263. [CrossRef]
24. Stellato, B.; Banjac, G.; Goulart, P.; Bemporad, A.; Boyd, S. OSQP: An operator splitting solver for quadratic programs. *Math. Program. Comput.* **2020**, *12*, 637–672. [CrossRef]