Literature Notes

Brittany Hall

1 Numerical Optimal Control-Parametric Optimization

1.1 Parametric NLPs

A generic non-linear programming problem can be written in the following way:

$$\min_{w} \quad \Phi(w,p)) \tag{1a}$$
$$\text{s.t.} \quad g(w,p) = 0, \tag{1b}$$
$$h(w,p) \leq 0 \tag{1c}$$

where $\vec{p}$ is a vector of parameters. Changes of active set yield non-smooth points in the solution manifold $w(p)$. It is important to understand that $w(p) : p \in \mathbb{R}^p \to w \in \mathbb{R}^n$ as a genuine function defined by the parametric NLP. Solution w is an implicit function of the parameters p so we need to use IFT.

When looking at problems like this, we consider four things:

- continuity and differentiability
- sensitivities $\frac{\partial w}{\partial p}$
- Predictors: using $w(p_0)$ what can I say about $w(p)$?

$$w(p) \approx w(p_0) + \frac{\partial w}{\partial p}(p - p_0)$$

- Path-following: how to keep track of $w(p)$ for a (continuously) changing p?

Theorem 1.1. *If $w(p)$ fulfils LICQ and strict SOSC (positive definite Hessian), then $w(p)$ is continuous around p.*

1.1.1 Differentiability

Theorem 1.2. *Consider $w(p)$ at a given p with*

- *LICQ and strict SOSC*
- *no weakly active constraint h*

then $\nabla_p w(p)$ exists.

1.1.2 Sensitivity in Newton

- Newton as an implicit function

$$\begin{aligned} \min_{w} \quad & \Phi(w,p)) && \text{(2a)} \\ \text{s.t.} \quad & g(w,p) = 0 && \text{(2b)} \end{aligned}$$

Solution $w(p), \lambda(p)$ implicitly given by the KKT conditions:

$$\begin{aligned} \nabla_w \mathcal{L}(w, p\lambda) &= 0 \\ g(w,p) &= 0 \end{aligned}$$

Theorem 1.3 (Implicit Function Theorem)**.** *Let z be implicitly given by the $\mathcal{C}^1$ function:*

$$R(z,p) = 0 \textit{ with } \nabla_z R(z,p_0) \textit{ full rank}$$

Then for any p_0 there is a $\mathcal{C}^2$ function $\xi(p)$ such that:

$$R(\xi(p),p) = 0$$

holds in a neighborhood of p_0. That means we have $z(p) = \xi(p)$ around p_0.

In other words, $z(p)$ is locally well defined and differentiable if $\nabla_z R(z,p)$ exists and is full rank.

We will have then:

$$\begin{aligned} z &= \begin{bmatrix} w \\ \lambda \end{bmatrix} \\ R &= \begin{bmatrix} \nabla_w \mathcal{L}(w,\lambda,p) \\ g(w,p) \end{bmatrix} \end{aligned}$$

We now check that $\nabla_z R(z,p)$ for the KKT conditions:

$$\nabla_z R(z,p) = \begin{bmatrix} \nabla_w^2 \mathcal{L}(w,\lambda,p) & \nabla_w g(w,p) \\ \nabla_w g(w,p)^T & 0 \end{bmatrix}$$

This is the KKT matrix providing the Newton step, recall:

$$\begin{bmatrix} \nabla_w^2 \mathcal{L}(w,\lambda,p) & \nabla_w g(w,p) \\ \nabla_w g(w,p)^T & 0 \end{bmatrix} \begin{bmatrix} \Delta w \\ \lambda \end{bmatrix} = - \begin{bmatrix} \nabla \Phi(w) \\ g(w,p) \end{bmatrix}$$

Theorem 1.4. *The parametric solution $z(p)$ is well defined and $\mathcal{C}^1$ in a neighborhood of p if the KKT matrix is full rank at p*

- Differentiating Implicit functions
  The sensitivity $\frac{\partial}{\partial p} z(p)$ is given by

$$\frac{dR(z,p)}{dp} = \frac{\partial R(z,p)}{\partial z}\frac{\partial z}{\partial p} + \frac{\partial R(z,p)}{\partial p} = 0$$

or in other words

$$\frac{\partial z}{\partial p} = -\frac{\partial R(z,p)^{-1}}{\partial z}\frac{\partial R(z,p)}{\partial p}$$

with

$$\begin{aligned}\frac{\partial R(z,p)}{\partial z} &= \begin{bmatrix} H(w,\lambda,p) & \nabla_w g(w,p) \\ \nabla_w g(w,p)^T & 0 \end{bmatrix} \\ \frac{\partial R(z,p)}{\partial p} &= \begin{bmatrix} \nabla_{w,p}\mathcal{L}(w,\lambda,p) \\ \nabla_p g(w,p)^T \end{bmatrix}\end{aligned}$$

If p enters linearly in $g(w,p)$ then

$$\frac{\partial R(z,p)}{\partial p} = \begin{bmatrix} 0 \\ Cst. \end{bmatrix}$$

Sensitivities are for free since a factorization of the KKT matrix is available from the Newton algorithm.

- Computing the Sensitivities-Implementation
  - M re-used in the sensitivities, computationally cheap
  - Sensitivities are inexact if Newton is not properly converged
  - Must use $\nabla_{w,p}\mathcal{L}$ and not $\nabla_{w,p}\Phi$ in sensitivities

Algorithm: NLP solution with sensitivities

```
Input: $\mathbf{w}$, $\boldsymbol{\lambda}$, $\mathbf{p}$
while not converged do
    Compute:
        $M = \begin{bmatrix} H & \nabla_{\mathbf{w}}\mathbf{g} \\ \nabla_{\mathbf{w}}\mathbf{g}^{\mathsf{T}} & 0 \end{bmatrix}^{-1}$
    Newton step
        $\begin{bmatrix} \Delta\mathbf{w} \\ \boldsymbol{\lambda}^{+} \end{bmatrix} = -M \begin{bmatrix} \nabla_{\mathbf{w}}\Phi \\ \mathbf{g} \end{bmatrix}$
    Update: $\mathbf{w} \leftarrow \mathbf{w} + t\Delta\mathbf{w}, \quad \boldsymbol{\lambda} \leftarrow t\boldsymbol{\lambda}^{+} + (1-t)\boldsymbol{\lambda}_k$
Compute sensitivities at the solution:
    $\frac{\partial}{\partial \mathbf{p}} \begin{bmatrix} \mathbf{w} \\ \boldsymbol{\lambda} \end{bmatrix} = -M \begin{bmatrix} \nabla_{\mathbf{w},\mathbf{p}}\mathcal{L} \\ \nabla_{\mathbf{p}}\mathbf{g}^{\mathsf{T}} \end{bmatrix}$
return $\mathbf{w}$, $\boldsymbol{\lambda}$, $\frac{\partial \mathbf{w}}{\partial \mathbf{p}}$, $\frac{\partial \boldsymbol{\lambda}}{\partial \mathbf{p}}$
```

1.1.3 Sensitivity with inequality constraints

$$\min_{w} \quad \Phi(w,p)) \tag{3a}$$

$$\text{s.t.} \quad g(w,p) = 0, \tag{3b}$$

$$h(w,p) \le 0 \tag{3c}$$

The solution $w(p), \lambda(p), \mu(p)$ is given by the KKT conditions:

$$\begin{aligned} \nabla_w \mathcal{L}(w,p,\lambda,\mu) &= 0 \\ g(w,p) &= 0 \\ h(w,p) &\le 0 \\ \mu_i h_i(w,p) &= 0 \end{aligned}$$

Now we have non-smooth conditions; however, they are piecewise smooth.

If we let $\mathbb{A}$ be the active set, then we have:

$$\begin{aligned}\nabla_w \mathcal{L}(w,p,\lambda,\mu) &= 0\\ g(w,p) &= 0\\ h_{\mathbb{A}}(w,p) &= 0\\ \mu_{\mathbb{A}} &= 0\end{aligned}$$

and $h_{\mathbb{A}}(w,p) < 0, \mu_{\mathbb{A}} > 0$. The conditions are smooth as long as all constraints are strictly active. Then we avoide the "corner" of the complementarity slackness manifold.

We know let $\mathbb{A}$ be the strictly active set and we have:

$$\begin{aligned}\nabla_w \Phi(w) + \nabla_w g(w,p)\lambda + \nabla_w h_{\mathbb{A}}(w,p)\mu_{\mathbb{A}} &= 0\\ g(w,p) &= 0\\ h_{\mathbb{A}}(w,p) &= 0\\ \mu_{\mathbb{A}} &= 0\end{aligned}$$

and $h_{\mathbb{A}}(w,p) < 0, \mu_{\mathbb{A}} > 0$. We then define

$$\begin{aligned}R(z,p) &= \begin{bmatrix}\nabla_w \Phi(w) + \nabla_w g(w,p)\lambda + \nabla_w h_{\mathbb{A}}(w,p)\mu_{\mathbb{A}}\\ g(w,p)\\ h_{\mathbb{A}}(w,p)\end{bmatrix}\\ z &= \begin{bmatrix} w\\ \lambda\\ \mu_{\mathbb{A}}\end{bmatrix}\end{aligned}$$

The sensitivity is then given by $\frac{\partial z}{\partial p} = -\frac{\partial R(z,p)}{\partial z}^{-1}\frac{\partial R(z,p)}{\partial p}$ with :

$$\frac{\partial R}{\partial z} = \begin{bmatrix} H & \nabla_w g & \nabla_w h_{\mathbb{A}}\\ \nabla_w g^T & 0 & 0\\ \nabla_w h_{\mathbb{A}}^T & 0 & 0\end{bmatrix}$$

The matrix $\frac{\partial R}{\partial z}$ is factorized inside Active Set QP solvers (i.e. we get the sensitivities for free when using SQP).

Example 1.1.

$$\min_{w} \quad \|w\|^2 \tag{4a}$$

$$\text{s.t.} \quad w_2 - w_1(1+w_1^2) + \theta = 0, \tag{4b}$$

$$\frac{1}{5}(\tanh\theta + 1) - w_2 \leq 0 \tag{4c}$$

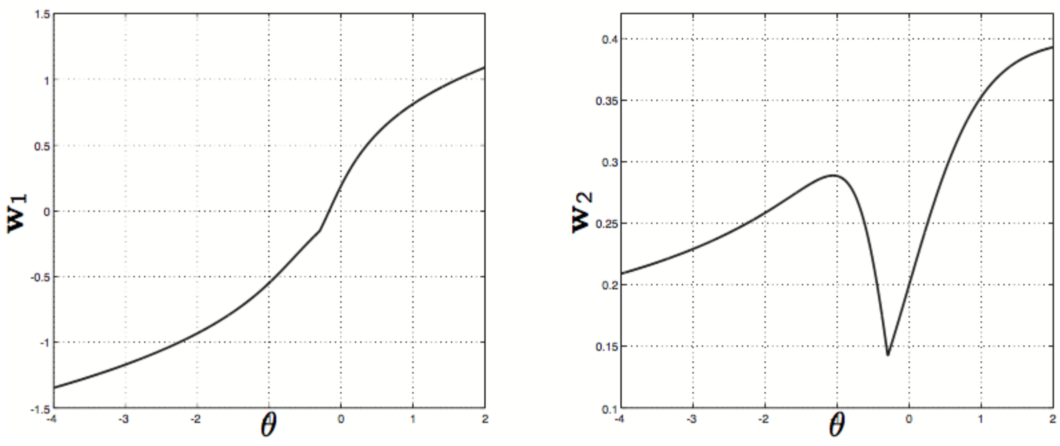


Figure 1

Writing the KKT conditions:

$$\frac{\partial}{\partial p}\begin{bmatrix} w \\ \lambda \end{bmatrix}$$
$$= -\begin{bmatrix} H & \nabla_w g \\ \nabla_w g^T & 0 \end{bmatrix}^{-1} \frac{\partial}{\partial p}\begin{bmatrix} \nabla_w \mathcal{L} \\ g \end{bmatrix}$$
$$\frac{\partial \mu}{\partial p} = 0$$

Note that the constraint h is inactive, $\mu = 0$, $\mathbb{A} = \emptyset$. We then check $\frac{\partial z}{\partial \theta}$.

At the "corner" the derivative does not exist but the directional derivatives do, i.e. $\lim_{p \to p^-} \frac{\partial z}{\partial p}$ *and* $\lim_{p \to p^+} \frac{\partial z}{\partial p}$ *exist.*

1.2 Non-Smooth "Linear" Approximator- QP approximation

$$\min_{w} \quad \Phi(w,p) \tag{5a}$$
$$\text{s.t.} \quad g(w,p) = 0, \tag{5b}$$
$$h(w,p) \le 0 \tag{5c}$$

With the following sensitivities:

$$\frac{\partial}{\partial p}\begin{bmatrix} w \\ \lambda \\ \mu \end{bmatrix} = -\begin{bmatrix} H & \nabla_w g & \nabla_w h \\ \nabla_w g^T & 0 & 0 \\ \nabla_w h^T & 0 & 0 \end{bmatrix} \frac{\partial}{\partial p}\begin{bmatrix} \nabla_w \mathcal{L} \\ g \\ h \end{bmatrix}$$

We can write the approximating QP:

$$\begin{aligned}
\min_{\Delta w} \quad & \frac{1}{2}\Delta w^T H \Delta w + \Delta p^T \nabla_{pw}\mathcal{L}\Delta w && \text{(6a)}\\
\text{s.t.} \quad & g + \nabla_w g^T \Delta g^T \Delta w + \nabla_p g^T \Delta p = 0, && \text{(6b)}\\
& h + \nabla_w h^T \Delta h^T \Delta w + \nabla_p h^T \Delta p \leq 0 && \text{(6c)}
\end{aligned}$$

where $\Delta w = w - w_0$ and $\Delta p = p - p_0$.

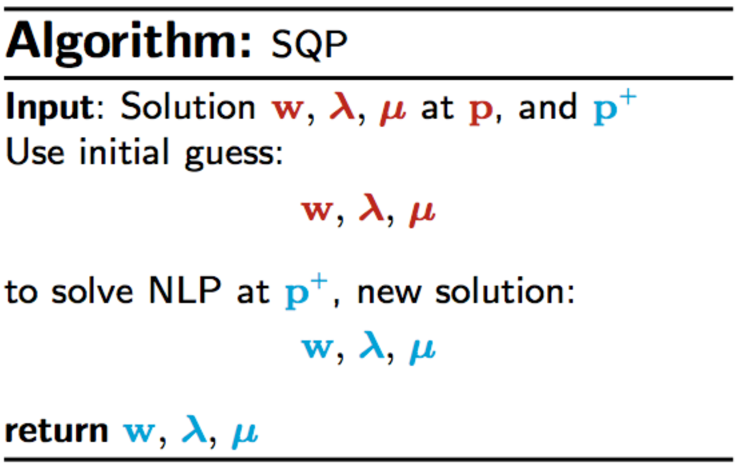
QP predictor holds the sensitivities implicitly (current active set) but also catches a linear approximation of the "kinks" resulting from changes of active set.

1.2.1 Implementation

- Solved at p_0 yields w_0
- New parameter p, perturbation $\Delta p = p - p_0$ inserted into the QP
- Predicted solution $w = w_0 + \Delta w$ from the QP

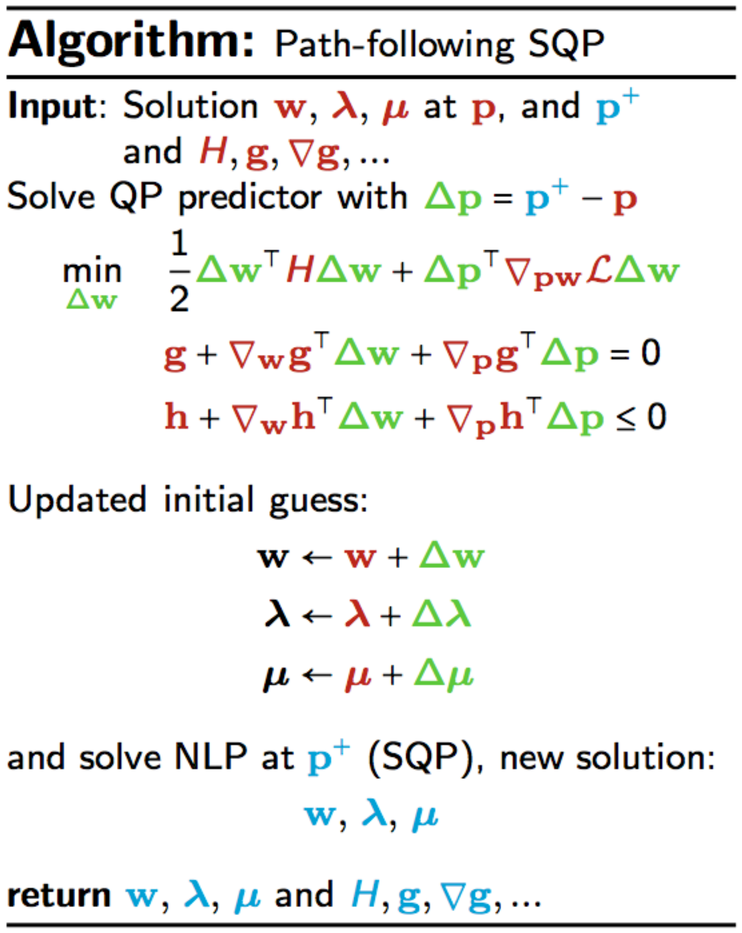
1.3 Off-line Path Following

Follow a solution path $w(p)$, for a given "parameter trajectory" p. Below is the traditional SQP method using Newton steps.



```
Algorithm: SQP
Input: Solution $\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\mu}$ at $\mathbf{p}$, and $\mathbf{p}^+$
Use initial guess:
                    $\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\mu}$
to solve NLP at $\mathbf{p}^+$, new solution:
                    $\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\mu}$
return $\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\mu}$
```

The off-line path following SQP algorithm is outlined below.



```
Algorithm: Path-following SQP
Input: Solution $\mathbf{w}$, $\boldsymbol{\lambda}$, $\boldsymbol{\mu}$ at $\mathbf{p}$, and $\mathbf{p}^+$
       and $H, \mathbf{g}, \nabla\mathbf{g}, \ldots$
Solve QP predictor with $\Delta\mathbf{p} = \mathbf{p}^+ - \mathbf{p}$
    $\min_{\Delta\mathbf{w}} \frac{1}{2}\Delta\mathbf{w}^\top H \Delta\mathbf{w} + \Delta\mathbf{p}^\top \nabla_{\mathbf{pw}}\mathcal{L}\Delta\mathbf{w}$
        $\mathbf{g} + \nabla_{\mathbf{w}}\mathbf{g}^\top \Delta\mathbf{w} + \nabla_{\mathbf{p}}\mathbf{g}^\top \Delta\mathbf{p} = 0$
        $\mathbf{h} + \nabla_{\mathbf{w}}\mathbf{h}^\top \Delta\mathbf{w} + \nabla_{\mathbf{p}}\mathbf{h}^\top \Delta\mathbf{p} \leq 0$
Updated initial guess:
    $\mathbf{w} \leftarrow \mathbf{w} + \Delta\mathbf{w}$
    $\boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda} + \Delta\boldsymbol{\lambda}$
    $\boldsymbol{\mu} \leftarrow \boldsymbol{\mu} + \Delta\boldsymbol{\mu}$
and solve NLP at $\mathbf{p}^+$ (SQP), new solution:
    $\mathbf{w}$, $\boldsymbol{\lambda}$, $\boldsymbol{\mu}$
return $\mathbf{w}$, $\boldsymbol{\lambda}$, $\boldsymbol{\mu}$ and $H, \mathbf{g}, \nabla\mathbf{g}, \ldots$
```

1.4 Predictor-Corrector

The QP predictor predicts solution for a parametric change in comparison to the SQP (corrector) which corrects the solution until KKT conditions are satisfied.

$$\min_{\Delta w} \quad \frac{1}{2}\Delta w^T H \Delta w + (p^+ - p)^T \nabla_{pw}\mathcal{L}\Delta w \tag{7a}$$

$$\text{s.t.} \quad g + \nabla_w g^T \Delta w + \nabla_p g^T (p^+ - p) = 0, \tag{7b}$$

$$h + \nabla_w h^T \Delta w + \nabla_p h^T (p^+ - p) = 0 \tag{7c}$$

The implementation is carried out as follows where the solution w at p is known, together with $H, g, \nabla g, \ldots$. For the new parameter p^+:

- Solve Predictor-Corrector, using $H, g, \nabla g, \ldots$ (already known). Predicts the next solution + correct w if KKT not yet fulfilled
- Update $p \leftarrow p^+$, re-evaluate $H, g, \nabla g, \ldots$, back to step 1. Correct solution until KKT satisfied.

1.4.1 Predictor-Corrector and Parametric Embedding

NLP with parametric embedding is equivalent to the "Path-following SQP" algorithm. This is a "cheap" way of implementing the algorithm. The predictor-correct QP is as follows:

$$\min_{\Delta w, \Delta\theta} \quad \frac{1}{2}\Delta w^T H \Delta w + \nabla\Phi^T \Delta w + \Delta\theta^T \nabla_{\theta,w}\mathcal{L}\Delta w \tag{8a}$$

$$\text{s.t.} \quad g + \nabla_w g^T \Delta w + \nabla_p g^T \Delta\theta = 0, \tag{8b}$$

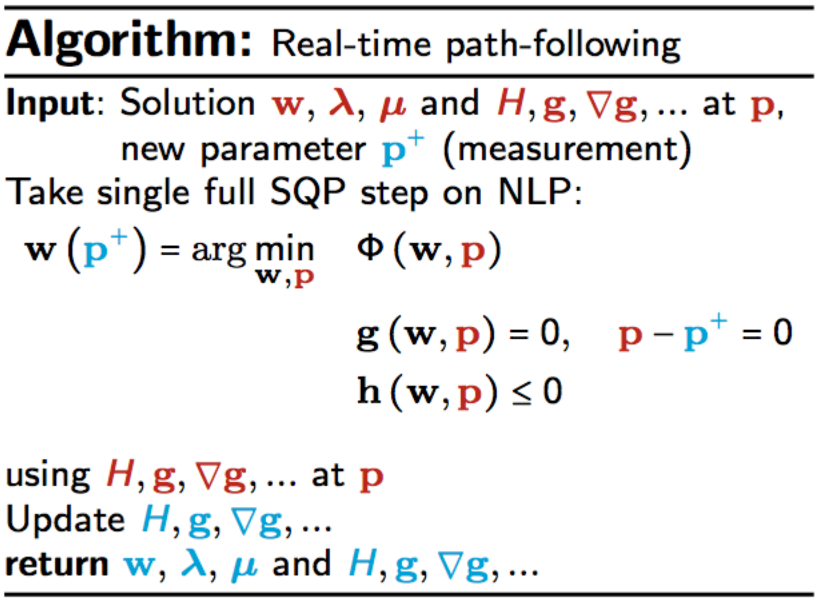
$$h + \nabla_w h^T \Delta w + \nabla_p h^T \Delta\theta \leq 0, \tag{8c}$$

$$p - p^+ + \Delta\theta = 0 \tag{8d}$$

We note that if $p \neq p^+$, then the QP does a predictor-corrector step and sets $p \leftarrow p^+$ (full SQP step); if $p = p^+$, then QP does a classic SQP (corrector) step and converges to the KKT conditions.

1.5 Real-time Path Following: The real-time dilemma

Suppose that $p(t)$ is continuously changing with time t and continuously measured. In NMPC, $p(t)$ is a state estimation and the solution w contains the control. We want to minimize time from $p(t)$ to update $w(p(t))$ (control delay). We also want to avoid linearization between the new $p(t)$ and the update of w. The algorithm for this method is outlined below.



Algorithm: Real-time path-following

```
Input: Solution $\mathbf{w}$, $\boldsymbol{\lambda}$, $\boldsymbol{\mu}$ and $H, \mathbf{g}, \nabla\mathbf{g}, \ldots$ at $\mathbf{p}$,
       new parameter $\mathbf{p}^+$ (measurement)
Take single full SQP step on NLP:
  $\mathbf{w}(\mathbf{p}^+) = \arg\min_{\mathbf{w},\mathbf{p}} \; \Phi(\mathbf{w},\mathbf{p})$
                      $\mathbf{g}(\mathbf{w},\mathbf{p}) = 0, \quad \mathbf{p} - \mathbf{p}^+ = 0$
                      $\mathbf{h}(\mathbf{w},\mathbf{p}) \leq 0$
using $H, \mathbf{g}, \nabla\mathbf{g}, \ldots$ at $\mathbf{p}$
Update $H, \mathbf{g}, \nabla\mathbf{g}, \ldots$
return $\mathbf{w}$, $\boldsymbol{\lambda}$, $\boldsymbol{\mu}$ and $H, \mathbf{g}, \nabla\mathbf{g}, \ldots$
```

1.5.1 Real-Time Iteration (RTI) for NMPC

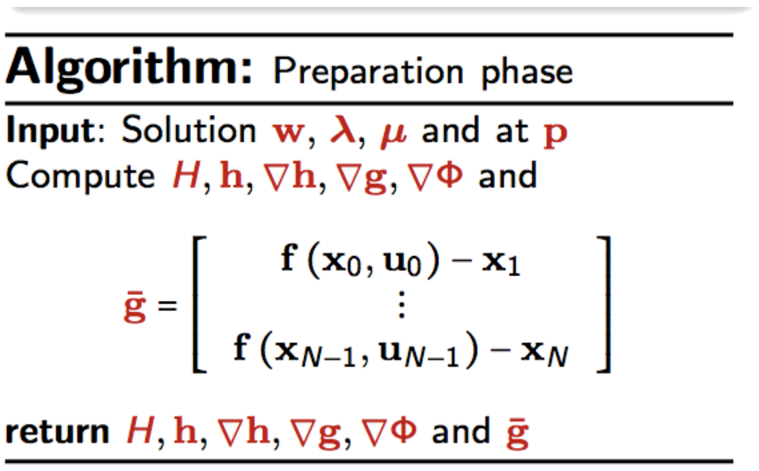
NMPC at physical time i is an NLP with $w = \{x_0, u_0, \ldots, x_{N-1}, u_{N-1}, x_N\}$

$$\min_{w} \quad \Phi(w) \tag{9a}$$

$$\text{s.t.} \quad g(w, \hat{x}_i) = \begin{bmatrix} \hat{x}_i - x_0 \\ f(x_0, u_0) - x_1 \\ \ldots \\ f(x_{N-1}, u_{N-1}) - x_N \end{bmatrix}, \tag{9b}$$

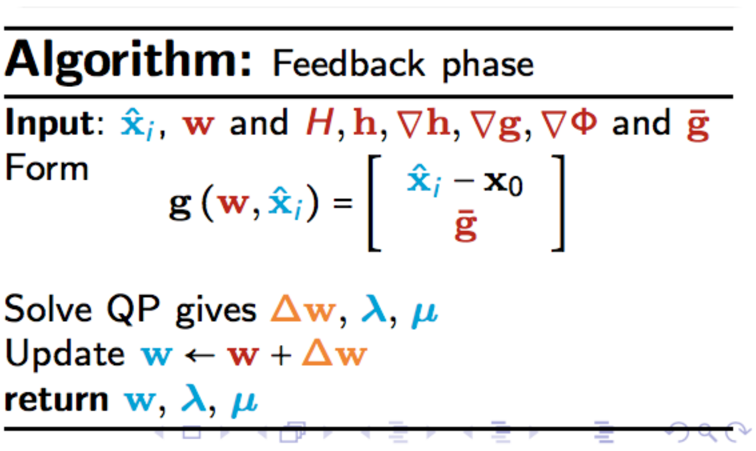
$$h(w) = \begin{bmatrix} h(x_0, u_0) \\ \ldots \\ h(x_{N-1}, u_{N-1}) \\ h(x_N) \end{bmatrix} \tag{9c}$$

Note that the initial conditions $\hat{x}_i$ at time i are already embedded in the formulation. We perform an initial preparation phase between $\hat{x}_{i-1}$ and $\hat{x}_i$:



```
Algorithm: Preparation phase
Input: Solution $\mathbf{w}$, $\boldsymbol{\lambda}$, $\boldsymbol{\mu}$ and at $\mathbf{p}$
Compute $H, \mathbf{h}, \nabla\mathbf{h}, \nabla\mathbf{g}, \nabla\Phi$ and
    $\bar{\mathbf{g}} = \begin{bmatrix} \mathbf{f}(\mathbf{x}_0, \mathbf{u}_0) - \mathbf{x}_1 \\ \vdots \\ \mathbf{f}(\mathbf{x}_{N-1}, \mathbf{u}_{N-1}) - \mathbf{x}_N \end{bmatrix}$
return $H, \mathbf{h}, \nabla\mathbf{h}, \nabla\mathbf{g}, \nabla\Phi$ and $\bar{\mathbf{g}}$
```

We then perform a feedback phase upon receiving $\hat{x}_i$:



```
Algorithm: Feedback phase
Input: $\hat{\mathbf{x}}_i$, $\mathbf{w}$ and $H, \mathbf{h}, \nabla\mathbf{h}, \nabla\mathbf{g}, \nabla\Phi$ and $\bar{\mathbf{g}}$
Form
    $\mathbf{g}(\mathbf{w}, \hat{\mathbf{x}}_i) = \begin{bmatrix} \hat{\mathbf{x}}_i - \mathbf{x}_0 \\ \bar{\mathbf{g}} \end{bmatrix}$
Solve QP gives $\Delta\mathbf{w}$, $\boldsymbol{\lambda}$, $\boldsymbol{\mu}$
Update $\mathbf{w} \leftarrow \mathbf{w} + \Delta\mathbf{w}$
return $\mathbf{w}$, $\boldsymbol{\lambda}$, $\boldsymbol{\mu}$
```

RTI reduces the control delay by moving the linearization “out of the way” into the preparation phase. The feedback phase becomes a simple QP problem to solve.

2 Sensitivity-Based Economic NMPC with a Path-following Approach

2.1 Introduction

The idea of economic model predictive control (MPC) is to integrate the economic optimization layer and the control layer in the process control hierarchy into a single dynamic optimization layer. Economic MPC adjusts the inputs to minimize the economic cost of operation directly.

Since nonlinear process models are often used for economic optimization, a potential drawback of economic MPC is that it requires solving a large-scale nonlinear optimization problem (NLP) associated with the nonlinear model predictive control (NMPC) problem at every sample time. The solution of this NLP may take a significant amount of time and this can lead to performance degradation and even to instability of the closed-loop system.

Instead of solving the full nonlinear optimization problem when new measurements of the state become available, these approaches use the sensitivity of the NLP solution at a previously-computed iteration to obtain fast approximate solutions to the new NMPC problem. These can be implemented in the plant with minimal delay.

The concept of real-time iteration (RTI), in which the full NLP is not solved at all during the MPC iterations. Instead, at each NMPC sampling time, a single quadratic programming (QP) related to the sequential quadratic programming (SQP) iteration for solving the full NLP is solved. The real-time iteration scheme contains two phases: (1) the preparation phase and (2) the feedback phase. In the preparation phase, the model derivatives are evaluated using a predicted state measurement, and a QP is formulated based on data of this predicted state. In the feedback phase, once the new initial state is available, the QP is updated to include the new initial state and solved for the control input that is injected into the plant.

A different approach known as advanced-step NMPC (asNMPC) involves solving the full NLP at each sample time. However, the full NLP solution is computed in advance for a predicted initial state. Once the new state measurement is available, the NLP solution is corrected using a fast sensitivity update to match the measured or estimated initial state.

In this case, we apply an improved path-following method for correcting the NLP solution within the advanced-step NMPC framework. We use asNMPC with a predictor-corrector path-following algorithm that is able to perform even in the presence of measurement noise.

2.2 NMPC Problem Formulations

2.2.1 The NMPC Problem

We consider a nonlinear discrete-time dynamic system:

$$x_{k+1} = f(x_k, u_k)$$

where $x_k \in \mathbb{R}^{n_x}$ denotes the state variable, $u_k \in \mathbb{R}^{n_u}$ is the control input and $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to R^{n_x}$ is a continuous model function, which calculates the next state x_{k+1} from the previous state x_k and control input u_k, where $k \in \mathbb{N}$. This system is optimized by a nonlinear model predictive controller, which solves the problem:

$$\min_{z_l, v_l} \quad \Psi(z_N) + \sum_{l=0}^{N-1} \psi(z_l, v_l) \tag{10a}$$

$$\text{s.t.} \quad z_{i+1} = f(z_l, v_l) l = 0, \ldots, N-1, \tag{10b}$$

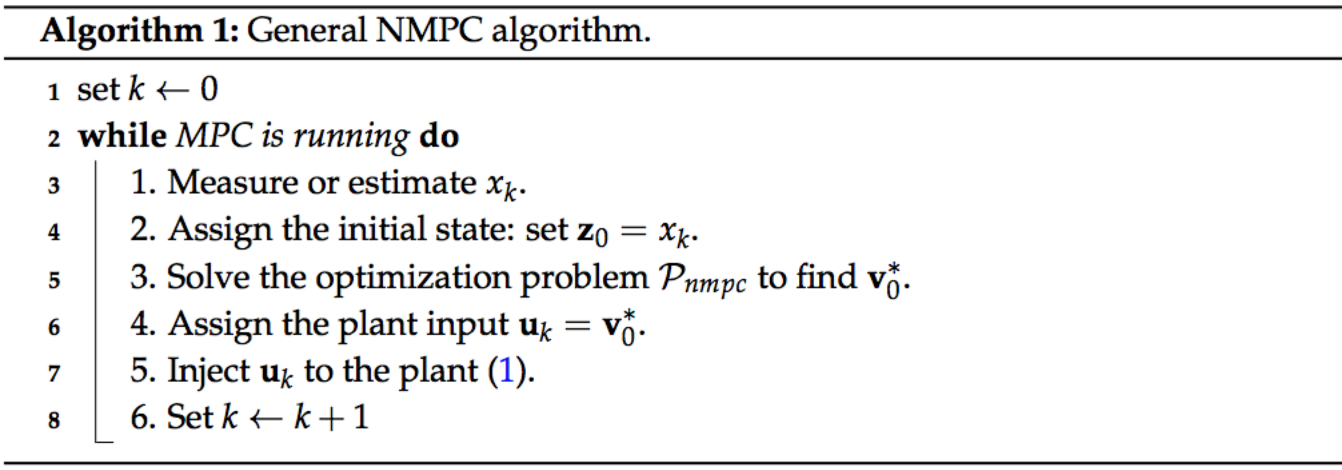
$$z_0 = x_k, \tag{10c}$$

$$(z_l, v_l) \in \mathcal{Z}, l = 0, \ldots, N-1, \tag{10d}$$

$$z_N \in X_f \tag{10e}$$

at each sample time. Here, $z_i \in \mathbb{R}^{n_x}$ is the predicted state variable; $v_l \in \mathbb{R}^{n_u}$ is the predicted control input; and $z_N \in \mathcal{X}_f$ is the final predicted state variable restricted to the terminal region $\mathcal{X}_f \in \mathbb{R}^{n_x}$. The stage cost is denoted by $\psi : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}$ and the terminal cost by $\Psi : \mathcal{X}_f \to \mathbb{R}$. Further $\mathcal{Z}$ denotes the path constraints, i.e. $\mathcal{Z} = \{(z, v) | q(z, v) \leq 0\}$, where $q : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \leftarrow \mathbb{R}^{n_q}$.

The solution of the optimization problem is denoted $\{z_0^*, \ldots, z_N^*, v_0^*, \ldots, v_{N-1}^*\}$. At sample time k, an estimate or measurement of the state x_k is obtained and the problem shown above (10e) is solved. Then, the first part of the optimal control sequence is assigned as plant input such that $u_k = v_0^*$. The first part of the solution defines an implicit feedback law and the system will evolve according to $x_{k+1} = fr(x_k, \kappa(x_k))$. At the next sample time $k+1$, when the measurement of the new state x_{k+1} is obtained, the procedure is repeated. The NMPC algorithm is summarized below.



Algorithm 1: General NMPC algorithm.

```
set $k \leftarrow 0$
while MPC is running do
    1. Measure or estimate $x_k$.
    2. Assign the initial state: set $\mathbf{z}_0 = x_k$.
    3. Solve the optimization problem $\mathcal{P}_{nmpc}$ to find $\mathbf{v}_0^*$.
    4. Assign the plant input $\mathbf{u}_k = \mathbf{v}_0^*$.
    5. Inject $\mathbf{u}_k$ to the plant (1).
    6. Set $k \leftarrow k+1$
```

2.2.2 Ideal NMPC and Advanced-Step NMPC Framework

We refer to the hypothetical case where the NMPC problem can be solved instantaneously as ideal NMPC. In practice, there will be of course some time delay between obtaining the updated values of the states and injecting the updated inputs into the plant. The main reason for this delay is the time it requires to solve the optimization problem.

This has lead to the development of fast senstivity-based NMPC approaches. We will now look at the advanced-step NMPC (asNMPC) approach, which is based on the following steps:

(a) Solve the NMPC problem at time k with a predicted state value of time $k+1$

(b) When the measurement x_{k+1} becomes available at time $k+1$, compute an approximation of the NLP solution using fast sensitivity methods

(c) Update $k \leftarrow k+1$ and repeat from Step 1

It has been proposed that a fast one-step sensitivity update that is based on solving a linear system of equations. Under some assumptions, this corresponds to a first-order Taylor approximation of the optimal solution. In particular, this approach requires strict complementarity of the NLP solution, which ensures no changes in the active set. A more general approach involves allowing for changes in the active set and making several sensitivity updates. This approach will be focused on here.

2.3 Sensitivity-Based Path-Following NMPC

2.3.1 Sensitivity Properties of NLP

The dynamic optimization problem shown in Problem (10e) can be cast as a general parametric NLP problem.

$$\begin{aligned} \min_{\chi} \quad & F(\chi, p) && \text{(11a)} \\ \text{s.t.} \quad & c(\chi, p) = 0, && \text{(11b)} \\ & g(\chi, p) \leq 0 && \text{(11c)} \end{aligned}$$

where $\chi \in \mathbb{R}^{n_\chi}$ are the decision variables (typically the state variables and the control input) and $p \in \mathbb{R}^{n_p}$ is the parameter, which is typically the initial state variable x_k. In addition $F : \mathbb{R}^{n_\chi} \times \mathbb{R}^{n_p} \rightarrow \mathbb{R}$ is the scalar objective function; $c : \mathbb{R}^{n_\chi} \times \mathbb{R}^{n_p} \rightarrow \mathbb{R}^{n_c}$ denotes the equality constraints; and finally, $g : \mathbb{R}^{n_\chi} \times \mathbb{R}^{n_p} \rightarrow \mathbb{R}^{n_g}$ denotes the inequality constraints. The instances in Problem (11c) that are solved at each sample time differ only in the parameter p.

The Lagrangian of this problem is:

$$\mathcal{L}(\chi, p, \lambda, \mu) = F(\chi, p) + \lambda^T c(\chi, p) + \mu^T g(\chi, p) \tag{12}$$

and the KKT conditions are:

$$\begin{aligned} c(x, p) = 0, \quad & g(x, p) \leq 0 \quad \text{(primal feasibility)} && \text{(13)} \\ \mu \geq 0, \quad & \text{(dual feasibility)} \\ \nabla_x \mathcal{L}(x, p, \lambda, \mu) = 0, \quad & \text{(stationary condition)} \\ \mu^T g(x, p) = 0, \quad & \text{(complementary slackness)} \end{aligned}$$

In order for the KKT conditions to be a necessary condition of optimality, we require a constraint qualification (CQ) to hold. From now on we assume that the linear independence constraint qualification (LICQ) holds:

Definition 2.1 (LICQ). *Given a vector p and a point χ, the LICQ holds at χ if the set of vectors $\{\{\nabla_\chi c_i(\chi, p)\}_{i \in \{1,\ldots,n_c\}} \cup \{\nabla_\chi g_i(\chi, p)\}_{i:g_i(\chi,p)=0}\}$ is linearly independent.*

The LICQ implies that the multipliers (λ, μ) satisfying the KKT conditions are unique. If additionally, a suitable second-order condition holds, then the KKT conditions guarantee a unique local minimum. A suitable second-order condition states that the Hessian matrix has to be positive definite in a set of appropriate directions, defined by the following property:

Definition 2.2 (SSOSC)**.** *The strong second-order sufficient condition (SSOSC) holds at χ with multipliers λ and μ if $d^T \nabla_\chi^2 \mathcal{L}(\chi, p, \lambda, \mu) d > 0$ for all $d \neq 0$, such that $\nabla_\chi c(\chi, p)^T d = 0$ and $\nabla_\chi g_i(\chi, p)^T d = 0$ for i such that $g_i(\chi, p) = 0$ and $\mu_i > 0$.*

For a given p, we denote the solution to (11c) by $\chi^*(p), \lambda^*(p), \mu^*(p)$, and if possible, we omit the argument. We want to know how the solution changes with a perturbation in the parameter p. One more important concept is outlined below:

Definition 2.3 (SC)**.** Given a vector p and a solution χ^* with vectors of multipliers λ^* and μ^*, strict complimentary (SC) holds if $\mu_i^* - g_i(\chi^*, p) > 0$ for each $i = 1, \ldots, n_g$.

We now look at the result.

Theorem 2.1 (Implicit function theorem applied to optimality conditions)**.** *Let $\chi^*(p)$ be a KKT point that satisfies (13), and assume that LICQ, SSOSC, and SC hold at χ^*. Further, let the function F, c, g be at least $k+1$ times differentiable in χ and k-times differentiable in p. Then:*

- *χ^* is an isolated minimizer, and the associated multipliers λ and μ are unique.*
- *for p in a neighborhood of p_0, the set of active constraints remains unchanged*
- *for p in a neighborhood of p_0, there exists a $k-$ times differentiable function $\sigma(p) = \begin{bmatrix} \chi^*(p)^T & \lambda^*(p)^T & \mu^*(p)^T \end{bmatrix}$, that corresponds to a locally unique minimum for (13).*

Using this result, the sensitvity of the optimal solution χ^*, λ^*, μ^* in a small neighborhood of p_0 can be computed by solving a system of linear equations that arises from applying the implicit function theorem to the KKT conditions:

$$\begin{bmatrix} \nabla^2_{\chi\chi} \mathcal{L}(\chi^*, p_0, \lambda^*, \mu^*) & \nabla_\chi c(\chi^*, p_0) & \nabla_\chi g_A(\chi^*, p_0) \\ \nabla_\chi c(\chi^*, p_0)^T & 0 & 0 \\ \nabla_\chi g_A(\chi^*, p_0)^T & 0 & 0 \end{bmatrix} \begin{bmatrix} \nabla_p \chi \\ \nabla_p \lambda \\ \nabla_p \mu \end{bmatrix} = - \begin{bmatrix} \nabla^2_{p\chi} \mathcal{L}(\chi^*, p_0, \lambda^*, \mu^*) \\ \nabla_p c(\chi^*, p_0) \\ \nabla_p g_A(\chi^*, p_0) \end{bmatrix} \tag{14}$$

Here, the constraint gradients with subscript g_A indicate that we only include the vectors and components of the Jacobian corresponding to the active inequality constraints at χ, i.e., $i \in A$ if $g_i(\chi, p_0) = 0$. Denoting the solution of the equation above as $\begin{bmatrix} \nabla_p \chi & \nabla_p \lambda & \nabla_p \mu \end{bmatrix}^T$, for small Δp, we obtain a good estimate:

$$\chi(p_0 + \Delta p) = \chi^* + \nabla_p \chi \Delta p \tag{15}$$
$$\lambda(p_0 + \Delta p) = \lambda^* + \nabla_p \lambda \Delta p \tag{16}$$
$$\mu(p_0 + \Delta p) = \mu^* + \nabla_p \mu \Delta p \tag{17}$$

of the solution to the NLP Problem (11c) at the parameter value $p_0 + \Delta p$.

If Δp becomes large, the approximate solution may no longer be accurate enough because the SC assumption implies that the active set cannot change. While this is true for small perturbations, large changes in Δp may very well induce active set changes.

It can be seen that the sensitivity system corresponds to the stationary conditions for a particular QP. If Δp is small enough, we can show that the set $\{i : \mu(\bar{p})_i > 0\}$ is constant for $\bar{p} = p_0 + \Delta p$. Thus we can form a QP wherein we are potentially moving off of weakly-active constraints while staying on the strongly-active ones. The primal-dual solution of this QP is in fact the directional derivative of the primal-dual solution $\chi^*(p), \lambda^*(p), \mu^*(p)$.

Theorem 2.2. *Let F, c, g be twice continuously differentiable in p and χ near (χ^*, p_0) and let the LICQ and SSOSC hold at (χ^*, p_0). Then, the solution $(\chi^*(p), \lambda^*(p), \mu^*(p))$ is Lipschitz continuous in a neighborhood of $(\chi^*, \lambda^*, \mu^*, p_0)$ and the solution function $(\chi^*(p), \lambda^*(p), \mu^*(p))$ is directionally differentiable.*

Moreover, the directional derivative uniquely solves the following quadratic problem:

$$\min_{\Delta\chi} \quad \frac{1}{2}\Delta\chi^T \nabla^2_{\chi\chi}\mathcal{L}(\chi^*, p_0, \lambda^*, \mu^*)\Delta\chi + \Delta\chi^T \nabla^2_{p\chi}\mathcal{L}(\chi^*, p_0, \lambda^*, \mu^*)\Delta p \tag{18a}$$

$$\text{s.t.} \quad \nabla_\chi c_i(\chi^*, p_0)^T \Delta\chi + \nabla_p c_i(\chi^*, p_0)^T \Delta p = 0, \qquad i = 1, \dots, n_c, \tag{18b}$$

$$\nabla_\chi g_j(\chi^*, p_0)^T \Delta\chi + \nabla_p g_j(\chi^*, p_0)^T \Delta p = 0, \qquad j \in K_+, \tag{18c}$$

$$\nabla_\chi g_j(\chi^*, p_0)^T \Delta\chi + \nabla_p g_j(\chi^*, p_0)^T \Delta p \leq 0, \qquad j \in K_0 \tag{18d}$$

where $K_+ = \{j \in \mathbb{Z} : \mu_j > 0\}$ is the strongly-active set and $K_0 = \{j \in \mathbb{Z} : \mu_j = 0$ and $g_j(\chi^*, p_0) = 0\}$ denotes the weakly active set.

The theorem above gives the solution of the perturbed NLP by solving a QP problem. Note that as the solution to this QP is the directional derivative of the primal-dual solution of the NLP, it is a predictor step, a tangential first-order estimate of the change in the solution subject to a change in the parameter. We refer to the QP (18d) as a pure-predictor. Note that obtaining the sensitivity via this method has the advantage that changes in the active set can be accounted for correctly and strict complementarity (SC) is not required.

2.3.2 Path-Following Based on Sensitivity Properties

Equation 14 and the QP (18d) describes the change in the optimal solutions for small perturbations. They are not guaranteed to reproduce the optimal solution accurately for larger perturbations. One approach to handle such cases is to divide the overall perturbation into several smaller intervals and to iteratively use the sensitivity to track the path of optimal solutions.

The general idea of a path-following method is to reach the solution of the problem at a final parameter value p_f by tracing a sequence of solutions $(\chi_k, \lambda_k, \mu_k)$ for a series of parameter values $p(t_k) = (1 - t_k)p_0 + t_k p_f$ with $0 = t_0 < t_1 < \dots < t_k < \dots < t_N = 1$. The new direction is found by evaluating the sensitivity at the current point (i.e. similar Euler integration for ODE).

A path-following algorithm that is only based on the sensitivity calculated by the pure predictor QP may fail to track the solution accurately enough and may lead to poor solutions. A common approach to address this problem is to include elements that are similar to a Newton step, which force the path-following algorithm towards the true solution. If we approximate (11c)by a QP, linearizing with respect to both χ and p, and enforce the equality of the strongly-active constraints:

$$\min_{\Delta\chi\Delta p} \quad \tfrac{1}{2}\Delta^2_{\chi\chi}\mathcal{L}(\chi^*,p_0,\lambda^*,\mu^*)\Delta\chi + \Delta\chi^T\nabla^2_{p\chi}\mathcal{L}(\chi^*,p_0,\lambda^*,\mu^*)\Delta p + \nabla_\chi F^T\Delta\chi + \nabla_p F\Delta p + \tfrac{1}{2}\Delta p^T\nabla^2_{pp}\mathcal{L}(\chi^*,p_0,\lambda^*,\mu^*)\Delta p \tag{19a}$$

$$\text{s.t.} \quad c_i(\chi^*, p_0) + \nabla_\chi c_i(\chi^*, p_0)^T \Delta\chi + \nabla_p c_i(\chi^*, p_0)^T \Delta p = 0 \qquad i = 0, \dots, n_c, \tag{19b}$$

$$g_j(\chi^*, p_0) + \nabla_\chi g_j(\chi^*, p_0)^T \Delta\chi + \nabla_p g_j(\chi^*, p_0)^T \Delta p = 0 \qquad j \in K_+, \tag{19c}$$

$$g_j(\chi^*, p_0) + \nabla_\chi g_j(\chi^*, p_0)^T \Delta\chi + \nabla_p g_j(\chi^*, p_0)^T \Delta p \leq 0 \qquad j \in \{1, \dots, n_g\}/K_+ \tag{19d}$$

In the NMPC problem, the parameter p corresponds to the current "initial" state x_k. Moreover, the cost function is independent of p and we have that $\nabla_p F = 0$. We then get that $\nabla_p c$

and $\nabla_p g$ are constants. Thus the above QP formulation can simplify to:

$$\min_{\Delta\chi\Delta p} \quad \frac{1}{2}\Delta^2_{\chi\chi}\mathcal{L}(\chi^*, p_0, \lambda^*, \mu^*)\Delta\chi + \nabla_\chi F^T \Delta\chi \tag{20a}$$

$$\text{s.t.} \quad c_i(\chi^*, p_0) + \nabla_\chi c_i(\chi^*, p_0)^T \Delta\chi = 0 \qquad i = 0, \dots, n_c, \tag{20b}$$

$$g_j(\chi^*, p_0) + \nabla_\chi g_j(\chi^*, p_0)^T \Delta\chi = 0 \qquad j \in K_+, \tag{20c}$$

$$g_j(\chi^*, p_0) + \nabla_\chi g_j(\chi^*, p_0)^T \Delta\chi \leq 0 \qquad j \in \{1, \dots, n_g\}/K_+ \tag{20d}$$

We denote the QP formulation (20d) as the predictor-corrector. While this appears similar to the QP proposed in the real-time iteration scheme, we note that it differs because here we enforce the strongly-active constraints as as equality constraints in the QP.

The predictor-corrector QP (20d) is well suited for use in a path-following algorithm, where the optimal solution path is tracked from p_0 to a final value p_f along a sequence of parameter points $p(t_k) = (1 - t_k)p_0 + t_k p_f$ with $0 = t_0 < t_1 < \ldots < t_k < \ldots < t_N = 1$. At each point $p(t_k)$, the QP is solved and the primal-dual solutions updated as:

$$\chi(t_{k+1}) = \chi(t_k) + \Delta\chi \tag{21}$$

$$\lambda(t_{k+1}) = \Delta\lambda \tag{22}$$

$$\mu(t_{k+1}) = \Delta\mu \tag{23}$$

where $\Delta\chi$ is obtained from the primal solution of QP (20d) and where $\Delta\lambda$ and $\Delta\mu$ correspond to the Lagrange multipliers of QP (20d).

Changes in the active set along the path are detected by the QP as follows: If a constraint becomes inactive at some point along the path, the corresponding multiplier μ_j will first become weakly active, i.e, it will be added to the set K_0. Since this is not included as an equality constraint, the next QP solution could move away from the constraint. Similarly, if a new constraint g_j becomes active along the path, it will make the corresponding linearized inequality constraint in the QP active.

The resulting path-following algorithm is summarized with its main steps in Algorithm 2.

Algorithm 2: Path-following algorithm.

```
Input: initial variables from NLP $\chi^*(\mathbf{p}_0), \lambda^*(\mathbf{p}_0), \mu^*(\mathbf{p}_0)$
fix stepsize $\triangle t$, and set $N = \frac{1}{\Delta t}$
set initial parameter value $\mathbf{p}_0$,
set final parameter value $\mathbf{p}_f$,
set $t = 0$,
set constant $0 < \alpha_1 < 1$.
Output: primal variable $\chi$ and dual variables $\lambda, \mu$ along the path
1  for $k \leftarrow 1$ to $N$ do
2      Compute step $\Delta\mathbf{p} = \mathbf{p}_k - \mathbf{p}_{k-1}$
3      Solve QP problem ;                         /* to obtain $\Delta\chi, \Delta\lambda, \Delta\mu$ */
4      if QP is feasible then
5          /* perform update                                        */
6          $\chi \leftarrow \chi + \Delta\chi$;               /* update primal variables */
7          Update dual variables appropriately; using Equations (8) and 9 for the pure-predictor
           method or (14) and (15) for the predictor-corrector method
8          $t \leftarrow t + \Delta t$ ;                          /* update stepsize */
9          $k \leftarrow k + 1$
10     else
11         /* QP is infeasible, reduce QP stepsize                  */
12         $\triangle t \leftarrow \alpha_1 \triangle t$
13         $t \leftarrow t - \alpha_1 \triangle t$
```

2.3.3 Discussion of the Path-Following asNMPC Approach

We now concentrate on the advanced-step NMPC framework, i.e., at every time step, the full NLP is solved for a predicted state. When the new measurement becomes available, the precomputed NLP solution is updated by tracking the optimal solution curve from the predicted initial state to the new measured or estimated state. Note that any numerical homotopy algorithm can be used to update the NLP solution. The solution of the last QP along the path corresponds to the updated NLP solution and only the inputs computed in this last QP will be injected into the plant.

This situation differs from the RTI scheme previously described where the NLP is not solved at all during the MPC sampling times. Instead, at each sampling time, a single QP is solved and the computed input is applied to the palnt. This requires rapid sampling times, and if the QP fails to track the true solution then other measures have to be performed to get the controller "on-track" again.

In comparison to previously published studies, strongly- and weakly- active inequality constraints are distinguished. Strongly-active inequalities are included as linearized equality contraints in the QP and weakly-active constraints are linearized and added as inequality constraints to the QP. This helps ensure that the true solution path is tracked more accurately when dealing with a non-convex full Hessian.

2.4 Numerical Case Study

2.4.1 Process Description

We now look at the path-following NMPC (pf-NMPC) on an isothermal reactor and separator process depicted in Figure 2. The CSTR is fed with a stream F_0 containing 100% component $\mathcal{A}$ and a recycler R from the distillation column. A first-order reaction A $\longrightarrow$ B takes place in the CSTR where $\mathcal{B}$ is the desired product and the product with flow rate F is fed to the column. In the distillation column, the unreacted raw material is separated from the product and recycled back to the reactor. The desired product $\mathcal{B}$ leaves the distillation column as the bottom product, which has a purity requirement. The model for the process has 84 state variables of which 82 of which are from the distillation and two from the CSTR.

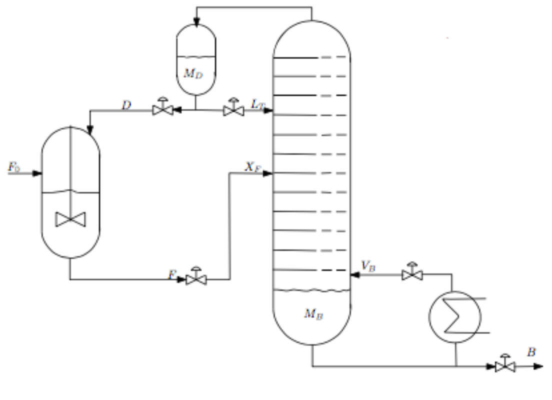


Figure 2: CSTR and distillation column

The stage cost of the economic objective function to optimize under operation is:

$$J = p_F F_0 + p_V V_B - p_B B \tag{24}$$

where p_F is the feed cost p_V is the steam cost and p_B is the product price. The operational constraints are the concentration of the bottom product as well as the liquid holdup at the bottom and top of the distillation column and in the CSTR. The control inputs are the reflux flow (L_T), boil-up flow (V_B), feeding rate to the distillation (F), distillate (top) and bottom product flow rates (D and B).

First we ran a steady-state optimization with a feed rate of $F_0 = 0.3(\text{kmol}\,\text{min}^{-1})$. This gives us the optimal values for control inputs and state variables; the optimal steady state input values are $u_s = \begin{bmatrix}1.18 & 1.92 & 1.03 & 0.74 & 0.29\end{bmatrix}^T$. These values are used to construct the regularization term added to the objective function. We thus end up with the regularized stage:

$$J_m = p_F F_0 + p_V V_B - p_B B - p_D D + (z - x_s)^T Q_1 (z - x_s) + (v - u_s)^T Q_2 (v - u_s) \tag{25}$$

The weights Q_1 and Q_2 are selected to make the rotated stage cost of the steady state problem strongly convex; this ensures that the NMPC controller is stable.

We then set up the NLP for calculating the predicted state variables z and predicted control inputs v. We use direct collocation on finite elements using Lagrange collocation to discretize the dynamics, using three colocation points in each finite element. This allows the state variables and control inputs to become optimization variables.

The economic NMPC case study is simulated with 150 MPC iterations with a sample time of 1 min. The prediction horizon is set to 30 min. These settings result in an NLP with 10,314 optimization variables. CasADi and IPOPT are used as the NLP solver. The QPs are solved using MINOS QP from TOMLAB.

2.4.2 Open-Loop Results

We observe that the one-step pure-predictor tracks the ideal NMPC solution the worst and the four-step path-following with predictor-corrector tracks best. This is due to the fact that the predictor-correct path-following QP has an additional linear term in the objective function and constraint for the purpose of moving closer to the solution of the NLP, as well as tracing the first-order estimate of the change in the solution.

In this case study, we note that a single predictor-corrector step has almost as good performance as the four predictor-corrector steps along the path.

2.4.3 Closed-Loop Results

If no measurement noise is present, then we note that the prediction and the true solution differ only due to numerical noise. This means that there is no need to update the predication, so all approaches give the exact same closed-loop behavior.

When we have measurement noise on all of the holdups in the system, we note that it is no longer possible to avoid the violation of the active constraints in the holdup of the CSTR and the bottom composition in the distillation column. However, fast sensitivity NMPC approaches are still very close to the ideal NMPC inputs. If we look at the accumulated stage cost we see that the proposed predictor-corrector path-following algorithm performs identically to the ideal NMPC.

2.5 Discussion and Conclusion

This method of predictor-corrector path-following asNMPC has the advantage of allowing for active set changes to be handled rigorously. In addition, solving a sequence of a few QPs is much faster than solving the full NLP such that we have a very small computational delay.

While it is true that path-following algorithms may get lost, the application of them in the asNMPC framework has the desirable property that the solution of the full NLP acts as a correct, such that if the algorithm diverges from the true solution, this will most likely only be for one sample time, before the next full NLP is solved.

3 Sequential quadratic programming methods for parametric nonlinear optimization Kungurtsev and Diehl

3.1 Introduction

We will analyze properties of parametric exact Hessian sequential quadratic programming (SQP) methods. SQP methods are well known to have desirable properties, in contrast to interior point methods. This is due to the fact that in conventional SQP method, if the active set has stabilized, the algorithm behaves like Newton's method on the optimality conditions that are satisfied at the solution. However, a QP solver may not estimate the optimal active set. Furthermore, typical NLP solvers may modify the Hessian or use a positive definite quasi-Newton variant in order to prevent the possibility of forming nonconvex QPs. We therefore propose an exact second-derivative procedure that is able to take advantage of fast local convergence without attempting to solve nonconvex QPs. We will use robust active set estimates and constrain the QP problem to the active sets, allowing the use of exact Hessians with each QP subproblem lying in the Newton domain of convergence in a sequence of homotopy steps.

We consider a parameteric nonconvex nonlinear optimization problem of the form

$$\min_{x \in \mathbb{R}^n} \quad f(x,t) \tag{26a}$$

$$\text{s.t.} \quad c(x,t) \geq 0 \tag{26b}$$

where $f : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}, c : \mathbb{R}^n \times \mathbb{R}^m$ are nonlinear two times Lipschitz continuously differentiable functions and $t \in [0,1]$.

We recall the definition of the first-order optimality conditions:

Definition 3.1 (First-order necessary conditions)**.** A primal-dual pair (x^*, y^*) satisfies the first-order necessary optimality conditions for (26b) if,

$$\begin{aligned} \nabla_x f(x^*,t) &= J(x^*,t)^T y^* \\ c(x^*,t) &\geq 0 \\ c(x^*,t)^T y^* &= 0 \\ y^* &\geq 0 \end{aligned} \tag{27}$$

where $J(x,t)$ denotes the Jacobian constraint matrix for problem t at x.

These conditions necessarily hold for any local minimzer x^* which satisfies a constraint qualification.

We also define the second-order sufficiency condition.

Definition 3.2 (Second-order sufficient conditions (SOSC))**.** A primal-dual pair (x^*, y^*) satisfies the second-order sufficient optimality conditions if it satisfies the first-order conditions (27) and

$$d^T H(x^*, y^*, t)d > 0 \qquad \text{for all} \qquad d \in \mathcal{C}(x^*, y^*, t)/\{0\} \tag{28}$$

where $d \in \mathcal{C}(x^*, y^*, t)$ if $\nabla c_i(x^*, t)^T d = 0$ for $i \in \mathcal{A}_+(x^*, y^*, t)$ and $\nabla c_i(x^*, t)^T d \geq 0$ for $i \in \mathcal{A}_0(x^*, y^*, t)$

If we want strict complementarity to hold, then $\mathcal{A}_0(x^*, y^*, t) = \emptyset$ must hold.

We make one main assumption in this paper:

Assumption 3.1. *The functions $f(x, t)$ and $c(x, t)$ are two times Lipschitz continuously differentiable for all x and $t \in [0, 1]$ with respect to both x and t.*

We define the distance of a point to the nearest primal-dual solution by

$$\delta(x, y, t) = \sqrt{\|x - x^*(t)\|^2 + \text{dist}(y, \Lambda(x^*(t)))^2}$$

where $x^*(t)$ is the closest primal solution to (26b) at t and $\Lambda(x^*(t))$ is the set of Lagrange multipliers $\{y^*\}$ satisfying (27) together with x^*.

The optimality residual $\eta(x, y, t)$ is defined as

$$\eta(x, y, t) = \left\| \begin{pmatrix} \nabla f(x, t) - J(x, t)^T y \\ \min(c(x, t), y) \end{pmatrix} \right\|_\infty$$

These two quantities bound each other under the SOSC.

Lemma 3.1. *If the second-order sufficiency condition for optimality holds at $(x^*, y^*)(t)$, then for (x, y) sufficiently close to $(x^*, y^*)(t)$, there exist constants $C_1(t) > 0$ and $C_2(t) > 0$ such that it holds that*

$$C_1(t)\delta(x, y, t) \leq \eta(x, y, t) \leq C_2(t)\delta(x, y, t)$$

Alternatively, if the second-order sufficiency condition holds for all $(x^*, y^*)(t)$, with y^* in a compact subset of $\Lambda(x^*)$ and (x, y) is sufficiently close to this subset, the estimate also holds.

3.2 Predictor corrector methods for parametric nonlinear optimization

We now consider the more general problem of finding a solution of $F(x, 1) = 0$, given a known $\bar{x}$ such that $F(\bar{x}, 0) = 0$. Predictor-corrector methods are a class of methods that include a predictor step that initially attempts to estimate the solution and a corrector step to refine the estimate.

At the simplest level, for a homotopic method to be successful at the minimum it should have a sequence of Newton iterates as described below in the algorithm

```
Newton corrector homotopy.
1: Begin with $x$ a solution to $F(x, t) = 0$ at $t = 0$. Initialize $\Delta t = 1$.
2: Choose constants $\gamma_1$ and $\gamma_2$ satisfying $0 < \gamma_1 < 1$ and $\gamma_2 > 1$.
3: while $t < 1$ do
4:     Solve $\Delta x = -F'(x, t + \Delta t)^{-1} F(x, t + \Delta t)$.
5:     if $\|F(x + \Delta x, t + \Delta t)\|$ is sufficiently small then
6:         Set $t = t + \Delta t$.
7:         Set $x = x + \Delta x$.
8:         Let $\Delta t = \min(\gamma_2 \Delta t, 1 - t)$.
9:     else
10:        Let $\Delta t = \gamma_1 \Delta t$.
11:    end if
12: end while
```

In this case, there is only a corrector step. The initial point at which Newton's method is applied is the original x. If x happens to lie in the Newton domain of rapid local convergence for the problem at t_2, then the procedure is effective at generating a point that sufficiently contracts $\|F(x,t)\|$ for $t_2 - t_2$ small enough that the domain of convergence does not change much.

However, if we include a predictor step, we incorporate first order information in order to estimate the location of the solution at t_2. Here if the first-order estimate is sufficiently accurate, we may take a large step along t and still achieve the desired level of contraction in $\|F\|$ by staying within the domain of local convergence. The algorithm below uses a canonical predictor-corrector algorithm using one step of a linear tangential predictor and a step of Newton's method.

Algorithm 2.1 Predictor–corrector.

```
1: Begin with $x$ a solution to $F(x,t) = 0$ at $t = 0$. Initialize $\Delta t = 1$.
2: Choose constants $\gamma_1$ and $\gamma_2$ satisfying $0 < \gamma_1 < 1$ and $\gamma_2 > 1$.
3: while $t < 1$ do
4:     Calculate $d = \frac{\partial x(t)}{\partial t}$.
5:     Set $\widehat{x} = x + \Delta t d$.
6:     Solve $\Delta x = -F'(\widehat{x}, t + \Delta t)^{-1} F(\widehat{x}, t + \Delta t)$.
7:     if $\|F(\widehat{x} + \Delta x, t + \Delta t)\|$ is sufficiently small then
8:         Set $t = t + \Delta t$.
9:         Set $x = \widehat{x} + \Delta x$.
10:        Let $\Delta t = \min(\gamma_2 \Delta t, 1 - t)$.
11:    else
12:        Let $\Delta t = \gamma_1 \Delta t$.
13:        Go to Step 5
14:    end if
15: end while
```

This is the Euler-Newton method using exact function and derivative definitions.

3.2.1 Predictors for parametric NLP

In the context of the parametric NLP, the direction of the predictor is the first-order estimate of the change in x and y that would solve the problem corresponding to the change in the parameter p. it also indicates which constraints become inactive with the parameter change. We shall explore a strategy where a predictor and corrector are computed together in one QP subproblem. In a general sense, the predictor estimates the newly active or inactive constraints and the corrector refines the step on a properly reduced space with a Newton type iteration.

The directional derivative provides a one-sided estimate of the active set. In particular, for small perturbations of an NLP, the optimal inactive constraints remain inactive and a subset of the weakly active constraints may become inactive. The estimation of newly active constraints is done either by linearized constraints at the QP subproblem of the predicted point, or a separate active set estimation procedure. One estimate of the active set is given by

$$\mathcal{A}_\gamma(x,y,t) = \{i : c_i(x,t) \le \eta(x,y,t)^\gamma\} \tag{29}$$

where γ is a scalar satisfying $0 < \gamma < 1$. This estimate is used in the case of the separate predictor and correct step strategy algorithm.

We will now look at the directional differentiability of primal-solution pairs. Consider a step $(\Delta t, \Delta x, \Delta y)$ for a fixed Δt from a base point $(t, \bar{x}, \bar{y})$.