Journal of Process Control 24 (2014) 29–40

Contents lists available at ScienceDirect

Journal of Process Control

journal homepage: www.elsevier.com/locate/jprocont

Optimal control of grinding mill circuit using model predictive static programming: A new nonlinear MPC paradigm

Johan D. le Roux[a,*], Radhakant Padhi[b,1], Ian K. Craig[a]

[a] Department of Electrical, Electronic, and Computer Engineering, University of Pretoria, Pretoria, South Africa
[b] Indian Institute of Science Bangalore, Bangalore, India

ARTICLE INFO

Article history:
Received 14 April 2014
Received in revised form 15 October 2014
Accepted 15 October 2014
Available online 11 November 2014

Keywords:
Comminution
Grinding mill
Model predictive control
Model predictive static programming
Optimal control

ABSTRACT

The recently developed reference-command tracking version of model predictive static programming (MPSP) is successfully applied to a single-stage closed grinding mill circuit. MPSP is an innovative optimal control technique that combines the philosophies of model predictive control (MPC) and approximate dynamic programming. The performance of the proposed MPSP control technique, which can be viewed as a 'new paradigm' under the nonlinear MPC philosophy, is compared to the performance of a standard nonlinear MPC technique applied to the same plant for the same conditions. Results show that the MPSP control technique is more than capable of tracking the desired set-point in the presence of model-plant mismatch, disturbances and measurement noise. The performance of MPSP and nonlinear MPC compare very well, with definite advantages offered by MPSP. The computational speed of MPSP is increased through a sequence of innovations such as the conversion of the dynamic optimization problem to a low-dimensional static optimization problem, the recursive computation of sensitivity matrices and using a closed form expression to update the control. To alleviate the burden on the optimization procedure in standard MPC, the control horizon is normally restricted. However, in the MPSP technique the control horizon is extended to the prediction horizon with a minor increase in the computational time. Furthermore, the MPSP technique generally takes only a couple of iterations to converge, even when input constraints are applied. Therefore, MPSP can be regarded as a potential candidate for online applications of the nonlinear MPC philosophy to real-world industrial process plants.

1. Introduction

A grinding mill circuit forms a crucial part in the energy and cost-intensive comminution process of extracting valuable metals and minerals from mined ore. The ability of downstream processes to extract the greatest benefit from milled ore is dependent on the particle size distribution of the product that leaves the mill. In order to achieve the desired product specification in terms of quality and production rate, adequate control of the circuit is required. The usual control objectives for a grinding mill circuit are to improve the quality of the product, to maximise the throughput, to decrease the power consumption, to reduce the usage of grinding media and to improve process stability [1]. Yet, these objectives are interrelated and necessitates trade-offs to be made, i.e. both throughput and product particle size cannot be independently controlled to arbitrary set point values. The challenges when controlling a grinding process are the strong coupling between variables, large time delays, uncontrollable disturbances, the variation of parameters over time, the nonlinearities in the process and instrumentation inadequacies [2,3].

The majority of industrial mineral processing plants make use of proportional integral derivative (PID) controllers to achieve the objectives mentioned above even though the success of model-based controllers in other process industries are well attested [2,4]. Model-based controllers such as model predictive control (MPC) provides significant advantages over PID when applied to grinding mill circuits [5–8]. The economic performance of PID control compared to nonlinear MPC when applied to a grinding mill circuit was evaluated by Wei and Craig [9] and results showed that nonlinear MPC (NMPC) can improve performance with respect to recovered mineral value in downstream flotation circuits. Further improvements to overall MPC performance can be achieved by

* Corresponding author. Tel.: +27 12 420 2201; fax: +27 12 362 5000.
E-mail address: derik.leroux@up.ac.za (J.D. le Roux).
[1] This work was carried out while the author was a visiting professor at the University of Pretoria, South Africa.

http://dx.doi.org/10.1016/j.jprocont.2014.10.007
0959-1524/

incorporating peripheral control tools such as inferential measurements, disturbance observers or model-plant mismatch detection [10–13].

The following studies all make use of linearized models when applying MPC: Chen et al. [3], Niemi et al. [5], Pomerleau et al. [6], Ramasamy et al. [7], Remes et al. [8], Apelt and Thornhill [11], Yang et al. [12], Chen et al. [14]. Because of the highly nonlinear nature of a grinding process, the use of NMPC with fundamental nonlinear models is more desirable. Even though the modelling of comminution processes has improved over the past years [15], many of the available fundamental nonlinear models are not necessarily suitable for process control. These nonlinear models are mainly used for steady-state plant design and for a better understanding of load behaviour, breakage mechanisms and energy dissipation [16]. Moreover, the computational burden of detailed fundamental nonlinear models with large parameter sets and large state vectors increases the difficulty of developing feasible nonlinear model based controllers.

In Coetzee et al. [17], robust NMPC was applied to a grinding mill circuit in simulation and showed excellent results in the presence of large disturbances and parameter uncertainties. Even though the nonlinear model used in the study had the minimum number of states and parameters necessary for control and estimation purposes [18], the robust NMPC controller was not regarded suitable for online application unless computational time was significantly reduced.

In an effort to address the problem of computational cost for MPC, Bemporad et al. [19] showed that for a discrete-time linear time-invariant system, the control law can be obtained through a linear function instead of quadratic programming. Another technique for reducing the computational cost of MPC can be found in Wang and Boyd [20], where the dimensionality of the problem is reduced by restructuring the quadratic programs found in MPC and performing only a few iterations to solve the quadratic program with an appropriate interior-point method.

Recently, a novel suboptimal control design technique called model predictive static programming (MPSP) has been developed by Padhi and Kothari [21] for finite-horizon nonlinear problems with terminal constraints. This technique combines the philosophies of MPC and approximate dynamic programming to reduce a dynamic optimisation problem to a low dimensional static optimisation problem which significantly reduces computational complexity. Additional innovations are used such as the recursive computation of sensitivity matrices and using a closed form expression to update the control history to further reduce computational time. The computational effectiveness of the MPSP control technique is well illustrated in a number of problems in the aerospace industry, e.g. see Padhi and Kothari [21], Halbe et al. [22], Oza and Padhi [23], Bhitre and Padhi [24], Joshi and Padhi [25], Maity et al. [26]. The MPSP philosophy was recently extended by Kumar and Padhi [27] to include output tracking for infinite horizon nonlinear problems. Similar to NMPC, it was proposed that the MPSP technique can be applied with a receding horizon mechanism for output tracking problems.

This paper describes the mathematical formulation of MPSP for output tracking where the output is a nonlinear function of both the states and the input and applies it in simulation to a grinding mill circuit. The proposed MPSP technique is evaluated against a conventional NMPC in terms of the ability to reject noise and disturbances while tracking a desired setpoint. The aim of this paper is to illustrate the ability of MPSP to track a desired setpoint, in the presence of significant disturbances and model uncertainties, with similar performance to NMPC, but without the computational burden associated with it.

This study shows that improved output regulation can be achieved by means of MPSP compared to NMPC, primarily because the control horizon and prediction horizon are equal for MPSP. Note that even though the horizons are equal, this does not substantially increase the computational burden. MPSP converges to a solution within a few iterations, even in the presence of model-plant mismatch, disturbances and measurement noise. Although more iterations are necessary to converge to a solution when input constraints apply, convergence remains quite fast in general.

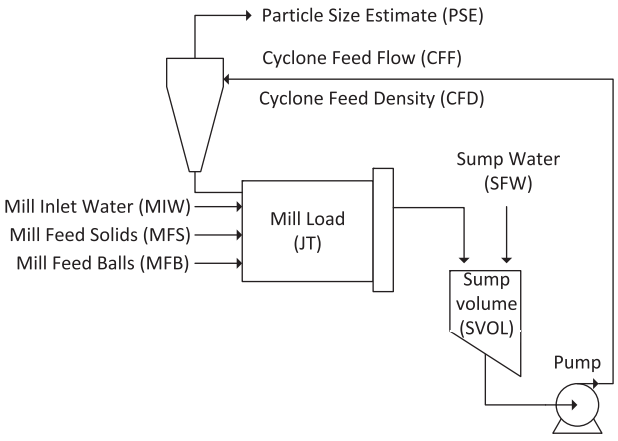

Fig. 1. A single-stage closed grinding mill circuit.

The rest of the paper is organized as follows: Section 2 of the paper discusses the nonlinear model used to describe the grinding mill circuit, Section 3 gives an overview of MPSP for nonlinear problems, Section 4 gives a brief description of the NMPC used in this study, Section 5 describes the simulation of both controllers and Section 6 discusses the results. Preliminary results of this technique applied to a grinding mill circuits over a finite time horizon was presented at the 19th IFAC World Congress held in Cape Town in August 2014 [28].

2. Milling circuit and model description

Both NMPC and MPSP are applied in simulation to a single-stage closed grinding mill circuit as described in this section. The three main elements of the circuit shown in Fig. 1 are the mill, sump and hydrocyclone. The mill receives four streams: mined ore, water, steel balls and underflow from the hydrocyclone. The ground ore in the mill mixes with water to create a slurry. For this circuit, the slurry in the mill is discharged through an end-discharge-screen where the aperture size of the end-discharge-screen limits the particle size of the discharged slurry. The discharged slurry is collected in a sump where it is diluted with water before it is pumped to the cyclone for classification. The hydrocyclone is responsible for the separation of the in- and out-of-specification ore discharged from the sump. The in-specification particles of the slurry pass to the overflow of the hydrocyclone, while the out-of-specification particles pass to the underflow. The underflow is passed back to the mill for the out-of-specification particles to be ground further. The overflow contains the final product which is passed to downstream processes [29]. The input and output variables of the circuit shown in Fig. 1 are described in Table 1.

The reduced complexity nonlinear dynamic model of Roux et al. [30] can be used to describe the circuit shown in Fig. 1. The approach in the derivation of this nonlinear phenomenological population balance model was to use as few fitted parameters as possible for reasonably accurate model responses in the correct direction.

The model uses six states to represent the constituents of the charge in the milling circuit. The six states are rocks, solids, coarse, fines, balls and water. Rocks are defined as ore too large to pass through the end-discharge screen, whereas solids are ore that can

Table 1
Description of circuit variables.

Input variables	
MIW	mill inlet water [m^3/h]
MFS	mill feed solids [t/h]
MFB	mill feed balls [t/h]
SFW	sump feed water [m^3/h]
CFF	cyclone feed flow-rate [m^3/h]
Output variables	
JT	mill total charge fraction [–]
SVOL	sump slurry volume [m^3]
PSE	particle size estimate [–]

Table 2
Description of subscripts.

Subscript	Description
$X_{\Delta-}$	f-feeder; m-mill; s-sump; c-cyclone
$X_{-\Delta}$	r-rocks; s-solids; c-coarse; f-fines; b-balls; w-water
$V_{--\Delta}$	i-inflow; o-outflow; u-underflow

be discharged from the mill. The solids consist of the sum of fine and coarse ore, where fine ore is smaller than the product specification size and coarse ore is larger than the product specification size. Although there are only three size classes used to describe the ore in the circuit (rocks, coarse ore and fine ore), these size classes are sufficient for a reasonably accurate model with responses in the correct directions [18]. The second last state, balls, represent the steel balls added to the mill to assist with grinding. The balls and rocks are the main grinding material and are only found in the mill as they are too large to pass through the apertures in the end-discharge screen. Finally, water is added to the ore to create a slurry which eases the transportation of ore through the circuit.

Only a brief overview of the model is given here. A detailed description of the model can be found in Roux et al. [30]. The model divides the circuit into four modules: a feeder, a semi-autogenous mill with an end-discharge screen, a sump and a hydrocyclone. For the model equations, V denotes a flow-rate in m^3/h and X denotes the states of the model as volumes in m^3. Table 2 provides a description of the subscripts for V and X. The first subscript indicates the module considered (feeder, mill, sump or cyclone), the second subscript specifies which of the six states are considered (rocks, solids, coarse, fines, balls, water), and in the case of flow-rates the final subscript shows if it is an inflow, overflow or underflow. The continuous time state-space description of the grinding mill circuit is shown below

$$
\begin{aligned}
\dot{X}_{mw} &= MIW - \frac{\varphi V_V X_{mw} X_{mw}}{X_{ms}+X_{mw}} + V_{cwu} \\
\dot{X}_{ms} &= \frac{MFS}{D_S}(1-\alpha_r) - \frac{\varphi V_V X_{mw} X_{ms}}{X_{ms}+X_{mw}} + V_{csu} + \frac{\varphi P_{mill}}{D_S \phi_r}\left(\frac{X_{mr}}{X_{mr}+X_{ms}}\right) \\
\dot{X}_{mf} &= \frac{MFS}{D_S}\alpha_f - \frac{\varphi V_V X_{mw} X_{mf}}{X_{ms}+X_{mw}} + V_{cfu} + \frac{P_{mill}}{D_S \phi_f} \Big/ \left[1 + \alpha_{\phi_f}\left(\frac{X_{mw}+X_{mr}+X_{ms}+X_{mb}}{v_{mill}} - v_{P_{\max}}\right)\right] \\
\dot{X}_{mr} &= \frac{MFS}{D_S}\alpha_r - \frac{\varphi P_{mill}}{D_S \phi_r}\left(\frac{X_{mr}}{X_{mr}+X_{ms}}\right) \\
\dot{X}_{mb} &= \frac{MFB}{D_B} - \frac{\varphi P_{mill}}{\phi_b}\left(\frac{X_{mb}}{D_S(X_{mr}+X_{ms}) + D_B X_{mb}}\right) \\
\dot{X}_{sw} &= \frac{\varphi V_V X_{mw} X_{mw}}{X_{ms}+X_{mw}} - \frac{CFFX_{sw}}{X_{sw}+X_{ss}} + SFW \\
\dot{X}_{ss} &= \frac{\varphi V_V X_{mw} X_{ms}}{X_{ms}+X_{mw}} - \frac{CFFX_{ss}}{X_{sw}+X_{ss}} \\
\dot{X}_{sf} &= \frac{\varphi V_V X_{mw} X_{mf}}{X_{ms}+X_{mw}} - \frac{CFFX_{sf}}{X_{sw}+X_{ss}}
\end{aligned}
\tag{1}
$$

Table 3
Circuit parameter values and uncertainty.

Parm	Value	Δ	Description
α_f	0.05	50%	Fraction fines in the ore
α_r	0.47	50%	Fraction rock in the ore
α_P	1.0		Fractional power reduction per fractional reduction from maximum mill speed
α_{ϕ_f}	0.01		Fractional change in kW/fines produced per change in fractional filling of mill
α_{speed}	0.71		Fraction of critical mill speed
α_{su}	0.87	5%	Parameter related to fraction solids in underflow
C_1	0.6		Constant
C_2	0.7		Constant
C_3	4.0		Constant
C_4	4.0		Constant
δ_{P_S}	0.5		Power-change parameter for fraction solids in the mill
δ_{P_V}	0.5		Power-change parameter for volume of mill filled
D_B	7.85		Density of steel balls [t/m^3]
D_S	3.2		Density of feed ore [t/m^3]
ε_{sv}	0.6		Max fraction solids by volume of slurry at 0 slurry flow
ε_c	129	5%	Parameter related to coarse split [m^3/h]
ϕ_b	90.0	5%	Steel abrasion factor [kWh/t]
ϕ_f	29.5	50%	Power needed per tonne of fines produced [kWh/t]
ϕ_r	6.00	20%	Rock abrasion factor [kWh/t]
$\varphi_{P_{\max}}$	0.57		Rheology factor for maximum mill power draw
$P_{\max}$	1662		Maximum mill motor power draw [kW]
v_{mill}	100		Mill volume [m^3]
$v_{P_{\max}}$	0.34		Fraction of mill volume filled for maximum power draw
V_V	84.0		Volumetric flow per "flowing volume" driving force [h^{-1}]
χ_P	0		Cross-term for maximum power draw

where X_{mw}, X_{ms}, X_{mf}, X_{mr} and X_{mb} are the volume of water, solids, fines, rocks and balls within the mill respectively, X_{sw}, X_{ss} and X_{sf} are the volume of water, solids and fines within the sump respectively, and V_{cwu}, V_{csu} and V_{cfu} are the underflow of water, solids and fines at the hydrocyclone respectively. Because solids are the sum of fine and coarse ore, only the change in the solids and fines are calculated rather than the change in coarse ore. The nomenclature for the model is shown in Table 3.

The three outputs of the model considered in this paper are the fraction of the mill filled with charge *JT*, the volume of the sump

filled with charge *SVOL* and the particle size estimate at the overflow of the cyclone *PSE*. These can be calculated as follow

$$\begin{aligned} JT &= (X_{mw} + X_{ms} + X_{mr} + X_{mb})/v_{mill} \\ SVOL &= X_{ss} + X_{sw} \\ PSE &= V_{cfo}/V_{cso} \end{aligned} \tag{2}$$

where V_{cfo} and V_{cso} are the volumetric flow-rate of fines and solids at the overflow of the cyclone respectively.

The intermediate equations required in (1) related to the mill are

$$\begin{aligned} \varphi &= \max\left\{0, \left(1 - \left(\frac{1}{\varepsilon_{sv}} - 1\right)\frac{X_{ms}}{X_{mw}}\right)^{0.5}\right\} \\ P_{mill} &= P_{\max}\{1 - \delta_{Pv}Z_x^2 - 2\chi_P\delta_{Pv}\delta_{Ps}Z_xZ_r - \delta_{Ps}Z_r^2\}(\alpha_{speed})^{\alpha_P} \\ Z_x &= \frac{X_{mw} + X_{mr} + X_{ms} + X_{mb}}{v_{mill}v_{P_{\max}}} - 1 \\ Z_r &= \frac{\varphi}{\varphi_{P_{\max}}} - 1 \end{aligned} \tag{3}$$

where φ is an empirically defined rheology factor, P_{mill} is the mill power draw, Z_x is the effect of the charge within the mill on the power draw, and Z_r is the effect of the rheology of the charge in the mill on the power draw.

The intermediate equations required in (1) and (2) related to the cyclone are

$$\begin{aligned} V_{ccu} &= \frac{CFF\left(X_{ss} - X_{sf}\right)}{X_{sw} + X_{ss}}\left(1 - C_1\exp\left(\frac{-CFF}{\varepsilon_c}\right)\right) \\ &\quad \times \left(1 - \left(\frac{X_{ss}}{C_2\left(X_{sw} + X_{ss}\right)}\right)^{C_3}\right)\left(1 - \left(\frac{X_{sf}}{X_{ss}}\right)^{C_4}\right) \\ F_u &= 0.6 - \left(0.6 - \frac{X_{ss}}{X_{sw} + X_{ss}}\right)\exp\left(\frac{-V_{ccu}}{\alpha_{su}\varepsilon_c}\right) \\ V_{cwu} &= \frac{X_{sw}\left(V_{ccu} - F_uV_{ccu}\right)}{F_uX_{sw} + F_uX_{sf} - X_{sf}} \\ V_{cfu} &= \frac{X_{sf}\left(V_{ccu} - F_uV_{ccu}\right)}{F_uX_{sw} + F_uX_{sf} - X_{sf}} \\ V_{csu} &= V_{ccu} + \frac{X_{sf}\left(V_{ccu} - F_uV_{ccu}\right)}{F_uX_{sw} + F_uX_{sf} - X_{sf}} \\ V_{cso} &= V_{sso} - V_{csu} \\ V_{cfo} &= V_{sfo} - V_{cfu} \end{aligned} \tag{4}$$

where the flowrate of water, solids, coarse and fines at the underflow of the cyclone are V_{cwu}, V_{csu}, V_{ccu} and V_{cfu} respectively, the flowrate of solids and fines at the sump outflow are V_{sso} and V_{sfo} respectively, and F_u represents the fraction of solids in the cyclone underflow.

The parameter values shown in Table 3, the operating point of the circuit shown in Table 4, and the initial state conditions shown in Table 5 were taken from the sampling campaign of the industrial grinding mill circuit described in Roux et al. [30].

Table 4
Operating point of milling circuit.

Variable	Nominal	Min	Max	Unit
Input				
CFF	374	100	500	m^3/h
MIW	4.64	0	20	m^3/h
MFB	5.69	0	10	t/h
MFS	65.2	0	100	t/h
SFW	140.5	0	400	m^3/h
Output				
JT	0.34	0.25	0.45	Fraction
SVOL	5.99	1.0	8.0	m^3
PSE	0.67	0.5	0.8	Fraction

3. Output tracking using model predictive static programming

3.1. Algorithm derivation

The MPSP control technique for output tracking is discussed here. The state dynamics and output equation of a general discrete nonlinear system can be written as

$$\begin{aligned} X_{k+1} &= F_k(X_k, U_k) \\ Y_k &= H_k(X_k, U_k) \end{aligned} \tag{5}$$

where $X \in \mathbb{R}^n$, $U \in \mathbb{R}^m$, $Y \in \mathbb{R}^p$ represent the states, the input and the output of the system respectively, and k are time steps.

The primary objective of output tracking by means of MPSP is to find an input projection U_k, $k = 1, 2, \ldots, N$ so that the output Y_k goes to the desired output value Y_k^* for time steps 1 to N, i.e. $Y_k \to Y_k^* \; \forall\, k = 1, 2, \ldots, N$. It is important to note that the output Y_k is a function of both the states X_k and the input U_k of the system. The MPSP algorithm predicts the output for N time steps and calculates inputs for N time steps. Compared to MPC, N represents both the prediction and control horizon for MPSP.

For the control technique presented here, it is necessary to start with a "guess" initial input. Obviously the method to obtain a good estimate or intelligent guess of the initial input is problem specific. Because the optimal input will not necessarily be achieved by the guessed input, the input has to be improved by an iterative process where i is the iteration index which increases until the algorithm converges. Convergence can be measured as $\frac{\|Y_k^i - Y_k^*\|}{\|Y_k^*\|} < \epsilon_k$, $\forall\, k = 1, 2, \ldots, N$, where Y_k^* is the desired output and ϵ_k is a user defined tolerance limit on the output error.

The system shown in (5) can now be written as

$$\begin{aligned} X_{k+1}^i &= F_k(X_k^i, U_k^i) \\ Y_k^i &= H_k(X_k^i, U_k^i) \end{aligned} \tag{6}$$

Table 5
Initial states for mill and sump.

State	Value	Unit	State	Value	Unit
Mill states			*Sump states*		
X_{mw}	4.85	m^3	X_{sw}	4.11	m^3
X_{ms}	4.90	m^3	X_{ss}	1.88	m^3
X_{mf}	1.09	m^3	X_{sf}	0.42	m^3
X_{mr}	1.82	m^3			
X_{mb}	8.51	m^3			

The relationship of variables between consecutive iterations i and $i+1$ at time step k are

$$\begin{aligned} Y_k^{i+1} &= Y_k^i + \Delta Y_k^i \\ X_k^{i+1} &= X_k^i + \Delta X_k^i \\ U_k^{i+1} &= U_k^i + \Delta U_k^i \end{aligned} \tag{7}$$

The output Y_k^{i+1} at time step k and iteration $(i+1)$ can be expanded by Taylor series expansion, retaining only first order terms

$$\begin{aligned} Y_k^{i+1} &= H_k(X_k^{i+1}, U_k^{i+1}) \\ &= H_k(X_k^i + \Delta X_k^i, U_k^i + \Delta U_k^i) \\ &\approx Y_k^i + \left[\frac{\partial H_k}{\partial X_k}\right] \Delta X_k^i + \left[\frac{\partial H_k}{\partial U_k}\right] \Delta U_k^i \end{aligned} \tag{8}$$

Combining (8) and the expression for the outputs in (7), it is possible to write

$$\begin{aligned} \Delta Y_k^i &= Y_k^{i+1} - Y_k^i \\ \Delta Y_k^i &\approx \left[\frac{\partial H_k}{\partial X_k}\right] \Delta X_k^i + \left[\frac{\partial H_k}{\partial U_k}\right] \Delta U_k^i \end{aligned} \tag{9}$$

where ΔY_k^i is the error in the output at time k and iteration i.

The state X_{k+1}^{i+1} at time step $(k+1)$ and iteration $(i+1)$ can be expanded by Taylor series expansion retaining only first order terms

$$\begin{aligned} X_{k+1}^{i+1} &= F_k\left(X_k^{i+1}, U_k^{i+1}\right) \\ &= F_k\left(X_k^i + \Delta X_k^i, U_k^i + \Delta U_k^i\right) \\ &\approx F\left(X_k^i, U_k^i\right) + \left[\frac{\partial F_k}{\partial X_k}\right] \Delta X_k^i + \left[\frac{\partial F_k}{\partial U_k}\right] \Delta U_k^i \\ &\approx X_{k+1}^i + \left[\frac{\partial F_k}{\partial X_k}\right] \Delta X_k^i + \left[\frac{\partial F_k}{\partial U_k}\right] \Delta U_k^i \end{aligned} \tag{10}$$

Combining (10) and the expression for states in (7), it is possible to write

$$\begin{aligned} \Delta X_{k+1}^i &= X_{k+1}^{i+1} - X_{k+1}^i \\ \Delta X_{k+1}^i &= \left[\frac{\partial F_k}{\partial X_k}\right] \Delta X_k^i + \left[\frac{\partial F_k}{\partial U_k}\right] \Delta U_k^i \end{aligned} \tag{11}$$

where ΔX_k^i is the error in the state and ΔU_k^i is the error in the input solution at time step k and iteration i. If small input deviations ($\Delta U_k^i = dU_k^i$), small state deviations ($\Delta X_k^i = dX_k^i$) and small output errors are assumed ($\Delta Y_k^i = dY_k^i$), the output error dY_k^i in (9) can be written in terms of the state and input error of (11)

$$\begin{aligned} dY_k^i &= \left[\frac{\partial H_k}{\partial X_k}\right] dX_k^i + \left[\frac{\partial H_k}{\partial U_k}\right] dU_k^i \\ &= \left[\frac{\partial Y_k}{\partial X_k}\right] \left[\frac{\partial F_{k-1}}{\partial X_{k-1}}\right] dX_{k-1}^i + \left[\frac{\partial H_k}{\partial X_k}\right] \left[\frac{\partial F_{k-1}}{\partial U_{k-1}}\right] dU_{k-1}^i \\ &\quad + \left[\frac{\partial H_k}{\partial U_k}\right] dU_k^i \end{aligned} \tag{12}$$

The state error dX_{k-1}^i in (12) can be expanded further in terms of dX_{k-2}^i and dU_{k-2}^i. And the state error dX_{k-2}^i can be expanded further in terms of dX_{k-3}^i and dU_{k-3}^i, and so on. This expansion procedure can continue until state error dX_1^i (where $k=1$) is reached. Finally,

$$\begin{aligned} dY_k^i &= \left[A^k\right]^i dX_1^i + \left[B_1^k\right]^i dU_1^i + \left[B_2^k\right]^i dU_2^i + \cdots + \left[B_{k-1}^k\right]^i dU_{k-1}^i \\ &\quad + \left[B_k^k\right]^i dU_k^i \end{aligned} \tag{13}$$

where

$$\begin{aligned} \left[A^k\right]^i &= \left[\frac{\partial H_k}{\partial X_k}\right] \left[\frac{\partial F_{k-1}}{\partial X_{k-1}}\right] \left[\frac{\partial F_{k-2}}{\partial X_{k-2}}\right] \cdots \left[\frac{\partial F_1}{\partial X_1}\right] \\ \left[B_j^k\right]^i &= \left[\frac{\partial H_k}{\partial X_k}\right] \left[\frac{\partial F_{k-1}}{\partial X_{k-1}}\right] \cdots \left[\frac{\partial F_{j+1}}{\partial X_{j+1}}\right] \left[\frac{\partial F_j}{\partial U_j}\right] \\ \left[B_k^k\right]^i &= \left[\frac{\partial H_k}{\partial U_k}\right] \end{aligned} \tag{14}$$

If it is assumed that with full-state feedback the initial condition of the system is known, i.e. X_1 is known, the error $dX_1^i = X_1^{i+1} - X_1^i$ has to be zero, i.e. $dX_1 = 0$. Therefore, the error in the output in (13) reduces to

$$dY_k^i = \sum_{j=1}^{k} \left[B_j^k\right]^i dU_j^i \tag{15}$$

Note that for the derivation of (15) the input variables at each time step are independent of the previous values of the states and/or inputs. The input variables are seen as decision variables and independent decisions can be made at every point in time. During the implementation of the algorithm, the entire input projection is computed, but only the first input move is performed. For the next time step the algorithm repeats, calculates a new input trajectory and again only performs the first input move of the new trajectory. Feedback is implemented via the cost function, described at a later stage in this section.

Eq. (15) represent the output sensitivity at time step k with respect to change in the input at all time steps prior to and including k. It is intuitively clear that the effect of input changes at future time steps will not change the output vector at the current time step. Therefore, $\left[B_j^k\right]^i$ can be defined for all $k = 1, 2, \ldots, N$ and $j = 1, 2, \ldots, N$. In order to reduce the computational requirements of the algorithm, $\left[B_j^k\right]^i$ can be computed recursively.

$$\left.\begin{aligned} \left[\phi_k^k\right]^i &= I_{n\times n} \\ \left[\phi_j^k\right]^i &= \left[\phi_{j+1}^k\right]^i \left[\frac{\partial F_j}{\partial X_j}\right] \\ \left[B_j^k\right]^i &= \left[\frac{\partial H_k}{\partial X_k}\right] \left[\phi_{j+1}^k\right]^i \left[\frac{\partial F_j}{\partial U_j}\right] \end{aligned}\right\} \quad \forall j < k$$

$$\left[B_j^k\right]^i = \left[\frac{\partial H_k}{\partial U_k}\right] \quad \forall j = k$$

$$\left[B_j^k\right]^i = [0]_{p\times m} \quad \forall j > k \tag{16}$$

The primary objective of the control technique can be defined by the following cost function

$$\begin{aligned} J^i &= \frac{1}{2}\sum_{k=1}^{N}\left(Y_k^{i+1}-Y_k^*\right)^T Q_k\left(Y_k^{i+1}-Y_k^*\right)+\frac{1}{2}\sum_{k=1}^{N}\left(U_k^{i+1}-U_k^i\right)^T R_k\left(U_k^{i+1}-U_k^i\right)\\ &= \frac{1}{2}\sum_{k=1}^{N}\left(Y_k^i+dY_k^i-Y_k^*\right)^T Q_k\left(Y_k^i+dY_k^i-Y_k^*\right)+\frac{1}{2}\sum_{k=1}^{N}\left(dU_k^i\right)^T R_k\left(dU_k^i\right)\\ &= \frac{1}{2}\sum_{k=1}^{N}\left(dY_k^i-dY_k^{*i}\right)^T Q_k\left(dY_k^i-dY_k^{*i}\right)+\frac{1}{2}\sum_{k=1}^{N}\left(dU_k^i\right)^T R_k\left(dU_k^i\right) \end{aligned} \tag{17}$$

where $dY_k^{*i}=Y_k^i-Y_k^*$. The cost function can be written in terms of the input error dU_k by using (15)

$$J^i=\frac{1}{2}\sum_{k=1}^{N}\beta^T Q_k\beta+\frac{1}{2}\sum_{k=1}^{N}\left(dU_k^i\right)^T R_k\left(dU_k^i\right) \tag{18}$$

where

$$\beta=\sum_{j=1}^{k}\left[B_j^k\right]^i dU_j^i-dY_k^{*i} \tag{19}$$

The iteration index i is dropped throughout the rest of the document for the sake of simplicity. The objective is to minimize the cost function J in (18) for dU_1, dU_2, …, dU_N, which requires the calculation of the partial derivatives $\frac{\partial J}{\partial(dU_1)}$, $\frac{\partial J}{\partial(dU_l)}$, and $\frac{\partial J}{\partial(dU_N)}$.

The equation corresponding to $\frac{\partial J}{\partial(dU_z)}=0$ can be simplified to

$$\frac{\partial J}{\partial(dU_z)}=\sum_{k=1}^{N}\left(B_z^{kT}Q_k\sum_{j=1}^{k}B_j^k dU_j\right)-\sum_{k=1}^{N}B_z^{kT}Q_k dY_k^*+R_z dU_z \tag{20}$$

The first term in (20) can be simplified further as

$$\begin{aligned} \sum_{k=1}^{N}\left(B_z^{kT}Q_k\sum_{j=1}^{k}B_j^k dU_j\right) &= \sum_{k=1}^{N}\left(B_z^{kT}Q_kB_1^k dU_1+B_z^{kT}Q_kB_2^k dU_2+\cdots+B_z^{kT}Q_kB_k^k dU_k\right)\\ &= \left(B_z^{1T}Q_1B_1^1dU_1\right)+\left(B_z^{2T}Q_2B_1^2dU_1+B_z^{2T}Q_2B_2^2dU_2\right)+\cdots+\left(B_z^{NT}Q_NB_1^NdU_1+B_z^{NT}Q_NB_2^NdU_2+\cdots+B_z^{NT}Q_NB_N^NdU_N\right)\\ &= \sum_{l=1}^{N}B_z^{lT}Q_lB_1^l dU_1+\sum_{l=2}^{N}B_z^{lT}Q_lB_2^l dU_2+\cdots+B_z^{NT}Q_NB_{N-1}^N dU_N\\ &= C_{z1}dU_1+C_{z2}dU_2+\cdots+C_{zN}dU_N \end{aligned} \tag{21}$$

From the simplification above, matrix $C\in\mathbb{R}^{N\times N}$ can be defined for $e=1,\ldots,N$ and $j=1,\ldots,N$ as

$$C_{ej}=\sum_{l=j}^{N}\left(B_e^l\right)^T Q_lB_j^l \tag{22}$$

Thus, $\frac{\partial J}{\partial(dU_z)}=0$ can now be written as

$$\sum_{k=1}^{N}B_z^{kT}Q_kdY_k^*=C_{z1}dU_1+C_{z2}dU_2+\cdots+C_{zN}dU_N+R_zdU_z \tag{23}$$

3.2. Final calculations

Compiling all the equations for all times steps, the system of equations can be written as

$$[dU_e]=\left[C_{ej}+\delta_{ej}R_e\right]^{-1}[b_e] \tag{24}$$

where δ_{ej} is the Kronecker-delta function and vector $b\in\mathbb{R}^{N\times 1}$ is defined for $e=1,\ldots,N$ as

$$b_e=\sum_{k=1}^{N}B_e^{kT}Q_kdY_k^* \tag{25}$$

Finally, the updated input at time step $k=1,\ldots,N$ is

$$U_k^{i+1}=U_k^i+dU_k^i \tag{26}$$

Because output tracking using MPSP is a relatively new development, some issues remain open for exploration, such as convergence guarantees and the consolidation of input, state and output equality, inequality and rate constraints. A possible approach to address state constraints is shown in Bhitre and Padhi [24] where slack variables can be used to handle state variable inequality constraints. However, the technique developed by Bhitre and Padhi [24] is not for the type of system nor the type of cost function considered here.

Using the grinding mill circuit as control problem, this paper investigates the effect of modelling errors on the MPSP algorithm, its ability to reject noise and large disturbances, and the computational efficiency of the algorithm.

3.3. General procedure

The general procedure to implement the receding horizon MPSP control algorithm is described below.

i. Start the iteration procedure of the MPSP algorithm by estimating an initial input trajectory U_k^1, $\forall k=1,\ 2\ldots N$. Initialize the iteration index as $i=0$.
ii. Use the known initial condition X_1 and input projection U_k^i to propagate the system dynamics in (6). Obtain the state trajectory X_k^i, $\forall k=1,2\ldots N$.
iii. From the state trajectory X_k^i determine the output trajectory Y_k^i. Use the desired output trajectory Y_k^* to calculate $dY_k^i=Y_k^i-Y_k^*$, $\forall k=1,2\ldots N$.
iv. Terminate the algorithm if the output error is smaller than the user-defined tolerance value, i.e. $\frac{\|Y_k^i-Y_k^*\|}{\|Y_k^*\|}<\epsilon_Y$, $\forall k=1,\ldots,N$, or if

the input projection has converged, i.e. $\frac{\|U_k^{i+1}-U_k^*\|_\infty}{\|U_k^*\|_\infty} < \epsilon_U$, $\forall k = 1, \ldots, N$. If either of these conditions are met for $i \geq 1$, then use U_k^i as the optimal input projection. Otherwise, continue with steps v. and vi.

v. Increase the iteration index i. Using X_k^i and U_k^i for all $k = 1, 2, \ldots, N$, calculate the result vector shown in (25) and the matrices shown in (22).
vi. Compute the input deviation dU_k^i using (24). Update the input for the ith iteration with (26) and return to step ii.

4. Nonlinear MPC

Given the system described by (5), the aim of nonlinear MPC can be described as

$$\min_U J(U, X_0) \tag{27}$$

where

$$J = \frac{1}{2}\sum_{k=1}^{N_P}\left(Y_k - Y_k^*\right)^T Q_k \left(Y_k - Y_k^*\right) + \frac{1}{2}\sum_{k=1}^{N_C}(U_{k+1} - U_k)^T \times R_k (U_{k+1} - U_k) \tag{28}$$

where N_C is the control horizon and N_P is the prediction horizon. The input is constrained between $U_{lb} < U < U_{ub}$ where U_{lb} and U_{ub} are the lower and upper bounds for input U respectively.

Comparing the cost function J above to the cost function for MPSP in (17), there is a difference in how the input U_k is evaluated. In (17) the difference between the input projections for consecutive iterations is evaluated (i.e. $U_k^{i+1} - U_k^i$), whereas in (28) the change in the control between time steps is evaluated (i.e. $U_{k+1} - U_k$). The effect of the difference in the cost function formulations can be reduced by increasing the ratio between matrix R and Q. If R is much smaller than Q, the contribution by the change in the input to the cost function will be much less than the contribution by the deviation of the output from setpoint to the cost function.

The problem described above is a constrained minimization problem and can be solved using the *fmincon* function in MATLAB. In this paper, the minimization technique used by *fmincon* to solve (28) was *sequential quadratic programming*. A description of *sequential quadratic programming* can be found in Grüne and Pannek [31]. The principal idea is the formulation of a quadratic subproblem based on the quadratic approximation of the Lagrangian function. An approximation of the Hessian of the Lagrangian function using a quasi-Newton updating method is made at each major iteration. This is used to generate the quadratic programming subproblem whose solution is used to form a search direction for a line search procedure.

5. Simulation

5.1. Simulation environment

The parameter values, the operating point of the circuit and the initial states can be viewed in Tables 3–5, respectively. The state, input and output vectors are defined as

$$\begin{aligned} X &= \left[X_{mw}, X_{ms}, X_{mf}, X_{mr}, X_{mb}, X_{sw}, X_{ss}, X_{sf}\right]^T \\ U &= [MFS, SFW, CFF]^T \\ Y &= [JT, SVOL, PSE]^T \end{aligned} \tag{29}$$

The grinding mill circuit was simulated for both controllers with the following general conditions:

i. Full-state feedback is assumed in this paper, which is a significant assumption as the states in the mill and the sump cannot be measured directly. As seen from Wei and Craig [2], the measurements available in industrial grinding mill circuit are limited. Various attempts have been made to estimate the states and parameters of the plant, which can then be used for the feedback control [32–35]. However, state and parameter estimation for grinding mill circuits is both challenging and interesting, which remains open for further research.
ii. Sampling time of $T_s = 10$ s.
iii. Simulation time of 4 h, i.e. 1440 time sampling points.
iv. The ball feed-rate *MFB* is kept as a constant ratio of 16.7 of the volume of the mill filled with charge *JT* in an attempt to keep the volume of balls in the mill constant, i.e. *MFB*/*JT* = 16.7.
v. Mill water inlet *MIW* is a constant ratio of 7% of *MFS*.
vi. Model-plant-mismatch between the controller and the plant is achieved by maintaining all the model parameters constant for the controller, but varying the following plant parameters every 3 min according to the respective uncertainties shown in Table 3: $\alpha_f, \alpha_r, \alpha_{su}, \epsilon_c, \phi_b, \phi_f, \phi_r$. Each uncertainty follows a uniform distribution around the nominal value of the parameter to produce large changes in the parameter. A uniform distribution was chosen instead of a normal distribution so that the model-plant-mismatch rejection capabilities of the controllers are clearly visible from the simulation results.
vii. A disturbance is simulated as a change in the mill feed size distribution by increasing the fraction of rocks in the ore fed to the mill α_r with 50% of its nominal value. This change is applied between $t = 1.2$ h and $t = 2.8$ h.
viii. A disturbance is simulated as a change in the ore hardness by increasing the power needed per ton of fines produced ϕ_f with 50% of its nominal value. This change is applied between $t = 2.2$ h and $t = 3.8$ h.
ix. The input U is hard-constrained for both MPSP and NMPC between the limits shown in Table 4.

To evaluate the noise and disturbance rejection of the controllers, two scenarios are simulated:

- For the first set of simulations, the milling circuit experience parameter variations and disturbances, but no measurement noise is added to the measured states.
- The same parameter variations and disturbances are used for the second set of simulations, but now measurement noise with a distribution of $\mathcal{N}\left(0, (0.01X_0)^2\right)$ is added to the measured states. Because large and fast unexpected changes to the volume of material in the circuit are not expected, the standard deviation of the noise is kept small.

Both controllers made use of the same Q and R matrices. The particle size estimate *PSE* is regarded as the most important output variable to control as this variable determines the economic efficiency of the milling circuit. Therefore, the Q weighting matrix for the output variables was determined such that a 1% deviation from setpoint for *PSE* will produce an error in the cost function equal to a 5% deviation of *JT* from setpoint and equal to a 20% change in *SVOL* from setpoint, i.e.

$$Q_1(5\%JT_{SP})^2 = Q_2(20\%SVOL_{SP})^2 = Q_3(1\%PSE_{SP})^2$$

Of the three weighting variables, Q_2 will be the smallest. Thus, choosing $Q_2 = 1$, the output weighting matrix is defined as

$$Q = 10^4\,\mathrm{diag}([0.50, \quad 0.0001, \quad 3.11])$$

The R weighting matrix for the input variables was determined such that 2% changes of half the ranges of *CFF*, *MFS* and *SFW* will

produce the same error in the cost function. The R matrix was scaled to produce 1% of the error compared to the Q matrix, i.e.

$$100R_1\left(\frac{2\%MFS_{range}}{2}\right)^2 = Q_1(5\%JT_{SP})^2$$

and

$$R_1\left(\frac{2\%MFS_{range}}{2}\right)^2 = R_2\left(\frac{2\%SFW_{range}}{2}\right)^2 = R_3\left(\frac{2\%CFF_{range}}{2}\right)^2$$

Therefore, the input weighting matrix is

$$R = 10^{-4}\,\mathrm{diag}([36, 16, 23])$$

5.2. MPSP and NMPC implementation details

In this paper, the MPSP and NMPC algorithms are simulated for a control horizon $T_C = 0.1$ h. In the case of NMPC the prediction horizon is set equal to the control horizon. For MPSP the horizon of prediction is by definition the same as the horizon of control.

Both MPSP and NMPC are applied as receding horizon controllers, i.e. the control is calculated from time t_0 to $t_0 + T_C$ where T_C [h] is the fixed control time and t_0 [h] is the current time. Thus, the input trajectory is always $N_C = T_C/T_s$ sampling instances long, where T_s [h] is the sampling time. The time sampling points can be expressed as $\{t_1, t_2, \ldots, t_{N_C}\} = \{t_0, t_0 + T_s, \ldots, t_0 + N_C T_s\}$.

When the controller is initiated, i.e. $t_0 = 0$ s, the initial states of the system are provided as well as an initial input trajectory of N_C sampling instances long. The initial input trajectory is not optimal and has to be improved iteratively by the MPSP or NMPC algorithm. Once the algorithm has determined a new input trajectory for all the time sampling points when $t_1 = t_0 = 0$ s, the input at $t_1 = 0$ s is implemented and the system moves to the next time step. The time is now at $t_1 = T_s$ and the MPSP or NMPC algorithm repeats. The time series was

$$\{t_1, t_2, \ldots, t_{N_C}\} = \{0, T_s, \ldots, N_C T_s\}$$

and is now

$$\{t_1, t_2, \ldots, t_{N_C}\} = \{T_s, 2T_s, \ldots, (N_C + 1)T_s\}$$

A new input trajectory of N_C sampling instance long has to be defined before the control algorithm can start again. However, the initial input trajectory for the first minimization routine when $t_1 = 0$ s cannot be used for the next minimization routine when $t_1 = T_s$. The input trajectory for the first minimization routine when $t_1 = 0$ s was only defined until time $t_{N_C} = N_C T_s$. This input trajectory does not define an input for $t_{N_C+1} = (N_C + 1)T_s$ when $t_1 = 0$. Therefore, the input trajectory for the second minimization routine requires the definition of an input value for the final time step at $t_{N_C} = (N_C + 1)T_s$ when $t_1 = T_s$. The input trajectory for the second minimization routine at $t_1 = T_s$ is obtained by shifting the input trajectory calculated during the first minimization routine at $t_1 = 0$ s one sampling instant to the left and equating the input at t_{N_C} to the input at $t_{N_C} - T_s$.

Termination of the MPSP algorithm occurs if all of the following conditions below are met for the three outputs

$$\begin{aligned} &\| Y(1)_k^i - Y(1)_k^* \| / \| Y(1)_k^* \| < 0.05 \\ &\| Y(2)_k^i - Y(2)_k^* \| / \| Y(2)_k^* \| < 0.1 \\ &\| Y(3)_k^i - Y(3)_k^* \| / \| Y(3)_k^* \| < 0.001 \end{aligned} \tag{30}$$

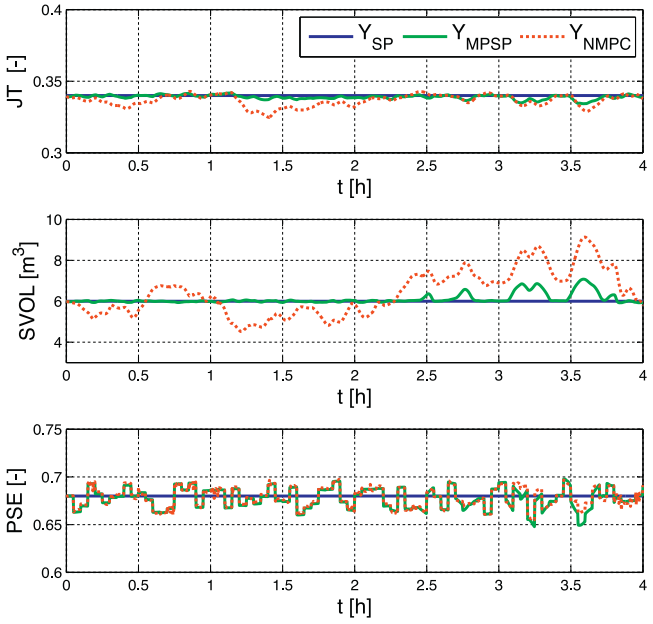

Fig. 2. Output of the plant when no measurement noise is added to the states. [Top: mill total charge fraction *JT*. Middle: sump slurry volume *SVOL*. Bottom: particle size estimate *PSE*. Legend: Y_{SP} is the desired setpoint, Y_{MPSP} is the output from the MPSP controller ($T_C = 0.1$ [h]), Y_{NMPC} is the output from the NMPC controller ($T_C = 0.1$ [h]).].

or if all of the following conditions below are met for the three inputs

$$\begin{aligned} &\| U(1)_k^{i+1} - U(1)_k^* \|_\infty / \| U(1)_k^* \|_\infty < 0.01 \\ &\| U(2)_k^{i+1} - U(2)_k^* \|_\infty / \| U(2)_k^* \|_\infty < 0.01 \\ &\| U(3)_k^{i+1} - U(3)_k^* \|_\infty / \| U(3)_k^* \|_\infty < 0.01 \end{aligned} \tag{31}$$

If these conditions are not met after 10 iterations, the algorithm terminates automatically.

Termination of the NMPC algorithm occurs either after 10 iterations, if the minimization algorithm has reached a feasible minimum for the cost function in (28), or if the cost function values is less than 0.1.

Finally, the partial derivatives of (5) necessary for the MPSP algorithm were explicit functions obtained through symbolic differentiation.

5.3. Results

The results of the two simulation cases discussed in Section 5.1 are shown in Figs. 2–7. Figs. 2 and 3 show the output of the milling circuit when there is no measurement noise and when there is measurement noise added to the states respectively. Figs. 4 and 5 show the input to the milling circuit when there is no measurement noise and when there is measurement noise added to the states respectively. Fig. 6 shows the particle size estimate *PSE* at the overflow of the cyclone for both simulation cases zoomed in between 2 and 2.5 h. Fig. 7 shows the variation of the parameters every 3 min, and also shows the constant disturbance in the fraction of rock in the ore fed to the mill (α_r) and the power needed per ton of fines produced (ϕ_f). Fig. 8 shows the frequency of the number of iterations per time step. For each figure, '$X+0$ ' indicates the case where no measurement noise was added to the states, and '$X + \mathcal{N}$' indicates the case where measurement noise was added to the states.

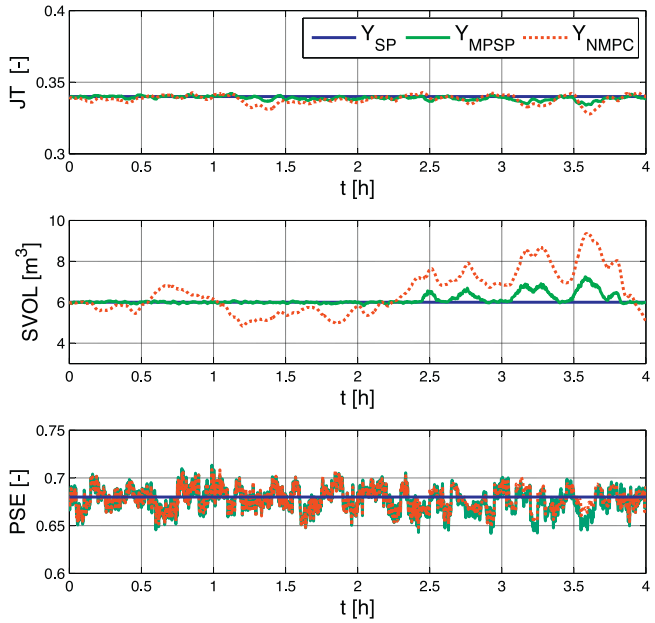

Fig. 3. Output of the plant when measurement noise is added to the states. [Top: mill total charge fraction *JT*. Middle: sump slurry volume *SVOL*. Bottom: particle size estimate *PSE*. Legend: Y_{SP} is the desired setpoint, Y_{MPSP} is the output from the MPSP controller ($T_C = 0.1$ [h]), Y_{NMPC} is the output from the NMPC controller ($T_C = 0.1$ [h]).].

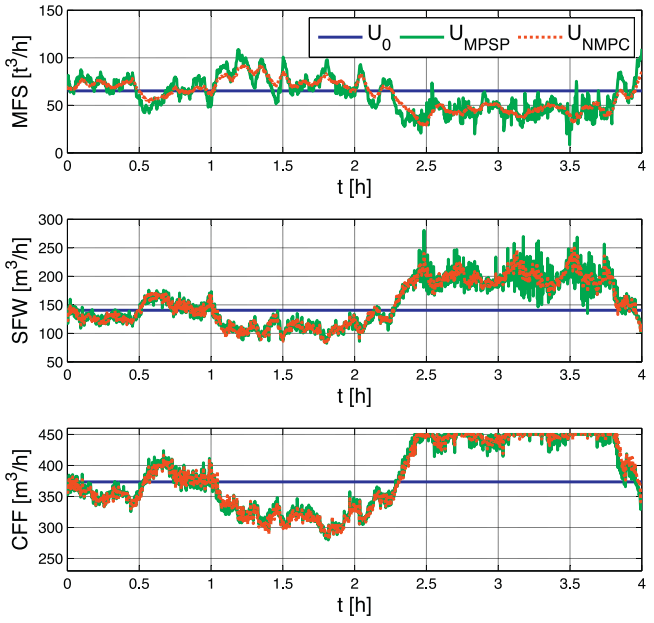

Fig. 5. Input determined by the controller when measurement noise is added to the states. [Top: mill feed solids *MFS*. Middle: sump feed water *SFW*. Bottom: cyclone feed flow-rate *CFF*. Legend: U_0 is original operating point of the plant, U_{MPSP} is the input from the MPSP controller ($T_C = 0.1$ [h]), U_{NMPC} is the input from the NMPC controller ($T_C = 0.1$ [h]).].

6. Discussion

Table 6 shows the normalized root mean squared error between the output of the controllers and the desired setpoint. It is interesting to note that the MPSP controller achieves better output regulation than the NMPC controller, especially with regards to *SVOL*. The trends the outputs follow for the NMPC and MPSP controllers are fairly similar, although there is some difference in the trends for fraction of the mill filled with charge *JT*. The large deviation of the outputs from setpoint after 3 h occurs because the

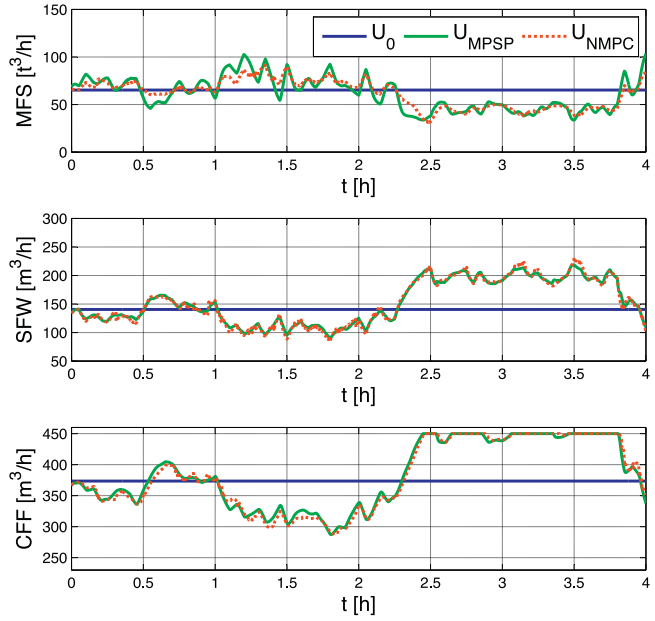

Fig. 4. Input determined by the controller when no measurement noise is added to the states. [Top: mill feed solids *MFS*. Middle: sump feed water *SFW*. Bottom: cyclone feed flow-rate *CFF*. Legend: U_0 is original operating point of the plant, U_{MPSP} is the input from the MPSP controller ($T_C = 0.1$ [h]), U_{NMPC} is the input from the NMPC controller ($T_C = 0.1$ [h]).].

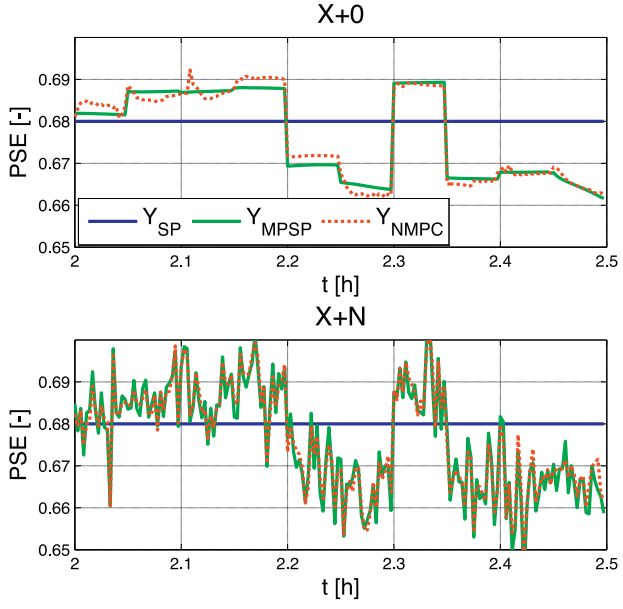

Fig. 6. A zoomed in plot of *PSE* between 2 and 2.5 h for both simulation cases. The top plot clearly shows the effect of the parameter variation every 3 min. For the bottom plot, the noise added to the states dominate the plot.

Table 6
Normalized root mean squared error between desired setpoints and outputs for both controllers. ($(X+0)$ – No measurement noise added to states. $(X+N)$ – Measurement noise added to states.)

		JT	SVOL	PSE
$X+0$	MPSP	0.53%	4.4%	1.6%
	NMPC	1.6%	18%	1.5%
$X+\mathcal{N}$	MPSP	0.51%	4.7%	1.9%
	NMPC	1.1%	18%	1.7%

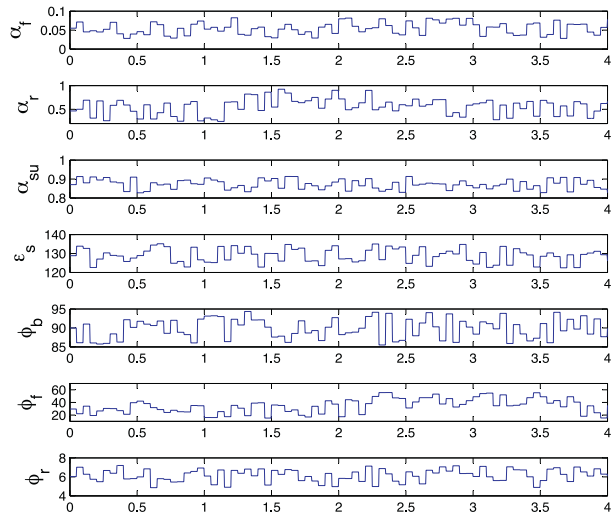

Fig. 7. The variation of parameters α_f, α_r, $\alpha_s u$, ϵ_c, ϕ_b, ϕ_f and ϕ_r every 3 min (0.1 h) according to their respective uncertainties shown in Table 3. The step disturbances in α_r between 1.2 h and 2.8 h and ϕ_f between 2.2 and 3.8 h can be seen in the figures for α_r and ϕ_f respectively.

cyclone feed flow-rate *CFF* is constrained to its maximum allowable rate of 450 m^3/h, as shown in the plots of the inputs in Figs. 4 and 5.

The measurement noise added to the states appears to have the greatest effect on the particle size estimate *PSE* at the cyclone overflow. This is shown in more detail in Fig. 6 where the top plot clearly shows the effect of the parameter variation every 3 min and the bottom plot shows the effect of the measurement noise added to the states. Because of the integrating effect of the charge in the mill and the charge in the sump, the effect of the high frequency measurement noise added to the states is low-pass filtered for outputs *JT* and *SVOL*. However, *PSE* is calculated using an algebraic function, and the small variations in the state have a significant effect on its value. Since the measurement noise added to the mill states is filtered by the sump, which acts as a buffer between the mill and the cyclone, the noisy states which contribute most to the variations in *PSE* is the volume of water X_{sw}, solids X_{ss} and fines X_{sf} in the sump respectively. Of specific importance is the noise added to X_{sf}, since the volume of fines in the sump is the most significant contributor to *PSE*, as shown in (2) and (4). What is apparent from Fig. 6 is that both controllers struggle to reject measurement noise added to the states.

The model-plant mismatch created by varying the plant parameters every 3 min does not appear to have a detrimental effect on overall plant performance. The mismatch effect can be seen most clearly in the top plot of Fig. 6. Both controllers are able to maintain an acceptable level of plant performance, but neither controller can fully reject the mismatch.

The step disturbance between 1.2 and 2.8 h in the fraction of rock in the ore fed to the mill α_r, shown in Fig. 7, does not seem to have a significant impact on the overall performance of the milling circuit. No significant change in the input can be seen to compensate for this disturbance. Another contributing factor that may possibly diminish the effect of this disturbance can be the variation of the rock breakage rate ϕ_r.

The effect of the step disturbance between 2.2 and 3.8 h in the power needed per ton of fines produced ϕ_f can be seen most clearly in the controller's use of the cyclone feed flow *CFF*. An increase in this parameter simulates a case where the ore has hardened and it takes longer for the ore to grind sufficiently fine. In order to maintain the required particle size estimate *PSE*, it is necessary for *CFF* to increase. However, an increase in *CFF* will result in a decrease of the sump volume *SVOL*. The decrease in *SVOL* is negated by increasing the flow of the sump dilution water *SFW*. Because more energy is required to break the ore, it takes longer to break. As shown in Figs. 4 and 5 the feed-rate of ore the mill *MFS* decreases slightly to maintain a constant fraction of the mill filled *JT*. Because the cyclone feed flow-rate *CFF* is constrained after approximately 3 h as shown in Figs. 4 and 5, *PSE* cannot be maintained at setpoint, as shown in Figs. 2 and 3.

Fig. 8 shows the frequency of the number of iterations per time step needed by both the MPSP and NMPC minimization algorithms to reach a solution. It should be remembered that the MPSP and NMPC routines were limited to a maximum 10 iterations. As seen from Fig. 8, for the simulation scenarios considered in this paper, 2 iterations of the MPSP routine is usually sufficient to find an optimal solution. When noise is added to the states, the MPSP algorithm may require a few more iterations to achieve a feasible solution. On the other hand, the NMPC algorithm usually achieves convergence within 3 iterations if no noise is added to the states, and between 3 and 6 iterations if noise is added to the states.

It is regarded as unfair to compare the computational time for the custom programmed MPSP routine to the NMPC routine implemented with the *fmincon* function of the Optimization Toolbox of MATLAB. The custom programmed MPSP routine does not call as many subroutines and error handling routines as *fmincon*. However, for interest sake the average computational time required for one iteration of the MPSP routine was 0.16 s. The average computational

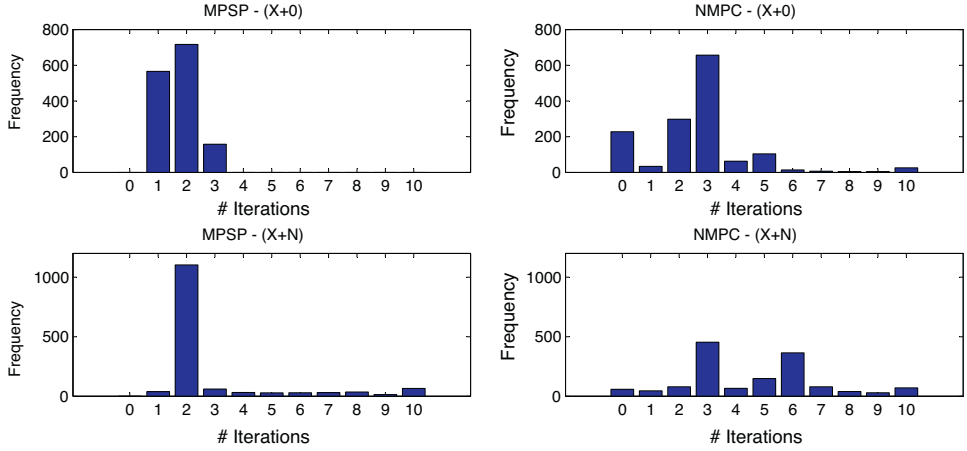

Fig. 8. Algorithm iterations frequency. [$(X+0)$ - no noise added to states. $(X+\mathcal{N})$ – noise added to states.].

time for the *fmincon* function to complete one iteration was 1.6 s. Simulations were carried out on a 64-bit system with 4GB RAM and an Intel Core i7-2600 CPU @ 3.4 GHz in MATLAB R2013a. Obviously the computational time of both routines can be decreased, but it is encouraging to note that the large number of matrix operations for the MPSP routine is handled quickly and effectively.

In order to demonstrate repeatability, the simulations for the MPSP controller, with and without measurement noise added to states, was repeated 50 times and a new noise and model-plant mismatch vector was generated for each repetition. The MPSP controller was limited to a maximum of 2 iterations per time step. For each repetition, the MPSP controller was able to track the desired setpoint irrespective of the parameter variations and measurement noise. There was no unusual behaviour for any of the simulations even though only 2 iterations per time step was allowed. The normalized root mean squared error between the plant output and the setpoint was less than 3% for *PSE*, less than 1.5% for *JT* and less than 12% for *SVOL*. The reason for the larger error in *SVOL* is that *CFF* starts to increase to compensate for the disturbance applied between 2.2 and 3.8 h. When the upper constraint on *CFF* is applied, *SVOL* starts to increase and moves away from the setpoint.

7. Conclusion

Output tracking using MPSP was successfully applied to a nonlinear model of a single-stage closed grinding mill circuit. The performance of the proposed MPSP controller was evaluated against the performance of a standard constrained NMPC controller applied to the same plant for the same conditions. The aim of this paper is not to compare the mathematical details of each minimization algorithm, but rather to show the control abilities of MPSP in the presence of noise, model-plant mismatch, disturbances and input box constraints.

Results indicate that the performance of the MPSP controller compares favourably to the performance of an NMPC controller. Compared to NMPC, MPSP achieved improved output regulation in the presence of model-plant mismatch, disturbances and measurement noise. Both controllers showed similar performance to reject the mismatches, disturbances and measurement noise. Even if the MPSP algorithm is limited to two iterations per time step, it is still capable of controlling the plant with adequate accuracy. The advantage of MPSP is that the computational burden is decreased by converting the dynamic optimization problem to a low-dimensional static optimization problem, calculating the sensitivity matrices recursively and using a closed form expression to update the control. Although the prediction and control horizons are equal in MPSP, this does not lead to substantially increased computational time. If these horizons are decreased further the computational time should also decrease, but at the cost of control performance. Further improvements to the computational time of MPSP can be achieved by using improved matrix operation programming techniques for matrix inverse and multiplication. Future work for MPSP will involve the application of equality and inequality constraints for states, inputs and outputs in the algorithm formulation, as well as creating a framework for a robust MPSP controller.

MPSP shows promise as a suitable option for nonlinear model-based optimal control for output tracking in large industrial processes, especially if computational time and complexity is a limiting factor for the real-time application of a model-based optimal controller.

References

[1] I.K. Craig, I.M. MacLeod, Specification framework for robust control of a run-of-mine ore milling circuit, Control Eng. Pract. 3 (1995) 621–630.

[2] D. Wei, I. Craig, Grinding mill circuits – a survey of control and economic concerns, Int. J. Miner. Process. 90 (2009) 56–66.

[3] X. Chen, Q. Li, S. Fei, Constrained model predictive control in a ball mill grinding process, Powder Technol. 186 (2008) 31–39.

[4] I. Craig, C. Aldrich, R. Braatz, F. Cuzzola, E. Domlan, S. Engell, J. Hahn, V. Havlena, A. Horch, B. Huang, M. Khanbaghi, A. Konstantellos, W. Marquardt, T. McAvoy, T. Parisini, S. Pistikopoulos, T. Samad, S. Skogestad, N. Thornhill, J. Yu, Control in the process industries, in: The Impact of Control Technology, IEEE Control Systems Society, 2011, Available at: http://www.ieeecss.org/main/IoCT-report

[5] A.J. Niemi, L. Tian, R. Ylinen, Model predictive control for grinding systems, Control Eng. Pract. 5 (1997) 271–278.

[6] A. Pomerleau, D. Hodouin, A. Desbiens, E. Gagnon, A survey of grinding circuit control methods: from decentralized PID controllers to multivariable predictive controllers, Powder Technol. 108 (2000) 103–115.

[7] M. Ramasamy, S.S. Narayanan, C.D.P. Rao, Control of ball mill grinding circuit using model predictive control scheme, J. Process Control 15 (2005) 273–283.

[8] A. Remes, J. Aaltonen, H. Koivo, Grinding circuit modeling and simulation of particle size control at Siilinjärvi concentrator, Int. J. Miner. Process. 96 (2010) 70–78.

[9] D. Wei, I. Craig, Economic performance assessment of two ROM ore milling circuit controllers, Miner. Eng. 22 (2009) 826–839.

[10] D. Hodouin, Methods for automatic control, observation and optimization in mineral processing plants, J. Process Control 21 (2011) 211–225.

[11] T.A. Apelt, N.F. Thornhill, Inferential measurement of SAG mill parameters V: MPC simulation, Miner. Eng. 22 (2009) 1045–1052.

[12] J. Yang, S. Li, X. Chen, Q. Li, Disturbance rejection of ball mill grinding circuits using DOB and MPC, Powder Technol. 198 (2010) 219–228.

[13] L.E. Olivier, I.K. Craig, Model-plant mismatch detection and model update for a run-of-mine ore milling circuit under model predictive control, J. Process Control 23 (2013) 100–107.

[14] X. Chen, J. Zhai, S. Li, Q. Li, Application of model predictive control in ball mill grinding circuit, Miner. Eng. 20 (2007) 1099–1108.

[15] M.S. Powell, R.D. Morrison, The future of comminution modelling, Int. J. Miner. Process. 84 (2007) 228–239.

[16] D. Hodouin, S.L. Jamsa-Jounela, M.T. Carvalho, L. Bergh, State-of-the-art and challenges in mineral processing control, Control Eng. Pract. 9 (2001) 995–1005.

[17] L.C. Coetzee, I.K. Craig, E.C. Kerrigan, Robust nonlinear model predictive control of a run-of-mine ore milling circuit, IEEE Trans. Control Syst. Technol. 18 (2010) 222–229.

[18] J.D. le Roux, I.K. Craig, Reducing the number of size classes in a cumulative rates model used for process control of a grinding mill circuit, Powder Technol. 246 (2013) 169–181.

[19] A. Bemporad, M. Morari, V. Dua, E.N. Pistikopoulos, The explicit solution of model predictive control via multiparametric quadratic programming, in: Proceedings of the American Control Conference, Chicago, USA, June 2000, pp. 872–876.

[20] Y. Wang, S. Boyd, Fast model predictive control using online optimization, IEEE Trans. Control Syst. Technol. 18 (2010) 267–278.

[21] R. Padhi, M. Kothari, Model predictive static programming: a computationally efficient technique for suboptimal control design, Int. J. Innov. Comput. Inf. Control 5 (2009) 399–411.

[22] O. Halbe, R.G. Raja, R. Padhi, Robust re-entry guidance of a reusable launch vehicle using model predictive static programming, J. Guidance Control Dyn. 37 (2014) 134–148.

[23] H.B. Oza, R. Padhi, Impact-angle-constrained suboptimal model predictive static programming guidance of air-to-ground missiles, J. Guidance Control Dyn. 35 (2012) 153–164.

[24] N.G. Bhitre, R. Padhi, State constrained model predictive static programming: a slack variable approach, in: Proceedings of the 3rd International IFAC Conference on Advances Control Optimization Dynamical Systems, Kanpur, India, March 2014, pp. 832–839, http://dx.doi.org/10.3182/20140313-3-IN-3024.00175.

[25] G. Joshi, R. Padhi, Formation flying of small satellites using suboptimal MPSP guidance, in: Proceedings of the American Control Conference, Washington, DC, USA, June 2013, pp. 1584–1589.

[26] A. Maity, H.B. Oza, R. Padhi, Generalized model predictive static programming and its application to 3D impact angle constrained guidance of air-to-surface missiles, in: Proceedings of the American Control Conference, Washington, DC, USA, June 2013, pp. 4999–5004.

[27] P. Kumar, R. Padhi, Extension of model predictive static programming for reference command tracking, in: Proceeding of the 3rd International IFAC Conference on Advances Control Optimization Dynamical Systems, Kanpur, India, March 2014, pp. 855–861, http://dx.doi.org/10.3182/20140313-3-IN-3024.00174.

[28] J.D. le Roux, I.K. Craig, R. Padhi, Output tracking for a milling circuit using model predictive static programming, in: Proceedings of the 19th IFAC World Congress, Cape Town, South Africa, August 2014, pp. 9792–9797, http://dx.doi.org/10.3182/20140824-6-ZA-1003.01902.

[29] T.J. Napier-Munn, S. Morrell, R.D. Morrison, T. Kojovic, Mineral Communition Circuits: Their Operation and Optimisation, in: JKMRC Monograph Series in Mining and Mineral Processing, 2nd ed., 1999.

[30] J.D. le Roux, I.K. Craig, D.G. Hulbert, A.L. Hinde, Analysis and validation of a run-of-mine ore grinding mill circuit model for process control, Miner. Eng. 43–44 (2013) 121–134.

[31] L. Grüne, J. Pannek, Nonlinear Model Predictive Control, Theory and Algorithms, Springer-Verlag, 2011.
[32] J.D. le Roux, I.K. Craig, R. Padhi, State and parameter estimation for a grinding mill circuit from operational input–output data, in: Proceedings of the 10th IFAC Symposium Dynamics Control Process Systems, Mumbai, India, December 2013, pp. 178–183, http://dx.doi.org/10.3182/20131218-3-IN-2045.00046.
[33] M.A. Naidoo, L.E. Olivier, I.K. Craig, Combined neural network and particle filter state estimation with application to a run-of-mine ore mill, in: Proceedings of the 10th IFAC Symposium Dynamics Control Process Systems, Mumbai, India, December 2013, pp. 397–402, http://dx.doi.org/10.3182/20131218-3-IN-2045.00103.
[34] L.E. Olivier, B. Huang, I.K. Craig, Dual particle filters for state and parameter estimation with application to a run-of-mine ore mill, J. Process Control 22 (2012) 710–717.
[35] T.A. Apelt, S.P. Asprey, N.F. Thornhill, Inferential measurement of SAG mill parameters. II: State estimation, Miner. Eng. 15 (2002) 1043–1053.