A Modified Sphere Decoder for Online Adjustment of the Switching Frequency

Qifan Yang*, Eyke Liegmann*, Petros Karamanakos†, and Ralph Kennel*
*Chair of Electrical Drive Systems and Power Electronics, Technical University of Munich, 80333 Munich, Germany
Email: *qifan.yang@tum.de, eyke.liegmann@tum.de, ralph.kennel@tum.de
†Faculty of Information Technology and Communication Sciences, Tampere University, 33101 Tampere, Finland
Email: †p.karamanakos@ieee.org

***Abstract*—In this paper we present a modified sphere decoder that can adjust the switching frequency in real time. To achieve this, the underlying integer least-squares (ILS) problem is reformulated and the lattice generator matrix is modified. By doing so, computational demanding operations required to be performed in real time are avoided altogether, rendering the proposed method computationally feasible. Moreover, the optimization process is kept computationally modest by appropriately manipulating the geometry of the ILS problem. The effectiveness of the introduced method is tested with a medium voltage variable speed drive system consisting of a three-level neutral point clamped (NPC) inverter and an induction machine.**

I. Introduction

In the recent years, model predictive control (MPC) has received increasing interest in the field of power electronics [1], [2]. MPC uses the plant model to predict its future behavior and a cost function that quantifies the control objectives. By doing so, the nonlinearities and constraints of the system can be included in a straightforward manner [3], thus fully exploiting the potential of the system. Moreover, direct MPC (DMPC)—which is the most popular MPC-based method in the power electronic community—exhibits a favorable dynamic performance owing to the direct manipulation of the converter switch positions [4].

The optimization problem underlying DMPC is an integer program, which is, in general, difficult to solve [5]. In power electronics the DMPC problem is most often solved with the brute-force approach of exhaustive search, i.e., the optimal control input is selected by evaluating all candidate solutions. This implies that the computational complexity increases exponentially with the prediction horizon. Because of this, the horizon is in most cases limited to one step. Nevertheless, it has been shown that long prediction horizons lead to a significant performance improvement under steady-state operation conditions, especially when considering higher order systems, like drives with LC filters [6].

The sphere decoding algorithm (SDA) introduced in [7] can be adopted to solve the long-horizon DMPC problem in a computational efficient manner and promising results have been shown, see, e.g. [7]–[11]. To this end, the DMPC problem has to be formulated as a (truncated) integer least-squares (ILS) one. Following, owing to its branch-and-bound nature [12], SDA effectively excludes suboptimal choices at the early stages of the search process. By doing so, the computational burden is significantly reduced and optimality is not sacrificed.

However, despite the fact that SDA can solve the optimization problem significantly more efficiently than the exhaustive search, long-horizon DMPC remains computationally challenging. In order to save computational power some demanding computations—required for the derivation of the ILS problem—have to be done offline [3]. In doing so, however, the weighting factor that decides on the trade-off between the tracking performance of the controller and the switching frequency has to be tuned offline since some entries of the matrix that define the ILS problem depend on it. As a result, this factor has to stay constant throughout the whole operation of the drive. The latter implies that the switching frequency cannot be adjusted among different operating points, something that may adversely affect the drive performance.

To overcome this issue, this paper proposes a modified sphere decoder that allows adjustment of the above-mentioned weighting factor in real time. To this aim, the formulation of the underlying ILS problem is modified such that the offline calculated matrices become independent of the weighting factor. By doing so, the switching frequency can be adjusted in real time without requiring lengthy and computationally demanding operations. Furthermore, by exploiting and manipulating the geometry of the formulated ILS problem the computational load of the problem can be kept modest. To highlight the potential benefits of the proposed control scheme a variable speed drive system, consisting of a three-level NPC voltage inverter and a medium-voltage (MV) induction machine (IM) is chosen as a case study.

II. Optimal Control Problem

In this section the formulation of the control problem is presented. For simplicity, a constant dc-link voltage and a fixed neutral point potential of the NPC inverter are assumed, as shown in Fig. 1. Moreover, the modeling of the system and the controller design are performed in the stationary orthogonal $\alpha\beta$-plane. Therefore, any variable $\boldsymbol{\xi}_{abc} = [\xi_a \ \xi_b \ \xi_c]^T$ in the abc-plane is mapped into the $\alpha\beta$-plane $\boldsymbol{\xi}_{\alpha\beta} = [\xi_\alpha \ \xi_\beta]^T$ through the transformation matrix $\boldsymbol{K}$, i.e., $\boldsymbol{\xi}_{\alpha\beta} = \boldsymbol{K}\boldsymbol{\xi}_{abc}$, with

$$\boldsymbol{K} = \frac{2}{3}\begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix}. \tag{1}$$

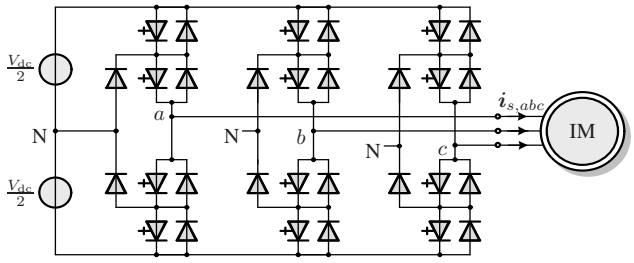


Fig. 1: Three-level three-phase neutral point clamped (NPC) voltage source inverter driving an induction motor (IM).

A. Control Model

Let $\boldsymbol{u}_{abc} = [u_a \ u_b \ u_c]^T$ denote the three-phase switch position of the NPC inverter, where $u_x \in \mathcal{U} = \{-1, 0, 1\}$, with $x \in \{a, b, c\}$, is the single-phase position. Given $\boldsymbol{u}_{abc}$ the three-phase inverter output voltage (equal to the applied stator voltage) is given by $\boldsymbol{v}_{abc} = [v_a \ v_b \ v_c]^T = (v_{\mathrm{dc}}/2)\boldsymbol{u}_{abc}$.

The dynamics of the IM can be described by [13][1]

$$\frac{\mathrm{d}\boldsymbol{i}_s}{\mathrm{d}t} = -\frac{1}{\tau_s}\boldsymbol{i}_s + \left(\frac{1}{\tau_r}\boldsymbol{I}_2 - \omega_r \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}\right)\frac{X_m}{D}\boldsymbol{\psi}_r + \frac{X_r}{D}\boldsymbol{v}_s \tag{2a}$$

$$\frac{\mathrm{d}\boldsymbol{\psi_r}}{\mathrm{d}t} = \frac{X_m}{\tau_r}\boldsymbol{i}_s - \frac{1}{\tau_r}\boldsymbol{\psi}_r + \omega_r \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}\boldsymbol{\psi}_r\,, \tag{2b}$$

where the stator current $\boldsymbol{i}_s = [i_{s\alpha} \ i_{s\beta}]^T$ and rotor flux $\boldsymbol{\psi}_r = [\psi_{r\alpha} \ \psi_{r\beta}]^T$ are the state variables, while the rotor angular speed ω_r is considered to be a time-varying parameter. For the definition of all parameters in (2) the reader is referred to [7].

For the derivation of the MPC problem in Section II-B, the model of the drive system in a state-space representation is required. By utilizing (2), and by choosing the state vector as $\boldsymbol{x} = [i_{s\alpha} \ i_{s\beta} \ \psi_{r\alpha} \ \psi_{r\beta}]^T$, and the three-phase switch position and the stator current as the system input and output, respectively, i.e., $\boldsymbol{u}_{abc} = [u_a \ u_b \ u_c]^T$ and $\boldsymbol{y} = [i_{s\alpha} \ i_{s\beta}]^T$, the state-space form of the drive is

$$\frac{\mathrm{d}\boldsymbol{x}(t)}{\mathrm{d}t} = \boldsymbol{F}\boldsymbol{x}(t) + \boldsymbol{G}\boldsymbol{u}_{abc}(t) \tag{3a}$$

$$\boldsymbol{y}(t) = \boldsymbol{C}\boldsymbol{x}(t), \tag{3b}$$

where the matrices $\boldsymbol{F}$, $\boldsymbol{G}$, and $\boldsymbol{C}$ are provided in the appendix.

By using exact discretization the discrete-time state-space model of the system is of the form

$$\boldsymbol{x}(k+1) = \boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{B}\boldsymbol{u}_{abc}(k) \tag{4a}$$

$$\boldsymbol{y}(k) = \boldsymbol{C}\boldsymbol{x}(k), \tag{4b}$$

with $k \in \mathbb{N}$, $\boldsymbol{A} = \mathrm{e}^{\boldsymbol{F}T_s}$, and $\boldsymbol{B} = \int_0^{T_s} \mathrm{e}^{\boldsymbol{F}\tau}\mathrm{d}\tau\,\boldsymbol{G}$.

B. Optimization Problem

The control objectives of the direct model predictive current control (DMPCC), see Fig. 2, relate to the (a) regulation of the stator current to its reference, and (b) the minimization of switching losses. Accordingly, the cost function that describes these objectives over a finite prediction horizon of length N_p time steps is defined as

$$J(k) = \sum_{l=k}^{k+N_p-1} ||\boldsymbol{y}_{\mathrm{err}}(l+1)||_2^2 + \lambda_u ||\Delta\boldsymbol{u}_{abc}(l)||_2^2\,. \tag{5}$$

The first term in (5) penalizes the output tracking error $\boldsymbol{y}_{\mathrm{err}}(l) = \boldsymbol{y}_{\mathrm{ref}}(l) - \boldsymbol{y}(l)$ within the prediction horizon, where $\boldsymbol{y}_{\mathrm{ref}}(k)$ is the stator current reference generated by the outer control loop. The second term, i.e., $\Delta\boldsymbol{u}_{abc}(l) = \boldsymbol{u}_{abc}(l) - \boldsymbol{u}_{abc}(l-1)$, penalizes the switching effort. Finally, the parameter $\lambda_u > 0$ is a weighting factor; it chooses an operating point on the trade-off curve by prioritizing between the two competing control objectives.

To find the optimal switching sequence $\boldsymbol{U}^*(k) = [\boldsymbol{u}_{abc}^{*T}(k) \ \boldsymbol{u}_{abc}^{*T}(k+1) \ \ldots \ \boldsymbol{u}_{abc}^{*T}(k+N_p-1)]^T$ that minimizes function (5), problem

$$\underset{\boldsymbol{U}(k)}{\text{minimize}} \ J(k) \tag{6a}$$

$$\text{subject to } \boldsymbol{x}(l+1) = \boldsymbol{A}\boldsymbol{x}(l) + \boldsymbol{B}\boldsymbol{u}_{abc}(l) \tag{6b}$$

$$\boldsymbol{y}(l+1) = \boldsymbol{C}\boldsymbol{x}(l+1) \tag{6c}$$

$$\boldsymbol{U}(k) \in \mathbb{U} \tag{6d}$$

$$\|\Delta\boldsymbol{u}_{abc}(l)\|_\infty \leq 1, \forall l = k, \cdots, k+N_p-1 \tag{6e}$$

needs to be solved in real time. In (6), $\boldsymbol{U}(k) = [\boldsymbol{u}_{abc}^T(k) \ \boldsymbol{u}_{abc}^T(k+1) \ \ldots \ \boldsymbol{u}_{abc}^T(k+N_p-1)]^T$ is the switching sequence within the prediction horizon, and $\mathbb{U}$ is the feasible set defined as $\mathbb{U} = \boldsymbol{\mathcal{U}}^{N_p} \subset \mathbb{Z}^n$, i.e., the N_p-times Cartesian product of the input set $\boldsymbol{\mathcal{U}} = \mathcal{U} \times \mathcal{U} \times \mathcal{U} = \mathcal{U}^3$, with $n = 3N_p$. Furthermore, (6e) introduces the switching constraint added to prevent a shoot-through in the converter by not allowing switching from $u_x = -1$ to 1, or vice versa, in one step [7]. Finally, in line with the receding horizon policy [14], only the first element of the solution, i.e., $\boldsymbol{u}_{abc}^*(k)$, is used as gating signals for the converter. At the next time step, the optimization problem is repeated over a shifted horizon with updated measurements and estimates.

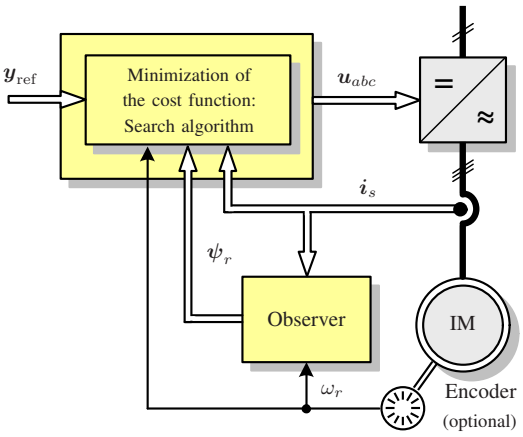


Fig. 2: Model predictive current control with reference tracking for the three-phase three-level NPC inverter with an IM.

[1]To simplify the notation, from this section on the subscript $\alpha\beta$ for vectors in the $\alpha\beta$-plane is omitted.

An efficient method to solve problem (6) is to formulate it as an ILS program. In the following, only some key points are summarized; for the detailed derivation see [7].

By successively using (4) it yields

$$\boldsymbol{Y}(k) = \boldsymbol{\Gamma}\boldsymbol{x}(k) + \boldsymbol{\Upsilon}\boldsymbol{U}(k), \tag{7}$$

where $\boldsymbol{Y}(k) = [\boldsymbol{y}^T(k+1)\ \boldsymbol{y}^T(k+2)\ \dots\ \boldsymbol{y}^T(k+N_p)]^T$ is the output trajectory. By introducing the unconstrained solution $\boldsymbol{U}_{\mathrm{unc}}(k)$, i.e., the solution to (6) when constraints (6d) and (6e) are neglected (i.e., $\boldsymbol{U}(k) \in \mathbb{R}^n$), and after some algebraic manipulations, function (5) can be written as

$$J(k) = (\boldsymbol{U}(k) - \boldsymbol{U}_{\mathrm{unc}}(k))^T \boldsymbol{H} (\boldsymbol{U}(k) - \boldsymbol{U}_{\mathrm{unc}}(k))\,, \tag{8}$$

with

$$\boldsymbol{H} = \boldsymbol{\Upsilon}^T\boldsymbol{\Upsilon} + \lambda_u \boldsymbol{S}^T\boldsymbol{S} \tag{9a}$$

$$\boldsymbol{U}_{\mathrm{unc}}(k) = -\boldsymbol{H}^{-1}\boldsymbol{\Theta}(k) \tag{9b}$$

$$\boldsymbol{\Theta}^T(k) = -\left(\boldsymbol{Y}_{\mathrm{ref}}(k) - \boldsymbol{\Gamma}\boldsymbol{x}(k)\right)^T\boldsymbol{\Upsilon} - \lambda_u(\boldsymbol{E}\boldsymbol{u}_{abc}(k-1))^T\boldsymbol{S}, \tag{9c}$$

where matrices $\boldsymbol{\Upsilon}$, $\boldsymbol{\Gamma}$, $\boldsymbol{S}$ and $\boldsymbol{E}$ are given in the appendix.

As $\boldsymbol{H}$ is (by definition) positive definite for $\lambda_u \in \mathbb{R}^{++}$, a unique invertible and lower triangular matrix $\boldsymbol{V} \in \mathbb{R}^{n\times n}$ exists, which satisfies

$$\boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{H}\,. \tag{10}$$

As shown in [7], matrix $\boldsymbol{V}$ can be computed by applying the Cholesky decomposition to $\boldsymbol{H}^{-1}$. By doing so, and with $\bar{\boldsymbol{U}}_{\mathrm{unc}}(k) = \boldsymbol{V}\boldsymbol{U}_{\mathrm{unc}}(k)$, (8) can be rewritten as

$$J(k) = (\boldsymbol{V}\boldsymbol{U}(k) - \bar{\boldsymbol{U}}_{\mathrm{unc}}(k))^T(\boldsymbol{V}\boldsymbol{U}(k) - \bar{\boldsymbol{U}}_{\mathrm{unc}}(k))\,. \tag{11}$$

Therefore, problem (6) can be stated as the (truncated) ILS problem

$$\underset{\boldsymbol{U}(k)\in\mathbb{U}}{\text{minimize}} \quad \left\|\bar{\boldsymbol{U}}_{\mathrm{unc}}(k) - \boldsymbol{V}\boldsymbol{U}(k)\right\|_2^2\,. \tag{12}$$

The solution to (8) is the lattice point[2] which has the smallest Euclidean distance ρ to the unconstrained solution. This point can be efficiently found by adopting the branch-and-bound algorithm referred to as sphere decoder [15], [16], see [7] for more details.

III. Sphere Decoder for Online Adjustment of the Switching Frequency

The computational cost of Cholesky decomposition of an $m \times m$ matrix is $\frac{1}{3}m^3$ floating-point operations (flops) [17]. As can be understood, the computational load of the Cholesky decomposition of $\boldsymbol{H}$—thus the computation of $\boldsymbol{V}$—can be very high, especially when long horizons are implemented. Moreover, given the very short sampling intervals adopted in direct control methods for power electronics, it can be concluded that the Cholesky decomposition of $\boldsymbol{H}$ in real time becomes very challenging, if not impossible. As a result, unless if the size of $\boldsymbol{H}$ remains reasonably small [11], the computation of the generator matrix $\boldsymbol{V}$ is performed offline [7]. The downside of this approach is that adjustment of the switching frequency (through λ_u, see (5)) in real time is impossible since there are entries of $\boldsymbol{H}$ (and $\boldsymbol{V}$) that depend on λ_u. In the remainder of this section a method that achieves online adjustment of the switching frequency with minimal additional computational overhead is presented.

A. Reformulation of the ILS Problem

To achieve online adjustment of the switching frequency the ILS problem (12) needs to be reformulated. The aim is to make the lattice generator matrix independent of λ_u such that it can still be computed offline. To this end, (8) is written as

$$\boldsymbol{H} = \underbrace{\boldsymbol{\Upsilon}^T\boldsymbol{\Upsilon} + \lambda_o\boldsymbol{S}^T\boldsymbol{S}}_{:=\boldsymbol{H}_1} + (\lambda_u - \lambda_o)\underbrace{\boldsymbol{S}^T\boldsymbol{S}}_{:=\boldsymbol{H}_2}\,. \tag{13}$$

As can be seen, in (13) the term $\lambda_o \in \mathbb{R}^{++}$ is introduced. This is done to turn the positive semidefinite matrix $\boldsymbol{H}_1$ [3] into a positive definite one[3]. Thus, since both $\boldsymbol{H}_1$ and $\boldsymbol{H}_2$ are by definition positive definite, two unique invertible and lower triangular matrices $\boldsymbol{R}_1 \in \mathbb{R}^{n\times n}$ and $\boldsymbol{R}_2 \in \mathbb{R}^{n\times n}$ exist such that

$$\boldsymbol{R}_1^T\boldsymbol{R}_1 = \boldsymbol{H}_1 \tag{14a}$$

$$\boldsymbol{R}_2^T\boldsymbol{R}_2 = \boldsymbol{H}_2\,. \tag{14b}$$

Combining (8), (13), and (14), it yields

$$\begin{aligned} J(k) &= (\boldsymbol{U}(k) - \boldsymbol{U}_{\mathrm{unc}}(k))^T\boldsymbol{R}_1^T\boldsymbol{R}_1(\boldsymbol{U}(k) - \boldsymbol{U}_{\mathrm{unc}}(k)) + \\ &\quad (\boldsymbol{U}(k) - \boldsymbol{U}_{\mathrm{unc}}(k))^T(\lambda_u - \lambda_o)\boldsymbol{R}_2^T\boldsymbol{R}_2(\boldsymbol{U}(k) - \boldsymbol{U}_{\mathrm{unc}}(k)) \\ &= (\boldsymbol{L}\boldsymbol{U}(k) - \widetilde{\boldsymbol{U}}_{\mathrm{unc}}(k))^T(\boldsymbol{L}\boldsymbol{U}(k) - \widetilde{\boldsymbol{U}}_{\mathrm{unc}}(k))\,, \end{aligned} \tag{15}$$

where $\boldsymbol{L} = \begin{bmatrix} \boldsymbol{R}_1 \\ \sqrt{\lambda_u - \lambda_o}\boldsymbol{R}_2 \end{bmatrix}$ is the new generator matrix and $\widetilde{\boldsymbol{U}}_{\mathrm{unc}}(k) = \boldsymbol{L}\boldsymbol{U}_{\mathrm{unc}}(k)$. With this, the ILS problem (12) can be rewritten as

$$\underset{\boldsymbol{U}(k)\in\mathbb{U}}{\text{minimize}} \quad \left\|\widetilde{\boldsymbol{U}}_{\mathrm{unc}}(k) - \boldsymbol{L}\boldsymbol{U}(k)\right\|_2^2\,. \tag{16}$$

As can be seen, problems (12) and (16) are equivalent. With the latter formulation, however, λ_u is removed from the offline calculated matrices $\boldsymbol{R}_1$ and $\boldsymbol{R}_2$. In doing so, the switching frequency can be adjusted in real time by appropriately tuning λ_u and updating $\boldsymbol{L}$ accordingly.[4] Therefore, the computationally demanding approach of recalculating $\boldsymbol{H}$ and $\boldsymbol{V}$ in (12) (via Cholesky decomposition) is avoided altogether.

[2]Matrix $\boldsymbol{V}$ generates a n-dimensional lattice—ergo its name "lattice generator matrix"—each point of which corresponds to a candidate solution, i.e., switching sequence $\boldsymbol{U}(k) \in \mathbb{U}$.

[3]The Cholesky decomposition is applicable only to positive definite matrices.

[4]It should be mentioned that $\lambda_u > \lambda_o$ is implied, meaning that the switching effort is by default penalized with a penalty greater than λ_o. This imposes a limit to the maximum achievable (average) switching frequency. Note also that the parameter λ_o is chosen offline. Its value affects the structure of the lattice, and, consequently, the computational complexity of the problem, as shown in Section IV.

B. Modified Sphere Decoder Algorithm

To solve problem (16) the sphere decoding algorithm proposed in [7] is refined, as shown in Algorithm 1. This algorithm operates on a search tree of n levels—generated by the set $\mathbb{U}$—the branches of which correspond to the elements of the sequence of control inputs $\boldsymbol{U}$. According to the sphere decoding principle, this search tree is traversed in a depth-first search manner and, consequently, a relatively tight upper bound is computed at the beginning of the optimization procedure. By doing so, the algorithm can visit only a small subset of $\mathbb{U}$ while optimality is still guaranteed. Specifically, only the candidate solutions inside a hypersphere (n-dimensional sphere) of radius ρ centered at the unconstrained solution $\widetilde{\boldsymbol{U}}_{\mathrm{unc}}(k)$, i.e.,

$$\left\| \widetilde{\boldsymbol{U}}_{\mathrm{unc}}(k) - \boldsymbol{L}\boldsymbol{U}(k) \right\|_2 \leq \rho(k)\,, \tag{17}$$

need to be evaluated.

With regards to the optimization process in question, matrix $\boldsymbol{L}$ generates two n-dimensional lattices. Thanks to the same lower-triangular structure of $\boldsymbol{R}_1$ and $\boldsymbol{R}_2$, the Euclidean distances d between $\boldsymbol{L}\boldsymbol{U}(k)$ and $\widetilde{\boldsymbol{U}}_{\mathrm{unc}}(k)$ in both lattices can be calculated in parallel and added together, see line 4 of Algorithm 1. Thus, the solution set $\boldsymbol{U}(k)$ can be built component by component, and at each step only a one-dimensional problem needs to be solved. If the intermediate distance d' exceeds the radius ρ, the branch is pruned and the remaining elements of the associated switching sequence are not explored; then the algorithm backtracks to find new paths towards the lower levels of the search tree. Once a complete switching sequence $\boldsymbol{U}(k)$ is constructed, while constraint (6e) is met, then it is considered as the tentative solution to (16). At this point, the radius of the hypersphere is tightened to d' and backtracking occurs so that previously unexplored–and yet not pruned—switching sequences are examined until the certificate (i.e., proof) of optimality is obtained. The latter implies that only the optimal solution (i.e., switching sequence $\boldsymbol{U}^*(k)$) remains inside the sphere and $\boldsymbol{u}^*_{abc}(k)$ is applied to the inverter. The algorithm is evoked with the following initial values of the arguments

$$\boldsymbol{U}^*(k) = \mathrm{ModSphDec}([\,], 0, 1, \rho^2_{\mathrm{ini}}(k), \widetilde{\boldsymbol{U}}_{\mathrm{unc}}(k)), \tag{18}$$

where $[\,]$ is the empty vector, and $\rho_{\mathrm{ini}}(k)$ is the initial radius of the hypersphere, computed based on an educated guess $\boldsymbol{U}_{\mathrm{ed}}(k)$, as introduced in (40) in [7], i.e.,

$$\rho_{\mathrm{ini}}(k) = \left\| \widetilde{\boldsymbol{U}}_{\mathrm{unc}}(k) - \boldsymbol{L}\boldsymbol{U}_{\mathrm{ed}}(k) \right\|_2 . \tag{19}$$

Algorithm 1 Modified Sphere Decoder

1: **function** MODSPHDEC($\boldsymbol{U}$, d^2, i, ρ^2, $\widetilde{\boldsymbol{U}}_{\mathrm{unc}}$)
2: **for** each $u \in \mathcal{U}$ **do**
3: $u_i \leftarrow u$
4: $d'^2 \leftarrow \|\widetilde{u}_{\mathrm{unc},i} - \boldsymbol{L}_{(i,1:i)}\boldsymbol{U}_{1:i}\|_2^2 + \|\widetilde{u}_{\mathrm{unc},n+i} - \boldsymbol{L}_{(n+i,1:i)}\boldsymbol{U}_{1:i}\|_2^2 + d^2$
5: **if** $d'^2 \leq \rho^2$ **then**
6: **if** $i < n$ **then**
7: ModSphDec($\boldsymbol{U}$, d'^2, $i+1$, ρ^2, $\widetilde{\boldsymbol{U}}_{\mathrm{unc}}$)
8: **else**
9: **if** $\boldsymbol{U}$ meets constraints (6e) **then**
10: $\boldsymbol{U}^* \leftarrow \boldsymbol{U}$
11: $\rho^2 \leftarrow d'^2$
12: **end if**
13: **end if**
14: **end if**
15: **end for**
16: **return** $\boldsymbol{U}^*$
17: **end function**

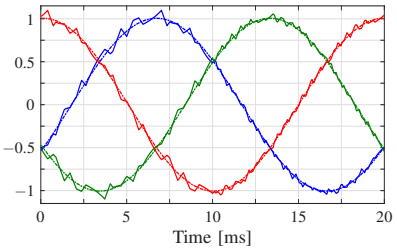
IV. Performance Evaluation

The simulation results presented in this section are obtained based on the MV drive system shown in Fig. 1. The three-level NPC has a constant dc-link voltage $V_{\mathrm{dc}} = 5.2\,\mathrm{kV}$. The squirrel cage IM has rated values $3.3\,\mathrm{kV}$, $356\,\mathrm{A}$, $2\,\mathrm{MVA}$, $50\,\mathrm{Hz}$ nominal frequency, and 0.25 per unit (p.u.) total leakage inductance. For all cases examined, the sampling interval $T_s = 25\,\mu\mathrm{s}$ and the prediction horizon $N_p = 10$ were used. Moreover, all results are shown in the p.u. system. The performance of the proposed sphere decoder is compared with the sphere decoder in [7], hereafter referred to as "standard sphere decoder".

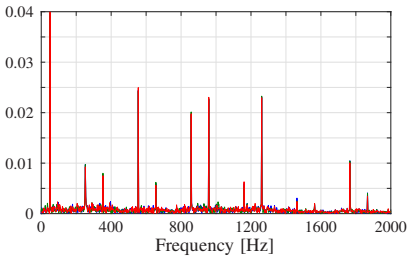
The effectiveness of the proposed method is examined both under steady state and torque transients. Fig. 3 shows the steady-state behavior of the drive system while operating at rated speed and torque at an average device switching frequency $f_{\mathrm{sw}} = 200\,\mathrm{Hz}$ ($\lambda_u = 0.15$). At $t = 10\,\mathrm{ms}$ λ_u is changed from 0.15 to 0.01. As a result, f_{sw} increases to $500\,\mathrm{Hz}$, and, consequently, the stator current total harmonic distortion (THD) changes from 5.46% to 3.00%.

Another advantage of the proposed method is that λ_u can be adjusted when the load condition varies to keep the same switching frequency at all operating points. Fig. 4(a) shows the stator current and Fig. 4(b) the switch positions during torque transients, i.e., when the reference torque changes from 1 to 0 p.u., and after a quarter of period, back to 1 p.u. (the torque reference is translated into the corresponding current steady-state reference). As can be seen, a switching frequency $f_{\mathrm{sw}} = 200\,\mathrm{Hz}$ can be maintained under different torque conditions by appropriately tuning λ_u. To this end, a lookup table can be built where the values of λ_u for different operating conditions (e.g. torque references) and f_{sw} are stored.

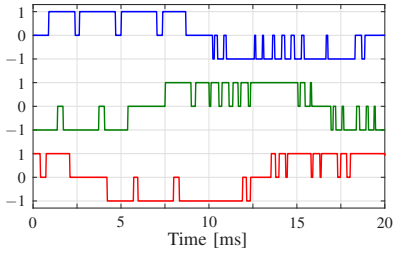
The computational complexity of an ILS problem depends on the condition of the lattice generator matrix [9]. To have a well-conditioned one, the columns of the lattice generator matrix should be close to orthogonal and of (relatively) small length [18]. The generator matrix of the modified sphere decoder consists of two parts, i.e., $\boldsymbol{R}_1$ and $\sqrt{\lambda_u - \lambda_o}\boldsymbol{R}_2$. Since $\boldsymbol{R}_2$ is by definition constant (in our case $\boldsymbol{R}_2 = \boldsymbol{S}$), only the columns of $\boldsymbol{R}_1$ can be manipulated (through λ_o) with the goal to meet the aforementioned criteria. To show the effect of λ_o on the structure of the lattice generator matrix,



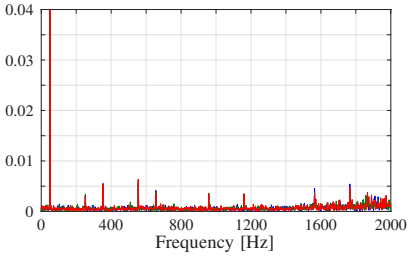
(a) Three-phase current, output (solid line) and its reference (dashed line).



(b) Stator current spectrum before λ_u adjustment, the THD is 5.46%.

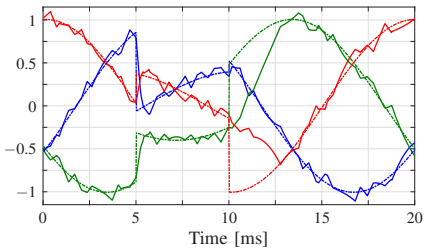


(c) Three-phase switch position.

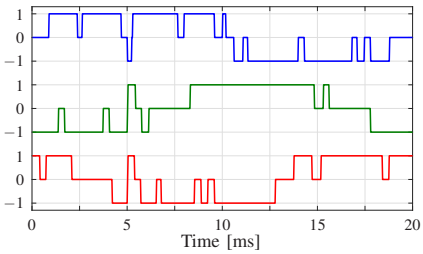


(d) Stator current spectrum after λ_u adjustment, the THD is 3.00%.

Fig. 3: Simulated waveform of steady-state operation with online adjustment of the switching frequency. For $t < 10\,\text{ms}$, the switching frequency f_{sw} is 200 Hz, whereas for $t \geq 10\,\text{ms}$ $f_{\text{sw}} = 500\,\text{Hz}$.

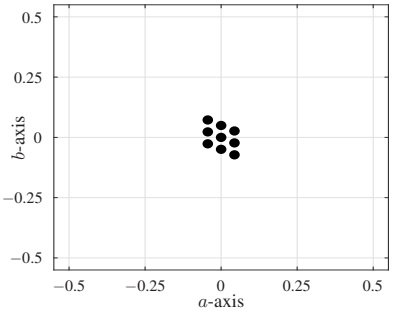


(a) Three-phase current, output (solid line) and its reference (dashed line).

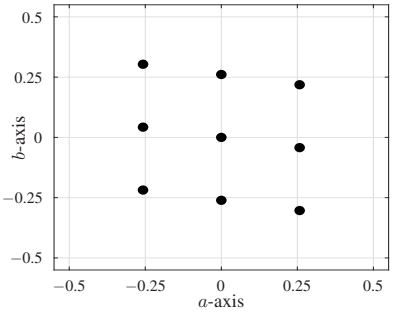


(b) Three-phase switch position.

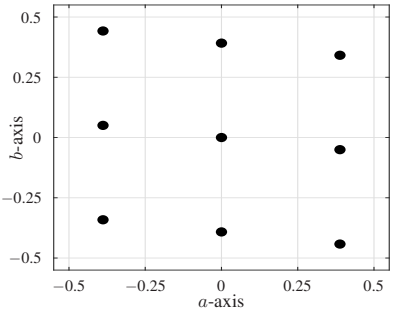
Fig. 4 Three-phase current and switch position during torque reference steps, In all operating points f_{sw} is 200 Hz by adjusting λ_u in real time.



(a) Generated by $\boldsymbol{R}_1$ with $\lambda_o = 0.001$.

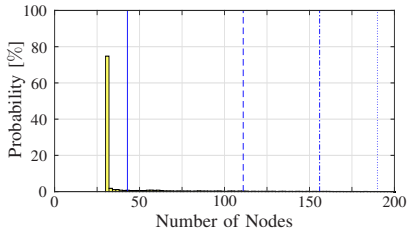


(b) Generated by $\boldsymbol{R}_1$ with $\lambda_o = 0.05$.

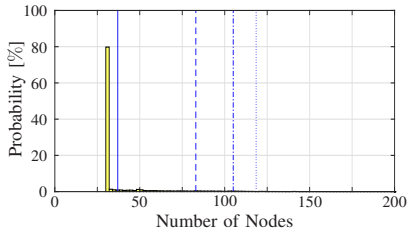


(c) Generated by $\boldsymbol{V}$.

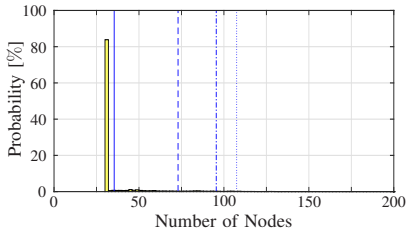
Fig. 5: The transformed coordinate system (lattice) of the first predicted step in ab-plane for $f_{\text{sw}} = 250\,\text{Hz}$.



(a) Modified sphere decoder with $\lambda_o = 0.001$.



(b) Modified sphere decoder with $\lambda_o = 0.05$.



(c) Standard sphere decoder.

Fig. 6: Probability distribution of the number of nodes explored by the sphere decoder algorithm for $f_{\text{sw}} = 250\,\text{Hz}$. The average number of nodes explored is indicated by the solid vertical line. The 95, 98, and 99 percentiles are shown as dashed, dashed-dotted, and dotted vertical lines, respectively.

and thus the computational complexity of (16), the following test is conducted. First, λ_u is set equal to 0.12 such that $f_{\text{sw}} = 250\,\text{Hz}$ results. By choosing $\lambda_o = 0.001$ the lattice generated by $\boldsymbol{R_1}$ is shown in Fig. 5(a), where the black solid circles correspond to the ab-phase switch positions of the first prediction step. For comparison purposes, the lattice generated by $\boldsymbol{V}$ (see (12)) when $\lambda_u = 0.12$ is shown in Fig. 5(c). As can be observed, the lattice in Fig. 5(a) is much more skewed than that in Fig. 5(c). Moreover, the lattice points in the former case are very close to each other. As a consequence, the hypersphere centered at $\widetilde{\boldsymbol{U}}_{\text{unc}}$ is very likely to include more lattice points when a lattice as in Fig. 5(a) exists. In other words, the sphere decoding algorithm has to evaluate more candidate solutions since the bounding is less effective. Thus, the modified ILS problem (16) with $\lambda_o = 0.001$ is ill-posed.

This conditioning of the problem, however, can be improved by increasing λ_o. In doing so, the diagonal entries of $\boldsymbol{H}_1$—and thus of $\boldsymbol{R}_1$—become relatively larger than the off-diagonal ones. Consequently, $\boldsymbol{R}_1$ and the lattice generated by it get closer to being orthogonal, as can be seen in Fig. 5(b) where λ_o is set to 0.05. As can be observed, the generated lattice is close to that shown in Fig. 5(c), implying that the search

TABLE I: Number of nodes explored by the modified and standard sphere decoder for $f_{\mathrm{sw}} = 250\,\mathrm{Hz}$.

Number of nodes explored	Modified $\lambda_o = 0.001$	Modified $\lambda_o = 0.05$	Standard
μ_a	43	37	35
μ_m	536	299	266

process is significantly sped up. The latter is verified in Table I, where the computational complexity—in terms of the number of nodes μ explored by the sphere decoder at each time step—of the aforementioned three cases is investigated. Specifically, the average μ_a and maximum μ_m number of the explored nodes are presented. As can be seen, the modified sphere decoder needs to visit slightly more nodes than the standard sphere decoder when $\lambda_o = 0.05$. Moreover, given that from an implementation point of view, the maximum number of nodes explored is the decisive quantity [9], it can be concluded that the 12% increase in the visited nodes is not detrimental. To further explore the computational complexity of the discussed cases, Fig. 6 illustrates the probability distribution of the number of nodes explored. The solid, dashed, dash-dotted and dotted lines indicate the 50, 95, 98, and 99 percentiles of the number of nodes explored, respectively. As can be observed, these lines are shifted towards the higher number of nodes in the case of the modified sphere decoder with $\lambda_o = 0.001$, see Fig. 6(a). On the other hand, when $\lambda_o = 0.05$, the percentiles exist at similar numbers as those of the standard sphere decoder, see Figs. 6(b) and 6(c), respectively.

V. Conclusions

This paper proposes a modified sphere decoder for online weighting factor adjustment. To this end, the integer least-square (ILS) problem is reformulated and a new lattice generator matrix is introduced. Thanks to the proposed method, the drive system can operate at the same average switching frequency over the whole operating region while avoiding demanding computations. Furthermore, as shown, by appropriately manipulating the lattice generator matrix of the modified ILS problem, the computational burden of the proposed sphere decoder is very close to that of the algorithm in [7]. It can be concluded that the proposed approach facilitates the implementation of the sphere decoding algorithm on a real-time system.

Appendix

The matrices of the continuous-time model (3) are

$$\boldsymbol{F} = \begin{bmatrix} -\frac{1}{\tau_s} & 0 & \frac{X_m}{\tau_r D} & \omega_r \frac{X_m}{D} \\ 0 & -\frac{1}{\tau_s} & -\omega_r \frac{X_m}{D} & \frac{X_m}{\tau_r D} \\ \frac{X_m}{\tau_r} & 0 & -\frac{1}{\tau_r} & -\omega_r \\ 0 & \frac{X_m}{\tau_r} & \omega_r & -\frac{1}{\tau_r} \end{bmatrix},$$

$$\boldsymbol{G} = \frac{v_{\mathrm{dc}}}{2}\frac{X_r}{D}\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}\boldsymbol{K},\ \boldsymbol{C} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}.$$

The matrices in (9) are

$$\boldsymbol{\Upsilon} = \begin{bmatrix} \boldsymbol{CB} & \boldsymbol{0}_{2\times 3} & \cdots & \boldsymbol{0}_{2\times 3} \\ \boldsymbol{CAB} & \boldsymbol{CB} & \cdots & \boldsymbol{0}_{2\times 3} \\ \vdots & \vdots & & \vdots \\ \boldsymbol{CA}^{N_p-1}\boldsymbol{B} & \boldsymbol{CA}^{N_p-2}\boldsymbol{B} & \cdots & \boldsymbol{CB} \end{bmatrix}, \boldsymbol{E} = \begin{bmatrix} \boldsymbol{I}_3 \\ \boldsymbol{0}_{3\times 3} \\ \boldsymbol{0}_{3\times 3} \\ \vdots \\ \boldsymbol{0}_{3\times 3} \end{bmatrix},$$

$$\boldsymbol{\Gamma} = \begin{bmatrix} \boldsymbol{CA} \\ \boldsymbol{CA}^2 \\ \vdots \\ \boldsymbol{CA}^{N_p} \end{bmatrix}, \boldsymbol{S} = \begin{bmatrix} \boldsymbol{I}_3 & \boldsymbol{0}_{3\times 3} & \cdots & \boldsymbol{0}_{3\times 3} \\ -\boldsymbol{I}_3 & \boldsymbol{I}_3 & \cdots & \boldsymbol{0}_{3\times 3} \\ \boldsymbol{0}_3 & -\boldsymbol{I}_3 & \cdots & \boldsymbol{0}_{3\times 3} \\ \vdots & \vdots & & \vdots \\ \boldsymbol{0}_{3\times 3} & \boldsymbol{0}_{3\times 3} & \cdots & \boldsymbol{I}_3 \end{bmatrix}.$$

References

[1] J. Rodriguez, J. Pontt, C. Silva, M. Salgado, S. Rees, U. Ammann, P. Lezana, R. Huerta, and P. Cortes, "Predictive control of three-phase inverter," *Electron. Letters*, vol. 40, no. 9, pp. 561–563, Apr. 2004.

[2] S. Vazquez, J. Rodriguez, M. Rivera, L. G. Franquelo, and M. Norambuena, "Model predictive control for power converters and drives: Advances and trends," *IEEE Trans. Ind. Electron.*, vol. 64, no. 2, pp. 935–947, Feb. 2017.

[3] T. Geyer, *Model Predictive Control of High Power Converters and Industrial Drives*. Hoboken, NJ: Wiley, 2016.

[4] J. Rodriguez, M. P. Kazmierkowski, J. R. Espinoza, P. Zanchetta, H. Abu-Rub, H. A. Young, and C. A. Rojas, "State of the art of finite control set model predictive control in power electronics," *IEEE Trans. Ind. Informat.*, vol. 9, no. 2, pp. 1003–1016, May 2013.

[5] L. A. Wolsey, *Integer Programming*. New York, NY: Wiley, 1998.

[6] T. Geyer, P. Karamanakos, and R. Kennel, "On the benefit of long-horizon direct model predictive control for drives with LC filters," in *Proc. IEEE Energy Convers. Congr. Expo.*, Pittsburgh, PA, Sep. 2014, pp. 3520–3527.

[7] T. Geyer and D. E. Quevedo, "Multistep finite control set model predictive control for power electronics," *IEEE Trans. Power Electron.*, vol. 29, no. 12, pp. 6836–6846, Dec. 2014.

[8] ——, "Performance of multistep finite control set model predictive control for power electronics," *IEEE Trans. Power Electron.*, vol. 30, no. 3, pp. 1633–1644, Mar. 2015.

[9] P. Karamanakos, T. Geyer, and R. Kennel, "A computationally efficient model predictive control strategy for linear systems with integer inputs," *IEEE Trans. Control Syst. Technol.*, vol. 24, no. 4, pp. 1463–1471, Jul. 2016.

[10] M. Dorfling, H. Mouton, P. Karamanakos, and T. Geyer, "Experimental evaluation of sphere decoding for long-horizon direct model predictive control," in *Proc. Eur. Power Electron. Conf.*, Warsaw, Poland, Sep. 2017, pp. P.1–P.10.

[11] P. Acuna, C. A. Rojas, R. Baidya, R. P. Aguilera, and J. E. Fletcher, "On the impact of transients on multistep model predictive control for medium-voltage drives," *IEEE Trans. Power Electron.*, vol. 34, no. 9, pp. 8342–8355, Sep. 2019.

[12] E. L. Lawler and D. Woods, "Branch-and-bound methods: A survey," *Operations Research*, vol. 14, no. 4, pp. 699–719, Jul./Aug. 1966.

[13] J. Holtz, "The representation of ac machine dynamics by complex signal flow graphs," *IEEE Trans. Ind. Electron.*, vol. 42, no. 3, pp. 263–271, Jun. 1995.

[14] J. B. Rawlings and D. Q. Mayne, *Model Predictive Control: Theory and Design*. Madison, WI: Nob Hill, 2009.

[15] U. Fincke and M. Pohst, "Improved methods for calculating vectors of short length in a lattice, including a complexity analysis," *Math. Comput.*, vol. 44, no. 170, pp. 463–471, Apr. 1985.

[16] B. Hassibi and H. Vikalo, "On the sphere-decoding algorithm I. Expected complexity," *IEEE Trans. Signal Process.*, vol. 53, no. 8, pp. 2806–2818, Aug. 2005.

[17] J. W. Demmel, *Applied Numerical Linear Algebra*. Philadelphia, PA: SIAM, 1997.

[18] P. Karamanakos, T. Geyer, and R. Kennel, "Constrained long-horizon direct model predictive control for power electronics," in *Proc. IEEE Energy Convers. Congr. Expo.*, Milwaukee, WI, Sep. 2016, pp. 1–8.