FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

UNIVERZITY KOMENSKÉHO

v Bratislave

Ekonomická a finančná matematika

DIPLOMOVÁ PRÁCA

2003

Ľubomír Koršňák

FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY
UNIVERZITY KOMENSKÉHO
v Bratislave

Ekonomická a finančná matematika

DIPLOMOVÁ PRÁCA

Metódy vnútorného bodu v lineárnych úlohách optimalizácie dlhopisového portfólia

Autor: Ľubomír Koršňák
Vedúci diplomovej práce: Doc. RNDr. Margaréta Halická, CSc.

Bratislava 2003

Čestne vyhlasujem, že som túto diplomovú prácu vypracoval samostatne, len s použitím uvedenej literatúry.

....

V Bratislave 4.apríla 2003 Ľubomír Koršňák

Poďakovanie

Ďakujem aj touto cestou Doc.RNDr.Margaréte Halickej, CSc. za sprístupnenie literatúry a v neposlednom rade za cenné rady a konzultácie pri písaní diplomovej práce.

Obsah

Úvod 1

1. Úloha lineárneho programovania 2

1.1 Formulácia úlohy lineárneho programovania 2

1.2 Teória duality 4

2. Homogénna samoduálna metóda 7

2.1 Homogénna samoduálna úloha 8

2.2 Princíp algoritmu na riešenie homogénnej samoduálnej úlohy 9

2.3 Prediktor-korektor algoritmus 11

2.4 Transformácia štandartnej úlohy na homogénnu samoduálnu úlohu 15

2.5 Vzťah optimálneho riešenia homogénnej samoduálnej úlohy a optimálneho riešenia štandartnej úlohy 16

2.6 Redukovaný KKT systém pre štandartnú úlohu 18

3. Lineárna optimalizácia vo financiách 22

3.1 Deterministický model s minimalizáciou nákladov 23

3.2 Deterministický model s maximalizáciou výnosu na konci 27

3.3 Stochastický prípad MAX modelu 28

3.4 Generovanie scenárov 29

4. Optimalizácia dlhopisového portfólia s využitím homogénnej samoduálnej metódy 31

4.1 S-MAX model a úloha lineárneho programovania 31

4.2 MPS formát 34

4.3 Výsledky experimentov 37

Záver 43

Literatúra 44

Úvod

Lineárne programovanie má rozsiahle praktické použitie nielen v oblasti vedy a techniky, ale aj v procese ekonomického a finančného rozhodovania. Medzi takéto procesy finančného rozhodovania patrí aj problém optimalizácie dlhopisového portfólia. Teória lineárneho programovania poskytuje viacero metód a algoritmov pre hľadanie optimálneho riešenia. Medzi najstaršie, ale zároveň aj najpoužívanejšie patrí simplexová metóda (Dantig, 1947). Nevýhodou simplexovej metódy je, že teoreticky má exponenciálnu zložitosť, a teda nie je vhodná pre optimalizáciu úloh veľkých rozmerov. Na druhej strane, aby bol ekonomický model použiteľný v praxi, nesmie byť miera abstrakcie od reality príliš veľká. To však vedie k formulácií modelov relatívne veľkých rozmerov.
Alternatívou k simplexovej metóde sú metódy súhrne označované ako metódy vnútorného bodu. Tieto metódy na rozdiel od simplexovej metódy vykazujú polynomiálnu zložitosť. Ďalšou zaujímavou vlastnosťou týchto metód je, že v prípade existencie alternatívnych optimálnych riešení poskytujú iné optimálne riešenie ako simplexová metóda.
Cieľom tejto práce bolo nájsť na internete softvér používajúci niektorú z metód vnútorného bodu, naštudovať a popísať túto metódu a sfunkčniť softvér. Pomocou neho potom riešiť problém optimalizácie dlhopisového portfólia a porovnať získané výsledky s výsledkami získanými použitím simplexovej metódy. Tento cieľ sa podarilo naplniť. Na internete sme našli softvér využívajúci homogénnu samoduálnu metódu a ten sme aplikovali na lineárny model optimalizácie dlhopisového portfólia. Úlohy sme generovali v rozmeroch do 24002 ohraničení a 21009 premenných.
Práca je členená do štyroch kapitol. V prvej kapitole kapitole sú zhrnuté základné pojmy lineárneho programovania a teórie duality. Druhá kapitola vychádza z knihy Roberta J. Vanderbeia[2] a je venovaná popisu homogénnej samoduálnej metódy. V tretej kapitole sme popísali problém optimalizácie dlhopisového portfólia a naformulovali modely popisujúce tento model. Cenným zdrojom informácií pre túto kapitolu bol článok S.Nielsena[1]. V poslednej štvrtej kapitole sú uvedené výsledky experimentov.

1. Úloha lineárneho programovania

1.1 Formulácia úlohy lineárneho programovania

Jedným z najdôležitejších problémov v ekonómii je hľadanie optimálneho riešenia. Riešenie takýchto optimalizačných problémov často vedie k úlohám lineárneho programovania. Uveďme teda, čo to úloha lineárneho programovania vlastne je. Lineárne programovanie sa zaoberá špeciálnymi úlohami na viazaný extrém funkcií viac premenných. Výnimočnosť týchto úloh spočíva v tom, že ide o extrémy lineárnych funkcií viazaných podmienkami v tvare lineárnych rovníc a nerovníc.

Uveďme niektoré základné typy úloh lineárneho programovania. Nech matica $A = (a_{ij}) \in R^{m \times n}$ a vektory $b = (b_i) \in R^m$, $c = (c_j) \in R^n$. V tejto práci budeme vektory chápať ako stĺpce. Optimalizačnú úlohu maximalizovať lineárnu funkciu v tvare:

$$\sum_{j=1}^{n} c_j x_j,$$

ktorú nazývame účelová funkcia, na množine riešení lineárnych nerovníc:

$$\begin{aligned} \sum_{j=1}^{n} a_{ij} x_j \leqslant b_i \qquad & i = 1, ..., m \\ x_j \geqslant 0 \qquad & j = 1, ..., n \end{aligned}$$

budeme nazývať úlohou lineárneho programovania v štandartnom tvare (Vanderbei[2]) resp. stručne štandartnou úlohou. Takáto úloha sa často označuje aj ako úloha lineárneho programovania v tvare nerovností. Skrátene budeme túto úlohu zapisovať v maticovom tvare:

$$\max \{ \ c^T x \mid \quad Ax \leqslant b \ ; \quad x \geqslant 0 \ \}$$

Vo všeobecnosti nemusíme riešiť maximalizačný problém, ale rovnako aj minimalizačný. Keďže pre ľubovoľnú množinu $M \in R^n$ a ľubovoľnú funkciu $f : M \to R$ platí

$$\min_{x \in M} f(x) = -\max_{x \in M}(-f(x))$$

môžeme ľubovoľnú minimalizačnú úlohu previesť na maximalizačnú.

Niekedy požadujeme od všetkých ohraničení, aby sa nadobúdali ako rovnosti ,t.j. máme úlohu maximalizovať

$$\sum_{j=1}^{n} c_j x_j,$$

pri ohraničeniach

$$\begin{aligned} \sum_{j=1}^{n} a_{ij} x_j = b_i & \qquad i = 1, ..., m \\ x_j \geqslant 0 & \qquad j = 1, ..., n. \end{aligned}$$

Potom takúto úlohu lineárneho programovania nazývame úlohou lineárneho programovania v rovnicovom tvare. Každú úlohu v rovnicovom tvare vieme previesť na úlohu v tvare nerovností a to nahradením každej rovnosti $\sum_{j=1}^{n} a_{ij} x_j = b_i$ dvojicou nerovností $\sum_{j=1}^{n} a_{ij} x_j \leqslant b_i$ a $-\sum_{j=1}^{n} a_{ij} x_j \leqslant b_i$. Naopak každú úlohu v tvare nerovností vieme previesť na rovnicový tvar zavedením nezáporných doplnkových premenných $z_i \geqslant 0$ a nahradením nerovností rovnosťami v tvare $\sum_{j=1}^{n} a_{ij} x_j + z_i = b_i$

Nakoniec uvedieme najvšeobecnejšiu formuláciu úlohy lineárneho programovania. Nech $n_1 < n, \ m_1 < m$, potom úlohu maximalizovať

$$\sum_{j=1}^{n} c_j x_j,$$

pri ohraničeniach:

$$\begin{aligned} \sum_{j=1}^{n} a_{ij} x_j \leqslant b_i & \qquad i = 1, ..., m_1 \\ \sum_{j=1}^{n} a_{ij} x_j = b_i & \qquad i = m_1 + 1, ..., m \\ x_j \geqslant 0 & \qquad j = 1, ..., n_1 \end{aligned}$$

nazývame úlohou lineárneho programovania v zmiešanom tvare. Takúto úlohu vieme však previesť do rovnicového alebo nerovnicového tvaru zavedením doplnkových premenných resp. nahradením každej rovnosti dvojicou nerovností. V týchto úlohách ale požadujeme nezápornosť všetkých premenných, čo dosiahneme rozložením premenných

$x_j,\ j = n_1 + 1, ..., n$ na dve nové nezáporne premenné $x_j^+ \geqslant 0$ a $x_j^- \geqslant 0$, pričom budeme požadovať aby platilo $x_j = x_j^+ - x_j^-$. Na základe vyššie uvedeného sa budeme ďalej v tejto práci venovať len úlohám v štandartnom tvare.

Pre úlohu v štandartnom tvare teraz zadefinujeme niekoľko základných pojmov:
Množinu $M = \{\ x \in R^n \mid \ \ Ax \leqslant b\ ,\ x \geqslant 0\ \}$ nazveme množinou prípustných riešení, jej prvky prípustnými riešeniam. Prípustné riešenie $x^* \in M$, pre ktoré platí $c^T x^* \geqslant c^T x$, pre všetky $x \in M$ budeme nazývať optimálnym riešením. Niekedy sa môže stať, že množina prípustných riešení M je prázdna, t.j. neexistuje žiadne prípustné riešenie, potom takúto úlohu budeme nazývať neprípustnou. Ďalšia zaujímavá možnosť, ktorá môže nastať je, že síce existuje prípustné riešenie, ale optimálna hodnota účelovej funkcie je nekonečno. Takúto úlohu potom nazveme neohraničenou.

1.2 Teória duality

Jedným z dôležitých výsledkov teórie lineárneho programovania je poznatok, že s každou maximalizačnou úlohou je spojená minimalizačná úloha, ktorá je danou maximalizačnou úlohou jednoznačne určená. Táto minimalizačná úloha sa nazýva duálnou úlohou k maximalizačnej úlohe, ktorú budeme nazývať primárnou úlohou.
Ukážme si ako takáto dvojica úloh vyzerá. Uvažujme primárnu úlohu v tvare

$$(P) \qquad \max\ \{\ c^T x \mid \ \ Ax \leqslant b,\ x \geqslant 0\ \}.$$

Potom tejto úlohe zodpovedajúca duálna úloha má tvar:

$$(D) \qquad \min\ \{\ b^T y \mid \ \ A^T y \geqslant c,\ y \geqslant 0\ \}.$$

Poznamenajme, že prívlastky primárna a duálna sú podmienené tým, z ktorej úlohy sme vyšli. Ak by sme vyšli z úlohy (D), potom k nej duálna úloha je úloha (P). Preto sa často hovorí o dvojici vzájomne duálnych úloh. V tejto práci však budeme pod primárnou úlohou vždy rozumieť maximalizačnú úlohu a teda k nej prislúchajúcu duálnu úlohu zasa minimalizačnú úlohu.

Pre dvojicu vzájomne duálnych úloh platia mnohé užitočné tvrdenia, ktoré napomáhajú pri hľadaní optimálneho riešenia úloh lineárneho programovania.

Veta 1.1. (Slabá veta o dualite) [Vanderbei[2], str.58]
Pre každé prípustné riešenie $\overline{x}$ primárnej úlohy a každé prípustné riešenie $\overline{y}$ duálnej úlohy platí nerovnosť $c^T\overline{x} \leqslant b^T\overline{y}$.

Dôkaz.
Vychádzajme z hodnoty účelovej funkcie primárnej úlohy $c^T\overline{x}$. Keďže $\overline{y}$ je prípustné riešenie duálnej úlohy musí spĺňať nerovnosť $c^T \leqslant \overline{y}^T A$. Teda:

$$c^T\overline{x} \leqslant \overline{y}^T A\overline{x}$$

Nakoniec keďže $\overline{x}$ je prípustné riešenie primárnej úlohy platí $A\overline{x} \leqslant b$ a teda:

$$c^T\overline{x} \leqslant \overline{y}^T b = b^T\overline{y}. \qquad \square$$

Aj napriek názvu slabá veta o dualite, má táto veta veľký význam. Umožňuje totiž získať postačujúce podmienky optimálnosti prípustných riešení. Stačí si všimnúť, že podľa nerovnosti $c^T\overline{x} \leqslant b^T\overline{y}$ hodnota účelovej funkcie minimalizačnej úlohy v ktoromkoľvek prípustnom riešení je horným odhadom hodnôt účelovej funkcie príslušnej maximalizačnej úlohy na množine všetkých prípustných riešení a naopak hodnota účelovej funkcie maximalizačnej úlohy v ktoromkoľvek prípustnom riešení je dolným odhadom hodnoty účelovej funkcie minimalizačnej úlohy na množine všetkých jej prípustných riešení.
Rozdiel medzi hodnotou účelových funkcií primárnej a duálnej úlohy $b^Ty - c^Tx$ nazývame duálnou medzerou. Sledovanie veľkosti duálnej medzery poskytuje vhodný nástroj na overenie optimálnosti riešenia. Ukážeme totiž, že ak x^* je prípustné riešenie primárnej úlohy a y^* je prípustné riešenie duálnej úlohy také, že platí rovnosť $b^Ty^* = c^Tx^*$, tak potom x^* je optimálnym riešením primárnej úlohy a y^* je optimálnym riešením duálnej úlohy. Nech x je ľubovoľné prípustné riešenie primárnej úlohy. Potom zo slabej vety o dualite máme: $c^Tx \leqslant b^Ty^* = c^Tx^*$ a keďže x^* je prípustné riešenie, vidíme že musí byť optimálne. Analogicky sa ukáže, že y^* je optimálne pre duálnu úlohu.

Veta 1.2. (Silná veta o dualite) [Vanderbei[2], str.60]
Ak má jedna z dvojice vzájomne duálnych úloh optimálne riešenie, tak majú optimálne riešenie obe úlohy. Navyše ak x^* je optimálne riešenie primárnej úlohy a y^* je optimálne riešenie duálnej úlohy platí rovnosť $c^T x^* = b^T y^*$.

Zo silnej vety o dualite dostávame, že nulová duálna medzera je nielen postačujúcou ale aj nutnou podmienkou optimality.
Pozrime sa teraz na možnosti, ktoré môžu nastať pre primárno-duálnu dvojicu riešení úlohy lineárneho programovania. Silná veta o dualite hovorí, že ak primárna úloha má optimálne riešenie, tak optimálne riešenie má aj duálna úloha a naopak. Čo sa však stane, ak primárna úloha nemá optimálne riešenie, napríklad je neohraničená. Neohraničenosť primárnej úlohy spolu so silnou vetou o dualite nám nevyhnutne dáva neprípustnosť duálnej úlohy, pričom veľkosť duálnej medzery je nulová a rovnosť sa nadobúda v ∞. Obdobne ak je duálna úloha neohraničená, potom primárna úloha je neprípustná, duálna medzera je nulová a rovnosť sa nadobúda v $-\infty$. Posledným prípadom, ktorý môže nastať je, že primárna aj duálna úloha sú neprípustné. V takomto prípade máme nenulovú duálnu medzeru.
Niekedy je potrebné nájsť optimálne riešenie duálnej úlohy, pričom optimálne riešenie primárnej úlohy je známe. Nasledujúca veta pomôže takýto problém vyriešiť.

Veta 1.3. (Veta o komplementarite) [Vanderbei[2], str.67]
Predpokladajme, že $x = (x_1, ..., x_n)^T$ je prípustné riešenie primárnej úlohy (P) a $y = (y_1, ..., y_m)^T$ je prípustné riešenie duálnej úlohy (D). Nech $w = (w_1, ..., w_m)^T$ reprezentuje prislúchajúcu primárnu doplnkovú premennú a $z = (z_1, ..., z_n)^T$ prislúchajúcu duálnu doplnkovú premennú. Potom x je optimálne pre primárnu úlohu (P) a y je optimálne pre duálnu úlohu (D) vtedy a len vtedy keď

$$\begin{aligned} x_i z_i &= 0, \qquad i = 1, ..., n, \\ w_j y_j &= 0, \qquad j = 1, ..., m. \end{aligned} \tag{1.1}$$

Rovnosti (1.1) sa často nazývajú aj podmienkami komplemantarity a sú ekvivalentné s nulovou veľkosťou duálnej medzery. Ak (x^*, w^*) je optimálne riešenie primárnej úlohy v rovnicovom tvare a (y^*, z^*) je optimálne riešenie duálnej úlohy v rovnicovom tvare, potom veta o komplementarite hovorí, že pre každé $i = 1, ..., n$ je buď $x_j^* = 0$ alebo $z_j^* = 0$ alebo oba a pre každé $j = 1, ..., m$ je buď $y_j^* = 0$ alebo $w_j^* = 0$ alebo oba. Nasledujúca veta hovorí o tom, že existuje taká dvojica optimálnych riešení primárnej a duálnej úlohy v rovnicovom tvare, pre ktoré prípad "oba" nenastáva, t.j. pre ktoré je práve jedna premenná z každej dvojice (x_i^*, z_i^*) nulová a taktiež nulová je práve jedna premenná z každej dvojice (y^*, w^*). V takomto prípade nazývame optimálne riešenie ostro komplementárnym a často sa táto ostrosť komplementarity vyjadruje tvrdením, že $x^* + z^* > 0$ a $y^* + w^* > 0$.

Veta 1.4. (Veta o ostrej komplmentarite) [Vanderbei[2], str.169]
Ak má úloha lineárneho programovania optimálne riešenie tak, potom existuje také optimálne riešenie (x^*, w^*) a optimálne riešenie duálnej úlohy (y^*, z^*), že platí: $x^* + z^* > 0$ a $y^* + w^* > 0$.

2. Homogénna samoduálna metóda

V tejto kapitole ukážeme ako možno riešiť úlohu lineárneho programovania. Najčastejšie sa úlohy lineárneho programovania riešia simplexovou metódou. Tá však nie je polynomiálna, a preto nemusí byť vhodná pre úlohy veľkých rozmerov.

Iný možný prístup predstavujú metódy vnútorného bodu, ktorých algoritmy vykazujú polynomiálnu zložitosť. Metódy vnútorného bodu zahŕňajú širokú škálu rôznych algoritmov a metód. Jednou z nich je tzv. homogénna samoduálna metóda. Jej princíp spočíva v tom, že sa štandartná úloha pretransformuje na tzv. homogénnu samoduálnu úlohu (viď odsek 2.4) a úloha sa vyrieši prostredníctvom riešenia homogénnej samoduálnej metódy. Hoci homogénna samoduálna úloha neobsahuje vnútorné body, je ju možné efektívne riešiť postupmi, ktoré boli vyvinuté pre úlohy s vnútornými bodmi. Výhodou je, že nepotrebujeme vnútorný bod na naštartovanie algoritmu, a že analýza konvergencie a polynomiálnej zložitosti je relatívne jednoduchá. V nasledujúcich častiach predstavíme základné myšlienky tejto metódy. Budeme sa pritom pridržiavať výkladu z knihy Roberta J. Vanderbeia[2].

Predtým ako sa začneme venovať samotnej metóde, upozorníme ešte na neštandartné označenie používané v práci pre diagonálnu maticu, ktorej diagonálne prvky pochádzajú z prislúchajúceho vektora. Napríklad:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \qquad => \qquad X = \begin{bmatrix} x_1 & & & \\ & x_2 & & \\ & & \ddots & \\ & & & x_n \end{bmatrix}$$

2.1 Homogénna samoduálna úloha

Uvažujme úlohu lineárneho programovania v štandartnom tvare:

$$(P) \quad \max \{ \ c^T x \mid \quad Ax \leqslant b \ ; \quad x \geqslant 0 \ \}$$
$$(D) \quad \min \{ \ b^T y \mid \quad A^T y \geqslant c \ ; \quad y \geqslant 0 \ \},$$

kde $A \in R^{m \times n}$; $c, x \in R^n$; $b, y \in R^m$. Takúto úlohu budeme nazývať samoduálnou, ak $m = n$, $b = -c$ a matica A je *kososymetrická* t.j.: $A = -A^T$. Dôvodom pre názov samoduálna je, že primárna aj duálna úloha sú totožné. Úloha lineárneho programovania s nulovými pravými stranami sa nazýva homogénnou. Teda samoduálnu úlohu, v ktorej $b = c = 0$ nazývame homogénnou samoduálnou úlohou. Poznamenajme ešte, že v takomto prípade je aj účelová funkcia nulová.

Po zavedení doplnkovej premennej z na odstránenie nerovností v ohraničeniach môžeme homogénnu samoduálnu úlohu zapísať v tvare:

$$(PD) \quad \max \{ \ 0 \mid \quad Ax + z = 0 \ ; \quad x, z \geqslant 0 \ \}$$

Pripomeňme, že primárna a duálna úloha sú zhodné.

Veta 2.1. [Vanderbei[2], str.351]

Pre homogénnu samoduálnu úlohu platí:

(a) Existuje prípustné riešenie a každé prípustné riešenie je optimálne.

(b) Množina prípustných riešení má prázdne vnútro. To značí, ak (x, z) je prípustné, potom $z^T x = 0$.

Dôkaz.

(a) Triviálne riešenie $(x, z) = 0$ je prípustné riešenie a keďže účelová funkcia je konštantná, každé prípustné riešenie je optimálne.

(b) Najskôr si ukážeme, že pre ľubovoľnú kososymetrickú maticu A a ľubovoľný vektor x platí: $x^T Ax = 0$.
Ak matica A je kososymetrická platí $A = -A^T$. Prenásobme túto rovnosť x^T resp. x zľava aj sprava a upravme:

$$x^T Ax = -x^T A^T x = -(x^T Ax)^T = -x^T Ax.$$

A teda naozaj musí platiť, že $x^T Ax = 0$.
Teraz už môžeme pristúpiť k samotnému dôkazu tvrdenia (b). Nech (x, z) je prípustné riešenie úlohy (PD), t.j. $Ax + z = 0,\ x, z \geqslant 0$. Ak prenásobíme ohraničenia $Ax + z = 0$ zľava x^T dostávame $x^T Ax + x^T z = 0$. Využijúc vyššie uvedenú vlastnosť kososymetrickej matice máme $z^T x = 0$ a teda nemôže byť $x > 0$ a súčasne aj $z > 0$. Z toho máme, že úloha (PD) nemá vnútorný bod. □

2.2 Princíp algoritmu na riešenie homogénnej samoduálnej úlohy

V časti 2.1 sme uviedli ako vyzerá homogénna samoduálna úloha. V nasledujúcich častiach naznačíme algoritmus pre nájdenie optimálneho riešenia (PD) úlohy.
Aby riešenie (x^*, z^*) bolo optimálne musí byť samozrejme prípustné a podľa vety o komplementarite (Veta 1.3) musí spĺňať aj podmienky komplementarity. Teda musí preň platiť:

$$Ax^* + z^* = 0$$

$$x^{*T} z^* = 0.$$

Princíp algoritmu spočíva v tom, že začneme s ľubovoľnou kladnou hodnotou premenných (x, z), dajme tomu $x = z = e$, ktorú budeme v nasledujúcich iteráciach postupne upravovať, a to tak, aby sme postupne znižovali neprípustnosť a nekomplementaritu. Podstatné je určiť správny smer a veľkosť kroku, ktorým budeme postupovať v jednotlivých iteráciach.

Vo všeobecnosti štartovací bod (x, z) nie je ani prípustný a nespĺňa ani podmienky komplementarity. Preto si zadefinujeme premenné merajúce veľkosť neprípustnosti a nekomplementarity:

$$\rho(x, z) := Ax + z$$

$$\mu(x, z) := \frac{1}{n} x^T z,$$

kde ρ vyjadruje neprípustnosť riešenia (x, z) a μ meria stupeň nekomplementarity medzi x a z. Pokiaľ x a z budú zrejmé z kontextu budeme jednoducho písať ρ pre $\rho(x, z)$ a μ pre $\mu(x, z)$.

Je zrejmé, že pre dosiahnutie optimálneho riešenia je nutné každou iteráciou približovať hodnoty ρ a μ k nule. Teda smer kroku $(\Delta x, \Delta z)$ zvolíme tak, aby sme redukovali neprípustnosť a nekomplementaritu nového bodu $(x + \Delta x, z + \Delta z)$ pri danom redukčnom parametri $0 \leqslant \delta \leqslant 1$. Takto dostávame nelineárny systém rovníc, ktorého riešením získame smer, v ktorom sa zníži neprípustnosť a nekomplementarita $(x + \Delta x, z + \Delta z)$ na δ násobok oproti (x, z):

$$A(x + \Delta x) + (z + \Delta z) = \delta(Ax + z)$$

$$(X + \Delta X)(Z + \Delta Z)e = \delta\mu(x, z)e$$

Zanedbaním nelinearity v Δ-premenných (vyskytujúcej sa v 2.rovnici) dostávame nasledovný systém lineárnych rovníc pre určenie smeru $(\Delta x, \Delta z)$:

$$A\Delta x + \Delta z = -(1 - \delta)\rho(x, z) \tag{2.1}$$

$$Z\Delta x + X\Delta z = \delta\mu(x, z)e - XZe \tag{2.2}$$

Tento systém rovníc budeme nazývať podľa Vanderbeia[2] Karush-Kuhn-Tucker systém (KKT systém). Po nájdení smeru, ktorým máme postupovať, určíme dĺžku kroku $\theta > 0$, pričom nesmieme porušiť nezápornosť premenných. Takýmto spôsobom sa dostaneme do nového bodu $(\widetilde{x}, \widetilde{z})$:

$$\widetilde{x} = x + \theta \Delta x$$

$$\widetilde{z} = z + \theta \Delta z,$$

pričom nové hodnoty vektorov merajúcich neprípustnosť a nekomplementaritu v novom bode $(\widetilde{x}, \widetilde{z})$ sú:

$$\widetilde{\rho} = \rho(\widetilde{x}, \widetilde{z})$$

$$\widetilde{\mu} = \mu(\widetilde{x}, \widetilde{z}).$$

Na záver ešte uvedieme niektoré dôležité vlastnosti kroku, ktoré sú zhrnuté v nasledujúcej vete.

Veta 2.2. [Vanderbei[2], str.352]
Platia nasledujúce vzťahy:
1. $\Delta z^T \Delta x = 0$.
2. $\widetilde{\rho} = (1 - \theta + \theta\delta)\rho$.
3. $\widetilde{\mu} = (1 - \theta + \theta\delta)\mu$.
4. $\widetilde{X}\widetilde{Z}e - \widetilde{\mu}e = (1 - \theta)(XZe - \mu e) + \theta^2 \Delta X \Delta Z e$.

Vzťah 1. hovorí, že smery Δx a Δz sú ortogonálne. Platnosť vzťahov 2. a 3. zaručuje, že v každej iterácií algoritmu klesá neprípustnosť ρ a nekomplementarita μ riešenia.

2.3 Prediktor-korektor algoritmus

Na určenie smeru a dĺžky kroku použijeme prediktor-korektor algoritmus. Myšlienka algoritmu spočíva v snahe udržať komponenty XZe blízko seba. Uvedieme dve metódy, ktoré sa budú líšiť vo výbere dĺžky kroku.

Metóda s krátkym krokom:

Pre $0 \leqslant \beta \leqslant 1$ zadefinujme množinu:

$$\mathcal{N}(\beta) = \{(x,z) > 0 : \|XZe - \mu(x,z)e\| \leqslant \beta\mu(x,z)\}$$

Pri prediktor-korektor algoritme však budeme narábať len z hodnotami parametra $\beta = 1/2$ a $\beta = 1/4$. Poznamenajme ešte, ak $\beta < \beta'$ tak aj $\mathcal{N}(\beta) \subset \mathcal{N}(\beta')$. Tiež $\mathcal{N}(0)$ je iba množina takých bodov (x,z), pre ktoré komponenty XZe sú totožné.

Prediktor-korektor algoritmus pozostáva z dvoch typov krokov. V prvej a každej ďalšej nepárnej iterácií postupujeme prediktor krokom, pred ktorým budeme predpokladať:

$$(x,z) \in \mathcal{N}(1/4).$$

Smer kroku vypočítame z rovníc (2.1) a (2.2) použitím hodnoty parametra $\delta = 0$. Dĺžku kroku θ určíme tak, aby $(x+\theta\Delta x, z+\theta\Delta z) \in \mathcal{N}(1/2)$, ale pritom bola maximálna možná:

$$\theta = \max\,\{t : (x + t\Delta x, z + t\Delta z) \in \mathcal{N}(1/2)\}.$$

V párnych iteráciach algoritmus postupuje korektor krokom, pred ktorým budeme predpokladať:

$$(x,z) \in \mathcal{N}(1/2),$$

čo je garantované výberom dĺžky prediktor kroku. Smer kroku vypočítame opäť z rovníc (2.1) a (2.2) použitím hodnoty parametra $\delta = 1$, pričom dĺžku kroku θ stanovíme na 1.

Nasledujúca veta ukazuje, že výsledok po každej iterácii algoritmu spĺňa predpoklad pre nasledujúcu iteráciu algoritmu a zároveň, že nekomplementarita μ klesá pri prediktor kroku pokiaľ je to možné a nemení sa pri korektor kroku.

Veta 2.3. [Vanderbei[2], str.354]

Nasledujúce tvrdenia sú pravdivé:

1. Po prediktor kroku $(\widetilde{x},\widetilde{z}) \in \mathcal{N}(1/2)$ *a* $\widetilde{\mu} = (1-\theta)\mu$

2. Po korektor kroku $(\widetilde{x},\widetilde{z}) \in \mathcal{N}(1/4)$ *a* $\widetilde{\mu} = \mu$

Veta 2.3 ukazuje, že prediktor-korektor algoritmus je dobre definovaný. Nasledujúca veta udáva dolnú hranicu, ktorou napreduje každý prediktor krok.

Veta 2.4. [Vanderbei[2], str.357]
V každom prediktor kroku $\theta \geqslant \frac{1}{2\sqrt{n}}$.

Teraz ukážeme, že prediktor-korektor algoritmus nie je len dobre definovaný, ale že aj konverguje. Nech $(x^{(k)}, z^{(k)})$ označuje riešenie po k-tej iterácií a nech:

$$\rho^{(k)} = \rho(x^{(k)}, z^{(k)}) \quad \text{a} \quad \mu^{(k)} = \mu(x^{(k)}, z^{(k)}).$$

Algoritmus začína so štartovacími hodnotami $x^{(0)} = z^{(0)} = e$. Teda $\mu^{(0)} = 1$. Naším cieľom je ukázať, že ak ide k do nekonečna, potom $\rho^{(k)}$ a $\mu^{(k)}$ idú k nule. Predchádzajúca veta spolu s vetou 2.3 spolu dávajú, že po každej párnej iterácií, dajme tomu 2k-tej, platí nasledujúca nerovnosť:

$$\mu^{(2k)} \leqslant \left(1 - \frac{1}{2\sqrt{n}}\right)^k.$$

Keďže korektor krok nemení hodnotu μ platí:

$$\mu^{(2k-1)} = \mu^{(2k)}.$$

Z týchto dvoch výrazov je teraz zrejmé, že platí:

$$\lim_{k\to\infty} \mu^{(k)} = 0$$

Pozrime sa teraz na $\rho^{(k)}$. Z časti (2) a (3) vety 2.2 dostávame, že súčasne s redukciou neprípustnosti redukujeme aj nekomplementaritu. Teda

$$\rho^{(k)} = \mu^{(k)}\rho^{(0)}.$$

A preto, keď $\mu^{(k)}$ ide k nule, aj $\rho^{(k)}$ ide k nule. Vyššie uvedené môžeme zhrnúť do nasledujúcej vety:

Veta 2.5. [Vanderbei[2], str.358]
Limity $x^* = \lim_{k\to\infty} x^{(k)}$ *a* $z^* = \lim_{k\to\infty} z^{(k)}$ *existujú a* (x^*, z^*) *je optimálne riešenie. Navyše vektory* x^* *a* z^* *sú ostro komplementárne, t.j.* $\forall j:\ x_j^* z_j^* = 0$*, pričom buď* $x_j^* > 0$ *alebo* $z_j^* > 0$*.*

Samozrejme v reálnej praxi nemôžeme nechať bežať nekonečne veľa iterácií. Namiesto toho stanovíme nejakú hranicu a skončíme keď $\mu^{(k)}$ klesne pod túto hranicu. Hranica sa zvyčajne udáva na 2^{-L}, kde L je nejaké kladné číslo. Obvykle požadujeme, aby hranica bola okolo 10^{-8}, čo zodpovedá hodnote $L \approx 26$.

Metóda s dlhým krokom:

Stručne uvedieme ešte iný, veľmi podobný, algoritmus. Variant algoritmu s dlhým krokom spočíva v zväčšení množiny $\mathcal{N}(\beta)$. Definujme teda pre $0 \leqslant \beta \leqslant 1$ množinu

$$\mathcal{M}(\beta) = \{(x, z) > 0 : \min XZe \geqslant (1-\beta)\mu(x, z)\},$$

kde minXZe znamená minimum zo všetkých komponentov XZe. Poznamenajme, že $\mathcal{N}(\beta) \subset \mathcal{M}(\beta) \subset \mathcal{M}(1) = (x, z) > 0$.

Obdobne ako v metóde s krátkym krokom pred prediktor krokom budeme predpokladať, že:

$$(x, z) \in \mathcal{M}(1/4).$$

Smer kroku vypočítame rovnako ako v predchádzajúcej metóde a dĺžku kroku zvolíme:

$$\theta = \max\{t : (x + t\Delta x, z + t\Delta z) \in \mathcal{M}(1/2)\}.$$

Takisto dĺžku a smer korektor kroku určíme obdobne ako v metóde s krátkym krokom.

Redukovaný KKT systém:

Časovo najnáročnejším krokom každej iterácie prediktor-korektor algoritmu je určenie smeru kroku. Ten sa určuje riešením KKT systému:

$$A\Delta x + \Delta z = -(1-\delta)\rho(x, z) \tag{2.1}$$

$$Z\Delta x + X\Delta z = \delta\mu(x, z)e - XZe \tag{2.2}$$

Najschodnejšou cestou ako tento systém vyriešiť sa zdá byť vyjadrenie Δz z rovnice (2.2):

$$\Delta z = -X^{-1}Z\Delta x + \delta\mu X^{-1}e - z \tag{2.3}$$

a následne dosadenie (2.3) do (2.1):

$$(A - X^{-1}Z)\Delta x = -(1-\delta)\rho + z - \delta\mu X^{-1}e \tag{2.4}$$

Systém rovníc (2.4) budeme nazývať podľa Vanderbeia[2] redukovaný KKT systém a jeho riešením sa budeme zaoberať neskôr.

2.4 Transformácia štandartnej úlohy na homogénnu samoduálnu úlohu

V reálnej praxi sa homogénna samoduálna úloha vyskytuje len zriedkavo. Ako však ukážeme v tejto časti úlohu lineárneho programovania v štandartnom tvare vieme veľmi ľahko previesť na problém riešenia homogénnej samoduálnej úlohy.

Uvažujme teda primárno-duálnu dvojicu úloh lineárneho programovania v štandartnom tvare:

$$\begin{array}{ll} (P) & \max \{ \; c^T x \mid \quad Ax \leqslant b \; ; \quad x \geqslant 0 \; \} \\ (D) & \min \{ \; b^T y \mid \quad A^T y \geqslant c \; ; \quad y \geqslant 0 \; \}, \end{array}$$

Tento problém môžeme vyriešiť pomocou iného väčšieho problému, ktorý vznikne kombináciou primárnej a duálnej úlohy.

$$\begin{array}{rr} \max & 0 \\ \text{pri ohraničeniach:} & -A^T y + c\phi \leqslant 0 \; , \\ & Ax - b\phi \leqslant 0 \; , \\ & -c^T x + b^T y \leqslant 0 \; , \\ & x, y, \phi \geqslant 0. \end{array} \tag{2.5}$$

Takouto kombináciou primárnej a duálnej úlohy nám pribudla jedna nová premenná ϕ a jedno nové ohraničenie $-c^T x + b^T y \leqslant 0$. Teda celkovo dostávame n+m+1 premenných a tiež n+m+1 ohraničení. Účelová funkcia a pravé strany v (2.5) sú nulové a navyše matica ohraničení takejto úlohy

$$\widetilde{A} = \begin{bmatrix} 0 & -A^T & c \\ A & 0 & -b \\ -c^T & b^T & 0 \end{bmatrix} \tag{2.6}$$

je *kososymetrická*. Teda táto úloha je homogénnou samoduálnou úlohou lineárneho programovania.

Nerovnosti v ohraničeniach úlohy (2.5) odstránime ako obvykle pridaním doplnkových premenných:

$$
\begin{aligned}
\max \qquad & \quad 0 \\
\text{pri ohraničeniach:} \qquad & -A^T y + c\phi + z = 0 \ , \\
& Ax - b\phi + w = 0 \ , \\
& -c^T x + b^T y + \psi = 0 \ , \\
& x, y, \phi, w, z, \psi \geqslant 0.
\end{aligned} \tag{2.7}
$$

Označme vektor nových premenných ako $\widetilde{x}$ a vektor doplnkových premenných ako $\widetilde{z}$, t.j.:

$$
\widetilde{x} = \begin{bmatrix} x \\ y \\ \phi \end{bmatrix}, \quad \widetilde{z} = \begin{bmatrix} z \\ w \\ \psi \end{bmatrix} \tag{2.8}
$$

Potom úlohu (2.7) môžeme zapísať v tvare:

$$
\max \{ \ 0 \mid \quad \widetilde{A}\widetilde{x} + \widetilde{z} = 0 \ ; \quad \widetilde{x}, \widetilde{z} \geqslant 0 \ \}. \tag{2.9}
$$

2.5 Vzt'ah optimálneho riešenia homogénnej samoduálnej úlohy a optimálneho riešenia štandartnej úlohy

Časť 1 Vety 2.1 o existencii prípustného riešenia a jeho optimálnosti zaručuje, že riešením homogénnej samoduálnej úlohy dostaneme nejaké prípustné riešenie $(\overline{x}, \overline{y}, \overline{\phi}, \overline{w}, \overline{z}, \overline{\psi})$, Toto prípustné riešenie $(\overline{x}, \overline{y}, \overline{\phi}, \overline{w}, \overline{z}, \overline{\psi})$ je podľa tej istej vety navyše aj optimálne a teda musia preň platiť podmienky komplementarity. Teda platí:

$$
\begin{aligned}
&\overline{x}_j.\overline{z}_j = 0, && \forall \ j = 1, ..., n \\
&\overline{y}_i.\overline{w}_i = 0, && \forall \ j = 1, ..., m \\
&\overline{\phi}.\overline{\psi} = 0.
\end{aligned}
$$

Pretože homogénna samoduálna úloha má prípustné riešenie, a teda podľa Vety 2.1 aj optimálne riešenie, tak potom z Vety 1.4 o existencii ostro komplentárneho riešenia máme zaručenú aj existenciu ostro komplementárneho riešenia pre homogénnu samo-duálnu úlohu:

$$
\begin{aligned}
&\overline{x}_j + \overline{z}_j > 0, && \forall j = 1, ..., n\\
&\overline{y}_i + \overline{w}_i > 0, && \forall i = 1, ..., m\\
&\overline{\phi} + \overline{\psi} > 0.
\end{aligned}
$$

Naším cieľom je teraz ukázať, že úlohu lineárneho programovania je možné riešiť pomocou homogénnej samoduálnej úlohy a to v zmysle, že z optimálneho riešenia homogénnej samoduálnej úlohy (2.5) vieme určiť optimálne riešenia primárnej úlohy (P) resp. duálnej úlohy (D). Nech $(\overline{x}, \overline{y}, \overline{\phi})$ je optimálne riešenie úlohy (2.5). Optimálna hodnota $\overline{\phi}$ môže byť kladná alebo nulová. V prípade, že $\overline{\phi} > 0$ zadefinujme $x^* = \overline{x}/\overline{\phi}$ a $y^* = \overline{y}/\overline{\phi}$. V ďalšej časti ukážeme, že x^* je optimálne riešenie primárnej úlohy (P) a y^* je optimálne riešenie duálnej úlohy (D). Pre prípad $\overline{\phi} = 0$ ukážeme, že nastáva neprípustnosť v primárnej alebo duálnej úlohe (alebo v oboch). Tieto poznatky môžeme sformulovať do nasledujúcej vety:

Veta 2.6. [Vanderbei[2], str.360]
Nech $(\overline{x}, \overline{y}, \overline{\phi}, \overline{w}, \overline{z}, \overline{\psi})$ je ostro komplementárnym prípustným (teda aj optimálnym) riešením úlohy 2.6:
1. Ak $\overline{\phi} > 0$, potom $x^* = \overline{x}/\overline{\phi}$ je optimálnym riešením prímarnej úlohy (P) a $y^* = \overline{y}/\overline{\phi}$ je optimálnym riešením duálnej úlohy (D).
2. Ak $\overline{\phi} = 0$, potom buď $c^T\overline{x} > 0$ alebo $b^T\overline{y} < 0$.
a) Ak $c^T\overline{x} > 0$, potom duálna úloha je neprípustná.
b) Ak $b^T\overline{y} < 0$, potom primárna úloha je neprípustná.

Dôkaz. (1) Prenásobme ohraničenia úlohy 2.5 výrazom $\overline{\phi}^{-1}$. Dostaneme:

$$
\begin{array}{rrrl}
 & -A^Ty^* & +c & \leqslant 0\\
Ax^* & & -b & \leqslant 0\\
-c^Tx^* & +b^Ty^* & & \leqslant 0
\end{array}
$$

Je zrejmé, že x^* aj y^* sú nezáporné (lebo $\overline{x} \geqslant 0, \overline{y} \geqslant 0, \overline{\phi} \geqslant 0$). Preto teda x^* je prípustné riešenie primárnej úlohy a y^* je prípustné riešenie duálnej úlohy. Navyše, ak dáme dokopy slabú vetu o dualite (Veta 1.1) a tretiu nerovnosť dostávame $c^T x^* = b^T y^*$, čo značí, že x^* je optimálne riešenie primárnej úlohy a y^* je optimálne riešenie duálnej úlohy.

(2) Keďže $\overline{\phi} = 0$, z ostrej komplementarity prípustného riešenia nutne dostávame $\overline{\psi} > 0$. Z toho $\overline{x}$ a $\overline{y}$ musia spĺňať:

$$A^T\overline{y} \geqslant 0$$

$$A\overline{x} \leqslant 0$$

$$b^T\overline{y} < c^T\overline{x}$$

Z poslednej nerovnosti vyplýva, že nemôže nastať prípad, keď $b^T\overline{y} \geqslant 0$ a súčasne $c^T\overline{x} \leqslant 0$. Preto teda buď $c^T\overline{x} > 0$ alebo $b^T\overline{y} < 0$ (alebo oba).
Časť 2a) vety dokážeme sporom. Predpokladajme, že existuje nejaké prípustné riešenie duálnej úlohy y^0. To znamená, že $y^0 \geqslant 0$ a $A^T y^0 \geqslant c$. Ak prenásobíme nerovnosť $A^T y^0 > c$ zľava $\overline{x}^T$ dostávame:

$$\overline{x}^T A^T y^0 \geqslant \overline{x}^T c \tag{2.10}$$

($\overline{x} \geqslant 0$, teda nerovnosť ostáva zachovaná v pôvodnom smere).
Keďže platí $A\overline{x} \leqslant 0$ a súčasne $y^0 \geqslant 0$, potom:

$$\overline{x}^T A^T y^0 \leqslant 0. \tag{2.11}$$

Ak dáme dokopy nerovnosti (2.10) a (2.11) spolu z predpokladom časti 2a) $\overline{x}^T c > 0$ dostávame nasledujúci sled nerovností:

$$0 \geqslant \overline{x}^T A^T y^0 \geqslant \overline{x}^T c > 0,$$

čo je spor a teda duálna úloha nemá prípustné riešenie.
Časť 2b) sa dokáže analogicky. □

2.6 Redukovaný KKT systém pre štandartnú úlohu

V časti 2.5 sme ukázali ako pomocou ostro komplementárneho optimálneho riešenia príslušnej homogénnej samoduálnej úlohy určiť riešenie primárnej a duálnej úlohy. Na získanie ostro komplementárneho riešenia homogénnej samoduálnej úlohy budeme používať algoritmus, ktorý sme popísali v častiach 2.2 - 2.3. Ako sme uviedli časovo najnáročnejším krokom takéhoto algoritmu je vyriešenie KKT systému. Preto v tejto kapitole ukážeme ako vyriešiť KKT systém pre úlohu (2.7). Princíp bude spočívať v transformácii redukovaného KKT systému na systém s kvázidefinitnou maticou. Pod kvázidefinitnou maticou budeme rozumieť symetrickú maticu v tvare

$$B = \begin{bmatrix} -E & A \\ A^T & D \end{bmatrix},$$

kde E a D sú symetrické, kladne definitné matice.

Systém rovníc $Bx = b$, kde B je kvázidefinitná matica vieme ľahko riešiť pomocou Choleského rozkladu matice B: $B = LDL^T$, kde L je dolná trojuholníková matica a D je diagonálna matica. Pomocou Choleského rozkladu potom môžeme prepísať pôvodný systém $Bx = b$ na dva čiastkové systémy rovníc:

$$Ly = b$$

$$DL^T x = y,$$

ktoré sú ľahko riešiteľné, keďže L je dolná trojuholníková a DL^T horná trojuholníková matica.

Načrtli sme ako riešiť redukovaný KKT systém s kvázidefinitnou maticou. Teraz sa bližšie pozrieme na úlohu (2.9), k nej prislúchajúci redukovaný KKT systém a ako tento tento redukovaný KKT systém previesť na systém s kvázidefinitnou maticou.

Najskôr zadefinujeme:

$$\begin{aligned} \sigma &:= -A^T y + c\phi + z \\ \rho &:= Ax - b\phi + w \\ \gamma &:= -c^T x + b^T y + \psi. \end{aligned}$$

Poznamenajme, že podobne ako v kapitole 2.2 vektorová veličina ρ vyjadrovala neprípustnosť (x,z), tak aj tu vektor(σ, ρ, γ) vyjadruje neprípustnosť riešenia (x, y, ϕ, w, z, ψ).

Ak pre úlohu (2.9) teraz použijeme analógiu vzťahu (2.4) pre redukovaný KKT systém a dosadíme doň namiesto matice A maticu $\widetilde{A}$ (2.6) a namiesto matíc X,Z matice $\widetilde{X}, \widetilde{Z}$ prislúchajúce k vektorom $\widetilde{x}$ a $\widetilde{z}$ (2.8), získame redukovaný KKT systém pre úlohu (2.7):

$$\begin{bmatrix} 0 & -A^T & c \\ A & 0 & -b \\ -c^T & b^T & 0 \end{bmatrix} - \begin{bmatrix} X^{-1}Z & & \\ & W^{-1}Y & \\ & & \psi/\phi \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta\phi \end{bmatrix} =$$
$$= -(1-\delta)\begin{bmatrix} \sigma \\ \rho \\ \gamma \end{bmatrix} + \begin{bmatrix} z \\ w \\ \psi \end{bmatrix} - \delta\mu \begin{bmatrix} X & & \\ & Y & \\ & & \phi \end{bmatrix}^{-1} e \tag{2.12}$$

Po úprave môžeme tento systém zapísať nasledovne:

$$\begin{bmatrix} -X^{-1}Z & -A^T & c \\ A & -Y^{-1}W & -b \\ -c^T & b^T & -\psi/\phi \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta\phi \end{bmatrix} = \begin{bmatrix} \widehat{\sigma} \\ \widehat{\rho} \\ \widehat{\gamma} \end{bmatrix}, \tag{2.13}$$

kde

$$\begin{bmatrix} \widehat{\sigma} \\ \widehat{\rho} \\ \widehat{\gamma} \end{bmatrix} = \begin{bmatrix} -(1-\delta)\sigma + z - \delta\mu X^{-1}e \\ -(1-\delta)\rho + w - \delta\mu Y^{-1}e \\ -(1-\delta)\gamma + \psi - \delta\mu/\phi \end{bmatrix}.$$

Vidíme,že tento systém nie je symetrický. Ako sme však už uviedli na začiatku tejto kapitoly, naším cieľom bude previesť tento systém na systém s kvázidefinitnou maticou. Transformáciu začneme vyjadrením Δx a Δy z prvých dvoch rovníc redukovaného KKT systému (2.13):

$$\begin{bmatrix} -X^{-1}Z & -A^T \\ A & -Y^{-1}W \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} + \begin{bmatrix} c \\ -b \end{bmatrix} \Delta\phi = \begin{bmatrix} \widehat{\sigma} \\ \widehat{\rho} \end{bmatrix}$$

Teda

$$\begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} -X^{-1}Z & -A^T \\ A & -Y^{-1}W \end{bmatrix}^{-1} \left(\begin{bmatrix} \widehat{\sigma} \\ \widehat{\rho} \end{bmatrix} - \begin{bmatrix} c \\ -b \end{bmatrix} \Delta\phi \right). \tag{2.14}$$

Zavedením nových premenných $f_x,\ f_y,\ g_x,\ g_y$ môžeme zapísať systém rovníc (2.14) v tvare:

$$\begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} f_x \\ f_y \end{bmatrix} - \begin{bmatrix} g_x \\ g_y \end{bmatrix} \Delta\phi, \tag{2.15}$$

pričom vektory $f = \begin{bmatrix} f_x \\ f_y \end{bmatrix}$ a $g = \begin{bmatrix} g_x \\ g_y \end{bmatrix}$ nájdeme ako riešenia dvoch systémov rovníc:

$$\begin{bmatrix} -Y^{-1}W & A \\ A^T & X^{-1}Z \end{bmatrix} \begin{bmatrix} f_y \\ f_x \end{bmatrix} = \begin{bmatrix} \widehat{\rho} \\ -\widehat{\sigma} \end{bmatrix}$$

a

$$\begin{bmatrix} -Y^{-1}W & A \\ A^T & X^{-1}Z \end{bmatrix} \begin{bmatrix} g_y \\ g_x \end{bmatrix} = \begin{bmatrix} -b \\ -c \end{bmatrix}.$$

Ak označíme $X^{-1}Z = D$ a $Y^{-1}W = E$ je zrejmé, že tieto dva systémy rovníc sú systémy s kvázidefinitnou maticou, ktoré už vieme riešiť.

Teda teraz už sme schopní nájsť optimálne riešenie úlohy (2.9). Ak dosadíme (2.15) do tretej rovnice (2.13) získame rovnicu pre určenie smeru $\Delta\phi$ premennej ϕ:

$$\begin{bmatrix} -c^T & b^T \end{bmatrix} \left(\begin{bmatrix} f_x \\ f_y \end{bmatrix} - \begin{bmatrix} g_x \\ g_y \end{bmatrix} \Delta\phi \right) - \frac{\psi}{\phi}\Delta\phi = \widehat{\gamma}.$$

Z toho

$$\Delta\phi = \frac{c^T f_x - b^T f_y + \widehat{\gamma}}{c^T g_x - b^T g_y - \psi/\phi}.$$

Pri danom $\Delta\phi$ systém rovníc (2.14) určuje Δx a Δy. Pre výpočet smeru kroku doplnkových premenných použijeme analógiu rovnice (2.3) a už vyrátane hodnoty Δx, Δy a $\Delta\phi$:

$$\begin{aligned} \Delta z &= -X^{-1}Z\Delta x + \delta\mu X^{-1}e - z \\ \Delta w &= -Y^{-1}W\Delta y + \delta\mu Y^{-1}e - w \\ \Delta\psi &= -\frac{\psi}{\phi}\Delta\phi + \delta\frac{\mu}{\phi} - \psi. \end{aligned}$$

Teraz už môžeme uviesť celý algoritmus homogénnej samoduálnej metódy [Vanderbei[2], str.364]:

incializácia

$(x, y, \phi, z, w, \psi) = (e, e, 1, e, e, 1)$

pokiaľ *(nie je optimálne)* {

$\mu = (z^T x + w^T y + \psi\phi)/(n + m + 1)$

$$\delta = \begin{cases} 0, & \text{v párnych iteráciach} \\ 1, & \text{v nepárnych iteráciach} \end{cases}$$

$\widehat{\rho} = -(1-\delta)(Ax - b\phi + w) + w - \delta\mu Y^{-1}e$

$\widehat{\sigma} = -(1-\delta)(A^T y - c\phi + z) + z - \delta\mu X^{-1}e$

$\widehat{\gamma} = -(1-\delta)(b^T y - c^T x + \psi) + \psi - \delta\mu/\phi$

riešiť dva $(n+m) \times (n+m)$ **kvázidefinitné systémy:**

$$\begin{bmatrix} -Y^{-1}W & A \\ A^T & X^{-1}Z \end{bmatrix} \begin{bmatrix} f_y \\ f_x \end{bmatrix} = \begin{bmatrix} \widehat{\rho} \\ -\widehat{\sigma} \end{bmatrix}$$

a

$$\begin{bmatrix} -Y^{-1}W & A \\ A^T & X^{-1}Z \end{bmatrix} \begin{bmatrix} g_y \\ g_x \end{bmatrix} = \begin{bmatrix} -b \\ -c \end{bmatrix}$$

vyrátať smer kroku:

$$\Delta\phi = \frac{c^T f_x - b^T f_y + \widehat{\gamma}}{c^T g_x - b^T g_y - \psi/\phi}$$

$$\begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} f_x \\ f_y \end{bmatrix} - \begin{bmatrix} g_x \\ g_y \end{bmatrix} \Delta\phi$$

$$\Delta z = -X^{-1}Z\Delta x + \delta\mu X^{-1}e - z$$

$$\Delta w = -Y^{-1}W\Delta y + \delta\mu Y^{-1}e - w$$

$$\Delta\psi = -\frac{\psi}{\phi}\Delta\phi + \delta\frac{\mu}{\phi} - \psi$$

vyrátať dĺžku kroku:

$$\theta = \begin{cases} \max\{t : (x(t), ..., \psi(t)) \in \mathcal{N}(1/2)\}, & \text{v párnych iteráciach} \\ 1, & \text{v nepárnych iteráciach} \end{cases}$$

vyrátať nový bod:

$$\begin{array}{ll} x \leftarrow x + \theta\Delta x, & z \leftarrow z + \theta\Delta z \\ y \leftarrow y + \theta\Delta y, & w \leftarrow w + \theta\Delta w \\ \phi \leftarrow \phi + \theta\Delta\phi, & \psi \leftarrow \psi + \theta\Delta\psi \end{array}$$

}.

3. Lineárna optimalizácia vo financiách

Väčšina finančných a bankových inštitúcií potrebuje zhodnocovať svoj majetok formou investovania finančných prostriedkov. Dôvody môžu byť rôzne, či už potrebujú vykonať nejaké platby v budúcich obdobiach ako napríklad splátky pôžičiek, alebo len chcú dosiahnuť zisk. Iným príkladom budúcich záväzkov sú domáce hypotéky či už s fixovanou alebo premenlivou platbou typicky splatné v priebehu 15-30 rokov. Veľmi často veľkosť platieb a čas ich realizácie nemusia byť v súčasnosti presne známe. Príkladom takýchto spoločnosti, ktoré očakávajú takéto stochastické platby sú poisťovne s niektorými svojími produktmi ako je napríklad životné poistenie. Pre prípad životného poistenia závisí výška platby a čas splátky od dĺžky života zákazníka.

Často používaným spôsobom ako sa zabezpečiť voči takýmto budúcim platbám je investovanie finančných prostriedkov do nejakých aktív. Ako takéto aktívum sa veľmi často používajú dlhopisy. Cash-flow takéhoto portfólia dlhopisov (aktív) musí kopírovať v každom časovom okamihu výšku záväzkov splatných v danom období. Úlohou tejto časti práce je ukázať ako takéto portfólio skonštruovať, t.j. bude nás zaujímať aké dlhopisy a v akých množstvách máme teraz nakúpiť. Ukážeme dva typy modelov, jednak s minimalizáciou nákladov na obstaranie portfólia a jednak s maximalizáciou výnosu na konci sledovaného obdobia. Uvidíme, že takýto problém vedie k úlohe lineárneho programovania, ktorej metódu riešenia sme popísali v kapitole 2.

3.1 Deterministický model s minimalizáciou nákladov

Uvažujme najskôr deterministický príklad, kde aktíva ako aj záväzky sú dopredu známe. Pred formuláciou modelu si uvedieme označenie, ktoré v modeloch budeme používať. Označme teda:

$T = \{0, ..., m\}$ - množina diskrétnych časových okamihov (periód) t udávaných v rokoch, pričom $t = 0$ (teraz) až $t = m$ (horizont)

$U = \{1, ..., n\}$ - indexová množina možných dlhopisov

L_t - výška záväzku splatná v čase $t \geqslant 1$

$F_{i,t}$ - cash-flow vyplývajúci z dlhopisu i v čase t.

Poznamenajme, že veľkosť časového okamžiku je určená intenzitou vyplácania kupónov. Teda napríklad ak sa kupón vypláca každých 6 mesiacov, veľkosť jednej periódy bude 0.5 roka. Ďalej cash-flow $F_{i,t}$ môže byť kladný aj záporný v závislosti od smeru toku peňazí. Zaužívaná konvencia však hovorí, že kladný cash-flow ukazuje na prichádzajúce peniaze, kým záporný cash-flow na odchádzajúce peniaze. Teda cash-flow pre k-ročný dlhopis vyplácajúci kupóny raz za rok je v čase 0 záporný a to vo výške $F_{i,0} = -P_i$, kde P_i je cena, ktorú treba zaplatiť za kúpu jedného dlhopisu. V nasledujúcich rokoch $t = 1, ..., k-1$ je výška cash-flowu pochádzajúceho z dlhopisu rovná výške vyplácaného dlhopisu: $F_{i,t} = c_i$. V poslednom roku k v čase expirácie okrem kupónu získame aj nominálnu hodnotu (face value): $F_{i,k} = c_i + FV$. Ako sme uviedli cash-flow z dlhopisového portfólia využívame na splátky našich záväzkov. Niekedy je ale možné uvažovať, že nesplácame len naše záväzky ale dostávame aj splátky našich pohľadávok. V takomto prípade prichádzajúce platby označujeme záporným znamienkom a odchádzajúce kladným.

Prvý model, ktorý uvedieme je deterministický buy&hold model, t.j. na začiatku v čase $t = 0$ určíme koľko a akých dlhopisov zahrnieme do portfólia a tie budeme držať v portfóliu v určených množstvách počas celého sledovaného obdobia.

$$
\begin{aligned}
&\min_{x,\lambda} \lambda \\
&\text{pri ohraničeniach:} \\
&\sum_{i \in U} F_{i,0} x_i + \lambda \geqslant 0, \\
&\sum_{i \in U} F_{i,t} x_i \geqslant L_t, \quad t \geqslant 1, \\
&x_i \geqslant 0, \quad i \in U,
\end{aligned}
$$

kde x je vektor, ktorého zložky x_i vyjadrujú množstvo dlhopisu typu i a λ zodpovedá nákladom potrebným na obstaranie portfólia na začiatku, a preto sa niekedy aj nazýva paušál (lump sum). Tento model budeme nazývať Cash-Flow Matching Model 1 (CFM-1) [Nielsen[1]].

Takto formulovaná úloha nám zaručuje, že v každom časovom okamihu t cash-flow prislúchajúci k nášmu portfóliu dosahuje aspoň hodnotu záväzku splatného v čase t. Zároveň z pomedzi všetkých prípustných portfólií vyberá to s najnižšími obstarávacími nákladmi.

Zdá sa byť prirodzené formulovať tento model v tvare s účelovou funkciou

$$\min_x \sum_{i \in U} P_i x_i,$$

odstrániť tak prvé ohraničenie a vždy držať cash-flow kladný. V prípade dlhopisového portfólia je táto myšlienka správna, ale v prípadoch kedy cash-flow môže byť kladný aj záporný (napr.: investície, ktoré sú platené niekoľko rokov predtým ako začnú prinášať kladný cash-flow alebo fixed-for-floating swapy) už takýto postup nemôžeme aplikovať.

Teda formulácia s účelovou funkciou v tvare ako v CFM-1 je všeobecnejšia a zahŕňa širšiu triedu investícií.

V CFM-1 modeli jediná cesta k dosiahnutiu cash-flow v čase t, ktorý by pokrýval záväzky je investícia v čase $t = 0$, ktorá prináša v čase t požadovaný cash-flow. Niekedy môže byť ale výhodnejšie kúpiť za oveľa lepšiu cenu dlhopis, ktorý maturuje pred časom t a jednoduchým uložením peňazí v banke dosiahneme požadovanú hodnotu záväzku v čase t. Obdobne si môžeme požičať peniaze z banky pri nejakom úroku oproti cenine, ktorá maturuje v budúcnosti. Krátkodobé reinvestovanie a požičiavanie môže byť modelované pridaním premenných r_t a b_t, ktoré reprezentujú množstvo peňazí reinvestovaných z periódy t do t+1 resp. množstvo peňazí požičaných požičaných z periódy t do t+1. Predpokladáme, že budúce krátkodobé investície majú návratnosť ρ_t za jednú periódu a peniaze sú požičiavané pri úroku β_t. Tento model budeme nazývať CFM-2 alebo tiež aj Cash-Flow Matching with reinvesting and borrowing [Nielsen[1]]:

$$
\begin{aligned}
&\min_{x,\lambda,r,b} \lambda \\
&\text{pri ohraničeniach:} \\
&\sum_{i \in U} F_{i,0} x_i + b_0 + \lambda = r_0, \\
\sum_{i \in U} F_{i,t} x_i + (1+\rho_t) r_{t-1} + b_t &= L_t + r_t + (1+\beta_t) b_{t-1}, \quad t \geqslant 1, \\
&b_m = 0, \\
&x_i, r_i, b_i \geqslant 0, \quad i \in U.
\end{aligned}
$$

Podmienka $b_m = 0$ nám vyjadruje skutočnosť, že v poslednej perióde je požičiavanie zakázané. Reinvestovanie v poslednej perióde je síce povolené, ale v podstate zbytočné. Premenná r_m môže byť však interpretovaná ako prebytok na konci, v horizonte. Oproti modelu CFM-1 sa v dôsledku možnosti reinvestovania resp. požičiavania zmenili nerovnosti v ohraničeniach na rovnosti. Je totiž pre nás výhodnejšie peniaze neinvestované do dlhopisov vložiť do banky.

V nasledujúcej časti uvedieme niektoré rozšírenia modelu, ktoré sa snažia model priblížiť viac realite. S každou investíciou sú spojené transakčné náklady ako burzové poplatky a bid-ask spread. Budeme rozlišovať medzi dvoma typmi transakčných nákladov a to fixnými a variabilnými. Fixné náklady sa uplatňujú na každú ceninu, ktorá je obchodovaná. Môžu byť modelované zavedením binárnej premennej $z_i \in \{0, 1\}, i \in U$ pre každú ceninu a pridaním výrazu

$$F \sum_{i \in U} z_i$$

do účelovej funkcie, pričom F je výška fixných nákladov. Táto nová premenná z_i potrebuje byť prepojená na x_i tak, že ak $x_i > 0$ potom $z_i = 1$, inak $z_i = 0$. Množina ohraničení $x_i \leqslant M z_i$, kde M je dostatočne veľké číslo nám zabezpečí takýto vzťah. Takto namodelovaný problém je však veľmi ťažko riešiteľný, keďže obsahuje celočíselné premenné to je aj jeden z dôvodov prečo sa transakčné náklady často zanedbávajú.

Variabilné transakčné náklady sú na druhej strane veľmi ľahko modelovatelné bez zvýšenia zložitosti alebo obtiažnosti modelu. Variabilné transakčné náklady sa uplatňujú na jednotku nákladov a tak môžu byť jednoducho pridané k nákladom na obstaranie ceniny alebo presnejšie odčítané z $F_{i,0}$ v CFM modeloch.

Jedným z dôvodov prečo zahrnúť transakčné náklady do modelu je, že väčšina investorov nedrží rovnaké portfólio počas celého obdobia, ale reaguje na situáciu na trhu a reoptimalizuje alebo rebalancuje svoje portfólio. Vynechaním transakčných nákladov v prípadoch kedy investor predá celé svoje portfólio a kúpi úplne nové môže dôjsť k pomerne veľkému rozdielu oproti skutočnosti.

Ďalším problémom je, že v modeli predpokladáme, že prichádzajúci cash-flow v danej perióde je schopný "stretnúť" sa so záväzkom. To ale vo všeobecnosti nemusí nastať. Ak časové okamžiky sú roky a záväzok je splatný k prvému dňu roka, potom cash-flow z aktíva ako napríklad dlhopis ešte nebude dostupný. Jedným z riešení je rozšíriť množinu časových okamžikov T zahrnúc do nej všetky cash-flow vrátane dátumov splatnosti záväzkov. To ale vedie k zbytočne veľkej úlohe. Iným spôsobom je predpokladať, že peniaze môžu byť požičané oproti budúcim prichádzajúcim cash-flow (stále v tej istej perióde), a že prichádzajúca hotovosť, ktorá prichádza predtým ako je potrebná môže byť investovaná kým nie je potrebná. V takomto prípade môže byť cash-flow prispôsobený času. Napríklad ak máme kupónový dlhopis vyplácajúci kupón 6 mesiacov po splatnosti záväzku môžeme si pri aktuálnom úroku β požičať oproti tomuto kupónu hotovosť na splatenie záväzku.

Potom do modelu namiesto hodnoty kupónu c_i zoberieme hodnotu $c_i(1+\beta)^{-1/2}$ pri diskrétnom úročení resp. $c_i e^{-\beta/2}$ pri spojitom úročení.

3.2 Deterministický model s maximalizáciou výnosu na konci

V predchádzajúcej kapitole sme uviedli CFM modely, ktorých cieľom je minimalizovať náklady na obstaranie portfólia pri splnení istých ohraničení. Pre väčšinu spoločností však nie je podstatné mať to najlacnejšie portfólio, ale dosiahnuť čo najväčší zisk. Preto v nasledujúcej časti uvedieme iný model založený na maximalizácii zisku.

Oproti modelom CFM zavedieme dve zmeny:

1. účelová funkcia bude teraz namiesto minimalizovania počiatočných nákladov maximalizovať konečnú hotovostnú pozíciu,
2. namiesto pôvodnej účelovej funkcie minimalizácie počiatočných nákladov budeme v modely predpokladať, že máme k dispozícií iba istý obmedzený rozpočet pre investovanie.

$$
\begin{gathered}
\max_{x,h,r,b} h \\
\text{pri ohraničeniach:} \\
\sum_{i\in U} F_{i,0}x_i + b_0 + B = r_0, \\
\sum_{i\in U} F_{i,t}x_i + (1+\rho_t)r_{t-1} + b_t = L_t + r_t + (1+\beta_t)b_{t-1}, \quad t \geqslant 1, \\
b_m = 0, \\
r_m = h, \\
x_i, r_i, b_i \geqslant 0, \quad i \in U,
\end{gathered}
$$

kde B je daný rozpočet a h je hotovostná pozícia v horizonte. Tento model budeme nazývať podľa Nielsena[1] Maximizing horizon position (MAX) model.

Tak ako pri CFM modeli aj pri MAX modeli môžeme urobiť obdobným spôsobom rozšírenia modelu, s tým rozdielom, že pri fixných transakčných nákladoch nebudeme meniť účelovú funkciu ale dostupný rozpočet B.

3.3 Stochastický prípad MAX modelu

Pozrime sa bližšie na úlohu, ktorú sme sformulovali v predchádzajúcej časti. Cieľom je stanoviť množstvá a typ dlhopisov na začiatku sledovaného obdobia tak, aby sme dosiahli maximálny zisk pri známych budúcich cash-flow. Nanešťastie v reálnej praxi nie vždy poznáme budúce dáta. Známymi parametrami modelu sú hodnota vstupného kapitálu na začiatku, ceny a kupóny dlhopisov vypisovaných na začiatku sledovaného obdobia. Nepoznáme kupóny dlhopisov vypisovaných v iných časoch. Takže v čase, keď chceme riešiť úlohu, t.j. na začiatku sledovaného obdobia, poznáme iba výnosy niektorých dlhopisov z ponúkaného balíka, ale nie všetky. Rovnako nepoznáme ani úroky, pri ktorých si budeme požičiavať resp. ukladať peniaze v banke. Nakoniec nepoznáme ani súčasnú hodnotu dlhopisov na konci sledovaného obdobia, ktorú potrebujeme na to, aby sme určili hodnotu držaných prostriedkov na konci. Táto hodnota závisí od doby, ktorú má konkrétny dlhopis do vypršania, od aktuálnych úrokových mier v čase určovania súčasnej hodnoty dlhopisu a od výšky transakčných nákladov na trhu. K tomu ešte často veľké finančné inštitúcie ako banky a poisťovne pracujú so stochastickými záväzkami, takže nepoznáme presne ani budúce záväzky. Teraz sa pokúsime túto stochastiku dostať do MAX modelu. Parametre, ktoré sú stochastické budeme označovať indexom $\widetilde{\ }$:

$$
\begin{gathered}
\max_{x,r,b,h} \widetilde{h} \\
\text{pri ohraničeniach:} \\
\sum_{i \in U} F_{i,0} x_i + b_0 + B = r_0, \\
\sum_{i \in U} \widetilde{F}_{i,t} x_i + (1+\widetilde{\rho}_t)\widetilde{r}_{t-1} + \widetilde{b}_t = \widetilde{L}_t + \widetilde{r}_t + (1+\widetilde{\beta}_t)\widetilde{b}_{t-1}, \quad t \geqslant 1, \\
\widetilde{b}_m = 0, \\
\widetilde{r}_m = \widetilde{h}, \\
x_i, \widetilde{r}_i, \widetilde{b}_i \geqslant 0, \quad i \in U.
\end{gathered}
$$

Tak ako je tento model postavený je príliš všeobecný pre praktické použitie, pričom vôbec nie je jasné, čo znamená max $\widetilde{h}$. Ak stochastický parameter $\widetilde{h}$ má všeobecné spojité rozdelenie pravdepodobnosti je takýto problém ťažko riešiteľný, aj keď je rozmer úlohy veľmi malý. Preto budeme predpokladať, že rozdelenie pravdepodobnosti je diskrétne. Tento predpoklad môžeme považovať za rozumný, pretože ľubovoľné rozdelenie môžeme aproximovať ľubovoľne blízko diskrétnym rozdelením.

Teraz teda môžeme predpokladať, že náhodné dáta sú dané v množinách možných scenárov. Pri daných dátach $(F_{i,t}^s, L_t^s, \rho^s, \beta^s)_{t\in T, i\in U}$ pre každý scenár $s \in S$, kde S je množina všetkých scenárov, vypočítame optimálne hodnoty premenných x, h^s, r^s a b^s. Tiež budeme predpokladať, že pravdepodobnosť realizácie scenára s je p_s. Poznamenajme ešte, že premenné r^s, b^sa h^s sú stochastické v zmysle, že sú závislé od výberu scenára, ale stále ostávajú výstupmi modelu a nie vstupmi. Teraz môžeme na základe vyššie uvedeného sformulovať nový model, ktorý budeme v súlade s Nielsenom[1] označovať ako S-MAX model alebo Maximizing expected horizon return:

$$\max_{x,h,r,b} \sum_{s\in S} p_s h^s$$

pri ohraničeniach:

$$\sum_{i\in U} F_{i,0} x_i + b_0 + B = r_0,$$

$$\sum_{i\in U} F_{i,t}^s x_i + (1+\rho_t^s) r_{t-1}^s + b_t^s = L_t^s + r_t^s + (1+\beta_t^s) b_{t-1}^s, \quad t \geqslant 1,\ s \in S,$$

$$b_m^s = 0, \quad s \in S,$$

$$r_m^s = h^s, \quad s \in S,$$

$$x_i, r_i^s, b_i^s \geqslant 0, \quad i \in U,\ s \in S.$$

3.4 Generovanie scenárov

Je zrejmé, že výber scenárov v S-MAX modeli má veľký vplyv na optimálne riešenie. Preto od množiny scenárov budeme požadovať, aby bola:

1. dostatočne obsiahla t.j. aby zachycovala ako "normálne" tak i extrémne prípady,
2. konzistentná t.j. aby dobre zachycovala korelácie medzi náhodnými dátami.

Obsiahnutie extrémnych prípadov je dôležité, na druhej strane, ale model nemôže byť zaslepený takýmito prípadmi, ktorých nastatie má veľmi nízku pravdepodobnosť a musí sa "pozerať" najmä na "normálne" prípady. Pre S-MAX model takéto extremálne prípady nemajú veľký vplyv na účelovú funkciu ale pri S-CFM modeli môžu výrazne ovplyvniť hľadané portfólio.

Konzistencia je rovnako dôležitá. V skutočnom živote sú udalosti často korelované. Vo finančníctve niektoré ceniny môžu mať platby práve vtedy keď iná cenina nie. Jednoduchým príkladom takýchto cenín môžu byť puty a cally. Ak riziko-neutrálny investor nájde záporne korelované aktíva alebo aktíva, ktoré sú kladne korelované môže ich zahrnúť do modelu tak, aby ich cash-flow zahrňoval ich koreláciu. Množina scenárov musí byť konzistentná aj v čase, keďže niektoré ceniny vyplácajú predpovedateľnú čiastku, ale doba vyplácania je ťažko odhadnuteľná.

New York 7:

Jedným z často používaných príkladov množiny scenárov je The New York 7 (NY-7). Je to množina úrokových scenárov, ktoré používa regulátor trhu v New Yorku. Zahŕňa 7 scenárov vývoja úrokových sadzieb a každá finančná inštitúcia v štáte New York musí každý rok demonštrovať regulátorovi, že pri ich súčasných aktívach a záväzkoch v portfóliu ostanú solventní pri ktoromkoľvek scenári. NY-7 uvažuje paralelné posuny krivky úrokových mier. Začína s aktuálnou krivkou úrokových mier a má časový horizont 10 rokov:

1. Úrokové miery sa nezmenia.
2. Úrokové miery vzrastú o 50 bb ročne počas 10 rokov.
3. Úrokové miery vzrastú o 100 bb ročne počas 5 rokov, a potom poklesnú o 100 bb ročne počas 5 rokov.
4. Úrokové miery vzrastú o 300 bb a potom sa už nebudú meniť.
5. Úrokové miery poklesnú o 50 bb ročne počas 10 rokov.
6. Úrokové miery poklesnú o 100 bb ročne počas 5 rokov a potom vzrastú o 100 bb ročne počas 5 rokov.
7. Úrokové miery poklesnú o 300 bb a potom sa už nebudú meniť.

Pričom bb označuje bázický bod, čo je vlastne $\frac{1}{100}$ percentuálneho bodu. Takže nárast o 50 bb úrokovej miery, povedzme, 5% bude na hodnotu 5,5%.

Binomický strom:

Iným spôsobom ako generovať krivku úrokových mier je použitie binomického stromu. To znamená, že máme binomický strom a rozhodujeme sa, po ktorej vetve stromu sa vyberieme. Predpokladáme, že pravdepodobnosť toho, že úroková miera v danom časovom okamžiku vzrastie je rovnaká ako pravdepodobnosť poklesu úrokovej miery. Generáciu úrokového scenára začneme v časovom okamžiku 0. Pomocou náhodného generátora, ktorý nám zaručí rovnakú pravdepodobnosť nárastu aj poklesu úrokovej miery určíme, ktorou vetvou sa dostaneme v binomickom strome z času 0 do času 1. Príkladmi takýchto náhodných generátorov môžu byť hod mincou, generovanie párnych a nepárnych čísel a podobne. Ďalej budeme pokračovať z bodu v binomickom strome, do ktorého sme sa už dostali a znova pomocou náhodného generátora rozhodneme či úroková miera vzrastie alebo poklesne. Takto pokračujeme až do konca sledovaného obdobia a tým dostaneme jeden kompletný scenár vývoja úrokovej miery. Tento postup opakujeme toľko krát, koľko scenárov vývoja úrokovej miery potrebujeme. Tomuto spôsobu generovania úrokovej miery možno vytknúť, že ponúka len malé množstvo prípadov. Neznamená to však, že tento spôsob je nepoužiteľný. Možným vylepšením je zvýšiť počet možností (vetiev) pri rozhodovaní použitím trinomického alebo multinomického stromu.

4. Optimalizácia dlhopisového portfólia s využitím homogénnej samoduálnej metódy

4.1 S-MAX model a úloha lineárneho programovania

V predchádzajúcej kapitole sme naformulovali problém optimalizácie dlhopisového portfólia pomocou niekoľkých modelov. Pozrime sa teraz bližšie na stochastický prípad

MAX modelu: S-MAX model. Uvažujme, že v modeli máme zahrnutých n dlhopisov, použili sme k scenárov vývoja úrokových mier a časový horizont modelu je m. Potom môžeme S-MAX model prepísať do tvaru

$$\max_x \{\ c^T x \mid \quad Ax = b\ , \quad x \geqslant 0\ \},$$

kde $x,\ c \in R^{n+2+2mk+k},\ b \in R^{1+mk+2k}$ sú vektory:

$$\begin{aligned}
x &= [\, x_1 \quad \ldots \quad x_n \quad r_0 \quad b_0 \quad r_1^1 \quad \ldots \quad r_m^k \quad b_1^1 \quad \ldots \quad b_m^k \quad h_1 \quad \ldots \quad h_k\,]^T \\
c &= [\, 0 \quad \ldots \quad 0 \quad p_1 \quad \ldots \quad p_k\,]^T, \\
b &= [\, -B \quad L_1^1 \quad \ldots \quad L_m^1 \quad \ldots \quad L_1^k \quad \ldots \quad L_m^k \quad 0 \quad \ldots \quad 0\,]^T
\end{aligned}$$

a $A \in R^{(1+mk+2k)\times(n+2+2mk+k)}$ je bloková matica

$$A = \begin{bmatrix}
F^0 & C^0 & & & & & & & & & \\
F^1 & C^1 & D^1 & & & -D^1 & & & & & \\
F^2 & C^2 & & D^2 & & & -D^2 & & & & \\
\vdots & \vdots & & & \ddots & & & \ddots & & & \\
F^k & C^k & & & D^k & & & & -D^k & & \\
 & & E & & & & & & & -1 & & \\
 & & & E & & & & & & & -1 & \\
 & & & & \ddots & & & & & & & \ddots \\
 & & & & E & & & & & & & & -1 \\
 & & & & & E & & & & & & & \\
 & & & & & & E & & & & & & \\
 & & & & & & & \ddots & & & & & \\
 & & & & & & & & E & & & &
\end{bmatrix},$$

kde $F^0 \in R^{1\times n},\ C^0 \in R^{1\times 2},\ E \in R^{1\times m},\ F^i \in R^{m\times n},\ C^i \in R^{m\times 2},\ D^i \in R^{m\times m}$ sú matice:

$$\begin{aligned}
F^0 &= [\, F_{1,0} \quad F_{2,0} \quad \ldots \quad F_{n,0}\,], \\
C^0 &= [\, -1 \quad 1\,], \\
E &= [\, 0 \quad \ldots \quad 0 \quad 1\,], \\
F^i &= \begin{bmatrix}
F_{1,1}^i & F_{2,1}^i & \ldots & F_{n,1}^i \\
F_{1,2}^i & F_{2,2}^i & \ldots & F_{n,2}^i \\
\vdots & \vdots & \ddots & \vdots \\
F_{1,m}^i & F_{2,m}^i & \ldots & F_{n,m}^i
\end{bmatrix}, \qquad i = 1, ..., k,
\end{aligned}$$

$$C^i = \begin{bmatrix} (1+\rho_1^i) & -(1+\rho_1^i) \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \end{bmatrix}, \qquad i = i, ..., k,$$

$$D^i = \begin{bmatrix} -1 & & & & & \\ (1+\rho_2^i) & -1 & & & & \\ & (1+\rho_3^i) & -1 & & & \\ & & \ddots & \ddots & & \\ & & & & (1+\rho_m^i) & -1 \end{bmatrix}, \qquad i = 1, ..., k.$$

Teda S-MAX model je úlohou lineárneho programovania v rovnicovom tvare. V kapitole 2 sme popísali homogénnu samoduálnu metódu na riešenie štandartnej úlohy lineárneho programovania. Ako však vieme z kapitoly 1 každú úlohu v rovnicovom tvare vieme prepísať do štandartného tvaru nahradením každej rovnosti dvojicou nerovnosti. Teda na S-MAX model môžeme aplikovať algoritmus homogénnej samoduálnej metódy.

Je zrejmé, že rozmer S-MAX úlohy rastie jednak s počtom uvažovaných dlhopisov n, jednak s časovým horizontom m a jednak s počtom použitých scenárov vývoja úrokových mier k. Konkrétne S-MAX model zapísaný ako štandartná úloha lineárneho programovania obsahuje $n + 2 + (2m + 1)k$ premenných a $2(1 + (m + 2)k)$ ohraničení.

Pre dosiahnutie dostatočnej obsiahlosti použitých scenárov vývoja úrokových mier ich počet výrazne zväčšuje rozmer riešenej úlohy. To spôsobuje, že časová náročnosť riešenia takejto úlohy použitím simplexovej metódy môže byť relatívne veľká. Cieľom tejto časti práce bude aplikovať algoritmus homogénnej samoduálnej metódy na S-MAX model a porovnať dosiahnuté výsledky so simplexovou metódou. K dosiahnutiu tohto cieľa použijeme voľne dostupný softvér z internetu [6], v ktorom je naprogramovaná homogénna samoduálna metóda ako aj simpexová metóda. Program pracuje pod operačným systémom MS DOS. Ako vstup do softvéru slúži úloha zapísana v MPS formáte. Jeho nevýhodou je, že neposkytuje možnosť zadávania ďalších paremetrov ako je napríklad meranie času.

4.2 MPS formát

MPS formát je stĺpcovo orientovaný formát zápisu úlohy lineárneho programovania, ktorý slúži ako alternatíva k zápisu úlohy pomocou rovníc a nerovníc (riadkovo orientovaný formát). Pri MPS formáte všetky premenné a riadky dostávajú svoje mená. Keďže MPS je pomerne starý formát, ktorý sa používal ešte v časoch diernych štítkov, všetko má presne danú pozíciu. Vertikálne je delený do šiestich polí, ktoré začínajú v stĺpcoch 1, 5, 15, 25, 40 a 50. Horizontálne je členený do sekcií. Každý MPS súbor musí obsahovať v danom poradí tieto sekcie: NAME (meno), ROWS (riadky), COLUMNS (stĺpce), RHS (pravé strany) a ENDATA (koniec dát). Niektoré MPS súbory môžu obsahovať navyše ešte sekcie RANGES (rozsah) a BOUNDS (ohraničenia), ktoré sú umiestnené na konci pred sekciou ENDDATA.

Teraz bližšie uvedieme obsah a štruktúru jednotlivých sekcií:

1. NAME. Táto sekcia pozostáva len zo slova NAME umiestneného v stĺpcoch 1-4 a pomenovania problému v stĺpcoch 15-22. Pomenovanie problému môže byť ľubovoľné maximálne však 8 znakov. Niektoré programy používajúce ako vstup súbor vo formáte MPS používajú pomenovanie problému pre názov výstupného súboru.

2. ROWS. V tejto časti sa pomenujú riadky[1] a bližšie sa špecifikuje typ riadku. Sekcia začína slovom ROW umiestneným v stĺpcoch 1-4 nasledovaným popisom riadkov. Typ riadku je definovaný v stĺpcoch 2 alebo 3, názov riadku v stĺpcoch 5-12. Kódy pre špecifikáciu typu riadku sú nasledovné:

E - rovnosť ($=$)

L - menšie alebo rovné ($\leqslant$)

G - väčšie alebo rovné ($\geqslant$)

N - účelová funkcia

N - riadok bez ohraničení

Poznamenajme, že ak je viac riadkov špecifikovaných ako typ N za účelovú funkciu sa považuje prvý z nich.

[1] riadkom rozumieme účelovú funkciu alebo ohraničenie okrem ohraničení na premenné

3. COLUMNS. V tejto časti sú definované mená premenných, koeficienty účelovej funkcie a všetky nenulové členy matice A. Sekcia začína slovom COLUMNS umiestneným v stĺpcoch 1-7. V poliach 2-6 sú potom špecifikované premenné a ich koeficienty v jednotlivých riadkov. V poli 2 (t.j. v stĺpcoch 5-12) je uvedený názov premennej. V poliach 3 a 5 (t.j. v stĺpcoch 15-22 a 40-47) je uvedený názov riadku, v ktorom sa vyskytuje pri danej premennej nenulový koeficient. V poliach 4 a 6 (t.j. v stĺpcoch 25-36 a 50-61) je koeficient premennej v prislúchajúcom riadku. Na záver poznamenajme, že dáta musia byť zapisované súvislé pre jednotlivé premenné, pričom polia 5 a 6 sa nemusia využívať.

4. RHS. Táto časť zahŕňa jeden alebo viac vektorov pravých strán. Väčšinou nie je potrebné používať viac ako jeden vektor pravej starny. Sekcia začína slovom RHS umiestneným v stĺpcoch 1-3. Štruktúra zápisu je podobná ako v sekcii COLUMNS, t.j v poli 2 je uvedený názov pravej strany, v poliach 3 a 5 názov riadku, v ktorom sa nachádza nenulový prvok a v poliach 4 a 6 veľkosť pravej strany v príslušnom riadku. V riadkoch, ktoré nie sú uvedené v tejto sekcii sa považujú pravé strany za nulové.

5. RANGES. Táto časť nie je povinná, t.j. v MPS súboroch sa nemusí nachádzať. Sekcia RANGES bližšie špecifikuje ohraničenia typu $h_i \leqslant a_{i,1}+...+a_{i,n} \leqslant u_i$, t.j. ohraničenia, v ktorých sa súčasne vyskytuje dolné aj horné ohraničenie. Rozsah takého ohraničenia je potom $r_i = u_i - h_i$. Hodnota r_i s ľubovoľným znamienkom je zapísaná v sekcii RANGES. Hodnoty hraníc u_i a h_i sú určené v sekcii RHS spoločnou hodnotou b_i, ktorá je rovná buď dolnej alebo hornej hranici v závislosti od znamienka rozsahu r_i v sekcii RANGES a typu riadku uvedeného v sekcii ROWS. Hodnoty b_i sú uvedené v nasledujúcej tabuľke:

Typ riadku	Znamienko r_i	Hodnota b_i
$G(\geqslant)$	$+/-$	h_i
$L(\leqslant)$	$+/-$	u_i
$E(=)$	$+$	h_i
$E(=)$	$-$	u_i

Rovnako ako predchádzajúce aj táto sekcia začína slovom RANGES v stĺpcoch 1-6. V poli 2 je očakávané meno rozsahu, v poli 3 meno riadku a v poli 4 hodnota rozsahu r_i.

6. BOUNDS. Tak ako časť RANGES, tak aj táto časť nie je povinná. Sekcia BOUNDS bližšie špecifikuje ohraničenia na premenné. Ak premenná nie je v tejto sekcii uvedená považuje sa za nezápornú (t.j. typ PL). Sekcia začína slovom BOUNDS v poli 1. V poli 2 je zapísaný typ ohraničenia:

LO - dolné ohraničenie:	$b_j \leqslant x_j (< \infty)$
UP - horné ohraničenie:	$(0 \leqslant) x_j \leqslant b_j$
FX - fixovaná premenná:	$x_j = b_j$
FR - voľná premenná:	$-\infty < x_j < +\infty$
MI - dolné ohraničenie $-\infty$:	$-\infty < x_j (\leqslant 0)$
PL - horné ohraničenie ∞:	$(0 \leqslant) x_j < \infty$.

V poli 3 sa očakáva meno premennej a v poli 4 sa určuje hodnota hranice b_j.

7. ENDATA. Poslednou sekciou je sekcia ENDATA, ktorá pozostáva len zo slova ENDATA napísaného v poli 1 v stĺpcoch 1-6 a signalizuje koniec vstupného súboru.

Pre väčšiu názornosť uvádzame príklad úlohy zapísanej v MPS formáte [9]:

```
NAME          TESTPROB
ROWS
 N  COST
 L  LIM1
 G  LIM2
 E  MYEQN
COLUMNS
    XONE      COST                 1   LIM1                 1
    XONE      LIM2                 1
    YTWO      COST                 4   LIM1                 1
    YTWO      MYEQN               -1
    ZTHREE    COST                 9   LIM2                 1
    ZTHREE    MYEQN                1
RHS
    RHS1      LIM1                 5   LIM2                10
    RHS1      MYEQN                7
BOUNDS
 UP BND1      XONE                 4
 LO BND1      YTWO                -1
 UP BND1      YTWO                 1
ENDATA
```

Pre porovnanie uvádzame tú istú úlohu zapísanú v riadkovom formáte.

```
Optimize
 COST:    XONE + 4 YTWO + 9 ZTHREE
Subject To
 LIM1:    XONE + YTWO <= 5
 LIM2:    XONE + ZTHREE >= 10
 MYEQN:   - YTWO + ZTHREE  = 7
Bounds
 0 <= XONE <= 4
-1 <= YTWO <= 1
End
```

Nakoniec ešte poznamenajme, že v MPS formáte nikde nie určený smer optimalizácie, t.j. či sa jedná o úlohu minimalizačnú alebo maximalizačnú. Smer optimalizácie sa potom určuje pomocou softvéru, kde slúži súbor vo formáte MPS ako vstupný súbor a to buď pomocou parametra, alebo daný softvér požaduje ako vstup striktne maximalizačnú resp. minimalizačnú úlohu. Ako sme uviedli v kapitole 1 každú maximalizačnú úlohu vieme previesť na minimalizačnú a naopak a teda v prípade, ak softvér požaduje maximalizačnú úlohu a naša úloha je minimalizačná resp. naopak, stačí prenásobiť koeficienty prislúchajúce účelovej funkcii číslom -1. Nami používaný softvér na optimalizáciu [6] si vyžadoval maximalizačnú úlohu.

MPS formát je pomerne starý formát a pri práci s veľkými úlohami sa s ním pracuje pomerne ťažko, keďže všetky prvky musia byť zapísané na konkrétnom mieste. Preto sme v tejto práci na vytváranie súborov v MPS formáte používali softvér z internetu [7], ktorý transformoval úlohu v riadkovom tvare (zapísanú pomocou rovníc a nerovníc) do formátu MPS. Aj tento softvér pracuje pod operačným systémom MS DOS. Program požaduje dva parametre, názov vstupného a názov výstupného súboru. Od vstupného súboru s úlohou zapísanou v riadkovom formáte sa požaduje, aby každý riadok (ohraničenie, účelová funkcia) bol ukončený bodkočiarkou. Účelová funkcia môže byť umiestnená buď na začiatku alebo na konci súboru a požadovaná syntax je: max: (resp. min:) účelová funkcia;.

4.3 Výsledky experimentov

Deterministický MAX model:

Ako prvým sa budeme zaoberať deterministickým prípadom MAX modelu. Do množiny dlhopisov zahrnieme 1-ročné, 2-ročné, 3-ročné, 4-ročné, 5-ročné, 7-ročné a 10-ročné dlhopisy vypísané v čase 0. Teda dokopy 7 dlhopisov. Budeme uvažovať, že dlhopisy sú typu par-bond, t.j. ich nákupná cena je rovnaká ako ich nominálna hodnota, vyplácajú kupón raz ročne a cena každého dlhopisu je 1 peňažná jednotka. Ďalej budeme predpokladať, že máme k dispozícii rozpočet B=50000 peňažných jednotiek a výška očakávaných záväzkov je v každom roku 6000 peňažných jednotiek. Časový horizont úlohy m je 10 rokov a veľkosť jednej periódy jeden rok.

Počty dlhopisov, ktoré máme kúpiť v čase 0, a ktoré sme získali z MAX modelu použitím simplexovej metódy a použitím homogénnej samoduálnej metódy sú rovnaké a sú uvedené v tabuľke 1 v riadku MAX. Porovnanie obidvoch metód z hľadiska počtu iterácií je uvedené v tabuľke 2 v riadku MAX.

Rozmer úlohy je pomerne malý a teda aj časová náročnosť riešenia úlohy optimalizácie dlhopisového portfólia formulovanej bez zahrnutia náhodnosti budúceho vývoja úrokových mier nie je veľká. Na druhej strane, keďže úrokové miery sa v čase menia, získaný výsledok nemusí zodpovedať realite. Preto sa ďalej budeme zaoberať už len S-MAX modelom, ktorý zohľadňuje stochastický vývoj úrokových mier použitím scenárov.

S-MAX model:

S-MAX modelom budeme riešiť rovnakú úlohu ako v prípade deterministického MAX modelu. Navyše budeme predpokladať, že veľkosť záväzkov splatných v každom časovom okamihu je deterministická a teda sa nemení a je rovná 6000 peňažných jednotiek pre všetky $t = 1, ..., m$. Pre generáciu scenárov vývoja úrokových mier použijeme postupne NY-7, binomický, trinomický a multinomický strom.

V nami použitom binomickom strome sme zvolili v každom časovom okamihu s pravdepodobnosťou 50% nárast úroku o 0,5% a s pravdepodobnosťou 50% pokles úroku o 0,5%. V trinomickom strome sme zvolili opäť s rovnakými pravdepodobnosťami nárast úroku o 0,5%, pokles o 0,5% a možnosť, že sa úrok nezmení. Ďalej sme použili dva multinomické stromy, s 11 a 21 vetvami v každom uzle stromu. Zmena úroku prislúchajúca jednotlivým vetvám stromu bola v rozpätí od nárastu o 1,5% až po pokles o 1,5%. Ak by bol úrok na začiatku stromu v čase 0 dostatočne malý mohlo by sa stať, že v niektorých vetvách stromu by predpokladaný úrok bol nulový alebo záporný, čo v nie je v skutočnosti možné. Tento problém sme vyriešili zavedením minimálneho úroku, ktorý sme stanovili na hodnotu 0,5%. To znamená, že tam kde je úrok v strome menší ako 0,5% budeme uvažovať úrok s výškou 0,5%.

Na generáciu náhodného pohybu sme použili náhodný generátor v programovacom jazyku C++, pomocou ktorého sme vygenerovali 30 scenárov pre každý strom. Výsledky experimentov použitím metód sú uvedené v tabuľke 1 v riadkoch prislúchajúcich k S-MAX . Porovnanie metód z hľadiska počtu iterácií je uvedené v tabuľke 2 v riadkoch prislúchajúcich k S-MAX.

		1-ročný	2-ročný	3-ročný	4-ročný	5-ročný	7-ročný	10-ročný
MAX		2511	2689	2878	3076	3288	11336	24222
S-MAX	NY7	2505	4161	1402	3071	866	7320	30675
	binomický	2758	3177	2082	3070	3554	7909	27450
	trinomický	2761	3180	2428	3306	5329	10018	22978
	multinomický (11)	2808	3716	1833	3602	2701	8460	26880
	multinomický (21)	2507	3602	1099	3577	2160	7825	29230

Tab.1.

<table>
<tr><th colspan="4" rowspan="2"></th><th colspan="3">Počet</th></tr>
<tr><th>Premenných</th><th>Ohraničení</th><th>Iterácií</th></tr>
<tr><td colspan="2" rowspan="3">MAX</td><td rowspan="2">HSD</td><td>krátky krok</td><td rowspan="3">30</td><td rowspan="3">26</td><td>26</td></tr>
<tr><td>dlhý krok</td><td>28</td></tr>
<tr><td colspan="2">Simplex</td><td>33</td></tr>
<tr><td rowspan="15">S-MAX</td><td rowspan="3">NY7</td><td rowspan="2">HSD</td><td>krátky krok</td><td rowspan="3">156</td><td rowspan="3">170</td><td>32</td></tr>
<tr><td>dlhý krok</td><td>30</td></tr>
<tr><td colspan="2">Simplex</td><td>230</td></tr>
<tr><td rowspan="3">binomický</td><td rowspan="2">HSD</td><td>krátky krok</td><td rowspan="3">639</td><td rowspan="3">722</td><td>34</td></tr>
<tr><td>dlhý krok</td><td>34</td></tr>
<tr><td colspan="2">Simplex</td><td>1001</td></tr>
<tr><td rowspan="3">trinomický</td><td rowspan="2">HSD</td><td>krátky krok</td><td rowspan="3">639</td><td rowspan="3">722</td><td>34</td></tr>
<tr><td>dlhý krok</td><td>34</td></tr>
<tr><td colspan="2">Simplex</td><td>947</td></tr>
<tr><td rowspan="3">multinomický (11)</td><td rowspan="2">HSD</td><td>krátky krok</td><td rowspan="3">639</td><td rowspan="3">722</td><td>34</td></tr>
<tr><td>dlhý krok</td><td>32</td></tr>
<tr><td colspan="2">Simplex</td><td>972</td></tr>
<tr><td rowspan="3">multinomický (21)</td><td rowspan="2">HSD</td><td>krátky krok</td><td rowspan="3">639</td><td rowspan="3">722</td><td>38</td></tr>
<tr><td>dlhý krok</td><td>36</td></tr>
<tr><td colspan="2">Simplex</td><td>950</td></tr>
</table>

Tab.2. Porovnanie homogénnej samoduálnej (HSD) a simplexovej metódy

Jednou z vlastností metód vnútorného bodu, medzi ktoré zaraďujeme aj homogénnu samoduálnu metódu, je, že v prípade existencie alternatívnych optimálnych riešení poskytujú iné optimálne riešenie ako simplexová metóda. Riešenia, ktoré sme získali použitím homogénnej samoduálnej aj simplexovej metódy boli totožné a teda nami riešený problém má jediné riešenie.

Z výsledkov experimentov vidieť, že použitý scenár ovplyvňuje ako počet jednotlivých dlhopisov, ktoré máme zahrnúť do portfólia, tak aj počet iterácií. Teda pre správny výber portfólia je dôležité vybrať takú množinu scenárov, ktorá sa najviac približuje realite.

Porovnanie metód z hľadiska počtu iterácií

Teraz sa bližšie pozrieme na metódy z hľadiska počtu iterácií. Budeme riešiť vyššie naformulovanú úlohu použitím binomického stromu, pričom budeme jednotlivo pozorovať ako sa mení počet iterácií s narastajúcim rozmerom úlohy pre homogénnu samoduálnu metódu a ako sa mení pre simplexovú metódu. Rozmer úlohy budeme zväčšovať zvyšovaním počtu scenárov a to tak, že k už vygenerovaným scenárom budeme dopĺňať nové. To isté potom urobíme aj pre multinomický strom s 21 vetvami. Výsledky experimentov sú zhrnuté v tabuľkách 3 a 4.

Počet scenárov	Rozmer úlohy	Počet iterácií		
		HSD		Simplex
		krátky krok	dlhý krok	
1	26x30	26	30	33
5	122x114	34	32	179
7	170x156	34	34	258
10	242x219	30	34	335
15	362x324	38	34	495
20	482x429	30	32	648
25	602x534	32	32	803
30	722x639	34	32	957
40	962x849	34	34	1310
50	1202x1059	36	34	1602
75	1802x1584	40	34	2392
100	2402x2109	40	34	3077
125	3002x2634	44	36	NA
150	3602x3159	52	38	NA
200	4802x4209	60	274	NA
250	6002x5259	104	154	NA
300	7202x6309	104	120	NA

Tab.3. Porovnanie homogénnej samoduálnej (HSD) a simplexovej metódy z hľadiska počtu iterácií (binomický strom)

NA - použitý softvér nedokázal spočítať

Počet scenárov	Rozmer úlohy	Počet iterácii		
		HSD		Simplex
		krátky krok	dlhý krok	
1	26x30	28	30	32
5	122x114	30	30	154
7	170x156	30	30	212
10	242x219	32	32	301
15	362x324	32	32	459
20	482x429	36	34	619
25	602x534	36	34	767
30	722x639	36	34	917
40	962x849	40	36	1208
50	1202x1059	40	36	1542
75	1802x1584	38	36	2241
100	2402x2109	42	36	3039
125	3002x2634	52	42	NA
150	3602x3159	122	54	NA
200	4802x4209	194	206	NA
250	6002x5259	182	560	NA
300	7202x6309	154	274	NA
400	9602x8409	126	194	NA
500	12002x10509	154	196	NA
600	14402x12609	198	250	NA
700	16802x14709	196	420	NA
800	19202x16809	152	370	NA
900	21602x18909	132	118	NA
1000	24002x21009	138	128	NA

Tab.4. Porovnanie homogénnej samoduálnej (HSD) a simplexovej metódy z hľadiska počtu iterácii (multinomický strom)
NA - použitý softvér nedokázal spočítať

Môžeme pozorovať, že s narastajúcim rozmerom úlohy počet iterácií pri použití simplexovej metódy rastie rýchlejšie ako počet iterácií pri použití homogénnej samoduálnej metódy. Pri použití 125 scenárov nami používaný softvér so simplexovou metódou už takúto úlohu nedokázal zrátať. Ďalej môžeme pozorovať pri homogénnej samoduálnej metóde, že väčší rozmer úlohy neznamená nevyhnutne aj väčší počet iterácií. Tento jav si môžeme vysvetliť tak, že pridávaním náhodných scenárov sa mení kvalita riešenej úlohy. Môžeme tiež pozorovať, že metóda z dlhým krokom je citlivejšia na zmenu scenárov. Vo všeobecnosti je však zrejmý trend zvyšovania počtu iterácií s narastajúcim rozmerom úlohy, i keď čím je rozmer úlohy väčší, tým je trend rastu pomalší.

Žiaľ nami používaný softvér neumožňoval merať časovú náročnosť. Približný čas činnosti celého programu však bol pre malé úlohy približne rovnaký pre homogénnu samoduálnu metódu aj pre simplexovú metódu, rádovo 1-5 sekúnd. Pre väčšie úlohy (rádovo od rozmeru 500x500) sme pozorovali zvyšovanie časovej náročnosti pri simplexovej metóde, čo mohlo byť spôsobené nielen očakávanou vlastnosťou simplexovej metódy (exponenciálna zložitosť), ale aj väčším počtom výpisov a prácou operačného systému[2] s uvoľňovaním miesta v operačnej pamäti. Pre úlohy väčšie, ako úlohy prislúchajúce k 125 použitým scenárom vývoja úrokových mier (t.j. väčšie ako 2752x2509), operačný systém už nedokázal alokovať dostatočné množstvo pamäte[3], a preto nami používaný softvér zlyhal. Časová náročnosť pri softvéri používajúcom homogénnu samoduálnu metódu bola pomerne nízka a pre najväčšiu úlohu (22002x20009) bola približne 5 minút[3].

[2]MS Windows XP Professional

[3]všetky experimenty boli prevádzané na počítači s procesorom AMD® Duron 700Mhz a operačnou pamäťou SDRAM 128MB

Záver

V tejto práci sme popísali jednu z metód vnútorného bodu (t.j. homogénnu samoduálnu metódu) našli sme na internete príslušný softvér, ktorý sme použili na riešenie problému optimalizácie dlhopisového portfólia. Ako sme uviedli v práci tento problém môže viesť k lineárnym úloham veľkých rozmerov. Podarilo sa nám zostaviť a spočítať úlohy relatívne veľkých rozmerov (najväčšia úloha mala rozmer 24002x21009). Keďže sme používali nekomerčný softvér voľne prístupný na internete, ten nám neumožňoval zrátať úlohy väčších rozmerov (obmedzenie na počet iterácii bolo 199 pri homogénnej samoduálnej metóde s krátkym krokom a 599 pri tej istej metóde s dlhým krokom). Taktiež sa nám nepodarilo presne určiť ani časovú náročnosť metódy.

Ďalším cieľom bolo porovnať na úlohách optimalizácie dlhopisového portfólia homogénnu samoduálnu metódu so simplexovou metódou. Aj napriek tomu, že sme používali nekomerčný softvér a nepodarilo sa nám spočítať simplexovou metódou až tak veľké úlohy ako homogénnou samoduálnou metódou, podarilo sa nám ukázať správnosť nášho predpokladu, že homogénna samoduálna metóda by mala byť pre úlohy veľkých rozmerov efektívnejšia.

Na záver ešte poznamenajme, že nami namodelovaný problém otimalizácie by sa dal ešte viac priblížiť realite. Lineárna účelová funkcia často nepostihuje dobre realitu. Preto ju je možné v modeli nahradiť kvadratickou účelovou funkciou alebo všeobecnou nelineárnou úžitkovou funkciou. Ďalšími vylepšeniami modelu by mohlo byť zahrnutie transakčných nákladov (popísané v časti 3.1) alebo zahrnutie možnosti prerovnávania portfólia, ktoré sa dá namodelovať zavedením ďalších premenných do modelu (viď Nielsen[1], str.15-22).

Literatúra:

[1] Nielsen, S.S., *Mathematical modeling and optimization with applications in finance*, 1997, http://www.math.ku.dk/~ nielsen.

[2] Vanderbei, J.R., *Linear programming: Foundations and extensions (Second Edition)*, 2000, http://www.princeton.edu/~ rvdb/LPbook/online.html.

[3] Vanderbei, J.R., *Linear programming: Foundations and extensions*, Kluwer Academic Publishers, Boston, 1997.

[4] Plesnik, J., Dupačová, J., Vlach, M., *Lineárne programovanie*, Alfa, Bratislava, 1990.

[5] Luknár, I., *Optimalizácia portfólia dlhopisov pri stochastickom vývoji úrokových mier*, Diplomová práca, Bratislava, 2001.

[6] Vanderbei, J.R., *Softvér na optimalizáciu používajúci homogénnu samoduálnu metódu a simplexovú metódu,* http://www.sor.princeton.edu/~ rvdb/LPbook/src/index.html.

[7] Dirks, J.J., *Softvér na transformáciu úlohy vo formáte LP do formátu MPS,* http://elib.zib.de/pub/Packages/mathprog/linprog/lp-solve/lp2mps.c.

[8] *About MPS Input Format*, MCS Division at Argonne National Laboratory, www.mcs.anl.gov/otc/Guide/OptWeb/continuous/ constrained/linearprog/mps.html.

[9] *MPS format*, Industrial Engeneering and Management Science at Northwestern University, Evanston, http://users.iems.nwu.edu/~ neos/mps_format.txt.