2021 European Control Conference (ECC)
June 29 - July 2, 2021. Rotterdam, Netherlands

Establishing local strong accessibility of large-scale nonlinear systems by replacing the Lie algebraic rank condition

L. Gerard Van Willigenburg, Johannes D. Stigter, Jaap Molenaar[1]

1. Biometris Group of Wageningen University, P.O. Box 17, 6700 AA Wageningen, The Netherlands

E-mail: gerard.vanwilligenburg@wur.nl

Abstract: Within a recent development of algorithms to establish local structural identifiability, local observability and local strong accessibility of nonlinear systems, it turned out that *sensitivities*, governed by *linear time-varying dynamics*, are fundamental. As to local strong accessibility of nonlinear systems, the algorithm essentially checks controllability of linearizations along trajectories of the nonlinear system. In the literature concerning local controllability of nonlinear systems on the other hand, examples are regularly presented illustrating that controllability of linearizations is a *stronger* property than local strong accessibility. This paper clarifies these apparently contradicting results by using an important theoretical result from the literature and several illustrative examples. These reveal that the Lie algebraic rank condition (LARC), that is currently used to check local strong accessibility, may indeed be replaced by a rank condition based on sensitivities (SERC), that essentially checks controllability of linearizations along trajectories, provided that these trajectories are taken to be *non-singular*. This replacement is important for two reasons. One is that the computation of LARC requires a finite, but a-priori unknown number of steps, which may be very large. The other is especially important for large-scale nonlinear systems, for which the large number of symbolic differentiations involved in LARC results in excessive computation times or even renders the calculation infeasible. Both phenomena are also illustrated with examples.

Keywords: Nonlinear systems, local controllability, local strong accessibility, Lie algebraic rank condition (LARC), sensitivity rank condition (SERC), linearization, trajectories.

1 Introduction.

Controllability of a dynamic system concerns the ability to steer the system from an arbitrary state to any other, by means of control input signals applied to the system. This property ensures that proper control of such a system is always possible. If the system is linear, controllability is a property that holds globally while for nonlinear systems this may no longer be the case. Local definitions of controllability have been developed for nonlinear systems. Of these definitions local controllability (LC) and local strong accessibility (LSA) became especially important [1]. LSA has the advantage that it can be verified computationally through what is called the Lie Algebraic Rank Condition (LARC) [1]-[8]. This condition is necessary and sufficient for LSA, and necessary but *not* sufficient for LC [1]. Still, algorithms to compute LARC may be infeasible for two different reasons. Firstly, the computation of LARC requires an a-priori unknown number of steps, which may be very large [9]. Secondly, for large-scale nonlinear systems, the computation of LARC requires a large number of derivatives that have to be computed symbolically [1]. Therefore an alternative that is much more efficient computationally would be very attractive [9]-[13].

To establish local structural identifiability of nonlinear systems a very efficient algorithm was presented in [14]. The key idea to approach this problem is to calculate *sensitivities* along trajectories. These sensitivities are governed by *linear time-varying dynamics*. It turned out that the same approach is easily adapted to also establish local observability and LSA of nonlinear systems [15]. The analysis in [15], [16] reveals that the rank condition based on sensitivities (SERC) to establish LSA, essentially establishes controllability of linearizations (CL) along trajectories. But statements are regularly made in the literature that CL is a *stronger* property than LSA [7], [8]. There is even commonly used terminology for systems "having an uncontrollable linearization" while being LSA [17]-[21]. On the other hand, a considerably less known but important theoretical result from the literature states that LSA implies CL along *non-singular* trajectories. Moreover, trajectories are *generically* non-singular [22]-[24]. All this implies the following relations between the system properties LC, LSA and CL and satisfaction of SERC and LARC,

$$\mathrm{LC} \Rightarrow \mathrm{LSA} \Leftrightarrow \mathrm{LARC} \underset{\textit{generically}}{\overset{\Leftarrow}{\Rightarrow}} \mathrm{SERC} \Leftrightarrow \mathrm{CL} \qquad (1.1)$$

The relations (1.1) are further explained in section 2. By means of examples, in section 3, the generic equivalence between LARC and SERC is further illustrated together with problems to compute LARC. In section 4, using SERC, these problems are solved. For one large-scale example we show how SERC heavily outperforms LARC in terms of computational efficiency. The efficient algorithm by which we compute SERC is presented in a companion paper [16] that extends results presented in [14], [15].

2 Relating LC, LSA, LARC, SERC and CL.

Consider a nonlinear, continuous-time system represented in state-space form by,

$$\dot{x} = f\left(x(t), u(t)\right), \;\; f, x \in \mathbb{R}^n, \; u \in \mathbb{R}^m, \qquad (2.1)$$

In equations (2.1), t denotes continuous time, x is a vector containing the states and u a vector containing the controls of

the system. Furthermore f is an analytic vector function determining the system dynamics. Let,

$$u(t),\ x(t),\ t_0 \le t \le t_f, \tag{2.2}$$

denote a control and state trajectory satisfying differential equation (2.1) with t_0, t_f representing the initial and final time. SERC is based on sensitivities that propagate along a trajectory (2.2) according to *linear time-varying dynamics*. These dynamic equations are precisely those that describe the *linearization along the trajectory* (2.2) [15], [16]. Linearizations along trajectories are often conceived as approximations that hold close to the trajectory. The crucial insight provided by the development of SERC is that the linearization is *not* an approximation but an *exact description* of the propagation of *sensitivities* along trajectories [15], [16]. As a result SERC in (1.1), which we will use to determine LSA, also determines controllability of the linearization along the trajectory (CL) [15], [16]. This explains the equivalence between SERC and CL in (1.1).

Next consider LARC used to check LSA. To compute it, the system (2.1) is assumed to be linear in the input,

$$\begin{aligned} &f\left(x(t),u(t)\right) = f_0\left(x(t)\right) + \sum_{i=1}^{m} f_i\left(x(t)\right)u_i(t), \\ &f_i\left(x(t)\right) \in \mathbb{R}^n,\ i = 0,1,2,..,m. \end{aligned} \tag{2.3}$$

If the so called drift term $f_0\left(x(t)\right) = 0$ in (2.3), then LSA in (1.1) becomes equivalent to LC [1], [8]. LARC that verifies LSA is fully determined by the vector fields $f_i(x)$, $i = 1,2,..,m$. These determine directions in state-space in which the system can be steered locally [1], [8]. When evaluated at some $x = x_e$, vectors $f_i(x_e)$, $i = 1,2,..,m$ are obtained. If these together span $\mathbb{R}^n$ the system is LSA from $x = x_e$. If not, *additional directions* in which the system can be steered locally are obtained from so called Lie brackets operating on two vector fields. For some $i = 1,2,..,m$ and $j = 1,2,..,m$ the Lie bracket denoted by $\left[f_i(x), f_j(x)\right]$ operates on the two vector fields $f_i(x)$, $f_j(x)$ and generates from them another vector field of dimension n as follows,

$$\left[f_i(x), f_j(x)\right] = \frac{\partial f_j(x)}{\partial x} f_i(x) - \frac{\partial f_i(x)}{\partial x} f_j(x). \tag{2.4}$$

When the Lie Bracket (2.4) is evaluated at $x = x_e$, denoted by $\left[f_i(x), f_j(x)\right](x_e)$, this vector represents a possibly new direction in which the system can be steered locally at $x = x_e$. Every new vector field obtained from (2.4) may be used in consecutive Lie brackets (2.4) replacing either $f_i(x)$ or $f_j(x)$. One may *stop* computing additional vector fields if their evaluations at $x = x_e$ together span $\mathbb{R}^n$. Equivalently, the matrix with as columns evaluations of the vector fields at $x = x_e$ then has full rank n. In that case, at $x = x_e$, LARC is satisfied and the system is LSA from $x = x_e$. If LARC fails, i.e. if $\mathbb{R}^n$ is not spanned, one may also stop if any additional new vector field no longer increases the dimension of the space spanned by evaluations at $x = x_e$. Unfortunately, in general there is no known stop criterion for this [9], except when the system has no singular points [2] and the rank obtained from LARC is identical for each state. For this reason, computing LARC can be problematic and is also *very exhaustive*, since the number of vector fields obtained from recursive application of (2.4) grows heavily [1], [9], [25]. These problems will be illustrated and discussed by means of an example in section 3. To solve them, SERC will replace LARC in section 4.

An important difference between LARC and SERC is that LARC restricts computation to a *single state* $x = x_e$ for which it computes finitely many terms of a Fliess series expansion determined by Lie Brackets [7], [8]. SERC on the other hand considers sensitivities propagating along *a full trajectory* (2.2) containing $x = x_e$ as a state. This captures all terms of the Fliess series at once. A trajectory of an LSA system is called *non-singular* in [22] if the linearization along it is controllable. It is proved in [22] that trajectories of LSA systems are *generically* non-singular. This is represented by the important generic implication in (1.1) that justifies the replacement of LARC by SERC.

It is important to distinguish between the singular trajectories defined in [22], and singular trajectories on which LARC *itself* produces a reduced rank. The latter implies that LSA does not hold on the trajectory. In section 3, Example 2 illustrates that LARC and SERC also produce the same rank on such trajectories. This is proved in [16].

3 Examples illustrating SERC versus LARC.

Through examples, in this section, we further explore the *generic* equivalence in (1.1) of LARC with SERC/CL as presented in [15], [16]. Next we explore the *high efficiency* of SERC as compared to LARC. To further explore the generic equivalence of LARC with SERC/CL, consider a common, well-known example from the literature that is often used to argue *against* this equivalence.

Example 1 *(Classic car parking problem from [2])*

The dynamics of the car is given by (2.1) with,

$$f(x,u) = \begin{bmatrix} \cos(x_3 + x_4) \\ \sin(x_3 + x_4) \\ \sin(x_4) \\ 0 \end{bmatrix} u_1 + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u_2. \tag{3.1}$$

This is also described by equation (2.3) with,

$$f_0(x)=\begin{bmatrix}0\\0\\0\\0\end{bmatrix},\ f_1(x)=\begin{bmatrix}\cos(x_3+x_4)\\ \sin(x_3+x_4)\\ \sin(x_4)\\ 0\end{bmatrix},\ f_2(x)=\begin{bmatrix}0\\0\\0\\1\end{bmatrix}. \quad (3.2)$$

Thus the system is linear in the input, as is required for LARC. Common trajectories around which to linearize are steady-state trajectories. For the system (2.3), (3.2) each state is a steady state when one selects,

$$u=\begin{bmatrix}0\\0\end{bmatrix}. \quad (3.3)$$

Such a linearization yields,

$$\frac{df}{dx}=\begin{pmatrix}0 & 0 & -u_1\sin(x_3+x_4) & -u_1\sin(x_3+x_4)\\ 0 & 0 & u_1\cos(x_3+x_4) & u_1\cos(x_3+x_4)\\ 0 & 0 & 0 & u_1\cos(x_4)\\ 0 & 0 & 0 & 0\end{pmatrix}$$
$$\frac{df}{du}=\begin{bmatrix}\cos(x_3+x_4) & 0\\ \sin(x_3+x_4) & 0\\ \sin(x_4) & 0\\ 0 & 1\end{bmatrix} \quad (3.4)$$

In view of equations (3.3), (3.4), the time-invariant linearization around a steady state is described by,

$$\left.\frac{df}{dx}\right|_{x,\,u=\begin{bmatrix}0\\0\end{bmatrix}}=A(t)=A=\begin{bmatrix}0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{bmatrix},$$
$$\left.\frac{df}{du}\right|_{x,\,u=\begin{bmatrix}0\\0\end{bmatrix}}=B(t)=B=\begin{bmatrix}\cos(x_3+x_4) & 0\\ \sin(x_3+x_4) & 0\\ \sin(x_4) & 0\\ 0 & 1\end{bmatrix} \quad (3.5)$$

It is well known that to determine controllability of these time-invariant linearizations we may calculate the rank of the *controllability matrix* C given by [27],

$$C=\left[B, AB, A^2B, ..., A^{n-1}B\right],\ n=4. \quad (3.6)$$

Since $A=0$, we find that the rank of controllability matrix C in (3.6) equals the rank of B, which equals 2 according to equation (3.5). Therefore controllability matrix C does not have full rank 4, and all linearizations *around steady state trajectories* are indeed uncontrollable. On the other hand, LARC for any state comes out at 4 establishing LSA from each state [2]. The difference between these two results is, however, only caused by the *choice of trajectories* around which to linearize. From (3.3) these have the special property $u(t)=0,\ \forall t$ resulting in steady state trajectories. From df/dx in (3.4) observe that the choice $u_1=0$ "switches off" a large part of the linearized dynamics, destroying controllability of these. The steady state trajectories obtained in this way are called *singular trajectories* whereas the vast majority of trajectories is non-singular [22] causing the *generic* equivalence between LARC and SERC in equation (1.1) Therefore, if we select $u(t)$, $t_0 \le t \le t_f$ in (2.2) *arbitrarily*, steady state trajectories are not generally obtained but trajectories having linearizations that are controllable [15], [16]. In the car parking example, linearization around an *arbitrarily* selected trajectory *is* controllable as we will calculate in section 4. This reflects the generic equivalence in equation (1.1). If in doubt about the non-singularity of the trajectory, one could analyze several trajectories.

The next example concerns a nonlinear system having *singularities* for which there is no known termination criterion when computing LARC.

Example 2

Consider the system (2.3) with,

$$f_0(x)=\begin{bmatrix}0\\0\\x_1^2-1\end{bmatrix},\ f_1(x)=\begin{bmatrix}x_2^2\\0\\0\end{bmatrix},\ f_2(x)=\begin{bmatrix}0\\x_3^2\\0\end{bmatrix} \quad (3.7)$$

One can easily see that for states $x\neq[1,0,0]^T$, the time derivative $\dot{x}_3$ will be nonzero and therefore x_3 will become non-zero. With a non-zero u_2, also time derivative $\dot{x}_2$ and with it x_2 and x_2^2 will become non-zero, and finally, with a non-zero u_1, also time derivative $\dot{x}_1$ and with it x_1. Since x_1 determines the time derivative $\dot{x}_3$, the system is LSA from all states $x\neq[1,0,0]^T$. On the other hand, state $x=[1,0,0]^T$ is a steady state that moreover is unaffected by the control u. Therefore it is a singular state with zero as the outcome of LARC.

The algorithm to compute LARC, as presented in [2], is implemented in a Mathematica package called *ProPac*. It implements a stopping condition that relies on the assumption that no singular states exist. Although this assumption is violated, we still ran the algorithm from *ProPac* version 7.0 in Mathematica 11.2 on a PC having a 3.10 gigahertz Intel Core i5-8600 processor with operating system Windows 10. It gives 3 for the rank, implying that the system is LSA from each state. Apart from the singular state $x=[1,0,0]^T$ this is correct. The algorithm also offers the possibility to specify a specific state for evaluation. When specifying such states however, results do become erroneous for several states, as listed in Table 1. The *ProPac* algorithm then stops prematurely because the assumption underlying the stopping condition is invalid.

Since there is no known stopping condition in this case, the only thing one can do is to continue recursive computation of Lie brackets (2.4). We implemented an efficient algorithm to

do so in Mathematica and used it on the PC described above. The algorithm recursively computes so called P. Hall bases of increasing orders [25], [26] generated by the vector fields $f_i(x)$, $i = 0,1,..,m = 3$ in (3.7). After each increase of the order, it checks if the basis spans $R^n = R^3$. If so, the algorithm terminates. Otherwise it continues until a specified maximum order is reached. The P. Hall basis incorporates the anti-symmetry and Jacobi identity properties of Lie Brackets which significantly reduces the number of recursive Lie Bracket computations (2.4) without affecting the rank [25], [26]. To determine *LSA* from $x = x_e$, when checking the rank one must exclude the drift term $f_0(x)$ from these bases [8].

State:	$\begin{bmatrix}0\\0\\0\end{bmatrix}$	$\begin{bmatrix}1\\0\\0\end{bmatrix}$	$\begin{bmatrix}0\\1\\0\end{bmatrix}$	$\begin{bmatrix}0\\0\\1\end{bmatrix}$	$\begin{bmatrix}1\\1\\0\end{bmatrix}$	$\begin{bmatrix}1\\0\\1\end{bmatrix}$	$\begin{bmatrix}0\\1\\1\end{bmatrix}$
ProPac rank:	0	0	1	1	2	1	2
Correct rank:	3	0	3	3	3	3	3
Order:	15	1	3	7	5	4	3

Table 1: *Rank obtained from the ProPac algorithm [1] when specific states are specified. The correct rank and corresponding order of the P. Hall basis are also shown.*

Ord:	4	5	6	7	8	9
L:	29	77	193	505	1315	3499
CPU:	1.469	3.328	7.672	19.313	49.547	131
Ord:	10	11	12	13	14	15
L:	9379	25483	69703	192343	533827	1490403
CPU:	350	954	2636	7494	22463	73209

Table 2: *Number of Lie Brackets L and corresponding CPU times in seconds of P. Hall bases up to order 15 for Example 2 and state* $[0,0,0]^T$.

Then, for several states, Table 1 also records the order of the P. Hall basis needed to find the correct rank. The interesting case is $x = [0,0,0]^T$ where the appropriate rank only comes out for a P. Hall basis having a very high order of 15. Table 2 records the number of Lie Brackets and corresponding CPU times to compute P. Hall bases up to order 15 for Example 2 and state $[0,0,0]^T$. The number of Lie Brackets is 3 less than the number of elements of the P. Hall bases [26] since $f_i(x)$, $i = 0,1,2$ are the only 3 elements of each basis not obtained from Lie Bracketing. They are called the generators of each basis [25], [26]. As can be seen, with increasing order, the computations become very time consuming.

In the next section we will demonstrate how SERC, which will replace LARC, very efficiently computes correct results for all states, even if these are singular.

Example 3: *(High dimensional example of a truck with N trailers [28])*

This system concerns a truck with N trailers, where N can be selected arbitrarily. Index $k = 0$ refers to the truck and indices $k = 1,2,..,N$ to the N trailers. The state variables x_{2k+1}, $k = 0,1,..,N$ represent orientation angles of the axes of the truck and N trailers, respectively. There are two control variables u_1, u_2, being truck velocity and truck steering wheel angular velocity. Then the first order differential equations represented in state-space form (2.1) are given in recursive form by,

$$\begin{aligned}
&\dot{x}_1 = x_2,\ \dot{x}_2 = u_2,\ v_0 = u_1,\\
&v_k = v_{k-1}\cos(x_{2k-1} - x_{2k+1}) - M_{k-1}\sin(x_{2k-1} - x_{2k+1})x_{2k},\\
&\dot{x}_{2k+1} = x_{2k+2},\\
&\dot{x}_{2k+2} = \frac{v_k \sin(x_{2k-1} - x_{2k+1})}{L_k} - \frac{M_{k-1}\cos(x_{2k-1} - x_{2k+1})x_{2k}}{L_k},\ k = 1,2,..,N.
\end{aligned} \tag{3.8}$$

Here v_k is the velocity of truck/trailer k, L_k distance from the axle of the previous truck/trailer $k-1$ to the hitching point of trailer k, and M_k distance of the axle of trailer k to the hitching point of the same trailer. For this specific example, an analysis that does not require LARC could be used to prove that the system is LSA [28]. With SERC replacing LARC, the same result is obtained in the next section. Moreover, using SERC we will also establish LSA when only u_2 is used to control the system, taking $u_1 = 0$. Finally, using SERC we will find that using only u_1 to control the system, taking $u_2 = 0$, will fail to control states x_1 and x_2.

4 Replacing LARC with SERC

Using the efficient algorithm from [16], which is implemented in Matlab 2019, on the very same PC as described in section 3, we will now compute SERC to determine LSA. SERC determines the rank of a sensitivity matrix through a singular value decomposition (SVD). If the sensitivity matrix is not full rank, implying that one or more singular values are zero, this SVD also provides a so called "LSA signature". This signature displays the left singular vectors of the sensitivity matrix corresponding to zero singular values. These singular vectors represent directions in state-space along which the state cannot be changed. For details concerning the algorithms and their implementation, and the display of the "LSA signature" we refer to [15], [16]. In this section we only present and explain their outcome for the examples presented in the previous section, as well as CPU times.

For Examples 1-2, Table 3 lists the singular values of the sensitivity matrix. If these are not *numerically zero*, LSA is established. Roughly, singular values are considered

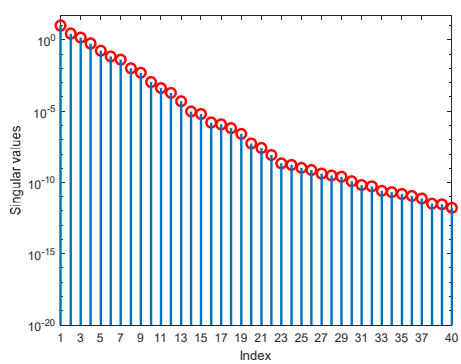
numerically zero if they are in the order of the machine precision (10^{-16}) or when the singular values contain a significant gap [29]. The singular values below such a gap are then considered to be numerically zero. From Table 3, LSA is correctly established in each case except for $x = [1,0,0]^T$ in Example 2. In that case the algorithm correctly produces only zero singular values corresponding to rank zero.

Example 1:	2.236806	2.235359	0.064377	0.022854
Example 2: $x \neq [1,0,0]^T$	1.723037	0.389012	0.017462	
Example 2: $x = [1,0,0]^T$	0	0	0	

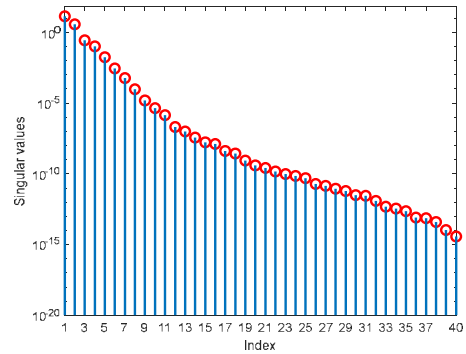
Table 3: *Singular values of the sensitivity matrix for Examples 1-2.*

For the large-scale Example 3, with $N = 19$ trailers and 40 state variables, Fig. 1 graphically represents the singular values of the sensitivity matrix on a logarithmic scale, for three different cases. The high efficiency of the algorithm results in computation times of only 11.6, 7.6 and 5.6 seconds. The results confirm LSA, except when only u_1 is used to control the system. In that case the two left singular vectors making up the LSA signature in Fig. 2 indicate that the state cannot be controlled in the directions x_1 and x_2. The latter implies that both truck position angle x_1 and its time-derivative $x_2 = \dot{x}_1$ cannot be controlled when truck velocity u_1 is the only control variable, while taking $\dot{x}_2 = u_2 = 0$.

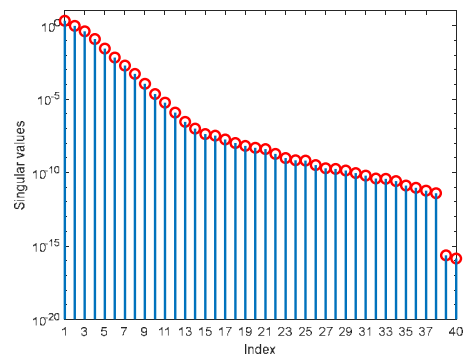
Fig. 1: *Singular values of the sensitivity matrix for Example 4 with $N = 19$ trailers and 3 different sets of control variables $\{u_1, u_2\}$, $\{u_2\}$, $\{u_1\}$ respectively. CPU times: 11.6, 7.6, 5.6 seconds respectively.*



Singular values when both u_1, u_2 are used for control

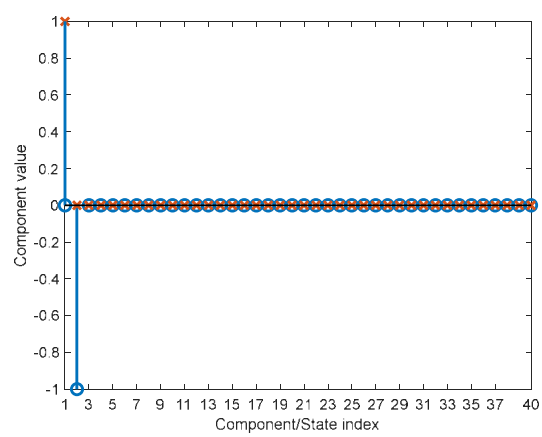


Singular values when only truck steering wheel angular velocity u_2 is used for control



Singular values when only truck velocity u_1 is used for control.

Fig. 2: *LSA signature corresponding to control with only u_1. Components of the two left singular vectors are shown; blue dots for the first and red crosses for the second singular vector. These indicate control is not possible in the x_1 and x_2 directions.*



5 Conclusions

As opposed to the Lie algebraic rank condition (LARC), the sensitivity rank condition (SERC) is computed with a very high efficiency. This makes SERC especially suitable for large-scale nonlinear systems as illustrated by Example 3 and in [16]. Satisfying SERC along a trajectory is equivalent to controllability of the linearization (CL) along that trajectory. Controllability of the linearization along an *arbitrarily* selected trajectory of a nonlinear system is generically equivalent to satisfying LARC for all states on that trajectory. The generic nature of the equivalence is caused by the fact that the equivalence only holds for *non-singular* trajectories, as defined in [22], which represent the vast majority. When in doubt about the non-singularity, several trajectories should be considered. It is important to distinguish between the singular trajectories defined in [22], which fail the equivalence between LARC and SERC/CL and singular trajectories on which LARC *itself* produces a reduced rank. The latter implies that local strong accessibility (LSA) does not hold on the trajectory. As shown by Example 2, when $x = [1,0,0]^T$, LARC and SERC also produce the same rank on such trajectories. A proof of this is presented in [16].

Using the well-known classic car parking problem, the *generic* equivalence between LARC and SERC/CL was illustrated. Ironically, this problem has so far been used to argue *against* the equivalence, which formally is correct when singular trajectories, mostly steady states, are considered too. For a very long time, this argument has probably withheld researchers in the systems and control community to use controllability of linearizations along trajectories (CL) to establish local strong accessibility (LSA) of nonlinear systems. The first author has been investigating properties of time-varying linearizations along trajectories for the purpose of designing linear perturbation (output) feedback controllers for nonlinear systems. Only when presented with an efficient algorithm to determine local structural identifiability of nonlinear systems, based on *sensitivities*, by the second and third author [14], the *generic* equivalence became apparent as confirmed by [22]. Considering sensitivities reveals that linearizations along trajectories are *exact descriptions* of these [15], [16], not approximations which they are when designing perturbation feedback controllers.

As to realizing nonlinear controllability in actual practice, control variations along a trajectory are much more easily realized than controls switching very fast over a very small time-interval. The former is approximately described by linearizations along trajectories while the latter underlies LARC.

References

[1] Imran M., Haitham T., Eisa S.A. Maqsood A. (2018). A controllability perspective of dynamic soaring. *Nonlinear Dynamics*, 94, 4, 2347–2362.

[2] Kwatny, H. G., & Blankenship, G. L. (2000). Nonlinear control and analytical mechanics: a computational approach. Boston: Birkhäuser.

[3] Sussmann H.J., Jurdjevic V. (1972). Controllability of nonlinear systems, *Journal of Differential Equations*, 12, 95-116.

[4] Hermann R., Krener A.J. (1977). Nonlinear Controllability and Observability, *IEEE Trans. Aut. Contr.*, AC-22, 5, 728-740.

[5] Hermes, H. (1982), On local controllability. *SIAM J. Contr. Opt.*, 20, 2, 211–220.

[6] Sussmann, H. J. (1987), A general theorem on local controllability. *SIAM J. Contr. Opt.* 25, 1, 158–194.

[7] A. Isidori (1995), Nonlinear Control Systems, 3rd ed. London: Springer-Verlag.

[8] Nijmeijer, H. and Van der Schaft, A. (1990), Nonlinear dynamical control systems, Vol. 175, Springer.

[9] A. Kawsky (2006). On the problem whether controllability is finitely determined, Proceedings of the 17th International Symposium on Mathematical Theory of Networks and Systems, Kyoto, Japan, July 24-28.

[10] V. Permana and R. Shoureshi (2010). Controllability and Observability of a Large Scale Thermodynamical System via Connectability Approach, *ASME 2010 Dynamic Systems and Control Conference*, Cambridge, Massachusetts, USA, September 12–15, Vol. 2, 217-224.

[11] Yang-Yu Liu, J.J. Slotine, A.L. Barabasi (2011). Controllability of complex networks, *Nature*, 473, 167-173

[12] N.J. Cowan, E.J. Chastain, D.A. Vilhena, J.S. Freudenberg, C.T. Bergstrom (2012). Nodal Dynamics, Not Degree Distributions, Determine the Structural Controllability of Complex Networks, *Plos One*, 2012, 7, 6, e38398.

[13] A.M. Hassan, H.E. Taha (2021). Design of a Nonlinear Roll Mechanism for Airplanes Using Lie Brackets for High Alpha Operation, *IEEE Trans. Aer. and Electr. Systems*, 57, 1, 462-475.

[14] J.D Stigter., J. Molenaar (2015). A fast algorithm to assess local structural identifiability, Automatica, 58, 118–124.

[15] J.D. Stigter, L.G. Van Willigenburg, J. Molenaar (2018). An Efficient Method to Assess Local Controllability and Observability for Non-Linear Systems, Preprints of the 9th Vienna International Conference on Mathematical Modelling, Vienna, Austria, February 21-23.

[16] L.G. Van Willigenburg, J.D. Stiger, J. Molenaar (2020), Sensitivity matrices as keys to local structural system properties of large-scale nonlinear systems. Submitted.

[17] Behtash S., Sastry S. (1988). Stabilization of Nonlinear Systems with Uncontrollable Linearization, *IEEE Trans. Aut. Contr.*, 33, 6, 585-591.

[18] Kim T., Abed E.H. (2001), Stationary bifurcation control of systems with uncontrollable linearization, *Int. J. Contr.*, 74, 5, 445 –452.

[19] Yang B., Lin W. (2005). Robust Output Feedback Stabilization of Uncertain Nonlinear Systems With Uncontrollable and Unobservable Linearization, *IEEE Trans. Aut. Contr.*, 50, 5, 619-630.

[20] Zhu G., Praly, J.L.L. (2007), Stabilization of an Electrostatic MEMS Including Uncontrollable Linearization, Proc. 46th IEEE CDC, New Orleans, LA, USA.

[21] Tall, I.A., Respondek W. (2010). Analytic Normal Forms and Symmetries of Strict Feedforward Control Systems, *Int. J. Robust Nonlinear Contr.*, 20, 1431-1454.

[22] E.D. Sontag (1992), Universal non-singular controls, *Systems & Control Letters*, 19, 221-224

[23] T. Fliess, T.S. Glad, 1993, An Algebraic Approach to Linear and Nonlinear Control, H. L. Trentelman *et al*. (eds.), Essays on Control, Springer Science+Business Media, New York.

[24] J.M. Coron (1994), Linearized control systems and applications to smooth stabilization, SIAM J. Contr. Opt., 32, 2, 358-386.

[25] I. Duleba (1997). Checking controllability of nonholonomic systems via optimal generation of P. Hall basis, IFAC Robot Control, Nantes, France.

[26] H. Munthe-Kaas, B. Owren (1999). Computations in a free Lie algebra, Phil. Trans. R. Soc. Lond. A, 357, 957-981.

[27] R.E. Kalman (1963). Mathematical description of linear systems, SIAM J. Contr., A, 1, 2, 152–192.

[28] C. Altafini, (1999). Controllability and singularities in the N-trailer system with kingpin hitching, Proceedings lFAC 14th Triennial World Congress, Beijing, 2169-2174.

[29] V.C. Klema, A.J. Laub, 1980, The Singular Value Decomposition: Its Computation and Some Applications, IEEE Trans. Aut. Contr., AC-25, 2, 164-176.