Vorbereitung

Diskretisierung des Optimierungsproblems. Für jeden Zeitschritt k erhalten wir ein Problem $P^k(x_k)$

$$\min_{\substack{s_k,\dots,s_N\\ q_k,\dots,q_N}} \sum_{i=k}^{N-1} f_i(s_i,q_i) \quad s.t. \quad \begin{cases} x_k - s_k = 0 \\ h_i(s_i,q_i) - s_{i+1} = 0 \quad \forall i = k,\dots,N-1 \end{cases}$$

Mit

$$h_i(s_i,q_i) = s_i + \Delta t g_i(s_i,q_i)$$

In diesem Fall steht g für die diskretisierte ODE

$$\dot{x}_i = g_i(s_i,q_i)$$

Die zu den Problemen $P^k(x_k)$ gehörenden Lagrangegleichungen lauten wie folgt:

$$L^k(y) = \sum_{i=k}^{N-1} f_i(s_i,q_i) + \lambda_k^T(x_k - s_k) + \sum_{i=k}^{N-1} \lambda_{i+1}^T(h_i(s_i,q_i) - s_{i+1})$$

In dieser Lagrangegleichung wird $y := (\lambda_k, s_k, q_k, \lambda_{k+1}, s_{k+1}, q_{k+1}, \dots, \lambda_N, s_N)$ verwendet. Mit KKT-Bedingung

$$\nabla_y L^k(y) = 0$$

und das exakt Newton-Raphson-Verfahren

$$y_{i+1} = y_i + \Delta y_i$$

bei dem jedes Δy_i die Lösung des linearen approximierten Systems

$$\nabla_y L^k(y_i) + J^k(y_i)\Delta y_i = 0$$

ist.

Pseudo Algorithmus

Suche Lösung für das Problem

$$\min \sum_{i=k}^{N-1} \frac{1}{2}\|l_i(s_i,q_i)\|_2^2 + \frac{1}{2}\|e(s_N)\|_2^2$$

0. Wähle Startschätzung $y^0 := (\lambda_0, s_0, q_0, \lambda_1, s_1, q_1, ..., \lambda_{N-1}, s_{N-1}, q_{N-1}, \lambda_N, s_N) \in R^{N(n_\lambda+n_s+n_q)+(n_{\lambda_N}+n_{s_N})}$

Für $k = 0, 1, 2, ...$

1. Berechne $\nabla_y L^k(y^k)$ ohne erste Komponente x_k und berechne

$$J^k(y^k) = \begin{pmatrix}
 & -E & & & & & & & & & & \\
-E & Q_k^H & M_k^H & A_k^T & & & & & & & & \\
 & (M_k^T)^H & R_k^H & B_k^T & & & & & & & & \\
 & A_k & B_k & & -E & & & & & & & \\
 & & & -E & Q_{k+1}^H & M_{k+1}^H & A_{k+1}^T & & & & & \\
 & & & & (M_{k+1}^T)^H & R_{k+1}^H & B_{k+1}^T & & & & & \\
 & & & & A_{k+1} & B_{k+1} & & & & & & \\
 & & & & & & & \ddots & & & & \\
 & & & & & & \ddots & Q_{N-1}^H & M_{N-1}^H & A_{N-1}^T & & \\
 & & & & & & & (M_{N-1}^T)^H & R_{N-1}^H & B_{N-1}^T & & \\
 & & & & & & & A_{N-1} & B_{N-1} & & -E & \\
 & & & & & & & & & -E & Q_N^H &
\end{pmatrix}$$

Mit $A_i := \dfrac{\partial h_i}{\partial s_i}$, $B_i := \dfrac{\partial h_i}{\partial q_i}$, $\begin{pmatrix} Q_i^H & M_i^H \\ (M_i^H)^T & R_i^H \end{pmatrix} := \left(\dfrac{\partial l_i(s_i,q_i)}{\partial(s_i,q_i)}\right)^T \left(\dfrac{\partial l_i(s_i,q_i)}{\partial(s_i,q_i)}\right)$,
$Q_N^H := \left(\dfrac{\partial e(s_N)}{\partial s_N}\right)^T \left(\dfrac{\partial e(s_N)}{\partial s_N}\right)$

2. Löse $(J^k(y^k))^{-1}\nabla_{\lambda_k} L^k(y^k)$ soweit wie möglich ohne x_k (Riccati Recursion)

3. x_k bekannt

4. Löse $\Delta y^k = (J^k(y^k))^{-1}\nabla_{y_k} L^k(y^k)$

5. Gib $u_k := q_k + \Delta q_k$ an das System

6. STOP, wenn $k = N - 1$

7. Berechne $y^{k+1} = \prod^{k+1}(y^k + \Delta y^k) \in R^{(N-k-1)(n_\lambda+n_s+n_q)+(n_{\lambda_N}+n_{s_N})}$

Konkretes Beispiel

Löse das Problem zur minimalen Steuerung l_i.

$$\min_{\substack{s_k,\dots,s_N\\ q_k,\dots,q_N}} \sum_{i=k}^{N} \frac{1}{2}\|S_i(s_i)\|_2^2 + \sum_{i=k}^{N-1} \frac{1}{2}\|l_i(q_i)\|_2^2 \quad s.t. \quad \begin{cases} x_k - s_k = 0 \\ h_i(s_i, q_i) - s_{i+1} = 0 \quad \forall i = k, \dots, N-1 \end{cases}$$

S_i ist in diesem Fall eine Penaltyfunktion, die für die Abweichung vom Kurs bestraft.

0. Wähle Startschätzung $y^0 := (\lambda_0, s_0, q_0, \lambda_1, s_1, q_1, \dots, \lambda_{N-1}, s_{N-1}, q_{N-1}, \lambda_N, s_N)$

Für $k = 0, 1, 2, \dots$

1. Berechne $\nabla_y L^k(y^k)$ ohne erste Komponente

$$\nabla_y L^k(y^k) = \begin{pmatrix} \nabla_{\lambda_k} L^k(y^k) \\ \hline \nabla_{s_k} L^k(y^k) \\ \nabla_{q_k} L^k(y^k) \\ \vdots \\ \nabla_{\lambda_{N-1}} L^k(y^k) \\ \nabla_{s_{N-1}} L^k(y^k) \\ \nabla_{q_{N-1}} L^k(y^k) \\ \nabla_{\lambda_N} L^k(y^k) \\ \nabla_{s_N} L^k(y^k) \end{pmatrix}$$

und berechne Komponenten von $J^k(y^k)$

$$\begin{aligned} A_i &= \frac{\partial h_i(s_i, q_i)}{\partial s_i} = (\nabla_{s_i} h_i)^T \\ B_i &= \frac{\partial h_i(s_i, q_i)}{\partial q_i} = (\nabla_{q_i} h_i)^T \\ Q_i &= \nabla_{s_i} S_i * (\nabla_{s_i} S_i)^T \\ M_i &= 0 \\ R_i &= \nabla_{q_i} l_i(q_i) * (\nabla_{q_i} l_i(q_i))^T \end{aligned}$$

In diesem Beispiel hängt l_i nicht von s_i ab, daher ist M_i gleich Null.

2. Löse $(J^k(y^k))\Delta y^k = -\nabla_{\lambda_k} L^k(y^k)$ mit Riccati Recursion.
Betrachte

$$\begin{pmatrix} -E & Q_{N-1} & 0 & A_{N-1}^T & \\ & 0 & R_{N-1} & B_{N-1}^T & \\ & A_{N-1} & B_{N-1} & & -E \\ & & & -E & Q_N \end{pmatrix} \begin{pmatrix} \Delta\lambda_{N-1} \\ \Delta s_{N-1} \\ \Delta q_{N-1} \\ \Delta\lambda_N \\ \Delta s_N \end{pmatrix} = - \begin{pmatrix} \nabla_{s_{N-1}} L^k(y^k) \\ \nabla_{q_{N-1}} L^k(y^k) \\ \nabla_{\lambda_N} L^k(y^k) \\ \nabla_{s_N} L^k(y^k) \end{pmatrix}$$

Zur einfacheren Schreibweise ist ab jetzt $\nabla_{s_N} := -\nabla_{s_N} L^k(y^k)$

$$\begin{pmatrix} 0 & A_{N-1} & B_{N-1} \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \Delta\lambda_{N-1} \\ \Delta s_{N-1} \\ \Delta q_{N-1} \end{pmatrix} + \begin{pmatrix} 0 & -E \\ -E & Q_N \end{pmatrix} \begin{pmatrix} \Delta\lambda_N \\ \Delta s_N \end{pmatrix} = \begin{pmatrix} \nabla_{\lambda_N} \\ \nabla_{s_N} \end{pmatrix}$$

Für das Verfahren setzen wir $P_N = Q_N$.

$$\begin{aligned}
\begin{pmatrix}\Delta\lambda_N \\ \Delta s_N\end{pmatrix} &= \begin{pmatrix}0 & -E \\ -E & P_N\end{pmatrix}^{-1} \left[\begin{pmatrix}\nabla_{\lambda_N} \\ \nabla_{s_N}\end{pmatrix} - \begin{pmatrix}0 & A_{N-1} & B_{N-1} \\ 0 & 0 & 0\end{pmatrix}\begin{pmatrix}\Delta\lambda_{N-1} \\ \Delta s_{N-1} \\ \Delta q_{N-1}\end{pmatrix}\right] \\
&= \begin{pmatrix}-P_N & -E \\ -E & \end{pmatrix} \left[\begin{pmatrix}\nabla_{\lambda_N} \\ \nabla_{s_N}\end{pmatrix} - \begin{pmatrix}0 & A_{N-1} & B_{N-1} \\ 0 & 0 & 0\end{pmatrix}\begin{pmatrix}\Delta\lambda_{N-1} \\ \Delta s_{N-1} \\ \Delta q_{N-1}\end{pmatrix}\right]
\end{aligned}$$

Dann werden $\Delta\lambda_{N-1}$,Δs_{N-1} und Δq_{N-1} gelöst.

$$\begin{pmatrix}-E & Q_{N-1} + A_{N-1}^T P_N A_{N-1} & A_{N-1}^T P_N B_{N-1} \\ 0 & B_{N-1}^T P_N A_{N-1} & R_{N-1} + B_{N-1}^T P_N B_{N-1}\end{pmatrix}\begin{pmatrix}\Delta\lambda_{N-1} \\ \Delta s_{N-1} \\ \Delta q_{N-1}\end{pmatrix} = \begin{pmatrix}\nabla_{s_{N-1}} \\ \nabla_{q_{N-1}}\end{pmatrix} + \begin{pmatrix}A_{N-1}^T P_N & A_{N-1}^T \\ B_{N-1}^T P_N & B_{N-1}^T\end{pmatrix}\left(\begin{matrix}\nabla \\ \nabla\end{matrix}\right.$$

Zuerst wird Δq_{N-1} gelöst

$$\Delta q_{N-1} = (R_{N-1} + B_{N-1}^T P_N B_{N-1})^{-1}(\nabla_{q_{N-1}} + B_{N-1}^T P_N \nabla_{\lambda_N} + B_{N-1}^T \nabla_{s_N} - B_{N-1}^T P_N A_{N-1}\Delta s_{N-1})$$

Für $\Delta\lambda_{N-1}$ und Δs_{N-1} ergibt sich dann

$$-\Delta\lambda_{N-1} + P_{N-1}\Delta s_{N-1} = \nabla_{s_{N-1}}^*$$

mit

$$\begin{aligned}
P_{N-1} = \quad & Q_{N-1} + A_{N-1}^T P_N A_{N-1} - A_{N-1}^T P_N B_{N-1}(R_{N-1} + B_{N-1}^T P_N B_{N-1})^{-1} B_{N-1}^T P_N A_{N-1} \\
\nabla_{s_{N-1}}^* = \quad & \nabla_{s_{N-1}} + A_{N-1}^T P_N \nabla_{\lambda_N} + A_{N-1}^T \nabla_{s_N} \\
& -(R_{N-1} + B_{N-1}^T P_N B_{N-1})^{-1}(\nabla_{q_{N-1}} + B_{N-1}^T P_N \nabla_{\lambda_N} + B_{N-1}^T \nabla_{s_N})
\end{aligned}$$

Damit ergibt sich für das anfängliche System $J^k(y^k)\Delta y^k = -\nabla_{y^k} L^k(y^k)$

$$\begin{pmatrix}
 & -E & & & & & & & \\
-E & Q_k & 0 & A_k^T & & & & & \\
 & 0 & R_k & B_k^T & & & & & \\
 & A_k & B_k & & \ddots & & & & \\
 & & & \ddots & Q_{N-2} & 0 & A_{N-2}^T & & \\
 & & & & 0 & R_{N-2} & B_{N-2}^T & & \\
 & & & & A_{N-2} & B_{N-2} & & -E & \\
 & & & & & & -E & P_{N-1} &
\end{pmatrix}\begin{pmatrix}\Delta\lambda_k \\ \Delta s_k \\ \Delta q_k \\ \vdots \\ \Delta\lambda_{N-1} \\ \Delta s_{N-1}\end{pmatrix} = \begin{pmatrix}\nabla_{\lambda_k} \\ \nabla_{s_k} \\ \nabla_{q_k} \\ \vdots \\ \nabla_{\lambda_{N-1}} \\ \nabla_{s_{N-1}}^*\end{pmatrix}$$

Die weiteren P_i ergeben sich für $i = k+1, ..., N-1$

$$\begin{aligned}
P_{i-1} = \quad & Q_{i-1} + A_{i-1}^T P_i A_{i-1} - A_{i-1}^T P_i B_{i-1}(R_{i-1} + B_{i-1}^T P_i B_{i-1})^{-1} B_{i-1}^T P_i A_{i-1} \\
\nabla_{s_{i-1}}^* = \quad & \nabla_{s_{i-1}} + A_{i-1}^T P_i \nabla_{\lambda_i} + A_{i-1}^T \nabla_{s_i}^* \\
& -(R_{i-1} + B_{i-1}^T P_i B_{i-1})^{-1}(\nabla_{q_{i-1}} + B_{i-1}^T P_i \nabla_{\lambda_i} + B_{i-1}^T \nabla_{s_i}^*)
\end{aligned}$$

Schließlich ergibt sich

$$\begin{pmatrix}\Delta\lambda_k \\ \Delta s_k\end{pmatrix} = \begin{pmatrix}-P_k & -E \\ -E & 0\end{pmatrix}\begin{pmatrix}\nabla_{\lambda_k} \\ \nabla_{s_k}^*\end{pmatrix}$$

3. x_k bekannt. Berechne $\nabla_{\lambda_k} = x_k - s_k$

$$\Delta q_k = (R_k + B_k^T P_{k+1} B_k)^{-1}(\nabla_{q_k} + B_k^T P_{k+1} \nabla_{\lambda_{k+1}} + B_k^T \nabla_{s_{k+1}}^* - B_k^T P_{k+1} A_k \Delta s_k)$$

Gib dem System den Wert $u_k = q_k + \Delta q_k$.
Berechne mit Forward Recursion die restlichen Werte von Δy^k

$$\begin{pmatrix} \Delta\lambda_{i+1} \\ \Delta s_{i+1} \end{pmatrix} = \begin{pmatrix} -P_{i+1} & -E \\ -E & 0 \end{pmatrix} \left[\begin{pmatrix} \nabla_{\lambda_{i+1}} \\ \nabla^*_{s_{i+1}} \end{pmatrix} - \begin{pmatrix} 0 & A_i & B_i \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \Delta\lambda_i \\ \Delta s_i \\ \Delta q_i \end{pmatrix} \right]$$

Berechne $y^{k+1} = \prod^{k+1}(y^k + \Delta y^k)$

Dieses Verfahren empfiehlt sich nicht wenn N zu groß ist.
Eine Alternative wäre, für große N, ein fixes n zu wählen und das Verfahren darauf anzuwenden ohne kleiner werdenden Horizont.

Für bestimmte Funktionen vereinfacht sich das Problem. Wenn man die Funktion $l_i(q_i) = q_i$ betrachtet, was einer Minimierung über die Steuerungssignale entspricht, vereinfacht sich R_i

$$R_i = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & & \\ \vdots & & \ddots & \vdots \\ 1 & & \cdots & 1 \end{pmatrix} \qquad \forall i = k, ..., N-1 \quad \forall k = 0, ...N-1$$

und muss nicht mehr für jeden Zeitschritt neu berechnet werden.