České vysoké učení technické v Praze
Fakulta elektrotechnická
Katedra řídicí techniky

DIPLOMOVÁ PRÁCE

Algoritmy pro nelineární prediktivní řízení

Praha, 2006 **Miroslav Čermák**

Prohlášení

Prohlašuji, že jsem svou diplomovou práci vypracoval samostatně a použil jsem pouze podklady uvedené v přiloženém seznamu.

Nemám závažný důvod proti užití tohoto školního díla ve smyslu § 60 Zákona č.121/2000 Sb. , o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

V Praze dne 27. ledna 2006

_____

podpis

Poděkování

Děkuji především vedoucímu diplomové práce Prof. Ing. Vladimíru Havlenovi, CSc. a dále Ing. Jaroslavu Pekařovi za pomoc a cenné rady při tvorbě této práce.

Abstrakt

Tato diplomová práce se zabývá formulací algoritmu pro návrh nelineárního prediktivního řízení. V práci je popsána metoda nelineární optimalizace využívající sekvenční kvadratické programování, které je následně využito k nalezení optimálního řízení v multiple shooting prediktivním regulátoru. Navržené algoritmy jsou otestovány na modelu odparky.

Abstract

This diploma thesis deals with the model predictive control design and especially with the numerical algorithms for the nonlinear model predictive control (NMPC). In the work, the nonlinear optimization methods based on the sequential quadratic programming suitable for the nonlinear predictive control are reviewed and the basic methods are compared on the simple examples. The nonlinear optimization is then used for the multiple shooting NMPC formulation. The properties of the designed NMPC algorithm are shown on the simple evaporator model.

Obsah

Seznam obrázků

Seznam tabulek

Kapitola 1

Úvod

Prediktivní řízení (Model Predictive Control - MPC), také nazýváno jako řízení s klouzavým horizontem, se dnes začíná rozšiřovat v průmyslu jako efektivní prostředek vícerozměrového řízení s omezeními. Při návrhu lze zahrnout omezení na vstupní, výstupní i stavové proměnné. MPC je počítačové řízení, které hledá optimální vstupní trajektorii minimalizující rozdíl mezi predikovaným a požadovaným chováním soustavy při splnění zadaných omezení. Prediktivní regulátor používá k získání odezvy na vstupní trajektorii predikční model soustavy. Ze získané optimální trajektorie se k řízení soustavy použije pouze první krok a výpočet se opakuje v následující vzorkovací periodě. Tento přístup se nazývá klouzavý horizont a umožňuje potlačovat příchozí poruchy tím, že zavádí do regulační smyčky zpětnou vazbu.

Podle použitého predikčního modelu se MPC rozdělují na lineární [3, 7] a nelineární [1]. Optimalizační problém lineárního MPC je většinou snáze řešitelný a je vhodný pro soustavy, které nevykazují mezi vstupem a výstupem silnou nelinearitu. Naopak řešení nelineárního optimalizačního problému pro nelineární soustavu je často časově náročné a proto se nelineární MPC obvykle používá pro řízení pomalých procesů v chemickém a petrochemickém průmyslu.

Cílem této práce je navrhnout algoritmus prediktivního regulátoru. Hlavní úkol MPC regulátoru je minimalizace ztrátové funkce, která může být obecně nelineární. Proto se budeme nejprve zabývat metodami pro hledání minima nelineární funkce. Ve druhé kapitole uvedeme metodu jednorozměrového hledání minima vícerozměrné funkce využívající tzv. Wolfeho podmínky a ve třetí kapitole se zaměříme na metodu sekvenčního kvadratického programování. Dále zde ukážeme podobnost sekvenčního kvadratického programování a Newton-Lagrangeovy metody při hledání extrému ztrátové funkce. Zaměříme se též na možné aproximace hesiánu v kvadratickém modelu ztrátové funkce. Ve čtvrté kapitole odvodíme tzv. multiple shooting MPC

regulátor a uvedeme jeho algoritmus využívající metodu sekvenčního kvadratického programování. V páté kapitole provedeme ověření vlastností navržených algoritmů na modelu odparky.

1.1 Značení

V této části definujeme použité značení vektorů, které bude v této práci dále použito. Derivace skalární funkce $f(x) : \mathbb{R}^n \to \mathbb{R}$ podle vektorového argumentu $x \in \mathbb{R}^n$ je řádkový vektor, který značíme $\nabla f(x)$. Platí tedy

$$\frac{\partial f(x)}{\partial x} = \begin{bmatrix} \frac{\partial f(x)}{\partial x_1} & \frac{\partial f(x)}{\partial x_2} & \cdots & \frac{\partial f(x)}{\partial x_n} \end{bmatrix} = \nabla f(x). \tag{1.1}$$

Gradient skalární funkce f v bodě x je sloupcový vektor, platí $\mathrm{grad} f(x) = \nabla^T f(x)$. Druhá derivace skalární funkce $f(x)$ podle vektorového argumentu x je Hesseho matice, kterou značíme $H(x) = \nabla^2 f(x)$.

$$H(x) = \nabla^2 f(x) = \frac{\partial^2 f(x)}{\partial x^2} = \begin{bmatrix} \frac{\partial^2 f(x)}{\partial x_1^2} & \frac{\partial^2 f(x)}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_1 \partial x_n} \\ \vdots & \vdots & & \vdots \\ \frac{\partial^2 f(x)}{\partial x_n \partial x_1} & \frac{\partial^2 f(x)}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_n^2} \end{bmatrix}. \tag{1.2}$$

Derivace vektorové funkce $f(x) : \mathbb{R}^n \to \mathbb{R}^m$ podle vektorového argumentu $x \in \mathbb{R}^n$ je Jacobiho matice, kterou značíme $J(x) = \nabla f(x)$

$$J(x) = \nabla f(x) = \frac{\partial f(x)}{\partial x} = \begin{bmatrix} \frac{\partial f_1(x)}{\partial x_1} & \frac{\partial f_1(x)}{\partial x_2} & \cdots & \frac{\partial f(x)}{\partial x_n} \\ \vdots & \vdots & & \vdots \\ \frac{\partial f_m(x)}{\partial x_1} & \frac{\partial f_m(x)}{\partial x_2} & \cdots & \frac{\partial f(x)_m}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \nabla f_1(x) \\ \vdots \\ \nabla f_m(x) \end{bmatrix}. \tag{1.3}$$

Kapitola 2

Metoda jednorozměrového hledání

Metoda jednorozměrového hledání (Line Search - LS) je jedna z metod hledání extrému vícerozměrné funkce. V našem případě hledáme minimum ztrátové funkce f. Hlavním znakem této metody je hledání extrému funkce podél polopřímky ve směru jejího poklesu. V každém kroku této metody spočítáme směr hledání p_k a poté rozhodneme, jak daleko se v tomto směru můžeme posunout. Krok je dán vztahem

$$x_{k+1} = x_k + \alpha_k p_k, \tag{2.1}$$

kde kladný skalár α_k nazveme délkou kroku. Úspěch line search metody závisí na efektivní volbě jak směru p_k, tak i délky kroku α_k.

Mnoho jednorozměrových hledacích algoritmů požaduje, aby p_k byl směr klesání ztrátové funkce, protože tím bude zaručeno, že v tomto směru bude funkce f snížena. Jak spočítat směr hledání je popsáno v další kapitole o Sekvenčním kvadratickém programování, proto se nejprve zaměříme na volbu parametru délky kroku α_k.

Při výpočtu délky kroku α_k jsme postaveni před kompromis — na jedné straně chceme, aby zvolené α_k zajistilo značné snížení hodnoty ztrátové funkce f, ale na druhé straně nechceme strávit mnoho času jeho hledáním. Ideální volba kroku α_k je taková, která dosáhne globálního minima jednorozměrné funkce $\phi(.)$ definované jako

$$\phi(\alpha) = f(x_k + \alpha p_k), \quad \alpha > 0. \tag{2.2}$$

Obecně je však příliš náročné nalézt toto minimum přesně, proto se v mnoha praktických strategiích používá přibližný line search k nalezení takové délky kroku, která dosáhne dostatečného poklesu ztrátové funkce za minimální cenu.

Typický line search algoritmus testuje posloupnost možných hodnot α a skončí tehdy, pokud nalezne takovou hodnotu, která vyhovuje určitým podmínkám. Algoritmus lze rozdělit do dvou fází, v první fázi najdeme interval obsahující vhodné

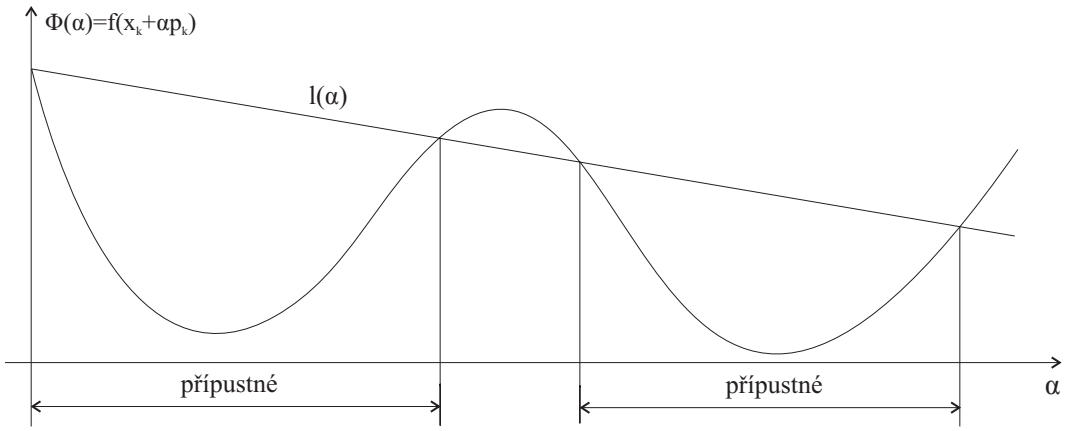
délky kroku a ve druhé fázi na tomto intervalu pomocí interpolace nebo bisekce [5] vypočteme příznivou délku kroku. K ukončení line search algoritmu lze použít několik různých podmínek [8]. Pro naši úlohu je vhodné použít tzv. Wolfeho podmínky, které nám zaručí, že daná délka kroku povede k výraznému zmenšení ztrátové funkce.

2.1 Wolfeho podmínky

Tyto podmínky můžeme zapsat pomocí následujících dvou nerovnic

$$\begin{aligned} f(x_k + \alpha_k p_k) &\leqslant f(x_k) + c_1 \alpha_k \nabla f_k^T p_k \\ \nabla f(x_k + \alpha_k p_k)^T p_k &\geqslant c_2 \nabla f_k^T p_k, \end{aligned} \tag{2.3}$$

kde konstanta $c_1 \in (0, 1)$ a konstanta $c_2 \in (c_1, 1)$. První nerovnice, kterou můžeme nazvat jako podmínku dostatečného poklesu, požaduje, aby pro danou délku kroku byl zajištěn dostatečný pokles ztrátové funkce, tedy aby zvolený krok nebyl příliš dlouhý. Tato podmínka je znázorněna na obr. 2.1. Jak je vidět na obr. 2.1, první



Obrázek 2.1: První Wolfeho podmínka

podmínka je splněna pro všechny dostatečně malé hodnoty α, proto je zde ještě druhá podmínka, podmínka křivosti, která je vyjářena druhou nerovnicí (2.3). Ta požaduje, aby strmost klesání ztrátové funkce v následujícím kroku byla menší než v předchozím kroku ($\alpha = 0$). Druhá podmínka je znázorněna na obr. 2.2. Obě Wolfeho podmínky, tedy podmínka dostatečného poklesu a podmínka zakřivení, jsou zobrazeny na obrázku obr. 2.3, na kterém je také vidět, že přípustné délky kroku nemusí vždy ležet blízko minima funkce ϕ. Proto upravíme podmínku zakřivení tak, aby α_k ležel v blízkém okolí minima funkce ϕ a získáme tzv. silné Wolfeho podmínky

$$f(x_k + \alpha_k p_k) \leqslant f(x_k) + c_1 \alpha_k \nabla f_k^T p_k \tag{2.4}$$

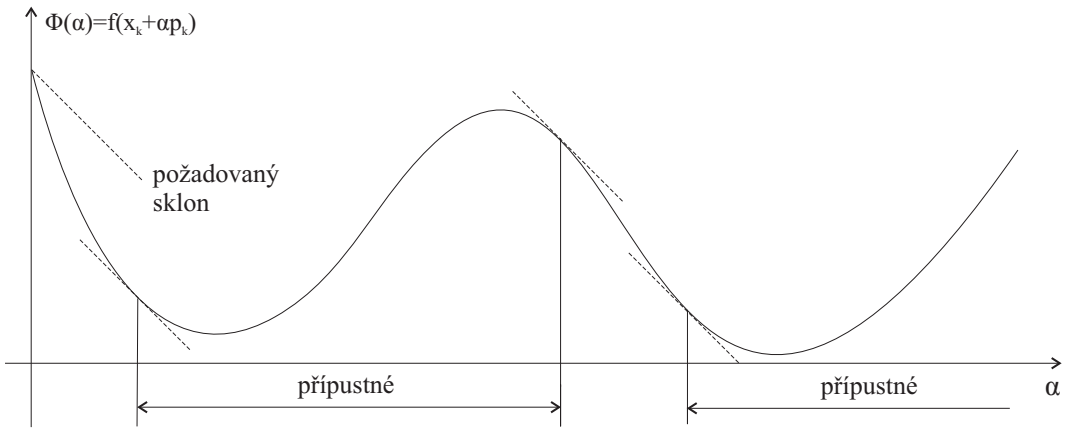
$$\nabla f(x_k + \alpha_k p_k)^T p_k \quad \leqslant \quad c_2 \left| \nabla f_k^T p_k \right|,$$

s konstantami $0 < c_1 < c_2 < 1$. Jediný rozdíl oproti původním Wolfeho podmínkám je, že již nepřipouštíme, aby derivace $\phi'(\alpha_k)$ byla příliš pozitivní. Tedy vyloučíme body, které jsou příliš vzdálené od stacionárních bodů funkce ϕ.

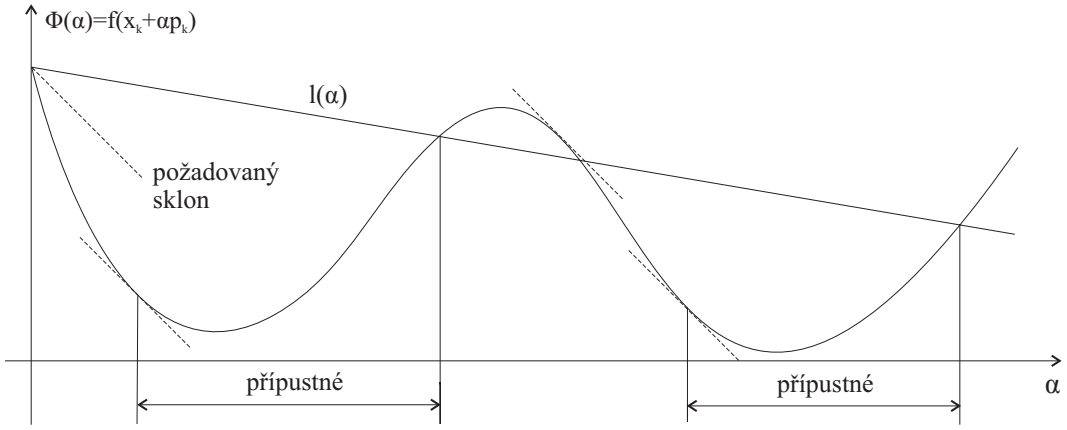
Obdobný vztah jako je první Wolfeho podmínka (2.3) využívá i tzv. Armiův test [8], při němž je hledáno tak velké α, které ještě splňuje podmínku dostatečného poklesu ztrátové funkce.

V literatuře [5] je doporučeno volit hodnotu konstanty $c_1 = 10^{-4}$ a hodnotu $c_2 = 0,9$ pokud jsme směr p_k získali pomocí Newtonovy nebo kvazi-Newtonovy metody. Pokud jsme směr p_k získali pomocí metody nejrychlejšího sestupu, je vhodné volit $c_2 = 0,1$.

Řád konvergence line search algoritmu závisí na metodě, kterou získáme směr hledání p_k. Pokud použijeme metodu nejrychlejšího sestupu, je řád konvergence



Obrázek 2.2: Druhá Wolfeho podmínka



Obrázek 2.3: Wolfeho podmínky

lineární. V případě kvazi-Newtonových metod konverguje superlineárně a při použití Newtonovy metody konverguje kvadraticky.

2.2 Algoritmus jednorozměrového hledání

Nyní popíšeme jednodimenzionální prohledávací proceduru, která nám zajistí nalezení délky kroku splňující silné Wolfeho podmínky (2.4). Algoritmus sestává ze dvou částí:

1. řídící část algoritmu
2. funkce ZOOM

V řídící části se nejprve pro zvolenou délku kroku α_i vypočte příslušná hodnota funkce $\phi(\alpha_i)$ a otestuje se, zda je splněna první Wolfeho podmínka dostatečného poklesu. Pokud není, zavolá se funkce Zoom, která vrátí optimální délku kroku α^* a algoritmus může skončit. Když je první Wolfeho podmínka splněna, spočte se hodnota směrnice tečny v daném bodě α_i. Pokud je v dané toleranci, tedy je splněna i druhá Wolfeho podmínka zakřivení, algoritmus skončí s aktuální hodnotou délky kroku. Když je směrnice tečny v daném bodě kladná, funkce ϕ roste, zavolá se funkce Zoom, která opět vrátí α^* a algoritmus se ukončí. Jinak algoritmus najde novou hodnotu $\alpha_{i+1} \in (\alpha_i, \alpha_{max})$, na které opět otestuje Wolfeho podmínky.

Pro získání nové hodnoty α_{i+1} (předposlední krok algoritmu) použijeme extrapolaci. Nejjednodušší je volba nového kroku jako násobek předchozí hodnoty α_i. Důležité je, aby se následný krok dostatečně rychle zvětšoval a dosáhl horní meze α_{max} v konečném počtu kroků.

Dále se zaměříme na funkci Zoom. Ta je vždy volána s dvěma parametry α_{lo} a α_{hi} vymezujícími interval, který zaručeně obsahuje délky kroků splňující Wolfeho podmínky. Pořadí parametrů je vždy voleno tak, aby určovaly správný směr pro interpolaci nové hodnoty α_j. Algoritmus pro novou hodnotu α_j ověří, zda splňuje podmínku dostatečného poklesu. Pokud ji nesplňuje, zmenší prohledávaný interval nahrazením α_{hi} hodnotou α_j a opakuje interpolaci na zmenšeném intervalu. Při splnění první podmínky testuje ještě druhou Wolfeho podmínku. Pokud je i ona splněna, může funkce skončit s návratovou hodnotou α_j. Pokud ji však nesplňuje, funkce upraví meze intervalu tak, aby určovaly správný směr pro interpolaci a zmenší prohledávaný interval nahrazením α_{lo} hodnotou α_j. V tomto intervalu opět nalezne novou hodnotu kroku pomocí interpolace. Pro správnou funkci line search algoritmu je nutné, aby funkce Zoom byla ukončena v konečném počtu kroků.

Algoritmus 1 (Algoritmus jednorozměrového hledání)

Nastav počáteční hodnotu $\alpha_0 \leftarrow 1$, zvol hodnotu $\alpha_1 > 1$ a α_{max};

$i \leftarrow 1$;

repeat

Vypočti $\phi(\alpha_i)$;

if $\phi(\alpha_i) > \phi(0) + c_1\alpha_i\phi'(0)$ nebo $[\phi(\alpha_1) \geqslant \phi(\alpha_{i-1})$ a $i > 1]$ **then**

$\alpha^* \leftarrow$ ZOOM(α_{i-1}, α_i) a KONEC;

end if

Vypočti $\phi'(\alpha_i)$;

if $|\phi'(\alpha_i)| \leqslant -c_2\phi'(0)$ **then**

$\alpha^* \leftarrow \alpha_i$ a KONEC;

end if

if $\phi'(\alpha_i) \geqslant 0$ **then**

$\alpha^* \leftarrow$ ZOOM(α_i, α_{i-1}) a KONEC;

end if

Vyber $\alpha_{i+1} \in (\alpha_i, \alpha_{max})$;

$i \leftarrow i + 1$;

until $i = i_{max}$

Algoritmus 2 (Zoom)

$j \leftarrow 1$;
repeat
  Najdi interpolací (kvadratická, kubická) délku kroku α_j v rozmezí α_{lo} až α_{hi};
  Vypočti $\phi(\alpha_j)$;
  if $\phi(\alpha_j) > \phi(0) + c_1\alpha_j\phi'(0)$ nebo $\phi(\alpha_1) \geqslant \phi(\alpha_{lo})$ **then**
    $\alpha_{hi} \leftarrow \alpha_j$;
  else
    Vypočti $\phi'(\alpha_j)$;
    if $|\phi'(\alpha_j)| \leqslant -c_2\phi'(0)$ **then**
      $\alpha^* \leftarrow \alpha_j$ a KONEC;
    end if
    if $\phi'(\alpha_j)(\alpha_{hi} - \alpha_{lo}) \geqslant 0$ **then**
      $\alpha_{hi} \leftarrow \alpha_{lo}$;
    end if
    $\alpha_{lo} \leftarrow \alpha_j$;
    $j \leftarrow j + 1$;
  end if
until $j < j_{max}$

Kapitola 3

Sekvenční kvadratické programování

Sekvenční kvadratické programování (Sequential Quadratic Programming - SQP) je jedna z nejefektivnějších metod řešení nelineárních optimalizačních úloh [2]. SQP neřeší nelineární úlohu přímo, ale převádí ji na sekvenci optimalizačních podproblémů. Nejprve se zaměříme na řešení minimalizační úlohy s omezením typu rovnosti a později řešení rozšíříme i na minimalizační úlohy s omezením typu nerovnosti.

Uvažujme následující problém:

$$\begin{aligned} &\min_x f(x) \\ \text{s omezením} \quad &h(x) = 0, \end{aligned} \tag{3.1}$$

kde $f : \mathbb{R}^n \to \mathbb{R}$ je ztrátová funkce a $h : \mathbb{R}^n \to \mathbb{R}^m$ je funkce vyjadřující omezení. Obě funkce jsou obecně nelineární. Protože uvedený nelineární problém neumíme vyřešit přímo, musíme nelineární funkce aproximovat. SQP používá pro ztrátovou funkci aproximaci kvadratickou formou a pro funkci omezení lineární formou. Nelineární program, ve kterém je ztrátová funkce kvadratická a funkce omezení je lineární, se nazývá kvadraticé programování (QP). Základní myšlenka SQP metody je pro každou iteraci x_k vytvořit podúlohu kvadratického programování, jejímž vyřešením získáme novou iteraci x_{k+1}. Úkolem je navrhnout danou podúlohu kvadratického programování tak, aby výsledkem bylo přiblížení se k řešení původního problému a aby celkový SQP algoritmus konvergoval. Nejjednodušší přiblížení metody SQP je ukázka aplikace Newtonovy metody na Karush-Kuhn-Tuckerovy (KKT) nutné podmínky optimality.

Bod x^* nazveme bodem relativního minima problému (3.1), pokud splňuje násle-

dující podmínky (KKT):

$$\nabla f(x^*) + \sum_{i=1}^{m} \nabla h_i(x^*)\lambda_i = 0 \tag{3.2}$$
$$h(x^*) = 0,$$

kde λ je vektor Lagrangeových koeficientů.

3.1 Newton-Lagrangeova metoda

Problém (3.1) rozšíříme přidáním omezení do ztrátové funkce a dostaneme Lagrangeovu funkci[1] ve tvaru

$$L(x, \lambda) = f(x) + \sum_{i=1}^{m} h_i(x)\lambda_i. \tag{3.3}$$

Aplikací Karush-Kuhn-Tuckerových nutných podmínek prvního řádu (3.2) na problém (3.3) dostaneme následující rovnice:

$$\begin{aligned} \nabla_x L(x, \lambda) &= \nabla_x f(x) + \sum_{i=1}^{m} \nabla h_i(x^*)\lambda_i = 0 \\ \nabla_\lambda L(x, \lambda) &= h(x) = 0 \end{aligned} \tag{3.4}$$

Označíme $A(x)$ Jacobiho matici omezení, která se vypočte jako

$$A(x)^T = [\nabla h_1(x), \nabla h_2(x), ..., \nabla h_m(x)], \tag{3.5}$$

kde $h_i(x)$ je i-tá složka vektoru $h(x)$, a získáme soustavu $n + m$ rovnic o $n + m$ neznámých x a λ:

$$F(x, \lambda) = \begin{bmatrix} \nabla f(x) + A(x)^T\lambda \\ h(x) \end{bmatrix} = 0. \tag{3.6}$$

Pokud má matice $A(x^*)$ plnou hodnost, jakékoliv řešení (x^*, λ^*) problému (3.1) s omezením typu rovnosti splňuje také (3.6). Jedna z možných metod, jak řešit nelineární rovnice (3.6) je použití Newtonovy metody, která řeší problém pomocí rozvoje prvního řádu.
Určíme Jakobián matice (3.6) jako

$$K(x, \lambda) = \begin{bmatrix} W(x, \lambda) & A(x)^T \\ A(x) & 0 \end{bmatrix}, \tag{3.7}$$

[1] Lagrangeova funkce nabývá minima ve stejném bodě jako původní ztrátová funkce.

kde W označuje hesián z Lagrangeovy funkce,

$$W(x, \lambda) = \nabla^2_{xx} L(x, \lambda). \tag{3.8}$$

Krok Newtonovy metody od iterace (x_k, λ_k) je dán

$$\begin{bmatrix} x_{k+1} \\ \lambda_{k+1} \end{bmatrix} = \begin{bmatrix} x_k \\ \lambda_k \end{bmatrix} + \begin{bmatrix} p_k \\ p_\lambda \end{bmatrix}, \tag{3.9}$$

kde p_k a p_λ je řešení rovnice

$$K \begin{bmatrix} p_k \\ p_\lambda \end{bmatrix} + F = 0, \tag{3.10}$$

ze které dosazením získáme soustavu KKT podmínek

$$\begin{bmatrix} W_k & A_k^T \\ A_k & 0 \end{bmatrix} \begin{bmatrix} p_k \\ p_\lambda \end{bmatrix} = \begin{bmatrix} -\nabla f_k - A_k^T \lambda_k \\ -h_k \end{bmatrix}. \tag{3.11}$$

Tato iterace se také nazývá Newton-Lagrangeova metoda a je dobře určena, pokud je matice (3.8) pozitivně definitní. Pozitivní definitnost je důsledek následujících podmínek.

Podmínky 1:

1. Jakobián omezení A_k má plnou řádkovou hodnost
2. matice W_k je pozitivně definitní na tečné rovině k omezením[2] v bodě x_k

První podmínka je požadavek lineárně nezávislých omezení a druhá podmínka je splněna, když (x, λ) je blízko optima (x^*, λ^*). Je možné dokázat, že Newton-Lagrangeova iterace za těchto podmínek konverguje kvadraticky [5]. V další části uvedeme řešení minimalizační úlohy využívající SQP metodu.

3.2 SQP metoda

Je zde ovšem i další možnost, jak řešit uvedený problém (3.1). Pokud Lagrangeovu funkci z (3.3) aproximujeme Taylorovým rozvojem druhého řádu a omezující funkci aproximujeme lineárním rozvojem, získáme následující kvadratický program pro iteraci (x_k, λ_k)

$$\begin{aligned} \min_p \frac{1}{2} p_k^T W_k p_k + \nabla f_k^T p_k \\ \text{s omezením} \quad A_k p_k + h_k = 0, \end{aligned} \tag{3.12}$$

[2] platí, že $d^T W_k d > 0 : \forall d \neq 0$ a $A_k d = 0$

kde $p \in \mathbb{R}^n$ je přírůstek řešení ve směru klesání funkce $f(x)$. Pro tento program vytvoříme novou Lagrangeovu funkci

$$L(p_k, \mu_k) = \frac{1}{2} p_k^T W_k p_k + \nabla f_k^T p_k + \mu_k (A_k p_k + h_k), \tag{3.13}$$

kde μ je Lagrangeův koeficient.

Pokud je matice $A(x)$ regulární, tedy jsou splněné výše uvedené Podmínky 1, má tento problém jediné řešení (p_k^*, μ_k^*), které vyhovuje nutným podmínkám prvního řádu

$$\begin{aligned} W_k p_k + \nabla f_k^T + A_k^T \mu_k &= 0 \\ A_k p_k + h_k &= 0. \end{aligned} \tag{3.14}$$

Obdobnou soustavu rovnic získáme, pokud přičteme $A_k^T \lambda_k$ k oběma stranám první rovnice v (3.11) a získáme tak

$$\begin{bmatrix} W_k & A_k^T \\ A_k & 0 \end{bmatrix} \begin{bmatrix} p_k \\ \lambda_{k+1} \end{bmatrix} = \begin{bmatrix} -\nabla f_k \\ -h_k \end{bmatrix}. \tag{3.15}$$

Z porovnání (3.14) a (3.15) plyne, že $p_k = p_k$ a $\lambda_{k+1} = \mu_k$, tedy je vidět, že řešení problému (3.1) metodou SQP a Newtonovou metodou, aplikovanou na podmínky prvního řádu, jsou ekvivalentní pouze s tím rozdílem, že řešením Newtonova algoritmu získáme přírůstek p_λ, naproti tomu řešením SQP získáme přímo λ_{k+1}. Tedy jinými slovy, pokud jsou v bodě x_k splněny Podmínky 1, tak nová iterace (x_{k+1}, λ_{k+1}) může být definována buď jako řešení kvadratického programu (3.12), nebo jako iterace získaná Newtonovou metodou (3.9,3.11) použitou na nutné podmínky optimality daného problému. Tento dvojí pohled je užitečný. Newtonův pohled ulehčuje analýzu, zatímco SQP nám umožňuje odvodit praktický algoritmus a rozšířit danou techniku i na omezení typu nerovnosti.

Nyní můžeme uvést algoritmus Lokáloního sekvenčního programování v té nejjednodušší formě. Pro tento algoritmus je dokázáno, že pokud je počáteční bod (x_0, λ_0) dostatečně blízko optima (x^*, λ^*) a jsou splněny Podmínky 1, tak jím generované iterace konvergují k optimu kvadraticky, stejně jako Newton-Lagrangeova metoda uvedena v kapitole 3.1. Dosud jsme se zabývali pouze řešením minimalizační úlohy s omezením typu rovnosti a nyní jej rozšíříme i na minimalizační úlohy s omezením typu nerovnosti.

Algoritmus 3 (Lokální SQP algoritmus)

Zvol počáteční pár (x_0, λ_0);
for $k = 0, 1, 2, \ldots$ **do**
  Vypočti hodnotu $f_k, \nabla f_k, W_k = W(x_k, \lambda_k), h_k$ a A_k;
  Vyřeš (3.12) a získej p_k a μ_k;
  $x_{k+1} = x_k + p_k; \quad \lambda_{k+1} = \mu_k;$
  if je splněna konvergence **then**
    STOP s přibližným řešením (x_{k+1}, λ_{k+1});
  end if
end for

3.2.1 Omezení typu nerovnosti

Algoritmus řešící SQP program může být jednoduše rozšířen na obecný nelineární problém

$$\begin{aligned} \min_x f(x) & \\ \text{s omezením} \quad h_i(x) &= 0, \quad i \in I, \\ g_j(x) &\leqslant 0, \quad j \in J. \end{aligned} \tag{3.16}$$

Pro problém (3.16) zavedeme Lagrangeovu funkci ve tvaru

$$L(x, \lambda, \mu) = f(x) + \lambda^T h(x) + \mu^T g(x). \tag{3.17}$$

Bod x^* (relativní minimum problému (3.16) splňující zadaná omezení) musí vyhovovat následujícím podmínkám (KKT):

$$\begin{aligned} \nabla f(x^*) + \lambda^T \nabla h(x^*) + \mu^T \nabla g(x^*) &= 0 \\ h(x^*) &= 0 \\ g(x^*) &\leqslant 0 \\ \mu^T g(x^*) &= 0 \\ \mu &\geqslant 0 \end{aligned} \tag{3.18}$$

Abychom mohli pro řešení problému (3.16) použít algoritmus kvadratického programování, musíme opět Lagrangeovu funkci aproximovat Taylorovým rozvojem druhého řádu a obě funkce omezení aproximovat lineárním rozvojem:

$$\begin{aligned} \min_p \frac{1}{2} p^T W_k p + \nabla f_k^T p & \\ \text{s omezením} \quad \nabla h_i(x_k)^T p + h_i(x_k) &= 0, \quad i \in I, \\ \nabla g_j(x_k)^T p + g_j(x_k) &\leqslant 0, \quad j \in J. \end{aligned} \tag{3.19}$$

Pokud zanedbáme ta omezení typu nerovnosti, která se v bodě optimálního řešení x^* neuplatňují a ostatní převedeme na omezení typu rovnosti, pak se můžeme na problém (3.19) dívat, jako kdyby obsahoval pouze omezení typu rovnosti a můžeme k jeho řešení použít dříve uvedený Algoritmus 1.

3.3 Hesián pro kvadratický program

Nyní se zaměříme na volbu matice W_k v kvadratickém modelu (3.12). Pro jednoduchost se nejprve zaměříme na optimalizační problémy s omezením typu rovnosti. První možnost volby matice W_k vychází z ekvivalence mezi SQP a Newtonovou metodou aplikovanou na podmínky optimality (3.6), tedy matici W_k volíme jako hesián z Lagrangiánu. Tato volba vede na kvadratický řád konvergence za odpovídajících podmínek a často, když jsou iterace vzdálené od řešení, vede k rychlému postupu. Avšak takto získaná matice je vytvořena z druhých derivací ztrátové funkce a omezení, které nemusejí být jednoduše spočítatelné a nemusí být ani vždy pozitivně definitní na prostoru omezení. Jedna z možností, jak zajistíme její pozitivní definitnost je, že použijeme místo matice W_k její aproximaci B_k, kterou získáme pomocí kvazi-Newtonova algoritmu.

3.3.1 Kvazi-newtonovské metody

Kvazi-newtonovské metody jsou gradientní metody, které leží někde mezi metodou nejrychlejšího sestupu a Newtonovou metodou. Snaží se využít přednosti obou metod. Gradientní metody mají zaručenou konvergenci a Newtonova metoda má v okolí optima řád konvergence rovný dvěma. Newtonova metoda ale vyžaduje výpočet Hesseho matice. Kvazi-newtonovské metody, podobně jako metoda nejrychlejšího sestupu, potřebují v každé iteraci pouze gradient ztrátové funkce. Jeden z kvazi-newtonských algoritmů je BFGS metoda [5], pojmenovaná po svých tvůrcích Brodyen, Fletcher, Goldfarb and Shanno, jejíž odvození zde stručně zopakujeme.
Uvažujme následující kvadratický model ztrátové funkce

$$m_k(p) = f_k + \nabla f_k^T p + \frac{1}{2} p^T B_k p, \tag{3.20}$$

kde B_k je symetrická pozitivně definitní matice $n \times n$. Tato matice je v každém kroku akualizovaná.

Pro minimum tohoto konvexního kvadratického modelu platí, že jeho první derivace je v tomto bodě rovna nule, tedy $\nabla_p m(p) = \nabla f_k + B_k p = 0$. Explicitně lze

napsat

$$p_k = -B_k^{-1} \nabla f_k, \tag{3.21}$$

kde $p_k \in \mathbb{R}^{n \times n}$ je směr poklesu funkce f a použijeme jej k získání nového kroku

$$x_{k+1} = x_k + \alpha_k p_k, \tag{3.22}$$

kde $\alpha_k \in \mathbb{R}$ je délka kroku a volíme ji tak, aby splňovala Wolfeho podmínky (2.3). Takto získaná iterace je obdobná line search metodě, pouze s tím rozdílem, že jsme hesián nahradili jeho aproximací B_k.

Když jsme získali nový krok x_{k+1}, vytvoříme pro něj nový kvadratický model

$$m_{k+1}(p) = f_{k+1} + \nabla f_{k+1}^T p + \frac{1}{2} p^T B_{k+1} p, \tag{3.23}$$

Nyní požadujeme, aby gradient funkce m_{k+1} odpovídal gradientu ztrátové funkce v posledních dvou krocích x_k a x_{k+1}. Jelikož $\nabla m_{k+1}(0) = \nabla f_{k+1}$, tak je podmínka pro následující krok splněná automaticky. Podmínku pro aktuální krok můžeme zapsat matematicky jako

$$\nabla m_{k+1}(-\alpha_k p_k) = \nabla f_{k+1} - \alpha_k B_{k+1} p_k = \nabla f_k. \tag{3.24}$$

Úpravou obdržíme

$$B_{k+1} \alpha_k p_k = \nabla f_{k+1} - \nabla f_k. \tag{3.25}$$

Pro zjednodušení zápisu zavedeme následující vektory

$$s_k = x_{k+1} - x_k, \quad y_k = \nabla f_{k+1} - \nabla f_k, \tag{3.26}$$

potom se (3.25) změní na

$$B_{k+1} s_k = y_k. \tag{3.27}$$

Máme dáno posunutí s_k a změnu gradientu y_k, rovnice (3.27) požaduje, aby pozitivně definitní matice B_{k+1} zobrazovala s_k na y_k. Toto je možné pouze pokud s_k a y_k splňují podmínky zakřivení

$$s_k^T y_k > 0, \tag{3.28}$$

Pokud je f silně konvexní [5], můžeme přenásobením rovnice (3.27) vektorem s_k^T dokázat, že nerovnice (3.28) je splněna pro jakékoliv dva body x_k a x_{k+1}. Avšak pokud je funkce f nekonvexní, tak tato podmínka nemusí být vždy splněna a pro tento případ musíme ověřit nerovnost (3.28) přímo, a to přidáním omezení do line

search postupu pro nalezení α. Tedy podmínka (3.28) bude zaručeně splněna, když do line search metody přidáme Wolfeho podmínku.

Jelikož do rovnice (3.21) pro výpročet p_k potřebujeme dosadit inverzi matice B_k, je mnohem efektivnější počítat přímo její inverzi. Zavedeme matici $H_k = B_k^{-1}$. Potom dosazením do (3.27) získáme

$$H_{k+1} y_k = s_k. \tag{3.29}$$

Když je podmínka zakřivení (3.28) splněna, potom řešením rovnice (3.29) vždy získáme matici H_{k+1}. Prakticky však tento vztah vede na nekonečně mnoho řešení, jelikož v symetrické matici máme $n(n+1)/2$ stupňů volnosti a rovnice (3.29) představuje pouze n podmínek. Požadavek pozitivní definitnosti sice vkládá dalších n nerovnic, ale ani tyto podmínky nedokáží ubrat zbývající stupně volnosti.

K jednoznačnému určení matice H_{k+1} proto musíme přidat další omezení tak, aby matice H_{k+1} byla v jistém smyslu co nejblíže k aktuální matici H_k. Jinými slovy musíme vyřešit následující problém

$$\begin{aligned} &\min_H \|H_{k+1} - H_k\| \\ \text{s omezením} \quad &H_{k+1} = H_{k+1}^T, \quad H_{k+1} y_k = s_k, \end{aligned} \tag{3.30}$$

kde s_k a y_k splňují (3.28) a matice H_k je symetrická a pozitivně definitní. K řešení tohoto problému můžeme použít mnoho maticových norem a každá nám vytvoří rozdílnou kvazi-Newtonovu metodu. My použijeme váženou Frobeniovu normu, která nám umožní jednoduše řešit minimalizační problém (3.30) a vede na scale-invariant[3] optimalizační metodu. Vážená Frobeniova norma

$$\|A\|_P \equiv \left\| P^{1/2} A P^{1/2} \right\|_F, \tag{3.31}$$

kde $\|\cdot\|_F$ je definována jako $\|C\|_F^2 = \sum_{i=1}^n \sum_{j=1}^n c_{ij}^2$. Váhová matice P může být zvolena jako libovolná matice splňující vztah $P s_k = y_k$. Konkrétně zvolíme $P = \bar{G}_k^{-1}$, kde $\bar{G}_k$ je průměrný hesián definovaný jako

$$\bar{G}_k = \left[\int_0^1 \nabla^2 f(x_k + \tau \alpha_k p_k) d\tau \right]. \tag{3.32}$$

Při použití Frobeniovy normy s uvedenou váhovou maticí, obdržíme řešením rovnice (3.30) jednoznačné určenou matici H_{k+1} danou následujícím vztahem

$$H_{k+1} = (I - \rho_k s_k y_k^T) H_k (I - \rho_k y_k s_k^T) + \rho_k s_k s_k^T, \tag{3.33}$$

[3]je invariantní vůči změně měřítka jednotlivých souřadnic

kde

$$\rho_k = \frac{1}{y_k^T s_k}. \tag{3.34}$$

Poslední věc, kterou musíme vyřešit předtím, než budeme moci definovat kompletní BFGS algoritmus, je volba počáteční hodnoty aproximace H_0. Jednoduchý vztah, který by platil pro všechny druhy úloh, ovšem neexistuje. Můžeme použít určité informace o úloze a určit ji jako inverzi aproximovaného hesiánu, který spočítáme pomocí konečných diferencí [5] v bodě x_0, nebo ji můžeme zvolit jednoduše jako jednotkovou matici, popřípadě jako násobek jednotkové matice. K nastavení matice H_0 je vhodné použít následující heuristiku. Počáteční matici nastavíme až po prvním kroku, který spočteme ještě dříve, než provedeme první BFGS aktualizaci. Poté změníme prozatimní hodnotu $H_0 = I$ na

$$H_0 \leftarrow \frac{y_k^T s_k}{y_k^T y_k} I. \tag{3.35}$$

Algoritmus 4 (Metoda BFGS)

Zvol počáteční bod (x_0), toleranci konvergence $\epsilon > 0$ a inverzi aproximovaného Hesiánu H_0;
$k \leftarrow 0$;
while $\|\nabla f_k\| > \epsilon$ **do**
  Vypočti směr hledání: $p_k = -H_k \nabla f_k$;
  Nastav ($x_{k+1} = x_k + \alpha_k p_k$, kde α_k je vypočtena z line search procedury, aby splňovala Wolfeho podmínky;
  Definuj $s_k = x_{k+1} - x_k$ a $y_k = \nabla f_{k+1} - \nabla f_k$;
  Spočti H_{k+1} pomocí (3.33);
  $k \leftarrow k + 1$;
end while

Tento algoritmus je robustní a jeho řád konvergence je superlineární [5]. Ačkoliv Newtonova metoda konverguje mnohem rychleji (kvadraticky), je cena její iterace mnohem větší, protože je třeba řešit lineární systém(3.11). Další, mnohem důležitější výhoda BFGS je, že není potřeba počítat druhé derivace ztrátové funkce.

Dále můžeme ještě odvodit verzi BFGS algoritmu, která pracuje přímo s aproximací hesiánu B_k. Použitím Sherman-Morrison-Woodburyho rovnice [5] na (3.33) získáme vztah pro aktualizaci B_k

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k}. \tag{3.36}$$

Zde jsme odvodili základní BFGS algoritmus sloužící k aproximaci hesiánu obecného kvadratického modelu. Tento algoritmus nyní použijeme k aproximaci hesiánu W_k v kvadratickém modelu SQP (3.12). Ztrátová funkce SQP je daná Lagrangeovou funcí (3.13). Vztah pro s_k a y_k potřebný v BFGS algoritmu upravíme na

$$s_k = x_{k+1} - x_k, \quad y_k = \nabla_x L(x_{k+1}, \lambda_{k+1}) - \nabla_x L(x_k, \lambda_{k+1}). \tag{3.37}$$

Poté můžeme pomocí vztahu (3.36) vypočíst novou aproximaci Hesiánu B_{k+1}.

Tento přístup má určité přednosti i nedostatky. Pokud je matice $\nabla^2_{xx} L$ pozitivně definitní v oblasti, kde se nalézá minimum, odráží kvazi-Newtonova aproximace B_k určité informace o zakřivení problému a iterace konvergují robustně a rychle, podobně jako BFGS metoda bez omezení. Obsahuje-li však matice $\nabla^2_{xx} L$ záporná vlastní čísla, potom BFGS přístup může být neefektivní. Důležitou podmínkou BFGS aktualizace je splnění podmínky zakřivení (3.28), která nemusí být splněna, pokud jsou vektory s_k a y_k definovány dle (3.37) a iterace nejsou dostatečně blízko řešení.

Abychom se vyvarovali těmto obtížím, můžeme v daném kroku vynechat BFGS aktualizaci matice ($B_{k+1} = B_k$), pokud není splněna podmínka

$$s_k^T y_k \geqslant \theta s_k^T B_k s_k, \tag{3.38}$$

kde θ je kladný parametr (např. 10^{-2}). Tato úprava, která v některých krocích vynechává aktualizaci matice B_k, je použita v některých implementacích SQP a dosahuje dobrých výsledků pro mnoho problémů [5]. Ale pro některé problémy tento postup selhává. Proto se zaměříme na efektivnější modifikaci, která zaručí vždy dobře vymezenou aktualizaci.

3.3.2 Tlumená BFGS aktualizace pro SQP

Z vektorů s_k a y_k definovaných dle (3.37) vypočteme

$$r_k = \theta_k y_k + (1 - \theta_k) B_k s_k, \tag{3.39}$$

kde skalár θ_k je

$$\theta_k = \begin{cases} 1 & \text{pokud} \quad s_k^T y_k \geqslant 0,2 s_k^T B_k s_k, \\ (0,8 s_k^T B_k s_k)/(s_k^T B_k s_k - s_k^T y_k) & \text{pokud} \quad s_k^T y_k < 0,2 s_k^T B_k s_k. \end{cases} \tag{3.40}$$

Vztah pro aktualizaci B_k je následující:

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{r_k r_k^T}{s_k^T r_k}. \tag{3.41}$$

Tato rovnice je stejná jako standartní BFGS rovnice (3.36) pouze s tím rozdílem, že vektor y_k je nahražen vektorem r_k. Tím je zaručena pozitivní definitnost matice B_{k+1}, což můžeme dokázat [5]. Pokud $\theta_k \neq 1$, tak platí

$$s_k^T r_k = 0, 2 s_k^T B_k s_k > 0. \tag{3.42}$$

Dále můžeme ještě poznamenat, že pokud je $\theta_k = 0$ dostaneme $B_{k+1} = B_k$ a že volba $\theta_k = 1$ vede na stejné řešení jako původní BFGS metoda. Tedy hodnota $\theta_k \in (0, 1)$ interpoluje aktuální hodnotu aproximace B_k a aproximaci získanou původní BFGS metodou. Volba patrametru dle (3.40) zajistí, že nová aproximace bude dostatečně blízko aktuální aproximace B_k a bude zaručena její pozitivní definitnost. Dále uvedeme aproximaci hesiánu ztrátové funkce pomocí exact penalty funkce.

3.4 Exact penalty funkce

Další možnost, jak se vypořádat s tím, že matice W_k v kvadratickém modelu (3.12) nemusí být vždy pozitivně definitní, je rozšíření ztrátové funkce v (3.1) o penalizační funkci $\frac{\pi}{2} \|h(x)\|^2$. Minimalizační problém se nám tím změní na [6]

$$\begin{aligned} &\min_x f(x) + \frac{\pi}{2} \|h(x)\|^2 \\ &\text{s omezením} \quad h(x) = 0, \end{aligned} \tag{3.43}$$

kde π je skalár. Takto zadaný minimalizační problém má stejné lokální minimum jako náš původní problém (3.1) [6]. Pro tento problém sestavíme Lagrangeovu funkci

$$L_c(x, \lambda) = f(x) + \lambda^T h(x) + \frac{\pi}{2} \|h(x)\|^2 . \tag{3.44}$$

hesián této funkce je v bodě optima x^* a λ^* roven

$$\nabla^2_{xx} L_c(x^*, \lambda^*) = \nabla^2_{xx} L(x^*, \lambda^*) + \pi \nabla h(x^*) \nabla h(x^*)^T. \tag{3.45}$$

Pokud bude π dostatečně veliké ($\pi > \pi_0$), kde π_0 je prahová hodnota, pak bude Hesián Lagrangeovy funkce

$$\nabla^2_{xx} L_c(x^*, \lambda^*) : \text{pozitivně definitní}, \quad \forall \pi > \pi_0. \tag{3.46}$$

Tedy jinými slovy přidáním členu $\pi \nabla h(x^*) \nabla h(x^*)^T$ k původnímu hesiánu W_k zajistíme jeho pozitivní definitnost a tím splnění Podmínek 1. V nasledující části ukážeme další možnost výpočtu hesiánu ztrátové funkce, která nám opět zajistí jeho pozitivní definitnost.

3.5 Nelineární nejmenší čtverce

Ztrátovou funkci, kterou se snažíme minimalizovat, můžeme napsat v následujícím tvaru

$$f(x) = \frac{1}{2}\sum_{j=1}^{m} r_j^2(x), \tag{3.47}$$

kde každá r_j je hladká funkce $\mathbb{R}^n \to \mathbb{R}$ a nazvěme ji reziduem.

Minimum takto zadané funkce můžeme nalézt pomocí metody nejmenších čverců. Ta je pro takto definovanou ztrátovou funkci mnohem snáze řešitelná než obecný minimalizační problém. Nejprve sestavíme vektor reziduí $r : \mathbb{R}^n \to \mathbb{R}^m$, který bude vytvořený z jednotlivých komponent r_j v rovnici (3.47)

$$r(x) = (r_1(x), r_2(x), \ldots, r_m(x))^T. \tag{3.48}$$

Použitím této notace můžeme přepsat ztrátovou funkci f jako $f(x) = \frac{1}{2}\|r(x)\|_2^2$. Derivaci funkce f můžeme vyjádřit pomocí Jakobiánu vektoru r, což je matice prvních parciálních derivací o velikosti $m \times n$

$$\nabla r(x) = \begin{bmatrix} \frac{\partial r_1}{\partial x_1} & \frac{\partial r_1}{\partial x_2} & \cdots & \frac{\partial r_1}{\partial x_n} \\ \vdots & \vdots & & \vdots \\ \frac{\partial r_m}{\partial x_1} & \frac{\partial r_m}{\partial x_2} & \cdots & \frac{\partial r_m}{\partial x_n} \end{bmatrix}. \tag{3.49}$$

Poté první a druhá derivace funkce f jsou

$$\nabla f(x) = \sum_{j=1}^{m} r_j(x)\nabla r_j(x) = \nabla r(x)^T r(x), \tag{3.50}$$

$$\begin{aligned} \nabla^2 f(x) &= \sum_{j=1}^{m} \nabla r_j(x)\nabla r_j(x)^T + \sum_{j=1}^{m} r_j(x)\nabla^2 r_j(x) \\ &= \nabla r(x)^T \nabla r(x) + \sum_{j=1}^{m} r_j(x)\nabla^2 r_j(x). \end{aligned} \tag{3.51}$$

V mnoha aplikacích je možné explicitně spočítat parciální derivace, tím získat Jakobián $\nabla r(x)$ a ten poté použít k výpočtu gradientu $\nabla f(x)$ dle vztahu (3.50). Avšak hlavním rysem metody nejmenších čtverců je, že ze znalosti Jakobiánu $\nabla r(x)$ můžeme snadno získat první část hesiánu $\nabla^2 f(x)$. Tento první člen v rovnici (3.51) je často mnohem důležitější než člen druhý, který, pokud je dostatečně malý, můžeme zanedbat. Zanedbání je možné tehdy, pokud jsme blízko optima, tedy jsou malá rezidua $r_j(x)$, nebo pokud je model jen slabě nelineární, tedy $\nabla^2 r_j$ jsou malá. Zanedbáním zajistíme, že $\nabla^2 f(x)$ bude pozitivně definitní. Takto jednoduchý způsob výpočtu hesiánu můžeme použít například v line search metodě.

3.5.1 Gauss-Newtonova metoda

Nyní ukážeme metodu pro minimalizaci ztrátové funkce (3.47) využívající výše popsanou strukturu gradientu $\nabla f(x)$ a hesiánu $\nabla^2 f(x)$. Tato metoda vychází z obecné Newtonovy metody. Směr poklesu ztrátové funkce p_k získáme řešením Newtonovy rovnice $\nabla^2 f(x_k)p_k = -\nabla f(x_k)$, do které dosadíme výše odvozené vztahy pro ∇f a $\nabla^2 f$. Přičemž v rovnici pro vypočet $\nabla^2 f$ zanedbáme druhý člen $\sum_{j=1}^{m} r_j \nabla^2 r_j$. Tím získáme Gauss-Newtonovu metodu

$$\nabla r_k^T \nabla r_k p_k = -\nabla r_k^T r_k. \tag{3.52}$$

Touto jednoduchou modifikací obdržíme několik výhod oproti původní Newtonově metodě. První výhoda je, že použitím aproximace

$$\nabla^2 f_k \approx \nabla r_k^T \nabla r_k. \tag{3.53}$$

se vyhneme obtížím při výpočtu jednotlivých hesiánů $\nabla^2 r_i, i = 1, 2, \ldots m$ reziduí, které jsou potřeba v druhém členu (3.51). Tedy po spočtení Jakobiánu ∇r_k, který potřebujeme pro výpočet gradientu ztrátové funkce, můžeme jednoduše vypočíst aproximaci hesiánu ztrátové funkce a ušetřit tím čas, který by byl potřebný k výpočtu druhých parciálních derivací jednotlivých reziduí. Druhou výhodou je, že v mnoha aplikacích a také v prediktivním řízení, má první člen $\nabla r_k^T \nabla r_k$ v rovnici (3.51) mnohem větší význam než druhý člen a proto Gauss-Newtonova metoda dává velice podobná řešení jako původní Newtonova metoda, přestože jsme zanedbali druhý člen $\sum_{j=1}^{m} r_j \nabla^2 r_j$. Dostatečnou podmínkou pro to, aby první člen (3.51) převládal nad druhým členem je, aby velikost každého členu druhého řádu (který je $|r_j(x)| \, \|\nabla^2 r_j(x)\|$) byla mnohem menší než vlastní čísla matice $\nabla r_k^T \nabla r_k$. Tato podmínka je splněna v případě, že jsou malá rezidua $|r_j|$ (tzv. případ malých reziduí) nebo když se každá r_j blíží k lineární funkci a tedy $\|\nabla^2 r_j\|$ je malé. Třetí výhodou Gauss-Newtonovy metody je, že když $\nabla r(x_k)$ má plnou hodnost a gradient ∇f je nenulový, potom směr p_k je směr poklesu ztrátové funkce $f(.)$, a je to tedy použitelý směr pro line search metodu. Z rovnic (3.50) a (3.52) dostaneme

$$p_k^T \nabla f_k = p_k^T \nabla r_k^T r_k = -p_k^T \nabla r_k^T \nabla r_k p_k = -\left\|\nabla r_k p_k\right\|_2^2 \leqslant 0. \tag{3.54}$$

Poslední nerovnost je striktní právě když $\nabla r_k p_k = 0$, což dle (3.52) je ekvivalentní $\nabla r_k r_k = \nabla f_k = 0$. A konečně čtvrtá výhoda této metody vyplývá z podobnosti Gauss-Newtonovy rovnice (3.52) a obecné rovnice pro lineární nejmenší čtverce [5]. Z této podobnosti je vidět, že p_k můžeme získat jako řešení problému lineárních

nejmenších čtverců

$$\min_p \|\nabla r_k p_k + f_k\|_2^2 . \tag{3.55}$$

Předchozí podúloha naznačuje další motivaci pro Gauss-Newtonův krok. Místo abychom vytvořili kvadratický model funkce $f(x)$, vytvoříme lineární model vektorové funkce $r(x + p_k) \approx r(x) + \nabla r(x)p_k$. Poté krok p_k získáme nahražením tohoto lineárního modelu do výrazu $f(x) = \frac{1}{2}\|r(x)\|_2^2$ a jeho minimalizací přes p_k.

Jak již bylo uvedeno výše, směr získaný řešením Gauss-Newtonovy rovnice můžeme použít v line search metodě. Jedná se tedy o další možnost, jak aproximovat hesián ztrátové funkce a vyhnout se tím nutnosti počítat druhé derivace v případě, že ztrátová funkce má tvar (3.47).

Výše uvedený postup lze použít nejen pro minimalizační úlohy bez omezení, jak zde bylo odvozeno, ale také pro minimalizační úlohy s omezeními. Pro tento případ opět sestavíme Lagrangeovu funkci, tím zahrneme omezení do ztrátové funkce, a pokud i Lagrangeova funkce má tvar dle (3.47), můžeme použít Gauss-Newtonovu metodu k nalezení nového směru postupu.

3.6 Metoda SQP s jednorozměrovým hledáním

Z předchozích částí této kapitoly můžeme vidět, že existuje široké spektrum SQP metod, které využívají jednorozměrové hledání (line search). Tyto metody se liší výpočtem aproximace hesiánu nebo volbou tvaru ztrátové funkce. Nyní uvedeme praktický kvazi-Newtonův algoritmus pro řešení problému nelineárního programování.

Od tohoto algoritmu lze odvodit jeho několik variant tím, že použijeme různé způsoby aproximace hesiánu B_k, např. BFGS metodou nebo Gauss-Newtonovu metodu pro tvar ztrátové funkce dle (3.47). Můžeme také použít místo B_k přesný hesián z Lagrangiánu W_k. Jedinou podmínkou je, aby B_k byl pozitivně definitní.

V následující části ověříme na zvoleném příkladě výše uvedené algoritmy s různými aproximacemi hesiánu a provedeme jejich porovnání.

Algoritmus 5 (Algoritmus SQP pro nelineární program)

Zvol patrametry $\eta \in (0, 0.5), \tau \in (0, 1)$;

Zvol počáteční pár (x_0, λ_0) a počáteční symetrickou pozitivně definitní aproximaci Hesiánu B_0 o velikosti $n \times n$;

Vypočti hodnotu $f_0, \nabla f_0, h_0$ a A_0;

for $k = 0, 1, 2, \ldots$ **do**

  if je splněna konvergence **then**

    STOP s přibližným řešením (x_k, λ_k);

  end if

  Řešením (3.16) získej p_k;

  Zvol μ_k tak, aby p_k byl směr poklesu funkce ϕ v bodě x_k;

  Nastav $\alpha_k = 1$;

  while $\phi(x_k + \alpha_k p_k, \mu_k) > \phi(x_k, \mu_k) + \eta \alpha_k \phi'(x_k, \mu_k)$ **do**

    Uprav $\alpha_k \leftarrow \tau_\alpha \alpha_k$ pro nějaké $\tau_\alpha \in (0, \tau)$;

  end while

  Nastav $x_{k+1} = x_k + \alpha_k p_k$;

  Vypočti hodnotu $f_{k+1}, \nabla f_{k+1}, h_{k+1}$ a A_{k+1};

  Spočti λ_{k+1} řešením rovnice: $\lambda_{k+1} = -[A_{k+1} A_{k+1}^T]^{-1} A_{k+1} \nabla f_{k+1}$;

  Nastav $s_k = \alpha_k p_k, \quad y_k = \nabla_x L(x_{k+1}, \lambda_{k+1}) - L(x_k, \lambda_{k+1})$;

  Použitím kvazi-Newtonovy rovnice aktualizuj B_k a získej B_{k+1};

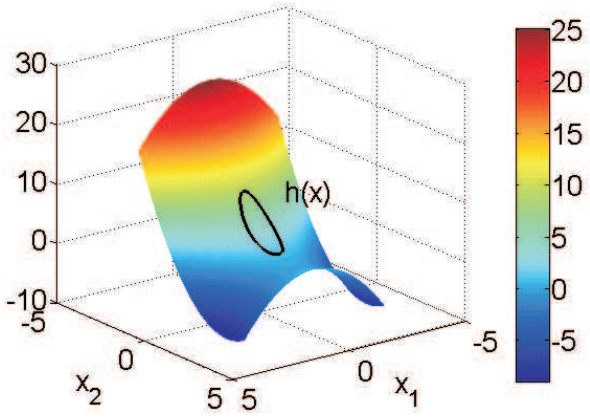
end for

3.7 Experimentální ověření

Uvažujme následující optimalizační problém

$$\min_x (x_2 - 2)^2 - x_1^2, \tag{3.56}$$
$$\text{s omezením} \quad 4x_1^2 + x_2^2 - 1 = 0,$$

Průběh ztrátové funkce $f(x)$ s omezující funkcí $h(x)$ je zobrazen na obr. 3.1. Odkud je vidět, že ztrátová funkce nabývá minima za daných omezení v bodě $x^* = [0, 1]$ (s odpovídajícím Lagrangeovým koeficientem $\lambda^* = 1$).



Obrázek 3.1: Ztrátová funkce

3.7.1 Lokální metody

Pro nalezení minima funkce $f(x)$ jsme použili postupně čtyři metody: Newton-Lagrangeovu metodu, metodu sekvenčního kvadratického programování, metodu nejrychlejšího sestupu (SD - steepest descent) a BFGS metodou. Všechny čtyři algoritmy jsme porovnávali pro dva počáteční body blízké optimu $x_0 = [3, 3]$ a $x_0 = [1, -2]$ se zvoleným počátečním odhadem Lagrangeova koeficientu $\lambda_0 = 0, 5$. Pro metody Newton-Lagrange, SQP a BFGS byla nastavena délka kroku $\alpha = 1$, u metody nejrychlejšího sestupu musela být délka kroku snížena na $\alpha = 0, 3$ pro zajištění konvergence výsledku. U metody BFGS bylo nutno navíc zvolit počáteční aproximaci hesiánu $B_0 = 1, 6 * I$.

V tabulce 3.1 jsou zaznamenány hodnoty ztrátové funkce v jednotlivých krocích daných metod pro oba počáteční body x_0. Jelikož je Newton-Lagrangeova metoda ekvivalentní metodě SQP (pokud jsou splněny Podmínky 1), měly obě metody v

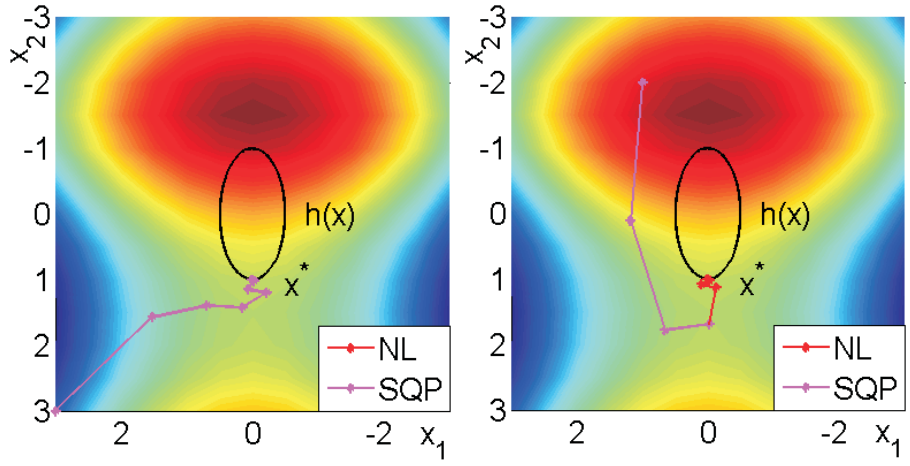
	$x_0 = [3, 3]$			$x_0 = [1, -2]$		
krok	SQP	SD	BFGS	SQP	SD	BFGS
0	-8,00	-8,00	-8,00	15,00	15,00	15,00
1	-2,14	-7,44	-0,98	-0,97	-0,49	-5,78
2	-0,19	-5,29	-0,34	-0,39	-1,92	-1,00
3	0,22	-3,66	0,16	0,09	-1,35	0,27
4	0,56	-2,50	0,23	-	-0,84	0,19
5	0,58	-1,66	0,46	-	-0,46	0,48
6	0,96	-1,05	0,83	-	-0,19	0,57
7	1,00	-0,61	0,90	-	0,02	0,91
8	**1,00**	-0,29	0,98	-	0,17	0,96
9		-0,06	1,00	-	0,28	0,99
10		0,11	**1,00**	-	0,39	1,00
11		0,24		-	0,50	**1,00**
⋮		⋮		⋮	⋮	
24		0,98		-	**1,00**	
⋮		⋮		⋮		
27		**1,00**		-		

Tabulka 3.1: Porovnání metod

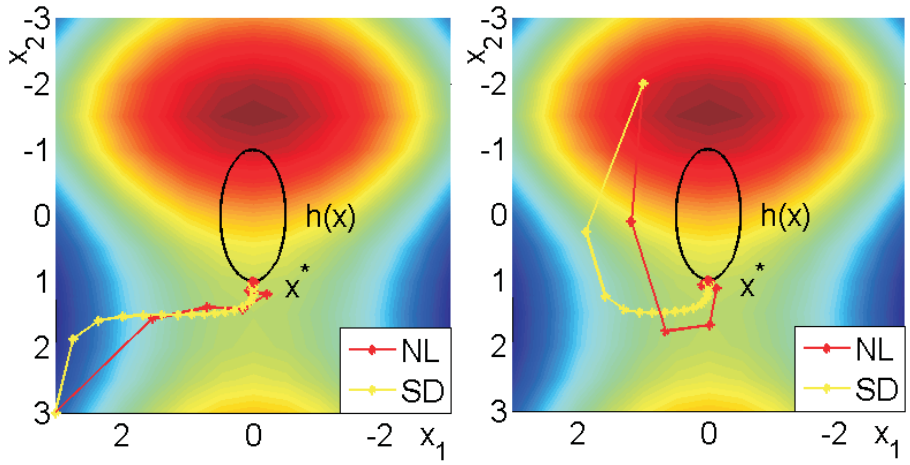
každém kroku stejné řešení. Ve čtvrtém kroku metody SQP pro druhý počáteční bod $x_0 = [1, -2]$ však nebyly tyto podmínky splněny a SQP algoritmus selhal .

Jak je vidět z tabulky 3.1, nejrychleji nalezla řešení pro první počáteční bod metoda SQP (po 8 iteracích), používající přesný hesián Larangeovy funkce. Metoda BFGS potřebovala k nalezení řešení 10 iterací, protože používá pouze aproximaci hesiánu. Největší počet iterací potřebovala metoda nejrychlejšího sestupu (27 iterací), která používá místo přesného hesiánu jednotkovou matici. Pro druhý počáteční bod se nepodařilo metodě SQP nalézt řešení, protože použitý přesný Hessián Lagrangeovy funkce byl ve čtvrtém kroku negativně definitní. Naproti tomu metoda BFGS nalezla řešení po 11 iteracích a postupnou aproximací hesiánu zajistila v každé iteraci jeho pozitivní definitnost. Metoda nejrychlejšího sestupu opět potřebovala k nalezení řešení nejvíce iterací (24 iterací).

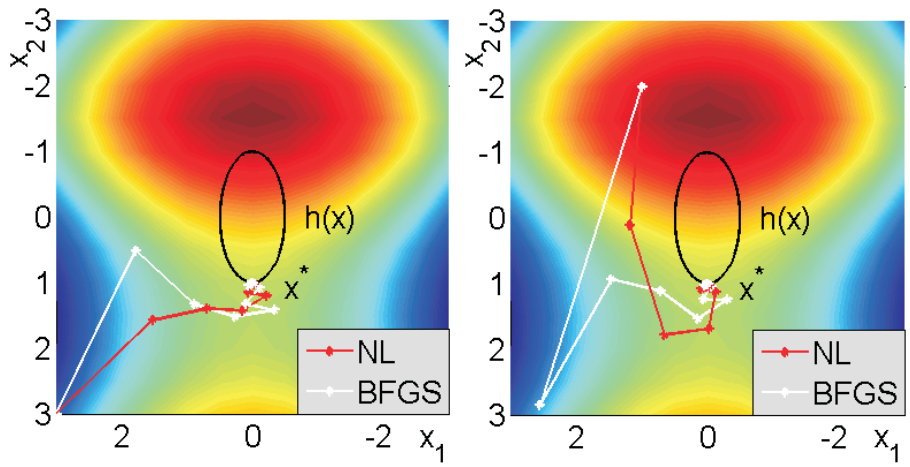
Na obrázcích 3.2, 3.3 a 3.4 jsou zobrazeny průběhy jednotlivých metod: červeně - metoda Newton-Lagrange, falově - metoda SQP, žlutě - metoda SD a bíle - metoda BFGS. Černou barvou je znázorněna funkce omezení $h(x)$.



Obrázek 3.2: Newton-Lagrangeova a SQP metoda



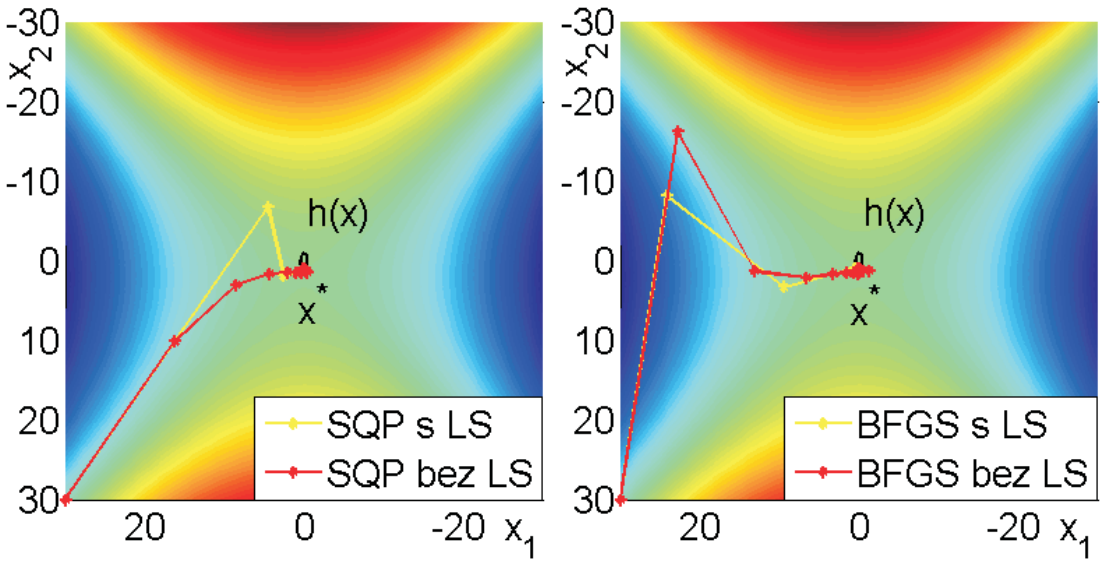
Obrázek 3.3: Metoda nejrychlejšího sestupu



Obrázek 3.4: BFGS metoda

3.7.2 Metody využívající jednorozměrové hledání

V předchozím příkladě jsme začali hledat minimum ztrátové funkce z bodu, jenž ležel v jeho blízkosti, nyní budeme hledat minimum stejné funkce (3.56), avšak počáteční bod x_0 bude více vzdálen od optima. K hledání použijeme dvě metody: SQP a BFGS. Obě metody jsou stejné jako v předchozím příkladu, ale navíc jsou rozšířené o line search algoritmus, který vrací délku kroku α splňující Wolfeho podmínky (2.3). Výpočet provedeme pro počáteční bod $x_0 = [30, 30]$ s odhadem Lagrangeova koeficientu $\lambda_0 = 2$. Pro BFGS metodu byl zvolena počáteční aproximace hesiánu $B_0 = 2 * I$.



Obrázek 3.5: Porovnání metod využívající line search

Na obrázku (3.5) vlevo je zobrazen průběh hledání optima metodou SQP, kde je žlutou barvou označen průběh s použitím line search algoritmu a červenou barvou bez jeho použití. Na druhém obrázku vpravo je obdobně zobrazen růběh hledání optima metodou BFGS. Z obou obrázků je vidět funkce line search algoritmu, kdy v každé iteraci prodlouží nebo naopak zkrátí délku kroku tak, aby se v daném směru hledání co nejvíce přiblížil optimu ztrátové funkce. Tím se sníží počet potřebných iterací pro nalezení minima. V našem případě použitím line search algoritmu byl snížen počet potřebných iterací u obou metod o čtyři iterace, jak je také možno vidět i v následující tabulce (3.2).

	bez LS	s LS
SQP	14	10
BFGS	17	13

Tabulka 3.2: Počet iterací při použití line search algoritmu

3.8 Shrnutí

V této kapitole jsme uvedli postup řešení nelineární optimalizační úlohy za použití sekvenčního kvadratického programování a ukázali jsme jeho ekvivalenci s Newtonovou metodou aplikovanou na podmínky prvního řádu. Byly odvodozeny následující možné způsoby aproximace hesiánu W_k v kvadratickém modelu ztrátové funkce, které nám pomohou vyhnout se nutnosti počítat její druhé parciální derivace.

1. BFGS metoda - provedeme aktualizaci aproximovaného hesiánu B_k tak, aby byl v každém kroku pozitivně definitní

2. Exact penalty funkce - rozšíření ztrátové funkce o penalizační funkci, která opět zajistí pozitivní definitnost hesiánu.

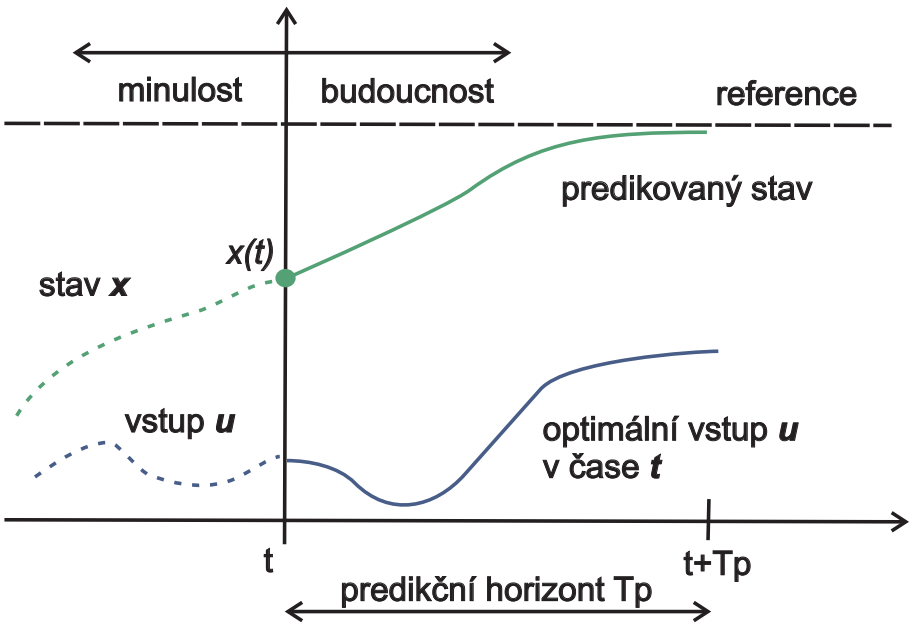
3. Gauss-Newtonova metoda - tu můžeme použít v případě, že je ztrátová funkce ve tvaru nelineárních nejmeších čtverců.

Dále byl uveden praktický algoritmus sekvenčního kvadratického programování využívající line search metodu k nalezení optimální délky kroku. Nakonec byly na příkladě porovnány algoritmy používající sekvenční kvadratické programování, metodu nejrychlejšího sestupu a metodu BFGS.

Kapitola 4

Prediktivní regulátor

Prediktivní řízení je metoda, která uplatňuje on-line optimalizaci na model systému s cílem řídit systém k požadovanému cílovému stavu. Na obr.4.1 je zobrazen základní princip prediktivního řízení. Odezva soustavy na potenciální vektor řízení je



Obrázek 4.1: Princip prediktivního řízení

získávána pomocí predikčního modelu daného systému. Proto je pro správnou funkci prediktivního regulátoru důležitá volba správného modelu a také přesná identifikace parametrů tohoto modelu. Další věc, která má vliv na kvalitu a vlastnosti regulace, je volba ztrátové funkce, jež je minimalizována MPC algoritmem. Jedna z důležitých výhod prediktivního řízení je, že lze do návrhu jednoduše zahrnout omezení. Pomocí MPC regulátoru lze řídit i systémy s více vstupy a více výstupy (Multiple Input

Multiple Output - MIMO).

Prediktivní regulátory lze rozdělit dle použitého predikčního modelu na dvě základní skupiny: lineární a nelineární. Řešení úlohy lineárního MPC je většinou jednodušší, avšak pokud řízená soustava obsahuje silnou nelinearitu a pokud se nepohybujeme v blízkém okolí pracovního bodu, stává se lineární prediktivní regulátor nepřesným. Této nepřesnosti se můžeme zbavit použitím nelineárního modelu soustavy. Nevýhodou nelineárního prediktivního regulátoru je však časově náročný výpočet optimalizace. MPC algoritmus musí v čase mezi dvěma vzorkovacími okamžiky vyřešit nelineární optimalizační úlohu, proto nelineární prediktivní regulátor lze použít pouze pro pomalejší soustavy s dostatečně dlouhou periodou vzorkování. Nelineární predikční model popíšeme v následující části spolu s odvozením citlivostních matic.

4.1 Predikční model

Dynamické chování nelineárního dynamického systému můžeme popsat následujícími rovnicemi

$$\begin{aligned} \dot{x}(t) &= f(x(t), u(t)) \\ y(t) &= h(x(t), u(t)), \end{aligned} \tag{4.1}$$

kde $u(t) \in \mathbb{R}^m$, $x(t) \in \mathbb{R}^n$ a $y(t) \in \mathbb{R}^p$ jsou vstupy, stavy a výstupy systému. Predikce budoucích stavů systému závisí na počátetční hodnotě stavu v aktuální čase a na budoucí posloupnosti vstupů. Matice, které popisují vliv změny dané proměnné na změnu počáteční podmínky, se nazývají citlivostní matice.

Citlivostní matice stavu $x(t)$ na počáteční podmínku je definována jako derivace $x(t)$ podle počáteční podmínky $x(t_0)$

$$\Phi(t, t_0) = \frac{\partial x(t)}{\partial x(t_0)}. \tag{4.2}$$

V čase $t + dt$ platí

$$\Phi(t + dt, t_0) = \frac{\partial x(t + dt)}{\partial x(t_0)} = \frac{\partial [x(t) + f(x(t), u(t))dt]}{\partial x(t_0)}, \tag{4.3}$$

kde jsme použili aproximaci Taylorovým rozvojem[1] prvního řádu. Nyní můžeme pokračovat v úpravách

$$\Phi(t + dt, t_0) \;=\; \frac{\partial x(t)}{\partial x(t_0)} + \left[\frac{\partial f(x(t), u(t))}{\partial x(t)} \frac{\partial x(t)}{\partial x(t_0)}\right] dt \tag{4.4}$$

[1] Taylorův rozvoj: $x(t + dt) = x(t) + \dot{x}(t)dt + \frac{1}{2!}\ddot{x}(t)dt^2 \ldots$

$$= \Phi(t, t_0) + \frac{\partial f(x(t), u(t))}{\partial x(t)} \Phi(t, t_0) dt \tag{4.5}$$

a dále můžeme napsat

$$\frac{\Phi(t + dt, t_0) - \Phi(t, t_0)}{dt} = \frac{\partial f(x(t), u(t))}{\partial x(t)} \Phi(t, t_0) \tag{4.6}$$

Použitím definice derivace

$$\dot{\Phi}(t, t_0) = \lim_{dt \to 0} \frac{\Phi(t + dt, t_0) - \Phi(t, t_0)}{dt}, \tag{4.7}$$

lze dynamiku citlivostní matice $\Phi(t, t_0)$ vyjádřit pomocí následující diferenciální rovnice

$$\dot{\Phi}(t, t_0) = \nabla_{x(t)} f(x(t), u(t)) \Phi(t, t_0) \tag{4.8}$$

s počáteční podmínkou

$$\Phi(t_0, t_0) = \frac{\partial x(t_0)}{\partial x(t_0)} = I_{n \times n}.$$

Obdobně potom citlivostní matice stavu $x(t)$ na vstupní vektor $u(t_0)$ je definována jako

$$\Gamma(t, t_0) = \frac{\partial x(t)}{\partial u(t_0)} \tag{4.9}$$

a časová změna citlivostní matice $\Gamma(t, t_0)$ je

$$\dot{\Gamma}(t, t_0) = \nabla_{x(t)} f(x(t), u(t)) \Gamma(t, t_0) + \nabla_{u(t)} f(x(t), u(t)) \tag{4.10}$$

s počáteční podmínkou

$$\Gamma(t_0, t_0) = \frac{\partial x(t_0)}{\partial u(t_0)} = 0_{n \times m}.$$

Tyto citlivostní matice použijeme pro sestavení linearizovaného predikčního modelu nelineárního systému. Predikční model použijeme k získání budoucí trajektorie stavu systému.

Vliv změny počátečního stavu $\delta x(t_{i-1})$ a řídící veličiny $\delta u(t_{i-1})$ v čase t_{i-1} na stav systému $x(t_i)$ v čase t_i můžeme popsat následujícím lineárním modelem

$$\delta x(t_1) = \Phi(t_1, t_0) \delta x(t_0) + \Gamma(t_1, t_0) \delta u(t_0) \tag{4.11}$$

$$\delta x(t_2) = \Phi(t_2, t_1) \delta x(t_1) + \Gamma(t_2, t_1) \delta u(t_1) \tag{4.12}$$

$$\vdots$$

$$\delta x(t_i) = \Phi(t_i, t_{i-1}) \delta x(t_{i-1}) + \Gamma(t_i, t_{i-1}) \delta u(t_{i-1}). \tag{4.13}$$

Pokud chceme vícekrokovou odezvu stavu systému na počáteční podmínku, která je potřeba např. při použití single-shooting optimalizace, dosadíme do rovnice (4.12) za $\delta x(t_1)$ předchozí rovnici a obdobně dosadíme i do ostatních rovnic, získáme tím linearizovaný predikční model stavu systému ve tvaru

$$\delta x(t_i|t_0) = \Phi_0^i \delta x(t_0) + \sum_{j=1}^{i} \Phi_j^i \Gamma(t_j, t_{j-1}) \delta u(t_{j-1}|t_0), \tag{4.14}$$

kde

$$\Phi_j^i = \begin{cases} \prod_{k=0}^{i-j-1} \Phi(t_{i-k}, t_{i-k-1}) & \text{if } i > j \\ I_{n\times n} & \text{if } i = j \end{cases} \tag{4.15}$$

Doposud jsme popisovali spojitý model systému. Nyní pro další potřeby návrhu MPC algoritmu budeme stavy a vstupy systému uvažovat v diskrétních okamžicích, proto upravíme značení stavu systému dle vztahu $x_k = x(t_0 + kT_s)$ a obdobně i vstupy systému $u_k = u(t_0 + kT_s)$, kde T_s je perioda vzorkování. Periodu vzorkování je možno jednoduše měnit změnou doby, po kterou budeme integrovat diferenciální rovnice citlivostních matic (4.8 a 4.10).

4.2 Formulace problému

Kvalita řízení prediktivního regulátoru velmi závisí na volbě ztrátové funkce, kterou MPC algoritmus minimalizuje. Ztrátová funkce na konečném horizontu využívající kvadratickou normu l_2 je

$$\min_{x,u} J(x,u) = \sum_{k=0}^{N-1} \left\{ \|Qx_k\|_2^2 + \|Ru_k\|_2^2 \right\} + \|Qx_N\|_2^2 \tag{4.16}$$

s omezeními

$$\begin{aligned} x_0 &\quad \text{dáno} \\ x_{k+1} &= F(x_k, u_k), \\ Du_k &\leqslant d, \\ u_{min} &\leqslant u_k \leqslant u_{max} \quad k = 0, 1, 2, \ldots, N-1, \end{aligned}$$

kde N je predikční horizont, matice $Q > 0$, $R > 0$ jsou váhové matice (pozitivně definitní), x a u označují posloupnost vektorů reprezentující stavy a vstupy soustavy, tak že:

$$x = (x_0, x_1, \ldots, x_N), \quad u = (u_0, u_1, \ldots, u_{N-1}). \tag{4.17}$$

4.3 Metody řešení

K řešení MPC algoritmu můžeme použít dvě metody optimalizace: single shooting a multiple shooting. Při single shooting metodě minimalizujeme ztrátovou funkci pouze přes vektor vstupu u. Stav systému je modelován přes celou dobu predikce a konečný stav v předchozím kroku je počátečním stavem pro následující krok. Naopak multiple shooting metoda modeluje stav systému v každém kroku odděleně a proto se konečný stav předchozího kroku nemusí rovnat počátečnímu stavu následujícího kroku. Multiple shooting metoda proto minimalizuje ztrátovou funkci nejen přes vektor vstupu u, ale talé přes vektor stavů x.

4.4 Řešení optimalizačního problému

Při návrhu ztrátové funkce je možno použít různé normy (např. l_1,l_2 nebo l_∞). My použijeme kvadratickou normu l_2, protože její použití vede na řešení kvadratického programu (QP) a také konečná řídící smyčka dává relativně dobré výsledky [9]. Tedy použitím l_2 normy můžeme k řešení daného problému použít SQP algoritmus (viz. kapitola 3.2). SQP přístupem vytvoříme podproblém, který má podobnou strukturu jako problém (4.16), s tím rozdílem, že rovnice modelu jsou linearizované (vyskytují se v podobě omezení typu rovnosti) a ztrátová funkce je nahražena kvadratickou funkcí, jejíž členy druhého řádu jsou aproximovány hesiánem z Lagrangeovy funkce pro (4.16). Podproblém má následující tvar:

$$\begin{aligned} \min_{\delta x, \delta u} \quad & \frac{1}{2}\delta x_N^T Q_N \delta x_N + q_N^T \delta x_N + \\ & + \sum_{k=0}^{N-1} \left\{ \frac{1}{2} \begin{bmatrix} \delta x_k \\ \delta u_k \end{bmatrix} \begin{bmatrix} Q_k & M_k \\ M_k & R_k \end{bmatrix} \begin{bmatrix} \delta x_k \\ \delta u_k \end{bmatrix} + \begin{bmatrix} q_k \\ r_k \end{bmatrix}^T \begin{bmatrix} \delta x_k \\ \delta u_k \end{bmatrix} \right\} \end{aligned} \tag{4.18}$$

s omezením

$$\begin{aligned} \delta x_0 &= 0, \\ \delta x_{k+1} &= \Phi_k \delta x_k + \Gamma_k \delta u_k, \\ D(u_k + \delta u_k) &\leqslant d, \quad k = 0, 1, 2, \ldots, N-1, \end{aligned}$$

Přidáním omezení do ztrátové funkce (4.16) vytvoříme Lagrangeovu funkci pro daný problém:

$$L(x, u, \lambda, \mu) = J(x, u) + \sum_{k=0}^{N-1} \lambda_k^T (F(x_k, u_k) - x_{k+1}) + \mu_k^T (D u_k - d) \tag{4.19}$$

$$= \sum_{k=0}^{N-1} (x_k^T Q x_k + u_k^T R u_k + \lambda_k^T (F(x_k, u_k) - x_{k+1}) + \mu_k^T (D u_k - d))$$

Dále můžeme Lagrangeovu funkci následujícím způsobem rozložit:

$$L(x, u, \lambda, \mu) = L_0(x_0, u_0, \mu_0) + \sum_{k=1}^{N-1} L_k(x_k, u_k, \lambda_{k-1}, \lambda_k, \mu_k) + L_N(x_N, \lambda_{N-1}), \quad (4.20)$$

kde

$$\begin{aligned} L_0(x_0, u_0, \mu_0) &= J(x_0, u_0) + \lambda_0^T (F(x_0, u_0) + \mu_0^T (D u_0 - d) \quad (4.21) \\ L_k(x_k, u_k, \lambda_{k-1}, \lambda_k, \mu_k) &= J(x_k, u_k) + \lambda_k^T (F(x_k, u_k) - \lambda_{k-1}^T x_k) + \mu_k^T (D u_k - d) \\ L_N(x_N, \lambda_{N-1}) &= -\lambda_{N-1}^T x_N. \end{aligned}$$

Poznamenejme, že každý člen x_k a u_k se vyskytuje pouze v L_k a každá L_k závisí nanejvýše na dvou složkách vektoru Lagrangeových koeficientů.

Nyní se zaměříme na různé možnosti volby hesiánu v kvadratickém modelu, k čemuž použijeme rovnice uvedené v této podkapitole.

4.4.1 Přesný hesián

První možnost je použití přesného hesiánu z Lagrangeovy funkce, který má následující blokovou strukturu

$$W = \nabla^2_{x,u} L = \begin{bmatrix} Q_0 & M_0 & & & & & & \\ M_0^T & R_0 & & & & & & \\ & & Q_1 & M_1 & & & & \\ & & M_1^T & R_1 & & & & \\ & & & & \ddots & & & \\ & & & & & Q_{N-1} & M_{N-1} & \\ & & & & & M_{N-1}^T & R_{N-1} & \\ & & & & & & & Q_N \end{bmatrix}, \quad (4.22)$$

kde se matice pro jednotlivé kroky vypočítají jako druhé derivace Lagrangeovy funkce

$$Q_k = \frac{\partial^2 L_k}{\partial x_k^2}, \quad R_k = \frac{\partial^2 L_k}{\partial u_k^2}, \quad M_k = \frac{\partial^2 L_k}{\partial u_k \partial x_k}. \quad (4.23)$$

Poznamenejme, že hesián z Lagrangiánu má stejnou strukturu jako ztrátová funkce (4.16) a jako hesián v lineárním MPC problému, protože se zde nevyskytují vzájemné vazby mezi budoucími a minulými stavy a vstupy, které jsou pouze v omezujících podmínkách typu rovnosti.

Použití přesného hesiánu v SQP podproblému má výhodu rychlé lokální konvergence k optimálnímu řešení. Avšak k jeho sestavení potřebujeme získat druhé

derivace modelu F, jejichž výpočet může být obtížný nebo časově náročný. A navíc jednotlivé bloky diagonálních matic nemusí být vždy pozitivně definitní.

4.4.2 Gauss-Newtonova aproximace

Další možnost je nepoužívat přesný hesián, ale pouze jeho aproximaci. Jednoduchá aproximace, která je často efektivní, je zanedbání příspěvku z F a použít přímo

$$Q_k = Q, \quad R_k = R, \quad M_k = M. \tag{4.24}$$

Tato aproximace se nazývá Gauss-Newtonova. Celý hesián se poté poskládá obdobně jako u přesného hesiánu dle (4.22). V případě, že je ztrátová funkce kvadratická a konvexní (což je obvykle splněno), je získaný hesián konstantní a pozitivně definitní. Pokud SQP podproblém s aproximovaným hesiánem startujeme z blízkého okolí optima, pak se získané řešení blíží k řešní získané použitím přesného hesiánu.

4.4.3 BFGS aproximace

Jiná možnost, jak aproximovat hesián kvadratické funkce, je použít BFGS aproximaci. Její princip je popsán v kapitole 3.3.1. Při jednotlivich optimalizacích aktualizujeme celý hesián, tím však hesián ztratí symetrickou blokovou strukturu a řešení kvadratického programu se stane složitější. Abychom zachovali symetrickou blokovou strukturu hesiánu, můžeme aktualizovat postupně jednotlivé čtvercové bloky matic, složené z váhových matic Q, R a M, příslušející daným krokům viz. (4.22). Použitím tlumené BFGS aktualizace popsané v kapitole 3.3.2 zajistíme, že získaný hesián bude pozitivně definitní.

4.4.4 Exact penalty funkce

Další způsob aproximace hesiánu kvadratické funkce je použití Exact penalty funkce dle vztahu (3.45). Přidáním penalizační funkce opět zajistíme pozitivní definitnost hesiánu a urychlíme konvergenci řešení k optimu.

4.5 Algoritmus prediktivního regulátoru

Nyní uvedeme algoritmus prediktivního regulátoru využívající sekvenční kvadratický program se ztrátovou funkcí dle (4.19) a predikční model sestevený z citlivostních matic.

Algoritmus 6 (Algoritmus prediktivního regulátoru)

Zvol váhové matice Q, R a M, dobu predikce T_p, omezení vstupu (u_{min}, u_{max}), omezení výstupu (y_{min}, y_{max}) a referenční průběh výstupu;
Sestav počáteční vektor řízení u_0;
for $k = 0, 1, 2, \ldots$ **do**
  Vyber vektor reference na dobu T_p dopředu;
  for $l = 1, \ldots,$ počet optimalizací **do**
    Odsimuluj odezvu systému a citlivostní matice na vstupní vektor u_k;
    Sestav ztrátovou funkci dle (4.19) s přesným hesiánem (4.22);
    Sestav podle (4.19) matice omezení;
    Vyřeš kvadratický program s danými omezeními a získej δu a δx;
    Uprav vektor řízení $u_k = u_k + \delta u$;
  end for
  Aplikuj první krok z vektoru řízení a získej nový stav systému x_{k+1};
  Rotuj vektor řízení o jeden krok dopředu;
end for

Použitím rotace vektoru řízení o jeden krok dopředu získáme velmi přesnou aproximaci počátečního řešení pro další dobu periody T_s, pokud do systému nevstupují vnější poruchy.

4.6 Shrnutí

V této kapitole jsme provedli návrh nelineárního prediktivního regulátoru. Ukázali jsme vhodné řešení optimalizační úlohy pomocí SQP a několik možných voleb hesiánu pro kvadratický model. Nakonec jsme uvedli algoritmus prediktivního regulátoru použitelný k impementaci v Matlabu.

Kapitola 5

Experimentální ověření na modelu odparky

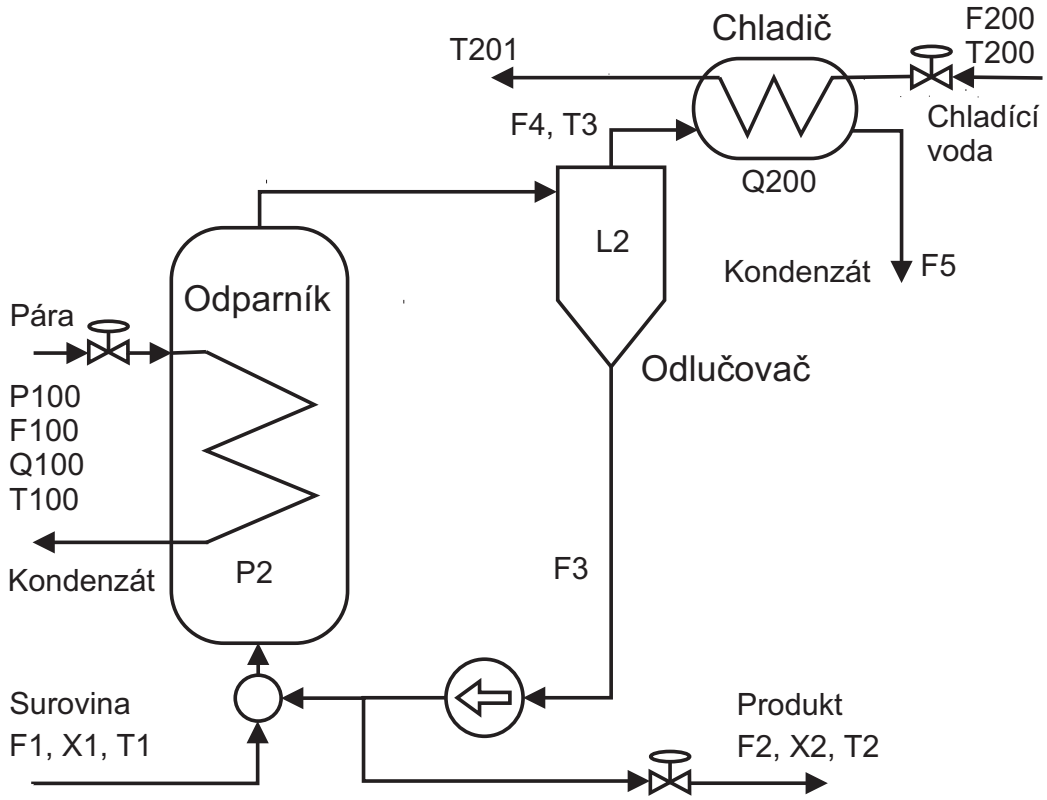
V této kapitole provedeme ověření prediktivního řízení na modelu odparky. Odparka je průmyslové zařízení používané k zahušťování příchozí suroviny odpařením obsaženého rozpouštědla. Typické použití je v cukrovarnictví, kde se používá k zahušťování cukerného roztoku odpařením vody. V první části této kapitoly nejprve popíšeme matematický model odparky, pro který následně navrhneme prediktivní regulátor.

5.1 Model odparky

Model odparky [4] je znázorněn na obr. 5.1 a lze jej pro zjednodušení rozdělit na tří hlavní části: odparník, odlučovač páry a chladič. Odparník je nádoba, ve které se příchozí surovina za pomoci páry zahřívá. Pára je přiváděna do výměníku z externího zdroje a její množství lze regulovat ventilem. Ohřátá surovina je společně s jejími výpary odváděna do odlučovače, kde se od kapaliny odloučí pára. Tato pára dále přechází do chladiče, kde je ochlazena chladící vodou a opouští proces jako kondenzát. Zahuštěná surovina je odčerpávána z odlučovače. Částečně je na výstupu odebírána jako výsledný produkt a větší část jí je opět navrácena do odparníku.

Dynamiku systému [4] lze popsat třemi diferenciálními rovnicemi. První rovnice popisuje závislost výšky hladiny $L2(t)$ v odlučovači na množství: přitékající suroviny $F1$, odebíraného produktu $F2(t)$ a odpařené vody $F4(t)$

$$\frac{dL2(t)}{dt} = \frac{1}{\rho A}(F1(t) - F2(t) - F4(t)), \tag{5.1}$$



Obrázek 5.1: Schéma modelu odparky

kde ρ je měrná hustota kapaliny a A je plocha hladiny v odlučovači. Můžeme uvažovat hodnotu konstant $\rho A = 20$ kg/m.

Zbylé dvě diferenciální rovnice popisují dění v odparníku:

$$\frac{dX2(t)}{dt} = \frac{1}{M}(F1(t) * X1(t) - F2(t) * X2(t)) \tag{5.2}$$

$$\frac{dP2(t)}{dt} = \frac{1}{C}(F4(t) - F5(t)) \tag{5.3}$$

$$T2(t) = 0,5616P2(t) + 0,3126X2(t) + 48,43$$

$$T3(t) = 0,507P2(t) + 55,0$$

$$F4(t) = \frac{Q100(t) - F1(t) * C_p(T2(t) - T1(t))}{\lambda},$$

kde první z nich vyjadřuje změnu koncentrace výstupního produktu $X2(t)$. Zjednodušeně lze říci, že množství užitečné látky přicházející na vstupu se musí rovnat množství užitečné látky odebírané na výstupu. A druhá diferenciální rovnice udává změnu tlaku v nádobě odparky, v závislosti na množství odpařené látky $F4(t)$ odcházející z odlučovače a na množství zkondenzovaných par v chladiči $F5(t)$. Dalšími třemi algebraickými rovnicemi vypočteme hodnotu teploty produktu $T2(t)$,

teploty $T3(t)$ a průtoku odpařené látky $F4$. Konstanty v předchozích rovnicích mají tyto hodnoty:

$$\begin{aligned} M &= 20 \text{ kg} \\ C &= 4 \text{ kg/kPa} \\ C_p &= 0,07 \text{ kW/K(kg/min)} \\ \lambda &= 38,5 \text{ kW/(kg/min)} \end{aligned}$$

Parní výměník tepla, který v odparníku ohřívá zahušťovanou látku popisují tyto rovnice

$$\begin{aligned} T100(t) &= 0,1538P100(t) + 90,0 \\ Q100(t) &= 0,16(F1(t) + F3(t))(T100(t) - T2(t)) \\ F100(t) &= \frac{Q100(t)}{\lambda_s}, \end{aligned}$$

kde $T100(t)$ je teplota páry vstupující do odparníku,$P100(t)$ je tlak páry, $Q100(t)$ je množství tepla, které pára předá zahušťované látce, $F100(t)$ je hmotnostní průtok páry a konstanta λ_s má hodnotu $\lambda_s = 36,6$ kW/(kg/min).

Poslední částí odparky je chladič, který za pomoci chladící vody zchladí odpařenou páru na kondenzát. Dle následujících rovnic se vypočítá teplo odebrané páře $Q200(t)$, teplota chladící vody na výstupu chladiče $T201(t)$ a hmotnostní průtok zkondenzovaných par $F5(t)$:

$$\begin{aligned} Q200(t) &= \frac{UA2(T3(t) - T200(t))}{1 + \frac{UA2}{2*C_p*F200(t)}} \\ T201(t) &= T200(t) + \frac{Q200(t)}{F200(t) * C_p} \\ F5(t) &= \frac{Q200(t)}{\lambda}, \end{aligned}$$

kde konstanta $UA2 = 6,48$ kW/K.

Z uvedených rovnic popisujících chování odparky sestavíme matematický model systému. Za vstupy systému zvolíme: hmotnostní průtok suroviny $F1(t)$, průtok chladící kapaliny $F200(t)$ a tlak páry sloužící k ohřevu $P100(t)$. Hmotnostní průtok produktu $F2(t)$ budeme modelovat jako předem známou poruchovou veličinu. Stavy systému jsou: výška hladiny v odlučovači $L2(t)$, koncentrace výstupního produktu $X2(t)$ a tlak v odparníku $P2(t)$. Stavy systému jsou současně i jeho měřené výstupy, které je nutno řídit. Nejdůležitější proměnná, kterou je třeba řídit, je koncentrace produktu $X2(t)$. Přesné dodržení požadované koncentrace produktu a její minimální

výkyvy maximalizují ziskovost odparky, protože se tím sníží výroba nekvalitního produktu (který je buď neprodejný nebo se prodává za nižší cenu). Další řízenou proměnnou je tlak v odparníku $P2(t)$, který z důvodu bezpečnosti nesmí přesáhnout povolenou mez, aby nedošlo k poškození nádoby nebo jiných částí odparky. A třetí řízenou proměnnou je výška hladiny v odlučovači $L2(t)$, která nesmí přetéci do chladiče a naopak nesmí klesnout na nulu, aby nedošlo k poškození oběhového čerpadla.

Zvolíme-li vektor stavu $x = (L2, X2, P2)$, vektor vstupu $u = (F1, P100, F200)$ a dosadíme-li do diferenciálních rovnic (5.1-5.3) zbylé rovnice popisující jednotlivé části odparky, získáme rovnice ve tvaru (4.1). Získanou vektorovou funkci $f(x, u)$ zderivujeme nejprve podle vektoru x a poté podle vektoru u. Obdržíme tím dvě Jakobiho matice

$$\nabla_x f(x,u) = \begin{bmatrix} 0 & \frac{0{,}0719u_1+2{,}5}{\lambda\rho A} & \frac{0{,}1292u_1+4{,}495}{\lambda\rho A} \\ 0 & \frac{-F2}{M} & 0 \\ 0 & \frac{-0{,}0719u_1-2{,}5}{\lambda C} & \frac{-0{,}1292u_1-4{,}495}{\lambda C} - \frac{0{,}4855u_3}{\lambda C(0{,}14u_3+6{,}48)} \end{bmatrix} \tag{5.4}$$

$$\nabla_u f(x,u) = \begin{bmatrix} \frac{\lambda-0{,}0246u_2+0{,}0719x_2+0{,}1292x_3-6{,}0611}{\lambda\rho A} & \frac{-0{,}0246u_1-1{,}23}{\lambda\rho A} & 0 \\ \frac{5}{M} & 0 & 0 \\ \frac{0{,}0246u_2-0{,}0719x_2-0{,}1292x_3+6{,}0611}{\lambda C} & \frac{0{,}0246u_1+1{,}23}{\lambda C} & \frac{\partial f_3}{\partial u_3} \end{bmatrix} \tag{5.5}$$

$$\frac{\partial f_3}{\partial u_3} = \frac{-3,3208x_3 - 196,5}{\lambda C(0,14u_3 + 6,84)^2},$$

které použijeme k výpočtu citlivostních matic stavu Φ a vstupu Γ. Citlivostní matice poté použijeme k vytvoření predikčního modelu systému. Odezva systému na vstupní vektor, současně s výpočtem citlivostních matic, bude odsimulována v Simulinku pomocí S-funkce.

V tabulce 5.1 jsou uvedeny všechny proměnné použité v modelu [4] a také jejich hodnota v ustáleném stavu pro vzorkovací periodu $T_s = 1$ min.

Proměnná	Popis	Hodnota	Jednotky
F1	Průtok suroviny	10,0	kg/min
F2	Průtok produktu	2,0	kg/min
F3	Cirkulační průtok	50,0	kg/min
F4	Průtok výparů	8,0	kg/min
F5	Průtok kondenzátu	8,0	kg/min
X1	Koncentrace suroviny	5,0	%
X2	Koncentrace produktu	25,0	%
T1	Teplota suroviny	40,0	°C
T2	Teplota produktu	84,6	°C
T3	Teplota výparů	80,6	°C
L2	Výška hladiny	1,0	m
P2	Provozní tlak	50,5	kPa
F100	Průtok páry	9,3	kg/min
T100	Teplota suroviny	119,9	°C
P100	Tlak páry	194,7	kPa
Q100	Výkon ohřevu	339,0	kW
F200	Průtok chladící vody	9,3	kg/min
T200	Teplota chladící vody	25,0	°C
T201	Teplota chladící vody	46,1	°C
Q200	Výkon chlazení	307,9	kW

Tabulka 5.1: Procesní proměnné a jejich nominální hodnoty

5.2 Prediktivní regulátor pro model odparky

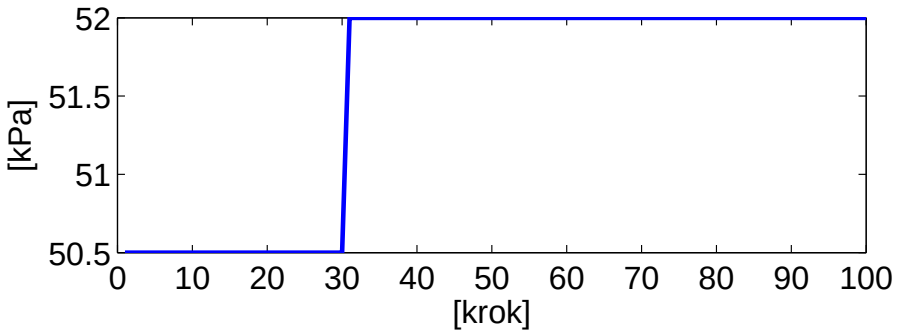
K vytvoření predikivního regulátoru použijeme algoritmus 6. Zvolíme následující váhové matice pro problém (4.16)

$$Q = \begin{bmatrix} 0,025 & 0 & 0 \\ 0 & 0,1 & 0 \\ 0 & 0 & 0,01 \end{bmatrix}, R = \begin{bmatrix} 0.05 & 0 & 0 \\ 0 & 0.0001 & 0 \\ 0 & 0 & 0.0001 \end{bmatrix}, M = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

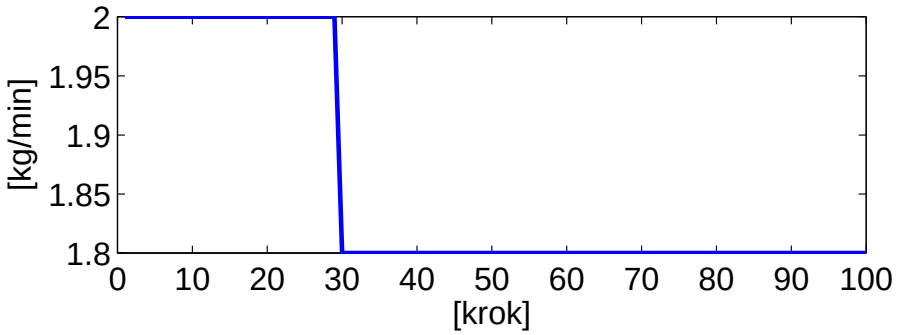
dobu predikce $T_p = 100$ min, vektor vstupů a výstupů

$$\begin{aligned} u &= (F1, P100, F200), \\ y &= (L2, X2, P2), \end{aligned}$$

referenční průběhy výstupů $L2(t) = 1$, $X2(t) = 25$ a $P2(t)$ dle obr. (5.2). Pro počáteční vektor vstupů u_0 zvolíme ustálené hodnoty vstupů dle tabulky 5.1. Průběh poruchové veličiny $F2(t)$ vstupující do soustavy je na obr. (5.3). Na tomto příkladě ukážeme vliv volby hesiánu ztrátové funkce na kvalitu a rychlost regulace.



Obrázek 5.2: Průběh reference tlaku $P2$



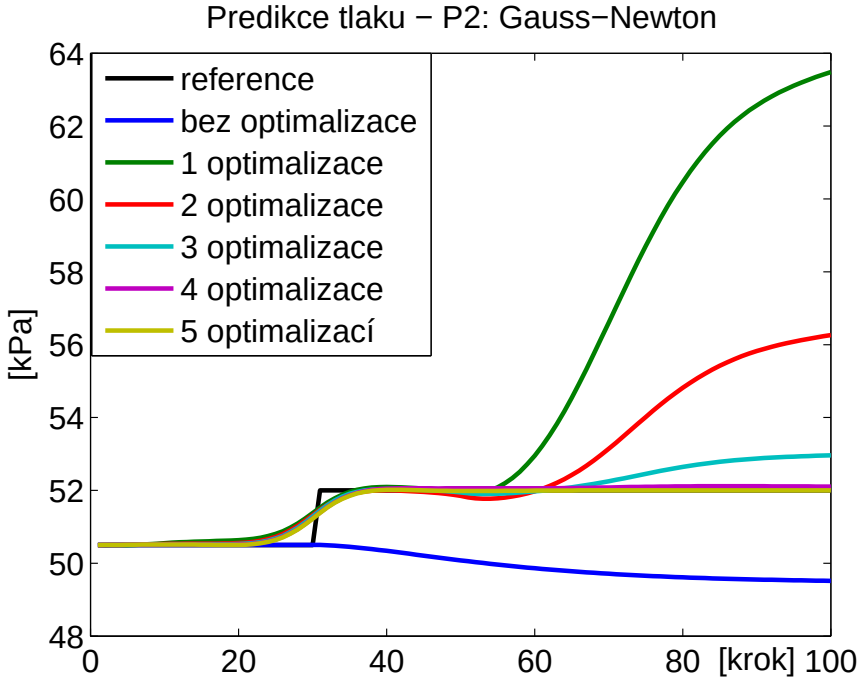
Obrázek 5.3: Průběh poruchové veličiny $F2$

5.2.1 Přesný hesián

K získání přesného hesiánu je třeba znát funkci $F(x, u)$, která popisuje omezení daného problému. V naší úloze však máme k dispozici pouze její derivaci, proto bychom museli pro její získání analyticky vyřešit časový integrál. Poté bychom takto získanou funkci omezení $F(x, u)$ parciálně zderivovali dle vektoru stavů x a vektoru vstupů u. Toto je pro náš příklad příliš obtížné a proto nebudeme přesný hesián používat.

5.2.2 Gauss-Newtonova aproximace

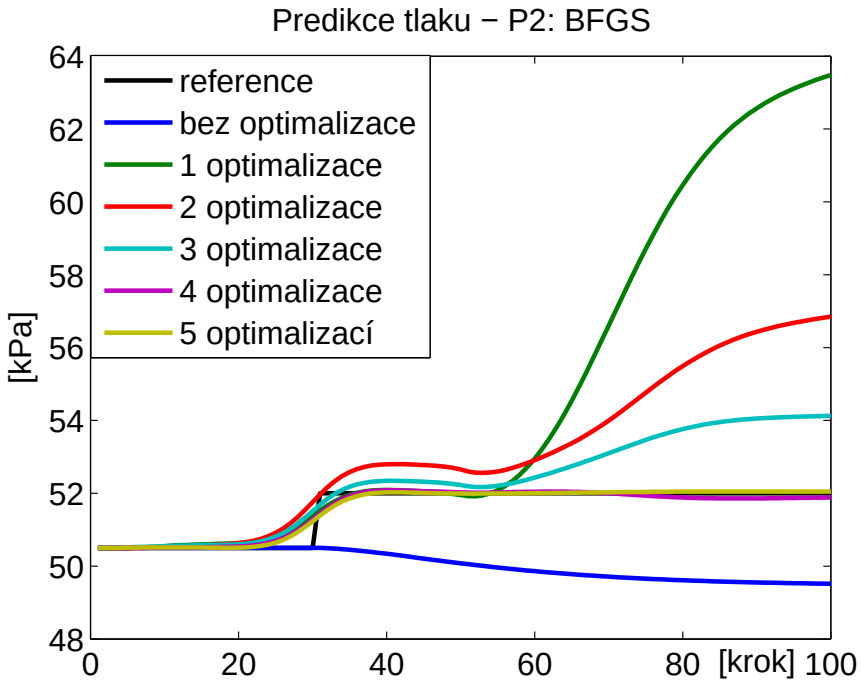
S použitím Gauss-Newtonovy aproximace hesiánu (4.24) provedeme jeden krok algoritmu 6, k jehož řešení použijeme postupně pět optimalizací SQP. Výsledek řešení je zobrazen na obr. 5.4, kde je vidět predikovaný průběh tlaku $P2$ po jednotlivých optimalizacích. Na obrázku je vidět postupné přibližování výstupu požadované referenci, kdy po čtyřech optimalizacích dosáhneme ustáleného řešení, které se již dále nemění.



Obrázek 5.4: Predikce tlaku P2: Gauss-Newton

5.2.3 BFGS aproximace

Poté provedeme stejný krok algoritmu jako v předchozím příkladě, ale použijeme BFGS aproximaci hesiánu. Počáteční podmínky jsou totožné. K odhadu počáteční hodnoty hesiánu použijeme Gauss-Newtonovu aproximaci[1]. Výsledek řešení je ukázán na obr. 5.5. Zde je opět vidět snaha algoritmu o přiblížení se referenčnímu signálu. Tentokrát potřebuje k nalezení optimálního řešení pět optimalizací.



Obrázek 5.5: Predikce tlaku P2: BFGS

5.2.4 Exact penalty funkce

Nakonec provedeme tentýž krok algoritmu s použitím aproximace hesiánu využívající Exact penalty funkce. Jelikož nemáme k dispozici přesný hesián, použijeme místo něj ve vztahu (3.45) jeho Gauss-Newtonovu aproximaci. Jelikož je v našem příkladě zajištěna pozitivní definitnost Gauss-Newtonovy aproximace hesiánu a je použita multiple shooting metoda optimalizace, obdržíme stejný výsledek jako u samotné Gauss-Newtonovy aproximace viz. obr. 5.4.

[1]Proto jsou průběhy po první optimalizaci u obou metod totožné.

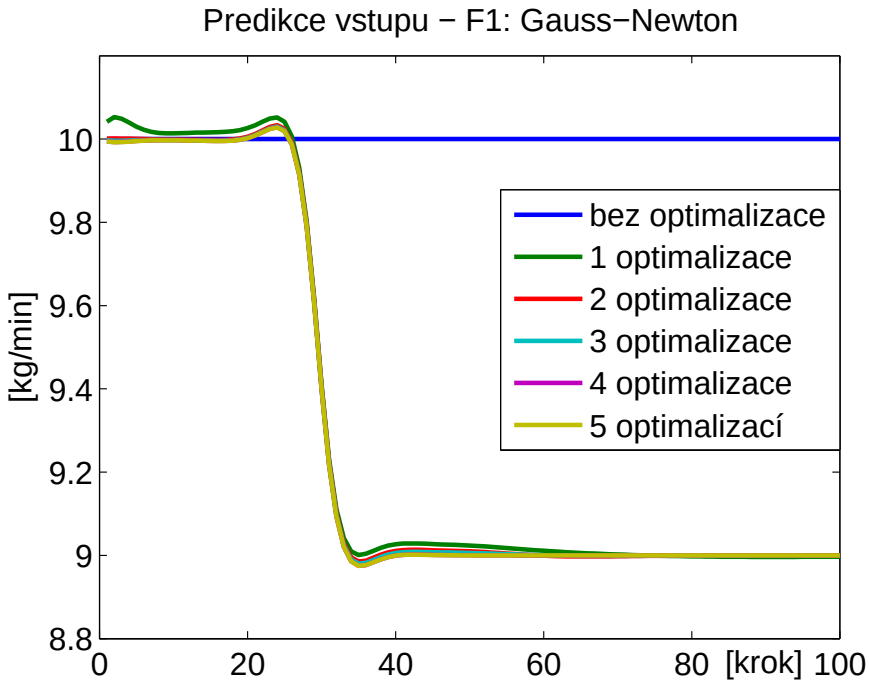
Rozdíly mezi použitými aproximacemi hesiánu nejsou příliš veliké, proto průběhy vstupů a zbylých výstupů ukážeme pouze pro Gauss-Newtonovu metodu. Na obrázcích 5.6, 5.7 a 5.8 jsou zobrazeny průběhy vstupů soustavy. Predikce zbylých dvou výstupů $L2$ a $X2$ jsou uvedeny na obr. 5.9 a 5.10. Také na těchto průbězích jsou vidět změny po jednotlivých optimalizacích a konvergence k optimálnímu řešení.

V tabulce 5.2 jsou uvedeny časy (v sekundách) jednotlivých optimalizací pro použité metody. Z tabulky vyplývá větší časová náročnost BFGS metody a Exact penalty funkce způsobená delším řešením kvadratického programu, jelikož hesián po první aktualizaci ztrácí symetrickou blokovou strukturu. BFGS metoda navíc v každém kroku provádí aktualizaci hesiánu.

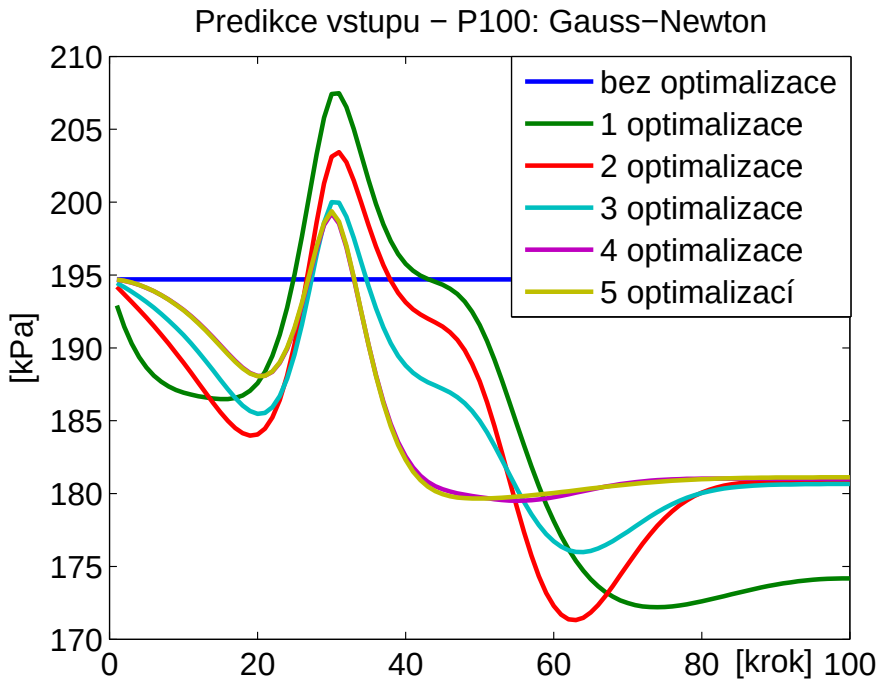
Optimalizace	Gauss-Newton	BFGS	Exact penalty
1	121,5	125,3	130.2
2	71,0	81,9	78.9
3	62,7	74,9	72.0
4	24,7	64,6	32.4
5	24,8	32,7	32.3

Tabulka 5.2: Porovnání časů jednotlivých optimalizací (v sekundách)

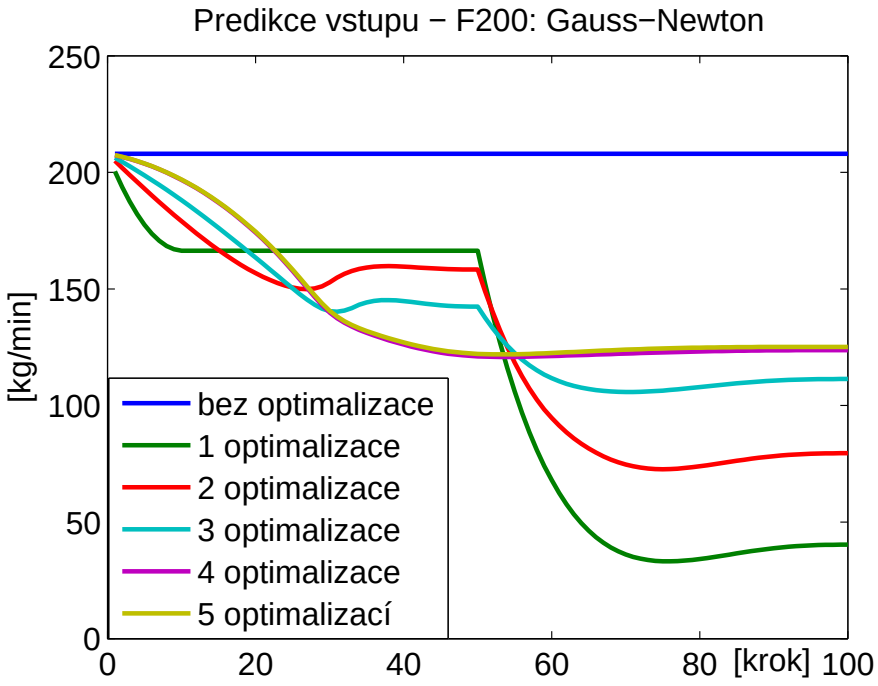
V tomto experimentu jsme provedli pouze jeden krok MPC algoritmu a zaměřili jsme se jen na vliv volby hesiánu ztrátové funkce. Nyní provedeme experiment při němž necháme běžet MPC algoritmus více kroků a do návrhu zahrneme také omezení na vstupní a výstupní proměnné. K aproximaci hesiánu ztrátové funkce v následujícím příkladě použijeme Gauss-Newtonovu metodu, která v tomto příkladě dosáhla nejlepších výsledků.



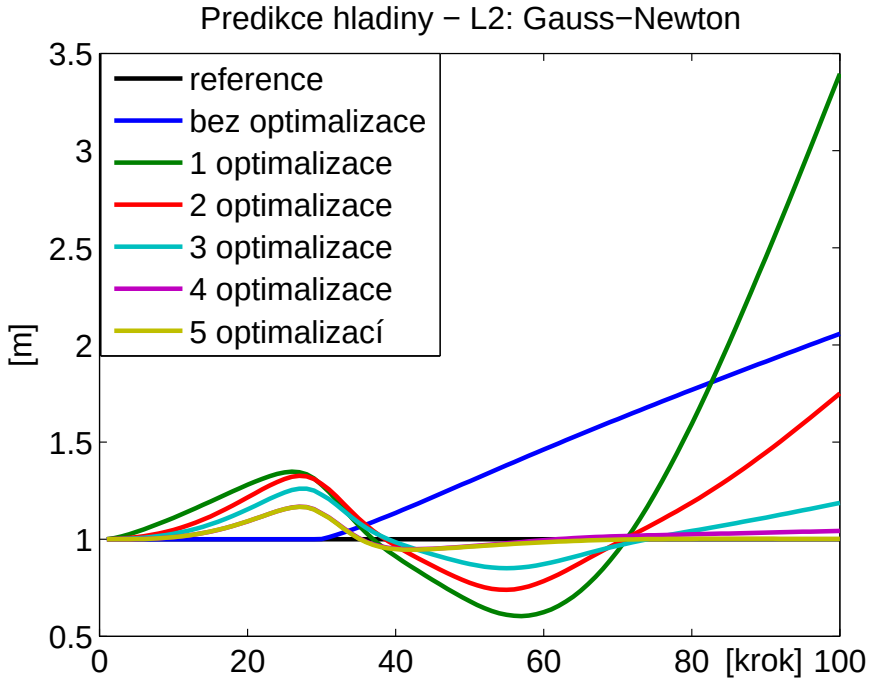
Obrázek 5.6: Predikce vstupu F1: Gauss-Newton



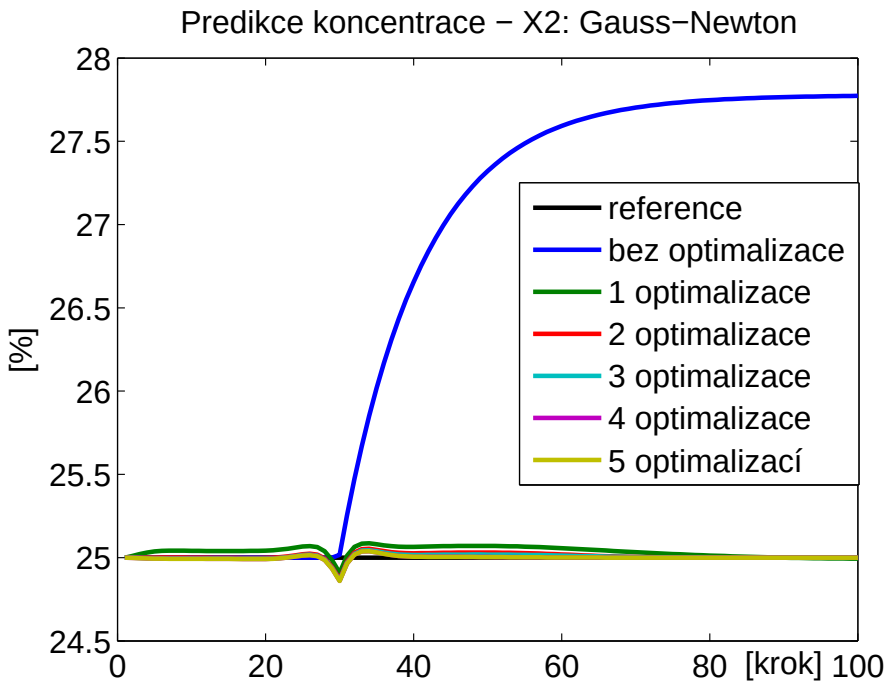
Obrázek 5.7: Predikce vstupu P100: Gauss-Newton



Obrázek 5.8: Predikce vstupu F200: Gauss-Newton



Obrázek 5.9: Predikce hladiny L2: Gauss-Newton



Obrázek 5.10: Predikce koncentrace X2: Gauss-Newton

5.3 Prediktivní regulátor s omezeními

Pro tento příklad zvolíme následující váhové matice

$$Q = \begin{bmatrix} 500 & 0 & 0 \\ 0 & 10000 & 0 \\ 0 & 0 & 100 \end{bmatrix}, R = \begin{bmatrix} 500 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, M = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

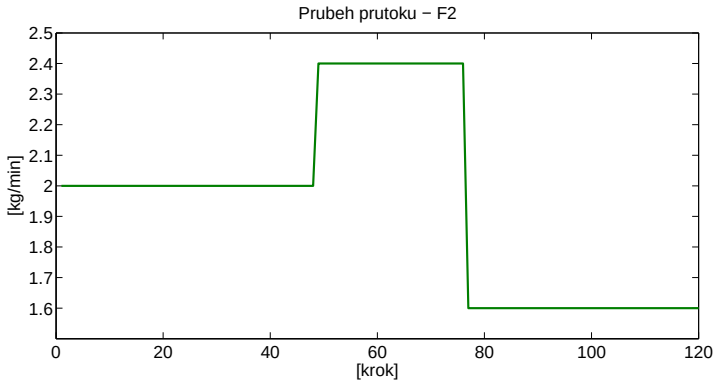
dobu predikce $T_p = 40$ min, vektor vstupů a výstupů

$$\begin{aligned} u &= (F1, P100, F200), \\ y &= (L2, X2, P2). \end{aligned}$$

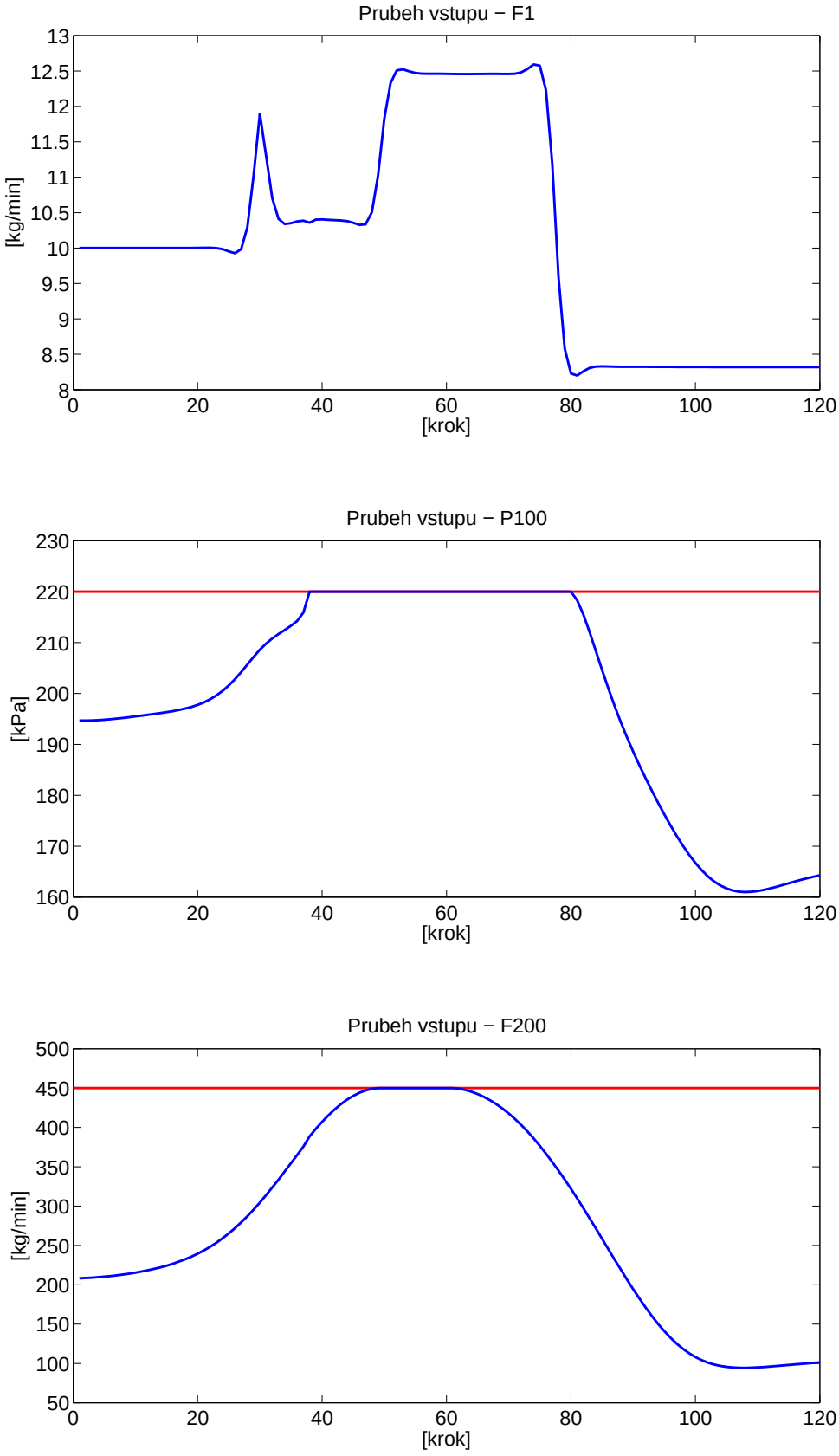
Pro počáteční vektor vstupů u_0 opět zvolíme ustálené hodnoty dle tab. 5.1. Průběh poruchové veličiny $F2(t)$ vstupující do soustavy je na obr. (5.11). V tomto příkladě necháme MPC algoritmus běžet po dobu 120 kroků a v každém kroku provést pět optimalizací ztrátové funkce. Při optimalizacích bude muset prediktivní regulátor dodržet omezení na velikosti vstupních a výstupních proměnných dle tab. 5.3.

Proměnná	Min	Max	Jednotky
F1	0	20	kg/min
P100	0	220	kPa
F200	0	450	kg/min
L2	0,5	2,0	m
X2	24	28	%
P2	0	51,5	kPa

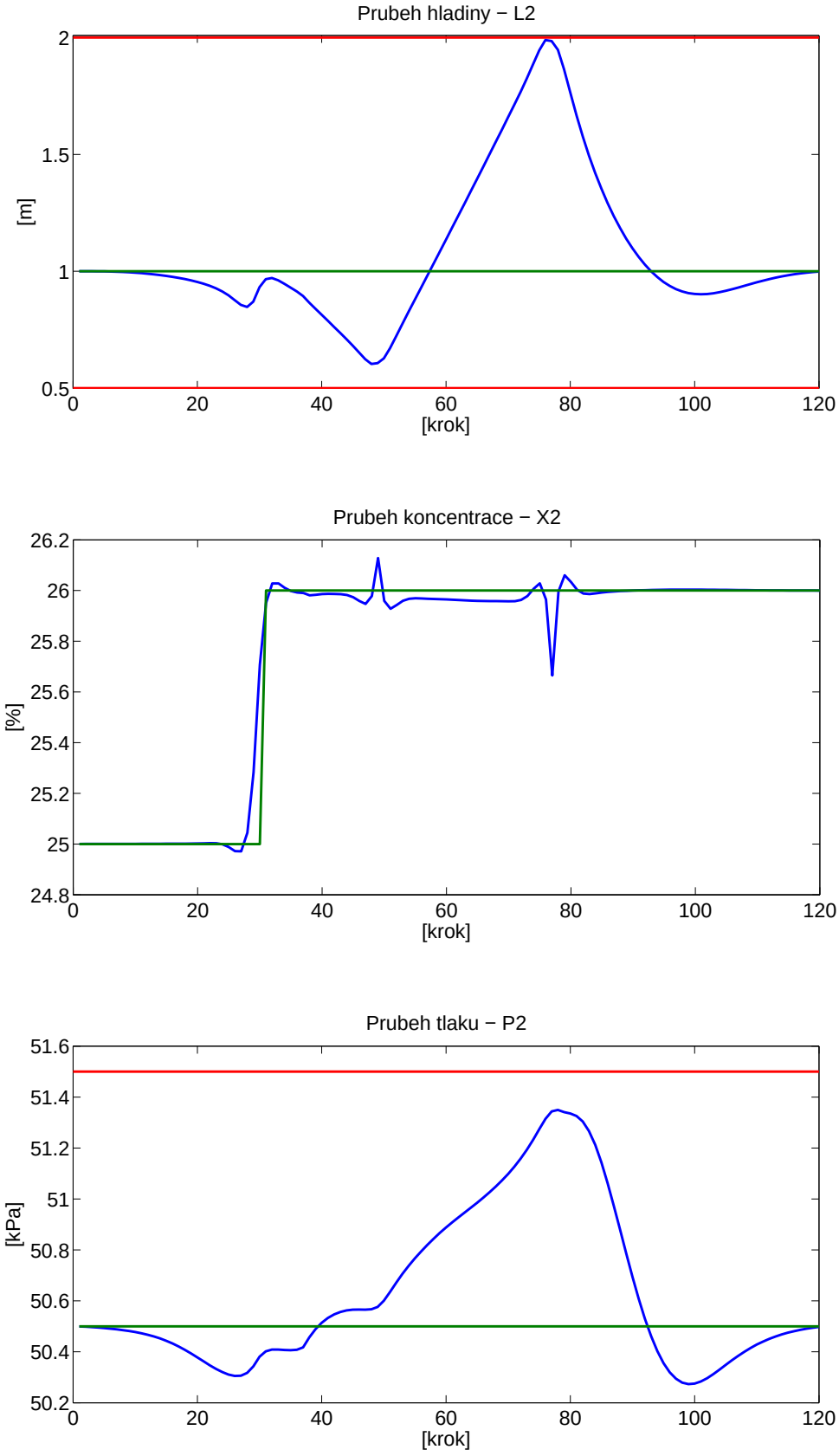
Tabulka 5.3: Omezení vstupních a výstupních proměnných



Obrázek 5.11: Průběh poruchové veličiny $F2$



Obrázek 5.12: Průběh vstupů: F1, P100, F200



Obrázek 5.13: Průběh výstupů: L2, X2, P2

Na obrázku 5.12 jsou zobrazeny průběhy vstupních veličin získané prediktivním regulátorem. Jak je z obrázků vidět, prediktivní regulátor dodržel předepsaná omezení na velikost vstupů, které jsou na obrázcích zobrazeny červeně. Obdobně jako vstupy jsou na obr. 5.13 zobrazeny optimální průběhy výstupních veličin. Zde jsou opět červeně zobrazeny omezení a zeleně průběhy referencí. Velké odchylky průběhu koncentrace od referenčního signálu jsou způsobeny přísným omezením vstupů tlaku páry $P100$ a průtoku chladící kapaliny $F200$, čímž prediktivní regulátor neměl dostatečný prostor k přesnější regulaci.

5.4 Shrnutí

V této kapitole jsme ověřili správnou funkci prediktivního regulátoru, jehož algoritmus 6 je uvedený v předchozí kapitole. Ověření jsme provedli na modelu odparky. V prvním příkladě jsme na tomto modelu porovnali různé volby hesiánu ztrátové funkce prediktivního regulátoru. Všemi třemi použitými metodami aproximace hesiánu jsme obdrželi obdobné výsledky, čož je způsobeno jednak použitím Gauss-Newtonovy aproximace jako počátečního odhadu hesiánu pro BFGS metodu a Exact penalty metodu, ale také použitím modelu se slabou nelinearitou, která se nejvíce projevuje na výstupu tlaku $P2$. Přestože v dosažených výsledcích nebyly příliš veliké rozdíly, tak pro tento příklad vychází nejlépe použití Gauss-Newtonovy aproximace. Tato aproximace dosáhla nejen vyšší rychlosti konvergence řešení, ale také výpočet optimalizace trval kratší dobu než pro BFGS aproximaci a Exact penalty funkci. V druhém příkladě jsme do návrhu prediktivního regulátoru zahrnuli omezení na vstupní a výstupní veličiny. Také v tomto příkladě se nám podařilo ověřit správnou funkci regulátoru a dodržení předepsaných omezení.

Kapitola 6

Závěr

V této práci jsme se zabývali použitelnými algoritmy pro návrh nelineárního MPC regulátoru. Nejprve jsme se zaměřili na různé způsoby řešení nelineární optimalizační úlohy, jež je třeba řešit při hledání optimální vstupní trajektorie v prediktivním řízení. Odvodili jsme úlohu sekvenčního kvadratického programování a ukázali jsme jeho ekvivalenci s Newton-Lagrangeovu metodou. Dále jsme porovnávali vliv volby hesiánu v kvadratickém modelu ztrátové funkce optimalizační úlohy. Na zvoleném příkladě jsme porovnali přesnost a rychlost konvergence SQP algoritmu při použití: přesného hesiánu, aproximovaného hesiánu BFGS metodou a jednotkového hesiánu (tzv. metoda nejrychlejšího sestupu). Algoritmus nejrychleji konvergoval při použití přesného hesiánu, kdy kvadratický model nejpřeněji popisoval nelineární optimalizační úlohu, naopak nejpomalejší byl při použití jednotkového hesiánu. Nevýhoda použití přesného hesiánu je, že není vždy zajištěna jeho pozitivní definitnost a SQP algorimus může z toho důvodu selhat, jak je ukázáno na našem příkladě. Naopak metoda nejrychlejšího sestupu a metoda BFGS mají vždy zajištěnu pozitivní definitnost hesiánu. Na stejném příkladě jsme ukázali vliv použití metody jednorozměrového hledání (line search), která nám pomůže zrychlit konvergenci algoritmu, pokud je počáteční bod více vzdálen od optima.

V další kapitole jsme formulovali nelineární prediktivní regulátor využívající tzv. multiple shooting metodu optimalizace. K řešení optimalizační úlohy jsme použili již dříve odvozené sekvenční kvadratické programování.

Nakonec jsme provedli experimentální ověření formulovaného prediktivního regulátoru na dodaném modelu odparky. Nejprve jsme na modelu odparky porovnali jednotlivé volby hesiánu ztrátové funkce prediktivního regulátoru. Jelikož získání přesného hesiánu pro tuto úlohu by bylo obtížné, zaměřili jsme se pouze na jeho různé metody aproximace. Použili jsme následující metody aproximace: Gauss-Newtonovu,

BFGS a Exact penalty funkce. Nejlepších výsledků jsme dosáhli při použití Gauss-Newtonovy metody. Nejen že ke konvergenci stačilo méně kroků, ale i časová náročnost výpočtu v jednotlivých krocích byla u této metody menší než u dvou ostatních. Při druhém experimentu jsme ověřili správnou funkci MPC algoritmu s přidanými omezeními na vstupní a výstupní veličiny.

Literatura

[1] R. Findeisen and F. Allgöwer. An introduction to nolinear model predictive control. *21st Benelux Meeting on Systems and Control*, 2002.

[2] R. Fletcher. *Practical Methods of Optimalization (2nd ed.)*. Wiley, 2003.

[3] J.M. Maciejowski. *Predictive Control with Constraints*. Prentice Hall, 2001.

[4] J.M. Maciejowski. Evaporator model. *Postgraduate course at The Univerzity of British Columbia*, 2005.

[5] J. Nocedal and S.J. Wright. *Numerical optimalization*. Springer, 1999.

[6] E. Polak. *Optimalization: Algorithms and Consistent Approximations*. Springer, 1997.

[7] J.A. Rossiter. *Model-Based Predictive Control*. CRC Press, 2003.

[8] J. Štecha. *Optimální rozhodování a řízení*. ČVUT, 1999. skripta.

[9] M.J. Tenny; S.J. Wright and J.B. Rawlings. Nolinear model predictive control via feasibility-pertrubed sequential quadratic programming. *Computational Optimization and Applications*, 2004.

Příloha

Přílohou této práce je CD-ROM disk s následujícím adresářovou strukturou a obsahem:

- V adresáři *Dokument* je zdrojový kód této práce ve formátu LaTeX.
- V adresáři *M-files* jsou soubory se zdrojovým kódem pro program Matlab 7, které byly použity k simulaci navržených algoritmů a k získání obrázků uvedených v této práci.
- V adresáři *Model* je dokument ve formátu *pdf*, ve kterém je popsán použitý model odparky.