Porteføljeoptimering for danske realkreditlån

Kourosh Marjani Rasmussen (s972269)

12. marts 2004

Vejleder: Prof. Jens Clausen

Institut for matematisk modellering
Danmarks Tekniske Universitet

Forord

Dette eksamensprojekt er udfærdiget som afslutning på min uddannelse som civilingeniør med retningsbetegnelsen anvendt matematik ved Danmarks Tekniske Universitet. Projektet er blevet udarbejdet i perioden fra den 1. juli 2003 til den 12. marts 2004 under vejledning af Professor Jens Clausen, institut for matematisk modellering, DTU. Desuden har Svend Bondorf, kontorchef i kontor for finansielle analyser og Michael Carlsen, kontorchef for produktlaboratoriet hos Nykredit, været eksterne vejledere på projektet.

Formålet med projektet er at udvikle et matematisk analyseværktøj til rådgivning af realkreditkunder - låntagere - således at de, til at begynde med, vælger det optimale lån eller en optimal kombination af lån, og dernæst plejer låneporteføljen mest hensigtsmæssigt gennem låneperioden. Til det formål vil vi - læseren og jeg - betragte en sammenkobling af matematisk finansiering og operationsananlyse. Inden for matematisk finansiering møder vi lånrelaterede beregninger, diskret rentetræ udvikling og prisfastsættelse af obligationer og optioner på obligationer. Inden for operationsanalyse anvender vi multi–stadie heltal stokastisk optimering.

Vi vil ikke se en detaljeret gennemgang af de enkelte discipliner, der udformer projektet som en helhed. Vi vil dog danne os en intuitiv forståelse for de forskellige komponenter i projektet og se, hvordan komponenternes grænseflader møder hinanden. Uddybninger og eksempler kan ses mange steder i rapporten, når de er led i forståelsen af helheden.

Til brug for en hurtig og nem adgang til betydning af finansieringstekniske begreber har jeg udarbejdet en til dette projekt skræddersyet ordbog, som løbende kan konsulteres i appendix A. Alle ord, der er skrevet med *kursiv* i teksten, kan findes i ordbogen. Desuden bliver de tekniske ord beskrevet i selve teksten, når en manglende forståelse af begreberne skønnes for at være hæmmende for en flydende læsning.

Mit håb og min ambition er, at denne rapport kan danne grundlag for et videre arbejde, der resulterer i et nyt analyseværktæj for låneanbefalinger.

Lyngby, 12. marts 2004

Kourosh Marjani Rasmussen

Anerkendelse

Professor Jens Clausen har som hovedvejleder på mit eksamensprojekt vejledt mig med sine konstruktive kritik af de mange udkast, som han har læst og rettet gennem denne opgaveskrivning. Som min underviser i flere videregående kurser inden for operationsanalyse har han på bedste vis formidlet den viden, der har været nødvendig for udarbejdelsen af dette projekt. Han har med sit overblik og store erfaring med håndteringen af praktiske og tværfaglige projekter motiveret mig gennem projektets forløb, først med sin positive indstilling til ideen bag projektet og dernæst ved at guide mig gennem et resultatorienteret forløb.

Ligeledes har Svend Bondorf, kontorchef i kontor for finansielle analyser i Nykredit, lært mig de komplicerede finansielle begreber og beregningsmetoder, der er nødvendige for at beskrive låneuniverset matematisk. Han har med stor entusiasme fulgt projektforløbet, læst, rettet og kommenteret min rapport.

Michael Carlsen og Steen Bertelsen fra produktlaboratoriet i Nykredit har hjulpet mig med faglig sparring og estimering af kurspriser på konverterbare obligationer.

Min hustru, Anne Mette Rasmussen, har med stor omhu læst og bidraget til det sproglige i rapporten.

Jeg skylder en stor tak til alle disse personer, uden hvem færdiggørelsen af denne rapport ikke havde været mulig. Jeg er dybt taknemmelig for al den vejledning og støtte, de har givet mig.

Executive summary

The Danish mortgage loan system is among the most complex of its kind in the world. Both adjustable–rate and fixed–rate loans are available. It is also possible to mix loans. As for the fixed–rate loans there are both call and delivery embedded options, which give the mortgagor the possibility to refinance the loan in both directions. When the interest rates are low the call option will be used to obtain a new loan with less interest payment. When the interest rates are high the delivery option can be used to reduce the amount of outstanding debt. A mortgagor with an adjustable–rate loan can buy a series of caps to hedge against possible rate increase.

The complexity of the mortgage loan system makes it a non–trivial task to decide on an initial choice of mortgage loan portfolio and on finding a continuing plan to readjust the portfolio optimally. There exists as of today no functional optimization model to advise the individual mortgagor on his choice of loan.

In this report we will develop and solve a realistic model for the optimal debt management problem of the Danish private mortgagor. The thesis is inspired by the research work of Nielsen and Poulsen (N&P, [18]). Some aspects of N&P model have been reformulated and extra constraints have been added, making the model more realistic. Both market risk and shape risk have been explicitly taken into consideration. Fixed transaction costs have been modeled by use of binary variables.

An effective algorithm to price a series of caps on an adjustable–rate loan is outlined. Likewise ways to use the optimization model for risk analysis regarding the price of these caps have been explained. The multi–stage stochastic integer program describing the mortgagors problem has been implemented in GAMS. Scenario reduction algorithms (GAMS/SCENRED module) have been applied to the problem with promising results.

The results obtained in this project show that the average Danish mortgagor should take better advantage of his options for mixing loans in his loan portfolio. Likewise he should readjust the portfolio more often than is the case today. The developed model and software in this project can be used as an analytical tool to develop new loan products, where an optimal portfolio of mortgage loans is chosen initially and then it is readjusted according to an optimal strategy given the individual customer input.

Indhold

1 Indledning **1**

1.1 Problembeskrivelse 1

1.2 Opbygning af rapporten 2

2 En beskrivelse af lånprodukter **5**

2.1 Grundlæggende elementer 5

2.2 Låntyper 6

2.2.1 Fastforrentede obligationslån 6

2.2.2 Fastforrentede kontantlån 7

2.2.3 Rentetilpasningslån 8

2.3 Valg af lånevaluta 11

2.4 Renten på et realkreditlån 11

2.5 Omkostningerne for et realkreditlån 12

2.6 Nuværende praksis inden for låneanbefaling 13

3 De deskriptive modeller **15**

3.1 Obligationer og renter 16

3.2 Black, Derman & Toy modellen 18

3.2.1 Prisfastsættelse af obligationer 19

3.2.2 Tilpasning af BDT–rentemodel til markedsdata 21

3.3 Bjerksund og Stenslunds approksimerende algoritme til løsning af BDT–modellen 26

3.3.1 Det korte rentetræ 26

3.3.2 Markedsinformation 27

3.3.3 Kalibreringsmetoden 29

3.3.4 Pseudokoden 30
3.4 Prisfastsættelse af rentesikring på rentetilpasningslån 31
3.4.1 Notation 33
3.4.2 Beregning af prisen på rentesikring 33
3.4.3 Beregning af prisen på rentesikring med leveling 35
3.4.4 Risikovurdering og prisfastsættelse i praksis 39
3.5 Prisfastsættelse af inkonverterbare og konverterbare obligationer 40
3.6 Forbehold for brugen af en rentemodel i et stokastisk optimeringsproblem 42

4 Optimeringsmodellen 45

4.1 Beskrivelse af optimeringsproblemet 46
4.2 En to–stadie stokastisk heltals–model 47
4.3 N&P optimeringsmodellen 60
4.3.1 Den principielle balanceligning 60
4.3.2 Beregning af betalinger 61
4.3.3 Målfunktioner 62
4.3.4 kommentarer til N&P optimeringsmodellen 62
4.4 En multi–stadie stokastisk heltals–model 62
4.5 Løsningsmetoder og implementering 69
4.5.1 Scenario–reducerende algoritmer i stokastisk programmering 70
4.5.2 GAMS–formuleringen 72
4.5.3 GAMS/SCENRED–formuleringen 77

5 Tilføjelser til optimeringsmodellen 83

5.1 Udvidelser til modellen 83
5.1.1 Tilføjelse af nye typer af lån 84
5.1.2 En metode til risikovurdering af rentesikring 88
5.1.3 Førtidig indfrielse 90
5.2 Modellering af risiko 91
5.2.1 Nyttefunktion 91
5.2.2 Minimax kriteriet 93

5.2.3 Budget begrænsninger og straffunktion 94
5.2.4 Formuerisiko–aversitet 95
5.3 Tilføjelser til Nielsen & Poulsen modellen 97

6 Resultater 99
6.1 Datagrundlaget 100
6.2 Likviditetsrisikoneutrale modeller 102
6.2.1 Fastforrentede obligationslån 102
6.2.2 Fastforrentede obligationslån og F1 lån 105
6.2.3 Fastforrentede obligationslån, F1 lån og F1 lån med rentesikring 106
6.2.4 Hvorfor får vi mindre målfunktionsværdier med GAMS/SCENRED formuleringer? 107
6.3 Risikoaverse modeller med GAMS formuleringer 109
6.3.1 Minimax modellen 109
6.3.2 Nyttefunktionsmodellen med budgetbegrænsninger . . 110
6.3.3 Formuerisikoaversitet 112
6.4 Risikoaverse modeller med GAMS/SCENRED formuleringer . 113
6.4.1 Minimax modellen 115
6.4.2 Nyttefunktionsmodellen med budgetbegrænsninger . . 116
6.4.3 Formuerisikoaversitet 118
6.5 Afsluttende kommentarer 118

7 Konklusion 121
7.1 Rådgivning 122
7.2 Produktudvikling 122
7.3 Analyse af rentesikring 123
7.4 Scenarioreducering 124
7.5 Forslag til videre arbejde 125
7.5.1 LP relaksation 125
7.5.2 Heuristiske løsninger 125
7.5.3 Parallel programmering 126
7.6 Slutord 126
Litteraturliste 129

A Ordbog **131**

B Fakta om realkreditlån **135**

B.1 Prisblad realkreditlån i kr. 135

B.1.1 Nykredit Kernekunde 135

B.1.2 Opkrævning af gebyrer 135

B.1.3 Tinglysningsafgift 136

B.1.4 Kurtage og kursfradrag ved obligationshandel 137

B.1.5 Bidrag 137

C Implementering af de deskriptive modeller **139**

C.1 Brugervejledning til applikationen 139

C.2 Kilekode 140

C.2.1 BDT knappen 140

C.2.2 BDTbuilderen 140

C.2.3 BDT funktioner 143

C.2.4 Rentesikring pris knappen 144

C.2.5 Rentesikring funktioner 146

D Koden til generering af inputdata til GAMS **149**

D.1 VBA program til generering af GAMS tabeller 149

D.2 Hjælpefunktioner og klasser 150

D.2.1 Funktionerne 150

D.2.2 Klasserne 156

E Implementering af optimeringsmodellen **161**

E.1 Grundmodellen med risikoneutral målfunktion 161

E.1.1 GAMS formulering 161

E.1.2 GAMS/SCENRED formulering 162

E.2 Risikoaverse modeller 166

E.2.1 GAMS/formuleing 166

E.2.2 GAMS/SCENRED–formuleringer 168

F Matlab kode til generering af scenariotrægrafer **173**

F.1 Grafen for alle scenarier 173

F.2 Grafen for udvalgte scenarier 174

Kapitel 1

Indledning

Baggrunden for dette eksamensprojekt er kompleksiteten i det danske marked for huslån. Der udbydes både fast–forrentede lån (der kan op– og ned-konverteres) og tilpasningslån i DK eller Euro. Man kan også sikre sit tilpasningslån ved at sætte et loft for, hvor højt renten må stige ved at benytte sig af tillægsproduktet rentesikring.

Vi ønsker en matematisk model til optimal styring af huslån, som er realistisk og nøjagtig nok i detaljerne til at kunne bruges operationelt. Den kunne f.eks. bruges som det bagvedliggende beregningsgrundlag til et låneanbefalingsprodukt.

1.1 Problembeskrivelse

Købere af fast ejendom er normalt nødt til at låne penge for at finansiere deres køb. Dermed står de over for at skulle udvælge en blandt et antal konkurrerende finansieringsmuligheder.

For de fleste vil det i denne sammenhæng være fordelagtigt at optage lån i et realkreditinstitut med sikkerhed i den købte ejendom. Det er denne situation, der betragtes i denne rapport.

Realkreditinstitutterne tilbyder imidlertid forskellige låntyper, så låntagerne skal foretage nogle valg. Disse valg afhænger af flere faktorer:

1. **Lånprodukter**: hvilket valgmuligheder er der af udbudte lånprodukter på realkreditmarkedet og hvordan er disse produkter skruet sammen?

2. **Markedssituationen**: hvordan svinger markedsrenten i lånets levetid?

3. **Låntagers risikoprofil**: er låntageren risikoavers, risikoneutral eller risikovillig?

En forståelse af disse tre emner er nødvendig for at kunne fremstille en brugbar beslutningsmodel. Modellen skal således forsynes med information fra punkterne 1 og 2 overfor, koble disse informationer sammen og rådgive de 3 typer låntagere fra punkt 3 om deres optimale beslutning.

En sådan model har en naturlig interesse blandt låntagere, som gerne vil minimere deres låneomkostninger og have en større klarhed over deres valgs konsekvenser. Realkreditinstitutter har også interesse i den type model. De kan nemlig bruge modellen som et analytisk værktøj til rådgivning af kunderne om eventuelle låneomlægninger og låneanbefalinger. De kan også bruge modellen til at analysere mulighederne for at sammensætte helt nye produkter.

1.2 Opbygning af rapporten

Dette projekt kan bedst beskrives som en integrering af matematisk finansiering og operationsanalyse. Vi skal således bygge en operationsanalytisk optimeringsmodel baseret på en kompliceret finansieringsproblemstilling. Selvom fokus i denne rapport er rettet mod optimeringsmodellen, kan denne model ikke afspejle markedets virkelighed, uden vi først forstår og beskriver det bagvedliggende finansieringsproblem.

Sammenhængen mellem emnerne i de forskellige kapitler og afsnit i denne rapport er skitseret i diagram 1.1 (side 4). I resten af rapporten studerer vi detaljerne bag de enkelte komponenter af dette diagram.

I kapitel 2 betragter vi de væsentligste realkreditlånprodukter. Vi vil introducere de faste og variable omkostninger i forbindelse med optagelse af lån, og vi definerer rammerne for de lånspecifikke data, der skal anvendes i optimeringsmodellen.

Som kan ses i diagrammet, vil vi betragte 5 matematiske modeller og deres vekselvirkning. Vi har 4 deskriptive modeller og 1 optimeringsmodel. En deskriptiv model er en matematisk model, som udelukkende beskriver et systems tilstand. En optimeringsmodel er derimod en matematisk model, som bestemmer en optimal beslutning eller en optimal sekvens af beslutninger til løsning af et aktuelt problem. I særlige tilfælde, som for eksempel i låneanbefalingsproblemet, kan det aktuelle problem være beskrevet nøje ved hjælp af deskriptive modeller.

Vi retter særligt fokus mod den multi–stadie stokastiske binære optimeringsmodel. Men eftersom optimeringsmodellen er tæt knyttet til de underliggende deskriptive modeller, vil vi analysere de deskriptive modeller i et kapitel for sig, (3).

I kapitel 3 udvikler vi tre af de fire deskriptive modeller, nemlig BDT–modellen til estimering af fremtidens renter, en model til prisfastsættelse af inkonverterbare obligationer og en prisfastsættelsesmodel til rentesikring.

Vi vil ikke udvikle en model til prisfastsættelse af konverterbare obligationer, men bruge markedets empiriske metoder til at estimere disse priser, (Argumenter for dette diskuteres i kapitel 3.).

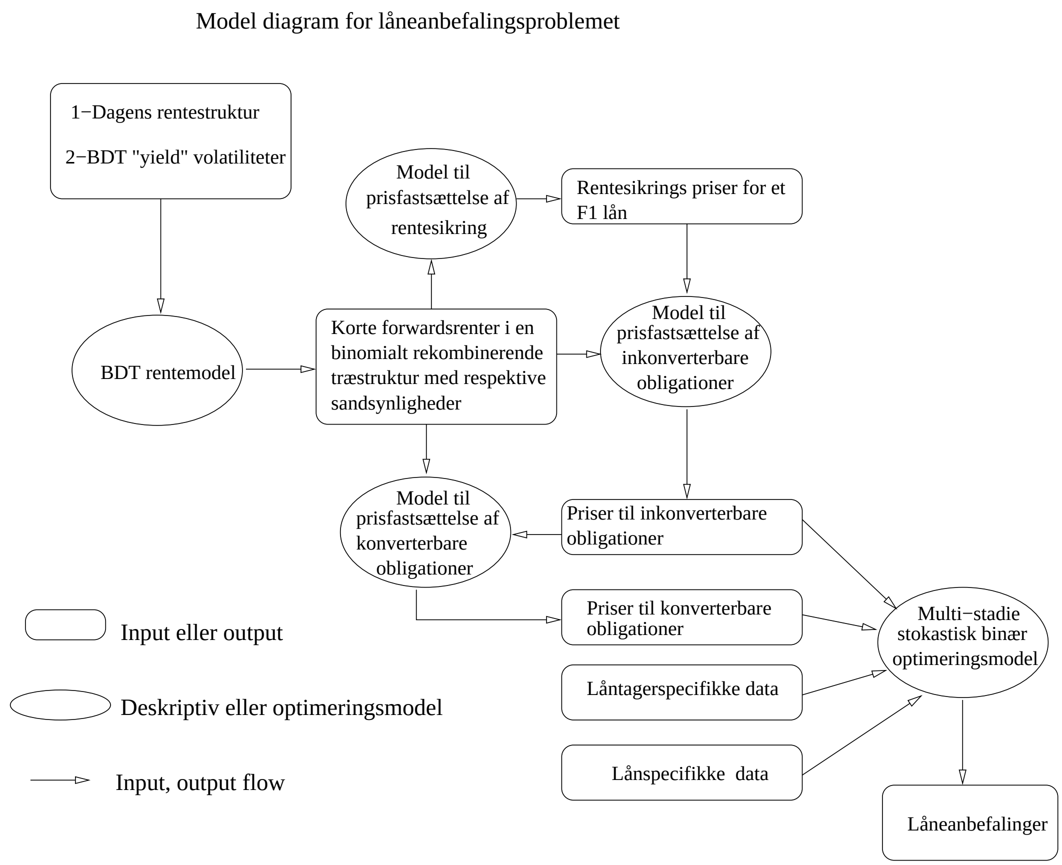
Stokastikken i låneanbefalingsproblemet kommer fra, at vi ikke kender alle omkostninger forbundet med forskellige typer lån, eftersom disse omkostninger er rentefølsomme. Det betyder, at vi først kender den totale låneomkostning med sikkerhed, når lånet er betalt. Vi kan dog estimere markedets rentebevægelse baseret på priserne på *nulkuponobligationer*[1], der også er rentefølsomme. Der findes en stor mængde kilder, der beskriver hvordan man kan estimere markedets forventning om den fremtidige rentebevægelse. (Se Hull, [14] og Björk, [5] for indføring i emnet og henvisning til flere artikler). Vi vil ikke her foretage en udtømmende analyse af rentemodeller. Vi vil dog i kapitel 3 danne os en intuitiv forståelse af diskrete rentemodeller og rentetrædannelsesprocesser ved at udvikle en rentemodel til brug for prisfastsættelsesmodellerne og den endelige optimeringsmodel.

Søren Nielsen og Rolf Poulsen har skrevet artiklen “A Two–Factor, Stochastic Programming Model of Danish Mortgage-Backed Securities” ($N\&P$, [18]), hvori de beskriver en model for bestemmelse af de optimale låneanbefalinger på det danske realkreditmarked. $N\&P$ modellen rådgiver den enkelte realkreditkunde om, hvilken type lån han oprindeligt skal vælge, og hvordan dette lån løbende skal plejes for at opnå den maksimale nytte.

Optimeringsdelen af $N\&P$ modellen har inspireret det meste af arbejdet i kapitel 4. $N\&P$ modellen er dog ikke fuldt realistisk og indeholder ikke samtlige relevante markedsdata. Vi vil præcisere disse forhold i kapitel 5 og analysere og implementere alternative modeller. Der bliver lagt stor vægt på praktisk anvendelighed af modellen. Derfor vil vi i kapitel 6 afprøve modellen ved at foretage et antal kørsler af den implementerede model. Vi vil se, om de fundne resultater stemmer overens med markedets praksis inden for låneanbefaling og undersøge, om robuste resultater kan opnås.

Rapporten afsluttes i kapitel 7 med konklusion og forslag til fremtidigt arbejde med dette emne.

[1] Se forklaringer på de finansielle udtryk brugt i denne rapport i appendix A (side 131)



Figur 1.1: Et generelt modeldiagram, der viser de forskellige delmodeller og deres respektive input og output.

Kapitel 2

En beskrivelse af lånprodukter

Vi vil i dette kapitel se på en beskrivelse af og kommentarer til lånrelaterede produkter og begreber.[1] Kapitlet er ment som en introduktion til realkreditlåneuniverset.[2] De væsentligste punkter, som en låntager burde vide inden han optager et lån er beskrevet i ord. Det er disse ord der blive omsat til en optimeringsmodel i kapitel 4. De fleste oplysninger i dette kapitel har en bred gyldighed blandt de danske realkreditinstitutter, (forkortelse: RI), men visse termer og satser er specifikke for Nykredit.

2.1 Grundlæggende elementer

- **Lån:** Et realkreditlån er baseret på udstedelse af obligationer og ydes mod pant i fast ejendom. Sikkerhed for lånet registreres ved, at der tinglyses et pantebrev. Lånene tilbydes med forskellig obligationsrente og løbetider. Der tilbydes både realkreditlån, hvor renten er fast i hele løbetiden (fastforrentede obligations- og kontantlån) og lån, hvor renten kan variere (Tilpasningslån). Der tilbydes realkreditlån med en løbetid på op til 30 år.

  I ejerboliger kan der tilbydes op til 80% af den værdi, som RI vurderer ejendommen til. Ved fritidshuse er grænsen 60%, og ved ubebyggede grunde er den 40%. RI fastsætter det endelige lånebeløb ud fra bl.a. ejendommmens størrelse, vedligeholdelsesstand, beliggenhed, omsættelighed, økonomi og en personlig kreditvurdering. Denne fremstilling omhandler kun realkreditdelen af lånet.

[1] De faktuelle oplysninger i dette kapitel stammer fra Nykredit hjemmeside ([17]) og bogen rentesregning (Bjarne A. Jensen, [1]).

[2] For en teknisk indføring i realkreditlån generelt se (Bjarne A. Jensen, [1]). For en teknisk introduktion til teorien bag rentetilpasningslån se (Svend Bodorf, [7]) og (Dan E. K. Sørensen, [23]).

- **Løbetid:** Et låns løbetid er det antal år, man har til at betale lånet tilbage. Man kan vælge mellem forskellige løbetider på et realkreditlån fra Nykredit. Normalt ligger løbetiden på 10, 15, 20 eller 30 år.

  Valget mellem de forskellige løbetider er først og fremmest et spørgsmål om likviditet. Man kan vælge at betale mindre ydelser over en længere periode eller betale højere ydelser i en kortere periode. Et lån med kortere løbetid betyder højere terminsydelser - til gengæld bliver lånet hurtigere afdraget, og man bliver hurtigere gældfri.

  Man kan selv vælge lånets løbetid. Man skal derfor gøre op med sig selv, hvilket lån, der passer bedst til ens situation. Man bør overveje, om man her og nu har brug for penge til andre formål - f.eks. til at nedbringe højt forrentet gæld, pensionsopsparing, boligindretning eller andet forbrug - eller om man eksempelvis ønsker lånet er tilbagebetalt, inden man går på pension.

  Her skal vi overveje om vi vil tage disse elementer med i beslutningsprocessen. Valget står mellem at betragte løbetiden som en konstant, der besluttes suverænt af låntageren eller en variabel, der afhænger af de nævnte forhold. Man kan med rimelighed sige, at løbetidsvalget i første omgang kan betragtes som en konstant. En tilføjelse til modellen vil dog være at introducere en løbetids variabel, hvis længde kan variere. Denne længde vil selvfølgelig også afhænge af låntagers budget. Så hvis vi erklærer løbetiden som variabel, skal vi tilføje passende budgetbegrænsninger, som bestemmer overgrænsen for hvor meget låntageren kan betale til en given termin.

- **Ydelsen før og efter skat:** Ydelse på et Realkreditlån afhænger af flere ting, blandt andet lånets størrelse, lånets sikkerhedsmæssige placering i ejendommen, den valgte rentesats og lånets løbetid. Ydelsen efter skat afhænger af ens skatteprocent og dermed af den skattemæssige værdi af ens rentefradrag. I dette projekt forudsættes, at alle låntageres indkomst er stor nok til at kunne udnytte denne rentefradrags mulighed. For tiden er denne rentefradrag på 32% af rentebetalingen.

2.2 Låntyper

Nykredit kan tilbyde 3 forskellige former for realkreditlån til private.

2.2.1 Fastforrentede obligationslån

Et obligationslån er baseret på udstedelse af obligationer, og lånet kan udbetales i obligationer eller ved, at RI sælger disse. I praksis sker salget af obligationerne, før man får lånet udbetalt kontant, idet RI normalt sælger obligationerne i forbindelse med lånets udbetaling.

Et obligationslån tilbydes typisk som annuitetslån. Det betyder, at ydelsen før skat (inkl. bidrag) tilnærmelsesvis er konstant i hele lånets løbetid. Ydelsen efter skat vil normalt være stigende, hvilket skyldes, at afdragenes andel af den samlede ydelse stiger, samtidig med at afdrag på realkreditlån ikke er fradragsberettiget.

Når obligationerne bliver solgt, sker det normalt til en kurs, der løbende fastsættes på Københavns Fondsbørs med fradrag af 0,10 kurspoint. Den kurs, som obligationerne bliver solgt til, ligger normalt under 100. Det betyder, at det beløb, man får udbetalt, er mindre end obligationsbeløbet (lånets hovedstol). Det er dog obligationsbeløbet, der er det faktiske lånebeløb, og det er dette beløb, man skal betale tilbage til RI. Obligationsbeløbet (hovedstolen) bliver - på grund af kurstabet - højere end det beløb, man får udbetalt i kontanter.

Det kontantbeløb, som obligationerne indbringer ved salget, kaldes obligationernes kursværdi. Det beløb, man får udbetalt ved optagelse af lånet, er kursværdien fratrukket omkostningerne ved optagelse af lånet. Forskellen mellem lånets hovedstol og obligationernes kursværdi udgør kurstabet. Kurstabet betaler man tilbage over hele lånets løbetid. Tilbagebetalingen sker som en del af afdragene på lånet.

Ved et obligationslån betales den procentvise tinglysningsafgift til staten af lånets hovedstol. Et procentvis bidrag betales til RI af lånets obligationsrestgæld.

Hvis kursen er over 100 på salgstidspunktet, skal man være opmærksom på, at kursgevinsten kan være skattepligtig.

2.2.2 Fastforrentede kontantlån

Ved et kontantlån kan man kun få lånebeløbet udbetalt i kontanter. RI skal derfor sælge obligationer for at skaffe det kontantbeløb, som låntageren skal låne og have udbetalt. Salget sker normalt til en kurs, der løbende fastsættes på Københavns Fondsbørs med fradrag af 0,10 kurspoint.

Hvis kursen på de obligationer, som RI sælger, er under 100, opstår der et kurstab. Dette kurstab skal tilbagebetales over hele lånets løbetid. Tilbagebetalingen af kurstabet sker ved, at man betaler en højere rente på lånet (kontantrenten).

Kontantrenten, der bliver beregnet af lånets kontantrestgæld, er derfor højere end den tilsvarende obligationsrente. Fordelen ved at optage et kontantlån er, at det opståede kurstab bliver omdannet til rente - og renter er fradragsberettigede. Fordelen ved et kontantlån skal dog holdes op mod den ulempe, at kursgevinster ved ekstraordinær indfrielse af kontantlån i mange tilfælde bliver skattepligtige.

Et kontantlån bliver altid tilbudt som et annuitetslån. Ved kontantlån betales den procentvise tinglysningsafgift til staten af det kontante lånebeløb, der svarer til lånets kursværdi. Bidraget betales af lånets kontantrestgæld.

2.2.3 Rentetilpasningslån

Realkreditlån er normalt kendt som lån med en fast rente og en fast ydelse, der ikke ændrer sig i lånets løbetid. Der kan dog også tilbydes en låntype med en helt anden rentemæssig fleksibilitet, nemlig rentetilpasningslån.

Den væsentligste forskel på et rentetilpasningslån og et almindeligt, fast forrentet realkreditlån er, at renten på et rentetilpasningslån kun er fast i en kortere periode, da lånet tilpasses markedsrenten med jævne mellemrum. Det betyder, at ydelsen på et rentetilpasningslån vil ændre sig, når lånet er tilpasset markedsrenten.

Nykredit kan tilbyde to forskellige typer rentetilpasningslån: F-lån og P-lån. På et F-lån fastlægges lånets rente for en kortere eller længere periode ad gangen. Det kan være perioder på 1, 2, 3, 5 eller 10 år.

Med et P-lån bliver renten delvis tilpasset én gang om året. Man vælger selv ved optagelse af lånet, hvor stor en procentdel af restgælden, der hvert år skal refinansieres. Procentdelen kan være 10, 20, 25, 33 eller 50%.

Der er mulighed for at gå tilbage fra et rentetilpasningslån til et traditionelt 20- eller 30-årigt obligationslån, hvis man på et tidspunkt skulle ønske det.

Sammenligning af rentetilpasningslån og obligationslån:
I tabel (2.1) kan man se en sammenligning af ydelserne for et 5% obligationslån i danske kroner og et rentetilpasningslån (F1) i danske kroner, begge med en løbetid på 30 år.[3]

Det ser umiddelbart ud til, at et rentetilpasningslån er billigere end et fastforrentet lån. Der er imidlertid en risiko for, at renterne stiger og at ren-

[3]Forudsætninger for beregningerne: Lånestørrelsen er angivet som det provenu, man modtager efter, at alle omkostninger til Nykredit og staten er betalt. Begge lån har 4 årlige terminer. Hele restgælden for et rentetilpasningslån F1 refinansieres hvert år, med første refinansiering ultimo 2003. Kunden er Kernekunde, bidrag er beregnet i belåningsintervallet 0-80%. Følgende omkostninger i forbindelse med låneoptagelsen er indeholdt i beregningen: tinglysningsafgift til staten (procentdel) på 1,5 % af hovedstolen, lånesagsgebyr, kurtage, tinglysningsafgift til staten (fast del) samt gebyr for tingbogsattest. Den anvendte skattesats er 33 %, hvilket svarer til gennemsnittet for alle kommuner og amter incl. kirkeskat. Beregningerne er baseret på Nykredits tilbudskurser, som er "Kurs alle handler"på Københavns Fondsbørs fratrukket kursskæring på 0,1 kurspoint. Kurserne for obligationslånet er fra børsdagen den 1. juni 2003. Kurserne for Tilpasningslånet er fra børsdagen den 1. juni 2003.

Lånetype	Obligationslån	Obligationslån	Tilpasningslån	Tilpasningslån
Provenu efter låneoptagelsen	1. års månedlige ydelse før skat	1. års månedlige ydelse efter skat	1. års månedlige ydelse før skat	1. års månedlige ydelse efter skat
100.000	636	469	476	386
200.000	1.221	904	910	743
500.000	3.024	2.238	2.253	1.839
1.000.000	6.025	4.459	4.488	3.663

Tabel 2.1: Sammenligninger af forskellige typer låns afdragsprofil.

tetilpasningslånet derved ender med at blive dyrere end det fastforrentede lån.

Fakta om rentetilpasningslån:
Løbetiden for et rentetilpasningslån kan vælges op til 30 år lige som for almindelige, fastforrentede realkreditlån.

Obligationerne bag et rentetilpasningslån har typisk en kortere løbetid end lånet. Derfor må RI sælge nogle nye obligationer for at kunne erstatte de gamle obligationer, der udløber. Lånet bliver med andre ord refinansieret. På den måde tilpasses renten på tilpasningslånet til den markedsrente, der gælder for obligationer med kort løbetid, den såkaldt "korte" rente.

Tilpasningslån med Nykredit Rentesikring:
Med Nykredit Rentesikring kan man lægge en øvre grænse for, hvor meget renten på ens tilpasningslån kan stige i forbindelse med de årlige refinansieringer. Hvis man ved optagelsen af et rentetilpasningslån vil udnytte den lavere korte rente, men samtidig ønsker et loft over, hvor meget renten kan stige på ens lån, bør man overveje rentesikring. Hvis man allerede har et rentetilpasningslån med fuld årlig refinansiering, kan dette også rentesikres.

Rentesikringen kan tegnes for op til 3 refinansieringer. Man kan vælge mellem forskellige øvre grænser for, hvor meget renten kan stige, de såkaldte sikringsrenter. Nykredit Rentesikring kan tegnes for rentetilpasningslån af typen F1 i både danske kroner og euro.

Man kan undre sig over, hvorfor det ikke er muligt at tegne en rentesikring for mere end 3 år. RI påtager sig en risiko i.f.m. salget af rentesikring. Jo længere sikringsperioden bliver, jo større risiko løber RI. Hvis RI kræver en alt for høj pris for rentesikring ville det opveje fordelen ved at optage en rentesikret rentetilpasningslån. I så fald vil det være mere fordelagtigt for låntageren at optage et fastforrentet lån.

Produktet rentesikring er stadig under videre udvikling hos RI, hvorfor et studie af rentesikring og alternative algoritmer til beregning af prisen for dette produkt, samt en metode til vurdering af risiko ønskes som en integreret del af dette projekt.[4]

[4] Vi vil udvikle to algoritmer til prisfastsættelsen af rentesikring i afsnit 3.4 (side 31).

Fordele ved et rentetilpasningslån:

1. Lavere renteudgifter og dermed lavere ydelser frem til første refinansiering af lånet, når renten på de "korte" obligationer ligger under renten på de "lange" obligationer på optagelsestidspunktet.
2. Tilpasning af renten på lånet, hvis der er sket et rentefald i forbindelse med refinansieringen, hvorved ydelserne falder.
3. I perioden op til en refinansiering har man mulighed for at låse kursen fast, hvis man tegner en fastkursaftale.

Ulemper ved et rentetilpasningslån:

1. Usikkerhed med hensyn til de fremtidige ydelser. Den aktuelle rente på refinansieringstidspunkterne er afgørende for ydelserne.
2. Usikkerhed ved en ekstraordinær indfrielse. Rentetilpasningslån er baseret på inkonverterbare obligationer, og det er obligationernes kurs på indfrielsestidspunktet, der bestemmer indfrielsesprisen. Traditionelle obligations- og kontantlån er derimod baseret på konverterbare obligationer, hvilket indebærer, at man altid har mulighed for at indfri lånene til kurs 100.
3. Beskatning af en eventuel kursgevinst ved indfrielse "før tid".
4. Tilpasning af renten på lånet, hvis der er sket en rentestigning i forbindelse med refinansieringen, hvorved ydelsen stiger.

Skal man vælge obligations-, kontant- eller rentetilpasningslån?
RIs anbefaling ved valg af realkreditlån afhænger af låntagers økonomiske situation og hans villighed til at tage økonomiske risici. I sidste ende er valget selvfølgelig låntagers eget. Det følgende er Nykredits retningslinjer når optagelse af realkreditlån skal anbefales.

- **Obligationslån:**
  Nykredit anbefaler fastforrentede obligationslån, når den pålydende rente giver en kurs tæt på kurs 100. Nykredit må ikke tilbyde obligations– og kontantlån, hvis "Kurs alle handler" på Københavns Fondsbørs er over kurs 100 på seneste børsdag, hvor obligationen er handlet. Når kursen er tæt på kurs 100, vil kurstabet ved låneoptagelsen blive relativt lille. Det betyder, at obligationsrestgælden bliver mindre, end hvis man i stedet for valgte et obligationslån med en lavere pålydende rente og dermed en lavere kurs.

Fordelen ved at vælge et obligationslån med en kurs tæt på 100 er, at restgælden kan blive billigere at indfri i forbindelse med et rentefald, hvor man vil nedkonvertere lånet. Omvendt er ydelserne på obligationslån og kontantlån med en kurs tæt på 100 normalt højere, end hvis man vælger en lavere pålydende rente og dermed en lavere kurs.

- **Kontantlån:**
  I stedet for et fastforrentet obligationslån kan man også vælge et kontantlån. Ydelserne på et kontantlån er typisk lidt lavere end på et tilsvarende obligationslån. Men i modsætning til obligationslån, kan man blive beskattet af en eventuel kursgevinst ved en ekstraordinær indfrielse af lånet. Dette kunne f.eks. være i forbindelse med en rentestigning, hvor man vil opkonvertere sit lån, mens renteniveauet er steget i forhold til det niveau, der var gældende, da lånet blev optaget.

- **Rentetilpasningslån:**
  Hvis Nykredit anbefaler rentetilpasningslån, er det med baggrund i, at ydelserne på et sådant lån er lavere end på de fastforrentede lån, samtidig med at Nykredit ikke forventer væsentlige rentestigninger i den nærmeste fremtid. Ulempen ved et rentetilpasningslån er bl.a. risikoen for ydelsesstigninger som følge af en rentestigning. Derfor anbefaler Nykredit heller ikke rentetilpasningslån uden at tage hensyn til låntagers økonomiske situation og hans risikovillighed.

2.3 Valg af lånevaluta

Nykredit tilbyder realkreditlån i enten kroner eller euro. Der tilbydes de samme realkreditlån i euro som i danske kroner. Dette gælder også rentetilpasningslån. Et lån i euro bliver udbetalt i euro og skal tilbagebetales i euro. Lånet er baseret på udstedelse af obligationer i euro. Lån i euro har, bortset fra valutaen, samme egenskaber som tilsvarende lån i danske kroner. Når et eurolån skal udbetales, kan provenuet naturligvis omveksles til danske kroner.

2.4 Renten på et realkreditlån

Man vil normalt kunne vælge mellem lån, der er baseret på obligationer med forskellige pålydende renteprocenter - f.eks. 5% eller 6%.

De forskellige renteprocenter har betydning for obligationernes kurs. Jo lavere renteprocent, jo lavere vil kursen blive på den pågældende obligation. En lavere kurs betyder samtidig, at man får et større kurstab, når obligationerne bliver solgt.

2.5 Omkostningerne for et realkreditlån

I forbindelse med optagelsen af et realkreditlån har vi med to typer af omkostninger at gøre: Engangsomkostninger ved oprettelsestidspunktet og løbende omkostninger gennem lånets løbetid. Både engangs– og løbende omkostninger kan være variable eller faste. Variable omkostninger varierer, som navnet antyder, med størrelsen af restgælden. De faste omkostninger er uafhængige af størrelsen af restgælden.

Afdrag og *rentebetaling* udgør den største del af lånets omkostninger. Vi vil i denne rapport beregne disse løbende variable omkostninger efter *annuitetsprincippet*. Her hæfter vi os ved, at langt den største del af private låntagere foretrækker annuitetslån, d.v.s. lån med lige store terminslige ydelsesbetalinger.

De andre løbende variable omkostninger, som vi betragter i denne rapport er *bidrag*, der betales som en procentdel af restgælden til RI, og kurtage til obligationshandler. For lån mellem 200.000 og 5.000.000 kr. er bidragssatsen 0,5876% af restgælden for tiden. Kurtage er 0,15% af den handlede obligationsmængde for både *solgte* og *indfriede* obligationer. Den væsentligste løbende faste afgift er statens tinglysningsafgift på 1400 kr. og administrationsgebyr på cirka 1000 kr.

De løbende faste omkostninger samt kurtagen har afgørende betydning for hyppigheden af låneomlægninger, eftersom disse omkostninger kun indtræder, hvis vi omlægger et lån. Derfor er det vigtigt at have disse omkostninger med i den endelige optimeringsmodel.

Nogle omkostninger er tilknyttet det enkelte lån, hvor andre omkostninger er tilknyttet et lånesag. En lånesag er en sammensætning af de enkelte lån i en låneportefølje. Således bliver statens tinglysningsafgift og andre ekspeditionsafgift, (i alt cirka 2500 kr. pr. lån), ganget med antallet af lån i låneporteføljen. På den måde kan disse afgifter have betydning for antallet af valgte lån i den optimale låneportefølje.

Vi betragter også de forskellige engangsomkostninger, der kan være variable eller faste. Den største engangsomkostning er tinglysningsstempelafgiften, der er 1,5% af hovedstolen. De faste engangsomkostninger er lånesagsgebyret på 1050 kr. og tinglysnigsekspeditionsafgift på 2800 kr.

Disse engangsomkostninger tilføjes direkte til målfunktion i optimeringsmodellen, og de har således ingen inflydelse på valget af lån eller hyppigheden af låneomlægninger.

I appendix (B.1) kan vi se en detaljeret gennemgang af størrelsen af de enkelte engangs– og løbende omkostninger i forbindelse med oprettelsen og omlægning af et lån hos Nykredit.

2.6 Nuværende praksis inden for låneanbefaling

Betragtningerne i dette kapitel danner grundlag for valg af lån. Jo flere af disse informationer der betragtes i rådgivningen, jo mere værdifuld bliver anbefalingen til den enkelte kunde.

Den nuværende praksis inden for låneanbefaling baserer sig på en kombination af overordnede betragtninger af egenskaberne bag de forskellige typer af lån samt konsekvensberegninger. I en konsekvensberegning genererer man først nogle scenarier for fremtidige renter og obligationskurser. Dernæst vil man, for hvert scenario, beregne priser for de forskellige typer af lån. På den måde foretager analytikeren en stikprøvekontrol af sine forskellige anbefalinger til forskellige kundegrupper. Til disse konsekvensberegninger bruger man de deskriptive modeller, som vi så i diagram 1.1 (side 4).

Det er hensigten med dette projekt at bygge videre på nuværende praksis. De samme deskriptive modeller skal anvendes til generering af scenarier. Der skal yderligere bygges en skræddersyet optimeringsmodel til selve låneanbefalingsproblemet.

Et naturligt spørgsmål ville her være: hvorfor skal man bruge en optimeringsmodel, når der i forvejen findes vejledende anbefalinger baseret på sunde, fornuftige finansielle argumenter, hvoraf nogle blev nævnt tidligere i dette kapitel. Et muligt svar kan gives som følgende:

Selv om det er muligt at rådgive kunderne udfra fornuftige finansielle argumenter, er det tidskrævende at skulle gennemgå samtlige forhold, der gør sig gældende for netop den enkelte kunde. Det er ønskeligt at have et automatiseret rådgivningssystem, således at den enkelte kunde kan forsyne systemet med forudbestemte input data, og systemet til gengæld, i sand tid, kommer med de bedste forslag for optagelsen af lån. Et sådant system kan med fordel bygges på en optimeringsmodel.

Det skal bemærkes, at der ikke er tale om entydige optimale løsninger, men en forventet optimal løsning, eftersom vi har med en stokastisk proces at gøre. Modellen bør derfor betragtes som tilfredsstillende, når løsningerne kan forsvares med gængse finansielle argumenter.

Men hvorfor skal vi bruge en optimeringsmodel fremfor en automatiseret konsekvensberegner:

En optimeringsmodel beskriver alene det underliggende problem ved at introducere en ønskelig målfunktion og de begrænsninger, der identificerer mængden af mulige løsninger. Konsekvensberegning er derimod en løsningsmetode. Den kan for eksempel bruges til at løse optimeringsproblemet, ved at beregne værdien af målfunktionen for samtlige mulige løsninger og vælge den bedste. Denne metode kaldes enumeration og bruges som udgangspunkt for nogle løsningsalgoritmer til løsningen af kombinatoriske problemer i heltalsprogrammering, (se Wolsey, [24]).

Der er således ikke tale om en erstatning for konsekvensberegninger, men derimod indføring af en matematisk beskrivelse og begrænsning af beslutningsproblemet. Optimeringsmodellen er heller ikke en erstatning for de finansielle argumenter der bruges i rådgivningen. Det er lige netop disse argumenter, der skal afspejles i modellen.

Det er vigtigt at slå fast, at det ikke er muligt at tale om en entydig besparelse på f.eks. 5 eller 10 procent, eftersom vi har med en multikriterie stokastisk optimeringsmodel at gøre. Forbedringen kan umiddelbart måles i en hurtigere rådgivning med flere konstruktive informationer til den enkelte kunde. Givet specifikke forudsætninger, som kunden er indforstået med, kan vi dog tale om forventede optimale løsninger.

En optimeringsmodel giver også mulighed for "what if" analyser. D.v.s. at vi kan bruge modellen til hurtigt at generere konsekvenser af forskellige inputdata. Analytikeren kan således bruge optimeringsmodellen som en konsekvensberegner, dog på en højere abstraktionsniveau end de allerede brugte konsekvensberegninger.

I kapitlerne 4, 5 og 6, hvor optimeringsmodellen bliver udviklet og testet, får vi en afklaring på de konceptuelle forskelle mellem nuværende praksis og anvendelse af en optimeringsmodel.

Vi vil dog først analysere og implementere de deskriptive modeller, der er nødvendige for generering af scenarier i næste kapitel (kap. 3).

Kapitel 3

De deskriptive modeller

Dette kapitel består af følgende afsnit: I afsnit 3.1 introducerer vi begrebsapparatet for udviklingen af de deskriptive modeller, som blev præsenteret i diagram 1.1 (side 4). I afsnit 3.2 vil vi, med udgangspunkt i BDT–artiklen (BDT, [6]), udarbejde et generelt ligningssystem, der giver BDT–renterne som løsning. I afsnit 3.3 analyserer og implementerer vi en approksimerende algoritme til løsning af BDT–ligningssystemet.

Et produkt, som ikke hidtil har haft den store interesse hos realkreditkunder er rentesikring på rentetilpasningslån. I afsnit 3.4 beskrives, hvordan rentesikring kan anvendes til at sikre et loft for rentesatsen på et F1 lån i en periode op til 3 refinansierigner. Vi ser, hvordan inkonverterbare obligationer kan prisfastsættes og gør rede for vores valg af metode til bestemmelse af priser for konverterbare obligationer i afsnit 3.5.

Endelig diskuterer vi i afsnit 3.6 rimeligheden i at bruge rentemodellen som grundlag for generering af scenarier, som skal bruges i et stokastisk optimeringsmodel.

Som supplement til den teoretiske gennemgang implementeres BDT–rentemodellen i programmeringsproget Visual Basic for Applications (VBA) i appendix C.1. Ligeledes implementerer vi "leveling" algoritmen til prisfastsættelse af rentesikring i appendix C.2. Den fundne rentemodel, samt priserne for rentesikring fra disse implementeringer, vil senere blive brugt i optimeringsmodellen. Koden der formatterer inputdata til GAMS[1] parametre og tabeller til brug for løsning af optimeringsmodellen findes i appendix D.1.

[1] GAMS står for General Algebraic Modeling Systems.

3.1 Obligationer og renter

Ordet rente dækker over en lang række forskellige begreber: den nominelle rente og realrenten, den korte, mellemlange eller lange rente, den effektive rente osv. Generelt kan man sige, at renten er prisen på udbudte penge. Renten skal sikre, at der er ligevægt mellem mængden af udbudte penge og mængden af efterspurgte penge. (Obligationsinvestering, [10])

Når det handler om realkreditlån, er det obligationsrenter, der har vores interesse, eftersom realkreditlån finansieres ved, at låntageren gennem realkreditinstituttet sælger obligationer til en eller flere investorer. En obligation har en kurs, en pålydende værdi, en pålydende rente (kuponrente) og en udløbsdato. En obligation indfris til kurs pari eller kurs 100 ved udløbsdatoen. Når man udsteder (sælger) en obligation til en kurs på 98 med en årlig kupon på 5% med udløbsdato om et år, betyder dette, at man modtager 98 DKK nu for at betale 105 DKK (den pålydende værdi + kuponrente) om et år. For at beregne den effektive rente, beregner man først den direkte rente på obligationen:

$$\text{Direkte rente} = 5/98 = 5,10\%$$

Den direkte rente angiver dog alene den del af det samlede obligationsafkast, der vedrører rentekomponenten. Hertil kommer, at der også vil være en kursregulering, som for vores eksempel svarer til en kursgevinst på 2 kurspoint. Denne kursgevinst skal også beregnes i forhold til investeringsbeløbet, som igen svarer til kursen. Kursreguleringen i vores eksempel giver:

$$\text{Kursregulering} = (100 - 98)/98 = 2,04\%$$

Den effektive rente kan nu beregnes som den simple sum af den direkte rente og kursreguleringen. Den effektive rente bliver således:

$$\text{Effektiv rente} = 5,10 + 2,04 = 7,14\%$$

Hvis obligationen i stedet var en såkaldt nulkuponobligation, ville kuponrenten være nul, og det samlede afkast ville derfor alene bestå af kursreguleringer. En af grundene til, at det er hensigtsmæssigt at opdele en obligations afkast i en rente– og en kursreguleringskomponent er, at private investorer skal betale skat af renteindtægter, mens kursgevinster er skattefrie. Til gengæld kan man heller ikke fradrage et kurstab i den skattemæssige opgørelse, (Obligationsinvestering, [10]). Den skattemæssige asymmetri gør, at man skal tage højde for skatten, når man beregner den effektive rente på en obligation med kuponer, hvorimod dette ikke er nødvendigt for en nulkuponobligation.

I fortsættelse af vores eksempel, vil vi forestille os en investor, der beskattes med 36% af renteindtægterne. Den direkte rente efter 36% skat beregnes som:

$$\text{Direkte rente efter 36\% skat} = 5,10 \cdot (1 - 0,36) = 3,26\%$$

Da kursgevinsten på 2,04 ikke beskattes, kan den effektive disponible rente opgøres til:

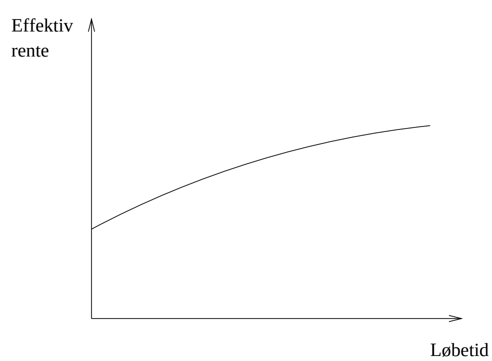
$$\text{Effektiv disponibel rente} = 3,26 + 2,04 = 5,30\%$$

Den samme investor kan med rimelighed forvente at opnå den samme effektive disponible rente, hvis han skulle investere i nulkuponobligationer. Det svarer til, at han vil købe en nulkuponobligation med en kurs på 94,97 med udfaldsdato om et år.

$$\text{Effektiv disponibel rente for nulkuponobligaitonen} = (100 - 94,97)/94,97 = 5,30\%$$

Som det fremgår af dette eksempel spiller det ingen rolle at beregne den effektive disponible rente udfra en obligation med kuponrente eller en nulkuponobligation. I udvikling af rentemodeller er det dog oplagt at tage udgangspunkt i nulkuponobligationer for at gøre beregningerne enklere.

Når en låntager står overfor en beslutning om, hviket lån han skal optage, kender han gennem det finansielle marked de effektive renter, der er gældende på optagelsestidspunktet, for obligationer med forskellige udløbsdatoer. Sammenhængen mellem renteniveauet og løbetider benævnes som ren-



Figur 3.1: Rentestrukturen på et givet tidspunkt for obligationer med forskellige løbetider.

testrukturen. På et givet tidspunkt er det oftest sådan, at obligationer med korte løbetider har lavere effektive renter og obligationer med lange løbetider har højere effektive renter. Renterne stiger typisk logaritmisk, hvilket fremgår af skitsen i figur (3.1). Hvis en rentestruktur, som vist i figuren, har en konveks form, benævnes rentestrukturen en normal rentestruktur. Det kan undertiden forekomme, at en rentestruktur får en konkav form. Hvis dette er tilfældet, kalder vi rentestrukturen for en invers rentestruktur.

For at kunne foretage sammenligninger af forskellige typer af lån har låntageren brug for at kende rentestrukturen også på fremtidige tidspunkter. Men fremtidige renter kan man kun gisne om på nuværende tidspunkt. Fremtidige renter styres af mange økonomiske, politiske og samfundsmæssige variable. Ikke desto mindre er det muligt at udlede markedets forventning om disse renter i form af en stokastisk proces. Rentestrukturteori, som er læren om udledning af disse stokastiske processer, handler om at udlede stokastiske funktioner, som simulerer rentebevægelsen i fremtiden. I de seneste årtier har der været en bemærkelsesværdig udvikling i rentestrukturteorien. I dette kapitel dog nøjes vi med at gennemgå en rentemodel, der har været en af markedets foretrukne gennem 1990'erne, nemlig Black, Derman og Toy modellen. (BDT–modellen, [6])

Grunde til dette valg er mange. For det første giver BDT modellen en intuitiv forståelse for mekanismen bag udviklingen af rentemodeller ved at bruge et diskret rentetræ. For det andet er BDT modellen den blandt de diskrete rentemodeller, der er mest brugt i det finansielle marked. For det tredje får vi renterne fra BDT–modellen som diskrete værdier, som umiddelbart kan bruges som input i optimeringsmodellen.

Det skal også bemærkes, at en vilkårig rentemodel kan anvendes som inputkilde til optimeringsmodellen, idet optimeringsmodellen er uafhængig af valget af rentemodel. En naturlig udvidelse af dette projekt kan derfor være at eksperimentere med andre rentemodeller, deriblandt kontinuerte rentemodeller.

3.2 Black, Derman & Toy modellen

Black, Derman & Toy modellen fra 1990 er en en–faktor rentemodel, hvor markedsinformationen om nulkuponobligationers effektive renter og volatilitet bruges som input.

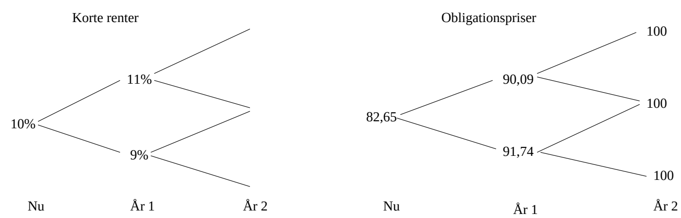
En vigtig forudsætning i BDT–modellen er, at de korteste renter i modellen er lognormal fordelte, (se BDT, [6]). Denne forudsætning medfører, at negative renter ikke vil forekomme i modellen.

En anden interessant egenskab af BDT–modellen er, at modellen bruger dagens rente og volatilitetsstruktur som input. Dagens rentestruktur er nemt tilgængelig på det finansielle marked og volatiliteterne er implicit givet i markedspriser på rentefølsomme derivater. En Kalibreringsteknik bruges til at finde de implicitte volatiliteter. Her forudsætter vi, at volatiliteterne er kendte. [2]

BDT–modellen anvender et rekombinerende binomialtræ med sandsynlighed p for en opadgående bevægelse og sandsynlighed $1 - p$ for en nedadgående bevægelse. Vi vil i det følgende sætte $p = \frac{1}{2}$.

Intuitionen bag modellen kan bedst forstås ved at anvende eksemplet fra den originale artikel i de føgende afsnit.

3.2.1 Prisfastsættelse af obligationer



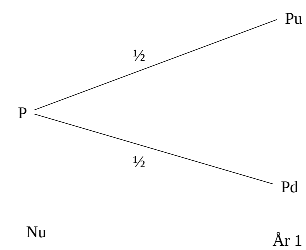
Figur 3.2: To–trin træer med korte renter og tilsvarende obligationspriser.

For at få en intuitiv forståelse af formålet med en rentemodel (BDT–modellen i vores tilfælde) vil vi i dette afsnit se på anvendelsen af outputtet fra en rentemodel til prisfastsættelse af obligationer. Outputtet fra BDT–modellen er et rentetræ. Et eksempel på et sådant træ kan ses til venstre i figur (3.2). Her forestiller vi os, at dagens korte rente (1–årig i vores eksempel) er givet som 10%. Vi forestiller os yderligere, at outputtet fra BDT–modellen giver, at den 1–årige rente enten med sandsynlighed $\frac{1}{2}$ vil stige til 11% eller falde til 9% om et år.

Målet er at finde den indeværende pris P på en to–årig nulkuponobligation. Vi definerer P_u som prisen for denne obligation et år frem i tiden, når renten stiger til 11% med sandsynlighed $\frac{1}{2}$. Ligeledes definerer vi P_d som prisen for obligationen et år frem i tiden, når renten falder til 9% med sandsynlighed $\frac{1}{2}$. Det vil sige, at den forventede pris for obligationen om et år er $\frac{1}{2}(P_u + P_d)$. (fig. 3.3) Givet at den 1–årige effektive obigationsrente betegnes r, kan vi finde obligationens nuværende pris ved at tilbagediskontere den forventede pris om et år:

$$P = \frac{\frac{1}{2}P_u + \frac{1}{2}P_d}{1 + r}, \tag{3.1}$$

[2] For mere om volatiliteter se (Hull, [14]).



Figur 3.3: Prisprocessen for en nulkuponobligation i et et–step binomialtræ.

Formel (3.1) kaldes evalueringsformelen. I vores eksempel er $r = 10\%$. Vi mangler værdierne for P_u og P_d for at bestemme P. Betragt nu træet til højre i figur (3.2). Her ser vi prisprocessen for vores to–årige nulkuponobligation. Om to år indfries obligationen til kurs pari (kurs 100). Obligationsprisen et år fra nu vil være enten 90,09 eller 91,74 alt efter om den korte rente om et år er 11% eller 9%.

$$\frac{\frac{1}{2}100 + \frac{1}{2}100}{1 + 0,11} = 90,09 \qquad , \qquad \frac{\frac{1}{2}100 + \frac{1}{2}100}{1 + 0,09} = 91,74$$

Ved nu at bruge evalueringsformlen (3.1) kan vi nemt finde obligationens nuværende pris:

$$\frac{\frac{1}{2}90,09 + \frac{1}{2}91,74}{1 + 0,10} = 82,65$$

Som det kan ses fra eksemplet, kan vi prisfastsætte en obligation med enhver løbetid til ethvert tidspunkt, bare vi har et rentetræ, der går langt nok frem i tiden. I den virkelige finansielle marked findes eksplicitte priser for obligationer på dagsbasis[3]. Disse priser afspejler dagens effektive renter for obligationer med forskellige løbetider. Der opgives derimod ikke eksplicitte fremtidspriser for obligationer på det finansielle marked. Det er lige netop disse priser med deres tilknyttede sandsynligheder, der skal bruges i vores stokastiske optimeringsmodel i næste kapitel. Disse priser med dertilhørende sandsynligheder kan dog findes implicit ved at finde en rentemodel, der afspejler markedets forventninger om den fremtidige rentebevægelse. Det er en sådan rentemodel, der tilpasses markedsinformationen i BDT–artiklen ([6]).

[3]Fondsbørsen udgiver priser for samtlige danske stats– og realkreditobligationer hver dag.

3.2.2 Tilpasning af BDT–rentemodel til markedsdata

BDT–modellen bruger som input dagens effektive obligationsrenter for obligationer med forskellige løbetider samt dagens volatiliter på obligationsrenter et trin frem i tiden for obligationer med forskellige løbetider. Modellens output er det markedstilpassede korte rentetræ. Det er normalt på det finansielle marked at arbejde med en rentestruktur, som angiver dagens effektive renter på obligationer med forskellige udløbsdatoer samt disses indeværende volatiliteter. Obligationsrenterne i BDT–artiklen ([6]) betegnes som "yields" og volatiliteterne som "yield" volatiliteter. Artiklens eksempel på sådan en rentestruktur er givet her:

Eksempel på en rentestruktur		
Udløbsår	Dagens "yield" (%)	"Yield" volatilitet (%)
1	10	
2	11	19
3	12	18
4	12,5	17
5	13	16

Tabel 3.1: Rentestrukturen normalt angives på denne tabelform i den finansielle marked.

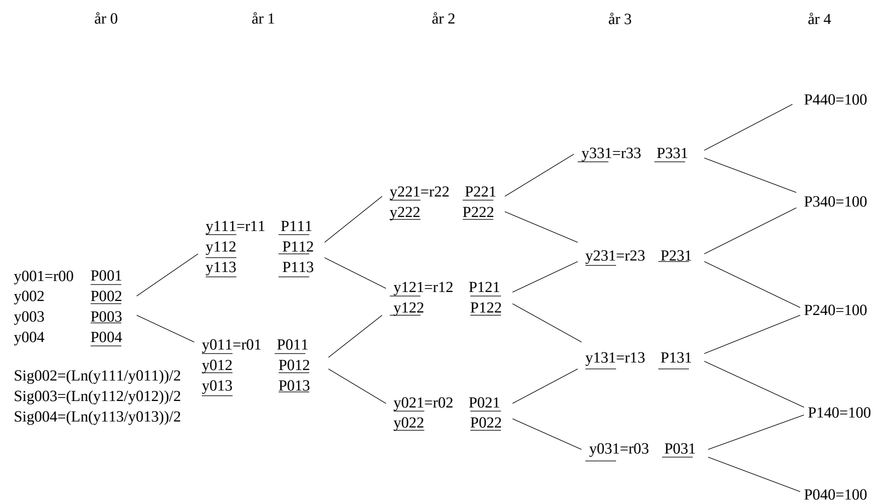
Begrebet "yield", y_{ijm}, bruges som et muligt udfald af den effektive rente over en givet skridtlængde, på niveau i, j år frem i tiden for en m–årig obligation. Hvis vi betragter 1–årige skridtlængder for en 4–årig horisont ($N = 4$), fortolker vi y_{122} som et muligt udfald af den årlige rente ("yield") på niveau 1, 2 år frem i tiden for en 2–årig obligation. I det følgende differentierer vi mellem begreberne korte renter, r_{ij}, og "yields", y_{ijm}. De korte renter er de årlige renter for obligationer med løbetider svarende til skridtlængden (1 år i vores eksempel), hvorimod "yields" er de årlige renter for obligationer med løbetider, der er multipla af skridtlængden og op til horisonten. Sammenhængen mellem de korte renter og "yields" fremgår af figur (3.4). For eksempel er y_{121} en "yield" på niveau 1, i år 2 for en 1–årig nulkuponobligation. Vi refererer til denne "yield" som den korte rente (i vores eksempel en 1–årig rente). y_{122} er derimod ikke en kort rente men "yield" på en 2–årig obligation.

Obligationspriserne P_{ijm} har et entydigt forhold til obligationernes "yields" y_{ijm} nemlig:

$$P_{ijm} = \frac{100}{(1 + y_{ijm})^m} \quad \forall i, j, m \quad , \tag{3.2}$$

hvor vi tilbagediskonterer kurs pari (kurs 100) m år med den 1–årige rente for en m–årig obligation med den tilsvarende "yield".

"Yield" volatiliteten σ_{ijm} udtrykker den spændevidde, der er mellem de forskellige udfald af "yields" i fremtiden. Et år frem har vi, som det fremgår af



Figur 3.4: Et eksempel på et 4–årigt rentetræ med 1–årige skridt. I hver knude kan vi se de kendte og ubekendte parametre. De understregede parametre er ubekendte. Resten er givet som input.

figur (3.4) to sæt mulige "yields". Dagens input "yield" volatiliteter defineres som:[4]

$$\sigma_{00m+1} = \big(\ln \frac{y_{11m}}{y_{01m}}\big)/2, \quad m \in \{1, \cdots, N-1\}. \tag{3.3}$$

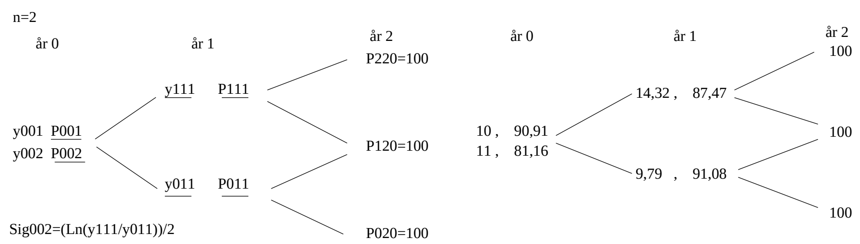
"Yield" volatiliteter er gemt implicit i priserne for de rentefølsomme finansielle derivater såsom rente "caps" eller "floors" (se e.g. J. Hull [14]). Vi vil ikke her gå nærmere ind i, hvordan "yield" volatiliteter kan beregnes, idet dette ligger uden for projektets emneområde. Derimod forudsætter vi, at "yield" volatiliteterne er kendte.

I det følgende finder vi de ubekendte variabler i træet trinvis, indtil vi kan bestemme et generelt ligningssystem for tilpasningen af BDT–modellen til markedsdataen. Tallene kommer fra eksemplet i tabel (3.1).

Vores mål nu er at finde $y_{ij1} = r_{ij}$ værdier. Ifølge tabel (3.1) og figur (3.4) er $r_{00} = y_{001} = 0,10$. Så vi starter med at finde de korte renter et år frem, d.v.s. r_{01} og r_{11}. Det fremgår af figur (3.4) at $r_{ij} = y_{ij1} \;\; i \in \{0, \cdots, j\}, j \in \{0, \cdots, n-1\}$. Derfor nøjes vi i det følgende med at betragte y_{ijm} værdierne. Nedenstående figur viser et to–trin (to år i vores eksempel) forløb:

Dagens rentestruktur (y_{001}, y_{002} samt σ_{002}) er givet som input. Ligeledes ved vi, at 2 år frem i tiden skal obligationerne indfris til kurs 100. Som det

[4] I eksemplet i den originale BDT–artikel blandes begreberne "yield" volatilitet og "spot" volatilitet, hvilket gør det forvirrende at skulle afprøve resultaterne opnået i artiklens eksempel. Bjerksund og Stensland antyder dog i deres artikel fra 1996 (BS, [4]) en mere entydig definition af "yield" volatiliteten: $\sigma_{ijm+1} = (\ln \frac{y_{i+1j+1m}}{y_{ij+1m}})/2 \;\; \forall i, j, m$. Inputtet til BDT–træet består dog kun af de indeværende volatiliteter, d.v.s $i = j = 0$



Figur 3.5: Et to–trin BDT–træproces. Træet til venstre: y_{001}, y_{002} og σ_{002} er givet som input. De understregede variabler er ubekendte. Træet til højre: Løsningen til vores taleksempel.

fremgår af figuren, har vi 6 ubekendte, og for at finde disse har vi brug for 6 ligninger. Evalueringsformlen (3.1) giver os følgende ligning:

$$\frac{\frac{1}{2}P_{111} + \frac{1}{2}P_{011}}{1 + y_{001}} = P_{002}.$$

Formel 3.2 giver os de næste 4 ligninger:

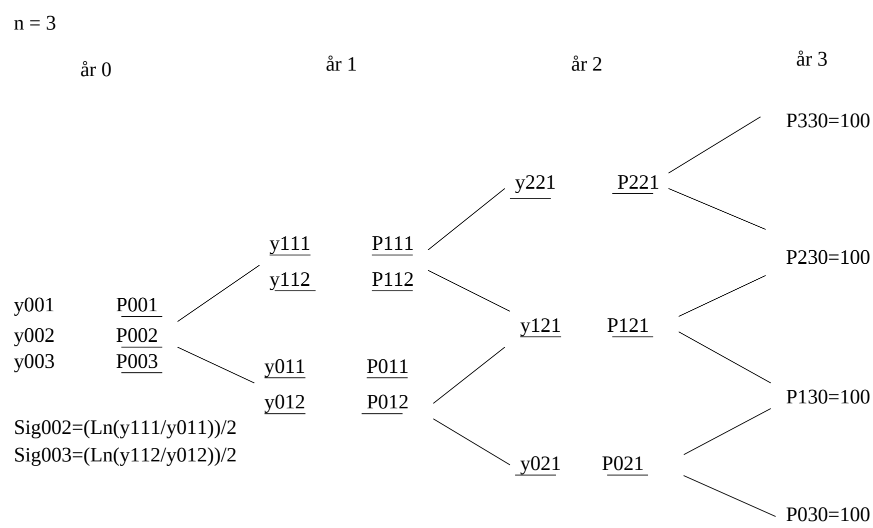
$$\frac{100}{1 + y_{001}} = P_{001}, \qquad \frac{100}{(1 + y_{002})^2} = P_{002}$$
$$\frac{100}{1 + y_{111}} = P_{111}, \qquad \frac{100}{1 + y_{011}} = P_{011}.$$

Endelig får vi den 6. ligning fra (3.3):

$$\big(\ln \frac{y_{111}}{y_{011}}\big)/2 = \sigma_{002}$$

Ved at bruge Excel–solveren finder vi værdierne til de 6 ubekendte, som kan ses i højre side af figur (3.5). Der findes ingen analytiske løsninger til sådanne ikke–linære ligningssystemer. Derimod er der flere numeriske metoder til løsningen af disse. I BDT–artiklen foreslår forfatterne at bruge en gættemetode for at tilpasse rentemodellen til markedsinformationen uden yderligere at komme ind på detaljerne omkring en sådan gættemetode.

Den generelle Newton Raphson metode ([22]) er den mest velkendte numeriske løsning, blandt de generelle løsningsmetoder for løsningen af systemer af ikke–linære ligninger. Metoden udnytter dog ikke strukturen i BDT–ligningerne. Vi vil senere i kapitlet se på implementering af en skræddersyet metode til løsningen af BDT–ligningerne. Vi vil dog først se et trin videre i trædannelsesproceduren, idet ikke alle aspekter af BDT–trædannelsen fremgår fra et 2–trin træ. I BDT–artiklen stopper forklaringerne efter et 2–trin træ, og der bliver heller ikke introduceret en entydig notation, der antyder en generel mønster i processen. Det, sammen med manglende forklaring på definitionen af "yield" volatiliteterne, gør det vanskeligt for læseren at implementere en metode til tilpasningen af BDT–træet, alene baseret på BDT–artiklen. Eksemplet her har derfor til formål at klarlægge BDT–trædannelsesprocessen entydigt og introducere en konsistent notation for ligningssystemet bag denne process for et vilkårligt BDT–træ. I næste trin



Figur 3.6: Et tre–trin BDT–træproces. De 17 understregede variabler er ubekendte.

betragter vi træet i figur (3.6) med 3 trin (3 år i vores eksempel). Som kan ses fra figuren opstår der nu 17 variabler, som vi skal finde 17 ligninger til. Læg mærke til, at vores inputinformation kun vedrører de første 2 trin af træet i form af den indeværende rentestruktur og "yield" volatiliteter. Læg også mærke til, at 6 af de 17 variabler er de samme som i figur (3.5), hvilket vil sige, at vi kan bruge resultaterne fra to–trin problemet i 3–trin problemet. Vi vil dog i det følgende finde frem til samtlige 17 variabler uden at bruge resultaterne fra 2–trin problemet for fuldstændighedens skyld.

Evalueringsformlen (3.1) giver os nu følgende 4 ligninger:

$$\frac{\frac{1}{2}P_{111} + \frac{1}{2}P_{011}}{1 + y_{001}} = P_{002}$$
$$\frac{\frac{1}{2}P_{112} + \frac{1}{2}P_{012}}{1 + y_{001}} = P_{003}$$
$$\frac{\frac{1}{2}P_{221} + \frac{1}{2}P_{121}}{1 + y_{111}} = P_{112}$$
$$\frac{\frac{1}{2}P_{121} + \frac{1}{2}P_{021}}{1 + y_{011}} = P_{012}$$

Formel 3.2 giver 10 ligninger:

$$\frac{100}{1+y_{001}} = P_{001}, \qquad \frac{100}{(1+y_{002})^2} = P_{002}, \qquad \frac{100}{(1+y_{003})^3} = P_{003}$$
$$\frac{100}{1+y_{111}} = P_{111}, \qquad \frac{100}{(1+y_{112})^2} = P_{112}, \qquad \frac{100}{1+y_{011}} = P_{011}, \qquad \frac{100}{(1+y_{012})^2} = P_{012}$$
$$\frac{100}{1+y_{221}} = P_{221}, \qquad \frac{100}{1+y_{121}} = P_{121}, \qquad \frac{100}{1+y_{021}} = P_{021}.$$

Volatilitetsformlen (3.3) giver os 2 ekstra ligninger:

$$\big(\ln \frac{y_{111}}{y_{011}}\big)/2 = \sigma_{002}, \qquad \big(\ln \frac{y_{112}}{y_{012}}\big)/2 = \sigma_{003}.$$

Vi har fundet 16 ligninger med 17 ubekendte. Et sådant system kan som bekendt have uendelig mange løsninger. Imidlertid ved vi, at vi i BDT–modellen forudsætter, at de korte renter er log–normal fordelte i grænsen, d.v.s. volatiliteten et år frem for en obligation, der udløber 3 år frem, er givet ved $\sigma_{112} = \big(\ln \frac{y_{221}}{y_{121}}\big)/2$, hvis renterne stiger og $\sigma_{012} = \big(\ln \frac{y_{121}}{y_{021}}\big)/2$ hvis renterne falder. Men vi også ved, at volatiliteterne kun afhænger af tid og ikke af renternes niveau i fremtiden, derfor har vi at $\sigma_{112} = \sigma_{012}$. Det betyder:

$$\big(\ln \frac{y_{221}}{y_{121}}\big)/2 = \big(\ln \frac{y_{121}}{y_{021}}\big)/2$$
$$\Rightarrow \frac{y_{221}}{y_{121}} = \frac{y_{121}}{y_{021}} \quad \Rightarrow y_{021} = \frac{y_{121}^2}{y_{221}}$$

Herved har vi fundet den 17. ligning.[5] Disse 17 ligninger med de 17 ubekendte løses i Excel. Resultatet er vist i figur (3.7).

Vi har nu al den nødvendige information for at generalisere BDT–trædannelsesprocessen:

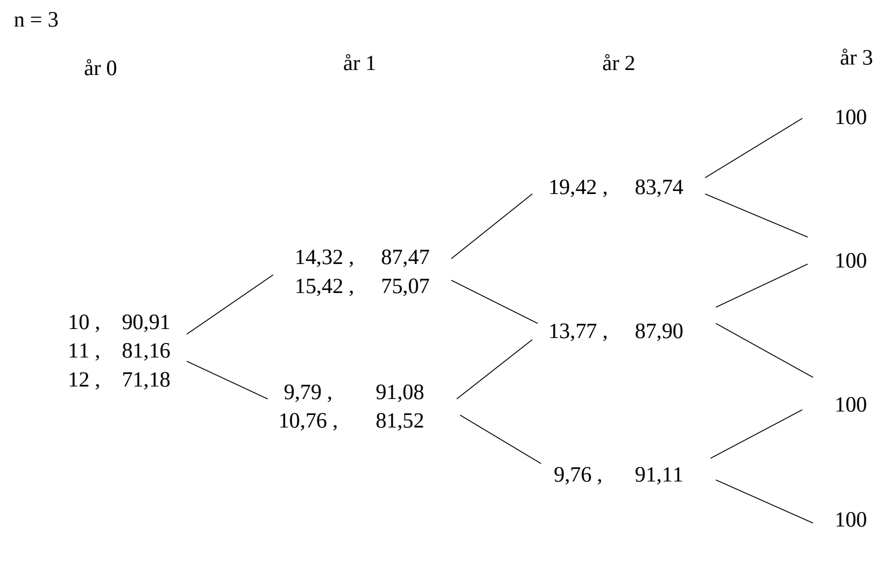
$$\frac{\frac{1}{2}P_{ijm}+\frac{1}{2}P_{i-1,jm}}{1+y_{i-1,j-1,1}} = P_{i-1,j-1,m+1} \qquad \begin{array}{l} i \in \{1,\cdots,j\}, \\ j \in \{1,\cdots,n-1\}, \\ m \in \{1,\cdots,n-j\} \end{array} \tag{3.4}$$

$$\frac{100}{(1+y_{ijm})^m} = P_{ijm} \qquad \begin{array}{l} i \in \{0,\cdots,j\}, \\ j \in \{0,\cdots,n-1\}, \\ m \in \{1,\cdots,n-j\} \end{array} \tag{3.5}$$

$$\big(\ln \frac{y_{11m}}{y_{01m}}\big)/2 = \sigma_{00m+1} \qquad m \in \{1,\cdots,n-1\} \tag{3.6}$$

$$y_{ij1} = \frac{y_{j-1,j1}^{j-i}}{y_{jj1}^{j-i-1}} \qquad \begin{array}{l} i \in \{0,\cdots,j-2\}, \\ j \in \{2,\cdots,n-1\}. \end{array} \tag{3.7}$$

[5] Vi skal være opmærksomme på, at de ligninger, der kobler renterne nederst i træet til de to øverste renter alene kan udledes for de korte renter, d.v.s kun for $m = 1$. Dette er et direkte resultat af, at de korte renter (og ikke "yields") er forudsat log–normal fordelte.



Figur 3.7: Løsningen til vores taleksempel, svarende til figur (3.6).

Bjerksund og Stensland's artikel fra 1996 (BS, [4]) introducerer to lukkede formler, som approksimerer de korte renter i BDT–modellen. I det følgende gennemgår vi denne fremstilling af BDT–modellen samt den algoritme, der udspringer fra approksimationen.

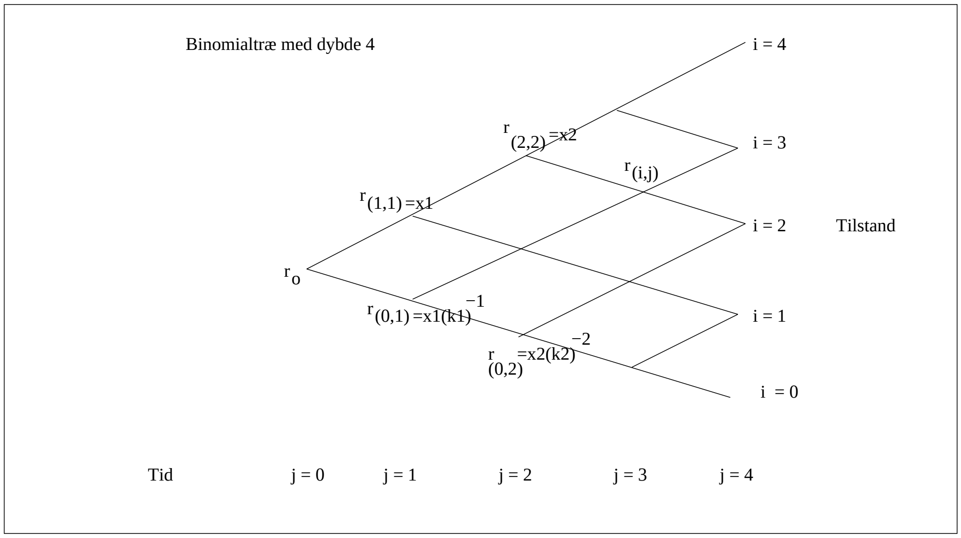
3.3 Bjerksund og Stenslunds approksimerende algoritme til løsning af BDT–modellen

Det ikke linære ligningssystem (3.4 til 3.7) kan løses ved at bruge den generelle Newton–Raphson–metode. Det kan også omskrives til en optimeringsmodel og løses ved for eksempel CONOPT solveren eller i excel. Der findes dog specialiserede metoder til løsning af BDT–modellen. I det følgende betragter vi Bjerksund og Stenslund metoden fra 1996.

3.3.1 Det korte rentetræ

Vi betragter BDT–træet endnu engang (fig. 3.8), nu med fokus på de korte renter. Træet er bygget i to dimensioner; en tidsdimension $j = 0, 1, 2, \cdots, N$, og en tilstandsdimension $i = 0, 1, .., j$. Vi definerer r_{ij} som den korte rente ved tilstand i og tidspunkt j. De forskellige forekomster af den korte rente, idet renterne er forudsat lognormalfordelte i grænsen, kan beskrives som:

$$r_{ij} = x_j (k_j)^{i-j} \qquad i = 0, 1, 2, \cdots, j \quad , \tag{3.8}$$



Figur 3.8: Det binomiale rekombinerende korte rentetræ

hvor x_j er den højeste korte rente ved hvert tidspunkt og k_j angiver forholdet mellem renterne i to nabotilstande, d.v.s. $k_j = \frac{r_{i+1,j}}{r_{ij}}$. Bjerksund og Stenslund udnytter, at k_j alene afhænger af tiden. Træet er dermed entydigt defineret ved den gældende korte rente r_0 samt de to vektorer $X \equiv (x_1, x_2, \cdots, x_{N-1})$ og $K \equiv (k_1, k_2, \cdots, k_{N-1})$. Eftersom vi kan nøjes med at betragte de to øverste tilstande for hvert tidspunkt, dropper vi i det følgende indexet i og bruger bogstavet u og d for op og ned istedet for. Når vektorene X og K er bestemt, kan vi dog bruge formlen (3.8) til at bestemme det endelige rentetræ.

3.3.2 Markedsinformation

I afsnit (3.2) betragtede vi for overskuelighedens skyld kun et–årige tidsskridt. Vi ser i det følgende på en generalisering af BDT–modellen for vilkårlige tidsskridt. Lad t betegne tiden med enheden år og T tidshorisonten. Tidsdimensionen i det korte rentetræ er delt i N skridt. Tidsskridtslængden kan derfor defineres som $\Delta t \equiv T/N$. Tiden t og det j'te tidsskridt er relateret som $t = j\Delta t$. Ligeledes har vi at $m = N - j$, hvor m angiver antallet af tidsperioder til forfaldsdatoen.

De 2 inputdata for tilpasning af den korte rentestrukturmodel kan nu defineres som:

1. Den indeværende implicitte nulkuponobligations "yield" vektor:

   $$Y = [y_0(\Delta t), y_0(2\Delta t), y_0(3\Delta t), \cdots, y_0(N\Delta t)] = [y_{01}, y_{02}, y_{03}, \cdots, y_{0N}],$$

idet vi definerer y_{0m} som den indeværende implicitte årlige "yield" for en nulkuponobligation med forfaldsdato om m tidsperioder.

2. Den indeværende "yield" volatilitetsvektor:

$$\Sigma = [\sigma_0(2\Delta t), \sigma_0(3\Delta t), \cdots, \sigma_0(N\Delta t)] = [\sigma_{02}, \sigma_{03}, \cdots, \sigma_{0N}],$$

hvor σ_{0m} står for den indeværende rentevolatilitet for en nulkuponobligation med forfaldsdato om m perioder.

De to forekomster af "yield" fra en nulkuponobligation et tidsskridt frem er betegnet som y_{1m}^u hhv. y_{1m}^d. Antagelsen om lognormalfordeling af de korte renter giver anledning til følgende definition af "Yield" volatiliteter:

$$y_{1m}^u = y_{1m}^d e^{(2\sigma_{0m+1}\sqrt{\Delta t})}. \tag{3.9}$$

Dette er en generalisering af ligning (3.3). For $t = 0$ har vi at $y_{1m}^u = y_{1m}^d$. For større værdier af t får vi en meget stor forskel mellem y_{1m}^u og y_{1m}^d, eftersom forholdet mellem y_{1m}^u og y_{1m}^d vokser eksponentielt som funktion af $\sqrt{t}$. For $t = 1$ fås ligning (3.3):

$$\sigma_{0m+1} = \big(\ln \frac{y_{1m}^u}{y_{1m}^d}\big)/2.$$

For at opnå fornuftige approksimationer, foreslår Bjerksund og Stensland (se [4]), at vælge tidsskridtet mellem 0 og 1.

Vi definerer P_{0m} som nutidsværdien af en nulkuponobligation med en pålydende værdi af 1 kr ved forfaldstid om m perioder. Af definitionen på "yield" følger at:

$$P_{0m} = (1 + y_{0m})^{-t}, \tag{3.10}$$

hvor $t = j\Delta t$, $m = N - j$ og $m = 1, 2, 3, \cdots, N$.

På tidspunkt 1 defineres P_{1m}^u og P_{1m}^d som værdierne af en nulkuponobligation med pålydende værdi 1 og udløb om m perioder. For $m = 1, 2, 3, \cdots, N - 1$ fås:

$$P_{1m}^u = [1 + y_{1m}^u]^{-(t-\Delta t)}, \tag{3.11}$$
$$P_{1m}^d = [1 + y_{1m}^d]^{-(t-\Delta t)}, \tag{3.12}$$

hvor $t = j\Delta t$ og $m = N - j$. Fra evalueringsreglen (3.1) fås tilsvarende for P_{0m}:

$$P_{0m} = (1 + r_0)^{-\Delta t}\left[\frac{1}{2}P_{1m}^u + \frac{1}{2}P_{1m}^d\right] \tag{3.13}$$

Ligningerne (3.9) til (3.13) bruges til at sikre, at modellen er konsistent med markedsinformationen.

3.3.3 Kalibreringsmetoden

I kalibreringsmetoden anvender vi to approksimerende lukkede formler[6], som genererer rentemodellen ud fra markedsinformationen. Desuden bruges en iterativ procedure, som forbedrer præcisionen.

Fra ligning 3.8 vides, at de korte renter entydigt kan bestemmes ved hjælp af r_0 og de to vektorer K og X, hvor r_0 er kendt på forhånd. De to vektorer K og X kan approksimeres med de to vektorer $\hat{K} \equiv (\hat{k}_1, \hat{k}_2, \hat{k}_3, \cdots, \hat{k}_{N-1})$ og $\hat{X} \equiv (\hat{x}_1, \hat{x}_2, \hat{x}_3, \cdots, \hat{x}_{N-1})$, hvor $\hat{k}_j$ og $\hat{x}_j$ findes ved:

$$\hat{k}_j = \frac{(P^u_{1,j-1}/P^u_{1j}) - 1}{(P^d_{1,j-1}/P^d_{1j}) - 1} \tag{3.14}$$

$$\hat{x}_j = \frac{(P_{0j}/P_{0,j+1}) - 1}{(\hat{k}_j)^{-j}\left(\frac{1}{2}\hat{k}_j + \frac{1}{2}\right)^j}. \tag{3.15}$$

Læg mærke til, at til P vektoren bruger vi indexet j i stedet for den sædvanlige m. Kalibreringsmetoden kan nu beskrives i følgende trin:

1. Ligningerne (3.9 - 3.13) bruges til at bestemme P_{0j}, P^u_{1j} og P^d_{1j}.
2. Formlerne (3.14) og (3.15) bruges til at bestemme $\hat{K}$ og $\hat{X}$.
3. Formel (3.8) bruges til at bestemme det korte rentetræ.

Udfra det approksimerede korte rentetræ, fundet i trin 3, kan vi nu, ved hjælp af ligningerne (3.8) til (3.13)) udlede de implicitte "yields"- og volatilitetsvektorer, $\hat{Y}$ og $\hat{\Sigma}$. Hvis approksimationerne (3.14) og (3.15) er præcise, vil $\hat{Y}$ og $\hat{\Sigma}$ svare til de rigtige markedsinformationer Y og Σ, d.v.s. input. Der er imidlertid ingen garanti for at få den ønskede præcision. For at forbedre præcisionen foreslår Bjerksund og Stensland ([4]) at bruge følgende opdateringer og gentage trin 2 og 3 i kalibreringsprocessen, indtil $\hat{Y}$ og $\hat{\Sigma}$ matcher markedsinformationen med den ønskelige nøjagtighed:

$$P^{u\ (iter)}_{1m} = P^u_{1m} + [P^{u\ (iter-1)}_{1m} - \hat{P}^{u\ (iter-1)}_{1m}], \tag{3.16}$$

$$P^{d\ (iter)}_{1m} = P^d_{1m} + [P^{d\ (iter-1)}_{1m} - \hat{P}^{d\ (iter-1)}_{1m}]. \tag{3.17}$$

Her er $P^{u\ (iter)}_{1m}$ og $P^{d\ (iter)}_{1m}$ de nye input til trin 2 af kalibreringsprocessen. $P^{u\ (iter-1)}_{1m}$ og $P^{d\ (iter-1)}_{1m}$ er input til trin 2 fra den tidligere iteration.

[6]Vi vil i denne rapport ikke gå ind på den teoretiske baggrund for approksimationsformlerne, blot henviser vi til artiklen fra Bjerksund og Stenslund, [4]

$\hat{P}_{1m}^{u\ (iter-1)}$ og $\hat{P}_{1m}^{d\ (iter-1)}$ er de approksimerede priser, der er udledt fra det approksimerede korte rentetræ ved iteration $iter - 1$. P_{1m}^{u} og P_{1m}^{d} er de gældende obligationspriser fra markedsinformationen. For $iter = 1$ gælder: $P_{1m}^{u\ (iter-1)} = P_{1m}^{u}$ og $P_{1m}^{d\ (iter-1)} = P_{1m}^{d}$.

Endelig kan $P_{0m}^{(iter)}$, som også er input til trin 2 af kalibreringsprocessen, findes ved hjælp af formel (3.13).

3.3.4 Pseudokoden

Koden, der genererer BDT–træet, er skrevet i VBA (Visual Basic for Applications) og kan ses i appendix (C). Følgende pseudokode (Alg. 1) repræsenterer hovedparten af implementeringen af BDT–træet. Data indlæses fra linje 1 til 5. Fra linje 6 til 9 genereres en nulkuponprisvektor, som kan bruges til at kontrollere præcisionen af approksimationen til det korte rentetræ. I linje 10 kaldes en funktion, som tager sig af opdateringen af prisvektoren for nulkuponobligationerne. For detaljer omkring denne funktion henvises til appendix (C.2.3). Fra linje 11 til 19 produceres Pd, Yu og Yd vektorer. Pd vektoren sammen med Pu vektoren bruges i linjerne 20 og 21 til at producere K og X vektorene. Her kaldes to funktioner, der svarer til formlerne (3.14) og (3.15) fra det forrige underafsnit. Yu og Yd vektorer kan bruges til at kontrollere, om vi får den samme volatilitet som input. Til slut genereres det korte rentetræ i linjerne 22 til 30. Linjerne 20 til 30 gentages indtil en tilfredsstillende nøjagtighed er opnået jævnfør opdateringsproceduren i ligningerne (3.16) og (3.17).

Algorithm 1 MakeBDTtree

```
 ' Indlæs input
z0 ← GetCurrentShortRate
δ ← GetStepSize
N ← GetNumberofSteps
Zvector ← GetZeroYields
ZVolvector ← GetZeroVolatilities
' Generér nulkuponpriser fra nulkuponrenter
ZeroPrices(1) = (1 + z0)^(−1·δ)
for (k = 2 : N) do
  ZeroPrices(k) = (1 + Zvector(k-1))^(−k·δ)
end for
' Generér pu vektor
puvector = MakePuvec(z0, Zvector, ZVolvector, δ)
' Generér pd vektoren fra pu vektoren
for (k = 1 : N − 1) do
  pdvector(i) = 2 · Zvector(k + 1) · (1 + z0) − puvector(k)
end for
' Generér Yu vektor fra puvector
for (k = 1 : N − 1) do
  Yu(k) = puvector(k)^(−1/(k·δ))−1
end for
' Generér Yd vektor fra pdvector
for (k = 1 : N − 1) do
  Yd(i) = pdvector(k)^(−1/(k·δ))−1
end for
' Følgende kode eksekveres mindst en gang
DO
' Generér K vektor
Kvector = MakeKvec(puvector, pdvector)
' Generér X vektor
Xvector = MakeXvec(puvector, pdvector, Kvector)
' Generér det korte rentetræ
shortRatesTree(1, 1) = z0
for (j=2:N) do
  shortRatesTree(1, j) = Xvector(j - 1)
end for
for (i=2:N) do
  for (j=i:N) do
    shortRatesTree(i, j) = shortRatesTree(i - 1, j) / Kvector(j - 1)
  end for
end for
While   ((margin = Opdater_Puvec_og_Pdvec) > ϵ)
```

3.4 Prisfastsættelse af rentesikring på rentetilpasningslån

Rentesikring er et finansielt instrument, der tilbydes til kunder som i forvejen har optaget et rentetilpasningslån af type F1. Tilknytning af rentesikring til rentetilpasningslånet giver garanti for, at den kontantrente, rentetilpasningslånet skal refinansieres til, over en given periode ikke bliver højere end en aftalt sikringsrente.

Prisfastsættelse af rentesikring er en kompliceret opgave, idet restgælden i et rentetilpasningslån er stiafhængig i det stokastiske rentetræ. Stiafhængigheden betyder, at vi ikke umiddelbart kan bruge et *kombinerende rentetræ*. Formålet med dette afsnit er at udvikle en algoritme til prisfastsættelsen af rentesikring.

Rentesikring er p.t. ikke et eftertragtet produkt på markedet. Dette skyldes blandt andet en undervurdering af den likviditetsrisiko, der er forbundet med rentetilpasningslån, især F1 lån. Manglende markedsføring af rentesikring har også en del af skylden. Når produktet ikke er blevet markedsført i højere grad, skyldes det blandt andet en manglende tilbundsgående undersøgelse af prisfastsættelse og risikovurdering af rentesikring. Af disse grunde kunne et studie af allerede fundne metoder til prisfastsættelse af rentesikring samt en analyse af risici forbundet med produktet være interessant . I dette afsnit vil vi dog hovedsageligt koncentrere os om den første del, nemlig prisfastsættelsesspørgsmålet.

Foreløbig tilbyder Nykredit kun sikringen til rentetilpasningslån af type F1. Til gengæld kan en aftale tegnes for lån med enhver tænkelig afdragsform og løbetid og på et vilkårligt tidspunkt. Aftaler tilbydes med løbetider på op til 4 år (d.v.s. op til 3 refinansieringer). I det følgende vil vi dog begrænse os til at betragte annuitetslån med a årlige terminer på låntagersiden. Den beskrevne metode vil dog også være anvendelig for andre type af F lån og med andre afdragsformer.

En rentesikring (en *cap* på renten) af et F1[7] rentetilpasningslån, svarer til, hvis man ser det fra realkreditinstituttets side, at sælge en serie af *putoptioner* til låntageren, som køber retten til at sælge 1-årige nulkuponobligationer til en given kurs.
Den mængde af obligationer, som låntager ønsker at sælge, svarer selvfølgelig til restgælden på lånet. Denne restgæld er stiafhængig, da størrelsen af det samlede afdrag på et givet tidspunkt afhænger af, hvilken rentesats lånet tidligere har været finansieret til.
Dette gør prisfastsættelsen af rentesikringen temmelig kompliceret, da man ikke kan se på putoptionerne enkeltvis, men må betragte dem som en portefølje af putoptioner, hvis værdi afhænger af, hvor stor restgælden er.

[7] Et fastforrentet annuitetslån, som refinansieres hvert år ved udstedelse af 1-årige nulkuponobligationer

3.4.1 Notation

N	Restløbetiden på realkreditlånet i antal år
a	Antal årlige terminer
j	Terminen, man befinder sig i $j = 0, ...N \cdot a - 1$
i	Tilstanden i knuden, man er i
i^*	Tilstanden i knuden, man kom fra
r_{ij}	Årlige rentesats i tilstand i på tidspunkt j uden rentesikring
$\bar{r}_{ij}$	Årlige rentesats i tilstand i på tidspunkt j med rentesikring
RG_{ij}	Restgælden i tilstand i på tidspunkt j uden rentesikring
$\bar{RG}_{ij}$	Restgælden i tilstand i på tidspunkt j med rentesikring
$\bar{RG}_{ij0}$	Normeret restgæld[8] i tilstand i på tidspunkt j med rentesikring
Y_{ij}	Ydelsen i tilstand i på tidspunkt j uden rentesikring
$\bar{Y}_{ij}$	Ydelsen i tilstand i på tidspunkt j med rentesikring
V_{ij}	Værdi af rentesikringen i tilstand i på tidspunkt j
P_0	Prisen på rentesikringen

3.4.2 Beregning af prisen på rentesikring

Uden tab af generalitet vil beregningerne bygge på et eksempel med et lån med en restløbetid på 4 år, 4 årlige terminer, en restgæld på 1, samt en sikringsrente på 7% i år 2 og 3.
Det korte rentetræ er genereret ud fra nulkuponrenter på hhv. 6,27%, 6,30%, 6,41% og 6,51% og volatiliteter på hhv. 17%, 16% og 15%.

Man vil kunne beregne værdien af rentesikringen ved først at beregne restgæld og ydelser ud fra standardformlerne for annuitetslån:
Den første restgæld er givet: $RG_{0,0} = \bar{RG}_{0,0} = 1$. Herefter beregnes ydelserne og de nye restgælde for tilfældet uden rentesikring iterativt efter annuitets formlerne:

$$Y_{ij} = RG_{i,j-1} \cdot \frac{\frac{r_{ij}}{a}}{1 - (1 + \frac{r_{ij}}{a})^{-(N \cdot a - j + 1)}}$$

$$RG_{ij} = RG_{i^*,j-1} - (Y_{ij} - RG_{i^*,j-1}\frac{r_{ij}}{a})$$

Tilsvarende kan ydelserne og restgældene for tilfældet med rentesikring findes ved formlerne:

$$\bar{Y}_{ij} = \bar{RG}_{i,j-1} \cdot \frac{\frac{\bar{r}_{ij}}{a}}{1 - (1 + \frac{\bar{r}_{ij}}{a})^{-(N \cdot a - j + 1)}}$$

[8] Vi normerer restgælden til 1 i starten af hver periode, svarende til at vi har en restgæld på 1 DKK primo hver periode.

$$\bar{RG}_{ij} = \bar{RG}_{i^*,j-1} - (\bar{Y}_{ij} - \bar{RG}_{i^*,j-1}\frac{\bar{r}_{ij}}{a})\bar{Y}_{ij}$$

Værdien af rentesikringen, V_{00}, kan nu beregnes ved baglæns induktion ved hjælp af følgende iterative formel:

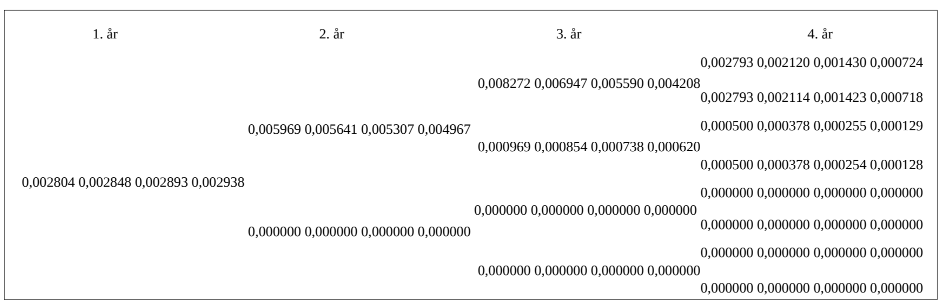
$$V_{i,15} = \frac{(Y_{i,15} - \bar{Y}_{i,15})}{1 + \frac{r_{i,15}}{a}} \qquad \forall i \tag{3.18}$$

$$V_{ij} = \frac{\left[(Y_{ij} - \bar{Y}_{ij}) + \frac{1}{2}(V_{2i,j+1} + V_{2i+1,j+1})\right]}{1 + \frac{r_{ij}}{a}} \qquad \forall i, j \in \{3, 7, 11\} \tag{3.19}$$

$$V_{ij} = \frac{\left[(Y_{ij} - \bar{Y}_{ij}) + V_{i,j+1}\right]}{1 + \frac{r_{ij}}{a}} \qquad \forall i, j \backslash \{3, 7, 11, 15\} \tag{3.20}$$

V_{ij} skal forstås som værdien af rentesikringen fra knude (ij) til og med de sidste efterkommere af knude (ij). Eftersom betalingen sker ultimo en perode, j, og værdien beregnes primo perioden, tilbagediskonterer vi den vundne besparelse fra brugen af rentesikring med den gældene rente for perioden. Formlen initialiseres i (3.18) for den sidste periode, d.v.s. $j = N \cdot a - 1$ ($j = 15$ i vores eksempel). Formlerne (3.19) og (3.20) bruges for de knuder, hvor der følger umidelbart en refinansiering, h.h.v. for de knuder, hvor der ikke følger en refinansiering.

Ifølge eksemplet, vil dette give værditræet i figur (3.9):



Figur 3.9: Værdier af rentesikring.

Som kan ses i knude (00), får vi en samlet pris på $V_{00} = 0,00280442$ for en initial restgæld på 1 DKK eller $P_0 = 2.804,42$ for et lån med en restgæld på 1.000.000.

I eksemplet ender vi med 8 eller 2^{N-1} knuder ved udløb, og ser vi på et tilsvarende lån med en løbetid på 30 år, vil vi komme op på $2^{29} = 536.870.912$ knuder. Så problemet med denne intuitive prisfastsættelsesmetode er, at den er meget regnetung. Der findes ikke skræddersyede metoder i den gængse litteratur til prisfastsættelsen af rentesikring. Der findes dog metoder til prisfastsættelse af instrumenter, hvor *betalingsstrømmen*[9] er stiafhængig. Disse metoder er oftest approksimerende eller heuristiske. Monte Carlo simulering er mest anvendt i disse metoder. De heuristiske løsningsmetoder går ud på at vælge et antal af scenarierne efter en tilfældighedsprincip i stedet for at betragte alle scenarierne. David Luenberger har dog i sin bog "Investment Science" ([16], Luenberger p. 391-395) beskrevet en generel metode til en klasse af prisfastsættelsesproblemer, hvor det er muligt at bebeholde strukturen i det kombinerende rentetræ og alligevel finde den eksakte pris til instrumentet. Dette betyder, at antallet af knuder ved udløb bliver reduceret til N (30 for et 30–årigt F1 lån). Metoden er kendt under navnet "leveling". Vi vil i det følgende analysere leveling og skræddersy den til prisfastsættelsen af rentesikringen.

3.4.3 Beregning af prisen på rentesikring med leveling

Prisen på rentesikring beregnes ud fra ydelserne på rentetilpasningslånet, hvis størrelse ikke kun afhænger af den rentesats, som er aktuel i den enkelte knude i træet, men også restgælden, som afhænger af udviklingen i rentesatsen i stien op til knuden.

Umiddelbart ser det ud til, at vi er nødt til at anvende et ikke–kombinerende træ for at prisfastsætte rentesikringen, som vi gjorde i sidste afsnit. Vi vil dog i det følgende se, hvordan "leveling" kan skræddersys til at omdanne problemet således, at det følger et rekombinerende træ.

Metoden tager udgangspunkt i et problem, hvor betalingsstrømmen afhænger af to variabler: y, som er stiuafhængig, og x, som er stiafhængig. Hvis betalingsstrømmen er lineært afhængig af x, kan problemet omskrives til at være stiuafhængigt ved at fastlægge et fast niveau af x_0 og anvende dette niveau i alle knuder. Prisen V_{ij} i knude (ij) bestemmes på følgende måde:

Vi definerer K_{ij} som knude (ij) og dennes efterkommeres bidrag til instrumentets samlede pris når $x = x_0$ primo periode j. Det rigtige bidrag, d.v.s når $x \neq x_0$, er nu, idet betalingsstrømmen er lineært afhængig af x givet ved $\frac{x}{x_0} \cdot K_{ij}$. Hvis man vælger $x_0 = 1$, kan man finde prisen i hver knude som $V_{ij}(x) = K_{ij} \cdot x$.

Per definition giver V_{00} instrumentets samlede pris. Problemet går nu ud på at bestemme værdien for K_{00}. Vi vil i det følgende vise, hvordan K_{00} og derved V_{00} kan bestemmes for rentesikring på et F1 lån.

Den terminslige ydelse, $\bar{Y}_{ij}$, er betalingsstrømmen i vores F1 lån. Denne ydelse afhænger lineært af den stiafhængige variabel, nemlig restgælden

[9] Betalingsstrømmen i vores tilfælde svarer til låntagers terminslige nettobetaling. I litteraturen dog er betalingsstømmen typisk afkastet på forskellige instrumenter.

$\bar{RG}_{ij}$. Ydelsen afhænger også af renten, $\bar{r}_{ij}$.

$$\bar{Y}_{ij} = \bar{RG}^{primo}_{ij0} \cdot \frac{\frac{\bar{r}_{ij}}{a}}{1 - (1 + \frac{\bar{r}_{ij}}{a})^{-(N \cdot a - j + 1)}} \quad ,$$

hvor $\bar{RG}^{primo}_{ij0} = 1$ og N er udløbstiden. Når vi bruger taleksemplet fra sidste afsnit, får vi ydelserne med rentesikring på et F1 lån med restgæld lig 1 primo hver periode :

1. år	2. år	3. år	4. år
			0,26558 0,34988 0,51854 1,0246
		0,13504 0,15303 0,17702 0,21062	
			0,26195 0,34604 0,51426 1,0190
	0,093114 0,100730 0,109875 0,12105		
		0,13428 0,15226 0,17624 0,20982	
0,071151 0,075330 0,080109 0,085626			
			0,25918 0,34310 0,51096 1,01458
	0,090630 0,098241 0,107377 0,118547		
		0,13184 0,14979 0,17373 0,20725	
			0,25705 0,34084 0,50843 1,0112

Figur 3.10: Ydelser med rentesikring for et F1 lån med restgæld 1 primo hver periode.

Når ydelserne er betalt i hver knude, har vi så følgende restgæld ultimo hver periode:

$$\bar{RG}^{ultimo}_{ij0} = \bar{RG}^{primo}_{ij0} - (\bar{Y}_{ij} - \bar{RG}^{primo}_{ij0} \frac{\bar{r}_{ij}}{a})^{-(N \cdot a - j + 1)}$$

Figur (3.11) giver disse værdier for vores taleksempel:

1. år	2. år	3. år	4. år
			0,75905 0,67474 0,50608 0,00000
		0,88246 0,86447 0,84048 0,80688	
	0,92439 0,91677 0,90763 0,89644		0,75699 0,67290 0,50469 0,00000
		0,88192 0,86394 0,83996 0,80638	
0,94452 0,94035 0,93557 0,93005			
	0,92253 0,91491 0,90578 0,89461		0,75540 0,67148 0,50362 0,00000
		0,88015 0,86220 0,83826 0,80474	
			0,75417 0,67038 0,50279 0,00000

Figur 3.11: Restgæld efter ydelse.

Værdien af rentesikringen i hver knude kan nu findes ved at bevæge sig baglæns i træet.
Med udgangspunkt i, at der er lige stor sandsynlighed for, at renten går op

eller ned ved starten af hver periode, kan værdien i en knude findes som:

$$K_{ij} = \frac{\bar{RG}_{ij0}^{ultimo} \cdot \left[1 + \frac{1}{2} \cdot \frac{K_{i+1,j+1} + K_{i,j+1}}{\bar{RG}_{ij0}^{primo}}\right] + \bar{Y}_{ij}}{1 + \frac{r_{ij}}{a}} - \bar{RG}_{ij0}^{primo} \ .$$

Vi har ved indføring af denne formel skræddersyet "leveling" til bestemmelesen af prisen på en rentesikring på et F1 lån. Formlens korrekthed kan vi argumentere for som følger:

Vi sætter $\bar{RG}_{ij0}^{primo} = 1$, som vi hidtil har gjort i eksemplet for overskuelighedens skyld:

$$K_{ij} = \frac{\bar{RG}_{ij0}^{ultimo} \cdot \left[1 + \frac{1}{2}\left(K_{i+1,j+1} + K_{i,j+1}\right)\right] + \bar{Y}_{ij}}{1 + \frac{r_{ij}}{a}} - 1 \quad \forall i, j \in \{3, 7, 11, 15\} \tag{3.21}$$

For den sidste termin, $j = 15$, ved vi, at $K_{i,j+1} = K_{i+1,j+1} = 0$, idet den sidste betaling falder ved den sidste termin i år 4, og derfor har sikringen selvfølgelig ingen værdig ved starten af år 5, idet lånet er helt udbetalt. Af samme grund er $\bar{RG}_{i,15,0}^{ultimo} = 0$, eftersom vi skal sikre os, at lånet er helt betalt ved den 16. termin (j=15).

Derved får vi startformlen for den sidste periode:

$$K_{i,15} = \frac{\bar{Y}_{i,15}}{1 + \frac{r_{i,15}}{a}} - 1 \quad \forall i \ .$$

Læg mærke til, at ydelsen er beregnet med rentesikring, men den rente

1. år	2. år	3. år	4. år
			0,0 0,0 0,0 0,0
		−0,013706 −0,010904 −0,007818 −0,004290	
			0,0 0,0 0,0 0,0
	−0,007723 −0,007446 −0,00720 −0,006997		
		0,000000 0,000000 0,000000 0,000000	
−0,002804 −0,003016 −0,003257 −0,003536			
			0,0 0,0 0,0 0,0
	0,000000 0,000000 0,000000 0,000000		
		0,000000 0,000000 0,000000 0,000000	
			0,0 0,0 0,0 0,0

Figur 3.12: Værdier af rentesikring.

ydelsen bliver tilbagediskonteret med, er renten uden rentesikring. Dette er årsagen til, at den tilbagediskonterede ydelse er mindre end eller lig med 1. Dette medfører, at værdien af rentesikring udtrykkes enten som en negativ værdi, svarende til en besparelse, eller er værdien 0 svarende til, at renten i den gældende knude har været lavere end sikringsniveauet eller at der ikke er købt en rentesikring i den gældende knude. Idet vi i eksemplet har forudsat, at rentesikringen kun gælder for år 2 og 3, så ser vi i figur (3.12), at de sidste 4 koloner indeholder nul værdier. Dette er en interessant observation, idet det betyder, at vi alene skal regne K_{ij} værdierne fra den sidste periode hvor rentesikringen er blevet anvendt, i modsætning til standard–metoden fra sidste afsnit, hvor hele forløbet skulle tages i betragtning.

Læg også mærke til, at formel (3.21) kun gælder for terminerne før en rentetilpasning, d.v.s for terminerne 4, 8, 12 og 16 (d.v.s. $j \in \{3, 7, 11, 15\}$). For de øvrige terminer erstatter vi $\frac{1}{2} \cdot (K_{i+1,j+1} + K_{i,j+1})$ med $K_{i,j+1}$ da træet her udvikler sig deterministisk:

$$K_{ij} = \frac{\bar{RG}_{ij0}^{ultimo} \cdot \left[1 + K_{i,j+1}\right] + \bar{Y}_{ij}}{1 + \frac{r_{ij}}{a}} - 1 \quad \forall i, j \backslash \{3, 7, 11, 15\}. \tag{3.22}$$

For $j = 14 \cdots 0$ bruger vi formlerne (3.21) eller (3.22). Her er det mere kompliceret at vise formlens korrekthed. Formlen kan bedst forstås ved at dele den op i flere omgange. Lad os definere:

$$A = \bar{RG}_{ij0}^{ultimo} \cdot \left[1 + \frac{1}{2}\big(K_{i+1,j+1} + K_{i,j+1}\big)\right] + \bar{Y}_{ij}.$$

Det første og det tredje element i A definerer vi nu som:

$$B = \bar{RG}_{ij0}^{ultimo} + \bar{Y}_{ij}.$$

B er udtryk for vores rentesikrede betalingsforpligtelser, delvis indfriet gennem ydelsen og delvis i form af ultimo restgælden. Beløbet $\frac{B}{1+\frac{r_{ij}}{a}}$ (hvor r_{ij} er periodens rente uden rentesikring), svarer til primorestgælden uden rentesikring, givet vi ville betale det samme som vi ville have betalt for denne periode med rentesikring og med en primorestgæld 1. Med andre ord giver $\frac{B}{1+\frac{r_{ij}}{a}} - 1$ værdien af rentesikring for termin j niveau i. Denne værdi er mindre end eller lig nul, svarende til en besparelse. Den tredje element i A definerer vi som:

$$C = \bar{RG}_{ij0}^{ultimo} \cdot \frac{1}{2}\big(K_{i+1,j+1} + K_{i,j+1}\big)$$

C er rentesikringens gennemsnitlige værdi i de efterfølgende terminer, vægtet med ultimorestgælden.[10] Det er her, at forudsætningen om et lineært forhold mellem ydelsen (vores betalingsstrøm) og restgælden (den stiafhængige variabel) er nødvendig. Eftersom ydelsen og restgælden har et lineært forhold i annuitetsformlen, er denne forudsætning opfyldt. Alt i alt får vi, at $\frac{B+C}{1+\frac{r_{ij}}{a}} - 1$ giver os bidraget til rentesikringens pris fra knude ij og dennes samtlige efterkomere.

Når vi er kommet til tidspunkt 0, har vi fået rentesikringens pris givet at vi starter ud med en restgæld på 1. Tilbage er der kun at multiplicere værdien i tidspunkt 0 med den rigtige restgæld, så har man prisen på rentesikringen:

$$P_0 = V_{0,0} = K_{0,0} \cdot \bar{RG}_{0,0} \ ,$$

[10] Eftersom C er et udtryk for en besparelse har vi at $C \leq 0$, hvilket vil sige $B + C \leq B$.

som i eksemplet bliver $K_{0,0} = 0,00280442$ og $P_0 = 2.804,42$ for et lån på 1.000.000. Dette er som forventet det samme resultat, som fås ved beregning ved hjælp af metoden i sidste afsnit, d.v.s. at vi, ved anvendelse af metoden "leveling", har fundet en ny effektiv metode til at beregne prisen på en rentesikring på.

Prisfastsættelsen af rentesikring kan altså foretages eksakt ved hjælp af leveling, hvilket reducerer problemet fra at være stiafhængigt til at være stiuafhængigt. En yderligere fordel er, som allerede påpeget, at man ikke behøver at regne baglæns fra lånets udløbstid, men kun fra den sidste periode med rentesikring, hvor sikringsrenten er lavere end den forventede markedsrente i mindst én tilstand. I alle knuder, som kommer senere i træet, er værdien 0. Implementeringen af prisfastsættelsen af rentesikring kan ses i appendix (C.2).

3.4.4 Risikovurdering og prisfastsættelse i praksis

Ovenstående prisfastsættelse er udelukkende bygget på, hvordan man forventer, at rentekurven vil udvikle sig. Når prisen beregnes, tages der selvfølgelig hensyn til den forventede volatilitet, men da det er en gennemsnitlig betragtning, kan det give et stort tab, hvis renten udvikler sig til et niveau, som ligger langt over det forventede. Samtidig betyder det, at hvis låntager ønsker en høj sikringsrente (i eksemplet en rente over 8,75 %) vil prisen på rentesikringen blive 0, selvom realkreditinstituttet (RI) i værste fald vil kunne tabe på forretningen.

Når RI skal vurdere risikoen ved at sælge en rentesikring, skal de altså huske på, at tabet i teorien er ubegrænset, hvorimod gevinsten aldrig bliver højere end prisen på rentesikringen.

RI må derfor vurdere, om de skal lægge et tillæg til prisen, som kan kompensere for en del af risikoen. RI kan selvfølgelig gå ud i markedet og afdække rentesikringen helt eller delvist, men dette er ikke uden omkostninger.

Argumentet for at sælge produktet til den teoretiske pris kan være, at RI ønsker at yde denne service over for sine låntagere. Hvis RI ikke skal risikere at tabe på denne service, må de tage et tillæg/gebyr over den teoretiske pris ved salg af rentesikring. Hvor stort dette tillæg skal være, og om der overhovedet skal være et tillæg, er en helt anden problemstilling, som vi ikke kommer nærmere i denne rapport. Senere i rapporten vil vi dog bruge rentesikringspriser som input til optimeringsmodellen. Resultater fra optimeringsmodellen vil give os en ide om, hvorvidt vi har underpriset eller overpriset rentesikringen.

3.5 Prisfastsættelse af inkonverterbare og konverterbare obligationer

I afsnit 3.2.1 (side 19) så vi, som en indledende motivation inden gennemgangen af BDT–rentemodellen, hvordan de fundne renter i modellen kan bruges til at bestemme priser på nulkuponobligationer. De obligationer, vi hidtil har betragtet, kaldes *inkonverterbare* obligationer. Det betyder, at der ikke er indbyggede *call optioner* i obligationerne. I vores lånesammenhænge betyder det, at låntageren (sælgeren af obligationen) skal betale dagens kurs på obligationen for at indfri sit gæld. Hvis obligationskursen ved oprettelsestidspunktet er 100, og hvis kursen en måned senere stiger til 200, betyder det et kurstab på 100% eller en fordobling af låntagerens restgæld, hvis låntageren skal indfri lånet.

Det er grunden til, at fastforrentede lån oprettes i *konverterbare* obligationer. En konverterbar obligation har en indbygget call option, der giver sælgeren (låntageren) ret til at købe obligationen tilbage (indfri lånet) til højst kurs 100. Når lånet oprettes i en obligation med en kurs tæt på kurs pari (kurs 100), er låntageren sikret en begrænsning af kurstabet, i tilfælde af en kursstigning.

Til låneanbefalingsproblemet har vi brug for priser på både inkonverterbare og konverterbare obligationer. Dette skyldes at tilpasningslån oprettes i inkonverterbare obligationer, hvorimod fastforrentede lån oprettes i konverterbare obligationer. Vi kan heller ikke nøjes med kurserne på nulkuponobligationerne, men har derimod brug for obligationer med pålydende kuponer. Vi vil i det følgende se, hvordan syntetiske obligationer med dertil hørende kurser og kuponrenter kan estimeres baseret på de fundne renter i BDT–modellen.

Prisfastsættelse af inkonverterbare obligationer

Vi forudsætter, at vi har løst BDT–ligningerne. Det er ensbetydende med, at vi allerede har priserne på nulkuponobligationerne fra ligning 3.5 (side 25), som vi gengiver her:

$$P_{ijm} = \frac{1}{(1+y_{ijm})^m} \qquad \begin{array}{l} i \in \{0, \cdots, j\}, \\ j \in \{0, \cdots, n-1\}, \\ m \in \{1, \cdots, n-j\}. \end{array}$$

Her er P_{ijm} kursen for en m–årig nulkuponobligation ved knude ij, niveau i år j. Vi minder også, at n er antallet af perioder i BDT–træet og y_{ijm} er den effektive rente på den m–årige nulkuponobligation i knude ij. Læg også mærke til, at vi repræsenterer kurs 100 med et 1–tal, efter markedets praksis, så vi vil arbejde med en kurs på 0,98 og en rente på 0,03 i stedet for en kurs på 98 og en rente på 2. Problemet går nu ud på at finde prisen til en obligation med kuponer. Vi ser først på et eksempel:

Hvis den 3–årige effektive rente på en nulkuponobligation i knude ij er 0,035 findes kursen på nulkuponobligationen ved:

$$P_{ij3}^{(nul)} = \frac{1}{(1+0,035)^3} = 0,9019 \ .$$

Denne kurs svarer til en kuponrente på 0, ifølge definitionen af en nulkuponobligation. Hvis vi ønsker at finde kursen på en tilsvarende obligation med en årlig kupon på 0,02, kan vi beskrive denne obligations betalingsrække i følgende ligning:

$$P_{ij3}^{(2\%)} = (1-skat)\cdot kupon\cdot\left((1+y_{ij1})^{-1}+(1+y_{ij2})^{-2}+(1+y_{ij3})^{-3}\right)+1\cdot(1+y_{ij3})^{-3}.$$

Her har vi at $kupon = 0,02$, skattefradraget fra rentebetalingen sættes til $skat = 0,36$ og vi har allerede fundet 1–, 2– og 3–årige renter som mellemregninger i BDT–modellen. Vi forudsætter, at disse renter er fundet til $y_{ij1} = 0,025, y_{ij2} = 0,030, y_{ij3} = 0,035$. Når vi indsætter værdierne, får vi en kurs på 0,9380. Denne er kursen på en 3 årig inkonverterbar obligation med årlige kuponer på 0,02.

Helt generelt fås kursen på en m–årig obligation i knude ij med given kuponrnete, given skattesats for kuponbetalinger og fundne 1– til m–årige effektive renter som følger:

$$P_{ijm} = (1-skat)kupon\Big(\sum_{l=1}^{m}(1+y_{ijl})^{-l}\Big) + \frac{1}{(1+y_{ijm})^m} \qquad \begin{array}{l} i \in \{0,\cdots,j\}, \\ j \in \{0,\cdots,n-1\}, \\ m \in \{1,\cdots,n-j\}. \end{array} \tag{3.23}$$

Prisfastsættelse af konverterbare fastforrentede obligationer

Det er mere kompliceret med prisfastsættelsen af konverterbare obligationer. Som sagt i starten af dette afsnit har konverterbare obligationer indbyggede call optioner, der giver sælgeren (låntageren) ret til at indfri lånet til højest kurs 100 på et hvilket som helst tidspunkt under låneforløbet (i praksis 4 gange om året til forudbestemte terminer). Dette indebærer, at konverterbare obligationer handles højest til kurs 100 plus det, der svarer til transaktionsomkostninger for obligationshandel. Hvis kurserne var meget højere end 100, vil sælgeren udnytter sin call option øjeblikkeligt og derved tjene risikofrie penge.

Foreløbig har vi argumenteret for, at konverterbare obliagationer skal have en kurs, som ikke er meget højere end 100. Udover det skal den effektive rente for konverterbare obligationer være højere end den effektive rente for inkonverterbare obligationer. Forskellen skal svare til værdien af call–optionen på obligationen. Vi har allerede set et eksempel på prisfastsættelse af optioner på obligationer under vores gennemgang af rentesikring, hvor vi fortolkede rentesikringen som en portefølje af *europæiske* put optioner. De indbyggede call optioner på de konverterbare obligationer er imidlertid *amerikanske* optioner. Forskellen mellem de to typer optioner er, at de europæiske optioner kan *exercises* kun til et forudbestemt tidspunkt, hvorimod en amerikansk option kan exercises i hele perioden op til et forudbestemt tidspunkt.

Der findes mange metoder til prisfastsættelse af amerikanske optioner, (se [10, 14, 5]). Vi vil dog ikke i denne rapport komme denne problemstilling nærmere. I stedet for har produktlaboratoriet i Nykredit, ved at bruge empiriske erfaringer, estimeret kurserne på konverterbare obligationer ud fra de teoretiske kurser på inkonverterbare obligationer som beskrevet i ligning (3.23).

I appendix D.1 fidnes koden, der tilrettelægger de teoretisk fundne kurser og renter, samt de empiriske kurser og renter, i et format, der kan bruges i GAMS implementering af optimerigsmodellen.

3.6 Forbehold for brugen af en rentemodel i et stokastisk optimeringsproblem

For at prisfastsætte et rentefølsomt finansiel derivat er der brug for en rentemodel, der simulerer den forventede rentebevægelse. Dette har vi set et eksempel på i afsnit (3.4). Et andet eksempel på anvendelse af rentemodeller er prisfastsættelse af optioner på konverterbare obligationer (se [10, 14, 5]). Fællesnævneren for disse prisfastsættelsesmodeller er, at de er *arbitragefrie* modeller (se [5]). At modellerne er arbitragefrie betyder, at priserne skal bestemmes således, at det ikke burde være muligt at tjene risikofrie penge. [11]

Med andre ord, rentemodeller bruges i prisfastsættelsessammenhænge for at skabe en konsistent prispolitik, der lukker mulighederne for spekulation. Det er vigtigt at være klar over, at prisfastsættelsesmodeller ikke bruger rentemodellerne til at forudse fremtiden. Og dette er lige netop skillelinjen mellem den måde, prisfastsættelsesmodeller og optimeringsmodeller bruger rentemodellerne. I en optimeringsmodel har vi brug for estimater, der fortæller os noget om fremtiden med en vis sandsynlighed. Når vi sammenligner et F1 lån med et fastforrentet obligationslån og til det formål bruger en kort

[11] Hvis priserne til de forskellige instrumenter er usammenhængende, kan vi tjene frie penge ved for eksempel at købe en underpriset syntetisk put option (sammensat ud fra en future og en call option) og sælge en overpriset standard put option, (se [10], p. 177).

rentemodel, så forudsætter vi, at vi regner med rentemodellens forudsigelse om fremtiden. Det er den eneste måde, vi kan komme med en estimat for, hvad F1 lånet kommer til at koste i gennemsnit, i værste og i bedste fald. Optimeringsmodellen kan derfor ikke komme med fornuftige resultater, hvis den tager en ufornuftig rentemodel som input. Dette forbehold gælder hver gang en rentemodel bruges i en optimeringsmodel.

Kritikere af brugen af en optimeringsmodel inden for låneanbefaling hævder, at rentemodeller ikke kan bruges til at forudse fremtiden, og derfor er anbefalingen fra en optimeringsmodel i princippet lige så god eller lige så dårlig som en hvilken som helst anden anbefaling, idet man kan få en optimeringsmodel til at generere forskellige anbefalinger, alt efter hvilken rentemodel der tages udgangspunkt i. Som svar på dette argument kan vi give følgende begrundelser for brugen af en optimeringsmodel:

- Optimeringsmodellens funktion er at optimere en målfunktion under hensyntagen til givne kriterier og begrænsninger. Rentemodellen skal blot betragtes som et kvalificeret gæt på mulige fremtidige rentescenarier. Alternativet til optimeringsmodellen ville være at anvende konsekvensberegninger for forskellige rentescenarier. Der findes imidlertid kontinuert mange lineære kombinationer af to eller flere forskellige lån, og det er praktisk taget umuligt at foretage konskvensberegninger for alle disse kombinationer. Optimeringsmodellen derimod betragter samtlige scenrier på en gang, og genererer optimale anbefalinger givet fastlagte inputdata, deriblandt en bestemt renteudvikling.

- Ofte bruger vi en optimeringsmodel som et analytisk værktøj til at bekræfte eller afkræfter en formodning om rimeligheden i en beslutning. Vi kan således bruge forskellige rentemodeller som input til optimeringsmodellen og teste derved robustheden i de fundne løsninger. Hvis vi, i en given situation, får den samme løsning for forskellige rentemodeller, har vi styrket vores formodning om rigtigheden af løsningen.

- Selvom rentemodeller, strengt taget, ikke burde bruges til forudsigelse af fremtidens renter, afspejler de ikke desto mindre markedets forventning om en fremtidig rentebevægelse. De mest brugte rentemodeller, som for eksempel BDT–modellen, er i overensstemmelse med den måde, markedet har prisfastsat obligationer med forskellige løbetider. De estimerede renter giver derfor, under normale omstændigheder, et realistisk billede af spændvidden af renterne, især inden for en kort horisont.

Med normale omstændigheder menes de omstændigheder, markedet betragter som afgørende for bevægelsen af renterne. Eftersom lånemarkedet er bygget på obligationerne, og disse betragtes som sikre investeringer i forhold til aktierne, er den eneste form for ekstreme tilfælde, som kan få virkeligheden til at afvige betydeligt fra rentemodellen, en uventet økonomisk fallit på nationalt eller internationalt niveau. I teorien kan man tilsvarende betragte en uventet økonomisk opblomstring, men dette har ikke interesse i et land som Danmark med en solid national økonomi. Det skal bemærkes, at selvom obligationsinvestering er en sikker investering i forhold til aktieinvestering, er der selvfølgelig en renterisiko forbundet med obigationsinvesteringen. Det er denne renterisiko, der bliver afspejlet i rentemodellen. Vi forudsætter i denne rapport, at der ikke forekommer en ekstrem høj rente, der kan være forårsaget af en ekstrem hændelse som krig eller en naturkatastrofe og som ikke bliver fanget i rentemodellen. Hvis låntageren lægger vægt på realisieringsmuligheden af dette ekstreme scenario, burde han optage den sikreste form for lån, som er et fastforrentet obligationslån med indbygget konverteringsret.

Vi er nu klar til at gå videre til optimeringsmodellen, der gradvis bliver opbygget og analyseret i næste kapitel.

Kapitel 4

Optimeringsmodellen

I kapitel (2) og (3) definerede vi den underliggende problemstilling og dannede os en intuitiv forståelse af de deskriptive finansielle modeller, der producerer inputdata til den stokastiske optimeringsmodel, (se diagram 1.1 (side 4)). Nu er vi kommet frem til kapitlet, hvor denne optimeringsmodel trinvis skal udvikles.

Vi udvikler i resten af dette kapitel en stokastisk optimeringsmodel, der trinvis bliver skræddersyet til at afspejle markedets parametre og derved giver et realistisk bud på værdifulde anbefalinger. Disse låneanbefalinger kan, givet fastlagte forudsætninger, betragtes som optimale løsninger.

I afsnit 4.1 beskriver vi det grundlæggende optimeringsproblem uden at betragte en egentlig modelleringsstrategi.

Vi vil få en intuitiv forståelse for stokastisk programmering i afsnit (4.2). Her vil vi formulere en to–stadie stokastisk heltals–model.

Modeludviklingen i dette kapitel er inspireret af optimeringsmodellen beskrevet i Nielsen og Poulsen artiklen, (N&P, [18]). Vi vil i afsnit (4.3) gøre rede for N&P–modellen i sin oprindelige form, inden vi formulerer en multi–stadie optimeringsmodel.

Vi vil dernæst i afsnit (4.4) formulere og løse en multi–stadie stokastisk heltals–model.

Afsnit (4.5) handler om at analysere de forskellige løsningsmetoder, der er anvendt i dette projekt. GAMS ("General Algebraic Modeling Systems") er blevet brugt som modelleringssprog. Til optimal reducering af antallet af scenarier benytter vi et scenario–reducerende modul "SCENRED", (se [11, 12, 13]). Vi viser forskelle mellem en GAMS–formulering og en GAMS/SCENRED–formulering af modellen.

4.1 Beskrivelse af optimeringsproblemet

For at kunne tale om en optimeringsmodel er det nødvendigt at kunne definere en målfunktion. Når vi gerne vil rådgive om det optimale lån, har vi desværre ikke en entydig målfunktion. Vi har valgt mellem enten at minimere lånets omkostninger eller at minimere risici. Når vi ser på låneomkostninger, skal vi beslutte os for en passende *tilbagediskonteringsfaktor*. Og når vi betragter låntagers risici, har vi med to former for risici at gøre, nemlig *likviditetsrisiko* og *formuerisiko*. Vi vil i det følgende se på en beskrivelse af, og overvejelserne omkring disse begreber.

Låneomkostninger

For erhvervslåntagere er det ofte mere relevant at betragte nutidsværdien af de totale omkostninger. Dette er rimeligt, eftersom en virksomhed vægter en tidlig betaling tungere end en betaling langt ude i fremtiden. Nutidsværdibetragtningen vil afspejle denne tidsværdi af penge. Man kan således forestille sig optimale løsninger, der favoriserer låntyper med små betalinger i starten og større betalinger i fremtiden. Dette er ikke nødvendigvis det rigtige valg for en privatkunde, der ikke i samme grad kan eller vil udnytte tidsværdien af penge. Løsningen på dette problem er at lade den enkelte bruger af modellen vælge en passende tilbagediskonteringsfaktor.

Risicibetragtninger

Likviditetsrisiko er den risiko, der er forbundet med ekstra betalinger som følge af stigninger i renten. Et F1–lån må derfor antages at have en høj likviditetsrisko. Et stort fokus på likviditetsrisikoen vil betyde, at de fastforrentede lån bliver favoriseret i en optimalløsning. Formuerisikoen er den risiko, låntageren har i forbindelse med en voldsom kursstigning for de bagvedliggende obligationer. Hvis man eksempelvis optager et 30 årigt fastforrentet obligationslån, hvor man udsteder obligationer til kurs 80, har man en stor formuerisiko. Et lille rentefald betyder en stor stigning i kursen og dermed en forøgelse af ens restgæld i tilfældet af for tidlig indfrielse af lånet, eller en tilsvarende formindskelse af ejendommens friværdi. Hvis man ikke skal indfri lånet utidigt eller optage nyt lån i ejendommens friværdi, behøver man således ikke at bekymre sig om formuerisikoen. Formuerisikoen er nemlig kun uønskelig, hvis den kan omsættes til likviditetsrisiko, og dette sker enten ved en tidlig indfrielse af lånet eller udnyttelsen af ejendommens friværdi.

Der findes ikke et entydigt svar på, hvilket risikomål der skal tages højde for i optimeringsmodellen. Vi vil studere begge typer risici og implementere dem i den stokastiske optimeringsmodel.

Multikriterieoptimering

Hvis vi ikke tager hensyn til ovenstående spørgsmål, får vi ikke meningsfulde løsninger fra optimeringsmodellen. Hvis vi for eksempel minimerer omkostninger uden hensyntagen til likviditetsrisiko, så maksimerer vi ofte risikoen. Og omvendt, hvis vi vil minimere likviditetsrisikoen er det oftest ensbetydende med, at vi maksimerer omkostningerne. At dette "ofte" men ikke "altid" er tilfældet har baggrund i følgende situation:

Under tiden kan det forekomme, at det billigste lån også er det mindst risikofyldte. Dette vil være tilfældet, hvis for eksempel et fastforrentet lån viser sig at være billigere end et rentetilpasningslån uanset den formodede renteudvikling. Med andre ord har vi en invers rentestruktur. Hvis dette forekommer, er det nemt at beslutte sig for valget af lån: et fastforrentet lån ville være at foretrække. Markedsmekanismer vil dog hurtigt sørge for, at denne mulighed forsvinder. I takt med, at flere (alle rationelle) låntagere vælger et fastforrentet lån overfor et rentetilpasningslån, vil (udstedelses–)prisen til de lange obligationer falde og prisen på de korte obligationer stige, og det bliver hurtigt billigere (i hvert fald på kort sigt) at optage et rentetilpasningslån.

I det følgende ser vi på eksempler fra et normalt marked, d.v.s. et marked med en normal rentestruktur, (se skitse 3.1 (side 17)), men optimeringsmodellen har ingen forudsætninger med hensyn til rentestrukturformen.

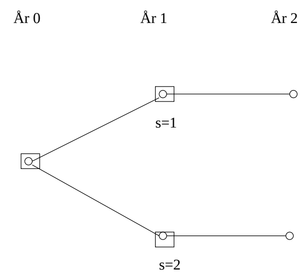
Problemer af denne type, hvor vi har med modstridende interesser at gøre kaldes multikriterieoptimeringsproblemer. I resten af dette kapitel minimerer vi de totale låneomkostninger uden hensyntagen til risici. Vi vil dog i kapitel 5 (side 83) vende tilbage til risicibetragtningerne og vi vil forslå flere metoder til at håndtere de modstridende interesser i optimeringsmodellen.

4.2 En to–stadie stokastisk heltals–model

Stokastisk programmering handler om at optimere en beslutningsproces under usikkerhed over en given tidshorisont. Den simpleste form for et stokastisk program er et to–stadie stokastisk program, hvor man står over for et beslutningsproblem her og nu, men senere får mulighed for at foretage en justerende beslutning alt efter hvilken information man har fået i løbet af den første periode. Justeringsdelen af problemet er kendt under navnet "recourse" problemet, og de tilhørende justerende variable kaldes "recourse" variable.

Det vanskelige ved stokastisk programmering er, at man i sin nuværende beslutning skal tage højde for effekten af sine fremtidige beslutninger på sin nuværende beslutning. Dette indebærer, at man skal kunne tilskrive en sandsynlighedsfordeling til de stokastiske hændelser i problemet. Rentemodellen og prisfastsættelsesmodellen fra kapitel (3) forsyner os med en diskret sandsynlighedsfordeling, som vi vil bruge i resten af denne rapport. For at have vores fokus alene rettet mod optimeringsdelen af problemet, vil vi dog først forudsætte, at estimaterne for obligationsrenter og obligationskurser samt dertil knyttede sandsynligheder er kendte. Senere i afsnit (4.4) vil vi koble inputmodellerne til optimeringsmodellen.

I figur (4.1) ser vi et simpelt scenariotræ, der skitserer udviklingen i en to–periode model. De knuder, der er omskrevet med firkanter, er beslutnings-



Figur 4.1: Et scenario træ for en to–stadie stokastisk programmeringsmodel.

knuderne. Vi skal således vælge mellem et givet antal lån, eller en kombination af disse lån på tidspunkt 0. Derefter på tidspunkt 1 kan vi revidere vores beslutning alt efter hvilket scenario der er blevet realiseret. Det er vigtigt, at beslutningen ved tidspunkt 0 ikke afhænger af, hvilket scenario ved tidspunkt 1 der realiseres. Denne betingelse kaldes Ikke–forudseenhed betingelsen (engelsk: “Non–anticipativity constraint”).

Selvom en to–stadie stokastisk heltals–model er betydelig vanskeligere at løse end en tilsvarende LP (lineær programmering) model, er ideerne bag selve modelleringen af problemet identiske. Idet vi til at starte med ikke bekymrer os om løsningsmetoden, springer vi en forsimplet LP model over og betragter med det samme en mere realistisk model. I afsnit (4.5) og (6) vil vi se på problemet omkring løsningsmetoder, der er til rådighed. Vi starter med en naiv, men intuitiv, “mixed integer” deterministisk ækvivalent formulering af problemet. Følgende konstanter defineres:

c_{i0}: Variable omkostninger forbundet med låntype i i periode 1.[1]
m_{i0}: Faste omkostninger forbundet med optagelsen af låntype i i periode 1.
p_s: Sandsynligheden for scenario s ved tidspunkt 1.
c_{i1s}: Variable omkostninger forbundet med låntype i ved scenario s i periode 2.
m_{i1s}: Faste omkostninger forbundet med optagelsen af låntype i ved scenario s i periode 2.

[1]Læg mærke til at vi henviser til periode 1 som perioden mellem tidspunkt 0 og 1 og periode 2 som perioden mellem tidspunkt 1 og 2.

Endvidere defineres følgende beslutningsvariable:

x_{i0} : procentdel af låntype i optaget i periode 1. $\quad 0 \leq x_{i0} \leq 1$

$$l_{i0} : \begin{cases} 1 & \text{hvis der løber faste omkostninger ved tidspunkt 0, låntype } i. \\ 0 & \text{ellers.} \end{cases}$$

x_{i1s}: procentdel af låntype i optaget ved scenario s i periode 2. $\quad 0 \leq x_{i1s} \leq 1$

$$l_{i1s} : \begin{cases} 1 & \text{hvis der løber faste omkostninger ved tidspunkt 1, låntype } i\text{, scenario } s. \\ 0 & \text{ellers.} \end{cases}$$

Modellen kan nu formuleres som følgende:

$$\min \sum_i c_{i0} x_{i0} + \sum_{is} p_s c_{i1s} x_{i1s} + \sum_i m_{i0} l_{i0} + \sum_{is} p_s m_{i1s} l_{i1s} \tag{4.1}$$

$$\sum_i x_{i0} \geq 1 \tag{4.2}$$

$$l_{i0} - x_{i0} \geq 0 \qquad \forall i \tag{4.3}$$

$$\sum_i x_{i1s} \geq 1 \qquad \forall s \tag{4.4}$$

$$l_{i0} + l_{i1s} - x_{i1s} \geq 0 \qquad \forall is \tag{4.5}$$

$$x_{i0}\ ,\ x_{i1s} \geq 0 \ \ \forall is, \qquad l_{i0}\ ,\ l_{i1s} \in \{0,1\} \ \ \forall is \tag{4.6}$$

Målfunktionen (4.1) er summen af variable og faste omkostninger forbundet med de to perioder. Vi har endnu ikke besluttet os for, om omkostningerne skal tilbagediskonteres til startpunktet. Det bliver diskuteret i afsnit (6). Denne målfunktion repræsenterer en likviditetsrisikoneutral låntager, idet vi betragter et simpelt vægtede gennemsnit af mulige omkostninger i periode 2. Den eneste form for risiko taget i betragtning er likviditetsrisiko, idet vi forudsætter at utidig indfrielse ikke vil forekomme. Vi har nemlig et lån der betales fuldt ud i 2 år.

Begrænsning (4.2) siger, at 100% af lånebeløbet skal dækkes i periode 1. Ligeledes siger begrænsning (4.4), at for hvert scenario i periode 2 skal hele restgældsbeløbet dækkes. Her kunne vi også bruge lig med (=) begrænsninger, men eftersom vi har et minimeringsproblem og låneomkostningerne er positive, er vi sikre på, at disse begrænsninger er bindende og derfor nøjes vi med at bruge den mere relakserede ($\geq$) begrænsningsform.

Begrænsning (4.3) sørger for, at de faste omkostninger forbundet med lån i betales såfremt lån i indgår i vores låneportefølje i periode 1. Begrænsning (4.5) sikrer os, at vi ikke betaler faste omkostninger to gange for det samme lån: Hvis $l_{i0} = 1$, d.v.s. hvis vi har betalt faste omkostninger for lån i i periode 1, så bliver $l_{i1s} = 0$ idet $x_{i1s} \leq 1$ og vi har et minimeringsproblem. Omvendt hvis $l_{i0} = 0$, d.v.s. hvis vi ikke har betalt faste omkostninger for lån i i periode 1, så bliver $l_{i1s} = 1$ hvis $x_{i1s} > 0$. Endelig har vi ikke–negativitets og binære betingelserne (4.6).

Denne model giver en intuitiv forståelse for optimeringsproblemet, men desværre melder to problemer sig umiddelbart:

For det første forudsætter modellen, at man kender de variable omkostninger på forhånd. Man kan imidlertid kun regne de variable omkostninger på forhånd, hvis man kendte al restgæld i alle knuder. Desværre kender vi ikke den præcise restgæld i knuderne ved tidspunkt 1 før vi har bestemt os for valget af et lån. Men den optimale lånestrategi findes kun, hvis vi kender de dertilhørende omkostninger. Ved tidspunkt 0 kender vi lånets *kursværdi*, hvorfor vi for et bestemt type lån, i, kan beregne lånets restgæld og derved den respektive variable omkostning i periode 1, d.v.s. c_{i0} værdierne kan beregnes på forhånd. Problemet er i den anden periode, hvor restgælden afhænger af valget af lån ved tidspunkt 0. For eksempel, hvis man ved tidspunkt 0 optager et lån med kurs tæt ved pari, har man en mindre restgæld ved tidspunkt 1 i forhold til hvis man havde optaget et lån med et større kurstab. Havde man valgt en kombination af disse to lån vil ens restgæld ligge imellem de to restgæld. Det svarer til, at vi har kontinuert mange mulige restgæld ved tidspunkt 1. Og det kræver, at vi skal beregne kontinuert mange c_{i1s} værdier svarende til kombinationsmulighederne. Dernæst skal vi sørge for at de kontinuert mange låneomkostninger forbindes med de respektive lånekombinationer. Dette er ikke praktisk muligt. Forudsætningen for ens restgæld i hver knude er heller ikke acceptabelt, idet restgældene i starten af anden periode kan afvige betydeligt fra hinanden for de forskellige lånekombinationer.

Det andet problem i modellen er en følge af det første problem. Det går på, at man på forhånd skal fastlægge den endelige restgæld ved horisonten. Dette er nødvendigt for at kunne beregne de variable omkostninger der skal bruges som inputdata. Vi har således ikke mulighed for at betragte slutrestgælden som en variabel, der kan optimeres på. Såfremt vi kun ser på de tilfælde, hvor lånet skal afvikles op til horisonten, er dette ikke en begrænsning. Men hvis vi er interesserede i at integrere formuebetragtninger[2] i vores model, har vi igen et problem med generering af inputdata.

Meget tyder på, at vi skal lade restgælden afhænge af valget af lån, d.v.s. restgælden skal betragtes som en variabel i optimeringsmodellen. I det følgende vil vi belyse hvordan begge problemer kan løses med indførelse af en balance–begrænsning samt en betalingsbegrænsning, som involverer restgælden. Med indførelse af disse nye begrænsninger kommer optimeringsmodellen tættere på det underliggende finansieringsproblem. Med andre ord får vi nu brug for at involvere obligationspriser og renter direkte i optimeringsmodellen. Vi betragter en betalingsstrøm fra år 0 til år 2. Vi henviser til lånets restgæld som RG og afdrag som A. Følgende balance–ligning skal nødvendigvis gælde:

$$RG_{i0} - A_{i0} = RG_{i1s} \quad \forall i, s \quad . \tag{4.7}$$

Vi kender ikke restgælden RG_{i0} direkte ved tidspunkt 0 før vi har valgt et lån eller en kombination af lån. Vi ved derimod hvor mange penge der skal

[2]Vi vil diskutere formuerisikoen i detaljer i afsnit 5.2.4 (side 95)

skaffes, eksempelvis 1000.000 DKK. Vi kender ligeledes de forskellige kurser på de forskellige obligationer. Idet vi definerer kursen som $0 < k_{i0} \leq 1$, kan vi nu tilføje startbetingelsen:

$$\sum_i RG_{i0} \cdot k_{i0} \geq 1000.000 \quad . \tag{4.8}$$

Denne betingelse siger, at kursværdien skal være mindst på 1000.000 DKK. Igen kan vi bruge (=) tegn i stedet for ($\geq$) tegn. Men eftersom vi har et minimeringsproblem med positive omkostninger, er betingelsen bindende og derfor kan vi nøjes med denne mere relakserede betingelse. Vi er også interesserede i at afspejle konverteringsmuligheden i balance–ligningen (4.7). Til dette formål skal vi introducere en købevariabel, P_{i1s}, som angiver mængden af købte (indfriede) obligationsenheder af obligation i på tidspunkt 1 for alle scenarier s. Ligeledes indfører vi salgsvariablen S_{i1s}, som er mængden af solgte obligationsenheder af obligation i ved scenario s på tidspunkt 1. Vores nye balanceligning bliver:

$$RG_{i0} - A_{i0} - P_{i1s} + S_{i1s} = RG_{i1s} \quad \forall i, s \ . \tag{4.9}$$

Vi skal også sørge for at de penge vi køber obligationer for kommer fra vores salg af andre obligationer. Dette gøre vi ved at introducere følgende betalingsbegrænsning:

$$\sum_i (k_{i1s} \cdot S_{i1s}) = \sum_i (Callk_{i1s} \cdot P_{i1s}) \quad \forall s \quad , \tag{4.10}$$

hvor vi definerer k_{i1s} som kursen for obligation i for scenario s på tidspunkt 1. Ligeledes definerer vi $Callk_{i1s} = \min\{1, k_{i1s}\}$, som indfrielseskursen for den konverterbare obligation i ved tidspunkt 1, scenario s. Her har vi implicit forudsat, at vi har alene med konverterbare obligationer at gøre, d.v.s. at låntageren kan til enhver tid indfri sit lån til kurs pari. Dette er dog ikke en begrænsning i modellen, eftersom vi kan sætte $Callk_{i1s} = k_{i1s}$ for alle inkonverterbare obligationer.

Læg mærke til, at i ligning (4.9) ganger vi S_{i1s} og P_{i1s} med 1 (svarende til kurs pari), hvorimod i ligning (4.10) ganger vi disse variable med henholdsvis k_{i1s} og $Callk_{i1s}$. Dette kan begrundes med, at i ligning (4.9) ser vi på det restgældsmæssige beløb der lægges til eller trækkes fra restgælden. I ligning (4.10) derimod er vi interesserede i det aktuelle beløb (det kursværdimæssige beløb) der kan skaffes ved salget af S_{i1s} obligationsenheder, og tilsvarende det aktuelle beløb der skal betales når vi indfrier P_{i1s} obligationsenheder.

På tidspunkt 0 kan vi kun sælge (udstede) obligationer, idet vi endnu ikke har en restgæld der kan indfries ved at købe obligationer. D.v.s.:

$$RG_{i0} = S_{i0} \quad \forall i. \tag{4.11}$$

Restgælden ved tidspunkt 1, RG_{i1s}, skal afdrages helt i anden periode:

$$RG_{i1s} - A_{i1s} = 0 \quad \forall i, s. \tag{4.12}$$

Næste skridt går ud på at etablere en kobling mellem afdragene A_{i0} og hovedstolene HS_{i0} i balanceligningen (4.9). Annuitetsformlen giver:

$$Y_j = RG_{optagelse} \frac{r_j}{1-(1+r_j)^{-N}} \quad ,$$

hvor Y_j er den faste ydelse til tid j, r_j er renten og N er lånehorisonten. Denne ydelse deles op i en rentebetaling $RB_j = r_j \cdot RG_j$ og afdrag $A_j = Y_j - RB_j$. Nu kan vi beskrive A_{i0} og A_{i1s} som en funktion af de tilsvarende restgæld og renter:

$$A_{i0} = RG_{i0}\Big[\frac{r_{i0}}{1-(1+r_{i0})^{-2}} - r_{i0}\Big] \quad \forall i \tag{4.13}$$

$$A_{i1s} = RG_{i1s}\Big[\frac{r_{i1s}}{1-(1+r_{i1s})^{-1}} - r_{i1s}\Big] \quad \forall i, s. \tag{4.14}$$

Nu har vi koblet de forskellige obligationsrenter r_{i0} og r_{i1s} til modellen og er klar til at formulere vores reviderede 2–stadie stokastiske heltalsmodel:

$$\min \quad \tau_0 \cdot B_0 + \sum_s p_s \cdot \tau_s \cdot B_{1s} \tag{4.15}$$

$$\sum_i k_{i0} \cdot RG_{i0} \geq 1000.000 \tag{4.16}$$

$$RG_{i0} = S_{i0} \tag{4.17}$$

$$RG_{i0} - A_{i0} - P_{i1s} + S_{i1s} = RG_{i1s} \qquad \forall i, s \tag{4.18}$$

$$\sum_i (k_{i1s} \cdot S_{i1s}) = \sum_i (Callk_{i1s} \cdot P_{i1s}) \qquad \forall s \tag{4.19}$$

$$RG_{i1s} - A_{i1s} = 0 \qquad \forall i, s \tag{4.20}$$

$$A_{i0} = RG_{i0}\Big[\frac{r_{i0}}{1-(1+r_{i0})^{-2}} - r_{i0}\Big] \qquad \forall i \tag{4.21}$$

$$A_{i1s} = RG_{i1s}\Big[\frac{r_{i1s}}{1-(1+r_{i1s})^{-1}} - r_{i1s}\Big] \qquad \forall i, s \tag{4.22}$$

$$B_0 = \sum_i \Big(A_{i0} + (1-\gamma)RG_{i0} \cdot r_{i0} + (1-\beta)RG_{i0} \cdot b_{i0} + \eta \cdot S_{i0} + m_{i0} \cdot l_{i0}\Big) \tag{4.23}$$

$$B_{1s} = \sum_i \Big(A_{i1s} + (1-\gamma)RG_{i1s} \cdot r_{i1s} + (1-\beta)RG_{i1s} \cdot b_{i1s} + \eta \cdot (S_{i1s} + P_{i1s}) + m_{i1s} \cdot l_{i1s}\Big) \qquad \forall s \tag{4.24}$$

$$BigM \cdot l_{i0} - S_{i0} \geq 0 \qquad \forall i \tag{4.25}$$

$$BigM \cdot l_{i1s} - S_{i1s} \geq 0 \qquad \forall is \tag{4.26}$$

$$RG_{i0}\ ,\ RG_{i1s}\ ,\ S_{i0}\ ,\ S_{i1s}\ ,\ P_{i1s} \geq 0 \qquad \forall is \tag{4.27}$$

$$l_{i0}\ ,\ l_{i1s} \in \{0,1\} \qquad \forall is \tag{4.28}$$

Ligningerne (4.21) til (4.24) definerer "bogholderi" variablene A for afdrag og B for betaling. Konstanten γ er procentsatsen for skattefradraget fra rentebetalingen.[3] Bidragssatserne er b_{i0} henholdsvis b_{i1s} for de to perioder og β er procentsatsen for skattefradraget fra bidragbetalingerne. Konstanten η er kurtagen på obligationshandlen. Hver gang der bliver solgt eller indfriet en obligation betales der en procentdel (0,15%) af handlens beløb som kurtage. "Bogholderi" variablene bruges, så resten af modellen kan skrives mere overskueligt. Vi bruger for eksempel B_0 og B_{is} til at definere målfunktionen (4.15), som har til formål at minimere den samlede betaling, bestående af betalingen i periode 1 og den gennemsnitlige betaling i periode 2. Læg mærke til, at vi har introduceret koefficienterne, τ_0 og τ_s, som tilbagediskonteringsfaktorene. Disse er defineret som $\tau_0 = \frac{1}{1+d_0}$ og $\tau_s = \frac{1}{(1+d_s)^2}$, hvor d_0 og d_s er brugerens valgte diskonteringsrente.

[3]For tiden er skattefradragssatsen for private ca. 0,32.

Dynamikken i modellen kan deles op i en startfase, en mellemfase og en slutfase. I startfasen sikrer begrænsningen (4.16) os, at der er penge nok til at dække vores initiale behov (1000.000 DKK eller et hvilket som helst andet beløb) og ligning (4.17) initialiserer vores restgæld for hver obligation.

I mellemfasen sikrer ligning (4.18) os, at restgælden i næste periode er lig med restgælden i første periode minus afdrag og det indfriede værdi plus nye solgte obligationer. Ligning (4.19) siger, at der ikke tilføres systemet penge udefra.

I slutfasen siger ligning (4.20), at vi skal betale hele restgælden i den sidste periode.

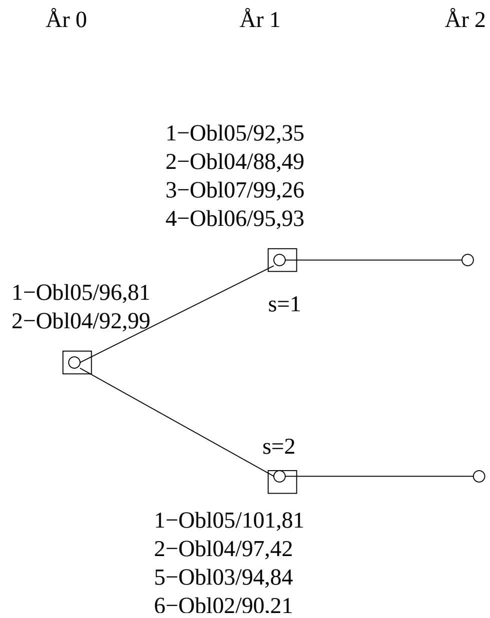
Begrænsningerne (4.25 og 4.26) har samme formål som ligninger (4.3 og 4.5) fra den tidligere model. Der er dog to forskelle: For det første ganger vi nu vores binære variable med en stor konstant[4], idet vi ikke længere betragter procentdelen af et valgt lån, x_{i0} og x_{i1s}, men derimod betragter vi den egentlige mængde obligationer der bliver udstedt, nemlig S_{i0} og S_{i1s}. Den anden forskel er, at begrænsning (4.26) bliver simplere end (4.5), idet S_{i1s} er kun positiv, hvis vi udsteder nye obligationer, i modsætning til x_{i1s}, fra den tidligere model, som er positiv, så længe vi bibeholder lån type i. Endelig har vi ikke–negativitets– og heltalsbegrænsningerne (4.27 og 4.28).

Vi har med vilje ikke skrevet modellen mere kompakt m.h.t. tidsvariablen, idet en to–stadie model i åben form m.h.t. tiden viser dynamikken i problemet mere tydeligt. Når vi i afsnit (4.4) ser på en multi–stadie model, introducerer vi en mere kompakt formulering.

Vi afslutter dette afsnit med et eksempel, som bliver videreudviklet i de kommende afsnit i takt med modellens trinvise udvikling. Vi betragter scenariotræet i figur 4.2 (side 55), hvor ved år 1 er der sandsynlighed $\frac{1}{2}$ for scenario 1 og sandsynlighed $\frac{1}{2}$ fro scenario 2. Obligationerne er nummereret fra 1 til 6, hvor $1-Obl05/96,81$ er, for eksempel, en obligation med 5% rente og kurs på $96,81$.

Ved år 0 skal vi skaffe DKK 1.000.000 og vi har kun to lange obligationer at vælge imellem. Forfaldsdatoen for obligationerne forudsættes at være langt nok ude i fremtiden til at vi kan beholde lånet indtil horisonten. Læg mærke til, at selvom renterne på obligationerne 1 og 2 er uændret ved år 1, så har kurserne ændret sig. Det svarer til, at vi betaler stadig den samme rente men hvis lånet skal indfries, så sker det til kurset ved år 1, dog højest kurs pari, idet obligationerne forudsættes at være konverterbare. Ligeledes har vi fået tilføjet 2 nye obligationer, 3 og 4, ved år 1 scenario 2 og obligationerne 5 og 6 ved år 1 scenario 3.

[4]Vi kan nøjes med at gange variablene med en værdi, der er større end $\sum_i RG_{i0}$. Idet vi kender ikke den præcise restgæld før vi har valgt en strategi, kan vi vælge en værdi som er 10-20% større end hovedstolen. I vores eksempel kan vi nøjes med at sætte $BigM = 1.200.000$.



Figur 4.2: Et scenariotræ for en to–stadie stokastisk programmeringsmodel, med angivelse af et univers af obligationer.

Lånet skal betales over 2 år og det kan refinansieres ved år 1. Refinansieringen er tilladt både som ned–konvertering, d.v.s. som konvertering til et lån med lavere rente eller en op–konvertering, d.v.s. som konvertering til et lån med højere rente. Ved år 1, scenario 1 har vi valget mellem at op–konvertere til obligationerne 3 og 4 eller at beholde år 0's valgte låneportefølje. Ved scenario 2 kan vi enten ned–konvertere til obligationerne 5 og 6 eller beholde år 0's låneportefølje.

Renterne, r_{i0} og r_{i1s}, kurserne, k_{i0} og k_{i1s} samt konverteringskurserne, $Callk_{i1s} = \min\{1, k_{i1s}\}$, kan sammenfattes i følgende tabel:

	Renter				Kurser				Konverteringskurser		
Lån \ År , Scenario	0	1 , 1	1 , 2		0	1 , 1	1 , 2		0	1 , 1	1 , 2
1	0,05	0,05	0,05		0,9681	0	0		–	0,9235	1
2	0,04	0,04	0,04		0,9299	0	0		–	0,8849	0,9742
3	–	0,07	–		–	0,9926	–		–	–	–
4	–	0,06	–		–	0,9593	–		–	–	–
5	–	–	0,03		–	–	0,9488		–	–	–
6	–	–	0,02		–	–	0,9021		–	–	–

Tabel 4.1: Renter, kurser og konverteringskurser for de 6 obligationer i vores låneunivers.

Læg mærke til, at ved år 1 scenario 1 og 2 har vi med vilje sat kurserne for obligationerne 1 og 2 til 0. Dette skyldes, at vi ikke vil tillade at indfri en obligation til kurs pari for dernæst at optage lån i den samme obligation for et kurs der er højere end pari og på den måde tjene risikofrie penge. Når kurserne bliver sat til 0 ved disse knuder, tvinger vi modellen til ikke at sælge obligationerne 1 og 2, eftersom der ikke kommer noget kursværdi ud af disse handler.

For konverteringskurser giver det kun mening at betragte obligationerne 1 og 2, idet disse er de eneste obligationer, der kan indfries ved de to scenarier i år 1.

Vi sætter skattefradragssatsen for rentebetaling, $\gamma = 0,32$, og for bidragsbetaling $\beta = 0,32$. Bidragsprocenten sætter vi til $b_{i0} = b_{i1s} = 0,005$ og tilbage diskonteringsfaktoren sættes til $\tau_0 = \tau_s = 1$, d.v.s. ingen tilbagediskontering. Vi forestiller os endvidere, at de faste etableringsomkostninger er $m_{i0} = 10.000$ og de faste omlægningsomkostninger er $m_{i1s} = 1000$. Konstanten $BigM$ sættes til $BigM = 1.500.000$. Modellen kan nu skrives i åben form som følgende:

```
---- Totale omkostninger: Objektfunktionen

MINIMER  Z = B(tid0) + 0.5*B(tid1,Scen1) + 0.5*B(tid1,Scen2);

---- EQ1: Hele lånet skal dækkes ved år 0

0.9681*RG(laan1,tid0) + 0.9299*RG(laan2,tid0) >= 1000000;

---- EQ2: Restgælden ved år 0 svarer til de solgte obligationer

RG(laan1,tid0) = S(laan1,tid0);
RG(laan2,tid0) = S(laan2,tid0);
RG(laan3,tid0) = 0;
RG(laan4,tid0) = 0;
RG(laan5,tid0) = 0;
RG(laan6,tid0) = 0;

---- EQ3: Balanceligninger

RG(laan1,tid0) - A(laan1,tid0) - P(laan1,tid1,Scen1)
               + S(laan1,tid1,Scen1) = RG(laan1,tid1,Scen1);
RG(laan1,tid0) - A(laan1,tid0) - P(laan1,tid1,Scen2)
               + S(laan1,tid1,Scen2) = RG(laan1,tid1,Scen2);
RG(laan2,tid0) - A(laan2,tid0) - P(laan2,tid1,Scen1)
               + S(laan2,tid1,Scen1) = RG(laan2,tid1,Scen1);
RG(laan2,tid0) - A(laan2,tid0) - P(laan2,tid1,Scen2)
               + S(laan2,tid1,Scen2) = RG(laan2,tid1,Scen2);
RG(laan3,tid0) - A(laan3,tid0) - P(laan3,tid1,Scen1)
               + S(laan3,tid1,Scen1) = RG(laan3,tid1,Scen1);
RG(laan3,tid0) - A(laan3,tid0) - P(laan3,tid1,Scen2)
               + S(laan3,tid1,Scen2) = RG(laan3,tid1,Scen2);
RG(laan4,tid0) - A(laan4,tid0) - P(laan4,tid1,Scen1)
               + S(laan4,tid1,Scen1) = RG(laan4,tid1,Scen1);
RG(laan4,tid0) - A(laan4,tid0) - P(laan4,tid1,Scen4)
               + S(laan4,tid1,Scen2) = RG(laan4,tid1,Scen2);
```

```
RG(laan5,tid0) - A(laan5,tid0) - P(laan5,tid1,Scen1)
               + S(laan5,tid1,Scen1) = RG(laan5,tid1,Scen1);
RG(laan5,tid0) - A(laan5,tid0) - P(laan5,tid1,Scen2)
               + S(laan5,tid1,Scen2) = RG(laan5,tid1,Scen2);
RG(laan6,tid0) - A(laan6,tid0) - P(laan6,tid1,Scen1)
               + S(laan6,tid1,Scen1) = RG(laan6,tid1,Scen1);
RG(laan6,tid0) - A(laan6,tid0) - P(laan6,tid1,Scen2)
               + S(laan6,tid1,Scen2) = RG(laan6,tid1,Scen2);

---- EQ4: Betalingsstrømligninger

0.9926*S(laan3,tid1,Scen1) + 0.9493*S(laan4,tid1,Scen1) =
    0.9235*P(laan1,tid1,Scen1) + 0.8849*P(laan2,tid1,Scen1);

0.9984*S(laan5,tid1,Scen2) + 0.9521*S(laan6,tid1,Scen2) =
    1.0000*P(laan1,tid1,Scen2) + 0.9642*P(laan2,tid1,Scen2);

---- EQ5: Hele restgælden skal betales i anden periode

RG(laan1,tid1,Scen1) - A(laan1,tid1,Scen1) = 0;
RG(laan1,tid1,Scen2) - A(laan1,tid1,Scen2) = 0;
RG(laan2,tid1,Scen1) - A(laan2,tid1,Scen1) = 0;
RG(laan2,tid1,Scen2) - A(laan2,tid1,Scen2) = 0;
RG(laan3,tid1,Scen1) - A(laan3,tid1,Scen1) = 0;
RG(laan3,tid1,Scen2) - A(laan3,tid1,Scen2) = 0;
RG(laan4,tid1,Scen1) - A(laan4,tid1,Scen1) = 0;
RG(laan4,tid1,Scen2) - A(laan4,tid1,Scen2) = 0;
RG(laan5,tid1,Scen1) - A(laan5,tid1,Scen1) = 0;
RG(laan5,tid1,Scen2) - A(laan5,tid1,Scen2) = 0;
RG(laan6,tid1,Scen1) - A(laan6,tid1,Scen1) = 0;
RG(laan6,tid1,Scen2) - A(laan6,tid1,Scen2) = 0;

---- EQ6: Definition af afdrag for de to perioder

A(laan1,tid0) = 0.48780487804878*RG(laan1,tid0);
A(laan2,tid0) = 0.49019607843137*RG(laan2,tid0);
A(laan3,tid0) = 0;
A(laan4,tid0) = 0;
A(laan5,tid0) = 0;
A(laan6,tid0) = 0;

A(laan1,tid1,Scen1) = 1*RG(laan1,tid1,Scen1);
A(laan1,tid1,Scen2) = 1*RG(laan1,tid1,Scen2);
A(laan2,tid1,Scen1) = 1*RG(laan2,tid1,Scen1);
A(laan2,tid1,Scen2) = 1*RG(laan2,tid1,Scen2);
A(laan3,tid1,Scen1) = 1*RG(laan3,tid1,Scen1);
A(laan3,tid1,Scen2) = 1*RG(laan3,tid1,Scen2);
A(laan4,tid1,Scen1) = 1*RG(laan4,tid1,Scen1);
A(laan4,tid1,Scen2) = 1*RG(laan4,tid1,Scen2);
A(laan5,tid1,Scen1) = 1*RG(laan5,tid1,Scen1);
A(laan5,tid1,Scen2) = 1*RG(laan5,tid1,Scen2);
A(laan6,tid1,Scen1) = 1*RG(laan6,tid1,Scen1);
A(laan6,tid1,Scen2) = 1*RG(laan6,tid1,Scen2);
```

```
---- EQ7: Definition af betaling for de tre knuder

B(tid0) = 0.0374*RG(laan1,tid0) + 0.0306*RG(laan2,tid0) + 0.6834*RG(laan3,tid0)
        + 0.6834*RG(laan4,tid0) + 0.6834*RG(laan5,tid0) + 0.6834*RG(laan6,tid0)
        + A(laan1,tid0) + A(laan2,tid0) + A(laan3,tid0) + A(laan4,tid0)
        + A(laan5,tid0) + A(laan6,tid0) + 10000*L(laan1,tid0)
        + 10000*L(laan2,tid0) + 10000*L(laan3,tid0) + 10000*L(laan4,tid0)
        + 10000*L(laan5,tid0) + 10000*L(laan6,tid0);

B(tid1,Scen1) = 0.0374*RG(laan1,tid1,Scen1) + 0.0306*RG(laan2,tid1,Scen1)
              + 0.0442*RG(laan3,tid1,Scen1) + 0.0374*RG(laan4,tid1,Scen1)
              + 0.6834*RG(laan5,tid1,Scen1) + 0.6834*RG(laan6,tid1,Scen1)
              + A(laan1,tid1,Scen1) + A(laan2,tid1,Scen1) + A(laan3,tid1,Scen1)
              + A(laan4,tid1,Scen1) + A(laan5,tid1,Scen1) + A(laan6,tid1,Scen1)
              + 1000*L(laan1,tid1,Scen1) + 1000*L(laan2,tid1,Scen1)
              + 1000*L(laan3,tid1,Scen1) + 1000*L(laan4,tid1,Scen1)
              + 1000*L(laan5,tid1,Scen1) + 1000*L(laan6,tid1,Scen1);

B(tid1,Scen2) = 0.0374*RG(laan1,tid1,Scen2) + 0.0306*RG(laan2,tid1,Scen2)
              + 0.6834*RG(laan3,tid1,Scen2) + 0.6834*RG(laan4,tid1,Scen2)
              + 0.0306*RG(laan5,tid1,Scen2) + 0.0238*RG(laan6,tid1,Scen2)
              + A(laan1,tid1,Scen2) + A(laan2,tid1,Scen2) + A(laan3,tid1,Scen2)
              + A(laan4,tid1,Scen2) + A(laan5,tid1,Scen2) + A(laan6,tid1,Scen2)
              + 1000*L(laan1,tid1,Scen2) + 1000*L(laan2,tid1,Scen2)
              + 1000*L(laan3,tid1,Scen2) + 1000*L(laan4,tid1,Scen2)
              + 1000*L(laan5,tid1,Scen2) + 1000*L(laan6,tid1,Scen2);

---- EQ8: Faste etableringsomkostninger

1500000*L(laan1,tid0) - S(laan1,tid0) >= 0;
1500000*L(laan2,tid0) - S(laan2,tid0) >= 0;
1500000*L(laan3,tid0) - S(laan3,tid0) >= 0;
1500000*L(laan4,tid0) - S(laan4,tid0) >= 0;
1500000*L(laan5,tid0) - S(laan5,tid0) >= 0;
1500000*L(laan6,tid0) - S(laan6,tid0) >= 0;

---- EQ9: Faste omlægningsomkostninger

1500000*L(laan1,tid1,Scen1) - S(laan1,tid1,Scen1) >= 0;
1500000*L(laan1,tid1,Scen2) - S(laan1,tid1,Scen2) >= 0;
1500000*L(laan2,tid1,Scen1) - S(laan2,tid1,Scen1) >= 0;
1500000*L(laan2,tid1,Scen2) - S(laan2,tid1,Scen2) >= 0;
1500000*L(laan3,tid1,Scen1) - S(laan3,tid1,Scen1) >= 0;
1500000*L(laan3,tid1,Scen2) - S(laan3,tid1,Scen2) >= 0;
1500000*L(laan4,tid1,Scen1) - S(laan4,tid1,Scen1) >= 0;
1500000*L(laan4,tid1,Scen2) - S(laan4,tid1,Scen2) >= 0;
1500000*L(laan5,tid1,Scen1) - S(laan5,tid1,Scen1) >= 0;
1500000*L(laan5,tid1,Scen2) - S(laan5,tid1,Scen2) >= 0;
1500000*L(laan6,tid1,Scen1) - S(laan6,tid1,Scen1) >= 0;
1500000*L(laan6,tid1,Scen2) - S(laan6,tid1,Scen2) >= 0;
```

Før vi løser problemet bemærker vi, at EQ5, d.v.s. de ligninger der sørger for, at hele restgælden skal betales i anden periode, er overflødige. Disse ligninger er nemlig indeholdt i EQ6 som definition af afdrag for periode 2. I de følgende optimeringsmodeller, der bliver udviklet i denne rapport, dropper vi derfor EQ5.

Når vi løser problemet ved hjælp af Cplex[5], får vi $Z = 1.086.113, 5853$ som optimal objektværdi og følgende løsning:

```
----    118 VARIABLE Sale.L  Sale variable

        tid0       tid1.Scen1

laan1 1032951.141
laan3             492241,071


----    118 VARIABLE P.L  Purchase variable

       tid1.Scen1

laan1  529072,536


----    118 VARIABLE L.L

            tid0      tid1.Scen1

laan1         1
laan3                     1


----    118 VARIABLE RG.L  Outstanding debt

          tid0     tid1.Scen1   tid1.Scen2

laan1 1032951.141              529072.536
laan3             492241.071
```

Låntageren skal derfor optage hele lånet i obligation 1 ved år 0. Ved år 1 skal han/hun op–konvertere lånet til obligation 3 hvis scenario 1 indtræffer og beholde obligation 1 hvis scenario 2 indtræffer. En kontrol af løsningen viser, at løsningen er mulig og i øvrigt i overensstemmelse med finansieringsargumenter. Givet vores forudsætninger, er løsningen tilmed optimal.

Vi er nu klar til at analysere N&P optimeringsmodellen, (se N&P, [18]), i sin originale form.

[5]Her nøjes vi med at præsentere løsningen. I de næste afsnit vil vi beskrive programmeringsaspekterne i GAMS, og diskutere valget af løsere ("solvers").

4.3 N&P optimeringsmodellen

Optimeringsmodellen i N&P artiklen, (N&P, [18]) er en multi–stadie lineær model, som afspejler den grundlæggende dynamik i låneanbefalingsproblemet.

N&P artiklen er, så vidt jeg ved, den eneste artikel, der handler specifikt om optimering af låneporteføljer set fra låntagerside. Modellen i artiklen giver dog kun en overordnet beskrivelse af problematikken, og den kan ikke betragtes som en funktionel model. Det overordnede formål med dette eksamensprojekt er at belyse låneanbefalingsproblemet, med udgangspunkt i N&P artiklen, og videreudvikle N&P modellen til en fuld funktionel model.

Vi vil i det følgende gøre rede for N&P modellen og kommentere fejl og de manglende informationer.

4.3.1 Den principielle balanceligning

Den centrale del af N&P modellen er en balanceligning, som beskriver udvikling af restgælden i scenariotræet. Omsat til vores notation fra sidste afsnit ser N&Ps balanceligning således ud:

$$RG_{ia} - A_{ia} - P_{is} - Q_{is} + S_{is} = RG_{ik} \quad \forall i, k,$$

hvor a symboliserer en forælderknude og s en barnknude i scenriotræet. Vores balanceligning for 2–stadie modellen (ligning 4.18 (side 53)) er en modificering af denne balanceligning. Den eneste forskel mellem ligning 4.18 (side 53) og N&P ligningen er brugen af variablene P_{is} og Q_{is}. N&P forklarer, at P_{is} er købsvariablen for opkonvertering og Q_{is} er købsvariablen for nedkonvertering. Artiklen forklarer ikke, hvorfor det er nødvendigt at bruge to særskilte købsvariable, ligesom der ikke findes en betalingsstrømligning, der sikrer, at pengene, der skal finansiere Q_{is}, kommer indefra systemet. Det bliver nævnt, at der er brug for en betalingsstrømligning uden at præcisere, hvordan en sådan ligning skal se ud. Vores betalingsstrømligning 4.19 (side 53) er et bud på, hvordan vi sikrer, at der ikke tilføres balanceligningen penge udefra.

Eftersom der ikke umiddelbart findes nogle grunde til at bruge 2 købsvariable i balanceligningen, vil vi, også i det følgende, nøjes med købsvariablen P. Hvis renterne stiger, falder kurserne og givet, at det er optimalt at opkonvertere, kan P bruges til at opkonvertere. Ligeledes hvis renterne falder, stiger kurserne, og her kan den samme P bruges til nedkonvertering. Opkonvertering og nedkonvertering betyder blot, med balanceligningens øjemed, at værdien af variabel P stiger så værdien af restgæld i næste knude i stien for den pågældende obligation bliver mindre end ellers.

4.3.2 Beregning af betalinger

N&P definerer den totale betaling ved knude s, omsat til vores notation, som følgende:

$$B_s = \sum_i \Big(A_{is} + (1-\gamma) \cdot RG_{is} \cdot r_{is} - (1-\lambda) \cdot k_{is} \cdot S_{is} + Q_{is} + CallK_{is} \cdot P_{is} \Big) \qquad \forall s.$$

De første to led, d.v.s. afdrag og rentebetaling fratrukket skattefradrag, er det samme som vi så i vores definition af betaling i ligning 4.24 (side 53). De andre led kræver en forklaring. Hvis vi omskriver de resterende led får vi

$$Restled_s = \sum_i \Big(\lambda \cdot k_{is} \cdot S_{is} - k_{is} \cdot S_{is} + Q_{is} + CallK_{is} \cdot P_{is} \Big)$$

Det første led i restleddet svarer til transaktionsomkostninger for låneomlægning. I virkeligheden er der både faste og variable omkostninger forbundet med omlægninger, men N&P ser bort fra de faste transaktionsomkostninger for at kunne formulere en stokastisk LP model. Ved at bruge en procentsats λ, der er større end virkelighedens variable omlægningsomkostninger, vil de kompensere for effekten af faste omlægningsomkostninger. Problemet er imidlertid, at ved at bruge en fast procentsats λ kræver de en stor omlægningsomkostning for de første år, og i takt med, at restgælden bliver mindre med årene, kræver de en mindre omlægningsomkostning. Dette ville favorisere sene omlægninger p.g.a. deres relativ lave transaktionsomkostning. Ideen kan dog bruges med lidt justering. I stedet for at bruge λ kan vi bruge λ_s, som vi tilskriver stigende værdier som funktion af tiden. Vi skal også huske, at vi skal tilpasse værdien af parameteret λ_s for hvert enkelt problem, alt efter størrelsen af hovedstolen og længden af løbetiden. I denne rapport bruger vi en stokastisk binær model, som eksplicit tager højde for faste transaktionsomkostninger.

De sidste 3 led i restleddet har ingen betydning og kan droppes. Dette kan begrundes med, at den nye kursværdi skal lige præcis dækker indfrielsesbeløbet. Dette er faktisk vores betalingsstrøm i ligning 4.19 (side 53). Hvis vi medtager Q_{is} som en eksplicit købsvariabel for nedkonvertering, skal det gælde at:

$$\sum_i (k_{is} \cdot S_{is}) = \sum_i (Callk_{is} \cdot P_{is} + Q_{is}) \qquad \forall s$$

En anden detalje er, at bidragsbetaling, som er en procentvis betaling af restgælden, er blevet glemt. Et forslag til et forbedret version af N&P knudebetalinger kunne se således ud:

$$B_s = \sum_i \Big(A_{is} + (1-\gamma) \cdot RG_{is} \cdot r_{is} + (1-\beta) RG_{is} \cdot b_{is} + \lambda_s \cdot S_{is} \Big) \quad \forall s.$$

4.3.3 Målfunktioner

Der er to formuleringer af målfunktionen i N&P artiklen. Den første er baseret på den samme gennemsnitlige betragtning som vi så i målfunktionen 4.15 (side 53).

Den anden formulering er baseret på maksimering af en konveks nyttefunktion. Objektfunktionen formuleres som:

$$\max \sum_{s=1}^{S} p_s \cdot \log(\tau_s \cdot (B^{maks} - B_s)),$$

hvor p_s er sandsynligheden ved knude s, τ_s er tilbagdiskonteringsfaktoren, B^{maks} er en maksimal budget til rådighed, og B_s er betalingen ved knude s. Målet er nu at maksimere budgetoverskuddet. Denne form for nyttefunktion bruges til at tildele de store besparelser en relative mindre nytte end de små besparelser. Ideen er, at der typisk er forbundet en større risiko for tab, når indtjeningmuligheden er størst. Vi vil vende tilbage til denne nyttefunktion i afsnit 5.2 (side 91).

4.3.4 kommentarer til N&P optimeringsmodellen

N&P modellen, på trods af de nævnte fejl og mangler, formår at belyse grundideerne for modellering af låneanbefalingsproblemet. Det er læserens opgave at regne ud, hvordan de ikke–formulerede begrænsninger, såsom betalingsstrømligningen og startbetingelserne, skal formuleres.

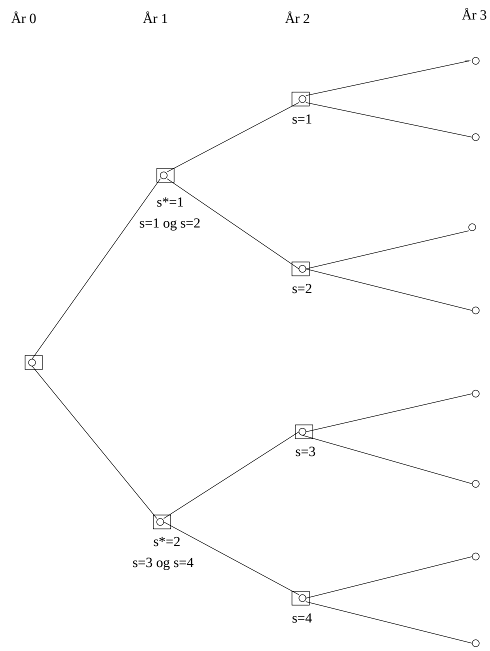
Meningen med artiklen har ikke været at dokumentere en fuld funktionel model, der afspejler samtlige forhold på lånemarkedet. Og eftersom der ikke findes andre offentlig tilgængelige artikler om låneanbefalingsproblemet, skal N&P artiklen betragtes som en værdifuld førstehånds arbejde med introduktion og analyse af låneanbefalingsproblemet.

I næste kapitel vil vi se på en detaljeret gennemgang af nogle af de centrale emner omkring multi–stadie stokastisk modeludvikling samt selve udviklingen af låneanbefalingsmodellen.

4.4 En multi–stadie stokastisk heltals–model

Vi definerer startpunktet som tid 0, og vi betragter en endelig tidshorisont N. Undervejs fra tid 0 til tid N skal vi foretage en række sekventielle beslutninger under usikkerhed. En beslutning taget i tid t skal alene være baseret på den information, der er til rådighed til tid t, hvor $0 \leq t \leq N$. Vi forudsætter, at informationen er givet som en tidsdiskret stokastisk proces, der er beskrevet i træstrukturen i kapitel (3).

Vi bruger ideerne fra to–stadie låneoptimeringsmodellen til at generalisere til en multi–stadie model. For demonstrationens skyld ser vi på et scenariotræ over tre perioder, hvor realiseringer af den stokastiske proces er begrænset til to udfald i hver periode (figur 4.3). Igen angiver knuder omskrevet af

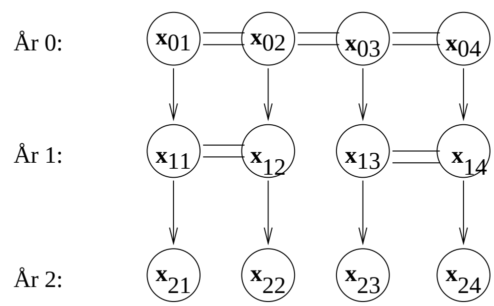


Figur 4.3: Et scenariotræ for en multi–stadie stokastisk programmeringsmodel.

firkanter beslutningsknuderne. Den oprindelige beslutning for optagelse af lån skal træffes i år 0. Der er mulighed for justering af beslutningen i år 1 og år 2. Med andre ord består "recourse" problemet nu af to stadier. Det første problem for udvidelsen fra en 2–stadie til en multi–stadie model er repræsentationen af scenarierne i år 1 og 2. I litteraturen findes to måder at repræsentere scenarierne på. I den første, (Birge, [2]), består hver stadium af et antal scenarier. D.v.s., at hvis vi definerer vektoren $\mathbf{x}_{ts}$ som beslutningsvariabelvektoren ved tid t, scenario s, så har vi for vores eksempel i figur (4.3): I tid 0, $\mathbf{x}_{01}$, i tid 1, $\mathbf{x}_{11}$ og $\mathbf{x}_{12}$ og i tid 2, $\mathbf{x}_{21}$, $\mathbf{x}_{22}$, $\mathbf{x}_{23}$ og $\mathbf{x}_{24}$. Fordelen med denne formulering er, at vi implicit tager højde for "ikke forudseenhed" betingelser. Til gengæld er modellen ikke nemt tilgængelig for en eventuel parallel implementering.

Alternativt kan vi benytte split–variabel formuleringen, (f.eks. Rockafellar og Wets, [20] eller Ruszczynski, [21]). Det går ud på at betragte et scenario som en sti fra roden til en af de sidste beslutningsknuder. Vi har således 4 scenarier i figur (4.3), $(s = 1, s = 2, s = 3, s = 4)$. I år 1 skal vi derfor tænke på $s^* = 1$, fra Birge' scenarioformulering, som værende et mellempunkt på scenarierne 1 og 2, ($s = 1$ og $s = 2$), og tilsvarende ligger $s^* = 2$ på $s = 3$ og $s = 4$ stierne. Det svarer til, at vi har følgende beslutningsvariable: I tid 0, ($\mathbf{x}_{01}, \mathbf{x}_{02}, \mathbf{x}_{03}, \mathbf{x}_{04}$), i tid 1, ($\mathbf{x}_{11}, \mathbf{x}_{12}, \mathbf{x}_{13}, \mathbf{x}_{14}$) og i tid 2 ($\mathbf{x}_{21}, \mathbf{x}_{22}, \mathbf{x}_{23}$ og $\mathbf{x}_{24}$).

Med denne formulering skal vi indføre separate ikke–forudseenhedsbegrænsninger, (Engelsk. "non–anticipativity constraints"). Formålet med disse begrænsninger er at sikre, at en beslutning i stadium t alene afhænger af den information der er tilgængelig til tiden t. For eksemplet i figur (4.3) skal vi indføre følgende begrænsninger: For år 0, ($\mathbf{x}_{01} = \mathbf{x}_{02} = \mathbf{x}_{03} = \mathbf{x}_{04}$). For år 1, ($\mathbf{x}_{11} = \mathbf{x}_{12}$) og ($\mathbf{x}_{13} = \mathbf{x}_{14}$):



Figur 4.4: Demonstration af ikke–forudseenhedsbegrænsninger for split variabel formulering.

Birge og Louveaux ([3]) formulerer disse ligninger som følger:

$$(\sum_{s' \in S^t_{J(ts)}} p_{s'} \mathbf{x}_{ts'}) - (\sum_{s' \in S^t_{J(ts)}} p_{s'}) \mathbf{x}_{ts} = 0 \quad \forall t, s, \tag{4.29}$$

hvor p_s er sandsynligheden for scenario s og s' er en alias for s. Mængden $S^t_{J(ts)}$ repræsenterer scenarier med samme fortid fra tidspunkt t og op til scenario s. I vores eksempel svarer $S^0_{J(0,3)}$ således til scenarierne 1, 2 og 3 og $S^1_{J(1,4)}$ til scenarierne 3 og 4. Denne notation brugt i [3] er ikke intuitiv, så vi ser på betydning af det (som antydet i implementeringen på side 27 i [3] i det følgende.

I åben form giver begrænsningen følgende ligninger for $t = 0$:

$$\frac{1}{4}\mathbf{x}_{01} = \frac{1}{4}\mathbf{x}_{01}$$
$$\frac{1}{4}\mathbf{x}_{01} + \frac{1}{4}\mathbf{x}_{02} = \frac{2}{4}\mathbf{x}_{02}$$
$$\frac{1}{4}\mathbf{x}_{01} + \frac{1}{4}\mathbf{x}_{02} + \frac{1}{4}\mathbf{x}_{03} = \frac{3}{4}\mathbf{x}_{03}$$
$$\frac{1}{4}\mathbf{x}_{01} + \frac{1}{4}\mathbf{x}_{02} + \frac{1}{4}\mathbf{x}_{03} + \frac{1}{4}\mathbf{x}_{04} = \frac{4}{4}\mathbf{x}_{04}$$

Man kan hurtigt overbevise sig selv om, at dette system af lineære ligninger betyder: $\mathbf{x}_{01} = \mathbf{x}_{02} = \mathbf{x}_{03} = \mathbf{x}_{04}$. For $t = 1$ fås:

$$\begin{aligned}
&\frac{1}{4}\mathbf{x}_{11} = \frac{1}{4}\mathbf{x}_{11} \\
&\frac{1}{4}\mathbf{x}_{11} + \frac{1}{4}\mathbf{x}_{12} = \frac{2}{4}\mathbf{x}_{12} \\
&\frac{1}{4}\mathbf{x}_{13} = \frac{1}{4}\mathbf{x}_{13} \\
&\frac{1}{4}\mathbf{x}_{13} + \frac{1}{4}\mathbf{x}_{14} = \frac{2}{4}\mathbf{x}_{14}
\end{aligned}$$

Disse ligninger er ensbetydende med $\mathbf{x}_{11} = \mathbf{x}_{12}$ og $\mathbf{x}_{13} = \mathbf{x}_{14}$. Inspireret af åbenformen af begrænsning (4.29) kan vi nu introducere en tre–dimensional matrix S_Link_{tsk}, med $t \in J, s \in S, k \in S$, hvor S er mængden af scenarier og J er mængden af stadier. Matricen består udelukkende af 0 og 1–taller. For vores eksempel indfører vi for $t = 0$:

$$S\_Link_{0sk} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

For $t = 1$ indfører vi:

$$S\_Link_{1sk} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

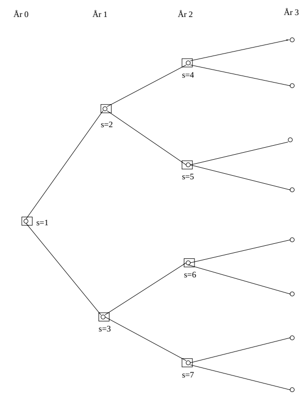
Nu kan begrænsning (4.29) omskrives, således at den kan implementeres direkte i et algebraisk modelleringssprog såsom GAMS.[6]

$$\Big(\sum_{k\in S} S\_Link_{tsk} \cdot p_k \cdot \mathbf{x}_{tk}\Big) - \Big(\sum_{k\in S} S\_Link_{tsk} \cdot p_k\Big)\mathbf{x}_{ts}, \quad \forall s,\ \forall t \in J\backslash(N-1)$$

Vi kan se, at split–variabel formuleringen bruger betydelig flere begrænsninger og variable end formuleringen med implicitte ikke–forudseenhedsbegrænsninger. Det interessante ved split–variabel formuleringen er, at hvis man midlertidigt ser bort fra ikke–forudseenhedsbegrænsninger, kan man betragte problemet som adskilte scenarier, der kan løses parallelt. Den egenskab kan udnyttes til at udvikle effektive parallelle algoritmer til løsning af problemet. (se f.eks. Nielsen og Zenios, [19])

Eftersom vi ikke, i dette projekt, vil betragte sådanne parallelle algoritmer, vil vi i det følgende udvikle en multistadie model baseret på implicitte ikke–forudseenhedsbegrænsninger (Birges formulering). Vi kan dog udnytte den specielle træstruktur, vi har i det underliggende problem, til yderligere at reducere antallet af variablene i vores formulering. Som kan ses i figur (4.5) kan vi definere et scenario som en beslutningsknude, i stedet for en sti. Således

[6] Man kan nemt definere flerdimensionale tabeller i algebraiske programmeringssprog.



Figur 4.5: En alternativ scenario definition for en multi–stadie stokastisk programmeringsmodel.

kan vi definere 7 scenarier for vores eksempel (figur 4.5). Hvert stadium har særskilte scenarier. Dette gøre det muligt at droppe tidsindekset t, idet tiden fremgår implicit af scenarierne. Dette forhold er givet ved $t = \lfloor \frac{\ln s}{\ln 2} \rfloor$, eftersom der er 2^t knuder i hvert stadie t. Med denne definition af scenarierne er vi nu klar til at beskrive vores formulering af multi–stadie problemet.

Konstanterne:
p_s: Sandsynligheden for scenario s, $\forall s \in \{1, \cdots, 2^N - 1\}$.
τ_s: Diskonteringsfaktoren ved scenario s.
k_{is}: Kursen for obligation i, scenario s.
$Callk_{is}$: Konverteringskursen for obligation i, scenario s. Vi sætter $Callk_{is} = \min\{1, k_{is}\}$ for konverterbare obligationer og $Callk_{is} = k_{is}$ for ikke konverterbare obligationer.
r_{is}: Renten for obligation i, scenario s.
n: Restløbetid i år.
γ: Skattefradragsprocent fra rentebetaling.
β: Skattefradragsprocent fra bidragsbetaling.
b_{is}: Bidragsprocent som procentdel af restgæld for obligation i, scenario s.
m_{is}: Faste omkostninger forbundet med optagelsen af lån i, scenario s.

Variablene:
B_s: Den samlede betaling ved scenario s.
RG_{is}: Restgæld for obligation i, scenario s.
S_{is}: De solgte enheder af obligation i ved scenario s.
P_{is}: De købte (indfriede) enheder af obligation i ved scenario s.
A_{is}: Afdrag fra obligation i ved scenario s.

$$l_{is} : \begin{cases} 1 & \text{hvis der løber faste omkostninger for obligation } i, \text{ scenario } s. \\ 0 & \text{ellers.} \end{cases}$$

Multi–stadie modellen kan nu formuleres som følgende:

$$\min \sum_s p_s \cdot \tau_s \cdot B_s \tag{4.30}$$

$$\sum_i k_{i1} \cdot RG_{i1} \geq 1.000.000 \tag{4.31}$$

$$RG_{i1} = S_{i1} \qquad \forall i \tag{4.32}$$

$$RG_{i\lfloor \frac{s}{2} \rfloor} - A_{i\lfloor \frac{s}{2} \rfloor} - P_{is} + S_{is} = RG_{is} \qquad \forall i, \forall s \backslash 1 \tag{4.33}$$

$$\sum_i (k_{is} \cdot S_{is}) = \sum_i (Callk_{is} \cdot P_{is}) \qquad \forall s \backslash 1 \tag{4.34}$$

$$A_{is} = RG_{is} \Big[\frac{r_{is}}{1-(1+r_{is})^{-n+\lfloor \frac{\ln s}{\ln 2} \rfloor}} - r_{is} \Big] \qquad \forall i, s \tag{4.35}$$

$$B_s = \sum_i \Big(A_{is} + (1-\gamma) RG_{is} \cdot r_{is} + (1-\beta) RG_{is} \cdot b_{is} + \eta \cdot (S_{is} + P_{is}) + m_{is} \cdot l_{is} \Big) \qquad \forall s \tag{4.36}$$

$$BigM \cdot l_{is} - S_{is} \geq 0 \qquad \forall i, s \tag{4.37}$$

$$RG_{is}\ ,\ S_{is}\ , P_{is} \geq 0\ ,\ l_{is} \in \{0, 1\} \qquad \forall is \tag{4.38}$$

Modellen er en direkte udvidelse af 2–stadie modellen, dog er der visse ændringer, som vi vil diskutere her:

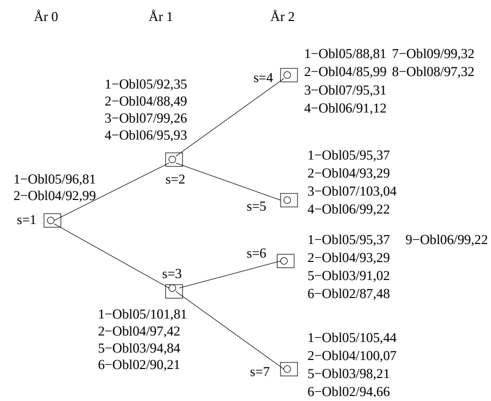
Tidsindekset er forsvundet p.g.a. den nye scenarioformulering. Ikke desto mindre indgår tiden implicit i modellen. Sandsynligheden, p_s, er ikke ligesom før lig med 2^{-N} for alle scenarier men derimod lig med 2^{-t} for hvert stadium t, hvor $t = \lfloor \frac{\ln s}{\ln 2} \rfloor$. Udover beregningen af sandsynlighedskonstanterne indgår tiden implicit i beregning af afdragene som fremgår af ligning (4.35).

En anden detalje kan ses i begrænsning (4.33), hvor vi anvender en heltalsdivision (eller en division med efterfølgende nedtrunkering) for at bestemme forælder–scenariet.

Modellen beskriver dynamikken i det danske lånemarked. Vi har dog ikke beskrevet en måde, hvorpå data kan repræsenteres hensigtsmæssigt, således at modellen kan håndtere de forskellige obligationer i overensstemmelse med obligationernes egenskaber. Vi vil i det følgende eksempel give et bud på, hvordan obligationsdata kan repræsenteres. I næste afsnit vil de specifikke modelleringstekniske detaljer bliver diskuteret.

Vi betragter et univers af obligationer i figur (4.6):

Hver obligation har et nummer, en rente og en kurs. Vi forudsætter, at der kun er tale om fastforrentede obligationslån. I afsnit (5.1) vil vi vise, hvordan data–repræsentationen på en naturlig måde også kan repræsentere fastforrentede kontantlån, rentetilpasningslån og rentetilpasningslån med rentesikring.



Figur 4.6: Et scenariotræ for en multi–stadie stokastisk programmeringsmodel, med angivelse af et univers af obligationer.

Vi starter med 2 obligationer ved scenario 1 (knude 1). Der genereres nye obligationer i takt med, at vi går frem i scenariotræet. En obligation må kun udstedes, hvis kursværdien på obligationen er under pari.[7] De obligationer, som har en kurs over pari, skal derfor betragtes som lukkede. Obligation 3 kan således alene udstedes i scenario 2 (3-Obl07/99,26), og i scenario 4 (3-Obl07/95,31), men ikke i scenario 5 (3-Obl07/103,04). Ved scenario 5 kan obligation 3 enten bibeholdes, eller indfris, hvis den vel at mærke er blevet udstedt i scenario 2.

Når vi bygger scenariotræet, skal vi også huske på, at den underliggende model for generering af obligationsdata er et kombinerende binomialtræ. I vores eksempel betyder det, at nulkuponrenterne ved scenarierne 5 og 6 er ens. Hvis vi genererer en ny obligation i en af disse knuder, skal den samme obligation derfor også eksistere i den anden knude. I vores eksempel i figur (4.6) har vi genereret obligation 9 (9-Obl06/99,22) i scenario 6. Eftersom denne obligation i forvejen er åben for udstedelse i scenario 5 som obligation 4 (4-Obl06/99,22), behøver vi ikke at foretage os noget. Hvis dette ikke var tilfældet, skulle vi generere obligation 9 i både scenario 5 og 6.

I næste afsnit (afsnit 4.5) vil vi se, hvordan det stokastiske program kan modelleres i GAMS. Herunder vil vi se på en måde at repræsentere obligationsdata i GAMS–tabeller. Vi vil fortsatte med eksemplet fra figur (4.6) i næste afsnit.

[7] Realkreditinstitutter låner ikke penge i obligationer med kurs over pari (kurs 100), eftersom stort set alle fastforrentede lån er i dag konverterbare. Hvis der kunne udstedes lån med kurs over pari, vil dette betyde, at lånet kunne indfris øjeblikkeligt på kurs 100 ved at udstede en ny obligation, eventuelt med kurs over 100 igen. På den måde kunne man reducere sin restgæld øjeblikkeligt.

4.5 Løsningsmetoder og implementering

Stokastiske optimeringsmodeller kan formuleres som deterministiske modeller. Det har vi set et eksempel på i de sidste to afsnit. En sådan model kaldes for det deterministiske ækvivalente problem til det stokastiske problem. Sådanne modeller kan løses direkte ved hjælp af de eksisterende metoder til løsning af deterministiske problemer. Vi har hidtil anvendt CPLEX solver til løsning af lineære og heltal–lineære problemer.

Eftersom antallet af scenarier vokser eksponentielt i vores multi–stadie stokastiske program, vil de eksakte metoder, når antallet af scenarierne er vokset tilstrækkeligt, enten være for langsomme til at producere løsninger eller computeren vil mangle hukommelse til at kunne løse problemerne. I sådanne tilfælde kan det betale sig at reducere antallet af scenarierne på en sådan måde, at når vi løser det reducerede problem, får vi en optimal løsning, der er så tæt på løsningen fra det originale problem som muligt. Dette giver anledning til introduktion af scenario–reduceringsproblemet (SRP), som skal løses separat, inden vi løser det egentlige problem.

Nicole Gröwe–Kuska, Holger Heitsch, Jitka Dupačová og Werner Römisch i to artikler ([11, 13]) definerer SRP som et optimeringsproblem og udvikler flere approksimerende algoritmer til løsning af dette problem.

SRP introduceres som følgende: Givet et konvekst stokastisk problem, hvor de stokastiske data er givet som et sæt af scenarier, og givet et ønske om at reducere antallet af scenarierne med et bestemt antal eller procentdel, find en submængde af de oprindelige scenarier og denne submængdes tilknyttede sandsynlighedsfordeling, således at den nye sandsynlighedsfordeling har den mindste afstand, givet en afstandsmål for sandsynlighedsfordelinger, til den oprindelige fordeling som muligt.

Forfatterne bag artiklerne ([11, 13]), i samarbejde med "GAMS Software GmbH" og "GAMS Development Corporation", har udviklet en samling af C++ rutiner, SCENRED, til optimal reducering af scenarierne i et scenariotræ, samt et link, GAMS/SCENRED, der forbinder GAMS programmet til SCENRED algoritmerne.

Vi vil i det følgende først gøre rede for intuitionen bag de algoritmer, der løser SRP. Dernæst vil vi vise og analysere GAMS implementeringen bag vores multi–stadie model fra sidste afsnit. Endelig vil vi omskrive GAMS modellen til en GAMS/SCENRED model og tilføje de nødvendige SCENRED parametre til modellen. Obligationsdata præsenteret i dette afsnit stammer fra scenariotræet i figur 4.6 (side 68).

4.5.1 Scenario–reducerende algoritmer i stokastisk programmering

Som vi har set i tidligere afsnit, udformer scenarierne og deres tilknyttede sandsynligheder en diskret approksimation til den fordeling, der beskriver den underliggende stokastiske dataproces. For scenarioreduceringsformål definerer vi et scenario som den unikke sti, der løber fra roden til et af bladene i scenariotræet.

Sværheden i at løse stokastiske programmer akkumuleres i takt med stigningen i antallet af scenarierne. Dette er uafhængig af, om vi bruger deterministiske algoritmer, eller om vi bruger skræddersyede dekomponeringsalgoritmer, som udnytter stokastiske programmers specielle struktur. Det er derfor naturligt at forsøge at reducere antallet af scenarierne samtidig med, at vi beholder en god præcision i scenariotræet.

Scenario–reducerende algoritmer kontrollerer, at det reducerede scenariotræ er så tæt på det originale scenariotræ som muligt. Til dette formål bruger de en sandsynlighedsafstandsmål (Engelsk: probability metric). Afstandsmålet[8] mellem to scenarier defineres som $c(\xi^i, \xi^j)$, hvor ξ^i og ξ^j repræsenterer scenarierne i og j.

Vi definerer yderligere antallet af de oprindelige scenarier som S, indekssættet af de slettede scenarier som J, antallet af indeks i sæt J som $\#J$, antallet af bevarede scenarier som $s = S - \#J$ og sandsynligheden for et scenario som p_i, hvor $i \in J$. SRP kan nu formuleres som følgende:

$$\min\Big\{D_J := \sum_{i \in J} p_i \min_{j \notin J} c(\xi^i, \xi^j) : J \subset \{1, \cdots, S\}, \#J = S - s\Big\}, \tag{4.39}$$

hvor D_J i ord kan beskrives som summen af de vægtede mindste afstand fra et scenario i sæt J til et scenario i sæt $\{1, \cdots, S\} \backslash J$. Problemet svarer til, at sættet $\{1, \cdots, S\}$ skal dækkes af to sæt $J \subset \{1, \cdots, S\}$ og $\{1, \cdots, S\} \backslash J$, således at J har fast kardinalitet $S - s$ og D_J har minimal omkostning. Dette problem kan omformuleres til et "set covering problem", (SCP), som er $\mathcal{NP}$–hårdt. Der findes ikke effektive eksakte algoritmer til løsning af SCP, hvorfor SRP–algoritmerne bruger heuristiske metoder til løsning af disse underliggende SCP–problemer. Vi vil ikke gå nærmere ind på detaljerne bag disse algoritmer men blot beskrive de konceptuelle ideer, der ligger til grund for algoritmerne.

[8]Som afstandsmål bruges den såkaldte "Monge–Kantorovich" eller "transportaion" afstandsmål, (se ([11, 13])).

Backward reduction–algoritmen

Vi betragter det specielle tilfælde, hvor $\#J = 1$, d.v.s. at der skal slettes et enkelt scenario. Problem (4.39) reduceres nu til:

$$\min_{l \in \{1, \cdots, S\}} p_l \min_{j \neq l} c(\xi^l, \xi^j). \tag{4.40}$$

Den optimale løsning kan findes på følgende måde: For hvert scenario, ξ^l, beregn afstanden til de øvrige scenarier. Vælg den mindste af disse beregnede værdier og gang den med vægten, p_l, som er sandsynligheden for scenario ξ^l. Vælg endelig det scenario, der giver den laveste af disse vægtede værdier. Hvis den optimale løsning er fundet som $l_* \in \{1, \cdots, S\}$, betyder dette, at scenario ξ^l_* skal slettes, eftersom dette scenario har den mindste vægtede afstand til et andet scenario. Denne fremgangsmåde kan gentages rekursivt indtil den ønskede kardinalitet $\#J$ er opnået.

Forward selection–algoritmen

Lad os nu betragte det andet specielle tilfælde, hvor $\#J = S - 1$, d.v.s. at alle scenarier bortset fra 1 skal slettes. Problem (4.39) reduceres derved til:

$$\min_{u \in \{1, \cdots, S\}} \sum_{i=1}^{S} p_i c(\xi^i, \xi^u). \tag{4.41}$$

Den optimale løsning findes ved at beregne summen af de vægtede afstande fra et scenario, ξ^u, til samtlige andre scenarier og vælge det scenario med mindste sum. Når vi finder den optimale løsning, $u_* \in \{1, \cdots, S\}$, beholder vi scenario ξ^u_*, eftersom dette scenario kan betragtes som det mest centrale scenario. Vi har nu $S - 1$ scenarier tilbage. Vi kan gentage denne procedure på de ikke valgte scenarier indtil vi opnår den ønskede kardinalitet $\#J$ blandt disse ikke valgte scenarier. Det svarer til, at vi har slettet disse scenarier.

Numeriske erfaringer

SCENRED algoritmerne er baseret på ideerne fra algoritmerne “backward reduction” og “forward selection”. De endelige algoritmer, som er et resultat af videreudvikling af de grundlæggende ideer, er kendt under følgende navne: “Fast backward method”, “mix of fast backward/forward method” og “mix of fast backward/backward method”.

Når vi anvender SCENRED, skal vi enten vælge en af disse metoder eller vælge default, som vælger den bedste metode m.h.t. køretid. For store scenariotræer har "fast backward method" den bedste køretid, hvorimod de andre to metoder er mere præcise. "Mix of fast backward/forward method" er den mest præcise, men den har også den længste køretid, derfor kan den kun anbefales, hvis vi ønsker en stærk reducering i scenariotræet.

Ifølge ([11, 13]) finder algoritmerne gode approksimationer til de optimale løsninger. De numeriske erfaringer med reducering af scenarier i scenariotræerne fra el.–produktionsstyring under usikkerhed viser, at i gennemsnit er 90% af den relative præcision bibeholdt efter en 50% reducering.

Den relative præcision[9] m.h.t. afstandsmålet $c(\xi^i, \xi^j)$, defineres som $\zeta_c^{rel}(P, Q_s)$, hvor P er sandsynlighedsfordelingen for det oprindelige scenariosæt og Q_s er sandsynlighedsfordelingen for det reducerede scenariosæt, hvor der er s scenarier tilbage.

Den relative præcision måles som:

$$\zeta_c^{rel}(P, Q_s) := \frac{\zeta_c(P, Q_s)}{\zeta_c(P, \delta_{\xi_*^i})},$$

hvor tælleren, $\zeta_c(P, Q_s)$ er den absolutte afstand mellem de to scenariosæt P og Q, og nævneren, $\zeta_c(P, \delta_{\xi_*^i})$, normerer den absolute afstand, således at den kan præsenteres som en procentsats.

4.5.2 GAMS–formuleringen

For at SCENRED kan bruges på et GAMS–program, skal GAMS–programmet formuleres så generelt, at det kan bruge et vilkårligt scenariotræ som datainput. Hvis derimod SCENRED ikke skal bruges, er det en god ide at skræddersy GAMS–programmet til at udnytte den underliggende scenariostruktur, så vi kan opnå en mere effektiv implementering. Formålet med dette afsnit er at vise, hvordan et sådant skræddersyet program kan udformes.

Når man modellerer i GAMS er det en god praksis at holde data og model adskilt. Eftersom måden vi præsenterer data på i vores GAMS–formulering, med passende indføring af "dummy" koefficienter, har betydning for modellens korrekthed i.f.t. virkeligheden, betragter vi først datafilen, *data.txt*:

```
$ontext
data.txt : Datafilen med to fastforrentede lån
$offtext
```

[9] Vi gengiver blot, for fuldstændighedens skyld, definitionen på den relative præcision som beskrevet i artiklerne ([11, 13]). En fuld forståelse af de nævnte begreber er inden for området "measure theory", som ligger uden for rækkevidden af dette eksamensprojekt.

```
SETS
      I    laan produkter            / laan1*laan9/
      S    scenarier                 / Scen1*Scen7 / ;

ALIAS(S,S2)

SCALAR BigM     /1500000/;
SCALAR gamma    /.32/;      // Skattefradragsprocent fra rentebetaling
SCALAR betta    /.32/;      // Skattefradragsprocent fra bidrag
SCALAR bidrag   /0.005/;    // Bidragsprocent fra restgaeld
SCALAR N        /3/;        // Antal perioder
SCALAR Provenu  /1000000/;  // Den initielle kursvaerdi

PARAMETERS
 Prob(S)  Sandsynligheden for de enkelte scenarier
        /    Scen1       1
             Scen2       0.5
             Scen3       0.5
             Scen4       0.25
             Scen5       0.25
             Scen6       0.25
             Scen7       0.25      /

 Fast(S)  Faste omkostninger i forbindelse med forskellige scenarier
       /     Scen1     10000
             Scen2     1000,     Scen3     1000
             Scen4     1000,     Scen5     1000
             Scen6     1000,     Scen7     1000 /;


TABLE R(I,S) scenariotrae for renter
       Scen1 Scen2 Scen3 Scen4 Scen5 Scen6 Scen7
laan1  .05   .05   .05   .05   .05   .05   .05
laan2  .04   .04   .04   .04   .04   .04   .04
laan3  1     .07   1     .07   .07   1     1
laan4  1     .06   1     .06   .06   1     1
laan5  1     1     .03   1     1     .03   .03
laan6  1     1     .02   1     1     .02   .02
laan7  1     1     1     .09   1     1     1
laan8  1     1     1     .08   1     1     1
laan9  1     1     1     1     1     .06   1;

TABLE K(I,S) scenariotrae for obligationskurser
       Scen1 Scen2 Scen3 Scen4 Scen5 Scen6 Scen7
laan1  .9681 .9235 0     .8881 .9537 .9537 0
laan2  .9299 .8849 .9742 .8599 .9329 .9329 0
laan3  0     .9926 0     .9531 0     0     0
laan4  0     .9593 0     .9112 .9922 0     0
laan5  0     0     .9484 0     0     .9102 .9821
```

```
laan6  0     0     .9021 0     0     .8748 .9466
laan7  0     0     0     .9932 0     0     0
laan8  0     0     0     .9732 0     0     0
laan9  0     0     0     0     0     .9922 0;

TABLE CallK(I,S) scenariotrae for konverteringskurser
       Scen1 Scen2 Scen3 Scen4 Scen5 Scen6 Scen7
laan1  1    .9235 1.000 .8881 .9537 .9537 1.000
laan2  1    .8849 .9742 .8599 .9329 .9329 1.000
laan3  1    1     1     .9531 1.000 1     1
laan4  1    1     1     .9112 .9922 1     1
laan5  1    1     1     1     1     .9102 .9821
laan6  1    1     1     1     1     .8748 .9466
laan7  1    1     1     1     1     1     1
laan8  1    1     1     1     1     1     1
laan9  1    1     1     1     1     1     1;
```

I filen *data.txt* er det specielt præsentationen af datatabellerne R(I,S), K(I,S), og CallK(I,S), der er interessant i modelleringssammenhænge.

Tabel K(I,S) sikrer os, at det ikke er muligt at udstede en obligation, der ikke eksisterer ved et givent scenario, eller en obligation, der ikke længere er åben for handel. Et par eksempler kan klarlægge dette.

Først betragter vi obligation 3 i scenario 1. Ifølge figur 4.6 (side 68) findes obligation 3 slet ikke i scenario 1. En k værdi på 0 i tabel K(I,S) betyder, at udstedelsen af denne obligation i scenario 1 giver en kursværdi på nul, og at den er derved ikke interessant. Tilsvarende i tabel R(I,S) sætter vi renten for lån 3 i scenario 1 til 1, svarende til en rente på 100%. Selvom dette strengt taget ikke er nødvendigt, idet vi ikke er interesserede i en kursværdi på 0, kan 1 tallet bruges blot som en ekstra indikation på en ikke–eksisterende obligation.

Dernæst betragter vi obligation 1 ved scenario 3. Obligation 1 må ikke udstedes i scenario 3, eftersom kursen på denne obligation er over 100 ved dette scenario ifølge 4.6 (side 68). Derfor har vi sat kursen lig nul i tabel K(I,S) for at indikere, at obligationen er lukket for oprettelse af nyt lån ved scenario 3. Den tilsvarende rente er dog uberørt i tabel R(I,S). Vi skal nemlig bruge denne rente til beregning af restgæld, afdrag og betaling for dette scenario i det tilfælde, hvor lånet er oprettet i obligation 1 i scenario 1.

Tabel CallK(I,S) viser indfrielseskurserne. Obligationerne må ikke indfris ved optagelsestidspunktet, ligesom de ikke kan indfris, hvis de ikke eksisterer ved et scenario. Vi skal derfor sørge for, at indfrielseskursen er højere end kursen på en obligation, som lige er blevet udstedt, eller højere end kursen på en obligation, der ikke eksisterer i et scenario.[10] Dette sikrer os, at indfrielsen

[10] Kursen på en obligation, der ikke eksisterer, er ifølge tabel K(I,S) lig med nul. Eftersom

ikke er optimal og derved ikke vil finde sted. De steder, hvor indfrielsen er tilladt, skriver vi den gældende indfrielseskurs, der er defineret $Callk_{is} = \min\{1, k_{is}\}$.

Vi kan nu formulere GAMS–filen *multi1_4.gms*, som er skræddersyet til multi–stadiemodellen fra sidste afsnit. Programmet kan bruges for vilkårlig mange perioder, blot det bliver forsynet med en datafil, der er udformet som *data.txt*.

```
$ontext
multi1_4.gms : Vælger et lån eller en kombination af lån ved scenario 1
og dernæst plejer denne låneportefølje optimalt. Med faste omkostninger,
mip formulering.(likviditetsrisiko neutral låntager)
En multi-stage stokastisk heltalslåneoptimeringsmodel med en scenario
repræsentation, hvor tiden indgår kun implicit
$offtext

$eolcom //
option
iterlim=999999999,reslim=300,optcr=0.0,solprint=OFF,limrow=0,limcol=0;
$include datafil.txt  // 2 fastforrentede og 1 F1 laan

VARIABLES
       RG(I,S)   Restgaeld
       Sale(I,S) Salgsvariabel
       P(I,S)    Koebevariabel
       A(I,S)    Afdrag
       B(S)      Betaling
       L(I,S)    Indikator for faste omkostninger
       Z         Objektfunktionsvaerdien ;
POSITIVE VARIABLES RG, Sale , P;
BINARY VARIABLE L;

EQUATIONS
  COST        Definere objektfunktionen
  EQ1         Hele laanet skal daekkes til at starte med
  EQ2(I)      Solgte obligationer ved scenario 1 giver startrestgaelden
  EQ3(I)      Det er ikke tilladt at koebe obligationer ved scenario 1
  EQ4(I,S,S2) Balanceligninger
  EQ5(S)      Betalingsstroemligninger
  EQ6(I,S)    Definition af afdrag
  EQ7(S)      Definition af betaling
  EQ8(I,S)    Faste omlaegningsomkostninger     ;

  COST         ..   Z  =E=  SUM(S, Prob(S)*B(S));
  EQ1          ..   SUM(I, K(I,'scen1')*RG(I,'scen1')) =G= Provenu;
```

kurserne for obligationer, der kan handles, er mindre end eller lige med 1, skal vi blot vælge et tal, der er større eller lige med 1. Vi har valgt at sætte indfrielseskursen lig 1 de steder, hvor indfrielsen ikke er mulig eller tilladt.

```
EQ2(I)       ..   RG(I,'scen1') - Sale(I,'scen1') =E= 0;
EQ3(I)       ..   P(I,'scen1') =E= 0;
EQ4(I,S,S2)$(Floor(ORD(S)/2)=ORD(S2))  .. RG(I,S2) - A(I,S2)
                              - P(I,S) + Sale(I,S) - RG(I,S) =E= 0;
EQ5(S)$(ORD(S) > 1)    ..   sum(I,K(I,S)*Sale(I,S))
                                  - sum(I,CallK(I,S)*P(I,S)) =E= 0;
EQ6(I,S)     ..   A(I,S) - RG(I,S)*( ( R(I,S)/(1- (1+R(I,S))
              **(-N+floor(log(ORD(S))/log(2))) )- R(I,S) ) ) =E= 0;
EQ7(S)       ..   B(S) - sum(I, A(I,S)+ (1- gamma)*RG(I,S)*R(I,S)
            + (1-betta)*RG(I,S)*bidrag + Fast(S) * L(I,S)  ) =E= 0;
EQ8(I,S) .. BigM * L(I,S) - Sale(I,S) =G= 0;

MODEL LOANOPTIM /ALL/ ;
OPTION MIP=CPLEX;        // CPLEX bruges som solver
SOLVE LOANOPTIM USING MIP MINIMIZING Z ;
display Sale.l, P.l, L.l, RG.l; // De optimale vaerdier bliver vist i tabelform
```

GAMS–programmet *multi1_4.gms* stemmer linje for linje overens med den stokastiske multi–stadie model fra sidste afsnit, (4.30 til 4.38). Læg dog mærke til EQ4, der svarer til balanceligningerne, 4.33 (side 67). Vi sætter S til at være barnescenariet, og vi sætter S2, som er alias til S, til at være forælderscenariet. Dollar–betingelsen \$(Floor(ORD(S)/2)=ORD(S2)) sikrer os nu, at vi tager højde for forælder–barn forholdet i træet, hvor S=s er barnescenariet og S2=$\lfloor \frac{s}{2} \rfloor$ er forælderen.

Når vi kører programmet *multi1_4.gms*, får vi en objektværdi på DKK 1.097.823, 85. Løsningen er givet som følger:

```
----    123 VARIABLE Sale.L  Salgsvariabel

            Scen1       Scen2       Scen4       Scen6

laan1   1032951,141
laan3               656191,308
laan7                           325496,006
laan9                                       347228,883


----    123 VARIABLE P.L  Koebevariabel

            Scen2       Scen4       Scen6

laan1    705290,193              361246,196
laan3               339190,676


----    123 VARIABLE L.L  Indikator for faste omkostninger
```

```
            Scen1       Scen2       Scen4       Scen6

laan1         1
laan3                     1
laan7                                 1
laan9                                             1


----    123 VARIABLE RG.L  Restgaeld

         Scen1     Scen2     Scen3     Scen4    Scen5    Scen6       Scen7

laan1 1032951,14           705290,19                               361246,20
laan3            656191,31                     339190,68
laan7                                792496,01
laan9                                                    347228,88
```

Vi ser fra løsningen, at for et 3–årigt lån, når valget er begrænset til fastforrentede obligationslån, er det optimalt at starte med den obligation, der har størst rente (kurs tættest på pari) og dernæst op–konvertere til de obligationer, der har de højeste renter, hver gang renterne stiger, d.v.s. vi starter med obl1 (1-Obl05/96,81) i scenario 1. Dernæst op–konverterer vi til obl3 (3-Obl07/99,26), obl7 (7-Obl09/99,32) og obl9 (9-Obl06/99,22) ved scenarierne 2, 4 og 6. Løsningen kan forsvares med, at for et så kort låneperiode er det effekten af kursgevinsten[11] der overstiger den ekstra betaling, der kommer fra højere renter. Det kan derimod ikke betale sig at ned–konvertere, eftersom ned–konvertering betyder en forøgelse af restgælden, og dette kan en formindskelse af rentesatsen ikke kompensere for på så kort tid som 3 år.

4.5.3 GAMS/SCENRED–formuleringen

GAMS–formuleringen i sidste afsnit er afhængig af den underliggende struktur, og kan derfor ikke umiddelbart benyttes som input til SCENRED modulet i GAMS. I det følgende ser vi på en GAMS/SCENRED–formulering af det samme problem, hvor GAMS–programmet er blevet omskrevet til et GAMS/SCENRED–program. Derudover introducerer vi de øvrige GAMS/SCENRED specifikke kald og parametre.

Eftersom SCENRED er lavet til at håndtere et vilkårligt scenariotræ, kan vi desværre ikke længere bruge forælder–barn forholdet mellem knuderne, som vi beskrev under GAMS–formuleringen. Vi er nødt til at introducere en tidsdimension i modellen, og vi skal forsyne modellen med en specifik "mapping" mellem tiderne og scenarierne og en "mapping", der afgør forholdet

[11]Når renterne stiger, falder kurserne, og vi kan reducere vores restgæld ved at op–konvertere til en obligation med en højere rente. Kursgevinsten er det beløb, vi har reduceret vores restgæld med.

mellem forælderscenariet og børnescenarier. Dette kan gøres ved at tilføje følgende kode sidst i filen *data.txt*:

```
SET    T    tidsperioder                /time0*time2/;

SET

ts(t,s)  tid scenario mapping/
                        time0.(Scen1),
                        time1.(Scen2*Scen3),
                        time2.(Scen4*Scen7)/,
tree(s,s) Foraelder barn forhold mellem scenarier/
                        Scen1.(Scen2*Scen3),
                        Scen2.(SCen4*Scen5),
                        Scen3.(Scen6*Scen7)/;

set ss(s) 'submaengde af scenarier i det reducerede subtrae',
    leaf(s) 'bladknuder i det originale trae';
leaf(s)$[sum{ts('time2',s),1}] = YES;
display leaf;
display ts;
display prob;

parameter sprob(s) sandsynligheder for knuderne i det reducerede trae;
```

Tid–scenario–mappingen ts(t,s) viser et entydigt forhold mellem tidsperioderne og scenarierne. Ligeledes viser tree(s,s) et entydigt forhold mellem forælderknuden og dennes efterkommere. SCENRED bruger disse informationer til at identificere det originale scenariotræ. Dernæst reducerer SCENRED scenariotræet ved at omskrive disse "mappings", dog uden at ændre logikken i det oprindelige træ. Det vil sige, at forholdet mellem tid og de tilbageværende scenarier, og ligeledes det indbyrdes forhold mellem de tilbageværende scenarier, er i overensstemmelse med det oprindelige træ.

Med denne opdatering af datafilen er vi nu klar til at omformulere GAMS–filen *multi1_4.gms*, som en GAMS/SCENRED–formulering i *multi1_4_SR.gms*. Læg mærke til, at filen er delt i to for at demonstrere de to dele, der udgør en GAMS/SCENRED–model, nemlig GAMS/SCENRED–formuleringen af selve den underliggende model og GAMS/SCENRED kald.

```
$ontext
multi1_4_SR.gms : GAMS/SCENRED formulering af multi1_4.gms.
Første del er en GAMS/SCENRED formulering af selve den
underliggende model. Her indgår tiden eksplicit.
Anden del viser GAMS/SCENRED specifikke kald.
$offtext
*********1. del: GAMS/SCENRED formuleringen af modellen*********
$eolcom //
option
```

```
iterlim=999999999,reslim=300,optcr=0.0,limrow=0,limcol=0;
$include datafil.txt  // 2 fastforrentede og 1 F1 laan

VARIABLES
       RG(I,T,S)   Restgaeld
       Sale(I,T,S) Salgsvariabel
       P(I,T,S)    Koebevariabel
       A(I,T,S)    Afdrag
       B(T,S)      Betaling
       L(I,T,S)    Indikator for faste omkostninger
       Z           Objektfunktionsvaerdien ;
POSITIVE VARIABLES RG, Sale , P;
BINARY VARIABLE L;

EQUATIONS
 COST        Definere objektfunktionen
 EQ1         Hele laanet skal daekkes til at starte med
 EQ2(I)      Maengden af solgte obligationer ved scen1 giver startrestgaelden
 EQ3(I)      Det er ikke tilladt at koebe obligationer ved scenario 1
 EQ4(I,T,S) Balanceligninger
 EQ5(T,S)    Betalingsstroemligninger
 EQ6(I,T,S) Definition af afdrag
 EQ7(T,S)    Definition af betaling
 EQ8(I,T,S) Faste omlaegningsomkostninger       ;

  COST   ..   Z  =E=  SUM(ts(t,ss), sprob(ss)*B(T,ss));
  EQ1    ..   SUM(I, K(I,'scen1')*RG(I,'time0','scen1')) =G= Provenu;
  EQ2(I) ..   RG(I,'time0','scen1') - Sale(I,'time0','scen1') =E= 0;
  EQ3(I) ..   P(I,'time0','scen1') =E= 0;
  EQ4(I,ts(t,ss)) .. sum(tree(s2,ss), RG(I,T-1,S2) - A(I,T-1,S2)
                     - P(I,T,SS) + Sale(I,T,SS) - RG(I,T,SS)) =E= 0;
  EQ5(ts(t,ss))$(ord(t)>1) .. sum(I,K(I,SS)*Sale(I,T,SS)) -
                        sum(I,CallK(I,SS)*P(I,T,SS)) =E= 0;
  EQ6(I,ts(t,ss)) .. A(I,T,SS) - RG(I,T,SS)*( ( R(I,SS)/(1- (1+R(I,SS))
                               **(-N-1+ord(t)) )- R(I,SS) ) ) =E= 0;
  EQ7(ts(t,ss)) .. B(T,SS) - sum(I, A(I,T,SS)+ (1- gamma)*RG(I,T,SS)*R(I,SS)
               + (1-betta)*RG(I,T,SS)*bidrag + Fast(SS) * L(I,T,SS) ) =E= 0;
  EQ8(I,ts(t,ss)) ..  BigM * L(I,T,SS) - Sale(I,T,SS) =G= 0;

  MODEL LOANOPTIM /ALL/ ;
  OPTION MIP=CPLEX;
********************1. del slutter her*******************
```

Læg mærke til brugen af ts(t,ss) i EQ4 til EQ8. Læg især mærke til balanceligningen EQ4, hvor en kombineret brug af ts(t,ss) og tree(s2,ss) bestemmer forælder–barnforholdet.[12] Nu kan ligningerne acceptere en hvilken som helst reduceret inputdata præsenteret via ts(t,ss) og tree(s2,ss).

[12]Formuleringen i EQ4 kan bruges til at præsentere en vilkårlig balanceligning i et vilkårligt scenariotræ, og derfor er formuleringen af stor interesse i modelleringssammenhænge.

```
*****************2. del: SCENRED kald******************
$if set noscenred $goto noscenreduction

* Nu er vi klar til at koere ScenRed

* Her kommer nogle sets & parameters til brug for scenred I/O
$libinclude scenred.gms

scalar psum, rc, runCount, runMax;
set    run  / run1 * run1 /;
set    method 'reduction method used' /
        '0-default',
        '1-fastback',
        '2-fastback+forw',
        '3-fastback+back'  /;
parameter report(method,run, *);
set       rleaf(method,run,s)  'leaf set of reduced tree';


runMax = INF;
$if set runmax runMax = %runmax%;

* Opsaetning af scenred options file
file opts  /'scenred.opt'/;
putclose opts  'log_file   multilog.txt'
             / 'input_gdx  multiwin.gdx'
             / 'output_gdx multiout.gdx';


* Disse parms kommer fra inputtraeet
ScenRedParms('num_leaves') = sum {leaf, 1};
ScenRedParms('num_random') = 3*card(I);
ScenRedParms('num_nodes')  = card(s);
ScenRedParms('num_time_steps') = card(t);
* optional SCENRED input parameters
ScenRedParms('num_stages') = ScenRedParms('num_time_steps');
ScenRedParms('where_random') = 10;
ScenRedParms('report_level') = 0;
ScenRedParms('run_time_limit') = 30;

runCount = 0;
loop {method$(runCount < runMax),
    ScenRedParms('reduction_method') = ord(method)-1;
    loop {run$(runCount < runMax),
*       these parms control the tree output from ScenRed
*       at least one of the following two parameters is required
        ScenRedParms('red_num_leaves') = ord(run);
*       ScenRedParms('red_percentage') = 0.5;
```

```
        execute_unload 'multiwin.gdx',  ScenRedParms, s, tree, prob, r, k, callk ;
        execute 'rm -f multiout.gdx';
        execute 'scenred scenred.opt %system.redirlog%';
        rc = errorlevel;
        abort$rc "Return code from scenred was nonzero : ", rc;
        execute_load   'multiout.gdx', ScenRedReport, sprob=red_prob;

        ss(s) = sprob(s);
        display ScenRedParms, ScenRedReport;
        display sprob, ss;

        psum = sum {leaf(ss), sprob(ss)};
        abort$[abs(psum-1) gt 1e-8]
          "Error in reduced tree: leaf probabilities do not sum to 1";

        solve loanoptim min z us mip;
        runCount = runCount + 1;

        report(method,run, 'obj') = loanoptim.objval;
        report(method,run, 'red_percentage') =
             ScenRedReport('red_percentage');
        report(method,run, 'reduction_method') =
             ScenRedReport('reduction_method');
        report(method,run, 'run_time') =
             ScenRedReport('run_time');
        rleaf (method,run, leaf(ss)) = YES;
    };
};

display report;
display rleaf;
display r;
display k;
display callk;
*******************2. del slutter her*******************

$goto alldone

$label noscenreduction
* set "reduced tree" to be the whole tree
ss(s) = yes;
sprob(s) = prob(s);

SOLVE LOANOPTIM USING MIP MINIMIZING Z ;
display Sale.l, P.l, RG.l;

$label alldone
```

Vi vil ikke se på en detaljeret beskrivelse af den 2. del af programmet her. I stedet henvises til GAMS/SCENRED–manualen, ([12]).

Vi forklarer dog de modelspecifikke parameterangivelser og kald. De vigtigste informationer, som SCENRED skal forsynes med, er antallet af blade i scenariotræet, antallet af stokastiske variable, antallet af knuder og antallet af tidsskridt. Disse informationer er angivet i det følgende:

```
ScenRedParms('num_leaves') = sum {leaf, 1};
ScenRedParms('num_random') = 3*card(I);
ScenRedParms('num_nodes')  = card(s);
ScenRedParms('num_time_steps') = card(t);
```

Ligeledes skal vi angive omfanget af reduceringen. Dette gøres enten ved at angive det ønskede antal blade efter reducering, eller ved at angive procentdelen, vi gerne vil reducere scenariotræet med:

```
 ScenRedParms('red_num_leaves') = ord(run);
*ScenRedParms('red_percentage') = 0.5;
```

Endelig skal vi forsyne funktionen execute_unload med alle de modelspecifikke parametre:

```
 execute_unload 'multiwin.gdx', ScenRedParms, s, tree, prob, r, k, callk;
```

Vi har nu færdigudviklet en multi–stadie stokastisk optimeringsmodel til låneanbefalingsproblemet. Yderligere har vi formuleret modellen som en standard GAMS formulering og en GAMS/SCENRED formulering. I næste kapitel udvider vi modellen til at kunne håndtere forskellige typer af obligationer, og vi vil se på forskellige måder at håndtere risici på.

Kapitel 5

Tilføjelser til optimeringsmodellen

Vi kan betragte den udviklede model i sidste kapitel som en basemodel, der yderligere skal skræddersys for at beskrive samtlige markedsforhold, så modellen kan bruges i praksis. Det er formålet med dette kapitel.

I afsnit (5.1) vil vi vise, hvordan en vilkårlig portefølje af obligationer kan repræsenteres i modellen. Vi vil også studere effekten af tilføjelsen af rentesikring på vores univers af lånprodukter. Rentesikringens eventuelle effekter på en optimal løsning kan klarlægge, om vores prisfastsættelsesmetode fra afsnit (3.4) underpriser eller overpriser rentesikringen. På den måde kan vi bruge optimeringsmodellen ikke alene som et rådgivningsværktøj til låneanbefaling, men også som et risikostyringsværktøj for realkreditinstituttet. Endelig ser vi på modellering af tidlig indfrielse af lån.

Vi studerer og implementerer forskellige strategier til håndtering af likviditets– og formuerisiko i afsnit (5.2). Ved at bruge de videreudviklede modeller i afsnit (5.1 og 5.2) kan vi finde et skræddersyet lån, der ikke nødvendigvis findes blandt standardprodukterne, men som kan sammensættes efter kundens ønske.

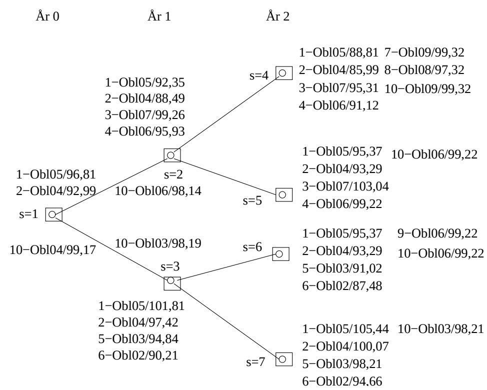
I afsnit (5.3) kommenterer vi denne rapports forskelle med og tilføjelser til Nielsen & Poulsen modellen, ($N\&P$, [18]).

5.1 Udvidelser til modellen

Indtil videre har vi kun set på fastforrentede obligationslån, hvor gælden amortiseres indtil horisonten. I dette afsnit udvider vi modellen til også at kunne håndtere rentetilpasningslån, rentetilpasningslån med rentesikring og fastforrentede kontantlån. Vi vil endvidere betragte situationen med tidlig indfrielse af lånet.

5.1.1 Tilføjelse af nye typer af lån

Vi starter med at se, hvordan modellen kan håndtere et F1–lån. Et F1–lån udmærker sig ved, at det skal refinansieres hvert år. Vi forstiller os, at vi har tilføjet et F1–lån, (Obl10), til vores låneunivers:



Figur 5.1: Et scenariotræ for en multi–stadie stokastisk programmeringsmodel, med angivelse af et univers af obligationer. Obl10 er rentetilpasningslån.

Tabellerne fra afsnit 4.5.2 (side 72) skal nu opdateres som følgende:

```
TABLE R(I,S) scenariotrae for renter
       Scen1 Scen2 Scen3 Scen4 Scen5 Scen6 Scen7
laan1  .05   .05   .05   .05   .05   .05   .05
laan2  .04   .04   .04   .04   .04   .04   .04
laan3  1     .07   1     .07   .07   1     1
laan4  1     .06   1     .06   .06   1     1
laan5  1     1     .03   1     1     .03   .03
laan6  1     1     .02   1     1     .02   .02
laan7  1     1     1     .09   1     1     1
laan8  1     1     1     .08   1     1     1
laan9  1     1     1     1     1     .06   1
laan10 .04   .06   .03   .09   .06   .06   .03;


TABLE K(I,S) scenariotrae for obligationskurser
       Scen1 Scen2 Scen3 Scen4 Scen5 Scen6 Scen7
laan1  .9681 .9235 0     .8881 .9537 .9537 0
laan2  .9299 .8849 .9742 .8599 .9329 .9329 0
laan3  0     .9926 0     .9531 0     0     0
laan4  0     .9593 0     .9112 .9922 0     0
laan5  0     0     .9484 0     0     .9102 .9821
laan6  0     0     .9021 0     0     .8748 .9466
laan7  0     0     0     .9932 0     0     0
laan8  0     0     0     .9732 0     0     0
```

```
laan9  0     0     0     0     0     .9922 0
laan10 .9917 .9814 .9819 .9932 .9922 .9922 .9821;

TABLE CallK(I,S) scenariotrae for konverteringskurser
       Scen1 Scen2 Scen3 Scen4 Scen5 Scen6 Scen7
laan1  1     .9235  1.000 .8881 .9537 .9537 1.000
laan2  1     .8849  .9742 .8599 .9329 .9329 1.000
laan3  1     1      1     .9531 1.000 1     1
laan4  1     1      1     .9112 .9922 1     1
laan5  1     1      1     1     1     .9102 .9821
laan6  1     1      1     1     1     .8748 .9466
laan7  1     1      1     1     1     1     1
laan8  1     1      1     1     1     1     1
laan9  1     1      1     1     1     1     1
laan10 1     1.000  1.000 1.000 1.000 1.000 1.000;
```

Renterne for Obl10 varierer fra år til år, hvilket kan ses i tabel R(I,S). Modellen tager dog ikke automatisk hensyn til refinansieringer ved slutningen af hvert år. Vi kan tilføje begrænsninger, der tvinger modellen til at indfri gælden fra Obl10 hvert år og dernæst sælge Obl10 på ny. Det vil dog medføre, at vi skal indføre yderligere begrænsninger, der sørger for, at der ikke blive unødigt betalt faste omkostninger for disse transaktioner. I stedet for at indfører disse begrænsninger introducerer vi sættet i' som sættet af obligationer for tilpasningslån. Dernæst kan vi tilføje særskilte balancebegrænsninger for obligationer tilhørende dette sæt:

$$RG_{i'\lfloor\frac{s}{2}\rfloor} - A_{i'\lfloor\frac{s}{2}\rfloor} - P_{i's} + S_{i's} = k_{i's} \cdot RG_{i's} \qquad \forall i', \forall s\backslash 1. \tag{5.1}$$

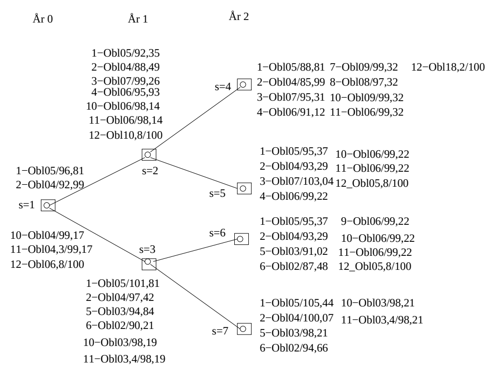
Ved at gange kursen på den nye restgæld justerer vi for restgælden, svarende til at vi har indfriet og solgt samme obligation, uden dog at dette skal registreres i variablerne P og S.

Vi kan bruge det samme princip for et tilpasningslån med rentesikring. Vi så i afsnit 3.4 (side 31), hvordan vi finder nutidsprisen på en rentesikring. I praksis fordeles denne pris som en forøgelse af rentebetaling og bidrag jævnt over rentesikringens levetid, så låntageren kan drage nytte af skattefradraget på rentebetaling og bidrag. Lad os forestille, at vi har tilpasningslån med rentesikring i vores låneunivers, hvor rentesikringen gælder for hele lånets levetid (3 år). Rentesikringens niveau sætter vi til at være 6%. Tabellerne i datafilen skal nu opdateres med følgende 3 linier:

```
TABLE R(I,S) scenariotrae for renter
       Scen1 Scen2 Scen3 Scen4 Scen5 Scen6 Scen7
laan10 .04   .06   .03   .09   .06   .06   .03
laan11 .043  .06   .034  .06   .06   .06   .034;
```

```
TABLE K(I,S) scenariotrae for obligationskurser
       Scen1 Scen2 Scen3 Scen4 Scen5 Scen6 Scen7
laan10 .9917 .9814 .9819 .9932 .9922 .9922 .9821
laan11 .9917 .9814 .9819 .9932 .9922 .9922 .9821;

TABLE CallK(I,S) scenariotrae for konverteringskurser
       Scen1 Scen2 Scen3 Scen4 Scen5 Scen6 Scen7
laan10 1     1.000 1.000 1.000 1.000 1.000 1.000
laan11 1     1.000 1.000 1.000 1.000 1.000 1.000;
```

Vi har sprunget de første 9 obligationer over for at undgå gentagelse. Sættet af tilpasningsobligationer i' er nu opdateret med $Obl11$. Givet at bidragsbetalingen for $Obl11$ er sat op i forhold til andre obligationer, skal vi også opdatere datafilen, således at hver obligation tildeles det rigtige bidrag.[1]



Figur 5.2: Et scenariotræ for en multi–stadie stokastisk programmeringsmodel, med angivelse af et univers af obligationer. $Obl1$ til $Obl9$ repræsenterer fastforrentede obligationslån, $Obl10$ er rentetilpasningslån, $Obl11$ er rentetilpasningslån med rentesikring og $Obl12$ er fastforrentet kontantlån.

Endelig viser vi, hvordan et fastforrentet kontantlån kan tilføjes programmet. Et fastforrentetlån udmærker sig ved, at låntageren betaler en højere rente, kontantrenten. Til gengæld har han intet kurstab ved oprettelsestidspunktet, (se kapitel 2). Vi tilføjer et fastforrentet kontantlån til vores låneunivers. Dette lån er kontantlån–versionen af obligationslånet $Obl1$. I stedet for at håndtere kontantlånet i modellen kan vi introducere en fiktiv obligation, $Obl12$, hvis kontantrente er beregnet ved at bruge $Obl1$. Som kan ses i figur 5.2, har vi indført $Obl12$ for scenarierne 1, 2, 4, 5 og 6. For scenarierne 3 og 7, hvor kursen for $Obl1$ er over pari, udsteder vi heller ikke et kontant lån i denne obligation. Vi kan desværre ikke nøjes med at tilføje en linje i

[1]Foreløbig har vi brugt en skalar, bidrag, for at bestemme bidragssatsen. Vi kan nemt lave det om til en parameter, bidrag(I), eller endnu mere specifik et tabel, bidrag(I,S), hvilket stemmer overens med den måde, vi har indført bidragssatsen i modellen på, nemlig b_{is}.

hvert tabel i datafilen for at repræsentere det nye lån. Vi kan bedst beskrive grunden til dette ved at betragte følgende (forkerte) tilføjelse til tabellerne:

```
TABLE R(I,S) scenariotrae for renter
       Scen1 Scen2 Scen3 Scen4 Scen5 Scen6 Scen7
laan1  .05   .05   .05   .05   .05   .05   .05
laan12 .068  .108  1     .068  .182  .058  .058;

TABLE K(I,S) scenariotrae for obligationskurser
       Scen1 Scen2 Scen3 Scen4 Scen5 Scen6 Scen7
laan1  .9681 .9235 0     .8881 .9537 .9537 0
laan12 1.000 1.000 0     1.000 1.000 1.000 0;

TABLE CallK(I,S) scenariotrae for konverteringskurser
       Scen1 Scen2 Scen3 Scen4 Scen5 Scen6 Scen7
laan1  1     .9235 1.000 .8881 .9537 .9537 1.000
laan12 1     .9235 1.000 .8881 .9537 .9537 1.000;
```

Medmindre vi tilføjer ekstra betingelser i modellen, vil denne datarepræsentation betyde, at hvis vi optage kontantlånet (laan12), har vi med varierende renter for forskellige scenarier at gøre. Desuden vil vi, i scenarierne 2, 4, 5 og 6, sælge og indfri den samme obligation og tjene risikofrie penge. Hvis vi vil undgå ekstra betingelser, der gælder for kontantlån, kan vi benytte os af en ny obligation for hvert scenario, hvor $Obl12$ kan udstedes:

```
TABLE R(I,S) scenariotrae for renter
         Scen1 Scen2 Scen3 Scen4 Scen5 Scen6 Scen7
laan1    .05   .05   .05   .05   .05   .05   .05
laan12_1 .068  .068  .068  .068  .068  .068  .068
laan12_2 1     .108  1     .108  .108  1     1
laan12_4 1     1     1     .182  1     1     1
laan12_5 1     1     1     1     .058  .058  1;

TABLE K(I,S) scenariotrae for obligationskurser
          Scen1 Scen2 Scen3 Scen4 Scen5 Scen6 Scen7
laan1    .9681 .9235 0     .8881 .9537 .9537 0
laan12_1 1.000 0     0     0     0     0     0
laan12_2 0     1.000 0     0     0     0     0
laan12_4 0     0     0     1.000 0     0     0
laan12_5 0     0     0     0     1.000 1.000 0;

TABLE CallK(I,S) scenariotrae for konverteringskurser
         Scen1 Scen2 Scen3 Scen4 Scen5 Scen6 Scen7
laan1    1    .9235 1.000 .8881 .9537 .9537 1.000
laan12_1 1    .9235 1.000 .8881 .9537 .9537 1.000
laan12_2 1    1     1     .8881 .9537 1     1
laan12_4 1    1     1     1     1     1     1
laan12_5 1    1     1     1     1     1     1;
```

Laan12_1 til Laan12_5 kan nu opfattes som kontantlån udstedt i scenarierne 1 til 6 (Laan12_5 genbruges for scenario 6 eftersom de to scenarier er ens fra et lånetilbudsmæssigt synspunkt.), hvor lånet er oprettet ved at sælge $Obl12$.

Vi mangler kun at modellere kontantlånets skattepligtige kursgevinst.[2] Kursgevinsten realiseres hver gang vi indfrier en obligation. Vi definerer kursgevinsten matematisk på følgende måde:

$$Kursgevinst = P_{is} - Callk_{is}cdotP_{is} = P_{is}(1 - Callk_{is}),$$

hvor P_{is} er det indfriede beløb og $Callk_{is}$ er det faktisk betalte beløb. Kursgevinsten har hidtil indgået implicit i modellen som en reduktion af restgælden i balanceligningen, ligning 4.33 (side 67). Eftersom denne reduktion er skattefri for obligationslån, har den ikke indgået som led i definitionen på betalingen i ligning 4.36 (side 67). Med indføring af kontantlån skal vi dog opdatere ligningen på følgende måde:

$$B_s = \sum_i \Big(A_{is} + (1-\gamma)RG_{is} \cdot r_{is} + (1-\beta)RG_{is} \cdot b_{is} + m_{is} \cdot l_{is} + \alpha_i \cdot P_{is}(1 - Callk_{is}) \Big) \quad \forall s,$$

hvor α_i er skatteprocenten for kursgevinsten på obligation i. For obligationslån sættes denne sats lig nul og for kontantlån til den givne skattesats.

5.1.2 En metode til risikovurdering af rentesikring

I afsnit 3.4 (side 31) så vi to måder at regne prisen på en rentesikring på. I sidste afsnit tilføjede vi tilpasningslån og tilpasningslån med rentesikring til vores låneunivers. Et centralt spørgsmål omkring prisfastsættelsen af rentesikring er, hvorvidt prisen på rentesikring er i overensstemmelse med låntagerens eller realkreditinstituttets (RI) opfattelse af risiko. Med den udviklede optimeringsmodel i hånden har vi et ekstra værktøj til kontrol og eventuel justeringer af prisen på rentesikring.

Som udgangspunkt ved vi, at en låntager, som er likviditetsrisikoneutral, ingen præferencer har m.h.t. valget mellem et tilpasningslån og et tilpasningslån med rentesikring. Det var netop forudsætningen for prisfastsættelsen af rentesikringen. I praksis er den teoretiske pris dog ikke nødvendigvis den pris, som låntageren betaler for produktet. Givet at RI har ikke prisfastsat rentesikringen udelukkende på et teoretisk grundlag, kan optimeringsmodellen bruges til at teste, om prisen er over eller under den teoretiske pris. Vi kan således forsyne den likviditetsrisikoneutrale model alene med et tilpasningslån og et tilpasningslån med rentesikring. Hvis den optimale løsning anbefaler valget af et tilpasningslån med rentesikring, er det ensbetydende

[2] Som vi så i kapitel 2, er kursgevinsten på et obligationslån skattefri mens kursgevinsten på et kontantlån er skattepligtig for private låntagere.

med, at prisen er under den teoretiske pris. Forskellen til den teoretiske pris kan måles ved at forøge de faste omkostninger forbundet med optagelsen af et tilpasningslån med rentesikring, lige præcis indtil det ikke længere er optimal at optage dette lån.

Hidtil har vi kun betragtet det likviditetsrisikoneutrale tilfælde. Det er dog først når vi har en likviditetsrisikoavers låntager, at interessen for rentesikring melder sig for alvor. Desto mere risikoavers låntageren er, desto højere en pris vil han betale for at sætte et loft på rentestigningen. Denne pris skal dog ikke være så høj, at fordelen ved at optage et rentesikring forsvinder helt. Låntageren betaler en pris til RI for at beskytte sig mod en rentestigning, hvorimod RI køber denne risiko mod en betaling. Det er for tiden kun muligt at rentesikre et tilpasningslån for en periode op til 3 refinansieringer. Denne korte sikringsperiode er ikke nok til at interessere mange låntagere, når de fleste lån har en horisont på 30 år. RI er ikke villig til at købe likviditetsrisikoen for så lang en periode, til den risikoneutrale pris. Dette kan være udtryk for, at RI ikke har en likviditetsrisikoneutral–opfattelse/politik. Desuden er der juridiske begrænsninger m.h.t. det risikoniveau RI må påtage sig når et tilkøbsprodukt som rentesikring skal udbydes.

Vi har nu med den likviditetsrisikoaverse optimeringsmodel mulighed for at justere prisen på rentesikring, således at dette stemmer mere overens med RIs likviditetsrisikoopfattelse. Så længe der findes låntagere, der er mere likviditetsrisikoaverse end RI[3], er der teoretisk mulighed for at sælge rentesikring. Når RI sælger rentesikringer til likviditetsrisikoneutrale priser for maksimum af 3 refinansieringer, kan vi udarbejde et skema, som tildeler højere priser til rentesikringer, der strækker sig ud over de 3 år.

En heuristisk metode til at bestemme denne varierende pris på beskrives i det følgende:

1. Startprisen på rentesikringen for op til N refinansieringer findes via den likviditetsrisikoneutrale metode beskrevet i afsnit 3.4 (side 31).

2. Objektfunktionen i optimeringsmodellen omskrives således, at den afspejler en stigende grad af likviditetsrisikoneutralitet som funktion af tiden.

3. Begrænsninger der afspejler loftet for den risiko RI er villig til at påtage sig i ekstreme tilfælde, introduceres i modellen.

4. Modellen udstyres med et univers af lån bestående af et rentetilpasningslån, et rentetilpasningslån med rentesikring og et fastforrentet lån.

[3] Den mest likviditetsrisikoaverse låntagere optager et fastforrentet lån, og den mest likviditetsrisikovillige låntagere optager et F1 lån. Likviditetsrisikoskalaen for købere af rentesikring ligger imellem disse to yderligheder.

5. Den initielle optimale løsning, givet at objektfunktionen og begrænsninger afspejler et risikoniveau mellem et rentetilpasningslån og et fastforrentet lån, ville være et rentetilpasningslån med rentesikring.

6. Prisen på rentesikring forøges gradvist lige præcis indtil den optimale løsning skifter til et rentetilpasningslån. Denne er den nye pris på rentesikring, som stemmer overens med RI's risikoopfattelse/politik.

5.1.3 Førtidig indfrielse

Et lån kan indfris på et vilkårligt tidspunkt T under amortiseringsforløbet, således at $T < N$, hvor N er horisonten. Sættet af scenarierne reduceres til at repræsentere de første T år under låneforløbet, mens horisonten stadigvæk er N. Det betyder, at de terminslige afdrag og rentebetalinger bliver beregnet for et N årigt lån, mens vi kun betragter rentebevægelsen og derved lånetilbudsbevægelsen indtil tid T. Tidlig indfrielse kan bedst beskrives gennem følgende eksempel.

Vi betragter det sidste låneunivers i vores tidligere eksempel, figur 5.2 (side 86). Til forskel fra før forestiller vi os nu, at lånet har en horisont på 30 år, $N = 30$, men at vi skal indfri lånet efter 3 år, $T = 3$. Vi behøver kun to ændringer i datafilen: Tidshorisonten sættes til $N = 30$, og vi introducerer en ny skalar T og sætter den til 3 for vores eksempel. De enkelte betalinger for hvert scenario er nu væsentlig mindre end før, da vi havde en horisont på 3 år. Til gengæld er lånet ikke tilbagebetalt helt i løbet af de 3 år. Med andre ord gælder det for bladknuderne i scenariotræet:

$$RG_{is} - A_{is} \neq 0 \qquad \forall i, \forall s \in \{2^{T-1}, \cdots, 2^T - 1\},$$

hvor RG_{is} er restgældene ved de sidste beslutningsknuder og A_{is} er de sidste afdrag på lånet før indfrielsen indtræffer. Vi er selvfølgelig interesserede i at have så lille et indfrielsesbeløb som muligt. Vi betegner det samlede indfrielsesbeløb ved scenario s som IB_s, der defineres som:

$$IB_s = \sum_i (RG_{is} - A_{is}) \qquad \forall i, \forall s \in \{2^{T-1}, \cdots, 2^T - 1\}.$$

Vi tilføjer denne ligning til programmet for at tage højde for tidlig indfrielse. Objektfunktionen, 4.30 (side 67), skal dernæst opdateres som følgende:

$$\min \sum_s p_s \cdot \tau_s \cdot B_s + \sum_{s=2^{(T-1)}}^{2^T-1} p_s \cdot \tau_s \cdot IB_s. \tag{5.2}$$

Nu minimerer vi den gennemsnitlige betaling over alle scenarier lagt sammen med det gennemsnitlige infrielsesbeløb.

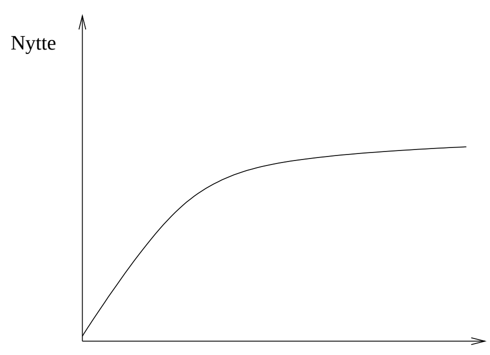
5.2 Modellering af risiko

Hidtil har vi forudsat, at låntageren er likviditetsrisikoneutral, d.v.s. at han gerne vil betale mindst muligt under det gennemsnitlige scenario. Dette er dog ikke særlig realistisk, eftersom huslånet typisk er den største investeringsobjekt for de fleste af låntagers vedkommende. De fleste låntagere er likviditetsrisiko–averse, d.v.s. at de gerne vil betale mindst muligt under det gennemsnitlige scenario, dog med den betingelse, at de ikke kan tåle at betale mere end en given overgrænse. I det følgende betragter vi 3 forskellige måder til at håndtere likviditetsrisiko–aversitet.

Til sidst vil vi betragte en måde, hvorpå vi kan tage højde for risiko–aversitet mod den anden form for risiko i realkreditlånesammenhænge nemlig formuerisikoen.

5.2.1 Nyttefunktion

I stedet for at minimere den totale omkostning ved et lån er det muligt at maksimere en konkav nyttefunktion af følgende form:



Figur 5.3: Skitsen viser en konkav nyttefunktion, hvor forøgelse af en stor besparelse ikke har den samme interesse som forøgelse af en lille besparelse.

Ideen med den faldende interesse for større besparelser er baseret på den tankegang, at de store gevinster typisk er mere risikofyldte end de små gevinster. Nielsen og Poulsen (se [18]) forslår brugen af en logaritmisk nyttefunktion i objektfunktionen:

$$\max \sum_s p_s \cdot \log(\tau_s \cdot (B_s^{maks} - B_s)), \tag{5.3}$$

hvor B_s^{maks} er det maksimale beløb, låntageren kan tåle at betale for et givet scenario. Besparelsen er defineret som $(B_s^{maks} - B_s)$, og nytten er defineret som $\log(B_s^{maks} - B_s)$. Nu maksimerer vi nytten i stedet for at minimere de totale omkostninger. Det gælder fortsat, at p_s er sandsynligheden, τ_s er tilbagediskonteringsfaktoren og B_s er den samlede betaling for scenario s.

Eftersom vi ikke måtte overskride budgetovergrænsen, B_s^{maks}, tilføjer vi følgende begrænsning i modellen:

$$\tau_s \cdot (B_s^{maks} - B_s) \geq 0 \qquad \forall s. \tag{5.4}$$

Problemet med denne nyttefunktion er dens ikke–linearitet. Sammen med heltallige variabler, l_{is}, kommer vi ind på en særdeles udfordrende familie af problemer, nemlig de ikke–lineære heltalsproblemer ("mixed integer non–linear programming", minlp, problemer). Der findes effektive solvere, f.eks. CPLEX, OSL m.f., til løsning af stor–skala heltallige problemer, (mip problemer). Ligeledes findes der effektive solvere såsom Mosek, Conopt m.f. til løsning af ikke–lineære problemer, (nlp problemer). For minlp problemer findes dog ikke idag lige så effektive standard solvere som findes for mip og nlp problemer, (se Bussieck og Pruessner, [9]).[4]

For at komme omkring dette problem, uden at bruge en minlp solver, har Nielsen og Poulsen ([18]) valgt ikke at tage hensyn til faste omkostninger og har derved reduceret problemet fra minlp til nlp. Transaktionsomkostninger beregnes alene som en fast procentdel af solgte obligationer. Dette er en approksimation til det oprindelige problem, som vi vil se på i afsnit 6 (side 99).

Alternativt kan vi bruge følgende målfunktion og begrænsning:

$$\max \sum_s p_s \cdot \tau_s (B_s^{maks} - B_s) \tag{5.5}$$

$$B_s^{maks} - B_s \geq 0 \qquad \forall s. \tag{5.6}$$

Målfunktion 5.5 bruger en lineær nyttefunktion, hvor en forøgelse af store besparelser er lige så interessant som en forøgelse af små besparelser. Vi kan bruge denne målfunktion samt begrænsning 5.6 til at sikre låntageren, at overgrænsen på budgettet ikke vil overskride for samtlige scenarier. Dette forudsætter, at renterne ikke vil stige eller falde mere end vores rentemodel forudsiger. Eftersom obligationsmarkedet er betydelig mere stabilt end for eksempel aktiemarkedet, er denne forudsætning ikke urimelig. Vi vil også teste denne lineære nyttefunktion i afsnit 6 (side 99).

[4] Dette er grunden til, at Institut for matematisk modellering på DTU har valgt ikke at investere i en licens til en minlp solver. Via GAMS er det muligt at bruge 4 minlp solvere, nemlig Baron, Dicopt, OQNLP og SBB.

5.2.2 Minimax kriteriet

For den ekstremt risikoaverse låntager kan det anbefales at bruge en min–max objektfunktion af følgende form:

$$\min \qquad MB, \tag{5.7}$$

hvor MB er en ny variabel, og den er defineret som maksimum betaling over alle mulige samlede scenarieforløb[5]:

$$MB \geq \sum_{s \in SF_f} B_s \qquad \forall f \in \{1, \cdots, 2^{T-1}\}, \tag{5.8}$$

hvor SF_f består af 2^{T-1} sæt af scenarier. Hvert af disse sæt beskriver et scenarieforløb. For vores eksempel med $T = 3$ har vi:

$$\begin{aligned} SF_1 &= \{1, 2, 4\} \\ SF_2 &= \{1, 2, 5\} \\ SF_3 &= \{1, 3, 6\} \\ SF_4 &= \{1, 3, 7\} \end{aligned}$$

Når vi benytter os af målfunktionen 5.7 samt begrænsningen 5.8, sikrer vi låntageren, at det bedste opnås, hvis det værste sker.

Minimax–formuleringen kan implementeres i GAMS ved at introducere et sæt af blade i beslutningstræet og et tabel, der med brug af 0 og 1–taller indikerer scenarieforløbene:

```
SET
       bl    bladnoderne      / blad1*blad4/;

table SF(s,bl) flag til brug for minimax formuleringen
        blad1   blad2   blad3   blad4
Scen1   1       1       1       1
Scen2   1       1       0       0
Scen3   0       0       1       1
Scen4   1       0       0       0
Scen5   0       1       0       0
Scen6   0       0       1       0
Scen7   0       0       0       1;
```

Følgende objektfunktion og begrænsning kan nu anvendes i modellen:

```
COST          ..   Z =E= MB;
DefMB(bl)     ..   MB =G= SUM(S, SF(S,bl)*B(S));
```

Vi vil se på resultater af denne strategi i afsnit 6 (side 99).

[5]Et samlet scenarieforløb skal forstås som en realisering af scenarier fra roden til et af bladene i beslutningstræet.

5.2.3 Budget begrænsninger og straffunktion

Minimax–kriteriet fra sidste afsnit er ekstremt begrænsende. I praksis betyder det, at låntageren ikke er villig til at tage nogle chancer. Mens det kan være en god strategi for nogen, er der andre, der er villige til at acceptere en begrænset risiko, hvis det betyder, at den forventede totale omkostning er mindre end minimax strategien. Minimax strategien kan således bruges som et benchmark, som vi kan sammenligne andre, mere risikofyldte, strategier med. Strategien med at bruge nyttefunktion er en af disse strategier. I det følgende udvider vi nyttefunktionskonceptet, således at vi tillader en vis overskridelse af budgetet.

Vi bruger målfunktionen (5.5) med den lineære nyttefunktion, og budgetbegrænsningen (5.6) fra afsnit 5.2.1 (side 91). Vi definerer en ny variabel BO_s, som står for budgetoverskridelsen ved scenario s. Ligeledes definerer vi en skalar BO_s^{maks}, som den maksimalt tilladte budgetoverskridelse og introducerer parametren $Straf_s$, som tilknytter en passende straf til en budgetoverskridelse ved scenario s. Vi kan nu benytte følgende målfunktion og tilføje de to begrænsninger til den oprindelige multi–stadie model[6]:

$$\max \sum_s \Big(p_s \cdot \tau_s \big((B_s^{maks} - B_s) - Straf_s \cdot BO_s \big) \Big) \tag{5.9}$$

$$B_s^{maks} + BO_s - B_s \geq 0 \qquad \forall s \tag{5.10}$$

$$BO_s \leq BO_s^{maks} \qquad \forall s. \tag{5.11}$$

Låntageren skal forsyne modellen med sit maksimale budgetniveau B_s^{maks}. Tilsvarende skal han bestemme en overgrænse for hvor meget dette budget højest må overskrides, BO_s^{maks}. Hvis budgettet ikke må overskrides, d.v.s. $BO_s^{maks} = 0$, får vi den almindelige nyttefunktionsformulering som et specielt tilfælde.

Her er det vigtigt at vælge værdierne B_s^{maks} og BO_s^{maks} med omhu. Et alt for unødvendig stramt budget kan føre til, at der ikke kan findes mulige løsninger, eller at nogle potentiale gode løsninger, med lidt risiko men god gennemsnitlig opførsel, bliver skåret væk. Af samme grund skal $Straf_s$ værdierne ikke være uhensigtsmæssig store. Eftersom både besparelsen og overskridelse af budgetet måles i penge–enheder, kan vi for eksempel benytte en høj rentesats som straf, svarende til, at pengene skal lånes fra en bank. Hvis vi sætter $Straf_s = 1,20$, svarer det til, at vi, ved scenario s, kan optage et banklån med en engangsrente på 20% til at betale budgetoverskridelsen med.

Vi vil i afsnit 6 sammenligne resultater fra denne formulering med de øvrige formuleringer i dette kapitel.

[6]Se formuleringen 4.30 til 4.38 (side 67).

5.2.4 Formuerisiko–aversitet

Forløbig har vi kun betragtet likviditetsrisiko, d.v.s. risikoen for, at låntageren kommer til at betale mere end beregnet for det gennemsnitlige scenario. Som nævnt i starten af dette kapitel findes dog en anden form for risiko, formuerisikoen, som er relevant for en låntager i et realkreditmarked.

Formuerisiko er risikoen for, at låntagerens restgæld bliver større end den forventede restgæld under amortiseringsforløbet. Formuerisikoen kan bedst beskrives i forbindelse med en voldsom kursstigning for de bagvedliggende obligationer. Hvis man eksempelvis optager et 30–årigt fastforrentet obligationslån, hvor man udsteder obligationer til kurs 80, har man en stor formuerisiko. Et lille rentefald betyder en stor stigning i kursen og dermed en forøgelse af ens restgæld i tilfældet af for tidlig indfrielse af lånet, eller en tilsvarende formindskelse af ejendommens friværdi. Formuerisikoen er en potentiel risiko, som bliver realiseret i det øjeblik, lånet pludselig skal indfris, eller når friværdien i ejendommen skal benyttes til optagelse af et andet lån.

Vi har allerede set på et specielt tilfælde af realiseret formuerisiko i afsnittet om tidlig indfrielse, (afsnit5.1.3 (side 90)), hvor vi præcis vidste, hvornår lånet skulle indfris. Til det tilfælde har vi allerede en model, der kan håndtere formuerisikoen. De tidligere beskrevne strategier til håndtering af likviditetsrisiko kan også benyttes til tilfældet med tidlig indfrielse, idet formuerisiko omdannes til likviditetsrisiko ved indfrielsestidspunktet. Vi vil se eksempler på dette i afsnit 6.

I resten af dette afsnit udvikler vi en strategi til at håndtere formuerisikoen i al almindelighed, d.v.s. når låntageren ingen anelse har om en mulig indfrielses– eller friværdiafbenyttelsestidspunkt. Vi vil også betragte den situation, hvor låntageren alene med en vis sandsynlighed kan pege på et tidsinterval, hvor formuerisikoen kan realiseres.

Formuerisikoen kan realiseres når som helst:

Vi tager udgangspunkt i målfunktionen (5.9). Denne målfunktion udtrykker det besparede beløb i forhold til et givet budget, samt eventuel straf af budgetets overskridelse. Denne værdi er sammenlignelig med den potentielle besparelse, der er gemt i en relativ lav restgæld. Vi definerer den gennemsnitlige restgæld for hver periode som $\overline{RG}_t$, hvor $t = \lfloor \frac{\ln s}{\ln 2} \rfloor$. Afvigelsen fra denne gennemsnitlige restgæld ved scenario s, er derfor:

$$AGR_s = \overline{RG}_t - \sum_i RG_{is}, \quad \forall s.$$

Hvis dette beløb er positivt, er det ensbetydende med, at vi har sparet i.f.t. gennemsnittet, eftersom vi har en restgæld, der er mindre end den gennemsnitlige restgæld, og hvis beløbet er negativt, har vi en formuerisiko.

Beløbet AGR_s kan, med en passende vægtning, indgå i målfunktionen. Hvis $AGR_s < 0$ tillægger vi beløbet en straf, og hvis $AGR_s > 0$, tillægger vi beløbet en belønning. Vi har dog brug for et par indikatorvariable, som registrerer den numeriske værdi af AGR_s.[7] Til det formål introducerer vi variablene RGB_s for restgældsbesparelse og RGU_s for restgældsunderskud. Vi tilføjer dernæst følgende begrænsning til modellen.

$$\big(\overline{RG}_t - \sum_i RG_{is}\big) - RGB_s + RGU_s = 0 \qquad \forall s, \tag{5.12}$$

Variablene RGB_s og RGU_s kan nu indgå i målfunktionen med en passende vægtning som følgende:

$$\max \sum_s p_s \cdot \tau_s \Big((B_s^{maks} - B_s) - Straf_s \cdot BO_s + PV_s \cdot RGB_s - NV_s \cdot RGU_s \Big), \tag{5.13}$$

hvor PV_s, en forkortelse for positiv vægt, er en vægt der bruges til belønning af positive AGR_s værdier og NV_s, kort for negativ vægt, bruges til at straffe en negativ værdi af AGR_s. Hvis vi sætter $PV_s = NV_s$, er det ensbetydende med, at vi er indifferente m.h.t. formuerisikoen. Hvis $PV_s < NV_s$, betyder det derimod, at vi er formuerisikoaverse, eftersom vi straffer et potentielt underskud hårdere, end vi belønner en potentiel overskud. Nu mangler vi kun at definere et hensigtsmæssigt niveau for den gennemsnitlige restgæld, $\overline{RG}_t$. Denne værdi kan bestemmes af låntageren eller som skøn. Denne værdi er nemlig alene udtryk for en opfattelse af, hvilken restgæld, der er acceptabel til hvilket scenario. Alternativt kan vi bruge formuleringen, hvor tiden indgår eksplicit. $\overline{RG}_t$ kan nu defineres som den gennemsnitlige betaling for hver periode:

$$\overline{RG}_t = \sum_{is} p_{ts} \cdot RG_{its}, \quad \forall t.$$

Igen skal vi være opmærksomme på valg af parametrene, $\overline{RG}_t$, PV_s og NV_s. En for lav værdi af $\overline{RG}_t$ samt en for høj værdi af NV_s betyder for eksempel, at modellen er ekstrem formuerisikoavers. Vi vil i afsnit 6 se på et eksempel med formuerisikoaversitet.

Formuerisikoen kan alene realiseres i en given periode med en vis sandsynlighed:

Det forekommer undertiden, at en låntager kan pege på en begrænset periode i fremtiden, hvor lånet muligvis skal indfris. Det drejer sig om udsagn som "Jeg regner med at sælge huset efter 5 til 10 år".

[7] Sådanne variable betegnes som "surplus" og "slack", svarende til en overskud eller underskud i.f.t. en given værdi.

Vi kan bruge den allerede udviklede formuerisikoaverse model til sådanne situationer. For at modellere det nævnte udsagn skal vi for eksempel alene sætte PV_s og NV_s parametrene lig nul for alle beslutningsknuder, som ligger uden for perioden mellem det 5. og 10. år. Ligeledes skal en tidlig indfrielse medtages, hvis lånet har en horisont, der er over 10 år.

5.3 Tilføjelser til Nielsen & Poulsen modellen

Vi har i afsnit 4.2 og 4.4 set på en detaljeret gennemgang af den grundlæggende optimeringsmodel, inspireret af N&P–modellen, beskrevet i afsnit 4.3 (side 60). I afsnit 4.5 gennemgik vi modelleringstekniske spørgsmål og vi analyserede en ny scenario–reducerende teknik beskrevet i [11, 12, 13]. Modellen blev videreudviklet til at tage højde for flere vigtige praktiske detaljer i afsnit 5.1 og 5.2.

Vi vil i det følgende opsummere denne rapports væsentligste bidrag til modellering og løsning af låneanbefalingsproblemet:

- Grundmodellen, inspireret af N&P–artiklen, er blevet analyseret i detaljer, trinvist opbygget og præsenteret som en samlet model, (afsnit 4.2–4.4).
- De faste transaktions– og omlægningsomkostninger er der blevet taget højde for, (afsnit 4.2–4.4).
- En scenarioreducerende algoritme, (SCENRED modulen i GAMS), er beskrevet i afsnit 4.5.
- En modelleringsstrategi til præsentation af inputdata forbundet med fastforrentede obligationslån, kontantlån, såvel som tilpasningslån er foreslået, (afsnit 5.1).
- En algoritme til prisfastsættelse af tilkøbsproduktet rentesikring er udviklet, og en strategi til risikovurdering af dette produkt er blevet skitseret (afsnit 3.4 samt 5.1).
- Førtidig indfrielse er blevet tilføjet modellen, (afsnit 5.1).
- Nyttefunktion og minimax–strategier til håndtering af likviditetsrisikoen, er blevet analyseret og modelleret, (afsnit 5.2).
- Nye begrænsninger til bestemmelse af en budgetovergrænse med en straffunktion til eventuelle overskridelser af budgetet blevet tilføjet modellen, (afsnit 5.2).
- Formuerisiko er blevet introduceret i modellen således, at den kan afspejles i dele af eller alle knuder i beslutningstræet, (afsnit 5.2).

Optimeringsmodellen med alle tilføjelser er implementeret i GAMS og koden til de forskellige versioner af modellen kan ses i appendix E.

I næste kapitel ser og kommenterer vi resultater, opnået fra adskillige kørsler af GAMS programmerne.

Kapitel 6

Resultater

Der er to overordnede formål med dette kapitel:

1. At validere de forskellige versioner af optimeringsmodellen, der blev udviklet i de sidste to kapitler. Vi skal med andre ord se om modellerne beskriver den virkelighed, som de skal afspejle.

2. At observere og kommentere effekten af scenarioreducering (GAMS/SCENRED modulen) på de opnåede resultater.

Flere GAMS programmer, hvert svarende til en af de i de sidste to kapitler udviklede versioner af modellen, er blevet udviklet under udarbejdelsen af dette projekt. Til hver version af modellen findes en GAMS formulering til løsning af det ikke–reducerede problem og en GAMS/SCENRED formulering til løsning af det tilsvarende reducerede problem.

Vi vil i dette kapitel se på de opnåede resultater, der er baseret på kørsler af disse GAMS programmer. Koden findes i appendix E (side 161).

I afsnit 6.1 ser vi en beskrivelse af de underliggende data, som er blevet brugt til generering af et sceanriotræ. Vi vil dernæst i afsnit 6.2 se resultater fra de likviditetsrisikoneutrale modeller. Her vil vi også kommentere effekten af scenarioreducering på vores grundproblem. Ligeledes vil vi foretage en analyse af prisen på rentesikring.

Cplex kan løse grundproblemet både i sin originale form og i sin reducerede form inden for 300 CPU sekunder. Til gengæld er det ekstremt tidskrævende at løse de risikoaverse modeller med det ikke–reducerede træ. I afsnit 6.3 ser vi på løsning af de risikoaverse versioner af modellen med det originale scenariotræ. Scenarioreducering bliver brugt i afnit 6.4 til hurtig generering af gode løsninger. Endelig ser vi på nogle afsluttende kommentarer til de fundne resultater i dette kapitel i afsnit 6.5.

6.1 Datagrundlaget

Vi så i starten af denne rapport i diagram 1.1 (side 4), at det er nødvendigt at anvende en rentemodel og flere prisfastsættelsesmetoder for at generere de nødvendige kurser og renter til de forskellige obligationer, der skal bruges som input i optimeringsmodellen.

I kapitel 3 udviklede vi BDT modellen, som leverer den underliggende rentestruktur for optimeringsmodellen. Vi har også set på en prisfastsættelsesmodel til nutidsværdibestemmelse af prisen på en rentesikring. Ligeledes har vi anvendt en simpel prisfastsættelsesmodel til at generere priser på inkonverterbare F1 lån. Priserne på konverterbare obligationer i denne rapport stammer fra approksimationer, foretaget af produktlaboratoriet i Nykredit baseret på BDT renterne og priserne på inkonverterbare obligationer.

I alle datasæt brugt i følgende kørsler betragter vi 30–årige annuitetslån, der bliver indfriet om 11 år. Det vil sige, at vi anvender modellen med førtidig indfrielse, beskrevet i afsnit 5.1.3 (side 90), som udgangspunkt. Basemodellen, d.v.s. modellen uden førtidig indfrielse, er indeholdt i denne model, og vi vil derfor ikke foretage adskilte kørsler med basemodellen.

Hvis priserne til konverterbare obligationer bliver genereret for hele 30 års–perioden, kan modellen uden yderligere justeringer bruges til at generere anbefalinger for lån, der ikke skal indfries før tid.

I de følgende testkørsler har vi årlig "recourse", d.v.s. vi kan foretage beslutninger ved oprettelsestidspunktet og ved slutning af hvert år i de næste 10 år. Antallet af blade i det ikke reducerede scenariotræ er 1024, og der er i alt 2047 beslutningsknuder. I det følgende bruger vi ordet scenario som et hændelsesforløb fra roden til et af bladene, d.v.s. vi har 1024 scenarier i det ikke–reducerede problem.

Vi bruger 24 fastforrentede obligationer med følgende kuponer, gennemsnitlige startkurser samt udstedelses– og udløbsdato:

Det er ikke praktisk muligt at angive de enkelte kurser for alle lån i alle knuder i BDT–træet her. Vi nøjes derfor i datatabellen 6.1 med at angive de gennemsnitlige startkurser, hver gang en ny obligation bliver åbnet til udstedelse. Som kan ses fra tabellen, bliver 30–årige fastforrentede lån oprettet i højest 33–årige obligationer. Dette skyldes, at RI åbner fastforrentede obligationsserier hver 3. år.

I det følgende, med mindre andet angives, vil laan25 svare til et F1 lån, hvor den effektive obligationsrente er på cirka 2% ved starttidspunktet. Den effektive rente kan højest stige til ca. 21% og falde til lidt under 1% for de yderste scenarier ved den 10. refinansiering. Tilsvarende er laan26 et F1 lån med laan25 som det underliggende lån. Sikringsniveauet er på 6% for hele perioden, d.v.s. den effektive rente må ikke stige mere end 6%.

GAMS betegnelse	Kupon	Gnmst. startkurs	Udstd. dato	Udl. dato
laan1	6%	103.06	3/10-02	3/10-35
laan2	5%	98.5	3/10-02	3/10-35
laan3	4%	89.4	3/10-02	3/10-35
laan4	9%	107.33	3/10-05	3/10-38
laan5	8%	103.16	3/10-05	3/10-38
laan6	7%	103.09	3/10-05	3/10-38
laan7	6%	100.51	3/10-05	3/10-38
laan8	5%	94.01	3/10-05	3/10-38
laan9	4%	84.55	3/10-05	3/10-38
laan10	3%	74.46	3/10-05	3/10-38
laan11	9%	105.4	3/10-08	3/10-41
laan12	8%	101.98	3/10-08	3/10-41
laan13	7%	100.3	3/10-08	3/10-41
laan14	6%	96.19	3/10-08	3/10-41
laan15	5%	89.5	3/10-08	3/10-41
laan16	4%	80.74	3/10-08	3/10-41
laan17	3%	71.32	3/10-08	3/10-41
laan18	9%	104.41	3/10-11	3/10-44
laan19	8%	100.9	3/10-11	3/10-44
laan20	7%	98.51	3/10-11	3/10-44
laan21	6%	94.07	3/10-11	3/10-44
laan22	5%	87.49	3/10-11	3/10-44
laan23	4%	79.25	3/10-11	3/10-44
laan24	3%	70.26	3/10-11	3/10-44

Tabel 6.1: De underliggende konverterbare obligationer til testkørslerne i dette kapitel.

Et praktisk problem i afprøvning af optimeringsmodellen er udskrivning af GAMS datatabeller. Problemet opstår ikke alene p.g.a. omfanget af tabellerne, men også fordi optimeringsmodellen bruger et ikke kombinerende binomialtræ som scenariotræ, hvorimod de underliggende data kommer fra et kombinerende binomialtræ (en “lattice”). Dette problem er løst ved at implementere VBA rutiner til generering af GAMS tabeller ved at mappe knuderne fra det kombinerende træ over til det ikke–kombinerende træ. Koden kan ses i appendiks D.1 (side 149).

På grund af problemets omfang og begrænset adgang til data for fastforrentede obligationer er det ikke praktisk muligt, inden for projektets rammer, at foretage en udtømmende afprøvning af modellen.

Vi vil dog teste adskillige versioner af modellen på de tilgængelige obligationsdata og kommentere resultaterne. På de fleste problemer anvender vi både en GAMS formulering, hvor alle scenarier er med, og en GAMS/SCENRED formulering med et reduceret antal scenarier.

6.2 Likviditetsrisikoneutrale modeller

Vi vil i dette afsnit se på forskellige kørsler af modellen med førtidig indfrielse for den likviditetsrisikoneutrale låntager (se afsnit 5.1.3 (side 90)). Vi vil sammenligne løsninger for det ikke–reducerede problem (GAMS formulering) med de reducerede problemer (GAMS/SCENRED formulering). Vi vil også se, hvordan modellen kan bruges til at vurdere prisniveauet for rentesikring.

6.2.1 Fastforrentede obligationslån

Her ser vi kun på de fastforrentede obligationer som angivet i tabel (6.1 (side 101)).

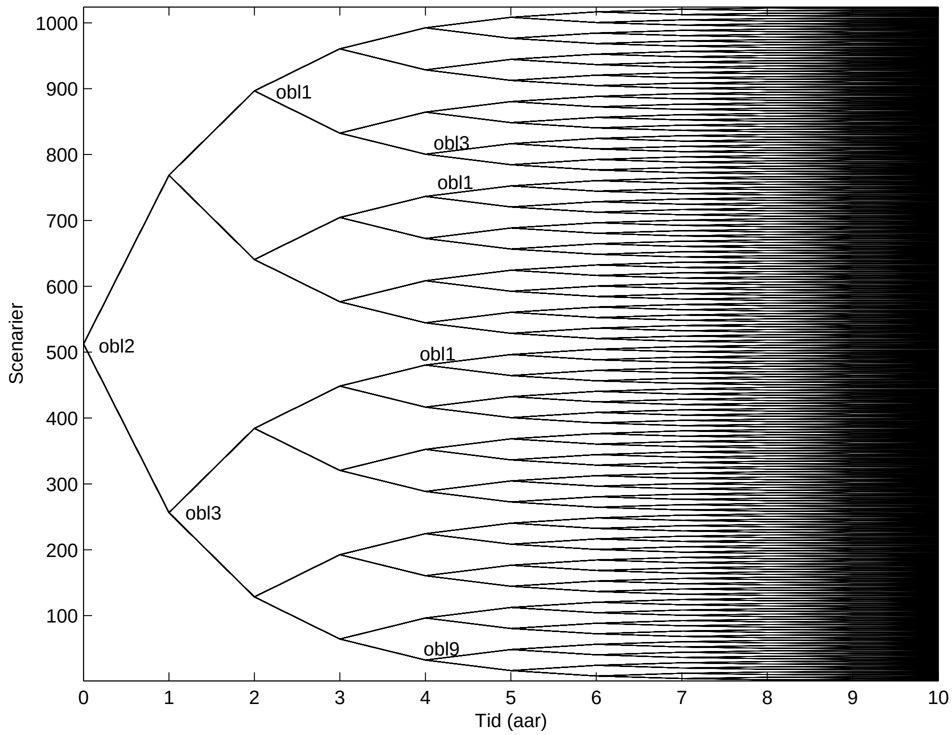
Løsning ved GAMS formulering

Når vi løser det originale problem for de 24 fastforrentede obligationer, får vi en målfunktionsværdi på **1.322.067,7810**. Løsningen repræsenteres som en sekvens af op– og nedkonverteringer. En fuldstændig gengivelse af løsningen for hele scenariotræet (med 2047 knuder) bidrager ikke med megen indsigt, hvorfor vi vil nøjes med at se på løsningen for de første 5 perioder (de første 31 beslutningsknuder). Resultatet repræsenteres både grafisk (i figur 6.1) og i tabel form.

```
----  15881 VARIABLE Sale.L  Salgsvariabel

              Sc1         Sc3         Sc4        Sc19        Sc20

laan1                              879678.866              855341.905
laan2  1018018.935
laan3              1039817.784              951846.432

    +    Sc24         Sc31

laan1     820022.731
laan9               1113490.551
```

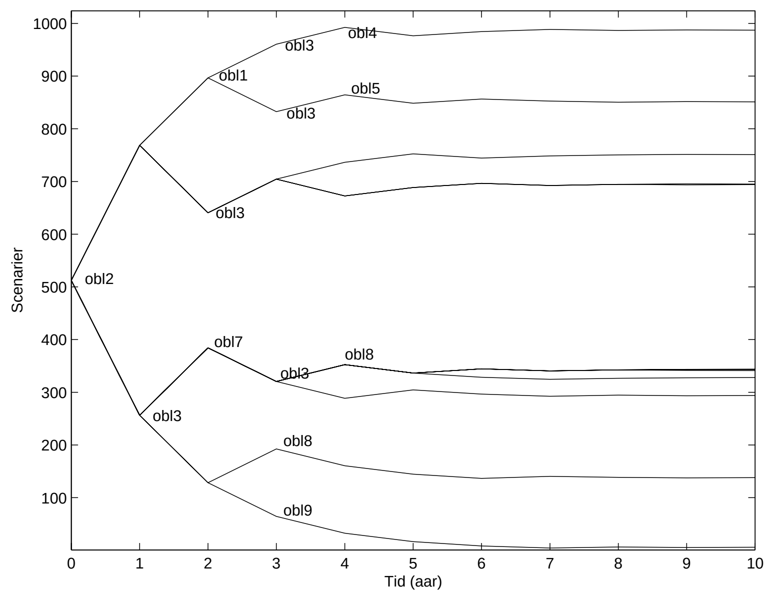
Løsning ved GAMS/SCENRED formulering

Vi løser samme problem med de 24 fastforrentede obligationer, men vi reducerer antallet af scenarier fra 1024 til 12 ved at bruge SCENRED fast-backward + forward algoritmen. Antallet af knuder bliver reduceret fra 2047 til 79. Vi får **1.244.874,2288** som værdien af målfunktionen. Løsningen ses både i tabelform og i figur 6.2, hvor vi også kan se, hvordan SCENRED har reduceret de oprindelige scenarier.



Figur 6.1: Scenariotræet med 10 perioder før reducering.

```
----  17045 VARIABLE Sale.L  Sale variable

                      Sc1         Sc3         Sc4         Sc5         Sc6

laan1 .time2                             879678.866
laan2 .time0  1018018.935
laan3 .time1              1039817.784
laan3 .time2                                        1051483.013
laan7 .time2                                                       933978.979

        +             Sc8         Sc9        Sc13        Sc14

laan3 .time3     1059514.567 1019094.882  976891.778
laan8 .time3                                            997326.788



        +            Sc15        Sc16        Sc18        Sc26

laan4 .time4               689738.053
laan5 .time4                             792811.983
laan8 .time4                                         928890.757
laan9 .time3  1123585.482
```



Figur 6.2: Scenariotræet med 10 perioder efter reducering.

Ved sammenligning af løsninger noterer vi, at selv med så kraftig en reducering af problemet, får vi den samme løsning for de første 2 perioder. Fra den 3. periode sker flere konverteringer i det reducerede træ i forhold til det oprindelige problem. Dette er naturligt, eftersom der er færre muligheder for bevægelse af renter i det reducerede træ, og det gøre det mere attraktivt at omlægge lånet, lige så snart der er udsigt til besparelse. I det oprindelige træ er der derimod mange muligheder for renternes bevægelse, og det vil, alt andet lige, øge muligheder for yderligere besparelse, hvis låntageren afholder sig fra omlægning i en tid for at omlægge til den store besparelse på et senere tidspunkt.

Vi kan ikke desto mindre nøjes med at bruge det reducerede scenariotræ, så længe vi får den samme strategi i starttidspunktet. Til næste refinansieringstidspunkt, når vi har fået nye informationer, kan vi køre programmet igen og revidere løsningen.

Selvom vi har fået den samme løsning i de første 2 perioder, er der ingen garanti for at, vi vil få den samme startløsning for et vilkårligt låneanbefalingsproblem. Vi vil i det følgende foretage flere kørsler både med det oprindelige problem (GAMS formulering) og det reducerede problem (GAMS/SCENRED formulering).

6.2.2 Fastforrentede obligationslån og F1 lån

Vi tilføjer nu et F1 lån (laan25) til vores låneunivers, så vi i alt har 25 underliggende obligationer at vælge imellem.

Løsning ved GAMS formulering

For det ikke reducerede problem får vi en målfunktionsværdi på **1.300.707,1531**. Ved at tilføje et F1 lån til vores betragtninger får vi en gennemsnitlig besparelse på **21.360** kr. over en 10–årig periode for et 30–årigt lån. Vi skal dog huske, at dette medfører en større likviditetsrisiko end tilfældet var med fastforrentede obligationer alene. Vi angiver løsninger for de første 3 år:

```
----  16495 VARIABLE Sale.L  Salgsvariabel

               Sc1         Sc2         Sc4

laan1                              915678.515
laan3              1128109.949
laan25 1000000.000
```

Løsning ved GAMS/SCENRED formulering

Vi vil igen reducere antallet af scenarier fra 1024 til 12 ved at bruge SCENRED fastbackward + forward algoritmen. Vi får **1.187.912,1961** som værdien af målfunktionen. Dette kan sammenlignes med det tilsvarende reducerede problem uden F1 lån. En besparelse på 56.962 kr. opnås her. Vi får følgende løsning i de første tre perioder:

```
----  17659 VARIABLE Sale.L  Sale variable

                      Sc1         Sc3         Sc4       Sc5       Sc6

laan1 .time2                          1009169.676
laan3 .time1              1010771.540
laan3 .time2                                      1049185.216
laan25.time0  1000000.000
laan25.time2                                                  900353.753
```

Her får vi den samme løsning, som vi fik ved det ikke reducerede tilfælde, d.v.s. vi starter med et F1 lån. Vi kan allerede i anden periode konstatere forskel mellem strategierne. Igen kan vi godt acceptere resultatet af det reducerede træ, eftersom vi får den samme startløsning.

6.2.3 Fastforrentede obligationslån, F1 lån og F1 lån med rentesikring

I dette afsnit demonstrerer vi hvordan optimeringsmodellen kan bruges til vurdering af en rimelig pris på en rentesikring på 6% for alle 10 refinansieringer. Når vi bruger optimeringsmodellen til prisfastsættelsesspørgsmål, kan vi ikke bruge et reduceret træ, eftersom vi har brug for al den information, der er repræsenteret i de enkelte knuder. Med andre ord har vi brug for så præcise målfunktionsværdier, som kan opnås ved at bruge det ikke–reducerede scenariotræ. Derfor betragter vi alene GAMS–formuleringen af modellen for det ikke–reducerede træ.

Kan rentesikring betale sig?

Vi løser problemet, med 26 lån, uden at kræve ekstra omkostninger for brugen af F1 lånet med rentesikring. Dette skal medføre, at F1 lånet med rentesikring bliver meget attraktivt. Vi får som forventet en lavere målfunktionsværdi på **1.285.529,1360**, en besparelse på **15.178** kr. Vi ser i løsningen, at ved starttidspunktet er F1 lånet blevet erstattet af F1 lånet med rentesikring. Men når vi kommer lidt længere frem i tiden ser vi, at F1 lånet (laan25) også bliver valgt. Det svarer til, at det ikke betyder noget for den optimale løsning at vælge mellem F1 lån og F1 lån med rentesikring i disse knuder. Denne observation kan bruges til at rådgive låntageren om, hvornår det kan svare sig at rentesikre sit lån.

```
----  17111 VARIABLE Sale.L  Salgsvariabel

             Sc1         Sc2         Sc4         Sc8        Sc11

laan1                               915678.515
laan3                   1128109.949
laan25                                                  1018713.348
laan26  1000000.000                             779511.163
```

Læg mærke til, at også andre lån end laan25 og laan26 bliver valgt i den optimale løsning. Det medfører, at vi får en mindre besparelse i.f.t. den teoretiske pris vores algoritme (som udviklet i afsnit 3.4 (side 31)) finder, nemlig **28.386** kr.

Denne forskel kan forklares med, at i vores prisfastsættelsesmodel for rentesikring betragter vi kun et F1 lån og et F1 lån med rentesikring, hvorimod vi i optimeringsmodellen har mulighed for at kigge på andre låneprodukter såsom fastforrentede obligationslån. For at se, om denne argumentation kan støttes af empiri, kan vi foretage 2 ekstra kørsler, hvor programmet bliver forsynet kun med et F1 lån i den første kørsel og dernæst med et F1 lån og et F1 lån med rentesikring.

Et F1 lån som inputdata

Når vi forsyner modellen alene med et F1 lån, vælger modellen naturligvis lånet ved startpunktet og derefter beholder lånet indtil indfrielsen. Den totale gennemsnitlige låneomkostning bliver **1.329.344,7568**.

Et F1 lån og et F1 lån med rentesikring som inputdata

Når vi forsyner modellen alene med et F1 lån og et F1 lån med et sikringsniveau på 6% for alle 10 refinansieringsperioder, (uden at indføre nogle ekstra omkostninger til målfunktionen), vælger modellen, som forventet, F1 lånet med rentesikring ved startpunktet og beholde lånet indtil indfrielsen. Den totale låneomkostning bliver **1.302.903,5672**. Besparelsen på **26.441** kr. stemmer godt overens med prisniveauet fra prisfastsættelsesmodellen nemlig **28.386** kr.[1] Vi så imidlertid, at hvis låntageren samtidig ville benytte sig af omlægningsmulighederne til fastforrentede lån, får rentesikringen en betydelig mindre værdi, nemlig 15.178 kr. Vi har hermed med anvendelse af optimeringsmodellen fået et effektivt analytisk værktøj til at studere rimeligheden i prisen på en rentesikring set med låntagerøjne.

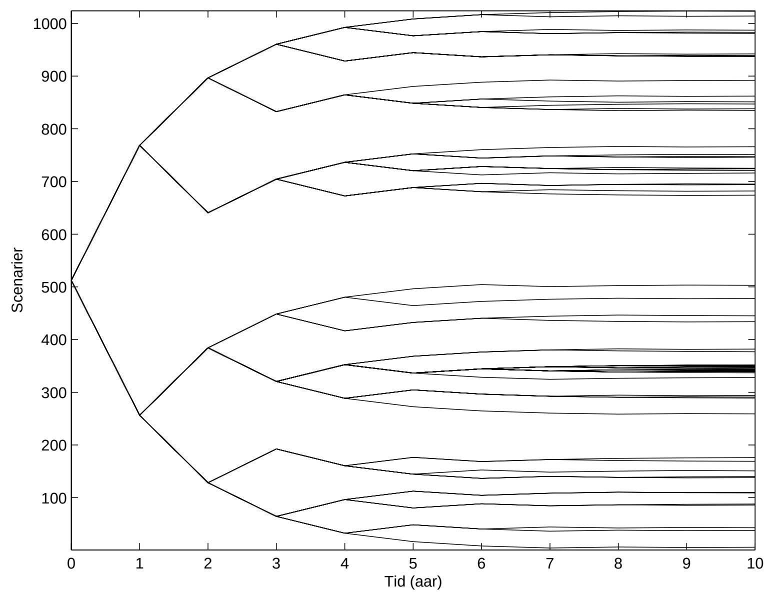
6.2.4 Hvorfor får vi mindre målfunktionsværdier med GAMS/SCENRED formuleringer?

Vi har hidtil observeret, at reducerede problemer resulterer i mindre målfunktionsværdier end de mindre reducerede problemer. Selvom vi hovedsageligt er interesserede i at opnå samme startløsninger for de reducerede og ikke–reducerede problemer, er det vigtigt at forstå, hvorfor vi systematisk får mindre målfunktionsværdier for de reducerede problemer.

Eftersom vi har lige sandsynligheder for alle blade ved det oprindelige scenariotræ, forventer vi at få et jævnt fordelt reduceret scenriotræ som output fra SCENRED. For at undersøge om dette er tilfældet, ser vi nærmere på fordeling af de tilbageværende scenarier, i forhold til de oprindelige scenarier.

Betragt figur (6.3), hvor 64 scenarier ud af 1024 oprindelige scenarier er tilbage. Umiddelbart er det svært at se et system i den måde, hvorpå scenarierne er blevet reduceret. Hvis vi inddeler de 1024 oprindelige scenarier i 4 lige store dele, kan vi dog konstatere, at der er 37 scenarier i de to midterste inddelinger i modsætning til de 27 scenarier i de 2 yderste inddelinger. Men eftersom vi ikke længere har lige sandsynligheder for alle scenarier skal vi betragte summen af sandsynlighederne for hver inddeling i stedet for summen af scenarier.

[1] Forskellen skyldes justeringer af det teoretiske F1 obligationspriser foretaget af produktlaboratoriet baseret på de empiriske erfaringer.



Figur 6.3: Scenariotræet med 10 perioder efter reducering fra 1024 til 64 blade.

I følgende tabel ser vi antallet af scenarier, summen af sandsynligheder for scenarierne for de 4 inddelinger for reduceringer fra de oprindelige 1024 scenarier til 12 scenarier, (figur 6.2 (side 104)), 64 scenarier, 400 og 800 scenarier.

Perioder	12	$\sum_s p_s$	64	$\sum_s p_s$	400	$\sum_s p_s$	800	$\sum_s p_s$	1024	$\sum_s p_s$
1-256	2	0,142	15	0,189	146	0,2521	176	0,2518	256	0,25
257-512	3	0,258	12	0,179	35	0,091	201	0,2461	256	0,25
513-768	5	0,479	25	0,463	132	0,443	206	0,2572	256	0,25
769-1024	2	0,12	12	0,183	87	0,2359	217	0,2537	256	0,25
Z*	1.183.652		1.205.759		1.246.465		1.278.484		1.285.529	

Tabel 6.2: Tabellen viser scenariofordelinger for det oprindelige ikke–reducerede problem og for 3 reducerede problemer. Nummereringen er fra 1, (højeste renteniveau), til 1024, (laveste renteniveau).

Vi ser, at den optimale målfunktionsværdi, (Z^*), stiger med antallet af scenarierne. Men vi også ser, at summen af sandsynligheder for de 2 nederste inddelinger konsekvent er større end de to øverste.

Hvis dette viser sig at være et generelt mønster for SCENRED algoritmerne, betyder det for vores løsninger, at vi får en underrepræsentation af scenarierne med høje renter.

Vi vil i det følgende foretage flere tests for at se, om dette er en systematisk favorisering af de nederste scenarier, eller om det kan tilskrives tilfældigheder.

6.3 Risikoaverse modeller med GAMS formuleringer

I de følgende afsnit består vores datagrundlag af de 24 fastforrentede obligationer (laan1-laan24) samt F1–obligationen (laan25).

6.3.1 Minimax modellen

Vi beskrev minimax modellen i afsnit 5.2.2 (side 93). Minimax formuleringen sikrer, at låntageren får det billigste lån, hvis det værste scenario i scenariotræet forekommer. En meget likviditetsrisikoavers låntager foretrækker et fastforrentet lån. Sådanne låntagere sikrer sig mod det værst tænkelige m.h.t. en eventuel rentestigning. Denne rapports meget likviditetsrisikoaverse låntager tror dog, at det værste scenario i vores scenariotræ vil forekomme. Givet, at vores model tillader de korte renter til at stige op til 21%, er det interessant at observere, om minimax–formuleringen fravælger F1–lånet i låneporteføljen.

GAMS–koden for minimax–formuleringen kan ses i appendix D.1. VBA–koden til generering af GAMS–tabellen til brug for identificeringen af samtlige scenarieforløb fra roden til blade i træet, kan ses i appendix E.

Minimax–formuleringen bidrager med 1024 ekstra begrænsninger til vores grundproblem i sidste afsnit. Tidsforbruget for at opnå en løsning stiger dog fra 275,6 sekunder for grundproblemet til over 10 timer for at finde den første heltalsløsning. Selv efter 100.000 sekunder (27,7 timer) har vi ikke fundet en løsning, der har den samme præcision som grundproblemet. Den fundne løsning har en relativ afstand på 8% fra en nedregrænse i "branch and cut"–træet. Dette er ikke nødvendigvis ensbetydende med, at vi ikke har fundet den optimale løsninger, men at vi ikke har ventet i tilstrækkelig lang tid for at få beviset på optimalitet fra Cplex. Foreløbig godtager vi løsningen som en god nær optimal minimax løsning. Vi vil vende tilbage til køretidsproblemet.

Vi får **1.353.713,43** kr. som værdien af målfunktionen. Til det beløb skal vi tilføje faste oprettelsesomkostninger på cirka 19000 kr., for at beløbet kan sammenlignes med den gennemsnitlige omkostning ved det risikoneutrale tilfælde. Følgende løsning opnås:

```
---- 226639 VARIABLE Sale.L  Salgsvariabel
                Sc1       Sc577       Sc592       Sc641       Sc672      Sc1759

laan2  1018018.935
laan10            639217.708              639217.708  639217.708
laan12                        637829.111
laan25                                                            825292.341
```

Modellen vælger et fastforrentet lån (5% obligationslån med en kurs på 98,23) og beholder dette lån helt indtil den 9. refinansiering, hvor lånet bliver enten nedkonverteret til et 3% obligationslån eller opkonverteret til et 8% obligationslån. Endelig vælges der ved den 10. refinansiering et F1 lån med en effektiv rente på 1,71%.

Med denne strategi får vi en standardafvigelse på **19.729** kr. fra den gennemsnitlige betaling på **1.353.713** kr. Med et tilsvarende univers af obligationer for den risikoneutrale model fik vi en målfunktionsværdi på 1.300.707,1531. Når vi fratrækker oprettelsesomkostningerne fås en gennemsnitlig total omkostning på **1.281.857,15** kr. Med andre ord skal låntageren betale **71.856** kr. mere for den sikkerhed, han får med minimax–strategien.

Vi samler de to foreløbige alternativer i tabel 6.3.

Lånestrategi	Totale omk.	Std. Afv.	max	min
Risikoneutral	1.281.857	92.289	1.502.042	1.004.583
Minimax	1.353.713	19.729	1.374.183	1.117.084

Tabel 6.3: De totale omkostninger er beregnet uden hensyntagen til engangsoprettelsesomkostninger, der er ens for begge lån.

Læg mærke til, at der er en forskel på 497.459 kr. mellem det bedste og det værste scenario for den risikoneutrale strategi, mens standardafvigelsen fra gennemsnittet er på 92.290 kr. Med mini–max strategien fås en betydelig mindre standardafvigelse på 19.729 kr. Faktisk i hele 98% af scenarierne er den totale betaling præcis 1.355.170 kr.

Når vi, for den risikoneutrale løsning, beregner procentdelen af scenarier med en betaling over 1.355.170 kr., som er den mest sandsynlige minimax–betaling, finder vi, at i cirka 22% af tilfældene vil de totale omkostninger stige denne betaling. En ekstrem risikoavers låntager tror på, at netop et af disse scenarier vil forekome, og derfor vælger han minimax–strategien. I praksis ser vi, at mange låntagere vælger lån, der minder om minimax anbefalingen.

Med risikoneutral– og minimax–modellen har vi nu to yderligheder inden for låneanbefaling. Vi kan bruge resultater fra disse to modeller som et benchmark til kvantitativt at vurdere placering af en given lånestrategi.

Vi vil i det følgende se på en model med nyttefunktionsformulering og budgetbegrænsninger og en model, der tillige tager hensyn til formuerisikoen.

6.3.2 Nyttefunktionsmodellen med budgetbegrænsninger

Vi er interesserede i løsninger, som beholder et for låntageren acceptabelt niveau af robusthed (lille standard afvigelse) samtidig med, at låneomkostninger bliver reduceret i.f.t. minimax løsningen.

I dette afsnit ser vi løsningen af en model med en lineær nyttefunktion og budgetbegrænsninger. Vi beskrev brugen af budgetbegrænsninger og en straffunktion til håndtering af likviditetsrisiko i afsnit 5.2.3 (side 94). Budgetoverskridelser er tilladt i modellen, dog tilføjer vi en procentdel af budgetoverskridelsen som straf i målfunktionen.

Den afgørende faktor for opnåelse af løsninger, der både risikomæssigt og prismæssigt ligger mellem minimax–løsninger og risikoneutrale løsninger, er et overvejet valg af konstanter til acceptable budgetniveau, maksimale budgetoverskridelser og dertilhørende straf.

Vi beregner lånets løbende omkostninger og indfrielsesbeløbet separat og bruger de gennemsnitlige værdier fra disse to omkostninger som acceptable niveauer af løbende omkostning, h.h.v. indfrielsesbeløb. Fra den risikoneutrale løsning har vi en gennemsnitlig løbende omkostning på 570.842 kr. og en gennemsnitlig indfrielsesbetaling på 711.015 kr. Hvis vi kræver, at disse budgetniveauer ikke må overskrides, får vi ingen mulige løsninger. Vi tillader derfor en overskridelse på 50.000 kr. for de løbende omkostninger og 100.000 kr. for indfrielsesbetaling, dog med en 10% ekstrabetaling til målfunktionen som en straf, når modellen benytter sig af det ekstraordinære budget.

Værdien af målfunktionen er på **8998.0621** kr. Denne værdi svarer til en gennemsnitlige besparelse i forhold til budgettet fratrukket straffen for overskridelse af budgettet. I det følgende ser vi Cplex tidsforbrug samt den bedst fundne målfunktionsværdi og den absolutte og relative afstand fra en overgrænse.

```
RESOURCE USAGE, LIMIT     3879.370      3600.000
ITERATION COUNT, LIMIT    132039     999999999
Resource limit exceeded.

**** OBJECTIVE VALUE            8998.0621
Absolute gap:         9177.892995
Relative gap:            1.019986
```

Læg mærke til, at efter en times kørsel har vi en relativ afstand på over 100% fra en overgrænse i "branch and cut" træet. Den absolutte afstand er dog på lidt over 9000 kr., hvorfor vi kan godtage løsningen som en acceptabel løsning.

Vi beregner værdien af den gennemsnitlige totale omkostning[2] på **1.288.405,13** kr. og en standardafvigelse på **66.019** kr. Vi kan nu opdatere tabel 6.3 (side 110):

Læg mærke til, at formuleringen med nyttefunktion og budgetbegrænsninger har sikret os en overgrænse, der svarer til minimax–strategien samtidig med, at de gennemsnitlige omkostninger kun er steget med cirka 7000 kr. i.f.t. den

[2]De totale omkostninger er beregnet uden hensyntagen til engangsoprettelsesomkostninger, der er stort set ens for samtlige lånestrategier.

Lånestrategi	Totale omk.	Std. Afv.	max	min
Risikoneutral	1.281.857	92.289	1.502.042	1.004.583
Minimax	1.353.713	19.729	1.374.183	1.117.084
Nytte	1.288.405	66.019	1.431.857	1.005.412

Tabel 6.4: En sammenligning af 3 anbefalingsstrategier. De første 2 modeller repræsenterer 2 yderligheder, hvor den 3. er en afvejning af likviditetsrisiko mod omkostning.

risikoneutrale løsning. Vi har således fundet en løsning, der er mere robust end risikoneutralløsningen og meget billigere end minimaxløsningen.

Som den nærved optimale konverteringsplan får vi følgende løsning samt restgæld for de første 3 beslutningsperioder:

```
---- 226666 VARIABLE Sale.L  Salgsvariabel

               Sc1         Sc3         Sc4         Sc7

laan1                           879678.866
laan2  1018018.935
laan3              1039817.784
laan25                                      952381.359

---- 226666 VARIABLE RG.L  Restgaeld

               Sc1         Sc2         Sc3         Sc4

laan1                                       879678.866
laan2  1018018.935 1002696.289
laan3                          1039817.784

+           Sc5         Sc6         Sc7

laan2    986607.511
laan3               1020186.093   52515.539
laan25                            952381.359
```

En interessant detalje ved denne løsning er, at vi for første gang har fået en løsning, hvor vi beholder 2 obligationer i den samme periode. Dette sker ved beslutningsknuden 7 (Sc7), hvor en del af et 4% obligationslån (laan3) er bibeholdt, mens den største del af lånet er omlagt til et F1 lån. Den slags blanding af obligationer forekommer jævnligt i resten af løsningen. Dette er tegn på, at modellen vælger at sammensætte lån for at mindske likviditetsrisiko.

6.3.3 Formuerisikoaversitet

Vi tilføjer nu modellen formuerisikobegrænsninger samt bidrag til målfunktionen som beskrevet i afsnit 5.2.4 (side 95). Dette komplicerer problemet i en grad, så vores 3 kørsler af Cplex, med henholdsvis 1, 5 og 10 timer, ikke udløser et heltalsresultat. Vi vil derfor nøjes med at betragte formuerisikoen alene for et reduceret problem.

6.4 Risikoaverse modeller med GAMS/SCENRED formuleringer

Da vi løste minimax–problemet for samtlige scenarier, skulle vi vente over 10 timer, før Cplex kunne finde den første heltalsløsning. Ligeledes brugte Cplex over 1 time på at finde løsninger til problemet med nyttefunktionsformulering for at finde en løsning, der er over 100% fra en overgrænse til problemet. Endelig opgav vi at finde en løsning på problemet med formuerisikobetragtning. Disse lange løsningstider begrænser anvendeligheden af modellen, når modellen skal bruges til meget individuelle låneanbefalinger. Med disse lange køretider er vi således nødt til at køre programmerne for adskillige inputdata og gemme de fundne løsninger i en database. Dernæst kan brugerne hurtigt få adgang til den løsning, der er tættest på netop deres ønsker.

Det er ønskeligt at reducere køretiden, så modellen kan bruges mere dynamisk. Vi har i denne rapport set anvendelsen af scenarioreducering som en løsning på dette problem. Vi så tidligere i dette kapitel, at selv med en reducering fra 1024 scenarier til 12 scenarier får vi samme startløsninger. I det følgende løser vi de tre udvidede modeller ved brug af GAMS/SCENRED formuleringer og reducering til 12 scenarier. Alle betragtede omkostninger er fratrukket engangsomkostninger på oprettelsestidspunktet. Disse omkostninger er på cirka 19000 kr. for vores eksempel. Inden vi præsenterer de egentlige løsninger samler vi de opnåede resultater i den følgende tabel for at se, om vores overordnede mål er opnået.

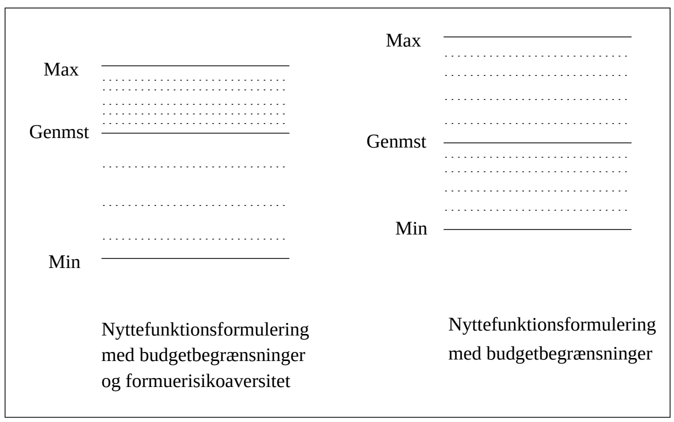
Lånestrategi	Totale omk.	Std. Afv.	max	min	tid
1 - Risikoneutral	1.169.173	49.765	1.274.079	1.064.525	1,7 s
2 - Minimax	1.187.938	0.00	1.187.938	1.187.938	42.2 s
3 - Budget beg.	1.171.926	24.270	1.229.897	1.136.655	300 s
4 - Likv./form. risikoavers	1.172.479	25.610	1.229.742	1.128.412	300 s

Tabel 6.5: Sammenligning af fire strategier. De første to strategier kan bruges som benchmark, hvor de sidste to kan justeres til at afspejle den enkelte låntagers ønsker.

De laveste gennemsnitlige omkostninger kommer fra den risikoneutrale model. Til gengæld har vi en standardafvigelse på cirka 50.000 kr. Minimax–modellen formår at give os en standardafvigelse på 0, mens omkostninger er steget med 18.765 kr. Denne omkostningsstigning virker for lille i forhold til den tilsvarende omkostningsstigning på 148.333 kr., som vi fik da alle scenarier var medtaget. Vi skal dog huske, at vi har en meget omfattende reducering fra 1024 scenarier til 12 scenarier samtidig med, at disse 12 scenarier underrepræsenterer de dyre scenarier. Vi kan fortolke minimax–løsningen for det reducerede scenariotræ som resultatet af en mere optimistisk “worst case” betragtning.

Den 3. formulering med nyttefunktion og budget–begrænsninger giver kun en lille stigning i.f.t. den risikoneutrale løsning, nemlig kun 2753 kr. til gengæld for en formindskelse af standardafvigelsen på cirka 25000 kr. Det er denne type løsninger, hvor en lille ekstra betaling medfører en forholdsvis stor robusthed, som vi er interesserede i.

I den sidste model, hvor der er blevet taget højde for formuerisikoen, ser vi en lille stigning på 553 kr. i.f.t. den 3. model. Samtidig er standardafvigelsen steget med 1340 kr. Som demonstreret i figur 6.4 (side 114) har



Figur 6.4: Vi ser i skitsen, at for modellen med indbygget formuerisikoaversitet fås en større koncentration af scenarier over gennemsnittet, hvor for modellen uden formuerisikoaversitet er fordelingen mere jævn.

både det dyreste og det billigste scenario for model 4 en mindre værdi i.f.t. den 3. model, mens den gennemsnitlige betaling er større for modellen med formuerisikoaversitet.

Hver streg i skitsen står for betalingsniveauet for et muligt scenario. Vi ser en større koncentration af dyre scenarier over gennemsnittet for modellen med indbygget formuerisikoaversitet end for modellen uden. Vi ved, at obligationer med kurs tæt ved pari har mindre formuerisiko end obligationer med kurs længere væk fra pari. Vi ved også, at obligationer med kurs tæt ved pari har højere kuponrenter og derved typisk er dyrere i løbende betalinger. Modellen har således valgt flere obligationer med kurs tættere på pari, end tilfældet er med modellen uden hensyntagen til formuerisiko. En kontrol af de valgte obligationer bekræfter dette.

Med hensyn til køretider ser vi, at alle modeller kan løses inden for en rimelig tid. Cplex kan dog ikke finde en løsning, der er 1% fra en overgrænse til model 3 og 4. I begge tilfælde stopper Cplex efter de 5 minutter, som vi har sat den maksimale køretid til.

De fundne løsninger for model 3 og 4 er 8% h.h.v. 5% fra overgrænser. Dette er ikke ensbetydende med, at vi ikke har fundet de optimale løsninger men med, at vi ikke har ventet i tilstrækkelig lang tid for at få beviset på optimalitet i "branch and cut"–træet. I begge tilfælde finder Cplex de bedst fundne løsninger i de første 100 sekunder, og resten af tiden bliver brugt til at finde bedre overgrænser således, at den fundne løsning får en reduceret afstand til overgrænsen. Udover problemernes kompleksitet kan denne lange tid m.h.t. opnåelse af meget små relative afstande skyldes, at vi får relativ små målfunktionsværdier for begge problemer. Vi skal huske på, at vi maksimerer en nyttefunktion i disse to modeller. Målfunktionsværdien afspejler en besparelse, som på grund af det meget stramme budget får en relativ lille numerisk værdi. Dette gør, at den absolutte afstand fra overgrænsen til målfunktionsværdien skal blive meget lille, før en relativ afstand på 1% kan opnås. Under alle omstændigheder er der højst tale om en yderligere besparelse, der er under 1000 kr. for et lån med en kursværdi på 1.000.000 kr.

Vi vil i de følgende afsnit præsenterer de fundne løsninger, der forårsager resultaterne fra tabel 6.5.

6.4.1 Minimax modellen

For overskuelighedens skyld betragter vi først løsningen fra den tilsvarende risikoneutrale model for de første tre år:

```
----  17727 VARIABLE Sale.L  Sale variable
                     Sc1         Sc3         Sc4         Sc5         Sc6
laan1 .time2                          1009169.676
laan3 .time1               1010771.540
laan3 .time2                                      1049185.216
laan25.time0  1000000.000
laan25.time2                                                   900353.753
```

Karakteristisk for risikoneutrale løsninger er, at der oprettes lån i en obligation ad gangen. I det følgende ser vi dog, at minimax–løsningen er udformet af flere obligationer i flere beslutningsknuder:

```
----  17783 VARIABLE Sale.L  Sale variable

                 Sc1         Sc2         Sc3         Sc4         Sc5          Sc6        Sc7

laan2 .time1             263565.223
laan3 .time0 143751.003
laan3 .time1                         877357.610
laan3 .time2                                                 923806.957
laan6 .time2                                     340207.804
laan25.time0 868007.829
laan25.time2                                                            907278.49  395452.88
```

```
----  17783 VARIABLE RG.L  Outstanding debt

                Sc1          Sc2          Sc3         Sc4        Sc5          Sc6          Sc7

laan2 .time1              263565.223
laan3 .time0 143751.003
laan3 .time1              141187.909 1018545.518
laan3 .time2                                                 1062329.247              597514.102
laan6 .time2                                     340207.804
laan25.time0 868007.829
laan25.time1              601447.416
laan25.time2                                     587790.645             907278.493  395452.883
```

Tabellen med restgæld ved hver knude er medtaget for at give overblik over låneporteføljens bevægelse. Ved beslutningsknude 2 ser vi for eksempel, at en del af F1 lånet (laan25) er blevet indfriet ved at udstede en 5% fastforrentet obligation (laan2). Hvis renterne stiger, d.v.s. hvis vi kommer til knude 4, opkonverterer vi de to fastforrentede obligationer til et 7% lån (laan6), og hvis renten falder, indfrier vi 5% obligationen ved at udstede flere 4% obligationer.

I modsætning til minimax–løsningen for det originale træ (1024 scenarier) ser vi en meget stor bevægelighed i løsningen. Dette kan begrundes med, at den fundne løsning (efter 100.000 sekunder) er 8,6% fra den bedst fundne nedre grænse. I absolut afstand svarer det til 123.607 kr. Givet at Cplex fik endnu mere tid, kunne vi forvente at få tilsvarende løsninger med lige så stor bevægelighed som i det reducerede problem. Vi forventer dog ikke, at sådanne løsninger medfører en så stor besparelse, som det fremgår fra afstanden til den nedre grænse. Hermed kan vi konkludere, at simple lånestrategier med få sikre obligationer i spil giver nær–optimale løsninger for minimax problemet. Den store bevægelighed i løsningen på det reducerede problem indikerer dog, at der er mulighed for at spare penge, hvis vi havde tid til at vente på en bedre løsning til det ikke reducerede problem.

6.4.2 Nyttefunktionsmodellen med budgetbegrænsninger

Vi har valgt følgende budgetbegrænsninger i programmet:

```
SCALAR BMAX    /565915/;    // den acceptable loebende bet. niveaue
SCALAR IBMAX   /601983/;    // den acceptabel indfrielsesbeloeb
SCALAR BOMAX    /50000/;    // max budgetoverskridelse pr sti
SCALAR IBOMAX   /35000/;    // max indfrielsesoverskridelse
```

Disse værdier er baseret på gennemsnits– og standardafvigelsesbetragtninger fra den risikoneutrale model.
Følgende målfunktionsværdi opnås efter 300 sekunder:

```
**** OBJECTIVE VALUE           -13118,2097

Absolute gap:         1114,52
Relative gap:            0,085
```

Den negative værdi af målfunktionen svarer til et gennemsnitligt underskud. En yderligere formindskelse af den relative afstand til en overgrænse (8,5% efter 300 sekunder) kan ikke forventes at have en afgørende betydning for målfunktionsværdien. Hvis vi forøger køretiden fra 300 sekunder til cirka 1200 sekunder, får vi en forbedring af målfunktionen på cirka 250 kr., mens den relative afstand falder til under 1%.

For at få denne gennemsnitlige målfunktionsværdi skal låntageren følge denne løsning:

```
----  17771 VARIABLE Sale.L  Salgsvariabel

                      Sc1         Sc3         Sc4         Sc5         Sc6

laan1 .time2                           379714.610
laan3 .time1              1010771.540
laan3 .time2                                      1049185.216
laan25.time0  1000000.000
laan25.time2                                                   900353.753



----  17771 VARIABLE RG.L  Restgaeld

                 Sc1       Sc2       Sc3       Sc4         Sc5         Sc6         Sc7

laan1 .time2                                 379714.610
laan3 .time1                     1010771.54
laan3 .time2                                            1049185.216              991688.24
laan25.time0  1000000
laan25.time1          974686.996
laan25.time2                                 594142.637              900353.753
```

Vi ser en mere varieret løsning i.f.t. løsningen på den risikoneutrale model. Hvis vi for eksempel betragter beslutningsknude 4 (Sc4), ser vi, at kun en del af F1–lånet (laan25) bliver lagt om til et 6% obligationslån (laan1). Udover blandede låneporteføljer sker der ned– og opkonverteringer i hele 76 beslutningsknuder ud af 79 mulige. I modsætning hertil havde vi 52 konverteringer for den risikoneutrale model. Ekstra konverteringer forøger den totale omkostning, men til gengæld sikrer de låntageren mod de dyreste scenarier. En optimal (billig) risikoavers strategi kan derfor betegnes som en strategi med en relativ stor antal konverteringer samt blandede låneporteføljer.

6.4.3 Formuerisikoaversitet

Vi straffer nu overskridelserne i.f.t. den gennemsnitlige restgæld ved at trække 10% af de overskridende beløb fra nyttefunktionen.
Cplex returnerer følgende målfunktionsværdi efter 300 sekunder:

```
**** OBJECTIVE VALUE           -25585.7769

Absolute gap:         1285.943865
Relative gap:            0.050260
```

Igen har vi en negativ målfunktionsværdi svarende til et budgetunderskud.

Følgende løsning sikrer os den fundne målfunktionsværdi:

```
----  17769 VARIABLE Sale.L  Salgsvariabel

                  Sc1      Sc3         Sc4         Sc5        Sc6

laan1 .time2                         377523.594
laan3 .time1           1010771.540
laan3 .time2                                    1049185.216
laan25.time0  1000000
laan25.time2                                                900353.753

----  17769 VARIABLE RG.L  Restgaeld

                Sc1      Sc2       Sc3        Sc4        Sc5        Sc6       Sc7

laan1 .time2                               377523.594
laan3 .time1                    1010771.54
laan3 .time2                                          1049185.216          991688.24
laan25.time0 1000000
laan25.time1        974686.996
laan25.time2                               596210.737             900353.753
```

Den eneste forskel vi observerer her i forhold til konverteringsplanen for den tilsvarende periode i den sidste model, kan ses ved beslutningsknude 4. Her beholder vi 2200 kr. ekstra af F1 lånet (laan25), svarende til, at vi udsteder 2200 kr. mindre i en 6% obligationslån (laan1). Dette er rimeligt i formuerisikobetragtninger, eftersom F1 lånet har en kurs, der er tættere på pari, end 6% obligationslånet har. Eftersom de fleste valgte lån i forvejen har kurs tæt på pari, er disse nye ændringer som følge af formuerisikobetragtningerne ganske små.

6.5 Afsluttende kommentarer

Samtlige af de opnåede resultater i dette kapitel viste sig at være i overensstemmelse med de finansielle retningslinjer til anbefaling af lån. Svend

Bondorf, kontorchef i kontor for finansielle analyser hos Nykredit, har kontrolleret de opnåede resultater. Og i den forbindelse har han verificeret, at resultaterne stemmer overens med de gængse finansielle argumenter i virkeligheden. Vi kan således konstatere modellens brugbarhed i praksis.

Grundproblemet kan løses både med det ikke–reducerede træ og med det reducerede træ inden for 300 CPU sekunder. Det gør det muligt at bruge modellen som et risikovurderingsværktøj for produktet rentesikring. Dette er nødvendigt, eftersom prisfastsættelsen af rentesikring, som vi så i afsnit 3.4 (side 31), er baseret på det ikke–reducerede scenariotræ.

Når modellen bruges til rådgivnings– og produktudviklingsformål kan vi nøjes med en reduceret version af problemet. Dette viste sig faktisk at være nødvendigt, eftersom de risikoaverse versioner af modellen er ekstremt tidskrævende at løse.

Selvom det reducerede scenariotræ viser sig at afspejle en mere optimistisk forudsigelse af fremtidens renter, kan det bruges til at generere værdifulde løsninger, der giver analytikeren information om nogle interessante lånestrategier til videre analyse. De reducerede træer kan nemt visualiseres, (se koden i appendix F (side 173)), så vi kan danne os et indtryk af det reducerede scenariotræs skævhed i.f.t. det ikke–reducerede træ. Vi kan dernæst tilføje flere forskellige scenarier til det reducerede træ for at teste robustheden i de fundne løsninger.

Flere generelle konklusioner om de opnåede resultater under udarbejdeslen af denne rapport kommer i næste kapitel.

Kapitel 7

Konklusion

En nuanceret investor er interesseret i en investeringsportefølje, der er afbalanceret m.h.t. både risiko og afkast. Hvis investoren for eksempel udelukkende skal investere sine penge i obligationer, vil han typisk sammensætte en blanding af korte, mellemlange og lange obligationer på en sådan måde, at det højest mulige afkast sikres, dog uden at risikoen for tab overstiger et på forhånd fastlagt niveau. For langvarige investeringer skal porteføljen desuden løbende justeres i takt med den ekstra information, der bliver til rådighed.

En nuanceret låntager skulle principielt gøre det samme som en nuanceret investor. Låneporteføljen skal bestå af både korte, mellemlange og lange obligationer, og porteføljen skal løbende justeres. De fleste låntagere vælger dog at oprette lån alene i én obligation, nemlig enten i en fastforrentet obligation eller i en kort 1–årig F1 obligation. De fleste låntagere med fastforrentede obligationslån vælger kun at justere deres låneporteføljer, når renterne er faldet markant (nedkonvertering). Det sker i endnu sjældnere grad, at de opkonverterer et lån for at reducere deres restgæld.

Den vigtigste forskel mellem investor– og låntagerside af obligationsmarkedet er de ekstra store faste og variable transaktionsomkostninger, som låntageren løbende skal betale, når han ønsker at justere sin låneportefølje. Desuden har låntageren flere begrænsninger m.h.t. sammensætning af sin portefølje, end investoren har med sin.

Vi har i denne rapport set, at det på trods af denne asymmetri kan betale sig at benytte sig af obligationssammensætnings– og konverteringsmuligheder i langt højere grad, end det er tilfældet i dag.

Når låntagere ikke benytter sig af deres muligheder (som de har betalt for) for at opnå den billigst mulige låneportefølje under hensyntagen til deres risikoopfattelse, skyldes det blandt andet følgende to faktorer:

1. Kompleksiteten på det danske realkreditmarked gør det uoverskueligt for de fleste private låntagere at danne sig et overblik over samtlige muligheder og konsekvenser af forskellige beslutninger.
2. Realkreditinstitutterne giver alene generelle anbefalinger. Selv i de mere individuelle rådgivninger giver rådgiveren kun de generelle retningslinjer til låntageren. Dette er ofte ikke tilstrækkeligt til, at låntageren selv kan indse de bedste muligheder for eventuelle besparelser.

Vi har i denne rapport anvendt stokastiske optimeringsmodeller til at generere optimale låneanbefalinger for en låntager, givet en bestemt renteudvikling og låntagerens specifikke input, deriblandt låntagerens risikoopfattelse.

Selvom ideen med brugen af stokastisk optimering er velkendt, især blandt investorer med store investeringer, findes der endnu ikke et funktionelt rådgivningssystem baseret på en optimeringsmodel i almindelighed og en multi–stadie heltals stokastisk optimeringsmodel i særdeleshed, hos realkreditinstitutter.

Vi har i denne rapport set, hvordan en optimeringsmodel kan bruges til fordel for både realkreditinstitutter og realkreditkunder. I de følgende afsnit ser vi på nogle væsentlige konklusioner og forslag til videre arbejde med dette emne:

7.1 Rådgivning

Optimeringsmodellen giver mulighed for en mere dynamisk rådgivning end det er tilfældet i dag. Med dynamisk menes det, at låntageren/RI på et højt niveau af interaktion kan foretage "what if"–analyser.

Rådgivning er i dag baseret på enkelte konsekvensberegninger på enkelte låneporteføljer for flere rentescenarier. I modsætning hertil har vi i denne rapport set, hvordan en optimeringsmodel kan håndtere samtlige blandingsmuligheder af låneporteføljer, samtlige konverteringsmuligheder og samtlige rentescenarier på en gang.

Låntageren skal således alene angive data om løbetid, budgetbegrænsninger og eventuelle planer om førtidig indfrielse af lånet, og modellen vil returnere den billigste lånestrategi, som tilfredsstiller låntageres samtlige krav, givet, at der findes en mulig løsning.

7.2 Produktudvikling

Modellen kan bruges som et underliggende analyseværktøj til en ny type af lån. Sådanne lån kan for eksempel gå under betegnelsen "dynamiske lån"

(D–lån). Lånet vil oprettes i en eller flere obligationer, hvilke løbende omlægges til nye obligationer i overensstemmelse med en optimal lånestrategi, givet kundens oprindelige præferencer med mulighed for løbende justering af præferencerne.

Dette produkt kan sammenlignes med lignende produkter på investorside af det finansielle marked, hvor en investor kan investere sine penge i en investeringspulje, der løbende justeres af investeringsforeningen.

Der findes i dag blandingsprodukter såsom P–lån, hvor låneporteføljen sammensættes af på forhånd fastlagte procentdele af korte og lange obligationer. Et "dynamisk lån" kan erstatte P–lån med henvisning til det mere fleksible og fordelagtige blandingsmønster, som optimeringsmodellen giver os mulighed for at opnå.

7.3 Analyse af rentesikring

Vi har i denne rapport beskrevet og implementeret Black, Derman og Toys en–faktor rentestrukturmodel. Derudover har vi beskrevet to måder at prisfastsætte en vilkårlig rentesikring på et F1–lån. Den mest effektive prisfastsættelsesalgoritme har vi implementeret i VBA.

Eftersom restgældsudviklingen i et rentetilpasningslån er stiafhængig i det binomiale træ, ser det umiddelbart ud til, at man skal bruge et ikke–rekombinerende træ til prisfastsættelsen. Dette er imidlertid ikke tilfældet, idet man, jævnfør metoden leveling ([16]), kan normere restgælden i alle knuder og derved beregne værdien af rentesikring ved hjælp af et rekombinerende binomialtræ. Her skal man være opmærksom på, at denne normering kun er mulig i kraft af, at der eksisterer et lineært forhold mellem restgæld og ydelse i annuitetsformlen.

Det skal bemærkes, at den fundne pris bygger på en gennemsnitlig sammenligning af priser på et F1–lån henholdsvis med og uden et renteloft. Der bliver ikke taget højde for, at låntageren har mulighed for at benytte sig af andre obligationer end netop F1–obligationerne. Dette er rimeligt set med sælgerens (RI) øjne, eftersom RI betragter produktet rentesikring som et isoleret produkt. Dette skyldes, at en låntager med rentesikret F1–lån ikke behøver at benytte sig af andre lånetilbud, der sikrer ham mod eventuelle rentestigninger. Set med låntagerens øjne vil rimeligheden i prisen på rentesikring variere alt efter hvilken risikoopfattelse han har, og hvor aktivt han vil deltage i vedligeholdelse af sit lån for at sikre de laveste omkostninger, der passer til lige netop hans risikoprofil.

Vi kan bruge optimeringsmodellen til, set med låntagerens øjne, at analysere rimeligheden i prisen på en rentesikring med. I virkeligheden er det dog RI, der foretager analyserne for at undersøge hvilken gruppe af låntagere der burde betragte RIs pris på rentesikring som acceptabel. Der er flere måder at foretage sådanne analyser på:

1. Som input til en risikoneutral version af optimeringsmodellen vælger vi et univers af obligationer bestående af korte, mellemlange og lange obligationer. Først løser vi problemet uden hensyntagen til rentesikring. Dernæst tilføjer vi en F1–obligation med et renteloft, svarende til sikringsniveauet, og løser problemet igen. Faldet i den totale omkostning som følge af tilføjelsen af rentesikringen giver os en nedre grænse for prisen på rentesikring. At denne er en nedre grænse kan begrundes med, at de fleste låntagere ikke har en risikoneutral holdning til deres lån.

2. Som input til en minimax–version af optimeringsmodellen vælger vi et univers af obligationer bestående af korte, mellemlange og lange obligationer, samt en F1–obligation med rentesikring, hvor startprisen på rentesikringen kan initialiseres som en nedre grænse fundet under punkt 1. Når vi løser problemet første gang, vælges F1–obligationen med rentesikring som en del af løsningen, eftersom den meget risikoaverse model foretrækker sikre løsninger med billige priser. Vi løser problemet adskillige gange hver gang med en lidt forhøjet pris på rentesikringen. Når modellen lige netop ikke vælger rentesikringen, har vi fundet en overgrænse på prisen af rentesikring. At denne er en overgrænse kan begrundes med, at mange låntagere ikke har en meget risikoavers holdnig til deres lån.

Når vi bruger optimeringsmodellen som analyseværktøj til prisfastsættelsen af rentesikring skal vi bruge den fulde information i træet, hvis resultaterne skal være sammenlignelige med den teoretiske pris fundet i afsnit 3.4 (side 31). Når vi derimod bruger optimeringsmodellen til rådgivning og produktudvikling, kan vi nøjes med at bruge reducerede scenariotræer.

7.4 Scenarioreducering

Konceptet scenarioreducering blev analyseret og afprøvet under udarbejdelsen af dette projekt. For den risikoneutrale model så vi, at selv med en reducering af antallet af scenarier fra 1024 til 12 får vi de samme startløsninger. Hvis vi kender den optimale startløsning i et stokastisk program med "recourse", kan vi droppe det første stadie og løse resten af problemet uafhængig af det første stadie. Vi vil opnå det samme endelige resultat, uanset om vi bruger det ikke–reducerede eller det reducerede scenariotræ, bare vi får de samme startløsninger i hvert delproblem.

Det viste sig at være nødvendigt med en kraftig reducering af scenarierne for overhovedet at kunne løse modellerne med indbygget risikoaversitet med det ønskelige præcisionsniveau indenfor rimelig tid (under 300 sekunder i vores tilfælde).

Her er det vigtigt at pointere, at når optimeringsmodellen bliver brugt til rådgivning eller produktudvikling er det mindre afgørende, om vi bruger det ikke–reducerede scenariotræ eller et reduceret træ. Det er dog vigtigt at reducere træet, således at de dyre og de billige scenarier i det reducerede træ har den samme fordelingsform som i det ikke–reducerede træ.

Selvom vi konstaterede en overpræsentation af de billige løsninger ved brug af GAMS/SCENRED, opnåede vi løsninger i fuld overensstemmelse med markedets retningslinjer for valg af lån for forskellige risikoopfattelser.

Vi kan betragte det fundne reducerede scenariotræ som et mere optimistisk forudsigelsesgrundlag end det ikke–reducerede scenariotræ. Under analysefasen af produktudviklingen kan vi tilføje nogle pessimistiske scenarier til træet for at teste de opnåede løsningers robusthed.

7.5 Forslag til videre arbejde

Den største udfordring i løsning af de præsenterede stokastiske modeller i denne rapport består af løsningen af det deterministisk ækvivalente heltalsproblem.

Arbejdet i dette projekt har hovedsageligt været fokuseret på modeludvikling og modelvalidering. Vi har dog set på anvendelsen af scenarioreducering som en måde at muliggøre relativt hurtig opnåelse af nær–optimale løsninger på.

I det følgende kan vi se 3 konkrete forslag til videre arbejde med låneanbefalingsproblemet.

7.5.1 LP relaksation

En løsning på køretidsproblemet består af en omformulering af problemet fra et heltalsproblem (MIP) til et lineært programmeringsproblem (LP). Udfordringen i denne omformulering ligger i udvikling af et regelsæt, der omsætter de faste løbende transaktionsomkostninger til variable løbende transaktionsomkostninger, således at det relakserede problem er så tæt på det oprindelige problem som muligt.

7.5.2 Heuristiske løsninger

Der findes få generelle heuristiske løsninger til multi–stadie stokastiske heltalsproblemer og ingen, der er skræddersyet til låneanbefalingsproblemet. Det er ikke svært at opnå gode løsninger til låneanbefalingsproblemet, selvom det kan være ekstremt tidskrævende at opnå en optimal løsning. Som et forslag til en heuristisk løsning kan man udvikle en algoritme, der følger et sæt af finansielle retningslinjer for at opnå gode resultater. Sådan et regelsæt kan sættes i forskellige heuristiske rammer, så de opnåede resultater kan sammenlignes.

7.5.3 Parallel programmering

Et mere krævende alternativ til løsning af låneanbefalingsproblemet er brugen af parallel programmering. Modellen kan formuleres med split–variable og ikke–forudseenhedsbegrænsninger. Disse ikke–forudseenhedsbegrænsninger kan droppes midlertidigt så delproblemerne kan løses parallelt. En stor udfordring ligger dernæst i en effektiv generering af snitflader, der udelukker de løsninger, der ikke respekterer ikke–forudseenhedsbegrænsningerne.

7.6 Slutord

Matematisk finansiering og matematisk optimering er to store discipliner indenfor både teoretisk og anvendt matematik. Der er et stort potentiale i brugen af en kombination af disse to discipliner i det virkelige liv. I flere lande er den finansielle sektor allerede godt igang med at anvende denne kombinationsmulighed. I Danmark har muligheden for brugen af matematisk optimering (operationsanalyse) i diverse beslutningsprocesser endnu ikke vundet indpas.

Vi har i denne rapport set et eksempel på et område, hvor brugen af en blanding af matematisk finansiering og operationsanalyse er oplagt. De fundne resultater er lovende for praktisk anvendelighed af modellen, og den udviklede software kan nemt integreres i eksisterende systemer hos de fleste finanshuse.

Mit håb er, at denne rapport har givet læseren et nyt analytisk værktøj til hurtigere at kunne levere individuelle og udførlige låneanbefalinger. Sådanne anbefalinger kan, givet vores rente– og låntager–forudsætninger, tilmed betragtes som optimale.

Litteratur

[1] Bjarne Astrup Jensen
Rentes regning
Jurist– og Økonomforbundets Forlag, 3. udgave, 2001.

[2] John R. Birge
Decomposition and partitioning methods for multistage stochastic linear programs
Operations Research, 33(5): 989-1007, 1985.

[3] John R. Birge & Francois Louveaux
Introduction to Stochastic Programming
1997 Springer–Verlag New York, Inc.

[4] Petter Bjerksund and Gunnar Stensland
Implementation of the Black–Derman–Toy Interest Rate Model
The Journal of Fixed Income, Volume 6 Number 2, September 1996.

[5] Tomas Björk
Arbitrage Theory in Continuous Time
ISBN 0-19-877518-0, Oxford University Press.

[6] Fischer Black, Emanuel Derman and William Toy
A One–Factor Model of Interest Rates and its Application to Treasury Bond Options
Financial Ananlysts Journal/January–February 1990.

[7] Svend Bondorf
Avanceret realkredit–teorien bag Tilpasningslån
Master's Thesis No. 1998-5, IM, DTU, februar 1998.

[8] Svend Bondorf, Dan Sørensen, Michael Carlsen
RENTESIKRING - nyt instrument til afdækning af likviditetsrisiko på rentetilpasningslån
FINANS/INVEST 3/00.

[9] Michael Bussieck, Armin pruessner
Mixed–Integer Nonlinear Programming
Overview for GAMS Development Corporation, February 19, 2003.

[10] Michael Christensen
Obligations investering
5. udgave, Jurist– og Økonomforbundets Forlag, 2001.

[11] J. Dupačová, N. Gröwe–Kuska, W. Römisch
Scenario reduction in stochastic programming: An appraoch using probability metrics
Revised version to appear in Mathematical Programming.

[12] *GAM/SCENRED Documentation*
Available from <www.gams.com/docs/document.htm>.

[13] H. Heitsch, W. Römisch
Scenario reduction algorithms in stochastic programming
Preprint 01-8, Institut für Matematik, Humboldt–Universität zu Berlin, 2001.

[14] John C. Hull
Options, Futures and Other Derivatives
Fifth edition,Prentice hall international 2003.

[15] Hull & White 93
Efficient procedures for valuing European and American path-dependent options
Hull, J. & White, A., Journal of Derivatives(fall 1993), 1993.

[16] Luenberger 98
Investment Science
Luenberger, D.G., Oxford Uni. Press, 1998.

[17] *Nykredit hjemmeside*
<www.nykredit.dk>.

[18] Søren S. Nielsen & Rolf Poulsen
A Two–Factor, Stochastic Programming Model of Danish Mortgage–Backed Securities
Journal of Economic Dynamics and Control, June 11, 2002.

[19] Søren S. Nielsen & Stavros A. Zenios
Solving Multistage Stochastic Network Programs on Massively Parallel Computers
Mathematical Programming 73, (1996), 227-250.

[20] R. T. Rockafellar and R.J.-B. Wets.
Scenarios and policy aggregation in optimization under uncertainty.
Math. of Operations Research, 16(1):119-147, February 1991.

[21] A. Ruszczynski.
Parallel decomposition of multistage stochastic programming problems.
Working paper WP-88-094,IIASA, October. 1988.

[22] L.F. Shampine, R.C. Allen. Jr. og S. Pruess
Fundamentals of Numerical Computing.
John Wiley &Sons, Inc. 1997.

[23] Dan E. K. Sørensen
Rentetilpasningslån, Flexlån, Tilpasningslån.

[24] Laurence A. Wolsey
Integer programming.
Wiley–Interscience Series in Discrete Mathematics and Optimization, 1998.

Indeks

afdrag, 12
Amerikansk option, 42
annuitet, 12
arbitrage, 42

betalingsstrøm, 35
bidrag, 12

Call option , 40
cap, 32

Europæisk option, 42
Exercise en option, 42

fast–forrentede lån, 1
formuerisiko, 46

Inkonverterbare obligationer, 40
invers, 47

kombinerende rentetræ, 32
Konverterbare obligationer, 40
kursværdi, 50

likviditetsrisiko, 46

ned–konvertering, 1
Nulkuponobligation, 3

op–konvertering, 1

putoptioner, 32

rentebetaling, 12
rentesikring, 1

tilbagediskonteringsfaktor, 46
tilpasningslån, 1

Bilag A

Ordbog

Afdrag og rentebetaling:
Den løbende ydelsesbetaling på et lån består af Afdrag og rentebetaling. Afdrag bliver trukket fra restgælden på et lån, hvorimod rentebetalingen bliver ikke trukket fra restgælden.

Amerikansk option:
En amerikansk option på en obligation er en option, som giver indehaveren ret til at indfri den underliggende obligation på et hvilket som helst tidspunkt inden udløbsdatoen. I praksis kan man dog kun indfri lånet til bestemte terminsdatoer, typisk 4 gange om året.

Amortiseringsplan:
Til hver termin betaler låntageren afdrag, rentebetaling og bidrag til realkreditinstituttet. En beskrivelse af disse betalinger gives i en amortiseringsplan.

Annuitet:
En annuitet defineres som en betalingsrække af lige store ydelser med lige store tidsmæssige mellemrum. Afdragene i et annuitetslån vokser over lånets løbetid, mens rentebetalinger falder.

Annuiteter er den dominerende type af betalingsrækker på markedet for realkreditobligationer. Alle betalingsrækker i denne rapport er annuiteter.

Bidrag:
Bidrag betales som en procentdel af den løbende restgæld til realkreditinstitutter. Bidragsbetalinger kompenserer for den risiko realkreditinstitutter løber ved at låne penge ud til låntagere, samt diverse administrative omkostninger.

Hvis lånets hovestol er under 200.000 DKK, så er der et fast bidrag og et variabelt bidrag forbundet med lånet. Dette er af hensyn til, at visse administrative opgaver er ens uanset størrelsen på lånet.

Formuerisiko:
Formuerisikoen er den risiko, låntageren har i forbindelse med en voldsom

kursstigning for de bagvedliggende obligationer. Hvis man eksempelvis optager et 30 årigt fastforrentede obligationslån hvor man udsteder obligationer til kurs 80, har man en stor formuerisiko. Et lille rentefald betyder en stor stigning i kursen og dermed en forøgelse af ens restgæld i tilfældet af for tidlig indfrielse af lånet, eller en tilsvarende formindskelse af ejendommens friværdi. Hvis man ikke skal indfri lånet utidigt eller optage nyt lån i ejendommens friværdi, behøver man således ikke at bekymre sig om formuerisikoen. Formuerisikoen er nemlig kun uønskelig, hvis den kan omsættes til likviditetsrisiko, og dette sker enten ved en tidlig indfrielse af lånet eller udnyttelsen af ejendommens friværdi.

Hovedstol:
Det beløb, som låntager forpligter sig til at forrente og afdrage, benævnes som hovedstolen. Hvis obligationskursen er under 100, har vi at hovedstolen er større end kursværdien og omvendt hvis obligationskursen er over 100, har vi at hovedstolen er mindre end kursværdien. Hovedstol er et andet navn for en restgæld på tidspunkt 0. For et obligationslån har vi: $kursvaerdi = obligationskurs \times Obligationshovedstol$. Obligationshovedstolen er typisk større end kursværdien p.g.a. kurstab, idet låntageren kun må udstede obligationer hvis kurs er mindre end 100. For et kontantlån sætter man hovedstolen og kursværdien lig med hinanden, til gengæld får lånet en højere rente.

Indfrielse (køb) af en obligation:
Når en låntager gerne vil komme ud af et lån, for eksempel når huset skal sælges eller når lånet skal konverteres (omlægges) til et andet lån, skal låntageren købe de underliggende obligationer tilbage. Prisen for tilbagekøb (indfrielse) af obligationer svarer til dagens kurs for inkonverterbare obligationer og til dagens kurs, dog højest til kurs pari (kurs 100), for konverterbare obligationer.

Invers rentestruktur
En invers rentestruktur er en rentestruktur, hvor de korte renter er større end de lange renter. Når vi har en invers rentestruktur, kan vi oprette fastforrentede lån, der er billigere end de korte rentetilpasningslån.

Kursværdi:
Det beløb, som låntager modtager ved oprettelsen af lån , benævnes som kursværdien. Hvis obligationskursen er under 100, har vi at hovedstolen er større end kursværdien og omvendt hvis obligationskursen er over 100, har vi at hovedstolen er mindre end kursværdien.

Likviditetsrisiko:
Likviditetsrisiko er risikoen forbundet med ekstra betalinger som følge af stigninger i renten. Et F1–lån må derfor antages at have en høj likviditetsrisko. Et stort fokus på likviditetsrisikoen vil betyde, at de fastforrentede lån bliver favoriseret i en optimalløsning.

Nulkuponobligation:
En nulkuponobligation er en obligation uden årlige renter. For eksempel kan en investor købe en nulkuponobliation for 94 DKK for at modtage 100 DKK om to år. Her taler man også ofte om den implicitte rente, som i dette eksempels tilfælde er $3,14\%$ per år: $94 \cdot (1 + 0,0314)^2 = 100$.

Obligation:
En obligation er et værdipapir, der kan købes af en investor for en markedsbestemt pris eller kurs. Investoren får til gengæld årlige renter, samt et engangsbeløb ved obligationens udløb. Dette engangsbeløb kaldes den pålydende værdi og er kendt på det tidspunkt investoren køber obligationen. Den obligationskurs for hvilken hovedstolen og kursværdien er ens kaldes ofte kurs pari eller kurs 100.

Tinglysning:
Alle danske ejendomme, d.v.s. jordarealer med eller uden bygninger, er registreret i tingbøger på såkaldte tinglysningskontorer rundt omkring i landet. Danmark er opdelt i 82 retskredse, hvor der er placeret et tinglysningskontor i hver kreds. Når man afgiver pant i sin faste ejendom, registreres det i tingbogen i prioritetsrækkefølge, d.v.s. i en rangorden, der i første omgang reguleres efter, hvem der tinglyser først, og derefter hvorledes de enkelte prioriteter respekterer hinanden indbyrdes. Systemet er reguleret af Tinglysningsloven, der omhandler og regulerer tinglysning af såvel gæld som andre rettigheder og forpligtelser.

Sælge en obligation:
Realkreditinstituttet udsteder (sælger) på vegne af en låntager en obligation. Låntageren modtager kursværdien for den solgte obligation, for til gengæld at betale afdrag, renter og bidrag til realkreditinstituttet efter en amortiseringsplan. Realkreditinstituttet afleverer renter og afdrag til investoren (køberen af obligationen) og selv beholder bidraget.

Bilag B

Fakta om realkreditlån

B.1 Prisblad realkreditlån i kr.

For at give et realistisk billede af diverse faste og variable omkostninger i forbindelse med et lån gennemgås de rigtige tal udgivet af Nykredit. De omkostninger der nævnes her har været gældende siden 1. oktober 2002.

B.1.1 Nykredit Kernekunde

Betingelserne for at blive Kernekunde i Nykredit koncernen er enkle. Har man lån i ejerbolig eller fritidshus, og har man betalt alle terminer til tiden gennem de seneste to år, kan man blive Kernekunde. Det gælder også, hvis man i øjeblikket har lån i et andet realkreditinstitut. Man kan også blive Kernekunde, hvis man bliver godkendt til 100% finansiering af sin ejerbolig eller sit fritidshus eller ved rettidig betaling af Bolig– eller Andelsboliglån i Nykredit Bank gennem de sidste to år. Dette gælder også, selvom man i dag har bolig– eller andelsboliglån i et andet pengeinstitut.

B.1.2 Opkrævning af gebyrer

Gebyrer modregnes ved låneudbetaling eller opkræves ved afslutning af sagsekspedition eventuelt i forbindelse med førstkommende terminsydelse.

Noter til tabel (B.1):
1) Ved a conto udbetaling af lån opkræves yderligere lånesagsgebyr
2) Der gælder særlige regler ved optagelse af nyt lån
3) I specielle sager kan der opkræves et særligt gebyr
4) Gebyret dækker:
- håndtering af tinglysning af nyt lån
- oprettelse og afregning af afregningskonto
- gebyr for Fastkursaftale
- én rykning af efterstående panthavere eksklusiv tinglysningsafgift 5)

Lånoptagelse		
	Kernekunde	Øvrige
Lånesagsgebyr ejerbolig og fritidshus 1)	1.050 kr.	2.100 kr.
Gældsovertagelse ved ejerskifte pr. ejendom 2)	1.050 kr.	1.050 kr.
Relaksation og opdeling af ældre lån pr. parcel 3)	1.050 kr.	1.050 kr.
Tinglysningsekspedition pr. ejendom 4)	2.200 kr.	2.600 kr.
Rykning af efterstående panthavere pr. lån/anmodning 5)	300 kr.	300 kr.
Indfrielse af andre lån end Nykredit-lån pr. lån 5)	300 kr.	300 kr.
Tinglysningsekspedition - omprioritering for sælger 6)	2.800 kr.	2.800 kr.

Tabel B.1: Omkostninger forbundet med låneoptagelsen.

- håndtering af indfrielse af maks tre andre lån end Nykredit lån 5)
5) Der kan forekomme omkostninger til kreditor
6) Gebyret dækker:
- håndtering af tinglysning af nyt ejerskiftelån
- oprettelse og afregning af afregningskonto/omprioriteringskonto
- gebyr for Fastkursaftale
- én rykning af efterstående panthavere eksklusiv tinglysningsafgift 5)
- håndtering af indfrielse af maks tre andre lån end Nykredit lån 5)

For hvert realkreditlån findes en række serviceydelser, og disse har omkostninger som fremgår af tabel (B.2) og (B.3).

Serviceydelser			
		Kernekunde	Øvrige
LånOvervågning	Aftale om LånOvervågning	0 kr.	0 kr.
Fastkursaftale mv.	Indgåelse eller ændring af Fastkursaftale (uden aftale om tinglysningsekspedition) 1)	500 kr.	500 kr.
Supplerende serviceydelser	Serviceydelser 2)	100 kr. pr. ydelse	100 kr. pr. ydelse

Tabel B.2: Liste over nogle af de standard serviceydelser i forbindelse med et realkreditlån hos Nykredit.

Noter til tabel (B.2):

1) Udbetaling af Fleksible Fastkursaftaler før tid er gebyrfri.

2) Gebyret omfatter bl.a. serviceudskrifter som låneoversigt og afdragsforløb samt udarbejdelse af kopier af pantebrev og regninger. Yderligere kræver Nykredit følgende gebyr for de angivne hændelser.

B.1.3 Tinglysningsafgift

Tinglysningsafgifter til staten for ejerboliger og fritidshuse udgør 1,5% af pantebrevets pålydende i procentafgift samt 1.400 kr. i fast afgift pr. dokument. Tinglysningsafgiften oprundes til nærmeste hele 100 kr.

Hændelsesbetinget gebyrer		
Produkt	Hændelse	Pris
Indfrielse	Gebyr for tilbud Gebyr for indfrielse	200 kr. pr. tilbud 500 kr. pr. lån
Refinansieringsændring	Gebyr for refinansieringsændring	200 kr.
Nedsparingslån	Gebyr for ændring af udbetalingsbeløb	500 kr.
Rentesikring	Gebyr for aftale Gebyr for ophævelsesberegning	400 kr. 400 kr.
Terminsrestance	Morarente	1,4% af terminsydelsen pr. påbegyndt måned regnet fra forfaldsdag
	Erindringsbrev	100 kr. pr. skrivelse
	Henstandsaftale	100 kr. pr. ejendom
	- den første inden for 1 år	gratis
	Misligholdelsesgebyr	100 kr. pr. ejendom

Tabel B.3: Liste over Nykrdits gebyrer i forbindelse med de hændelser der kan benyttes af låntageren.

B.1.4 Kurtage og kursfradrag ved obligationshandel

Ved salg af obligationer til dagskurs betales kurtage af kursværdien. Desuden foretages et kursfradrag.

Kurtageoversigt		
	Kursværdi mindre end 3 mio. kr.	Kursværdi større end 3 mio. kr.
Kurtage	0,15% af kursværdien min. 150 kr. og max. 3.000 kr.	0,10% af kursværdien
Kursfradrag	0,10 kurspoint	0,10 kurspoint
Kursfradrag uden for Københavns Fondsbørs' åbningstid	0,25 - 0,35 kurspoint	0,25 - 0,35 kurspoint

Tabel B.4: Liste over kurtage i forbindelse med Nykredits realkreditlån.

Ved Fastkursaftaler betales kurtage af kursværdien og kursfradrag som ved salg af obligationer til dagskurs.

B.1.5 Bidrag

Bidragets størrelse og beregningsmetode fastsættes i forbindelse med ydelsen af lånet. Bidragssatsen er afhængig af ejendomskategori, belåningsinterval (lånets prioritetsstilling på optagelsestidspunktet i forhold til den kontante låneværdi for ejendommen) samt lånets hovedstol. Bidraget kan ændres af Nykredit i lånets løbetid.

Forhøjelse af bidraget kan gennemføres med 3 måneders varsel til en 1/1, 1/4, 1/7 eller 1/10 med virkning for førstkommende terminsperiode.

Forhøjelse af bidraget kan gennemføres af Nykredit i lånets løbetid som følge af øgede omkostninger, herunder skatter og afgifter, tab og nedskrivninger eller i øvrigt, hvis det er ønskeligt for Nykredit af indtjeningsmæssige grunde, herunder, men ikke alene, behov for forbedring af kapitalgrundlaget.

Nykredit kan altid uden varsel nedsætte bidragets størrelse.

Bidraget beregnes med de anførte procenter af lånets restgæld (dvs. af kontantrestgælden for kontantlån og indekslån og af obligationsrestgælden for obligationslån). Satserne er angivet i procent pr. år, og betalingen sker forholdsmæssigt som en del af lånets kvartårlige eller halvårlige terminsydelse. For lån under 200.000 kr. beregnes på tilbudstidspunktet en samlet bidragssats. Beregningsmetoden for bidraget kan med tre måneders varsel til en 1/1, 1/4, 1/7 eller 1/10 ændres med virkning for førstkommende terminsperiode, hvis det er ønskeligt for Nykredit af administrative, markedsmæssige eller konkurrencemæssige grunde.

Bidraget ved omlægning af lån beregnes efter særlige regler, hvis lånet er optaget før 6. maj 1991, og der ikke skal ske ny vurdering af ejendommen: For lån, der oprindeligt er ydet som enhedslån, beregnes bidraget som ved fuld belåning inden for den pågældende ejendomskategori. For lån, der oprindeligt er ydet som basislån, beregnes bidraget ud fra det laveste belåningsinterval i den pågældende ejendomskategori.

Bidragssatser for lån til private		
Belåningsinterval	Hovedstol	Bidrag *)
0-50%	0-200.000 kr.	400 kr. + 0,1400%
	200.000- 5 mio. kr.	0.34%
	5-10 mio. kr.	0.29%
	10- mio. kr.	0.19%
50-80%	0-200.000 kr.	400 kr. + 0,8000%
	200.000- 5 mio. kr.	1%
	5-10 mio. kr.	0.95%
	10- mio. kr.	0.85%
0-80% **)	0-200.000 kr.	400 kr. + 0,3876%
	200.000- 5 mio. kr.	0.5876%
	5-10 mio. kr.	0.5376%
	10- mio. kr.	0.4376%

Tabel B.5: *) Kernekunder får automatisk 10% rabat på bidraget. **) Fritidshuse kan max. belånes med 60%.

Bilag C

Implementering af de deskriptive modeller

C.1 Brugervejledning til applikationen

VBA applikationen til at danne et BDT-træ og til beregning af prisen på en rentesikring kan nemt anvendes ved at følge nedenstående procedure:

1. Åben filen Rentesikring.xls

2. Under arket 'BDT' kan man bygge et kortrentetræ uafhængigt af resten af applikationen. Udfyld inputfelterne med nulkuponrenter og volatiliteter. Aktivér herefter 'BDT rentetræ' knappen og indtast antallet af år, for hvilket BDT-træet skal dannes. Dette må ikke være større end antallet af år, der er indtastet data for.

3. Når rentetræet er dannet, kan man finde prisen på rentesikringen. Under arket 'Pris' udfyld inputfelterne med: *Restgæld på tidspunkt 0*, *Restløbetid i år*, *Første år med sikring*, *Antal sikringer*, *Sikringsrenten* samt *Antal terminer pr år*.

4. Aktivér 'Find Sikringspris' knappen. Prisen kan nu ses i det blå felt.

Det skal bemærkes, at koden ikke er sikret imod at der indtastes forkerte input. Inputfelterne vil i forvejen være udfyldte med repræsentative data. Det kan være en god idé at gemme en kopi af de oprindelige data inden ændringer i inputdata foretages.

C.2 Kilekode

C.2.1 BDT knappen

```
Option Explicit
Private Sub MakeBDT_Click()
Dim i As Integer
Dim num As Integer

num = InputBox("Indtast antallet af år, for hvilket BDT træet skal dannes.")
Application.ScreenUpdating = False

MakeBDTtree num

''''''''formatting'''''''''
Range(Cells(6, 5), Cells(7, 44)).Select
    Selection.Font.ColorIndex = 0
    Selection.Borders(xlDiagonalDown).LineStyle = xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    With Selection.Borders(xlEdgeLeft)
        .LineStyle = xlContinuous
        .Weight = xlMedium
        .ColorIndex = xlAutomatic
    End With
    With Selection.Borders(xlEdgeTop)
        .LineStyle = xlContinuous
        .Weight = xlMedium
        .ColorIndex = xlAutomatic
    End With
    With Selection.Borders(xlEdgeBottom)
        .LineStyle = xlContinuous
        .Weight = xlMedium
        .ColorIndex = xlAutomatic
    End With
    With Selection.Borders(xlEdgeRight)
        .LineStyle = xlContinuous
        .Weight = xlMedium
        .ColorIndex = xlAutomatic
    End With
    Selection.Borders(xlInsideHorizontal).LineStyle = xlNone
    Selection.Interior.ColorIndex = 15
''''''''formatting'''''''''

End Sub
```

C.2.2 BDTbuilderen

```
Option Explicit

Public Sub MakeBDTtree(num As Integer)

Dim i, j As Integer
Dim N As Integer
Dim delta, z0 As Double

Dim ZeroYields() As Double
Dim ZeroYieldsVol() As Double
Dim ZeroPrices() As Double
```

```
Dim puvector As Variant
Dim pdvector() As Double
Dim Yu() As Double
Dim Yd() As Double
Dim testyield() As Double
Dim Kvector As Variant
Dim Xvector As Variant
Dim shortRatesTree() As Double
Dim zeroPricesTree() As Double

N = num
ReDim ZeroYields(1 To N - 1)
ReDim ZeroYieldsVol(1 To N - 1)
ReDim ZeroPrices(1 To N)
ReDim puvector(1 To N - 1)
ReDim pdvector(1 To N - 1)
ReDim Yu(1 To N - 1)
ReDim Yd(1 To N - 1)
ReDim testyield(1 To N - 1)
ReDim Kvector(1 To N - 1)
ReDim Xvector(1 To N - 1)
ReDim shortRatesTree(1 To N, 1 To N)
ReDim zeroPricesTree(1 To N + 1, 1 To N + 1)

' read in the input data
z0 = Cells(6, 5)
delta = Cells(8, 2)
For i = 1 To N - 1
    ZeroYields(i) = Cells(6, i + 5)
Next i
For i = 1 To num - 1
    ZeroYieldsVol(i) = Cells(7, i + 5)
Next i

' make zero prices from given yields and write them in row 9
Range(Cells(9, 1), Cells(21, 200)).ClearContents

Cells(9, 1) = "Nulkupon priser, fra givne nulkuponrenter"
ZeroPrices(1) = (1 + z0) ^ (-1 * delta)
Cells(9, 5) = ZeroPrices(1)
For i = 2 To num
    ZeroPrices(i) = (1 + ZeroYields(i - 1)) ^ (-i * delta)
    Cells(9, i + 4) = ZeroPrices(i)
Next i

' make the pu vector and write it's values in row 11
puvector = BDTPuvec(z0, ZeroYields, ZeroYieldsVol, delta)
Cells(11, 1) = "Pu (løsning) via JamshidianBDT"
For i = 1 To N - 1
    Cells(11, i + 5) = puvector(i)
Next i

' make the pd vector and write it's values in row 12
Cells(12, 1) = "Pd (løsning)"
For i = 1 To N - 1
    pdvector(i) = 2 * ZeroPrices(i + 1) * (1 + z0) - puvector(i)
    Cells(12, i + 5) = pdvector(i)
Next i

' make the Yu vector and write it's values in row 14
Cells(14, 1) = "Yu (løsning)"
For i = 1 To N - 1
```

```
    Yu(i) = puvector(i) ^ (-1 / (i * delta)) - 1
    Cells(14, i + 5) = Yu(i)
Next i

' make the Yd vector and write it's values in row 15
Cells(15, 1) = "Yd (løsning)"
For i = 1 To N - 1
    Yd(i) = pdvector(i) ^ (-1 / (i * delta)) - 1
    Cells(15, i + 5) = Yd(i)
Next i

' make the testyield vector and write it's values in row 16
' this is merely to test if we can match the input volatilities
Cells(16, 1) = "Rentevolatiliteter (til testning)"
For i = 1 To N - 1
    testyield(i) = 0.5 * Log(Yu(i) / Yd(i)) / Sqr(delta)
    Cells(16, i + 5) = testyield(i)
Next i

' make the K vector and write it's values in row 18
Kvector = BDTKvec(puvector, pdvector)
Cells(18, 1) = "Kvec (løsning) via JamshidianBDT"
For i = 1 To N - 1
    Cells(18, i + 5) = Kvector(i)
Next i

' make the X vector and write it's values in row 19
Xvector = BDTXvec(puvector, pdvector, Kvector)
Cells(19, 1) = "Xvec (løsning) via JamshidianBDT"
For i = 1 To N - 1
    Cells(19, i + 5) = Xvector(i)
Next i

Range(Cells(22, 1), Cells(200, 200)).Select
    Selection.Clear
    Selection.Interior.ColorIndex = 2
' make the shortRatesTree and write it's values in row 25 to 24+n
Cells(21, 1) = "Rentetræ:"
Cells(21, 1).Font.Bold = True
shortRatesTree(1, 1) = z0
Cells(22, 2) = z0
For j = 2 To N
    shortRatesTree(1, j) = Xvector(j - 1)
    Cells(22, j + 1) = shortRatesTree(1, j)
Next j
For i = 2 To N
  For j = i To N
      shortRatesTree(i, j) = shortRatesTree(i - 1, j) / Kvector(j - 1)
      Cells(21 + i, j + 1) = shortRatesTree(i, j)
  Next j
Next i

'''''''formating'''''''
Range(Cells(22, 2), Cells(21 + N, 1 + N)).Select
    With Selection.Interior
        .ColorIndex = 37
        .Pattern = xlSolid
    End With
    Selection.Font.Bold = True
    Selection.Borders(xlDiagonalDown).LineStyle = xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    With Selection.Borders(xlEdgeLeft)
```

```
        .LineStyle = xlContinuous
        .Weight = xlMedium
        .ColorIndex = xlAutomatic
    End With
    With Selection.Borders(xlEdgeTop)
        .LineStyle = xlContinuous
        .Weight = xlMedium
        .ColorIndex = xlAutomatic
    End With
    With Selection.Borders(xlEdgeBottom)
        .LineStyle = xlContinuous
        .Weight = xlMedium
        .ColorIndex = xlAutomatic
    End With
    With Selection.Borders(xlEdgeRight)
        .LineStyle = xlContinuous
        .Weight = xlMedium
        .ColorIndex = xlAutomatic
    End With
    Selection.Borders(xlInsideVertical).LineStyle = xlNone
    Selection.Borders(xlInsideHorizontal).LineStyle = xlNone
''''''''''''''''''''''

' make the zeroPricesTree and write it's values in row 27+n to 28+2n
Cells(23 + N, 1) = "Nulkuponpris træ:"
For i = 1 To N + 1
    zeroPricesTree(i, N + 1) = 1
    Cells(23 + i + N, N + 2) = 1
Next i
For j = N To 1 Step -1
  For i = 1 To j
    zeroPricesTree(i, j) = 0.5 * (zeroPricesTree(i, j + 1) _
              + zeroPricesTree(i + 1, j + 1)) / (1 + shortRatesTree(i, j))
    Cells(23 + i + N, j + 1) = zeroPricesTree(i, j)
  Next i
Next j

'''''''formating'''''''
Range(Cells(22, 2), Cells(21 + N, 1 + N)).Select
    Selection.NumberFormat = "0.00%"
Range(Cells(24 + N, 2), Cells(25 + 2 * N, 2 + N)).Select
    Selection.NumberFormat = "0.0000"
''''''''''''''''''''''

End Sub
```

C.2.3 BDT funktioner

```
Option Explicit
Option Base 1
Function BDTPuvec(z0, zvec, sigvec, delt)
'   Replicates Goal Seek to find initial Pu vector in BDT
'   Discrete interest rates only
'   See Jamshidian (1991)
    Dim atol, r0, jdt, zpj, puj, sigj, pdj, yuj, ydj, fval, fdashval
    Dim j As Integer, N As Integer
    Dim puvec() As Variant
    N = Application.Count(zvec)
    ReDim puvec(N)
    atol = 0.000001
```

```
    r0 = (1 + z0) ^ (delt) - 1
    For j = 1 To N
        jdt = j * delt
        zpj = (1 + zvec(j)) ^ -((j + 1) * delt)
        puj = zpj * (1 + r0)
        sigj = sigvec(j)
        Do
            pdj = 2 * zpj * (1 + r0) - puj
            yuj = (puj ^ (-1 / jdt)) - 1
            ydj = (pdj ^ (-1 / jdt)) - 1
            fval = 0.5 * Log(yuj / ydj) - sigj * Sqr(delt)
            fdashval = -(0.5 / jdt) * ((puj ^ -(1 / jdt + 1)) _
                                        / yuj + (pdj ^ -(1 / jdt + 1)) / ydj)
            puj = puj - (fval / fdashval)
        Loop While Abs(fval) > atol
        puvec(j) = puj
    Next j
    BDTPuvec = puvec
End Function

Function BDTKvec(puvec, pdvec)
'   From Pu and Pd vectors to approximation vector K in BDT
'   See Jamshidian (1991)
    Dim i As Integer, N As Integer
    Dim kvec() As Variant
    N = Application.Count(puvec)
    ReDim kvec(N)
    kvec(1) = ((1 / puvec(1)) - 1) / ((1 / pdvec(1)) - 1)
    For i = 2 To N
        kvec(i) = ((puvec(i - 1) / puvec(i)) - 1) -
                                        / ((pdvec(i - 1) / pdvec(i)) - 1)
    Next i
    BDTKvec = kvec
End Function

Function BDTXvec(puvec, pdvec, kvec)
'   From Pu and Pd vectors to approximation vector X in BDT
'   See Jamshidian (1991)
    Dim pvec0
    Dim i As Integer, N As Integer
    Dim xvec() As Variant
    N = Application.Count(puvec)
    ReDim xvec(N)
    xvec(1) = ((1 / (0.5 * (puvec(1) + pdvec(1)))) - 1) _
                               / (0.5 * (1 + 1 / kvec(1)))
    For i = 2 To N
        xvec(i) = (((puvec(i - 1) + pdvec(i - 1)) _
             / (puvec(i) + pdvec(i))) - 1) / ((0.5 * (1 + 1 / kvec(i))) ^ i)
    Next i
    BDTXvec = xvec
End Function
```

C.2.4 Rentesikring pris knappen

```
Option Explicit

Private Sub FindSikringspris_Click()
Dim SikretBDT As Variant
Dim BDT() As Double
```

```
Dim Paymentvec As Variant
Dim Restgaeldvec As Variant
Dim Valuevec As Variant
Dim N As Integer, b As Integer, NumberofCaps As Integer
Dim Xs As Double
Dim i As Integer, j As Integer
Dim RG_0 As Double
Dim Sik_start As Integer

Application.ScreenUpdating = False
Range(Cells(4, 13), Cells(4, 15)).ClearContents
Range(Cells(10, 1), Cells(200, 200)).Select
    Selection.Clear
    Selection.Interior.ColorIndex = 2
' read in the input data from sheet BDT and Pris
RG_0 = Cells(7, 1)
N = Cells(7, 4)
Sik_start = Cells(7, 6)
NumberofCaps = Cells(7, 9)
Xs = Cells(7, 11)
b = Cells(7, 13)

ReDim BDT(1 To N, 1 To N)
ReDim SikretBDT(1 To N, 1 To N)
ReDim Paymentvec(1 To N, 1 To b * N)
ReDim Restgaeldvec(1 To N, 1 To b * N)
ReDim Valuevec(1 To N + 1, 1 To b * N + 1)
' Read in the BDT tree from the BDT sheet
For j = 1 To N
  For i = 1 To j
    BDT(i, j) = Worksheets("BDT").Cells(21 + i, j + 1)
  Next i
Next j

' Genererer en rentemodel med rentesikring og skriver den ud
SikretBDT = BDTcapped(BDT, N, Sik_start, Xs, NumberofCaps)
For j = 1 To N
  For i = 1 To j
    Cells(i + 9, j).Select
    Selection = SikretBDT(i, j)
    With Selection.Interior
        .ColorIndex = 19
        .Pattern = xlSolid
    End With
    Selection.NumberFormat = "0.00%"
  Next i
Next j

' Genererer og udskrive et ydelsestræ som indeholder den første
' ydelse for et annuitetslån der starter i den givne knude og
' har løbetid indtil slut
Cells(11 + N, 1) = "Ydelse ultimo perioden med Rentesikring:"
Paymentvec = Makepayvec(SikretBDT, b, N)
For j = (b * N) To 1 Step -1
  For i = 1 To ((j - 1) \ b) + 1
    Cells(11 + N + i, j).Select
    Selection = Paymentvec(i, j)
    With Selection.Interior
        .ColorIndex = 19
        .Pattern = xlSolid
    End With
    Selection.NumberFormat = "0.0000"
```

```
  Next i
Next j

' Genererer og udskrive restgældstræet der indeholder restgælden
' for hver knude efter den første ydelse er betalt, hvor den
' oprindelige restgæld er 1 i hver knude
Cells(13 + (2 * N), 1) = "Restgæld ultimo perioden med Rentesikring:"
Restgaeldvec = MakeRGvec(SikretBDT, Paymentvec, b, N)
For j = (b * N) To 1 Step -1
  For i = 1 To ((j - 1) \ b) + 1
    Cells(13 + (2 * N) + i, j).Select
    Selection = Restgaeldvec(i, j)
    With Selection.Interior
        .ColorIndex = 19
        .Pattern = xlSolid
    End With
    Selection.NumberFormat = "0.0000"
  Next i
Next j

' Genererer værditræet for hver knude ved at anvende leveling
Cells(15 + (3 * N), 1) = "Værdi af rentesikring:"
Valuevec = Makevaluesvec(BDT, Paymentvec, Restgaeldvec, b, N, _
                              Sik_start, NumberofCaps)
For j = (Sik_start + NumberofCaps - 1) * b + 1 To 1 Step -1
  For i = 1 To ((j - 1) \ b) + 1
    Cells(15 + (3 * N) + i, j).Select
    Selection = Valuevec(i, j)
    With Selection.Interior
        .ColorIndex = 19
        .Pattern = xlSolid
    End With
    Selection.NumberFormat = "0.0000"
  Next i
Next j

Range(Cells(4, 13), Cells(4, 15)) = -1 * Valuevec(1, 1) * RG_0

End Sub
```

C.2.5 Rentesikring funktioner

```
Option Explicit
Option Base 1

' Input:
' BDT: En N årig BDT rentestrukturmodel
' Xs: Årlig sikringsrente
' NumberofCaps: Antal sikrede refinansieringer (maks N-1)
' Output:
' Et tilsvarende BDT træ med sikring
Function BDTcapped(BDT, N, Sik_start, Xs, NumberofCaps)
Dim i, j As Integer
Dim cappedvec() As Double
ReDim cappedvec(1 To N, 1 To N)

For j = 1 To N
  For i = 1 To j
    cappedvec(i, j) = BDT(i, j)
```

```
  Next i
Next j

For j = Sik_start To Sik_start - 1 + NumberofCaps
  For i = 1 To j
    If BDT(i, j) > Xs Then
      cappedvec(i, j) = Xs
    End If
  Next i
Next j

BDTcapped = cappedvec
End Function

' input:
' SikretBDT : Et BDT træ hvor de renter der overstiger markedsrenten
'             er erstattet af sikringsrenten
' b : Antal terminer pr år
' N : Restløbetid (år)
' output:
' Et ydelsestræ som indeholder den første ydelse for et annuitetslån,
' der starter i den givne knude og har løbetid indtil slut
' jævnfør "Investment Science, David G. Luenberger, § 14.5"
Function Makepayvec(SikretBDT, b, N)
Dim i, j As Integer
Dim RG, temp As Double
Dim payvec() As Double
ReDim payvec(1 To N, 1 To b * N)

RG = 1 ' Restgælden er 1 i alle knuder

For j = (b * N) To 1 Step -1
  For i = 1 To ((j - 1) \ b) + 1
    ' terminslig rente for det aktuelle år
    temp = SikretBDT(i, ((j - 1) \ b + 1)) / b
    payvec(i, j) = (RG * temp) / (1 - (1 + temp) ^ -(b * N + 1 - j))
  Next i
Next j

Makepayvec = payvec
End Function

' input:
' SikretBDT : Et BDT træ hvor de renter der overstiger markedsrenten
'             er erstattet af sikringsrenten
' payvec : Et ydelsestræ som indeholder den første ydelse for et
'           annuitetslån, der starter i den givne knude og har
'           løbetid indtil slut
' b : Antal terminer pr år
' N : Restløbetid (år)
' output:
' MakeRGvec : Restgældtræet efter den første betaling jævnfør
'              payvec træet
' jævnfør "Investment Science, David G. Luenberger, § 14.5"
Function MakeRGvec(SikretBDT, payvec, b, N)
Dim i, j As Integer
Dim RG, temp As Double
Dim RGvec() As Double
ReDim RGvec(1 To N, 1 To b * N)

RG = 1 ' Restgælden er 1 i alle knuder
```

```
For j = (b * N) To 1 Step -1
  For i = 1 To ((j - 1) \ b) + 1
    ' terminslig rente for det aktuelle år
    temp = SikretBDT(i, ((j - 1) \ b + 1)) / b
    RGvec(i, j) = RG - ((1 + temp) ^ -(b * N + 1 - j)) * payvec(i, j)
  Next i
Next j

MakeRGvec = RGvec

End Function

' input:
' BDT : Et BDT træ der representere markedsrenterne
' payvec : Et ydelsestræ som indeholder den første ydelse for et annuitetslån,
'          der starter i den givne knude og har løbetid indtil slut
' RGvec :  Restgældtræet efter den første betaling jævnfør payvec træet
' b : Antal terminer pr år
' N : Restløbetid (år)
' output:
' Makevaluevec : Værditræet for rentesikringen, hvor hver knude angiver
'                værdien af rentesikring for et lån på 1
'                jævnfør "Investment Science, David G. Luenberger, § 14.5"
Function Makevaluesvec(BDT, payvec, RGvec, b, N, Sik_start, NumberofCaps)
Dim i, j As Integer
Dim RG, temp As Double
Dim valuesvec() As Double
ReDim valuesvec(1 To Sik_start + NumberofCaps, 1 _
                 To (Sik_start + NumberofCaps - 1) * b + 1)

RG = 1 ' Restgælden er 1 i alle knuder

For i = 1 To Sik_start + NumberofCaps
    valuesvec(i, (Sik_start + NumberofCaps - 1) * b + 1) = 0
Next i

For j = (Sik_start + NumberofCaps - 1) * b To 1 Step -1
  For i = 1 To ((j - 1) \ b) + 1
    temp = BDT(i, ((j - 1) \ b + 1)) / b ' terminslig rente for det aktuelle år
    If j Mod b = 0 Then
      valuesvec(i, j) = (RGvec(i, j) * _
                         (1 + 0.5 * (valuesvec(i + 1, j + 1) _
                       + valuesvec(i, j + 1)) / RG) _
                       + payvec(i, j)) / (1 + temp) - RG
      Else
      valuesvec(i, j) = (RGvec(i, j) * (1 + valuesvec(i, j + 1) / RG) _
                              + payvec(i, j)) / (1 + temp) - RG
    End If
  Next i
Next j

Makevaluesvec = valuesvec

End Function
```

Bilag D

Koden til generering af inputdata til GAMS

D.1 VBA program til generering af GAMS tabeller

Følgende kode skal skrives i et microsoft excel objekt tilknyttet et excel ark, hvor tabellerne skal skrivers ud.

```
Option Explicit


Private Sub GRT_Click()
Dim i As Integer, j As Integer
Dim TabelFlag As Integer
Dim LaanFlag As Integer
Dim ALgentag As Integer
Dim NumScen As Integer
Dim NumSøjler As Integer
Dim NumLaan As Integer
Dim StartRække As Integer
Dim DataLatticeDepth As Integer

''''''''''''''''''''''INPUT DATA, SKAL TILPASSES TABELSTØRRELSEN''''''''''
DataLatticeDepth = 11 ' Antallet af niveauer
NumSøjler = 10        ' Antallet af søjler i tabellerne
NumLaan = 24          ' Antallet af fastforrentede obligationer i
kurser_steen arket
StartRække = 2        ' Rækkenummer hvorfra tabellerne udskrives i Tabel
arket
''''''''''''''''''''''END INPUT DATA''''''''''''''''''''''''''''''''''''

NumScen = (2 ^ (DataLatticeDepth)) - 1 ' Antal scenarier i det ikke
kombinerende træ

Application.ScreenUpdating = False
Range(Cells(StartRække + 1, 1), Cells(6000, 200)).Select
    Selection.ClearContents

TabelFlag = InputBox("1 for renter, 2 for kurser og 3 Kaldkuser")
```

```
LaanFlag = InputBox("1 for fastforrentet Obligationslån, 2 for fastforrentet
obligationslån og F1 lån, 3 for fastforrentet obligationslån, F1 lån og F1
lån med rentesikring.")

Select Case LaanFlag
    Case 2
        NumLaan = NumLaan + 1
    Case 3
        NumLaan = NumLaan + 2
End Select

'Her laver vi rammerne for tabellerne
For i = 0 To NumScen - 1
  Cells(StartRække + 1 + ((NumLaan + 1) * Int(i / NumSøjler)), _
        i + 2 - NumSøjler * Int(i / NumSøjler)) = "Sc" & i + 1
Next i

ALgentag = NumScen / NumSøjler

For i = 0 To ALgentag
    For j = 1 To NumLaan
      Cells(StartRække + 1 + j + ((NumLaan + 1) * i), 1) = "Laan" & j
    Next j
    Cells(1 + StartRække + 1 - 1 + j + ((NumLaan + 1) * i), 1) = "+"
Next i

'En sub bliver kaldt alt efter hvilken type data, der skal udfyldes
Select Case TabelFlag
  Case 1
    UdfyldRenter ALgentag, NumScen, NumSøjler, NumLaan, StartRække,
DataLatticeDepth, LaanFlag
  Case 2
    UdfyldKurser ALgentag, NumScen, NumSøjler, NumLaan, StartRække,
DataLatticeDepth, LaanFlag
  Case 3
    UdfyldCallKurser ALgentag, NumScen, NumSøjler, NumLaan, StartRække,
DataLatticeDepth, LaanFlag
End Select


End Sub
```

D.2 Hjælpefunktioner og klasser

Følgende funktioner og klasser skal skrives i et VBA projekts moduler og klasser. Funktionaliteterne bliver brugt i koden fra sidste afsnit.

D.2.1 Funktionerne

```
Option Explicit
Dim SteenRenteStart As Integer
Dim F1Start As Integer

'Sub til generering af renter i tabellen, CASE 1
Sub UdfyldRenter(ALgentag As Integer, NumScen As Integer, NumSøjler As
Integer, NumLaan As Integer, StartRække As Integer, DataLatticeDepth As
Integer, LaanFlag As Integer)
```

```
Dim i As Integer, j As Integer, k As Integer
Dim OblUdDato() As Integer
Dim DataLattice As Lattice
Dim AntalNoder As Integer
Dim ScenarioILattice As Integer
Dim FandtMapping As Boolean
''''''''''''''''''''''''INPUT DATA''''''''''''''''
F1Start = 40  'linjen hvor F1 lån starter DATA INPUT
''''''''''''''''''''''''INPUT DATA''''''''''''''''
Set DataLattice = New Lattice
DataLattice.Depth = DataLatticeDepth
AntalNoder = DataLattice.NumberOfNodes 'Gemmer antal noder i lattice og
initialiserer lattice

If NumLaan > 2 Then
  ReDim OblUdDato(1 To NumLaan - LaanFlag + 1)
End If

SteenRenteStart = 4   ' Rækken i Kurser_steen arket, hvor renterne begynder
Worksheets("Kurser_steen").Activate
For i = 1 To NumLaan - LaanFlag + 1
  For j = 2 To 256
    If Cells(i + SteenRenteStart - 1, j) <> 0 Then
      Exit For
    End If
  Next j
  OblUdDato(i) = Int((Cells(SteenRenteStart - 2, j)))
Next i

Worksheets("Tabel").Activate
Cells(StartRække, 1) = "TABLE R(I,S) scenariotrae for renter"

' Først udskriver vi fastforrentede obligationslån
For i = 0 To NumScen - 1
  For j = 1 To NumLaan - LaanFlag + 1
    If OblUdDato(j) > Int(Log(i + 1) / Log(2)) Then
      Cells(j + StartRække + 1 + ((NumLaan + 1) * Int(i / NumSøjler)), _
            i + 2 - NumSøjler * Int(i / NumSøjler)) = 1
    Else
      Cells(j + StartRække + 1 + ((NumLaan + 1) * Int(i / NumSøjler)), _
            i + 2 - NumSøjler * Int(i / NumSøjler)) = _
            Worksheets("Kurser_steen").Cells(SteenRenteStart + j - 1, 1) /
100
    End If
  Next j
Next i

'For at sikre os at vi udskriver de rigtige lån
Select Case LaanFlag
  Case 2  'Så tilføjer vi et F1 lån
    For i = 0 To NumScen - 1
      FandtMapping = False  ' Vi har ikke fundet en mapping endnu
      For k = 1 To AntalNoder
        For j = 1 To DataLattice.LatticeKnude(k).AntalElementer
          If DataLattice.LatticeKnude(k).MapElement(j).MappedNode = i + 1
Then
            ScenarioILattice = DataLattice.LatticeKnude(k).Number
            FandtMapping = True ' Nu er en mapping fundet
            Exit For            ' så behøver ikke tjekke de øvrige mapping
for denne knuder
          End If
        Next j
```

```
        If FandtMapping = True Then
          Exit For  ' og når en mapping er fundet springer vi resten af
knuderne over også
        End If
      Next k

      Cells(NumLaan + StartRække + 1 + ((NumLaan + 1) * Int(i / NumSøjler)),
_
            i + 2 - NumSøjler * Int(i / NumSøjler)) = _
      Worksheets("Kurser_steen").Cells(F1Start, ScenarioILattice + 1)
    Next i

  Case 3  ' Eller et F1 lån og et F1 lån med rentesikring
    For i = 0 To NumScen - 1
      FandtMapping = False  ' Vi har ikke fundet en mapping endnu
      For k = 1 To AntalNoder
        For j = 1 To DataLattice.LatticeKnude(k).AntalElementer
          If DataLattice.LatticeKnude(k).MapElement(j).MappedNode = i + 1
Then
            ScenarioILattice = DataLattice.LatticeKnude(k).Number
            FandtMapping = True ' Nu er en mapping fundet
            Exit For           ' så behøver ikke tjekke de øvrige mapping
for denne knuder
          End If
        Next j
        If FandtMapping = True Then
          Exit For  ' og når en mapping er fundet springer vi resten af
knuderne over også
        End If
      Next k

      For j = 0 To 1
        Cells(NumLaan + StartRække + j + ((NumLaan + 1) * Int(i /
NumSøjler)), _
              i + 2 - NumSøjler * Int(i / NumSøjler)) = _
        Worksheets("Kurser_steen").Cells(F1Start + j * 2, ScenarioILattice +
1)
      Next j
    Next i
End Select

Cells(1, 1).Select
End Sub
'Sub til generering af kurser i tabellen, CASE 2
Sub UdfyldKurser(ALgentag As Integer, NumScen As Integer, NumSøjler As
Integer, NumLaan As Integer, StartRække As Integer, DataLatticeDepth As
Integer, LaanFlag As Integer)
Dim i As Integer, j As Integer, k As Integer
Dim Kursen As Double
Dim DataLattice As Lattice
Dim AntalNoder As Integer
Dim ScenarioILattice As Integer
Dim FandtMapping As Boolean
''''''''''''''''''''''''INPUT DATA''''''''''''''''
F1Start = 40  'linjen hvor F1 lån starter DATA INPUT
''''''''''''''''''''''''INPUT DATA''''''''''''''''
Set DataLattice = New Lattice
DataLattice.Depth = DataLatticeDepth
AntalNoder = DataLattice.NumberOfNodes 'Gemmer antal noder i lattice og
initialiserer lattice

SteenRenteStart = 4   ' Rækken i Kurser_steen arket, hvor renterne begynder
```

```
Worksheets("Tabel").Activate
Cells(StartRække, 1) = "TABLE K(I,S) scenariotrae for kurser"
For i = 0 To NumScen - 1
  FandtMapping = False  ' Vi har ikke fundet en mapping endnu
  For k = 1 To AntalNoder
    For j = 1 To DataLattice.LatticeKnude(k).AntalElementer
      If DataLattice.LatticeKnude(k).MapElement(j).MappedNode = i + 1 Then
        ScenarioILattice = DataLattice.LatticeKnude(k).Number
        FandtMapping = True ' Nu er en mapping fundet
        Exit For            ' så behøver ikke tjekke de øvrige mapping for
denne knuder
      End If
    Next j
    If FandtMapping = True Then
      Exit For  ' og når en mapping er fundet springer vi resten af knuderne
over også
    End If
  Next k

  Select Case LaanFlag
    Case 1
      For j = 1 To NumLaan
        Kursen = Worksheets("Kurser_steen").Cells(SteenRenteStart + j - 1,
ScenarioILattice + 1)
        If Kursen > 0 And Kursen < 100 Then
          Cells(j + StartRække + 1 + ((NumLaan + 1) * Int(i / NumSøjler)), _
                i + 2 - NumSøjler * Int(i / NumSøjler)) = Kursen / 100
        Else
          Cells(j + StartRække + 1 + ((NumLaan + 1) * Int(i / NumSøjler)), _
                i + 2 - NumSøjler * Int(i / NumSøjler)) = 0
        End If
      Next j
    Case 2
      For j = 1 To NumLaan
        If j < NumLaan Then
          Kursen = Worksheets("Kurser_steen").Cells(SteenRenteStart + j - 1,
ScenarioILattice + 1)
          If Kursen > 0 And Kursen < 100 Then
            Cells(j + StartRække + 1 + ((NumLaan + 1) * Int(i / NumSøjler)),
_
                  i + 2 - NumSøjler * Int(i / NumSøjler)) = Kursen / 100
          Else
            Cells(j + StartRække + 1 + ((NumLaan + 1) * Int(i / NumSøjler)),
_
                  i + 2 - NumSøjler * Int(i / NumSøjler)) = 0
          End If
        Else
          Kursen = Worksheets("Kurser_steen").Cells(F1Start + 1,
ScenarioILattice + 1)
          Cells(j + StartRække + 1 + ((NumLaan + 1) * Int(i / NumSøjler)), _
                i + 2 - NumSøjler * Int(i / NumSøjler)) = Kursen / 100
        End If

      Next j
    Case 3
      For j = 1 To NumLaan
        Select Case j
          Case NumLaan - 1
            Kursen = Worksheets("Kurser_steen").Cells(F1Start + 1,
ScenarioILattice + 1)
          Case NumLaan
            Kursen = Worksheets("Kurser_steen").Cells(F1Start + 3,
```

```
ScenarioILattice + 1)
          Case Else
            Kursen = Worksheets("Kurser_steen").Cells(SteenRenteStart + j -
1, ScenarioILattice + 1)
        End Select

        If Kursen > 0 And Kursen <= 100 Then
          Cells(j + StartRække + 1 + ((NumLaan + 1) * Int(i / NumSøjler)), _
                i + 2 - NumSøjler * Int(i / NumSøjler)) = Kursen / 100
        Else
          Cells(j + StartRække + 1 + ((NumLaan + 1) * Int(i / NumSøjler)), _
                i + 2 - NumSøjler * Int(i / NumSøjler)) = 0
        End If
      Next j
  End Select
Next i

Cells(1, 1).Select
End Sub
'Sub til generering af Kaldkurser i tabellen, CASE 3
Sub UdfyldCallKurser(ALgentag As Integer, NumScen As Integer, NumSøjler As
Integer, NumLaan As Integer, StartRække As Integer, DataLatticeDepth As
Integer, LaanFlag As Integer)
Dim i As Integer, j As Integer, k As Integer
Dim Kursen As Double
Dim DataLattice As Lattice
Dim AntalNoder As Integer
Dim ScenarioILattice As Integer
Dim FandtMapping As Boolean
Dim TempString As String
''''''''''''''''''''''''''INPUT DATA'''''''''''''''''
F1Start = 40  'linjen hvor F1 lån starter DATA INPUT
''''''''''''''''''''''''''INPUT DATA'''''''''''''''''
Set DataLattice = New Lattice
DataLattice.Depth = DataLatticeDepth
AntalNoder = DataLattice.NumberOfNodes 'Gemmer antal noder i lattice og
initialiserer lattice

SteenRenteStart = 4   ' Rækken i Kurser_steen arket, hvor renterne begynder
Worksheets("Tabel").Activate
Cells(StartRække, 1) = "TABLE CallK(I,S) scenariotrae for Call kurser"
For i = 0 To NumScen - 1
  FandtMapping = False  ' Vi har ikke fundet en mapping endnu
  For k = 1 To AntalNoder
    For j = 1 To DataLattice.LatticeKnude(k).AntalElementer
      If DataLattice.LatticeKnude(k).MapElement(j).MappedNode = i + 1 Then
        ScenarioILattice = DataLattice.LatticeKnude(k).Number
        FandtMapping = True ' Nu er en mapping fundet
        Exit For            ' så behøver ikke tjekke de øvrige mapping for
denne knuder
      End If
    Next j
    If FandtMapping = True Then
      Exit For  ' og når en mapping er fundet springer vi resten af knuderne
over også
    End If
  Next k

  Select Case LaanFlag
    Case 1
      For j = 1 To NumLaan
        Kursen = Worksheets("Kurser_steen").Cells(SteenRenteStart + j - 1,
```

```
ScenarioILattice + 1)
        If Kursen > 0 And Kursen < 100 Then
          Cells(j + StartRække + 1 + ((NumLaan + 1) * Int(i / NumSøjler)), _
                i + 2 - NumSøjler * Int(i / NumSøjler)) = Kursen / 100
        Else
          Cells(j + StartRække + 1 + ((NumLaan + 1) * Int(i / NumSøjler)), _
                i + 2 - NumSøjler * Int(i / NumSøjler)) = 1
        End If
      Next j
    Case 2
      For j = 1 To NumLaan
        If j < NumLaan Then
          Kursen = Worksheets("Kurser_steen").Cells(SteenRenteStart + j - 1,
ScenarioILattice + 1)
        Else
          Kursen = 100
        End If
        If Kursen > 0 And Kursen < 100 Then
          Cells(j + StartRække + 1 + ((NumLaan + 1) * Int(i / NumSøjler)), _
                i + 2 - NumSøjler * Int(i / NumSøjler)) = Kursen / 100
        Else
          Cells(j + StartRække + 1 + ((NumLaan + 1) * Int(i / NumSøjler)), _
                i + 2 - NumSøjler * Int(i / NumSøjler)) = 1
        End If
      Next j
    Case 3
      For j = 1 To NumLaan
        Select Case j
          Case NumLaan - 1
            Kursen = 100
          Case NumLaan
            Kursen = 100
          Case Else
            Kursen = Worksheets("Kurser_steen").Cells(SteenRenteStart + j -
1, ScenarioILattice + 1)
        End Select

        If Kursen > 0 And Kursen < 100 Then
          Cells(j + StartRække + 1 + ((NumLaan + 1) * Int(i / NumSøjler)), _
                i + 2 - NumSøjler * Int(i / NumSøjler)) = Kursen / 100
        Else
          Cells(j + StartRække + 1 + ((NumLaan + 1) * Int(i / NumSøjler)), _
                i + 2 - NumSøjler * Int(i / NumSøjler)) = 1
        End If
      Next j
  End Select
Next i

''''Sandsynlighederne og faste omkostninger skrivers i parametre i ark
SF'''''''''
Worksheets("SF").Activate
Application.ScreenUpdating = False
Range(Cells(1, 1), Cells(2000, 100)).Select
    Selection.ClearContents
Dim TempSSDouble As Double
Dim TempSSInt As Integer

NumSøjler = NumSøjler / 2  ' Vi bruger kun det halve antal søjler for
parametrene
Cells(StartRække, 1) = "Prob(S)  Sandsynligheden for de enkelte scenarier"
For i = 0 To NumScen - 1
  Cells(StartRække + 1 + Int(i / NumSøjler), _
```

```
        2 * (i + 1 - NumSøjler * Int(i / NumSøjler))) = "Sc" & i + 1

  TempSSDouble = Log(i + 1) / Log(2)
  TempSSInt = Int(TempSSDouble)
  TempSSDouble = 2 ^ (-TempSSInt)
  If (Int((i + 1) / NumSøjler) <> (i + 1) / NumSøjler And i < NumScen - 1)
Then
    Cells(StartRække + 1 + Int(i / NumSøjler), _
    1 + 2 * (i + 1 - NumSøjler * Int(i / NumSøjler))) = TempSSDouble & ","
'2 ^ Fix((-(Log(i + 1) / Log(2)))) & ","
  Else
    Cells(StartRække + 1 + Int(i / NumSøjler), _
          1 + 2 * (i + 1 - NumSøjler * Int(i / NumSøjler))) = TempSSDouble
'2 ^ Fix((-(Log(i + 1) / Log(2))))
  End If
Next i


StartRække = StartRække + NumScen / NumSøjler + 3
Cells(StartRække, 1) = "Fast(S)  Faste omkostninger i forbindelse med
forskellige scenarier"
For i = 0 To NumScen - 1
  Cells(StartRække + 1 + Int(i / NumSøjler), _
        2 * (i + 1 - NumSøjler * Int(i / NumSøjler))) = "Sc" & i + 1
  If (Int((i + 1) / NumSøjler) <> (i + 1) / NumSøjler And i < NumScen - 1)
Then
    Cells(StartRække + 1 + Int(i / NumSøjler), _
    1 + 2 * (i + 1 - NumSøjler * Int(i / NumSøjler))) = "1000 ,"
  Else
    Cells(StartRække + 1 + Int(i / NumSøjler), _
          1 + 2 * (i + 1 - NumSøjler * Int(i / NumSøjler))) = 1000
  End If
Next i


Cells(1, 1).Select
End Sub
```

D.2.2 Klasserne

Klasse Lattice

```
Option Explicit
Public Depth As Integer
'Public NumberOfNodes As Integer

Private LatticeNodes() As Node
Dim AntalLatticeNodesLokal As Integer




Public Function NumberOfNodes() As Integer
  Dim i As Integer, j As Integer
  Dim NedreGrænse() As Integer

  NumberOfNodes = Depth * (Depth + 1) / 2
  ReDim LatticeNodes(1 To NumberOfNodes) As Node
  ReDim NedreGrænse(1 To Depth)
```

```
''''''''''Indsætning af numrene'''''''''''''''
For i = 1 To NumberOfNodes
  Set LatticeNodes(i) = New Node
  LatticeNodes(i).Number = i
Next i
''''''''''END Indsætning af numrene'''''''''''


''''''''''Indsætning af niveauerne''''''''''''
For i = 1 To Depth
  NedreGrænse(i) = i * (i + 1) / 2
Next i

For i = 1 To NumberOfNodes
  For j = 1 To Depth
    If LatticeNodes(i).Number <= NedreGrænse(j) Then
      LatticeNodes(i).Level = j - 1
      Exit For
    End If
  Next j
Next i
''''''''''END Indsætning af niveauerne''''''''


''''''''''Indsætning af børneknude''''''''''''
For i = 1 To NumberOfNodes - Depth
  For j = 1 To Depth
    If LatticeNodes(i).Number <= NedreGrænse(j) Then
      LatticeNodes(i).ChildUp = LatticeNodes(i).Number + j
      LatticeNodes(i).ChildDown = LatticeNodes(i).Number + j + 1
      Exit For
    End If
  Next j
Next i
''''''''''END Indsætning af børneknude''''''''


''''''''''Indsætning af forældreknude'''''''''
For i = 2 To NumberOfNodes
  For j = 1 To NumberOfNodes - Depth
    If LatticeNodes(i).Number = LatticeNodes(j).ChildDown Then
      LatticeNodes(i).ParrentUp = LatticeNodes(j).Number
    ElseIf LatticeNodes(i).Number = LatticeNodes(j).ChildUp Then
      LatticeNodes(i).ParrentDown = LatticeNodes(j).Number
    End If
  Next j
Next i
''''''''''END Indsætning af forældreknude'''''

'Lokale variable til brug for mapping algoritmen
Dim GemParrentDown As Integer
Dim GPDmapnode As Integer
Dim GemParrentUp As Integer
Dim GPUmapnode As Integer
Dim AntalDownMappings() As Integer
ReDim AntalDownMappings(1 To NumberOfNodes)
''''''''''Indsætning af mappings''''''''''''''
'Først initialiserer vi mappingen for knude 1
LatticeNodes(1).AntalElementer = 1
LatticeNodes(1).MapElement(1).MappedNode = 1
'Derefter opdaterer vi mappingerne for de øvrige knuder rekursivt
For i = 2 To NumberOfNodes
```

```
    If LatticeNodes(i).ParrentDown <> 0 Then
      GemParrentDown = LatticeNodes(i).ParrentDown
      AntalDownMappings(i) = LatticeNodes(GemParrentDown).AntalElementer
      For j = 1 To AntalDownMappings(i)
        GPDmapnode = LatticeNodes(GemParrentDown).MapElement(j).MappedNode
        LatticeNodes(i).AntalElementer = LatticeNodes(i).AntalElementer + 1
        LatticeNodes(i).MapElement(j).MappedNode = GPDmapnode * 2
      Next j
    End If

    If LatticeNodes(i).ParrentUp <> 0 Then
      GemParrentUp = LatticeNodes(i).ParrentUp
      For j = 1 To LatticeNodes(GemParrentUp).AntalElementer
        GPUmapnode = LatticeNodes(GemParrentUp).MapElement(j).MappedNode
        LatticeNodes(i).AntalElementer = LatticeNodes(i).AntalElementer + 1
        LatticeNodes(i).MapElement(j + AntalDownMappings(i)).MappedNode =
GPUmapnode * 2 + 1
      Next j
    End If
  Next i
  ''''''''''END Indsætning af mappings'''''''''''
End Function

'Dim Mapping() As NodeTilNode


'Public Property Let AntalElementer(ind As Integer)
'Dim i As Integer
'AntalMappingLokal = ind
'If ind > 0 Then
'  ReDim Preserve Mapping(1 To ind)
'End If
'
'For i = 1 To ind
'  If Mapping(i) Is Nothing Then
'    Set Mapping(i) = New NodeTilNode
'  End If
'Next i
'
'End Property

'Public Property Get AntalElementer() As Integer
'  AntalElementer = AntalMappingLokal
'End Property

Public Property Get LatticeKnude(i As Integer) As Node
  Set LatticeKnude = LatticeNodes(i)
End Property
```

Klasse Node

```
Option Explicit

Public Number As Integer
Public ParrentUp As Integer
Public ParrentDown As Integer
Public ChildUp As Integer
Public ChildDown As Integer
Public Level As Integer
```

```
Dim Mapping() As NodeTilNode
Dim AntalMappingLokal As Integer

Public Property Let AntalElementer(ind As Integer)
Dim i As Integer
AntalMappingLokal = ind
If ind > 0 Then
  ReDim Preserve Mapping(1 To ind)
End If

For i = 1 To ind
  If Mapping(i) Is Nothing Then
    Set Mapping(i) = New NodeTilNode
  End If
Next i

End Property

Public Property Get AntalElementer() As Integer
  AntalElementer = AntalMappingLokal
End Property
Public Property Get MapElement(i As Integer) As NodeTilNode
  Set MapElement = Mapping(i)
End Property
```

Klasse NodeTilNode

```
Public MappedNode As Integer
```

Bilag E

Implementering af optimeringsmodellen

Vi vil i det følgende nøjes med at gengive grundmodellerne i deres helhed. Alle tilføjelserne til grundmodellerne gengives dernæst som kodestykker som grundmodellen skal opdateres med.

E.1 Grundmodellen med risikoneutral målfunktion

E.1.1 GAMS formulering

```
$ontext
Resultater/Lneutral/MedF1/laan.gms : Vælger et lån eller en kombination
af lån ved scenario 1 og dernæst plejer denne låneportefølje optimelt.
Med faste omkostninger, mip formulering.(risiko neutral låntager)

En multi-stage stokastisk heltalslåneoptimeringsmodel
med en scenario represæntation, hvor tiden indgår kun implicit
$offtext

$eolcom //
option
iterlim=999999999,reslim=600,optcr=0.01,solprint=OFF,limrow=0,limcol=0;

$include ../../dataMedF1_11.txt // generelle data
$include ../../SFparms.txt // Sandsynligheder og faste omkostninger
$include ../../FastF1Tables.txt // rente, kurs og callkurs tabeller

VARIABLES
       RG(I,S)   Restgaeld
       Sale(I,S) Salgsvariabel
       P(I,S)    Koebsvariabel
       A(I,S)    Afdrag
       B(S)      Betaling
       IB(S)     Indfrielsesbetaling
       L(I,S)    Indikator for faste omkostninger
       MB        Maksimal betaling
       Z         Objektfunktionsvaerdien ;
```

```
POSITIVE VARIABLES RG, Sale , P;
BINARY VARIABLE L;

EQUATIONS
  COST        Definere objektfunktionen
  EQ1         Hele laanet skal daekkes til at starte med
  EQ2(I)  Maengden af solgte obliagationer ved scenario 1 udgoere startrestgaelden
  EQ3(I)      Det er ikke tilladt at koebe obligationer ved scenario 1
  EQ4(I,S,S2) Balanceligninger
  EQ5(S)      Betalingsstroemligninger
  EQ6(I,S)    Definition af afdrag
  EQ7(S)      Definition af betaling
  EQ8(I,S)    Faste omlaegningsomkostninger
  EQ9(S)      Definition af infrielsesbeloeb  ;

  COST         ..   Z  =E=  SUM(S, Prob(S)*B(S)) + SUM(S$(ORD(S) > 2**(Udloeb-1)-1), Prob(S)*IB(S))
                            + FastOpr + TinglP*sum(I, RG(I,'Sc1'));
  EQ1          ..   SUM(I, K(I,'Sc1')*RG(I,'Sc1')) =G= Provenu;
  EQ2(I)       ..   RG(I,'Sc1') - Sale(I,'Sc1') =E= 0;
  EQ3(I)       ..   P(I,'Sc1') =E= 0;
  EQ4(I,S,S2)$(Floor(ORD(S)/2)=ORD(S2))  .. RG(I,S2) - A(I,S2)
                                                - P(I,S) + Sale(I,S) - RG(I,S) =E= 0;
  EQ5(S)$(ORD(S) > 1)    ..   sum(I,K(I,S)*Sale(I,S)) - sum(I,CallK(I,S)*P(I,S)) =E= 0;
  EQ6(I,S)     ..   A(I,S) - RG(I,S)*( ( R(I,S)/(1- (1+R(I,S))
                               **(-N+floor(log(ORD(S))/log(2))) )- R(I,S) ) ) =E= 0;
  EQ7(S)     ..  B(S)  - sum(I, A(I,S) + (1- gamma)*RG(I,S)*R(I,S) + (1-betta) * RG(I,S)
                * bidrag + FastOml*L(I,S) + kurtage*(Sale(I,S)+P(I,S)) ) =E= 0;
  EQ8(I,S) .. BigM * L(I,S) - Sale(I,S) + P(I,S) =G= 0;
  EQ9(S)$(ORD(S) > 2**(Udloeb-1)-1) ..   IB(S) - sum(I, RG(I,S) - A(I,S)) =E= 0;


  MODEL LOANOPTIM /ALL/ ;
  OPTION MIP=CPLEX;
  SOLVE LOANOPTIM USING MIP MINIMIZING Z ;
  display Sale.l, P.l, L.l, RG.l, A.l, B.l, IB.l;
```

E.1.2 GAMS/SCENRED formulering

```
$ontext
multi1_4_SR.gms : Vælger et lån eller en kombination af lån ved scenario 1
og dernæst plejer denne låneportefølje optimelt. Uden faste omkostninger,
lp formulering. (risiko neutral låntager) Med scenario reduction

En multi-stage stokastisk heltalslåneoptimeringsmodel
med en scenariorepresæntation, hvor tiden indgår eksplicit
$offtext

$eolcom //
option
iterlim=999999999,reslim=300,optcr=0.01,solprint=OFF,limrow=12,limcol=0;
set
I    laan produkter             / laan1*laan25/,
tb   traeblade                  /bl1*bl12/;
$include ../../dataMedF1_11.txt // generelle data
$include ../../SFparms.txt // Sandsynligheder og faste omkostninger
$include ../../FastF1Tables.txt // rente, kurs og callkurs tabeller
$include ../../dataMedF1_11SR.txt // Scenario reduction speceifikke data

*display leaf;
*display ts;
```

```
*display prob;

VARIABLES
      RG(I,T,S)   Outstanding debt
      Sale(I,T,S) Sale variable
      P(I,T,S)    Purchase variable
      A(I,T,S)    Afdrag
      B(T,S)      Betaling
      SB(tb)      Total stibetaling
      IB(T,S)     Indfrielsesbeloeb
      IBT(tb)     Indfrielsesbeloeb
      L(I,T,S)    Indikator for faste omkostninger
      Z           Total value of the loan costs ;
POSITIVE VARIABLES RG, Sale , P;
BINARY VARIABLE L;

EQUATIONS
 COST       Definere objektfunktionen
 EQ1        Hele laanet skal daekkes til at starte med
 EQ2(I)     Maengden af solgte obliagationer ved scenario 1 udgoere startrestgaelden
 EQ3(I)     Det er ikke tilladt at koebe obligationer ved scenario 1
 EQ4(I,T,S) Balanceligninger
 EQ5(T,S)   Betalingsstroemligninger
 EQ6(I,T,S) Definition af afdrag
 EQ7(T,S)   Definition af betaling
 EQ8(I,T,S) Faste omlaegningsomkostninger
 EQ9(T,S)      Definition af infrielsesbeloeb
 DefIB1, DefIB2, DefIB3, DefIB4, DefIB5, DefIB6,
 DefIB7, DefIB8, DefIB9, DefIB10, DefIB11, DefIB12
 BerSB1, BerSB2, BerSB3, BerSB4,  BerSB5, BerSB6,
 BerSB7, BerSB8, BerSB9, BerSB10,  BerSB11, BerSB12;


  COST   ..   Z  =E=  SUM(ts(t,ss), sprob(ss)*B(T,ss)) + SUM(ts(t,ss)$(ord(t)>10), sprob(ss)*IB(T,ss))
                      + FastOpr + TinglP*sum(I, RG(I,'time0','sc1'));
  EQ1    ..   SUM(I, K(I,'sc1')*RG(I,'time0','sc1')) =G= Provenu;
  EQ2(I) ..   RG(I,'time0','sc1') - Sale(I,'time0','sc1') =E= 0;
  EQ3(I) ..   P(I,'time0','sc1') =E= 0;
  EQ4(I,ts(T,SS)) .. sum(tree(s2,ss), RG(I,T-1,S2) - A(I,T-1,S2)
                    - P(I,T,SS) + Sale(I,T,SS) - RG(I,T,SS)) =E= 0;
  EQ5(ts(t,ss))$(ord(t)>1) .. sum(I,K(I,SS)*Sale(I,T,SS)) -
                         sum(I,CallK(I,SS)*P(I,T,SS)) =E= 0;
  EQ6(I,ts(t,ss)) .. A(I,T,SS) - RG(I,T,SS)*( ( R(I,SS)/(1- (1+R(I,SS))
                                    **(-N-1+ord(t)) )- R(I,SS) ) ) =E= 0;
  EQ7(ts(t,ss)) .. B(T,SS) - sum(I, A(I,T,SS)+ (1- gamma)*RG(I,T,SS)*R(I,SS)
                + (1-betta)*RG(I,T,SS)*bidrag + FastOml*L(I,T,SS)
                + kurtage*(Sale(I,T,SS)+P(I,T,SS))  ) =E= 0;
  EQ8(I,ts(t,ss)) ..  BigM * L(I,T,SS) - Sale(I,T,SS) + P(I,T,SS) =G= 0;
  EQ9(ts(t,ss))$(ord(t)>10 ) ..  IB(T, SS) - sum(I, RG(I,T,SS) - A(I,T,SS)) =E= 0;
DefIB1   .. IBT('bl1') =E=  IB('time10','Sc1061');
DefIB2   .. IBT('bl2') =E=  IB('time10','Sc1197');
DefIB3   .. IBT('bl3') =E=  IB('time10','Sc1297');
DefIB4   .. IBT('bl4') =E=  IB('time10','Sc1353');
DefIB5   .. IBT('bl5') =E=  IB('time10','Sc1354');
DefIB6   .. IBT('bl6') =E=  IB('time10','Sc1704');
DefIB7   .. IBT('bl7') =E=  IB('time10','Sc1705');
DefIB8   .. IBT('bl8') =E=  IB('time10','Sc1707');
DefIB9   .. IBT('bl9') =E=  IB('time10','Sc1720');
DefIB10  .. IBT('bl10') =E= IB('time10','Sc1754');
DefIB11  .. IBT('bl11') =E= IB('time10','Sc1910');
DefIB12  .. IBT('bl12') =E= IB('time10','Sc2042');
BerSB1   .. SB('bl1') =E= B('time0','Sc1') + B('time1', 'Sc2') + B('time2', 'Sc4')
```

```
                + B('time3', 'Sc8') + B('time4', 'Sc16') + B('time5', 'Sc33')
                + B('time6', 'Sc66') + B('time7', 'Sc132') + B('time8', 'Sc265')
                + B('time9', 'Sc530') + B('time10','Sc1061') + IBT('bl1');
BerSB2   .. SB('bl2') =E= B('time0','Sc1') + B('time1', 'Sc2') + B('time2', 'Sc4')
                + B('time3', 'Sc9') + B('time4', 'Sc18') + B('time5', 'Sc37')
                + B('time6', 'Sc74') + B('time7', 'Sc149') + B('time8', 'Sc299')
                + B('time9', 'Sc598') + B('time10','Sc1197') + IBT('bl2');
BerSB3   .. SB('bl3') =E= B('time0','Sc1') + B('time1', 'Sc2') + B('time2', 'Sc5')
                + B('time3', 'Sc10') + B('time4', 'Sc20') + B('time5', 'Sc40')
                + B('time6', 'Sc81') + B('time7', 'Sc162') + B('time8', 'Sc324')
                + B('time9', 'Sc648') + B('time10','Sc1297') + IBT('bl3');
BerSB4   .. SB('bl4') =E= B('time0','Sc1') + B('time1', 'Sc2') + B('time2', 'Sc5')
                + B('time3', 'Sc10') + B('time4', 'Sc21') + B('time5', 'Sc42')
                + B('time6', 'Sc84') + B('time7', 'Sc169') + B('time8', 'Sc338')
                + B('time9', 'Sc676') + B('time10','Sc1353') + IBT('bl4');
BerSB5   .. SB('bl5') =E= B('time0','Sc1') + B('time1', 'Sc2') + B('time2', 'Sc5')
                + B('time3', 'Sc10') + B('time4', 'Sc21') + B('time5', 'Sc42')
                + B('time6', 'Sc84') + B('time7', 'Sc169') + B('time8', 'Sc338')
                + B('time9', 'Sc677') + B('time10','Sc1354') + IBT('bl5');
BerSB6   .. SB('bl6') =E= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc6')
                + B('time3', 'Sc13') + B('time4', 'Sc26') + B('time5', 'Sc53')
                + B('time6', 'Sc106') + B('time7', 'Sc213') + B('time8', 'Sc426')
                + B('time9', 'Sc852') + B('time10','Sc1704') + IBT('bl6');
BerSB7   .. SB('bl7') =E= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc6')
                + B('time3', 'Sc13') + B('time4', 'Sc26') + B('time5', 'Sc53')
                + B('time6', 'Sc106') + B('time7', 'Sc213') + B('time8', 'Sc426')
                + B('time9', 'Sc852') + B('time10','Sc1705') + IBT('bl7');
BerSB8   .. SB('bl8') =E= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc6')
                + B('time3', 'Sc13') + B('time4', 'Sc26') + B('time5', 'Sc53')
                + B('time6', 'Sc106') + B('time7', 'Sc213') + B('time8', 'Sc426')
                + B('time9', 'Sc853') + B('time10','Sc1707') + IBT('bl8');
BerSB9   .. SB('bl9') =E= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc6')
                + B('time3', 'Sc13') + B('time4', 'Sc26') + B('time5', 'Sc53')
                + B('time6', 'Sc107') + B('time7', 'Sc215') + B('time8', 'Sc430')
                + B('time9', 'Sc860') + B('time10','Sc1720') + IBT('bl9');
BerSB10  .. SB('bl10') =E= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc6')
                + B('time3', 'Sc13') + B('time4', 'Sc27') + B('time5', 'Sc54')
                + B('time6', 'Sc109') + B('time7', 'Sc219') + B('time8', 'Sc438')
                + B('time9', 'Sc877') + B('time10','Sc1754') + IBT('bl10');
BerSB11  .. SB('bl11') =E= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc7')
                + B('time3', 'Sc14') + B('time4', 'Sc29') + B('time5', 'Sc59')
                + B('time6', 'Sc119') + B('time7', 'Sc238') + B('time8', 'Sc477')
                + B('time9', 'Sc955') + B('time10','Sc1910') + IBT('bl11');
BerSB12  .. SB('bl12') =E= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc7')
                + B('time3', 'Sc15') + B('time4', 'Sc31') + B('time5', 'Sc63')
                + B('time6', 'Sc127') + B('time7', 'Sc255') + B('time8', 'Sc510')
                + B('time9', 'Sc1021') + B('time10','Sc2042') + IBT('bl12');

  MODEL LOANOPTIM /ALL/ ;
  OPTION LP=CPLEX;
**********************************ScenRed calls start her*********************
$if set noscenred $goto noscenreduction

* now we prepare to run ScenRed

* this includes some sets & parameters used for scenred I/O
$libinclude scenred.gms

scalar psum, rc, runCount, runMax;
set    run  / run1 * run1 /;
set    method 'reduction method used' /
        '0-default',
```

```
        '1-fastback',
        '2-fastback+forw',
        '3-fastback+back'  /;
parameter report(method,run, *);
set       rleaf(method,run,s)  'leaf set of reduced tree';


runMax = INF;
$if set runmax runMax = %runmax%;

* set up the scenred options file
file opts  /'scenred.opt'/;
putclose opts  'log_file   multilog.txt'
             / 'input_gdx  multiwin.gdx'
             / 'output_gdx multiout.gdx';


* these parms are based on the input tree
ScenRedParms('num_leaves') = sum {leaf, 1};
ScenRedParms('num_random') = 3*card(I);
ScenRedParms('num_nodes')  = card(s);
ScenRedParms('num_time_steps') = card(t);
* optional SCENRED input parameters
ScenRedParms('num_stages') = ScenRedParms('num_time_steps');
ScenRedParms('where_random') = 10;
ScenRedParms('report_level') = 0;
ScenRedParms('run_time_limit') = 3000;

runCount = 0;
*loop {method$(runCount < runMax),
*    ScenRedParms('reduction_method') = ord(method)-1;
    ScenRedParms('reduction_method') = 0;
    loop {run$(runCount < runMax),
*       these parms control the tree output from ScenRed
*       at least one of the following two parameters is required
        ScenRedParms('red_num_leaves') = 12;
*       ScenRedParms('red_num_leaves') = ord(run);
*       ScenRedParms('red_percentage') = 0.5;

        execute_unload 'multiwin.gdx',  ScenRedParms, s, tree, prob, r, k, callk ;
        execute 'rm -f multiout.gdx';
        execute 'scenred scenred.opt %system.redirlog%';
        rc = errorlevel;
        abort$rc "Return code from scenred was nonzero : ", rc;
        execute_load   'multiout.gdx', ScenRedReport, sprob=red_prob;

        ss(s) = sprob(s);
        display ScenRedParms, ScenRedReport;
        display sprob, ss;

        psum = sum {leaf(ss), sprob(ss)};
        abort$[abs(psum-1) gt 1e-8]
          "Error in reduced tree: leaf probabilities do not sum to 1";

        solve loanoptim min z us mip;
        runCount = runCount + 1;

        report(method,run, 'obj') = loanoptim.objval;
        report(method,run, 'red_percentage') =
             ScenRedReport('red_percentage');
        report(method,run, 'reduction_method') =
             ScenRedReport('reduction_method');
```

```
        report(method,run, 'run_time') =
             ScenRedReport('run_time');
        rleaf (method,run, leaf(ss)) = YES;
    };
*};

display report;
display Sale.l, B.l, RG.l, IB.l, SB.l;
*display rleaf;
*display r;
*display k;
*display callk;

*********************************ScenRed calls end her********************
$goto alldone

$label noscenreduction
* set "reduced tree" to be the whole tree
ss(s) = yes;
sprob(s) = prob(s);

SOLVE LOANOPTIM USING MIP MINIMIZING Z ;
display Sale.l, P.l, RG.l;

$label alldone
```

E.2 Risikoaverse modeller

kodestykkerne i dette afsnit skal opdateres eller tilføjes i grundmodellen.

E.2.1 GAMS/formuleing

Minimax–formulering

```
set
tb   traeblade                 / bl1*bl1024/  ;
$include ../../dataMedF1_11.txt // generelle data
$include ../../../../../SFtabel.txt // Scenarioforloeb
$include ../../SFparms.txt // Sandsynligheder og faste omkostninger
$include ../../FastF1Tables.txt // rente, kurs og callkurs tabeller


VARIABLES
       IB(tb)     Indfrielsesbetaling
       MB        Maksimal betaling


EQUATIONS
  COST        Definere objektfunktionen
  DefMB(tb)   Definitione paa maksimal betaling
  TOTBETAL    Den gennemsnitlige totale omkostning
  EQ9(tb,S)   Definition af infrielsesbeloeb  ;

  COST         ..   Z =E= MB;
  DefMB(tb)    ..   1*MB =G= sum(s, SF(tb,S)*B(S)) + IB(tb);
  TOTBETAL     ..   total =E= SUM(S, Prob(S)*B(S));
```

```
  EQ9(tb,s)$(ord(tb)=ord(s)-1023) .. IB(tb) - sum(I, RG(I,S) - A(I,S)) =E= 0;
```

Likviditetsrisikoaversitet

```
set
tb   traeblade                  / bl1*bl1024/  ;
$include ../dataMedF1_11.txt    // generelle data
$include ../SFparms.txt         // Sandsynligheder og faste omkostninger
$include ../../../../../SFtabel.txt // Scenarioforloeb beskrivelsen
$include ../FastF1Tables.txt    // rente, kurs og callkurs tabeller

VARIABLES
       SB(tb)    stiBetaling
       BO(tb)    stibetalingsoverskridelse
       IB(S)     Indfrielsesbetaling
       IBO(S)    Indfrielsesbetalingsoverskridelse

EQUATIONS
  COST        Definere objektfunktionen
  KBB(tb)     stibetalingsbegraensning
  KBO(tb)     stibetalings maksimale graense
  BIBB(S)     Blad indfrielsesbetalingsbegraensning
  BIBO(S)     Blad indfrielsesbetalings maksimale graense
  TOTBETAL    Den gennemsnitlige totale omkostning
  EQ9(S)      Definition af infrielsesbeloeb
  BerSB(tb)   Beregning af scenarieforloebsbetalinger;


   COST         ..   Z =E= (1/(2**(Udloeb-1)))*sum(tb, (BMAX - SB(tb) - StrafBO*BO(tb)))
                           + sum(S$(ORD(S) > 2**(Udloeb-1)-1), Prob(S)*((IBmax-IB(S))
                           - StrafBO*IBO(S))) + FastOpr + TinglP*sum(I, RG(I,'Sc1')) ;
  KBB(tb)       ..   BMAX + BO(tb) - SB(tb) =G= 0;
  KBO(tb)        ..   BO(tb) =L= BOMAX ;
  BIBB(S)$(ORD(S) > 2**(Udloeb-1)-1) .. IBmax + IBO(S) - IB(S) =G= 0;
  BIBO(S)$(ORD(S) > 2**(Udloeb-1)-1) .. IBO(S) =L= IBOMAX ;
  TOTBETAL     ..   total =E= SUM(S, Prob(S)*B(S))
                     + SUM(S$(ORD(S) > 2**(Udloeb-1)-1), Prob(S)*IB(S))
                     + FastOpr + TinglP*sum(I, RG(I,'Sc1')) ;
  EQ9(S)$(ORD(S) > 2**(Udloeb-1)-1) ..   IB(S) - sum(I, RG(I,S) - A(I,S)) =E= 0;
  BerSB(tb) .. SB(tb) =E= sum(S,B(S)*SF(tb,S));
```

Likviditets– og formuerisikoaversitet

```
set
I    laan produkter            / laan1*laan25/;

VARIABLES
       RGB(T,S)    Restgaeld eller formuebesparelse
       RGU(T,S)    Restgaeld eller formueunderskud
       GRG(T)      Gennemsnitlig restgaeld
       SB(tb)      stiBetaling
       BO(tb)      stibetalingsoverskridelse
       IB(T,S)     Indfrielsesbetaling
       IBO(T,S)    Indfrielsesbetalingsoverskridelse

EQUATIONS
  COST          Definere objektfunktionen
  KBB(tb)       stibetalingsbegraensning
```

```
 KBO(tb)       stibetalings maksimale graense
 BIBB(T,S)     Blad indfrielsesbetalingsbegraensning
 BIBO(T,S)     Blad indfrielsesbetalings maksimale graense
 RGBU(T,S)     Restgaeld eller formuebesparelse og underskud
 TOTBETAL      Den gennemsnitlige totale omkostning
 BerSB(tb)     Beregning af scenarieforloebsbetalinger
 BerGRG(T)     Beregning af den gennemsnitlige restgaeld per tidsperiode;



 COST         ..   Z =E= (1/(2**(Udloeb-1)))*sum(tb, (BMAX - SB(tb) - StrafBO*BO(tb)))
                         +  SUM(ts(t,s)$(ord(t)>10), prob(s)*(IBmax-IB(T,s))
                         - StrafBO*IBO(T,S))
                         + sum(ts(t,s), prob(s)*(PV*RGB(T,S)-NV*(RGU(T,S))) );
 KBB(tb)       ..   BMAX + BO(tb) - SB(tb) =G= 0;
 KBO(tb)        ..   BO(tb) =L= BOMAX ;
 BIBB(ts(t,s))$(ord(t)>10) .. IBmax + IBO(T,S) - IB(T,S) =G= 0;
 BIBO(ts(t,s))$(ord(t)>10) .. IBO(T,S) =L= IBOMAX ;
 RGBU(ts(t,s)) ..  GRG(T) - sum(i, RG(I,T,S)) - RGB(T,S) + RGU(T,S) =E=0;
 TOTBETAL     ..   total =E= (1/(2**(Udloeb-1)))*SUM(tb, SB(tb))
                    + SUM(ts(t,s)$(ORD(T) > 10), Prob(S)*IB(T,S))
                    + FastOpr + TinglP*sum(I, RG(I,'time0','Sc1')) ;
 BerSB(tb) .. SB(tb) =E= sum(ts(t,s),B(T,S)*SF(tb,S));
 BerGRG(t) .. GRG(T) =E= sum((I,ts(t,s)), Prob(S)*RG(I,T,S) );
```

E.2.2 GAMS/SCENRED–formuleringer

Minimax–formulering

```
set
I    laan produkter           / laan1*laan25/,
tb   traeblade                /bl1*bl12/;

VARIABLES
       IB(tb)      Indfrielsesbeloeb
       SB(tb)      Total stibetaling
       MB          Maksimal betaling

EQUATIONS
 COST       Definere objektfunktionen
 DefMB1, DefMB2, DefMB3, DefMB4, DefMB5, DefMB6,
 DefMB7, DefMB8, DefMB9, DefMB10, DefMB11, DefMB12,
 DefIB1, DefIB2, DefIB3, DefIB4, DefIB5, DefIB6,
 DefIB7, DefIB8, DefIB9, DefIB10, DefIB11, DefIB12
 BerSB1, BerSB2, BerSB3, BerSB4,  BerSB5, BerSB6,
 BerSB7, BerSB8, BerSB9, BerSB10,  BerSB11, BerSB12;


 COST         ..   Z =E= MB;
DefMB1   .. MB =G= B('time0','Sc1') + B('time1', 'Sc2') + B('time2', 'Sc4')
                + B('time3', 'Sc8') + B('time4', 'Sc16') + B('time5', 'Sc33')
                + B('time6', 'Sc66') + B('time7', 'Sc132') + B('time8', 'Sc265')
                + B('time9', 'Sc530') + B('time10','Sc1061') + IB('bl1');
DefMB2   .. MB =G= B('time0','Sc1') + B('time1', 'Sc2') + B('time2', 'Sc4')
                + B('time3', 'Sc9') + B('time4', 'Sc18') + B('time5', 'Sc37')
                + B('time6', 'Sc74') + B('time7', 'Sc149') + B('time8', 'Sc299')
                + B('time9', 'Sc598') + B('time10','Sc1197') + IB('bl2');
DefMB3   .. MB =G= B('time0','Sc1') + B('time1', 'Sc2') + B('time2', 'Sc5')
                + B('time3', 'Sc10') + B('time4', 'Sc20') + B('time5', 'Sc40')
                + B('time6', 'Sc81') + B('time7', 'Sc162') + B('time8', 'Sc324')
```

```
                + B('time9', 'Sc648') + B('time10','Sc1297') + IB('bl3');
DefMB4   .. MB =G= B('time0','Sc1') + B('time1', 'Sc2') + B('time2', 'Sc5')
                + B('time3', 'Sc10') + B('time4', 'Sc21') + B('time5', 'Sc42')
                + B('time6', 'Sc84') + B('time7', 'Sc169') + B('time8', 'Sc338')
                + B('time9', 'Sc676') + B('time10','Sc1353') + IB('bl4');
DefMB5   .. MB =G= B('time0','Sc1') + B('time1', 'Sc2') + B('time2', 'Sc5')
                + B('time3', 'Sc10') + B('time4', 'Sc21') + B('time5', 'Sc42')
                + B('time6', 'Sc84') + B('time7', 'Sc169') + B('time8', 'Sc338')
                + B('time9', 'Sc677') + B('time10','Sc1354') + IB('bl5');
DefMB6   .. MB =G= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc6')
                + B('time3', 'Sc13') + B('time4', 'Sc26') + B('time5', 'Sc53')
                + B('time6', 'Sc106') + B('time7', 'Sc213') + B('time8', 'Sc426')
                + B('time9', 'Sc852') + B('time10','Sc1704') + IB('bl6');
DefMB7   .. MB =G= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc6')
                + B('time3', 'Sc13') + B('time4', 'Sc26') + B('time5', 'Sc53')
                + B('time6', 'Sc106') + B('time7', 'Sc213') + B('time8', 'Sc426')
                + B('time9', 'Sc852') + B('time10','Sc1705') + IB('bl7');
DefMB8   .. MB =G= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc6')
                + B('time3', 'Sc13') + B('time4', 'Sc26') + B('time5', 'Sc53')
                + B('time6', 'Sc106') + B('time7', 'Sc213') + B('time8', 'Sc426')
                + B('time9', 'Sc853') + B('time10','Sc1707') + IB('bl8');
DefMB9   .. MB =G= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc6')
                + B('time3', 'Sc13') + B('time4', 'Sc26') + B('time5', 'Sc53')
                + B('time6', 'Sc107') + B('time7', 'Sc215') + B('time8', 'Sc430')
                + B('time9', 'Sc860') + B('time10','Sc1720') + IB('bl9');
DefMB10  .. MB =G= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc6')
                + B('time3', 'Sc13') + B('time4', 'Sc27') + B('time5', 'Sc54')
                + B('time6', 'Sc109') + B('time7', 'Sc219') + B('time8', 'Sc438')
                + B('time9', 'Sc877') + B('time10','Sc1754') + IB('bl10');
DefMB11  .. MB =G= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc7')
                + B('time3', 'Sc14') + B('time4', 'Sc29') + B('time5', 'Sc59')
                + B('time6', 'Sc119') + B('time7', 'Sc238') + B('time8', 'Sc477')
                + B('time9', 'Sc955') + B('time10','Sc1910') + IB('bl11');
DefMB12  .. MB =G= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc7')
                + B('time3', 'Sc15') + B('time4', 'Sc31') + B('time5', 'Sc63')
                + B('time6', 'Sc127') + B('time7', 'Sc255') + B('time8', 'Sc510')
                + B('time9', 'Sc1021') + B('time10','Sc2042') + IB('bl12');
DefIB1   .. IB('bl1') =E= SUM(I, RG(I,'time10','Sc1061') - A(I,'time10','Sc1061'));
DefIB2   .. IB('bl2') =E= SUM(I, RG(I,'time10','Sc1197') - A(I,'time10','Sc1197'));
DefIB3   .. IB('bl3') =E= SUM(I, RG(I,'time10','Sc1297') - A(I,'time10','Sc1297'));
DefIB4   .. IB('bl4') =E= SUM(I, RG(I,'time10','Sc1353') - A(I,'time10','Sc1353'));
DefIB5   .. IB('bl5') =E= SUM(I, RG(I,'time10','Sc1354') - A(I,'time10','Sc1354'));
DefIB6   .. IB('bl6') =E= SUM(I, RG(I,'time10','Sc1704') - A(I,'time10','Sc1704'));
DefIB7   .. IB('bl7') =E= SUM(I, RG(I,'time10','Sc1705') - A(I,'time10','Sc1705'));
DefIB8   .. IB('bl8') =E= SUM(I, RG(I,'time10','Sc1707') - A(I,'time10','Sc1707'));
DefIB9   .. IB('bl9') =E= SUM(I, RG(I,'time10','Sc1720') - A(I,'time10','Sc1720'));
DefIB10  .. IB('bl10') =E= SUM(I, RG(I,'time10','Sc1754') - A(I,'time10','Sc1754'));
DefIB11  .. IB('bl11') =E= SUM(I, RG(I,'time10','Sc1910') - A(I,'time10','Sc1910'));
DefIB12  .. IB('bl12') =E= SUM(I, RG(I,'time10','Sc2042') - A(I,'time10','Sc2042'));
BerSB1   .. SB('bl1') =E= B('time0','Sc1') + B('time1', 'Sc2') + B('time2', 'Sc4')
                + B('time3', 'Sc8') + B('time4', 'Sc16') + B('time5', 'Sc33')
                + B('time6', 'Sc66') + B('time7', 'Sc132') + B('time8', 'Sc265')
                + B('time9', 'Sc530') + B('time10','Sc1061') + IB('bl1');
BerSB2   .. SB('bl2') =E= B('time0','Sc1') + B('time1', 'Sc2') + B('time2', 'Sc4')
                + B('time3', 'Sc9') + B('time4', 'Sc18') + B('time5', 'Sc37')
                + B('time6', 'Sc74') + B('time7', 'Sc149') + B('time8', 'Sc299')
                + B('time9', 'Sc598') + B('time10','Sc1197') + IB('bl2');
BerSB3   .. SB('bl3') =E= B('time0','Sc1') + B('time1', 'Sc2') + B('time2', 'Sc5')
                + B('time3', 'Sc10') + B('time4', 'Sc20') + B('time5', 'Sc40')
                + B('time6', 'Sc81') + B('time7', 'Sc162') + B('time8', 'Sc324')
                + B('time9', 'Sc648') + B('time10','Sc1297') + IB('bl3');
BerSB4   .. SB('bl4') =E= B('time0','Sc1') + B('time1', 'Sc2') + B('time2', 'Sc5')
```

```
                 + B('time3', 'Sc10') + B('time4', 'Sc21') + B('time5', 'Sc42')
                 + B('time6', 'Sc84') + B('time7', 'Sc169') + B('time8', 'Sc338')
                 + B('time9', 'Sc676') + B('time10','Sc1353') + IB('bl4');
BerSB5   .. SB('bl5') =E= B('time0','Sc1') + B('time1', 'Sc2') + B('time2', 'Sc5')
                 + B('time3', 'Sc10') + B('time4', 'Sc21') + B('time5', 'Sc42')
                 + B('time6', 'Sc84') + B('time7', 'Sc169') + B('time8', 'Sc338')
                 + B('time9', 'Sc677') + B('time10','Sc1354') + IB('bl5');
BerSB6   .. SB('bl6') =E= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc6')
                 + B('time3', 'Sc13') + B('time4', 'Sc26') + B('time5', 'Sc53')
                 + B('time6', 'Sc106') + B('time7', 'Sc213') + B('time8', 'Sc426')
                 + B('time9', 'Sc852') + B('time10','Sc1704') + IB('bl6');
BerSB7   .. SB('bl7') =E= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc6')
                 + B('time3', 'Sc13') + B('time4', 'Sc26') + B('time5', 'Sc53')
                 + B('time6', 'Sc106') + B('time7', 'Sc213') + B('time8', 'Sc426')
                 + B('time9', 'Sc852') + B('time10','Sc1705') + IB('bl7');
BerSB8   .. SB('bl8') =E= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc6')
                 + B('time3', 'Sc13') + B('time4', 'Sc26') + B('time5', 'Sc53')
                 + B('time6', 'Sc106') + B('time7', 'Sc213') + B('time8', 'Sc426')
                 + B('time9', 'Sc853') + B('time10','Sc1707') + IB('bl8');
BerSB9   .. SB('bl9') =E= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc6')
                 + B('time3', 'Sc13') + B('time4', 'Sc26') + B('time5', 'Sc53')
                 + B('time6', 'Sc107') + B('time7', 'Sc215') + B('time8', 'Sc430')
                 + B('time9', 'Sc860') + B('time10','Sc1720') + IB('bl9');
BerSB10  .. SB('bl10') =E= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc6')
                 + B('time3', 'Sc13') + B('time4', 'Sc27') + B('time5', 'Sc54')
                 + B('time6', 'Sc109') + B('time7', 'Sc219') + B('time8', 'Sc438')
                 + B('time9', 'Sc877') + B('time10','Sc1754') + IB('bl10');
BerSB11  .. SB('bl11') =E= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc7')
                 + B('time3', 'Sc14') + B('time4', 'Sc29') + B('time5', 'Sc59')
                 + B('time6', 'Sc119') + B('time7', 'Sc238') + B('time8', 'Sc477')
                 + B('time9', 'Sc955') + B('time10','Sc1910') + IB('bl11');
BerSB12  .. SB('bl12') =E= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc7')
                 + B('time3', 'Sc15') + B('time4', 'Sc31') + B('time5', 'Sc63')
                 + B('time6', 'Sc127') + B('time7', 'Sc255') + B('time8', 'Sc510')
                 + B('time9', 'Sc1021') + B('time10','Sc2042') + IB('bl12');
```

Likviditetsrisikoaversitet

```
set
tb   traeblade                 /bl1*bl12/;

parameter stiprob(tb)/
   bl1 0.072, bl2 0.070, bl3 0.107, bl4 0.075
   bl5 0.076, bl6 0.057, bl7 0.083, bl8 0.095
   bl9 0.090, bl10 0.154, bl11 0.091, bl12 0.029/;


VARIABLES
       SB(tb)       stiBetaling
       BO(tb)       stibetalingsoverskridelse
       IB(T,S)      Indfrielsesbetaling
       IBT(tb)      Indfrielsesbeloeb temp
       IBO(T,S)     Indfrielsesbetalingsoverskridelse

EQUATIONS
  COST          Definere objektfunktionen
  KBB(tb)       stibetalingsbegraensning
  KBO(tb)       stibetalings maksimale graense
  BIBB(T,S)     Blad indfrielsesbetalingsbegraensning
  BIBO(T,S)     Blad indfrielsesbetalings maksimale graense
```

```
  TOTBETAL      Den gennemsnitlige totale omkostning ;

  COST   ..   Z =E= sum(tb, stiprob(tb)*(BMAX - SB(tb) - StrafBO*BO(tb)))
                         + SUM(ts(t,ss)$(ord(t)>10), sprob(ss)*(IBmax-IB(T,ss))
                         - StrafBO*IBO(T,SS));
  KBB(tb)       ..   BMAX + BO(tb) - SB(tb) =G= 0;
  KBO(tb)        ..   BO(tb) =L= BOMAX ;
  BIBB(ts(t,ss))$(ord(t)>10) .. IBmax + IBO(T,SS) - IB(T,SS) =G= 0;
  BIBO(ts(t,ss))$(ord(t)>10) .. IBO(T,SS) =L= IBOMAX ;
  TOTBETAL     ..   total =E= SUM(tb, stiprob(tb)* SB(tb))
                    + SUM(ts(t,ss)$(ORD(T) > 10), sProb(SS)*IB(T,SS))
                    + FastOpr + TinglP*sum(I, RG(I,'time0','Sc1')) ;
```

Likviditets– og formuerisikoaversitet

```
set
tb   traeblade                  /bl1*bl12/;

parameter stiprob(tb)/
   bl1 0.072, bl2 0.070, bl3 0.107, bl4 0.075
   bl5 0.076, bl6 0.057, bl7 0.083, bl8 0.095
   bl9 0.090, bl10 0.154, bl11 0.091, bl12 0.029/;

VARIABLES
       RGB(T,S)    Restgaeld eller formuebesparelse
       RGU(T,S)    Restgaeld eller formueunderskud
       GRG(T)      Gennemsnitlig restgaeld
       SB(tb)      stiBetaling
       BO(tb)      stibetalingsoverskridelse
       IB(T,S)     Indfrielsesbetaling
       IBT(tb)     Indfrielsesbeloeb temp
       IBO(T,S)    Indfrielsesbetalingsoverskridelse

EQUATIONS
  COST          Definere objektfunktionen
  KBB(tb)       stibetalingsbegraensning
  KBO(tb)       stibetalings maksimale graense
  BIBB(T,S)     Blad indfrielsesbetalingsbegraensning
  BIBO(T,S)     Blad indfrielsesbetalings maksimale graense
  TOTBETAL      Den gennemsnitlige totale omkostning
  RGBU(T,S)     Restgaeld eller formuebesparelse og underskud
  BerGRG(T)     Beregning af den gennemsnitlige restgaeld per tidsperiode
  DefIB1, DefIB2, DefIB3, DefIB4, DefIB5, DefIB6,
  DefIB7, DefIB8, DefIB9, DefIB10, DefIB11, DefIB12
  BerSB1, BerSB2, BerSB3, BerSB4,  BerSB5, BerSB6,
  BerSB7, BerSB8, BerSB9, BerSB10,  BerSB11, BerSB12;


  COST   ..   Z =E= sum(tb, stiprob(tb)*(BMAX - SB(tb) - StrafBO*BO(tb)))
                         + SUM(ts(t,ss)$(ord(t)>10), sprob(ss)*(IBmax-IB(T,ss))
                         - StrafBO*IBO(T,SS))
                         + sum(ts(t,ss), prob(ss)*(PV*RGB(T,SS)-NV*(RGU(T,SS))) );
  KBB(tb)       ..   BMAX + BO(tb) - SB(tb) =G= 0;
  KBO(tb)        ..   BO(tb) =L= BOMAX ;
  BIBB(ts(t,ss))$(ord(t)>10) .. IBmax + IBO(T,SS) - IB(T,SS) =G= 0;
  BIBO(ts(t,ss))$(ord(t)>10) .. IBO(T,SS) =L= IBOMAX ;
  TOTBETAL     ..   total =E= SUM(tb, stiprob(tb)* SB(tb))
                    + SUM(ts(t,ss)$(ORD(T) > 10), sProb(SS)*IB(T,SS))
                    + FastOpr + TinglP*sum(I, RG(I,'time0','Sc1')) ;
```

```
  RGBU(ts(t,ss)) ..  GRG(T) - sum(i, RG(I,T,SS)) - RGB(T,SS) + RGU(T,SS) =E=0;
  BerGRG(t) .. GRG(T) =E= sum((I,ts(t,ss)), SProb(SS)*RG(I,T,SS) );
DefIB1    .. IBT('bl1') =E=  IB('time10','Sc1061');
DefIB2    .. IBT('bl2') =E=  IB('time10','Sc1197');
DefIB3    .. IBT('bl3') =E=  IB('time10','Sc1297');
DefIB4    .. IBT('bl4') =E=  IB('time10','Sc1353');
DefIB5    .. IBT('bl5') =E=  IB('time10','Sc1354');
DefIB6    .. IBT('bl6') =E=  IB('time10','Sc1704');
DefIB7    .. IBT('bl7') =E=  IB('time10','Sc1705');
DefIB8    .. IBT('bl8') =E=  IB('time10','Sc1707');
DefIB9    .. IBT('bl9') =E=  IB('time10','Sc1720');
DefIB10   .. IBT('bl10') =E= IB('time10','Sc1754');
DefIB11   .. IBT('bl11') =E= IB('time10','Sc1910');
DefIB12   .. IBT('bl12') =E= IB('time10','Sc2042');
BerSB1    .. SB('bl1') =E= B('time0','Sc1') + B('time1', 'Sc2') + B('time2', 'Sc4')
                 + B('time3', 'Sc8') + B('time4', 'Sc16') + B('time5', 'Sc33')
                 + B('time6', 'Sc66') + B('time7', 'Sc132') + B('time8', 'Sc265')
                 + B('time9', 'Sc530') + B('time10','Sc1061') ;
BerSB2    .. SB('bl2') =E= B('time0','Sc1') + B('time1', 'Sc2') + B('time2', 'Sc4')
                 + B('time3', 'Sc9') + B('time4', 'Sc18') + B('time5', 'Sc37')
                 + B('time6', 'Sc74') + B('time7', 'Sc149') + B('time8', 'Sc299')
                 + B('time9', 'Sc598') + B('time10','Sc1197') ;
BerSB3    .. SB('bl3') =E= B('time0','Sc1') + B('time1', 'Sc2') + B('time2', 'Sc5')
                 + B('time3', 'Sc10') + B('time4', 'Sc20') + B('time5', 'Sc40')
                 + B('time6', 'Sc81') + B('time7', 'Sc162') + B('time8', 'Sc324')
                 + B('time9', 'Sc648') + B('time10','Sc1297') ;
BerSB4    .. SB('bl4') =E= B('time0','Sc1') + B('time1', 'Sc2') + B('time2', 'Sc5')
                 + B('time3', 'Sc10') + B('time4', 'Sc21') + B('time5', 'Sc42')
                 + B('time6', 'Sc84') + B('time7', 'Sc169') + B('time8', 'Sc338')
                 + B('time9', 'Sc676') + B('time10','Sc1353') ;
BerSB5    .. SB('bl5') =E= B('time0','Sc1') + B('time1', 'Sc2') + B('time2', 'Sc5')
                 + B('time3', 'Sc10') + B('time4', 'Sc21') + B('time5', 'Sc42')
                 + B('time6', 'Sc84') + B('time7', 'Sc169') + B('time8', 'Sc338')
                 + B('time9', 'Sc677') + B('time10','Sc1354') ;
BerSB6    .. SB('bl6') =E= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc6')
                 + B('time3', 'Sc13') + B('time4', 'Sc26') + B('time5', 'Sc53')
                 + B('time6', 'Sc106') + B('time7', 'Sc213') + B('time8', 'Sc426')
                 + B('time9', 'Sc852') + B('time10','Sc1704') ;
BerSB7    .. SB('bl7') =E= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc6')
                 + B('time3', 'Sc13') + B('time4', 'Sc26') + B('time5', 'Sc53')
                 + B('time6', 'Sc106') + B('time7', 'Sc213') + B('time8', 'Sc426')
                 + B('time9', 'Sc852') + B('time10','Sc1705') ;
BerSB8    .. SB('bl8') =E= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc6')
                 + B('time3', 'Sc13') + B('time4', 'Sc26') + B('time5', 'Sc53')
                 + B('time6', 'Sc106') + B('time7', 'Sc213') + B('time8', 'Sc426')
                 + B('time9', 'Sc853') + B('time10','Sc1707') ;
BerSB9    .. SB('bl9') =E= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc6')
                 + B('time3', 'Sc13') + B('time4', 'Sc26') + B('time5', 'Sc53')
                 + B('time6', 'Sc107') + B('time7', 'Sc215') + B('time8', 'Sc430')
                 + B('time9', 'Sc860') + B('time10','Sc1720') ;
BerSB10   .. SB('bl10') =E= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc6')
                 + B('time3', 'Sc13') + B('time4', 'Sc27') + B('time5', 'Sc54')
                 + B('time6', 'Sc109') + B('time7', 'Sc219') + B('time8', 'Sc438')
                 + B('time9', 'Sc877') + B('time10','Sc1754') ;
BerSB11   .. SB('bl11') =E= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc7')
                 + B('time3', 'Sc14') + B('time4', 'Sc29') + B('time5', 'Sc59')
                 + B('time6', 'Sc119') + B('time7', 'Sc238') + B('time8', 'Sc477')
                 + B('time9', 'Sc955') + B('time10','Sc1910') ;
BerSB12   .. SB('bl12') =E= B('time0','Sc1') + B('time1', 'Sc3') + B('time2', 'Sc7')
                 + B('time3', 'Sc15') + B('time4', 'Sc31') + B('time5', 'Sc63')
                 + B('time6', 'Sc127') + B('time7', 'Sc255') + B('time8', 'Sc510')
                 + B('time9', 'Sc1021') + B('time10','Sc2042') ;
```

Bilag F

Matlab kode til generering af scenariotrægrafer

F.1 Grafen for alle scenarier

Dette script kan skrives i en matlab editor og gemmes som en .m fil. Når scriptet køres får vi en graf med T perioder.

```
%Script til generering af plots for binomial traeer
clear all
close all

%%%%%INPUT: Skriv antallet af perioder i traet%%%%%
%(!Advarsel: Proev maks med 13 p.g.a tid)%
T = 10;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

L = 2^T; %Her regner vi antallet af blade

start(1) = 1;
start(2) = 1.5;
for i = 3:T+1,
    start(i) = start(i-1) + 2^(i-3);
end

for i=0:L-1,
    for j= 1:T+1
        y(j,i+1) = start(j) + (2^(j-1)) * fix(i/(2^(j-1)));
        x(j,i+1) = T+1-j;
    end
end

plot(x,y)
```

```
axis([0 T 1 L]);
%axis('off');
xlabel('Tid (aar)')
ylabel('Scenarier')
%title('Et ikke kombinerende binomial trae med 10 perioder.')
```

F.2 Grafen for udvalgte scenarier

Vi skal først definere en matrix, som indeholder numrerne for de udvalgte scenarier, (d.v.s. bladene i træet). Scenarierne tællede nedefra. For det reducerede træ til 12 scenarier har vi:

```
IndexVector =[ 5   137 293 327
               340 342 343 693
               694 750 850 986 ];
```

Dernæst køre vi følgende script:

```
%%%%%INPUT: Skriv antallet af perioder i traet%%%%%
%(!Advarsel: Proev maks med 13 p.g.a tid)%
T = 10;
Antalblade = 400;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

L = 2^T; %Her regner vi antallet af blade

start(1) = 1;
start(2) = 1.5;
for i = 3:T+1,
    start(i) = start(i-1) + 2^(i-3);
end

for i=0:L-1,
    for j= 1:T+1
        for k= 1:Antalblade
            if (i==IndexVector(k))
                y(j,i+1) = start(j) + (2^(j-1)) * fix(i/(2^(j-1)));
                x(j,i+1) = T+1-j;
                break
            end
        end
    end
end

plot(x,y)
axis([0 T 1 L]);
```

```
%axis('off');
xlabel('Tid (aar)')
ylabel('Scenarier')
%title('Et ikke kombinerende binomial trae med 10 perioder.')
```