Combined Speed and Current Model Predictive Control with Inherent Field-Weakening Features for PMSM Drives

Saverio Bolognani*, Silverio Bolognani**,
Mauro Zigliotto*, Luca Peretti*

* Department of Technique and Management of Industrial Systems
** Department of Electrical Engineering
University of Padova, Italy

May 5th, 2008

Key idea of MPC

Model Predictive controllers rely on the fact that the available model of the system is rich and accurate enough to

Key idea of MPC

Model Predictive controllers rely on the fact that the available model of the system is rich and accurate enough to

1. predict the future behavior of the outputs

Key idea of MPC

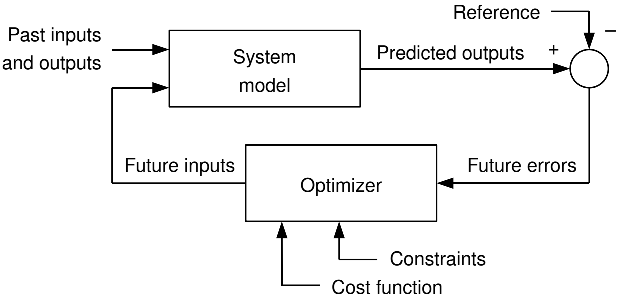
Model Predictive controllers rely on the fact that the available model of the system is rich and accurate enough to

1. predict the future behavior of the outputs
2. choose the inputs so that the future behavior
    - minimizes a given cost function
    - satisfies given constraints

Key idea of MPC

Model Predictive controllers rely on the fact that the available model of the system is rich and accurate enough to

1. predict the future behavior of the outputs
2. choose the inputs so that the future behavior
   - minimizes a given cost function
   - satisfies given constraints

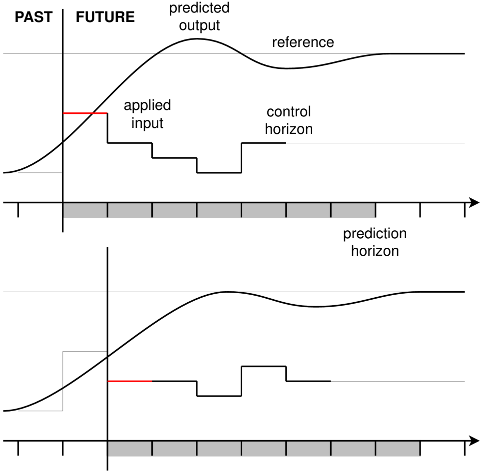


Receding horizon strategy

One of the key strategies of MPC is the receding horizon idea.

Receding horizon strategy

One of the key strategies of MPC is the receding horizon idea.



Mathematical framework

One of the mathematical framework that can be used for MPC is state space (linear) modeling with linear constraints.

$$\begin{aligned}\mathbf{x}(k+1) &= \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k)\\ \mathbf{y}(k) &= \mathbf{C}\mathbf{x}(k) + \mathbf{D}\mathbf{u}(k)\end{aligned}$$

Mathematical framework

One of the mathematical framework that can be used for MPC is state space (linear) modeling with linear constraints.

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

$$\mathbf{y}(k) = \mathbf{C}\mathbf{x}(k) + \mathbf{D}\mathbf{u}(k)$$

$$\mathbf{x} \in \mathbf{X} \subset \mathbb{R}^n, \mathbf{y} \in \mathbf{Y} \subset \mathbb{R}^p, \mathbf{u} \in \mathbf{U} \subset \mathbb{R}^m$$

Mathematical framework

One of the mathematical framework that can be used for MPC is state space (linear) modeling with linear constraints.

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

$$\mathbf{y}(k) = \mathbf{C}\mathbf{x}(k) + \mathbf{D}\mathbf{u}(k)$$

$$\mathbf{x} \in \mathbf{X} \subset \mathbb{R}^n, \mathbf{y} \in \mathbf{Y} \subset \mathbb{R}^p, \mathbf{u} \in \mathbf{U} \subset \mathbb{R}^m$$

The cost function that has to be minimized, generally takes the quadratic form

$$J_{N_p} = \sum_{j=k}^{k+N_p-1} \left[\mathbf{x}(j)^T \mathbf{Q}\mathbf{x}(j) + \mathbf{u}(j)^T \mathbf{R}\mathbf{u}(j) \right]$$

Mathematical framework

One of the mathematical framework that can be used for MPC is state space (linear) modeling with linear constraints.

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k)$$
$$\mathbf{y}(k) = \mathbf{C}\mathbf{x}(k) + \mathbf{D}\mathbf{u}(k)$$
$$\mathbf{x} \in \mathbf{X} \subset \mathbb{R}^n, \mathbf{y} \in \mathbf{Y} \subset \mathbb{R}^p, \mathbf{u} \in \mathbf{U} \subset \mathbb{R}^m$$

The cost function that has to be minimized, generally takes the quadratic form

$$J_{N_p} = \sum_{j=k}^{k+N_p-1} \left[\mathbf{x}(j)^T \mathbf{Q}\mathbf{x}(j) + \mathbf{u}(j)^T \mathbf{R}\mathbf{u}(j) \right]$$

This setup simplifies the resulting control law.

Fields of application

Model Predictive Control is particularly effective for plants that:

- can be modelled with a simple, known, system (possibly linear + time delays)

Fields of application

Model Predictive Control is particularly effective for plants that:

- can be modelled with a simple, known, system (possibly linear + time delays)
- constraints play a critical role in the system dynamics

Fields of application

Model Predictive Control is particularly effective for plants that:

- can be modelled with a simple, known, system (possibly linear + time delays)
- constraints play a critical role in the system dynamics
- have multiple inputs / outputs / states

Fields of application

Model Predictive Control is particularly effective for plants that:

- can be modelled with a simple, known, system (possibly linear + time delays)
- constraints play a critical role in the system dynamics
- have multiple inputs / outputs / states
- have long time constants

Fields of application

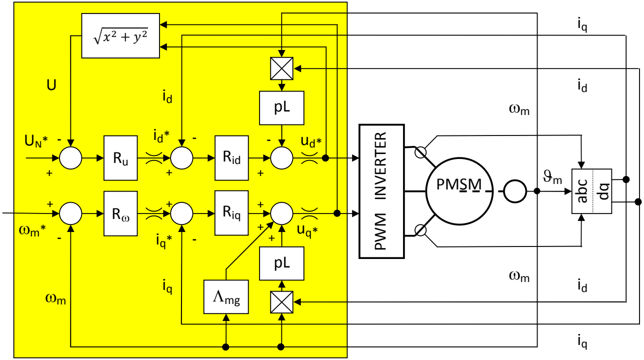
Model Predictive Control is particularly effective for plants that:

- can be modelled with a simple, known, system (possibly linear + time delays)
- constraints play a critical role in the system dynamics
- have multiple inputs / outputs / states
- have long time constants

Electrical drives fit in this category!

Once faster hardware and algorithm are available, MPC can be considered when designing an electrical motor drive, for example for a PMSM drive.

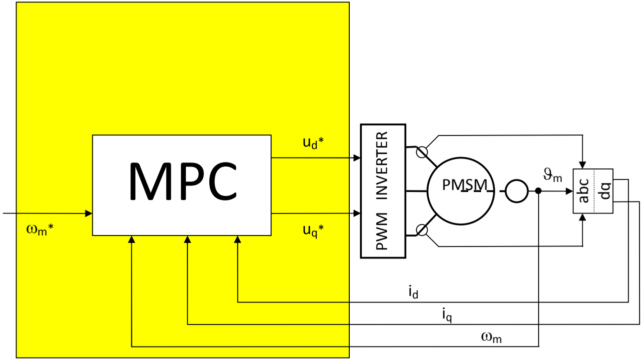
Controller structure



PID cascade controller

Classical solution: an inner loop for the current control and an outer loop for speed control. Bounds are enforced in every controller and flux weakening requires an external algorithm.

Controller structure



One MPC single controller

One only MPC controller is driven by the speed reference and commands the voltages that feed the motor. Bounds are enforced inside the controller.

Permanent Magnet Synchronous Motor equations

The electrical and mechanical dynamics of a PMSM motor is described by the following equations in the $(d - q)$ synchronous reference frame

Permanent Magnet Synchronous Motor equations

The electrical and mechanical dynamics of a PMSM motor is described by the following equations in the $(d - q)$ synchronous reference frame

$$\frac{\mathrm{d}i_d}{\mathrm{dt}} = \frac{1}{L}\left(u_d - Ri_d + \omega_{me} L i_q\right)$$
$$\frac{\mathrm{d}i_q}{\mathrm{dt}} = \frac{1}{L}\left(u_q - Ri_q - \omega_{me} L i_d - \omega_{me}\Lambda_{mg}\right)$$

Permanent Magnet Synchronous Motor equations

The electrical and mechanical dynamics of a PMSM motor is described by the following equations in the $(d-q)$ synchronous reference frame

$$\begin{aligned}
\frac{\mathrm{d}i_d}{\mathrm{dt}} &= \frac{1}{L}\left(u_d - Ri_d + \omega_{me}Li_q\right)\\
\frac{\mathrm{d}i_q}{\mathrm{dt}} &= \frac{1}{L}\left(u_q - Ri_q - \omega_{me}Li_d - \omega_{me}\Lambda_{mg}\right)\\
\frac{\mathrm{d}\omega_{me}}{\mathrm{dt}} &= \frac{p}{J}\left(k_t i_q - \frac{B}{p}\omega_{me} - \tau_L\right)
\end{aligned}$$

Permanent Magnet Synchronous Motor equations

The electrical and mechanical dynamics of a PMSM motor is described by the following equations in the $(d-q)$ synchronous reference frame

$$\begin{aligned}\frac{\mathrm{d}i_d}{\mathrm{dt}} &= \frac{1}{L}\left(u_d - Ri_d + \omega_{me}Li_q\right)\\ \frac{\mathrm{d}i_q}{\mathrm{dt}} &= \frac{1}{L}\left(u_q - Ri_q - \omega_{me}Li_d - \omega_{me}\Lambda_{mg}\right)\\ \frac{\mathrm{d}\omega_{me}}{\mathrm{dt}} &= \frac{p}{J}\left(k_t i_q - \frac{B}{p}\omega_{me} - \tau_L\right)\end{aligned}$$

which are continuous-time, non linear equations, even if iron saturation is assumed negligible (constant parameters), because of the motional coupling terms.

Linearized discrete–time model

The first step in MPC design consists in obtaining a linear, discrete–time model for the system. This mainly means getting rid of the motional coupling terms $\omega_{me} L i_d$ and $\omega_{me} L i_q$.

$$\mathbf{x} = \begin{bmatrix} i_d & i_q & \widehat{\omega_{me} i_q} & \omega_{me} \end{bmatrix}^T$$

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{dt}} = \begin{bmatrix} -\frac{R}{L} & 0 & 1 & 0 \\ -\omega_B & -\frac{R}{L} & 0 & -\frac{\Lambda_{mg}}{L} \\ 0 & 0 & 0 & 0 \\ 0 & \frac{pk_t}{J} & 0 & -\frac{B}{J} \end{bmatrix} \mathbf{x} + \begin{bmatrix} \frac{1}{L} & 0 \\ 0 & \frac{1}{L} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \mathbf{u}$$

Linearized discrete–time model

The first step in MPC design consists in obtaining a linear, discrete–time model for the system. This mainly means getting rid of the motional coupling terms $\omega_{me} L i_d$ and $\omega_{me} L i_q$.

$$\mathbf{x} = \begin{bmatrix} i_d & i_q & \widehat{\omega_{me} i_q} & \omega_{me} \end{bmatrix}^T$$

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{dt}} = \begin{bmatrix} -\frac{R}{L} & 0 & 1 & 0 \\ -\omega_B & -\frac{R}{L} & 0 & -\frac{\Lambda_{mg}}{L} \\ 0 & 0 & 0 & 0 \\ 0 & \frac{pk_t}{J} & 0 & -\frac{B}{J} \end{bmatrix} \mathbf{x} + \begin{bmatrix} \frac{1}{L} & 0 \\ 0 & \frac{1}{L} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \mathbf{u}$$

One can consider these terms as a measured disturbance. It has been done for $\omega_{me} L i_q$.

Linearized discrete–time model

The first step in MPC design consists in obtaining a linear, discrete–time model for the system. This mainly means getting rid of the motional coupling terms $\omega_{me} L i_d$ and $\omega_{me} L i_q$.

$$\mathbf{x} = \begin{bmatrix} i_d & i_q & \widehat{\omega_{me} i_q} & \omega_{me} \end{bmatrix}^T$$

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{dt}} = \begin{bmatrix} -\frac{R}{L} & 0 & 1 & 0 \\ -\omega_B & -\frac{R}{L} & 0 & -\frac{\Lambda_{mg}}{L} \\ 0 & 0 & 0 & 0 \\ 0 & \frac{pk_t}{J} & 0 & -\frac{B}{J} \end{bmatrix} \mathbf{x} + \begin{bmatrix} \frac{1}{L} & 0 \\ 0 & \frac{1}{L} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \mathbf{u}$$

One can consider these terms as a measured disturbance. It has been done for $\omega_{me} L i_q$. Something different has been done for the $\omega_{me} L i_d$ term, linearizing it around the base speed ω_B.

Linearized discrete–time model

The first step in MPC design consists in obtaining a linear, discrete–time model for the system. This mainly means getting rid of the motional coupling terms $\omega_{me} L i_d$ and $\omega_{me} L i_q$.

$$\mathbf{x} = \begin{bmatrix} i_d & i_q & \widehat{\omega_{me} i_q} & \omega_{me} \end{bmatrix}^T$$

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{dt}} = \begin{bmatrix} -\frac{R}{L} & 0 & 1 & 0 \\ -\omega_B & -\frac{R}{L} & 0 & -\frac{\Lambda_{mg}}{L} \\ 0 & 0 & 0 & 0 \\ 0 & \frac{pk_t}{J} & 0 & -\frac{B}{J} \end{bmatrix} \mathbf{x} + \begin{bmatrix} \frac{1}{L} & 0 \\ 0 & \frac{1}{L} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \mathbf{u}$$

One can consider these terms as a measured disturbance. It has been done for $\omega_{me} L i_q$. Something different has been done for the $\omega_{me} L i_d$ term, linearizing it around the base speed ω_B. Linearization then follows standard Euler techniques.

State augmentation for tracking

There are a couple of modification that are needed to achieve reference tracking.

State augmentation for tracking

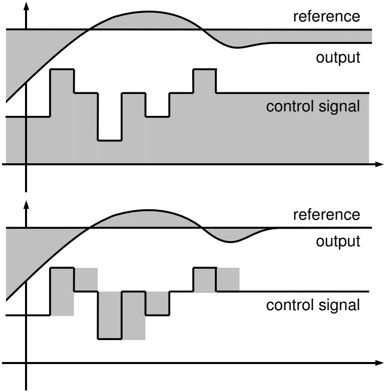
There are a couple of modification that are needed to achieve reference tracking.

- the reference ω_{me}^{ref} has to appear as a state variable (we'll assume it is constant in the prediction interval)

State augmentation for tracking

There are a couple of modification that are needed to achieve reference tracking.

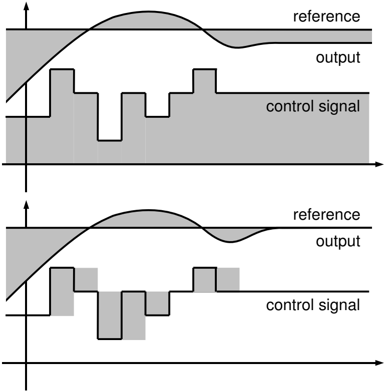
- the reference ω_{me}^{ref} has to appear as a state variable (we'll assume it is constant in the prediction interval)
- the input variation $\Delta\mathbf{u}$ is weighted instead of $\mathbf{u}$ (to achieve offset–free tracking)



State augmentation for tracking

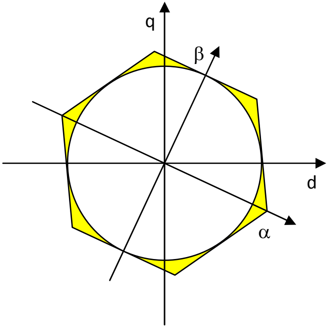
There are a couple of modification that are needed to achieve reference tracking.

- the reference ω_{me}^{ref} has to appear as a state variable (we'll assume it is constant in the prediction interval)
- the input variation $\Delta \mathbf{u}$ is weighted instead of $\mathbf{u}$ (to achieve offset–free tracking)



After this augmentation, the system state has dimension 7.

Input and state constraints

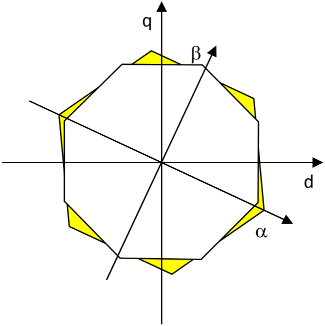


The second step consists in defining the input and state constraints of the system.

Due to inverter limitation, the voltage vector has to stay inside an hexagon which in fixed in the $\alpha - \beta$ reference frame. In the $d - q$ frame this can be approximated by a circular region.

.

Input and state constraints



The second step consists in defining the input and state constraints of the system.

Due to inverter limitation, the voltage vector has to stay inside an hexagon which in fixed in the $\alpha - \beta$ reference frame. In the $d - q$ frame this can be approximated by a circular region.

To have linear contraints in the $d - q$ frame, an octagonal region is adopted.

Cost function

Once the model of the system is fully defined, the cost function has to be tuned.

$$J_{N_p} = \sum_{j=k}^{k+N_p-1} \left[\mathbf{x}(j)^T \mathbf{Q} \mathbf{x}(j) + \mathbf{u}(j)^T \mathbf{R} \mathbf{u}(j) \right]$$

Cost function

Once the model of the system is fully defined, the cost function has to be tuned.

$$J_{N_p} = \sum_{j=k}^{k+N_p-1} \left[\mathbf{x}(j)^T \mathbf{Q} \mathbf{x}(j) + \mathbf{u}(j)^T \mathbf{R} \mathbf{u}(j) \right]$$

There are two main degrees of freedom

- the prediction horizon length N_p ($N_p = 5$ resulted to be a good tradeoff between enclosing most of the system dynamics and keeping the computational effort low)

Cost function

Once the model of the system is fully defined, the cost function has to be tuned.

$$J_{N_p} = \sum_{j=k}^{k+N_p-1} \left[\mathbf{x}(j)^T \mathbf{Q} \mathbf{x}(j) + \mathbf{u}(j)^T \mathbf{R} \mathbf{u}(j) \right]$$

There are two main degrees of freedom

- the prediction horizon length N_p ($N_p = 5$ resulted to be a good tradeoff between enclosing most of the system dynamics and keeping the computational effort low)
- the weigh matrices **Q** and **R**

Cost function

The matrices **Q** and **R** contain weighting coefficients for the state variables and the input signals.

Cost function

The matrices **Q** and **R** contain weighting coefficients for the state variables and the input signals.

- γ_{i_d}
  penalizes the direct current i_d, so that it is kept to zero unless it is really needed (i.e. for flux weakening)

Cost function

The matrices **Q** and **R** contain weighting coefficients for the state variables and the input signals.

- γ_{i_d}
- γ_{i_q}
  slightly penalizes the quadrature current i_q (mainly for stability purposes)

Cost function

The matrices **Q** and **R** contain weighting coefficients for the state variables and the input signals.

- γ_{i_d}
- γ_{i_q}
- $\gamma_{\omega_{me}}$

  weighs the speed error with respect to the gived speed reference

Cost function

The matrices **Q** and **R** contain weighting coefficients for the state variables and the input signals.

- γ_{i_d}
- γ_{i_q}
- $\gamma_{\omega_{me}}$
- $\gamma_{\Delta u_d}$ and $\gamma_{\Delta u_q}$
  weights the input signal variations (it has to be non–zero to have unique solution of the optimization problem)

Cost function

The matrices **Q** and **R** contain weighting coefficients for the state variables and the input signals.

- γ_{i_d}
- γ_{i_q}
- $\gamma_{\omega_{me}}$
- $\gamma_{\Delta u_d}$ and $\gamma_{\Delta u_q}$

These parameters are mainly tuned in a trial–and–error process based on simulations.

Affine state feedback control law

Finding the optimal inputs is Quadratic Programming

Having a linear system with linear constraints and a quadratic cost function, allows to solve the optimization problem with the standard tools of QP.

Affine state feedback control law

Finding the optimal inputs is Quadratic Programming

Having a linear system with linear constraints and a quadratic cost function, allows to solve the optimization problem with the standard tools of QP.

Moreover, the resulting control that is commanded (the first sample of the optimal input sequence) results to be an affine feedback of the state

$$\Delta \mathbf{u}_i(k) = \mathbf{F}_i\, \mathbf{x}(k) + \mathbf{G}_i$$

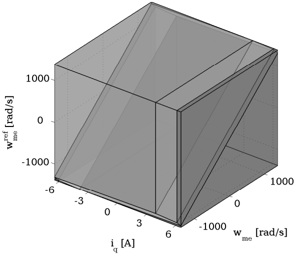
Affine state feedback control law

Finding the optimal inputs is Quadratic Programming

Having a linear system with linear constraints and a quadratic cost function, allows to solve the optimization problem with the standard tools of QP.

Moreover, the resulting control that is commanded (the first sample of the optimal input sequence) results to be an affine feedback of the state defined in different regions of the state space.

$$\Delta \mathbf{u}_i(k) = \mathbf{F}_i\, \mathbf{x}(k) + \mathbf{G}_i$$



Algorithm

The control algorithm then consists in

1. finding the active region
2. applying the corresponding feedback law

$$\Delta \mathbf{u}_i(k) = \mathbf{F}_i \, \mathbf{x}(k) + \mathbf{G}_i$$

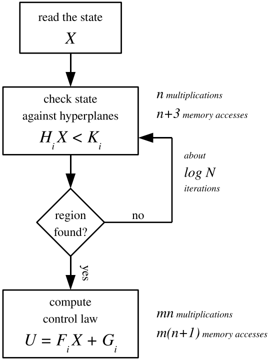
Algorithm

The control algorithm then consists in

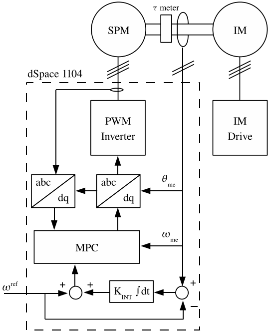
1. finding the active region
2. applying the corresponding feedback law

$$\Delta \mathbf{u}_i(k) = \mathbf{F}_i\,\mathbf{x}(k) + \mathbf{G}_i$$

It is not necessary to inspect all the regions! A dichotomic search can be done comparing the current state against a series of hyperplanes.
A binary search tree is built off-line. The online part of the algorithm is minimal!



Test bench



The MPC controller we just designed has been implemented on a dSpace 1104 fast prototyping board.

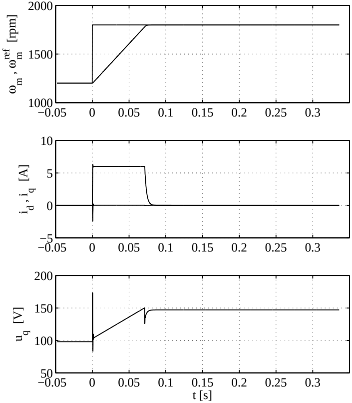
The *Multi Parametric Toolbox (MPT)* for *Matlab* provides both a solver for the optimization problem and some routines to generate the region partition and the binary search tree for simulation and actual implementation.

1st test

Speed reference step 1200 to 1800 rpm – no FW is needed.

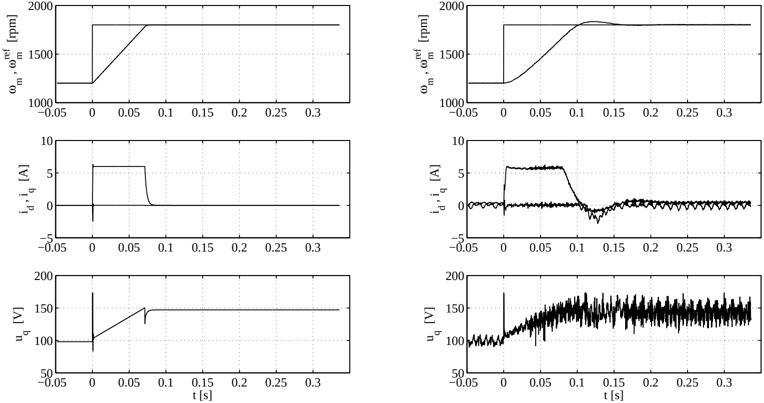
1st test

Speed reference step 1200 to 1800 rpm – no FW is needed.



1st test

Speed reference step 1200 to 1800 rpm – no FW is needed.

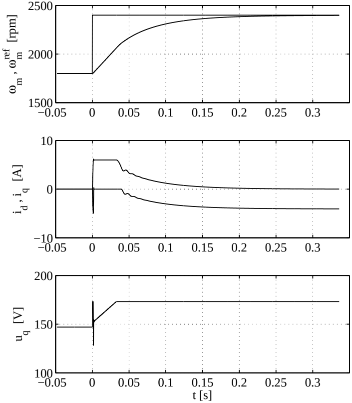


2nd test

Speed reference step 1800 to 2400 rpm – FW is needed.

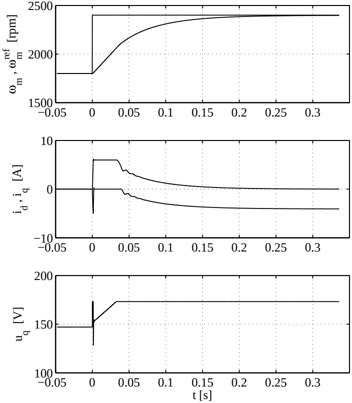
2nd test

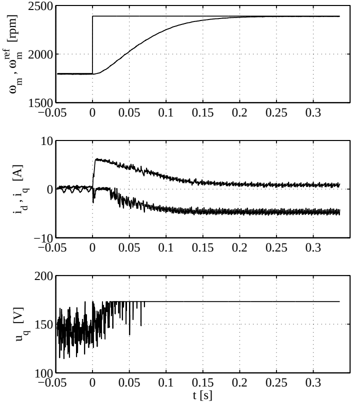
Speed reference step 1800 to 2400 rpm – FW is needed.



2nd test

Speed reference step 1800 to 2400 rpm – FW is needed.



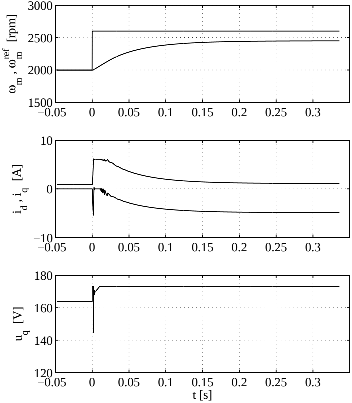


3rd test

Speed reference step 2000 to 2600 rpm – over speed limit.

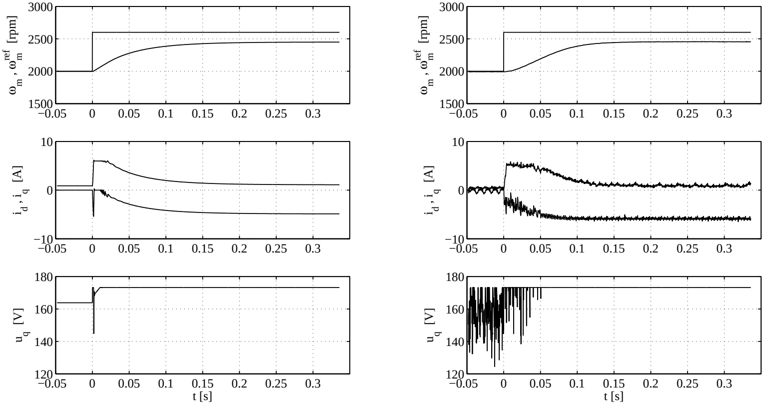
3rd test

Speed reference step 2000 to 2600 rpm – over speed limit.



3rd test

Speed reference step 2000 to 2600 rpm – over speed limit.



MPC for electrical drives

The proposed MPC controller for both speed and current of a PMSM drive, is able to command the motor both below and above base speed, by an effective and inherent flux weakening.

Details of the design, of the control algorithm generation, and of the actual implementation have been provided in the paper.

Simulation and experimental results prove the promising performance of the proposed controller.

Pros and Cons

Pros:

- one single controller effectively drives the motor in all its operating regions
- bounds are automatically enforced, with no need of saturations and anti–windup strategies
- more complex cost functions may be adopted
- reference and load dynamics can be added in the model

Pros and Cons

Pros:

- one single controller effectively drives the motor in all its operating regions
- bounds are automatically enforced, with no need of saturations and anti–windup strategies
- more complex cost functions may be adopted
- reference and load dynamics can be added in the model

Cons:

- the computational effort is higher than most of the conventional controllers
- drive and controller parameter changes require off-line algorithm tuning
- model mismatch has to be taken into account

Possible future works

This is the first complete application of MPC to current/speed control of a PMSM drive with flux weakening

Some interesting issues arose and are worth investigation.

Possible future works

This is the first complete application of MPC to current/speed control of a PMSM drive with flux weakening

Some interesting issues arose and are worth investigation.

- testing on different drives, SPM motors with wider flux weakening region, IPM motors

Possible future works

This is the first complete application of MPC to current/speed control of a PMSM drive with flux weakening

Some interesting issues arose and are worth investigation.

- testing on different drives, SPM motors with wider flux weakening region, IPM motors
- direct implementation with FPGA

Possible future works

This is the first complete application of MPC to current/speed control of a PMSM drive with flux weakening

Some interesting issues arose and are worth investigation.

- testing on different drives, SPM motors with wider flux weakening region, IPM motors
- direct implementation with FPGA
- stability and robustness analysis with respect to parameters mismatch

Thank you!