Problemstellung:

$$\min_{u(\cdot),x(\cdot)} f(x,u) \qquad s.t. \begin{cases} g(x,u) \leq 0 \\ h(x,u) = 0 \end{cases}$$

mit Zielfunktion f, Ungleichheitsbedingung g und Gleichheitsbedingung h

$$\begin{aligned} f &: R^{n_x} \times R^{n_u} \to R \\ g &: R^{n_x} \times R^{n_u} \to R \\ h &: R^{n_x} \times R^{n_u} \to R \end{aligned}$$

Zur einfacheren Rechnung fassen wir x und u zu einer gemeinsamen Variable zusammen:

$$v := (x,u)^T \qquad v \in R^{n_x+n_u}$$

Lagrangefunktion:

$$L(v,\lambda,\mu) := f(v) - \lambda^T g(v) - \mu^T h(v)$$

Das SQP-Verfahren beginnt bei einem Startwert $y_0 = (v_0, \lambda_0, \mu_0)$ und jede weiter Iterirte ergibt sich

$$y_{k+1} = y_k + \alpha_k \Delta y_k$$

mit $\alpha_k \in (0,1]$ und

$$\Delta y_k = \begin{pmatrix} \Delta v_k \\ \Delta\lambda_k \\ \Delta\mu_k \end{pmatrix} := \begin{pmatrix} \Delta v_k \\ \tilde{\lambda}_k - \lambda_k \\ \tilde{\mu}_k - \mu_k \end{pmatrix}$$

wird erhalten durch den Lösungspunkt $(\Delta_k, \tilde{\lambda}_k, \tilde{\mu}_k)$ des folgenden quadratischen Problems

$$\begin{aligned} &\min_{\Delta v \in \Omega_k} \frac{1}{2} \Delta v^T A_k \Delta v + \nabla_v f(v_k)^T \Delta v \\ &s.t. \begin{cases} g(v_k) + \nabla_v g(v_k)^T \Delta v \leq 0 \\ h(v_k) + \nabla_v h(v_k)^T \Delta v = 0 \end{cases} \end{aligned}$$

Die existierenden SQP-Verfahren unterscheiden sich hauptsächlich in der Wahl der Schrittlänge α_k, der Hesse-Matrix A_k und des Bereichs $\Omega_k \subset R^{n_v}$. Die entstehenden Iterierten y_k formen eine Reihe, von der erwartet wird, dass sie zu einem KKT-Punkt $y^* = (v^*, \lambda^*, \mu^*)$ konvergiert. In der Praxis wird die Iteration abgebrochen, wenn ein entsprechendes Konvergenzkriterium erreicht ist.

Full Step Exact Hessian SQP Method

In diesem Verfahren verwendet man $\alpha_k := 1$,$\Omega_k := R^{n_v}$ und am wichtigsten

$$A_k := \nabla_v^2 L(v_k, \lambda_k, \mu_k).$$

Veranschaulicht wird diese Wahl im folgenden Beispiel eines Gleichheitsproblems. In diesem Fall sind die nötigen Optimalitätsbedingungen für die QP Lösung $(\Delta v_k, \tilde{\mu}_k)$

$$\begin{aligned} \nabla_v^2 L(v_k,\mu_k)\Delta v_k + \nabla_v f(v_k) - \nabla_v h(v_k)\tilde{\mu}_k &= 0, \\ h(v_k) + \nabla_v h(v_k)^T \Delta v_k &= 0. \end{aligned}$$

Durch substituiere $\tilde{\mu}_k = \mu_k - \Delta\mu_k$ können wir equivalent schreiben

$$\begin{aligned} \nabla_v L(v_k, \mu_k) + \nabla_v^2 L(v_k, \mu_k)\Delta v_k - \nabla_v h(v_k)\Delta\mu_k = 0, \\ h(v_k) + \nabla_v h(v_k)^T \Delta v_k = 0. \end{aligned}$$

was der Newton-Raphson-Iterationsregel entspricht

$$\begin{pmatrix} \nabla_v L(v_k, \mu_k) \\ h(v_k) \end{pmatrix} + \frac{\partial}{\partial(v,\mu)} \begin{pmatrix} \nabla_v L(v_k, \mu_k) \\ h(v_k) \end{pmatrix} \begin{pmatrix} \Delta v_k \\ \Delta\mu_k \end{pmatrix} = 0,$$

für die Lösung des KKT Systems

$$\begin{pmatrix} \nabla_v L(v, \mu) \\ h(v) \end{pmatrix} = \begin{pmatrix} \nabla_v f(v) - \nabla_v h(v)\mu \\ h(v) \end{pmatrix} = 0.$$

Daraus ergibt sich dann für $(\Delta v_k, \Delta\mu_k)$

$$\begin{pmatrix} \Delta v_k \\ \Delta\mu_k \end{pmatrix} = -\left(\frac{\partial}{\partial(v,\mu)} \begin{pmatrix} \nabla_v L(v_k, \mu_k) \\ h(v_k) \end{pmatrix}\right)^{-1} \begin{pmatrix} \nabla_v L(v_k, \mu_k) \\ h(v_k) \end{pmatrix}$$

wenn die Ableitung in diesem Punkt regulär ist. Die Inverse sollte nicht berechnet werden, da dies zu aufwändig wird.

Vorbereitung

Diskretisieren wir nun unser Optimierungsproblem. Für jeden Zeitschritt k erhalten wir ein Problem $P^k(x_k)$

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

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 $y := (\lambda_k, s_k, q_k, \lambda_{k+1}, s_{k+1}, q_{k+1}, ..., \lambda_N, s_N)$ verwendet. Aus dieser Wahl für y ergibt sich eine praktische Diagonalform der Hesse-Matrix. Mit dieser Lagrangegleichung ergibt sich die 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 Systems

$$\nabla_y L^k(y_i) + \nabla_y^2 L^k(y_i)\Delta y_i = 0$$

ist. Für den im Diehlpaper vorgestellten Algorithmus verwendet man das oben vorgestellte Newton-Raphson-Verfahren nicht exakt. Die zweite Ableitung $\nabla_y^2 L^k$ besser gesagt die Hesse-Matrix $\nabla_{q,s}^2 L^k$ wird ersetzt durch eine (symmetrische) Approximierung. Die Approximierung von $\nabla_y^2 L^k(y)$ wird im folgenden mit $J^k(y)$ bezeichnet. Ebenso wird das Newton-Type-Verfahren approximiert:

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

J^k wird auch als Karush-Kuhn-Tucker Matrix bezeichnet.

$$\nabla_y^2 L^k(y) = \begin{pmatrix}
 & -E & & & & & & & & \\
-E & Q_k & M_k & A_k^T & & & & & & \\
 & M_k^T & R_k & B_k^T & & & & & & \\
 & A_k & B_k & & -E & & & & & \\
 & & & -E & Q_{k+1} & M_{k+1} & A_{k+1}^T & & & \\
 & & & & M_{k+1}^T & R_{k+1} & B_{k+1}^T & & & \\
 & & & & A_{k+1} & B_{k+1} & & & & \\
 & & & & & & & \ddots & & \\
 & & & & & & \ddots & Q_{N-1} & M_{N-1} & A_{N-1}^T \\
 & & & & & & & M_{N-1}^T & R_{N-1} & B_{N-1}^T \\
 & & & & & & & A_{N-1} & B_{N-1} & \\
 & & & & & & & & & -E \\
 & & & & & & & & -E & Q_N
\end{pmatrix}$$

Mit $A_i := \dfrac{\partial h_i}{\partial s_i}$, $B_i := \dfrac{\partial h_i}{\partial q_i}$, $\begin{pmatrix} Q_i & M_i \\ M_i^T & R_i \end{pmatrix} := \nabla_{s_i,q_i}^2 L^k$ und $Q_N := \nabla_{s_N}^2 L^k$.

In der Approximierung werden Q_i,R_i und M_i ersetzt durch $Q_i^H(s_i, q_i, \lambda_{k+1})$,$R_i^H(s_i, q_i, \lambda_{k+1})$ und $M_i^H(s_i, q_i, \lambda_{k+1})$.
Wir sehen, wenn wir $y = (\lambda_k, s_k, q_k, \tilde{y})$ betrachten, dass $\tilde{y}$ direkt zum nächsten Problem $P_{k+1}(x_{k+1}$ gehört.

$$\nabla_y^2 L^k(y) = \left(\begin{array}{ccc|c}
 & -E & & \\
-E & Q_k & M_k & A_k^T \\
 & M_k^T & R_k & B_k^T \\
\hline
 & A_k & B_k & \\
 & & & \nabla_{\tilde{y}}^2 L^k(\tilde{y}) \\
 & & &
\end{array}\right)$$

Im Paper wird erwähnt, dass diese vorteilhafte Form von $\nabla_y^2 L^k(y)$ bzw. $J^k(y)$ eine effiziente Lösung der Gleichung $J^k(y)x = b$ durch die Riccati Recursion ermöglicht.

Approximieren

Ein wichtiger Spezialfall der Newton-Type Verfahren ist die Constrained Gauss-Newton Methode welches sich auf die LEAST SQUARES Form der Funktion

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

anwenden lässt. In diesem Fall lässt sich die Approximierung wie folgt berechnen.

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

Algorithmus

Wir starten mit einem passend gewählten $y^0 \in R^{n_0}$ für das Problem $P_0(\cdot)$. Der Iterationsindes k wird auf Null gesetzt und wir starten mit den Schritten:

1. Vorbereitung. Basierend auf dem Wert $y^k \in R^{n_k}$ wird $\nabla_{y^k} L^k(y^k)$ und $J^k(y^k)$ berechnet. Beachte, dass $J^k(y^k)$ nicht von x_k abhängt und bei $\nabla_y L^k(y^k)$ nur die erste Komponente ($\nabla_{\lambda_k} L^k = x_k - s_k$). Diese wird nur im 2. Schritt benötig, daher berechne $(J^k(y^k))^{-1}\nabla_{y_k} L^k(y^k)$ soweit es geht ohne den Wert x_k.

2. Feedback. Zu dem Zeitpunkt an dem x_k bekannt ist beende die Berechnung von $\Delta y^k = (J^k(y^k))^{-1}\nabla_{y_k} L^k(y^k)$ und gib dem System den Steuerungsimpuls $u_k := q_k + \Delta q_k$.

3. Anpassen. Wenn k = N-1 STOP, ansonsten berechne den nächsten Startwert y^{k+1}. Verwende die Projektion ($\prod^{k+1} y = \tilde{y}$) und wende sie auf den Startwert zu dem der Iterationsschritt addiert wurde an. In etwa so:

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

Setzte $k = k + 1$ und gehe zu Schritt 1