Fakulta matematiky, fyziky a informatiky
Univerzity Komenského v Bratislave

Katedra aplikovanej matematiky a štatistiky

METÓDY VNÚTORNÉHO BODU VO FINANČNÝCH MODELOCH

DIPLOMOVÁ PRÁCA

2006 Václav Kolátor

Fakulta matematiky, fyziky a informatiky
Univerzity Komenského v Bratislave

Matematika

Ekonomická a finančná matematika

METÓDY VNÚTORNÉHO BODU VO FINANČNÝCH MODELOCH

DIPLOMOVÁ PRÁCA

Diplomant: Václav Kolátor
Vedúca diplomovej práce: doc. RNDr. Margaréta Halická, CSc.

Bratislava 2006

Týmto prehlasujem, že som diplomovú prácu vypracoval samostatne s použitím uvedenej literatúry.

V Bratislave, 27.4.2006

Václav Kolátor

Ďakujem vedúcej mojej diplomovej práce doc. RNDr. Margaréte Halickej, CSc. za cenné rady, pripomienky a odborné vedenie práce.

Zadanie

Cieľom diplomovej práce je naštudovať metódy vnútorného bodu a využiť ich spolu so simplexovou metódou pri riešení vhodných úloh z oblasti finančnej matematiky.

Abstrakt

Väčšina firiem potrebuje zhodnocovať svoj majetok, či už z dôvodu maximalizácie zisku alebo potreby vykonania akýchsi budúcich platieb. Typickým príkladom, že veľkosť, ako aj realizácia týchto budúcich záväzkov, nemusí byť dopredu známa, sú poisťovne, a to konkrétne so životným poistením, kde výška a čas platby závisí od dĺžky života daného zákazníka. Spôsob, ako sa voči takýmto platbám zabezpečiť, je investovanie do aktív, konkrétne sa často používajú práve dlhopisy. Našou úlohou je zostaviť dlhopisové portfólio, ktorého cash-flow bude kopírovať výšku našich záväzkov v budúcnosti. Predstavíme viacero modelov, pričom postupne prejdeme od tých najjednoduchších k zložitejším a lepšie popisujúcim reálnu situáciu. Zároveň ukážeme, že všetky modely sa dajú previesť na úlohu lineárneho programovania. Naformulované úlohy budeme následne riešiť pomocou metód vnútorného bodu a simplexového algoritmu, pričom budeme porovnávať ich výkonnosť.

Obsah

Predhovor

Lineárne programovanie má v súčasnosti široké využitie a nevyhýba sa ani oblasti ekonomických a finančných úloh, napríklad problémom optimálneho rozhodovania. Do tejto kategórie patrí aj úloha optimalizácie dlhopisového portfólia. Na hľadanie riešení nám teória lineárneho programovania poskytuje veľké množstvo použiteľných metód a algoritmov. Jedna z najstarších a najpoužívanejších, tzv. simplexová metóda, ktorú v roku 1947 vynašiel Dantzig, si rýchlo získala obľubu. Neskôr sa však podarilo ukázať, že má exponenciálnu zložitosť, a preto nie je vhodná na riešenie úloh veľmi veľkých rozmerov. Nastal preto dopyt po alternatívnych možnostiach. Prelomovou sa ukázala byť práca Karmarkára, ktorý prezentoval v roku 1984 algoritmus dosahujúci polynomiálnu zložitosť. Publikácia naštartovala revolúciu v optimalizácii a priviedla odborníkov na lineárne programovanie do oblasti konvexného programovania, známej pod názvom metódy vnútorného bodu. Tie zahŕňajú obrovské množstvo algoritmov, niektoré si získali aj softvérovú podobu a vďaka potvrdenej polynomiálnej zložitosti predstavujú významnú alternatívu k simplexovej metóde.

Cieľom tejto práce bolo získať voľne dostupný softvér z internetu, ktorý by používal niektoré z týchto metód, naštudovať ich a na ekonomických úlohách (v našom prípade optimalizácie dlhopisového portfólia) porovnať ich efektívnosť aj so simplexovou metódou. Stanovený cieľ sa nám podarilo naplniť. Získali sme program používajúci simplexový algoritmus a ďalšie dva programy využívajúce metódy vnútorného bodu. V prvom bola zakomponovaná homogénna samoduálna metóda, v druhom bol implementovaný v súčasnosti najpoužívanejší Mehrotrov prediktor-korektor algoritmus. Tie sme potom aplikovali na spomínané ekonomické úlohy optimálneho rozhodovania, ktoré sme generovali v rozsahu 56009 premenných a 26001 ohraničení.

Prvá kapitola obsahuje vysvetlené základné pojmy z oblasti lineárneho programovania a teórie duality. V druhej kapitole sme podľa knihy Stephena J. Wrighta[12] popísali Mehrotrov prediktor-korektor algoritmus. Tretia kapitola sa zaoberá optimalizáciiou dlhopisového portfólia. Postupne sme naformulovali viacero modelov popisujúcich tento problém. Zdrojom informácií bol článok Sorena S. Nielsena[1]. V záverečnej štvrtej kapitole sme uviedli výsledky numerických experimentov.

Kapitola 1

Teória lineárneho programovania

1.1 Úloha lineárneho programovania

V tejto diplomovej práci sa budeme zaoberať problémom z oblasti financií, optimalizáciou dlhopisového portfólia, ktorý možno naformulovať ako úlohu lineárneho programovania. Bolo by preto dobré si najskôr povedať, čo je úloha lineárneho programovania a ako vyzerá. Lineárne programovanie skúma hľadanie viazaného extrému funkcie viacerých premenných. Zvláštnosťou je fakt, že hľadáme extrém lineárnej účelovej funkcie, pričom sme viazaní podmienkami, ktoré majú podobu lineárnych rovníc alebo nerovníc. Zavedieme si nasledovné označenie: $A = (a_{ij})$ je matica z R^{mxn}, $b = (b_i)$ je vektor z R^m a $c = (c_j)$ je vektor z R^n, pričom vektory chápeme ako stĺpce. Našou úlohou bude hľadať minimum účelovej funkcie v tvare:

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

pri ohraničeniach v tvare:

$$\sum_{j=1}^{n} a_{ij} x_j = b_i, \quad i = 1, ..., m,$$
$$x_j \geq 0, \quad j = 1, ..., n.$$

Takto naformulovanú úlohu budeme nazývať úlohou lineárneho programovania v štandardnom tvare, skrátene štandardnou úlohou. Pre jednoduchosť je možné ju zapísať nasledujúcim spôsobom:

$$min\left\{c^T x | Ax = b, x \geq 0\right\}.$$

Často sa táto úloha označuje aj ako úloha lineárneho programovania v rovnicovom tvare. Môže sa stať, že hľadáme minimum lineárnej funkcie ako vyššie, avšak ohraničenia sú tentokrát v tvare nerovností. Potom takúto úlohu nazývame úlohou lineárneho programovania v tvare nerovností. Veľmi ľahko ju vieme previesť na úlohu v štandardnom tvare, a to pridaním doplnkových premenných $s_i \geq 0$ a úpravou nerovníc $\sum_{j=1}^{n} a_{ij}x_j \leq b_j$ na rovnice nasledovne: $\sum_{j=1}^{n} a_{ij}x_j + s_i = b_j$. Podobne každú úlohu v rovnicovom tvare vieme previesť na úlohu v tvare nerovností nahradením každej rovnosti $\sum_{j=1}^{n} a_{ij}x_j = b_i$ dvojicou nerovností $\sum_{j=1}^{n} a_{ij}x_j \leq b_i$, $\sum_{j=1}^{n} a_{ij}x_j \geq b_i$.

V praxi sa môžeme stretnúť s problémom, kedy namiesto minima budeme hľadať maximum danej účelovej funkcie. Vieme však, že pre ľubovoľnú funkciu $f : A \rightarrow R$, kde A je ľubovoľná množina, $A \subset R^n$, platí:

$$min\ (f(x)) = -max\ (-f(x)),\ x \in A.$$

Týmto spôsobom teda možno každú maximalizačnú úlohu previesť na úlohu minimalizačnú. Ďalšou možnosťou je, že okrem nezáporných premenných môže úloha obsahovať aj voľné premenné. V štandardnej úlohe však vyžadujeme nezápornosť všetkých premenných. Tú dosiahneme rozdelením voľných premenných (napr. a) na dve nové nezáporné premenné, ktoré označíme ako a^+ a a^-. Požadujeme však aby platilo $a = a^+ - a^-$. Na základe týchto vyššie uvedených skutočností sa ďalej budem zaoberať iba úlohami v štandardnom tvare.

1.2 Teória duality

Teória duality je jednou z najdôležitejších súčastí teórie lineárneho programovania. Jej výsledkom je fakt, že ku každej minimalizačnej úlohe (budeme ju nazývať primárnou úlohou) vieme jednoznačne priradiť maximalizačnú úlohu, pričom tú budeme nazývať duálnou úlohou. Označenie primárna a duálna vychádza z toho, z ktorej úlohy vychádzame. Ak z minimalizačnej úlohy, tak duálna úloha je maximalizačná a naopak, ak z maximalizačnej, tak duálna úloha je minimalizačná. Dá sa preto hovoriť o dvojici vzájomne duálnych úloh. Teraz ešte ukážeme, ako táto dvojica vyzerá. Máme primárnu úlohu v štandardnom tvare:

$$(P) \qquad min \left\{ c^T x | Ax = b, x \geq 0 \right\}.$$

K nej prislúchajúca duálna úloha vyzerá nasledovne:

$$max \left\{ b^T y | A^T y \leq c \right\},$$

ktorú možno pomocou doplnkových premenných zapísať v tvare

$$(D) \qquad max\left\{b^T y | A^T y + s = c, s \geq 0\right\}.$$

Pod primárnou úlohou budeme mať v ďalších stránkach na mysli minimalizačnú úlohu, pod duálnou maximalizačnú. Pre ne teraz ešte zadefinujem niekoľko pojmov.

$\mathcal{P} = \{x \in R^n | Ax = b; x \geq 0\}$ sa nazýva množinou primárne prípustných riešení. $x^* \in \mathcal{P}$ sa nazýva optimálne riešenie úlohy (P), ak platí, že $c^T x^* \leq c^T x$ pre každé $x \in \mathcal{P}$. V reálnej situácii môže nastať prípad, keď je $\mathcal{P}$ prázdna, t.j. neexistuje primárne prípustné riešenie. Potom je daná úloha neprípustná. Ďalšou možnosťou, ktorú treba spomenúť je prípad, ak $\mathcal{P}$ nie je prázdna, t.j. existuje primárne prípustné riešenie, avšak optimálna hodnota účelovej funkcie je rovná $-\infty$. Takáto úloha sa nazýva neohraničená. Podobne môžeme zadefinovať množinu duálne prípustných bodov ako $\mathcal{D} = \left\{(y, s) \in R^m \times R^n | A^T y + s = c; s \geq 0\right\}$.

Pre dvojicu úloh (P) a (D) platia nasledovné tvrdenia, ktoré využívame pri hľadaní optimálneho riešenia. Prvým, ktoré uvedieme je tzv. slabá veta o dualite.

Veta 1.2.1. (Slabá veta o dualite)
Pre každé prípustné riešenie $x = (x_1, ..., x_n)^T$ primárnej úlohy a každé prípustné riešenie $y = (y_1, ..., y_m)^T$ duálnej úlohy platí $c^T x \geq b^T y$.

Dôkaz.

Vieme, že y je prípustné riešenie duálnej úlohy. Preto musí spĺňať nerovnosť $A^T y \leq c$, ktorú transponovaním upravíme na $y^T A \leq c^T$. Teda aj $y^T Ax \leq c^T x$. Teraz ešte využijeme, že x je prípustným riešením primárnej úlohy, t.j. platí nerovnosť $Ax = b$. Dokopy dostávame $c^T x \geq y^T Ax = y^T b = b^T y$. □

Slabá veta o dualite má veľký význam. Podľa nerovnosti $c^T x \geq b^T y$ je hodnota účelovej funkcie minimalizačnej úlohy pre akékoľvek prípustné riešenie horným odhadom účelovej funkcie maximalizačnej úlohy pre všetky jej prípustné riešenia a naopak. Rozdiel $b^T y - c^T x$ sa označuje ako duálna medzera. Jej veľkosť je rozhodujúca pre overenie optimality prípustných riešení primárnej a duálnej úlohy. Ukážeme, že ak x^* je prípustné riešenie primárnej úlohy a y^* je prípustné riešenie duálnej úlohy a veľkosť duálnej medzery je rovná nule, t.j. $c^T x^* = b^T y^*$, potom x^* a y^* sú aj optimálne. Týmto spôsobom nám poskytuje postačujúce podmienky pre optimalitu prípustných riešení. Nech teda x je prípustné riešenie primárnej úlohy, potom podľa slabej vety o dualite platí $c^T x \geq b^T y^* = c^T x^*$. Ale x^* je prípustné, potom musí byť aj optimálne. Podobným spôsobom je možné ukázať, že y^* je optimálne riešenie duálnej úlohy.

Dôsledok 1.2.2. *Ak je x prípustné riešenie primárnej úlohy, (y, s) prípustné riešenie duálnej úlohy a je splnená rovnosť $c^T x = b^T y$ potom x, resp. (y, s) sú optimálne riešenia primárnej resp. duálnej úlohy.*

Druhým dôležitým výsledkom teórie duality je nasledovná veta, známa tiež ako silná veta o dualite.

Veta 1.2.3. (Silná veta o dualite)[Vanderbei[2]]
Ak primárna úloha má optimálne riešenie $x^* = (x_1^*, ..., x_n^*)^T$, potom aj duálna úloha má optimálne riešenie $y^* = (y_1^*, ..., y_m^*)^T$ také, že platí $b^T y^* = c^T x^*$.

Nulová veľkosť duálnej medzery je týmto postačujúcou (podľa Dôseldku 1.2.2), ale zároveň aj nutnou podmienkou optimality (podľa Vety 1.2.3). Zo silnej vety o dualite nám vyplýva, že ak jedna z dvojice vzájomne duálnych úloh má optimálne riešenie, tak aj tá druhá má optimálne riešenie. Otázne je, čo nastáva v situácii, ak napr. primárna úloha je neohraničená alebo neprípustná. Z neohraničenosti primárnej úlohy s využitím silnej vety o dualite dostávame, že duálna úloha je neprípustná. Veľkosť duálnej medzery je v tomto prípade nulová, pričom rovnosť sa nadobúda v $-\infty$. Podobne máme druhú možnosť, ak primárna úloha je neohraničená. Potom je duálne úloha neprípustná. Veľkosť duálnej medzery je rovná nule a rovnosť nastáva v ∞. Posledným prípadom, ktorý treba uvažovať, je ak sú obe úlohy neprípustné. Tu je veľkosť duálnej medzery nenulová. Metódy vnútorného bodu nám poskytujú optimálne riešenie, ktoré je ostro komplementárne. Preto by sme mali ešte spomenúť vety o komplementarite.

Veta 1.2.4. (Veta o komplementarite)[Vanderbei[2]]
Nech $x = (x_1, ..., x_n)^T$ je prípustné riešenie primárnej úlohy a $y = (y_1, ..., y_m)^T$ je prípustné riešenie duálnej úlohy. Nech $(s_1, ..., s_m)^T$ značí príslušné duálne doplnkové premenné. Potom x a y sú optimálne riešenia pre primárnu resp. duálnu úlohu vtedy a len vtedy, ak platí:

$$x_j s_j = 0, j = 1, ..., n, \tag{1.1}$$

Dôkaz.
V nasledujúcich pár riadkoch najprv ukážeme, že platí rovnosť:

$$c^T x - b^T y = x^T s. \tag{1.2}$$

Keďže $x, s \geq 0$ potom už z rovnosti (1.2) jednoznačne vyplýva platnosť tvrdenia.

Vieme, že pre y, s duálne prípustné platí $A^T y + s = c$. S využitím tejto rovnosti potom máme, že:

$$c^T x - b^T y = (A^T y + s)^T x - b^T y = s^T x + y^T Ax - b^T y. \tag{1.3}$$

Keďže x je prípustné riešenie, potom platí $Ax = b$, a dosadením do (1.3) získavame:

$$s^T x + y^T Ax - b^T y = s^T x + \underbrace{y^T b - b^T y}_{=0} = s^T x = x^T s. \tag{1.4}$$

Podarilo sa nám tu ukázať, že platí rovnosť (1.2). $\square$

Dôsledok 1.2.5. *Nech $x \in R^n$, $y \in R^m$ a $s \in R^n$. Potom x je optimálne riešenie (P) a (y,s) je optimálne riešenie (D), práve vtedy, keď je splnené:*

$$\begin{aligned} Ax = b,\ x \geq 0,\\ A^T y + s = c,\ s \geq 0,\\ x_i s_i = 0,\ \forall i. \end{aligned} \tag{1.5}$$

Podmienky (1.5) sú uvádzané v literatúre [Vanderbei[2]] pod názvom *Karush-Kuhn-Tuckerove podmienky*, skrátene *KKT podmienky.*

Veta o komplementarite zároveň pomáha v situácii, ak je potrebné nájsť optimálne riešenie jednej z dvojice vzájomne duálnych úloh, pričom optimálne riešenie tej druhej je nám známe. Rovnosti (1.1) sa označujú ako podmienky komplementarity. Vyplýva z nich, že ak máme optimálne riešenie x^* primárnej úlohy a (y^*, s^*) duálnej úlohy, potom jedno z x_j, s_j je rovné nule, alebo sú obe nulové pre $j = 1, ..., n$. Metódy vnútorného bodu nám dávajú riešenia, ktoré sú ostro komplementárne. To znamená, že riešenia spĺňajú podmienky komplementarity, avšak nenastáva prípad, že pre optimálne riešenie sú obe z x_j, s_j rovné nule pre $j = 1, ..., n$, vždy je nulové práve jedno z nich.

Veta 1.2.6. (Veta o ostrej komplementarite)[Vanderbei[2]]
Ak úloha lineárneho programovania má optimálne riešenia, potom existujú také optimálne riešenia x^*, (y^*, s^*) primárnej resp. duálnej úlohy, ktoré sú ostro komplementárne, t.j. $x^* + s^* > 0$.

Kapitola 2

Metódy vnútorného bodu

Teraz ukážeme, ako je možné riešiť úlohy lineárneho programovania. V minulosti sa najčastejšie používala simplexová metóda. Neskôr sa však ukázalo, že nie je vhodná na riešenie úloh veľmi veľkých rozmerov a do popredia sa dostali metódy vnútorného bodu, ktoré zahŕňajú pestrú škálu rôznych algoritmov. Najčastejšie používané sú tzv. primárno-duálne algoritmy, ktoré v nasledujúcich pár stranách popíšeme. Ešte predtým upozorníme na neštandardné označenie: Pod $x \in R^n$ budeme rozumieť vektor v tvare $x = (x_1, x_2, ..., x_n)^T$. Potom X budeme chápať ako diagonálnu maticu v tvare:

$$X = \begin{pmatrix} x_1 & & & \\ & x_2 & & \\ & & \ddots & \\ & & & x_m \end{pmatrix}.$$

Pod označením x^{-1}, e budeme rozumieť vektory:

$$x^{-1} = (1/x_1, 1/x_2, ..., 1/x_n)^T,$$

$$e = (1, 1, ..., 1)^T.$$

2.1 Centrálna trajektória

Majme primárnu úlohu v štandardnom tvare:

$$(P) \qquad min\left\{c^T x | Ax = b, x \geq 0\right\}. \tag{2.1}$$

K nej prislúchajúca duálna úloha je:

$$(D) \qquad max\left\{b^T y | A^T y + s = c, s \geq 0\right\}. \tag{2.2}$$

Bod x nazývame *ostro prípustným primárnym bodom*, ak $x > 0$ a $x \in \mathcal{P}$. Množinu všetkých ostro prípustných primárnych bodov označíme ako:

$$\mathcal{P}^0 := \{x | Ax = b, x > 0\}.$$

Podobne môžeme zadefinovať množinu *ostro prípustných duálnych bodov* ako:

$$\mathcal{D}^0 := \left\{(y, s) | A^T y + s = c, s > 0\right\}.$$

V metódach vnútorného bodu sú dôležité nasledovné dva predpoklady.

Predpoklad 2.1.1. *Matica A má plnú hodnosť, t.j. $h(A) = m$, pričom $m \leq n$.*

Predpoklad (2.1.1) nám zabezpečuje, že ak s vytvára nejakú duálne prípustnú dvojicu, tak ju tvorí s jediným y.

Metódy vnútorného bodu hľadajú optimálne riešenia úlohy (2.1) z vnútra množiny $\mathcal{P}$ ako limitu postupnosti riešení úloh, ktoré sú odvodené z úloh (2.1), (2.2). Preto je dôležitý nasledovný predpoklad.

Predpoklad 2.1.2. *Množiny $\mathcal{P}^0$, $\mathcal{D}^0$ sú neprázdne.*

V ďalšej časti vyložíme najdôležitejšie poznatky metód vnútorného bodu v oblasti lineárneho programovania, ktoré je možné nájsť napr. vo [Vanderbei [2]], [Wright [12]] alebo [Halická [11]].

Poznámka 2.1.3. Odteraz budeme v celej kapitole predpokladať, že sú splnené Predpoklady (2.1.1), (2.1.2).

Už sme spomínali, že metódy vnútorného bodu neriešia priamo pôvodnú úlohu, ale postupnosť úloh z nej odvodených. Teraz ukážeme, akú formu majú dané odvodené úlohy. S využitím logaritmickej bariérovej funkcie nahradíme pôvodnú účelovú funkciu úlohy (P) $c^T x$ funkciou

$$f_\tau(x) := c^T x - \tau \sum_{i=1}^{n} ln(x_i),$$

kde τ je kladná konštanta a namiesto úlohy (2.1) riešime úlohu:

$$(P_\tau) \qquad min\{f_\tau(x)\,|Ax = b, x > 0\}. \tag{2.3}$$

Lema 2.1.4. *Nech $\tau > 0$. Potom $x \in R^n$ je optimálne riešenie (P_τ) práve vtedy, keď existuje také $y \in R^m$, $s \in R^n$, že:*

$$\begin{aligned} Ax = b,\ x > 0, \\ A^T y + s = c,\ s > 0, \\ x_i s_i = \tau,\ \forall i. \end{aligned} \tag{2.4}$$

Dôkaz.

Úlohu na hľadanie viazaného extrému prevedieme na úlohu hľadania voľného extrému. Zostrojíme Lagrangeovu funkciu pre úlohu (P_τ), v tvare:

$$L(x, y) = c^T x - \tau \sum_{i=1}^{n} ln(x_i) - y^T (Ax - b); \tag{2.5}$$

kde $y \in R^m$, $x > 0$. Nutné a postačujúce podmienky pre extrém sú:

$$\nabla_x L(x, y) = c - \tau X^{-1} e - A^T y = 0, \tag{2.6}$$

$$\nabla_y L(x, y) = Ax - b = 0, x > 0. \tag{2.7}$$

Teda x je riešenie $(P_\tau) \Leftrightarrow$ existuje také y, že (x, y) spĺňajú podmienky (2.6), (2.7). Ak označíme: $\tau X^{-1} e \overset{ozn.}{=} s > 0$, z toho jednoduchou úpravou po zložkách dostaneme, že $\tau \frac{1}{x_i} = s_i \Leftrightarrow x_i s_i = \tau$. Tým sme sa dostali k podmienkam (2.4), čo bolo naším cieľom a dôkaz je skončený. $\square$

Podmienky (2.4) sú tzv. *centrujúce podmienky.* Jedná sa v podstate o perturbáciu systému (1.5), ak 0 v poslednej rovnici nahradíme za τ. Pomocou logaritmickej barérovej funkcie môžeme podobným spôsobom ako úlohu (2.1) transformovať aj úlohu (2.2) na:

$$(D_\tau) \qquad min \left\{ g_\tau (y, s) \,|\, A^T y + s = c, s > 0 \right\}, \tag{2.8}$$

kde $g_\tau(y, s) := b^T y + \tau \sum_{i=1}^{n} ln(s_i)$.

Lema 2.1.5. *Nech $\tau > 0$, $y \in R^m$, $s \in R^n_+$. Potom (y, s) je riešenie (D_τ) práve vtedy, keď existuje také $x \in R^n$, že sú splnené podmienky (2.4).*

Dôkaz.
Dôkaz možno urobiť podobným spôsobom ako pri Leme (2.1.4). □

Lema 2.1.6. *Pre každé $\tau > 0$ majú úlohy (P_τ), (D_τ) práve jedno riešenie.*

Dôsledok 2.1.7. *Pre každé $\tau > 0$ majú majú centrujúce podmienky (2.4) práve jedno riešenie x_τ, y_τ, s_τ.*

Teraz môžeme zadefinovať aj pojem centrálnej trajektórie. Centrálnu trajektóriu definujeme ako množinu riešení systému (2.4) závislých na hodnote parametra τ:

$$\mathcal{C} = \{x_\tau, y_\tau, s_\tau | \tau > 0\}.$$

V teórii metód vnútorného bodu sa ukázalo, že pre $\tau \to 0$ konverguje centrálna trajektória k ostrokomplementárnemu optimálnemu riešeniu úlohy lineárneho programovania, t.j. platí $(x_\tau, y_\tau, s_\tau) \to (x^*, y^*, s^*)$ a $x^* + s^* > 0$.

2.2 Newtonove smery

V metódach vnútorného bodu teda potrebujeme vypočítať x_τ, y_τ, s_τ, čo nás ako už vieme dovedie k optimálnemu riešeniu danej úlohy, pre $\tau \to 0$. V každej iterácii budeme teda hľadať riešenie systému:

$$\left\{ \begin{array}{l} Ax = b, \\ A^T y + s = c, \\ x_i s_i = \tau, \ \forall i \end{array} \right\} \Leftrightarrow \left\{ \begin{array}{l} Ax = b, \\ A^T y + s = c, \\ XSe = \tau e, \end{array} \right\} \tag{2.9}$$

pre fixnú hodnotu parametra τ. Úlohu riešiť systém (2.9) môžeme jednoducho previesť na úlohu hľadania nulového bodu funkcie $F(x,y,s) : R^{2n+m} \to R^{2n+m}$, ktorá je v tvare:

$$F(x,y,s) = \begin{pmatrix} Ax - b \\ A^T y + s - c \\ XSe - \tau e \end{pmatrix}. \tag{2.10}$$

Rovnicu $F(x,y,s) = 0$ môžeme riešiť *pomocou modifikovanej Newtonovej metódy so skrátenou dĺžkou kroku.* Predpokladáme, že máme k dispozícii ostro prípustný štartovací bod (x^0, y^0, s^0) a parameter $\tau > 0$ bude konštantný, rovný τ_0. V prvej iterácii získame ostro prípustné riešenie (x^1, y^1, s^1). Poznamenajme ešte, že prvé dve rovnice v (2.10) sú lineárne, a teda ich všetky nasledujúce newtonovské iterácie budú spĺňať. Potrebujeme však ešte splniť podmienku kladnosti x a s, čo sa dosiahne práve redukciou dĺžky kroku. K tomu sa ešte vrátime pri naznačení jednej konkrétnej iterácie. Do druhej iterácie zvolíme za štartovací bod (x^1, y^1, s^1) a zmenšíme hodnotu τ z τ_0 na τ_1. Spôsob, ako znižujeme veľkosť parametra τ po každej iterácii popíšeme neskôr. Získame nový ostro prípustný bod (x^2, y^2, s^2) a takto budeme pokračovať ďalej, pokým sa dostatočne nepriblížime k optimálnemu riešeniu, t.j. pokým v niektorej iterácii nenastane $x^T s < \epsilon$, kde ϵ je vopred stanovená tolerančná konštanta. Teraz naznačíme, ako pomocou Newtonovej metódy počítame nové iteračné body. Zoberme, že sme v $(k+1)$-tej iterácii, t.j. štartovací bod je (x^k, y^k, s^k), ktorý je ostro prípustný. Máme, že:

$$F(x^{k+1}, y^{k+1}, s^{k+1}) \approx F(x^k, y^k, s^k) + J(x^k, y^k, s^k)(\Delta x^k, \Delta y^k, \Delta s^k)^T = 0,$$

kde $(\Delta x^k, \Delta y^k, \Delta s^k) = (x^{k+1}, y^{k+1}, s^{k+1}) - (x^k, y^k, s^k)$ a $J(x,y,s)$ je *Jakobiho matica* funkcie $F(x,y,s)$. Našou úlohou je vypočítať z rovnosti

$$F(x^k, y^k, s^k) + J(x^k, y^k, s^k)(\Delta x^k, \Delta y^k, \Delta s^k)^T = 0$$

smery $(\Delta x^k, \Delta y^k, \Delta s^k)$. Dostaneme ich ako riešenie systému:

$$J(x^k, y^k, s^k) \begin{pmatrix} \Delta x^k \\ \Delta y^k \\ \Delta s^k \end{pmatrix} = -F(x^k, y^k, s^k), \tag{2.11}$$

čo je vlastne systém:

$$\begin{pmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S^k & 0 & X^k \end{pmatrix} \begin{pmatrix} \Delta x^k \\ \Delta y^k \\ \Delta s^k \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ -X^k S^k e + \tau_k e \end{pmatrix}. \tag{2.12}$$

$(\Delta x^k, \Delta y^k, \Delta s^k)$ sa podľa použitej metódy nazývajú *Newtonove smery.* Novú iteráciu by sme dostali ako $(x^{k+1}, y^{k+1}, s^{k+1}) = (x^k, y^k, s^k) + \alpha(\Delta x^k, \Delta y^k, \Delta s^k)$, kde $\alpha \in (0, 1]$ je parameter označujúci dĺžku kroku a určíme ho tak, aby bola splnená podmienka $(x^{k+1}, s^{k+1}) > 0$. Poznamenajme, že voľba $\alpha = 1$ sa nazýva *plným Newtonovým krokom.* Tento krok je však často neprípustný, a preto sa v praxi volí $\alpha < 1$. Je však $(x^{k+1}, y^{k+1}, s^{k+1})$ prípustné riešenie? Ak áno, musí platiť, že $Ax^{k+1} = b, A^T y^{k+1} + s^{k+1} = c$.

$$Ax^{k+1} = A(x^k + \alpha \Delta x^k) = Ax^k + \underbrace{\alpha A \Delta x^k}_{=0, \text{ podľa } (2.12)} = Ax^k = b,$$

keďže sme predpokladali, že (x^k, y^k, s^k) je prípustný. Podobne jednoducho sa dá ukázať, že aj $A^T y^{k+1} + s^{k+1} = c$. To znamená, že ak štartovací bod (x^0, y^0, s^0) je prípustný, potom aj všetky ďalšie vypočítané iterácie sú prípustné.

Ak by sme zvolili $\tau = 0$, znamenalo by to, že by sme hľadali riešenie pôvodného neperturbovaného systému rovností z podmienok (1.5). Takto získané smery $(\Delta x^k, \Delta y^k, \Delta s^k)$ sa nazývajú *čisté Newtonove smery*, alebo aj *afinno-škálovacie smery.* Väčšina primárno-duálnych algoritmov však volí $\tau > 0$. Dôvodom je skutočnosť, že touto voľbou získavame smery $(\Delta x^k, \Delta y^k, \Delta s^k)$ (v literatúre [Wright[12]] sa označujú ako *centrujúce smery*), ktoré sú orientované do vnútra kladného ortantu definovaného ako $(x, s) > 0$. Potom je možné zvoliť väčšiu dĺžku kroku ako pri afinno-škálovacích smeroch, u ktorých je pri dlhšom kroku väčšie riziko prekročenia hranice kladného ortantu, a preto sme nútení vziať $\alpha << 1$, čo spomaľuje konvergenciu algoritmu. My budeme voliť hodnotu parametra τ adaptívne. Za tým účelom si zavedieme centrujúci parameter $\sigma \in [0, 1]$ a parameter duálnej medzery $\mu(x, s)$, ktorý je definovaný ako $\mu(x, s) = \frac{1}{n} \sum_{i=1}^{n} x_i s_i = \frac{x^T s}{n}$. Ďalej budeme $\mu(x, s)$ označovať už iba ako μ a položíme $\tau = \sigma\mu$.

2.3 Neprípustné štartovacie body

Doteraz sme vôbec nerozoberali problematiku štartovacieho bodu, len sme predpokladali, že je ostro prípustný. To znamená, že (x^0, y^0, s^0), kde $x^0 > 0$ a $s^0 > 0$ spĺňa

$Ax^0 = b$ a $A^T y^0 + s^0 = c$. Daným rovnostiam následne vyhovujú aj všetky nasledujúce iterácie. Pre veľké množstvo úloh je však náročné takýto počiatočný bod nájsť. Daný problém sa dá odstrániť preformulovaním úlohy, čo však môže mať za následok zvýšenú náročnosť riešenia úlohy. Pohodlnejší prístup ponúkajú tzv. neprípustné metódy vnútorného bodu. V nich od štartovacieho bodu požadujeme iba splnenie podmienky $(x^0, s^0) > 0$. Daný prístup si nevyhnutne vyžaduje úpravu systému na hľadanie smeru (2.12) tak, aby sme každou iteráciou zároveň zlepšovali prípustnosť. Z toho dôvodu zadefinujeme reziduá r_b, r_c následne: $r_b = Ax^k - b$, $r_c = A^T y^k + s^k - c$ a systém (2.12) prevedieme na tvar:

$$\begin{pmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S^k & 0 & X^k \end{pmatrix} \begin{pmatrix} \Delta x^k \\ \Delta y^k \\ \Delta s^k \end{pmatrix} = \begin{pmatrix} -r_c \\ -r_b \\ -X^k S^k e + \sigma_k \mu_k e \end{pmatrix}. \tag{2.13}$$

Takto vypočítaný smer je stále Newtonov krok k bodu patriacemu centrálnej trajektórii, avšak zmenou systému pridaním reziduí je tlačený k odstráneniu neprípustnosti, a to už v prvom kroku. Pri voľbe dĺžky kroku $\alpha = 1$ sa r_b a r_c stanú nulové, t.j. úplne sa eliminuje neprípustnosť, čo sa zachová aj v ďalších iteráciách. Táto voľba je v praxi nemožná. Prípustnosť sa v jednotlivých ietráciách zlepšuje, rovnako ako aj komplementarita, znižovaním hodnoty hodnoty parametra μ k nule a cieľ o odstránení neprípustnosti sa podarí naplniť až neskôr. Do skupiny neprípustných metód vnútorného bodu patrí aj tzv. Mehrotrov prediktor-korektor algoritmus.

2.4 Mehrotrov prediktor-korektor algoritmus

Mehrotrov prediktor-korektor algoritmus používa v súčasnosti väčšina programov na lineárnu optimalizáciu. V nasledujúcich pár riadkoch sa pokúsime túto metódu popísať. Klasický Newtonov smer je vylepšený o korekčnú zložku, ktorej výpočtová náročnosť je relatívne malá. Výhodou je aj adaptívna voľba parametra σ v každej iterácii. Hľadaný smer pozostáva z troch komponentov:

2.4.1 Afinno-škálovací "prediktor"komponent

Jedná sa v podstate o čistý Newtonov smer funkcie $F(x, y, s)$ definovanej podľa (2.10), kde zvolíme $\tau = 0$, teda ako riešenie systému:

$$\begin{pmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S^k & 0 & X^k \end{pmatrix} \begin{pmatrix} \Delta x_k^{af} \\ \Delta y_k^{af} \\ \Delta s_k^{af} \end{pmatrix} = \begin{pmatrix} -r_c \\ -r_b \\ -X^k S^k e \end{pmatrix}, \tag{2.14}$$

kde $r_b = Ax^k - b$ a $r_c = A^T y^k + s^k - c$. Následne určíme maximálnu možnú dĺžku krokov (aby sme nevyšli z kladného ortantu) oddelene pre primárne a duálne premenné podľa vzťahu:

$$\begin{aligned} \alpha_{af}^{pri} &= \arg\max \left\{ \alpha \in [0,1] \, | x + \alpha \Delta x^{af} \geq 0 \right\}, \\ \alpha_{af}^{dual} &= \arg\max \left\{ \alpha \in [0,1] \, | s + \alpha \Delta s^{af} \geq 0 \right\}. \end{aligned} \tag{2.15}$$

Ďalej definujeme μ_{af} ako hypotetickú hodnotu μ, ak by sme zvolili plný krok: $\mu_{af} = (x + \alpha_{af}^{pri} \Delta x^{af})^T (s + \alpha_{af}^{dual} \Delta s^{af})/n$. Ak $\mu_{af} << \mu$, potom afinno-škálovací smer je dostatočne dobrý a umožní signifikantnú redukciu μ, teda môžeme zobrať parameter σ blízky 0. Ak je však μ_{af} len o trochu menšie ako μ, potom zvolíme hodnotu σ bližšie k 1, čo nám zaručí priblíženie sa k centrálnej trajektórii a následne lepšiu pozíciu pre pokles μ v ďalšej iterácii. V praxi sa voľba $\sigma = (\frac{\mu_{af}}{\mu})^3$ ukázala ako veľmi efektívna.

2.4.2 Centrujúci komponent

Jeho hodnota závisí od adaptívnej voľby centrujúceho parametra σ. Získa sa riešením systému, ktorý má rovnakú maticu ako (2.14), ale pravú stranu v tvare $(0, 0, \sigma_k \mu_k e)^T$. Výhodné je vypočítať ho spolu s korekčným členom. Ten tiež určíme podľa systému (2.14), avšak s pravou stranou v tvare $(0, 0, -\Delta X_k^{af} \Delta S_k^{af} e)^T$. Keďže sú navyše obe zložky nezávislé jedna na druhej, nie je potom žiaden dôvod, aby sa určovali oddelene. Môžeme ich zlúčiť do jedného smeru jednoduchým spočítaním prislúchajúcich pravých strán a následným zriešením systému rovníc.

2.4.3 "Korektor"komponent

Afinno-škálovací smer je získaný z lineárnej aproximácie KKT podmienok (1.5) a keďže je počítaný samostatne, je možné ho použiť na odhad chyby v aproximácii. Táto znalosť nám umožňuje vypočítať "korektor"smer a vylepšiť lineárny model na kvadratický. Ako motiváciu si najskôr ukážme, ako sa zmení i-ty súčin $x_i s_i$, ak by sme zvolili plný krok. Z (2.14), vynechajúc indexy značiace iteráciu, máme $(x_i +$

$\Delta x_i^{af})(s_i + \Delta s_i^{af}) = x_i s_i + x_i \Delta s_i^{af} + s_i \Delta x_i^{af} + \Delta x_i^{af} \Delta s_i^{af} = \Delta x_i^{af} \Delta s_i^{af}$. Súčin $x_i s_i$ sa pretransformoval na $\Delta x_i^{af} \Delta s_i^{af}$ namiesto predpokladanej nuly. Korekčná zložka kroku sa bude snažiť kompenzovať túto odchýlku tým, že vzájomné súčiny približuje k ich cieľovej hodnote 0. Táto zložka vyhovuje systému:

$$\begin{pmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S^k & 0 & X^k \end{pmatrix} \begin{pmatrix} \Delta x_k^{kor} \\ \Delta y_k^{kor} \\ \Delta s_k^{kor} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ -\Delta X_k^{af} \Delta S_k^{af} e \end{pmatrix}. \tag{2.16}$$

Kombinovaný centrujúco-korekčný smer budeme počítať ako riešenie systému:

$$\begin{pmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S^k & 0 & X^k \end{pmatrix} \begin{pmatrix} \Delta x_k^{c-k} \\ \Delta y_k^{c-k} \\ \Delta s_k^{c-k} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \sigma_k \mu_k e - \Delta X_k^{af} \Delta S_k^{af} e \end{pmatrix}. \tag{2.17}$$

2.4.4 Celkový smer

Celkový smer v k-tej iterácii bude súčet afinno-škálovacej a centrujúco-korekčnej zložky. Bude mať tvar:

$$\left(\Delta x^k, \Delta y^k, \Delta s^k\right) = \left(\Delta x_k^{af}, \Delta y_k^{af}, \Delta s_k^{af}\right) + \left(\Delta x_k^{c-k}, \Delta y_k^{c-k}, \Delta s_k^{c-k}\right). \tag{2.18}$$

Následne určíme maximálnu možnú dĺžku kroku pre primárne a duálne premenné zvlášť podľa vzťahov:

$$\begin{aligned} \alpha_k^{pri} &= \min(0.99\alpha_{max}^{pri}, 1), \\ \alpha_k^{dual} &= \min(0.99\alpha_{max}^{dual}, 1), \end{aligned} \tag{2.19}$$

kde

$$\begin{aligned} \alpha_{max}^{pri} &= \arg\max\left\{\alpha \geq 0 | x^k + \alpha \Delta x^k \geq 0\right\}, \\ \alpha_{max}^{dual} &= \arg\max\left\{\alpha \geq 0 | s^k + \alpha \Delta s^k \geq 0\right\}. \end{aligned} \tag{2.20}$$

Približné riešenie primárnej resp. duálnej úloy v k-tej iterácii je v tvare:

$$\begin{aligned} x^{k+1} &= x^k + \alpha_k^{pri} \Delta x^k, \\ \left(y^{k+1}, s^{k+1}\right) &= \left(y^k, s^k\right) + \alpha_k^{dual} \left(\Delta y^k, \Delta s^k\right). \end{aligned} \tag{2.21}$$

2.5 Redukovaný KKT systém

Časovo najnáročnejšou zložkou v každej iterácii je určenie smeru kroku, t.j. riešenie KKT systému:

$$\begin{pmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S & 0 & X \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta s \end{pmatrix} = \begin{pmatrix} -r_b \\ -r_c \\ -r_{xs} \end{pmatrix}, \tag{2.22}$$

kde

$$r_{xs} = \begin{cases} -XSe & \text{, pre afinno-škálovací smer,} \\ \sigma\mu e - \Delta X^{af} \Delta S^{af} e & \text{, pre centrujúco-korekčný smer} \end{cases} \tag{2.23}$$

Systém (2.22) prepíšeme z maticového tvaru a pokúsime sa ho čo možno najjednoduchšie vyriešiť. Po úprave dostávame:

$$A^T \Delta y + \Delta s = -r_c, \tag{2.24}$$

$$A \Delta X = -r_b, \tag{2.25}$$

$$S \Delta x + X \Delta s = -r_{xs}. \tag{2.26}$$

Najskôr z (2.26) vyjadríme $\Delta s = -X^{-1}(r_{xs} + S\Delta x)$, čo dosadíme do (2.24) a získavame:

$$A^T \Delta y - X^{-1}(r_{xs} + S\Delta x) = -r_c, \tag{2.27}$$

čo následne upravíme na:

$$A^T \Delta y - X^{-1} S \Delta x = -r_c + X^{-1} r_{xs}. \tag{2.28}$$

Ak označíme $D = S^{-1/2}X^{1/2}$, dostávame sa od (2.22) k systému:

$$\begin{pmatrix} A^T & -D^{-2} \\ 0 & A \end{pmatrix} \begin{pmatrix} \Delta y \\ \Delta x \end{pmatrix} = \begin{pmatrix} -r_c + X^{-1}r_{xs} \\ -r_b \end{pmatrix}, \tag{2.29}$$

$$\Delta s = -X^{-1}(r_{xs} + S\Delta x), \tag{2.30}$$

ktorý je z literatúry [Wright[12]] známy pod názvom *redukovaný KKT systém*. Teraz budeme eliminovať Δx. Z (2.29) máme:

$$A^T\Delta y - D^{-2}\Delta x = -r_c + X^{-1}r_{xs}. \tag{2.31}$$

Ak do (2.31) dosadíme z (2.24) $A^T\Delta y = -r_c - \Delta s$, dostávame:

$$-r_c - \Delta s - D^{-2}\Delta x = -r_c + X^{-1}r_{xs}, \tag{2.32}$$

z čoho nahradením $D^{-2} = X^{-1}S$ a následnými jednoduchými úpravami dostaneme:

$$\Delta x = -S^{-1}(r_{xs} + X\Delta s), \tag{2.33}$$

Keď (2.33) dosadíme do (2.25), dostávame, že:

$$A(-S^{-1}(r_{xs} + X\Delta s)) = -r_b, \tag{2.34}$$

čo jednoducho upravíme na:

$$-AS^{-1}X\Delta s = -r_b + AS^{-1}r_{xs}. \tag{2.35}$$

Ak do (2.35) dosadíme z (2.24) $\Delta s = -r_c - A^T\Delta y$, máme:

$$A\underbrace{S^{-1}X}_{=D^2}A^T\Delta y = -r_b - AS^{-1}Xr_c + AS^{-1}r_{xs}, \tag{2.36}$$

čo upravíme na:

$$AD^2A^T\Delta y = -r_b + A(-S^{-1}Xr_c + S^{-1}r_{xs}). \tag{2.37}$$

Celkovo sme sa od systému (2.24)-(2.26) prepracovali k systému:

$$AD^2A^T\Delta y = -r_b + A(-S^{-1}Xr_c + S^{-1}r_{xs}), \tag{2.38}$$

$$\Delta s = -r_c - A^T\Delta y, \tag{2.39}$$

$$\Delta x = -S^{-1}(r_{xs} + X\Delta s). \tag{2.40}$$

Systém (2.38)-(2.40) sa zvykne v literatúre [Wright[12]] označovať pod názvom *normálny systém rovníc* a na jeho riešenie, kontrétnejšie na určenie Δy z (2.38), sa používa tzv. *Choleského rozklad.*

Symbolom M označíme maticu $M = AD^2A^T$. Ak má matica A plnú hodnosť, t.j. ak $h(A) = m$, potom M je kladne definitná a použitím Choleského rozkladu získame dolnú trojuholníkovú maticu L a diagonálnu maticu D také, že platí $M = LDL^T$. (2.38) sa pretransformovalo na $LDL^T\Delta y = z \overset{ozn.}{=} -r_b+A(-S^{-1}Xr_c+S^{-1}r_{xs})$. Namiesto (2.38) budeme riešiť dva čiastkové systémy rovníc:

$$Lw = z,$$

$$DL^T\Delta y = w,$$

ktoré sú ľahko riešiteľné, vzhľadom na fakt, že L je dolná trojuholníková a DL^T je horná trojuholníková matica.

Hodnoty $\Delta x, \Delta s$ sa následne už jednoducho dopočítajú podľa vzťahov (2.39),(2.40).

2.6 Použitý algoritmus

Jedným z programov, ktorý budeme používať v tejto práci je, LOQO [5]. Tento softvér využíva pri riešení úloh lineárneho programovania oproti vyššie opísanému Mehrotrovmu prediktor-korektor algoritmu navyše nasledovnú modifikáciu.

2.6.1 Presolver

Základným predpokladom pri použití primárno-duálneho algoritmu je plná hodnosť matice A. Túto požiadavku dokážeme splniť použitím tzv. presolvera. Táto časť programu vykonáva nasledovné úpravy matice A:

- kontrola neprípustnosti, t.j. či nulovému riadku v matici zodpovedá nulová zložka vo vektore pravej strany,
- vynechanie nulového riadka, ak mu zodpovedá nulová zložka na pravej strane,

- odstránenie jedného z dvojice rovnakých riadkov (ak je jeden nenulovým násobkom druhého),
- odstránenie jedného z dvojice rovnakých stĺpcov,
- odstránenie fixnej premennej, t.j. takej, u ktorej je dolné a horné ohraničenie zhodné,
- vyradenie určenej premennej, ak riadok matice obsahuje jedinú nenulovú hodnotu, premennú možno jednoducho určiť.

Úprava matice s využitím presolvera je výpočtovo menej náročná než jedna iterácia, pričom program zároveň ponúka možnosť jeho nepoužitia.

2.7 Homogénna samoduálna metóda

Táto metóda, ktorú budeme v našej práci tiež používať, je dostatočne podrobne popísaná v diplomovej práci Ľubomíra Koršňáka [9], a preto ju tu uvádzať nebudeme.

Kapitola 3

Optimalizácia vo financiách

Banky a finančné inštitúcie potrebujú zhodnocovať svoj majetok, či už z dôvodu maximalizácie zisku, prípadne potreby splatenia akýchsi budúcich platieb. Veľkosť, ako aj realizácia týchto budúcich záväzkov, často nemusí byť dopredu známa.

Najtypickejším príkladom toho sú poisťovne, a to konkrétne tie, ktoré sa zaoberajú životným poistením, kde výška a čas pohľadávky závisí od dĺžky života daného zákazníka. Spôsob, ako sa voči takýmto platbám zabezpečiť, je investovanie do rozličných finančných aktív, z ktorých sa veľmi často používajú práve dlhopisy. Našou úlohou je teda zostaviť dlhopisové portfólio, ktorého cash-flow (príjmy) bude kopírovať výšku našich záväzkov v budúcnosti.

V tejto časti ukážeme, ako také portfólio skonštruovať, teda rozhodnúť o tom, aké množstvo ktorých dlhopisov potrebujeme zaobstarať. Predstavíme viacero modelov, pričom postupne prejdeme od tých najjednoduchších k zložitejším. Zároveň bude možné vidieť, že všetky vedú k úlohám lineárneho programovania, ktoré následne budeme riešiť metódami opísanými v kapitole 2.

3.1 Deterministický model s minimalizáciou vstupných nákladov

Najskôr rozoberieme deterministický prípad, v ktorom sú záväzky aj aktíva vopred známe. Predtým však ešte uvedieme označenie, ktoré budeme v následných formuláciách používať. Označme:

$T = \{0, ..., m\}$ - množina časových okamihov t, ktoré sú udávané v rokoch, pričom $t = 0$ značí súčasnosť a $t = m$ budúci horizont

$U = \{0, ..., n\}$ - množina použiteľných dlhopisov (1-ročné, 2-ročné, ...)

L_t - výška záväzku splatného v čase t, pričom $t \geq 1$

$F_{i,t}$ - cash-flow (príjem) z dlhopisu i v čase t

Treba ešte dodať, že veľkosť časového okamihu (periódy) je určená intenzitou s akou dlhopisy vyplácajú kupóny. To znamená, že ak z dlhopisu plynie kupón každých 6 mesiacov, tak potom bude veľkosť jednej periódy rovná polovici roka. Cash-flow $F_{i,t}$ môže byť kladný aj záporný, pričom znamienko závisí od toho, aký je smer toku peňazí. Je zaužívané, že kladný cash-flow značí prichádzajúce peniaze a záporný cash-flow reprezentuje odchádzajúce peniaze. Ako príklad uvedieme 5-ročný kupónový dlhopis. Na začiatku je potrebné dlhopis obstarať, t.j. máme záporný cash-flow vo výške $F_{i,0} = -P_i$, teda je rovný mínus cene dlhopisu, v ďalších rokoch $t = 1, ..., 4$ máme kladný cash-flow vo výške kupónu $F_{i,t} = c_i$ a v poslednom piatom roku dostaneme navyše aj nominál (face value), $F_{i,5} = c_i + FV$.

Prvý uvádzaný model je deterministický, typu buy&hold. To znamená, že na začiatku v čase $t = 0$ skonštruujeme portfólio, teda obstaráme požadovaný počet daných typov dlhopisov, ktoré potom následne máme počas celého sledovaného obdobia v držbe. Tento model budeme podľa Nielsena[1] nazývať *"Cash-Flow Matching Model"* (označíme ho ako CFM-1).

$$\min \lambda$$

pri ohraničeniach:

$$\sum_{i\in U} F_{i,0}\, x_i + \lambda \geq 0,$$

$$\sum_{i\in U} F_{i,t}\, x_i \geq L_t, \quad t \geq 1,$$

$$x_i \geq 0, \quad i \in U.$$

Vektor x vyjadruje množstvá konkrétnych dlhopisov, ktoré treba obstarať, λ vyjadruje vstupné náklady na obstaranie portfólia. Jedná sa o model s minimalizáciou vstupných nákladov. Druhé ohraničenie značí, že v každom časovom okamihu dosahuje cash-flow z portfólia aspoň výšku záväzku, ktorý treba v danom okamihu splatiť.

Podľa tohto modelu sa záväzok splatný v čase t zaistí kúpou dlhopisu v čase 0, ktorý v t prináša potrebný cash-flow. To by však bolo možné urobiť aj iným spôsobom. V skutočnosti môže byť výhodnejšie kúpiť dlhopis, ktorý maturuje skôr a z neho získané peniaze uložiť do banky, pričom úročením získame požadovanú hodnotu. Druhou možnosťou je požičanie peňazí z banky oproti cenine (dlhopisu), ktorá maturuje v budúcnosti. Toto investovanie a požičiavanie sa dá veľmi jednoducho modelovať a je zahrnuté aj v ďalšom modeli, ktorý podľa Nielsena[1] budeme nazývať *"Cash-Flow Matching with reinvesting and borrowing"* (ozn. CFM-2).

$$\min \lambda$$

pri ohraničeniach:

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

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

$$b_m = 0,$$

$$x_i, r_i, b_i \geq 0, \quad i \in U.$$

Premenné r_t, resp. b_t vyjadrujú objem reinvestovaných resp. požičaných finančných prostriedkov z periódy t do $t+1$. Návratnosť z investície za 1 periódu je označená ako ρ_t a úrok na požičiavanie ako β_t. Požičiavanie je v poslednej perióde zakázané, čo vyjadruje ohraničenie $b_m = 0$. Premenná r_m značí v tomto prípade akýsi prebytok na konci. Ohraničenia sa nám oproti prvému modelu zmenili na rovnosti. Je to zapríčinené tým, že zvyšné finančné prostriedky na konci každej periódy je výhodnejšie investovať do banky, než ich mať v držbe ako hotovosť.

3.2 Deterministický model s maximalizáciou výnosu

V predchádzajúcich dvoch modeloch minimalizujeme vstupné náklady. V skutočnosti je však pre väčšinu investorov oveľa dôležitejšie dosiahnutie najvyššieho možného zisku. Touto úvahou sa dostávame k nasledovnému modelu, ktorý Nielsen[1] nazýva *"Maximizing horizon position"* (ozn. MAX).

$$\max h$$

pri ohraničeniach:

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

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

$$b_m = 0,$$

$$r_m = h,$$

$$x_i, r_i, b_i \geq 0, \quad i \in U.$$

Vidíme, že účelová funkcia maximalizuje konečnú hotovostnú pozíciu. Zároveň nám však pribudla konštanta B. Tá vyjadruje rozpočet, ktorý máme k dispozícii na začiatku na obstaranie portfólia.

3.3 Stochastický variant MAX modelu

Doteraz sme predpokladali, že poznáme všetky budúce dáta. V skutočnosti sú nám na začiatku známe len rozpočet, ceny a kupóny dlhopisov. Úroky, súčasné hodnoty dlhopisov na konci sledovaného obdobia ani budúce záväzky nepoznáme. Týmto sa dostávame od deterministických modelov k tzv. stochastickým. V týchto je zahrnutá neistota ohľadom budúcnosti (t.j. fakt, že nepoznáme kompletné dáta) a je vyjadrená pomocou stochastických premenných. Tie označíme znakom $\tilde{}$.

$$\max \tilde{h}$$

pri ohraničeniach:

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

$$\sum_{i\in U} \tilde{F}_{i,t}\, x_i + (1+\tilde{\rho}_t)\tilde{r}_{t-1} + \tilde{b}_t = \tilde{L}_t + \tilde{r}_t + (1+\tilde{\beta}_t)\tilde{b}_{t-1}, \quad t \geq 1,$$

$$\tilde{b}_m = 0,$$

$$\tilde{r}_m = \tilde{h},$$

$$x_i, \tilde{r}_i, \tilde{b}_i \geq 0, \quad i \in U.$$

Takto naformulovaný model je však príliš všeobecný. Navyše, ak by sme predpokladali, že stochastické premenné majú spojité rozdelenie pravdepodobnosti, tak sa daná úloha stáva veľmi ťažko riešiteľnou. Preto budeme predpokladať diskrétne rozdelenia pravdepodobnosti (tie budú vyjadrené množinou možných scenárov ich vývoja). Tento predpoklad sa dá uznať, keďže vieme spojité rozdelenia dobre aproximovať diskrétnymi. Odteraz budeme mať k dispozícii dáta o možnom vývoji premenných $F_{i,t}$, L_t, ρ_t, β_t pre každý scenár s z množiny všetkých scenárov S, pričom každý scenár má priradenú pravdepodobnosť p_s. Dostávame sa tak k modelu, ktorý Nielsen[1] pomenúva ako *"Maximizing expected horizon return"* (ozn. S-MAX):

$$\max \sum_{s\in S} p_s h^s$$

pri ohraničeniach:

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

$$\sum_{i\in U} F^s_{i,t} x_i + (1+\rho^s_t) r^s_{t-1} + b^s_t = L^s_t + r^s_t + (1+\beta^s_t) b^s_{t-1}, \quad t \geq 1, s \in S$$

$$b^s_m = 0,$$

$$r^s_m = h^s,$$

$$x_i, r^s_i, b^s_i \geq 0, \quad i \in U, s \in S.$$

3.4 Dvojfázový stochastický model

Všetky doteraz spomenuté modely boli statického charakteru, prezentovali tzv. buy&hold situáciu. To znamená, že na začiatku nakúpime portfólio, ktoré máme až do konca v držbe, pričom ho nijakým spôsobom počas sledovaného obdobia neupravujeme (nerebalancujeme). Drvivá väčšina investorov však svoje portfóliá pravidelne upravuje v závislosti od vyvíjajúcej sa situácie budúcich záväzkov a úrokových mier, ich neistota ohľadom budúcnosti sa postupne mení. Nasledujúci model bude tým najjednoduchším dynamickým modelom, konkrétne dvojfázovým. Investor tu po prvej časovej perióde rebalancuje svoje portfólio, pričom ho však už do konca nemení. Týmto spôsobom môžeme lepšie modelovať reálnu situáciu. Realistickosť by sa následne dala zvýšiť pridaním množstva okamihov, v ktorých budeme portfólio prerovnávať, čo by nás priviedlo k multiperiodickým modelom. Tými sa však v našej práci zaoberať nebudeme. Nasledovný model Nielsen[1] značí ako *"Two-Stage Stochastic Program"* (ozn. 2S-SP).

$$\max \sum_{s \in S} p_s h^s$$

pri ohraničeniach:

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

$$\sum_{i \in U} F^s_{i,t} y^s_i + (1+\rho^s_t) r^s_{t-1} + b^s_t = L^s_t + r^s_t + (1+\beta^s_t) b^s_{t-1}, \quad t \geq 1, s \in S$$

$$\sum_{i \in U} {}^{sell}P^s_i x_i = \sum_{i \in U} {}^{buy}P^s_i y^s_i$$

$$b^s_m = 0,$$

$$r^s_m = h^s,$$

$$x_i, y^s_i, r^s_i, b^s_i \geq 0, \quad i \in U, s \in S,$$

Premenné x_i vyjadrujú množstvá dlhopisov pre prvú fázu, y^s_i pre druhú fázu. Vidíme, že tie závisia od vybraného scenára. Ak by tomu tak nebolo, v druhej fáze by sa vybralo rovnaké portfólio ako na začiatku a o rebalancovaní by nebolo možné hovoriť. Premenné ${}^{sell}P^s_i$, resp. ${}^{buy}P^s_i$ označujú cenu, za ktorú môžeme dlhopis i predať resp. obstarať, ak sa uberáme podľa scenára s.

3.5 Generovanie scenárov

Ako si môžeme všimnúť, výber množiny scenárov má veľký vplyv na optimálne riešenie úlohy. Preto musí množina S obsahovať "normálne", ale zároveň aj extrémne prípady (na tie však nemôže brať prílišný ohľad, keďže pravdepodobnosť ich nastania je malá). Scenáre budeme voliť viacerými spôsobmi. Prvým je tzv. New York 7. Jedná sa o nasledovných 7 scenárov vývoja úrokových mier počas 10 rokov, pričom každá finančná inštitúcia v štáte New York musí každoročne preukázať solventnosť, ak by sa situácia vyvíjala podľa ľubovoľného z týchto scenárov.

NY - 7:

- úrokové miery sa nezmenia,
- úrokové miery vzrastú o 50 bb ročne počas 10 rokov,
- úrokové miery poklesnú o 50 bb ročne počas 10 rokov,
- úrokové miery vzrastú o 100 bb ročne počas 5 rokov, a potom poklesnú o 100 bb ročne počas 5 rokov,
- úrokové miery poklesnú o 100 bb ročne počas 5 rokov, a potom vzrastú o 100 bb ročne počas 5 rokov,
- úrokové miery vzrastú o 300 bb a potom sa už nezmenia,
- úrokové miery poklesnú o 300 bb a potom sa už nezmenia.

Dodajme, že bb označuje bázický bod a ako príklad uveďme, že nárast o 50 bb znamená nárast o 0.5%. Ďalšou možnosťou, ktorou budeme generovať vývoj úrokových mier, je použitie binomického stromu. Začneme v časovom okamihu 0, pričom predpokladáme, že pravdepodobnosť nárastu a poklesu úrokovej miery je rovnaká. To nám zabezpečí náhodný generátor, pomocou ktorého rozhodneme, ktorou z dvoch možných vetiev sa dostaneme do časového okamihu 1. Teda, či zaznamenáme pokles alebo nárast úrokovej miery. Takto pokračujeme ďalej, až pokým sa nedostaneme na koniec sledovaného obdobia a získame tým jeden kompletný scenár. Postup opakujeme dovtedy, kým nezískame stanovený počet scenárov vývoja úrokových mier. Táto možnosť je ľahko použiteľná, ale ponúka iba obmedzené množstvo prípadov. Vylepšením môže byť zvýšenie počtu vetiev pri rozhodovaní v každom uzle, čím sa dostávame k trinomickému a multinomickým stromom.

3.6 Rozšírenie modelov

Pri investovaní do aktív (akcií, dlhopisov, ...) musíme počítať aj s transakčnými nákladmi, medzi ktoré patria napríklad brokerské poplatky alebo rozdiel medzi ponukou a dopytom (tzv. bid-ask spread). Dôvodom pre ich zahrnutie do modelu je skutočnosť, že investori v reálnom živote nedržia po celý čas rovnaké portfólio. Na meniacu sa situáciu na trhu reagujú úpravami portfólia, jeho rebalancovaním (presúvajú finančné prostriedky z menej výhodných na výhodnejšie aktíva). Vynechaním transakčných nákladov by mohlo dôjsť k rozdielu oproti skutočnosti v prípade, kedy by investor kompletne odpredal celé portfólio a obstaral si úplne nové. Existujú dva typy transakčných nákladov, a to fixné a variabilné. Fixné náklady sú uvalené na každé obchodované aktívum a modelujeme ich zavedením binárnych premenných: $z_i = 1$, ak $x_i > 0$, inak $z_i = 0$ (to dosiahneme pridaním nerovností $x_i \leq Mz_i$, kde M je dostatočne veľké číslo) a do účelovej funkcie pridáme výraz: $F \sum_{i \in U} z_i$, kde F predstavuje výšku fixných nákladov. Takto získaná úloha obsahuje celočíselné premenné, a preto je veľmi ťažko riešiteľná. Tým pádom sa v praxi používajú variabilné náklady, ktoré sú ľahko modelovateľné a nezvyšujú zložitosť ani obtiažnosť modelu. Uplatňujú sa na jednotku nákladov a možno ich jednoducho zahrnúť pridaním k obstarávacím nákladov jednotlivých aktív, v našom prípade odpočítaním z $F_{i,0}$. Tento spôsob môžeme uplatniť v CFM modeloch. V modeloch typu MAX sa toto rozšírenie robí podobne. Jediný rozdiel je v tom, že ak uvažujeme fixné transakčné náklady, nemeníme účelovú funkciu, ale upravíme dostupný rozpočet B. V našich konkrétnych experimentoch nebudeme brať transakčné náklady do úvahy.

Kapitola 4

Numerické experimenty

4.1 Transformácia ekonomických modelov

Všetky uvažované modely sú v podstate úlohami lineárneho programovania. Cieľom tejto kapitoly bude ich prepísať do tvaru štandardnej úlohy lineáneho programovania (skrátene ŠÚLP), teda na tvar:

$$min\left\{c^T x | Ax = b, x \geq 0\right\}.$$

Budeme predpokladať, že máme k dispozícii n možných dlhopisov, k scenárov vývoja stochastických premenných a časový horizont úlohy je m rokov.

4.1.1 S-MAX model ako ŠÚLP

S-MAX sa dá prepísať na tvar:

$$min\left\{c^T x | Ax = b, x \geq 0\right\},$$

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

$$x = (x_1 \; .. \; x_n \; r_0 \; b_0 \; r_1^1 \; .. \; r_m^1 \; .. \; r_1^k \; .. \; r_m^k \; b_1^1 \; .. \; b_m^1 \; .. \; b_1^k \; .. \; b_m^k \; h_1 \; .. \; h_k)^T,$$

$$c = (0 \; ... \; 0 \; -p_1 \; ... \; -p_k)^T,$$

$$b = (-B \; L_1^1 \; ... \; L_m^1 \; ... \; L_1^k \; ... \; L_m^k \; 0 \; ... \; 0)^T$$

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

$$
A = \left(\begin{array}{cccccccccccccc}
F^0 & C^0 & & & & & & & & & & & & \\
F^1 & C^1 & D^1 & & & & G^1 & & & & & & & \\
F^2 & C^2 & & D^2 & & & & G^2 & & & & & & \\
\vdots & \vdots & & & \ddots & & & & \ddots & & & & & \\
F^k & C^k & & & & D^k & & & & G^k & & & & \\
 & & E & & & & & & & & -1 & & & \\
 & & & E & & & & & & & & -1 & & \\
 & & & & \ddots & & & & & & & & \ddots & \\
 & & & & & E & & & & & & & & -1 \\
 & & & & & & E & & & & & & & \\
 & & & & & & & E & & & & & & \\
 & & & & & & & & \ddots & & & & & \\
 & & & & & & & & & E & & & &
\end{array}\right),
$$

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 v tvare:

$$
F^0 = \begin{pmatrix} F_{1,0} & F_{2,0} & \dots & F_{n,0} \end{pmatrix},
$$

$$
C^0 = \begin{pmatrix} -1 & 1 \end{pmatrix},
$$

$$
E = \begin{pmatrix} 0 & \dots & 0 & 1 \end{pmatrix},
$$

$$
F^i = \begin{pmatrix} F^i_{1,1} & F^i_{2,1} & \dots & F^i_{n,1} \\ F^i_{1,2} & F^i_{2,2} & \dots & F^i_{n,2} \\ \vdots & \vdots & \ddots & \vdots \\ F^i_{1,m} & F^i_{2,m} & \dots & F^i_{n,m} \end{pmatrix}, \quad i = 1, ..., k,
$$

$$
C^i = \begin{pmatrix} (1+\rho^i_1) & -(1+\beta^i_1) \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \end{pmatrix}, \quad i = 1, ..., k,
$$

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

$$G^i = \begin{pmatrix} 1 & & & & \\ -(1+\beta_2^i) & 1 & & & \\ & -(1+\beta_3^i) & 1 & & \\ & & \ddots & \ddots & \\ & & & -(1+\beta_m^i) & 1 \end{pmatrix}, \quad i = 1, ..., k.$$

4.1.2 2S-SP model ako ŠÚLP

Podobným spôsobom, ako v predchádzajúcej časti, prepíšeme model 2S-SP do tvaru:

$$min\left\{c^T x | Ax = b, x \geq 0\right\}.$$

V tomto prípade dostávame, že $x, c \in R^{n+2+nk+2mk+k}$, $b \in R^{1+mk+3k}$ sú vektory v tvare:

$$x = (x_1 \,..\, x_n \; y_1^1 \,..\, y_n^1 \,..\, y_1^k \,..\, y_n^k \; r_0 \; b_0 \; r_1^1 \,..\, r_m^1 \,..\, r_1^k \,..\, r_m^k \; b_1^1 \,..\, b_m^1 \,..\, b_1^k \,..\, b_m^k \; h_1 \,..\, h_k)^T,$$

$$c = (0 \,...\, 0 \; -p_1 \,...\, -p_k)^T,$$

$$b = (-B \; L_1^1 \,...\, L_m^1 \,...\, L_1^k \,...\, L_m^k \; 0 \,...\, 0)^T$$

a $A \in R^{(1+mk+3k)\times(n+2+nk+2mk+k)}$ je bloková matica v tvare:

$$A = \left(\begin{array}{cccccccccccccc}
F^0 & & & & C^0 & & & & & & & & & \\
 & F^1 & & & C^1 & D^1 & & & G^1 & & & & & \\
 & & \ddots & & \vdots & & \ddots & & & \ddots & & & & \\
 & & & F^k & C^k & & & D^k & & & G^k & & & \\
 & & & & & E & & & & & & -1 & & \\
 & & & & & & \ddots & & & & & & \ddots & \\
 & & & & & & & E & & & & & & -1 \\
 & & & & & & & & E & & & & & \\
 & & & & & & & & & \ddots & & & & \\
 & & & & & & & & & & E & & & \\
S^1 & -B^1 & & & & & & & & & & & & \\
\vdots & & \ddots & & & & & & & & & & & \\
S^k & & & -B^k & & & & & & & & & &
\end{array}\right),$$

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 v tvare:

$F^0 = (F_{1,0}\ F_{2,0}\ ...\ F_{n,0})$,
$C^0 = (-1\ 1)$,
$E = (0\ ...\ 0\ 1)$,
$S^i = ({}^{sell}P^i_1\ {}^{sell}P^i_2\ ...\ {}^{sell}P^i_n), \quad i = 1, ..., k$
$B^i = ({}^{buy}P^i_1\ {}^{buy}P^i_2\ ...\ {}^{buy}P^i_n), \quad i = 1, ..., k$

$$F^i = \begin{pmatrix} F^i_{1,1} & F^i_{2,1} & \dots & F^i_{n,1} \\ F^i_{1,2} & F^i_{2,2} & \dots & F^i_{n,2} \\ \vdots & \vdots & \ddots & \vdots \\ F^i_{1,m} & F^i_{2,m} & \dots & F^i_{n,m} \end{pmatrix}, \quad i = 1, ..., k,$$

$$C^i = \begin{pmatrix} (1+\rho^i_1) & -(1+\beta^i_1) \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \end{pmatrix}, \quad i = 1, ..., k,$$

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

$$G^i = \begin{pmatrix} 1 & & & & \\ -(1+\beta_2^i) & 1 & & & \\ & -(1+\beta_3^i) & 1 & & \\ & & \ddots & \ddots & \\ & & & -(1+\beta_m^i) & 1 \end{pmatrix}, \quad i = 1, ..., k.$$

4.2 Výsledky experimentov

Vo všetkých modeloch budeme uvažovať, že máme k dispozícii 7 rôznych dlhopisov, konkrétne 1-ročné, 2-ročné, 3-ročné, 4-ročné, 5-ročné, 7-ročné a 10-ročné. Všetky vyplácajú kupón raz ročne a sú typu par-bond, t.j. ich nominálna hodnota je rovnaká ako ich nákupná cena, ktorá bude 1 peňažná jednotka. Dostupný rozpočet bude vo výške B=50000 peňažných jednotiek a veľkosť záväzku splatného v každom období bude 6000 peňažných jednotiek. Veľkosť časového obdobia budeme uvažovať 10 rokov, pričom jedna perióda bude rovná jednému roku. Scenáre vývoja úrokových mier generujeme postupne podľa NY-7, binomického, trinomického a dvojice multinomických stromov. V binomickom mohla v každom časovom okamihu s pravdepodobnosťou 50% vzrásť úroková miera o 0,5% a s rovnakou pravdepodobnosťou klesnúť o 0,5%. V trinomickom strome sme s rovnakými pravdepodobnosťami nastatia vybrali možnosti nárastu o 0,5%, poklesu o 0,5% a možnosť, že úrok zostane na pôvodnej úrovni. V multinomických stromoch sme v prvom prípade používali 11 vetiev, pričom zmena úroku bola v rozsahu od poklesu o 1% po nárast o 1%, v druhom prípade sme uvažovali 21 vetiev a rozsah bol od poklesu o 1,5% po nárast o 1,5%. Tu však hrozí riziko, že by sme sa pri takomto náhodnom pohybe mohli dostať s úrokovou mierou na nulovú hladinu prípadne aj do záporných čísiel. To je však v praxi nemožné, a preto sme stanovili dolnú hranicu pre úrokovú mieru na úrovni 0,25%. Na získanie scenárov sme použili náhodný generátor naprogramovaný v jazyku C++, pomocou ktorého sme vygenerovali úlohy zahŕňajúce 1000 scenárov pre binomický a trinomický strom. Pre dvojicu multinomických stromov sme vytvorili úlohy obsahujúce až 2000 scenárov. Výsledky experimentov sú zhrnuté v nasledujúcich tabuľkách.

Scenáre		Rozmer úlohy	Počet iterácií		
Generovanie	**Počet**		**HSD**	**MPK**	**Simplex**
NY-7	7	156x85	32	21	228
binomický	15	324x181	32	21	491
	30	639x361	98	21	925
	50	1059x601	98	23	1524
	75	1584x901	164	30	2316
	100	2109x1201	74	32	3045
	150	3159x1801	94	31	4506
	250	5259x3001	76	216	NA
	400	8409x4801	94	228	NA
	500	10509x6001	72	22	NA
	650	13659x7801	78	X	NA
	750	15759x9001	188	X	NA
	850	17859x10201	124	X	NA
	1000	21009x12001	X	X	NA
trinomický	15	324x181	34	20	474
	30	639x361	50	20	935
	50	1059x601	62	22	1567
	75	1584x901	46	21	2396
	100	2109x1201	90	21	3184
	150	3159x1801	106	24	4769
	250	5259x3001	120	446	NA
	400	8409x4801	56	244	NA
	500	10509x6001	92	X	NA
	650	13659x7801	142	416	NA
	750	15759x9001	104	301	NA
	850	17859x10201	X	X	NA
	1000	21009x12001	84	X	NA

Tabuľka 4.1: Porovnanie homogénnej samoduálnej (HSD), simplexovej metódy a Mehrotrovho prediktor-korektor algoritmu (MPK) v prípade modelu S-MAX z hľadiska počtu iterácií (NY-7, binomický a trinomický strom)

NA - softvér nedokázal požadovanú úlohu vypočítať
X - softvér pri riešení úlohy prekročil iteračný limit

Scenáre		Rozmer úlohy	Počet iterácií		
Generovanie	**Počet**	**Rozmer úlohy**	**HSD**	**MPK**	**Simplex**
multinomický (11 vetiev)	15	324x181	110	21	472
	30	639x361	42	21	928
	50	1059x601	42	26	1626
	75	1584x901	52	23	2364
	100	2109x1201	52	23	3249
	150	3159x1801	86	21	4882
	250	5259x3001	116	89	NA
	400	8409x4801	114	29	NA
	500	10509x6001	130	25	NA
	650	13659x7801	56	X	NA
	750	15759x9001	94	X	462
	850	17859x10201	X	24	NA
	1000	21009x12001	80	24	NA
	2000	42009x24001	X	487	NA
multinomický (21 vetiev)	15	324x181	38	21	475
	30	639x361	48	23	936
	50	1059x601	78	23	1581
	75	1584x901	120	23	2400
	100	2109x1201	100	22	3130
	150	3159x1801	74	23	4871
	250	5259x3001	176	280	NA
	400	8409x4801	116	25	NA
	500	10509x6001	86	X	NA
	650	13659x7801	X	261	NA
	750	15759x9001	X	X	NA
	850	17859x10201	X	471	NA
	1000	21009x12001	X	98	NA
	2000	42009X24001	X	487	NA

Tabuľka 4.2: Porovnanie homogénnej samoduálnej (HSD), simplexovej metódy a Mehrotrovho prediktor-korektor algoritmu (MPK) v prípade modelu S-MAX z hľadiska počtu iterácií (multinomické stromy)

Scenáre		Rozmer úlohy	Počet iterácií		
Generovanie	**Počet**		**HSD**	**MPK**	**Simplex**
NY-7	7	205x92	34	20	228
binomický	15	429x196	36	18	502
	30	849x391	42	17	1014
	50	1409x651	32	18	1719
	75	2109x976	46	17	2563
	100	2809x1301	42	17	3389
	150	4209x1951	48	18	5106
	250	7009x3251	46	18	NA
	400	11209x5201	54	19	NA
	500	14009x6501	46	51	NA
	650	18209x8451	94	18	NA
	750	21009x9751	98	18	NA
	850	23809x11051	92	18	NA
	1000	28009x13001	142	18	NA
trinomický	15	429x196	34	17	500
	30	849x391	32	17	1021
	50	1409x651	42	17	1703
	75	2109x976	48	17	2540
	100	2809x1301	44	18	3394
	150	4209x1951	44	17	5097
	250	7009x3251	64	17	NA
	400	11209x5201	48	18	NA
	500	14009x6501	46	18	NA
	650	18209x8451	42	18	NA
	750	21009x9751	88	21	NA
	850	23809x11051	150	18	NA
	1000	28009x13001	34	19	NA

Tabuľka 4.3: Porovnanie homogénnej samoduálnej (HSD), simplexovej metódy a Mehrotrovho prediktor-korektor algoritmu (MPK) v prípade modelu 2S-SP z hľadiska počtu iterácií (NY-7, binomický a trinomický strom)

Scenáre			Počet iterácií		
Generovanie	**Počet**	**Rozmer úlohy**	**HSD**	**MPK**	**Simplex**
multinomický (11 vetiev)	15	429x196	32	18	503
	30	849x391	36	18	1025
	50	1409x651	36	18	1731
	75	2109x976	36	18	2532
	100	2809x1301	50	17	3384
	150	4209x1951	40	18	5060
	250	7009x3251	46	18	NA
	400	11209x5201	58	19	NA
	500	14009x6501	56	129	NA
	650	18209x8451	108	229	NA
	750	21009x9751	44	24	NA
	850	23809x11051	X	19	NA
	1000	28009x13001	198	20	NA
	2000	56009x26001	180	20	NA
multinomický (21 vetiev)	15	429x196	36	19	505
	30	849x391	36	19	202
	50	1409x651	38	18	1696
	75	2109x976	36	18	2504
	100	2809x1301	40	20	3415
	150	4209x1951	41	19	4879
	250	7009x3251	56	21	NA
	400	11209x5201	49	21	NA
	500	14009x6501	48	20	NA
	650	18209x8451	87	98	NA
	750	21009x9751	X	114	NA
	850	23809x11051	63	30	NA
	1000	28009x13001	187	21	NA
	2000	56009x26001	X	25	NA

Tabuľka 4.4: Porovnanie homogénnej samoduálnej (HSD), simplexovej metódy a Mehrotrovho prediktor-korektor algoritmu (MPK) v prípade modelu 2S-SP z hľadiska počtu iterácií (multinomické stromy)

Môžeme z nich vypozorovať, že iteračná náročnosť simplexového algoritmu významne narastá so zvyšujúcim sa počtom uvažovaných scenárov. Pri počte vyššom ako 150 scenárov už softvér nebol schopný nájsť optimálne riešenie príslušnej úlohy, program v týchto prípadoch zlyhal. Ďalej si môžeme všimnúť, že pri použití metód vnútorného bodu bol nárast v počte iterácií výrazne nižší, v niektorých prípadoch sme dokonca so zvýšením počtu zahrnutých scenárov zaznamenali pokles v počte potrebných iterácií. Dôvodom môže byť napríklad fakt, že zmena množiny scenárov má podstatný vplyv na riešenie úlohy, aj jeho kvalitu. Preto je nevyhnutné brať do úvahy prevažne reálne možnosti vývoja stochastických premenných (v našom prípade úrokovej miery) a extrémnym možnostiam prideliť nižšiu pravdepodobnosť nastatia. Z výsledkov je badateľné, že Mehrotrov prediktor-korektor algoritmus je efektívnejší (z hľadiska počtu ietrácií) ako algoritmus homogénnej samoduálnej metódy, čo sme aj predpokladali. Opak sme zaznamenali iba v minimálnom počte prípadov. Jednou z vecí, ktorá nás zaujímala bolo aj časové porovnanie jednotlivých metód. To však naneštastie nebolo možné. Meranie časovej náročnosti umožňoval iba program, v ktorom bol implementovaný Mehrotrov prediktor-korektor algoritmus, zostávajúce dva túto možnosť neponúkali. Iteračne najnáročnejšie úlohy dokázali programy využívajúce metódy vnútorného bodu vyriešiť do šiestich minút.

Záver

V diplomovej práci sme sa zaoberali modernými metódami na riešenie úloh lineárneho programovania. Oblasť lineárneho programovania má široké uplatnenie a využíva sa aj vo finančnej sfére. Ako sa nám podarilo ukázať, niektoré modely znázorňujúce problém optimalizácie dlhopisového portfólia sa dajú zapísať v tvare úlohy lineárneho programovania. V práci sme využívali dve metódy vnútorného bodu, Mehrotrov prediktor-korektor algoritmus a homogénnu samoduálnu metódu, ktoré sme potom spoločne so simplexovým algoritmom testovali na nami vygenerovaných úlohách, ktorých rozmer bol relatívne veľký a dosahoval až 56009 premenných pri 26001 ohraničeniach.

Pri práci sme používali nekomerčný softvér, voľne dostupný na internete, a preto sme boli obmedzení na maximálny počet možných iterácií, ktorý sme v niekoľkých prípadoch dosiahli(pri homogénnej samoduálnej metóde to bolo 199 iterácií, pri Mehrotrovom prediktor-korektor algoritme 500 iterácií). Z podobného dôvodu sme nemohli presne určiť časovú náročnosť jednotlivých metód.

Podarilo sa nám však preukázať správnosť predpokladu, že simplexová metóda pri riešení úloh veľkých rozmerov nemôže s modernejšími metódami vnútorného bodu súperiť a jej efektivita je nízka. Taktiež sa ukázalo, že momentálne najpoužívanejší Mehrotrov prediktor-korektor algoritmus je efektívnejší (tak tomu bolo v experimentoch až na pár výnimiek) ako homogénna samoduálna metóda.

Prínosom diplomovej práce je vysvetlenie metód, ktoré riešia úlohy lineárneho programovania v polynomiálnom čase a otestovanie príslušného softvéru. Môžeme ešte dodať, že naformulované modely vieme ešte viac priblížiť realite. Vylepšením môže byť zahrnutie transakčných nákladov popísané v časti 3.6 alebo zvýšenie počtu okamihov, v ktorých je možné prerovnávanie portfólia.

Literatúra

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

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

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

[4] Vanderbei R.J.: *Softvér na optimalizáciu využívajúci simplexovú metódu*, http://www.sor.princeton.edu/~rvdb/LPbook/src/index.html.

[5] Vanderbei R.J., LOQO: *LOQO - Softvér na optimalizáciu využívajúci Mehrotrov prediktor-korektor algoritmus*,
http://www.princeton.edu/~rvdb/loqo/install.html.

[6] Vanderbei R.J., LOQO user's manual: *Používateľská príručka pre program LOQO*, http://www.princeton.edu/~rvdb/tex/loqo/loqo405.pdf.

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

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

[9] Koršňák Ľ.: *Metódy vnútorného bodu v lineárnych úlohách optimalizácie dlhopisového portfólia*, Diplomová práca, Bratislava, 2003.

[10] Plesník J., Dupačová J., Vlach M.: *Lineárne programovanie*, Alfa, Bratislava, 1990.

[11] Halická M.: *Prednášky z lineárneho programovania 2*, Fakulta Matematiky, Fyziky a Informatiky Univerzity Komenského, 2005.

[12] Wright S.J.: *Primal-Dual Interior-Point Methods*, Society for Industrial and Applied Mathematics, Philadelphia, 1997.