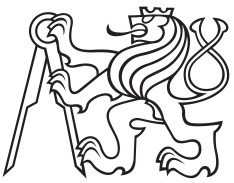
Bakalářská práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra řídicí techniky

Paralelní řešič pro prediktivní řízení

Jiří Burant

Květen 2015

České vysoké učení technické v Praze
Fakulta elektrotechnická

katedra řídicí techniky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: **Jiří Burant**

Studijní program: Kybernetika a robotika
Obor: Systémy a řízení

Název tématu: **Paralelní řešič pro prediktivní řízení**

Pokyny pro vypracování:

1. Popište problém prediktivního řízení a algoritmy pro jeho řešení.2. Implementujte prediktivní regulátor.3. Implementujte paralelní prediktivní regulátor.4. Zhodnoťte zrychlení způsobené paralelizací.

Seznam odborné literatury:

[1] J.A. Rossiter: Model-Based Predictive Control: A Practical Approach[2] S. Boyd, L. Vandenberghe: Convex Optimization[3] J. Nocedal, S.J. Wright: Numerical Optimization[4] Konferenční a časopisecké články

Vedoucí: Ing. Pavel Otta

Platnost zadání: do konce letního semestru 2015/2016

L.S.

prof. Ing. Michael Šebek, DrSc.
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 21. 10. 2014

Poděkování / Prohlášení

Chtěl bych poděkovat vedoucímu mé bakalářské práce Ing. Pavlovi Ottovi za pomoc při tvorbě této práce, a dále své rodině za podporu emocionální i materiální.

Prohlašuji, že svoji bakalářskou práci jsem vypracoval samostatně, pod vedením vedoucího bakalářské práce a použil jsem pouze zdroje, které jsou uvedeny v seznamu literatury na konci této práce.

V Praze dne 20. 5. 2015

.....

Abstrakt / Abstract

Model Predictive Control (MPC) je moderní metodou prediktivního řízení umožňující zahrnout do výpočtu budoucí vývoj řízeného systému a také omezení kladená na tento systém. V rámci regulace je řešena optimalizační úloha, kterou je možné převést na úlohu kvadratického programování. Pro řešení této úlohy je možné použít řadu metod, z nichž některé jsou v této práci uvedeny. Hlavní náplní této práce je paralelizace výpočtu Newtonova kroku pro metodu kombinované gradient/Newtonovy projekce, implementace regulátoru využívajícího této metody a zhodnocení efektivity této paralelizace.

Model Predictive Control (MPC) is a modern method of predictive control that is able to consider the future development of the controlled system as well as the limitations of the system itself. An optimalisation problem, which can be altered to quadratic programming problem, is solved within the regulation. It is possible to use a wide range of methods in order to solve this problem. Some of them are presented in this thesis. The main focus of this thesis are parallelization of the calculation of Newton step in the combined gradient/Newton projection method, implementation of the regulator using this method and evaluation of the effectivity of this parallelization.

Obsah /

Kapitola 1
Prediktivní řízení

1.1 Úvod

Tématem první kapitoly této práce je Model Predictive Control, neboli MPC. Jedná se o moderní metodu prediktivního řízení, založenou na modelu řízeného dynamického systému.

Řízení pomocí MPC je v posledních desetiletích na vzestupu, a zejména v rozsáhlých průmyslových aplikacích je stále populárnější [1]. První systémy řízené pomocí MPC se objevily v chemickém a zpracovatelském průmyslu v sedmdesátých letech [1]. Pro řízení těchto systémů s relativně pomalou dynamikou tehdejší výpočetní technika postačovala, a s narůstajícím výkonem výpočetních zařízení a se sofistikovanějšími řídícími algoritmy se MPC začalo rozšiřovat i do dalších oblastí průmyslu i mimo něj.

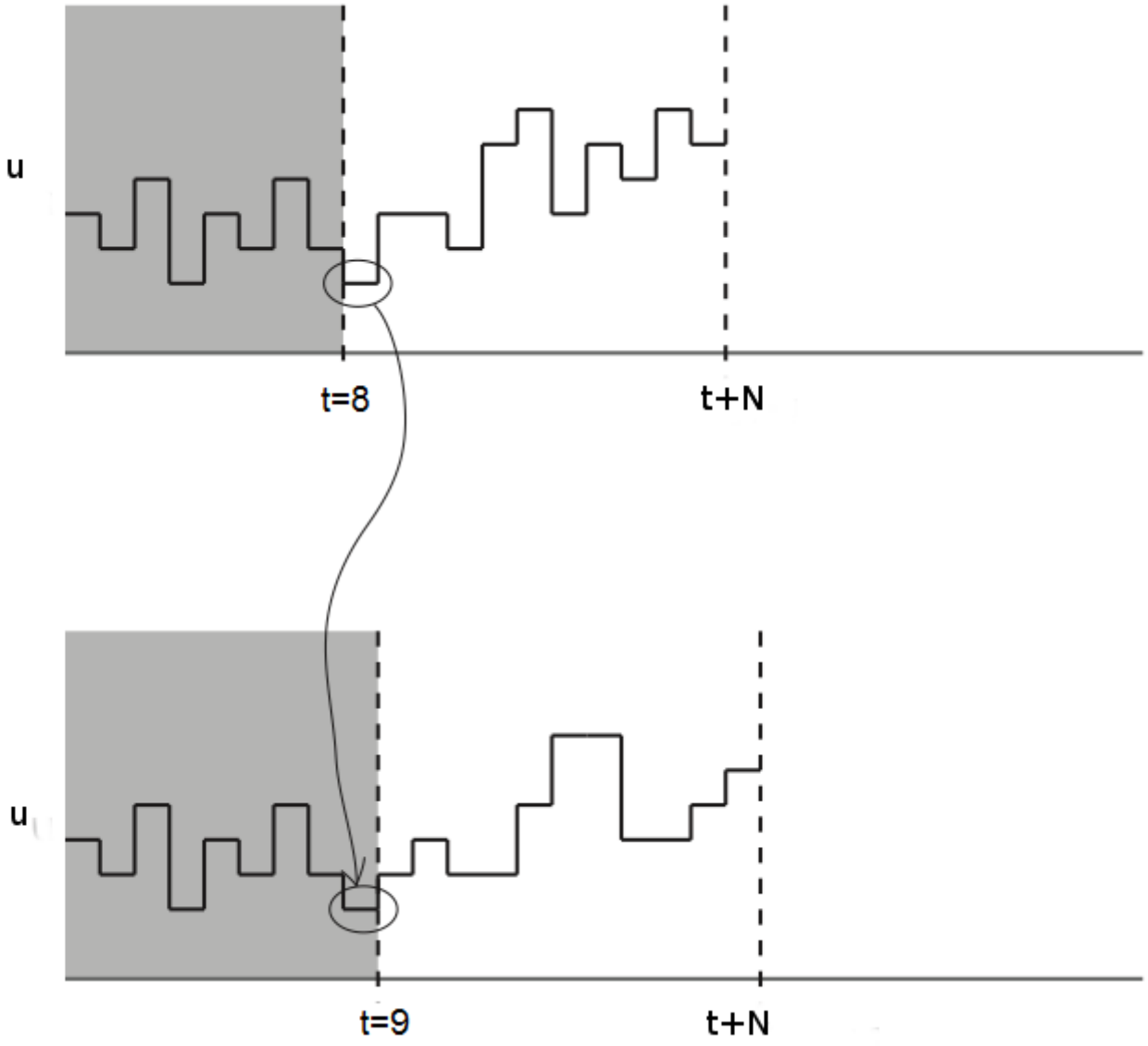
Hlavním úkolem MPC je minimalizace *ztrátové funkce* v každém kroce řízení, jedná se tedy o diskrétní regulaci. Ztrátová funkce je v každém kroce sestavena znovu, na základě požadavků na řízení, znalosti aktuálního stavu systému a znalosti modelu systému.

Charakteristickou vlastností MPC regulátoru je schopnost předpovídat vývoj systému v budoucnu a zahrnout informaci o vývoji do aktuálního řízení. V každém kroce je spočtena posloupnost akčních zásahů na takzvaném *horizontu predikce*, tato posloupnost je řešením úlohy minimalizace ztrátové funkce. Horizont predikce je tedy interval (počet kroků), pro kterou regulátor hledá tuto posloupnost. Pro zajištění zpětné vazby je důležitý *klouzavý horizont*, což znamená, že v každém kroce je série akčních zásahů spočtena znovu pro stejně dlouhý horizont, ovšem posunutý. V každém kroce je aplikován pouze první akční zásah. Ilustrace klouzavého horizontu je znázorněna na obrázku 1.1.

Hlavní předností MPC je jeho schopnost zahrnout do výpočtu omezení na stavy i vstupy řízeného systému, což řada jiných regulátorů neumožňuje. V této práci uvažujeme řízení diskrétního LTI systému, popsaného následujícím modelem.

$$\boldsymbol{x}_{t+1} = \boldsymbol{A}\boldsymbol{x}_t + \boldsymbol{B}\boldsymbol{u}_t,\ t \geq 0$$

Formulace ztrátové funkce, uvažovaná v této práci, vede na problém kvadratického programování. Některé z možných postupů jeho řešení jsou uvedeny v druhé kapitole.



Obrázek 1.1. Příklad dvou kroků klouzavého horizontu [2]

1.2 Regulace

Regulace systému v tomto případě znamená řízení stavů k nule, tedy snaha stabilizovat systém. Dalším požadavkem na rozumnou regulaci jsou co nejmenší akční zásahy. Malé hodnoty akčních zásahů a stavů mohou šetřit prostředky, a také je třeba brát v úvahu omezení výkonu reálných aktuátorů, například motorů nebo ventilů.
Jak již bylo uvedeno, řízení pomocí MPC je založeno na minimalizaci ztrátové funkce za daných omezení, budeme ji tedy minimalizovat ve stavech a vstupech.

$$\min_{\boldsymbol{x}_k, \boldsymbol{u}_k} J := \frac{1}{2} \sum_{k=0}^{N} (\boldsymbol{x}_k{}^T \boldsymbol{Q} \boldsymbol{x}_k + \boldsymbol{u}_k{}^T \boldsymbol{R} \boldsymbol{u}_k) \tag{1.1}$$

Za omezení :

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k,\ \boldsymbol{x}_0 = \boldsymbol{x}_t \tag{1.2}$$

$$\underline{\boldsymbol{u}}_k \leq \boldsymbol{u}_k \leq \overline{\boldsymbol{u}}_k$$

$$\underline{\boldsymbol{x}}_k \leq \boldsymbol{x}_k \leq \overline{\boldsymbol{x}}_k \tag{1.3}$$

Písmeno J označuje ztrátovou funkci, N je délka predikčního horizontu, $\boldsymbol{u}_k$ je vektor akčních zásahů a $\boldsymbol{x}_k$ je vektor stavů v daném kroce k, $\boldsymbol{x}_t$ je počáteční podmínka pro stavy. Matice $\boldsymbol{Q} = \boldsymbol{Q}^T \geq 0$ a $\boldsymbol{R} = \boldsymbol{R}^T > 0$ jsou váhové matice, které stanovují váhu stavů a vstupů. Symbol $\geq$ značí, že matice je pozitivně semidefinitní, resp. symbol $>$ značí, že matice je pozitivně definitní.
Matice $\boldsymbol{A}$ a $\boldsymbol{B}$ jsou stavové matice systému, který řídíme. Symboly $\underline{\boldsymbol{u}}_k$, $\overline{\boldsymbol{u}}_k$, $\underline{\boldsymbol{x}}_k$, $\overline{\boldsymbol{x}}_k$ označují omezení, která jsou kladena na vstupy a stavy. V této práci uvažujeme pouze omezení shora a zdola.

1.3 Sledování reference

V případě, že je zapotřebí sledovat referenci, je třeba ztrátovou funkci změnit. Uvažujeme-li konstantní referenci, můžeme ztrátovou funkci přeformulovat následovně.

$$\min_{\boldsymbol{x}_k, \boldsymbol{u}_k} J := \frac{1}{2}\sum_{k=0}^{N} (\boldsymbol{x}_k - \boldsymbol{r})^T \boldsymbol{Q} (\boldsymbol{x}_k - \boldsymbol{r}) + \boldsymbol{u}_k^T \boldsymbol{R} \boldsymbol{u}_k \tag{1.4}$$

Symbol $\boldsymbol{r}$ označuje vektor reference, tedy vektor hodnot, kterých mají stavy dosáhnout. Tímto způsobem se snažíme minimalizovat odchylku stavů od reference. Reference nemusí být konstantní, je možné závést i referenci v čase proměnnou, která bude mít v každém kroce jinou hodnotu. Potom by ve vzorci (1.4) bylo použito $\boldsymbol{r}_k$ místo $\boldsymbol{r}$.
Tento zápis je možné přeformulovat do tvaru, který bude více podobný původní ztrátové funkci(1.1). Nejprve přepíšeme vývoj dynamiky systému.

$$\begin{pmatrix} \boldsymbol{x}_{k+1} \\ \boldsymbol{r}_{k+1} \end{pmatrix} = \begin{pmatrix} \boldsymbol{A} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I} \end{pmatrix} \begin{pmatrix} \boldsymbol{x}_k \\ \boldsymbol{r}_k \end{pmatrix} + \begin{pmatrix} \boldsymbol{B} \\ \boldsymbol{0} \end{pmatrix} \boldsymbol{u}_k$$

Přidáním vektoru $\boldsymbol{r}_k$ jsme rozšířili stavový vektor $\boldsymbol{x}_k$ na rozšířený vektor stavů $\widehat{\boldsymbol{x}}_k = (\begin{array}{cc} \boldsymbol{x}_k^T & \boldsymbol{r}_k^T \end{array})^T$. Reference je zde uvažována konstantní.
Nyní přepíšeme vzorec (1.4) následovně.

$$\min_{\widehat{\boldsymbol{x}}_k, \boldsymbol{u}_k} J := \frac{1}{2}\sum_{k=0}^{N} (\widehat{\boldsymbol{x}}_k^T \boldsymbol{Q}_a \widehat{\boldsymbol{x}}_k + \boldsymbol{u}_k^T \boldsymbol{R} \boldsymbol{u}_k) \tag{1.5}$$

Symbol $\boldsymbol{Q}_a$ představuje rozšířenou váhovou matici stavů a platí pro ní následující vzorec.

$$\boldsymbol{Q}_a = (\begin{array}{cc} \boldsymbol{I} & -\boldsymbol{I} \end{array})^T \boldsymbol{Q} (\begin{array}{cc} \boldsymbol{I} & -\boldsymbol{I} \end{array})$$

1.3.1 nulová ustálená odchylka

V některých případech je výhodnější minimalizovat spíše změnu vstupu oproti předchozí hodnotě, než-li jeho absolutní hodnotu. Zavádíme proto nový člen $\Delta\boldsymbol{u}_k = \boldsymbol{u}_k - \boldsymbol{u}_{k-1}$.
Uvažujeme-li zároveň sledování reference z předchozí sekce, přepíšeme vzorec (1.4) následovně.

$$\min_{\widehat{\boldsymbol{x}}_k, \Delta\boldsymbol{u}_k} J := \frac{1}{2}\sum_{k=0}^{N} (\widehat{\boldsymbol{x}}_k^T \boldsymbol{Q}_a \widehat{\boldsymbol{x}}_k + \Delta\boldsymbol{u}_k^T \boldsymbol{R} \Delta\boldsymbol{u}_k) \tag{1.6}$$

1.4 Rozšíření na nekonečný horizont

Omezením prediktivního řízení je skutečnost, že minimalizaci je možno provádět jen na konečně dlouhém predikčním horizontu. Obecně platí, že čím delší horizont, tím lepší predikce, ale současně tím roste i výpočetní náročnost. Proto je v praxi obtížné dosáhnout skutečně optimálního řízení, tedy počítat na nekonečném predikčním horizontu. Náhradou za nekonečný predikční horizont však může být takzvaný koncový člen, který přidáme do ztrátové funkce. Má následující podobu.

$$\frac{1}{2}\boldsymbol{x}_N^T \boldsymbol{Q}_N \boldsymbol{x}_N \tag{1.7}$$

Vektor $\boldsymbol{x}_N$ je vektor stavů v kroce N, tedy posledním kroce predikčního horizontu a matice $\boldsymbol{Q}_N$ je *koncová váha*. Tu můžeme získat jako řešení Riccatiho rovnice. Poté je člen (1.7) ekvivalentní následujícímu výrazu, důkaz [3].

$$\frac{1}{2}\sum_{k=N}^{\infty} \boldsymbol{x}_k^T \boldsymbol{Q} \boldsymbol{x}_k + \boldsymbol{u}_k^T \boldsymbol{R} \boldsymbol{u}_k \tag{1.8}$$

1.5 Řídká formulace

Nyní je možné problém transformovat do formy vhodnější pro následnou optimalizaci. První z takovou formou je řídká formulace.
V případě řídké formulace uvažujeme minimalizační kritérium (1.1). Zapíšeme jej vektorově.

$$\min_{\boldsymbol{x},\boldsymbol{u}} J := \frac{1}{2}(\boldsymbol{x}^T \widehat{\boldsymbol{Q}} \boldsymbol{x} + \boldsymbol{u}^T \widehat{\boldsymbol{R}} \boldsymbol{u}) \tag{1.9}$$

$\widehat{\boldsymbol{Q}} = \boldsymbol{I}_N \otimes \boldsymbol{Q}, \widehat{\boldsymbol{R}} = \boldsymbol{I}_N \otimes \boldsymbol{R}, \boldsymbol{x} = (\boldsymbol{x}_1^T, \boldsymbol{x}_2^T \dots \boldsymbol{x}_N^T)^T, \boldsymbol{u} = (\boldsymbol{u}_0^T, \boldsymbol{u}_1^T \dots \boldsymbol{u}_{N-1}^T)^T$, symbol $\otimes$ značí Kroneckerův součin. Matice $\widehat{\boldsymbol{Q}}$ a $\widehat{\boldsymbol{R}}$ jsou rozšířené váhové matice, $\boldsymbol{x}$ a $\boldsymbol{u}$ jsou stavy, resp. vstupy ve zřetězené formě.

Rovnostní omezení (1.2) přepíšeme následovně.

$$\widehat{\boldsymbol{B}}\boldsymbol{u} + \widehat{\boldsymbol{A}}\boldsymbol{x} + \boldsymbol{r}_x(\boldsymbol{x}_0) = \boldsymbol{0} \tag{1.10}$$

$$\widehat{\boldsymbol{B}} = \begin{pmatrix} \boldsymbol{B} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{B} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{B} \end{pmatrix}, \widehat{\boldsymbol{A}} = \begin{pmatrix} -\boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{A} & -\boldsymbol{I} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{A} & -\boldsymbol{I} & \cdots & \boldsymbol{0} \\ \vdots & \vdots & & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{A} & -\boldsymbol{I} \end{pmatrix}, \boldsymbol{r}_x(\boldsymbol{x}_0) = \begin{pmatrix} \boldsymbol{A}\boldsymbol{x}_0 \\ \boldsymbol{0} \\ \boldsymbol{0} \\ \vdots \\ \boldsymbol{0} \end{pmatrix} \tag{1.11}$$

Dále uvažujeme substituci $\boldsymbol{z} = (\boldsymbol{u}^T, \boldsymbol{x}^T)^T$. Podle této nové proměnné budeme optimalizovat. Problém je možné přeformulovat do následující podoby.

$$\min_{\boldsymbol{z}} J := \frac{1}{2}\boldsymbol{z}^T\boldsymbol{H}\boldsymbol{z} \tag{1.12}$$
$$\text{Za omezení :}$$
$$\boldsymbol{C}\boldsymbol{z} + \boldsymbol{r}_z(\boldsymbol{z}_0) = \boldsymbol{0}$$
$$\underline{\boldsymbol{z}} \leq \boldsymbol{z} \leq \overline{\boldsymbol{z}}$$

pročež jsme zavedli nové matice $\boldsymbol{C}$, $\boldsymbol{H}$ a $\boldsymbol{r}_z(\boldsymbol{z}_0)$. Matice $\boldsymbol{H}$ je v této formulaci pozitivně semidefinitní.

$$\boldsymbol{C} = \begin{pmatrix} \widehat{\boldsymbol{B}} & \widehat{\boldsymbol{A}} \end{pmatrix}, \ \boldsymbol{H} = \begin{pmatrix} \widehat{\boldsymbol{R}} & \\ & \widehat{\boldsymbol{Q}} \end{pmatrix}, \ \boldsymbol{r}_z(\boldsymbol{z}_0) = \begin{pmatrix} \boldsymbol{0} \\ \boldsymbol{r}_x(\boldsymbol{x}_0) \end{pmatrix} \tag{1.13}$$

Omezení na stavy i vstupy se spojí do vektorů $\underline{\boldsymbol{z}} = \begin{pmatrix} \underline{\boldsymbol{u}}^T & \underline{\boldsymbol{x}}^T \end{pmatrix}^T$ a $\overline{\boldsymbol{z}} = \begin{pmatrix} \overline{\boldsymbol{u}}^T & \overline{\boldsymbol{x}}^T \end{pmatrix}^T$.

1.6 Kondenzace

Druhým způsobem formulace, uvedeným v této práci je takzvaná kondenzovaná formulace. Vztah (1.1) je možné, zavedením složených vektorů $\boldsymbol{x}$ a $\boldsymbol{u}$, přepsat do kompaktní formy.

$$\min_{\boldsymbol{x},\boldsymbol{u}} J := \frac{1}{2}(\boldsymbol{x}^T\widehat{\boldsymbol{Q}}\boldsymbol{x} + \boldsymbol{u}^T\widehat{\boldsymbol{R}}\boldsymbol{u}), \tag{1.14}$$

kde $\widehat{\boldsymbol{Q}} = \boldsymbol{I}_N \otimes \boldsymbol{Q}$, $\widehat{\boldsymbol{R}} = \boldsymbol{I}_N \otimes \boldsymbol{R}$, $\boldsymbol{x} = (\boldsymbol{x}_1^T, \boldsymbol{x}_2^T \dots \boldsymbol{x}_N^T)^T$, $\boldsymbol{u} = (\boldsymbol{u}_0^T, \boldsymbol{u}_1^T \dots \boldsymbol{u}_{N-1}^T)^T$.
V kondenzované formě vztah (1.2) přepíšeme následovně.

$$\begin{aligned} \boldsymbol{x}_0 &= \boldsymbol{x}_t \\ \boldsymbol{x} &= \boldsymbol{P}\boldsymbol{x}_0 + \boldsymbol{V}\boldsymbol{u} \end{aligned} \tag{1.15}$$

$$\boldsymbol{P} = \begin{pmatrix} \boldsymbol{A} \\ \boldsymbol{A}^2 \\ \boldsymbol{A}^3 \\ \vdots \\ \boldsymbol{A}^N \end{pmatrix}, \boldsymbol{V} = \begin{pmatrix} \boldsymbol{B} & \boldsymbol{0} & \boldsymbol{0} & \dots & \boldsymbol{0} \\ \boldsymbol{A}\boldsymbol{B} & \boldsymbol{B} & \boldsymbol{0} & \dots & \boldsymbol{0} \\ \boldsymbol{A}^2\boldsymbol{B} & \boldsymbol{A}\boldsymbol{B} & \boldsymbol{B} & \dots & \boldsymbol{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{A}^{N-1}\boldsymbol{B} & \boldsymbol{A}^{N-2}\boldsymbol{B} & \boldsymbol{A}^{N-3}\boldsymbol{B} & \dots & \boldsymbol{B} \end{pmatrix} \tag{1.16}$$

Nyní dosadíme (1.15) do kritéria (1.14) za $\boldsymbol{x}$ a roznásobíme.

$$\min_{\boldsymbol{u}} J = \frac{1}{2}(\boldsymbol{x}_0{}^T\boldsymbol{P}^T\widehat{\boldsymbol{Q}}\boldsymbol{P}\boldsymbol{x}_0 + 2\boldsymbol{u}^T\boldsymbol{V}^T\widehat{\boldsymbol{Q}}\boldsymbol{P}\boldsymbol{x}_0 + \boldsymbol{u}^T\boldsymbol{V}^T\widehat{\boldsymbol{Q}}\boldsymbol{V}\boldsymbol{u} + \boldsymbol{u}^T\widehat{\boldsymbol{R}}\boldsymbol{u}) \tag{1.17}$$

Toto kritérium minimalizujeme. Můžeme zanedbat konstantní členy, protože nás zajímají pouze souřadnice minima, nikoliv hodnota ztrátové funkce v minimu.

$$\min_{\boldsymbol{u}} J = \frac{1}{2}(\boldsymbol{u}^T(\boldsymbol{V}^T\widehat{\boldsymbol{Q}}\boldsymbol{V} + \widehat{\boldsymbol{R}})\boldsymbol{u} + 2\boldsymbol{u}^T\boldsymbol{V}^T\widehat{\boldsymbol{Q}}\boldsymbol{P}\boldsymbol{x}_0) \tag{1.18}$$

Dále můžeme označit $\boldsymbol{H} = \boldsymbol{V}^T\widehat{\boldsymbol{Q}}\boldsymbol{V} + \widehat{\boldsymbol{R}}$ a $\boldsymbol{f} = \boldsymbol{V}^T\widehat{\boldsymbol{Q}}\boldsymbol{P}\boldsymbol{x}_0$.
Matice $\boldsymbol{H}$ je pozitivně definitní, což plyne z toho, že $\widehat{\boldsymbol{R}} > \boldsymbol{0}$.

$$\min_{\boldsymbol{u}} J = \frac{1}{2}\boldsymbol{u}^T\boldsymbol{H}\boldsymbol{u} + \boldsymbol{f}^T\boldsymbol{u} \tag{1.19}$$

1.6.1 Nerovnostní omezení

V případě, že součástí optimalizační úlohy jsou omezení, je třeba transformovat tato omezení do obecné formy. Minimalizační úlohu totiž v další části textu řešíme jako úlohu kvadratického programování, a proto se snažíme ukázat ekvivalenci s obecným tvarem, uvedeným v druhé kapitole.
Předpokládáme-li jednoduchá omezení z (1.3), můžeme je zapsat ve sloučené formě.

$$\underline{\boldsymbol{x}} \leq \boldsymbol{x} \leq \overline{\boldsymbol{x}} \tag{1.20}$$

$$\underline{\boldsymbol{u}} \leq \boldsymbol{u} \leq \overline{\boldsymbol{u}} \tag{1.21}$$

Omezení na stavy z (1.20) je možné převést na omezení na vstupy, dosadíme-li do výrazu (1.20) z (1.15).

$$\underline{\boldsymbol{x}} \leq \boldsymbol{P}\boldsymbol{x}_0 + \boldsymbol{V}\boldsymbol{u} \leq \overline{\boldsymbol{x}} \tag{1.22}$$

Po úpravách dostaneme obecný tvar.

$$\boldsymbol{G}\boldsymbol{u} \leq \boldsymbol{S}\boldsymbol{x}_0 + \boldsymbol{W} \tag{1.23}$$

Pro tuto rovnici platí následující vztahy. $\boldsymbol{G} = \begin{pmatrix} \boldsymbol{V} \\ -\boldsymbol{V} \end{pmatrix}$, $\boldsymbol{S} = \begin{pmatrix} -\boldsymbol{P} \\ \boldsymbol{P} \end{pmatrix}$, $\boldsymbol{W} = \begin{pmatrix} \overline{\boldsymbol{x}} \\ -\underline{\boldsymbol{x}} \end{pmatrix}$.

Omezení na vstupy z (1.21) je možné zapsat formou maticové nerovnosti.

$$\boldsymbol{A}\boldsymbol{u} \leq \boldsymbol{b} \tag{1.24}$$

Matice $\boldsymbol{A} = \begin{pmatrix} \boldsymbol{I} \\ -\boldsymbol{I} \end{pmatrix}$, vektor $\boldsymbol{b} = \begin{pmatrix} \overline{\boldsymbol{u}} \\ -\underline{\boldsymbol{u}} \end{pmatrix}$.

1.6.2 Blokování vstupů

Myšlenkou blokování vstupů je zjednodušit výpočet tím, že vstupy budou po určitou dobu konstantní, tím se zmenší stupeň volnosti daného problému a sníží se tak velikost řešeného problému. Tato metoda je popsána například v [4]. V souvislosti s tímto zjednodušením zavádíme pojem *horizont řízení*. Horizont řízení je interval, pro který počítáme akční zásahy. Po konci horizontu řízení uvažujeme vstupy konstantní.

$$\begin{pmatrix} \boldsymbol{u}_0 \\ \vdots \\ \boldsymbol{u}_{N_r-1} \\ \vdots \\ \boldsymbol{u}_{N-1} \end{pmatrix} = \begin{pmatrix} \boldsymbol{I} & & \\ & \ddots & \\ & & \boldsymbol{I} \\ & & \vdots \\ & & \boldsymbol{I} \end{pmatrix} \begin{pmatrix} \boldsymbol{u}_0 \\ \vdots \\ \boldsymbol{u}_{N_r-1} \end{pmatrix}$$

Symbol N_r značí délku horizontu řízení.

Kapitola 2
Kvadratické programování

Kvadratické programování je druhem matematické optimalizace, ve které minimalizujeme kvadratickou funkci za daných omezení. Obecný problém kvadratického programování má následující podobu.

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) := \frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{c}^T\boldsymbol{x} \tag{2.1}$$

Za omezení :

$$\boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}$$

Matice $\boldsymbol{Q} \in R^{n\times n}$ charakterizuje kvadratickou složku problému, a dále platí $\boldsymbol{Q} \geq 0$ a $\boldsymbol{Q} = \boldsymbol{Q}^T$, $\boldsymbol{c} \in R^n$ je vektor a $\boldsymbol{x} \in R^n$ je vektor, jehož hodnoty optimalizujeme. Matice $\boldsymbol{A} \in R^{m\times n}$ společně s vektorem $\boldsymbol{b} \in R^m$ stanovuje omezení na řešení optimalizace.
Pro lepší popis optimalizace nyní definujme následující pojmy.
Hessova matice $\boldsymbol{H}$ představuje čtvercovou matici druhých parciálních derivací funkce.

$$\boldsymbol{H} = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_N} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_N} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_N \partial x_1} & \frac{\partial^2 f}{\partial x_N \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_N^2} \end{pmatrix} = \boldsymbol{Q}$$

Gradient $\nabla f(\boldsymbol{x})$ je vektorem prvních parciálních derivací.

$$\nabla f(\boldsymbol{x}) = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_N} \end{pmatrix} = \boldsymbol{Q}\boldsymbol{x} + \boldsymbol{c}$$

Jakobiho matice $\boldsymbol{J}$ je maticí prvních parciálních derivací jednotlivých funkcí soustavy rovnic. Protože v našem případě uvažujeme pouze jednu skalární funkci, Jakobián se zjednodušší na vektor.

$$\boldsymbol{J} = \begin{pmatrix} \frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} & \cdots & \frac{\partial f}{\partial x_N} \end{pmatrix} = \boldsymbol{x}^T\boldsymbol{Q} + \boldsymbol{c}^T$$

Úlohy minimalizace kvadratické funkce můžeme rozdělit podle složitosti jejich řešení do tří skupin od nejméně složitých po nejsložitější, například pro třetí skupinu je nutné použít iterativních metod:

1)Úlohy bez omezení
2)Úlohy s rovnostními omezeními
3)Úlohy s nerovnostními omezeními

V následujícím textu budou uvedeny základní metody řešení těchto úloh.

2.1 Minimalizace bez omezení

V případě, že na minimum nejsou kladena omezení, řešení se výrazně zjednodušší. nutnou podmínkou pro optimum je $\nabla f(\boldsymbol{x}) = \boldsymbol{0}$. Postačující podmínkou je $\nabla f(\boldsymbol{x}) = \boldsymbol{0}$ a zároveň $\boldsymbol{H} \geq \boldsymbol{0}$.
Na rovnici z (2.1) nejsou kladena omezení.

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) := \frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{c}^T\boldsymbol{x} \tag{2.2}$$

2.1.1 Gradientní metoda

Gradientní metoda je jedna ze základních iterativních metod. Vychází z následujícího vztahu.

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i - \alpha\nabla f(\boldsymbol{x}_i) \tag{2.3}$$

Symbol $\boldsymbol{x}_i$ označuje hodnotu vektoru $\boldsymbol{x}$ v iteraci i, $\boldsymbol{x}_{i+1}$ je hodnota $\boldsymbol{x}$ v iteraci následující a α je konstanta, která stanoví velikost kroku.
Tato metoda je metodou prvního řádu, protože využívá prvních dvou členů Taylorova rozvoje vyšetřované funkce.
Problémem je stanovení velikosti α tak, aby výraz konvergoval k minimu.
Koeficient α nesmí být příliš velký, protože poté by hrozilo, že algoritmus bude divergovat a řešení nebude nalezeno. Platí $0 < \alpha < \frac{1}{\lambda_{\max}(\boldsymbol{H})}$. Důkaz lze nalézt například v [5].

2.1.2 Metoda největšího spádu

Jedná se o druh gradientní metody, ve které uvažujeme α takové, že zaručí největší spád funkce v daném směru. Koeficient α však není konstantní, musíme jej v každém kroce vypočítat znovu. Metoda i celý postup řešení jsou uvedeny v [6].
Zaveďme $\boldsymbol{r}_i = -\nabla f(\boldsymbol{x}_i)$, kde $\boldsymbol{r}$ znamená reziduum.
Uvažujme rovnici kvadratického kritéria (2.2) pro $\boldsymbol{x}_{i+1}$ z (2.3).

$$f(\boldsymbol{x}_{i+1}) = \frac{1}{2}\boldsymbol{x}_i^T\boldsymbol{Q}\boldsymbol{x}_i + \alpha_i\boldsymbol{x}_i^T\boldsymbol{Q}\boldsymbol{r}_i + \frac{1}{2}\alpha_i^2\boldsymbol{r}_i^T\boldsymbol{Q}\boldsymbol{r}_i + \boldsymbol{x}_i^T\boldsymbol{c} + \alpha_i\boldsymbol{r}_i^T\boldsymbol{c}$$

Hledanou neznámou je zde α_i, to určíme z nutné podmínky pro minimum, tedy derivaci položíme rovnou nule.

$$\frac{df}{d\alpha_i} = \boldsymbol{x}_i^T\boldsymbol{Q}\boldsymbol{r}_i + \alpha_i\boldsymbol{r}_i^T\boldsymbol{Q}\boldsymbol{r}_i + \boldsymbol{r}_i^T\boldsymbol{c} = 0$$

Následně je možné vyjádřit α_i.

$$\alpha_i = \frac{\boldsymbol{r}_i^T\boldsymbol{r}_i}{\boldsymbol{r}_i^T\boldsymbol{Q}\boldsymbol{r}_i}$$

2.1.3 Metoda sdružených gradientů

Tato metoda vychází z následujícího vztahu.

$$f(\boldsymbol{x}_{i+1}) = \frac{1}{2}\boldsymbol{x}_i^T\boldsymbol{Q}\boldsymbol{x}_i + \alpha_i\boldsymbol{x}_i^T\boldsymbol{Q}\boldsymbol{r}_i + \frac{1}{2}\alpha_i^2\boldsymbol{r}_i^T\boldsymbol{Q}\boldsymbol{r}_i + \boldsymbol{x}_i^T\boldsymbol{c} + \alpha_i\boldsymbol{r}_i^T\boldsymbol{c}$$

V tomto případě aproximace neprobíhá ve směru opačného gradientu, ale obecně ve směru $\boldsymbol{s}_i$.

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \alpha_i \boldsymbol{s}_i \tag{2.4}$$

Parametr α_i je v každém kroce vypočten znovu a je optimální v tom smyslu, že najde minimum v daném směru.
Směr $\boldsymbol{s}_i$ se nazývá konjugovaným směrem a platí pro něj následující vztah.

$$\boldsymbol{s}_{i+1} = \nabla f(\boldsymbol{x}_{i+1}) - \beta_i \boldsymbol{s}_i \tag{2.5}$$

β_i je v každém kroce také vypočteno znovu, a to podle vztahu:

$$\beta_i = \frac{\nabla f(\boldsymbol{x}_{i+1})^T \nabla f(\boldsymbol{x}_{i+1})}{\nabla f(\boldsymbol{x}_i)^T \nabla f(\boldsymbol{x}_i)} \tag{2.6}$$

Detailní odvození a více informací o této metodě lze nalézt například v [6].

2.1.4 Zpětná gradientní metoda

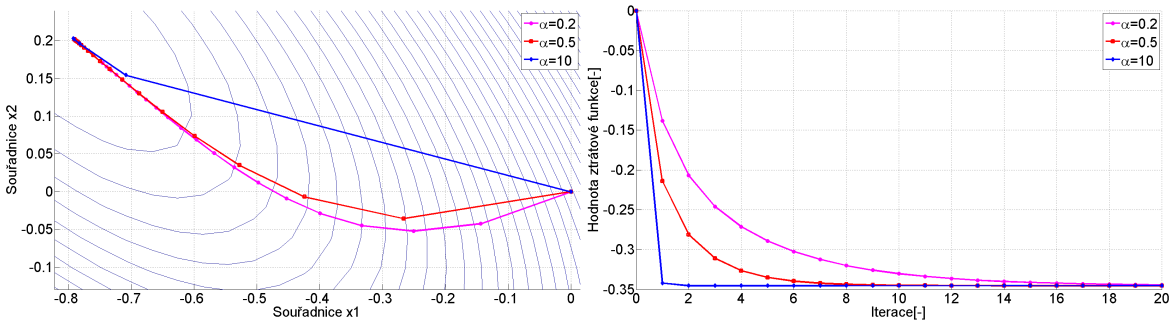
Další metodou je zpětná gradientní metoda. Tato metoda vychází z podobného vztahu jako gradientní metoda, pouze místo gradientu je v i-tém kroce zvolen gradient v kroce následujícím.

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i - \alpha \nabla f(\boldsymbol{x}_{i+1})$$

Po dosazení a úpravě vztahů získáme výsledný předpis.

$$\boldsymbol{x}_{i+1} = (\boldsymbol{I} + \alpha \boldsymbol{Q})^{-1} \boldsymbol{x}_i - \alpha (\boldsymbol{I} + \alpha \boldsymbol{Q})^{-1} \boldsymbol{c}$$

Nevýhodou této metody je především složitost výpočtu členu $(\boldsymbol{I} + \alpha \boldsymbol{Q})^{-1}$.
Výhodou je, že hodnota parametru α není omezena, jako v případě gradientní metody.



Obrázek 2.1. Zpětná gradientní metoda

Z obrázku 2.1 vidíme, že rychlost konvergence této metody se zvyšuje s rostoucím α.

2.1.5 Dvoukroková zpětná gradientní metoda

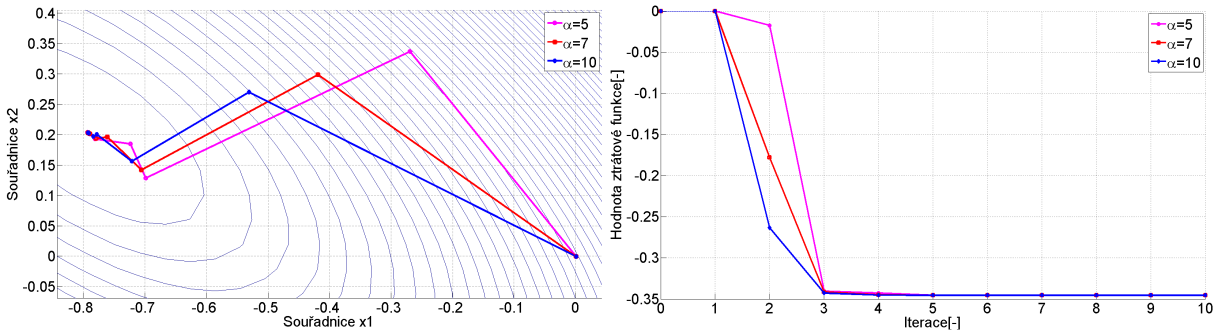
Jedná se opět o zpětnou gradientní metodu, s tím rozdílem, že počítáme s dvěma předchozími kroky.

$$\boldsymbol{x}_i = \boldsymbol{x}_{i-1} - \alpha \nabla f(\boldsymbol{x}_{i+1})$$

Což je možné přepsat do následující podoby.

$$\boldsymbol{x}_{i+1} = \frac{1}{\alpha}\boldsymbol{Q}^{-1}(\boldsymbol{x}_{i-1} - \boldsymbol{x}_i) - \boldsymbol{Q}^{-1}\boldsymbol{c} \tag{2.7}$$

Tato metoda se liší od zpětné gradientní metody především tím, že není třeba počítat $(\boldsymbol{I}+\alpha\boldsymbol{Q})^{-1}$. Výraz α totiž nemusí být konstantní a počítat v každém kroce znovu inverzi by bylo výpočetně náročné.
U této metody postačí vypočítat inverzi matice $\boldsymbol{Q}$ pouze jednou. Další zajímavostí je člen $-\boldsymbol{Q}^{-1}\boldsymbol{c}$ v rovnici (2.7). Ten je stejný jako v případě metody prosté inverze (2.11).
Z rovnice (2.7) je vidět, že pro $\alpha \to \infty$ je první člen roven nule a řešení úlohy je totožné s řešením Newtonovy metody.
Při výpočtu pomocí této metody je také nutné v první iteraci zvolit vhodně prvek $\boldsymbol{x}_{i-1}$.
Použití zpětných gradientních metod pro řešení neomezených problémů je nevýhodné, protože za zvýšení ceny výpočtu nepřináší žádné výhody.



Obrázek 2.2. Dvoukroková zpětná gradientní metoda

2.1.6 Gauss-Seidelova metoda

Matici $\boldsymbol{Q}$ lze rozložit pomocí následujícího vztahu.

$$\boldsymbol{Q} = \boldsymbol{L}^* + \boldsymbol{U}$$

V této rovnici platí:

$$\boldsymbol{L}^* = \begin{pmatrix} a_{11} & 0 & \dots & 0 \\ a_{21} & a_{22} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix}$$

$$\boldsymbol{U} = \begin{pmatrix} 0 & a_{12} & \dots & a_{1n} \\ 0 & 0 & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 \end{pmatrix}.$$

Řešení je nalezeno pomocí následující rovnice.

$$\boldsymbol{x}_{i+1} = -\boldsymbol{L}^{*-1}\boldsymbol{U}\boldsymbol{x}_i - \boldsymbol{L}^{*-1}\boldsymbol{c} \tag{2.8}$$

Matice $\boldsymbol{Q}$ musí být pozitivně definitní a symetrická nebo přísně či neredukovatelně diagonálně dominantní, tedy musí platit $|a_{ii}| \geq \sum_{j \neq i} |a_{ij}|$, pro všechna i, aby byla zajištěna konvergence. Tento výpočet je možné zefektivnit následujícím postupem. Díky tomu,

že matice $\boldsymbol{L}^*$ je horní trojúhelníková, prvky $\boldsymbol{x}$ mohou být vypočteny sekvenčně pomocí dopředné substituce.

$$x_j^{i+1} = \frac{1}{a_{jj}}(b_i - \sum_{k<j} a_{jk}x_k^{i+1} - \sum_{k>j} a_{jk}x_k^i)$$

a $j, k = 1, 2, ..., n$, kde n je rozměr matice.

2.1.7 Newtonova metoda

Newtonova metoda je další z iterativních metod hledajících extrém funkce. Je založena na hledání kořenů funkce. V optimalizaci je používána ke hledání kořenů derivace funkce, jejíž extrémy chceme naleznout. Vzhledem k tomu, že funkce, kterou se v tomto případě snažíme minimalizovat je kvadratická, má pouze jeden extrém, a tím je globální minimum.
Newtonova metoda je metoda druhého řádu, což znamená, že využívá tři první členy Taylorova rozvoje. Oproti gradientním metodám má tedy k dispozici větší množství informace. Následuje obecný předpis Newtonovy metody.

$$\boldsymbol{x} - \overline{\boldsymbol{x}} = -\boldsymbol{H}(\overline{\boldsymbol{x}}^{-1})\nabla f(\overline{\boldsymbol{x}}) \tag{2.9}$$

Vektor $\boldsymbol{x}$ značí souřadnice hledaného minima a $\overline{\boldsymbol{x}}$ je bod, ve kterém je funkce aproximovaná Taylorovým rozvojem. Pro případ kvadratické funkce se rovnice (2.9) výrazně zjednodušší a řešení vyšetřované funkce je nalezeno v pouhé jedné iteraci.

$$\boldsymbol{x} - \overline{\boldsymbol{x}} = -\boldsymbol{Q}^{-1}(\boldsymbol{Q}\overline{\boldsymbol{x}} + \boldsymbol{c}) \tag{2.10}$$

2.1.7.1 Prostá inverze

Rovnici (2.10) je možné vyřešit přímo pomocí inverze matice $\boldsymbol{Q}$.

$$\boldsymbol{x} = -\boldsymbol{Q}^{-1}\boldsymbol{c} \tag{2.11}$$

Výpočet inverzní matice má ale poměrně velkou složitost [7], je tedy třeba hledat způsoby jak tento výpočet obejít.

2.1.7.2 Choleského faktorizace

Každá symetrická, pozitivněděfinitní matice s reálnými prvky může být rozložena na součin dvou matic. Dolní trojúhelníkové matice $\boldsymbol{L}$ a transpozice této matice. Tento rozklad je jednoznačný [8].

$$\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T \tag{2.12}$$

Tento výraz je následně dosazen do řešení Newtonova kroku.

$$\boldsymbol{Q}\boldsymbol{x} = -\boldsymbol{c}$$

$$\boldsymbol{L}\boldsymbol{L}^T\boldsymbol{x} = -\boldsymbol{c} \tag{2.13}$$

Dále je řešena soustava maticových rovnic.

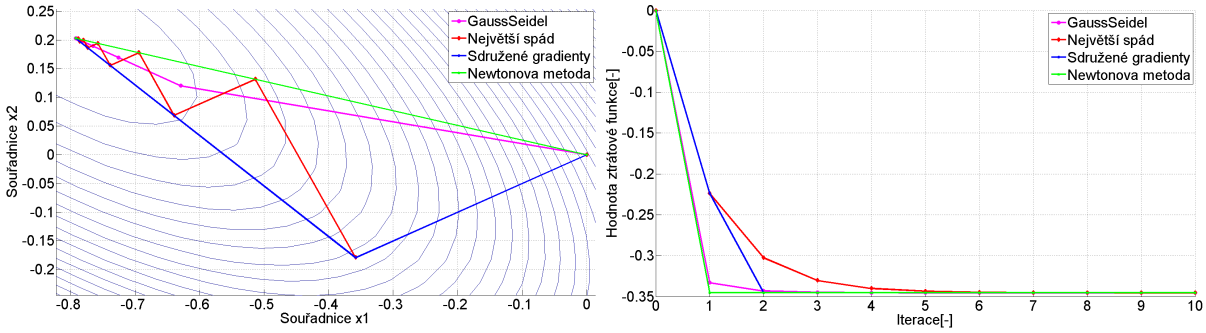
$$\boldsymbol{L}\boldsymbol{y} = -\boldsymbol{c} \tag{2.14}$$

$$\boldsymbol{L}^T\boldsymbol{x} = \boldsymbol{y} \tag{2.15}$$

Složitost výpočtu faktorizace je $O(\frac{1}{3}(n_x^3))$ [9], kde n_x je délka vektoru $\boldsymbol{x}$. Složitost výpočtu pro maticové rovnosti z (2.14) je $O(n_x^2)$ [9] pro každou. Výpočet je tedy méně složitý, avšak ne řádově.

2.1.8 Srovnání metod

Na obrázku 2.3 je znázorněno porovnání konvergence jednotlivých metod pro neomezené problémy.



Obrázek 2.3. Porovnání metod pro neomezené problémy

Z obrázku 2.3 vidíme, že nejrychleji konverguje Newtonova metoda. Oproti ostatním má tu výhodu, že je druhého řádu, má tedy větší množství informací. Druhá v pořadí je metoda sdružených gradientů. Ta konverguje v pouhých dvou krocích. Rychlost konvergence pro metodu sdružených gradientů je zaručena, viz [6]. Tato metoda vždy konverguje v počtu iterací rovném nejvýše rozměru problému.
Rychlost konvergence zbylých dvou metod není zaručena.

2.2 Rovnostní omezení

Pro řešení QP s rovnostními omezeními lze použít Lagrangeovy multiplikátory. Tato omezení se dají obecně zapsat jako $\boldsymbol{Ax} - \boldsymbol{b} = \boldsymbol{0}$. Problém pro rovnostní omezení vypadá následovně.

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) := \frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{c}^T\boldsymbol{x} \tag{2.16}$$

Za omezení :

$$\boldsymbol{Ax} = \boldsymbol{b}$$

Nejprve je třeba sestavit Lagrangeovu funkci $\boldsymbol{L}$ k našemu QP problému (2.16).

$$\boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\lambda}) := \frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{c}^T\boldsymbol{x} + \boldsymbol{\lambda}^T(\boldsymbol{Ax} - \boldsymbol{b}) \tag{2.17}$$

Dle nutné podmínky pro minimum této funkce musí platit, že parciální derivace $\boldsymbol{L}$ podle $\boldsymbol{x}$ a $\boldsymbol{\lambda}$ se rovnají nule.

$$\begin{pmatrix} \boldsymbol{Q} & \boldsymbol{A}^T \\ \boldsymbol{A} & \boldsymbol{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} -\boldsymbol{c} \\ \boldsymbol{b} \end{pmatrix} \tag{2.18}$$

Řešením této soustavy rovnic je možné určit $\boldsymbol{x}$ a $\boldsymbol{\lambda}$.

2.2.1 Schurova metoda

Schurova metoda řeší rovnici (2.18). Nejprve z první rovnice z (2.18) vyjádříme $\boldsymbol{x}$ a dosadíme ho do druhé rovnice.

$$\boldsymbol{A}\boldsymbol{Q}^{-1}\boldsymbol{A}^T\boldsymbol{\lambda} = -\boldsymbol{A}\boldsymbol{Q}^{-1}\boldsymbol{c} - \boldsymbol{b} \tag{2.19}$$

Z rovnice (2.19) vypočteme $\boldsymbol{\lambda}$ a dosadíme do první rovnice z (2.18).

$$\boldsymbol{x} = -\boldsymbol{Q}^{-1}(\boldsymbol{A}^T\boldsymbol{\lambda} + \boldsymbol{c}) \tag{2.20}$$

Soustava $\boldsymbol{A}\boldsymbol{Q}^{-1}\boldsymbol{A}^T$ z (2.19) má velikost $m \times m$, proto je tato metoda vhodná pokud je málo omezení.
Nevýhodou této metody je, že matice $\boldsymbol{Q}$ musí být invertovatelná.

2.2.2 Metoda nulového prostoru

Metoda nulového prostoru opět řeší rovnice (2.18). Nejprve rozepíšeme proměnnou $\boldsymbol{x}$.

$$\boldsymbol{x} = \boldsymbol{Y}\boldsymbol{x}_y + \boldsymbol{Z}\boldsymbol{x}_z \tag{2.21}$$

Pro $\boldsymbol{Z}$ platí, že $\boldsymbol{A}\boldsymbol{Z} = \boldsymbol{0}$, tedy $\boldsymbol{Z}$ je báze nulového prostoru matice $\boldsymbol{A}$. Dále platí, že $\text{hod}[\boldsymbol{Y}|\boldsymbol{Z}] = n$, kde n je počet řádků $\boldsymbol{A}$.
Dosazením (2.21) do (2.18) a s tím, že $\boldsymbol{A}\boldsymbol{Z} = \boldsymbol{0}$ získáme následující rovnici.

$$\boldsymbol{A}\boldsymbol{Y}\boldsymbol{x}_y = \boldsymbol{b} \tag{2.22}$$

Z této rovnosti můžeme odvodit, že $[\boldsymbol{A}\boldsymbol{Y}|\boldsymbol{Z}] = [\boldsymbol{B}\boldsymbol{Y}|\boldsymbol{0}]$, z čehož plyne, že matice $\boldsymbol{Y}$ musí mít plnou hodnost.
Vztah (2.21) dosadíme do (2.18) a tuto rovnici přenásobíme $\boldsymbol{Z}^T$.
Získáme tím výslednou rovnici.

$$\boldsymbol{Z}^T\boldsymbol{Q}\boldsymbol{Z}\boldsymbol{x}_z = -\boldsymbol{Z}^T\boldsymbol{Q}\boldsymbol{Y}\boldsymbol{x}_y - \boldsymbol{Z}^T\boldsymbol{c} \tag{2.23}$$

Hodnotu $\boldsymbol{x}_y$ známe již z rovnice (2.22), dopočítáme tedy $\boldsymbol{x}_z$ a určíme celkové $\boldsymbol{x}$ ze vztahu (2.21).
Výhodou této metody oproti Schurově metodě je ta, že není třeba počítat inverzi $\boldsymbol{Q}$. Na druhé straně je třeba hledat bázi nulového prostoru $\boldsymbol{Z}$.

2.3 Nerovnostní omezení

Nyní budeme uvažovat úlohy, na které jsou kladena nerovnostní omezení. Pro řešení těchto úloh je nutno použít jiné postupy než pro úlohy bez omezení, protože optimum kvadratické funkce může ležet mimo přípustnou oblast. K vyřešení těchto úloh můžeme použít pouze iterativní metody.
Při řešení úloh s nerovnostními omezeními uvažujeme omezení typu box, což znamená pouze prostá omezení shora a zdola tak jak byla uvedena v (1.24). Uvádím zde několik metod, které patří k nejběžněji používaným metodám pro řešení tohoto typu úloh. Jejich formulace jsou převzaty především z [10], [11], [8].
Připomeňme úlohu, kterou je třeba řešit.

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) := \frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{c}^T\boldsymbol{x} \tag{2.24}$$

Za omezení :

$$\boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}$$

2.3.1 Gradientní projekce

První metodou zabývající se nerovnostními omezeními, uvedenou v této práci, je metoda gradientní projekce. Gradientní projekce je v podstatě variace gradientní metody, která navíc obsahuje projekční krok. Gradientní krok musí mít fixní délku a musí platit $0 < \alpha < \frac{1}{\lambda_{\max}(\boldsymbol{H})}$.
Projekční krok znamená, že v každé iteraci po vypočtení nového vektoru $\boldsymbol{x}$ určíme medián z vektorů dolních omezení, vypočteného vektoru a horních omezení. Tím je zaručeno, že výsledek zůstane uvnitř daných omezení. Tento postup je vhodný pro naše omezení typu box, v případě složitějších omezení by se postup zkomplikoval a projekční krok by již nebyl prostý median.

$$\boldsymbol{x} = \text{median}(\underline{\boldsymbol{x}}, \boldsymbol{x}, \overline{\boldsymbol{x}}) \tag{2.25}$$

Gradientní projekce

$\boldsymbol{x}_0$, ležící v přípustné oblasti, $0 < \alpha < \frac{1}{\lambda_{\max}(\boldsymbol{H})}$
for $i = 0$ to i_{max} **do**
$\boldsymbol{y}_i = \boldsymbol{x}_i - \alpha(\boldsymbol{Q}\boldsymbol{x}_i + \boldsymbol{c})$;
$\boldsymbol{x}_{i+1} = \text{median}(\underline{\boldsymbol{x}}, \boldsymbol{y}_i, \overline{\boldsymbol{x}})$;
end

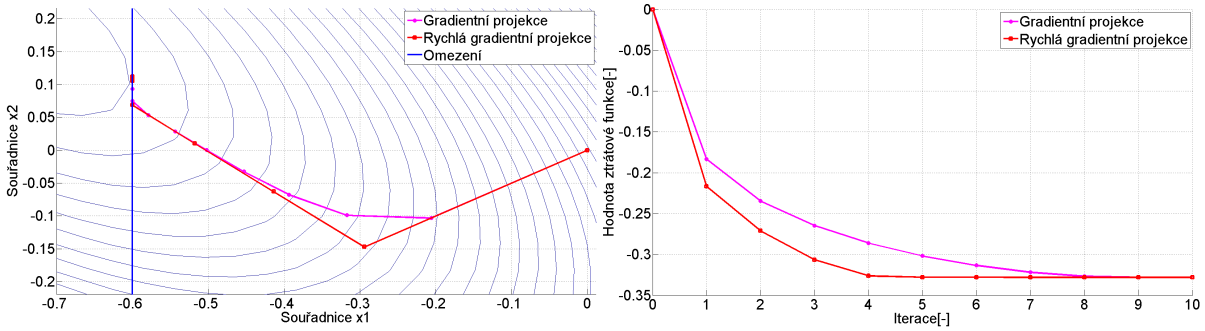
2.3.2 Rychlá gradientní projekce

Rychlá gradientní projekce byla publikována Yuriim Nesterovem v roce 1983, a posléze upravena S.Richterem. Jedná se o iterativní metodu, která využívá projekční krok k aplikaci omezení.
Tato metoda navíc přidává několik výpočetních kroků, s jejichž pomocí konverguje rychleji než obyčejná gradientní projekce.
Uvažujeme QP problém ve tvaru (2.1), na který aplikujeme omezení typu box.
Tento algoritmus je dobře popsán např. v [12]. Následuje algoritmus této metody.

Rychlá gradientní projekce [12]

$L = \lambda_{\max}(\boldsymbol{Q})$ a $\mu = \lambda_{\min}(\boldsymbol{Q})$, $\boldsymbol{y}_0 = \boldsymbol{x}_0$, ležící v přípustné oblasti, $0 < \sqrt{\frac{\mu}{L}} \leq \alpha_0 < 1$
for $i = 0$ to i_{max} **do**
$\boldsymbol{x}_i = \boldsymbol{y}_i - \frac{1}{L}(\boldsymbol{Q}\boldsymbol{y}_i + \boldsymbol{c})$;
$\boldsymbol{x}_{i+1} = \text{median}(\underline{\boldsymbol{x}}, \boldsymbol{x}_i, \overline{\boldsymbol{x}})$;
$\alpha_{i+1} = (-\alpha_i^2 L + \mu + \sqrt{4\alpha_i^2 L^2 + (\alpha_i^2 L - \mu)^2})/2L$;
$\beta_i = \alpha_i(1 - \alpha_i)/(\alpha_i^2 + \alpha_{i+1})$;
$\boldsymbol{y}_{i+1} = \boldsymbol{x}_{i+1} + \beta_i(\boldsymbol{x}_{i+1} - \boldsymbol{x}_i)$;
if Zkontroluj KKT podmínky
break
end
end

Dále uvádím srovnání předchozích dvou metod, jsou porovnávány na stejném problému jako metody pro neomezenou optimalizaci, ale je přidáno spodní omezení.



Obrázek 2.4. Srovnání metody gradientní projekce a metody Rychlé gradientní projekce

2.3.3 Metoda vnitřního bodu

Metoda vnitřního bodu je iterativní metodou, která převádí problém s nerovnostními omezeními na sérii problémů s rovnostními omezeními.
Hlavní myšlenkou této metody je zahrnutí nerovnostních omezení do ztrátové funkce pomocí bariérové funkce ϕ, která je přičtena ke ztrátové funkci. Uvažujme následující problém.

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) := \frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{c}^T\boldsymbol{x} \tag{2.26}$$

Za omezení :

$$\underline{\boldsymbol{x}} - \boldsymbol{x} \leq \boldsymbol{0}$$
$$\boldsymbol{x} - \overline{\boldsymbol{x}} \leq \boldsymbol{0}$$
$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$$

Parametr m je počet nerovnostních omezení.
Jako funkci ϕ volíme sumu logaritmických funkcí pro jednotlivá omezení.

$$\phi = -(\log(-\underline{\boldsymbol{x}} + \boldsymbol{x}) + \log(-\boldsymbol{x} + \overline{\boldsymbol{x}}))$$

Úloha optimalizace, kterou bude třeba v každém kroce vyřešit je poté následující.

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) := \ t(\frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{c}^T\boldsymbol{x}) + \phi \tag{2.27}$$

Za omezení :

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$$

Parametr $t > 0$ je takzvaný bariérový parametr, který určuje váhu členu $\frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{c}^T\boldsymbol{x}$ oproti členu ϕ. Čím menší bude parametr t, tím významější bude funkční hodnota ϕ, a tím více bude řešení odpuzováno od omezení. Se zvyšujícím se parametrem t bude výsledek rovnice (2.27) stále méně ovlivněn členem ϕ.
Výsledný algoritmus poté začíná s malou hodnotou t ze středu omezení a se zvyšujícím se t je mu umožněno přiblížit se omezením.
Pro řešení úlohy (2.27) lze využít například Newtonovu metodu.

Metoda vnitřního bodu [13]

$\boldsymbol{x}_0$, ležící v přípustné oblasti, $t_0 > 0$, $\mu > 0$, *tolerance* $\epsilon > 0$

for $i = 0$ to i_{max} **do**
  Spočti $\boldsymbol{x}$ z (2.27)
  if $m/t < \epsilon$
    break
  end
  $t = \mu t$
end

Symbol m značí počet omezení a parametr ϵ vyjadřuje toleranci nepřesnosti řešení.
V tomto algoritmu je nejprve vyřešena minimalizační úloha s rovnostními omezeními. Poté je provedena kontrola, zda-li byla dosažena požadovaná přesnost. Pokud nebyla dosažena, je t zvětšeno μ-krát a algoritmus pokračuje další iterací.
Detailnější popis této metody a jejího algoritmu lze nalézt např. v [13].

2.3.4 Metoda aktivních množin

Tato metoda má opět za úkol převod problému s nerovnostními omezeními na sérii problémů s rovnostními omezeními.
Metoda pracuje s takzvanou *pracovní množinou*. Cílem je najít *množinu aktivních omezení*, což je množina těch omezení, která jsou aktivní pro výsledné řešení. V každém kroce jsou do pracovní množiny přidána omezení, nebo je z ní jedno omezení odebráno. Každé omezení v pracovní množině omezuje volnost výběru dalšího kroku.
Proměnnou, kterou zde minimalizujeme je $\Delta\boldsymbol{x}$.

$$\Delta\boldsymbol{x} = \boldsymbol{x} - \boldsymbol{x}_i$$

Algoritmus metody aktivních množin [8]

$\boldsymbol{x}_0$, ležící v přípustné oblasti, počáteční pracovní množina $\boldsymbol{W}_0$.

for $i = 0$ to i_{max} **do**
  Najdi $\Delta\boldsymbol{x}$ z (2.28) za omezení (2.29).
  if $\Delta\boldsymbol{x} = 0$
    if $\lambda_j \geq 0, \lambda_j \in \boldsymbol{W}_i$,
      $\boldsymbol{x}^* = \boldsymbol{x}_i$
      break
    else
      Z pracovní množiny odeber omezení, pro které je λ nejmenší.
      $\boldsymbol{x}_{i+1} = \boldsymbol{x}_i$
    end
  else

$\boldsymbol{x}_{i+1} = \alpha_i \Delta \boldsymbol{x}_i$, kde parametr α_i je spočten z (2.30).
Pokud jsou nějaká blokující omezení, přidej to nejbližší do pracovní množiny.

end

end

V každé iteraci je vypočtena následující úloha pro krok $\Delta \boldsymbol{x}$.

$$\min_{\Delta \boldsymbol{x}_i} \frac{1}{2} \Delta \boldsymbol{x}_i{}^T \boldsymbol{Q} \Delta \boldsymbol{x}_i + \Delta \boldsymbol{x}_i{}^T \nabla f(\boldsymbol{x}_i) \tag{2.28}$$

Za omezení :

$$\boldsymbol{A}_i \Delta \boldsymbol{x}_i = \boldsymbol{0} \tag{2.29}$$

Matice $\boldsymbol{A}_i$ reprezentuje aktuální omezení z pracovní množiny.
Problém (2.28) za omezení (2.29) můžeme řešit pomocí Lagrangeových multiplikátorů, respektive metod nulového prostoru a Schurovy metody, které již byly zmíněny v příslušné kapitole o rovnostních omezeních.
Pokud platí $\Delta \boldsymbol{x} \neq \boldsymbol{0}$, je třeba zkontrolovat, zda uskutečněním tohoto kroku nebudou porušena některá omezení mimo pracovní množinu, resp. je třeba spočítat parametr α_i, $\alpha_i \in (0, 1]$, který udává největší možnou velikost kroku. Výsledný krok tedy bude mít následující podobu

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \alpha_i \Delta \boldsymbol{x}$$

.
Koeficient α_i spočteme ze vzorce

$$\alpha_i \leq \frac{b_j - \boldsymbol{A}_j \boldsymbol{x}_i}{\boldsymbol{A}_{ij} \Delta \boldsymbol{x}}, \tag{2.30}$$

kde b_j jsou prvky vektoru omezení a $\boldsymbol{A}_{ij}$ je příslušná část (příslušné řádky) matice omezení z (2.29), důkaz vzorce (2.30) lze nalézt v [8]. Ze všech možných takto spočtených koeficientů vybereme ten nejmenší.
Pro jednoduchá omezení typu box, která v této práci uvažujeme je možné tento vzorec zjednodušit.

$$\alpha_i \leq \frac{b_j - x_{ij}}{\Delta x_j}, \tag{2.31}$$

kde j vybírá příslušný prvek vektoru $\boldsymbol{x}_i$, resp. $\Delta \boldsymbol{x}$.

Pokud je $\alpha_i < 1$, znamená to, že algoritmus narazil na takzvané blokující omezení, tedy na některé omezení, které není v pracovní množině, a je nutno toto blokující omezení přidat do pracovní množiny.

Pokud platí $\Delta \boldsymbol{x} = \boldsymbol{0}$, a zároveň $\lambda_j \geq 0, j \in \boldsymbol{W}_i$, kde λ_j je příslušný Lagrangeův multiplikátor, který jsme mohli obdržet například při výpočtu kroku $\Delta \boldsymbol{x}$, našli jsme výsledné řešení a aktuální pracovní množina je množinou aktivní.
Pokud podmínka $\lambda_j \geq 0, j \in \boldsymbol{W}_i$ neplatí, je třeba z pracovní množiny odebrat jedno omezení a pokračovat další iterací.
Obecně lze použít pro řešení jednotlivých kroků algoritmu různé postupy nebo metody uvedené výše.

2.3.5 Metoda kombinované gradient/Newtonovy projekce

V této sekci je popsána metoda kombinované gradient/Newtonovy projekce, která je odvozena od metody aktivních množin.
Důležitou vlastností tohoto algoritmu je, že uvažuje omezení pouze na vstupy, tvrdá omezení na stavy totiž mohou způsobit neřešitelnost Newtonova kroku.

gradient/Newtonova projekce [14]

$\boldsymbol{u}_0$ ležící v přípustné oblasti, délka gradientního kroku $\alpha \in (0, \frac{1}{\lambda_{\max}(\boldsymbol{H})})$.
for $i = 0$ to i_{max} **do**
Spočti Newtonův krok $\Delta\boldsymbol{u}$ pro neaktivní omezení.
Spočti vektor $\boldsymbol{t}$.

$t_j = \frac{(\boldsymbol{u}_j - \overline{\boldsymbol{u}}_j)}{\Delta \boldsymbol{u}_j}$, pro $\Delta \mathbf{u}_\mathrm{j} < 0$
$t_j = \frac{(\boldsymbol{u}_j - \underline{\boldsymbol{u}}_j)}{\Delta \boldsymbol{u}_j}$, pro $\Delta \mathbf{u}_\mathrm{j} > 0$
$t_j = \infty$, jinak

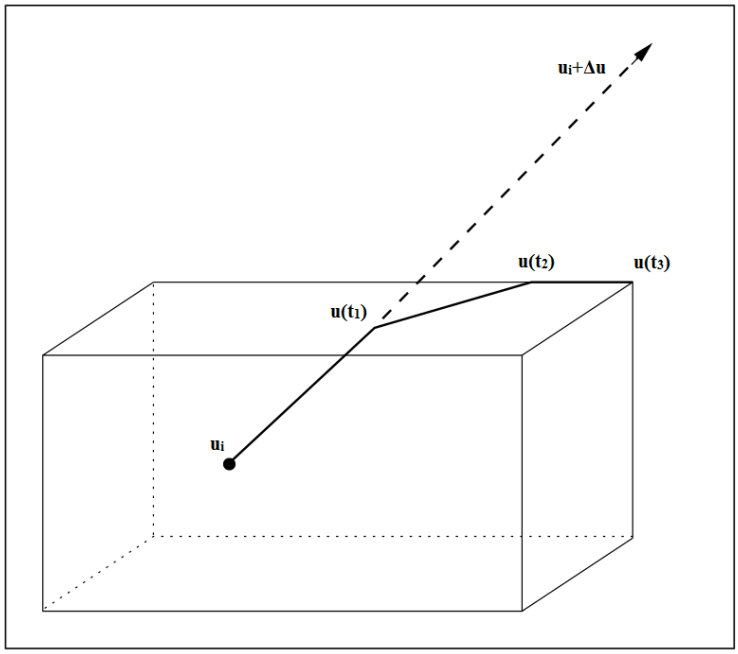
Spočti maximální délku kroku s.
$s = \min(1, \min(\boldsymbol{t}))$
$\boldsymbol{u}_a = \boldsymbol{u}_i + s\Delta\boldsymbol{u}$
if $s == 1$
$\nabla f(\boldsymbol{u}_a) = \boldsymbol{Q}\boldsymbol{u}_a + \boldsymbol{c}$
if Zkonroluj KKT podmínky
$\boldsymbol{u}^* = \boldsymbol{u}_a$.
break
else
$\boldsymbol{u}_{i+1} = P(\boldsymbol{u}_i - \alpha\nabla f(\boldsymbol{u}_a))$
end
else
$\boldsymbol{u}_{i+1} = P(\boldsymbol{u}_a + \beta(1-s)\Delta\boldsymbol{u})$
end
end

V hlavním cyklu je nejprve pomocí Newtonova kroku nalezeno optimum na podprostoru neaktivních omezení. Možnosti výpočtu tohoto kroku jsou uvedeny v dalším textu.
Dále je vypočten vektor koeficientů $\boldsymbol{t}$, který udává maximální délku kroku $\Delta\boldsymbol{u}$ pro jednotlivé souřadnice.
Poté je vypočten koeficient s

$$s = \min(1, \min(\boldsymbol{t})).$$

Následně je Newtonův krok aplikován až k prvnímu nejbližšímu omezení. V případě, že algoritmus na žádné omezení nenarazil, je možné, že nalezl optimum. Tento bod je však optimem pouze za předpokladu, že je aktuální pracovní množina finální množinou aktivních omezení (viz 2.3.4). To zjistíme pomocí testování KKT podmínek. V našem případě to znamená, že všechny Lagrangeovy multiplikátory jsou nezáporné.

Pokud KKT podmínky nejsou splněny, je aplikován gradientní krok, využívající metodu gradientní projekce. Algoritmus gradientní projekce byl popsán v sekci 2.3.1. Provádíme pouze jednu iteraci výpočtu, protože smyslem tohoto kroku není přímé hledání optima, ale pouze úprava pracovní množiny. Po provedení projekčního kroku tuto pracovní množinu upravíme dle toho, která omezení jsou aktivní.
Pokud algoritmus narazil na omezení, je nutné provést projekci Newtonova kroku, abychom přidali aktivní omezení do pracovní množiny. Projekce Newtonova kroku je provedena pomocí algoritmu *exact line search* popsaného v [8]. Jedná se o iterativní algoritmus, který hledá po částech lineární cestu k minimu. Pro ilustraci tohoto algoritmu uvádím následující obrázek převzatý z [8].



Obrázek 2.5. Funkce algoritmu Newtonova projekčního kroku

V tomto kroce je projektován dříve vypočtený Newtonův krok $\Delta \boldsymbol{u}$ na omezení. K tomu je využit vektor maximálních délek $\boldsymbol{t}$. Vektor $\boldsymbol{t}$ je v tomto případě seřazen od nejmenších hodnot po největší, tedy t_1 je nejmenší z prvků vektoru $\boldsymbol{t}$.

Kapitola 3
Výpočet Newtonova kroku

Způsob výpočtu a optimalizace Newtonova kroku je hlavní náplní této práce. Jsou zde uvedeny dva způsoby formulace tohoto kroku. V obou případech je použito Lagrangeových multiplikátorů a metody nulového prostoru pro aplikaci omezení. Tato omezení jsou, jak již bylo uvedeno, pouze jednoduchá omezení shora a zdola a jsou aplikována jen na vstupy.
V Newtonově kroku hledáme posun $\Delta \boldsymbol{u}$,

$$\boldsymbol{u} = \boldsymbol{u}_i + \Delta \boldsymbol{u}. \tag{3.1}$$

V případě prvního způsobu je využito kondenzovaného QP, tento způsob je v současné době velmi rozšířen a dlouhodobě se používá.
Druhým způsobem je Rychlý Newtonův krok, v této formulaci se problém skládá z většího počtu rovnic, ovšem zároveň je využita specifická podoba matic, a díky tomu je významným způsobem urychlen výpočet v porovnání s kondenzovanou formulací, především pro problémy s dlouhým predikčním horizontem.
Dále je v této práci uvedena paralelizace Rychlého Newtonova kroku. Problém je přeformulován tak, že vzroste dimenze problému, nicméně jeho struktura se stane vhodnou pro paralelizaci.

3.1 Newtonův krok pro kondenzované QP

Smyslem kondenzované formulace je v tomto případě eliminovat stavy, čímž je problém převeden na optimalizaci vstupů. V kondenzované formulaci, která je popsána v sekci 1.6 je přepsán následovně.

$$\min_{\Delta \boldsymbol{u}} J := \frac{1}{2} \Delta \boldsymbol{u}^T \boldsymbol{H} \Delta \boldsymbol{u} + \boldsymbol{f}^T \Delta \boldsymbol{u} \tag{3.2}$$

Za podmínky :

$$\underline{\boldsymbol{u}} \leq \boldsymbol{u} \leq \overline{\boldsymbol{u}},$$

kde $\boldsymbol{f}^T = {\boldsymbol{x}_0}^T \boldsymbol{P}^T \widehat{\boldsymbol{Q}} \boldsymbol{V}$ a $\boldsymbol{H} = \boldsymbol{V}^T \widehat{\boldsymbol{Q}} \boldsymbol{V} + \widehat{\boldsymbol{R}}$.

Dosadíme-li nyní z (3.1) do (3.2), dostaneme následující vztah.

$$\min_{\Delta \boldsymbol{u}} J := \frac{1}{2} \Delta \boldsymbol{u}^T \boldsymbol{H} \Delta \boldsymbol{u} + (\boldsymbol{f}^T + \boldsymbol{u}_i^T \boldsymbol{H}) \Delta \boldsymbol{u} + (\frac{1}{2} \boldsymbol{u}_i^T \boldsymbol{H} + \boldsymbol{f}^T) \boldsymbol{u}_i \tag{3.3}$$

Za podmínky :

$$\boldsymbol{F} \Delta \boldsymbol{u} = \boldsymbol{0}$$

Matice $\boldsymbol{F}$ je matice udávající, které vstupy leží na omezení. Členy rovnice (3.3) neobsahující $\Delta \boldsymbol{u}$ jsou v této iteraci konstantní, a proto při minimalizaci nehrají roli. Dále je tedy nebudeme uvažovat.

Zavedeme novou proměnnou

$$\boldsymbol{g}_i^T = \boldsymbol{f}^T + \boldsymbol{u}_i^T \boldsymbol{H}.$$

Použitím Lagrangeových multiplikátorů 2.2 vytvoříme Lagrangeovu funkci

$$\boldsymbol{L}(\Delta \boldsymbol{u}, \boldsymbol{\mu}) := \frac{1}{2}\Delta \boldsymbol{u}^T \boldsymbol{H} \Delta \boldsymbol{u} + \boldsymbol{g}_i^T \Delta \boldsymbol{u} + \boldsymbol{\mu}^T \boldsymbol{F} \Delta \boldsymbol{u}.$$

Nyní je třeba vyřešit následující soustavu rovnic.

$$\begin{pmatrix} \boldsymbol{H} & \boldsymbol{F}^T \\ \boldsymbol{F} & \boldsymbol{0} \end{pmatrix} \begin{pmatrix} \Delta \boldsymbol{u} \\ \boldsymbol{\mu} \end{pmatrix} = \begin{pmatrix} -\boldsymbol{g}_i \\ \boldsymbol{0} \end{pmatrix} \tag{3.4}$$

Následně je možné tuto soustavu řešit pomocí metody nulového prostoru, popsané v 2.2.2.
Nejprve rozepíšeme $\Delta \boldsymbol{u}$.

$$\Delta \boldsymbol{u} = \boldsymbol{Y} \Delta \boldsymbol{u}_y + \boldsymbol{Z} \Delta \boldsymbol{u}_z \tag{3.5}$$

Dosazením (3.5) do (3.4) a přenásobením $\boldsymbol{Z}^T$ získáme výslednou rovnici.

$$\boldsymbol{Z}^T \boldsymbol{H} \boldsymbol{Z} \Delta \boldsymbol{u}_z = -\boldsymbol{Z}^T \boldsymbol{g}_i \tag{3.6}$$

Tato rovnice je podobná výsledné rovnici obecného řešení problému v kondenzované formulaci, s tím rozdílem, že tato rovnice je upravena pomocí matic $\boldsymbol{Z}$, resp. $\boldsymbol{Z}^T$, které vybírají pouze řádky, resp. sloupce příslušné neaktivním omezením pro danou iteraci.
Pro získání plného vektoru použijeme vztah

$$\Delta \boldsymbol{u} = \boldsymbol{Z} \Delta \boldsymbol{u}_z.$$

V tomto případě vynásobení maticí $\boldsymbol{Z}$ doplní vypočtený vektor o nuly v souřadnicích, které jsou na omezení.

3.2 Rychlý Newtonův krok

Tento způsob výpočtu vychází opět z obecné formulace QP problému. Znovu je použita substituce za $\boldsymbol{u}$

$$\boldsymbol{u} = \boldsymbol{u}_i + \Delta \boldsymbol{u}.$$

Problém vypadá následovně

$$\begin{aligned} \min_{\boldsymbol{x}, \Delta \boldsymbol{u}} J &:= \frac{1}{2}\boldsymbol{x}^T \widehat{\boldsymbol{Q}} \boldsymbol{x} + \frac{1}{2}\Delta \boldsymbol{u}^T \widehat{\boldsymbol{R}} \Delta \boldsymbol{u} + \Delta \boldsymbol{u}^T \widehat{\boldsymbol{R}} \boldsymbol{u}_i \\ \text{Za omezení :} & \\ \widehat{\boldsymbol{B}} \Delta \boldsymbol{u} + \widehat{\boldsymbol{A}} \boldsymbol{x} &= \boldsymbol{q} - \widehat{\boldsymbol{B}} \boldsymbol{u}_i \\ \boldsymbol{F} \Delta \boldsymbol{u} &= \boldsymbol{0}. \end{aligned} \tag{3.7}$$

Matice $\widehat{\boldsymbol{Q}}$ a $\widehat{\boldsymbol{R}}$ představují rozšířené váhové matice, tak jak byly popsány v 1.5 a 1.6 .
V této formulaci bylo použito posunutí v proměnné $\boldsymbol{u}$, avšak proměnná $\boldsymbol{x}$ zůstala ve stejné podobě. Je to proto, že Newtonův krok provádíme pouze v proměnné $\boldsymbol{u}$, a hodnotu stavů pro jeho výpočet znát nemusíme.

Protože v této formulaci jsou opět kladena pouze rovnostní omezení na vstupy, použijeme Lagrangeovy multiplikátory

$$\boldsymbol{L}(\boldsymbol{\lambda},\boldsymbol{\mu}) := \frac{1}{2}\boldsymbol{x}^T\widehat{\boldsymbol{Q}}\boldsymbol{x} + \frac{1}{2}\Delta\boldsymbol{u}^T\widehat{\boldsymbol{R}}\Delta\boldsymbol{u} + \Delta\boldsymbol{u}^T\widehat{\boldsymbol{R}}\boldsymbol{u}_i + \boldsymbol{\lambda}^T(\widehat{\boldsymbol{B}}\Delta\boldsymbol{u} + \widehat{\boldsymbol{A}}\boldsymbol{x} - \boldsymbol{q} + \widehat{\boldsymbol{B}}\boldsymbol{u}_i) + \boldsymbol{\mu}^T(\boldsymbol{F}\Delta\boldsymbol{u}).$$

KKT podmínky problému (3.7) tedy budou vypadat následovně

$$\begin{pmatrix} \widehat{\boldsymbol{R}} & & \widehat{\boldsymbol{B}}^T & \boldsymbol{F}^T \\ & \widehat{\boldsymbol{Q}} & \widehat{\boldsymbol{A}}^T & \\ \widehat{\boldsymbol{B}} & \widehat{\boldsymbol{A}} & & \\ \boldsymbol{F} & & & \end{pmatrix} \begin{pmatrix} \Delta\boldsymbol{u} \\ \boldsymbol{x} \\ \boldsymbol{\lambda} \\ \boldsymbol{\mu} \end{pmatrix} = \begin{pmatrix} -\widehat{\boldsymbol{R}}\boldsymbol{u}_i \\ \boldsymbol{0} \\ \boldsymbol{q} - \widehat{\boldsymbol{B}}\boldsymbol{u}_i \\ \boldsymbol{0} \end{pmatrix}. \tag{3.8}$$

Dále přistoupíme k metodě nulového prostoru (viz 2.2.2). Obdobně jako v předchozím případě se tím zmenší problém o jednu maticovou rovnici.
Soustava rovnic poté vypadá následovně

$$\begin{pmatrix} \boldsymbol{\Psi} & & \boldsymbol{\Gamma}^T \\ & \widehat{\boldsymbol{Q}} & \widehat{\boldsymbol{A}}^T \\ \boldsymbol{\Gamma} & \widehat{\boldsymbol{A}} & \end{pmatrix} \begin{pmatrix} \Delta\boldsymbol{u}_z \\ \boldsymbol{x} \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \boldsymbol{c}_i \\ \boldsymbol{0} \\ \boldsymbol{q} - \widehat{\boldsymbol{B}}\boldsymbol{u}_i \end{pmatrix}, \tag{3.9}$$

kde platí $\boldsymbol{\Psi} = \boldsymbol{Z}^T\widehat{\boldsymbol{R}}\boldsymbol{Z}$, $\boldsymbol{\Gamma} = \widehat{\boldsymbol{B}}\boldsymbol{Z}$ a $\boldsymbol{c}_i = -\boldsymbol{Z}^T\widehat{\boldsymbol{R}}\boldsymbol{u}_i$.

Po Gaussově eliminaci je možné rovnici (3.9) zapsat následovně

$$\begin{pmatrix} \boldsymbol{\Psi} & & \boldsymbol{\Gamma}^T \\ & \widehat{\boldsymbol{Q}} & \widehat{\boldsymbol{A}}^T \\ & & \boldsymbol{G} \end{pmatrix} \begin{pmatrix} \Delta\boldsymbol{u}_z \\ \boldsymbol{x} \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \boldsymbol{c}_i \\ \boldsymbol{0} \\ \boldsymbol{v} \end{pmatrix}. \tag{3.10}$$

Soustava rovnic pro výpočet $\Delta\boldsymbol{u}_z$ je poté

$$\boldsymbol{G}\boldsymbol{\lambda} = \boldsymbol{v} \tag{3.11}$$

$$\boldsymbol{\Psi}\Delta\boldsymbol{u}_z = (\boldsymbol{c}_i - \boldsymbol{\Gamma}^T\boldsymbol{\lambda}), \tag{3.12}$$

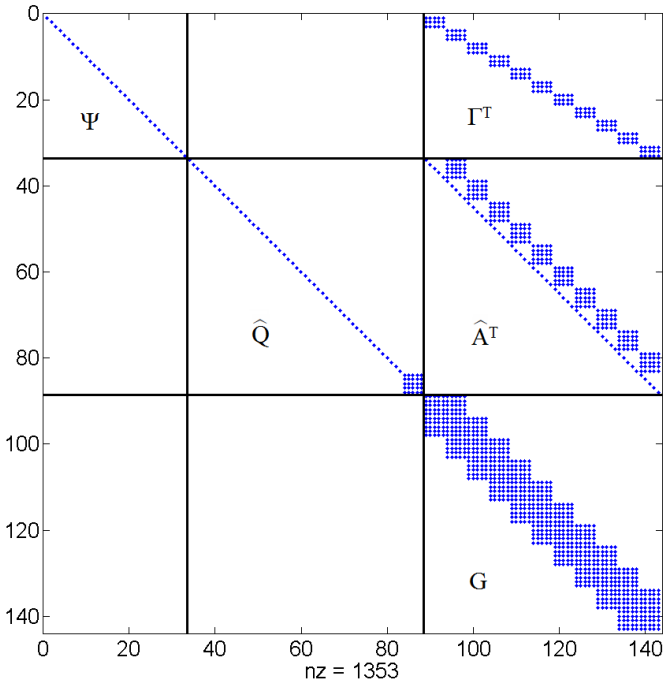
kde $\boldsymbol{G} = \boldsymbol{\Gamma}\boldsymbol{\Psi}^{-1}\boldsymbol{\Gamma}^T + \widehat{\boldsymbol{A}}\widehat{\boldsymbol{Q}}^{-1}\widehat{\boldsymbol{A}}^T$ a $\boldsymbol{v} = -\boldsymbol{q} + \widehat{\boldsymbol{B}}\boldsymbol{u}_i + \boldsymbol{\Gamma}\boldsymbol{\Psi}^{-1}\boldsymbol{c}_i$. Je nutné, aby $\widehat{\boldsymbol{Q}}$ bylo invertovatelné.

Výsledné $\Delta\boldsymbol{u}_z$ rozšíříme obdobně jako v případě kondenzované formulace

$$\Delta\boldsymbol{u} = \boldsymbol{Z}\Delta\boldsymbol{u}_z.$$

Struktura problému je znázorněna na obr. 3.1 jako jedna velká matice, skládající se z jednotlivých částí. Modré body představují nenulové prvky, na osách jsou vyznačeny rozměry matice.

Z obrázku 3.1 je vidět, že celý problém je velmi řídký, tedy obsahuje málo nenulových prvků, v porovnání s celkovým počtem prvků.
Matice $\boldsymbol{\Psi}$ je matice $\widehat{\boldsymbol{R}}$ s odebranými sloupci a řádky, zůstává diagonální. Matice $\widehat{\boldsymbol{Q}}$ je téměř diagonální, až na poslední člen, odpovídající matici $\boldsymbol{Q}_N$. Matice $\boldsymbol{\Gamma}$ je blokově diagonální, jedná se o matici $\widehat{\boldsymbol{B}}$ s odebranými sloupci.



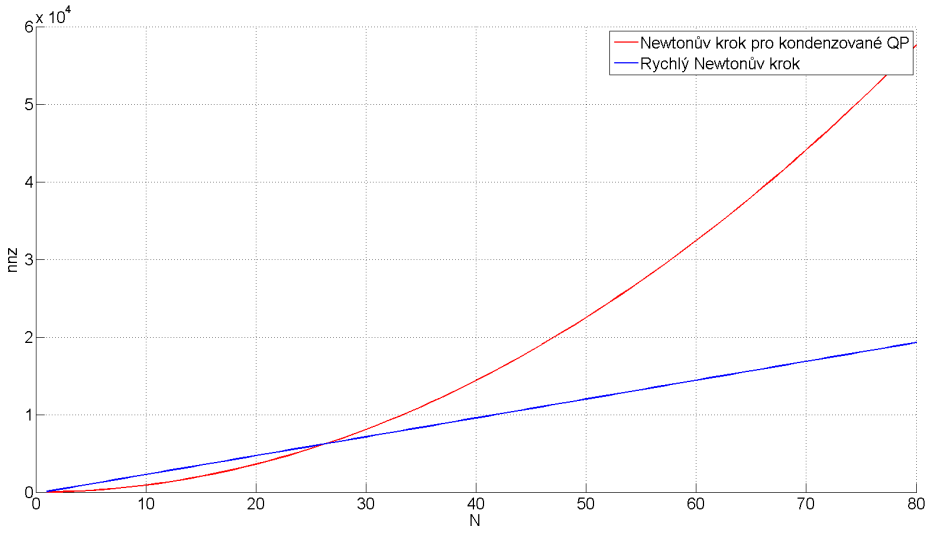
Obrázek 3.1. Struktura problému v Rychlé formulaci pro $n_x = 5$, $n_u = 3$, $N = 11$ a žádné aktivní omezení

Na diagonále matice $\widehat{\boldsymbol{A}}$ je záporně vzatá jednotková matice, pod hlavní diagonálou jsou bloky tvořené maticemi $\boldsymbol{A}$.
Pokud bychom uvažovali tvrdé omezení na stavy, právě v tomto kroce by mohlo dojít k tomu, že systém ztratí plnou hodnost v důsledku úpravy matic.
Zatímco matice $\boldsymbol{H}$ měla rozměr nejvýše $Nn_u \times Nn_u$, matice $\boldsymbol{G}$ má rozměr nejvýše $Nn_x \times Nn_x$, ovšem hlavní výhodou tohoto přístupu je fakt, že díky tridiagonální struktuře matice $\boldsymbol{G}$ nyní počet nenulových prvků této matice roste pouze lineárně, zatímco v případě matice $\boldsymbol{H}$ roste kvadraticky.

Výše uvedený fakt je demonstrován na obr. 3.2 na příkladu systému, kde $n_x = 9, n_u = 3$. Je vidět že pro systém s více stavy než-li vstupy, což je častý případ, je pro malý predikční horizont výhodnější kondenzovaná formulace. Ovšem s narůstající délkou predikčního horizontu se stává výhodnější Rychlý Newtonův krok.

3.2.1 Paralelizace výpočtu

Paralelní výpočet Newtonova kroku vychází z předchozí metody Rychlého Newtonova kroku. Smyslem této paralelizace je možnost rozdělit výpočetně náročnou část výpočtu na několik menších problémů, jejichž výpočet bude probíhat ve více procesech paralelně a tím urychlit celý výpočet Newtonova kroku.
Způsob paralelizace uvedený v této práci není jedinou možností paralelizace. Jiný postup je uveden například v [15].
Kritickou částí, tedy výpočetně náročnou je v tomto případě výpočet Choleskyho rozkladu matice $\boldsymbol{G}$ v rovnici (3.11). Jak již bylo zmíněno, matice $\boldsymbol{G}$ je blokově tridiagonální a smyslem paralelizace bude upravit tuto matici na blokově diagonální tvar, protože výpočet bude poté možné lehce paralelizovat.



Obrázek 3.2. Porovnání počtu nenulových prvků

Připomeňme, čemu je rovna matice $\boldsymbol{G}$.

$$\boldsymbol{G} = \boldsymbol{\Gamma}\boldsymbol{\Psi}^{-1}\boldsymbol{\Gamma}^T + \widehat{\boldsymbol{A}}\widehat{\boldsymbol{Q}}^{-1}\widehat{\boldsymbol{A}}^T$$

V této rovnici je první člen $\boldsymbol{\Gamma}\boldsymbol{\Psi}^{-1}\boldsymbol{\Gamma}^T$ blokově diagonální a člen $\widehat{\boldsymbol{A}}\widehat{\boldsymbol{Q}}^{-1}\widehat{\boldsymbol{A}}^T$ je blokově tridiagonální, a zároveň činí i matici $\boldsymbol{G}$ tridiagonální. Úkolem je tedy upravit matici $\widehat{\boldsymbol{A}}$ tak, aby $\boldsymbol{G}$ byla pouze blokově diagonální. Jednotlivé bloky $\boldsymbol{G}$ budou stále tridiagonální. Zavedeme nový vektor $\boldsymbol{p}$. Vektor $\boldsymbol{p}$ se skládá z vybraných prvků vektoru $\boldsymbol{x}$. Vybíráme je vhodně podle toho, jak moc chceme problém paralelizovat. Tyto prvky jsou stavové vektory v jednotlivých krocích řízení,

$$\boldsymbol{p} = \begin{pmatrix} \boldsymbol{x}_{j_1} \\ \boldsymbol{x}_{j_2} \\ \vdots \\ \boldsymbol{x}_{j_m} \end{pmatrix}, \text{pro } 0 < j_1 < j_2 < \ldots < j_m \leq N. \tag{3.13}$$

Matice $\widehat{\boldsymbol{A}}$ má následující podobu,

$$\widehat{\boldsymbol{A}} = \begin{pmatrix} -\boldsymbol{I} & & & & & \\ \boldsymbol{A} & -\boldsymbol{I} & & & & \\ & \boldsymbol{A} & -\boldsymbol{I} & & & \\ & & \boldsymbol{A} & -\boldsymbol{I} & & \\ & & & \ddots & \ddots & \\ & & & & \boldsymbol{A} & -\boldsymbol{I} \end{pmatrix}. \tag{3.14}$$

Další snahou bude odstranit matici $\widehat{\boldsymbol{A}}$ reformulací ztrátové funkce z (3.7) tak, aby byla umožněna paralelizace.

$$\min_{\boldsymbol{x},\Delta\boldsymbol{u}} J := \frac{1}{2}\boldsymbol{x}^T\widehat{\boldsymbol{Q}}\boldsymbol{x} + \frac{1}{2}\Delta\boldsymbol{u}^T\widehat{\boldsymbol{R}}\Delta\boldsymbol{u} + \Delta\boldsymbol{u}^T\widehat{\boldsymbol{R}}\boldsymbol{u}_i$$

Za podmínek :

$$\widehat{\boldsymbol{B}}\Delta\boldsymbol{u} + \boldsymbol{A}_x\boldsymbol{x} + \boldsymbol{A}_p\boldsymbol{p} = \boldsymbol{q} - \widehat{\boldsymbol{B}}\boldsymbol{u}_i$$

$$\begin{aligned} \boldsymbol{J}\boldsymbol{x} + \boldsymbol{I}\boldsymbol{p} &= \boldsymbol{0} \\ \boldsymbol{F}\Delta\boldsymbol{u} &= \boldsymbol{0} \end{aligned} \tag{3.15}$$

V této formulaci se již matice $\widehat{\boldsymbol{A}}$ nevyskytuje, bylo však zavedeno několik nových matic. Matice $\boldsymbol{A}_x$ je strukturou podobná matici $\widehat{\boldsymbol{A}}$, ovšem pod diagonálou jí chybí některé bloky matice $\boldsymbol{A}$. Tyto bloky chybějí na místech odpovídajících poloze prvků vektoru $\boldsymbol{p}$. Tuto strukturu demonstruje vztah (3.16). Zvolil jsem $N = 6$, a $|\{p_j\}| = 2$,

$$\boldsymbol{A}_x = \begin{pmatrix} -\boldsymbol{I} & & & & & \\ \boldsymbol{A} & -\boldsymbol{I} & & & & \\ & \boldsymbol{A} & -\boldsymbol{I} & & & \\ & & & -\boldsymbol{I} & & \\ & & & \boldsymbol{A} & -\boldsymbol{I} & \\ & & & & & -\boldsymbol{I} \end{pmatrix}. \tag{3.16}$$

Matice $\boldsymbol{A}_x$ je blokově diagonální a počet bloků této matice, n_b je o jedna větší než počet prvků vektoru $\boldsymbol{p}$, dále označeno jako n_p.
Matice $\boldsymbol{A}_p$ se skládá z bloků matice $\boldsymbol{A}$, které jsou na stejných řádcích, na kterých chybí v matici $\boldsymbol{A}_x$. Počet sloupců je stejný jako velikost vektoru $\boldsymbol{p}$.
Dále uvádím příklad možné podoby matice $\boldsymbol{A}_p$ (3.17) pro stejný problém jako v případě matice $\boldsymbol{A}_x$,

$$\boldsymbol{A}_p = \begin{pmatrix} & \\ \boldsymbol{A} & \\ & \\ & \boldsymbol{A} \end{pmatrix}. \tag{3.17}$$

Nově přidaná rovnice $\boldsymbol{J}\boldsymbol{x} + \boldsymbol{I}\boldsymbol{p} = \boldsymbol{0}$ vyjadřuje fakt, že vektor $\boldsymbol{p}$ je složen z vybraných souřadnic vektoru $\boldsymbol{x}$, viz (3.13). Matice $\boldsymbol{I}$ je jednotková matice.
Matice $\boldsymbol{J}$ vybírá příslušné souřadnice vektoru $\boldsymbol{x}$. Pro problém, na kterém byla demonstrována struktura matic $\boldsymbol{A}_x$ a $\boldsymbol{A}_p$ vypadá následovně.

$$J = \begin{pmatrix} \boldsymbol{0} & \boldsymbol{0} & -\boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & -\boldsymbol{I} & \boldsymbol{0} \end{pmatrix}$$

Dalším krokem je zavedení Lagrangeových multiplikátorů.

$$\boldsymbol{L}_{(\boldsymbol{\lambda},\boldsymbol{\mu},\boldsymbol{\phi})} = \frac{1}{2}\boldsymbol{x}^T\widehat{\boldsymbol{Q}}\boldsymbol{x} + \frac{1}{2}\Delta\boldsymbol{u}^T\widehat{\boldsymbol{R}}\Delta\boldsymbol{u} + \Delta\boldsymbol{u}^T\widehat{\boldsymbol{R}}\boldsymbol{u}_i + \boldsymbol{\lambda}^T(\widehat{\boldsymbol{B}}\Delta\boldsymbol{u} + \boldsymbol{A}_x\boldsymbol{x} + \boldsymbol{A}_p\boldsymbol{p} - \boldsymbol{q} + \widehat{\boldsymbol{B}}\boldsymbol{u}_i) + \boldsymbol{\mu}^T(\boldsymbol{F}\Delta\boldsymbol{u}) + \boldsymbol{\phi}^T(\boldsymbol{J}\boldsymbol{x} + \boldsymbol{I}\boldsymbol{p}) \tag{3.18}$$

Poté je využita metoda nulového prostoru, jako v předchozích postupech, čímž je problém zmenšen o poslední rovnici.
Po úpravách vypadá soustava následovně,

$$\begin{pmatrix} \boldsymbol{\Psi} & & \boldsymbol{\Gamma}^T & & \\ & \widehat{\boldsymbol{Q}} & \boldsymbol{A}_x^T & & \boldsymbol{J}^T \\ \boldsymbol{\Gamma} & \boldsymbol{A}_x & & \boldsymbol{A}_p & \\ & & \boldsymbol{A}_p^T & & \boldsymbol{I} \\ & \boldsymbol{J} & & \boldsymbol{I} & \end{pmatrix} \begin{pmatrix} \Delta\boldsymbol{u}_z \\ \boldsymbol{x} \\ \boldsymbol{\lambda} \\ \boldsymbol{p} \\ \boldsymbol{\phi} \end{pmatrix} = \begin{pmatrix} \boldsymbol{c}_i \\ \boldsymbol{0} \\ \boldsymbol{q} - \widehat{\boldsymbol{B}}\boldsymbol{u}_i \\ \boldsymbol{0} \\ \boldsymbol{0} \end{pmatrix}, \tag{3.19}$$

kde platí $\boldsymbol{\Psi} = \boldsymbol{Z}^T\widehat{\boldsymbol{R}}\boldsymbol{Z}$, $\boldsymbol{\Gamma} = \widehat{\boldsymbol{B}}\boldsymbol{Z}$, $\boldsymbol{c}_i = -\boldsymbol{Z}^T\widehat{\boldsymbol{R}}\boldsymbol{u}_i$.
Nyní tuto soustavu upravíme pomocí Gaussovy eliminační metody a získáme tím výslednou soustavu.

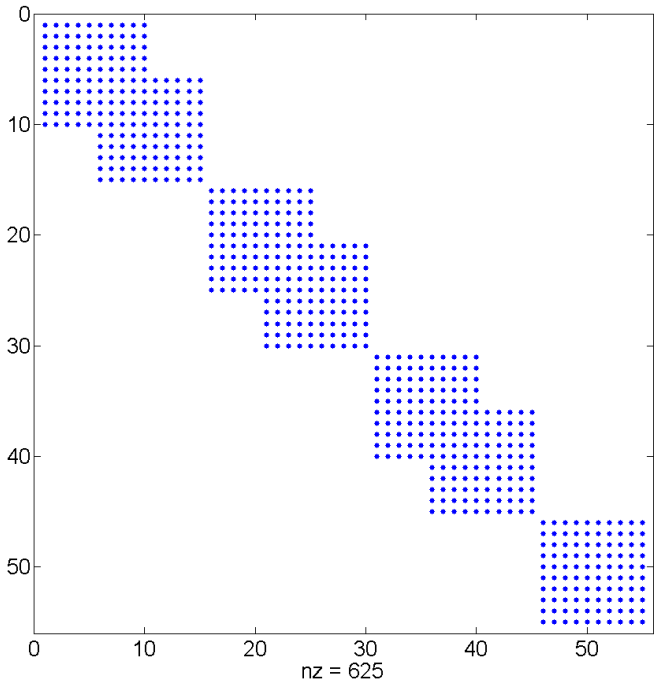
$$\begin{pmatrix} \boldsymbol{\Psi} & & \boldsymbol{\Gamma}^T & & \\ & \widehat{\boldsymbol{Q}} & \boldsymbol{A}_x^T & & \boldsymbol{J}^T \\ & & \boldsymbol{G} & -\boldsymbol{A}_p & -\boldsymbol{V}^T \\ & & & \boldsymbol{\Phi} & \boldsymbol{W}^T \\ & & & & \boldsymbol{M} \end{pmatrix} \begin{pmatrix} \Delta\boldsymbol{u}_z \\ \boldsymbol{x} \\ \boldsymbol{\lambda} \\ \boldsymbol{p} \\ \boldsymbol{\phi} \end{pmatrix} = \begin{pmatrix} \boldsymbol{c}_i \\ \boldsymbol{0} \\ -\boldsymbol{v}_i \\ \boldsymbol{w}_i \\ \boldsymbol{h}_i \end{pmatrix} \tag{3.20}$$

$$\begin{aligned} \boldsymbol{G} &= \boldsymbol{\Gamma}\boldsymbol{\Psi}^{-1}\boldsymbol{\Gamma}^T + \boldsymbol{A}_x\widehat{\boldsymbol{Q}}^{-1}\boldsymbol{A}_x^T & \boldsymbol{\Phi} &= \boldsymbol{A}_p^T\boldsymbol{G}^{-1}\boldsymbol{A}_p & \boldsymbol{V}^T &= -\boldsymbol{A}_x\widehat{\boldsymbol{Q}}^{-1}\boldsymbol{J}^T \\ \boldsymbol{v}_i &= \boldsymbol{r}_i - \boldsymbol{\Gamma}\boldsymbol{\Psi}^{-1}\boldsymbol{c}_i & \boldsymbol{w}_i &= \boldsymbol{A}_p^T\boldsymbol{G}^{-1}\boldsymbol{v}_i & \boldsymbol{h}_i &= -\boldsymbol{V}\boldsymbol{G}^{-1}\boldsymbol{v}_i + \boldsymbol{W}\boldsymbol{\Phi}^{-1}\boldsymbol{w}_i \\ \boldsymbol{W}^T &= \boldsymbol{I} + \boldsymbol{A}_p^T\boldsymbol{G}^{-1}\boldsymbol{V}^T & \boldsymbol{M} &= \boldsymbol{W}\boldsymbol{\Phi}^{-1}\boldsymbol{W}^T - \boldsymbol{V}\boldsymbol{G}^{-1}\boldsymbol{V}^T + \boldsymbol{J}\widehat{\boldsymbol{Q}}^{-1}\boldsymbol{J}^T \end{aligned}$$

Z této maticové rovnosti dopočítáme hodnoty proměnných dle následujících vztahů.

$$\begin{aligned} \boldsymbol{M}\boldsymbol{\phi} &= \boldsymbol{h}_i \\ \boldsymbol{\Phi}\boldsymbol{p} &= (\boldsymbol{w}_i - \boldsymbol{W}^T\boldsymbol{\phi}) \\ \boldsymbol{G}\boldsymbol{\lambda} &= (\boldsymbol{V}^T\boldsymbol{\phi} + \boldsymbol{A}_p\boldsymbol{p} - \boldsymbol{v}_i) \\ \boldsymbol{\Psi}\Delta\boldsymbol{u}_z &= (\boldsymbol{c}_i - \boldsymbol{\Gamma}^T\boldsymbol{\lambda}) \end{aligned}$$

Jak již bylo uvedeno, hlavním přínosem tohoto postupu je úprava struktury matice $\boldsymbol{G}$, která je nyní blokově diagonální.



Obrázek 3.3. Ukázka struktury matice $\boldsymbol{G}$ pro $n_x = 5,\ n_u = 3,\ N = 11$ a žádné aktivní omezení

Na obr. 3.3 je uvedena ukázka možné struktury matice $\boldsymbol{G}$. Tato matice je rozdělena na čtyři bloky, tedy $|\{p_j\}| = 3$.

Bloky je vhodné volit stejně velké, aby výpočetní složitost byla rovnoměrně rozložena.

Na obr. 3.4 je znázorněna celá matice levé strany rovnice (3.20), skládající se z jednotlivých matic, obdobně jako v případě Rychlého Newtonova kroku. Modré body opět představují nenulové prvky, na osách je vyznačen rozměr celého problému.

Matice $\boldsymbol{A}_x^T$ má plnou hodnost, díky plné diagonále, avšak matice $\boldsymbol{A}_p$ plnou hodnost nemá. Obě tyto nové matice jsou řídké.

Matice $\boldsymbol{J}$, která vybírá prvky $\boldsymbol{x}$, ze kterých se skládá vektor $\boldsymbol{p}$, je složena z jednotkových matic.

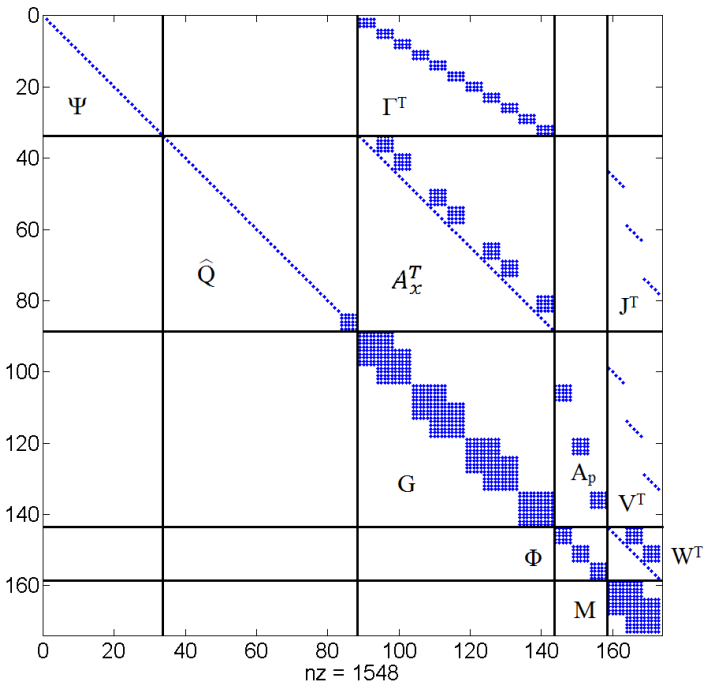
Matice $\boldsymbol{V}^T$ se skládá z diagonálních matic, což vychází z jejího předpisu $\boldsymbol{V}^T = -\boldsymbol{A}_x\widehat{\boldsymbol{Q}}^{-1}\boldsymbol{J}^T$.

Nakonec je zde několik matic, které se vyskytují v rovnicích příslušejících novým proměnným $\boldsymbol{p}$ a $\boldsymbol{\phi}$.

Matice $\boldsymbol{\Phi}$ je blokově diagonální, počet jejích bloků je o jedna menší než počet bloků matice $\boldsymbol{G}$. Struktura matice $\boldsymbol{W}^T$ je podobná struktuře matice $\boldsymbol{A}_x^T$, opět má nenulové prvky na diagonále, a nad diagonálou bloky nenulových prvků. Počet těchto bloků je závislý na počtu bloků matice $\boldsymbol{G}$. Je jich o dva méně než v matici $\boldsymbol{G}$.

Poslední maticí je matice $\boldsymbol{M}$, která je blokově tridiagonální. Její velikost opět závisí na počtu bloků matice $\boldsymbol{G}$. Čím více bloků bude tuto matici tvořit, tím větší bude matice $\boldsymbol{M}$.

Ze struktury matic tedy plyne, že zvyšování počtu bloků matice $\boldsymbol{G}$ sice vede k lepším možnostem paralelizace této matice, ale na druhé straně se tím zvětšuje velikost matic $\boldsymbol{M}$ a $\boldsymbol{\Phi}$.



Obrázek 3.4. Struktura paralelizovaného Newtonova kroku pro $n_x = 5$, $n_u = 3$, $N = 11$, $|\{p_j\}| = 3$ a žádné aktivní omezení

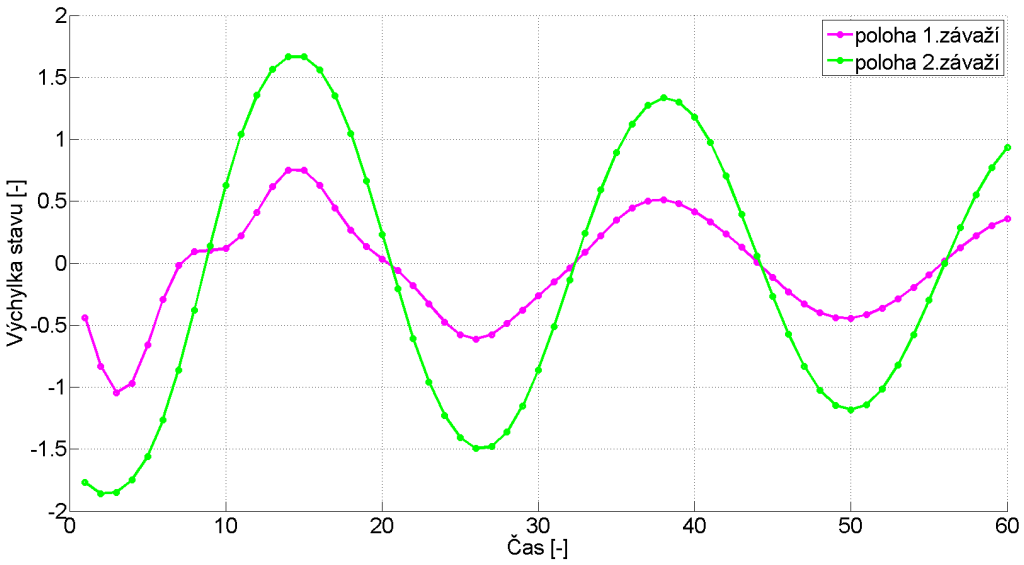
Hlavním omezením tohoto algoritmu je nemožnost omezit stavy pomocí tvrdých omezení. Tento nedostatek se dá částečně napravit přidáním měkkých omezení na stavy.

Kapitola 4
Porovnání efektivity

Algoritmus metody kombinované gradient/Newtonovy projekce s využitím paralelizovaného Newtonova kroku jsem implementoval ve vývojovém prostředí MATLAB. V této kapitole je nejprve demonstrována funkčnost regulátoru, poté je uvedena teoretická úvaha o zrychlení výpočtu, následovaná numerickým testem.

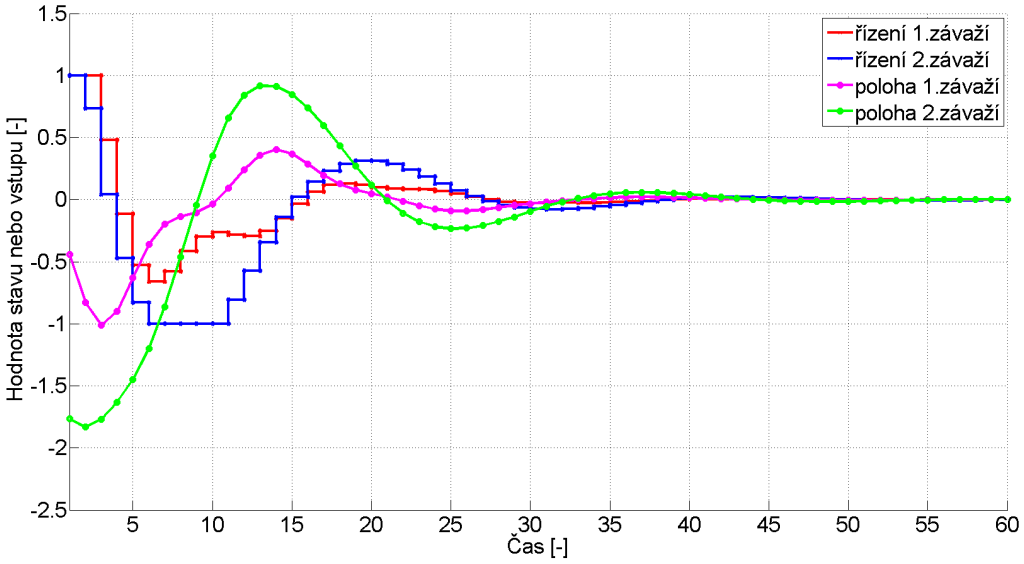
4.1 Demonstrace funkce

Nejprve je demonstrována samotná funkčnost algoritmu, a to na následujícím příkladu dynamického systému spojených závaží. Je nutné podotknout, že pro samotný průběh řízení není způsob výpočtu Newtonova kroku rozhodující, respektive všechny přístupy dávají stejné výsledky. Rozdíl je v rychlosti výpočtu Newtonova kroku.
Jedná se o model dvou závaží spojených pružinou, z nichž jedno je dále připojeno pružinou k nehybnému objektu, například zdi. Tento systém má čtyři stavy, rychlosti obou závaží a výchylky těchto závaží. Dále uvažujme, že můžeme přímo působit na obě závaží, systém má tedy dva vstupy. Parametry systému jsem volil tak, že je mírně tlumeným oscilátorem. Průběh výchylky v čase pro neřízený systém a nenulové počáteční podmínky je uveden v grafu 4.1.



Obrázek 4.1. Ukázka chování spojených závaží při nulových vstupech

Při řízení systému bude v tomto případě cílem co nejrychleji zastavit kmitání, tedy přivést stavy k nule. Dále stanovíme omezení na vstupy, jako maximum uvažujme 1 a jako minimum -1. Graf v závislosti na čase takto řízeného systému je na obrázku 4.2. Z tohoto grafu je vidět, že systém se dostane do ustáleného stavu poměrně rychle i přes to, že na vstupy jsou kladena omezení.



Obrázek 4.2. Ukázka chování spojených závaží při nulových vstupech

4.2 Zrychlení výpočtu

Hlavním cílem této práce je otestovat zrychlení výpočtu způsobené paralelizací Newtonova kroku. Smyslem paralelizace je rozdělit problém na několik menších, které mohou být vypočteny paralelně. S tím souvisí otázka jaký počet paralelních procesů je optimální. Tedy na kolik částí je nejvýhodnější problém rozdělit za předpokladu, že je k dispozici dostatek výpočetních jednotek.

4.2.1 Teoretická úvaha

Paralelizace, jak již bylo uvedeno, je důležitá pro výpočet Choleskyho faktorizace matice $\boldsymbol{G}$. Velikost této matice je závislá na délce predikčního horizontu N. Dalším důležitým parametrem určujícím složitost výpočtu je velikost vektoru $\boldsymbol{p}$. Jako velikost vektoru $\boldsymbol{p}$, označeno n_p, uvažujme počet stavových vektorů $\boldsymbol{x}$ v něm obsažených, nikoliv počet jeho prvků. Počet stavů, který je také důležitý v určení rozměrů uvažujme konstantní. Dále označme $n_b = n_p + 1$ počet bloků matice $\boldsymbol{G}$.
Se zvyšující se velikostí vektoru $\boldsymbol{p}$ je možné výpočet rozložit na více částí, ale na druhé straně roste náročnost výpočtu *pomocných* matic, a také dalších výpočtů s těmito maticemi.
Pro účely této úvahy bereme v potaz pouze nejsložitější výpočty, tedy takové které mají složitost řádově $O(n_x^3)$.
V případě, že vektor $\boldsymbol{p}$ je prázdný, tedy není použita paralelizace, jsou pro určení složitosti klíčové dva členy. Složitost sestavení matice $\boldsymbol{G}$ a výpočet Choleskyho faktorizace této matice. Matice $\boldsymbol{G}$ má následující předpis, $\boldsymbol{G} = \boldsymbol{\Gamma}\boldsymbol{\Psi}^{-1}\boldsymbol{\Gamma}^T + \widehat{\boldsymbol{A}}\widehat{\boldsymbol{Q}}^{-1}\widehat{\boldsymbol{A}}^T$. Druhý člen této rovnice je konstantní, lze ho tedy předvypočítat a pro naši úvahu nehraje roli. První člen má složitost přibližně $O(Nn_x^3)$, protože matice $\boldsymbol{\Gamma}$ je blokově diagonální matice s N bloky velikosti $n_x \times n_x$ a matice $\boldsymbol{\Psi}^{-1}$ je diagonální.
Choleskyho faktorizace matice $\boldsymbol{G}$ má složitost přibližně $O(\frac{1}{3}(3N-2)n_x^3)$, což plyne z tridiagonální struktury této matice.
Přistoupíme-li k paralelizaci, maticových operací bude více, avšak bude možné je paralelizovat.

Sestavení matice $\boldsymbol{G}$ je možno paralelizovat, a v takovém případě bude složitost přibližně $O(\frac{1}{n_b}(Nn_x^3))$. Choleskyho faktorizace této matice bude mít složitost $O(\frac{1}{3}\frac{(3N-2)n_x^3}{n_b})$. Další složitou operací je výpočet matice $\boldsymbol{\Phi}$, tedy výrazu $\boldsymbol{A}_p^T\boldsymbol{G}^{-1}\boldsymbol{A}_p$. Ten má složitost přibližně $O(n_p\ n_x^3)$, protože matice $\boldsymbol{A}_p$ obsahuje n_p matic $\boldsymbol{A}$, a protože výpočet je symetrický. Složitost výpočtu Choleskyho faktorizace matice $\boldsymbol{\Phi}$ se dá zaokrouhlit na $O(\frac{1}{3}(n_x^3))$, protože je možné jí paralelizovat. Předposledním výpočtem, braným v úvahu je člen $\boldsymbol{W}\boldsymbol{\Phi}^{-1}\boldsymbol{W}^T$, ten má složitost přibližně $O((3n_p-2)n_x^3)$, což plyne ze struktury matice $\boldsymbol{W}^T$. Posledním členem je Choleskyho faktorizace matice $\boldsymbol{M}$. Tato matice má tridiagonální strukturu, a proto je složitost této faktorizace přibližně $O(\frac{1}{3}(3n_p-2)n_x^3)$.

Sečtením těchto členů a s tím, že $n_b = n_p + 1$, získáváme výsledný vztah $\frac{(2N-\frac{2}{3})n_x^3}{n_b} + (5n_b - \frac{23}{3})n_x^3$. Derivací tohoto vztahu a položením této derivace nule dostaneme předpis pro teoreticky optimální počet paralelních bloků: $n_b = \sqrt{\frac{2}{5}N - \frac{2}{15}}$.

4.2.2 Numerický test

Na testovacím skriptu výpočtu Newtonova kroku jsem testoval dobu výpočtu tohoto kroku v závislosti na velikosti n_p.

Pro měření doby trvání výpočtu jsem použil funkce *tic* a *toc*. Problémem v případě MATLABu je fakt, že změřené doby trvání výpočtu se mohou lišit, protože přiřazení výpočetních prostředků funkci záleží na operačním systému. Proto jsem test desetkrát opakoval se stejnými vstupními daty, a následně jsem bral nejmenší dosažený čas, protože ten je nejblíže čisté době vykonávání funkce.

Pro zefektivnění maticových operací je použito *sparse* ukládání matic.

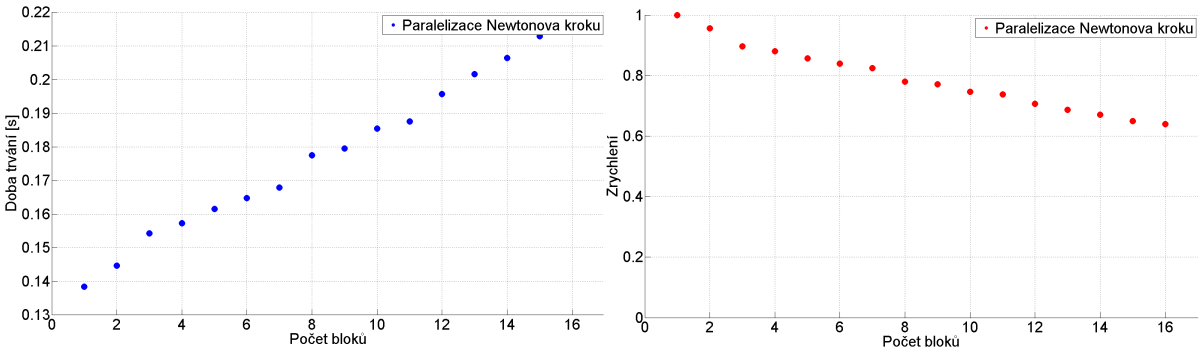
Pro paralelizaci jsem dále použil funkci *parpool*, pomocí které jsem vytvořil šestnáct procesů MATLABu, pro každý procesor jeden. Samotná paralelní část algoritmu, tedy výpočet Choleskyho faktorizace, byla vykonávána v $parfor$ smyčce.

Paralelizaci jsem testoval na systému s následujícími parametry.

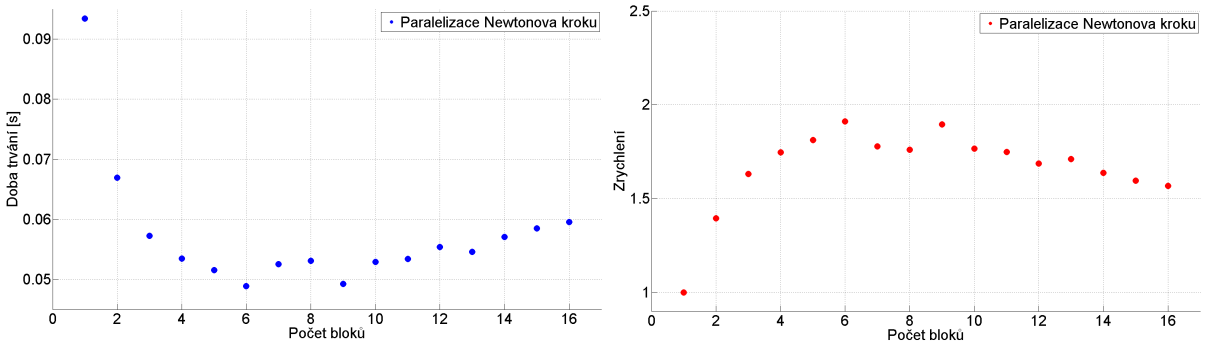
Procesor: Intel Xeon E5-2687W0; 3,10GHz; 16 cores, Operační paměť: 128GB RAM.

Software: MATLAB R2014b s Parallel Computing Toolboxem.

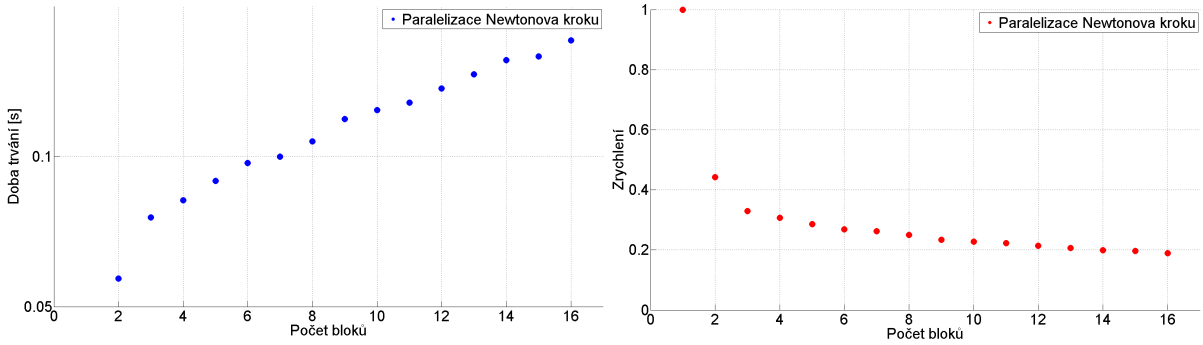
Jako testovací data jsem zvolil několik náhodně vygenerovaných systémů. Následně jsem na nich testoval tři skripty. První měřil celkovou dobu vykonávání Newtonova kroku v paralelní formulaci, druhý měřil jen čistou dobu vykonávání $parfor$ smyčky pro výpočet Choleskyho faktorizace matice $\boldsymbol{G}$. Třetí skript měřil dobu výpočtu pomocných matic a vektorů, tedy režii přidanou ve srovnání s neparalelizovanou formulací. Výsledky těchto skriptů jsem vykreslil do grafů, na kterých je znázorněna doba vykonávání příslušné funkce v závislosti na míře paralelizace, tedy na počtu bloků matice $\boldsymbol{G}$.



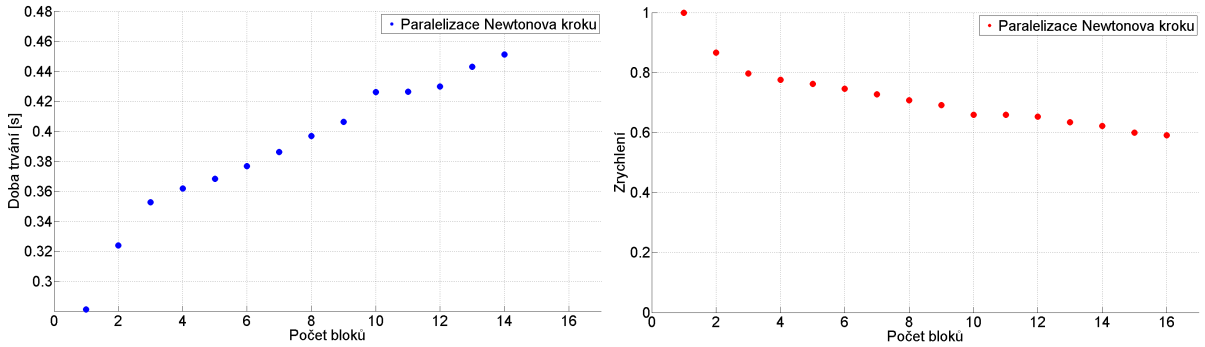
Obrázek 4.3. Graf doby trvání Newtonova kroku v závislosti na počtu bloků matice $\boldsymbol{G}$ pro $n_x = 50$ a $N = 80$



Obrázek 4.4. Graf doby trvání Choleskyho faktorizace v závislosti na počtu bloků matice $\boldsymbol{G}$ pro $n_x = 50$ a $N = 80$



Obrázek 4.5. Graf doby trvání výpočtu pomocných matic v závislosti na počtu bloků matice $\boldsymbol{G}$ pro $n_x = 50$ a $N = 80$

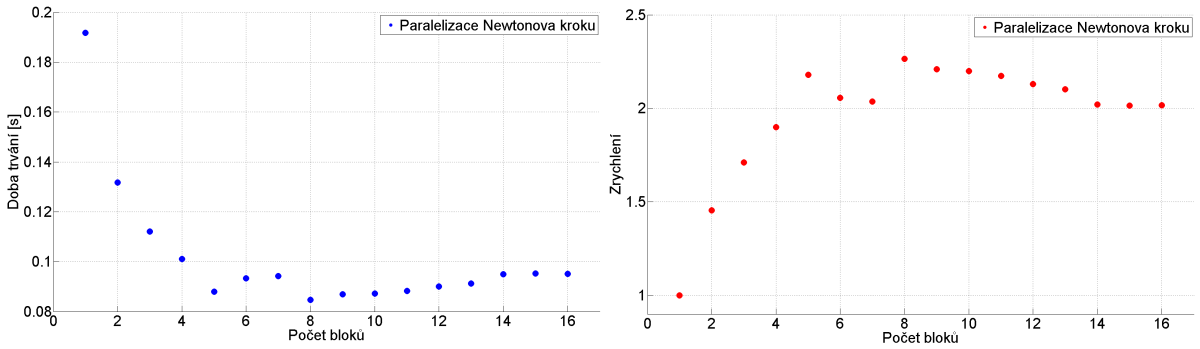


Obrázek 4.6. Graf doby trvání Newtonova kroku v závislosti na počtu bloků matice $\boldsymbol{G}$ pro $n_x = 75$ a $N = 80$

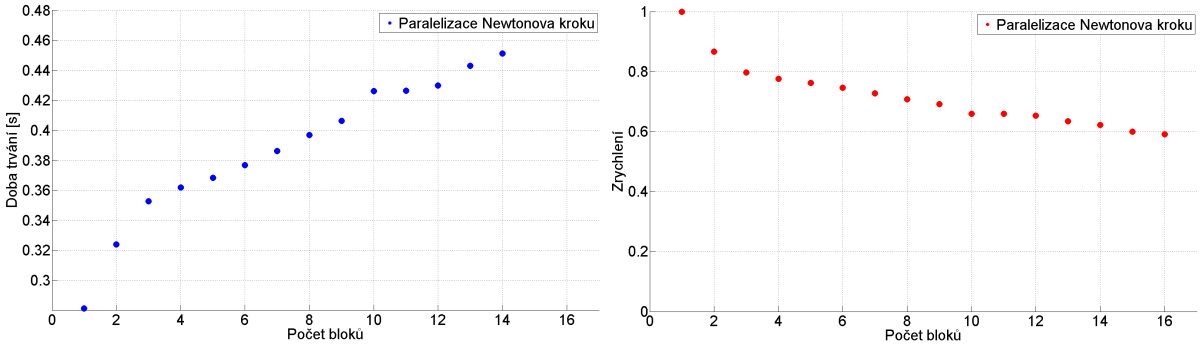
Z těchto grafů je vidět, že paralelizace představená v této práci a následně implementovaná v MATLABu výpočet nezrychlila, právě naopak. Výpočet Choleskyho faktorizace se zrychlil, ovšem režie nutná pro tuto paralelizaci byla příliš vysoká a se zvyšujícím se n_b rostla.

Pro zrychlení celého výpočtu by bylo pravděpodobně nutné zefektivnit maticové výpočty, což se v rámci této práce bohužel nepodařilo. Druhou možností by mohla být odlišná formulace paralelizace, obsahující méně maticových operací.

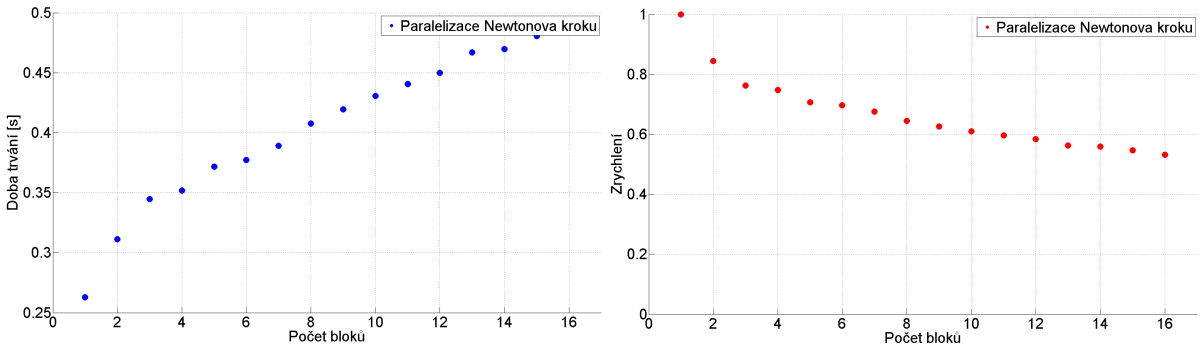
Závěrem praktického porovnání tedy je, že se v této práci nepodařilo prokázat, že paralelní varianta kombinované gradient/Newtonovy projekce v této práci uvedená je rychlejší než-li původní formulace.



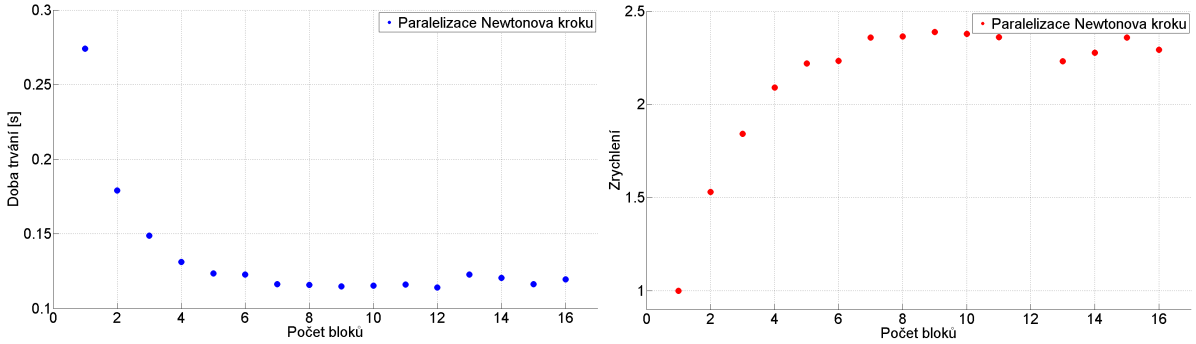
Obrázek 4.7. Graf doby trvání Choleskyho faktorizace v závislosti na počtu bloků matice $\boldsymbol{G}$ pro $n_x = 75$ a $N = 80$



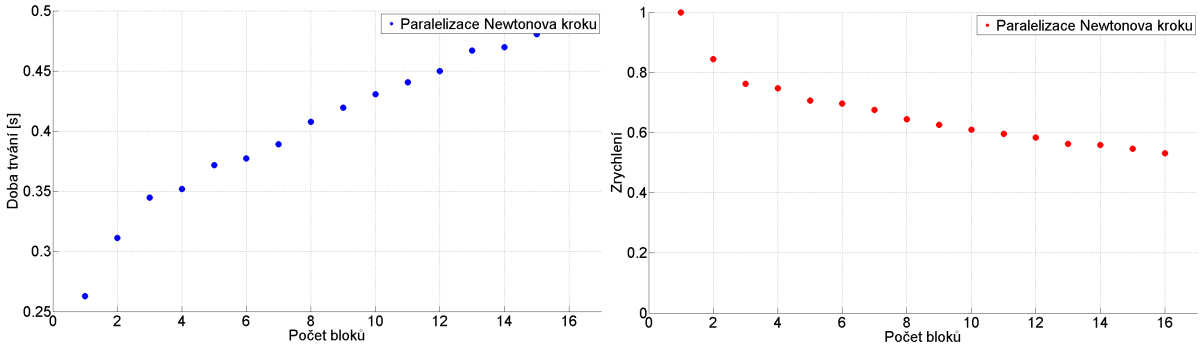
Obrázek 4.8. Graf doby trvání výpočtu pomocných matic v závislosti na počtu bloků matice $\boldsymbol{G}$ pro $n_x = 75$ a $N = 80$



Obrázek 4.9. Graf doby trvání Newtonova kroku v závislosti na počtu bloků matice $\boldsymbol{G}$ pro $n_x = 90$ a $N = 50$



Obrázek 4.10. Graf doby trvání Choleskyho faktorizace v závislosti na počtu bloků matice $\boldsymbol{G}$ pro $n_x = 90$ a $N = 50$



Obrázek 4.11. Graf doby trvání výpočtu pomocných matic v závislosti na počtu bloků matice $\boldsymbol{G}$ pro $n_x = 90$ a $N = 50$

Kapitola 5
Závěr

V úvodu této práce (1) byl stručně představen koncept MPC a několik možných formulací MPC úlohy. V kapitole 2 bylo popsáno několik metod pro řešení úlohy kvadratického programování s omezeními i bez omezení a některé z nich byly pro ilustraci porovnány na modelovém příkladu. V kapitole 3 bylo uvedeno několik možných způsobů výpočtu Newtonova kroku pro metodu kombinované gradient/Newtonovy projekce a byla porovnána struktura řešeného problému v jednotlivých formulacích. V předposlední kapitole (4) byla na příkladu spojených závaží znázorněna funkčnost implementovaného regulátoru založeného na metodě kombinované gradient/Newtonovy projekce, a poté zde byly uvedeny výsledky numerických testů paralelizace Newtonova kroku.

Implementovat prediktivní regulátor i paralelní prediktivní regulátor se podařilo, a srovnání s již existujícími výpočetními programy (qpOASES, [16]) ukázaly, že regulace pomocí těchto regulátorů dává správné výsledky. Při porovnání efektivity paralelizace se však ukázalo, že paralelizace uvedená v této práci a implementovaná v prostředí MATLAB není efektivnější než neparalelizovaná formulace. Paralelizace zefektivněla výpočet Choleskyho faktorizace, což byl předpoklad, nicméně výpočty přidané v důsledku reformulace problému byly natolik náročné, že celková doba vykonávání Newtonova kroku byla pro paralelizovanou formulaci výrazně delší než pro neparalelizovanou. Problémem byla řada maticových operací, které se v důsledku reformulace objevily a jejichž výpočetní složitost byla výrazně větší, než bylo očekáváno.

V budoucnu by bylo vhodné implementovat algoritmus efektivnějším způsobem, nebo dokonce v jiném programu než MATLAB. Maticové funkce by bylo vhodné programovat například přímo v programovacím jazyce C.

Závěrem této práce je tedy konstatování, že cíle této práce byly splněny, avšak nepodařilo se prokázat, že by zde představená paralelizace znamenala zrychlení výpočtu Newtonova kroku.

Literatura

[1] Ruchika, N. Raghu. Model Predictive Control: History and Development. *IJETT*, 4, 2013.

[2] Bembook. Klouzavý horizont. *http : //www.bembook.ibpsa.us/index.php?title = Model_Predictive_Control*. Staženo 11.11.2014.

[3] D. Chmielewski, V. Manousiouthakis. On constrained infinite-time linear quadratic optimal control. *Decision and Control, Proceedings of the 35th IEEE Conference on*, 1996.

[4] O. Mikulas. Quadratic Programming Algorithms for Fast Model-Based Predictive Control - Bachelor thesis, 2013.

[5] Z. Dostál. *Optimal Quadratic Programming Algorithms, with Applications to Variational Inequalities,1st Edition*. Springer, 2008.

[6] J.R. Shewchuk. An Introduction to the Conjugate Gradient Method Without the Agonizing Pain. *School of Computer Science Carnegie Mellon University*, 1994.

[7] V. Pan. Complexity of Computations with Matrices and Polynomials. *Society for Industrial and Applied Mathematics*, 1992.

[8] J. Nocedal, S.J. Wright. *Numerical Optimization*. Springer Science and Business Media, 1999.

[9] G. Hammerlin a K.H. Hoffmann. *Numerical Mathematics*. Springer Science and Business Media, 1991.

[10] P. Otta. Rychlé numerické metody pro prediktivní řízení - bakalářská práce, 2011.

[11] Y. Wang a S. Boyd. Fast Model Predictive Control Using Online Optimization. *IEEE*, 2009.

[12] S. Richter, N.C. Jones a M. Morari. Real-Time Input Constrained MPC Using Fast Gradient Methods. *Switzerland, Department of Electrical Engineering, Swiss Federal Institute of Technology Zurich*, 2010.

[13] S. Boyd, L. Vanderberghe. *Convex Optimization*. Cambridge University Press, 2004.

[14] P. Otta, O. Šantin, V. Havlena. Tailored QP Algorithm for Predictive Control with Dynamics Penalty. *Control and Automation (MED), 2014 22nd Mediterranean Conference of*, 2014.

[15] I. Nielsen a D. Axehill. An $O(\log n)$ Parallel Algorithm for Newton Step Computation in Model Predictive Control. *IFAC*, 2014.

[16] H.J. Ferreau, C. Kirches, A. Potschka, H.G. Bock a M. Diehl. qpOASES: A parametric active-set algorithm for quadratic programming. *Mathematical Programming Computation*, 6(4):327–363, 2014.

Příloha A
Obsah CD

Disk přiložený k této práci obsahuje několik souborových adresářů.

- MATLAB - Obsahuje zdrojové kódy regulátoru a dalších skriptů pro MATLAB.
- Tex - Obsahuje zdrojové kódy textu bakalářské práce v jazyce TEX, využívající makra balíčku *Pmac*, a dále obsahuje obrázky použité v bakalářské práci.
- BakalářskáPráce - Obsahuje text bakalářské práce ve formátu pdf